

FEDERAL UNIVERSITY OF PAMPA

Felipe Antunes Quirino

**Automatic Design of Analog Integrated
Circuits Under Process Parameter
Variability**

Alegrete
2020

Felipe Antunes Quirino

**Automatic Design of Analog Integrated Circuits
Under Process Parameter Variability**

Final Paper submitted to the Undergraduate Program in Computer Science of Federal University of Pampa in partial fulfillment of the requirements for the Bachelor's degree in Computer Science.

Supervisor: Prof. Dr. Alessandro Gonçalves Girardi

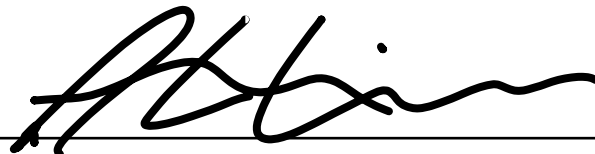
Alegrete
2020

Felipe Antunes Quirino

Automatic Design of Analog Integrated Circuits Under Process Parameter Variability

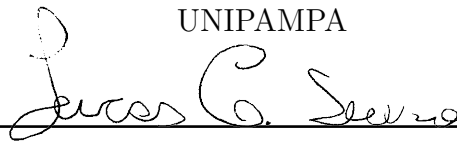
Final Paper submitted to the Undergraduate Program in Computer Science of Federal University of Pampa in partial fulfillment of the requirements for the Bachelor's degree in Computer Science.

Final Paper presented and approved in December 8th 2020
Examination board:



Prof. Dr. Alessandro Gonçalves Girardi

Supervisor
UNIPAMPA



Prof. Dr. Lucas Severo

UNIPAMPA



Prof. Dr. Marcelo Caggiani Luizelli

UNIPAMPA

This work is dedicated to automation of boring stuff.

ACKNOWLEDGEMENTS

I would first express my very profound gratitude to my parents for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching. This accomplishment would not have been possible without them.

I would also like to thank my thesis advisor Dr. Alessandro Girardi, professor at Federal university of Pampa. The door to Prof. Girardi office was always open whenever I ran into a trouble spot or had a question about my research or writing. He steered me in the right direction whenever necessary.

I would also like to thank my research group colleagues: Luiz Antonio Junior for always being available to help me make the tools work and Marcelo Romanssini for always doing his work with excellence in the projects in which I worked with him.

“A landed proprietor maintains that the use of machinery in agricultural operations, as practised in England, is an excellent institution, since an engine does the work of many men. You give him to understand that it will not be very long before carriages are also worked by steam, and that the value of his large stud will be greatly depreciated; and you will see what he will say.”

(Arthur Schopenhauer)

ABSTRACT

Analog integrated circuits have a high range of applications, from interface circuits to signal processing. These systems need to be carefully designed in order to achieve a suitable trade-off between performance and power consumption. Traditional method of designing an analog circuit is based on trial-and-error. The designer uses his own experience for testing and modifying circuit parameters with the aid of an electrical simulator, until achieving the desired solution. However, the process of sizing the circuit requires several hours of design. In addition, a SPICE simulation can take a long time. The designer needs to wait for the end of the simulation, verify the results, and from there to do another simulation. An alternative for design automation is to abstract the circuit as an optimization problem. Using an optimization algorithm it is possible to explore the design space in the search for an optimum solution. This work demonstrates an optimization analysis performed with a low-voltage bulk-driven operational transconductance amplifier. Previous works demonstrated the modeling of this circuit as an optimization problem, but only for nominal values. However, a nominal analysis presents the performance disregarding process parameter variability that affect the performance. In the present work, we propose a design automation tool of analog integrated circuits using yield analysis, i.e, estimating performance with Monte Carlo electrical in order to evaluate the impact of process variability on circuit performance. In general, optimization algorithms present random variables (i.e., even with the same parameters, different seeds for the random number generator function may converge to different results). In order to further understanding this behavior, this work analyses the behavior of many executions of the algorithm. The adjustable parameter on the present algorithm (Cuckoo Search) is the number of nests. We performed the simulation 30 times for the same number of nests, varying the number of nests from 10 to 490 with a total of 1470 executions. As result, this work demonstrates a statistic analysis of all designs intending to find the best parameters for the tool. After getting the parameters, we analysed the circuit behavior for the worst, median and best cases, with the fixed parameter. The tool is able to design analog integrated circuits automatically, however without a guarantee of a feasible solution. For instance, in the worst case example, the algorithm converges to an unfeasible solution for practical terms. However, this could be mitigated with a large member of iterations or with more than one execution. In the best case, the amplifier designed with the optimization tool presents improvements in terms of P_{cons} (8.86nW of mean \pm 0.22 of standard deviation in comparison to 18nW designed manually), PM ($59.99^\circ \pm 10.15$ in comparison to 52.50°) and GBW ($3.81\text{kHz} \pm 0.65$ in comparison to 1.88kHz).

Key-words: Analog IC design. Optimization algorithm. CAD tool.

RESUMO

Os circuitos analógicos integrados possuem uma ampla gama de aplicações e necessitam ser projetados de modo a atender a requisitos conflitantes de desempenho e consumo de energia. O método tradicional de projeto de um circuito analógico é baseado em tentativa e erro. O projetista utiliza sua própria experiência para testar e modificar os parâmetros até encontrar uma solução satisfatória. A análise do desempenho do circuito é feita com o auxílio de simulação elétrica SPICE. Todavia, este processo exige muito tempo de projeto e a simulação SPICE pode demorar um longo tempo. O projetista necessita esperar o fim da simulação para analisar os resultados e, a partir disso, alterar os parâmetros do circuito e fazer outra simulação. Uma alternativa para a automação do projeto é abstrair a tarefa de dimensionamento dos circuitos como problema de otimização. Utilizam-se algoritmos de otimização para explorar o espaço de projeto em busca de uma solução otimizada. O presente trabalho propõe o uso de otimização para o projeto de síntese analógica, com base em um amplificador operacional de transcondutância de ultra baixa tensão alimentado pelo substrato. Trabalhos anteriores já foram realizados com este circuito, porém realizando apenas simulações nominais no circuito. Contudo, somente com análise nominal não é possível determinar o comportamento do circuito sob a influência de variações nos parâmetros do processo de fabricação. No presente trabalho, propomos uma ferramenta para automação do projeto do circuito utilizando a análise de rendimento, isto é, utilizando simulação Monte Carlo para estimar o rendimento do circuito após a fabricação. Além disso, os algoritmos utilizados para a otimização de circuitos possuem variáveis aleatórias (i.e., diferentes execuções podem divergir os resultados conforme a semente da função de geração de números aleatórios, mesmo tendo os mesmos parâmetros). Para prever este comportamento, o objeto de estudo analisa o comportamento de diversas execuções do algoritmo Cuckoo Search. Como resultado, demonstra-se uma análise estatística dos projetos. A partir desta análise, é possível escolher os melhores parâmetros para o algoritmo. Dado os parâmetros, analisou-se o comportamento dos circuitos para pior, mediano e melhor caso das execuções. A ferramenta consegue projetar circuitos analógicos integrados de forma automática, porém não há garantias que ela sempre converge em um resultado viável. Por exemplo, o resultado que o algoritmo converge no pior caso não é viável. Todavia, isso é mitigado com um grande número de execuções ou maior número de iterações no algoritmo. No melhor caso, os resultados do projeto do amplificador projetado com a ferramenta de otimização proposta alcançou melhoria em termos de Potência (8.86nW de média \pm 0.22 de desvio padrão em comparação com 18nW em relação ao projetado de forma manual), Margem de fase (59.99 ° \pm 10.15 em comparação 52.50 °) e de produto ganho largura de banda (3.81kHz \pm 0.65 em comparação 1.88kHz).

Palavras-chave: Projeto de circuitos analógico integrados. Algoritmos de otimização. Ferramentas CAD.

LIST OF FIGURES

Figure 1 – Analog design optimization procedure.	24
Figure 2 – Gaussian distribution example.	28
Figure 3 – Example of a greedy optimization algorithm. The arrow follows the minimum path at each iteration, trying to find the minimum point of the function.	29
Figure 4 – Optimization methodology for the problem of analog sizing.	33
Figure 5 – Optimization procedure including statistical evaluation for best solution candidates.	34
Figure 6 – Schematics of a low-voltage bulk-driven OTA	36
Figure 7 – Cost function mean behavior according to the number of nests.	39
Figure 8 – Cost function standard deviation behavior according to the number of nests.	40
Figure 9 – Cost function behavior according to the number of nests considering the empirical rule and confidence interval.	41
Figure 10 – Cost function evaluation.	42
Figure 11 – Distribution performance.	44
Figure 12 – Bode diagram of the optimized solution.	45
Figure 13 – Phase x Frequency.	46
Figure 14 – Slew-rate.	46

LIST OF TABLES

Table 1 – Performance indicators.	43
Table 2 – Confidence interval. \bar{X} is the calculated mean via simulation. CI_{low} and CI_{high} are the bounds for the mean with 99.7% of confidence.	43

LIST OF ACRONYMS

A/D Analog-to-Digital

A_{v0} Low-frequency gain

CAD Computer-Aided Design

CS Cuckoo Search

CSA Cuckoo Search Algorithm

D/A Digital-to-Analog

EDA Electronic Design Automation

GA Genetic Algorithm

GBW Gain Bandwidth Product

GSA Gravitational Search Algorithm

ICMR Common Mode Input Voltage Range

OTA CMOS Miller Operational Transconductance Amplifier

PM Phase margin

P_{cons} Power consumption

PSO Particle Swarm Optimization

SPICE Simulation Program with Integrated Circuit Emphasis

SR Slew rate

SUMMARY

1	INTRODUCTION	23
2	THEORETICAL BACKGROUND	27
2.1	Monte Carlo	27
2.2	Confidence interval and Empirical Rule	27
2.3	Optimization Algorithms	29
2.4	Abstraction of a problem as a cost function	30
2.5	Operational amplifier case	31
2.6	Related work	32
3	PROPOSED METHODOLOGY	33
3.1	Optimization Algorithm	34
3.2	SPICE simulation	35
3.3	Case Study	36
3.4	Random behavior of the Algorithm	37
4	RESULTS	39
4.1	Statistic analysis of number of nests	39
4.2	Study case design and behavior	40
5	CONCLUSION	47
	BIBLIOGRAPHY	49

1 INTRODUCTION

Analog integrated circuits are very used in several applications, such as communication systems, Analog-to-Digital (A/D)/ Digital-to-Analog (D/A) converters, and interface circuits. The design flow of such circuits is different from its digital counterpart. It is required to size individually each device, such as transistors, resistors and capacitors, for achieving the desired electrical behavior. Each analog block has its own particular performance features, which turns the design automation more complex. Digital integrated circuit design, by the other side, takes advantage of the circuit regularity and relies on high automation level and well-established Computer-Aided Design (CAD) tools (GRAEB, 2007).

