FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Study of new paradigms of integrated circuit simulation

Luís Alberto Couto Da Silva Salgado De Abreu

Master Thesis

Supervisor at INTEL: Miguel Andrade Martins (PhD)

Supervisor at FEUP: Manuel Cândido Duarte Dos Santos (PhD)

April 7, 2019

© Luís Alberto Couto Da Silva Salgado De Abreu, 2019

Abstract

From the last fifty years, like Intel co-founder, Gordon Moore appraised, the density of transistor on integrated circuits has been doubling every eighteen months. Nowadays, is possible to find out in the modern smartphones and PCs, integrated circuits built with 5 nm technology, and each one can be composed of more than four billion transistors.

As the technology continues to scale down, it becomes harder to handle all the steps of the manufacturing process precisely. As a consequence, the variation in the manufacturing increases, leading to a need for an improvement in the integrated circuits design flow. Designers must deal with variation accurately and efficiently during the development of the integrated circuits to achieve the product specifications.

The rise of variation, complexity of the device models, circuits size and specifications, are pushing the designers to the resource limits (IT – information technology – and time).

In this thesis, a study of the standard IC design flow, the different types of variation and the current methods of simulation are presented. The primary focus is on Monte Carlo simulations and the benchmarking of their sampling methods, namely the Random, Latin Hypercube and Low Discrepancy sampling.

The Monte Carlo method was tested by resorting to multiple methods to generate the variation space of parameters (and their distributions) that can vary during the genesis of the integrated circuits. The description of the sampling methods is also included. In this same analysis, the possible points to be optimized are presented.

In this thesis, an optimized method for Monte Carlo simulations is proposed for large netlists. This method consists in using a pre-MC simulation and a Response Surface Model (RSM) to determine and filter the most important contributor for the measurement. It has been demonstrated that the proposed method archives more accuracy with fewer resources.

Keywords: Monte Carlo, Low Discrepancy Sampling, Latin Hypercube Sampling, Random Sampling, IC variation, global variation, mismatch variation, IC simulation, PVT variation, RF design, Analog design, FinFET, MOSFET, Screening.

Resumo

Nos últimos cinquenta anos, tal como o cofundador da Intel, Gordon Moore, estimou, a densidade de transístores em circuitos integrados tem duplicado a cada dezoito meses. Atualmente é possível encontrar nos smartphones e PCs mais recentes circuitos integrados com tecnologia de 5 nm. Cada circuito integrado pode ser composto por mais de quatro milhares de milhão de transístores.

À medida que a tecnologia encolhe, lidar com precisão em todas as etapas do processo de fabricação dos circuitos integrados torna-se difícil. Como consequência, o aumento da variação no processo de fabricação, leva a uma necessidade de uma melhoria no *design flow* dos circuitos integrados. Os designers devem lidar com a variação de maneira precisa e eficaz durante o desenvolvimento dos circuitos integrados, de modo alcançarem as especificações do produto.

O aumento da variação, da complexidade dos modelos dos componentes, o tamanho dos circuitos e das especificações, levam os designers aos limites dos recursos (IT – *information technology* – e tempo).

Nesta tese é exposto um estudo do *design flow* atual dos circuitos integrados, os diferentes tipos de variação e os métodos atuais de simulações. O foco deste relatório é em simulações de Monte Carlo.

Para estudar e testar o método de Monte Carlo foram utilizados múltiplos métodos para gerar o espaço amostral referente aos parâmetros (e as suas distribuições) que podem variar durante a génese dos circuitos integrados. A descrição dos métodos de amostragem do espaço de variação, a amostragem de baixa discrepância, o método *Latin Hypercube* e amostragem aleatória está também incluída. Nesta mesma análise são apresentados os possíveis pontos a serem otimizados.

Nesta tese é proposto um método otimizado de simulação de Monte Carlo para grandes netlists. Este método consiste em usar uma simulação pré-MC e um Modelo de Superfície de Resposta (RSM) para determinar e filtrar o contribuinte mais importante para a medição. Foi tambem demonstrado que o método proposto consegue mais precisão com menos recursos.

Palavras-chave: Monte Carlo, amostragem de baixa discrepância, *Latin Hypercube*, amostragem aleatória, variação em ICs, variação global, variação de disparidade, simulação em *IC design*, variação PVT, RF design, design analógico, FinFET, MOSFET, Screening.

Acknowledgements

I would like to express my gratitude to everyone that has been supporting me in this journey.

I want to thank, particularly, **Dr. Miguel Martins** for showing what guidance is, for the expertise and also for the patience and for the time dedicated to me. To my supervisor **Dr. Cândido Duarte** for the first introduction to the topic and for getting me this opportunity. I acknowledge **Intel Corporation** for providing me with all the necessary IT and Economic needs and especially for the opportunity.

I also want to thank:

- Particularly my parents **Olívia and Francisco** and grandparents **Lurdes and Antonio** for support, love, and motivation.
- To my sister **Dr. Susana Pinheiro** and my brother-in-law **Dr. Diogo Pinheiro** for support, guidance in the entire degree, and especially for the motivation and for being a inspiration to me.
- To my friend **Alexandre Truppel** for providing explanations and summaries during the entire degree.
- To **Catarina Soares** for the support, help, and motivation.
- To Nuno Fernandes, Ziad Bouhairy, Rajeeva KN and Ali Bitar for the company in this German adventure.
- To my girlfriend for the love and support.
- To my work colleagues for the welcome and support.

Without them, it would be not possible.

Luís Alberto Couto Da Silva Salgado De Abreu

"Intelligence is the ability to adapt to change."

Stephen Hawking

Contents

List of Figures

Abbreviations and Acronyms

ADC	Analog-digital converter
BJT	Bipolar junction transistor
CAD	Computer-aided design
CMP	Chemical mechanical polishing/planarization
CPU	Central Processing Unit
DAC	Digital-analog converter
DBB	Digital Baseband
DSP	Digital signal processing unit
ECAD	Electrical Computer-aided design
FinFET	Fin Field Effect Transistor
GSM	Global System for Mobile Communications
HB	Harmonic balance
I/O	Input/output
IC	Integrated Circuit
IT	Information Technology
LDE	Layout dependent effects
LDS	Low-Discrepancy-Sampling
LHS	Latin-Hyper-Cube sampling
LNA	Low noise amplifier
LO	Local Oscilator
MC	Monte-Carlo
MOSFET	Metal-oxide- semiconductor field-effect transistor
NMOS	N-channel MOSFET
OFAT	One factor at time
PA	Power amplifier
PMOS	P-channel MOSFET
PVT	Process, Voltage and Temperature
RAM	Random-access memory
RF	Radio frequency
ROM	Read-only memory
RTL	Resistor-Transistor Logic
SPICE	Simulation Program with Integrated Circuit Emphasis
TCAD	Three-dimensional technology computer-aided design
TFET	Tunnel field-effect transistor
VVO	Virtuoso Variation Options

Chapter 1

Introduction

The resources required for circuit simulations are increasing significantly due to the complexity rise of device models and circuit specifications. The scope of verifications that is required (by the product specifications) to be simulated nowadays is so high that it is impacting the time to market of products. New solutions need to be identified in order to reduce the products design cost (human resources, IT (information technology) needs and number of software licenses) while maintaining the required quality of results. The goal of this thesis was to study the current methods of simulations and look for new solutions that provide a reduction of resources, particularly simulations involving process, voltage, and temperature (PVT) variations. The tools required to test the current methodology, as well as the circuits and IT were provided by Intel Germany GmbH, which was where this thesis was developed.

1.1 Context and motivation

1.1.1 Transistor evolution and its limits

Intel co-founder Gordon Moore appraised that the transistors density on integrated circuits had doubled every year since their invention, in 1965. In 1975, the pace was adjusted to a doubling every two years. Figure 1.1 shows the rise of the transistor number in the Intel processors over time. Moore's law predicted that this trend would continue into the foreseeable future. The pace was revised again and, over roughly 50 years from 1961, the number of transistors density has since doubled approximately every 18 months. In 2016, Intel has suggested silicon transistors can only keep shrinking for another five years.

Shrinking transistors have powered 50 years of technological advances, translating on an exponential growth in the ICs (Integrated Circuits) capability. There are various

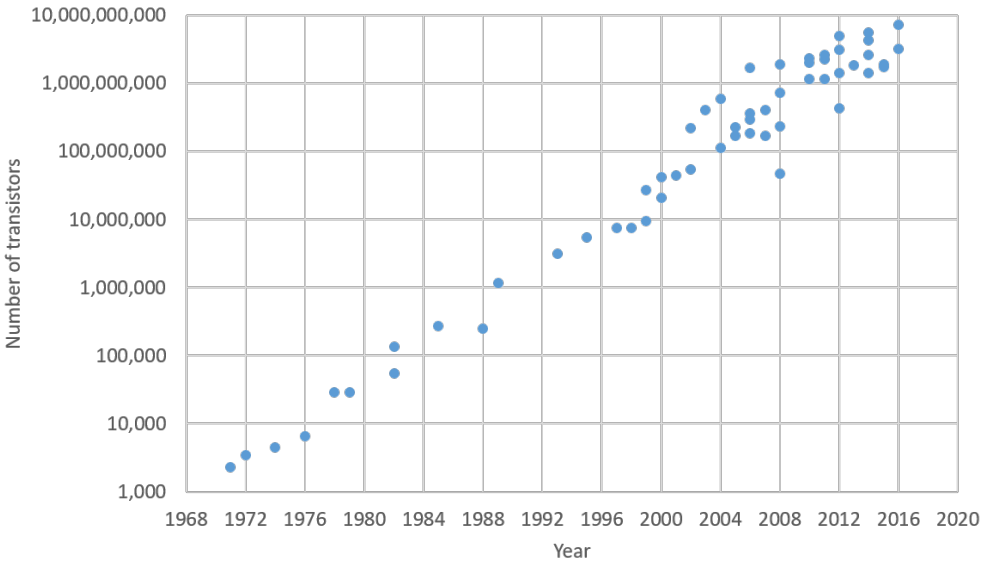
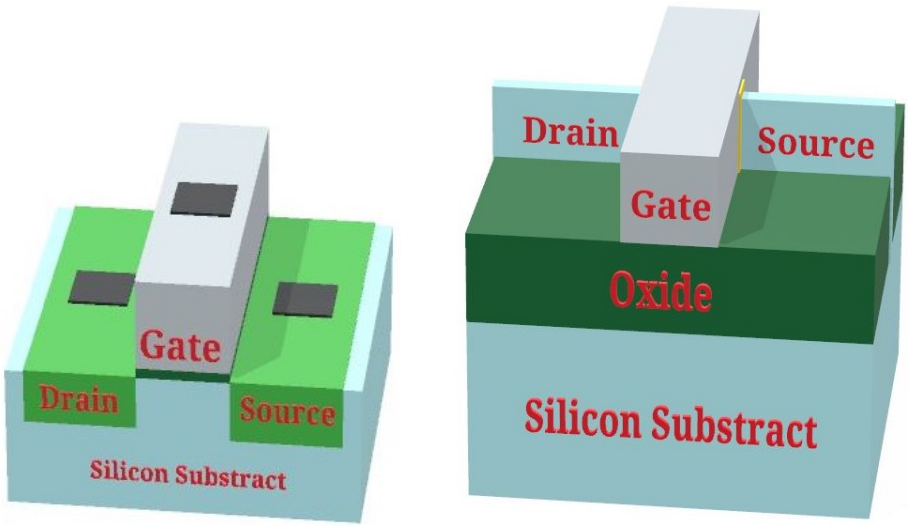


Figure 1.1: Density of transistor with the time evolution.

types of transistors, such as Field-effect transistors (FET) or Bipolar junction transistors (BJT). In this thesis, only the FETs, particularly the Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET) and the Fin-Field-effect Transistor (FinFET) are used. The difference between both structures is shown in figure 1.2.



(a) MOSFET structure. (b) FinFET structure.

Figure 1.2: Differences between MOSFET and FinFET transistors.

MOSFET transistors have nowadays approximately 10nm of channel length and approximately 1nm of gate oxide thickness, as shown in figure 1.3.

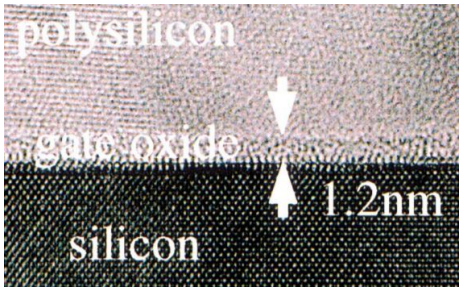


Figure 1.3: High resolution transmission electron micro-graph of a 35nm gate length with gate oxide thickness of approximately $1.2 \pm 0.3nm$" - Source: [1].

10nm of length means that according to the lattice constant of silicon of 0.5431nm [2] and, that the number of atoms is equal to:

$$\#N_{atoms} = \frac{length}{lattice\ constant} \tag{1.1}$$

The given cross section has:

$$\frac{10\,\text{nm}}{0.5431\,\text{nm}} = 18.413 \text{ atoms} \tag{1.2}$$

As the technology is going towards to the atomic size, one question arises: is it possible to stop the electrons effectively enough when source and drain are too close? This is a question addressed to quantum mechanics, and the answer is: even if there is a potential barrier between two electrodes, electrons can still flow because they have a "pesky ability to penetrate barriers, a quantum mechanical phenomenon known as **quantum tunneling**" [3], as shown in figure 1.4 and figure 1.5a.

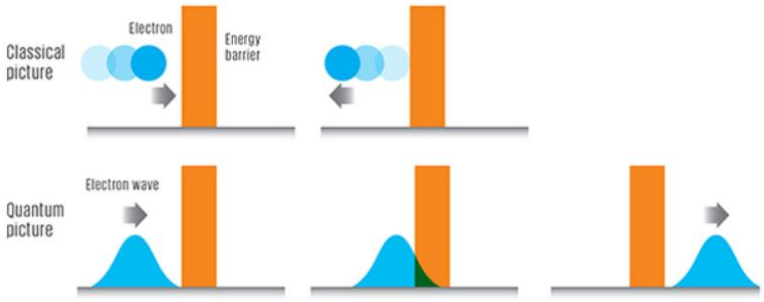
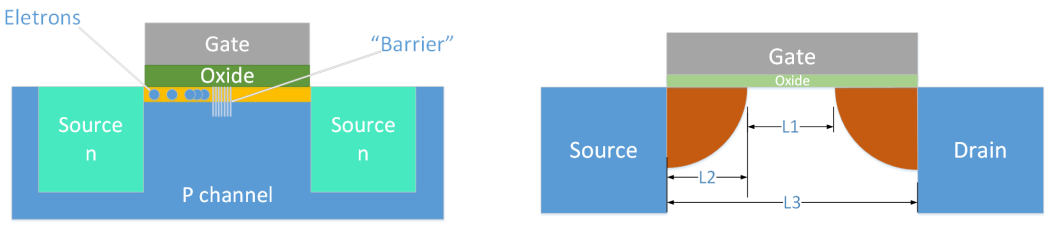


Figure 1.4: "Electrons have a probability of passing through the energy barrier. The thinner the barrier, the higher the probability that such a tunneling event might occur" - Source: [3].

Tunneling can cause large power dissipation, low drive current, and strong sensitivities to **process variation**, which greatly limit CMOS scaling [4]. As shown in figure 1.5b, assuming the limit caused by the Quantum Channel phenomenon, it is possible to illustrate the boundaries of the gate length, namely:

$L3$: Length of the gate;
$L2$: Length affect by Quantum Channel if $L1 = 0$;
$L1$: Length of "safety";



(a) Barrier illustration.

(b) MOSFET length limits.

Figure 1.5: MOSFET illustrations.

In this particular example, the limit is imposed by $L1$ length due to the need of having the minimum distance to avoid the quantum channel.

The oxide gate insulator also reached its limit of thickness. Is it not possible to induce more charge in the channel and make the transistor faster, due to this limitation. Quantum tunneling is one of the causes, mainly because it drives too much leakage current across the channel when the transistor is "off", whereas, ideally, no current should flow at all.

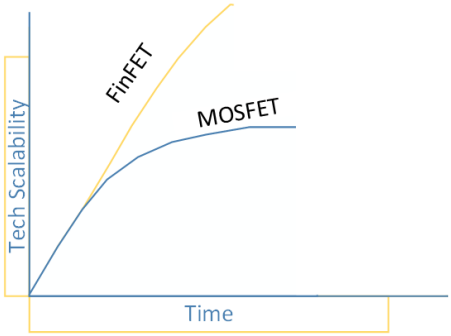


Figure 1.6: Paradigm shift.

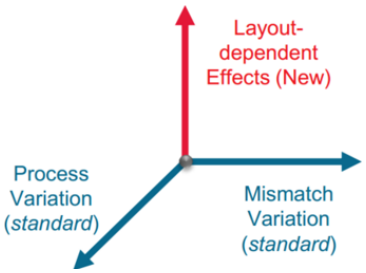
To overcome these issues it happened a so-called paradigm shift. Other FET technologies were developed, such as FinFET transistors. This transistor type could be pointed as the successor of MOSFET due to the good scalability properties, represented in figure 1.6.

1.1.2 Integrated circuit variations

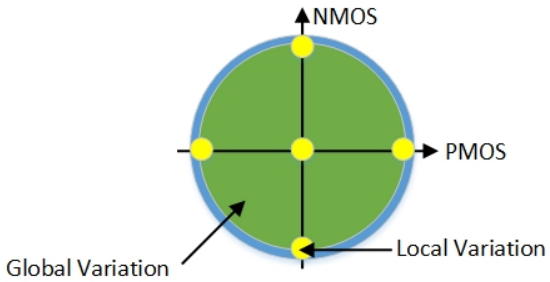
As the technology is going forward to the sub-nanoworld, the demand of performance increases and supply voltages decrease, all make variation effects are more pronounced. Even those variations such as electron mobility have now a higher impact on the design of the circuits and to maintain product quality, they must be taken into account, especially in RF and analog designs. One solution is to oversize the circuits, however, this leads to a waste of resources (like die area or power consumption).

Every time the technology scales down, the relative variation between technologies gets bigger. Mainly because it becomes harder to control and manage the precision of all steps in the manufacturing process. That is the major reason why statistical variation (and its impact) is essential. Handling accurately and efficiently with the variation is becoming more and more important and harder with advanced nodes [5].

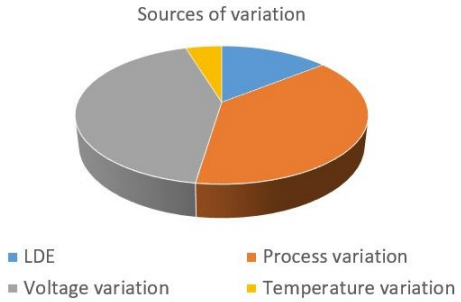
A parcel of the device electrical performance depends on global variation (between die, wafers, and lots), local variation (i.e., mismatch, an uncorrelated atomistic variation of each device instance), and it could also include the layout dependent effects (LDE) [6], as shown in figure 1.7a. Since the greater technology nodes (130 nm), local variation took the lead role considering that the global variation is too wide to cover. In order to include the majority of variation space of both variations, the current methodology is based on choosing three different points for each type of transistor (slow, typical and fast) and apply a local variation around this points, as shown in figure 1.7b.



(a) Performance dependencies.



(b) Manufacturing processes variations.



(c) Source of variations - Adaptation from: [7].

Figure 1.7: Transistor variations.

There is also another sort of variations that must be included in order to take into account all sources of variations, namely the temperature and the voltage variations. The proportion of all variation sources is shown in figure 1.7c. Voltage and process variation, are the major sources of variation followed by LDE and temperature.

1.1.2.1 Evaluation of variations: Methods

The device variation can be simulated using:

- Sweep Corners: It is a PVT variation technique (explored in section 2.3.1) that uses a simulation with a stipulated single set of points within the variation space of the circuit components. Within this methodology, it is possible to define multiple corners for each specific environment conditions and process variation and run a simulation for each case.
- Monte-Carlo simulation: Multiple simulations with single random sets of points within the variation space of the circuit components. The focus is on the global and mismatch (explored in sub-section 2.3.2).

This current simulation paradigm might be not valid anymore due to the increase of variation in the manufacturing process of the FinFET caused by:

- Line/fin-edge roughness;
- Distribution of dopant atoms;
- Chemical mechanical polishing/planarization (CMP) variations;

This will be explored with more detail in sub-section 2.2.

New technologies require new and more complex component models. This requires more IT resources such as memory and simulation time to process the equations. Bringing this together with the rise of functionality required by the circuit specifications, the computational power needed to simulate those circuits is now on a feasible edge. Since the simulation methods did not completely follow the evolution, in some cases the resources needed are now out of the development capability boundaries.

1.2 Complexity increase

1.2.1 Circuit complexity increase example: From GSM to 4G

The first mobile phone digital technology, GSM (Global System for Mobile Communications), became to replace the original full-analog mobile phone system, named AMPS (Advanced Mobile Phone System). In the beginning, GSM cell phones were phones with simple functionalities. They provided the required voice calls, SMS, and phonebook features. Some improvements were made like the ability to use them as a clock alarm, currency exchange, and other basic functionalities.

The PDA (personal digital assistant) was a pioneer handling additional features. It introduced the touch screen user interfaces and a wide range of application programs. Smartphone came when the PDA functionalities were added to the common cell-phone.

Mobile phones have the GSM modem interfacing with the GSM network, as shown in figure 1.8. This GSM modem consists of several parts [8]:

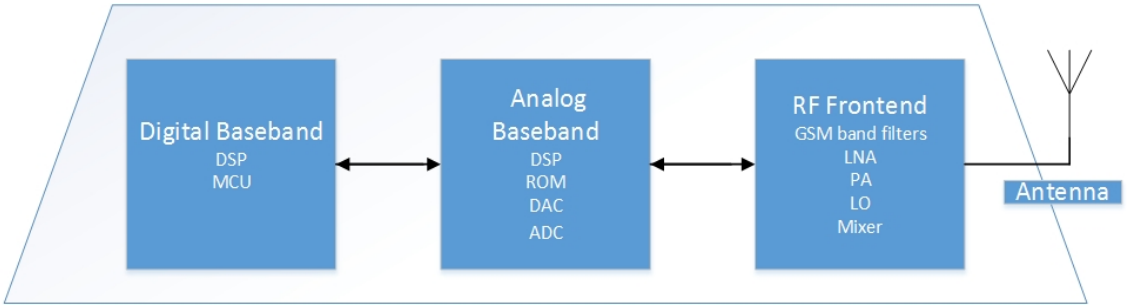


Figure 1.8: GSM-MODEM Integrated Circuit Model.

- RF Frontend, responsible for receiving and transmitting on GSM frequencies. It minimally consists of an antenna switch, GSM band filters, low-noise amplifier (LNA) for the receive path, a power amplifier (PA) for the transmit path, a local oscillator (LO) and a mixer.

- Analog Baseband, responsible for modulation and demodulation. It is a bridge between the digital domain and the analog domain. The receiver has a filter followed by an Analog to Digital Converter (ADC), passed to the Digital Signal Processor (DSP) in the Digital Baseband (DBB). The transmitter block has an inverse of the receive operation, using instead a Digital-to-Analog converter (DAC). Modulation ROM tables are necessary to modulation and demodulation processes.

- Digital Baseband is responsible for digital signal processing and the GSM protocol stack. The most common sets use a DSP plus a general-purpose processor (MCU).

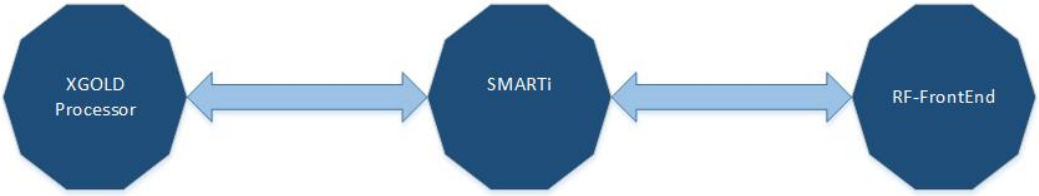


Figure 1.9: GSM MODEM product diagram.

Those three different blocks can be found at two different products plus the RF-Frontend, as shown in fig 1.9.

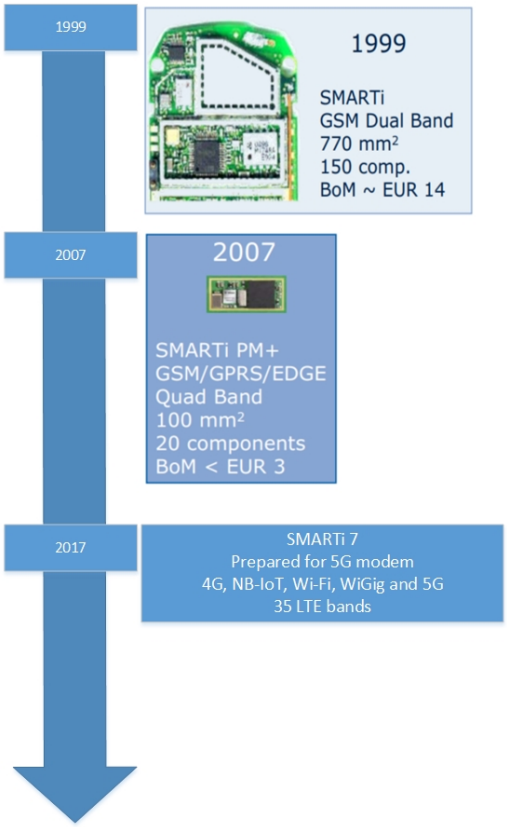


Figure 1.10: The evolution of SMARTi IC.

In order to represent the complexity of the evolution, the picture 1.10 shows an analysis regarding the evolution of SMARTi IC's. From the beginning of SMARTi history, back to 1999, the first single-chip GSM needed an area of 770mm^2, 150 components and it had a cost around 14 euros. The technology used was BiCMOS (a mix between BJT and CMOS) because MOSFET could handle high current circuits efficiently, and portions of specialized very high-performance circuits use bipolar transistors. It is possible to deduce that optimizing both transistor types in the process is barely impossible without adding extra fabrication steps and as a consequence, increasing the process cost. To conclude, in the area of high-performance logic, BiCMOS may never offer the low power consumption of the CMOS alone, due to the potential for higher standby leakage current. The SMARTi PM+, released eight years later, with 0.13um CMOS technology, needed for an entire transmission board almost half the space, 7.5 times fewer components needed and for a fifth of the price, compared with the original SMARTi.

1.2.2 The need for new tools: SPICE

During the 1960's, a need has arisen regarding the development process of circuit simulation techniques. The increase of the size and complexity of an electronic circuit made simulating a circuit on a computer a more attractive method.

To change the development flow, simulators had to provide reasonably accurate results in a reasonable amount of time. Simulation Program with Integrated-Circuit Emphasis or SPICE was a result of this new approach to circuit design. Within SPICE vision, the circuit can be visualized as a collection of various elements connected together at nodes. Thus, the entire circuit can be seen by the number of nodes, elements and the orientation of these nodes. Following this logic, with n nodes in a circuit, SPICE creates an $n \times n$ matrix. The values defined in the matrix are the values of voltages and/or currents at the external nodes, the values of the internal voltages and currents, or state variables, can be solved for each node using matrix manipulation techniques. To calculate those values, SPICE uses physical values that describe behavior, presented in the component model.

1.2.3 Transistor complexity: model evolution

Transistors are quite simple devices but it is arduous to describe their behavior. FET modeling had to follow the technology evolution in order to allow the use of simulators to observe the correct circuit behavior. Models were built to describe how the components work, and there is a large variety of purpose-dependent models. As described by [9], the earlier models had very simple and basic equations for C-V and I-V characteristics and also very few parameters to describe the equations. Those model parameters are representing particular and quantifiable physical values obtained from process information, electrical data and parameter extraction.

As the technology evolved, FET models became more complex. Equations and parameters needed to include various effects, brought, for example, from shorter channels and higher field strengths. This growth in the number of parameters and equation increased the resources needed to extract the model parameters and also to process those models within the simulations.

Since 1996, Compact Model Coalition (CMC) is a working collaborative group focused on the standardization of SPICE device models. The goal of this collaborative group is to promote the international, nonexclusive standardization of compact model formulations and the model interfaces.

The Department of Electrical Engineering and Computer Sciences of the University of California, Berkeley, is the major responsible for the release of the models. The following analysis was based on data provided by [10] and [11].

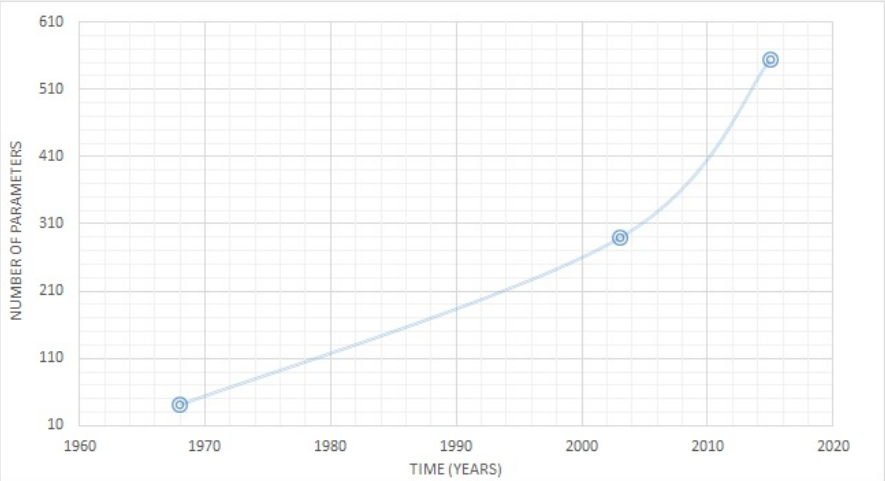


Figure 1.11: Number of parameters in models.

