

FACULTAD DE INGENIERÍA
UNIVERSIDAD NACIONAL DE MAR DEL PLATA

**SISTEMAS COMPLEJOS,
RUIDOS DISCRETOS Y SU
IMPLEMENTACIÓN EN FPGA**

TESIS

PARA OBTENER EL TÍTULO DE
DOCTOR EN INGENIERÍA CON ORIENTACIÓN EN ELECTRÓNICA
MAXIMILIANO ANTONELLI

DIRECTORA: LUCIANA DE MICCO
Co-DIRECTORA: HILDA A. LARRONDO

A Sonia y Eduardo, que hicieron la mejor versión de mí que pudieron.

A Lorena, que sigue intentándolo.

A Giuliana y Luca, que les sale sin querer.

Agradecimientos

Cuando se consiguen logros como este uno quiere agradecerle a todo el mundo, la lista es interminable e insólita, así que me voy a olvidar de muchos. No creo que lo noten.

Primero que nada, a mi familia. Desde Lorena que me banca las noches sin dormir, que me levante tratando de no hacer ningún ruido a la madrugada por que “tengo algo dando vueltas en la cabeza que no me deja dormir”, Giuli y Luca, que me obligan a hacer ejercicios de concentración cuando trato de escribir algún código con alguno aupa. Mi vieja, Sonia, de la que heredé esta forma de pensar tan lateral y mi viejo, Eduardo, que me metió de prepo en la cabeza este bicho de querer estudiar, curiosear, saber por qué... males que comparto con mis hermanos Anibal y Adriana que terminaron siendo una compañía en esto de ser un poco raro.

Por supuesto, a mis compañeros de trabajo, principalmente a mis directoras: Luciana, que me obliga a moverme cuando hay que hacer algo que no me gusta, dejándose esa sensación de que si no termino se termina el mundo e Hilda con la que lamentablemente pude compartir pocos años, persona a la que admiro y a la que traté de copiar en esa claridad para plantearse objetivos. Uno termina laburando mucho y feliz cuando estuvo un rato con Hilda. A mis compañeros de Laboratorio, Daniel, Karina, Miguel, Gustavo y Juan, con quienes se puede charlar de cualquier tema con niveles intelectuales insospechados. Hacen que la estadía sea amena.

No quisiera olvidarme que sin este sistema Universitario libre y gratuito no hubiera podido conseguir ninguno de mis logros profesionales. Un sistema en donde cualquier persona puede tener acceso a educación de primera calidad tiene que ser defendido a capa y espada, si es que alguna retribución pidiere. No puedo dejar de agradecer a los que lo pensaron, a los que lo sostienen y a los que lo defienden.

Abstract

Random numbers have been used successfully in a wide variety of applications such as games, cryptography, physical systems modeling, biological systems, etc. These numbers, can be generated from sources of randomness of a physical nature (TRNG) or from algorithmic generators (PRNG).

This thesis focuses on the implementation of RNGs in electronic hardware, particularly in answering two main questions: How do the statistical properties of chaotic systems vary when they are implemented in digital hardware? And, is it possible to implement a physical noise generator in hardware? The first question is directly related to PRNG generators. The second one introduces the possibility of implementing a TRNG (analog) in digital hardware.

When a system is calculated in finite precision, the result of each iteration is replaced by the nearest representable value. Because chaotic systems are inherently sensitive to initial conditions these variations are amplified and the resulting system may be totally different from the original one. The inherent sensitivity to the initial conditions, characteristic of chaotic systems, causes these perturbations to be amplified and the resulting system may be different from the original. The system can become pseudo-chaotic and its properties as Lyapunov exponent, stochasticity, mixing and period are degraded. One contribution of this thesis is the study of the degradation due to the arithmetic precision in a digital electronic system.

On the TRNG side, a ring oscillator (RO) presents phase fluctuations (jitter) that depend on factors such as gradients during the diffusion process in the manufacturing of the integrated circuit, gradients in the working temperature, thermal noise in the semiconductor junctions, etc. In this thesis, jitter is the source of physical randomness used to generate stochastic signals. A method based on differential entropies that allows to extract binary series randomness degree and, therefore, can indicate the level of jitter that it contains was proposed. This method is useful for cataloging a RO as a good RNG or clock generator.

Resumen

Los números aleatorios, han sido utilizados exitosamente en una gran variedad de aplicaciones como juegos, criptografía, modelado de sistemas físicos, sistemas biológicos, etc. Éstos pueden ser generados a partir de fuentes de aleatoriedad de naturaleza física (TRNG) o a partir de generadores algorítmicos (PRNG).

Esta tesis se centra en la implementación en hardware electrónico de RNGs, particularmente se trata de responder dos preguntas principales: ¿Cómo varían las propiedades estadísticas de los sistemas caóticos cuando son implementados en hardware digital? Y ¿Es posible implementar un generador físico de ruido en hardware? La primera pregunta está directamente relacionada con generadores PRNG. La segunda apunta a la posibilidad de implementar un TRNG (analógico) en hardware digital.

