Synchronization of Chaotic Oscillators Optimized by Applying Evolutionary Algorithms

by

Victor Hugo Carbajal Gómez

In partial fulfillment of the requirements for the degree of

DOCTOR IN SCIENCES WITH SPECIALITY IN ELECTRONICS

in the
National Institute of Astrophysics, Optics and Electronics

December 2015
Tonantzintla, Puebla

Supervised by:

Dr. Esteban Tlelo Cuautle, INAOE

©INAOE 2015
The author gives INAOE the rights to reproduce and distribute partial or full copies of this Thesis.
Doctoral Thesis

By:
Victor Hugo Carbajal Gómez

Supervised by:
Dr. Esteban Tlelo Cuautle

National Institute of Astrophysics, Optics and Electronics
Department of Electronics

Tonantzintla, Puebla. December 2015
Abstract

Nowadays, chaos generators of multi-scroll chaotic attractors have received considerable attention to model, simulate, design and use them in engineering applications. The interest is in both theoretical and practical issues, while a direct application is in the field of chaotic synchronization to implement private communication schemes, where the confidential information being transmitted is embedded into a chaotic signal by direct modulation, masking, or another technique. At the receiver end, if chaotic synchronization is perfectly achieved, one can extract the hidden information from the transmitted signal. In this Thesis, two multi-scroll chaotic oscillators are synchronized. The synchronization is performed by generalized Hamiltonian forms and observer approach from nonlinear control theory.

While realizations using commercially available amplifiers are given, the main contribution of this Thesis is focused on the design of oscillators with integrated circuit technology, where the goal is programmability of the amplifiers to accomplish the required optimized coefficients of the dynamical system. The proposed integrated circuit realization can generate from 2 to 7 scrolls and in each case the unpredictability of the chaotic oscillator is guaranteed by optimizing the maximum Lyapunov exponent (MLE). Three meta-heuristics are applied to optimize MLE: Genetic Algorithm (GA), Differential Evolution algorithm (DE) and Particle Swarm Optimization (PSO). Among the feasible solutions for the coefficients of the oscillator, the behavior on the synchronization for chaotic oscillators with low and high MLEs, is presented in both numerical simulations and experimental results. Design simulations are presented before and after the layout parasitics extraction, and additionally, a process-voltage-temperature (PVT) analysis is performed to show the robustness of
the integrated chaotic oscillator with respect to variations in temperature and process corners using integrated circuit technology of $0.35\mu m$. 
## Contents

### Abstract

1. **Introduction** ................................................................. 1
   1.1. Chaos ..................................................................... 2
   1.1.1. Chaos metrics .................................................. 2
   1.1.2. Optimization ..................................................... 3
   1.1.3. Synchronization .................................................. 4
   1.1.4. Applications ..................................................... 5
   1.2. Integrated design of chaotic oscillators ....................... 5
   1.3. Thesis objectives .................................................... 9
   1.3.1. General goal ...................................................... 9
   1.3.2. Specific objectives ............................................. 9

2. **Chaotic systems** .......................................................... 11
   2.1. Dynamical systems .................................................. 11
   2.1.1. State variables model ....................................... 11
   2.1.2. Autonomous systems ........................................ 13
   2.2. Geometry and stability of dynamical systems ............... 13
   2.2.1. Equilibrium points .......................................... 15
   2.2.2. Limit cycles .................................................... 15
   2.2.3. Strange attractors and chaos .............................. 17
   2.3. Multi-scrolls chaotic oscillators ................................ 17
   2.3.1. Chua’s circuit generalization ............................. 18
   2.3.2. Chaotic oscillator based on saturated functions series .... 22
   2.4. Design of multi-scroll chaos generators from behavioral modeling ... 27
2.4.1. Multi-scroll chaotic oscillator based on saturated non-linear functions ........................................... 28  
2.4.2. Behavioral modeling of SNLF .................................................. 30  
2.5. Circuit implementation ......................................................... 31  
  2.5.1. OpAmp implementation .................................................. 32  
  2.5.2. CFOA implementation .................................................. 35  
  2.5.3. OTA implementation .................................................. 37  
  2.5.4. OTA macromodel .................................................. 39  
  2.5.5. OTA SNLF synthesis .................................................. 40  

3. Optimizing the Maximum Lyapunov Exponent with Heuristics 47  
  3.1. Computing Lyapunov exponents .................................................. 47  
  3.2. Optimization algorithms .................................................. 49  
    3.2.1. Genetic algorithm .................................................. 51  
    3.2.2. Differential Evolution algorithm .................................................. 52  
    3.2.3. Particle Swarm Optimization .................................................. 55  
  3.3. Maximizing the unpredictability in multi-scroll chaotic oscillators .................................................. 57  

4. Synchronization 69  
  4.1. Hamiltonian synchronization approach .................................................. 69  
    4.1.1. Generalized Hamiltonian systems .................................................. 70  
  4.2. Synchronization of multi-scroll chaotic attractors .................................................. 72  
    4.2.1. Circuit implementation .................................................. 74  
    4.2.2. Experimental synchronization results .................................................. 75  
  4.3. OTA circuit implementation .................................................. 76  

5. Design of integrated chaotic oscillators 83  
  5.1. Integrated implementations of chaotic oscillators .................................................. 84  
  5.2. Integrated design approach .................................................. 87  
    5.2.1. Error sources and block requirements .................................................. 87  
    5.2.2. Linear design requirements .................................................. 88  
    5.2.3. Nonlinear design requirements .................................................. 89  
    5.2.4. CMOS Gm-C linear building block .................................................. 90  
  5.3. Synthesis of piecewise-linear functions .................................................. 96  
    5.3.1. CMOS PWL generation by current shaping .................................................. 96
## CONTENTS

5.4. OTAs based multi-scroll oscillator ........................................ 106  
  5.4.1. SNLF circuit design .................................................... 106  
  5.4.2. Adjustable $g_m$ OTA design ....................................... 139  

6. Design performance and results .............................................. 159  
  6.1. Multi-scrolls chaotic oscillator design Layout ...................... 160  
  6.2. Post-Layout Simulations ............................................... 163  

7. Conclusions ........................................................................... 171  

Bibliography ............................................................................... 173  

Appendix .................................................................................... 190  

A. Published Works .................................................................... 193  
  A.1. Book Chapters ................................................................. 193  
  A.2. International Journals ...................................................... 193  
  A.3. Conference Proceedings ................................................ 194
Chapter 1

Introduction

Chaotic systems have been known for long time but only recently, it was shown that chaos could be controlled and therefore, synchronized [1–6]. For this reason; this class of systems promise to have a major impact on many novel, time-and energy-critical applications, such as high-performance circuits and devices (e.g. delta-sigma modulators and power converters), liquid mixing, chemical reactions, biological systems (e.g. in the human brain, heart and perceptual process), crisis management (e.g. in power electronics), secure information processing, and critical decision-making in political, economic and military events [7–12]. This new and challenging research and development area has become a scientific interdiscipline, involving systems and control engineers, theoretical and experimental physicist, applied mathematicians, physiologists and above all, circuit and devices specialists.

Some nonlinear systems show chaotic behavior, which is a bounded unstable dynamic behavior that exhibits sensitive dependence on initial conditions and includes infinite unstable periodic motions. Although it appears to be stochastic, it occurs in a deterministic nonlinear system under deterministic conditions.

Nowadays, multi-scroll chaotic attractors has received considerable attention with both theoretical and experimental issues. Every new chaotic system [1, 4, 13–17] is a candidate to improve engineering applications, which in electronics, the designer has the challenge of providing the best circuit implementation of reliable nonlinear circuits.
1.1. Chaos

The term Chaos refers to one type of complex dynamic behavior that has some very special features, such as extremely sensitive to small variations in the initial conditions, having bounded trajectories in the phase space, but with at least one positive Lyapunov exponent and continuous power spectrum.

In other words, Chaos is an aperiodic long-term behavior of a deterministic system that exhibits a very sensitive dependence on initial conditions. According to [18, 19], there are three basic characteristics of chaos, namely:

- Chaos exhibits behavior which is “difficult to distinguish from random behavior” because the paths do not tend to a fixed point, periodic orbits or quasi-periodic orbits as $t \to \infty$.

- Chaos is “deterministic” because the system does not have random inputs or parameters. That is, its present condition is a consequence of previous system states. The irregular behavior arises from nonlinearities of the system.

- Chaos “is very sensitive to initial conditions ” because the paths that are very close, with time separate exponentially.

Since these characteristics are nonlinear; chaotic behavior is much more complicated than that of linear systems. In fact, even the simplest chaotic system exhibit a bulk of different behaviors [20–22], that can only be fully analyzed with the help of powerful software resources [23–28].

1.1.1. Chaos metrics

In general, the equations describing a dynamic system are dependent on their parameters or set of parameters and chaotic behavior occurs only for certain values of these parameters [14, 29, 30]. The main characterization of chaotic systems can be performed qualitatively by graphical representation of the bifurcation diagrams and Poincaré maps; and quantitatively by calculating the fractal dimension, Kolmogorov entropy, and Lyapunov spectrum [23, 29]. Among them, the Lyapunov exponents provide a means of ascertaining whether the behavior of a system is chaotic. In this manner, the presence of positive Lyapunov exponents has often been taken as a signature of chaotic motion. In addition, a high-value of the positive Lyapunov exponent
indicates a high degree of unpredictability of the system, therefore, the system has a more complex dynamic behavior [31].

1.1.2. Optimization

Mathematical Optimization is the process of the formulation and the solution of a constrained problem of the general mathematical form:

\[
\text{minimize } f(x), \quad x = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}
\]

subject to the constraints:

\[
g_j(x) \leq 0, \quad j = 1, 2, \ldots, m \\
h_j(x) = 0, \quad j = 1, 2, \ldots, r
\] (1.1.1)

where \( f(x) \), \( g_j(x) \) and \( h_j(x) \) are scalar functions of the real column vector \( x \).

The continuous components \( x_i \) of \( x = [x_1, x_2, \ldots, x_n]^T \) are called the (design) variables, \( f(x) \) is the objective function, \( g_j(x) \) denotes the respective inequality constraint functions and \( h_j(x) \) the equality constraint functions. The optimum vector \( x \) that solves problem (1.1.1) is denoted by \( x^* \) with corresponding optimum function value \( f(x^*) \). If no constraints are specified, the problem is called an unconstrained minimization problem [32].

To solve the optimization problem in (1.1.1), efficient search or optimization algorithms are needed. There are many optimization algorithms which can be classified in many ways, depending on the focus and characteristics [33, 34]. For instance, in the design of integrated circuits, evolutionary algorithms have proven to be a viable option to optimize circuits, because they provide a set of feasible or optimal solutions, which the best meet to the desired specifications are selected, such as: improve the gain, increase bandwidth, response speed, reduced power consumption, minimizing noise and distortion, tolerance variations in the manufacturing process, etc. In the case of chaotic oscillators, evolutionary algorithms can be applied to calculate the optimum values of parameters of the chaotic oscillator beginning at the system level of abstraction, or mathematical model and then find the values of the circuit elements that better generates chaotic behavior.

Computing a high value of the positive or maximum Lyapunov exponent (MLE), requires varying the coefficients of the chaotic oscillator, leading to a huge number of
combinations that are potential solutions when applying meta-heuristics.

1.1.3. Synchronization

The synchronization can be seen as the property that have some objects of a different nature to express a uniform rate of co-existence, generally different from their individual pace and necessarily in the presence of a physical connection or coupling means, which, in most cases, is extremely weak.

It is said that two harmonic oscillators are synchronized if their periods are equal. For no harmonic oscillators, including chaotic oscillators, the concepts of frequency and phase are not well defined and therefore is said that two chaotic oscillators are synchronized if eventually after the transitional (a long or short time span) the oscillations coincide exactly at all times, despite starting both oscillators in different initial conditions.

Chaos synchronization is an important problem in nonlinear science. In the last few decades, synchronization has received a great interest among scientists from various fields [9, 11, 35]. The various types of chaos synchronization, whose description may require different theoretical frameworks were found in natural systems. The idea of synchronizing two identical chaotic systems from different initial conditions was introduced by Pecora [2]. Several schemes can be used to achieve such type of synchronization, for example Pecora and Carroll method [36], active control [37,38], adaptive control [39,40], via nonlinear control [41]. Identical synchronization has clear application to secure communication due to special features of chaotic systems, such as being extremely sensitive to tiny variations of initial conditions and system parameters.

The practical applications of chaos synchronization show the limitation of identical synchronization: parameter mismatch will probably destroy the manifold of this synchronization. Consequently, generalized synchronization was introduced [42, 43], which can give much richer dynamics than identical ones.

Chaos synchronization seemed an attractive approach for secure communication since its discovery [2, 3, 30, 44]. Due to inherent sensitivity, chaos modulation is less attractive than chaos encryption. The latter has to perform well in several aspects: design flexibility, good statistical properties, large encryption key space and sufficient resistance to known attacks [45,46].
1.1.4. Applications

Chaotic circuits, capable of generating chaotic oscillations from audio frequencies up to the optical band, are used as sources of chaotic carriers in a variety of applications including broadband communications, signal masking, chaos modulation, spectrum spreading, radar and cryptography of high entropy sources [35, 47–49]. Chaos-based secure communication is one of the most prominent applications of chaos theory in engineering. This field has particularly received a great deal of interest having regard to the amount of related literature starting from the pioneering work of Pecora and Carroll (1990) [2]. Chaotic signals have suitable characteristics for communications such as:

- Their ultra wide bandwidth.
- Their stability to fading in multi-path environment.
- Their possibility of control and synchronization.

These advantages promote the study of applying chaotic signals to modern wideband wireless communication systems. The basic chaos-based secure communication architecture involves a chaotic transmitter and a chaotic receiver linked together and exhibiting a synchronized motion. The problem to deal with consists in processing the received signal in order to reconstruct the message injected into the chaotic model at the transmitter. As it clearly appears, the chaotic generator stands as the key module for chaos-based secure communication architecture. Therefore, designing the chaotic system that will synchronize has remained one of the most important engineering and operational issues. Nowadays, modern radar systems have been regarded to be very “mature” in all the aspects like performance, manufacturing techniques and reliability, and have been widely applied in either military and civil uses. For the same reason, radar systems, using chaotic signals or pseudo-random signals, are also reported in the literature [50–53].

1.2. Integrated design of chaotic oscillators

Chaotic oscillators have been investigated to generate multi-scroll attractors. Some of them can be modeled by piecewise-linear (PWL) approaches, so that the nonlinear
problem can be transformed into a linear one. However, the goal of research in chaotic systems is, thus, to understand how a deterministic dynamical system might exhibit chaotic behavior, the kind of systems capable of this behavior, the ways available to control it, the ways to implement it with electronic devices, and the practical and theoretical implications that follow [16, 43, 54, 55].

Electronic implementations of chaotic oscillators are relatively new and in some cases there is still mathematical research to be done. Most of the potential applications exploits the deterministic nature of chaotic signals [16, 55], while others are related to the random behavior [56]. For example, the synchronization and transmission of encrypted information [57, 58], show advantages based on the complexity of chaotic signals, which turns them suitable for message encryption.

It was since nineties, that much research has been oriented to implement chaotic oscillators with electronic devices. A remarkable design of a continuous chaotic oscillator is the case of the two-scroll Chua’s circuit [59–63], which generates a rich variety of chaotic dynamics in a relatively simple electronic implementation. Recently, many new chaotic oscillators have been proposed [64–66]. In all cases a nonlinear part is required to obtain more equilibrium points than the origin and eventually obtaining attraction regions. Some approaches use polynomial forms, sinusoidal functions, delay-based functions, or piecewise-linear (PWL) functions [56, 67]. In particular, the PWL-based implementations show numerous research works because of their capacity to obtain at least partial analytical solutions (for the linear segments), while the obtention of such solutions for other nonlinearities are hard to reach [56]. This motivates the development of new PWL-based chaotic oscillators showing more scrolls (multi-scroll attractors), and more directions [58, 64, 65, 68, 69].

Currently, a collection of major developments of Chaotic Systems can be found in [56], where several researchers summarize key guidelines on modeling, simulation, control synchronization, and applications of chaotic oscillators. Besides, although chaotic oscillators have been intensively studied since a few decades ago, their implementation using electronic devices has been considered only in a very few works. Henceforth, this survey describes the electronic implementation of theoretical chaotic oscillators using discrete active devices and integrated circuit realizations.

The majority of electronic implementations of chaotic oscillators consist of discrete active devices, mainly operational amplifiers (opamps) [69], whereas other designs are implemented with mixed-mode active devices [70, 71], such as the current-feedback
opamp (CFOA [61,72], e.g. the Analog Devices AD844), operational transconductance amplifier (OTA) [66], positive-type second-generation current conveyor (CCII+) [57, 73], and unity-gain cells (UGCs) [63–65]. Relevant chaotic continuous-time oscillator designs are summarized in Table 1.1, where the following issues can be identified:

- The majority of those designs have been obtained by generalizing previous others, for example, the most preferred dynamic system is the Chua’s chaotic oscillator.

- PWL-based oscillators are preferred because of the relatively simple mathematical description, dynamical analysis and circuit synthesis.

- Implementations using integrated circuit technology are few in all cases and multi-scroll designs scarce.

As one can infer, many multi-scroll chaotic oscillators are derived from the double-scroll Chua’s chaotic circuit. Basically, the nonlinear resistor known as Chua’s diode [61] is augmented to have more break points while combining the slopes [72, 74], to generate even or odd number of scrolls. Other PWL functions can be added to implement chaotic oscillators in more than one direction (1D) [58,69], e.g. 2D [75], 3D [76], and 4D [77].

The electronic implementations of those chaotic oscillators based on PWL functions use different kinds of active devices; the most used one is the traditional opamp [16]. The CFOA AD844 from Analog Devices is also quite useful, and it can be used as CCII+ [73], as already demonstrated in [56,57]. The OTA has been used to implement double-scroll chaotic oscillators in [59,60]. Those three active devices have been summarized in [66] to show the generic topologies for realizing the PWL saturated function series. From the integrated circuit design point of view, the unity-gain cells (UGCs) can be interconnected or superimposed to derive mixed-mode active devices like CCIIIs and CFOAs [71]. Besides, the UGCs not only can be implemented with the CFOA AD844 [63,72], but also with integrated circuit technology [64,65].
<table>
<thead>
<tr>
<th>System Name</th>
<th>Attractor type</th>
<th>Nonlinear function based on</th>
<th>Proposed circuit</th>
<th>Integrated design</th>
<th>Reference</th>
<th>Author</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chua’s circuit</td>
<td>double-scroll</td>
<td>PWL</td>
<td>Opamp, OTA, CFOA, UGC, CCII+</td>
<td>OTA</td>
<td>Chua [79], Delgado [60]</td>
<td>1993</td>
<td></td>
</tr>
<tr>
<td>1D</td>
<td>PWL</td>
<td>Opamp</td>
<td>Trigonometric function generator</td>
<td></td>
<td>Suykens [80]</td>
<td>2000</td>
<td></td>
</tr>
<tr>
<td>1D</td>
<td>Sinusoidal</td>
<td>Trigonometric function generator</td>
<td></td>
<td></td>
<td>Tang [81]</td>
<td>2001</td>
<td></td>
</tr>
<tr>
<td>1D</td>
<td>PWL</td>
<td>Opamp</td>
<td>FGMOS</td>
<td></td>
<td>Zhong [82]</td>
<td>2002</td>
<td></td>
</tr>
<tr>
<td>1D, 2D</td>
<td>PWL (sawtooth)</td>
<td>Opamp</td>
<td>FGMOS</td>
<td></td>
<td>Fujiwara [83]</td>
<td>2003</td>
<td></td>
</tr>
<tr>
<td>multi-scroll</td>
<td>PWL</td>
<td>UGC’s</td>
<td>FGMOS</td>
<td></td>
<td>Trejo-Guerra [65]</td>
<td>2012</td>
<td></td>
</tr>
<tr>
<td>Lorenz</td>
<td>double-scroll</td>
<td>Product</td>
<td>Multiplier</td>
<td>OTA, Multiplier</td>
<td>González [85]</td>
<td>2000</td>
<td></td>
</tr>
<tr>
<td>multi-scroll</td>
<td>Complex</td>
<td>DSP</td>
<td></td>
<td></td>
<td>Yu [86]</td>
<td>2006</td>
<td></td>
</tr>
<tr>
<td>Third order canonical system</td>
<td>double scroll</td>
<td>PWL</td>
<td>CFOA, Opamp</td>
<td>OTA, Opamp</td>
<td>Elwakil [87]</td>
<td>2000</td>
<td></td>
</tr>
<tr>
<td>1D-3D</td>
<td>PWL</td>
<td>CFOA, Opamp</td>
<td></td>
<td></td>
<td>Yalcin [88]</td>
<td>2002</td>
<td></td>
</tr>
<tr>
<td>1D-3D</td>
<td>PWL</td>
<td>Opamp</td>
<td></td>
<td></td>
<td>Lü [54]</td>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>1D-3D</td>
<td>Hysteresis</td>
<td>Opamp, diode</td>
<td></td>
<td></td>
<td>Lü [16]</td>
<td>2006</td>
<td></td>
</tr>
<tr>
<td>Second order hyster.</td>
<td>1D, 2D</td>
<td>Hysteresis</td>
<td>Opamp, diode</td>
<td></td>
<td>Han [89]</td>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>NA</td>
<td>1D, 2D</td>
<td>$tanh()$</td>
<td>Differential pair, OTA</td>
<td></td>
<td>Ozoguz [90]</td>
<td>2002</td>
<td></td>
</tr>
<tr>
<td>NA</td>
<td>double-scroll</td>
<td>$tanh()$</td>
<td>LC</td>
<td></td>
<td>Ozoguz [91]</td>
<td>2005</td>
<td></td>
</tr>
<tr>
<td>NA</td>
<td>1D</td>
<td>PWL, $tanh()$, $t$</td>
<td>CFOA, TX line</td>
<td></td>
<td>Yalcin [92]</td>
<td>2007</td>
<td></td>
</tr>
<tr>
<td>NA</td>
<td>double scroll</td>
<td>PWL</td>
<td>current mirror, C</td>
<td></td>
<td>Ozoguz [93]</td>
<td>2008</td>
<td></td>
</tr>
</tbody>
</table>
1.3. Thesis objectives

The purpose of this research is to generate scientific knowledge exploring and studying chaotic oscillators, determining the type of systems capable of generating this behavior, the conditions required for their generation, optimal system parameters, and ways of implementation with electronic devices, and the theoretical and practical implications that can be obtained to propose design guidelines.

In the design of integrated circuits, evolutionary algorithms have proven to be a viable option to optimize circuits, because they provide a set of feasible or optimal solutions, which best meet the desired specifications. In the case of chaotic oscillators, evolutionary algorithms can be applied to calculate the optimum values of parameters of the chaotic oscillator beginning from the system level down to the circuit level or even the integrated design finding feasible solutions for the transistor sizes.