The first SPICE MOSFET model MOS1 or 'Shichman-Hodges', was developed in 1968 and had just 41 parameters. In order to compare complexity over time, after 35 years was released the BSIM4.3.0 (Berkeley Short-Channel IGFET Model), addressed for physical effects into the sub-100nm regime, with 289 parameters. Thus, this is translated into 248 more elements in 35 years. In 2015, 12 years after the released of BSIM4.3.0, it was released the BSIM-CMG 110.0.0, reference for vertical transistors (FinFET). Then, the number of parameters raised more 265 elements, for the total of 554, as shown in chart 1.11.

The conclusion of this analysis is that in the last 12 years, the number of parameters of the models and the resources needed to process them, raised more than in 35 years.

1.3 Thesis objectives

The objectives outlined for this dissertation are:

1. Study and application of Monte Carlo (MC) simulation, analysis of performance when different techniques are applied on circuits with different design spaces (different number of varying parameters). The goal here is to identify the main MC limitations in this context.

2. Several techniques need to be studied for the problem of multidimensionality optimization approach. For this purpose, an appropriated solution needs to be found for the current circuit context.

1.4 Thesis structure

The present dissertation is organized in six chapters. In chapter two, it is shown the main contents of the current paradigm of simulations and their flow and structure. Additionally, it is reviewed the primary key concepts needed to understand the following topics, including the simulations structure and flow, the types of variations and the methodologies to deal with them.

Chapter three introduces the key-core concepts of this thesis, the sampling methods. It is presented the Random, the Latin Hypercube and the Low Discrepancy sampling methods, their flow, pros and cons.

Chapter four contains the study cases used to do the comparison of the methods introduced in chapter three. It is also presented the results obtained and a discussion of them.

In chapter five, a new optimized method is proposed and presented. It was based on the conclusion taken in the previous chapter four (overload of instances number).

The last chapter, six, includes the analysis conclusion and a list of improvement that is still possible to do in order to increase the performance of the Monte Carlo Simulations.

Chapter 2

An overview on IC simulation

Electronics is divided into two different worlds: analog and digital. On the one hand, digital circuits commonly require very fast switching mode functionality, thus benefiting from smaller transistors which have smaller parasitic capacitances. On the other hand, analog circuits, especially in small-signal applications, call for low-noise transistors. These need special modeling methods and use different fabrication processes.

In the analog world, to handle signals we need more complex methods and accurate transistors. Speed comes with shrinking, while accuracy comes with better processes and modeling.

The technology push is mainly done by the digital side because people seek more processing capability. This push made by the digital side enhances the difference between sizes of digital and analog technologies. For example, in digital design, it is currently used 7 nm and 5 nm technology and analog is still on 28 nm, in the most cases.

In this chapter, the current methods for IC simulation, what is and how the technology variations are handled as well as the different design flows are presented. The boundaries of the tools will be explored and it will be provided the key concept terms needed for the upcoming chapters.

2.1 Standard simulation structure

Simulations can promote the understanding of the behavior of a system without actually prototyping and testing the system in the "real world". Support experimentation that occurs entirely in software can save huge amounts of time, especially during the development time.

Several parts are needed to perform simulations. The focus will be on four main parts, as shown in figure 2.1. By using models as a logical representation of the real component,

and simulators that contain the circuit's behavior in the form of equations, it is possible to use SPICE (previously exposed in the introduction chapter 1) to test a schematic/layout.

First, netlist (SPICE code that describes the circuit) is built together with the models. The simulator can now run the algebraic calculations of the correspondent matrices also using the representative equations of the components.

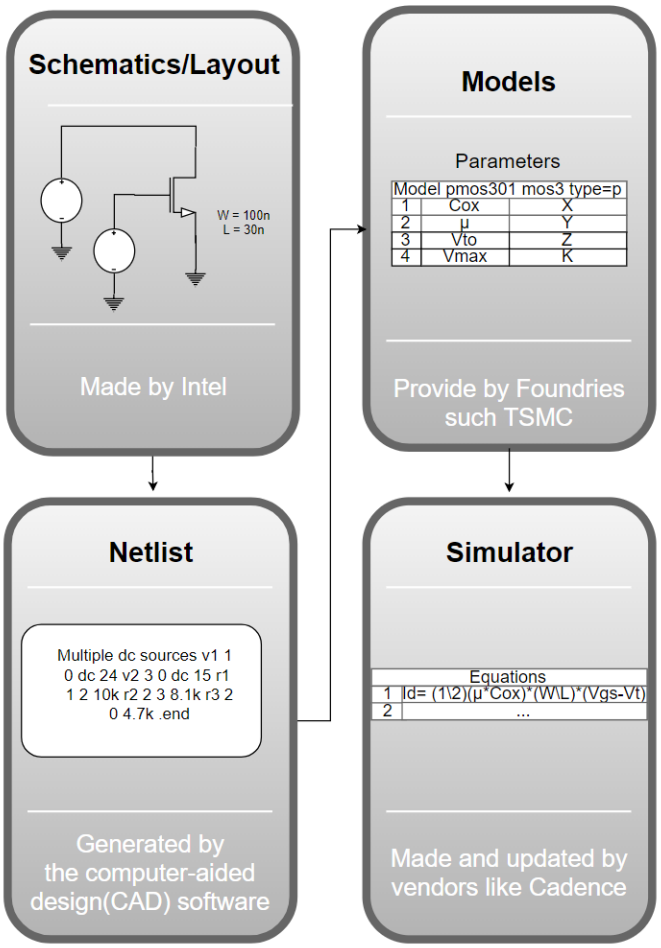


Figure 2.1: Simulation structure.

2.2 Technology and Environment variations

Variation is a hard issue to solve. Design fails lead to delays and yield loss. In this section, the types of variables will be discussed as well as the types of variations and terminology and the current approaches to deal with it.

Variations have always been around, but each time a technology shrinks, the effect of variation increases. For example, variance in electrical device performance doubled when went from CMOS 40nm to 28nm, as shown in figure 2.2.

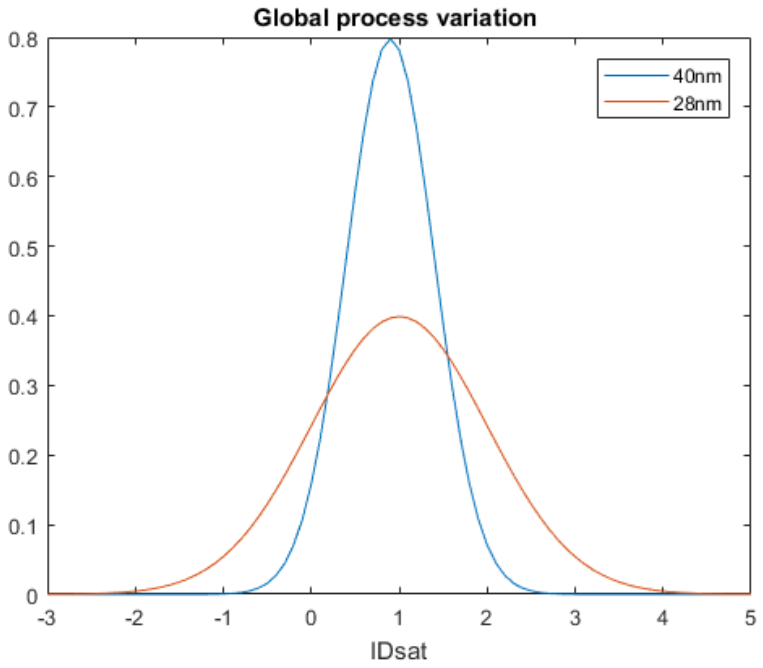


Figure 2.2: 40nm vs 28nm MOSFET Variation - 2x overall increase in Idsat variation (for global process variations) - Adaptation from: [12].

Currently, there are many different approaches to analyze variations. The most common are based on Corners and Monte-Carlo, explained in sections 2.3.1 and 2.3.2. In chapters 3 and 6, new and more effective methods to reach results with less time and IT resources, will be introduced.

2.2.1 Characterization of variables

Essentially, there are mainly two types of variables that could affect a circuit's behavior [13] :

-Controllable variables: Those that can be defined by designers. For example, topology, device sizes and placement.

-Uncontrollable variables: Those that cannot be set by the designer; its existence is due to various mechanisms such as variations on the fabrication process.

2.2.2 Types of uncontrollable variables

For further analysis, it will be considered the environmental variation, process variation, and layout parasitics.

2.2.2.1 Environmental variation

Environmental variables can affect the performance of the circuit components once the circuit is operating in the user environment. In order to ensure full-functionality, designers must be aware of target performance values across all environmental conditions and specifically, for the user environment, the worst-case must be known and pre-set. These conditions are different for different circuits/environments, for example, military devices typically must handle intense temperatures compared with normal automotive devices. This class of variables includes, for example, temperature and power supply voltage.

2.2.2.2 Layout parasitics

These parasitics crop up when two conductors at different potentials are close together. Their electric field interferes with each other and store opposite electric charges, forming a capacitor. There are three main categories in the 2.5D modeling[1], as shown in figure 2.3 [14]:

- Area Capacitance (CA) - Conductors in different layers overlap
- Fringe Capacitance (CF) - Lines that curve around the conductors (the hardest to estimate)
- Coupling Capacitance (CC) - Conductors in the same layer

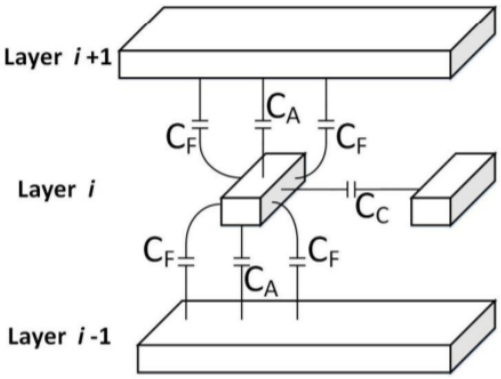


Figure 2.3: 2.5-D modeling capacitances - Source: [14].

Layout resistances and capacitances are not included in the schematic design, but they will be distributed throw the IC. They should be taken into account especially in circuits operating at higher frequencies, such as RF designs (RC filters will change the signal).

[1]This model does not cover all the FinFET capacitances, it is only possible with 3-D modeling.

The challenge with this variation is that they must be handled during front-end design, the step before layout design (for example, by over-margin the constraints).

2.2.2.3 Layout-dependent effects (LDEs)

The device layout structure can have strong effects in electrical performance for advanced technology nodes. They must be taken into account during the design cycle otherwise they will have an unfavorable impact into the circuit functionality. For technologies nodes of 90nm and older, this was not a big issue since the device characteristics were considered pretty much immune from layout. In other words, the electrical characteristics of the device were independent of the layout shape. However, it is no longer true for advanced technologies of 90nm and below [15].

A few examples of effects that could cause circuit performance or behavior deviations will be presented below:

- LOD (Length of Diffusion):

  It changes the stress effect on the active gate.

- WPE (Well Proximity Effect):

  It changes the doping concentration near the well edge and impacts the Threshold Voltage of the device.

- OSE (OD Spacing Effect):

  The Diffusion (OD) spacing to the neighbor which is called as OD-OD Spacing Effect impacts the behavior of the device.

- MBE (Metal Boundary Effect):

  With the sharing poly between PMOS and NMOS, Poly metal boundary effect (MBE) arises with two different doped regions.

2.2.2.4 Fabrication process variation

Integrated circuits are build in small blocks of semiconducting material, called dies. Each die is a small piece of a wafer that contains various copies of the circuit. Fluctuations during manufacturing means no die is exactly like any other. Fabrication process variation is related to these variations on die-to-die or wafer-to-wafer, introduced during manufacturing process. Process variations can be classified as global variations (on-die as well as inter-die, including die from different wafers and different wafer lots) and local variations[2] (within-die or intra-die), in other words, "the local variation occurs within a single die, while global variation occurs across wafers" [13], as shown in figure 2.4.

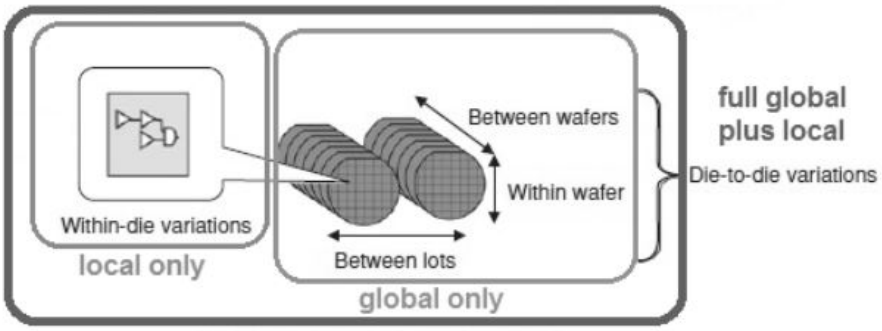


Figure 2.4: Types of process variations - Source: [16].

Inter circuit variation (global variation) is identical for all transistors of one circuit. For instance, can be represented as the variation on the oxide thickness (ΔT_{ox}), substrate doping ($\Delta Nsub$) or electron mobility ($\Delta\mu_0$).

Intra circuit variation (local variation) means that each transistor varies individually. Usually, it is possible to find parameters like the variation of the threshold voltage (ΔV_{th}) or electron mobility ($\Delta\mu_0$) and, the variance is dependable on the circuits area (inversely proportional). The easiest way to deal with this type of variation is to oversize the design but might have some impact in other points of interest.

Analog circuits are based on symmetries and mismatch can be understood as an asymmetry caused by local variation. For example, the output current (i_2) of the current mirror on figure 2.5 is dependable of the V_{th} of both transistors (eq. 2.1), and a mismatch of these parameters (eq. 2.2) can cause a different output current (eq. 2.3).

$$i_2 = k_2\left(\sqrt{\frac{i_1}{k_1}} + V_{th1} - V_{th2}\right)^2 \tag{2.1}$$

$$\Delta V_{th} = V_{th1} - V_{th2} \tag{2.2}$$

[2] Also called as mismatch variation

$$Mismatch = \Delta V_{th} \neq 0 \Leftrightarrow i_2 \neq \frac{k_2}{k_1} i_1 \tag{2.3}$$

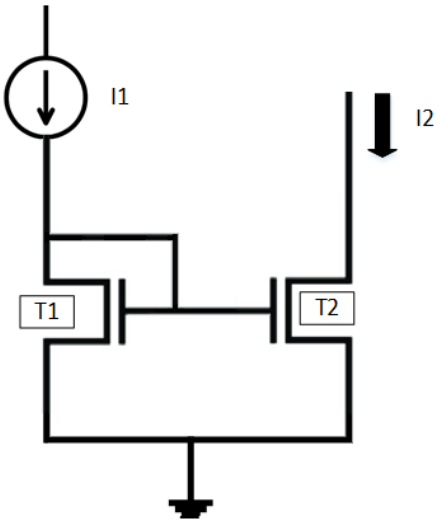


Figure 2.5: Current Mirror

Nowadays statistical models are supplied by the foundry as part of the Process Design Kit (PDK). These models, specify the local and global random variables, the distribution of those random variables (usually, normally distributed) and also the model-sets (corners). Each MOSFET model has three corners: fast (F), typical (T), and slow (S).

Monte Carlo (MC) sampling is used to determine those model parameters. To do that, mean (μ) and standard deviation (σ) of delay measurements are taken. The three models are represented by the sample closest to:

- Fast: $\mu - 3\sigma$
- Typical: μ
- Slow: $\mu + 3\sigma$

It is also possible to apply local variation, using Monte Carlo simulation, on top of each corner, as shown in figure 2.6 (it was used a single transistor scheme and the current flow from drain to source was measured).

The "corners" method is valid and it has been used for digital design as the Fast and Slow models can be used to bound speed limits and because speed is "inversely proportional to power, **F** and **S** indirectly bracketed power" [13].

However, for analog and RF circuits, these models usually do not support any performances bounds such as slew rate and power supply rejection ratio.

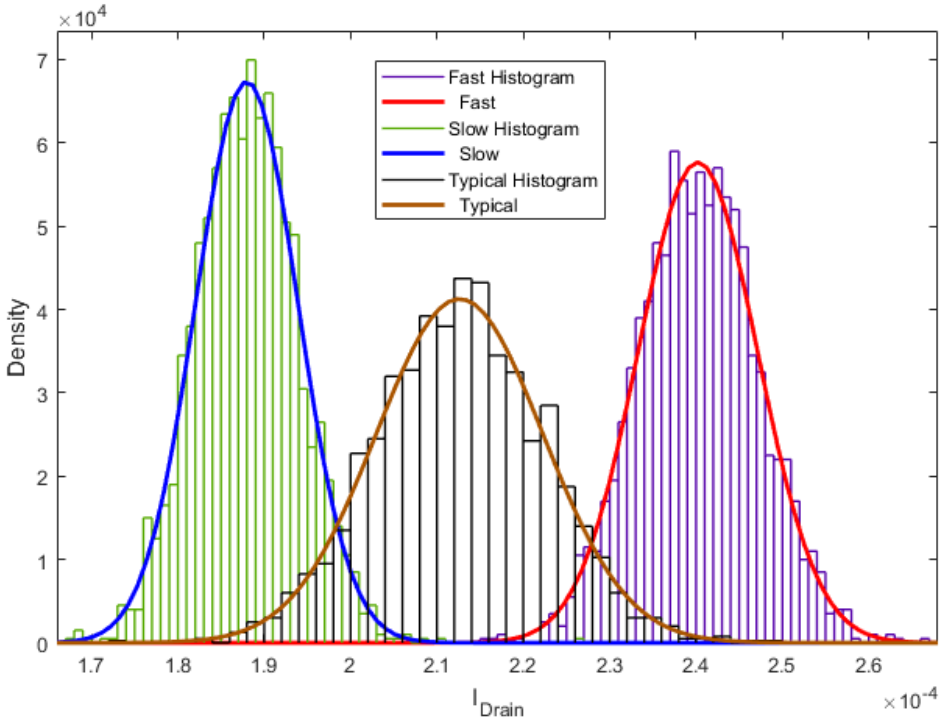


Figure 2.6: Typical S-T-F Variation.

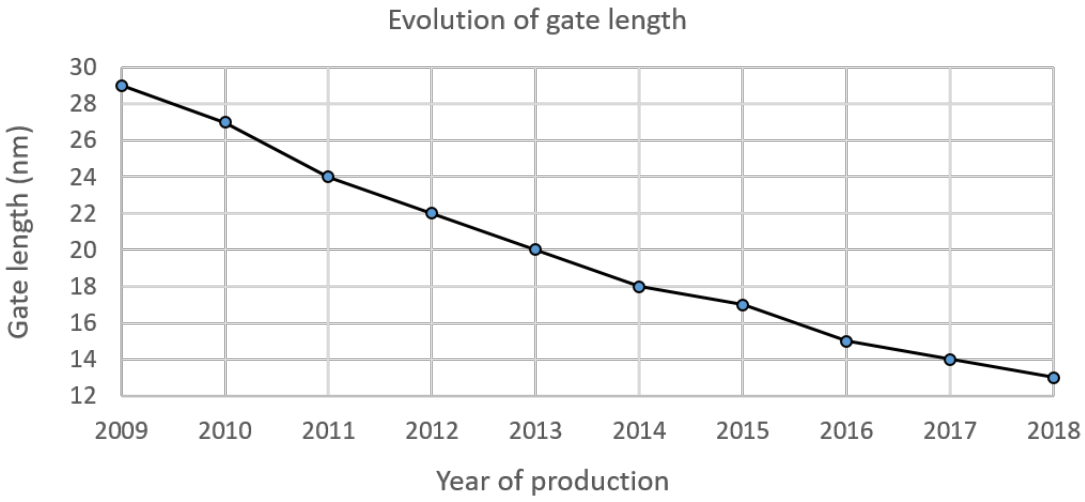


Figure 2.7: Shrinking of transistor's gate length [17].

As shown in figure 2.7, the gate lengths continue to shrink over time and as previously mentioned, a single atom out of place can have a significant impact on the device performance. The job for statistical models is to measure and capture these small variations. As the technology is going forward to small sizes, the global variation and local variations (or mismatch) are merging, as shown in figure 2.8, and are no longer valid to just sim-

ulate with corners [18]. In order to analyze the local variations, it will be needed many simulations to cover all the variation space.

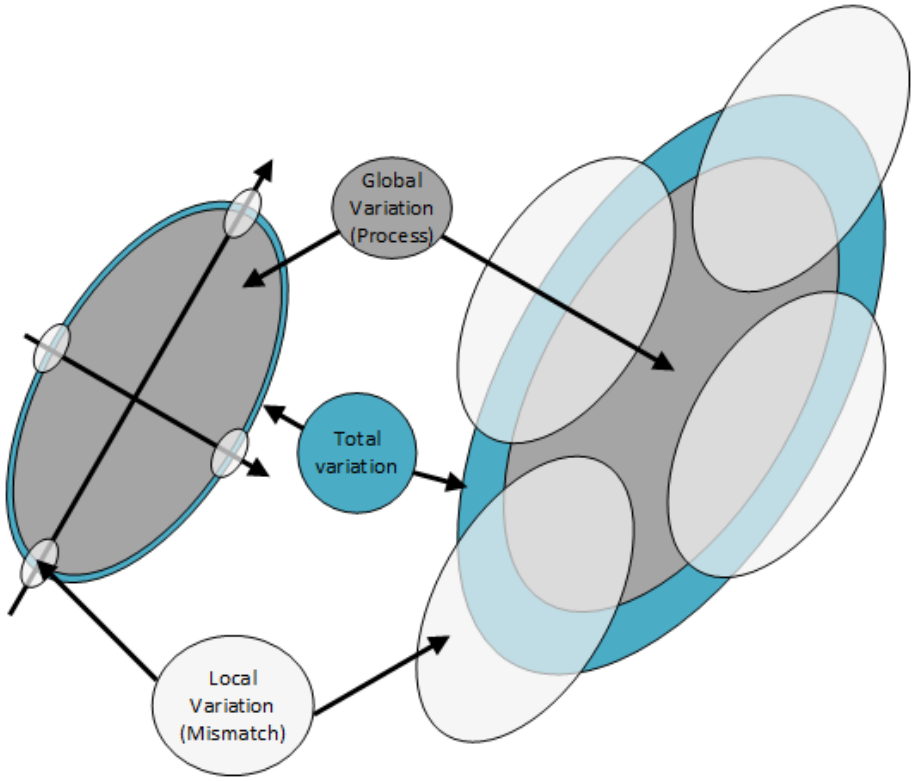


Figure 2.8: Effects on global and local parameters as the technology shrinks [18].

Looking at the variance of two random variables in these smaller technologies, a correlation among the parameters is possible to see [18]. Equation 2.4 shows the variance of two generic parameters X and Y. If the expected value is computed for the variance of these two parameters that are correlated, the result shows that a co-variance exists and cannot be ignored for high-performance analog circuits.

$$var(X+Y) = var(X) + var(Y) + 2 \cdot cov(X,Y) \tag{2.4}$$

2.3 Methodologies to analyze variation

The focus of this section is to explore with detail the techniques and main concepts that are used to analyze the variation introduced in the previous sections.

2.3.1 PVT variation

PVT variation is a technique used to combine model-sets based global process variation (P)[3], and environmental variation, such as power supply voltage (V) and temperature (T) variations. Different loads and power state settings are also used, like standby-state or active-state. This is the standard method used to analyze variations. These sets of variables that define a scenario in which the design performance is measured are called **corners**. Each corner is a representation of a single state, for example:

- Temperature = 25;
- Modelset = SS[4];
- V_{dd} = 2 V;

To evaluate PVT variation impact is also common to use swept variables as corners. Combining a swept of temperature from 0 to 200 degrees, with the three global process model-sets (S, T, F) is possible. In order to exemplify, the impact of the current flow from source to drain is shown in figure 2.9. With 20 measurements for each model-set and three model-sets, there are 60 corners in total.

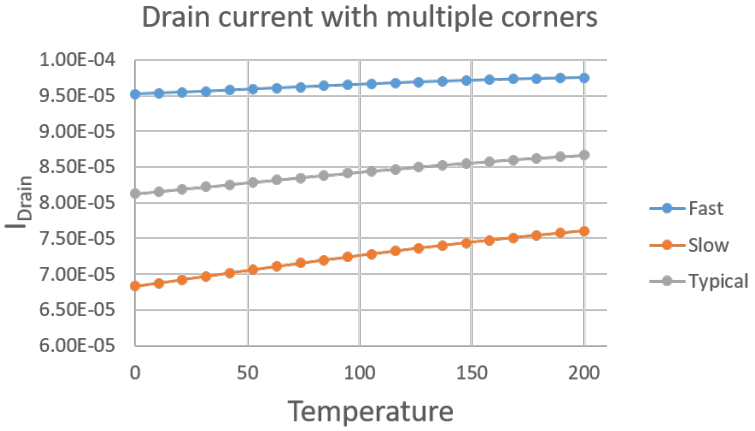


Figure 2.9: PVT variation example.

[3]Fast (F), Typical (T) or Slow (S).
[4]The first letter refers to the N-channel, and the second letter refers to the P channel.

2.3.2 Monte Carlo Simulation

Monte Carlo Simulations are a stochastic approach used in different areas such as finance, project management, energy, manufacturing, engineering, research and development, insurance, oil and gas, transportation, and environment.

The principle of this algorithm is to find out an approximation probability distribution of a certain variable(s) within the circuit by taking a large number of samples. With a large number of samples and, following the Strong Law of Large Numbers, the average of a certain variable must be close to the real value.

The Strong Law of Large Numbers is a key result in Probability Theory. It states that, with certainty, the sample mean of a sequence of independent and identically distributed random variables will converge to their common mean. More precisely, if $X_1, X_2, \ldots, X_n$ is a sequence of independent and identically distributed random variables with finite mean μ, then it holds that

$$\lim_{n \to +\infty} \frac{1}{n} \sum_{i=1}^{n} X_i = \mu \tag{2.5}$$

with probability 1.

In what concerns the present work, the Strong Law of Large Numbers is of great relevance to the implementation of Monte Carlo methods to estimate certain values of interest by simulation. To be more concrete, let λ be some parameter or quantity of relevance which we would like to approximate numerically. Suppose that $X_1, X_2, \ldots, X_N$ are independent and identically distributed random variables which are easy to simulate in the sense that there exists a (reasonably) low complexity algorithm yielding their values (either from their common probability distribution function or from another suitable process). Moreover, let F be a function such that

$$\mathbb{E}\left[f(X_i)\right] = \lambda \ , \qquad i = 1, 2, \ldots, N \ .$$

Then the Monte Carlo approximation to λ is obtained as

$$\lambda \approx \hat{\lambda}_N = \frac{1}{N} \sum_{i=1}^{N} f(X_i) \ ,$$

that is, we estimate λ by computing the sample mean of the random variables $f(X_1), f(X_2), \ldots, f(X_N)$. The Strong Law of Large Numbers, by ensuring that $\hat{\lambda}_N$ converges to λ as $N \to \infty$ (with probability 1), indicates that, except in an event of probability zero, $\hat{\lambda}_N$ may be taken as a good approximation to λ provided the sample size N is large enough.

It is a robust method to analyze the variations properties of a circuit.

The manner to apply Monte Carlo Simulations in the integrated circuit design can be described in the following steps:

1. Generate values by inputting random[5] values in the probability functions of the circuit components.

2. For each sample, a netlist is created.

3. Simulates all the generated netlists using SPICE. From each sample and simulation, one or more output values are measured.

4. Generate a probability distribution function of an output.

Depending on the confidence and accuracy needed for the specific problem, a Monte Carlo simulation could involve thousands or tens of thousands or millions of recalculations before it is complete.

Therefore, this method is suitable to use in any circuit with any number of varying parameters whenever the statistical distributions are available. It has become the standard technique for statistical analysis and modeling of integrated circuits [19–21].

By using a random algorithm to get the values from the probability functions of the circuit component, some issues show up. As shown in figure 2.10, some clusters [6] appear (inside the blue circle) and regions with no points (red circle). That poor spread leads to an unnecessary higher estimation error. Furthermore, to correct this error, it is necessary to have more MC samples to represent all regions in the same way otherwise the yield [7] estimation will be poorly determined. This means that the confidence interval for the yield will shrink fairly slowly, and therefore yield verification will take more simulations. To conclude, there is no need for randomness in MC sampling, what is needed is a fair full-coverage of all regions in the variation space [13].

[5] It is also possible to low-discrepancy algorithms.
[6] Regions with points nearly overlap.
[7] **Yield** is the amount(%) of pre-set circuits specifications reached.