Cuando un sistema es calculado en precisión finita, el resultado de cada iteración se sustituye por el valor representable más cercano. La sensibilidad inherente a las condiciones iniciales que presentan los sistemas caóticos hace que estas perturbaciones se vean amplificadas y que el sistema resultante difiera del original. El sistema puede pasar a ser pseudocaótico y sus propiedades como estocasticidad, mezcla, período y exponente de Lyapunov se ven degradadas. Uno de los aportes de esta tesis es el estudio de esta degradación en función de la precisión de la aritmética representada en un sistema electrónico digital.

Por el lado de los TRNG, un oscilador en anillo (RO) presenta fluctuaciones de fase (jitter) que dependen de factores como gradientes durante el proceso de difusión en la fabricación del circuito integrado, gradientes en la temperatura de trabajo, ruido térmico en las junturas semiconductoras, etc. En esta tesis, el jitter es la fuente de aleatoriedad física utilizada para generar señales estocásticas. Se propuso un método basado en entropías diferenciales que permite extraer un valor que indica la aleatoriedad de una serie binaria y, por lo tanto, puede indicar el nivel de jitter que contiene. Este método es útil para catalogar un dato RO como un buen generador de ruido o de señal de reloj.

Índice general

Abstract	V
Resumen	VII
1. Introducción	1
2. Sistemas de Dinámica Compleja	5
2.1. Teoría Cualitativa - Espacio de Fases	7
2.2. Sistemas Caóticos	16
2.3. Mapas Caóticos	23
2.3.1. Mapa Logístico	23
2.3.2. Mapa Tent	24
2.3.3. Mapas Cuadráticos Bidimensionales	25
2.4. El problema de la Aritmética Discreta	27
3. Cuantificadores de Aleatoriedad	29
3.1. Máximo Exponente de Lyapunov	31
3.1.1. Algoritmo Evolutivo para la Búsqueda de Caos	33
3.2. Cuantificadores de la Teoría de la Información	38
3.2.1. Entropía de Shannon y Complejidad Estadística	39
3.2.2. Determinación de la Función Distribución de Probabilidades	41
3.2.3. Planos Doble Entropía y Entropía-Complejidad	45
3.2.4. Entropías Diferenciales	48
3.2.5. Cuantificador de Entropías Implementado en FPGA	51
3.2.6. Dinámica de los ITQs con AWGN y Banda Limitada	58
3.3. Conclusiones	68

4. Generadores de Números Aleatorios Usando Caos	73
4.1. Caos en Redes Neuronales	75
4.1.1. El Modelo de Hopfield	76
4.1.2. Estudio de la RNA en Función de un Parámetro	78
4.2. Cripto-Codificación Caótica Variante en el Tiempo	81
4.2.1. Implementación	84
4.2.2. Resultados	87
4.3. Implementación de Atractor Determinístico - Estocástico	88
4.3.1. Discretización Temporal del Oscilador de Lorenz	90
4.3.2. Discretización de las Variables de Estado	91
4.3.3. Post-procesamiento de Aleatorización	94
4.3.4. Resultados	94
4.4. Mapas Cuadráticos 2-D Implementados en Punto Fijo	97
4.4.1. Resultados	99
4.5. Conclusiones	107
5. Mapas Conmutados en Precisión Finita	109
5.1. Introducción	109
5.2. Herramientas de Análisis Estadístico	110
5.3. Resultados	111
5.3.1. Período T en Función de B	114
5.3.2. Cuantificadores de Mapas Simples	116
5.3.3. Cuantificadores de Mapas Combinados	124
5.4. Conclusiones	127
6. Generadores de TRNG usando ROs en FPGA	133
6.1. Introducción	133
6.2. Determinación del <i>jitter</i> en <i>ROs</i>	135
6.2.1. Resultados	137
6.3. Implementación de TRNG basado en ROs	146
6.3.1. Introducción	146
6.3.2. Implementación en Hardware	147
6.3.3. Resultados	149
6.4. Conclusiones	153

Capítulo 1

Introducción

Los números aleatorios constituyen una de las bases del desarrollo tecnológico, han sido utilizados exitosamente en una gran variedad de aplicaciones como juegos, criptografía [1, 2], modelado de sistemas físicos [3, 4], sistemas biológicos [5], etc. Dado que cada aplicación tiene diferentes requerimientos, existe un amplio abanico de herramientas para analizar las propiedades de estas secuencias. Un ejemplo paradigmático es el generador Marsanne Twister, cuyas secuencias presentan un período extremadamente largo, poseen propiedades estadísticas excelentes, pero son perfectamente predecibles. Como resultado, este generador es inviable en el campo de la criptografía, sin embargo, es el más utilizado para el resto de las aplicaciones. Dentro de esta tesis se utilizan dos enfoques, cuando se quiere saber si un sistema es apto para ser utilizado como generador de números aleatorios para criptografía se utilizan test estadísticos estándar como NIST o DieHard [6]. Cuando lo que se desea es evaluar el grado de aleatoriedad de un sistema se utilizan cuantificadores de la Teoría de la Información para medir la estocasticidad (basados en entropías de valores) [7] y la mezcla (basados en entropías de patrones de orden) de las secuencias generadas [8]. Estas dos entropías son complementarias y cubren los dos principales aspectos a considerar: estocasticidad de los valores generados e independencia estadística de valores consecutivos. Además, también interesa conocer el período y el máximo exponente de Lyapunov [9], para evaluar la caoticidad de los sistemas implementados en hardware.