The importance of continuous chaotic oscillators is principally lying in the potential applications been still on research. The need of a CMOS integrated multi-scroll design arises with the appearance of multi-scroll oscillators, and the very limited amount of integrated realizations. A key point is that since these realizations are few, there is not enough criteria for selecting the design requirements for nonlinear systems. Also, the performance of such designs based on non-traditional active devices is unknown at integrated circuit design level.

1.3.1. General goal

Synchronization of two optimized chaotic oscillators coupled in a master-slave configuration and implement the system using CMOS integrated circuit technology.

1.3.2. Specific objectives

- Determining the values of the parameters of the chaotic system with which a more complex chaotic behavior occurs, e.g. evaluating the Lyapunov exponents optimizing MLE, and providing uniform phase diagrams distribution between different states.

- Explore and determine the dynamic behavior of the system when varying the characteristics of non-linear functions or disturbing, considering actual opera-
Proposing the design of active devices, using evolutionary algorithms.

- Propose the behavioral model of active devices, such as: operational amplifier (Opamp), opamp with current feedback (CFOA), transconductance opamp (OTA), and current conveyor.

- Propose a technical adjustment to the models of active devices to optimize the operation of the chaotic oscillator more often generating the highest number of scrolls, and to increase the complexity of chaotic behavior.

- Design the chaotic oscillator with optimized MLE using CMOS integrated circuit technology.

- Synchronizing two chaotic oscillators optimized in their MLE and connected in a master-slave configuration.
2.1. Dynamical systems

Dynamical systems have internal parameters (state variables) that can change over time in a way that at least in principle, is always predictable that external influences on this are known. Are called systems because they are described by a set of equations; and dynamical because their parameters vary with respect to any variable that is usually the time [21]. This Thesis show the integrated design of chaotic oscillators with optimized MLE and the synchronization of two of them, however, it takes no account of statistical properties [22].

2.1.1. State variables model

In order to make any quantitative progress in understanding a system, a mathematical model is required. The model may be formulated in many ways, but their essential feature allows us to predict the behavior of the system, sometimes by given its initial conditions and a knowledge of the external forces which affect it. In electronics, the mathematical description for a dynamical system, most naturally adopted for behavioral modeling, is done by using the so-called state-space representation, which basically consists of a set of differential equations describing the evolution of the variables whose values at any given instant determine the current state of the system. These are known as the state variables and their values at any particular time are supposed to contain sufficient information for the future evolution of the system to be predicted, given that the external influences (or input variables) which act upon it are known.

In the state-space approach, the differential equations are of first order in the
time-derivative, so that the initial values of the variables will suffice to determine the solution. In general, the state-space description is given in the form:

\[
\begin{align*}
\dot{x} &= f(x, u, t) \\
y &= h(x, u, t)
\end{align*}
\] (2.1.1)

where the dot denotes differentiation with respect to time \((t)\) and the functions \(f\) and \(h\) are in general nonlinear. In (2.1.1), the variety of possible nonlinearities is infinite, but it may nevertheless be worthwhile to classify them into some general categories. For example: There are simple analytic functions such as powers, sinusoids and exponentials of a single variable, or products of different variables. A significant feature of these functions is that they are smooth enough to possess convergent Taylor expansions at all points and consequently can be linearized [94]. A type of nonlinear function frequently used in system modeling is the piecewise-linear (PWL) approximation, which consists of a set of linear relations valid in different regions [14, 54, 72, 73, 88, 95–97]. The use of PWL approximations have the advantage that the dynamical equations become linear or linearized in any particular region, and hence the solutions for different regions can be joined together at the boundaries.

When applying PWL approximation to a system described by (2.1.1), the resulting linearized system has finite dimensional state-space representation, as a result the equations describing a linear behavioral model become:

\[
\begin{align*}
\dot{x} &= Ax + Bu \\
y &= Cx + Du
\end{align*}
\] (2.1.2)

where \(A, B, C,\) and \(D\) are matrices (possibly time-dependent) of appropriate dimensions. The great advantage of linearity is that, even in the time dependent case, a formal solution can immediately be constructed, which is moreover applicable for all initial conditions and all input functions.

An important point which must be kept in mind for a nonlinear dynamical system, is that the stability properties are essentially more complicated than in the linear case. For instance, when nonlinearities are present, several features can appear such as limit cycles or the phenomenon known as chaos [11, 19, 21, 69, 98, 99].
2.1.2. Autonomous systems

Although the equations describing a behavioral dynamical model will in general depend on the time, either explicitly or through the input function, or both, a large part of nonlinear system theory is concerned with cases where there is no time dependence at all [21]. Such systems are said to be autonomous, and they arise quite naturally in practice when, for example, the input vector is held fixed. In any such case, the differential equation for the state vector will become:

\[ \dot{x} = f(x, \hat{u}) \]  

(2.1.3)

where \( \hat{u} \) is a constant vector. Thus, the equilibrium points in the state-space are determined by \( f(x, \hat{u}) = 0 \). Assuming that \( f(x, \hat{u}) \) satisfies Lipschitz condition [99], the differential equation for \( x(t) \) will have a unique solution, for any given initial state \( x(0) \). The path traced out in the state-space by \( x(t) \) is called a trajectory of the system and because of the uniqueness property, there will be one and only one trajectory passing through any given point. If it is suppressed the dependence on \( \hat{u} \), the state-space differential equations for an autonomous system can be written simply as:

\[ \dot{x} = f(x) \]  

(2.1.4)

and the set of all trajectories of this equation provides a complete geometrical representation of the dynamical behavior of the system, under the specified conditions. As a result, it is possible to give an essentially complete classification of behavior in the phase plane, though not in higher-dimensional state-spaces. In general, the equations describing a nonlinear system cannot be solved analytically, so that, in order to construct the trajectories accurately, it is necessary to use numerical methods.

2.2. Geometry and stability of dynamical systems

The way to visualize the behavior of the state variables of a dynamical system can be in the form of time series (graph of a state variable versus time), or in a phase space portrait. The phase space portrait of an n-dimensional system \( \dot{x} = f(x) \) is the space where all possible states of a system are represented, each system parameter is
represented as an axis of a multidimensional space and each point in space representing each possible state of the system variables. In this type of representation time becomes an implicit parameter; the example shown in Figure 2.1 a time series and a flat phase of a dynamical system.

![Image](a) Time serie and (b) Phase space portrait.

Figure 2.1: Graphic representation: (a) Time serie and (b) Phase space portrait.

The phase space portrait is described by a vector field $\mathbf{F}$ which governs the path of the system variables $\mathbf{x}(t)$ in time, the path of these variables is called a path. Figure 2.2 shows the vector field in the phase space portrait of a dynamical system, it can be seen singularities (points, cycles or subsets of the phase space) that draw paths that pass near them and others that repel.

![Image](Vectorial field of a nonlinear dynamical system.)

Figure 2.2: Vectorial field of a nonlinear dynamical system.

It is said that a singularity of the phase space is stable attractor sink or if any path that starts near it approaches it as time passes. In fact if the region attracts all trajectories of phase space called the global attractor. Furthermore, a singularity of
the phase space is Lyapunov-stable if all paths starting close enough to it are kept close to the latter at all times. It may be a situation where the phase singularity is Lyapunov-stable attractor space but if this happens is said to be neutrally stable. However, usually two types of stability occur simultaneously, and if the singularity is said to be asymptotically stable. Finally, a singularity is unstable, repulsive or source when it is neither attractor or Lyapunov-stable, ie starting paths diverge near it as time passes. The importance of stability of singularities is that it determines the stability of the system in which the singularities occur. In linear systems, singularities can only be points, which are known as fixed points; instead nonlinear systems may have fixed points, limit cycles and regions called strange or chaotic attractors.

### 2.2.1. Equilibrium points

The equilibrium points of an autonomous system given by (2.1.4) are also known as singular points when \( f(\dot{x}) = 0 \); because they appear to violate the general rule that only the trajectory can pass trough any given point [11]. Actually, the violation is only apparent, since the trajectories which meet a singular point do not really pass through it, but only approach or depart from it asymptotically [19]. Assuming that \( f(x) \) is smooth enough for the equations to be linearized around the singular point \( \hat{x} \), the approximation will be sufficient to determinate the behaviour of the trajectories in the neighborhood of the equilibrium point. If \( A \) in (2) is nonsingular, the nature of the equilibrium point is essentially determined by its eigenvalues and can be classified by stable node, stable focus, unstable node, unstable focus, centre and saddle point [11,19–22], as illustrate in Figure 2.3.

If you have a linear dynamical system \( \dot{x} = Ax \), it is easy to know what kind of equilibrium point has the system; simply eigenvalues \( \lambda \) are calculated from the system characteristic equation \( \det(A - \lambda I) = 0 \) and the relationship between the eigenvalues is discussed. Furthermore, analysis of fixed points of a nonlinear system \( \dot{x} = f(x) \) can be done by linearizing the system around each of the fixed points and analyzing each separately linearized system [99].

### 2.2.2. Limit cycles

A common feature of autonomous systems is the occurrence of a special type of trajectory which takes the form of a closed curve. This is known as a limit cycle and...
represents a periodic solution of the system equation since, when the state vector returns to the initial value, it must repeat its previous motion and so continues indefinitely [22]. Limit cycles can occur in systems of any order, and indeed constitute the typical form of oscillatory behavior which arises when an equilibrium point of a nonlinear system becomes unstable according to the general condition for the existence of limit cycles defined by the Poincaré-Bendixon theorem [22]. Another concept also due to Poincaré, which is relevant to the occurrence of limit cycles in the phase plane, is the index of a close curve [20]. If the curve is simple, that is to say, it does not intersect itself, its index with respect to the vector function $f(x)$ is defined as the net total number of clockwise revolutions made by $f$ as $x$ traverses the curve once in the clockwise sense. Furthermore, it implies that the index of the curve can be computed by summing the contributions of the singular points which it surrounds, assuming they are isolated, where each node, focus, or centre counts $+1$, and each saddle point counts $-1$ [100]. Since the index of a limit cycle is clearly $+1$, this restricts its possible location with respect to the equilibrium points of the system. A typical case [22], with a stable limit cycle surrounding an unstable focus, is illustrated in Figure 2.4.
2.2.3. Strange attractors and chaos

Although singular points and closed curves constitute the only asymptotic terms of bounded trajectories for autonomous systems in the phase plane, this is no longer true in spaces of higher dimension [19]. In general, the term for a limit set where all trajectories in its vicinity approach it as $t \to \infty$ is an attractor, since it asymptotically attracts nearby trajectories to itself. For second-order dynamical systems, the only types of limit set normally encountered are singular points and limit cycles. Consequently, a continuous-time autonomous system requires more than two dimensions to exhibit chaos [11, 19, 21]. More complicated still are the so-called strange limit sets [11]. They may or may not be asymptotically attractive to neighboring trajectories; if so, they are known as strange attractors, though even then, the trajectories they contain may be locally divergent from each other, within the attracting set. Such structures are associated with the quasi-random behavior of solutions called chaos [19].

2.3. Multi-scrolls chaotic oscillators

Relevant chaotic continuous-time oscillator designs were summarized in Table 1.1. As one can see, the design of multi-scroll chaotic attractors can be performed via PWL functions [14, 54, 69, 72, 73, 88, 95–97, 101]. In the case of Chua’s circuit based oscillators, the PWL function is designed by introducing additional breakpoints to Chua’s diode [72], or by generalizing Chua’s circuit as proposed in [14, 102]. Furthermore, among the basic circuits used to generate multi-scroll generators, the step circuit, hysteresis circuit and saturated circuit are the three most used ones. In [54] a saturated multi-scroll chaotic system based on saturated function series, is introduced. That system can produce three different types of attractors, as follows: 1-D saturated $n$-scroll chaotic attractors, 2-D saturated $n \times m$-grid scroll chaotic attractors and 3-D saturated $n \times m \times l$-grid scroll chaotic attractors. Since this multi-scroll system can be designed...
by using PWL functions, it is a good chaos system suitable for the development of a systematic design automation process by applying behavioral modeling [69].

When implementing multi-scroll chaos generators with electronic devices, it is necessary to remark that it is quite difficult to generate attractors with a large number of scrolls due to the limitation of the real dynamical range of the physical devices.

2.3.1. Chua’s circuit generalization

A generalization of the Chua’s system described in [103] is chosen. The dynamical system is described by (2.3.1) with $\alpha = 10, \beta = 11.5$ and a given non-linear function (2.3.2).

\[
\begin{align*}
\dot{x} &= \alpha(y - x - f(x)) \\
\dot{y} &= x - y + z \\
\dot{z} &= -\beta y \tag{2.3.1}
\end{align*}
\]

\[
f(x) = m_{2n-1}x + \frac{1}{2} \sum_{i=q}^{2n-1} (m_{i-1} - m_i) (|x + b_i| - |x - b_i|) \tag{2.3.2}
\]

where $q = 1$ to generate $2n$ even scrolls, and $q = 2$ to generate $2n - 1$ odd scrolls. The vector $m$ represents slopes, while the vector $b$ shows the break points. For a number of even scrolls $m$ is taken as $[m_0 \ldots m_9] = [-4.416, -0.276, -3.036, -0.276, -3.036, -0.276, -3.036, -0.276, -3.036, -0.276]$ and $[b_1 \ldots b_9] = [0.1, 1.1, 1.55, 3.2, 3.85, 5.84, 6.6, 8.7, 9.45]$; whereas for an odd number of scrolls $[b_2 \ldots b_9] = [0.8, 1.4, 3.2, 3.9, 5.8, 6.4, 8.3, 9.2]$ is used.

![Figure 2.5: Nonlinear function.](image)
Dynamical analysis

This section discusses some of the dynamical and geometric properties of behavior system given in (2.3.1). First, the dynamics is considered in each PWL region and then observe the dynamics in the state space.

The system (2.3.1) presents chaotic behavior for a given set of parameters. Since the nonlinearity of Chua’s circuit is a piecewise linear (PWL) function, the circuit can be divided into a number of different related regions. Analyzing the behavior of the system in each of these regions is very useful to understand the overall behavior of the circuit.

In particular, for the case of generating 3 scrolls for $m_3, \ldots, m_7$ the circuit can be broken down into five distinct regions related:

$$
D_2 = \{(x, y, z)|x \geq E_2\}
$$

$$
D_1 = \{(x, y, z)|x \geq E_1\}
$$

$$
D_0 = \{(x, y, z)|-E_1 \leq x \leq E_1\}
$$

$$
D_{-1} = \{(x, y, z)|x \leq -E_1\}
$$

$$
D_{-2} = \{(x, y, z)|x \leq -E_2\}
$$

First the equilibrium points (EP) are given by:

$$
f(x) = 0
$$

$$
x = -z
$$

$$
y = 0
$$

In each of the five regions, $D_2, \ldots, D_{-2}$, the system has a unique equilibrium point given by the following equations:
\[ EP^+_2 = (k_1, 0, -k_1) \in D_2 \]
\[ EP^+_1 = (k, 0, -k) \in D_1 \]
\[ EP_0 = (0, 0, 0) \in D_0 \]
\[ EP^-_1 = (-k, 0, k) \in D_{-1} \]
\[ EP^-_2 = (-k_1, 0, k_1) \in D_{-2} \]

where \( k = (g_2 - g_1)E_1/(g_2 + 1) \) and \( k_1 = (g_2 - g_1)E_1 + (g_1 - g_2)E_2/(g_1 + 1) \), with \( g_1 = m_3 = m_5 = m_7 \) y \( g_2 = m_4 = m_6 \) according to (2.3.2) and Figure 2.5.

**Stability of equilibrium points**

In each of the regions \( D_2, \ldots, D_{-2} \), the Chua’s circuit equations (2.3.1) are linear and can be expressed in the form \( \dot{x} = Ax + b \), as:

\[
\dot{x} = \begin{cases} 
A(\alpha, \beta, m_3)(x - k_1) & x \in D_2 \\
A(\alpha, \beta, m_4)(x - k) & x \in D_1 \\
A(\alpha, \beta, m_5)x & x \in D_0 \\
A(\alpha, \beta, m_6)(x + k) & x \in D_{-1} \\
A(\alpha, \beta, m_7)(x + k_1) & x \in D_{-2} 
\end{cases} \tag{2.3.3}
\]

where \( x = [x, y, z]^T \), \( k = [k, 0, -k]^T \), \( k_1 = [k_1, 0, -k_1]^T \) and

\[
\begin{bmatrix} A(\alpha, \beta, m_i) \end{bmatrix} = \begin{bmatrix} -\alpha(m_i + 1) & \alpha & 0 \\
1 & -1 & 1 \\
0 & -\beta & 0 \end{bmatrix} \tag{2.3.4}
\]

whose eigenvalues are given by:

\[
\lambda = \lambda^3 + [1 + \alpha(m_i + 1)]\lambda^2 + [\beta + \alpha m_i] \lambda + \alpha \beta(1 + m_i) \tag{2.3.5}
\]

with \( m = g_1 \) for regions \( D_2, D_0, D_{-2} \) and \( m = g_2 \) for regions \( D_1, D_{-1} \).
The behavior in each respective region \( m_i \) is numerically analyzed using \( \alpha = 10, \beta = 11.5, g_1 = -0.276 \) and \( g_2 = -3.036 \) according to (2.3.2) and Figure 2.5.

Equilibrium points numerically evaluated are given by:

\[
EP_2^+ = (2.2873, 0, -2.2873) \in D_2
\]

\[
EP_1^+ = (1.0845, 0, -1.0845) \in D_1
\]

\[
EP_0 = (0, 0, 0) \in D_0
\]

\[
EP_1^- = (-1.0845, 0, 1.0845) \in D_{-1}
\]

\[
EP_2^- = (-2.2873, 0, 2.2873) \in D_{-2}
\]

For \( m_i = g_1 \) (regions \( D_2, D_0, D_{-2} \)) the Jacobian matrix of the system (2.3.1) in the equilibrium points \((EP_2^+, 0, EP_2^-)\) for the corresponding slope are obtained as:

\[
\left[ J(EP_2^+, 0) \right] = \left[ \frac{\partial F}{\partial x} \right] = \begin{bmatrix}
-7.2400 & 10.0000 & 0 \\
1.0000 & -1.0000 & 1.0000 \\
0 & -11.5000 & 0
\end{bmatrix}
\]

(2.3.6)

which have the following eigenvalues:

\[
\lambda_1^{EP_{2,0}} = -8.3823
\]

\[
\lambda_2^{EP_{2,0}} = 0.0712 + 3.1508i
\]

\[
\lambda_3^{EP_{2,0}} = 0.0712 - 3.1508i
\]

While for \( m = g_2 \) (regions \( D_1, D_{-1} \)) the Jacobian matrix of the system (2.3.1) in the equilibrium points \((EP_1^+, EP_1^-)\) for the corresponding slope are obtained as:

\[
\left[ J(EP_1^+) \right] = \left[ \frac{\partial F}{\partial x} \right] = \begin{bmatrix}
20.3600 & 10.0000 & 0 \\
1.0000 & -1.0000 & 1.0000 \\
0 & -11.5000 & 0
\end{bmatrix}
\]

(2.3.7)

which have the following eigenvalues:
\[ \lambda_1^{EP\pm1} = 20.8072 \]
\[ \lambda_2^{EP\pm1} = -0.7236 + 3.2755i \]
\[ \lambda_3^{EP\pm1} = -0.7236 - 3.2755i \]

Of the values calculated, these can be classified as a saddle point of index 2 and saddle point of index 1, respectively.

Numerical simulation using MATLAB and pack *ode45* for integrating the system of equations was performed for (2.3.1). Figure 2.6 shows the behavior of the nonlinear function and the phase diagram of the system for a 3-scrolls attractor using as initial conditions \( X = [0.1, 0, 0] \).

![Nonlinear function and phase space portrait](image-url)

**Figure 2.6:** 3-scrolls Chua’s circuit generalization.

### 2.3.2. Chaotic oscillator based on saturated functions series

Another attractor of n-scrolls pretty interesting and practical is based on series of saturated functions. This can be extended to generate multidimensional attractors [16, 54, 95, 104].

The equations of state variables of the system are shown in (2.3.8) [16], where \( a, b, c, d_1 \) are positive constants, and \( f(x; k, h, p, q) \) is defined by the PWL function given in (2.3.9).
\[ \begin{align*}
\dot{x} &= y \\
\dot{y} &= z \\
\dot{z} &= -ax - by - cz + d_1f(x; k, h, p, q) 
\end{align*} \] (2.3.8)

\[ f(x_1; k, h, p, q) = \sum_{i=-p}^{q} f_i(x_1; k, h) \] (2.3.9)

Equation (2.3.9), is called a saturated function series, where \( k > 0 \) is the slope of saturated function series, \( h > 2 \) is the saturated delay time of the saturated function series, \( p \) and \( q \) are positive integers, and

\[ f_i(x_1; k, h) = \begin{cases} 
2k, & \text{if } x_1 > ih + 1 \\
k(x_1 - ih) + k & \text{if } |x_1 - ih| \leq 1 \\
0, & \text{if } x_1 < ih - 1 
\end{cases} \] (2.3.10)

and

\[ f_{-i}(x_1; k, h) = \begin{cases} 
0, & \text{if } x_1 > -ih + 1 \\
k(x + ih) - k & \text{if } |x_1 + ih| \leq 1 \\
-2k, & \text{if } x_1 < -ih - 1 
\end{cases} \] (2.3.11)

One may recast the saturated function series \( f(x_1; \alpha, k, h, p, q) \) as follows:

\[ f(x_1; \alpha, k, h, p, q) = \begin{cases} 
(2q + 1)k & \text{if } x > qh + \alpha \\
k\alpha(x - ih) + 2ik & \text{if } |x - ih| \leq \alpha \\
-2p & \text{if } -p \leq i \leq q \\
(2i + 1)k & \text{if } ih + \alpha < x < (i + 1)h - \alpha \\
-2p & \text{if } -p \leq i \leq q - 1 \\
(2p + 1)k & \text{if } x < -ph - \alpha 
\end{cases} \] (2.3.12)

where the new slope \( k_n = \frac{k}{\alpha} \) allows that \( k \) and \( \alpha \) can be selected to allow \( \alpha < 1 \) and \( s \geq 2 \).

Figure 2.7 shows two saturated functions (SNLF), with 5 and 7 segments, respectively. These functions are composed of two segments. Segments without slope are
called saturated regions and segments with slope are called saturation slopes. Thus, the number of scrolls that these saturated functions can generate depends on the number of saturated regions.

![Diagram of saturated regions and segments with slope](image)

Figure 2.7: SNLF with 5 and 7 segments.

Saturated regions in (2.3.12) are in $\pm nk$, where $n$ is an odd integer to generate even scrolls or $n$ is an even integer to generate odd scrolls. $h$ is the saturation delay the center of the slopes, and must agree to $h_i = \pm mk$, where $i = 1, \ldots, [(s-2)/2]$ and $m = 2, 4, \ldots, (s-2)$ to generate even scrolls; $i = 1, \ldots, [(s-1)/2]$ and $m = 1, 3, \ldots, (s-2)$ to generate odd scrolls; $p$ and $q$ are positive integers.