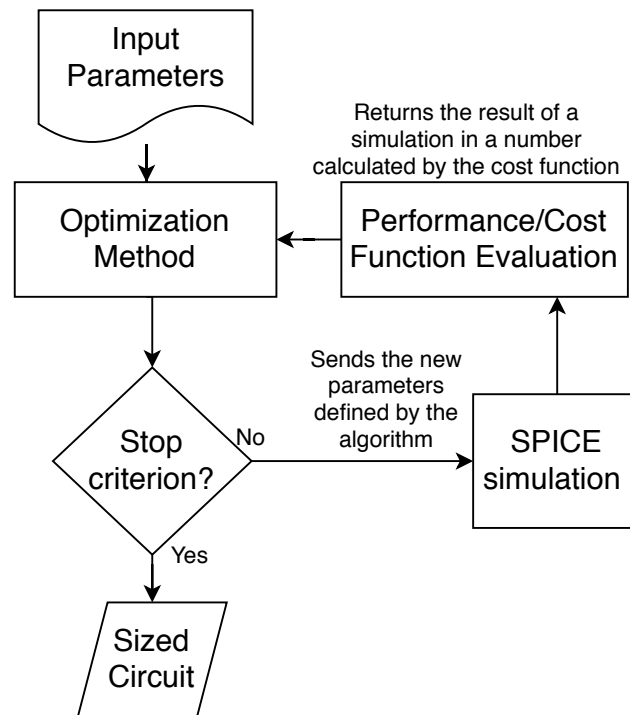
Traditional analog design methodology uses Simulation Program with Integrated Circuit Emphasis (SPICE) simulator (GRAEB, 2007) to estimate circuit performance without having to build the physical system. With a SPICE tool, it is possible to perform electrical simulation for a given process technology. Thus, a designer can estimate the features of the designed circuit and verify whether or not it is necessary to change some parameter to improve performance (ANTOGNETTI; MASSOBRIO, 1990). Therefore, the traditional trial-and-error procedure of sizing an integrated circuit consists in manual sizing each component of the system followed by electrical simulations using SPICE tool. The experience of the designer is fundamental in this case, since the search for a sized circuit with good performance is dependent on the ability to adjust the correct design parameters. However, an electrical simulation may take days or even months depending on the project complexity. The designer needs to wait this simulation finish for trying a new solution. Furthermore, it becomes increasingly difficult to find qualified professionals to size complex circuits in a short design time and with a high degree of reliability (BALKIR; DÜNDAR; ÖGRENCI, 2003).

An alternative to the manual design methodology is to abstract the problem into an optimization procedure. The circuit is modeled as a non-linear optimization problem in which devices sizes are the free variables. Each feature of the circuit is simulated by a SPICE tool and a cost function evaluates the circuit performance. From the cost function, the optimization algorithm perturbs the design parameters until the circuit performance converges to an optimal point. In the literature, it is possible to find optimization algorithms applied to this problem, such as Genetic Algorithm (GA) (JAFARI; SADRI; ZEKRI, 2010), Cuckoo Search (CS) (FORTES; SILVA; GIRARDI, 2018) and hybrid approaches with Particle Swarm Optimization (PSO), and Gravitational Search Algorithm (GSA) (MALLICK et al., 2017).

Figure 1 depicts a simplified process of analog design optimization using a CAD tool. The optimization method initialize randomly the design parameters and sends the circuit to the SPICE tool for simulation. The SPICE tool simulates and returns the output to the cost function that evaluates the result. If the stop criterion is not achieved,

the optimization method perturbs the design parameters and sends the circuit again to the simulator. The stop criterion can be defined as the number of runs or the minimal variation of the cost function. At the end, the algorithm returns the best circuit found thus far. This process is more detailed in Chapter 2.

Figure 1 – Analog design optimization procedure.



Source: the author.

The technology used on analog integrated circuits has a high degree of variability in their physical parameters. The variability affects directly circuit performance. For example, if a performance specification (e.g. low-frequency voltage gain) in a circuit has nominal value of 60 dB and standard deviation of 5 dB, and considering a normal distribution, we can calculate that the fabricated units having this specification ranges its values from 45 dB to 75 dB in 99.7% of the cases. Thus, this work also considers the variability of each performance specification.

A SPICE tool can estimate performance specification using different methods. The first method is with a nominal simulation, performing the simulation for only one sample, disregarding the variability of a circuit production. Other method is the corner simulation, to estimate the worst case (e.g., estimating that 99% circuits instances are above a value). It is a good approach for getting a circuit yield estimation. However, the corner parameters are, in general, extracted for digital circuits and might show a false worst case for analog integrated circuits because it tends to overestimate performance in the extreme cases. Finally, the method that shows the behavior of many circuits is with

the Monte Carlo simulation, performing the simulation of many samples and considering the variability procedure.

The Monte Carlo simulation randomly selects values for variables within a given range for estimating the system performance. The result is a Gaussian distribution curve that describes the output according to the variation of these variables. It has been applied to a diversity of problems such as in biology for estimating the likelihood of an individual having any disease, or in finance to estimate the risk of financial investment (BINDER et al., 1993). For analog integrated circuit design, Monte Carlo is used for simulating the performance of a circuit under process parameter variability. It is necessary to moderate the use of Monte Carlo simulation because each simulation can take days or even months. For example, if the number of Monte Carlo simulations is 1000, it is necessary to repeat SPICE simulations 1000 times with different values for process parameters (GRAEB, 2007). Thus, it is necessary to reduce the time spent with Monte Carlo simulations when exploring the search space. A strategy is to make pre-estimation of circuit performance with nominal parameters (with only one sample, without Monte Carlo). If the nominal simulation has an impracticable result, the Monte Carlo is not considered, otherwise it evaluates the Monte Carlo simulation.

This work evaluates the circuit described by (FERREIRA; SONKUSALE, 2014) automatically exploring and evaluating the design space. For this purpose, the work consists of modeling the analog sizing problem as an optimization problem and solving it, intending to search a feasible solution. Part of the work is a spin-off of the work of (FORTES; SILVA; GIRARDI, 2018), which intends to use algorithms for sizing analog integrated circuits with better performance compared to the manual sizing. The contribution of this work is the insertion of statistical analysis in the optimization loop for the search of solutions under process parameter variability. Also in this work, the tool was coded in Python.

This work is organized as follows: Chapter 2 provides a background of the methods, algorithms and related works used for the development of this work; Chapter 3 describes the methodology used for this work; Chapter 4 shows the results of the performance of this work; and, finally, Chapter 5 presents the final remarks.

2 THEORETICAL BACKGROUND

This chapter gives a background about the statistical methods, optimization algorithms, problem modeling and the state of art of Electronic Design Automation (EDA) tools for optimum sizing of analog integrated circuits.

2.1 Monte Carlo

The analog integrated systems projects are influenced by the variability procedure. Circuits with the same size, dimensions, project specifications and even from the same manufacturing round tends to present different performance. This occurs because of the randomness nature of some components (i.e., process parameter variability). Considering this variability, only one circuit instance is not enough to represent the circuit behavior. Many simulations considering the process parameter variability for a specification performance could demonstrate this behavior statistically with the evaluation of the solution quality.

Monte Carlo simulation is the basis for the methods used to estimate performance of many circuits samples considering the distribution of a parameter under process variability. It is possible to obtain mean values and standard deviation of each performance feature. This simulation is used to generate the distribution of a function from random values (BINDER et al., 1993).

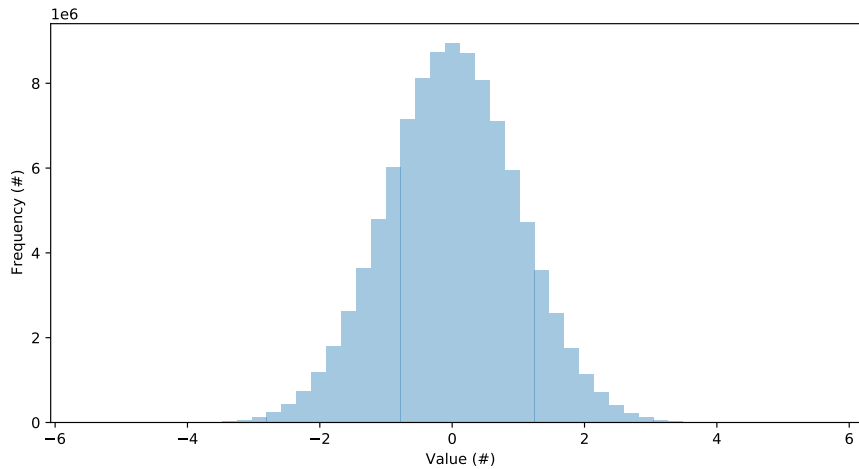
The random sampling is the simplest method to generate samples based on pseudo-random numbers. Basically, given constraints and bounds, the Random Sampling method generate a set of valid inputs. With these inputs it is possible to extract performance specifications from different circuit simulations. The random sampling tends to generate reliable solutions (similar to the real world), but requires many runs, which takes a lot of time to simulate.