Los números aleatorios pueden ser generados a partir de fuentes de aleatoriedad de naturaleza física (TRNG) o a partir de generadores algorítmicos (PRNG). Existen de las más variadas técnicas de generación de números aleatorios utilizando TRNGs, siendo el

ruido térmico generado por una resistencia caliente el más conocido. Las propiedades estadísticas de este generador son malas desde el punto de vista estadístico, sin embargo, el modelo de un TRNG nunca es totalmente conocido, por lo que sus secuencias son impredecibles en el corto y/o largo plazo. Por otro lado, los PRNGs pueden clasificarse en dos conjuntos, los basados en algoritmos y los basados en sistemas caóticos, estos últimos son los que se ocupa esta tesis. Los sistemas caóticos poseen dos propiedades que los hacen atractivos para ser utilizados como PRNGs, su dinámica está determinada por un modelo matemático y presentan sensibilidad a las condiciones iniciales. Como consecuencia son sistemas deterministas impredecibles a largo plazo, por lo que pueden generar señales estocásticas [10]. Estos sistemas pueden incluirse en la clase de sistemas estocásticos que se estudian mediante herramientas estadísticas, sin embargo, sus propiedades estadísticas no son óptimas dado que presentan estructuras internas. En consecuencia, es necesario un postprocesamiento de las secuencias generadas para mejorar su aleatoriedad.

Esta tesis se centra en la implementación en hardware electrónico de RNGs, particularmente se trata de responder dos preguntas principales: ¿Cómo varían las propiedades estadísticas de los sistemas caóticos cuando son implementados en hardware digital? y, ¿Es posible implementar un generador físico de ruido en hardware? La primera pregunta está directamente relacionada con generadores PRNG, se propone reemplazar los generadores algorítmicos por sistemas caóticos. Por otro lado, como todo PRNG tiene un modelo, existen propiedades que no puede satisfacer (como impredecibilidad o período infinito). Por lo tanto, es deseable contar con ambos generadores (PRNG y TRNG) en el mismo sistema. Aquí entra en juego la segunda pregunta, que apunta a la posibilidad de implementar un TRNG (analógico) en hardware digital.

Cuando un sistema es calculado en precisión finita, el resultado de cada iteración se sustituye por el valor representable más cercano, lo que desvía su trayectoria de la que tendría utilizando números reales. Frecuentemente, estas desviaciones del valor exacto son tomadas como incertezas en un resultado o como ruido de cuantificación y no presentan mayores inconvenientes para representar un sistema. Estos dos catálogos (ruido o incerteza) tienen validez cuando el sistema que se intenta describir es dinámicamente estable, entonces la cuantificación puede representarse como un ruido superpuesto a una órbita, por ejemplo; o cuando se caracteriza una fuente de ruido, entonces la cuantificación es sólo una incerteza en la medición. Los sistemas caóticos vienen a llenar la brecha existente entre estos dos conjuntos disjuntos (determinísticos vs. estocásticos) y la precisión juega un papel determi-

nante que puede tener varias interpretaciones. Una de ellas es modelar a la cuantificación como una perturbación del sistema. La inherente sensibilidad a las condiciones iniciales que presentan los sistemas caóticos hace que estas perturbaciones se vean amplificadas con cada iteración (vía el máximo exponente de Lyapunov) y el sistema resultante pueda tener poco que ver con el original. Con este panorama, ya no tenemos acceso a los sistemas caóticos ya que estos sólo existen cuando la base numérica puede representar toda la escala real. En el mejor de los casos el sistema pasa a ser pseudocaótico, en donde las propiedades como estocasticidad, mezcla, período y caoticidad se ven degradadas. En este caso, el sistema se mantiene oscilando sobre el mismo atractor que el calculado en números reales. Cuando la aritmética utilizada para calcular cada iteración no es suficiente, el sistema no oscilará y se perderá toda caoticidad. Un ejemplo muy interesante es el mapa Tent, que posee propiedades estadísticas ideales en el campo de los números reales, pero cuando es implementado en cualquier base 2, por preciso que sea, siempre converge a un punto fijo. Por lo que no puede implementarse en ningún dispositivo digital. Uno de los aportes de la tesis es el estudio de esta degradación en función de la precisión de la aritmética representada en un sistema electrónico digital.