In particular, for the case to generate two scrolls, the system can be decomposed into three distinct regions related given by (2.3.13) with $a = b = c = d_1 = 0.7$ and $k = 1$, $\alpha = 0.1$, $h = p = q = 0$ in (2.3.12).

\[
D_1 = \{(x, y, z)|x \geq 1\} \\
D_0 = \{(x, y, z)||x| \leq 1\} \\
D_{-1} = \{(x, y, z)|x \leq -1\}
\]

The equilibrium points of the system are given by:

\[
y = 0 \\
z = 0 \\
f(x) = 0
\]
The system has three equilibrium points, which correspond to three linear regions sections:

\[ EP_1^+ = \left( \frac{k_n d_1}{a}, 0, 0 \right) \in D_1 \]
\[ EP_0 = (0, 0, 0) \in D_0 \]
\[ EP_1^- = \left( -\frac{k_n d_1}{a}, 0, 0 \right) \in D_{-1} \]

**Stability of equilibrium points**

In each of the equilibrium points of the piecewise linear regions, the values are given by:

\[
\begin{bmatrix}
J(EP_1^+, 0) = \left[ \frac{\partial F}{\partial x} \right] = \\
0 & 1 & 0 \\
0 & 0 & 1.0000 \\
-0.7 & -0.7 & -0.7
\end{bmatrix}
\]

which have the following values:

\[
\lambda_1^{EP_1^\pm} = -0.8480 \\
\lambda_2^{EP_1^\pm} = 0.0740 + 0.9055i \\
\lambda_3^{EP_1^\pm} = 0.0740 - 0.9055i
\]

While for the equilibrium point \( EP_0 \)

\[
\begin{bmatrix}
J(EP_0) = \left[ \frac{\partial F}{\partial x} \right] = \\
0 & 1 & 0 \\
0 & 0 & 1 \\
6.3 & -0.7 & -0.7
\end{bmatrix}
\]

which have the following eigenvalues:
2. Chaotic systems

\[ \lambda_{1}^{EP_{0}} = 1.5309 \]
\[ \lambda_{2}^{EP_{0}} = -1.1154 + 1.6944i \]
\[ \lambda_{3}^{EP_{0}} = -1.1154 + 1.6944i \]

In general, every \(2(p + q) + 3\) equilibrium points of the system (2.3.8) are located along the \(x\) axis, and can be classified into two different groups:

\[ A_x = -\frac{(2p+1)d_1k_n}{a}, \frac{(-2p+1)d_1k_n}{a}, \ldots, \frac{(2q+1)d_1k_n}{a} \]
(2.3.15)

and

\[ B_x = \frac{pk_n d_1 (h - 2)}{k_n d_1 - a}, \frac{(-p+1)k_n d_1 (h - 2)}{k_n d_1 - a}, \ldots, \frac{qk_n d_1 (h - 2)}{k_n d_1 - a} \]
(2.3.16)

where \(k_n = \frac{k}{\alpha}\)

For all equilibrium points in the set \(A_x\), the characteristic equations are

\[ \lambda^3 + c\lambda^2 + b\lambda + a = 0 \]  
(2.3.17)

and the corresponding eigenvalues satisfy \(\lambda_1 < 0\) and \(\lambda_{2,3} = \alpha \pm \beta i\) with \(\alpha > 0\) and \(\beta \neq 0\). This is all the equilibrium points in set \(A_x\) are saddle points of index 2.

For all equilibrium points in the set \(B_x\), the characteristic equations are

\[ \lambda^3 + c\lambda^2 + b\lambda + a - d_1 k_n = 0 \]  
(2.3.18)

Solving the system (2.3.18) has one positive eigenvalue and two negative eigenvalues, or one positive eigenvalue and a pair of complex conjugate eigenvalues with negative real parts. To generate chaos from system (2.3.8), one may assume that (2.3.18) has a positive eigenvalue and a pair of complex eigenvalues with negative real parts. It means that all equilibria in set \(B_x\) are saddle points of index 1.

For the generation of a 4-scrolls chaotic oscillator, the saturated nonlinear function is shown in Figure 2.8a. Evaluating (2.3.8) and (2.3.12), by selecting \(a = b = c = d_1 = 0.7, k = 1, \alpha = 0.1, k_n = 10, h = 1, p = q = 1\), a chaotic attractor with 4-scrolls is generated as shown in Figure 2.8b.
2.4. Design of multi-scroll chaos generators from behavioral modeling

Behavioral modeling can be used to verify the successful development of electronic systems. In particular, analog electronic design automation (EDA) tools are gaining appreciation from the community of electronic design because various types of systems can be represented by means of an abstract model. In electronic design the abstraction levels are an indication of the degree of detail specified about how the function is to be implemented. Therefore, behavioral models try to capture as much circuit functionality as possible with far less implementation details than the device-level description of the circuit [105]. However, it is difficult to make a strict distinction between different abstraction levels for analog systems, in contrast to common practice in digital synthesis methodologies [106]. Instead, a division should be made between a description level and an abstraction level. A description level is a pair of two sets: a set of elementary elements and a set of interconnection types. The abstraction level of a description is the degree to which information about non-ideal effects or structure is neglected compared to the dominant behavior of the entire system.

In this manner, whereas a description level indicates how the analog system is represented; an abstraction level deals with the relation between the model of the system and its real behavior. Although it can be clear to consider the functional
level at a high abstraction level and the physical level at a low abstraction level, it is not straightforward to compare the abstraction levels of different description levels. Besides, an electronic system can be designed by converting the functional specification at the highest abstraction level to a physical realization at the lowest abstraction level via operations between description and abstraction levels.

In [69] it is presented the analog design automation process of chaotic systems. In that work, the chaotic systems can be modeled from the highest level of abstraction by applying state variables approach and PWL approximations.

2.4.1. Multi-scroll chaotic oscillator based on saturated non-linear functions

A multi-scroll chaotic oscillator can be described by the system of differential equations given in (2.4.1) [16, 54], where $a, b, c,$ and $d_1$ are positive constants, and can have values in the interval $[0, 1]$. The system is controlled by the PWL approximation, e.g. a saturated function series $f$,

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= x_3 \\
\dot{x}_3 &= -ax_1 - bx_2 - cx_3 + d_1 f(x_1; m)
\end{align*}
\]  

(2.4.1)

Now, it will be described how the saturated function $f$ in (2.4.1) is obtained in detail. Let $f_0$ be the saturated function:

\[
f_0(x_1; m) = \begin{cases} 
1, & \text{if } x_1 > m \\
\frac{a}{m}, & \text{if } |x_1| \leq m \\
-1, & \text{if } x_1 < -m
\end{cases}
\]  

(2.4.2)

where $\frac{1}{m}$ is the slope of the middle segment and $m > 0$. The upper radial $\{f_0(x_1; m) = 1|x_1 > m\}$, and the lower radial $\{f_0(x_1; m) = -1|x_1 < m\}$ are called saturated plateaus, and the segment $\{f_0(x_1; m) = \frac{a}{m}||x_1| \leq m\}$ between the two saturated plateaus is called saturated slope [69]. Let us consider also the saturated functions $f_h$ and $f_{-h}$ defined as:
2.4 Design of multi-scroll chaos generators from behavioral modeling

\[ f_h(x_1; m, h) = \begin{cases} 2, & \text{if } x_1 > h + m \\ \frac{x - h}{m}, & \text{if } |x_1 - h| \leq m \\ 0, & \text{if } x_1 < h - m \end{cases} \]  

(2.4.3)

and

\[ f_{-h}(x_1; m, -h) = \begin{cases} 0, & \text{if } x_1 > h + m \\ \frac{x - h}{m}, & \text{if } |x_1 - h| \leq m \\ -2, & \text{if } x_1 < h - m \end{cases} \]  

(2.4.4)

where \( h \) is called the saturated delay time and \( h > m \). Therefore, a saturated function series for a chaotic oscillator with \( s \) scrolls is defined as the function:

\[ f(x_1; m) = \sum_{i=0}^{s-2} f_{2i-s+2}(x_1; m, 2i - s + 2), \]  

(2.4.5)

where \( s > 2 \).

For example, using \( f = f_0 \) in (2.4.1), a 2-scroll chaotic attractor can be generated. Therefore, the saturated function series to generate 3 scrolls is \( f(x_1; m) = f_{-1}(x_1; m, -1) + f_1(x_1; m, 1) \). To generate 4-scroll it will be \( f(x_1; m) = f_{-2}(x_1; m, -2) + f_0(x_1; m) + f_2(x_1; m, 2) \), and so on. These function series are shown in Fig. 2.9 for \( m = 0.1 \). Note that the value of \( h \) in (2.4.3) and (2.4.4) represents the center of the saturated slopes.

Fig. 2.10 shows the simulation of 2 to 7-scroll chaotic oscillator attractors modeled by (2.4.1).
2. Chaotic systems

2.4.2. Behavioral modeling of SNLF

Existing models of continuous nonlinear functions used for generating multi-scroll chaotic attractors are based on a piecewise linear (PWL) approach [68]. These models, although relatively easy to build, do not include any information related to the performance parameters of active devices, in the context of a possible physical implementation. This is a serious drawback, since the use of a PWL model introduces...
a level of inaccuracy into a numerical analysis which is more evident when numerical and experimental results are compared. In this way, a chaotic system can numerically be analyzed and common metrics used to measure the complexity of the nonlinear system can be applied, like the Lyapunov exponent spectrum [67].

In brief, to gain insight on the behavior of the chaotic system, numerical simulations are often done and a PWL function is usually adopted as nonlinear function. In practice, however, the real behavior of the synthesized chaotic oscillator is far different from numerical simulations and these differences are more evident when the chaotic oscillator is pushed to operate at high frequency.

In electronics, the SNLF is often designed by stacking several high-gain voltage amplifiers and each of them is working in either saturation or linear region, which depends of the amplitude level of the input signal [54, 107–109]. Furthermore, the behavior of SNLF not only depends on the performance parameters of the active devices, but also on the operating frequency and on the nature of the excitation signal [109]. Therefore at low frequency, the behavior of SNLF can accurately be modeled by using a PWL approach, however, if the frequency increases, the nonlinear dynamics is more evident and the PWL approach cannot be used without significant precision loss. This is because a PWL model is only able to model a static behavior instead of a dynamical behavior.

Comparisons between the PWL model proposed in [109] and the real behavior of SNLF designed with OpAmps [108] are shown in Figure 2.11, where a chaotic waveform is applied as excitation signal [110]. According to Figure 2.11, one can infer that all PWL models proposed in [54, 72, 73, 84, 88, 107–109] can only be used at low frequency.

2.5. Circuit implementation

The dynamical system in (2.3.8) has the block diagram representation shown in Figure 2.12, which is realized with 3 integrators and an adder. Each block can be realized with different kinds of active devices, namely: OpAmps, CFOAs, unity-gain-cells (UGCs), OTA’s, current conveyors (CC) and so on.
2. Chaotic systems

Figure 2.11: PWL-based SNLF (blue line) and its real behavior using Op Amps and simulated with Hspice (black line) working at (a) 100 Hz and (b) 18 kHz.

![Block diagram](image)

Figure 2.12: Block diagram description of (2.3.8).

2.5.1. OpAmp implementation

The realization of the dynamical system in (2.3.8) using OpAmps is shown in Figure 2.18.

By applying Kirchhoff’s current law in Figure 2.18, equations given in (2.5.1) are obtained; where $SNLF = f(x_1)$. 

---

Department of Electronics National Institute of Astrophysics, Optics and Electronics
2.5 Circuit implementation

A voltage saturated function is synthesized using the opamp finite-gain model shown in Figure 2.14 to include its real limitations like gain, bandwidth, slew rate and saturation [110]. Therefore, if a shift-voltage ($\pm E$) is added, one gets the shifted-voltage determined by (2.5.3) for positive and negative shifts, respectively, as shown by Figure 2.15. The SNLF can be implemented as shown in Figure 2.16, where the number of opamps equals the number of scrolls to be generated minus 1. To generate a SNLF $E$ takes different values in (2.5.3) to synthesize the required plateaus and slopes. The value of the plateaus $k$, in voltage and current, the breakpoints $\alpha$, the slope $s$ and the saturated delays $h$ are evaluated by (2.5.2).

\[
\begin{align*}
  \dot{x}_1 &= \frac{x_2}{RC} \\
  \dot{x}_2 &= \frac{x_3}{RC} \\
  \dot{x}_3 &= -\frac{R_{fa} x_1}{R_{ia} C} - \frac{R_{fb} x_2}{R_{ib} C} - \frac{R_{fc} x_3}{R_{ie} C} + \frac{R_{fd} f(x_1)}{R_{id} C}
\end{align*}
\]  

\[
k = R_{iz} I_{sat}, \quad I_{sat} = \frac{V_{sat}}{R_c}, \quad \alpha = \frac{R_{iz} |V_{sat}|}{R_{iz}}, \quad h = \frac{E_i}{(1 + \frac{R_{iz}}{R_{iz}})}
\]  

Figure 2.13: OpAmp-based implementation of (2.3.8).
Simulating (2.3.8) using HSPICE, the model for the commercially available OpAmp TL081, and the diagrams shown in Figures 2.18 and 2.16, one can generate the attractors from 5 and 7 scrolls by selecting functional specifications: 

\( a = b = c = d = 0.7, k = 1, \alpha = 0.016, s = 60.606, h_1 \simeq 1, I_{sat} = 100 \mu A, R_{ix} = 10k\Omega, C = 1\mu F, R = 1M\Omega, R_{ia} = R_{ib} = R_{ic} = R_{id} = 10k\Omega, R_{fa} = R_{fb} = R_{fc} = R_{fd} = 7k\Omega, R_{x1} = R_{x2} = R_{x3} = R_{x4} = R_y = 10k\Omega, R_i = 10k\Omega \) and \( V_{sat} = \pm 16V \), \[111\]. The simulation results are shown in Figure 2.17.
2.5 Circuit implementation

\[ V_{in(x)} = I_{out(x)} \]

Figure 2.16: Realization of the SNLF using operational amplifiers.

Figure 2.17: Generating (a) 5-scrolls attractor, and (b) 7-scrolls attractor.

2.5.2. CFOA implementation

The realization of the dynamical system in (2.3.8) using CFOAs is shown in Figure 2.18. The circuit realization of the SNLF, can be implemented by using CFOAs working in the saturation region with shift bias-levels. For instance, the CFOA voltage behaviors can be modeled by the opamp finite-gain model shown in Figure 2.15, so that a SNLF can be described by

\[ V_o = \frac{A_v}{2} \left( |V_i + \frac{V_{sat}}{A_v}| - |V_i - \frac{V_{sat}}{A_v}| \right) \]

and if a shift-voltage \((\pm E)\) is added, one gets the shifted-voltage determined by (2.5.3) for positive...
and negative shifts, respectively. A resistor can be added to realize a current-to-voltage transformation. To generate the SNLF, $E$ takes different values in (2.5.3) to synthesize the required plateaus and slopes. The cell shown in Figure 2.19 is used to realize voltage and current SNLFs.

For instance, the cell in Figure 2.19 can realize the SNLF from Eq. (2.4.2), and the number of basic cells (BC) is determined by $BC = (\text{number of scrolls}) - 1$, which are parallel-connected [111].

By selecting $R_{ix} = 10K\Omega$, $C = 2.2nf$, $R = 7K\Omega$, $R_x = R_y = R_z = 10K\Omega$, $R_f = 10K\Omega$, $R_i = 10K\Omega$ in Figure 2.18, and $R_{ix} = 10K\Omega$, $R_c = 64K\Omega$, $R_{iz} = 1K\Omega$, $R_{fz} = 1M\Omega$, $E_1 = \pm1v$, $E_2 = \pm3v$ with $V_{sat} = \pm18v$ in the BC (SNLF), the result is $N = 5 - \text{scrolls}$, as shown in Figure 2.20. The circuit realization was performed by using the commercially available CFOA AD844.
2.5 Circuit implementation

This Thesis proposes using the state-variable synthesis method to define the system level topology, and using transconductors and capacitors (Gm-C techniques) as basic elements for the circuit level realization.

Figures 2.21 and 2.22 show a simplified $G_m - C$ realization of the system described by (2.4.1), based on the general architecture of Figure 2.12, and using differential-input transconductors. All the integrating capacitors are assumed to be identical. Linear transconductors have been implemented by building a unitary block with gain $g_{mf}$ and, consequently, the respectively transconductance ratios to achieve the coefficients values $a, b, c,$ and $d_1$ of the third-order chaotic system.

![Figure 2.21: OTA-based implementation with SNLF in voltage mode.](image)

2.5.3. OTA implementation

Figure 2.20: 5-scrolls chaotic attractor (a) SNLF, and (b) Phase space portrait.
Figure 2.22: OTA-based implementation with SNLF in current mode.

Figure 2.23 shows two different alternatives for the implementation of the nonlinear transconductor. In both cases, the PWL transfer function is obtained by shaping the characteristic of a linear transconductor. In Figure 2.23(a), shaping is realized in voltage-mode at the front of the unit transconductance element, while in Figure 2.23(b) shaping is done in current mode. Subsequent analyzes will use any of these representations as dictated by convenience although, in practice, the current shaping approach for the realization of the PWL transconductor, is adopted herein.

Figure 2.23: Alternatives for the physical realization of the nonlinear transconductor.

By applying Kirchhoff’s current law in Figure 2.22, equations given in (2.5.4) are obtained; where $SNLF = f(x_1)$. 
\[
\dot{x}_1 = \frac{g_{mx} x_2}{C} \\
\dot{x}_2 = \frac{g_{my} x_3}{C} \\
\dot{x}_3 = -\frac{g_{ma} x_1}{g_{mf} C} - \frac{g_{mb} x_2}{g_{mf} C} - \frac{g_{mc} x_3}{g_{mf} C} + \frac{g_{md} f(x_1)}{g_{mf} C}
\]

where \( g_{ma} = a g_{mf}, g_{mb} = b g_{mf}, g_{mc} = c g_{mf}, g_{md} = d_1 g_{mf} \) and \( g_{mx} = g_{my} = g_{mz} = C \)

The operating frequency of the system is determined by the time constant of the integrators and is defined by:

\[
f = \frac{g_m}{2\pi C}
\]

To perform frequency scaling is necessary to multiply (2.4.1) by the scale factor required, frequency scaling directly influences the capacitors value, where now the new value of the capacitor is evaluated by \( C_{fs} = \frac{C}{factor} \).

2.5.4. OTA macromodel

Figure 2.24(a) shows a macromodel valid both for the linear and the PWL differential input transconductors used in Figure 2.22. It includes the following second-order effects [33]: 1) finite input and output impedance; 2) frequency-dependent transconductance; and 3) output current saturation. Usually, the input conductance is very low and \( R_{in} \) can be disregarded with minor problems. Capacitances \( C_{in} \) and \( C_o \) (alternatively, \( C_q \) for the PWL transconductor) are the parasitics associated to the input and output nodes, respectively. \( R_o \) (alternatively, \( R_q \) for the PWL transconductor) represents the output resistance.
Figure 2.24: Alternatives for the physical realization of the nonlinear transconductor.

Figure 2.24(b) models the output current saturation observed for input voltage larger than $E_1$. In practice, the transfer gain of real transconductors varies continuously with the input signal level.

2.5.5. OTA SNLF synthesis

Figures 2.21 and 2.22 show two different alternatives for the implementation of the nonlinear transconductor. In both cases, the PWL transfer function is obtained by shaping the characteristic of a linear transconductor. In Figure 2.21, shaping is realized in voltage-mode at the front of the unit transconductance element, while in Figure 2.22 shaping is done in current mode.

PWL generation by voltage shaping

A voltage saturated function is synthesized using the OTA finite-gain model shown in Figure 2.25a. Therefore, if a shift-voltage ($\pm E$) is added, one gets the saturated function with a shifted-voltage determined by (2.5.6) for positive and negative shifts, respectively. The SNLF can be implemented as shown in Figure 2.25b, where the number of basic cells equals the number of scrolls to be generated minus 1.
\[ I_o = \frac{g_m}{2} (|V_i + \frac{I_{sat}}{g_m} \pm E| - |V_i - \frac{I_{sat}}{g_m} \pm E|) \]  

(2.5.6)

\[ I_o = \frac{\alpha}{2} (|I_i + \frac{I_{sat}}{\alpha} \pm I| - |I_i - \frac{I_{sat}}{\alpha} \pm I|) \]  

(2.5.7)

**PWL generation by current shaping**

A current saturated function is synthesized using the current comparator finite-gain model shown in Figure 2.26a. Therefore, if a shift-current (±I) is added, one gets the saturated functions with shifted-current determined by (2.5.7) for positive and negative shifts, respectively. The SNLF can be implemented as shown in Figure 2.26b, where the number of basic cells equals the number of scrolls to be generated minus 1.
Simulation results

The complete oscillator was designed following the scheme shown in Figure 2.22, and the saturated nonlinear function design shown in Figure 2.26 using the OTA macromodel from Figure 2.27.

Figures 2.28 to 2.31 show phase space portraits for the cases to generate 2 to 5 scrolls chaotic attractors. The macromodel circuit realization was performed by selecting coefficients $a = b = c = d_1 = 0.7$ that requires $g_{ma} = g_{mb} = g_{mc} = g_{md} = 140\mu A/V$, $g_{mf} = g_{mz} = g_{my} = g_{mz} = 200\mu A/V$, $I_{sat}$ and $I_{off}$ depends of the number of scrolls and the desired dynamical range. An integration capacitance $C = 10pF$ has been used, which implies an operating frequency of $3.1831 MHz$ according to (2.5.5). Figure 2.32 shows the spectrum of $x_1$ and $x_2$ for a 4-scrolls attractor.
2.5 Circuit implementation

Figure 2.28: 2-scrolls attractor using the OTA macromodel.

Figure 2.29: 3-scrolls attractor using the OTA macromodel.
2. Chaotic systems

Figure 2.30: 4-scrolls attractor using the OTA macromodel.

Figure 2.31: 5-scrolls attractor using the OTA macromodel.
Figure 2.32: Spectrum of signals: (a) $x_1$, and (b) $x_2$ for a 4-scrolls attractor.
3.1. Computing Lyapunov exponents

The deterministic, still unpredictable behavior of nonlinear dissipative dynamical systems is an important subject in more and more fields of science, from mathematics to biology; even in engineering. The Lyapunov exponents give the most characteristic description of the presence of a deterministic nonperiodic flow. Therefore, Lyapunov exponents are asymptotic measures characterizing the average rate of growth (or shrinkage) of small perturbations to the solutions of a dynamical system [112]. Lyapunov exponents provide quantitative measures of response sensitivity of a dynamical system to small changes in initial conditions [113]. The number of Lyapunov exponents equals the number of state variables, and if at least one is positive, this is an indication of chaos [23,31,113]. That way, an algorithm capable of computing the Lyapunov exponents in a simple fashion is very much in need to guarantee chaotic regime.