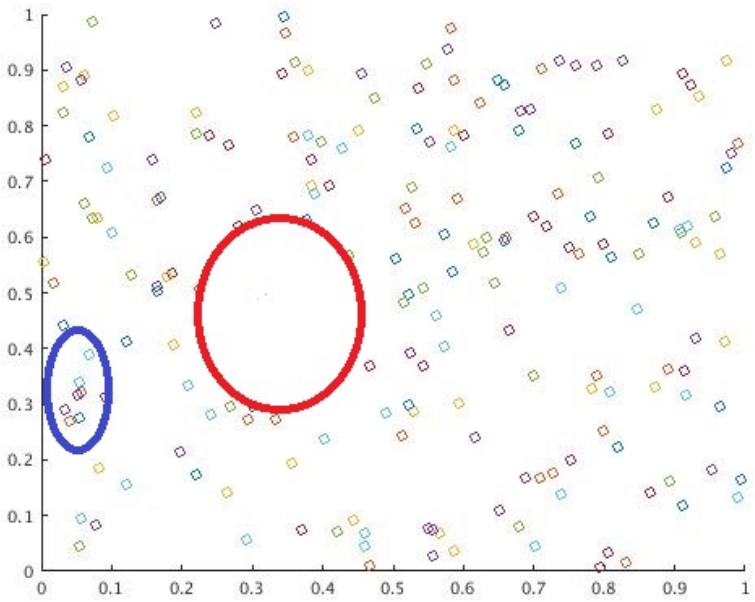


Figure 2.10: Spread Issue.

2.4 Standard design flows

In this section, two types of designs will be explored. First, the fastest and the most vulnerable, the design that does not take into account variations. As a consequence, the performance of this design flow is highly susceptible to any type of those variations referred before in section 2.2.2. This flow is shown in figure 2.11.

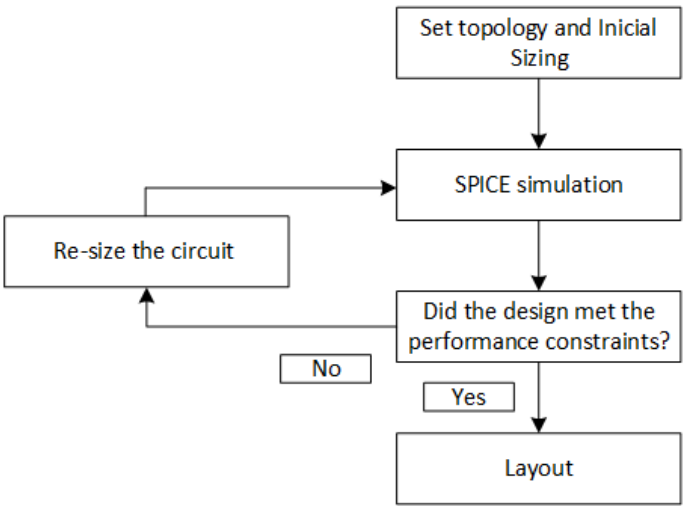


Figure 2.11: Variation unaware design flow - Adaptation from: [13].

In furtherance of getting a better design, accuracy, and fault-tolerance, the design

shown in figure 2.12 involves an improvement by approaching variations. Thus, the first step is followed by setting the corners that represent the user environment, for example, the worst case of temperature and power supply voltage. This step is followed by a Monte Carlo simulation to verify the global and local variation of the manufacturing process. If the results are within the performance constraints, this step is followed by the layout genesis, extract parasitics[8] and verification, that also includes a Monte Carlo verification. This way, it is possible to explore all cases which means the design is way more prepared to avoid fails in the user-final environment.

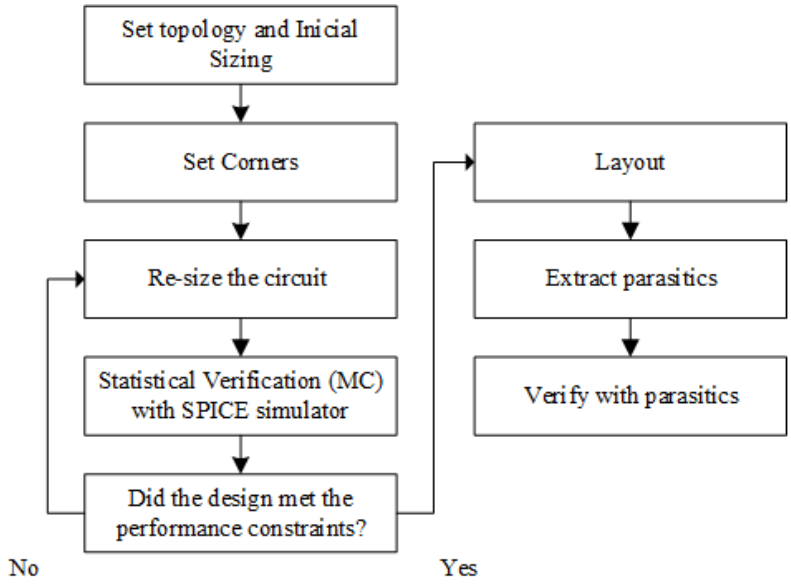


Figure 2.12: Variation aware design flow - Adaptation from: [13].

This approach also has some drawbacks, such as:

- To cover all the variation spectrum, it will need all the PVT corners which means a lot of simulations and time.
- For high-sigma designs, it is necessary to have billions of MC samples. The amount of necessary samples raises as the probability distribution functions get larger.
- Leads to either slow or less-accurate designs.

[8] If available, a parasitic reduction process should be included in order to save resources during the simulation.

2.5 Differences between 3Sigma and High-Sigma verification designs

In RF and analog designs, the standard methodology sets boundaries of three standard deviations[9] ($\mu \pm 3\sigma$) to measure the impact of variances on the circuit performance. Within this interval, it is possible to cover 99.73% of the variation space, assuming that the stochastic process is normally distributed[10], as shown in figure 2.13.

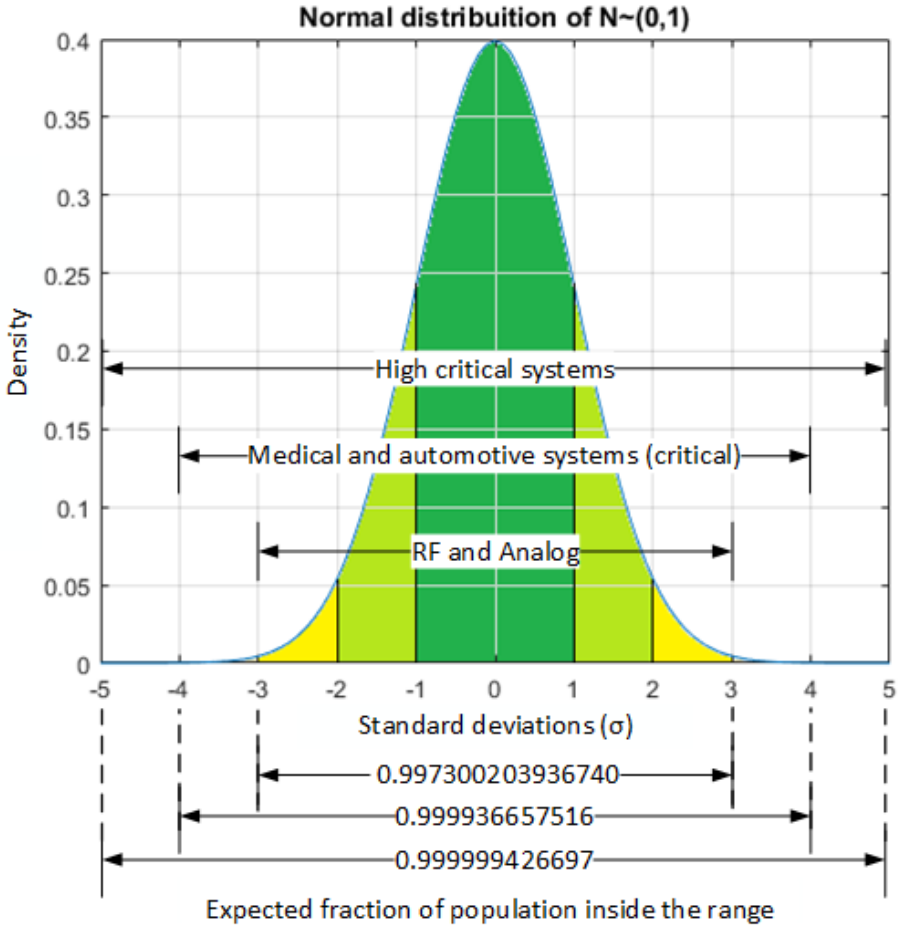


Figure 2.13: Difference between sigma designs.

High-yield estimation/verification is a mandatory request on devices that have high volume (i.e., memory devices), or in a critical system that one fails can lead to catastrophic consequences. In memory designs, one failed memory cell out of millions of cells will cause the whole memory circuit to fail without ECC (error checking and correction) techniques. That is why memory designers have high parametric yield requirements for

[9] Standard deviation can be defined as sigma, σ, stddv or s.
[10] Gaussian or Normal distribution.

the SRAM core cell. It is required no fails in hundreds of millions or billions of Monte Carlo simulations, if foundry statistical models are accurate up to the high sigma region. Memory circuit designers also have high yield requirements for other circuit block and memory partitions, such as sense amplifier for critical-path partitions.

For very high critical systems, 6-sigma designs are also used which include tests to determine circuit failures that are less than two parts-per-billion[11]. Using the traditional Monte Carlo analysis is completely impractical [22, 23].

2.6 Sensitivity Analysis

Sensitivity Analysis allows the designer to identify which components are critical, regarding the measurement goals of the circuit design. In other words, the sensitivity analysis tool examines how much each component affects the circuit behavior by itself, for each output.

The most known method to run a sensitivity analysis works by applying a variation on each component (or each parameter of a certain component if this has more than one parameter)[12]. This step is followed by a simulation where is measured the impact of this variation in the circuit response (for each output).

For instance, if a circuit has a capacitor C_1 and a response R, a variation $C_1' = C_1(1+x)$ will generate a response R', as shown in figure 2.14.

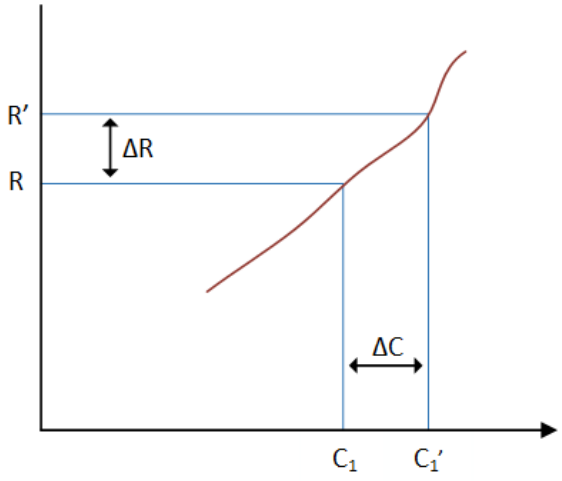


Figure 2.14: Sensitivity Analysis example

[11] 99.9999998% of coverage of the variation space.
[12] This method is named as "One Factor At a Time" (OFAT).

Where the sensitivity is equal to:

$$\text{Sensitivity} = \frac{\Delta R}{\Delta C} \tag{2.6}$$

In case of AC-Sensitivity analysis, in the frequency domain, lets for instance H(s) be the transfer function of a circuit network. In this case, the ac-sensitivity is defined to be the relative variation of H(s) regarding the variation of a circuit parameter, defined as p, which was written in [24] in the normalized form:

$$\text{Sens}(H,s) = \frac{\partial H(s)}{\partial lnp} = \frac{p}{H(s)}\frac{\partial H(s)}{\partial p} \tag{2.7}$$

In this way, it is possible to distinguish which components (or their parameters) have more impact, and if needed, redesign the circuit against high dependencies.

It also allows the designer to vary automatically all tolerances to create worst-case (minimum and maximum) measurement values.

Sensitivity Analysis can be used for optimization. After making this analysis, in order to evaluate yield versus cost trade-offs, it is possible to tighten tolerances of the sensitivity of the components and loosen tolerances on non-sensitive components. The typical flow to do this analysis is shown in figure 2.15.

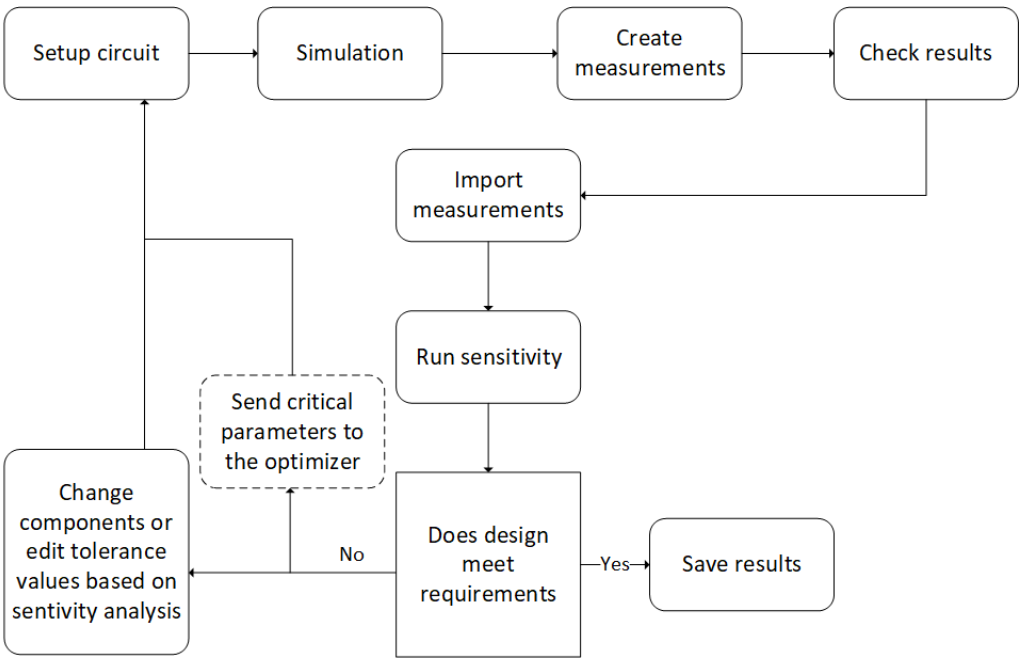


Figure 2.15: Sensitivity flows.

2.7 Verification and Validation

Nowadays, circuits can be composed of several billions of transistors size and ensuring that they work correctly is becoming increasingly difficult. Designers must use verification (testing the design before the chip is built) and validation (testing the fabricated chip) intensively [25].

Placing effective testing is crucial but pre-implementation verification boundaries are imposed by:

- Simulation coverage.
- Complexity.
- Design-Time.

The lack of control and observability places validation limits once devices are implemented in silicon. As the performance and complexity of these devices increase, the challenge to ensure that they work correctly also increases. However, even with all this effort, it is virtually impossible to eliminate all bugs in the design before it tapes-out. In fact, statistics show that close to 50% of chips require additional unplanned tape-outs because of functional bugs [26]. This factor creates an increasing need to connect these two domains by sharing methodologies and technologies. The IC development flow could be defined in several major steps, as shown in figure 2.16.

The following key concept terms are used regarding the IC development:

- Signoff: Series of verification steps that the design must pass before it can be tapeout.
- Tapeout (official tapeout): the final result of the design process for integrated circuits or printed circuit boards before they are sent for manufacturing. In other words, is the point at which the graphic for the photomask of the circuit is sent to the foundry.
- Tape in: could be interpreted as an internal tapeout.

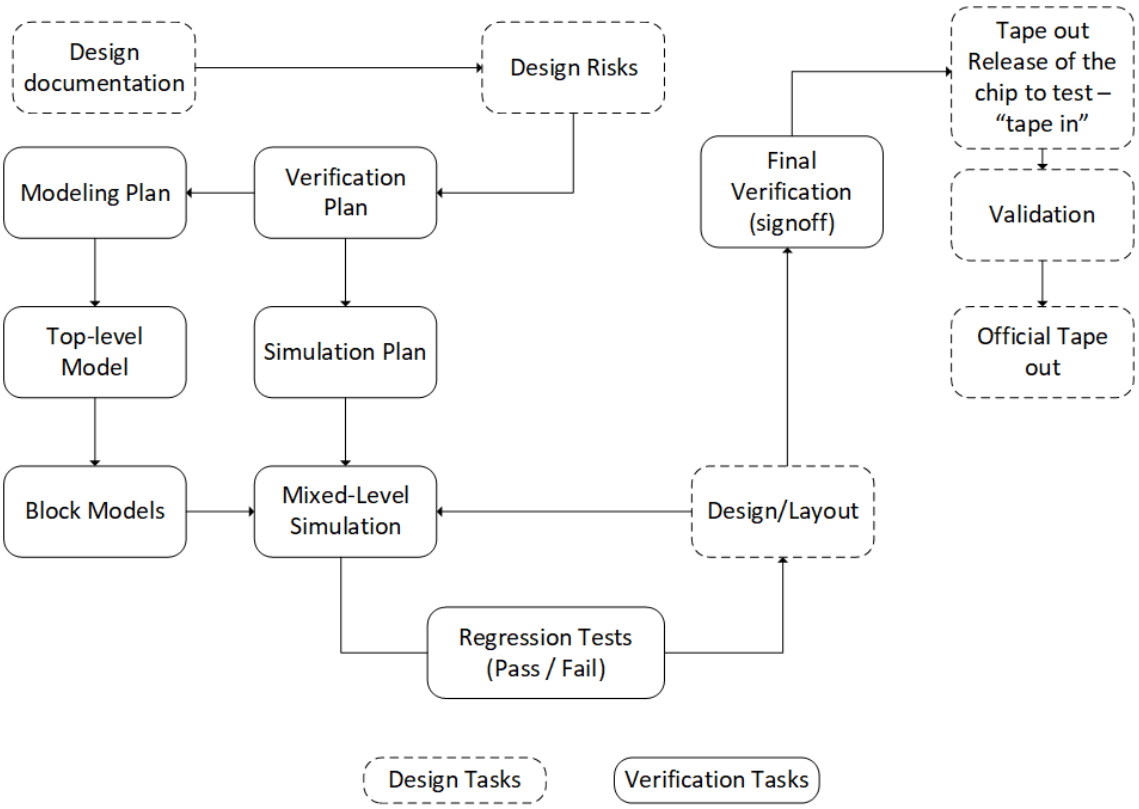


Figure 2.16: IC development flows - Adaptation from [27].

Chapter 3

Sampling Methods

Due to the necessity of fast and accurate design and verification, different methods and tools were developed during the last century, such as:

- Resorting to better pseudo-random generators such as Latin Hypercube Sampling (LHS) [19, 20, 28–32].
- Apply deterministic sampling method, so-called Low Discrepancy Sequences (LDS), such as Sobol or Halton [29, 30, 33, 34].
- Extract the worst case scenario as a statistical corner [35].
- Fast verification in high dimensional variation space [36].

In this chapter, techniques that enhance the efficiency but maintain the general Monte Carlo process (LHS and LDS) are presented. LHS and LDS allow the simulation process to get more well-spread variation spaces and stable estimations, as shown in the next sections.

A new method that combines the best characteristics of the Latin Hypercube and the Low Discrepancy is also presented. This new technique is described as an Optimized Latin Hypercube Sampling method.

The current circuit designs can have thousands of components and each component can have multiple varying parameters. The variation space of such circuits is consequently huge.

In this section, the current methods to generate the varying parameters' values are described. The pros and cons of the three methods, Random Monte Carlo, Latin Hypercube and Low Discrepancy and their functional mode will be described.

3.1 Random Monte Carlo Sampling

Monte Carlo[1] Sampling is a method that resorts to random or pseudo-random generators like the Mersenne Twister (MT) (Matsumoto and Nishimura 1998) to select values from the components' probability functions.

High requirements are imposed on these generators, for example [37]:

- Large period of the generated sequence (to avoid the sequence repeating during generation).
- Quick generation of values.
- Unpredictability of the generated sequence.
- Low correlation among subsequent values in the sequence of numbers.

As mentioned before in section 2.3.2, the Monte Carlo method has some weaknesses related to the coverage of the variation space (generation of clusters and empty spaces).

One of these weaknesses is that the generated sequence of numbers is highly dependant on the seed. In other words, some seeds lead to gaps in the variation space. Figure 3.1 shows an example of the seed's impact in the generation of empty spaces using the MT random generator.

It becomes clear that even with an increased number of samples the variation space is not necessarily better covered by those samples. The area of the largest empty space has a tendency to be reduced, but it is still large enough to be relevant.

Another weakness is the unfavorable rate of convergence of this method. The estimation error has the following dependency with the number of samples [30]:

$$\varepsilon \propto \frac{1}{\sqrt{n}} \tag{3.1}$$

with n equal to the number of samples/simulations and ε equal to the estimation error. This is quite slow and provides diminishing returns with the increase of n.

[1] Also called "Monte Carlo" or "Brute Force Monte Carlo."

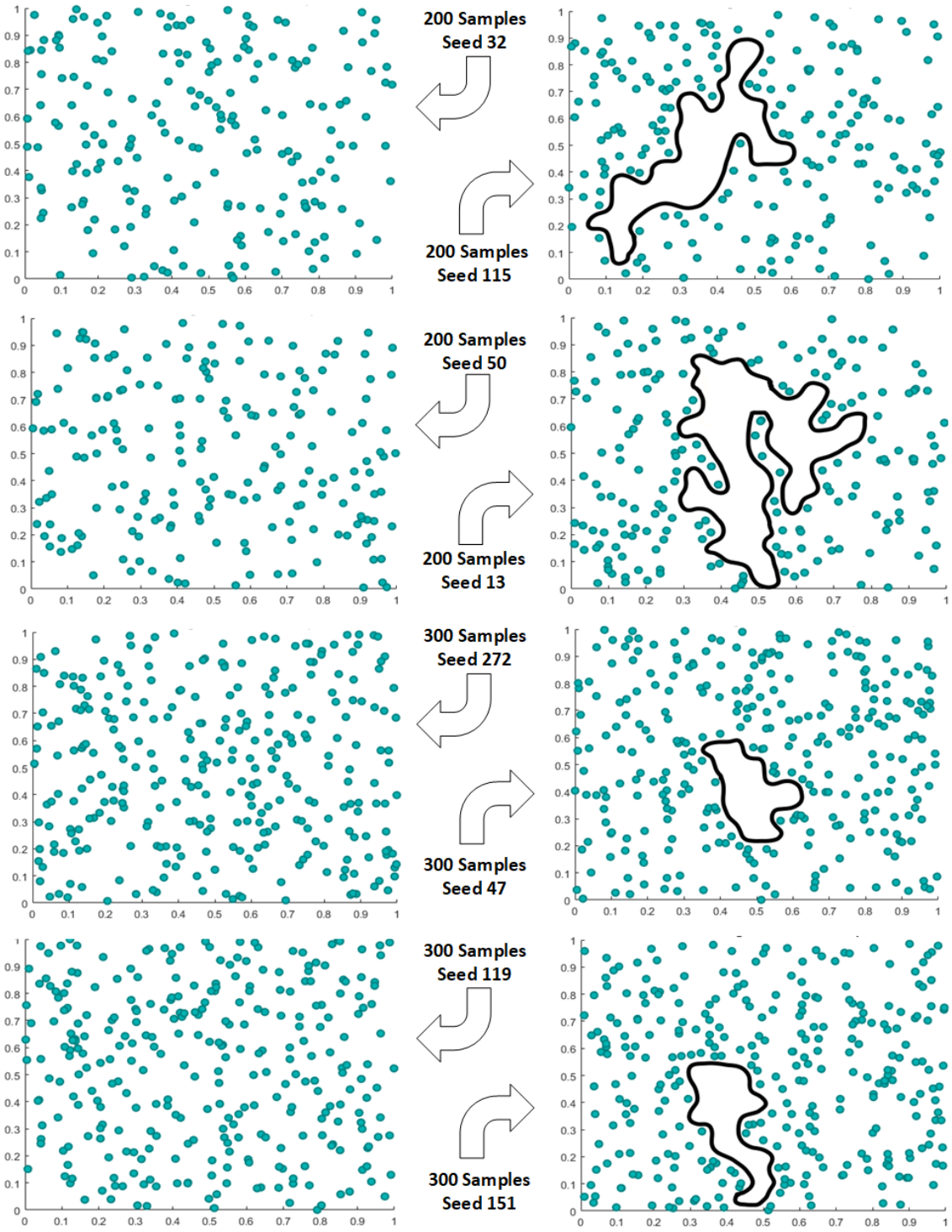


Figure 3.1: Mersenne Twister sampling issues. Left column shows relatively good spreading examples, whereas the right column shows poor spreading examples with big empty spaces. This image was generated by resorting to Matlab

3.2 Latin Hyper Cube Sampling

3.2.1 One dimensional spreading

For one dimensional spreading, the Latin Hypercube Sampling algorithm (LHS) is regarded as one of the best [30]. With this method, it is possible to achieve excellent representations of a one-dimensional variation space.

The Latin Hypercube method divides the variation space into sub-spaces (called bins) and then uses a random generator to create a sample within each sub-space. The number of sub-spaces is equal to the number of samples. For example, with a variation space between one and ten units, with ten simulations, each bin will have one unit, as shown in figure 3.2.

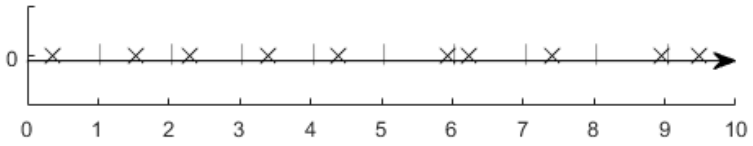


Figure 3.2: Latin Hypercube Sampling.

The mathematical formulation of this method can be described as:

$$A_i = \frac{i}{n} + \frac{U_i}{n} \tag{3.2}$$

With n LH samples, defined as $A_i, i \in \{0, 1, ..., n-1\}$, from a uniform distribution (U(0,n)) defined as U_i, and:

- The term $\frac{i}{n}$ is used to define n equal bins.
- The term $\frac{U_i}{n}$ is used to spread the points randomly through these bins.

This method has the advantage of enhancing the representation of the variation space while still resulting in a uniform random distribution. When compared with the random Monte Carlo, the improvement is clear, as shown in figures 3.3 and 3.4: LHS results in less empty spaces and clusters of points.

The global and local variations of the circuit components are normally distributed. The inverse cumulative distribution function (CDF) is used to transform samples from a uniform distribution into a normal distribution [30]. A detailed explanation about this method is shown in section 3.5.

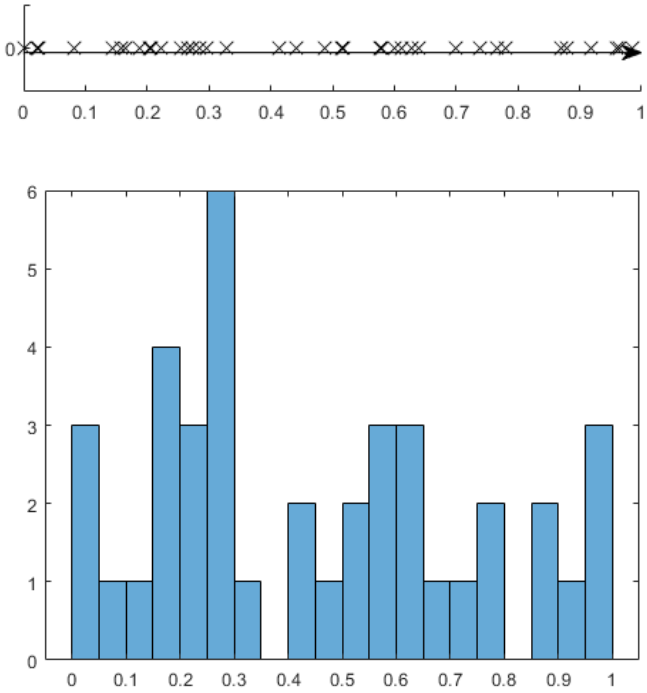


Figure 3.3: Random sampling

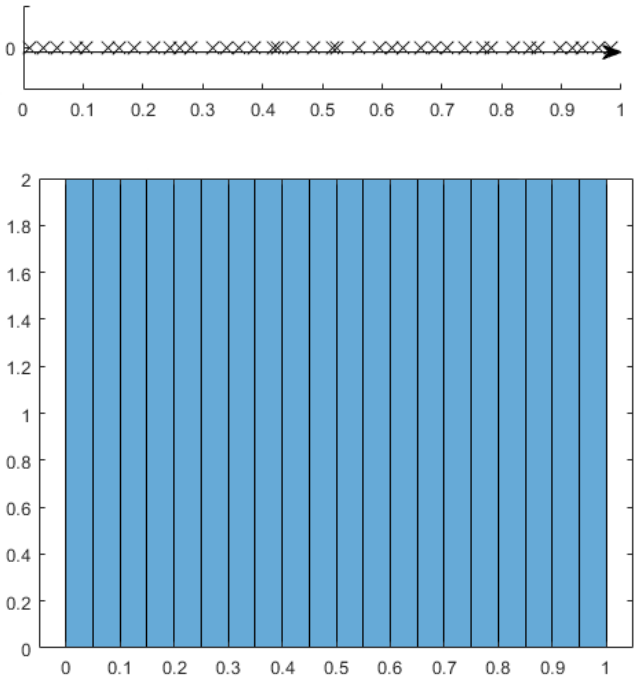


Figure 3.4: Latin Hypercube Sampling

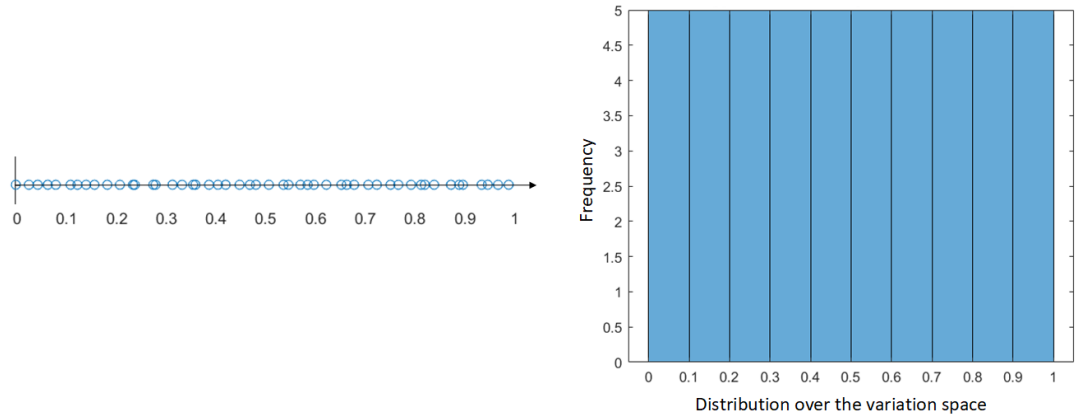


Figure 3.5: Latin Hypercube uniformly distributed.