Por el lado de los TRNG, el objetivo es implementar un generador que coexista en el mismo dispositivo que un PRNG. Está bien establecido que un oscilador en anillo (RO) presenta fluctuaciones de fase (jitter) que dependen de procesos puramente físicos como gradientes durante el proceso de difusión en la fabricación del circuito integrado, gradientes en la temperatura de trabajo, ruido térmico en las junturas semiconductoras, etc. Ninguna de estas variables pueden ser incluidas en el modelo del oscilador, lo que hace a este sistema muy atractivo para aprovechar esas incertezas. Como los ROs son comúnmente utilizados como generadores de señales de reloj para sincronizar sistemas, el jitter suele ser un problema [11]. Sin embargo, en esta tesis se lo utiliza como la fuente de aleatoriedad física para generar señales estocásticas. Existen diferentes topologías circuitales en donde se mezclan las señales de varios de estos ROs, de donde surge una señal binaria aleatoria [12]. En esta tesis se propuso un método basado en entropías diferenciales que permite extraer un valor que indica el grado de aleatoriedad de esta serie binaria y, por lo tanto, puede indicar el nivel de jitter que contiene. Este método es útil para catalogar un dado RO como bueno para generar ruido o como señal de reloj. Además, se implementó un TRNG basado en ROs mediante la mezcla de varios osciladores.

Esta tesis se centra principalmente en dos aportes de relevancia internacional, en donde

se publicaron los resultados de los dos últimos capítulos. En [13], se exploran técnicas de aleatorización de datos como switching y skipping desde el punto de vista de la implementación digital. Cuando se estudian técnicas de aleatorización, generalmente se pierde de vista la plataforma digital que calcula la salida del PRNG. Además, los mapas caóticos estudiados fueron caracterizados en trabajos anteriores midiendo el período de las secuencias que generan, por lo que el aporte de este trabajo es doble, por un lado, la caracterización desde el punto de vista estadístico y por otro, respecto de la precisión finita en base binaria. Luego, en [14] se propuso un método para medir el jitter basado en cuantificadores de la Teoría de la Información. Los métodos actuales más comunes (sin equipo especial) para medir jitter se basan en valores máximos, es decir, se detecta la desviación máxima medida y se verifica que quede dentro de ciertos márgenes preestablecidos. Como resultado de la técnica propuesta se obtiene un valor que cuantifica el jitter medio, pudiendo conseguir un grado de incertezza en la duración de cada período. Conocer las propiedades del jitter en un oscilador es útil para catalogarlo como buen generador de pulsos de reloj o como buen TRNG.

La organización de esta tesis es la siguiente: El Capítulo 2 es una introducción a los sistemas dinámicos caóticos utilizados a lo largo de la tesis. El Capítulo 3 contiene, por un lado, una introducción a los cuantificadores de aleatoriedad que se utilizan para medir los generadores de números, y por otro, algunos avances en la implementación de estos cuantificadores en hardware electrónico (FPGA). Además, en este capítulo se presentan los resultados publicados en [15], [16], [15] y [17]. El Capítulo 4 presenta avances en generadores de números aleatorios utilizando sistemas caóticos y sus aplicaciones. Se resumen los resultados de [18], [19] y [20]. En el Capítulo 5 se estudia la degradación estadística de los mapas caóticos cuando son implementados en precisión finita. Aquí se muestran los resultados de [13], el cual es uno de los aportes principales de esta tesis. Y por último en el Capítulo 6, primero se propone la utilización de cuantificadores de la teoría de la información para medir la mezcla y estocasticidad de la fuente de incertezas en ROs, que es el otro aporte principal presentado en [14]. Luego se muestran los resultados de la implementación en FPGA de un TRNG utilizando ROs, resultados presentados en [21].

Capítulo 2

Sistemas de Dinámica Compleja

En los últimos años se ha establecido que existen sistemas deterministas que rompen con el preconcepto de que los sistemas físicos pueden clasificarse en dos conjuntos disjuntos: sistemas deterministas y sistemas estocásticos. El concepto antiguo era que un sistema determinista es aquél para el cual conocemos el modelo y por lo tanto es posible predecir con exactitud la evolución de sus variables de estado. Se utilizan en su descripción ecuaciones diferenciales o de recurrencia. Por otra parte un sistema estocástico es aquél para el cual el modelo no se conoce o se lo supone sumamente complejo como para ser obtenido, de modo que se adopta la estrategia de estudiar sus variables de estado en forma estadística. Se utilizan entonces en la descripción ecuaciones diferenciales o de recurrencia estocásticas.

El caos determinista demostró que complejidad en la evolución temporal no es sinónimo de complejidad en el modelo, cuando hay alinealidad: modelos deterministas muy simples originan señales de aspecto estocástico. La sensibilidad a las condiciones iniciales hace que en estos sistemas la predictibilidad sea a corto plazo (luego de un tiempo finito es imposible predecir la evolución) lo que ubica a estos sistemas en una posición intermedia entre determinista y estocástico [10].

Como consecuencia se desarrollaron en los últimos años un número creciente de aplicaciones de los sistemas caóticos, empleándolos principalmente como generadores de ruido controlado [22], generadores de números pseudoaleatorios [23], portadoras de señales [24], sistemas de encriptado [1, 2], etc.