2.2 Confidence interval and Empirical Rule

The Monte Carlo algorithm is designed to achieve a Gaussian distribution (BINDER et al., 1993). The analog integrated circuits in the manufacturing process tends to have the same behavior of this distribution. The Gaussian distribution is characterized by a greater quantity of samples close to a median value and a small quantity close to the left and right limits. Figure 2 presents an example of Gaussian distribution with standard deviation ($\sigma = 1$) and sample mean ($\bar{x} = 0$). The majority of the samples tends to the mean (0). There are other rare samples distant from the mean. The sum or multiplication between Gaussian distributions result in another Gaussian distribution. Thus, the final distribution presents the same pattern of the original distribution.

For an analog integrated circuit design, it is necessary to know this behavior given a performance specification distribution (LUO et al., 2008). The empirical rule is a reliable

Figure 2 – Gaussian distribution example.



Source: the author

method to estimate the project behavior. According to the rule, 68% of the samples x are on the interval $\bar{x} - \sigma \leq x \leq \bar{x} + \sigma$. 95% of the samples x are on the interval $\bar{x} - 2 \cdot \sigma \leq x \leq \bar{x} + 2 \cdot \sigma$. 99.7% of the samples x are on the interval $\bar{x} - 3 \cdot \sigma \leq x \leq \bar{x} + 3 \cdot \sigma$ (PUKELSHEIM, 1994).

Even knowing the variability, there are another issue: the obtained mean is different from the real mean. This occurs because a random procedure with a finite number of samples could not represent perfectly a distribution. A larger number of samples tends to mitigate this problem. However, it is necessary to estimate this error. It can be done by estimating the confidence interval (CI, 1987). This rule defines the probability of the mean to be inside a given range. Equation 2.1 shows how it can be obtained:

$$CI = \bar{X} \pm t \frac{\sigma}{\sqrt{n}} \quad (2.1)$$

Here, σ is the standard deviation, \bar{X} is the mean, n is the number of samples and t is a variable calculated according the confidence level. Equation 2.2 shows an example for calculating the confidence interval with 95% of confidence:

$$CI = 1.77 \pm 1.96 \frac{0.24}{\sqrt{30}} \quad (2.2)$$

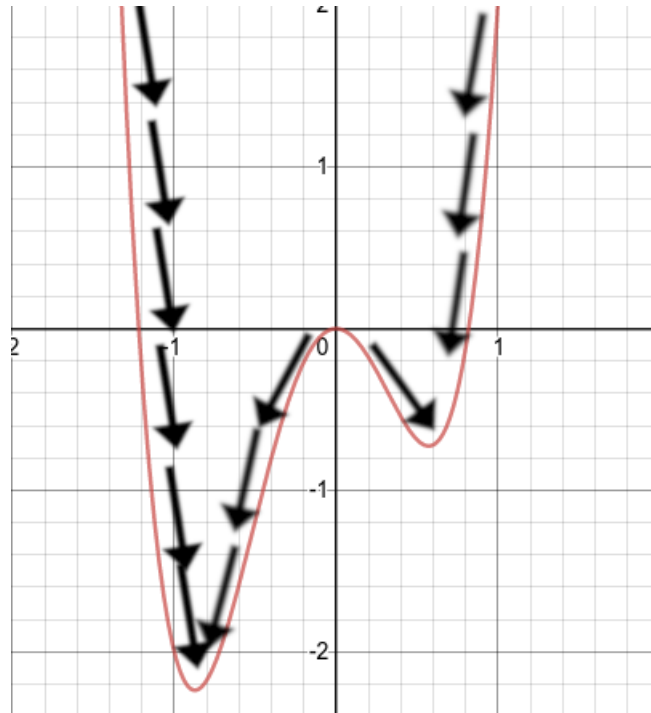
Here, the sample is equals to 1.77 ($\bar{X} = 1.77$). The variable $t = 1.96$ because it is the value determined for 95% of confidence (CI, 1987). The standard deviation is equals to 0.24 ($\sigma=0.24$). Finally, $N=30$ because it is the sample size. The result presents two confidence intervals. The low confidence interval equals to 1.68 and the high confidence interval equals to 1.86.

2.3 Optimization Algorithms

An optimization algorithm is used to search for a solution into a quantifiable problem. There are some classic optimization algorithms such as gradient descent and hill climbing (RUDER, 2016) (GOLDFELD; QUANDT; TROTTER, 1966). They are known as greedy methods, because they make a better choice at each step as they attempt to find the overall optimal way to solve the entire problem. These algorithms only guarantees the best solution in convex functions that have only a global minimum.

The Figure 3 illustrates a greedy algorithm behavior in a non-convex function. The algorithm (arrow) starts on random a point, following the minimum path, trying to find the minimum point of the function. In this case, the objective is to find the smallest possible value. Considering Figure 3, the algorithm may converge to the best solution, starting from a point less than zero (horizontal axis). However, it may converge to a local minimum if it starts from a point grater than 0. It occurs because the function is non-convex and have a global minimum (in some cases more than one with the same values) and a local minimum, making it difficult to find the global minimum. The problem of analog circuit sizing is non-convex. Thus, greedy methods such as the gradient descent may converge to a local minimum.

Figure 3 – Example of a greedy optimization algorithm. The arrow follows the minimum path at each iteration, trying to find the minimum point of the function.



Source: the author

There are in the literature other methods capable of find the best global solution,

such as Simplex (NELDER; MEAD, 1965). However, the Simplex method only guarantees the best solution in linear equations, which is not the case of analog integrated circuits. Even, it is possible to find in the literature algorithms for non-linear functions similar to Simplex methods. However, these similar methods require an equation to find the solution. In analog sizing, we can not model circuit performance features in simple equations, we can only estimate them from simulation. Even knowing the equations, they are very complex, making them unfeasible to solve the problem from an optimization method based on analytical equations.

Another way to optimize non-linear problems is via a heuristic. This procedure find a solution without guarantee to find the best solution. However, the generated solution might be close to the optimum. Some heuristics are capable to escape from a local minimum, such as Evolutionary Algorithms, Cuckoo Search and Simulated Annealing (ZITZLER; DEB; THIELE, 2000) (LAARHOVEN; AARTS, 1987) (YANG; DEB, 2009).

2.4 Abstraction of a problem as a cost function

Circuit sizing optimization problem can be modeled as a cost function. The optimization algorithm is responsible for minimizing this cost function, which can be an abstraction of a real problem in terms of circuit variables and performance features. The weight of each performance feature can be defined by the user, by the tool or even randomly. An example of cost function is shown in Equation 2.3 with a quantity of N features (x_i), each one with a weight w_i in which i is the index of each specification performance representation. This is a weighted average sum of performance features.

$$cost = \sum_{i=0}^N (w_i \cdot x_i) \quad (2.3)$$

Other example is according the sum of performance features x_i normalized by a reference parameter (divided by a reference value \hat{x}_i) as described on Equation 2.4. In this work, the reference (\hat{x}_i) means the results obtained by (FERREIRA; SONKUSALE, 2014) with the same circuit description. This cost function presents the same instance of Equation 2.3. However, with the normalized terms, the equation tends to have similar weights to the cost function compared to arbitrary weights.

$$cost = \sum_{i=0}^N \left(\frac{x_i}{\hat{x}_i} \right) \quad (2.4)$$

For evaluating the cost function of the Monte Carlo simulation it is required to use other approach for considering process variations that influence the performance features. An example is using the empirical rule. According to this rule: 68% of all samples are between $\bar{x}_i - \sigma_i$ and $\bar{x}_i + \sigma_i$, 95% are between $\bar{x}_i - 2\sigma_i$ and $\bar{x}_i + 2\sigma_i$, and 99.7% are between $\bar{x}_i - 3\sigma_i$ and $\bar{x}_i + 3\sigma_i$ (PUKELSHEIM, 1994). Using the empirical rule with Equation 2.4

it is possible to define Equation 2.5 that is the main way for evaluating the cost function in the present work.

$$cost = \sum_{i=0}^N \left(\frac{\bar{x}_i + 3\sigma_i}{\hat{x}_i} \right) \quad (2.5)$$

The example illustrated on Equation 2.4 is only for objective functions. However, in an optimization problem, there are also some hard constraints (features that could not exceed a threshold and do not contribute to the cost function), or soft constraints (when the condition is not satisfied, the objective cost function is penalized). In this work, we considered only soft constraints. Equation 2.6 represents a soft constraint function, in this case, y is the constraint feature and $softcons$ is the function that, given a constraint, returns if the cost function should be penalized.

$$\begin{aligned} softcons(y) &= y \quad if(y > \hat{y}); \\ &0 \quad if(y \leq \hat{y}) \end{aligned} \quad (2.6)$$

Equation 2.7 shows the final cost function, including objective and soft constraints. The x variable represents the performance features and y represents the constraints.

$$cost = \sum_{i=0}^N \left(\frac{\bar{x}_i + 3\sigma_i}{\hat{x}_i} \right) + \sum_{i=0}^N \left(\frac{softcons(\bar{y}_i + 3\sigma_i)}{\hat{y}_i} \right) \quad (2.7)$$

2.5 Operational amplifier case

An example of analog integrated circuit to be sized is the operational amplifier. The purpose of this circuit is to amplify an input signal with low distortion and low power consumption. It is also used for building filters, A/D and D/A converters and other systems.

There are some design parameters that impacts directly on circuit performance of operational amplifiers. They depend on the manufacturing technology used to design the circuit. In general, design parameters of the operational amplifier are transistors width and length, current source values, capacitor values and resistor values. The variation of these parameters impacts directly the circuit performance and behavior with external sources. These parameters are modeled as input of the algorithm described in Section 2.3.