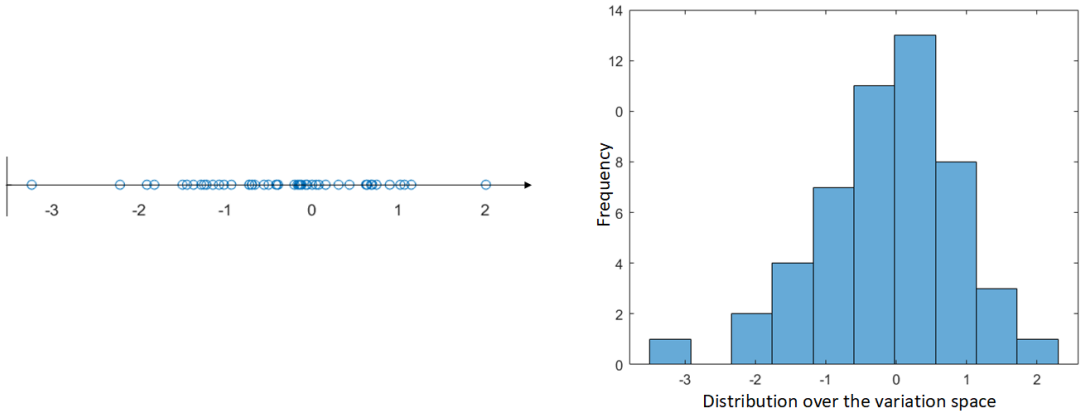


Figure 3.6: Latin Hypercube normally distributed.

To redefine the uniform set sample generated by LH, (3.3) is used:

$$A_i = \sqrt{2} \times erf^{-1}(2U_i - 1) \tag{3.3}$$

where U_i is the uniform set, and erf is the inverse error function given by [38]:

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \tag{3.4}$$

For non-negative values of x, the error function has the following interpretation: for a random variable Y which has a normal distribution, $erf(x)$ describes the probability of Y being in the interval [-x, x].

As shown in figures 3.5 and 3.6, after the transformation, the concentration of the points is clearly centered in the middle and now follows a normal distribution, as required.

3.2.2 Multi-Dimensional spreading

For multi-dimensional variation spaces, the same methodology for each dimension plus a "position" permutation of each point from one of the dimensions is applied. For example, with a two-dimensional variation space without permutations, the result is shown in figure 3.7. The first twenty samples are sorted as shown in figure 3.7, where the numbers on the figure represent the index of the vector that contains the values. Even with permutations, there is a chance of getting this order, but for this case specifically, the odds are 1 in $2.43290201 \times 10^{18}$.

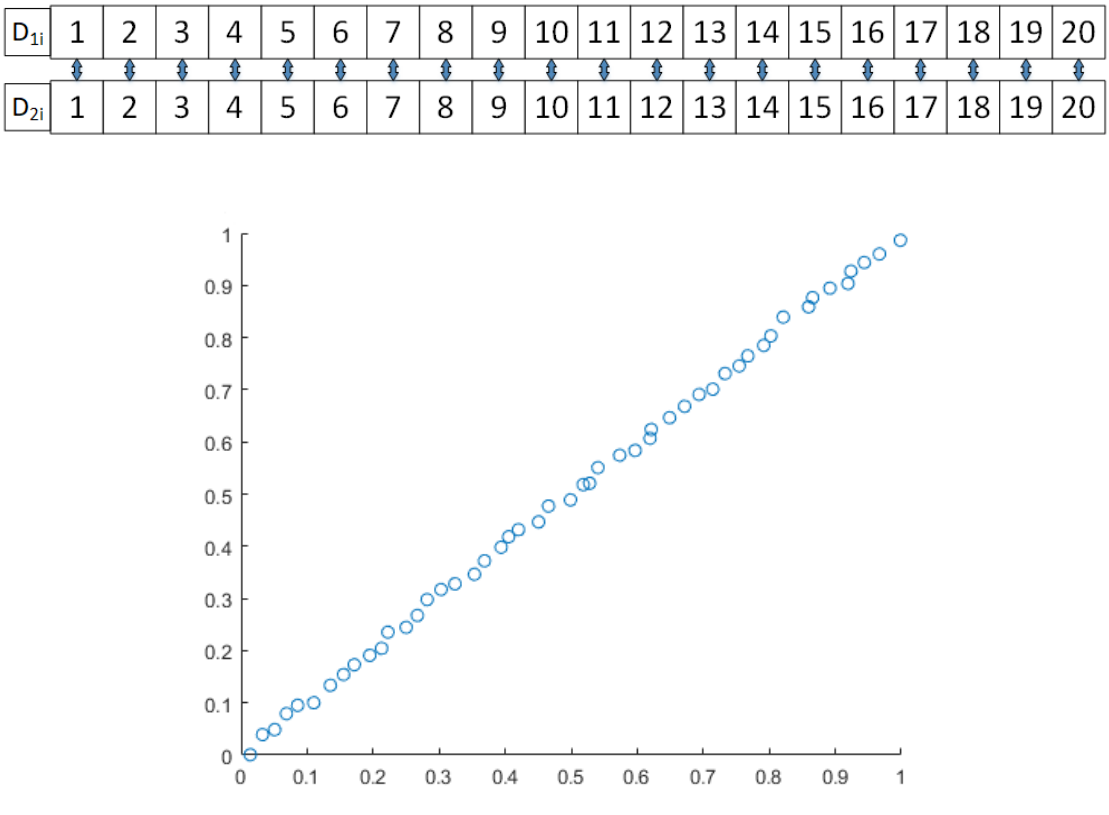


Figure 3.7: Latin Hypercube without permutations.

Performing a combination of the variables, with i representing the i_{th} index of the point from the variable x_1 and j representing the j_{th} index of the point from variable $x2$, if i is permutated resorting to random generator like MT, the spreading of the points in these two dimensions will be like the figure 3.8. An illustration of a vector with twenty positions permutated is shown in figure 3.8.

However, for multiple dimensions, the performance of this method is highly dependable on the exact permutation. In section 3.4 a new method that optimizes these permutations is shown.

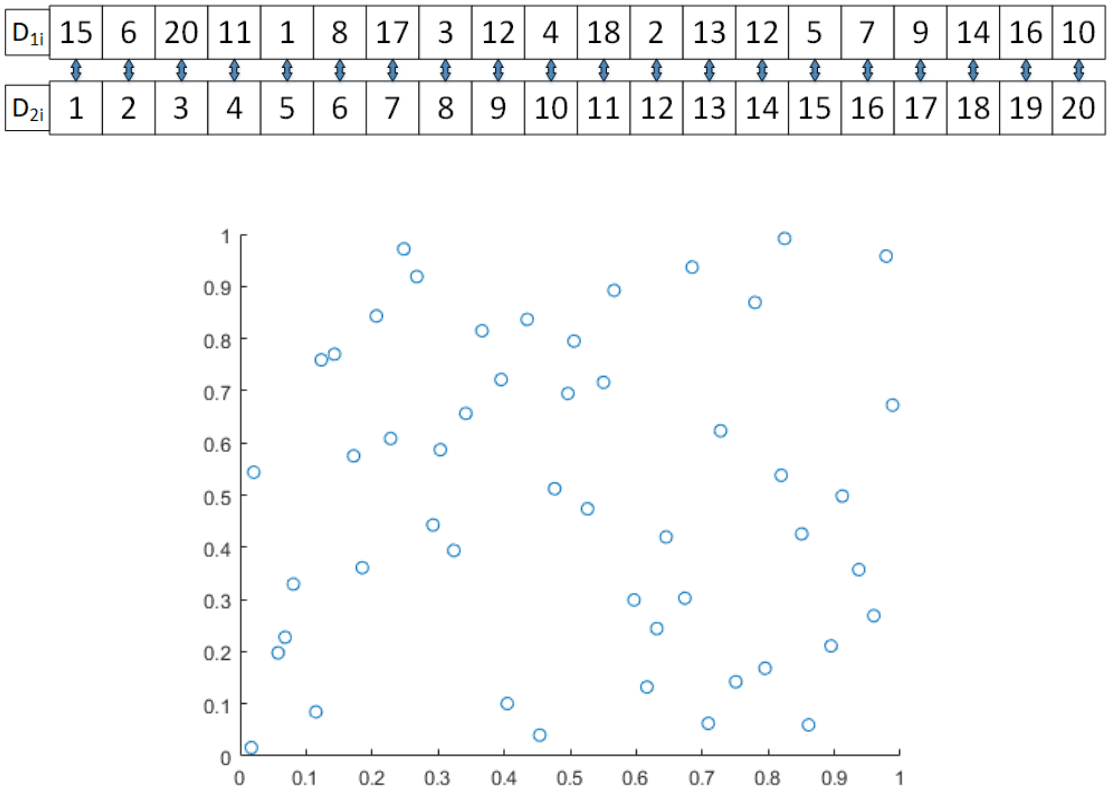


Figure 3.8: Latin Hypercube with permutations.

3.3 Low Discrepancy Sampling

Low Discrepancy Sampling[2] methods are, unlike Monte Carlo or Latin Hypercube, deterministic. The generation of points follows a complex algorithm in order to get uniform coverage of the variation space.

These methods were first tested in the finance field by IBM [29, 31] to solve high-dimensional (over 1400 dimensions) stochastic integration problems. The result was a speedup of 150 times with differences of 10^{-4} when compared with MC.

3.3.1 Low discrepancy concept

To understand these LDS methods, the notion of discrepancy must first be presented.

First, the problem is cast as a multidimensional integral, as follows:

$$I(f) = \int_{C^S} f(x)dx \tag{3.5}$$

[2] Also termed quasi-Monte Carlo.

where $f(x)$ is some function to integrate. Then, the sample mean is calculated:

$$Q(f) = \frac{1}{N} \sum_{z \in P} f(z) \tag{3.6}$$

where:

- C^S is the s-dimensional unit cube.
- P is some random or deterministic sample of N points in C^S.

It is possible to evaluate the error of the sampling process by applying the concept of discrepancy. The discrepancy can be described as a "quantity used to reflect this geometric non-uniformity of points in a set" [29] or a "measure of non-uniformity for the set of samples" [32]. Discrepancy bounds are derived from the Koksma-Hlawka inequality [32] as seen in 3.7:

$$\left| I(f) - Q(f) \right| \leq D^*(P) \times V(f) \tag{3.7}$$

The discrepancy bound thus depends on two terms:

- The first term, $V(f)$, depends only on $f(x)$ and is called the generalized variation of $f(x)$. Most importantly, this term is independent of the spreading method.
- The second term depends only on the discrepancy $D^*(P)$ and is highly dependant on the spreading method.

The discrepancy $D^*(P)$ is defined as:

$$D^*(P) = \sup_{J \in C^S} \left| \frac{n_J}{n} - Vol(J) \right| \qquad J = [0, a] : a \in C^S \tag{3.8}$$

where any s-dimensional hyper-rectangle $\in C^S$ is represented by J and n_J is the number of points within J.

For two-dimensional variation space, the discrepancy can be illustrated as shown in figure 3.9, where the star discrepancy can be defined as follows [28]:

$$D^* = \max_{a,b \in [0,1]} \left| \frac{n_{[0,a] \times [0,b]}}{n} - \frac{a \times b}{1 \times 1} \right| \tag{3.9}$$

The discrepancy for standard Monte Carlo is given by [29]:

$$D^*(P) = O\left(\frac{(log(log(n))^S}{\sqrt{n}} \right) \tag{3.10}$$

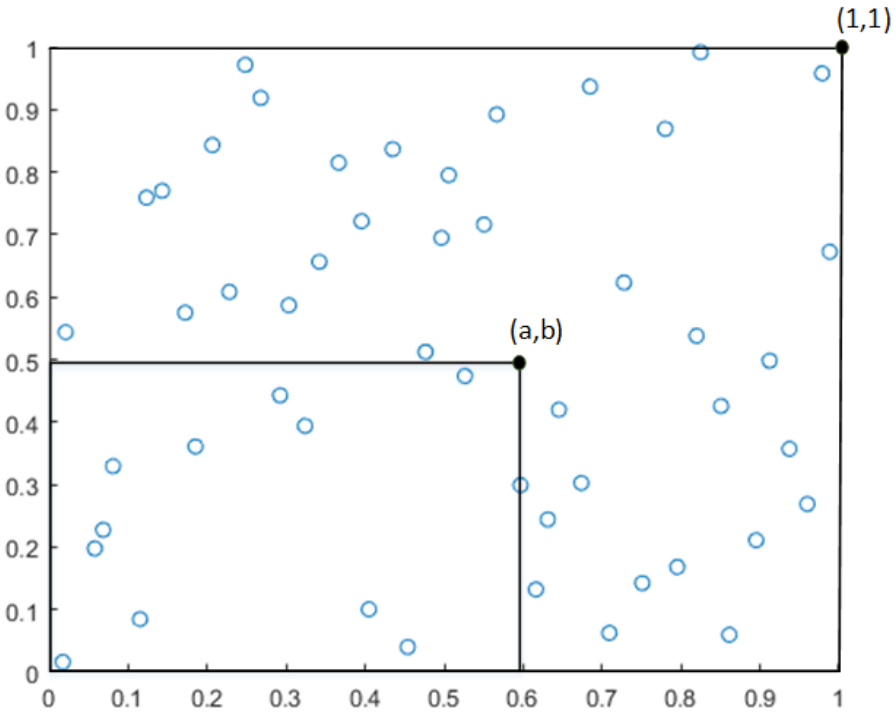


Figure 3.9: Star Discrepancy - axis-aligned anchored rectangles.

where O denotes the use of the standard *Big-O* notation.

The discrepancy for Low-Discrepancy sequences as Sobol is conjectured to be represented by [29]:

$$D^*(P) = O\left(\frac{(log(n))^S}{n}\right) \tag{3.11}$$

By analyzing the 3.10 and 3.11, comparing denominators, it becomes clear that MC $(\sqrt{n})$ will converge slowly compared with LDS (n) methods. For high dimensional variation spaces, as the numerator of the LDS expression is exponentially proportional to the dimension number, this method will be used only with a large number of simulations.

In order to solve high dimensional problems, tools such as *Analysis of Variance* (ANOVA) can be applied. These run a sensitivity analysis to identify and separate the dimensions that have a more significant impact on the output [29].

3.3.2 Low discrepancy algorithms

There is a large list of Quasi-Random Generators [39], and the focus will be on implementations that employ the following sequences:

- Sobol's 370[3] and 16384[4] sequences [29, 39].
- Halton sequence [40].

In [29] it is possible to see an implementation of Sobol's sequence, with an optimization by resorting to the Grey Code, simplifying the XOR operation and thus turning the algorithm faster. Since Sobol's sequence is a complex and lengthy algorithm, it will be only explored the Halton sequence, as an example.

The Halton sequence is constructed according to a deterministic method that uses co-prime[5] numbers as its bases. For example, to generate a base two sequence, we start by dividing the interval (0,1) by $\frac{1}{2^n}$ terms, with $n \geq$ number of samples. Then, a sum of the terms to fill the positions on the sequence is made, as shown in figure 3.10.

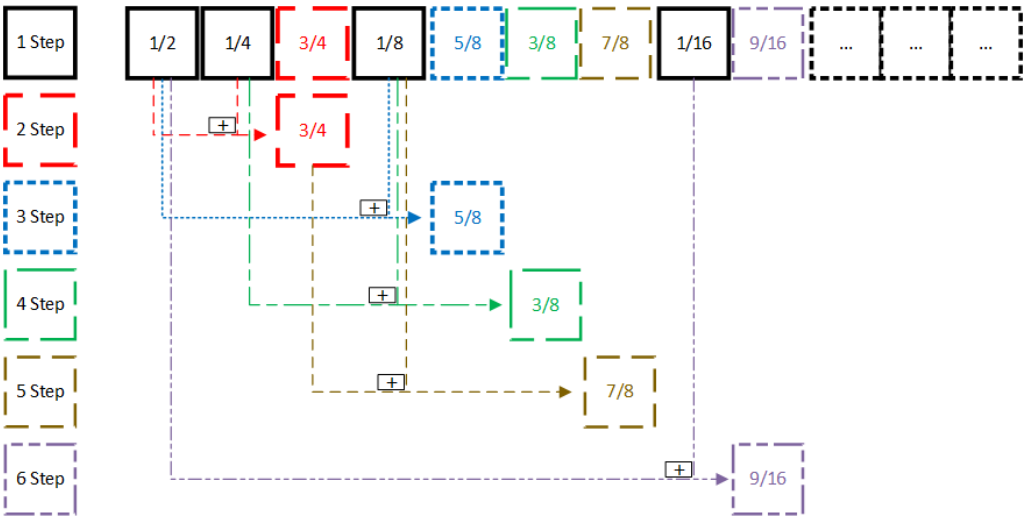


Figure 3.10: Halton Sequence.

The Halton sequence can be coded in Matlab using the code presented in A.1 (adapted from [41]).

Comparing the star discrepancy between Halton's and Sobol's sequence, using the sequence of 500 samples from both sequences, shown in figures 3.12 and 3.11, it is possible to find out that the discrepancy of Halton is two times superior to Sobol.

$$D^*(HaltonSequenceSample) = 0.004 \tag{3.12}$$

$$D^*(SobolSequenceSample) = 0.002 \tag{3.13}$$

[3]Maximum of 370 dimensions.
[4]Maximum of 16384 dimensions.
[5]Two integers a and b are said to be co-prime if the only positive integer (factor) that divides both of them is 1.

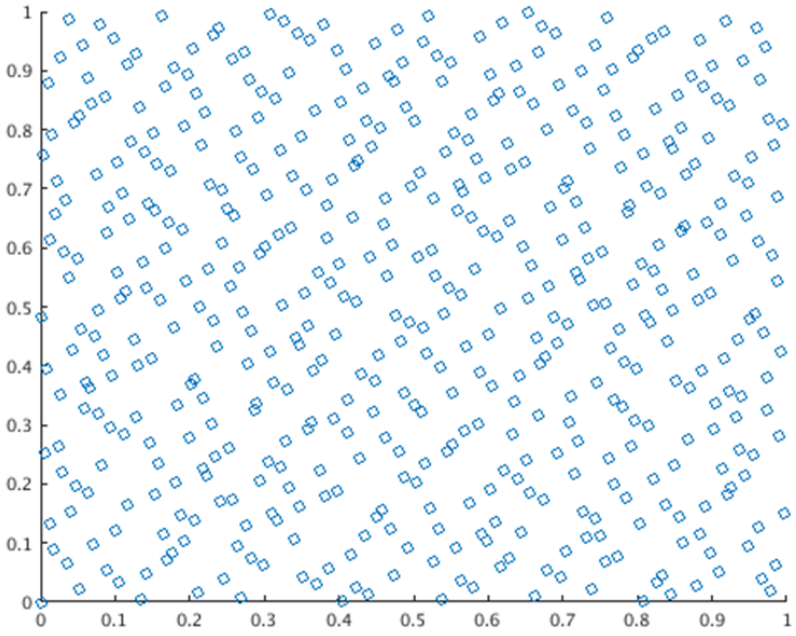


Figure 3.11: Halton Sequence – 500 samples.

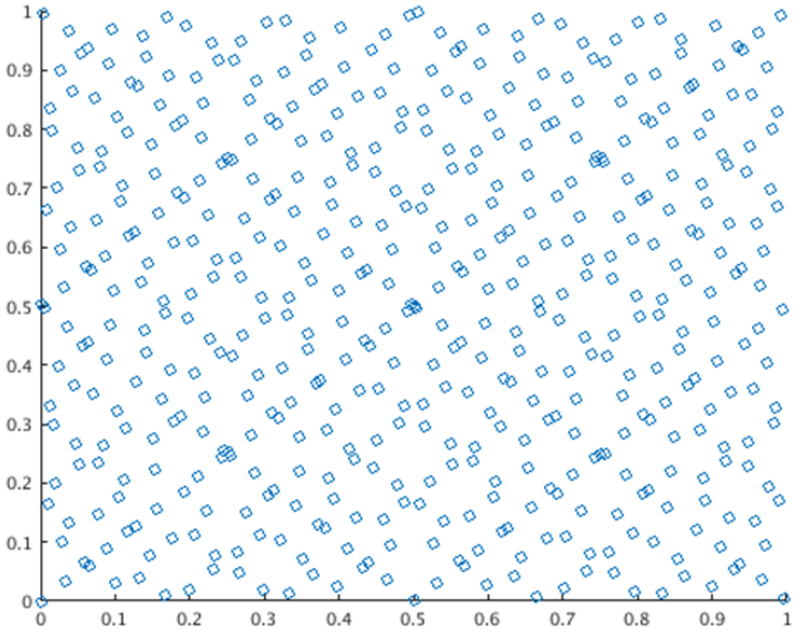


Figure 3.12: Sobol Sequence – 500 samples.

3.4 The optimized Latin Hypercube sampling scheme

Explored by [30], this method links the best characteristics of Low Discrepancy with Latin Hypercube method. It uses Latin Hypercube to generate points over one-dimension, due to the excellent spreading and projection properties that this method offers. The optimized step is the combination of all one-dimensional LH samples. To optimize this, the random manner to combine the one-dimensions samples' set, explained in 3.2.2, was changed to a deterministic procedure obtained by using the ordering characteristic from the Low Discrepancy method. To exemplify, two one-dimensional sets of ten points using Latin Hypercube were generated, as shown in figure 3.13.

LH_{v1}	0.015	0.169	0.277	0.383	0.456	0.577	0.679	0.752	0.877	0.980
LH_{v2}	0.034	0.13	0.210	0.333	0.482	0.546	0.629	0.782	0.869	0.983

Figure 3.13: Two one-dimensional LH sets.

This step is followed by the generation of two one-dimensional sets of ten points with the Low Discrepancy method, as shown in figure 3.14.

LD_{v1}	0	0.500	0.250	0.750	0.125	0.625	0.375	0.875	0.063	0.563
LD_{v2}	0	0.500	0.750	0.250	0.625	0.125	0.375	0.875	0.938	0.438

Figure 3.14: Two one-dimensional LD sets.

The two one-dimensional LH sets have the order shown in figure 3.15.

LD_{v1p}	1	6	4	9	3	8	5	10	2	7
LD_{v2p}	1	6	8	3	7	2	4	9	10	5

Figure 3.15: Order of the two one-dimensional LD sets.

By sorting the two one-dimensional LD sets, it is possible to obtain the set shown in figure 3.16.

LH_{vs1}	0.015	0.577	0.383	0.877	0.277	0.752	0.456	0.980	0.169	0.679
LH_{vs2}	0.034	0.546	0.782	0.210	0.629	0.13	0.333	0.869	0.983	0.482

Figure 3.16: Two one-dimensional LD sets sorted.

This method, evaluated in [30], shows a speedup of 2 times[6] for the Yield error (3.14) below 1% when compared with the standard sampling methods.

By using the C_{PK} method based on the process capability index to estimate yield, is possible to express the error of the yield as:

$$Err(Yield) = |Y_{CPK} - Y|, \tag{3.14}$$

where Y is the true yield and Y_{CPK} is defined as:

$$Y_{CPK} = 0.5 + 0.5 \times Erf[\frac{3}{\sqrt{2}} \times C_{PK}], \tag{3.15}$$

where C_{PK} is defined as:

$$C_{PK} = min\left[\frac{USL - \mu}{3\sigma}, \frac{\mu - LSL}{3\sigma}\right] \tag{3.16}$$

where USL stands for upper statistical limit, LSL for lower statistical limit, μ and σ are the estimated mean and standard deviation of the output, respectively.

[6] It takes two times less simulations.

3.5 Transformation from uniform to normal distribution

3.5.1 Theoretical review

The process variation distributions, global and mismatch, are usually normally distributed and their probability density function (pdf) is given by [42]:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-(x-\mu)^2/(2\sigma)^2}, -\infty < x < \infty \tag{3.17}$$

where $f(x)$ is the probability of the number x being chosen at random from the distribution.

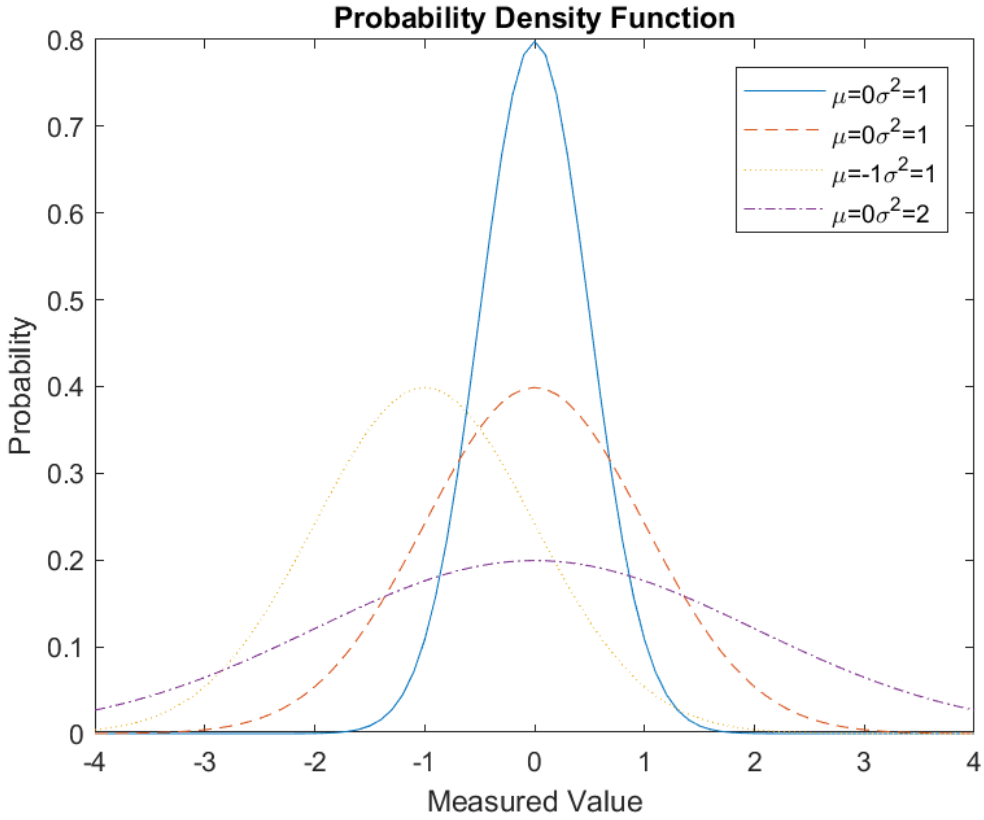


Figure 3.17: PDF with different μ and σ.

For a random variable X with mean μ and variance σ^2, and for any constant k and y, $aX + y$ is also normally distributed with:

$$\text{mean} = ku + y \tag{3.18}$$

$$\text{variance} = k^2\sigma^2 \tag{3.19}$$

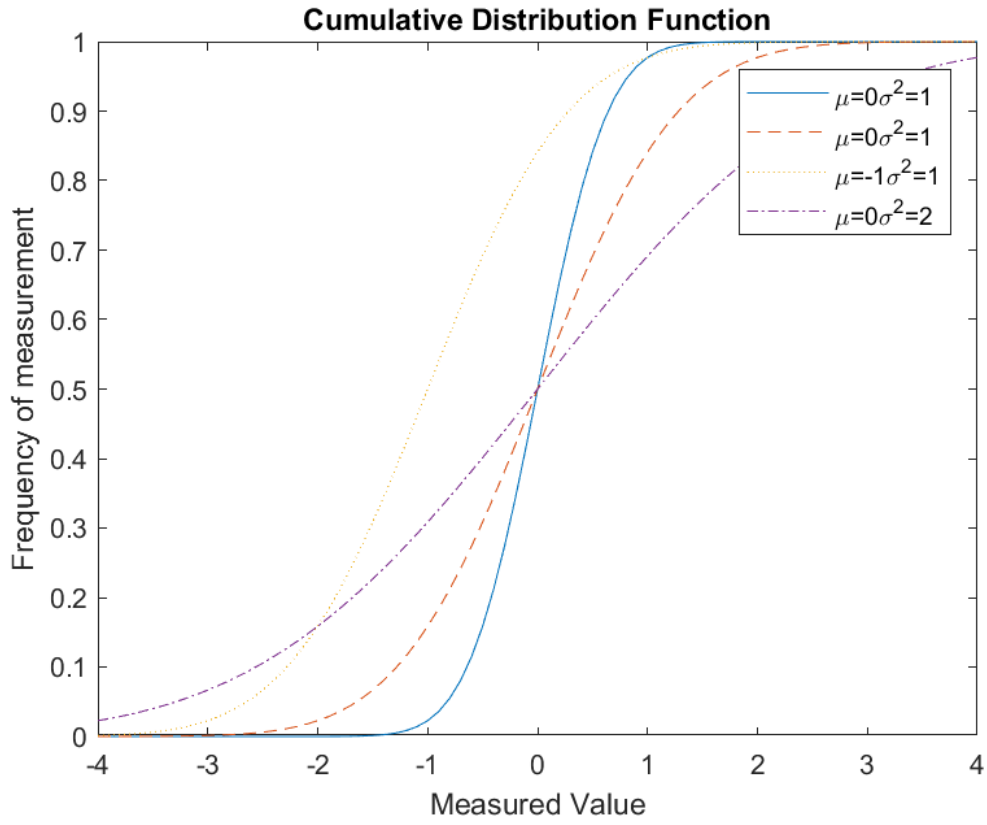


Figure 3.18: CDF with different μ and σ.