Hoy en día, los sistemas dinámicos son un objeto de estudio interdisciplinario, aunque originalmente fue una rama de la física. Todo comenzó a mediados del 1600, cuando

Newton inventó las ecuaciones diferenciales, descubriendo sus leyes del movimiento de gravedad universal, y las combinó con las leyes de Kepler sobre el movimiento planetario. Específicamente, Newton resolvió el problema de los dos cuerpos (por ejemplo, el sistema tierra-sol).

Subsecuentes generaciones de matemáticos y físicos intentaron extender los métodos analíticos de Newton al problema de los tres cuerpos (por ejemplo, luna-tierra-sol), pero curiosamente para resolver este problema se necesitó mucho más esfuerzo. Luego de décadas, se dieron cuenta de que el problema de los tres cuerpos era esencialmente imposible de resolver, en el sentido de obtener las fórmulas explícitas.

La ruptura vino con el trabajo de Poincaré a finales del 1800. Él introdujo un nuevo punto de vista que enfatizaba las cuestiones cualitativas más que las cuantitativas (por ejemplo, ¿es estable el sistema luna-tierra-sol?). Poincaré desarrolló una poderosa aproximación geométrica que hoy es usada para estudiar sistemas dinámicos y también fue el primero en vislumbrar la posibilidad del caos, en el cual un sistema determinístico exhibe un comportamiento aperiódico que depende sensiblemente de las condiciones iniciales, haciendo así imposible la predicción a largo plazo.

Pero el caos se mantuvo en segundo plano hasta la segunda mitad del 1900, en donde los osciladores no lineales jugaron un rol vital en el desarrollo de tecnologías de radio, radar, lazos de enganche de fase y láser. Por el lado matemático, los osciladores no lineales también estimularon la invención de nuevas técnicas matemáticas. Los métodos geométricos de Poincaré se fueron extendiendo para producir un conocimiento mucho más profundo de la mecánica clásica.

La invención de la computadora por el 1950 fue una línea divisoria en la historia de los sistemas dinámicos. La computadora nos permite experimentar con ecuaciones en una forma que antes era imposible, y así explorar la dinámica los sistemas no lineales de una forma mucho más directa. Estos experimentos llevaron a Lorenz a descubrir en 1963 el movimiento caótico de un atractor extraño, mientras estudiaba un modelo simplificado de la circulación de convección para comprender mejor la notoria impredecibilidad del clima. Lorenz encontró que la solución a sus ecuaciones nunca caían al equilibrio o a un estado periódico. Además, si comenzaba sus simulaciones de dos condiciones iniciales ligeramente diferentes, los comportamientos resultantes pronto serían totalmente diferentes. Como consecuencia de ello, el sistema es inherentemente impredecible, pequeños errores en las mediciones del estado actual de la atmósfera (o cualquier sistema caótico) sería

amplificado rápidamente. Pero Lorenz también mostró que había estructura en el caos, cuando las soluciones fueron dibujadas en tres dimensiones, las soluciones a sus ecuaciones cayeron sobre un conjunto de puntos en forma de mariposa. Él sostuvo que este sistema tenía que ser “un infinito complejo de superficies”, lo que hoy podríamos considerar como un ejemplo de fractal.

El trabajo de Lorenz tuvo un pequeño impacto hasta 1970, los años del boom del caos. Se desarrollaron teorías completamente nuevas basadas en consideraciones sobre atractores caóticos, como turbulencia de fluidos y biología de las poblaciones y se encontraron comportamientos caóticos en reacciones químicas [3], osciladores mecánicos [4], semiconductores [25] y oscilaciones biológicas como el ritmo cardíaco y circadiano [5]. Hoy, la teoría del caos es una herramienta más para el estudio de sistemas dinámicos y los sistemas caóticos son utilizados en una gran cantidad de dispositivos.

En este Capítulo se revisan los conceptos de espacio de fases, pasando por las soluciones típicas de sistemas de ecuaciones diferenciales, para luego poder entrar a la descripción de sistemas caóticos. Primero se abordan los sistemas caóticos continuos con derivada continua y presentamos tres ejemplos clásicos en la literatura. Luego se hace una reseña a los mapas caóticos, en donde presentamos los mapas cuadráticos bidimensionales, los cuales usaremos en algunas secciones subsiguientes.

2.1. Teoría Cualitativa - Espacio de Fases

En algunas aplicaciones puede interesar, más que conocer las soluciones de un sistema, sus propiedades cualitativas tales como la periodicidad, el comportamiento cuando crece la variable independiente (la que generalmente es el tiempo), si es constante, o si se aproxima a una solución conocida, etc. Una herramienta útil en este sentido es el diagrama de fase. El espacio de fase es el lugar geométrico que ocupan las posibles soluciones del sistema de ecuaciones diferenciales, en él se dibujan las trayectorias que son solución a un sistema de ecuaciones. La teoría cualitativa intenta clasificar los sistemas en función del tipo de trayectorias que poseen, en lugar de intentar resolver las ecuaciones diferenciales (EDs).