Circuit performance is measured by area, power consumption, GBW, Low-frequency gain (A_{v0}), PM, Slew rate (SR), Output Swing, Common Mode Input Voltage Range (ICMR), among others. Performance features can be divided in constraint specifications and objective specifications. Constraint specifications are the performance features that are bounded by a max/min value, but do not need to be optimized. The objective specifications are that ones that must be maximized (or minimized) in the optimization procedure (GRAY; MEYER, 1982). For example, the designer can determine that

power consumption must achieve the smallest possible value (objective specification) and low-frequency gain must be higher than 60 dB (constraint specification). As far as a low-frequency gain higher than 60 dB does not contribute to the reduction of the cost function, the search for a smaller power consumption continues regardless of its current value. These performances are combined in a single value by the evaluation of the cost function, as described in Section 2.4, and modeled as the output of the algorithm described in Section 2.3.

2.6 Related work

There are some research effort towards analog sizing optimization described in the literature.

The WiCkeD framework proposes the use of worst-case analysis for searching optimum solutions in a high-dimensional design space. It is suitable for designing robust circuits under the effect of process variability. However, specific design models need to be available for the target process technology (GRAEB, 2007) (ANTREICH et al., 2000).

The UCAF tool is another system for sizing analog circuits automatically. The method used to search for the best solution is the yield analysis. With this method it is possible to optimize a circuit with few interactions with the designer. UCAF uses HSPICE tool for electrical simulation, which provides models that guarantees the operation in all device regions (SEVERO; KEPLER; GIRARDI, 2015). The present work is also based on this tool.

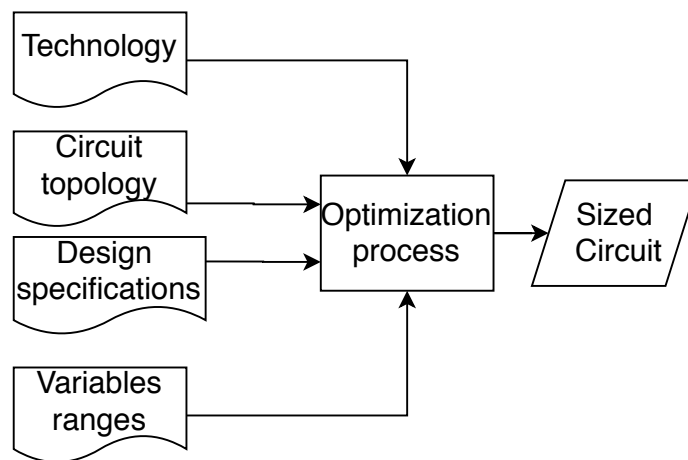
The Γ (Gamma) framework provides a sizing tool which uses a look-up table (LUT) method that is faster to converge to a solution than SPICE simulation and, according to the authors, has an accuracy similar to that of a SPICE simulation. However, this technique requires a set of equations exclusive for the target process technology. So, the migration for a new technology demands the algorithm to be rewritten. In addition, the tool does not allow a fully automated circuit analysis, requiring the expertise of a designer (STATTER; CHEN, 2016a) (STATTER; CHEN, 2016b).

Also, we can find in the literature the use of some algorithms for circuit optimization. (DEHBASHIAN; MAYMANDI-NEJAD, 2017) proposes an hybrid approach for analog integrated circuits optimization. An hybrid algorithm using concepts of PSO and GSA is responsible for the optimization. The algorithm is responsible for the reduction of power consumption and area in a two-stage CMOS op-amp. That work uses the HSPICE tool to simulate the circuits. (MALLICK et al., 2017) also proposes the same algorithm for optimization. The work proposes the optimization of two circuits: the differential amplifier circuit with current mirror load and the two-stage operational amplifier circuit. For simulation, it uses the Spectre tool. (SASIKUMAR; MUTHAIAH, 2017) proposes a hybrid approach of the PSO and GA. In this case, it is not used a SPICE tool for performance evaluation. Equations are responsible for estimating the circuit behavior.

3 PROPOSED METHODOLOGY

The methodology proposed in this work consists in modeling the analog sizing problem as an optimization problem and searching the design space for a near-optimum solution. Figure 4 depicts the adopted procedure. The designer selects the desired circuit topology and defines the design requirements, which are the circuit specifications that must be pursued. Using the target technology device parameters, an optimization algorithm performs the design space exploration searching for a circuit that attends the requirements.

Figure 4 – Optimization methodology for the problem of analog sizing.

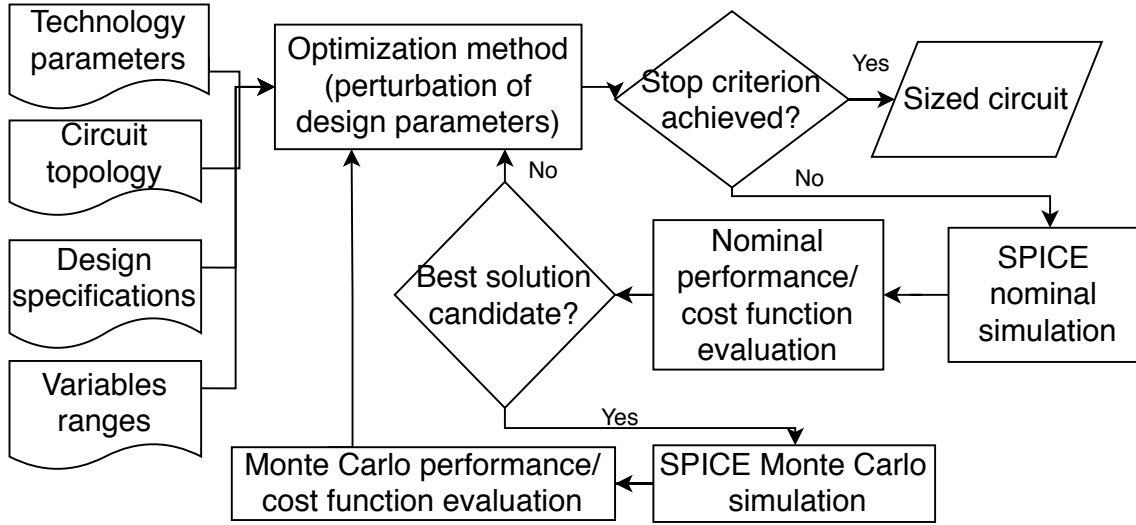


Source: the author

Figure 5 illustrates more details about the optimization procedure illustrated in Figure 4. The optimization method initializes randomly the design parameters and sends the circuit description to the SPICE tool for simulation. The simulated circuit is returned and the cost function is evaluated. If the simulation presents a feasible performance, the algorithm applies a Monte Carlo simulation. Otherwise, the algorithm considers a high-cost solution, without applying Monte Carlo. The strategy of applying Monte Carlo only to feasible solutions reduces considerably the total optimization time, avoiding unnecessary Monte Carlo simulations. After an iteration, if the stop criterion is not achieved, the optimization method perturbs the design parameters and sends the new circuit again to the simulator. The stop criterion can be defined as the maximum number of iterations or the minimal variation of the cost function, for example. At the end, the algorithm returns the best found circuit solution.

The optimization procedure is implemented as a script written in Python as described in Figure 5. Basically it is responsible for: a) randomly select the project parameters; b) run SPICE simulation with these parameters; c) read the simulation output; d) calculate each performance feature with the results of the output simulation; e) calculate the

Figure 5 – Optimization procedure including statistical evaluation for best solution candidates.



Source: the author

cost function; f) return the best found cost if reached the stop criterion.

3.1 Optimization Algorithm

The optimization algorithm is responsible for perturbing design parameters since the circuit is abstracted as an optimization problem. The chosen algorithm is the Cuckoo Search, due to its simplicity and efficiency (YANG; DEB, 2009).

The Cuckoo Search Algorithm (CSA) is based on the behavior of a bird species denominated cuckoo. The cuckoo put its eggs in nests belonging to other birds. It tries to minimize the probability of the host bird discovers the intruder egg by imitating its characteristics. In this case, an egg is an abstraction of a calculated cost. The good costs represent the survivor eggs. In addition, the algorithm uses the concept of Lévy flight for exploring the design space.

Algorithm 1 demonstrates the CSA via Lévy flights. Given a cost function, the algorithm generates a random population (i.e., a vector of costs obtained from random inputs applied to the function). After, the algorithm starts a loop intending to find a solution, repeating while a stop criterion is not archived. The algorithm obtains new solutions from the Lévy flight as following:

$$\mathbf{r}_k(t+1) = \mathbf{r}_k(t) + \alpha \cdot Lévy(\lambda) \quad (3.1)$$

Here, $r_k(t)$ is the current iteration, $r_k(t+1)$ is the next iteration and α is a scalar that defines the step between each iteration. This equation is multiplied by the Lévy

Algorithm 1 Cuckoo Search via Lévy Flights

Objective Function $f_c(\mathbf{r})$, $\mathbf{r} = (r_1, \dots, r_d)$;
 Generate initial population of N host nests $X_k (k = 1, 2, \dots, N)$;
while (NOT reaching the stop criteria) **do**
 Obtain new solutions by Lévy flights;
 Evaluate the quality of f_{c_k} ;
 Choose randomly a h nest between $1, \dots, N$;
 if $f_{c_k} > f_{c_h}$ **then**
 Replace h by the new solution;
 end if
 Replace a fraction (p) of the worst nests;
 Keep the best solutions of each nest;
 Sort the nests by cost;
end while

function with a step size λ . The Lévy flight step is defined by Equation 3.2.

$$p(l) = l^{-\lambda} \quad (3.2)$$

In this case, $1 < \lambda < 3$ and l are the steps in each iteration. In other words, Equation 3.1 with the Lévi flight is basically a random walking with a power function as step.

After the obtained solution set via Lévy flight, the algorithm evaluates the quality of these solutions. Then, the procedure chooses randomly a nest. If the new solution is better than the previous, it replaces the current solution by the previous. The best solution for each nest is maintained. Finally, the algorithm sorts the nests by their costs.