If X is normal with mean μ and variance σ^2, for a normal with mean 0 and variance 1, the random variable can be expressed as[7]:

$$Z = \frac{X - \mu}{\sigma} \tag{3.20}$$

For such distribution, the cumulative distribution function (CDF) can be obtained by integration the pdf and is given by $F(X)$ which can be expressed as follows [42]:

$$F(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-x^2/2} dx, -\infty < x < \infty \tag{3.21}$$

where $F(x)$ is the probability of the number x, or any lower number, being chosen randomly from that distribution.

For different μ and σ, the shape of a normal pdf and cdf is shown in figures 3.17 and 3.18.

[7]This distribution is named as standard normal distribution.

3.5.2 Transformation methods

There are various known methods to proceed this operation, for example:

- Box-Muller transform
- Ziggurat algorithm
- Ratio-of-uniforms
- Inverting the CDF

In this section, the Inverting the CDF is explored.

3.5.2.1 Inverting the CDF

In order to get a random number from a specific distribution, the opposite operation of what is shown in 3.5.1 is needed. Briefly explained, after generating a random number, from a uniform distribution $U(0,1)$, the same number will be "plugged" in a probability distribution, returning the number corresponding to that probability.

This method can be described as follows:

1. Generate a random number from a uniform distribution $U(0,1)$ -> U
2. Numerically invert the cdf function[8] (costly step) -> $F(X) = U$
3. Obtain Z from the previous step.
4. Obtain X from the 3.20

The shape of a normal pdf and cdf with different μ and σ is shown in figure 3.19

Is not possible to invert the PDF function because if we flip x and y's in a PDF, there would be multiple y values corresponding to the same x.

[8] Invert the cdf function is also known as the quantile function of a normal

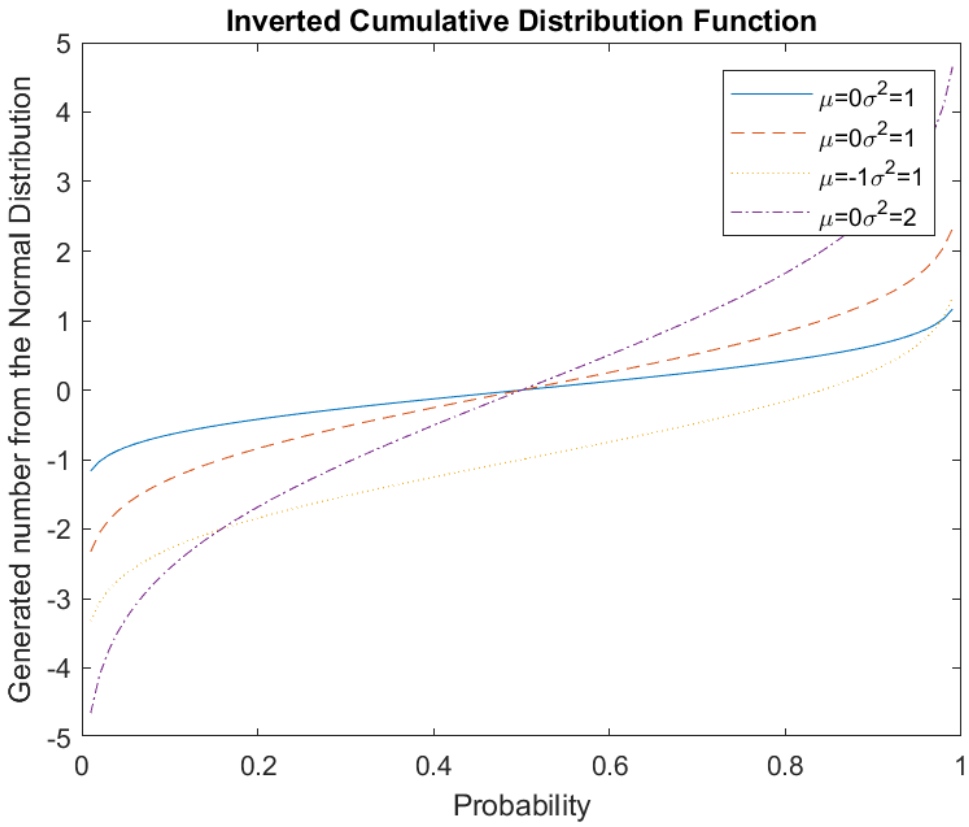


Figure 3.19: Inverse of CDF with different μ and σ.

Chapter 4

Study Cases

In the beginning of this chapter, the cases used to compare the techniques introduced in the previous chapters are presented. The result analysis of the variation test in both technologies and also the evaluation of the sampling methods are followed.

The first two circuits tested were composed of one single transistor, as shown in figure 4.1. Each circuit had a different transistor. The first had a MOSFET transistor with only four varying parameters, and the second had a FinFET transistor with seven varying parameters. The third circuit used was a modern large IC design, explored in section 4.2, with more than seventy thousand varying parameters.

Multiple Monte Carlo simulations following a mismatch variation (explained in section 2.2.2.4) were used. Each Monte Carlo simulation had different numbers of samples and three different sampling methods: Random, Latin Hypercube and Low Discrepancy.

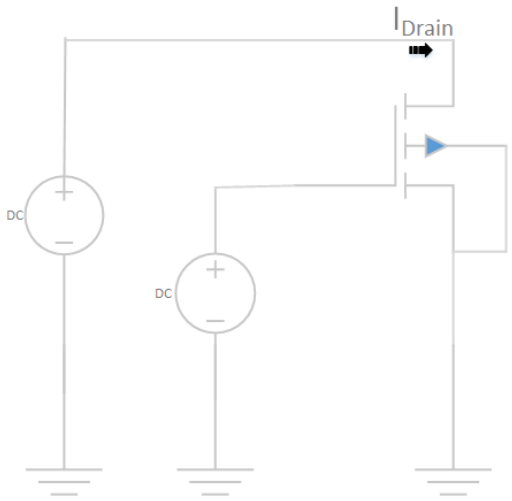


Figure 4.1: Single transistor

The comparison between each sampling method was based on the following characteristics:

- The number of clusters and empty spaces in the variation space, resorting to a two-dimensional projection.
- The spreading of the points along the variation space of each parameter, using a histogram.
- The speed of convergence of the mean by comparing to the "golden" mean reference.
- The speed of convergence of the mean confidence interval (shrink speed of the confidence interval size).

Note that all the absolute values, designs and descriptions are omitted in order to comply with Intel Corporation Confidential Policy.

4.1 Single transistor – Variation analysis in small IC designs

4.1.1 MOSFET transistor

For the first case, the MOSFET transistor had only four varying parameters, corresponding to a small variation space. By running a sensitivity analysis, it was possible to find out the impact of each parameter in the current flow of the transistor, as follows:

- Parameter 1: 0% of impact.
- Parameter 2: 0% of impact.
- Parameter 3: 68% of impact.
- Parameter 4: 32% of impact.

The varying parameters' distributions were defined in the models as normal distributions. In order to see the point spreading of each method, all the distributions were transformed into uniform distribution, as follows:

1. Normalize the data to $N(0,1)$ by subtracting the mean and dividing by the standard deviation.

$$y = \frac{x - \hat{\mu}}{\hat{\sigma}} \tag{4.1}$$

2. Transform the values using the cumulative distribution function (CDF) properties.

$$z = \frac{1}{2}\texttt{erfc}(-\frac{y}{\sqrt{2}}), \tag{4.2}$$

where `erfc` is the complementary error function.

With a uniform distribution, the performance of the sampling methods is clear.

4.1.1.1 Comparison between different spreading methods

The data used to analyze the spreading quality comes from nine different Monte Carlo simulations. A combination of three runs[1] for each method and three different number of runs, two hundred and fifty, five hundred and a thousand were used. The Monte Carlo simulations with a different number of runs were used to compare the number of runs needed to cover all the empty spaces generated by Random and LH sampling.

First, for the Monte Carlo simulation with two hundred and fifty points, the case which is possible to see the greatest payback of LDS is shown in figure 4.2 (best LD spreading and worst Random spreading).

The projection on the bottom shows a large empty space generated by the Random Sampling and also a vast number of clusters. This large empty space areas and a vast number of clusters lead to a slow convergence of the mean. As a consequence, a higher number of runs are needed to maintain the necessary accuracy.

In the same figure 4.2, is shown the spreading quality rising from Random to Latin Hypercube Sampling, with the empty spaces getting smaller when compared to the Random MC. The best method, from far, is the Low Discrepancy Sampling with all the points well spread in the variation space, without one single cluster. The LDS method provides a good representation of the variation space with only two hundred and fifty samples.

In figure 4.3 is shown the distributions of all sampling methods over the variation space of each varying parameter. The LHS and LDS had a good performance as a result of the proper spreading, almost evenly over the variation space. This factor was not clear by looking only to the projections in figure 4.2. Although, by looking at the Random sampling histogram, the lack of performance is clear, especially because of the empty bins.

[1] The SPICE simulation used to measure some output.

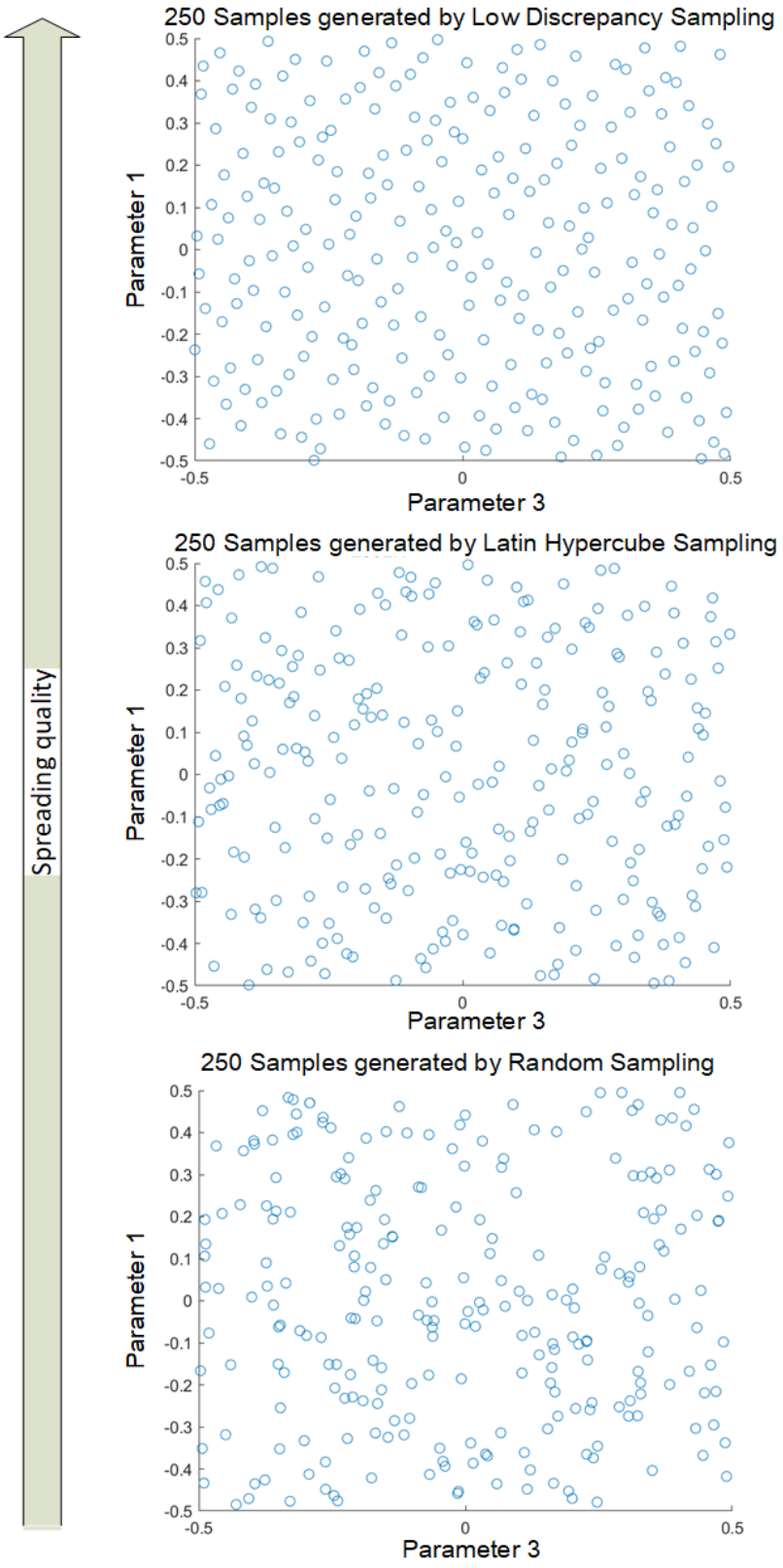


Figure 4.2: Spreading comparison

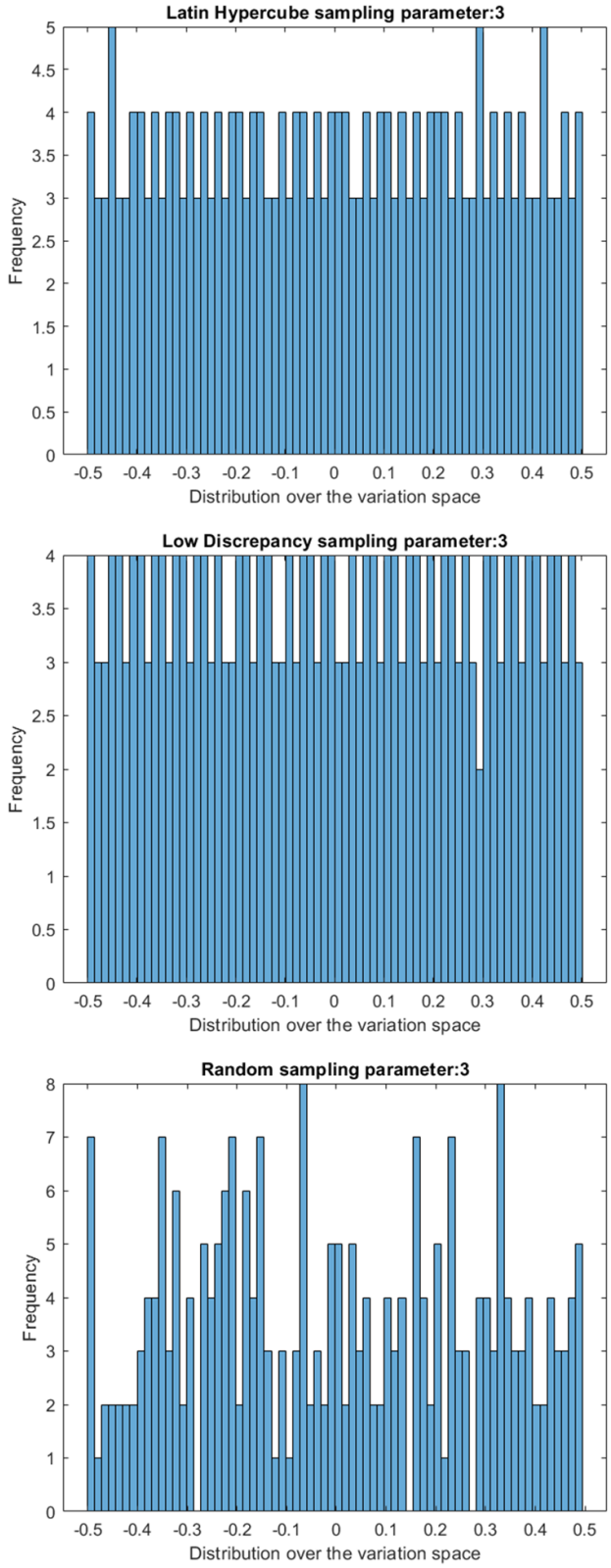


Figure 4.3: MOSFET histograms

With the rise of simulations going to five hundred and a thousand points, as shown in figure 4.4, it becomes even clearer the performance superiority of the Low Discrepancy Sampling. It is possible to see a large number of clusters with the LH and Random sampling and even some empty spaces with a thousand points. Although, with a thousand points, the LDS still with the low number of clusters and this reflects on the low number and area of the empty spaces.

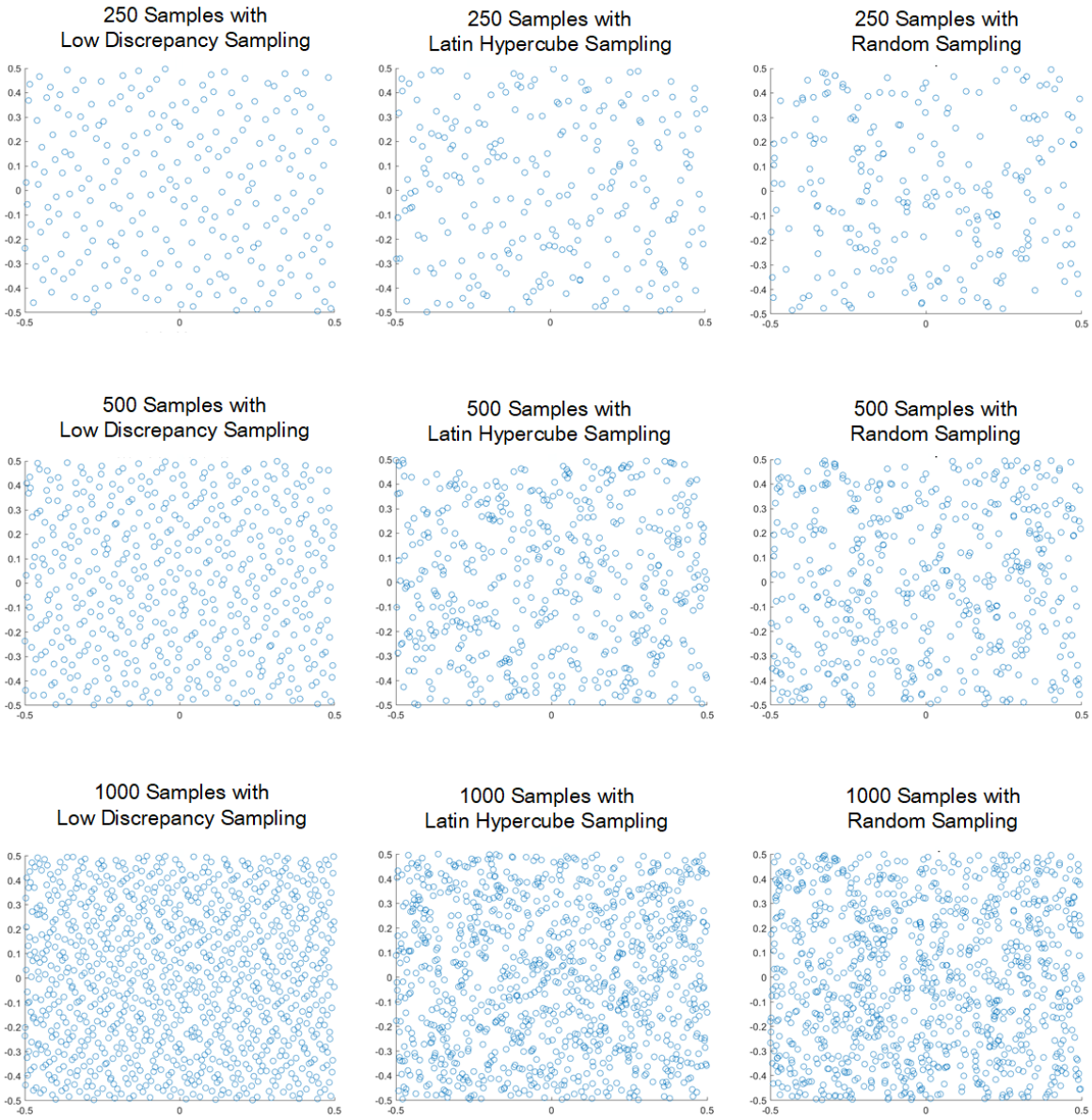


Figure 4.4: Comparison of all methods with the rise of the simulations number. Parameters 1 on the vertical axis and parameters 3 on the horizontal axis.

This spreading factor will have a major impact on the convergence of the mean, as shown in the next section.

4.1.1.2 Mean convergence for different spreading methods

In figure 4.5 is shown the mean absolute value calculated for the three methods. With LDS, after a hundred simulations, the mean value has a good approximation of the final value. For Random Sampling, with a hundred simulations, the mean value is far from the final value due to the large oscillation that is appearing during the firsts three hundred simulations. The same is occurring with LHS, with lower oscillations, but the mean can be only calculated with good accuracy ($\delta_x < 1\%$) only after two hundred and fifty simulations. It is possible to conclude that for the same circuit, the LDS offers a speedup of three times. A similar result can be visualized in figure 4.6, where the mean relative error is presented and calculated as follows:

$$\delta_x = \frac{\Delta_x}{x} = \frac{\text{actual mean value - reference}}{\text{reference}}, \tag{4.3}$$

Where Δx is the absolute error and the reference is the last LDS mean value (golden reference).

In figure 4.7 the confidence intervals for all three methods are presented. A higher stability of the LDS confidence interval in comparison with the other methods, is clear.

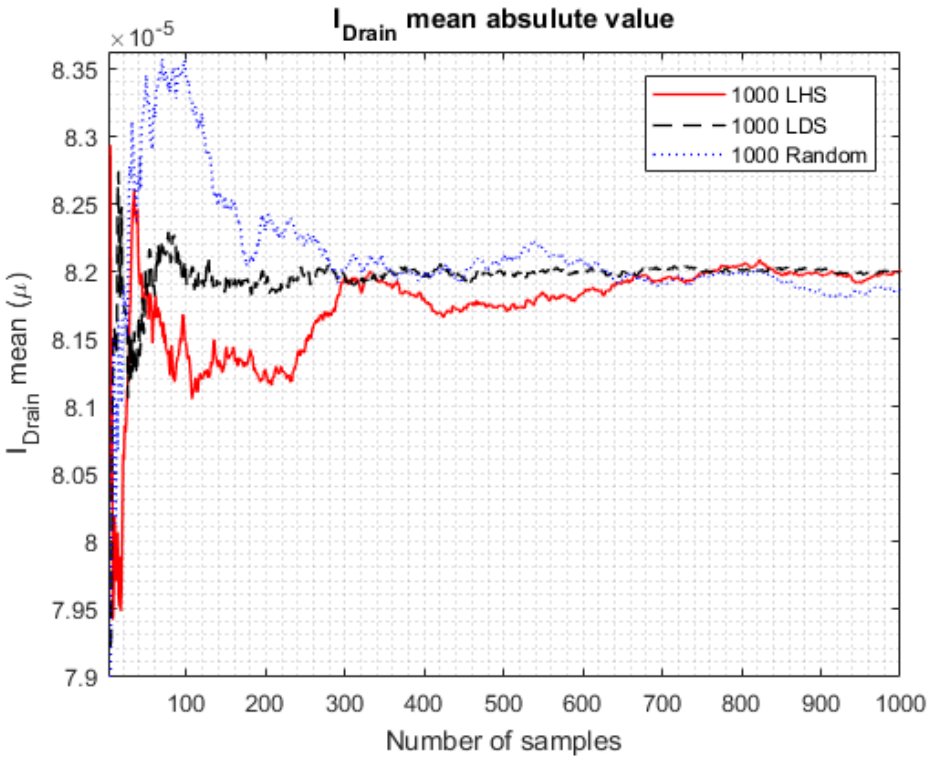


Figure 4.5: I_{Drain} Mean absolute value

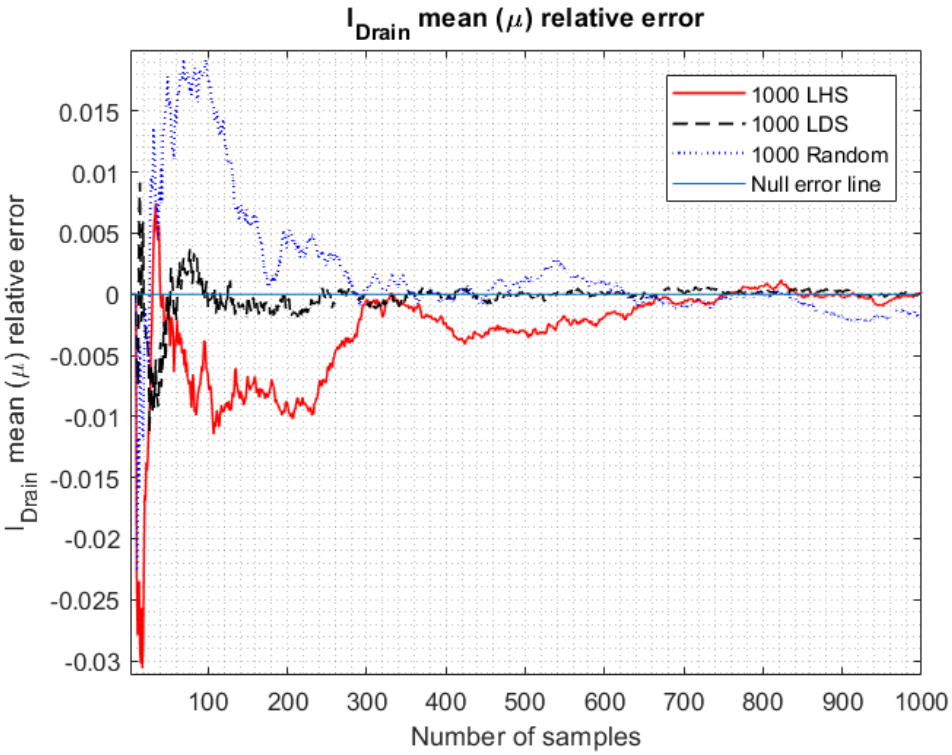


Figure 4.6: I_{Drain} mean relative error

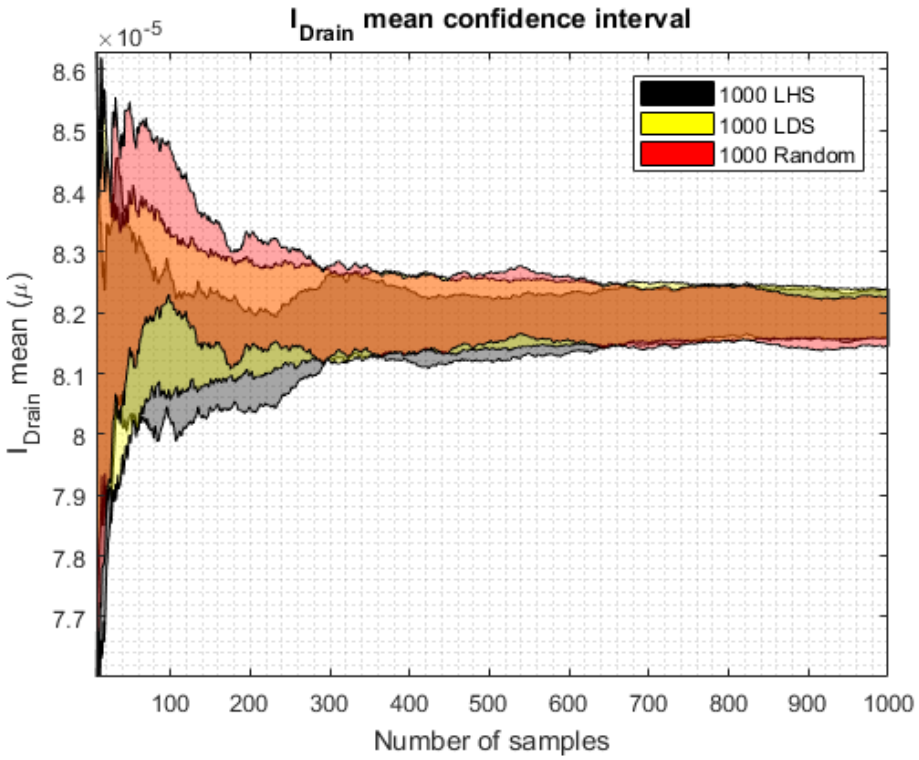


Figure 4.7: I_{Drain} mean confidence interval

4.1.2 FinFET transistor

The FinFET transistor used in this test had three more varying parameters when compared with the MOSFET, making a total of seven. By running a sensitivity analysis, it was possible to determine the mismatch contribution of each varying parameter, described as follows:

- Parameter 1 – 47% of impact
- Parameter 2 – 11% of impact
- Parameter 3 – 7% of impact
- Parameter 4 – 4% of impact
- Parameter 5 – 0% of impact
- Parameter 6 – 0% of impact
- Parameter 7 – 0% of impact

The process flow to analyze the variation was the same as the first case. This case is similar to the first one but the conclusions are slightly different, as shown in the next section.