Se denomina punto crítico de un sistema de ecuaciones diferenciales, al punto del espacio de estados que satisface:

$$X' = 0 \quad (2.1)$$

es el punto del espacio de estados a partir del cual el sistema no evoluciona.

Para sistemas homogéneos de ED lineales, el único punto crítico es el origen de coordenadas. Para sistemas no homogéneos de ED lineales, el punto crítico puede ser cualquier punto del espacio. Para sistemas no lineales, pueden existir varios puntos críticos, o ninguno.

Ahora, supongamos que encontramos todos los puntos críticos de un sistema de ecuaciones. En un entorno reducido de cada uno de ellos podemos linealizar el sistema, de modo que la Ecuación que representa la evolución en ese entorno sea:

$$X' = \mathbb{A} \cdot X \quad (2.2)$$

En donde X es el vector de soluciones y $\mathbb{A} \in \mathbb{R}^{n \times n}$ la matriz de coeficientes. Entonces, pueden calcularse los autovalores del sistema λ_i como:

$$\det(\mathbb{A} - \lambda \mathbb{I}) = 0 \quad (2.3)$$

Además, cada autovalor tiene un autovector asociado $K_i \in \mathbb{R}^{n \times 1}$ que cumple con la Ecuación:

$$\mathbb{A} \cdot K_i = \lambda_i \cdot K_i \quad (2.4)$$

Estos autovectores contienen la información de cuáles serán las direcciones naturales en las que evolucionará el sistema sobre el espacio de fases, mientras que los autovalores indican la dirección y velocidad de convergencia.

Para un sistema planar, pueden darse las siguientes trayectorias respecto de los dos autovalores:

Nodo estable Si ambos autovalores son negativos, la solución se acerca al origen asintóticamente. Las trayectorias de las soluciones son asintóticas a los autovalores de la matriz de coeficientes \mathbb{A} , excepto las soluciones con condiciones iniciales que pertenecen a las direcciones propias, entonces el sistema evoluciona sobre ellas.

Nodo inestable Si ambos autovalores son positivos, la solución se aleja del origen. Las trayectorias de las soluciones son asintóticas a los autovalores de la matriz de coeficientes \mathbb{A} , excepto las soluciones con condiciones iniciales que pertenecen a las direcciones propias, entonces el sistema evoluciona sobre ellas.

Punto silla Si los autovalores tienen signos opuestos, la solución se aleja del origen asintóticamente a uno de los autovectores y se aproxima asintóticamente al otro, excepto

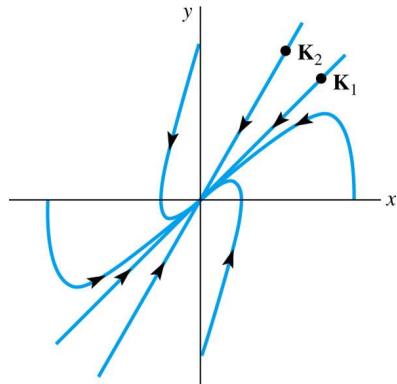


Figura 2.1: Nodo estable.

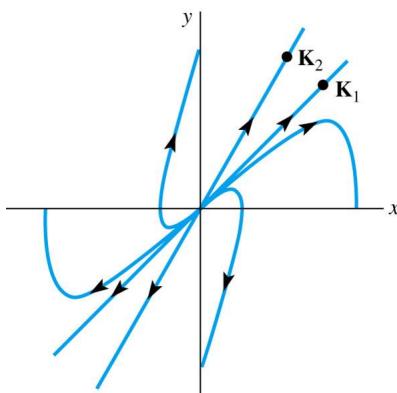


Figura 2.2: Nodo inestable.

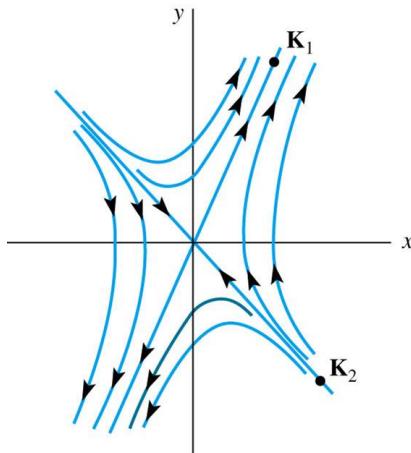


Figura 2.3: Punto silla.

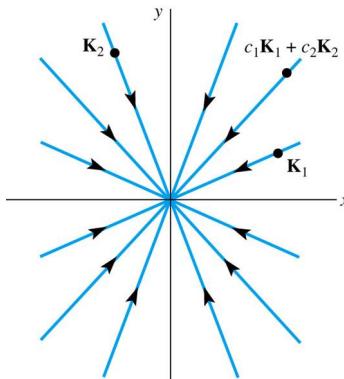


Figura 2.4: Nodo estable degenerado con autovalores negativos.

las soluciones con condiciones iniciales que pertenecen a las direcciones propias, entonces el sistema evoluciona sobre ellas.