Let us consider a continuous-time dynamical system depicted by

\[ \dot{x} = f(x), \quad x \in \mathbb{R}^n \]  

(3.1.1)

where \( x \) and \( f \) are vector fields of \( n \)-order. To determine the Lyapunov exponents of the system (3.1.1) its variational equation is considered, which is defined by
\[ \dot{v} = \frac{\partial f(x)}{\partial x} v = J(x)v \]  

(3.1.2)

where \( J \) is the Jacobian matrix of \( f(x) \) and \( v \) the variational state-variable [114]. A solution of (3.1.2) under initial conditions \( v(0) \) can be written as \( v = Vv(0) \), with matrix \( V \) satisfying

\[ \dot{V} = J(x)V, \quad V(0) = I \]  

(3.1.3)

Here \( I \) indicates the identity matrix. If we relate this solution to the evolution of the axes of an infinitesimal ellipsoid by means of \( \eta_i = V\eta_i(0) \) for \( i = 1, 2, \ldots, n \) where \( \eta_i(0) \) denotes an orthogonal basis; then the Lyapunov vectors are defined as the expansion rate of the axes length \( \eta_i \). Therefore, by solving (3.1.3) the LEs are computed with

\[ LE_i = \lim_{t \to \infty} \frac{1}{t} \ln \| \eta_i(t) \| \]  

(3.1.4)

The Lyapunov exponents can be computed by applying the methods given in [23,31,113]. To measure the three Lyapunov exponents of the original chaotic oscillator system in (2.4.1), this original system is observed by expanding it with other three systems that change according to the derivative of (2.4.1). If \( u = [\dot{x}, \dot{y}, \dot{z}]^T, \ u \in \mathbb{R}^3 \), represents one state of the original dynamical system at any \( t > 0 \), the state of the new observed system will be \( v = [u, u_1, u_2, u_3]^T, \ v \in \mathbb{R}^{12} \), where \( u_i \), for \( i = 1, 2, 3 \), are the three added systems that will measure precisely the change of those small perturbations on each orthogonal directions, for each of the three state variables in (2.4.1). The initial state of the expanded system is set to

\[ v_0 \in \mathbb{R}^{12} \]  

(3.1.5)

\[ v_0 = [u_0^T, e_1^T, e_2^T, e_3^T]^T \]

where \( u_0 \) is the vector of initial conditions \([x_0, y_0, z_0]^T\); \([e_1, e_2, e_3] = I \), and \( I \) is the identity matrix of size \( 3 \times 3 \). Thus, \( e_i \), for \( i = 1, 2, 3 \), are each unitary column vectors of the identity matrix \( I \).

The observational system is integrated by several steps until a period \( T_O \) is reached. After this step, the state of the variational system is orthonormalized by using the
3.2 Optimization algorithms

standard Gram-Schmidt method [114]. The next integration is carried out by using the new orthonormalized vectors as initial conditions.

The Lyapunov exponents measures the long time sensitivity of the flow in \( \mathbf{u} \) with respect to the initial data \( \mathbf{u}_0 \) at the directions of every orthonormalized vector. This measure is taken when the variational system is orthonormalized. If \( \mathbf{v} = [\mathbf{u}, \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3]^T \) is the state after the matrix \([\mathbf{u}, \mathbf{u}_2, \mathbf{u}_3]\) is orthonormalized, the Lyapunov exponents \( \lambda_i \), for \( i = 1, 2, 3 \) are calculated by

\[
\lambda_i \approx \frac{1}{T} \sum_{j=1}^{k} \ln \| \mathbf{p}_i \| \tag{3.1.6}
\]

where the number of summations \( k \) is calculated as \( \left\lceil \frac{T}{T_O} \right\rceil \), and \( T \) is the simulation time.

For instance, in [62, 115] the period of time \( T_O \) is selected by using the minimum absolute value of all the eigenvalues of the system as

\[
T_O = \frac{1}{l_{\text{min}}}
\]

where \( l_{\text{min}} \) represents the value of the minimum eigenvalue of the system in (2.4.1) [116].

Algorithm 1 shows the pseudo code to compute the \( n \)-Lyapunov exponents. The brackets indicate that the variable is a vector, double brackets denote a square matrix, and an index in the second bracket indicates a column vector, for example: \( \mathbf{v}[[i]] \) is the \( i \)-th column of \( \mathbf{v}[] \).

### 3.2. Optimization algorithms

Mathematical Optimization is the process of the formulation and the solution of a constrained problem of the general mathematical form:

\[
\text{minimize } f(x), \quad x = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}
\]

subject to the constraints:

\[
g_j(x) \leq 0, \quad j = 1, 2, \ldots, m
\]

\[
h_j(x) = 0, \quad j = 1, 2, \ldots, r
\]
Algorithm 1 Pseudo code to compute Lyapunov exponents

1: $u[ ][ ] = I$
2: for $i$ from 1 to $n$ do
3: $\lambda[i] = 0$
4: $sum[i] = 0$
5: end for
6: $k = 0$
7: repeat
8: $\lambda_{old}[ ] = \lambda[ ]$
9: $\delta_x[ ][ ] = \phi_T(x[ ])u[ ][ ]$
10: $x[ ] = \phi(x[ ])$
11: for $i$ from 1 to $n$ do
12: $v[ ][i] = \delta x[ ][ ]$
13: for $j$ from 1 to $(i-1)$ do
14: $v[ ][i] = v[ ][i] - \langle v[ ][i], u[ ][j]\rangle u[ ][j]$ 
15: end for
16: $u[ ][i] = v[ ][i]/\|v[ ][i]\|$
17: $sum[i] = sum[i] + \ln \|v[ ][i]\|$ 
18: $\lambda[i] = sum[i]/kT$
19: end for 
20: until $T_{total}$

where $f(x)$, $g_j(x)$ and $h_j(x)$ are scalar functions of the real column vector $x$.

The continuous components $x_i$ of $x = [x_1, x_2, \ldots, x_n]^T$ are called the (design) variables, $f(x)$ is the objective function, $g_j(x)$ denotes the respective inequality constraint functions and $h_j(x)$ the equality constraint functions. The optimum vector $x$ that solves problem (3.2.1) is denoted by $x^*$ with corresponding optimum function value $f(x^*)$. If no constraints are specified, the problem is called an unconstrained minimization problem [32].

To solve the optimization problem in (3.2.1), efficient search or optimization algorithms are needed. There are many optimization algorithms which can be classified in many ways, depending on their focus and characteristics [33,34].

If the derivative or gradient of a function is the focus, optimization can be classified into gradient-based algorithms and derivative-free or gradient-free algorithms. Gradient-based algorithms use derivative information, and they are often very efficient. Derivative-free algorithms do not use any derivative information but the values of the function itself. Some functions may have discontinuities or it may be expensive to calculate derivatives accurately, and thus derivative-free algorithms become very
useful [117]. From a different perspective, optimization algorithms can be classified into trajectory-based and population-based. A trajectory-based algorithm typically uses a single agent or one solution at a time, which will trace out a path as the iterations continue. On the other hand, population-based algorithms use multiple agents which will interact and trace out multiple paths [118]. Optimization algorithms can also be classified as deterministic or stochastic. If an algorithm works in a systematic deterministic manner without any random nature, it is called deterministic. For such an algorithm, it will reach the same final solution if it starts with the same initial point. On the other hand, if there is some randomness in the algorithm, it will usually reach a different point every time it is executed, even though the same initial point is used.

Genetic Algorithms (GA), Differential Evolution (DE) and Particle Swarm Optimization (PSO) are a class of evolutionary algorithms. They work with a population of tentative solutions to the problem, and new solutions are generated by somehow combining the information of the old ones and by surviving the ones with better fitness. These algorithms are used as solvers for global optimization problems, more commonly in problems with continuous representations [117,118].

The usefulness of these evolutionary algorithms relies in the fact that they need only the value of function $f$ to work, or in other words, it is not necessary that $f$ be continuous or get any information about the derivative of function $f$.

### 3.2.1. Genetic algorithm

Genetic algorithms (GAs) are probably the most popular evolutionary algorithms with a diverse range of applications. A vast majority of well-known optimization problems have been solved by genetic algorithms. In addition, genetic algorithms are population-based and many modern evolutionary algorithms are directly based on, or have strong similarities to, genetic algorithms [119]. Genetic algorithms, developed by John Holland and his collaborators in the 1960s and 1970s [120], are a model or abstraction of biological evolution based on Charles Darwin’s theory of natural selection. GAs operate on the principle of “survival of the fittest”. In this manner, a GA has the capability to generate new design solutions from a population of existing solutions, and discard the solutions which have an inferior performance or fitness. Holland was the first to use recombination, mutation and selection in the study of
adaptive and artificial systems. These genetic operators are the essential components of genetic algorithms as a problem-solving strategy [117, 120].

This is often done through the following procedure: 1) definition of an encoding scheme; 2) definition of a fitness function or selection criterion; 3) creation of a population of chromosomes; 4) evaluation of the fitness of every chromosome in the population; 5) creation of a new population by performing fitness-proportionate selection, crossover and mutation; 6) replacement of the old population by the new one. Steps 4), 5) and 6) are then repeated for a number of generations. At the end, the best chromosome is decoded to obtain a solution to the problem.

Each iteration, which leads to a new population, is called a generation. Fixed-length chromosomes are used in most genetic algorithms at each generation although there is substantial research on variable-length structures. The coding of the objective function is usually in the form of binary arrays or real-valued arrays in genetic algorithms. An important issue is the formulation or choice of an appropriate fitness function that determines the selection criterion in a particular problem.

The structure and the steps that execute the proposed GA are highlighted in the pseudocode depicted in Algorithm 2.

3.2.2. Differential Evolution algorithm

Differential evolution (DE) was developed by R. Storn and K. Price [121]. It is a vector-based evolutionary algorithm, and unlike genetic algorithms, differential evolution carries out operations over each component (or each dimension of the solution). Almost everything is done in terms of vectors, and DE can be viewed as a self-organizing search, directed towards the optimum.

DE is an evolutionary algorithm that works with a population of tentative solutions to the problem, and new solutions are generated by combining the old ones and by surviving the ones with better fitness [118].

The general convention used to denote the DE strategy is $DE/x/y/z$. DE stands for differential evolution algorithm, $x$ represents a string denoting the vector to be perturbed, $y$ is the number of difference vectors considered for perturbation of $x$, and $z$ is the type of crossover being used (exp: exponential; bin: binomial).

We use the most common version of DE: DE/rand/1/bin. Hence, the perturbation is on any randomly chosen vector, for perturbation a single vector difference is used
Algorithm 2 Genetic Algorithm

1: $N$ is the number of individuals
2: $G$ is the number of iterations (generations)
3: Variable bounds $x_i \in [l_i, u_i]$, for $i = 1, 2, \ldots D$
4: Procedure GA $(N, G, l_i, u_i)$
5: for $i = 1 : N$ do
6:   for $d = 1 : D$ do
7:     $x_i[d] = l_d + (u_d - l_d) \cdot \text{rand}()$
8:   end for
9:   $\text{Pop} \leftarrow x_i[d]$
10: $x_i.\text{fit} \leftarrow \text{evaluate Pop}$
11: end for
12: for $i = 1 : G$ do
13:   Rank the best $N/2$ solutions in Pop and save them in $\text{Pop}_1$ \hspace{1em} \triangleright \text{Elitism based selection}$
14:   Randomly select two solutions $x_A$ and $x_B$ from $\text{Pop}$ \hspace{1em} \triangleright \text{Crossover}$
15:   generate $x_C$ and $x_D$ by one-point crossover to $x_A$ and $x_B$
16:   save $x_C$ and $x_D$ to $\text{Pop}_2$
17:   for $i = 1 : N/2$ do
18:     Select a solution $x_j$ from $\text{Pop}_2$
19:     mutate each bit of $x_j$ under the rate $PM$ and generate a new solution $x'_j$ \hspace{1em} \triangleright \text{Mutation}$
20:     if $x'_j$ is unfeasible then
21:       update $x'_j$ with a feasible solution by repairing $x'_j$
22:     end if
23:     update $x_j$ with $x'_j$ in $\text{Pop}_2$
24:   end for
25: $\text{Pop} = \text{Pop}_1 + \text{Pop}_2$
26: end for
27: return the best solution $x_i[d]$ in $\text{Pop}$
and the type of crossover is binomial. For perturbation with a single vector difference, out of three distinct randomly chosen vectors, the weighted vector difference of any two vectors is added to the third one. In binomial crossover, the crossover is performed on each of the $D$ variables whenever a randomly picked number between 0 and 1 is below a certain threshold $R$. The pseudocode of DE is shown in Algorithm 3. Each individual is represented by a vector $x \in \mathbb{R}^D$, and its fitness value is represented as $x.fit$. The location $j$ of individual $i$ is represented as $x_i[j]$. $rand()$ is a function that returns a random number greater or equal to zero and less than one. $evaluate()$ is a function that calculates de fitness function (function to be optimized) [122]. The core of DE is in the loop on lines 15-24: a mutated individual is generated from three different randomly chosen individuals; each value of the new vector (a new individual) is calculated from the first individual, plus the difference of the other two individuals multiplied by $F$, the difference constant; the new vector value is calculated if random real number (between zero and one) is less than $R$, the DE’s recombination constant. To prevent the case when the new individual is equal to the first reference individual, at least one vector component is forced to be calculated from the mutated vector, it is in line 16 of the pseudocode, when $d = jrand$, and $jrand$ is an integer random number between 1 and $D$. Then the new individual is evaluated. If it is better than the original one (in line 26), then the child replaces it (line 27).
### Algorithm 3 Differential Evolution algorithm

3.2 Optimization algorithms

1. \( N \) is the number of individuals
2. \( G \) is the number of iterations (generations)
3. Variable bounds \( x_i \in [l_i, u_i] \), for \( i = 1, 2, \ldots D \)
4. Procedure \( \text{DE} (N, G, \{l_i\}, \{u_i\}) \)
5. \textbf{for} \( i = 1 : N \) \textbf{do}
6. \hspace{1em} \textbf{for} \( d = 1 : D \) \textbf{do}
7. \hspace{2em} \( x_i[d] = l_d + (u_d - l_d) \cdot \text{rand}() \)
8. \hspace{1em} \textbf{end for}
9. \hspace{1em} \( x_i.f\text{it} \leftarrow \text{evaluate}(x_i) \)
10. \hspace{1em} \textbf{end for}
11. \textbf{for} \( i = 1 : G \) \textbf{do}
12. \hspace{1em} Let \( j_1, j_2 \) and \( j_3 \) be three random numbers in \( \{1, N\} \)
13. \hspace{1em} without replacement and also different to \( i \).
14. \hspace{1em} \( j\text{rand} \leftarrow \lfloor \text{rand}() \cdot D \rfloor + 1 \)
15. \hspace{1em} \textbf{for} \( d = 1 : D \) \textbf{do}
16. \hspace{2em} \textbf{if} \( \text{rand}() < R \ \text{OR} \ d = j\text{rand} \textbf{then} \)
17. \hspace{2em} \hspace{1em} \( y[d] = x_{i2}[d] + F(x_{i0}[d] - x_{i1}[d]) \)
18. \hspace{2em} \hspace{1em} \textbf{if} \( y[d] < l_d \ \text{OR} \ y[d] > u_d \textbf{then} \)
19. \hspace{2em} \hspace{2em} \hspace{1em} \( y[d] = l_d + (u_d - l_d) \cdot \text{rand}() \)
20. \hspace{2em} \hspace{1em} \textbf{end if}
21. \hspace{2em} \textbf{else}
22. \hspace{2em} \hspace{1em} \( y[d] = x_i[d] \)
23. \hspace{2em} \textbf{end if}
24. \hspace{1em} \textbf{end for}
25. \hspace{1em} \( y.f\text{it} = \text{evaluate}(y) \)
26. \hspace{1em} \textbf{if} \( y.f\text{it} < x_i.f\text{it} \textbf{then} \)
27. \hspace{1em} \hspace{1em} \( x_i \leftarrow y; x_i.f\text{it} \leftarrow y.f\text{it} \)
28. \hspace{1em} \textbf{end if}
29. \hspace{1em} \textbf{end for}
30. \textbf{search} \( q = x_k | \min(x_k.f\text{it}), \text{for } k = 1, 2, \ldots, N \)
31. \( q \) is the solution at iteration \( i \)

3.2.3. Particle Swarm Optimization

Particle swarm optimization (PSO) was developed by Kennedy and Eberhart in 1995 [123], and was inspired on swarm behaviour observed in nature such as fish and bird schooling. Since then, PSO has generated a lot of attention, and now forms an exciting, ever-expanding research subject in the field of swarm intelligence. PSO has been applied to almost every area in optimization, computational intelligence, and
design applications. There are at least two dozens of PSO variants, as well as hybrid algorithms obtained by combining PSO with other existing algorithms, which are also increasingly popular [33, 117].

PSO searches the space of an objective function by adjusting the trajectories of individual agents, called particles. Each particle traces a piecewise path which can be modelled as a time-dependent positional vector. The movement of a swarming particle consists of two major components: a stochastic component and a deterministic component. Each particle is attracted toward the position of the current global best \( p_{best,i}.pos_d \) and its own best known location \( p_{best,gbest[i]}.pos \) in history.

When a particle finds a location that is better than any previously found locations, then it updates this location as the new current best for particle \( i \). There is a current best for all \( N \) particles at any time \( t \) at each iteration. The aim is to find the global best among all the current best solutions until the objective no longer improves or after a certain number of iterations [124].

The pseudocode for PSO is shown in Algorithm 4. Each particle \( p_i \) has three associated values: position \( p_i.pos \), velocity \( p_i.vel \), and the value of the fitness function \( p_i.fit \). Particle pbest has only position and fitness function value. \( gbest[] \) is a vector that stores indexes to reference pbest particles. \( rand() \) is a function that returns a random number greater or equal to zero and less that one. \( evaluate() \) is a function that calculates the value of the fitness for the problem to solve. This PSO version was inspired from [123] and [124, 125]. The main advantage of this PSO algorithm (not using extra parameters), consist on having only the essential parameters, i.e., the number of individuals (particles) and the number of iterations (generations).

Particles position \( p_i \) are initialized randomly and also their velocities (in lines 5-10 and 11-15 in Algorithm 4, respectively). Each particle is evaluated and \( p_{best,i} \) particles are initialized equal to the \( p_i \) ones. For a given number of iterations the following process is applied: (1) three random numbers are calculated in \([1,N]\) \( (N=\)population size) with replacement; \( gbest[i] \) points to the best particle inside this cluster of three particles.(2) A new particle is calculated and its velocity is updated (line 22-23). If this new particle is better than its associated \( p_{best} \) then \( p_{best} \) particle takes the values of the new particle. The core of PSO is in the loop of lines 17-23. The update rules are
$$p_i . pos_d \leftarrow w_i . vel_d + \varphi_1 U_1 (pbest_i . pos_d - p_i . pos_d) + \varphi_2 U_2 (pbest_{gbest[i]} . pos_d - p_i . pos_d)$$

where $w$ is a parameter called inertia weight, $\varphi_1$ and $\varphi_2$ are two parameters called acceleration coefficients, $U_1$ and $U_2$ are two random numbers uniformly distributed in the interval $[0, 1)$.

### 3.3. Maximizing the unpredictability in multi-scroll chaotic oscillators

Unpredictability is an important property of chaotic systems, since it means that the future events cannot be forecasted from the past events. Detecting the presence of chaos in a dynamical system is an important problem that is solved by measuring the largest, positive or maximum Lyapunov exponent (MLE). The Lyapunov exponents give the most characteristic description of the presence of a deterministic non-periodic flow. Therefore, Lyapunov exponents not only provide a qualitative characterization of dynamical behaviour but also the exponent itself determines the measure of predictability. This means that a large MLE is equivalent to a high unpredictability [31]. Hence, the estimation of the Lyapunov exponents as the useful dynamical classifier for deterministic chaotic systems is an important issue in nonlinear chaotic systems optimization.

The calculation of the Lyapunov exponents for the saturated nonlinear function series based chaotic oscillator described by (2.4.1), can be performed by simply setting: $a = b = c = d_1 = 0.7, \ m = 0.1 [16, 126]$. In most reported approaches using saturated nonlinear function series-based chaotic oscillator [16, 67, 69, 111, 111] the coefficients of the system are fixed to 0.7, but the MLE value is relatively small, as shown in Table 3.1. Furthermore, in this section the application and comparison of three computational intelligence algorithms: Genetic Algorithm (GA), Differential Evolution (DE) and Particle Swarm Optimization (PSO), to optimize MLE in a multi-scroll chaotic oscillator based on saturated nonlinear function series with a modification of the standard settings of the coefficient values of the mathematical description and taking into account the correct distribution of the scrolls drawing the phase space portrait, is presented.
Algorithm 4 Particle swarm optimization algorithm.