4.1.2.1 Comparison between different spreading methods

As in the first case, the quality of the points spreading increased from Random to LHS, and from LHS to LDS, as shown in figure 4.8. However, with more three varying parameters, it is possible to see some clusters appearing in the variation space with the Low Discrepancy generator, and as a consequence, empty spaces.

By looking to the histograms shown in figure 4.9, it is possible to see that there is no big difference on the one-dimensional points spreading when compared with the first case. Furthermore, one can conclude that even with the well points spreading for each varying parameter, the variation space with multiple varying parameters can be not well represented.

The worst two dimensional projections with LDS are represented in figure 4.10. With a strange spreading pattern occurring in the projection of Parameter 5 and Parameter 4, the quality of the spreading falls really quickly. The most important varying parameters, Parameter 1 and Parameter 2, are shown in figure 4.11, where is possible to see clusters and empty spaces. The case that did not happen in four dimensions. Furthermore, this affair will lead to slowness in the convergence of the mean.

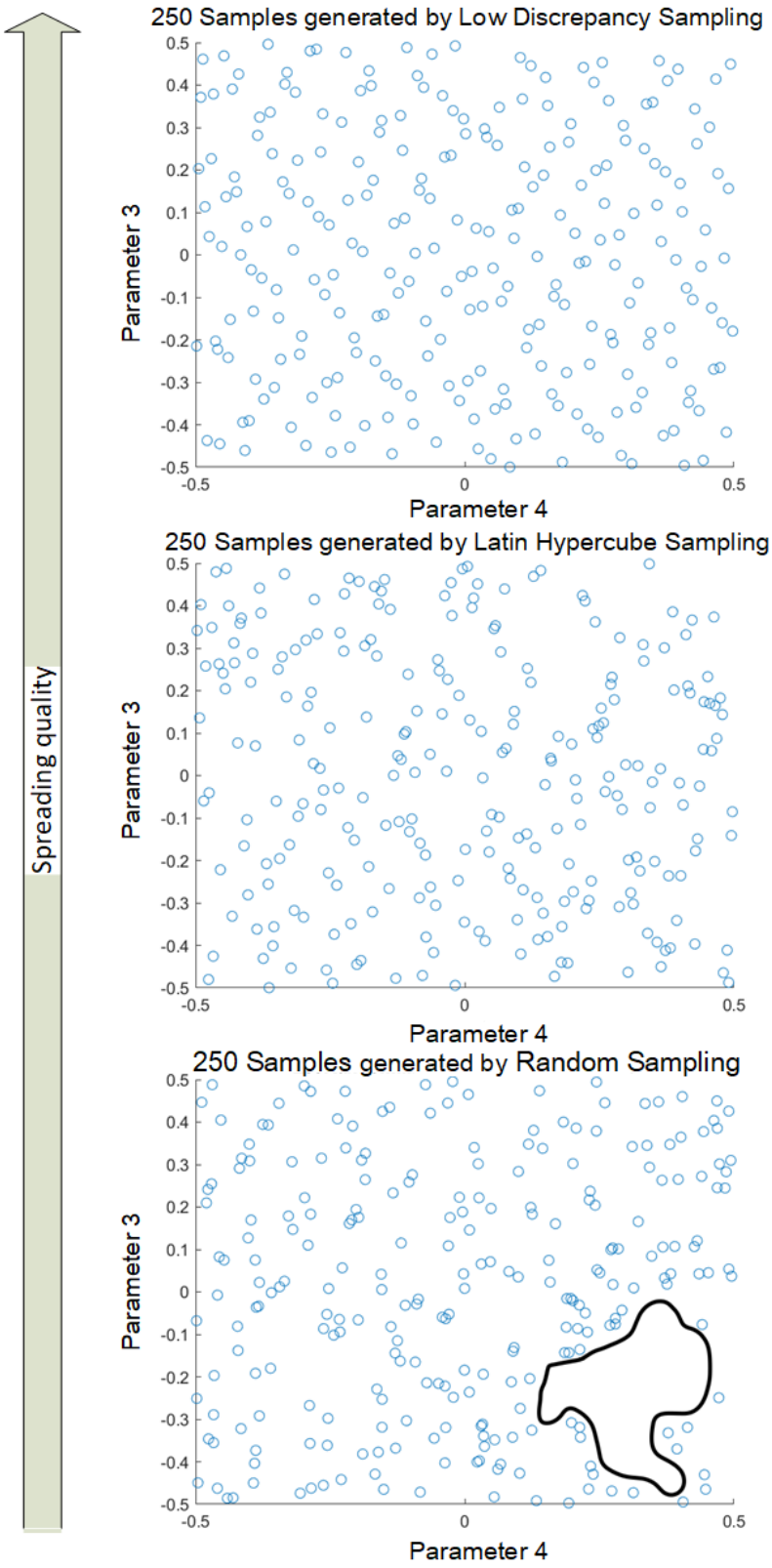


Figure 4.8: Spreading comparison

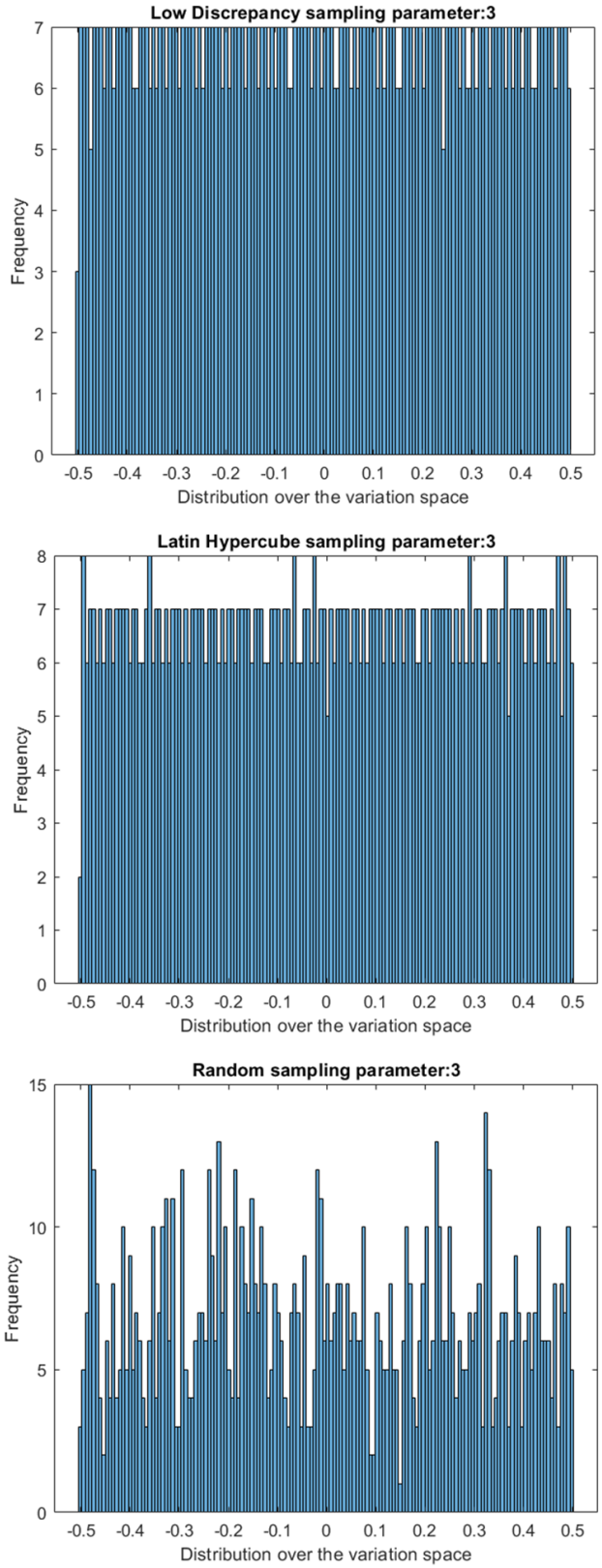


Figure 4.9: FinFET histograms

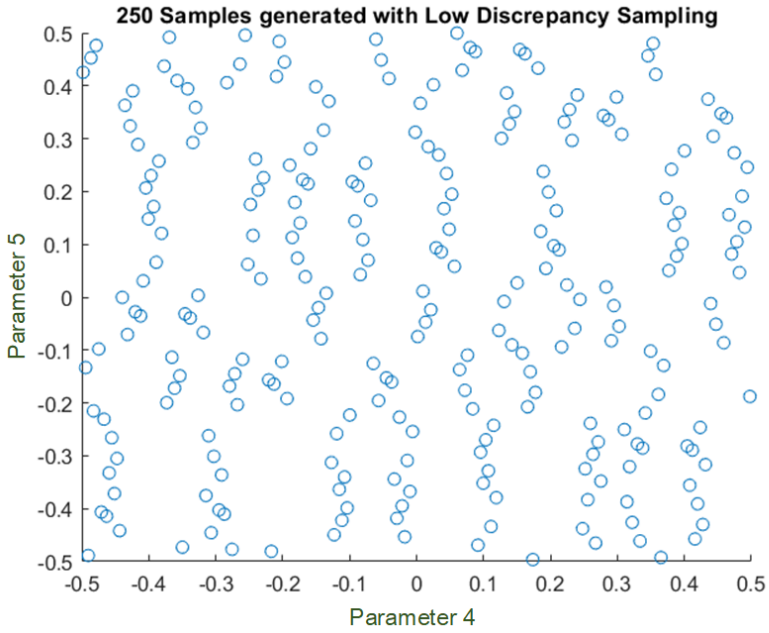


Figure 4.10: LDS point spreading - worst case

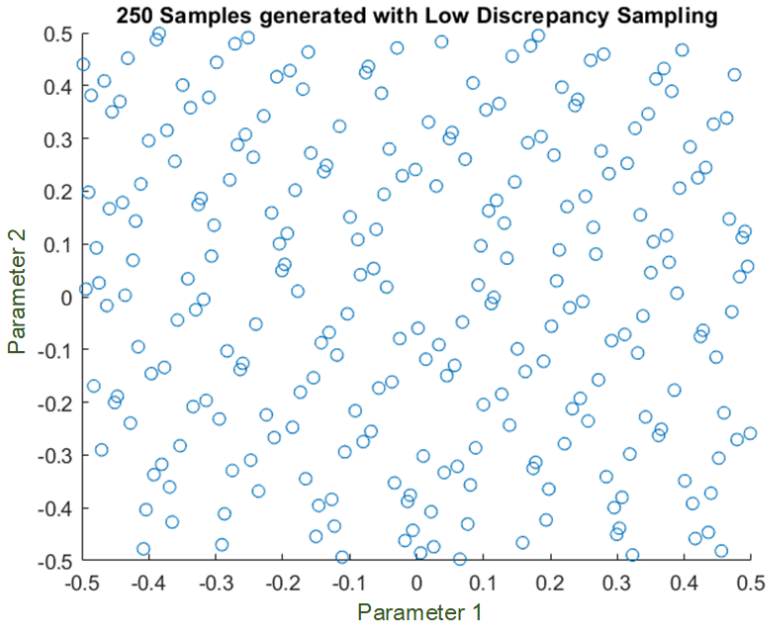


Figure 4.11: LDS Spreading - most important varying parameters

With the increase of the simulations number, shown in figure 4.12, in some projections, is possible to see that the issue is solved with a one thousand samples but not in all the cases. For example, the right column of figure 4.12 represents a case where even with a large number of samples (thousand), instead of spreading the points over the variation space, the LDS method is creating more clusters.

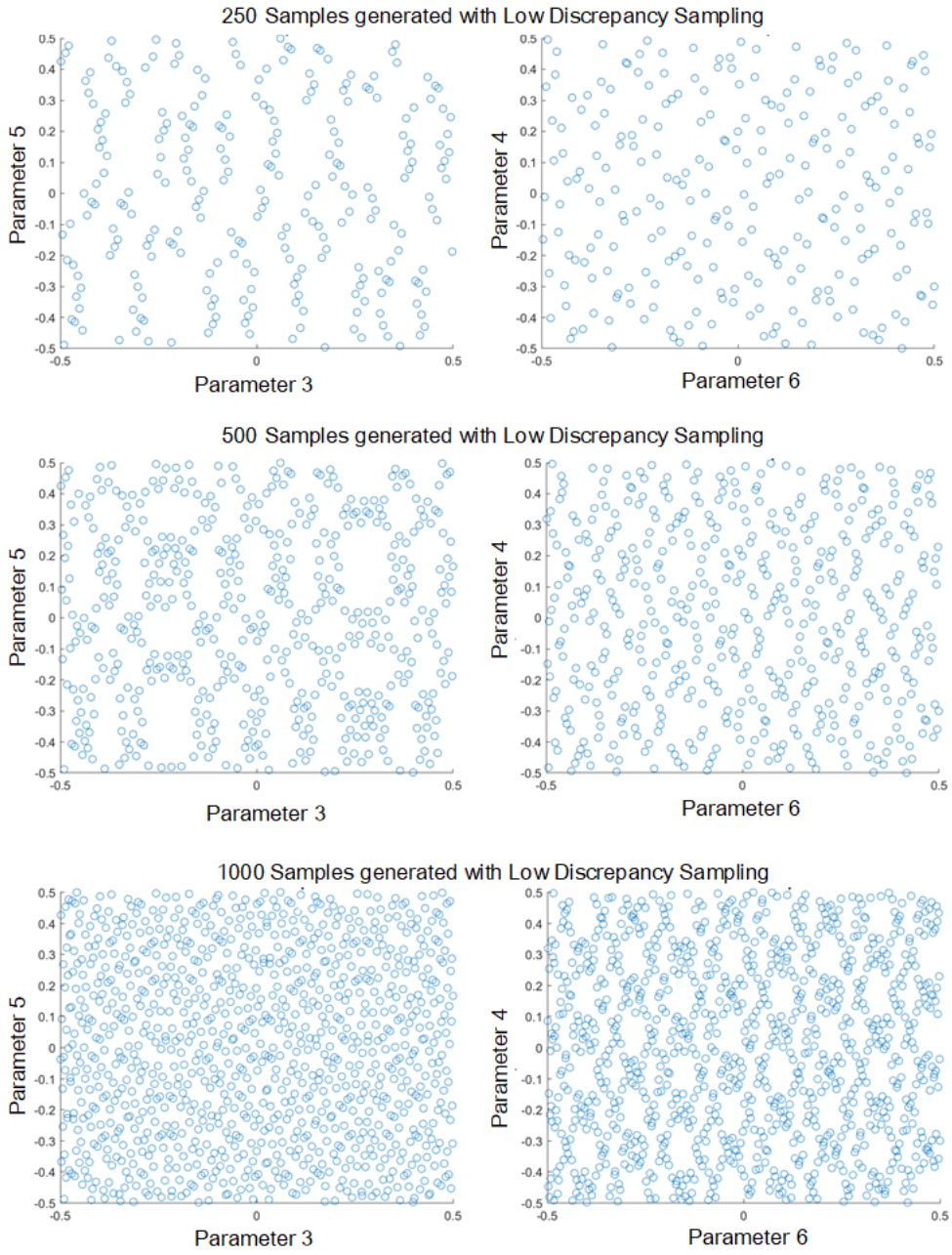


Figure 4.12: LDS Spreading comparison between different varying parameters

4.1.2.2 Mean convergence for different spreading methods

To evaluate the converge of the mean, the same procedure as in the MOSFET case was applied. Both Monte Carlo simulations with the Latin Hypercube Sampling and Random Sampling had an irregular behaviour, with multiple overshooting periods. The results of relative error comparison of all sampling methods, shown in figure 4.14, are present in the table below.

Mean relative error comparison table			
Sampling Method	Number of samples to converge above |0.5|%	Number of samples to converge above |0.1|%	MAX LDS speed up
Random Monte Carlo	325	920	3.28
Latin Hypercube	62	900	3.21
Low Discrepancy	65	280	—

Table 4.1: Mean relative error comparison table

Even with the spreading issue shown in the previous section, the Monte Carlo simulations with the Low Discrepancy Sampling had a maximum speedup of more than three times for a relative error below 0.1%, in comparison with both sampling methods.

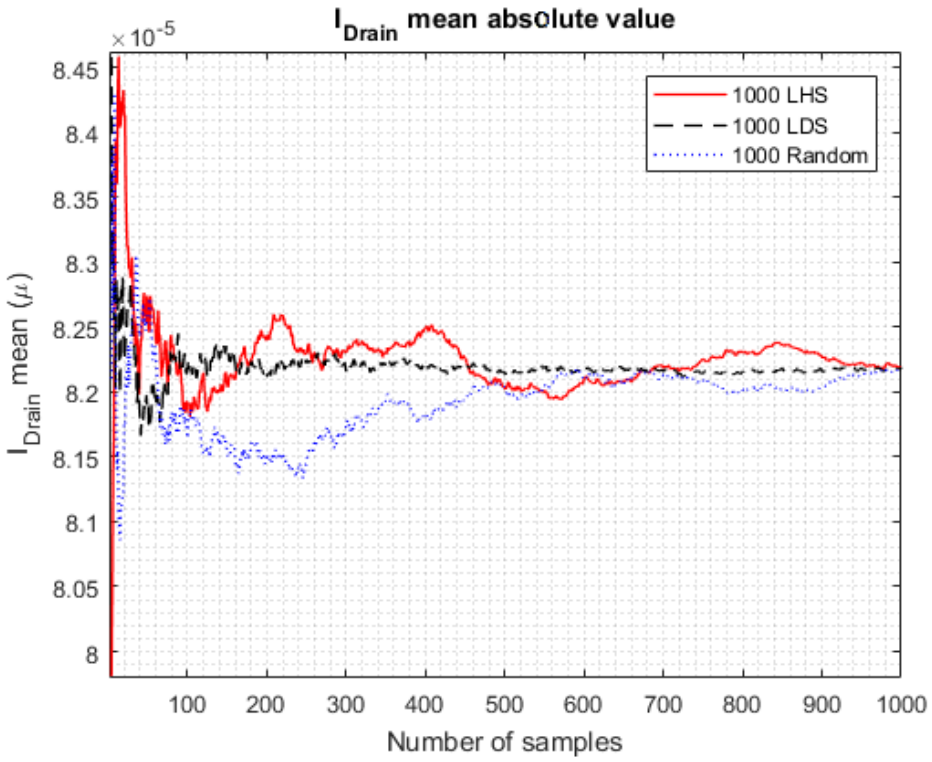


Figure 4.13: I_{Drain} Mean absolute value

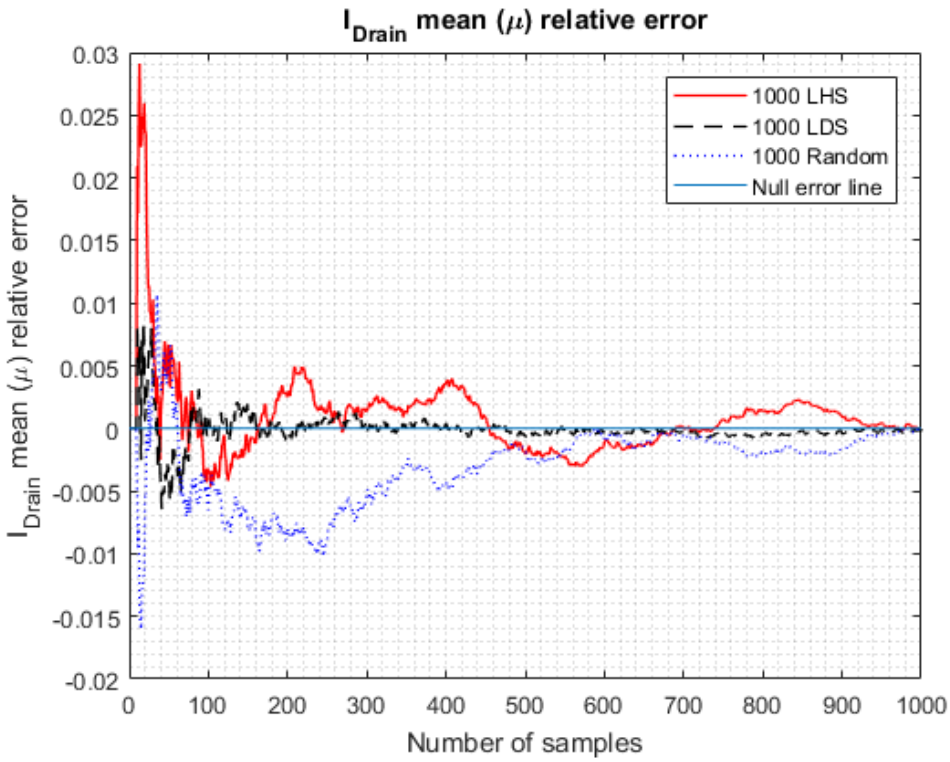


Figure 4.14: I_{Drain} mean relative error

The size of the confidence interval shrank faster with the Random Sampling compared with the LDS until approximated two hundred samples. After that period, the speed of convergence slowed down, as shown in figure 4.16. The large deviation of the Random Sampling confidence interval, when compared with the other two methods, makes the fast speed of convergence ineffective.

4.2 Micro-Receiver – Variation impact in large IC designs

In this sections, the variation impact with a modern large IC design is evaluated. The focus of this test is to appraise the method's performance with a very large variation space. Following the conjure presented in chapter 3, section 3.3, equation 3.11, a drop down on the LDS performance is expected due to the rise of dimensionality. This section is focused on testing the Low Discrepancy Sampling as well as expose the expected lack of performance during the generation of points in high dimensional variation spaces.

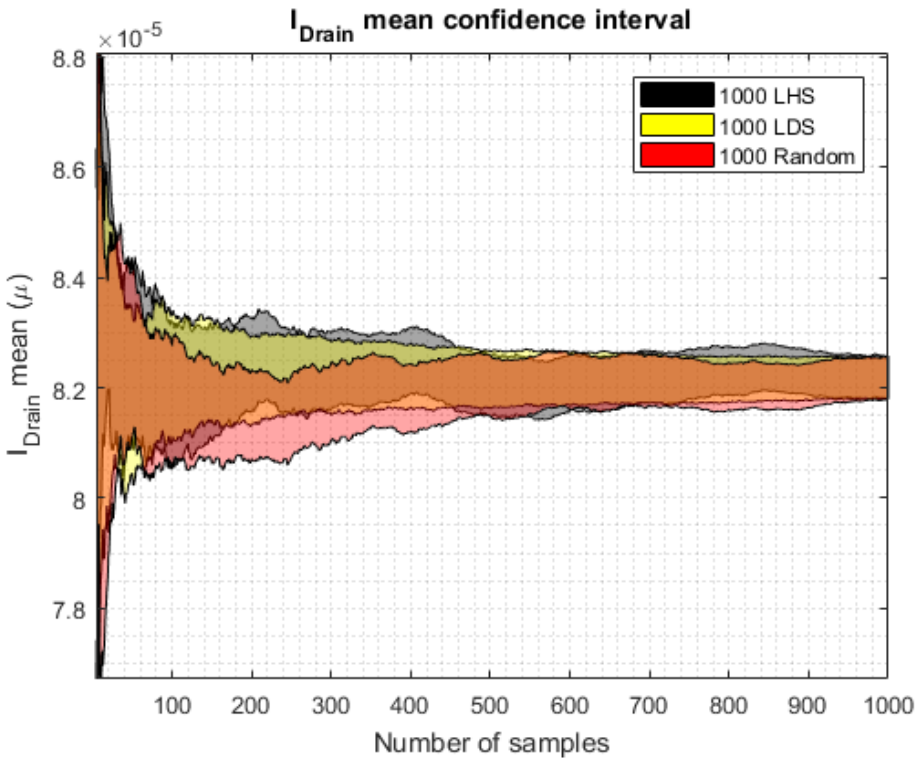


Figure 4.15: I_{Drain} mean confidence interval

4.2.1 Circuit description

The circuit had forty-six different outputs, each one with different mismatch sensitivity. To evaluate the performance of the sampling methods, two outputs with the most significant relative error and variation space were chosen, since the convergence of the mean is slower. They are identified as "Output 35" and "Output 36".

The list of varying parameters in this design is composed of more than seventy thousand varying parameters, a huge and complex variation space. To run Monte Carlo simulations, specifically for the LDS and LHS, the EDA tool needs to generate all the varying parameters' values for all the runs, taking into account the number of samples, since the set of points generated by LDS and LHS is dependent on the number of samples.

As shown in the previous sections, with only seven varying parameters composing the variation space, it is possible to see some irregular patterns as well as clusters and empty spaces. In this case, the variation space is composed of more than ten thousand more values than the single FinFET case, a factor that will have a major impact on the speed mean convergence.

As each sample takes more than five hours to be completed, the need for time and IT reduction is crucial. As a consequence, is not feasible to run more simulations to cover

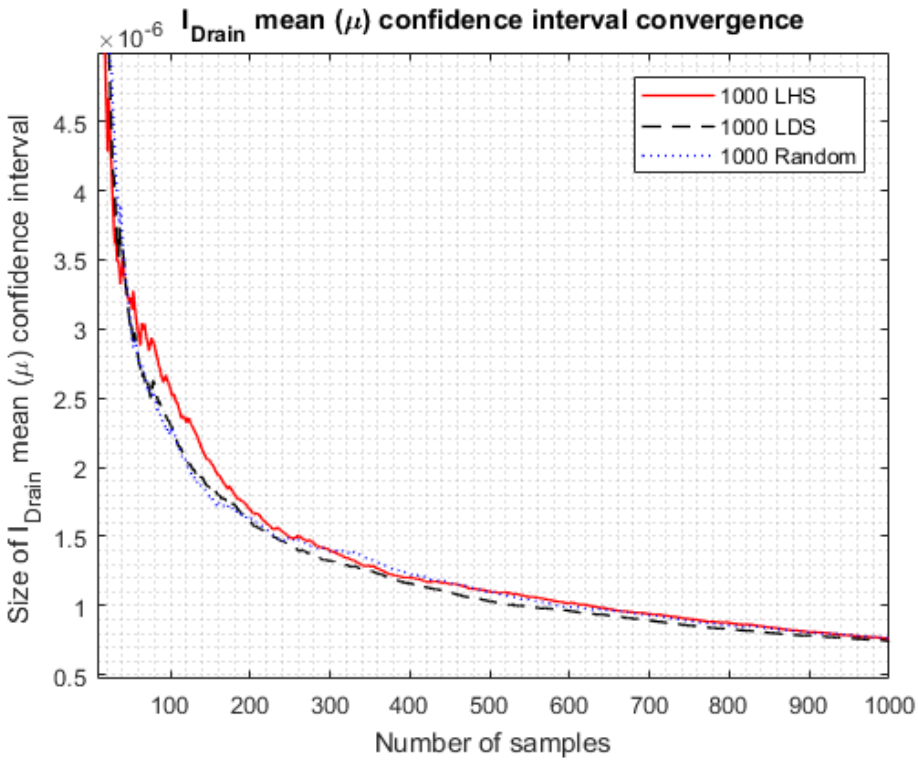


Figure 4.16: I_{Drain} mean confidence interval size convergence

this expected lack of performance of the Low Discrepancy Sampling method with high dimensional problems.

4.2.2 Mean error convergence for different spreading methods

The first step to evaluate the convergence of the mean was a generation of the golden references. The standard number of samples recommended is to run a Monte Carlo simulation with at least two thousand samples for three sigma designs. Considering the size of the circuit and the resources needed to run such a large simulation and the issues presented in the previous sections, the golden references were produced with two thousand and five hundred simulations using Latin Hypercube Sampling method. The reference to calculate the relative error is the last value of the mean since in both cases, after two thousand samples the mean value is well established. The mean values are shown in figure 4.17.
It was used a similar analysis flow to determine the performance of the methods. In this case, different seeds were also used to generate samples in order to have a fairer comparison.