3.2 SPICE simulation

The SPICE simulation is responsible for evaluating the circuit performance for the generated solution in the optimization loop. In order to facilitate the simulation we used a generic circuit description and the design variables are parameters in the SPICE netlist. With this strategy, only a parameter file is necessary to edit for each new evaluation.

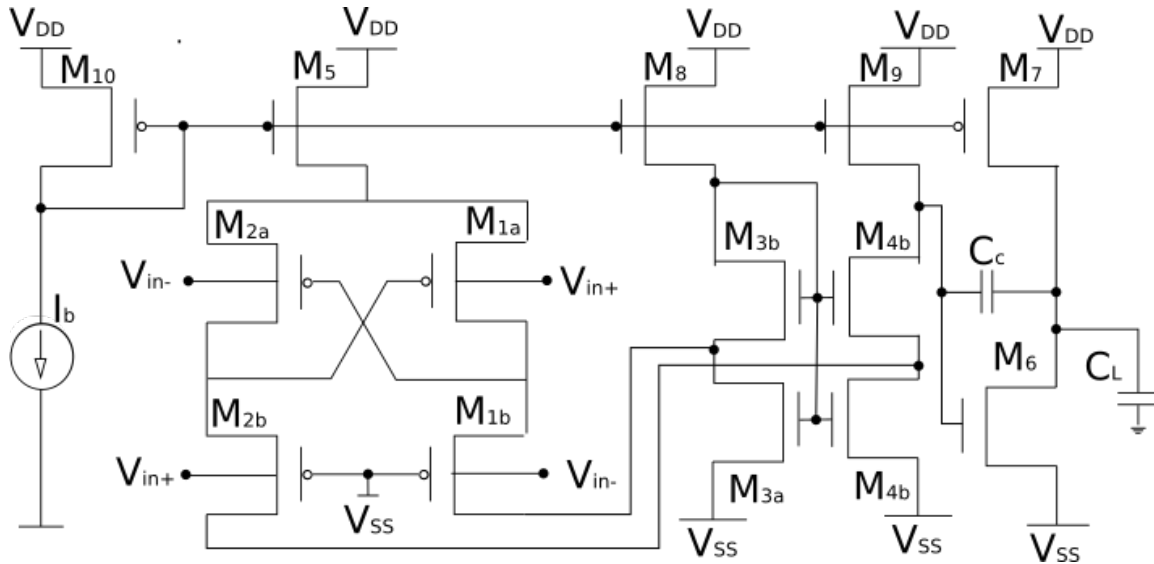
The script writes the parameter file and performs a system call to execute the SPICE simulation. Finally the script reads the output of the SPICE simulation and calculates the value of each specification of the specific analog circuit.

The HSPICE is the SPICE tool responsible for the simulation in this work because of its reliability and correct functioning in the technology used. This tool is developed and maintained by Synopsys (HSPICE, 2010). Another example of commercial SPICE tool is the Spectre, developed and maintained by Cadence (MARTIN, 2002). There are other SPICE tools, such as PSpice (TUINENGA, 1995), NgSpice (NENZI; VOGT, 2011) and LTSpice (MIKKELSEN, 2005), that can be also adapted to our system.

3.3 Case Study

We selected a case study for demonstrating the application of the proposed methodology. It is based on the bulk-driven operational amplifier proposed by (FERREIRA; SONKUSALE, 2014) and previously sized with a methodology proposed by (FORTES; SILVA; GIRARDI, 2018). The operational amplifier schematic is depicted in Figure 6. The work of (FORTES; SILVA; GIRARDI, 2018) used only nominal simulation for exploring the design space, thus not considering process variation. It means that the generated solution does not represent an optimum solution under process variability. It also used ideal current sources, which may not show the actual performance of the proposed system. Thus, this work intends to include yield analysis in the optimization procedure using real implementation for the current sources.

Figure 6 – Schematics of a low-voltage bulk-driven OTA



Source: adapted from (FERREIRA; SONKUSALE, 2014).

For evaluating each performance feature of the bulk-driven operational amplifier it is necessary to execute some procedures. Two types of simulations are necessary for extracting circuit characteristics: AC simulation for extracting Low frequency Gain, Phase Margin and GBW; and transient simulation for analysing the behavior of the circuit over time, in which it is possible to extract the slew rate of the circuit.

A_{v0} and the GBW are extracted from the output frequency response (Bode diagram). Equation 3.3 describes how A_{v0} is calculated:

$$A_v = 20 \cdot \log_{10}(V_{out}/V_{in}) \quad (3.3)$$

Here, V_{out} and V_{in} are the circuit output and input voltages, respectively. The

voltage gain is a complex function composed of a real (module) and an imaginary (phase) part. With the module we can extract the absolute low frequency gain and with the phase we can estimate phase margin. The extraction of GBW occurs at the frequency point in which the voltage gain is unitary.

The PM is possible calculated with the following equation:

$$PM = 180 - \text{abs}(\text{angle}(V_{out}/V_{in})) \quad (3.4)$$

The *angle* function gets the angle of a complex number. The *abs* function gets the absolute value of a real number.

Slew rate (SR) is calculated through transient simulation as:

$$SR = \min\left(\frac{\Delta V_{out_i}}{\Delta t_i}, \frac{\Delta V_{out_j}}{\Delta t_j}\right) \quad (3.5)$$

In this case, *min* is a function that returns the minimum element, ΔV_{out_i} is the variation of the output at the rising transition, Δt_i is the variation of time at the rising transition, ΔV_{out_j} is the variation of the output at the falling transition and Δt_j is the variation of time at the falling transition. Getting the minimum value guarantees the computation of the worst SR value.

The power consumption is calculated by equation

$$P_{cons} = I_{DD} \cdot V_{DD} \quad (3.6)$$

In this case, I_{DD} is the supply current and V_{DD} is the supply voltage.

Finally, area is computed as the sum of the gate area ($W \cdot L$) of all transistors.

We can separate circuit specifications in objectives and constraints. We define in this work P_{cons} , A_{v0} and GBW as objective functions to be minimized (maximized). The remaining performance features are constraints in the optimization problem, such as SR and PM.

The cost function is the sum of objective functions and constraint functions.

The constraint function contributes zero to the cost function if the obtained result x_i for the performance feature is greater (smaller) than a reference value \hat{x}_i . This reference value is defined by the user. The objective functions continuously contributes positively to the cost function and decreases the value as the performance function gets smaller (larger), even if it exceeds the reference.

3.4 Random behavior of the Algorithm

As shown in Section 3.1, the CSA presents a random behavior for calculating variable step. Because of this, the algorithm may present a different result for the same specifications, if using different seeds for random number generation.

Intending to mitigate this issue, it is performed a statistical analysis by executing many times the algorithm with same initial parameters, and getting different results. 30 executions for the same initial parameters seems to be reliable for demonstrating the algorithm behavior according to the parameters. This case does not require a very large number of execution because algorithm maintains a behavior according to the increase of the parameters.

This work uses the confidence interval applied in the cost results for verifying the mean deviation. According to this rule, as shown in Chapter 2, we could know the degree of confidence of the mean in a determined interval (CI, 1987). For this work, a confidence interval of 95 % is sufficient, considering that this confidence is tested in a total of 49 set of points.

Also, the empirical rule is responsible to verify the variation of the simulation, according to its randomness. This rule could guarantees that the mean more two times standard deviation of a set of points encompass 95% of the incidences.

4 RESULTS

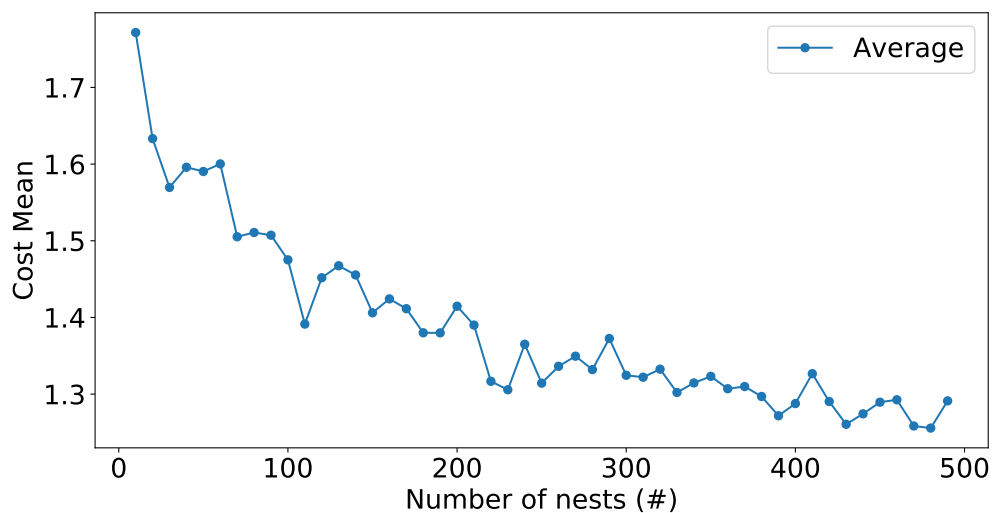
This chapter is divided in two parts. The first illustrates a statistical analysis of the CSA varying a parameter, intending to find the best parameters for a practical use of the tool. The second demonstrates some solutions achieved via the CSA with the defined parameter.

4.1 Statistic analysis of number of nests

CSA has some parameters that must be adjusted for the given application. The most important is the number of nests N . We tested the algorithm performance for different number of nests. Considering the algorithm randomness, it returns different results depending on the random seed. Thus, we performed the optimization search Thirty times for each number of nests. We varied the number of nests from 10 to 490. Considering that each search could take several computation hours, we used a step of 10 between the number of nests (in this case: 10, 20, ..., 490 nests). The tool designed a total of 1470 circuits with different number of nests as parameter and a fixed number of 50 iterations. The result was produced with simulations using Monte Carlo with 30 samples.

Figure 7 shows the mean cost for each number of nests. The cost tends to decrease quickly until around 200 nests. Thus, we could select the number of 200 nests as ideal for this application. However, the average is not always the real value of every simulation.