1: \( N \) is the number of particles
2: \( G \) is the number of iterations (generations)
3: Variable bounds \( x_i \in [l_i, u_i] \), for \( i = 1, 2, \ldots D \)
4: Procedure PSO \((N,G,\{l_i\}, \{u_i\})\)
5: \( \textbf{for} \ i = 1 : N \ \textbf{do} \quad \triangleright \text{Initialize particle’s positions} \\
6: \quad \textbf{for} \ d = 1 : D \ \textbf{do} \\
7: \quad \quad p_i.pos_d = l_d + (u_d - l_d) \cdot \text{rand}() \\
8: \quad \quad pbest_i.pos_d \leftarrow p_i.pos_d \\
9: \quad \quad p_i.fit \leftarrow \text{evaluate}(p_i.pos) \\
10: \quad \quad pbest_i.fit \leftarrow p_i.fit \\
11: \quad \textbf{for} \ i = N : D \ \textbf{do} \quad \triangleright \text{Initialize particle’s velocities} \\
12: \quad \textbf{for} \ d = 1 : D \ \textbf{do} \\
13: \quad \quad vmin = l_d - p_i.pos_d \\
14: \quad \quad vmax = u_d - p_i.pos_d \\
15: \quad \quad p_i.vel_d = vmin + (vmax - vmin) \cdot \text{rand}() \\
16: \quad \textbf{for} \ g = 1 : G \ \textbf{do} \quad \triangleright \text{Iterate G generations} \\
17: \quad \quad \textbf{for} \ i = 1 : N \ \textbf{do} \quad \triangleright \text{For each particle} \\
18: \quad \quad \quad \textbf{for} \ d = 1 : D \ \textbf{do} \quad \triangleright \text{For each particle} \\
19: \quad \quad \quad \quad \textbf{for} \ j = 1 : N \ \textbf{do} \quad \triangleright \text{For each dimension} \\
20: \quad \quad \quad \quad \quad \textbf{let} \ j_1, j_2 \text{ and } j_3 \text{ be three random numbers in } \{1, N\} \\
21: \quad \quad \quad \quad \quad \text{gbest}[i] = k|\min(pbest_k.fit), \text{for } k \in \{i, j_1, j_2, j_3\} \\
22: \quad \quad \quad \textbf{for} \ i = N : D \ \textbf{do} \quad \triangleright \text{For each particle} \\
23: \quad \quad \quad \quad \textbf{for} \ d = 1 : D \ \textbf{do} \quad \triangleright \text{For each particle} \\
24: \quad \quad \quad \quad \quad \textbf{if} \ p_i.pos_d < l_d \\
25: \quad \quad \quad \quad \quad \quad p_i.pos_d \leftarrow l_d; \quad p_i.vel_d = 0 \\
26: \quad \quad \quad \quad \quad \textbf{if} \ p_i.pos_d > u_d \textbf{ then} \\
27: \quad \quad \quad \quad \quad \quad p_i.pos_d \leftarrow u_d; \quad p_i.vel_d = 0 \\
28: \quad \quad \quad \quad \textbf{if} \ f < \text{gbest.fit}, \text{then} \\
29: \quad \quad \quad \quad \quad \quad pbest_i.pos \leftarrow p_i.pos \\
30: \quad \quad \quad \quad \quad \quad pbest_i.fit \leftarrow f \\
31: \quad \quad \textbf{search } q = \text{gbest}_k.pos - \min(gbest_k.fit), \text{for } k = 1, 2, \ldots , N. \\
32: \quad \textbf{q is the solution at iteration } g.$
3.3 Maximizing the unpredictability in multi-scroll chaotic oscillators

Table 3.1: Calculated MLE with coefficients values \((a, b, c, d_1 = 0.7)\)

<table>
<thead>
<tr>
<th>Scrolls</th>
<th>MLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.105422</td>
</tr>
<tr>
<td>3</td>
<td>0.138087</td>
</tr>
<tr>
<td>4</td>
<td>0.142087</td>
</tr>
<tr>
<td>5</td>
<td>0.134534</td>
</tr>
<tr>
<td>6</td>
<td>0.147785</td>
</tr>
<tr>
<td>7</td>
<td>0.148159</td>
</tr>
</tbody>
</table>

For this chaotic oscillator, the optimization problem tries to find the values of the four coefficient variables \(a, b, c\) and \(d_1\) in (2.4.1) that increase MLE. Those four coefficients can take values within the range \([0.0, 1.0]\). In our investigation, a resolution of 4 decimal digits for those variables, i.e. from 0.0001 to 1.0000 [115, 127], was used.

The MLE was measured like it was described in section 3.1. In addition, a very new procedure reported in [128] to measure the dispersions of the phase space portraits (PSP) coverture among all generated scrolls was included in the optimization loop. The procedure consist on counting the number of occurrences of the state trajectory in generating each scroll. Fig. 2.9 shows the PWL function to generate 4 scrolls. The procedure for distributing the trajectories in the PSP basically counts how many times the state variable, e.g. \(x\), crosses the center of saturated levels (horizontal zones) of function \(f\) in this Fig. 2.9 at the set of values \(x = \{-3, -1, 1, 3\}\). The quantitative measure taken in [128] is the standard deviation among all crossing values at the end of the simulation time. In this work, the average among all crossing values to decide if it is a feasible or unfeasible solution, is taken. For a feasible solution, 70\% of the average of the crosses is reached for each saturated region. e.g. for 4-scrolls \(crosses = \{224, 315, 301, 210\}\), the average of these crosses is \(mean = 262.5\), then, if in each case 183 crosses are reached, it is a feasible solution.

GA, DE and PSO are stochastic algorithms in nature, hence, different results can be obtained when executing different runs and such results may also depend on the parameter settings [67, 127, 129]. For instance, Table 3.2 shows the MLE obtained for 2 to 7-scroll chaotic attractors by applying the three selected algorithms. Each algorithm was executed 30 times, so Table 3.2 shows the best value of MLE, the mean Lyapunov exponent value and the standard deviation.

GA was executed with the crossover probability \(P_c = 0.9\) and mutation probability
Table 3.2: MLE values, mean values, standard deviation and coefficient value of 30 executions of each heuristic against number of scrolls.

<table>
<thead>
<tr>
<th>Scrolls</th>
<th>Algorithm</th>
<th>M.L.E.</th>
<th>Mean</th>
<th>St.dev.</th>
<th>Constants - [a, b, c, d1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>GA</td>
<td>0.221986</td>
<td>0.216023</td>
<td>0.005391</td>
<td>[0.9816, 0.8410, 0.4988, 0.6540]</td>
</tr>
<tr>
<td>2</td>
<td>DE</td>
<td>0.222767</td>
<td>0.218224</td>
<td>0.001765</td>
<td>[1.0000, 0.8284, 0.5321, 1.0000]</td>
</tr>
<tr>
<td>2</td>
<td>PSO</td>
<td>0.223114</td>
<td>0.219041</td>
<td>0.002024</td>
<td>[0.9970, 0.8469, 0.5098, 0.9221]</td>
</tr>
<tr>
<td>3</td>
<td>GA</td>
<td>0.298260</td>
<td>0.283042</td>
<td>0.011624</td>
<td>[0.9895, 0.7774, 0.3560, 1.0000]</td>
</tr>
<tr>
<td>3</td>
<td>DE</td>
<td>0.297813</td>
<td>0.290483</td>
<td>0.002884</td>
<td>[1.0000, 0.7782, 0.3416, 1.0000]</td>
</tr>
<tr>
<td>3</td>
<td>PSO</td>
<td>0.301033</td>
<td>0.294377</td>
<td>0.003385</td>
<td>[1.0000, 0.7724, 0.3618, 0.9927]</td>
</tr>
<tr>
<td>4</td>
<td>GA</td>
<td>0.303209</td>
<td>0.289411</td>
<td>0.014313</td>
<td>[0.9367, 0.6894, 0.3204, 0.9896]</td>
</tr>
<tr>
<td>4</td>
<td>DE</td>
<td>0.310734</td>
<td>0.300321</td>
<td>0.006029</td>
<td>[0.9399, 0.7037, 0.2854, 0.9660]</td>
</tr>
<tr>
<td>4</td>
<td>PSO</td>
<td>0.315349</td>
<td>0.306306</td>
<td>0.004998</td>
<td>[0.9607, 0.7028, 0.2728, 0.9880]</td>
</tr>
<tr>
<td>5</td>
<td>GA</td>
<td>0.296158</td>
<td>0.281553</td>
<td>0.012683</td>
<td>[0.9815, 0.7355, 0.1961, 1.0000]</td>
</tr>
<tr>
<td>5</td>
<td>DE</td>
<td>0.321793</td>
<td>0.302033</td>
<td>0.009817</td>
<td>[0.9770, 0.6622, 0.2180, 1.0000]</td>
</tr>
<tr>
<td>5</td>
<td>PSO</td>
<td>0.322885</td>
<td>0.309523</td>
<td>0.007469</td>
<td>[0.9497, 0.6494, 0.2749, 0.9966]</td>
</tr>
<tr>
<td>6</td>
<td>GA</td>
<td>0.313739</td>
<td>0.298833</td>
<td>0.008199</td>
<td>[0.9520, 0.5422, 0.2819, 1.0000]</td>
</tr>
<tr>
<td>6</td>
<td>DE</td>
<td>0.323515</td>
<td>0.307036</td>
<td>0.006663</td>
<td>[0.9167, 0.5410, 0.2467, 0.9521]</td>
</tr>
<tr>
<td>6</td>
<td>PSO</td>
<td>0.324055</td>
<td>0.310436</td>
<td>0.009127</td>
<td>[0.9502, 0.5745, 0.2395, 0.9916]</td>
</tr>
<tr>
<td>7</td>
<td>GA</td>
<td>0.322424</td>
<td>0.304251</td>
<td>0.016513</td>
<td>[0.9815, 0.7355, 0.1961, 1.0000]</td>
</tr>
<tr>
<td>7</td>
<td>DE</td>
<td>0.323100</td>
<td>0.307249</td>
<td>0.009793</td>
<td>[0.9692, 0.5269, 0.2312, 1.0000]</td>
</tr>
<tr>
<td>7</td>
<td>PSO</td>
<td>0.332127</td>
<td>0.320177</td>
<td>0.009676</td>
<td>[0.9391, 0.5217, 0.2172, 0.9699]</td>
</tr>
</tbody>
</table>

$P_m = 0.1$. A population of 80 individuals and 50 generations were used. DE was executed with the recombination constant $R = 0.8$ and difference constant $F = 0.6$, a population of 40 individuals and 100 generations. Finally, PSO was executed with the inertia weight $w = 0.721$ and the acceleration coefficients $\varphi_1 = \varphi_2 = 1.193$. A population of 20 particles and 200 generations were used. Therefore, a total of 4000 fitness evaluations were allowed for each algorithm [130].

Figures 3.1 to 3.6 show the transient evolution and phase space portraits for the cases listed in Table 3.2 and those corresponding to the non-optimized coefficients in Table 3.1.
3.3 Maximizing the unpredictability in multi-scroll chaotic oscillators

<table>
<thead>
<tr>
<th>Case</th>
<th>Description</th>
<th>Diagram</th>
</tr>
</thead>
</table>
| (a)  | Non-optimized $x_1$ value vs time | ![Diagram](image1)
| (b)  | Non-optimized $x_1$ vs $x_2$ values | ![Diagram](image2)
| (c)  | GA-optimized $x_1$ value vs time | ![Diagram](image3)
| (d)  | GA-optimized $x_1$ vs $x_2$ values | ![Diagram](image4)
| (e)  | DE-optimized $x_1$ value vs time | ![Diagram](image5)
| (f)  | DE-optimized $x_1$ vs $x_2$ values | ![Diagram](image6)
| (g)  | PSO-optimized $x_1$ value vs time | ![Diagram](image7)
| (h)  | PSO-optimized $x_1$ vs $x_2$ values | ![Diagram](image8)

Figure 3.1: Diagrams of the cases listed in Table 3.2 for a 2-scrolls chaotic oscillator. For each case, the top row and the following ones correspond to the non-optimized and optimized by applying GA, DE, and PSO, respectively.
Figure 3.2: Diagrams of the cases listed in Table 3.2 for a 3-scrolls chaotic oscillator. For each case, the top row and the following ones correspond to the non-optimized and optimized by applying GA, DE, and PSO, respectively.
Figure 3.3: Diagrams of the cases listed in Table 3.2 for a 4-scrolls chaotic oscillator. For each case, the top row and the following ones correspond to the non-optimized and optimized by applying GA, DE, and PSO, respectively.
Figure 3.4: Diagrams of the cases listed in Table 3.2 for a 5-scrolls chaotic oscillator. For each case, the top row and the following ones correspond to the non-optimized and optimized by applying GA, DE, and PSO, respectively.
Figure 3.5: Diagrams of the cases listed in Table 3.2 for a 6-scrolls chaotic oscillator. For each case, the top row and the following ones correspond to the non-optimized and optimized by applying GA, DE, and PSO, respectively.
Figure 3.6: Diagrams of the cases listed in Table 3.2 for a 7-scrolls chaotic oscillator. For each case, the top row and the following ones correspond to the non-optimized and optimized by applying GA, DE, and PSO, respectively.
3.3 Maximizing the unpredictability in multi-scroll chaotic oscillators

As it can be seen, the dynamical behavior of the chaotic system is more complex as MLE increases, because it achieves greater unpredictability.

The Runge-Kutta method of fourth order [1] was used to solve (2.4.1) and also to calculate the Lyapunov exponents. It was coded in C programming language, while the GA, DE and PSO algorithms were coded in MATLAB. The integration step was selected as $t_{\text{step}} = \frac{T_{0}}{50}$. For this type of oscillator, the simulations were executed first for 400 seconds and then for another $500 \times s$ (s =number of scrolls) seconds, where the Lyapunov exponents were measured. Also, the initial condition was set for all the simulations to $x_{0} = [0.1, 0.0, 0.0]^T$. 
Chapter 4

Synchronization

Although there are several synchronization approaches, in this chapter, synchronization is reviewed in the context of identical synchronization.

Two chaotic systems described by the sets of states \( x_1, x_2, \ldots, x_n \) and \( \xi_1, \xi_2, \ldots, \xi_n \) will synchronize if the following limit fulfills

\[
\lim_{t \to \infty} |x(t) - \xi(t)| = 0 \quad (4.0.1)
\]

For any initial conditions \( x(0) \neq \xi(0) \)

Due to the real limitations, a tolerance value is used in practical applications, where there are some other agents like noise, distortion, component mismatching, etc.

\[
|x(t) - \xi(t)| \leq \epsilon \forall t \geq t_f \quad (4.0.2)
\]

where \( \epsilon \) is the allowed tolerance value and \( t_f < \infty \) is assumed. Then the synchronization error is defined as

\[
e(t) = x(t) - \xi(t) \quad (4.0.3)
\]

4.1. Hamiltonian synchronization approach

The synchronization problem can be considered as a particular type of a generic control problem, in which the control objective is to keep the desired chaotic trajectory.

We adopt the synchronization technique already introduced in the seminal article [43], in which it is performed by applying Hamiltonian forms and observer approach.
due to its suitability of being automated [58, 131]. Hamiltonian synchronization technique has been applied to Chua’s circuit in previous works [41, 43, 57]. Their main advantages over other synchronization methods reported in the literature are:

- It can be successfully applied to several well-known chaotic systems.
- It does not require the computations of any Lyapunov exponent.
- It does not require initial conditions belonging to the same basin of attraction.

4.1.1. Generalized Hamiltonian systems

Let us consider the dynamical system

\[ \dot{x} = f(x) \]  

where \( x \in \mathbb{R}^n \) is the state vector and \( f : \mathbb{R}^n \rightarrow \mathbb{R}^n \) is a nonlinear function. In [43], it is reported how the system described by (4.1.1) can be written in the Generalized Hamiltonian canonical form:

\[ \dot{x} = J(x) \frac{\partial H}{\partial x} + S(x) \frac{\partial H}{\partial x} + F(x), \quad x \in \mathbb{R}^n \]  

where \( H(x) \) denotes a smooth energy function, which is globally positive definite in \( \mathbb{R}^n \). The gradient vector of \( H \), denoted by \( \frac{\partial H}{\partial x} \), is assumed to exist everywhere. In this work a quadratic energy function \( H(x) = \frac{1}{2} x^T M x \) with \( M \) being a constant, symmetric positive definite matrix, is used. In this case, \( \frac{\partial H}{\partial x} = M x \). The matrices \( J(x) \) and \( S(x) \) satisfy, for all \( x \in \mathbb{R}^n \), the following properties: \( J(x) + J^T(x) = 0 \) and \( S(x) = S^T(x) \). The vector field \( J(x) \frac{\partial H}{\partial x} \) exhibits the conservative part of the system and it is also referred to as the workless part, or work-less forces of the system, and \( J(x) \) denotes the working or nonconservative part of the system.

For certain systems, \( S(x) \) is negative definite or negative semidefinite. Thus, the vector field is referred to as the dissipative part of the system. If, on the other hand, \( S(x) \) is positive definite, positive semidefinite, or indefinite, it clearly represents the global, semi-global, or local destabilizing part of the system, respectively. In the last case, one can always (although nonuniquely) decompose such an indefinite symmetric matrix into the sum of a symmetric negative semidefinite matrix \( R(x) \) and a symme-
4.1 Hamiltonian synchronization approach

In the context of observer design, a special class of Generalized Hamiltonian forms with output $y(t)$ can be considered, it is given by

$$\dot{x} = J(y) \frac{\partial H}{\partial x} + (I + S) \frac{\partial H}{\partial x} + F(y), \quad x \in \mathbb{R}^n \tag{4.1.3}$$

$$y = C \frac{\partial H}{\partial x}, \quad y \in \mathbb{R}^m$$

where $S$ is a constant symmetric matrix, not necessarily of a definite sign. $I$ is a constant skew symmetric matrix, and $C$ is a constant matrix.

The estimate of the state $x(t)$ can be denoted by $\xi(t)$, and one can consider the Hamiltonian energy function $H(\xi)$ to be the particularization of $H$ in terms of $\xi(t)$. Similarly, $\eta(t)$ denotes the estimated output, computed in terms of $\xi(t)$. The gradient vector $\frac{\partial H(\xi)}{\partial \xi}$ is, naturally, of the form $M\xi$ with $M$ being a constant, symmetric positive definite matrix. A nonlinear state observer for the Generalized Hamiltonian form (4.1.3) is given by

$$\dot{\xi} = J(y) \frac{\partial H}{\partial \xi} + (I + S) \frac{\partial H}{\partial \xi} + F(y) + K(y - \eta), \quad \xi \in \mathbb{R}^n \tag{4.1.4}$$

$$\eta = C \frac{\partial H}{\partial \xi}, \quad \eta \in \mathbb{R}^m$$

where $K$ is the observer gain. The state estimation error, defined as $e(t) = x(t) - \xi(t)$, and the output estimation error, defined as $e_y(t) = y(t) - \eta(t)$, are governed by

$$\dot{e} = J(y) \frac{\partial H}{\partial e} + (I + S - KC) \frac{\partial H}{\partial e}, \quad e \in \mathbb{R}^n \tag{4.1.5}$$

$$e_y = C \frac{\partial H}{\partial e}, \quad e_y \in \mathbb{R}^m$$

where $\frac{\partial H}{\partial e}$ actually stands, with some abuse of notation, for the gradient vector of the modified energy function, $\frac{\partial H(e)}{\partial e} = \frac{\partial H}{\partial x} - \frac{\partial H}{\partial \xi} = M(x - \xi) = Me$. When needed, one can set $I + S = W$. 

---

Synchronization of Chaotic Oscillators Optimized by Applying Evolutionary Algorithms
Definition 1: (Chaotic synchronization) [43] The slave system (nonlinear state observer) (4.1.4) synchronizes with the chaotic master system in Generalized Hamiltonian form (4.1.3), if

\[
\lim_{t \to \infty} \|x(t) - \xi(t)\| = 0 \quad (4.1.6)
\]

no matter which initial conditions \(x(0)\) and \(\xi(0)\) have, where the state estimation error \(e(t) = x(t) - \xi(t)\) corresponds to the synchronization error.

4.2. Synchronization of multi-scroll chaotic attractors

The chaos generator model (4.1.1)-(4.1.3) in Generalized Hamiltonian form, according to (2.3.8) (master model) is given by

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3
\end{bmatrix} = \begin{bmatrix}
0 & \frac{1}{2b} & \frac{1}{2} \\
-\frac{1}{2b} & 0 & 1 \\
-\frac{1}{2} & -1 & 0
\end{bmatrix} \frac{\partial H}{\partial x} + \begin{bmatrix}
0 & \frac{1}{2b} & -\frac{1}{2} \\
\frac{1}{2b} & 0 & 0 \\
-\frac{1}{2} & 0 & -c
\end{bmatrix} \frac{\partial H}{\partial x} + \begin{bmatrix}
0 \\
0 \\
d_1 f(x)
\end{bmatrix} \quad (4.2.1)
\]

Hamiltonian energy function can be described by

\[
H(x) = \frac{1}{2} [ax_1^2 + bx_2^2 + x_3^2] \quad (4.2.2)
\]

and the gradient vector can be described by

\[
\frac{\partial H}{\partial x} = \begin{bmatrix}
a & 0 & 0 \\
0 & b & 0 \\
0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} = \begin{bmatrix}
a x_1 \\
b x_2 \\
x_3
\end{bmatrix}
\]

The destabilizing vector field calls for \(x_1\) and \(x_2\) signals to be used as the outputs, of the master model (4.2.1). In this work, \(y = x_1\) in (4.2.1), is used. The matrices \(C, S,\) and \(I\) are given by
4.2 Synchronization of multi-scroll chaotic attractors

\[
C = \begin{bmatrix} \frac{1}{a} & 0 & 0 \end{bmatrix}
\]

\[
S = \begin{bmatrix}
0 & \frac{1}{2b} & \frac{-1}{2} \\
\frac{1}{2b} & 0 & 0 \\
\frac{-1}{2} & 0 & -c
\end{bmatrix}
\]

\[
I = \begin{bmatrix}
0 & \frac{1}{2b} & \frac{1}{2} \\
\frac{-1}{2} & 0 & 1 \\
\frac{-1}{2} & -1 & 0
\end{bmatrix}
\]

The pair \((C, S)\) is observable. Therefore, the nonlinear state observer for (4.2.1), to be used as the slave model, is designed according to (4.1.4) as

\[
\begin{bmatrix}
\dot{\xi}_1 \\
\dot{\xi}_2 \\
\dot{\xi}_3
\end{bmatrix} = \begin{bmatrix}
0 & \frac{1}{2b} & \frac{1}{2} \\
\frac{-1}{2b} & 0 & 1 \\
\frac{-1}{2} & -1 & 0
\end{bmatrix} \frac{\partial H}{\partial \xi} + \begin{bmatrix}
0 & \frac{1}{2b} & \frac{-1}{2} \\
\frac{1}{2b} & 0 & 0 \\
\frac{-1}{2} & 0 & -c
\end{bmatrix} \frac{\partial H}{\partial \xi} + \ldots
\]

\[
\begin{bmatrix}
\dot{e}_1 \\
\dot{e}_2 \\
\dot{e}_3
\end{bmatrix} = \begin{bmatrix}
0 & \frac{1}{2b} & \frac{1}{2} \\
\frac{-1}{2b} & 0 & 1 \\
\frac{-1}{2} & -1 & 0
\end{bmatrix} \frac{\partial H}{\partial e} + \begin{bmatrix}
0 & \frac{1}{2b} & \frac{-1}{2} \\
\frac{1}{2b} & 0 & 0 \\
\frac{-1}{2} & 0 & -c
\end{bmatrix} \frac{\partial H}{\partial e} + \begin{bmatrix} k_1 \\
1 \\
k_3
\end{bmatrix} e_y
\]

With gain \(k_i, i = 1, 2, 3\) to be selected in order to guarantee asymptotic exponential stability to zero of the state reconstruction error trajectories (i.e., synchronization error \(e(t)\)). From (4.2.1) and (4.2.3) the synchronization error dynamics is governed by [58]

\[
\begin{bmatrix}
\dot{e}_1 \\
\dot{e}_2 \\
\dot{e}_3
\end{bmatrix} = \begin{bmatrix}
0 & \frac{1}{2b} & \frac{1}{2} \\
\frac{-1}{2b} & 0 & 1 \\
\frac{-1}{2} & -1 & 0
\end{bmatrix} \frac{\partial H}{\partial e} + \begin{bmatrix}
0 & \frac{1}{2b} & \frac{-1}{2} \\
\frac{1}{2b} & 0 & 0 \\
\frac{-1}{2} & 0 & -c
\end{bmatrix} \frac{\partial H}{\partial e} + \begin{bmatrix} k_1 \\
k_2 \\
k_3
\end{bmatrix} e_y
\]

By setting \(K = (k_1, k_2, k_3)^T\) with \(k_1 = 2, k_2 = 5, k_3 = 7\), and considering the initial condition \(X(0) = [0, 0, 0.1], \xi(0) = [1, -0.5, 3]\), one can carry out numerical simulations, for example by using ode45 integration algorithm in MATLAB with a full integration of \(T = 2000\) for 4-scrolls and taking into account the coefficients values.
listed in Table 3.2. Figures 4.1 and 4.2 show the state trajectories of the master and slave models described by (4.2.1) and (4.2.3), respectively, and their synchronization. The coincidence of the states is represented by a straight line with a unity-slope (identity function) in the phase plane of each state and the synchronization error is also shown in their transient evolution.