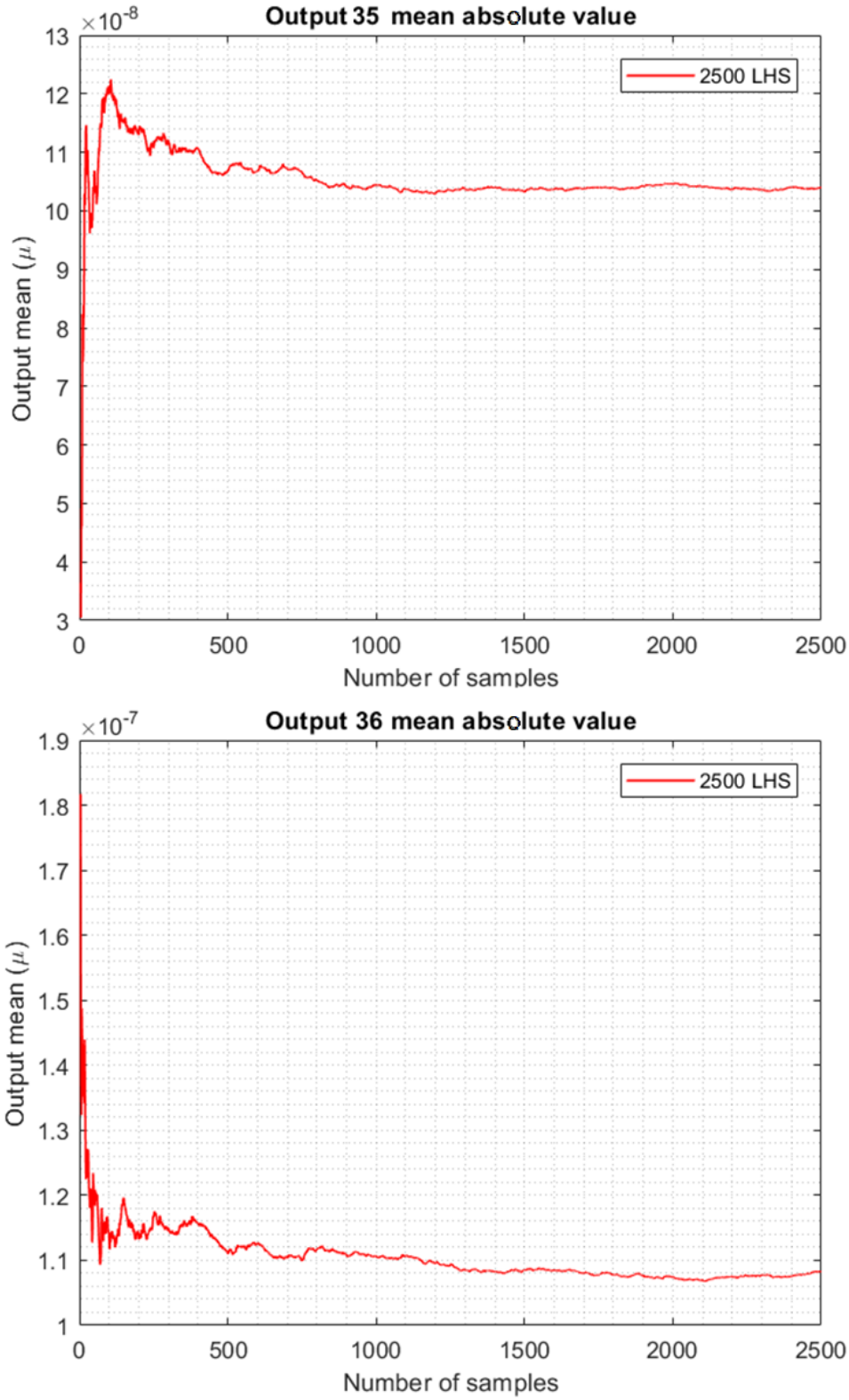


Figure 4.17: Golden references

In figure 4.18 is shown the relative error of both outputs. The relative error is substantially high, with values between 3% and 50%. The table below shows a comparison between all methods:

Mean relative error comparison table for "output 35"				
Sampling Method	Seed	Number of samples to converge above |10|%	Number of samples to converge above |5|%	Final error (%)
Random	12345	410	——	5.48
Random	98765	600	——	4.97
Low Discrepancy	——-	20	120	2.08
Latin Hypercube	12345	80	160	3.51
Latin Hypercube	98765	110	170	3.16

Mean relative error comparison table for "output 36"				
Sampling Method	Seed	Number of samples to converge above |10|%	Number of samples to converge above |5|%	Final error (%)
Random	12345	80	870	3.50
Random	98765	110	130	2.60
Low Discrepancy	——-	200	830	1.70
Latin Hypercube	12345	170	220	4.18
Latin Hypercube	98765	80	980	4.98

In both cases, with the Low Discrepancy Sampling, the error at the 1000th run is lower than the rest of the methods. However, looking at the speed of convergence of the output 36, it is possible to see that for a convergence lower than 10%, the LDS takes 17% more samples compared with the worst case (LHS with seed 12345). In one hand, comparing the number of samples needed to have a relative error lower than 5% with the best case of Random and LHS, it is possible to speed up to more than six times using Random sampling and 3.5 times using LHS. On the other hand, if the comparison is made using the worst case of Random and LHS, it is needed 4% and 18% more samples to achieve that accuracy.

The size of the confidence interval (with an inference error of 2.5%), for the LHS with seed 98765 is lower than the LDS for the output 35, as shown in figure 4.19, indicating that the estimation of the mean is better. For the "Output 36", the difference between LDS and LHS with seed 98765 is not significant.

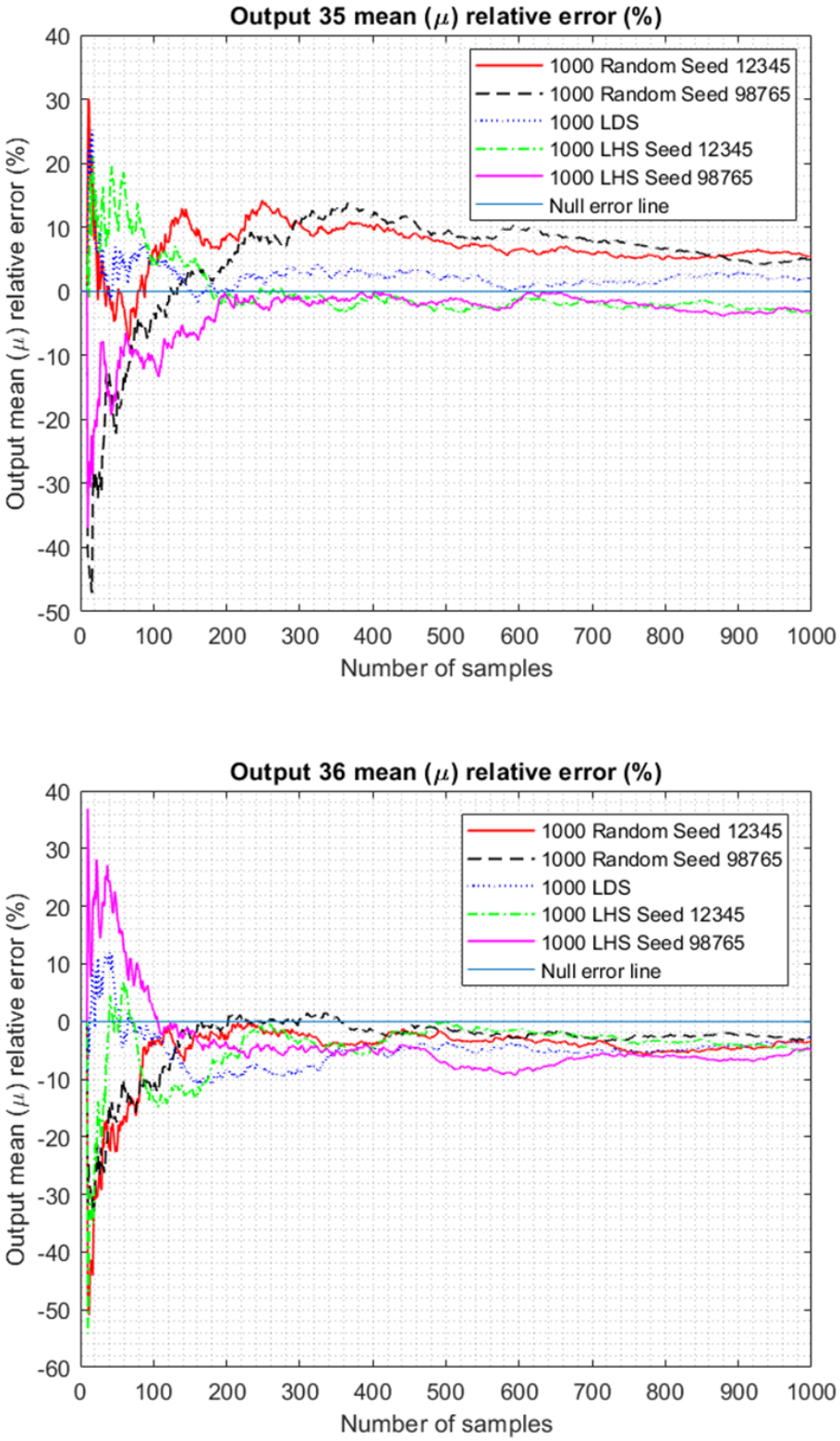


Figure 4.18: Output Comparison

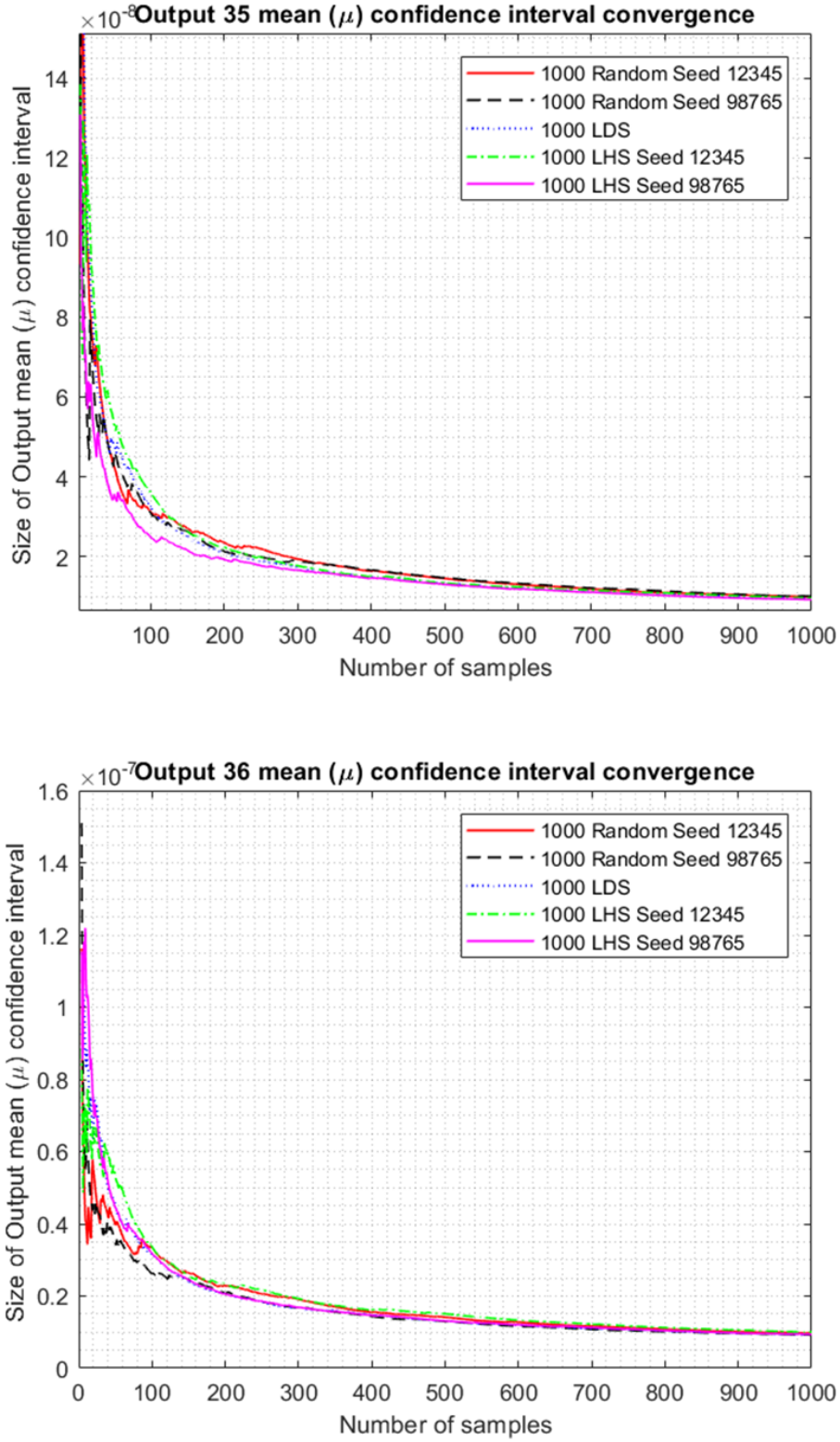


Figure 4.19: Output mean confidence interval size convergence

4.2.3 Low Discrepancy Sampling Performance

As shown in the previous section, the performance of the LDS method went down compared with the first two study cases. In some projections, the number of samples increased and the number of clusters also increased. As a consequence, the LDS is providing an awful representation of the variation space, as shown in figure 4.20.

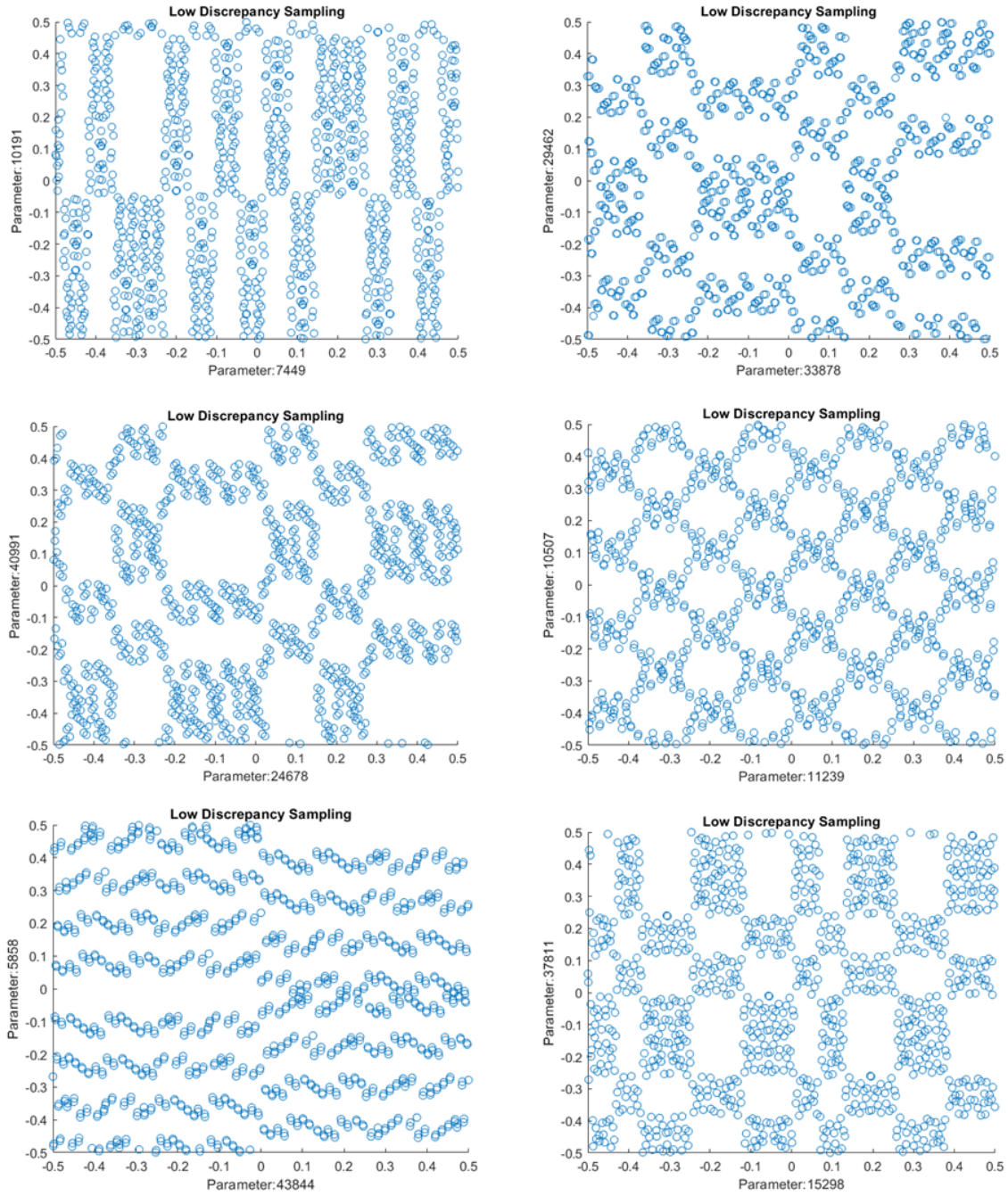


Figure 4.20: Low Discrepancy Sampling - Worst cases

By seeing the worst case scenario, shown in figure 4.21, it is possible to see a miserable spreading. If this method is used in verification, the worst case scenario[2] could be within this variation space holes. If that happens, the IC can fail and if applied to a critical system, could lead to a catastrophe.

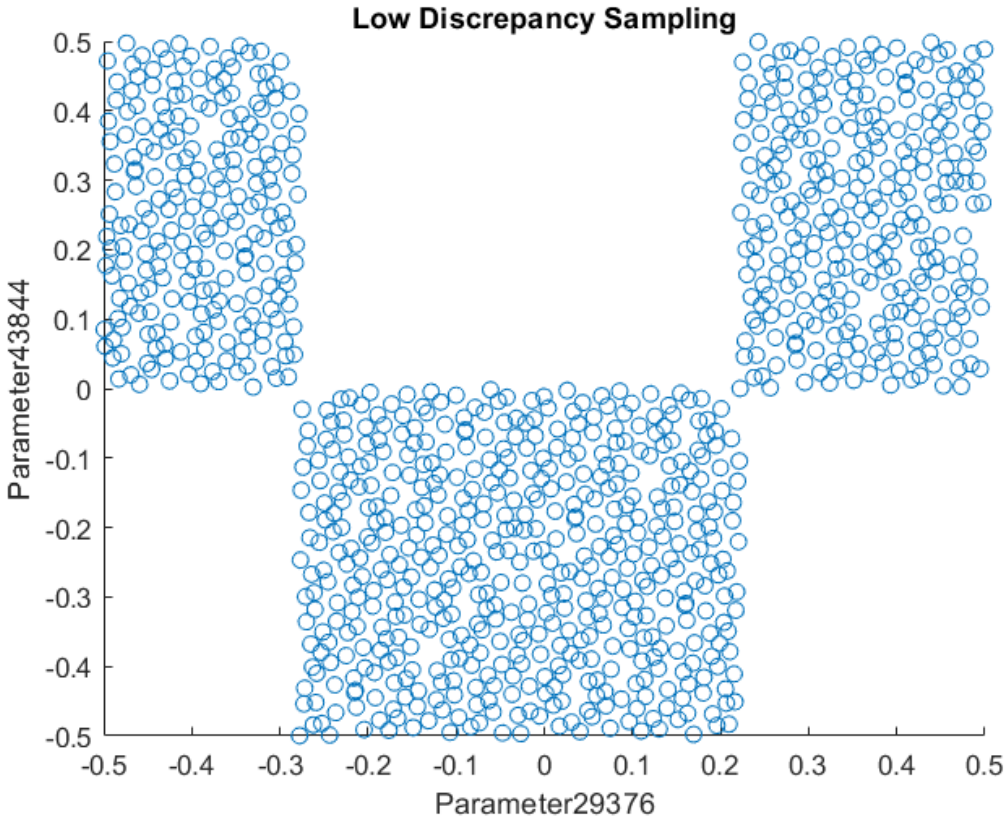


Figure 4.21: Low Discrepancy Sampling - Worst case

[2]The combination of all varying parameters that lead to a fail or a deviation of the circuit specifications.

Chapter 5

Efficient Monte Carlo simulation flow for large designs – Screening with RSM

5.1 Introduction

In this chapter, an improved methodology for Monte Carlo simulations on large and complex netlists like a complete receiver is described. In this chapter, it will be shown the importance of providing circuit insight to the MC Simulation before starting it in order to achieve results in a more efficient way.

As shown in the previous chapter, the increased complexity of circuits and devices leads to an increase in the number of process parameters that can vary within the circuit. An accurate variability analysis of a large circuit, like a complete receiver should not be simulated by simply turning on an MC simulation. To reach an accurate result, many simulations are required due to the increase of the geometric non-uniformity of points in a set (discrepancy) of the LDS method. An insight into the main contributors that affect the output must be known before performing an MC simulation.

Our proposed work will help the users to automatically select the most important instances to run a MC Simulation for a certain output or a group of them reducing substantially the number of MC runs required.

In the following sections of this chapter, the methodology is explained in detail as well as the improvement results, in comparison with the original methods.

5.2 Methodology proposed

Within this methodology we first propose to run a sensitivity analysis in order to know the most important contributors (instances) for each output and split the list of instances by groups, as shown in figure 5.1.
As the traditional sensitivity analysis required two simulations per parameter (One Factor At Time – OFAT), it is impractical to use it with RF designs. By applying another method named as "Sensitivity Accuracy", available on Cadence Virtuoso Variation Option [43], it is possible to run the required sensitivity analysis using a low number of MC runs. This method relies on a Response Surface Model (RSM) to evaluate the Sensitivity of certain output regarding a specific mismatch parameter.

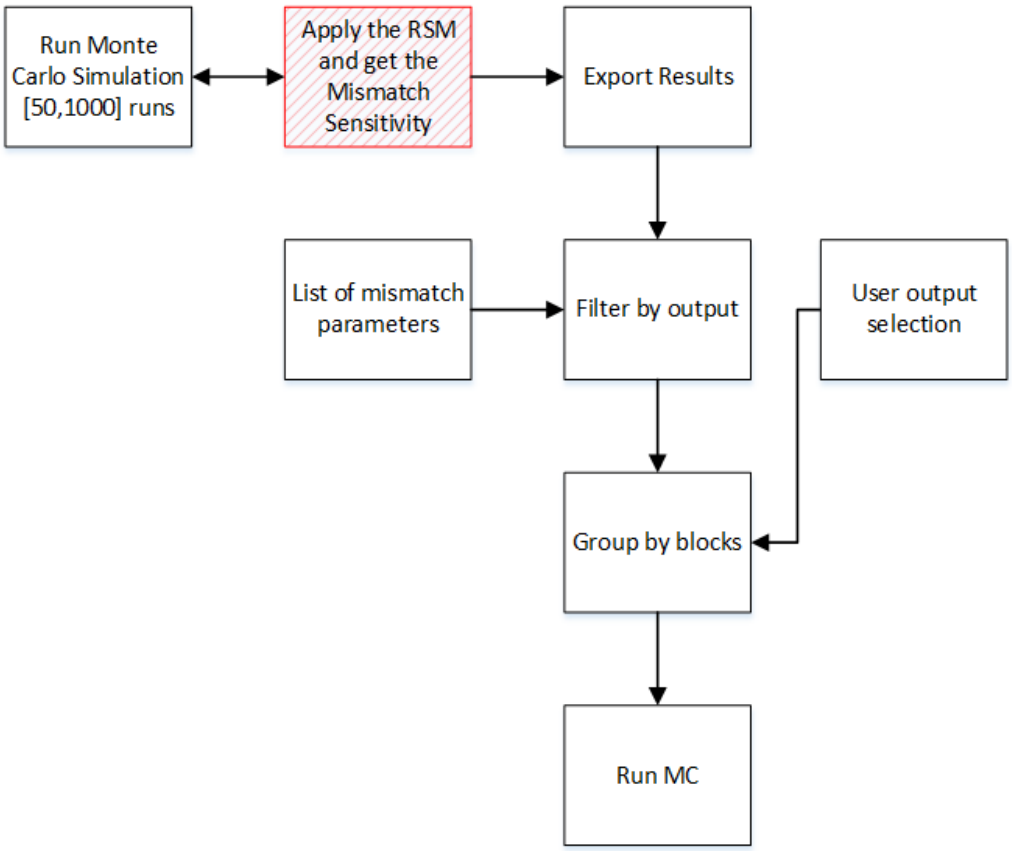


Figure 5.1: Screening technique

The sensitivity analysis is made by running MC simulations until an acceptable level of accuracy (R^2 – Squared Metric) to build a (RSM) is reached, as shown in figure 5.2.
R^2 value shows how much of the total variance can be explained by the computed model. In other words, the R^2 is the ratio of the variation in K that is explained by the systematic

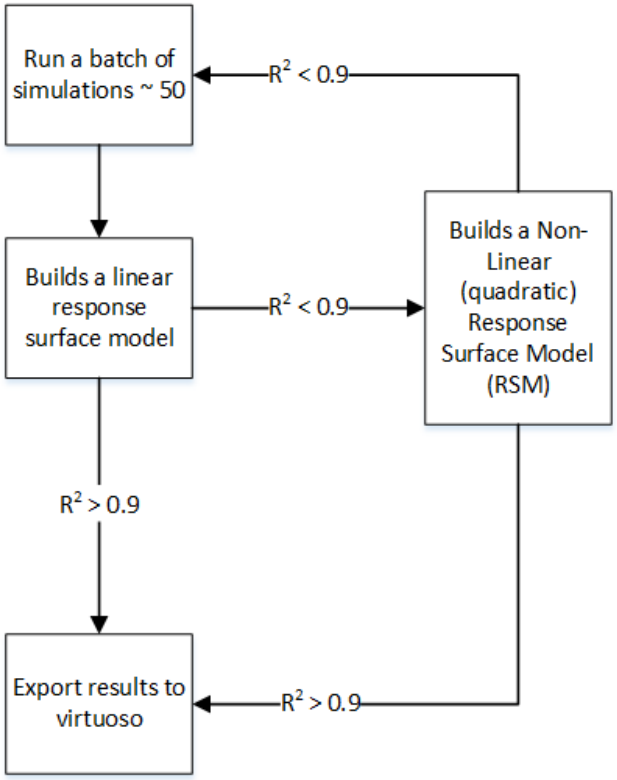


Figure 5.2: Sensitivity Analysis

proportion of the model (Z) and it can be expressed by:

$$R^2 = \frac{Var(K)}{Var(Z)}, 0 \leq R^2 \leq 1 \tag{5.1}$$

Where:

- $R^2 = 0$: The model does not explain anything.
- $R^2 = 1$: The model explains everything.

After importing the mismatch contribution results to a python dataframe, the file that contains all the instances of the circuit is also imported in order to identify parallel arrays. With both dataframes available, it is now possible to generate a list of instances with contribution greater than 0. It is now possible to select all desired outputs and compile the list of instances. In figure 5.3, the flow of the tool is shown.
The list is now imported to the simulator and a MC simulation is started, by applying variation only on those instances[1].

[1]This technique is named as Screening.

A faster convergence of the mean and standard deviation values and the detection of the worst case with a reduced number of MC runs can be obtained by applying these Screening techniques.

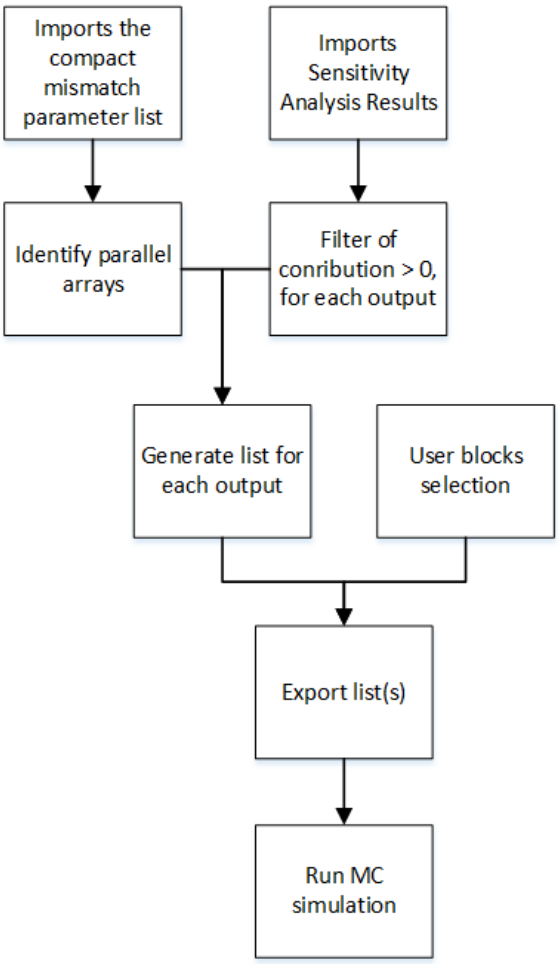


Figure 5.3: Algorithm Script

5.3 Algorithm Testbench

In order to test this method, it was used a similar testcase compared with the one shown in section 4.2. Due to a limit imposed by the simulator and to the hierarchical complexity of the circuit, it is only possible to select a certain number of instances (the limit comes from the string size of instances list), unabling us to test the same output refereed in section 4.2 (described as "output 36" and "output 35").
From the same circuit a set of 24 outputs was chosen, representing a list of instances with a total of 225 parameters (1064 parallel arrays included). An output from the set of 24

outputs was chosen based on the greatest mean and standard deviation. This output will be further described as "Output 1". The list of mismatch contributors for that specific "output 1" was also generated and it was determined that it was composed of 166 parameters (664 parallel arrays included).
A Monte Carlo simulation with 2500 runs sampled by the Latin Hypercube Sampling was used as a reference.
A comparison between the standard MC flow and the improved one was based in the following parameters:

- Output mean (μ) relative error
- Output standard deviation (σ) relative error
- Convergence of the mean confidence interval
- Convergence of the variance (S^2) confidence interval
- Output measure value distribution
- Discrepancy of 2D and 3D projections

5.4 Results

5.4.1 Discrepancy on 2D projections

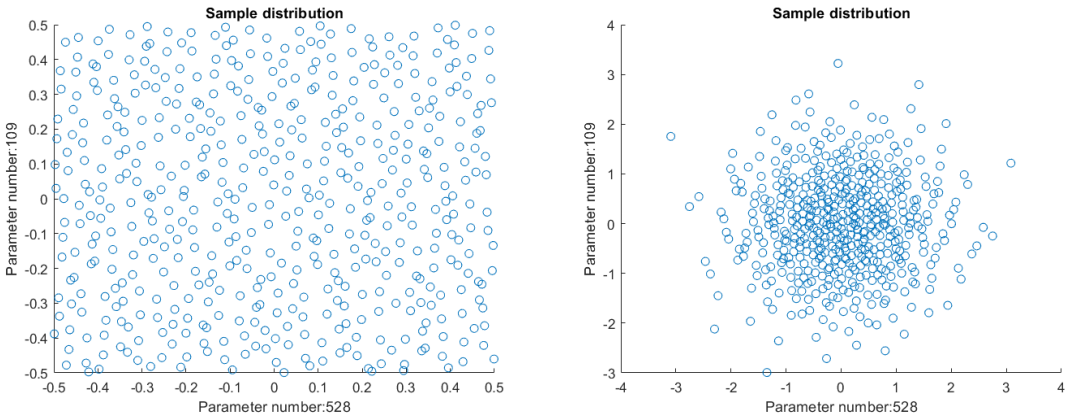
In figure 5.4, a well spread 2 dimensional variation space (sampled by LDS) is shown. The sub-figure 5.4a shows the variation space after being transformed to a uniform distribution and sub-figure 5.4b shows the original normal distribution (given by the simulator). The same can not be stated in figure 5.5 where a not so good spreading example is shown. The 3 clustered zones shown in sub-figure 5.5a exposes the ineffectiveness of the Low Discrepancy Sampling method even with a low number of instances. This condition leads to a slowness of the process and also to an redundant optimistic/pessimistic yield predictability. The worst case condition will be faulty in case these 2 two parameters have an high contribution to the sensitivity of a certain output[2]. Sub-figures 5.6a and 5.6b shows that even if we raise the number of simulations from 1000 to 5000 runs (an increase of 5x), the problem might not be solved. In some cases, like the one shown in figure 5.6c, the cluster issue is solved when the same increase of runs is made. A possible conclusion of these results is these even with a small number of instances selected (166), the LDS does not have a good performance and can compromise the Yield Analysis of the circuit.