Figure 7 – Cost function mean behavior according to the number of nests.

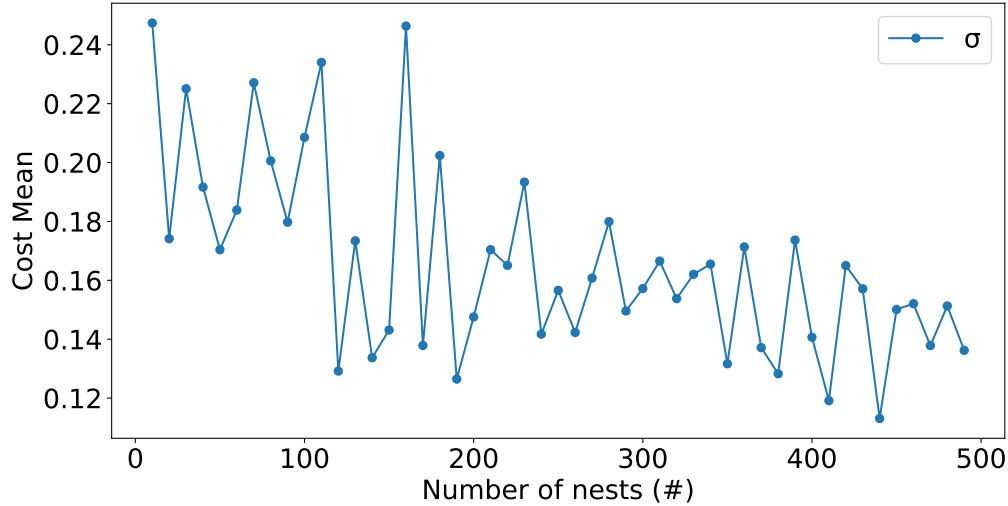


Source: the author

Intending to predict the solution variability of the solution, we measured the costs

standard deviation as illustrated in Figure 8. The standard deviation vary from a 0.12 to 0.24. It presents more variability and lower before approximately 200 nests, and more close to 0 and stable on the higher number of nests.

Figure 8 – Cost function standard deviation behavior according to the number of nests.



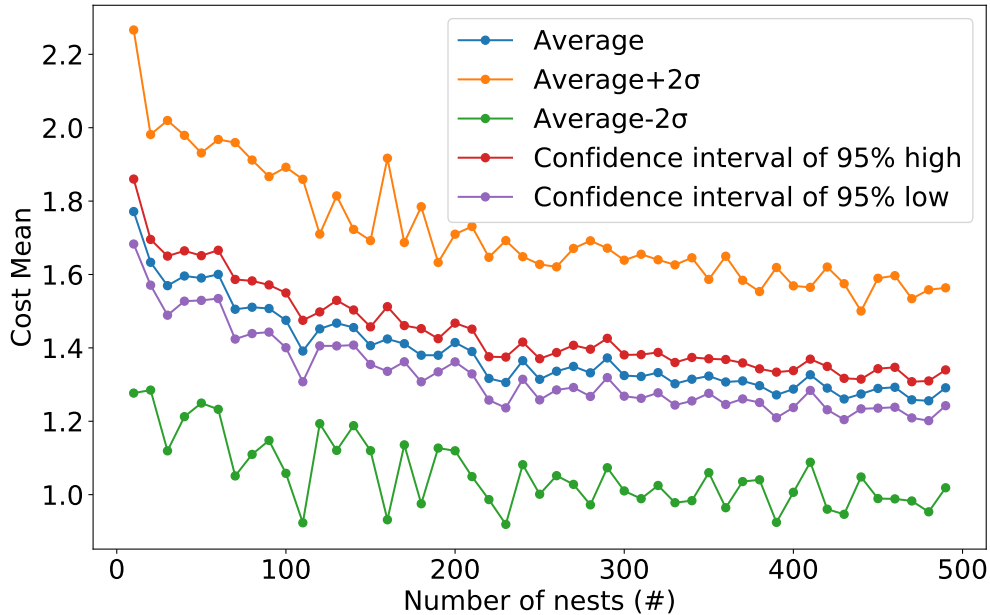
Source: the author

Knowing the mean and standard deviation, we could estimate the statistical behavior for new samples. According to the empirical rule, 95% of all samples are between $\bar{x} - 2\sigma$ and $\bar{x} + 2\sigma$. Considering this law, the green and orange lines on Figure 9 show the $\bar{x} - 2\sigma$, and $\bar{x} + 2\sigma$ solution behavior. The result shows that even the best solution, in the 5% of worst results, is worst than the mean of the worst solutions. Thus, in some cases, even a high number of nests can lead to a bad result. It is the greater problem of the tool. However, with a large quantity of simulations this problem is mitigated, tending to zero. Even considering the empirical rule, the mean of a sample might be reliable. Intending to predict how reliable is the sample, it is possible to use the confidence interval. As detailed in Chapter 2, according this law, we can predict how reliable is a sample (CI, 1987). The red and purple lines on Figure 9 demonstrates the behavior of the confidence interval with 95% of confidence in each point. The confidence interval demonstrates that the obtained mean, compared to the algorithm evolution with more nests, is close to the real mean.

4.2 Study case design and behavior

Considering the behavior of the best cost according the number of nests, we choose 180 nests, because the decrease in the best average cost is practically irrelevant after 180 nests. In addition, 180 nests is not a too large number, consequently it will not slow down

Figure 9 – Cost function behavior according to the number of nests considering the empirical rule and confidence interval.



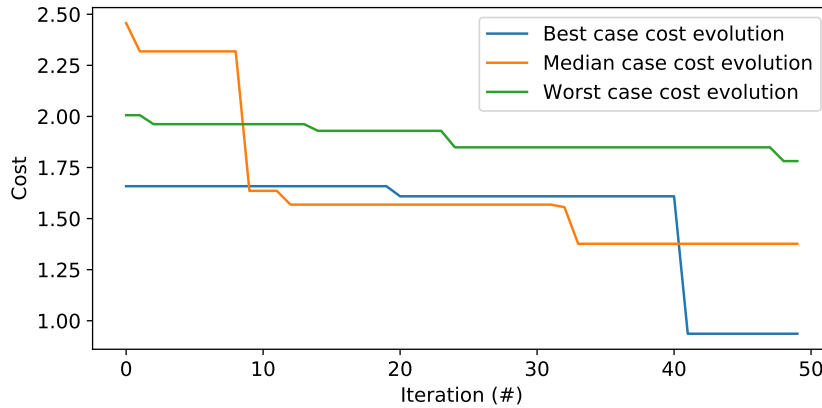
Source: the author

the algorithm convergence. The rest of this section consists in a behavior analysis of the best, median and worst solutions found for 180 nests, which are the worst, median and best cases for a practical use of the tool.

Figure 10 illustrates the evolution of the cost function in all three cases (best, median, and worst cases). Blue line illustrates the cost evolution for the best solution. It is possible to observe that the cost converge to a value less than 1. Comparing to the median solution (orange line) it has approximately 50% more cost, and the worst solution (green line) has approximately 1.8 times more cost. Thus, there are a large difference between the solution - practically the double of the cost - between the best and the worst solution. Also, the best solution demonstrates an abruptly decreasing, for less than 1, after iteration #40, meaning that the algorithm might converge to a good solution after some iterations with a poor cost.

Table 1 shows each performance specifications compared to (FERREIRA; SONKUSALE, 2014) and (FORTES; SILVA; GIRARDI, 2018). The first column is the name of each specification. The second, fourth and sixth columns shows the mean performance of the simulated circuits for the best, median and worst solutions, respectively. The third, fifth and seventh column show the standard deviation of each specification for the best, median and worst solution, respectively. The eighth column shows results obtained by

Figure 10 – Cost function evaluation.



Source: the author

(FORTES; SILVA; GIRARDI, 2018) for the same circuit with nominal simulation. Finally, the ninth column show the result obtained by (FERREIRA; SONKUSALE, 2014) using manual sizing. In all instances the area is larger compared to the reference. This occurs by two main reasons: the area is not considered in the cost function and this work implements the current sources with real transistors. Compared to Ferreira e Sonkusale (2014), the best solution of this work has better performance specifications mean in terms of GBW, PM and P_{cons} compared to the reference. The A_{v0} and SR are worst, however with a low difference and some instances of the circuits are better in all these specifications. The median solution has a similar solution compared to the best, however with a small GBW and one instance has an A_{v0} greater than 60 dB. The worst case converged to a poor solution, with exception the GBW. This solution is worse than the reference. In this case, the worst solution is infeasible in practice. Considering the used parameters, the algorithm can converge to an infeasible solution for commercial use. However, the algorithm explored only 50 iterations. In a real use case, the algorithm could explore for more time.

Table 2 shows the confidence interval of each specification performance for the best, mean and worst cases. The calculated confidence interval is for 99.7% of confidence, i.e., the method presents a finite number of samples to estimate the mean. Thus, the real mean could differ from the estimated mean. The confidence interval gives a reliability of 99.7% that the real mean is between the bounds. The results present a stable confidence interval. Even the lower bounds, for all specification performances, present similar solutions compared to the calculated mean. Table 1 shows only the mean and the standard deviation. Table 2 shows the specifications confidence interval. However, it is not possible to see the distribution of the simulated circuits.

Table 1 – Performance indicators.

Specification	Best cost (this work)		Median cost (this work)		Worst cost (this work)		Fortes (2018)	Ferreira (2014)
	mean	σ	mean	σ	mean	σ	nominal	measured
Gate Area (μa^2)	23892*	-	13858*	-	24479*	-	1719**	1372**
A_{v0} (dB)	48.22	10.69	42.86	2.32	36.70	5.86	67.64	60.00
GBW (kHz)	3.81	0.65	1.54	0.13	2.84	0.35	7.54	1.88
PM ($^\circ$)	59.99	10.15	53.68	7.56	51.59	4.78	88.00	52.50
SR ($\Delta V/\Delta ms$)	0.53	0.05	0.67	0.03	0.49	0.08	0.65	0.77
P_{cons} (nW)	8.86	0.22	9.57	0.24	18.19	0.45	10.12	18.00

* Considering the current sources area.