Nodos degenerados Aparecen en los casos en que los autovalores o autovectores sean iguales. Para iguales autovalores, pueden generarse todas las trayectorias radiales por combinación lineal de los autovectores. La forma de la solución será

$$X(t) = (c_1 K_1 + c_2 + K_2) e^{\lambda t}$$

Con c_i constantes, K_i los autovectores y λ un autovalor. Si los autovalores son negativos, la solución se acerca al origen en forma radial y el nodo resulta ser estable, de lo contrario será inestable. Si además tenemos iguales autovectores, la forma de la

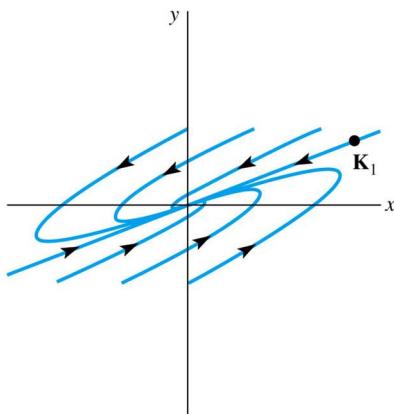


Figura 2.5: Nodo estable degenerado con autovalores negativos y un solo autovector.

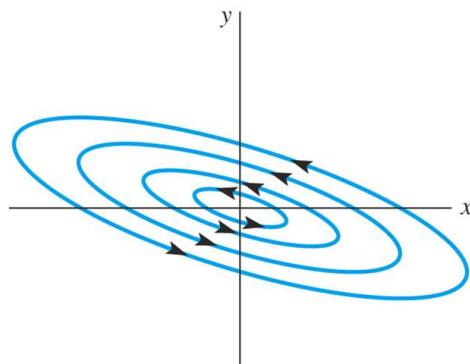


Figura 2.6: Centro.

solución sería

$$X(t) = (c_1 K + tc_2 K + c_2 P) e^{\lambda t}$$

Si los autovalores son negativos, la solución se acerca al origen y el nodo resulta ser estable, de lo contrario será inestable.

Centro Si los autovalores son imaginarios puros, la solución describe elipses concéntricas que pasan por el valor inicial.

Foco estable Si los autovalores son complejos con parte real negativa, la solución es una combinación de los casos anteriores. Será periódica conforme su parte imaginaria y se aproximará a cero según su parte real.

Foco inestable Si los autovalores son complejos, la solución será periódica conforme su parte imaginaria y tenderá a infinito según su parte real.

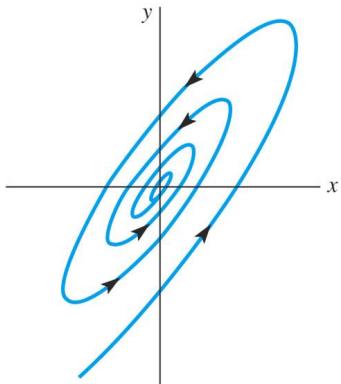


Figura 2.7: Foco estable.

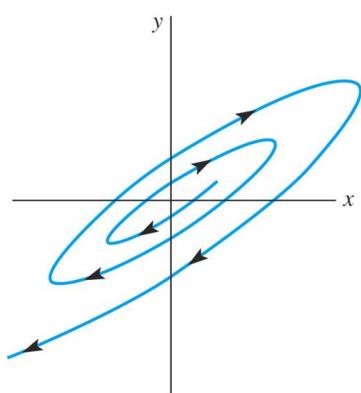


Figura 2.8: Foco inestable.

Para cualquier sistema de ED, primero se deben hallar todos los puntos críticos del sistema. Luego, se linealiza en torno a cada uno mediante el primer término de la serie de Taylor obteniendo tantos sistemas de ecuaciones como puntos críticos tenga el sistema. Estos sistemas son válidos en un entorno suficientemente pequeño del punto crítico.

Por ejemplo, supongamos que se necesita trazar el diagrama de fase para el péndulo físico de la Figura 2.9.

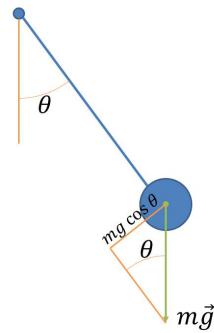


Figura 2.9: Péndulo físico ideal.

Primero hallamos la ecuación de la aceleración, según la Figura, la componente tangencial de la gravedad es la que acelera al cuerpo. Tomando como referencia positiva el sentido dextrógiro, queda

$$a = mg \cos \theta - \mu v$$

en donde μ es el coeficiente de roce viscoso con el aire y v la velocidad. Pero como nos interesa la aceleración angular α y la velocidad angular ω

$$\alpha = a/l; \quad \omega = v/l$$

siendo l la longitud de la cuerda.