Figure 4.1: 4-scrolls chaotic oscillator master-slave synchronization with fixed constants and equal to 0.7.

4.2.1. Circuit implementation

OpAmp implementation

The proposed circuit realization for the synchronization of multi-scroll chaotic oscillators of the form (2.3.8), by using OpAmps is shown in Fig. 4.3. The vector $K$ in (4.2.3) is the observer gain and it is adjusted by selecting $R_{io}, R_{fo}, R_{ko}$ according to the sufficiency conditions for synchronization [43].

The values of the circuit elements are calculated as shown in [126,131,132]. In this
4.2 Synchronization of multi-scroll chaotic attractors

Figure 4.2: Optimized 4-scrolls chaotic oscillator master-slave synchronization.

Figure 4.2: Optimized 4-scrolls chaotic oscillator master-slave synchronization.

manner, the SPICE simulation of the synchronization response is shown in Figure 4.4. The synchronization error is shown in Figure 4.5 which can be adjusted by tuning the gain of the observer. The coincidence of the states is represented by a straight line with a unity-slope (identity function) in the phase plane of each state.

4.2.2. Experimental synchronization results

The circuit realization of Figure 4.6 was performed by using the commercially available OpAmp TL081. By selecting \( a = 0.99, b = 0.836, c = 0.150, d = 1, k = 1, \alpha = 0.0165, s = 60.606, h_1 \approx 1 \), \( I_{sat} = 100\mu A, R_{ix} = 10K\Omega, C = 330pF, R = 100K\Omega, R_{tran} = 10.54K\Omega, R_{ua} = R_{ib} = R_{ic} = R_{id} = 16K\Omega, R_{fa} = 16K\Omega, R_{fb} = 13.28K\Omega, R_{fc} = 2.4K\Omega, R_{fd} = 16K\Omega, R_{x1} = R_{x2} = R_{x3} = R_{x4} = R_y = 16K\Omega, R_i = 16K\Omega \) and \( V_{sat} = \pm 18V \), the result is \( N=4 \)-scrolls as shown in Figure 4.6. The synchronization result of Figure 4.3 by selecting \( R_{xo} = 16k\Omega, R_{fo} = 48K\Omega \) and \( R_{ko} = 3\Omega \) is shown in Figure 4.6, and the coincidence of the states is represented by
a straight line with slope equal to unity in the phase plane for each state is shown in Figure 4.7.

4.3. OTA circuit implementation

The proposed scheme for the synchronization of multi-scroll chaotic oscillators of the form (2.3.8), by using OTAs is shown in Figure 4.8. The vector $K$ in (4.2.3) is the observer gain and it is adjusted by selecting $gm_{\text{sync}}$ according to the sufficiency
4.3 OTA circuit implementation

By setting \( g_{mf} = g_{mx} = g_{my} = g_{mz} = 200 \mu A/V \), and the values for \( a, b, c, d \) of the cases listed in Table 3.2 for PSO algorithm; the values for \( g_{ma} = a g_{mf}; g_{mb} = b g_{mf}; g_{mc} = c g_{mf}; g_{md} = d_1 g_{mf} \) are determined to generate 2 to 4 scrolls. SPICE simulation of the response of the synchronization system implemented with OTAs is shown in Figures 4.9 to 4.15. The coincidence of the states is represented by a straight line with a unity-slope (identity function) in the phase plane of each state.
4. Synchronization

(a) master circuit  
(b) slave circuit

Figure 4.6: 4-scrolls chaotic oscillator with optimized MLE.

Figure 4.7: Error phase diagram of (a) $X_1$ vs $\xi_1$, (b) $X_2$ vs $\xi_2$ and (c) $X_3$ vs $\xi_3$. 
4.3 OTA CIRCUIT IMPLEMENTATION

Figure 4.8: Circuit implementation for the synchronization using OTAs.
4. Synchronization

Figure 4.9: 2-scrolls chaotic attractor with optimized MLE.

Figure 4.10: Error phase diagram of (a) $X_1$ vs $\xi_1$, (b) $X_2$ vs $\xi_2$ and (c) $X_3$ vs $\xi_3$.

Figure 4.11: 3-scrolls chaotic attractor with optimized MLE.
Figure 4.12: Error phase diagram of (a) $X_1$ vs $\xi_1$, (b) $X_2$ vs $\xi_2$ and (c) $X_3$ vs $\xi_3$.

Figure 4.13: 4-scrolls chaotic attractor with optimized MLE.

Figure 4.14: Error phase diagram of (a) $X_1$ vs $\xi_1$, (b) $X_2$ vs $\xi_2$ and (c) $X_3$ vs $\xi_3$. 

Synchronization of Chaotic Oscillators Optimized by Applying Evolutionary Algorithms
Figure 4.15: 5-scrolls chaotic attractor with optimized MLE.

Figure 4.16: Error phase diagram of (a) $X_1$ vs $\xi_1$, (b) $X_2$ vs $\xi_2$ and (c) $X_3$ vs $\xi_3$. 
Chapter 5

Design of integrated chaotic oscillators

In PWL-based chaotic oscillators, the designs are governed by a set of slopes. This Chapter proposes a procedure to systematically obtain circuit parameters (given a circuit topology) from a nonlinear PWL function. The approach of a saturated nonlinear functions series is done by exploiting the saturation properties of linear amplifiers. Some examples and experiments for the circuit implementation of a multi-scroll oscillator based on saturated functions are reported in [16, 58, 66, 73, 107, 108, 110, 131, 132]. Whereas this method proves to work, the importance of a high speed switching component is highlighted in the generation of PWL functions. An analysis on the electrical requirements of integrated chaotic oscillators is also presented both for the linear part (integrators, amplifiers), and the nonlinear part (PWL function). In this way, a CMOS design is sketched based on a current saturation block, which has the ability to change the saturation level that requires the PWL function. It is worth to mention that the state of the art related to the design of multi-scroll chaotic oscillators, present solutions that are based on using external DC voltage references to shift the levels required to generate a PWL function. This issue seriously limits the reliability of integrated realizations. The proposed design that is introduced in this Thesis was simulated using integrated circuit technology of $0.35\mu m$ from AMS.
5.1. Integrated implementations of chaotic oscillators

As mentioned in the previous chapters, the majority of chaotic oscillator designs have been implemented mainly by using commercially available opamps in the majority of cases. Moreover, the use of practically recent mixed-mode active devices [71], with IC technology allows us to implement integrated systems, as recently demonstrated in [64, 65, 133]. However, still many open problems in the development of integrated chaotic oscillators, are good opportunities for future research. For instance, a brief discussion on the integrated implementations of chaotic oscillators that has been listed in [78, 134], is given below.

Cruz and Chua Design: One of the first monolithic implementations was the Chua’s chaotic oscillator [79]. The fabricated IC required external resistors connected between terminals 1 and 2, and between 5 and 6 of the developed chip shown in Figure 5.1 [79], with the purpose of adjusting the nonlinear function gain. That nonlinear function was generated by using two OTAs, each one with different transconductance values and labeled as A and B, and also each one with different current bias levels. The inductor was designed from the gyrator concept also using OTAs.

Rodríguez and Delgado Design: A similar Chua’s implementation was realized based on the state variables perspective [60] (the normalized system). Arranging identical OTAs in parallel, the authors used gm-C integrators. The nonlinear function was realized by two OTAs connected in parallel, but each one with independent bias control. A simplified version is shown in Figure 5.2

Elwakil et al. Design: The two-scroll version of the original canonical system was also realized by selecting the capacitance or resistances of the loads $Z_1, Z_2$, in Figure...
5.1 Integrated implementations of chaotic oscillators

Figure 5.2: Proposed Chua’s oscillator in [60].

5.3 as described in [87]. Thus allowing having positive and negative integrals. All the system parameters were available since all the loads were externally connected. Using an inverter as comparator made the nonlinear function.

Figure 5.3: Proposed two-scroll oscillator in [87].

Fujiwara et al. Design: These authors proposed a circuit similar to that introduced by Cruz and Chua [79]. A simulated inductor was also used, and transistors were biased to operate in the linear region to implement the resistor [83]. A particular change was introduced in the design of the nonlinear function, which was designed using floating-gate MOS transistors (FGMOS). The FGMOS transistors were exploited as switches in the current path. That way, the nonlinear function approached by a PWL one, allows increasing the number of linear segments by connecting several cells in parallel. The disadvantage of that design is the requirement of external controls to adjust the slope and breakpoints of the voltage-to-current (V-I) characteristic of the PWL function as shown in Figure 5.4. The authors in [83] reported a 3-scroll experimental attractor, obtained from the introduction of the first multi-scroll chaotic oscillator, implemented with IC technology. However, the poor common-mode rejection ratio (CMRR) did not allow proving the generation of more scrolls.

Trejo-Guerra Design: The author introduced a modified Chua’s circuit implemented using unity-gain cells (UGCs), and whose PWL function is a saw-tooth one as shown in Figure 5.5. That work shows the analysis of the dynamical system to ob-
tain the circuit design requirements and proposes an innovative design using FGMOS transistors to implement a voltage-to-current (V-I) PWL function to generate multi-scrolls. The experimental confirmation of that integrated design is given in [65], where the authors introduced for the first time: parameter, technology and process variation analysis. The authors in [65] show the experimental observation of 3 and 5 scrolls from the fabricated IC in technology of $0.5 \mu m$. A center frequency of $3.5 MHz$ was observed by using the internal capacitances, not the external ones already shown in [64,65].

Table 5.1 summarizes some details of these integrated chaotic oscillators, and the new one introduced by Trejo-Guerra and colleagues in [64,65]. Most important is that from this Table one can infer the necessity of developing novel IC designs for chaotic oscillators generating multi-scrolls.
Table 5.1: Chaotic oscillators implemented with CMOS IC technology.

<table>
<thead>
<tr>
<th>Author</th>
<th>Technology</th>
<th>Bias Levels</th>
<th>Center Freq.</th>
<th>PWL function strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cruz &amp; Chua [79]</td>
<td>2 µm</td>
<td>-</td>
<td>160 kHz</td>
<td>Distinct bias currents and saturation</td>
</tr>
<tr>
<td>Rodríguez &amp; Delgado [59, 60]</td>
<td>2.4 µm</td>
<td>±2.5 V</td>
<td>-</td>
<td>Distinct bias currents and saturation</td>
</tr>
<tr>
<td>Elwakil et al. [87]</td>
<td>1.2 µm</td>
<td>±2.5 V</td>
<td>118 kHz</td>
<td>Simple inverter</td>
</tr>
<tr>
<td>Fujiwara et al. [83]</td>
<td>0.35 µm</td>
<td>±1.65 V</td>
<td>7 MHz</td>
<td>Switching currents FGMOS transistors</td>
</tr>
<tr>
<td>Trejo-Guerra et al. [64, 65]</td>
<td>0.5 µm</td>
<td>±2.5 V</td>
<td>3.5 MHz</td>
<td>FGMOS inverter</td>
</tr>
</tbody>
</table>

5.2. Integrated design approach

PWL functions allow obtaining several multi-scroll oscillator schemes. To design those functions with electronic devices, a switching component may be used. Direct applications will achieve stair-like or saw-tooth functions; also, by controlling the biasing of some linear active cells, other PWL functions can be generated. But the implementation of the nonlinear or PWL function is a design challenge because the behavior must be accurate to keep the relations of the dynamical system and its equilibrium points, simultaneously. It means that the design of nonlinear functions has several dimensions of difficulty compared to a simple linear design. The first problem is how to mitigate input and output offset points on PWL segments, and active function blocks performing integration, addition or subtraction operations. The slew rate affects the bandwidth capacity in different manners (according to the specific PWL function), the hysteresis effects may produce more than two different signal paths, and so on.

Two further circuit-level considerations must be taken into account in the integrated circuit design of the schematic shown in Figure 2.22. One is that all the integration nodes are affected by the parasitics present at the input stages of the transconductors. A second consideration is that integrated components suffer severely from uncontrollable process variations (which may be around 25% of the absolute nominal values) due to the statistical deviations in technological parameters, temperature variations, and aging.

5.2.1. Error sources and block requirements

Hardware nonidealities make any physical realization of the schematic in Figure 2.22 deviate from the intended dynamics defined by (2.4.1) and (2.4.2) and may even preclude the appearance of chaotic behavior. Error sources may be dynamic or static.
The former are caused by the reactive behavior of the building blocks. Many of these errors can be grouped as integrator nonidealities, and have a large influence on the characteristic coefficients of the system. Another dynamic error source is due to the nonlinear transconductor, which can produce delays and ripples in the state variable $x_1$ when it crosses from one piece of the characteristics to another. Static deviations are those observable from the DC characteristics of the building blocks and also have a large influence on the dynamic behavior of the system. For instance, real transconductors deviate from the linear behavior, and this gives harmonic distortion, intermodulation, and makes the time constant of the circuit to be a function of the state variables. Another static error source is due to the gradual transition between the segments of the PWL characteristics in a real implementation of the nonlinear transconductor.

From a different perspective, error sources may be classified as deterministic or random. All the error sources referred to so far are deterministic. Attenuation of deterministic errors on the system performance does not require, in general, large device areas, but proper circuit topologies and design strategies. Also, these deviations can be easily included as modifications to the mathematical model of the system, and hence, compensations could be developed if required. On the other hand, mismatch errors are strongly dependent on the area of the devices, and there is no reduction technique other than proper sizing. Their effects can be summarized as system parameter deviations, offset terms in the state equations, and time constant variation. Random errors are of crucial importance during the synthesis route, and their effects are in many cases the limiting factor on the accuracy of the system [59].

In the following, three of the most relevant errors on the implementation of the schematic in Figure 2.22, are discussed, namely: the effect of nonideal integrators, the influence of nonlinear static deviations in real transconductors, and the influence of mismatch between circuit components.

### 5.2.2. Linear design requirements

*Swing.*- Since the IC design is focused to generate multi-scroll attractors, the increase on the number of scrolls on the same dynamic range will eventually cause the loss of the chaotic behavior due to the noise and circuit fault tolerances. The topology selection must be made in order to maximize swing, as well as to provide a
suitable circuit biasing. In the proposed integrated circuit design the dynamic range to be achieved is ±1 V.

*Gain.* Evidently, different gains will affect the final system parameters letting it to loose the chaotic behavior. The proposed IC design has a variable gain, which can be modified by a control signal. This will allow obtaining different values of the coefficients and constants of the integration system.

*Offset.* The use of low offset linear cells is essential. The influence of offset on autonomous systems is a complex issue that has not been addressed. Offset variations may alter the correct operation of the system, impairing the formation of scrolls and even saturate the system. The proposed IC design tries to minimize the offset through proper sizing of transistors.

*Linearity.* Very high linearity of any system term will not be required. However, the proposed design tries to get as linearity to match the ideal model.

*Parameter relation.* This part is a big issue on the design of integrated chaos generators because of the high parameter sensitivity that most autonomous chaotic systems exhibit. A parameter space analysis was implemented as a first glance to locate the appropriate values to optimize MLE of the chaotic oscillator. Considering the values obtained in the optimization chapter, the proposed design must be able to reproduce these coefficients values. It is proposed to have a transconductance as a unitary base and change the values of other gains to achieve the values of the coefficients.

*Bandwidth.* Since the linear part is in charge of the signal integration and other linear operations, the linear circuit bandwidth can be much lower than the nonlinear counterpart. Chaotic oscillators have a wide frequency spectra, which can be scaled by the integrator gain. If the spectrum is known a priory, the active devices bandwidth are set in accordance to this gain. In the proposed integrated design, the capacitor will be externally chosen allowing the spectrum to be scaled. It also may escalate the frequency using the variable control of the integrator’s transconductances.

### 5.2.3. Nonlinear design requirements

*Swing.* The saturated behavior that is typically designed, implies that an active cell may use different biasing than the rest of the circuit, which has to process the signal beyond saturation. In fact, some approaches have suggested this [14,54];
however, the circuit construction becomes very impractical as the number of desired scrolls grows. A typical solution is the use of saturated current-to-current (I-I) nonlinear cells, due to the natural gain of the signal. In this mode, lower currents can be easily handled while the associated voltages remain low and in the same dynamic ranges for all the oscillator.

*Gain.*- If the saturated function approached by a PWL one is considered, the gain is a factor affecting the behavior of the system, because the slope of the nonlinear function is modified. Therefore maintaining a slope $k > 10$ is necessary.

*Bandwidth.*- Among the numerous research on experimental designs, it is well known that autonomous chaotic behavior is severely limited in frequency. The diverse active cells, generally have a very high bandwidth than the observed one in the oscillation spectra. In fact, the delay response exhibited by the active devices is directly degrading the PWL function [109,110], thus changing critically the dynamical system properties. Consider, the time response for the circuit saturations that the PWL function requires. Several nonlinear cells has to surmount large signal changes and also change on/off states continuously as the system solutions change the region of the function behavior.

*Offset.*- This is well known to be one of the main drawbacks in comparator designs; thus, the saturated function output offset is proposed to be minimized to avoid delays in the signal. Furthermore, the nonlinear function may suffer from input offset.

*Modularity.*- Since the change on the number of scrolls is desirable on multi-scroll oscillator circuits and this effect depends directly on the applied nonlinear function; a desirable nonlinear function may be built to grow in a systematic mode. A current-current nonlinear function grows by simply connecting cells in parallel.

*DC references.*- Most of the multi-scroll designs base their nonlinear function on external voltage references. This means that the number of external components grow inadmissibly by extending the number of scrolls. A current shift is proposed in this Thesis to generate the adequate break points depending of the number of scrolls to be generated using the less number of currents biasing.

### 5.2.4. CMOS Gm-C linear building block

Linearization of CMOS differential transconductor characteristics has drawn strong attention in analog circuit design literature [135–143]. The basic objective is to obtain circuit structures whose linearity range is, by construction, larger than
that obtained for the simple differential amplifier. Following the classification scheme in [144], three major approaches can be identified:

1. Ohmic transconductors [145–148]

2. Degenerated differential pairs [142, 149, 150]

3. Square-law transconductors [137, 151, 152]

Ohmic transconductors exploit MOSFET operation in ohmic region and are well suited for high-frequency operation. Figure 5.6(a) illustrates the principle. Assuming that both transistors are perfectly matched and that $V_{DS1} = V_{DS2} \equiv V_{DS} < V_{DS,at}$, it follows that

$$I_{o1} - I_{o2} = (kW/L)V_{id}$$

(5.2.1)

whose voltage-to-current transfer characteristics and transconductance gain can be electrically tuned via $V_{DS}$. Actual implementation of this concept differs in the approach used to maintain drain terminals equipotential, independent of the differential input voltage. A major source of nonlinearity for all of them is due to dependence of parameter $k$ on terminal voltages [153].

Figure 5.6: (a) Concept of linearized ohmic transconductors, and (b) Concept of degenerated differential pair.

Figure 5.6 (b) illustrates the principle of degenerated differential pairs, which achieve large linearity at the cost of large area occupation and small transconductance gain values. The circuit exploits feedback to maintain $V_{GS}$ of the transistors practically constant for large differential input voltage variations. Proper operation requires that sufficiently large feedback is applied, which infers fulfilling $g_m' R >> 1$, where $g_m'$ denotes the transistor transconductance gain. Consequently, it obtains the following:
\[ I_{o1} - I_{o2} = \frac{g_m}{1 - g'_m R} V_{id} \approx R^{-1} V_{id} \] 

(5.2.2)

which shows that the external transconductance gain is determined by the feedback resistor. An excellent comparative review of different degenerated differential pair implementations can be found in [137,142,150].

The last group of linearized transconductors include structures that obtain linear operation by algebraic combination of square-law functions, for instance, the following:

\[ (a + b)^2 - (a - b)^2 = 4ab \] 

(5.2.3)

These structures hence exploit transfer characteristics of MOSFETs operating in strong inversion inside saturation. A great number of structures that follow this general principle have been reported in literature [137,139,140,154]. Figure 5.7 shows examples of some of them. As for ohmic region transconductors, a main limitation of these structures is the dependence of parameter \( k \) on terminal voltages, which produces deviation from the ideal square-law operation.

**Figure 5.7:** Illustrative square-law MOS transconductors. (a) Cross-coupling, (b) Adaptive biasing, (c) Class AB, and (d) Voltage shifting.
In this research different topologies of OTAs were tested. Below, topologies that allow to have a wide input dynamic range and high linearity, besides that the transconductance is as wide as possible, are shown. The basic objective is to obtain a large input voltage dynamic range and a variable transconductance device.

**OTA 1**

Figure 5.8 shows the schematics modeling for the design of a highly linear OTA. The proposed circuit implementation combines a cross-coupled quad cell and a source-coupled differential pair for linearisation [152]. As a result, improved linearity of the developed OTA over the large tuning range is obtained. The three parameters, \( n \), \( d \) and \( p \) enable to control the transfer characteristic of the OTA. One should notice that the parameter \( n \) is independent of the parameters \( d \) and \( p \). This is a significant advantage of the proposed circuit topology since it simplifies the linearization of the real OTA.

**OTA 2**

In [137] a novel circuit technique for realizing linear CMOS transconductance elements was proposed. The circuits discussed have superior linearity and input voltage range compared with the conventional source-coupled differential pair. Figure 5.9 shows the circuit realization of the above scheme. Transistors \( M_1 - M_4 \) form the cross-coupled quad, while \( M_6 - M_9 \) constitute a differential pair. An additional device \( M_5 \) and a current sink \( \alpha i \) are used to level shift the summed drain currents of \( M_1 \) and \( M_2 \) from the node A to the node B. The linearity and input-voltage range of the proposed circuits are superior to the characteristics of the source-coupled differential pair.

**OTA 3**

Figure 5.10 shows the schematic modeling the design of a highly linear and tuneable OTA. The proposed circuit implementation combines an OTA based on 3 current mirrors with degenerated source for linearisation [152]. As a result, improved linearity of the developed OTA over the large tuning range is obtained.

These three architectures were designed using the FRIDGE optimizer [155], with
the following restrictions; maximize input dynamic range, linearity, bandwidth; minimize output offset and area.