** Not considering the current sources area.

Source: the author

Table 2 – Confidence interval. \bar{X} is the calculated mean via simulation. CI_{low} and CI_{high} are the bounds for the mean with 99.7% of confidence.

Specification	Best cost			Median cost			Worst cost		
	CI_{low}	\bar{X}	CI_{high}	CI_{low}	\bar{X}	CI_{high}	CI_{low}	\bar{X}	CI_{high}
A_{v0} (dB)	47.22	48.22	49.22	42.65	42.86	43.08	36.15	36.70	37.25
GBW (kHz)	3.75	3.81	3.87	1.53	1.54	1.56	2.81	2.84	2.87
PM($^\circ$)	59.04	59.99	60.94	52.98	53.68	54.39	51.14	51.59	52.04
SR ($\Delta V/\Delta ms$)	0.52	0.53	0.53	0.67	0.67	0.68	0.49	0.49	0.50
P_{cons} (nW)	8.84	8.86	8.89	9.55	9.57	9.59	18.15	18.19	18.24

Source: the author

Figure 11 shows the distribution of each performance feature under variability for the best, median and worst solutions. Figures. 11 (a), (b) and (c) illustrate the PM. Compared to the reference (FERREIRA; SONKUSALE, 2014) (52.5°) the best case has 943 instances above the reference, the median 506 and the worst solution 393.

Figures 11 (d), (e) and (f) show the GBW. Comparing with (FERREIRA; SONKUSALE, 2014) (1.88 kHz of GBW), the best case has 981 instances with superior performance, the median only 1 and the worst solution 995.

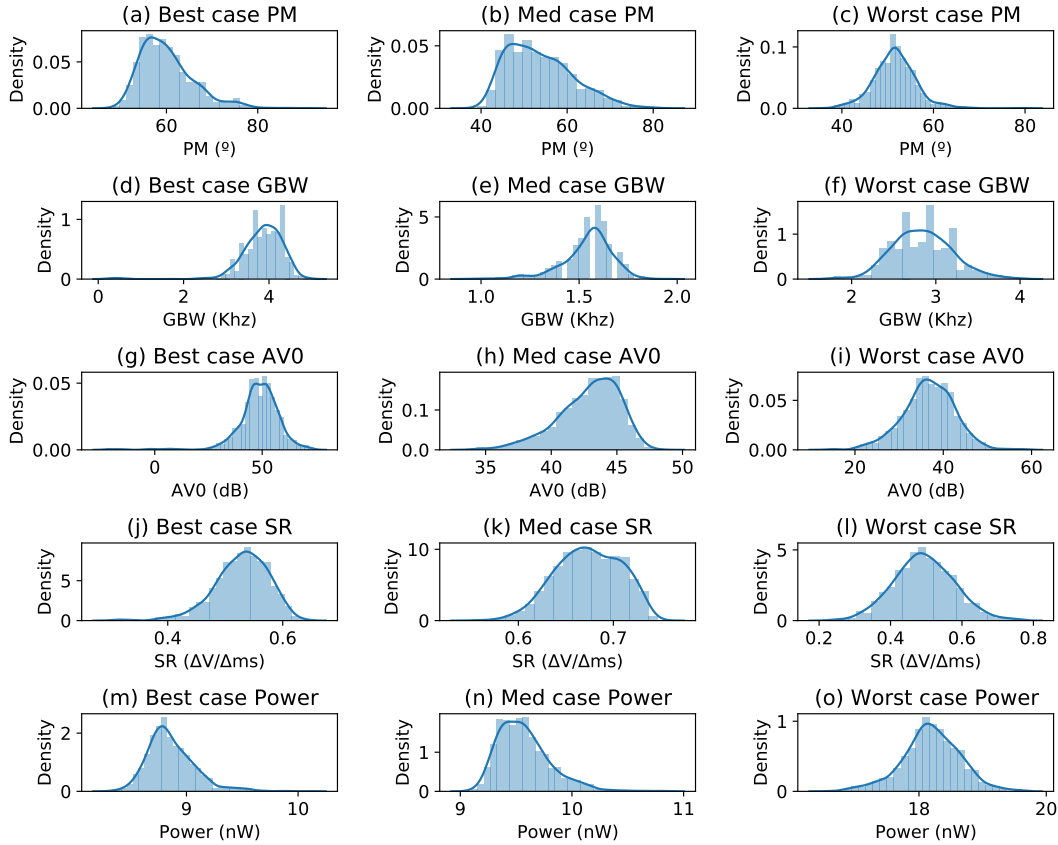
Figures 11 (g), (h) and (i) show A_{v0} , which is 60 dB on the reference. The best case has 70 instances with a performance superior to 60 dB and median and worst solutions have no solution above the reference.

For SR (Figures 11 (j), (k) and (l)), no sample overcame the reference of 0.77 V/ms. However, the best and the median solution achieved a SR close to this value.

Power consumption (Figures 11 (m), (n) and (o)) presented the best results, with all instances from best and median cases presenting smaller values than the reference of 18 nW. For the worst solution, only 304 instances overcame the reference.

Considering all features, with exception to the slew-rate, 53 instances overcame all references of (FERREIRA; SONKUSALE, 2014). Disregarding low-frequency gain, 934

Figure 11 – Distribution performance.

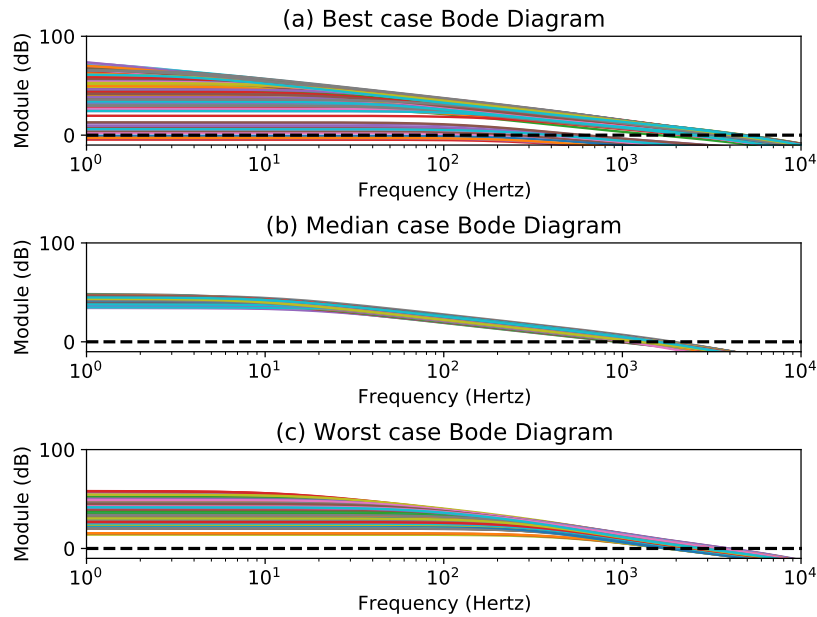


Source: the author

instances overcame the reference in the best cases.

Figure 12 shows the Bode diagram for the best, median and worst costs. Each line represents the behavior of one of 1000 simulated operational amplifiers with respect to frequency. In Bode diagram, the A_{v0} is the absolute value (dB) of the gain in low frequency. The value of the frequency (Hz) crossing the dotted black line represents GBW. Graphically, the best solution (Figure 12 (a)) appears the most influenced solution to the variability procedure (having instances with a negative A_{v0}). This occurs because some instances converges to a poor solution. Actually there are few instances with this poor solution and it is compensated on the final cost with greater majority of instances with a feasible result. The median solution (Figure 12 (b)), demonstrates the most stable result, however with a poor A_{v0} (42.86 dB) and GBW (1.54 kHz) mean. Finally, the worst solution (Figure 12 (c)) presents a poor and high variability A_{v0} . However, with a feasible GBW.

Figure 12 – Bode diagram of the optimized solution.

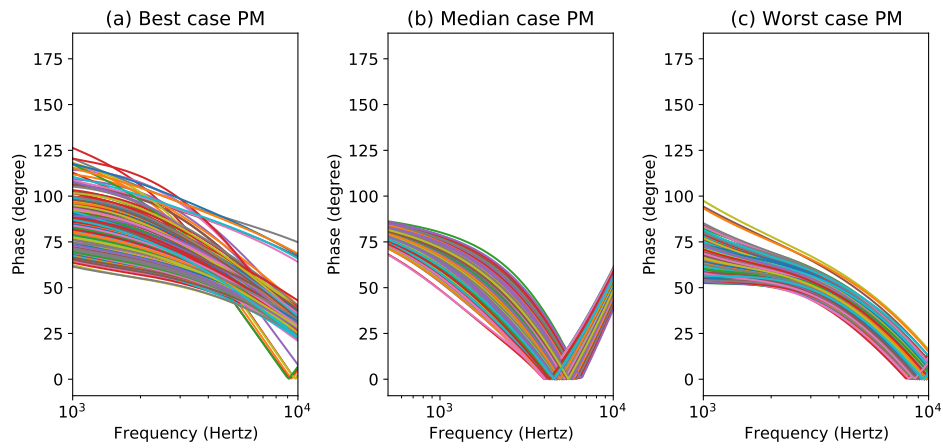


Source: the author

To calculate the PM, it is necessary to measure the phase in the unity gain frequency (dotted black line in Figure 12). Figure 13 shows the phase behavior in the interval which the module is equals to zero. It is possible to observe that all figures presents a similar pattern. The best result (Figure 13 (a)) presents the larger phase margin mean (59.99°), however with more sensitivity to the variability process ($\sigma = 10.15$). Even so, the other solution demonstrates a similar behavior. The median solution (Figure 13 (b)) presents a mean of 53.68° , and a bit less sensitive solution ($\sigma = 7.56$). Finally, the worst solution (Figure 13 (c)) demonstrates the stablest solution ($\sigma = 4.78$), however with the worst mean solution (51.59°).