5.4.2 Mean and std convergence – relative error and confidence interval

Since the LDS and the Random Sampling[3] show lack of performance that cannot be solved even if we rise the number of runs, the following analysis will only cover the results when the algorithm is applied to the LHS.
In figures 5.7 and 5.8 the convergence of the relative error and standard deviation variation is presented. It is possible to see that after 150 runs both simulations are converging to the true value of the mean (0% relative error). In the case of the standard deviation, after 150 runs, the relative standard deviation error starts to increase up to 8%, finishing with an error 4% greater when compared with the Latin Hypercube filtered simulation. The results of the comparison between both simulation are shown in the table below.

[2]The good spreading examples are typically the first parameters distributions picked to sample and the order of sampling is based on the order of the netlist. This means that the first instances of the netlist will have a better spreading when compared with the last. If the output is sensible to the last netlist instance, the chances of finding the best/worst case are low.

[3]In chapter 4, the Random sampling appears as the slowest method to estimate the standard deviation and mean.



(a) 2D uniform distribution (b) Normal 2D Distribution

Figure 5.4: Good spreading example

By analysing the variance and mean confidence interval convergence shown in figures 5.9 and 5.10, it is possible to conclude that the convergence is faster when only a subset is simulated (when the filter is applied).

In figure A.1 shown in appendix, the output measurement values are shown. By analysing the scatter, it is possible to see that the worst case point (for this specific output, it is the largest measured value) located near to the point 480, was sampled by the LHS with the subset selected.

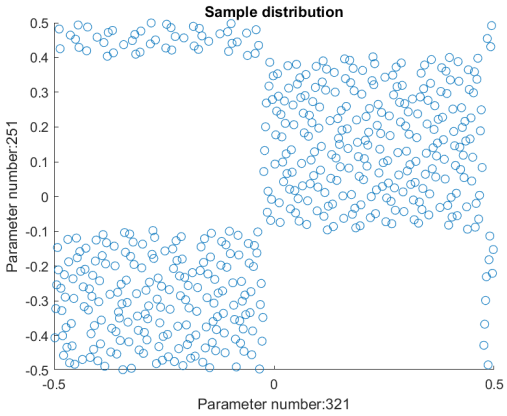
The table below shows the reduction of the mean and standard variation relative error, with special focus on the standard variation error decrease at the run 500. The error peak of the LHS filtered is also lower, 2.5 times less than the normal LHS. The proposed method is the Latin Hypercube Filtered.

Mean relative error comparison table		
Sampling Method	Number of samples to converge above |0.5|%	Relative error at run 500
Latin Hypercube Filtered	33	-0.0001077
Latin Hypercube	52	-0.00002158

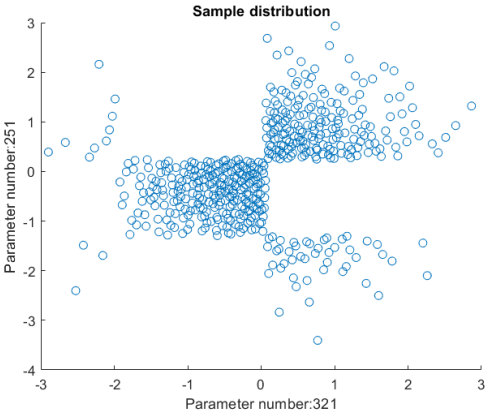
Table 5.1: Mean relative error comparison table

Standard deviation relative error comparison table			
Sampling Method	Number of samples to converge above |5|%	Relative error at run 500	Max error after run 50
Latin Hypercube Filtered	79	0.8%	3.2%
Latin Hypercube	64	4.8	8%

Table 5.2: Standard deviation relative error comparison table

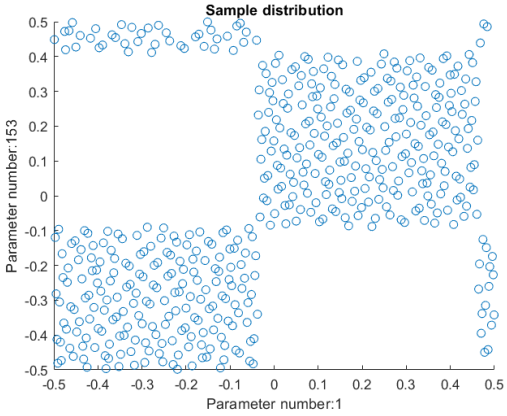


(a) 2D uniform distribution

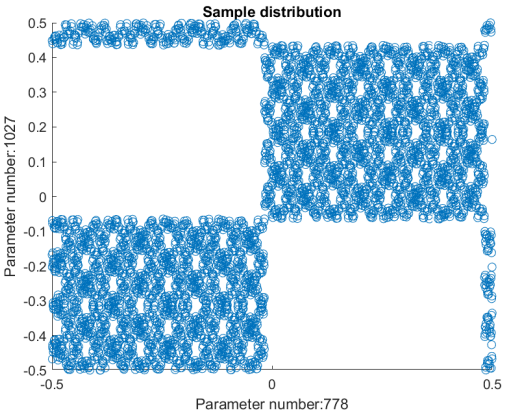


(b) Normal 2D Distribution

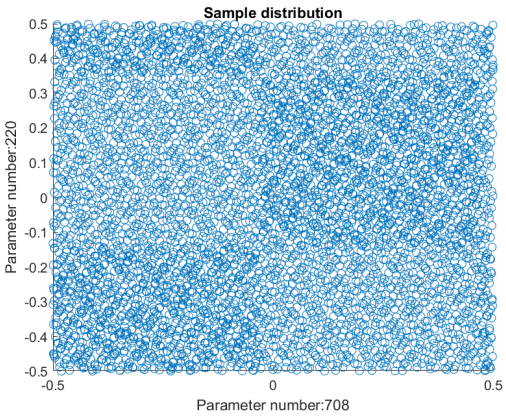
Figure 5.5: Bad spreading example – uniform and normal distribution



(a) 2D uniform distribution



(b) 2D uniform distribution – case where the increase has no improvement



(c) 2D uniform distribution – case where the increase covers the empty spaces

Figure 5.6: Bad spreading with different number runs

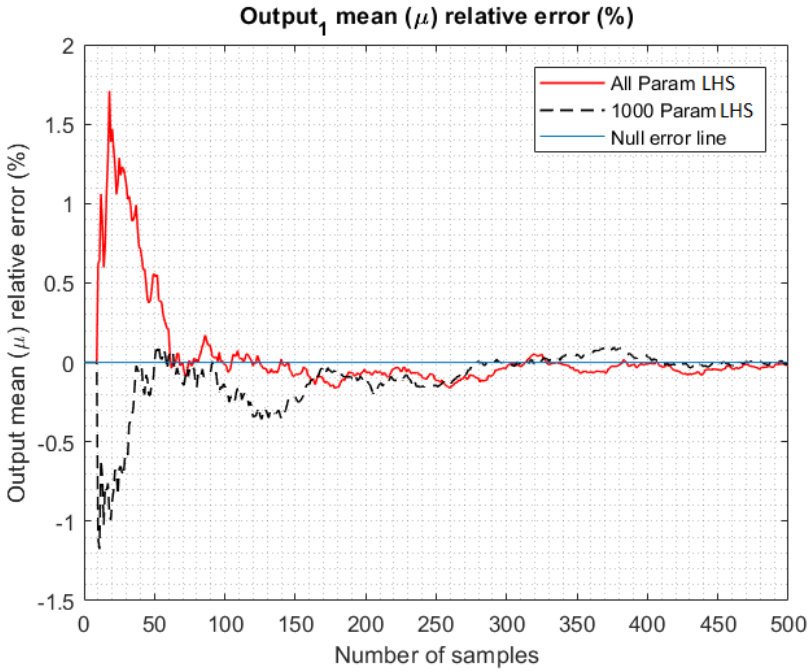


Figure 5.7: Output mean relative error convergence

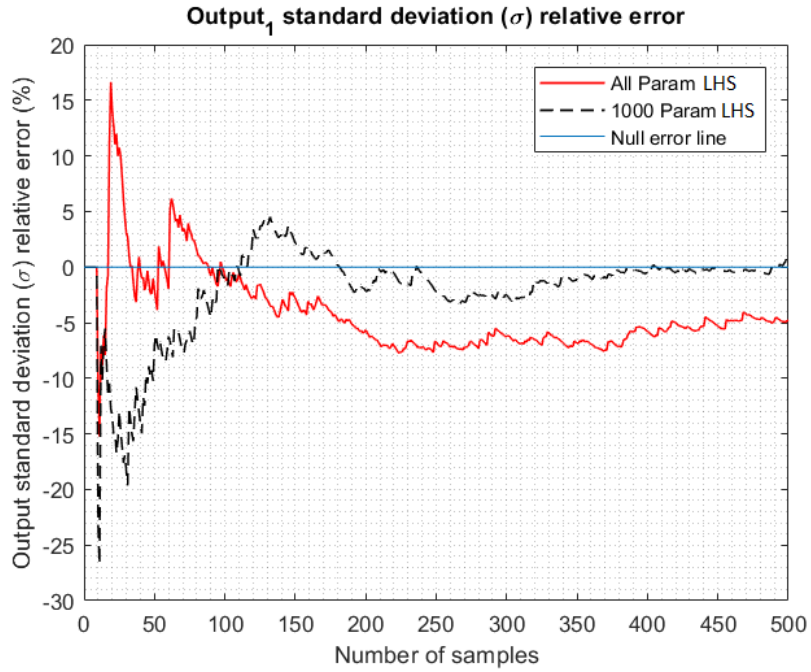


Figure 5.8: Standard deviation (σ) relative error convergence

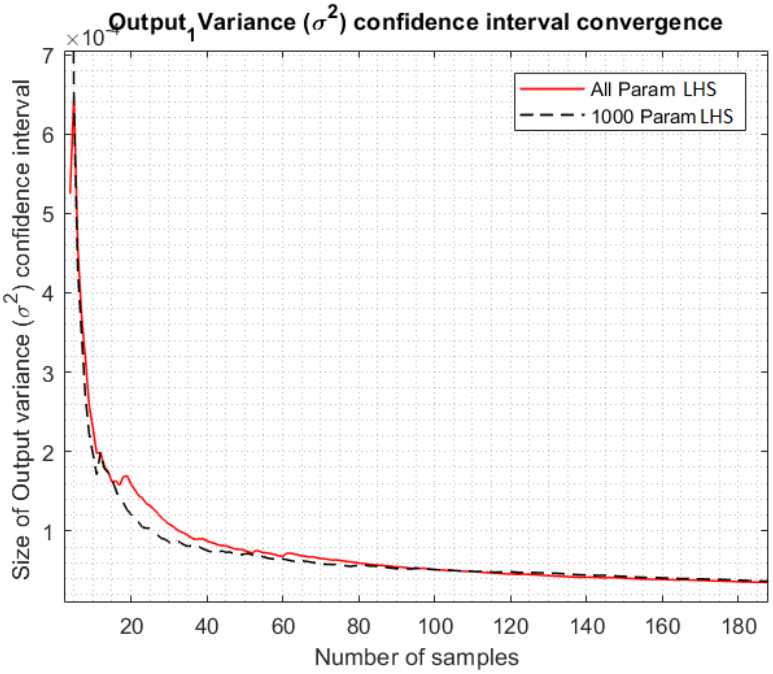


Figure 5.9: Variance confidence interval convergence

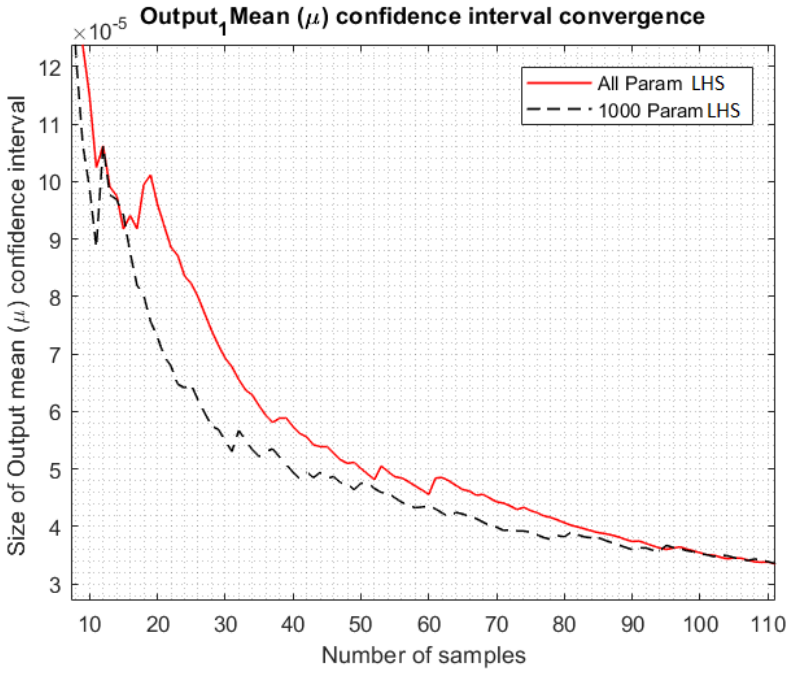


Figure 5.10: Mean confidence interval convergence

Chapter 6

Conclusions and Future Work

6.1 Conclusion

At the beginning of the verification process, the adoption of the Monte Carlo method using pseudo-random generators was appropriated for small circuits that were developed at that time. Small circuits are translated into variation space with a small number of dimensions, and it was why the random generator had good performance. With the rise of complexity and variation space dimensionality, the Latin Hypercube method and Low Discrepancy generator emerge as an update of the Random generator.
In this thesis, it is shown that for small dimensions, the usage of Low Discrepancy generators can speed up the Monte Carlo simulations more than three times. It was also shown that for large dimensions, the Low Discrepancy generator could not guarantee better performance than the Latin Hypercube method. As a consequence, it is possible to state that the feasibility of the Low Discrepancy generator is questionable for a problem with a large number of dimensions. Since the complexity and size of real-world circuits are increasing, and consequently the number of dimensions of their variation space, this method will cease to be feasible in the near future. The superiority of the Latin Hypercube compared with the Random generator was also shown.
The sampling method exposed in section 3.4 may handle these problems with better efficiency, but it was never tested for very high dimensionality problems. This thesis presented in chapter 5, a new optimized method that handles the high dimensionally problems. This method consists in using a pre-MC simulation and a RSM to determine and filter the most important contributor for the measurement in order to reduce the number of runs required. It has been demonstrated that the proposed method achieves more accuracy with fewer resources (see table 5.1 and 5.2).

6.2 Future work

There is a few more possible enhancements to do on the MC simulation flow as possible future work:

- Implementation of the optimized Latin Hypercube exposed in chapter 3, section 3.4.
- Benchmark of the optimized Latin Hypercube method with all methods reviewed in chapter 3.
- Use of the sensitivity analysis results to choose the best polynomials to the most important instances, during the generation of numbers with the Sobol's algorithm. This idea is pointed in [39], but it is purely theoretical and was never tested.

Appendix A

Appendix

A.1 Halton Matlab Code

```
% Inputs: nSamples - the length of the vector to generate
%         base - the base of the sequence
function GeneratedSequence = HaltonSequence(nSamples,base)
GeneratedSequence = zeros(nSamples,1); % Preallocate the output
for n = 1:nSamples
    GeneratedSequence(n) = HaltonNumber(n,base);
end

% Generate n samples with Halton's sequence
function GeneratedNumber = HaltonNumber(nSamples,base)
n0 = nSamples;
GeneratedNumber = 0;
f = 1/base;
while (n0>0)
    n1 = floor(n0/base);
    r = n0-n1*base;
    GeneratedNumber = GeneratedNumber + f*r;
    f = f/base;
    n0 = n1;
end
```

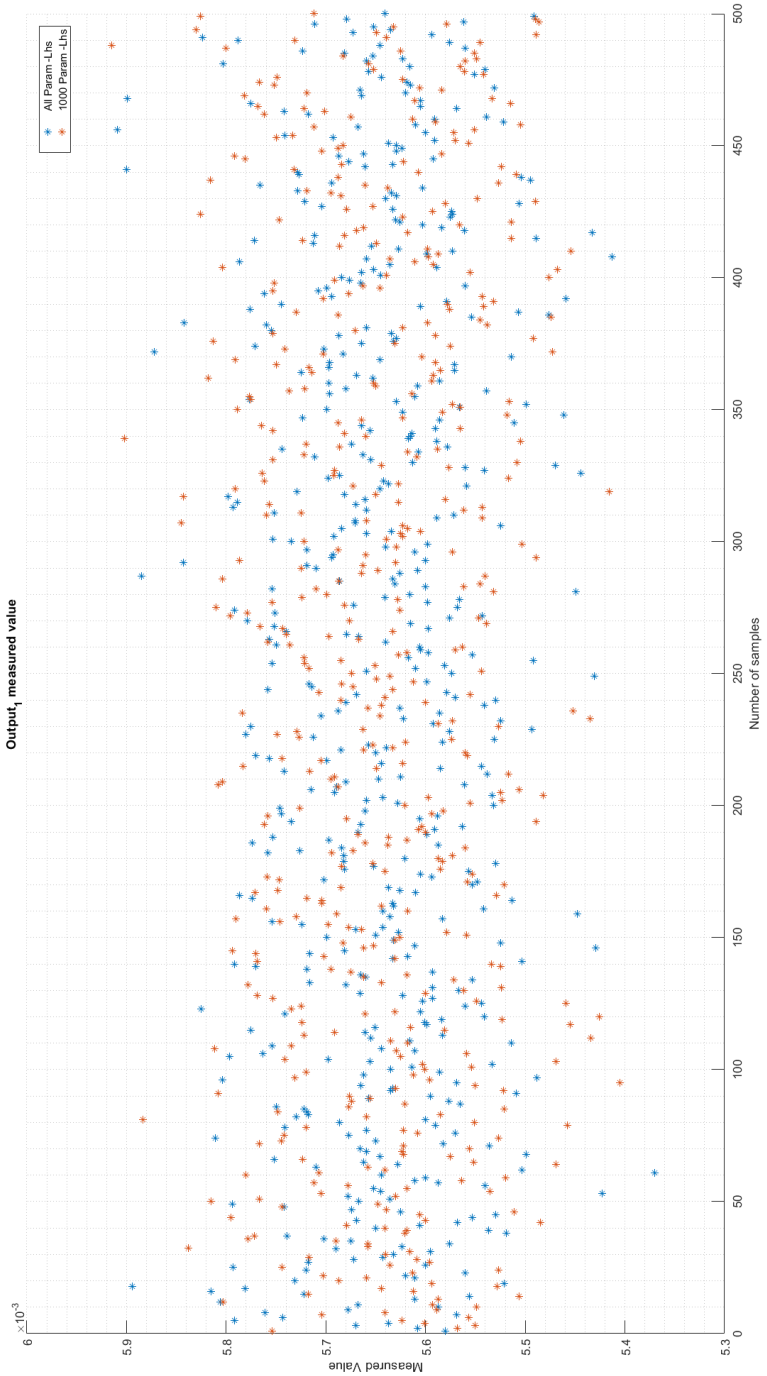


Figure A.1: Output measurement distribution

References

[1] F. Baumann, K.K. Bourdelle, T. Boone, J. Garno, A. Ghetti, M. Green, H. Gossmann, Y. Kim, R. Kleiman, A. Kornblit, F. Klemens, S. Moccio, D. Muller, J. Rosamilia, P. Silverman, T. Sorsch, W. Timp, D. Tennant, R. Tung, B. Weir G. Timp, J. Bude. The relentless march of the MOSFET gate oxide thickness to zero. *Microelectronics Reliability*, 40(4-5):557–562, 2000. doi:10.1016/S0026-2714(99)00257-7.

[2] Semiconductor NSM. Silicon basic parameters table at 300k – lattice constant, 2014.

[3] Alan Seabaugh. The Tunneling Transistor. *IEEE Spectrum*, 2013.

[4] Lihui Wang. Quantum mechanical effects on MOSFET scaling limit. 2006.

[5] Y. Lai. Accelerating Monte Carlo Analysis at Advanced Nodes. *International Meeting on Infomation Display*, pages 1069–1072, 2009.

[6] Colin C. Mcandrew, Ik-sung Lim, Brandt Braswell, and Freescale Garrity, Doug Semiconductor. Corner Models: Inaccurate at Best, and it Only Gets Worst... pages 13–16, 2013.

[7] Jim Dodrill. The Frontiers of Robust Circuit Design in Sub-28nm Process Technologies. Technical report, ARM, 2015.

[8] Harald Welte. Anatomy of contemporary GSM cellphone hardware. page 10, 2010.

[9] Nikhil M Kriplani. Transistor Modeling using Advanced Circuit Simulator Technology. pages 1–78, 2002.

[10] Xuemei Jane Xi, Mohan Dunga, Jin He, Weidong Liu, M Kanyu, Xiaodong Jin, Jeff J Ou, Mansun Chan, and Ali M Niknejad. User's Manual Developers. 2003.

[11] V. Sriramkumar, Navid Paydavosi, Darsen Lu, Chung-hsun Lin, Mohan Dunga, Shijing Yao, Tanvir Morshed, Ali Niknejad, Chenming Hu, Ali Niknejad, and Chenming Hu. Multi-Gate MOSFET Compact Model Technical Manual. 2012.

[12] Trent McConaghy. Analog behavior in custom IC variation-aware design. *IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, ICCAD*, pages 146–148, 2013.

[13] Trent McConaghy, Kristopher Breen, Jeffrey Dyck, and Amit Gupta. Variation-Aware Design of Custom Integrated Circuits: A Hands-on Field Guide. 2013. doi:10.1007/978-1-4614-2269-3.

[14] Bruno Cardoso. AIDA-PEx: Parasitic Extraction on Layout-Aware Analog Integrated Circuit Sizing Electrical and Computing Engineering Supervisors. 2015.

[15] Hau Yung Chen, Ming Juan, Hsin Hao Chen, and Arvin Guan. Practical electrical parameter aware methodology for analog designers with emphasis on LDE aware for devices. *Technical Papers of 2014 International Symposium on VLSI Design, Automation and Test, VLSI-DAT 2014*, pages 6–9, 2014. doi:10.1109/VLSI-DAT.2014.6834922.

[16] Prasanjeet Das and Sandeep K. Gupta. Efficient post-silicon validation via segmentation of process variation envelope – Global vs local variations. *Proceedings – International Symposium on Quality Electronic Design*, pages 115–122, 2014. doi:10.1109/ISQED.2014.6783314.

[17] ITRS. Technical report, International Technology Roadmap for Semiconductors, 2011.

[18] Brandt Braswell. Statistical simulations to replace corner simulations, 2016.

[19] Dale E. Hocevar, Dale E. Hocevar, Michael R. Lightner, Michael R. Lightner, and Timothy N. Trick. A Study of Variance Reduction Techniques for Estimating Circuit Yields. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2(3):180–192, 1983. doi:10.1109/TCAD.1983.1270035.

[20] Kishore Singhal and J. F. Pinel. Statistical Design Centering and Tolerancing Using Parametric Sampling. *IEEE Transactions on Circuits and Systems*, 28(7):692–702, 1981. doi:10.1109/TCS.1981.1085029.

[21] Norman Jay Elias. Acceptance Sampling: An efficient, accurate method for estimating and optimizing parametric yield. (3):1–4, 1993.

[22] Cadence Design Systems. Virtuoso Variation Option - Advanced statistical exploration of your design. Technical report, 2016.

[23] Stephan Weber, Tiago Ressurreição, and Cândido Duarte. Yield Prediction With a New Generalized Process Capability Index Applicable to Non-Normal Data. 35(6):931–942, 2016.

[24] Diming Ma, Guoyong Shi, and Alex Lee. A Design Platform for Analog Device Size Sensitivity Analysis and Visualization. *2010 IEEE Asia Pacific Conference on Circuits and Systems*, pages 48–51, 2010. doi:10.1109/APCCAS.2010.5774855.

[25] Prabhat Mishra, Ronny Morad, Avi Ziv, and Sandip Ray. Post-Silicon Validation in the SoC Era: A Tutorial Introduction. *IEEE Design and Test*, 34(3):68–92, 2017. doi:10.1109/MDAT.2017.2691348.

[26] A. Adir, S. Copty, S. Landa, A. Nahir, G. Shurek, A. Ziv, C. Meissner, and J. Schumann. A unified methodology for pre-silicon verification and post-silicon validation. *2011 Design, Automation & Test in Europe*, pages 1–6, 2011. doi:10.1109/DATE.2011.5763252.

[27] Ken Kundert and Henry Chang. Verification of complex analog integrated circuits. *Proceedings of the Custom Integrated Circuits Conference*, pages 177–184, 2006. doi:10.1109/CICC.2006.320883.

[28] Brian Hayes. IACS Seminar: Orderly Randomness: Quasirandom Numbers and Quasi-Monte Carlo. 2015.

[29] Amith Singhee and Rob A. Rutenbar. Why Quasi-Monte Carlo is Better than Monte Carlo or Latin Hypercube Sampling for Statistical Circuit Analysis. 29(11):1763–1776, 2010.

[30] Hiwa Mahmoudi and Horst Zimmermann. A new sampling technique for Monte Carlo-based statistical circuit analysis. *Proceedings of the 2017 Design, Automation and Test in Europe, DATE 2017*, pages 1277–1280, 2017. doi:10.23919/DATE.2017.7927188.

[31] Syoiti Ninomiya and Shu Tezuka. Toward real-time pricing of complex financial derivatives. *Applied Mathematical Finance*, 3(1):1–20, 1996. doi:10.1080/13504869600000001.

[32] Fred J. Hickernell. A generalized discrepancy and quadrature error bound. *Mathematics of computation*, 67(221):299–322, 1998.

[33] Michael Carter. The Sobol sequence – Monte Carlo Methods in Financial Engineering, 2007.

[34] Stephan Weber and Cândido Duarte. *Circuit Design – Anticipate, Analyze, Exploit Variations: Statistical Methods and Optimization*. Stylus Publishing, LLC, 2017.

[35] Cadence Design Systems. Accelerating Monte Carlo Analysis at Advanced Nodes, 2018.

[36] Shupeng Sun and Xin Li. Fast statistical analysis of rare circuit failure events via subset simulation in high-dimensional variation space. *IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers, ICCAD*, 2015-January(January):324–331, 2015. doi:10.1109/ICCAD.2014.7001370.

[37] Anton O. Prokofiev, Andrey V. Chirkin, and Vladislav A. Bukharov. Methodology for Quality Evaluation of PRNG , Investigating Distribution in a Multidimensional Space. pages 355–357, 2018.

[38] W. J. Cody. Rational Chebyshev Approximations Error Function. 1969.

[39] Ilya M. Sobol, Danil Asotsky, Alexander Kreinin, and Sergei Kucherenko. Construction and Comparison of High-Dimensional Sobol's Generators. pages 64–79.

[40] Hongshan Lu, Chongjian Wu, and Jian Xu. 2013 International Conference on Mechanical and Automation Engineering A Method of Quasi Monte Carlo for Computation of Ship Availability. (3), 2013. doi:10.1109/MAEE.2013.11.

[41] Goddard Consulting. Halton Sequence Matlab Function, 2018.

[42] Sheldon M. Ross. *Simulation*. Elsevier, fifth edit edition, 2013.

[43] Cadence Design Systems. *Virtuoso Variation Option User Guide*. Number March. 2018.