FRIDGE is a proprietary tool of the Institute of Microelectronics of Seville for analog circuits optimization and is used to size analog circuits automatically according to design requirements. The optimization process takes place in two stages: in the first, statistical optimization techniques are applied, while deterministic techniques are applied in the second. Given a set of design specifications to be met, it provides a set of design parameters combining optimization and electrical/behavioral simulation. This tool uses a statistical optimization algorithm (based on simulated annealing) as well as a guided algorithm and interacts automatically with the electrical/behavioral simulator [156].
5.2 Integrated design approach

(a) Simplified schematic diagram

(b) Transfer Function

Figure 5.9: OTA 2 behavior.

(a) Simplified schematic diagram

(b) Transfer Function

(c) Transconductance characteristic

Figure 5.10: OTA 3 behavior.
5.3. Synthesis of piecewise-linear functions

Piecewise-linear techniques have been used extensively in circuits and systems theory to model dc nonlinear characteristics of electronic devices [99, 157] and to study a large class of nonlinear resistive networks [158, 159]. In paper [160], Chua and Kang introduced new analytical representations for one dimensional piecewise-linear functions and multidimensional section-wise piecewise-linear functions. These representations led to the possibility of deriving explicit closed form expressions for system parameters and design formulas. They also allowed standard mathematical operations and manipulations to be carried out in various theoretical studies [161].

A very convenient approach to synthesize PWL characteristics is to decompose them into a summation of simpler functions, preferably exploiting intrinsic MOSFET nonlinearities.

5.3.1. CMOS PWL generation by current shaping

Another approach for GmC PWL function generation consists of generating the nonlinear function in current domain and using a front-end quasi-linear transconductance amplifier [162].

Let us first consider unidimensional functions. A first methodology relates to the extension operator concept [163] based on the following expression:

\[ f(x) = Ax + B + \sum_{j=1}^{N} M_j u_p(x - E_j) + \sum_{j=-N}^{-1} M_j u_n(x - E_j) \]  

(5.3.1)

where the functions \( u_p(.) \) and \( u_n(.) \) are defined as

\[
    u_p(x - E_j) = \begin{cases} 
    (x - E_j), & \text{if } x > E_j \\
    0, & \text{otherwise}
    \end{cases}
\]

\[
    u_n(x - E_j) = \begin{cases} 
    0, & \text{if } x > E_j \\
    (x - E_j), & \text{otherwise}
    \end{cases}
\]

(5.3.2)

and where parameters \( E_j \) represent breakpoints, and parameters \( A, B, \) and \( M_j \) can be easily calculated from the original function by inspection as shown in Figure 5.11b.

Another systematic decomposition technique for PWL functions uses radial base
functions \[164\], like

\[
f(x) = \sum_{j=1}^{N-1} f(P_j)\Phi(x, P_j)
\]  

(5.3.3)

where \(f(P_j)\) denotes the function value (interpolation data) at breakpoint \(P_j\), and the base functions \(\Phi(x, P_j)\) have the generic shape represented in Figure 5.11c, which involves two adjacent intervals of the PWL function domain partition. Unlike the extension operator technique, decomposition by base functions is unique and hence
more systematic. In addition, since each interpolation data, \( f(P_j) \), influences only one term in (5.3.3), it is more appealing for programmable implementations over predefined partitions \( (P_{fixed}) \).

The functions \( u_p(.) \) and \( u_n(.) \) required for an extension operator can be implemented by exploiting inherent mirror rectification as illustrated in Figure 5.12. The upper mirror of Figure 5.12(a) becomes cutoff for input currents above \(-I_n\), yielding \( I_{on} = 0 \). Otherwise, the net current entering the input device is scaled at the output, yielding \( I_{on} = P(I_i + I_n) \). The characteristics of Figure 5.12(b) are implemented in this manner. Similar considerations for the bottom mirror of Figure 5.12(a) lead to the characteristics of Figure 5.12(c).

Figure 5.12: Current rectification via current mirror cutoff.

Figure 5.13(b) shows a schematic to implement the PWL extension operator functions using a current switch. Because of the intrinsic class AB operation this structure allows a high operation speed. The current switch rectifier also provides a simple, natural way to implement PWL radial basis functions and the absolute value operator required for canonical PWL multidimensional functions. Corresponding generic schematics are shown in Figure 5.14. Output current \( I_{FWR} \) constitutes the shifted full-wave rectification of the input current \( I_i \), while \( I_{LBF} \) corresponds to a base function for positive interpolation data. Slopes of the base function are given by current mirror gains, while the position and height of the breakpoint are provided by the dc
biasing currents $I_{DFT}$ and $I_{REF}$, respectively. Negative interpolation data are realized by interchanging the functions of up and bottom transistors in Figure 5.14. Due to the integration performed at the input of the current switch, very high resolution and virtually zero current offset are obtained, without relying on precise device matching [60].

Figure 5.13: (a) CMOS current switch. (b) Current rectification via current switching.

Another interesting methodology for piecewise-linear (PWL) approximation using current mode circuits with very simple topologies is by using current mirrors as basic building blocks.

Figure 5.15 shows the four basic building blocks and the transfer characteristic for each block (defined in terms of output current $I_{out}$ vs. input current $I_{in}$) used to implement current mode PWL approximation circuits. These blocks are denoted as $TP(I_1, I_2)$, $TN(I_1, I_2)$, $BP(I_1, I_2)$ and $BN(I_1, I_2)$. They approximate ideal current rectifiers, DC currents $I_1$ and $I_2$ on the input and output sides of each circuit building block are used to shift the rectifying characteristic to any arbitrary position. Broken lines show the transfer characteristics for the case where $I_1$ and $I_2$ are zero. In this case, the breakpoint of the transfer characteristic is at the origin. The polarity of the transistors determines the quadrant where rectification takes place. The slope of the transfer characteristic corresponds to the gain of the mirror, which is determined by ratios of transistor geometries. Current mirrors have negative gain, positive current.
gains (slopes) are obtained by cascading two current mirrors as shown in Figure 5.15. For the PWL approximation, a number of building blocks equal to the number of segments of the PWL transfer characteristic, is used. The individual output currents of all blocks are added in a common (low impedance) load [165]. The input currents of each block are all replicas of the input signal current $I_{in}$. These replicas are easily generated using current mirrors with multiple outputs.

Figure 5.14: Full-wave current rectification and linear base function implementation with positive interpolation data via a current switch rectifier.

Figure 5.15: Basic building blocks for current-mode piecewise-linear approximation.
Class AB configurations can be used to avoid node saturation and improve high frequency performance [153, 166]. The delay required to discharge parasitic capacitances of saturated nodes is the main factor that limits high frequency operation of these circuits [153]. In the class AB current mirror of Figure 5.16 no nodal saturation takes places since there is always a path for the input current to flow. This circuit can be used as a generalized class AB circuit building block. It can generate (unrectified) replicas of the input current signal (output a), and simultaneously positive and negative rectifying outputs (outputs b and c respectively).

![Class AB current amplifier with: nonrectifying, positive rectifying, and negative rectifying outputs.](image)

Figure 5.16: Class AB current amplifier with: nonrectifying, positive rectifying, and negative rectifying outputs.

**SNLF 1**

A current saturation function can be synthesized using the current comparator finite-gain model.

\[
I_o = \frac{\alpha}{2} \left( |I_i + \frac{I_{sat}}{\alpha} \pm I| - |I_i - \frac{I_{sat}}{\alpha} \pm I| \right)
\]  

This expression can be modeled by adding an absolute value function and a shift current \( I \). Figure 5.17 shows an absolute value implementation.
thus, one can join two complementary structures to model the expression (5.3.4).

Hspice DC simulation of this implementation is shown in Figure 5.18.

\[ f(x_1; m) = \sum_{i=0}^{s-2} f_{2i-s+2}(x_1; m, 2i - s + 2), \]

where
5.3 Synthesis of piecewise-linear functions

\[ f_0(x_1; m) = \begin{cases} 
1, & \text{if } x_1 > m \\
\frac{x_1}{m}, & \text{if } |x_1| \leq m \\
-1, & \text{if } x_1 < -m
\end{cases} \]

\[ f_h(x_1; m, h) = \begin{cases} 
2, & \text{if } x_1 > h + m \\
\frac{x-h}{m}, & \text{if } |x_1 - h| \leq m \\
0, & \text{if } x_1 < h - m
\end{cases} \]

and

\[ f_{-h}(x_1; m, -h) = \begin{cases} 
0, & \text{if } x_1 > h + m \\
\frac{x-h}{m}, & \text{if } |x_1 - h| \leq m \\
-2, & \text{if } x_1 < h - m
\end{cases} \]

Using the configuration shown in Figure 5.19, one can generate the saturated functions \( f_0(x_1; m), f_h(x_1; m, h) \) and \( f_{-h}(x_1; m, -h) \). Where the number of basic cells equals the number of scrolls to be generated minus 1. To generate a SNLF \( X \) takes different values in Figure 5.19 to synthesize the required plateaus and slopes.

![Figure 5.19: SNLF basic building block.](image)

Hspice DC simulation of this implementation is shown in Figure 5.20
5. Design of integrated chaotic oscillators

Figure 5.20: SNLF 2 Hspice DC simulation (a) SNLF, and (b) Decomposition with basic building blocks.

SNLF 3

The configuration shown in Figure 5.21 can be used to generate the saturated functions $f_0(x_1; m), f_h(x_1; m, h)$ and $f_{-h}(x_1; m, -h)$. Where the number of basic cells equals the number of scrolls to be generated minus 1. To generate a SNLF $I_{sat}$ takes different values in Figure 5.21 to synthesize the required plateaus and slopes.

Figure 5.21: Schematics to generate a SNLF building block.

Hspice DC simulation of this implementation is shown in Figure 5.22.
An analysis in the time domain is shown in Figure 5.23, where the deterioration in each of the proposed structures by injecting a signal with frequency $F = 1 MHz$, can be appreciated.
The SNLF 1 function presents a significant deterioration in the generation of the absolute value signal in the intersection with \( x = 0 \), this is due to accumulation of charge on the part of the input to the circuit. The SNLF 2 structure, has a DC behavior that fails to generate saturated regions completely flat. While time behavior shows high overshoot.

The structure of function SNLF 3 achieves properly the generation of a saturated function, both in its DC behavior and transient analysis.

Some advantages of the selected circuit that allows implementing a PWL function are: 1) circuits are open loop configurations, so that they are unconditionally stable; 2) the circuits have very good high frequency performance; 3) the simplicity and modularity of the circuits, makes them very appropriate for SNLF implementation; 4) the small number of transistors required for each block; 5) the inherent programmability of the breakpoints of each segment.

5.4. OTAs based multi-scroll oscillator

In all the current literature related to the design of integrated chaotic oscillators, the ones based on PWL functions require the use of DC references to generate the desired shifts to realize the nonlinear function. The central idea on the proposed integrated design of a multi-scroll oscillator is to reduce such explicit dependence. In this manner, the objective of this research is to design a multi-scroll chaotic oscillator that is capable of changing the values of the coefficients to have different MLE values, and it can generate a different number of scrolls through the use of external control signals.

5.4.1. SNLF circuit design

A first diagram block approach to generate several scrolls is shown in Figure 5.24. The proposed diagram block was proven to work as expected generating up to 7-scroll attractors.

The main idea is based on the use of the parallel connection of the basic cell to generate the saturated function (SNLF 3), which is modified by the currents \( I_{\text{off}} \) in and \( I_{\text{off}} \) out to generate the various shifts in the global function. A global power signal \( I_{\text{sat}} \) is required to establish the limits of the saturated regions. In order to
select the number of segments required to generate \( n \)-scrolls, a 2-to-8 bits decoder is implemented, which is responsible for enabling the output switches to connect each segment of the nonlinear function.

The input currents of each block are all replicas of the input signal current \( I_{in} \). In order to generate a variable SNLF depending on the number of scrolls and dynamic range, the blocks \( I_{off\,in} \), \( I_{off\,out} \) and \( I_{sat} \) must be adjustable over a wide range by using an external control signal.

![Figure 5.24: Proposed SNLF Blocks.](image)

The input currents of each block are all replicas of the input signal current \( I_{in} \). These replicas are easily generated using current mirrors with multiple outputs [60]. Class AB current mirror configurations of Figure 5.25 is implemented to improve high frequency performance. It can generate replicas of the input current signal. In the proposed integrated design, in order to generate up to 7 scrolls, 6 replicates of the input current are required.
Replica current circuit design

The transistor sizing of the replication current circuit is shown in Table 5.2.

Table 5.3, shows the main characteristics of the designed circuit.

HSPICE simulations of the replica currents circuit are shown in Figure 5.26 for DC and transient analysis at frequency $F = 5MHz$.

Additionally, Figures 5.27 and 5.28 show a process-voltage-temperature (PVT) analysis to verify the robustness of replica current circuit design with respect to variations in process corners. Hspice simulations over the BSIM3v3 model have been made by using the typical process and the four corners: (NMOS-PMOS) typical-
Table 5.3: Replication current circuit electrical parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gain</td>
<td>1.005</td>
</tr>
<tr>
<td>DR</td>
<td>&gt; ±2</td>
</tr>
<tr>
<td>$R_{in}$</td>
<td>1.168</td>
</tr>
<tr>
<td>$R_{out}$</td>
<td>748.4</td>
</tr>
<tr>
<td>Offset</td>
<td>1.28</td>
</tr>
</tbody>
</table>

Figure 5.26: Replica current circuit Hspice simulation (a) DC analysis, and (b) time analysis.
Table 5.4: Saturated basic cell circuit transistor dimensions.

<table>
<thead>
<tr>
<th>Transistor</th>
<th>Length ($\mu$m)</th>
<th>Width ($\mu$m)</th>
<th>Multiplicity (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1, M_3 - M_4$</td>
<td>0.7</td>
<td>3.5</td>
<td>4</td>
</tr>
<tr>
<td>$M_2$</td>
<td>0.7</td>
<td>3.5</td>
<td>12</td>
</tr>
</tbody>
</table>

typical, fast-fast, fast-slow, slow-fast, slow-slow (TT, FF, FS, SF, SS, respectively) of the actual technology. The temperature has been sweep between $-20^\circ$ C and $100^\circ$ C in steps of 40 degrees.

Figure 5.27: Variations over the process corners (a) Dc analysis, and (b) time analysis.

Saturated basic cell design

The central SNLF generator is based on using saturated current mirrors as basic building blocks as shown in Figure 5.29. A current mode circuit with very simple topologies is used; the circuit is very compact, modular, and have very good high frequency performance.

The programmability of the breakpoints of each segment is performed by adjusting the currents $I_{off}^{in}$, $I_{sat}$ and $I_{off}^{out}$.

It should be noted that to generate a slope $k = 10$ it is necessary to inject an input signal $I'in = 10 \times Iin$.

The transistor sizing of the Saturated basic cell circuit is shown in Table 5.2.

HSPICE simulations of the main Saturated basic cell circuit are shown in Figure 5.30 for DC and transient analysis at frequency $F = 5MHz$. 
5.4 OTAs based multi-scroll oscillator

Figure 5.28: Time variations of the replication current circuit in temperature (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
5. Design of integrated chaotic oscillators

Figure 5.29: Schematic of the basic SNLF building block.

Figure 5.30: Saturated basic cell circuit (a) DC analysis, and (b) time analysis.
5.4 OTAs based multi-scroll oscillator

Table 5.5: Adjustable Ioff in circuit transistor dimensions.

<table>
<thead>
<tr>
<th>Transistor</th>
<th>Length (µm)</th>
<th>Width (µm)</th>
<th>Multiplicity (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_{b1},M_{n1}$</td>
<td>0.7</td>
<td>30</td>
<td>6</td>
</tr>
<tr>
<td>$M_{b2},M_{n2}$</td>
<td>0.7</td>
<td>30</td>
<td>2</td>
</tr>
<tr>
<td>$M_{n3}$</td>
<td>0.7</td>
<td>30.2</td>
<td>6</td>
</tr>
<tr>
<td>$M_{n4}$</td>
<td>0.7</td>
<td>30.2</td>
<td>2</td>
</tr>
<tr>
<td>$M_{p1}$</td>
<td>0.7</td>
<td>90</td>
<td>2</td>
</tr>
<tr>
<td>$M_{p2}$</td>
<td>0.7</td>
<td>90</td>
<td>6</td>
</tr>
<tr>
<td>$M_{p3}$</td>
<td>0.7</td>
<td>92.4</td>
<td>2</td>
</tr>
<tr>
<td>$M_{p4}$</td>
<td>0.7</td>
<td>92.4</td>
<td>6</td>
</tr>
</tbody>
</table>

Additionally, Figures 5.31 and 5.32 show a process-voltage-temperature (PVT) analysis to verify the robustness of the integrated chaotic oscillator with respect to variations in process corners.

![Graph 1](a)

![Graph 1](b)

Figure 5.31: Variations of the Saturated basic cell over the process corners with adjustment for each process (a) DC analysis, and (b) time analysis.

Adjustable Ioff in current circuit design

Figure 5.33 shows the proposed adjustable Ioff in current circuit of Figure 5.29. The transistor sizing of the adjustable Ioff in circuit is shown in Table 5.5. HSPICE simulations of the Ioff in current circuit are shown in Figure 5.34 for DC analysis.

Additionally, Figures 5.35 and 5.36 show a process-voltage-temperature (PVT) analysis to verify the robustness of the integrated chaotic oscillator with respect to variations in process corners.
Figure 5.32: Variations of the Saturated basic cell circuit in temperature (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
5.4 OTAs based multi-scroll oscillator

Figure 5.33: Adjustable Ioff in current circuit.

Figure 5.34: Adjustable Ioff in current circuit DC behavior.
5. Design of integrated chaotic oscillators

Figure 5.35: Variations of the Ioff in current circuit over the process corners without adjustment for each process.

Table 5.6: Adjustable Isat and Ioff out circuit transistor dimensions.

<table>
<thead>
<tr>
<th>Transistor</th>
<th>Length (µm)</th>
<th>Width (µm)</th>
<th>Multiplicity (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>M_{b1}, M_{n1}</td>
<td>0.7</td>
<td>3.35</td>
<td>6</td>
</tr>
<tr>
<td>M_{b2}, M_{n2}</td>
<td>2.5</td>
<td>3.35</td>
<td>2</td>
</tr>
<tr>
<td>M_{n1a}, M_{n2a}</td>
<td>2.5</td>
<td>3.4</td>
<td>6</td>
</tr>
<tr>
<td>M_{n1b}, M_{n2b}</td>
<td>2.5</td>
<td>3.4</td>
<td>2</td>
</tr>
<tr>
<td>M_{n3a}, M_{n2a}</td>
<td>2.5</td>
<td>3.5</td>
<td>6</td>
</tr>
<tr>
<td>M_{n3b}, M_{n2b}</td>
<td>2.5</td>
<td>3.4</td>
<td>2</td>
</tr>
</tbody>
</table>

Adjustable Isat and Ioff out current circuit design

Figure 5.37 shows the proposed adjustable Isat and Ioff out current circuit of Figure 5.29.

The transistor sizing of the adjustable Isat and Ioff out circuit is shown in Table 5.6.

HSPICE simulations of the Ioff in current circuit are shown in Figure 5.38 for DC analysis.

Additionally, Figures 5.39, 5.40 and 5.41 show a process-voltage-temperature (PVT) analysis to verify the robustness of the integrated chaotic oscillator with respect to variations in process corners.
5.4 OTAs based multi-scroll oscillator

Figure 5.36: Variations of the Ioff in current circuit in temperature with adjustment for each process. (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
Figure 5.37: Adjustable Isat and Ioff out current circuit.

Figure 5.38: Adjustable Isat and Ioff out current circuit DC behavior.
5.4 OTAs based multi-scroll oscillator

![Graph1](a)

![Graph1](b)

Figure 5.39: Variations of the Isat and Ioff out current circuit over the process corners without adjustment for each process. (a) Isat current circuit, and (b) Ioff out current circuit.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>o7</th>
<th>o6</th>
<th>o5</th>
<th>o4</th>
<th>o3</th>
<th>o2</th>
<th>o1</th>
<th>o0</th>
<th>scrolls</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>NA</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>NA</td>
</tr>
</tbody>
</table>

Table 5.7: Decoder 2-to-8 bits control.

Decoder design

A 2-to-8 bits digital decoder was implemented in order to select the number of building blocks for the generation of SNLF. Table 5.7 shows the digital control word and the activation outputs.
Figure 5.40: Variations of the Isat out current circuit in temperature with adjustment for each process. (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
Figure 5.41: Variations of the $I_{off}$ output current circuit in temperature with adjustment for each process. (a) TT, (b) SF, (c) FS, (d) SS and (e) FF.
The output functions are described by:

\[ O_0 = A + B + C \]
\[ O_1 = A + B \]
\[ O_2 = A + BC \]
\[ O_3 = A \]
\[ O_4 = AC + AB \]
\[ O_5 = AB \]
\[ O_6 = ABC \]
\[ O_7 = 0 \]

Figure 5.42: Logic gates decoder implementation.

HSPICE simulations of the 2-to-8 bits decoder are shown in Figure 5.43.
Figure 5.43: 2-to-8 bits decoder simulation.
SNLF circuit generator

The whole SNLF circuit generator design is now analyzed from the perspective of process variations. Hspice simulations over the BSIM3v3 model have been made by using the typical process and the four corners: (NMOS-PMOS) typical-typical, fast-fast, fast-slow, slow-fast, slow-slow (TT, FF, FS, SF, SS, respectively) of the actual technology. The temperature has been sweep between $-20^\circ C$ and $100^\circ C$.

A 3 segments SNLF to generate 2-scrolls is realized by setting $A = B = C = -1.65v$, $I_{sat} = I_{off}$ out$= 50\mu A$, $I_{off}$ in $= 0\mu A$. In Figures 5.44 to 5.46 a process-voltage-temperature (PVT) analysis is performed to show the robustness of the integrated SNLF generator with respect to variations in temperature and process corners.

![Graphs](a),(b),(c),(d)

Figure 5.44: 2 scrolls SNLF generator variations over the process corners (a,b) without adjustment; (c,d) with adjustment for each process.
Figure 5.45: DC variations of the 2-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
Figure 5.46: Time variations of the 2-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
A 5 segments SNLF to generate 3-scrolls is realized by setting $A = C = -1.65\, v, B = 1.65\, v$ $I_{sat} = I_{off\, out} = 40\, \mu A$, and $I_{off\, in1,2} = \pm 440\, \mu A$. In Figures 5.47 to 5.49 a process-voltage-temperature (PVT) analysis is performed to show the robustness of the integrated SNLF generator with respect to variations in temperature and process corners.

![Graph A](image)

(a)

![Graph B](image)

(b)

![Graph C](image)

(c)

![Graph D](image)

(d)

Figure 5.47: 3 scrolls SNLF generator variations over the process corners (a,b) without adjustment; (c,d) with adjustment for each process.