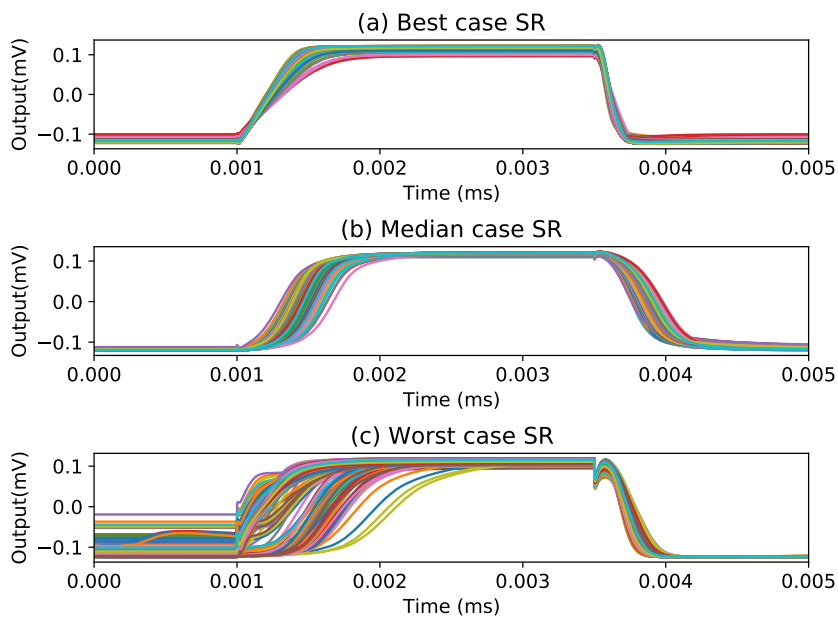
The SR is calculated as $\Delta V/\Delta ms$ on the maximum variation ascending and descending interval. The lowest SR mean the worst result. For this work we consider the worst result between the ascending and descending points for our SR. Figure 14 shows the pattern that SR was extracted for the best, median and worst solutions. The best case (Figure 14 (a)) presents a stable ($\sigma = 0.05$) and feasible SR (mean= $0.53\Delta V/\Delta ms$). The median solution (Figure 14 (b)) presents the lower variability ($\sigma = 0.03$) and best solution mean ($0.67\Delta V/\Delta ms$), with a small difference of the best solution. Also, the worst solution is close to the others, with a small variability ($\sigma = 0.08$) and a mean of $0.49\Delta V/\Delta ms$.

Figure 13 – Phase x Frequency.



Source: the author

Figure 14 – Slew-rate.



Source: the author

5 CONCLUSION

This work showed the behavior of a method for automatic design of analog integrated circuits under process variability. We abstracted the circuit design as an optimization problem and used CSA to find a feasible solution.

The methodology uses Monte Carlo simulation in the optimization loop to estimate circuit performance and calculate yield. Moreover, the analysis varying the number of nests shows the trade-off between solution quality and computational time. With the empirical rule and the confidence level, this work shows the statistical behavior of the method.

Results demonstrate that the tool can be used to design a bulk-driven OTA automatically, without human interference. The generated result reached a better solution than a hand-made design.

As future work, we propose the inclusion of a local search around the best solution found by CSA, intending to refine even more the final result. The generalization of the tool for other circuits topologies is also important to allow different kinds of circuits to be optimized. The analysis of the behavior of other optimization algorithms can be important for future improvements in the search for the best solution. Finally, the exploration of different sample methods can improve Monte Carlo execution in terms of simulation time, without losing a significant degree of reliability.

BIBLIOGRAPHY

- ANTOGNETTI, P.; MASSOBRIO, G. **Semiconductor device modeling with SPICE**. [S.l.]: McGraw-Hill, Inc., 1990. Cited at page 23.
- ANTREICH, K. et al. Wicked: Analog circuit synthesis incorporating mismatch. In: IEEE. **Proceedings of the IEEE 2000 Custom Integrated Circuits Conference (Cat. No. 00CH37044)**. [S.l.], 2000. p. 511–514. Cited at page 32.
- BALKIR, S.; DÜNDAR, G.; ÖGRENCI, A. S. **Analog VLSI design automation**. [S.l.]: CRC Press, 2003. Cited at page 23.
- BINDER, K. et al. Monte carlo simulation in statistical physics. **Computers in Physics**, AIP, v. 7, n. 2, p. 156–157, 1993. Cited 2 times at pages 25 and 27.
- CI, B. Confidence intervals. **Lancet**, v. 1, n. 8531, p. 494–7, 1987. Cited 3 times at pages 28, 38, and 40.
- DEHBASHIAN, M.; MAYMANDI-NEJAD, M. A new hybrid algorithm for analog ics optimization based on the shrinking circles technique. **Integration**, Elsevier, v. 56, p. 148–166, 2017. Cited at page 32.
- FERREIRA, L. H.; SONKUSALE, S. R. A 60-db gain ota operating at 0.25-v power supply in 130-nm digital cmos process. **IEEE Transactions on Circuits and Systems I: Regular Papers**, IEEE, v. 61, n. 6, p. 1609–1617, 2014. Cited 6 times at pages 25, 30, 36, 41, 42, and 43.
- FORTES, A.; SILVA, L. A. da; GIRARDI, A. Low power bulk-driven ota design optimization using cuckoo search algorithm. In: IEEE. **2018 31st Symposium on Integrated Circuits and Systems Design (SBCCI)**. [S.l.], 2018. p. 1–7. Cited 5 times at pages 23, 25, 36, 41, and 42.
- GOLDFELD, S. M.; QUANDT, R. E.; TROTTER, H. F. Maximization by quadratic hill-climbing. **Econometrica: Journal of the Econometric Society**, JSTOR, p. 541–551, 1966. Cited at page 29.
- GRAEB, H. E. **Analog design centering and sizing**. [S.l.]: Springer, 2007. v. 64. Cited 3 times at pages 23, 25, and 32.
- GRAY, P. R.; MEYER, R. G. Mos operational amplifier design-a tutorial overview. **Ieee journal of solid-state circuits**, IEEE, v. 17, n. 6, p. 969–982, 1982. Cited at page 31.
- HSPICE, S. Inc., dec. 2010. **Version E-2010.12**, 2010. Cited at page 35.
- JAFARI, A.; SADRI, S.; ZEKRI, M. Design optimization of analog integrated circuits by using artificial neural networks. In: IEEE. **2010 International Conference of Soft Computing and Pattern Recognition**. [S.l.], 2010. p. 385–388. Cited at page 23.
- LAARHOVEN, P. J. V.; AARTS, E. H. Simulated annealing. In: **Simulated annealing: Theory and applications**. [S.l.]: Springer, 1987. p. 7–15. Cited at page 30.
- LUO, P.-W. et al. Impact of capacitance correlation on yield enhancement of mixed-signal/analog integrated circuits. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, IEEE, v. 27, n. 11, p. 2097–2101, 2008. Cited at page 27.

MALLICK, S. et al. Optimal sizing of cmos analog circuits using gravitational search algorithm with particle swarm optimization. **International Journal of Machine Learning and Cybernetics**, Springer, v. 8, n. 1, p. 309–331, 2017. Cited 2 times at pages 23 and 32.

MARTIN, A. J. L. Cadence design environment. **New Mexico State University, Tutorial paper**, 2002. Cited at page 35.

MIKKELSEN, J. H. Ltspice—an introduction. **Technical report, Institute of Electronic Systems, Aalborg University, Aalborg**, 2005. Cited at page 35.

NELDER, J. A.; MEAD, R. A simplex method for function minimization. **The computer journal**, Oxford University Press, v. 7, n. 4, p. 308–313, 1965. Cited at page 30.

NENZI, P.; VOGT, H. **Ngspice Users Manual Version 23**. [S.l.]: June, 2011. Cited at page 35.

PUKELSHEIM, F. The three sigma rule. **The American Statistician**, Taylor & Francis Group, v. 48, n. 2, p. 88–91, 1994. Cited 2 times at pages 28 and 30.

RUDER, S. An overview of gradient descent optimization algorithms. **arXiv preprint arXiv:1609.04747**, 2016. Cited at page 29.

SASIKUMAR, A.; MUTHAIAH, R. Operational amplifier circuit sizing based on nsga-ii and particle swarm optimization. In: IEEE. **2017 International Conference on Networks & Advances in Computational Technologies (NetACT)**. [S.l.], 2017. p. 64–68. Cited at page 32.

SEVERO, L. C.; KEPLER, F. N.; GIRARDI, A. G. Automatic synthesis of analog integrated circuits including efficient yield optimization. In: **Computational Intelligence in Analog and Mixed-Signal (AMS) and Radio-Frequency (RF) Circuit Design**. [S.l.]: Springer, 2015. p. 29–58. Cited at page 32.

STATTER, Y.; CHEN, T. γ (gamma): A saas-enabled fast and accurate analog design system. **Integration**, Elsevier, v. 55, p. 67–84, 2016. Cited at page 32.

STATTER, Y.; CHEN, T. A novel high-throughput method for table look-up based analog design automation. **Integration**, Elsevier, v. 52, p. 168–181, 2016. Cited at page 32.

TUINENGA, P. W. **SPICE: a guide to circuit simulation and analysis using PSpice**. [S.l.]: Prentice Hall PTR, 1995. Cited at page 35.

YANG, X.-S.; DEB, S. Cuckoo search via lévy flights. In: IEEE. **2009 World congress on nature & biologically inspired computing (NaBIC)**. [S.l.], 2009. p. 210–214. Cited 2 times at pages 30 and 34.

ZITZLER, E.; DEB, K.; THIELE, L. Comparison of multiobjective evolutionary algorithms: Empirical results. **Evolutionary computation**, MIT Press, v. 8, n. 2, p. 173–195, 2000. Cited at page 30.