A 7 segments SNLF to generate 4-scrolls is realized by setting $A = -1.65\, v, B = C = 1.65\, v, I_{sat} = I_{off\, out} = 30\, \mu A$, $I_{off\, in1,2} = 0\, \mu A$ and $I_{off\, in3,4} = \pm 660\, \mu A$. In Figures 5.50 to 5.52 a process-voltage-temperature (PVT) analysis is performed to show the robustness of the integrated SNLF generator with respect to variations in temperature and process corners.

A 9 segments SNLF to generate 5-scrolls is realized by setting $A = 1.65\, v, B =$
Figure 5.48: DC variations of the 3-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
Figure 5.49: Time variations of the 3-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
5. Design of integrated chaotic oscillators

Figure 5.50: 4 scrolls SNLF generator variations over the process corners (a,b) without adjustment; (c,d) with adjustment for each process.
Synchronization of Chaotic Oscillators Optimized by Applying Evolutionary Algorithms

Figure 5.51: DC variations of the 4-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
Figure 5.52: Time variations of the 4-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS, and (e) FF.
$C = -1.65v$, $I_{sat} = I_{off \; out} = 24\mu A$, $I_{off \; in_{1,2}} = 264\mu A$, and $I_{off \; in_{3,4}} = \pm 792\mu A$. In Figures 5.53 to 5.55 a process-voltage-temperature (PVT) analysis is performed to show the robustness of the integrated SNLF generator with respect to variations in temperature and process corners.

A 11 segments SNLF to generate 6-scrolls is realized by setting $A = C = 1.65v, B = -1.65v$, $I_{sat} = I_{off \; out} = 20\mu A$, $I_{off \; in_{1,2}} = 0\mu A$, $I_{off \; in_{3,4}} = \pm 440\mu A$ and $I_{off \; in_{5,6}} = \pm 880\mu A$. In Figures 5.56 to 5.58 a process-voltage-temperature (PVT) analysis is performed to show the robustness of the integrated SNLF generator with respect to variations in temperature and process corners.

A 13 segments SNLF to generate 7-scrolls is realized by setting $A = B = 1.65v, C = -1.65v$, $I_{sat} = I_{off \; out} = 17.14\mu A$, $I_{off \; in_{1,2}} = 188.54\mu A$, $I_{off \; in_{3,4}} = \pm 880\mu A$. In Figures 5.56 to 5.58 a process-voltage-temperature (PVT) analysis is performed to show the robustness of the integrated SNLF generator with respect to variations in temperature and process corners.
Figure 5.54: DC variations of the 5-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS and (e) FF.
Figure 5.55: Time variations of the 5-scrolls SNLF generator in temperature (a)TT, (b)SF, (c)FS, (d)SS and (e)FF.
Figure 5.56: 6 scrolls SNLF generator variations over the process corners (a,b) without adjustment; (c,d) with adjustment for each process.
Figure 5.57: DC variations of the 6-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS and (e) FF.
Figure 5.58: Time variations of the 6-scrolls SNLF generator in temperature (a)TT, (b)SF, (c)FS, (d)SS and (e)FF.
5.4 OTAs based multi-scroll oscillator

\[ i_{n3,4} = \pm 565.73 \mu A \] and \[ I_{off} i_{5,6} = \pm 942.81 \mu A \]. In Figures 5.59 to 5.61 a process-voltage-temperature (PVT) analysis is performed to show the robustness of the integrated SNLF generator with respect to variations in temperature and process corners.

![Graphs](image)

**Figure 5.59:** 7 scrolls SNLF generator variations over the process corners (a,b) without adjustment; (c,d) with adjustment for each process.

5.4.2. Adjustable $g_m$ OTA design

Figure 5.62 shows the CMOS circuit design of the highly linear and tuneable OTA. The proposed circuit implementation combines an OTA based on 3 current mirrors with degenerated source for linearisation which shows that the external transconductance gain is determined by the feedback resistor. As a result, improved linearity of the developed OTA over the large tuning range is obtained.
Figure 5.60: DC variations of the 7-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS and (e) FF.
Figure 5.61: Time variations of the 7-scrolls SNLF generator in temperature (a) TT, (b) SF, (c) FS, (d) SS and (e) FF.
Figure 5.62: Adjustable gm OTA circuit.

Figure 5.63: CMOS Resistor implementation
Table 5.8: Adjustable gm OTA center in \( gm = 200 \mu A/V \) transistor dimensions.

<table>
<thead>
<tr>
<th>Transistor</th>
<th>Length (( \mu m ))</th>
<th>Width (( \mu m ))</th>
<th>Multiplicity (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M_{bias} )</td>
<td>1.05</td>
<td>6.15</td>
<td>8</td>
</tr>
<tr>
<td>( M_{bn} )</td>
<td>1.05</td>
<td>6.40</td>
<td>16</td>
</tr>
<tr>
<td>( M_{n1}, M_{n2} )</td>
<td>1.05</td>
<td>29.65</td>
<td>8</td>
</tr>
<tr>
<td>( M_{p1}, M_{p2} )</td>
<td>1.05</td>
<td>15.7</td>
<td>8</td>
</tr>
<tr>
<td>( M_{p3} )</td>
<td>1.05</td>
<td>14.8u</td>
<td>8</td>
</tr>
<tr>
<td>( M_{p4} )</td>
<td>1.05</td>
<td>16.8</td>
<td>8</td>
</tr>
<tr>
<td>( M_{n3} )</td>
<td>1.05</td>
<td>6.1u</td>
<td>8</td>
</tr>
<tr>
<td>( M_{n4} )</td>
<td>1.05</td>
<td>6.6</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 5.9: Adjustable gm OTA center in \( gm = 200 \mu A/V \) electrical parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC Gain</td>
<td>27.9-31.99</td>
<td>dB</td>
</tr>
<tr>
<td>Transconductance range</td>
<td>151.23-255.94</td>
<td>( \mu A/V )</td>
</tr>
<tr>
<td>Input dynamic Range</td>
<td>( \pm 1.50-0.993 )</td>
<td>V</td>
</tr>
<tr>
<td>GBW</td>
<td>18.04-29.52</td>
<td>MHz</td>
</tr>
<tr>
<td>Output offset</td>
<td>817.65-250.17</td>
<td>( \mu A )</td>
</tr>
<tr>
<td>Output capacitance</td>
<td>10</td>
<td>pF</td>
</tr>
<tr>
<td>Power consumption @ ( I_{bias} = 100 \mu A )</td>
<td>1.6-1.68</td>
<td>mW</td>
</tr>
</tbody>
</table>

Because the coefficients \( a, b \) and \( d_1 \) are in the range [0.7, 1] and coefficient \( c \) is in the range [0.15, 0.4], then the OTAs for the first case are designed with a central transconductance \( gm = 200 \mu A/V \) and a tuneable range of \( \pm 50 \mu A/V \), The OTA for the coefficient \( c \) is designed with a transconductance \( gm = 50 \mu A/V \) and tuneable range of \( \pm 50 \mu A/V \). As mentioned in the previous section, for a slope \( k \geq 10 \) in the SNLF, the input current to the nonlinear function must be \( I_{in}' = k \times I_{in} \) therefore an OTA for coefficient \( d \) is required with a transconductance \( gm = 2mA/V \) and tuneable range of \( \pm 500 \mu A/V \).

HSPICE simulations of the adjustable gm OTA design center in \( gm = 200 \mu A/V \) are shown in Figure 5.64. The transconductance gain is determined by the feedback resistor and is controlled by an external voltage \( V_c \).

The transistor sizing of the adjustable gm OTA center in \( gm = 200 \mu A/V \) is shown in Table 5.8.

Resistor CMOS transistor dimensions for \( M_{p1}-M_{p4} \) are \( L = 2.1 \mu m \) and \( W = 3.8 \mu m \) and \( M = 4 \).

Table 5.9, shows the main characteristics of the designed circuit.
Figure 5.64: Adjustable gm OTA circuit (a) DC analysis, (b) Time analysis, and (c) transconductance tuning range.
Table 5.10: Resistor voltage control for each process.

<table>
<thead>
<tr>
<th>Process</th>
<th>Vc (V)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TT</td>
<td>-3.89</td>
</tr>
<tr>
<td>SF</td>
<td>-3.46</td>
</tr>
<tr>
<td>FS</td>
<td>-4.18</td>
</tr>
<tr>
<td>SS</td>
<td>-5.19</td>
</tr>
<tr>
<td>FF</td>
<td>-2.84</td>
</tr>
</tbody>
</table>

Table 5.11: Adjustable gm OTA center in $gm = 50 \mu A/V$ transistor dimensions.

<table>
<thead>
<tr>
<th>Transistor</th>
<th>Length ($\mu m$)</th>
<th>Width ($\mu m$)</th>
<th>Multiplicity (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_{bias}$</td>
<td>1.05</td>
<td>6.15</td>
<td>8</td>
</tr>
<tr>
<td>$M_{bn}$</td>
<td>1.05</td>
<td>6.40</td>
<td>16</td>
</tr>
<tr>
<td>$M_{n1}, M_{n2}$</td>
<td>1.05</td>
<td>29.65</td>
<td>8</td>
</tr>
<tr>
<td>$M_{p1}, M_{p2}$</td>
<td>1.05</td>
<td>15.7</td>
<td>8</td>
</tr>
<tr>
<td>$M_{p3}$</td>
<td>1.05</td>
<td>14.8u</td>
<td>8</td>
</tr>
<tr>
<td>$M_{p4}$</td>
<td>1.05</td>
<td>16.8</td>
<td>2</td>
</tr>
<tr>
<td>$M_{n3}$</td>
<td>1.05</td>
<td>6.1u</td>
<td>8</td>
</tr>
<tr>
<td>$M_{n4}$</td>
<td>1.05</td>
<td>6.6</td>
<td>2</td>
</tr>
</tbody>
</table>

The adjustable gm OTA design is now analyzed from the perspective of process variations. Hspice simulations over the BSIM3v3 model have been made by using the typical process and the four corners: (NMOS-PMOS) typical-typical, fast-fast, fast-slow, slow-fast, slow-slow (TT, FF, FS, SF, SS, respectively) of the actual technology. The temperature has been swept between $-20^\circ C$ and $100^\circ C$ in steps of 40 degrees.

In Figures 5.65 to 5.67 a process-voltage-temperature (PVT) analysis is performed to show the robustness of the integrated adjustable OTA design with respect to variations in temperature and process corners. Table 5.10 shows the used voltage control values for each process.

HSPICE simulations of the adjustable gm OTA design center in $gm = 50 \mu A/V$ are shown in Figure 5.68. The transconductance gain is determined by the feedback resistor and is controlled by an external voltage $Vc$.

The transistor sizing of the adjustable gm OTA center in $gm = 50 \mu A/V$ is shown in Table 5.11.

Resistor CMOS transistor dimensions for $M_{p1}-M_{p4}$ are $L = 2.1 \mu m$ and $W = 3.8 \mu m$ and $M = 4$.

Table 5.12 shows the main characteristics of the designed circuit.

In Figures 5.69 to 5.71 a process-voltage-temperature (PVT) analysis is performed...
Figure 5.65: Adjustable gm OTA variations over the process corners (a,b) without adjustment (Vc); (c,d) with adjustment (Vc) for each process.

Table 5.12: Adjustable gm OTA center in \( gm = 50 \mu A/V \) electrical parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC Gain</td>
<td>27.89-31.98</td>
<td>dB</td>
</tr>
<tr>
<td>Transconductance range</td>
<td>37.8-63.98</td>
<td>( \mu A/V )</td>
</tr>
<tr>
<td>Input dynamic Range</td>
<td>( \pm 1.41-0.88 )</td>
<td>V</td>
</tr>
<tr>
<td>GBW</td>
<td>5.55-9.37</td>
<td>MHz</td>
</tr>
<tr>
<td>Output offset</td>
<td>817.65-250.17</td>
<td>( \mu A )</td>
</tr>
<tr>
<td>Output capacitance</td>
<td>10</td>
<td>pF</td>
</tr>
<tr>
<td>Power consumption @ ( I_{bias} = 100 \mu A )</td>
<td>1.4</td>
<td>mW</td>
</tr>
</tbody>
</table>
Figure 5.66: DC variations of the adjustable gm OTA in temperature (a)TT, (b)SF, (c)FS, (d)SS and (e)FF.
Figure 5.67: Time variations of the adjustable gm OTA in temperature (a)TT, (b)SF, (c)FS, (d)SS and (e)FF.
Figure 5.68: Adjustable gm OTA circuit (a) DC analysis, (b) Time analysis, and (c) transconductance tuning range.

Synchronization of Chaotic Oscillators Optimized by Applying Evolutionary Algorithms
to show the robustness of the integrated adjustable gm OTA design with respect to variations in temperature and process corners.

Figure 5.69: Adjustable gm OTA variations over the process corners (a,b) without adjustment; (c,d) with adjustment for each process.
5.4 OTAs based multi-scroll oscillator

Figure 5.70: DC variations of the adjustable gm OTA in temperature (a)TT, (b)SF, (c)FS, (d)SS and (e)FF.
Figure 5.71: Time variations of the adjustable gm OTA in temperature (a) TT, (b) SF, (c) FS, (d) SS and (e) FF.

HSPICE simulations of the adjustable gm OTA design center in $gm = 2mA/V$ are shown in Figure 5.72. The transconductance gain is determined by the feedback...
5.4 OTAs based multi-scroll oscillator

Table 5.13: Adjustable gm OTA center in $gm = 2mA/V$ transistor dimensions.

<table>
<thead>
<tr>
<th>Transistor</th>
<th>Length ($\mu$m)</th>
<th>Width ($\mu$m)</th>
<th>Multiplicity (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_{bias}$</td>
<td>1.05</td>
<td>6.15</td>
<td>8</td>
</tr>
<tr>
<td>$M_n$</td>
<td>1.05</td>
<td>6.40</td>
<td>16</td>
</tr>
<tr>
<td>$M_{n1},M_{n2}$</td>
<td>1.05</td>
<td>29.65</td>
<td>8</td>
</tr>
<tr>
<td>$M_{p1},M_{p2}$</td>
<td>1.05</td>
<td>15.7</td>
<td>8</td>
</tr>
<tr>
<td>$M_p3$</td>
<td>1.05</td>
<td>14.8u</td>
<td>8</td>
</tr>
<tr>
<td>$M_{p4}$</td>
<td>1.05</td>
<td>16.8</td>
<td>80</td>
</tr>
<tr>
<td>$M_{n3}$</td>
<td>1.05</td>
<td>6.1u</td>
<td>8</td>
</tr>
<tr>
<td>$M_{n4}$</td>
<td>1.05</td>
<td>6.6</td>
<td>80</td>
</tr>
</tbody>
</table>

Table 5.14: Adjustable gm OTA center in $gm = 2mA/V$ electrical parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC Gain</td>
<td>27.9-31.99</td>
<td>dB</td>
</tr>
<tr>
<td>Transconductance range</td>
<td>1.51-2.55</td>
<td>mA/V</td>
</tr>
<tr>
<td>Input dynamic Range</td>
<td>± 1.61-2.55</td>
<td>V</td>
</tr>
<tr>
<td>GBW</td>
<td>32.59-45.36</td>
<td>MHz</td>
</tr>
<tr>
<td>Output offset</td>
<td>817.65-45.36</td>
<td>$\mu$A</td>
</tr>
<tr>
<td>Output capacitance</td>
<td>10</td>
<td>pF</td>
</tr>
<tr>
<td>Power consumption @ $I_{bias} = 100\mu A$</td>
<td>7.47-7.92</td>
<td>mW</td>
</tr>
</tbody>
</table>

resistor and is controlled by an external voltage $V_c$.

The transistor sizing of the adjustable gm OTA center in $gm = 2mA/V$ is shown in Table 5.13.

Resistor CMOS transistor dimensions for $M_{p1}-M_{p4}$ are $L = 2.1\mu m$ and $W = 3.8\mu m$ and $M = 4$.

Table 5.14, shows the main characteristics of the designed circuit.

In Figures 5.73 to 5.75 a process-voltage-temperature (PVT) analysis is performed to show the robustness of the integrated adjustable gm OTA design with respect to variations in temperature and process corners.
Figure 5.72: Adjustable gm OTA circuit (a) DC analysis, (b) Time analysis, and (c) transconductance tuning range.
5.4 OTAs based multi-scroll oscillator

Figure 5.73: Adjustable gm OTA variations over the process corners (a,b) without adjustment; (c,d) with adjustment for each process.
Figure 5.74: DC variations of the adjustable gm OTA in temperature (a)TT, (b)SF, (c)FS, (d)SS and (e)FF.
5.4 OTAs based multi-scroll oscillator

Figure 5.75: Time variations of the adjustable gm OTA in temperature (a)TT, (b)SF, (c)FS, (d)SS and (e)FF.
Finally, after the design is verified using HSPICE, the complete layout of the SNLF block diagram for Figure 5.24 is shown in Figure 6.1.

![Proposed SNLF Blocks Layout](image)

In the same way, after the design verification of the OTA is done, the complete layout of the OTAs that will tune the desired coefficients in the OTA-based circuit
oscillator (the linear part) is shown in Figure 6.2.

![OTAs Layout](image)

**Figure 6.2: OTAs Layout.**

### 6.1. Multi-scrolls chaotic oscillator design Layout

The complete layout of the OTA-based multi-scroll chaotic oscillator is shown in Figure 6.3. Dimensions of the silicon area are $900 \times 350 \mu m$. A total of 2005 elements and 175 nodes were required and a total of 21 inputs/outputs were considered. For the linear part they are: $x, y, z, V_a, V_b, V_c, V_d, V_{int}$, and $Ad$. For the non-linear part they are: $A, B, C, V_{bn1}, V_{bn2}, V_{bn3}, V_{bn4}, I_{in}, I_{out}, BIAS_N$, and $V_{bias}$.
Figure 6.3: Multi-scrolls chaotic oscillator layout
Pad Frame 0.35μm design

This technology needs a pad frame that comply with the minimum size required area described by the manufacturer. Figure 6.4 shows the designed pad frame. This design contains protection diode, Vdd, Vss and open contacts to connect the manufacture designs.

Figure 6.4: Pad Frame Layout
6.2. Post-Layout Simulations

The multi-scrolls chaotic oscillator layout was made by using Tanner suite version 16.2. Hspice post-layout simulations resemble the attractor characteristics for 2 to 7-scrolls, as shown by Figures 6.5 to 6.10, respectively. These are in good agreement with the registered macromodel behavior in the chapter describing the OTAs implementation. Similar results are observed in the frequency analysis for all the attractors.

External integration capacitances are used to control the spectra scaling of the system. A $0.5pF$ parasitic capacitance and an inductor $L = 2nH$ has been supposed to exist at the outputs of variables $x, y, z, I_{in}$ and $I_{out}$, which resemble the internal IC parasitic elements of the circuit and the pad frame. A $20pF$ parasitic capacitance associated to the oscilloscope was supposed to exist at the outputs $x, y, z$ variables, and an external integrator capacitance of $C = 30pF$ was used; this corresponds to a $636.62kHz$ dominant frequency. In all cases, the synchronization is good for the state variables $x_1$ and $\xi_1$. This is a good indication for enhancing applications in secure communications, for instance.
Figure 6.5: Hspice post-layout simulation for the synchronization of a 2-scrolls chaotic oscillators whose MLE has been optimized.
6.2 Post-Layout Simulations

Figure 6.6: Hspice post-layout simulation for the synchronization of a 3-scrolls chaotic oscillators whose MLE has been optimized.
Figure 6.7: Hspice post-layout simulation for the synchronization of a 4-scrolls chaotic oscillators whose MLE has been optimized.
Figure 6.8: Hspice post-layout simulation for the synchronization of a 5-scrolls chaotic oscillators whose MLE has been optimized.
Figure 6.9: Hspice post-layout simulation for the synchronization of a 6-scrolls chaotic oscillators whose MLE has been optimized.
Figure 6.10: Hspice post-layout simulation for the synchronization of a 7-scrolls chaotic oscillators whose MLE has been optimized.
Chapter 7

Conclusions

In this Thesis, the simulation and the integrated circuit realization of the optimized multi-scroll chaotic oscillator based on saturated nonlinear function series using OTAs and the synchronization between two optimized oscillators in a master-slave configuration, have been introduced. The synchronization by generalized Hamiltonian forms and observer approach from nonlinear control theory showed good results from MATLAB simulations down to SPICE simulations and post-Layout ones.

As mentioned in the first chapter, recent integrated realizations of chaotic oscillators are based on PWL functions. Their implementations can be performed by using OpAmps, CFOAs and OTAs macro-models.

In addition, it has been shown that the unpredictability of the chaotic behavior can be improved by optimizing the MLE of the chaotic system. Three meta-heuristics were applied to optimize MLE: Genetic Algorithm (GA), Differential Evolution algorithm (DE) and Particle Swarm Optimization (PSO).

As a result, the main contribution of this Thesis is the design of chaotic oscillators using integrated circuit technology, where the amplifiers (OTAs) allow tuning the coefficient values to have high MLE value and to generate from 2 to 7 scrolls.

In this manner, the proposed integrated circuit realization allows one programming the transconductance of the amplifiers to accomplish the required optimized coefficients of the dynamical system and the required analog blocks were designed taking into account the main error sources like offset, swing, linearity, gain and bandwidth. Using CMOS IC technology of 0.35\(\mu m\), the SNLF was designed by using basic current saturated building blocks, allowing to have a high modularity to grow the PWL function of the chaotic system to generate a greater number of scrolls. The proposed saturated nonlinear block, generates 2 to 7-scrolls, which are selected by a 3-bits control word. In addition, the proposed modular model allows to modify the
saturation levels and break points of the saturated nonlinear function using only four variable current sources.

Design simulations were presented before and after the layout parasitic extraction, and additionally, a process-voltage-temperature (PVT) analysis was performed to verify the robustness of the integrated chaotic oscillator with respect to variations in temperature and process corners.

Although the challenge on generating more scrolls and in more directions remains, this Thesis has been organized also to be a guide in the integrated realization of multiscroll chaotic systems that are approached by PWL functions.

Some open research works can be associated to this Thesis. For instance, a comparative study on the relationship between the maximum frequency and the maximum number of generated scrolls is needed to compare with previous work [109]. Another measures to guarantee and quantify chaotic regimen that could be investigated are entropy and Poincaré maps.

From the integrated circuit point of view, the generation of nonlinear functions approached as PWL ones, is a complex problem that has several degrees of freedom (such as the amplitude, offset, slew rate, delay, bandwidth), and thus the need of proposing novel adjustment controls is mandatory.


---

**Synchronization of Chaotic Oscillators Optimized by Applying Evolutionary Algorithms**


Apendix
Appendix

Published Works

A.1. Book Chapters


A.2. International Journals

A. Published Works


A.3. Conference Proceedings

- **V.H. Carbajal-Gómez**, E. Tlelo-Cuautle, R. Trejo-Guerra, J.M. Muñoz-Pacheco, *Simulating the Synchronization of Multi-Scroll Chaotic Oscillators,*