La aceleración angular α es la derivada de la velocidad angular ω que a su vez es la derivada del ángulo θ

$$\alpha = \omega' = \theta''$$

entonces el sistema de ecuaciones queda

$$\begin{cases} \theta' = \omega \\ \omega' = \frac{mg}{l} \cos \theta - \frac{\mu}{l} \omega \end{cases}$$

Una vez planteado el sistema de ecuaciones, el primer paso es hallar los puntos críticos del sistema.

$$\begin{cases} 0 = \omega \\ 0 = \frac{mg}{l} \cos \theta - \frac{\mu}{l} \omega \end{cases} \rightarrow \begin{cases} 0 = \omega \\ \theta = (2n-1) \frac{\pi}{2} \quad n \notin \mathbb{Z} \end{cases}$$

Ahora estamos en condiciones de linealizar el sistema en torno de los puntos críticos.

$$\begin{cases} \theta' = \omega \\ \omega' = \left. \frac{\partial(\frac{mg}{l} \cos \theta)}{\partial \theta} \right|_{\theta=(2n-1)\frac{\pi}{2}} \theta - \frac{\mu}{l} \omega \end{cases} = \begin{cases} \theta' = \omega \\ \omega' = -\frac{mg}{l} \sin(\theta) \Big|_{\theta=(2n-1)\frac{\pi}{2}} \theta - \frac{\mu}{l} \omega \end{cases}$$

Según n sea par (incluyendo el cero) o impar, la ecuación lineal que representa al sistema será diferente.

Si n es par o cero

$$\begin{cases} \theta' = \omega \\ \omega' = \frac{mg}{l} \theta - \frac{\mu}{l} \omega \end{cases} \rightarrow A = \begin{pmatrix} 0 & 1 \\ \frac{mg}{l} & -\frac{\mu}{l} \end{pmatrix}$$

$$\det(A - \lambda I) = \begin{vmatrix} -\lambda & 1 \\ \frac{mg}{l} & -\frac{\mu}{l} - \lambda \end{vmatrix} = \lambda^2 + \frac{\mu}{l} \lambda - \frac{mg}{l}$$

los autovalores son reales y de distinto signo (punto silla).

Si n es impar

$$\begin{cases} \theta' = \omega \\ \omega' = -\frac{mg}{l} \theta - \frac{\mu}{l} \omega \end{cases} \rightarrow A = \begin{pmatrix} -\lambda & 1 \\ -\frac{mg}{l} & -\frac{\mu}{l} - \lambda \end{pmatrix} = \lambda^2 + \frac{\mu}{l} \lambda + \frac{mg}{l}$$

los autovalores son complejos conjugados con parte real negativa (foco estable).

La resolución numérica en Matlab (Figura 2.10, muestra las soluciones al sistema alineal en el espacio de estados. Puede verse que, en los puntos críticos, la solución se aproxima a un foco o a un punto silla, según el valor de n .

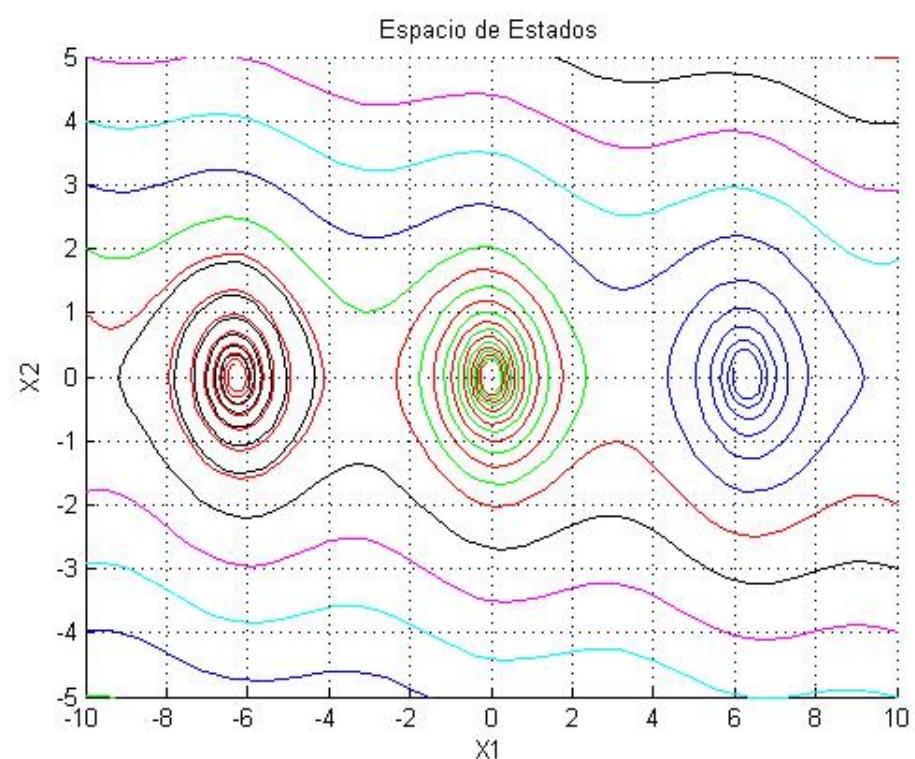


Figura 2.10: Diagrama de fase del péndulo físico real.

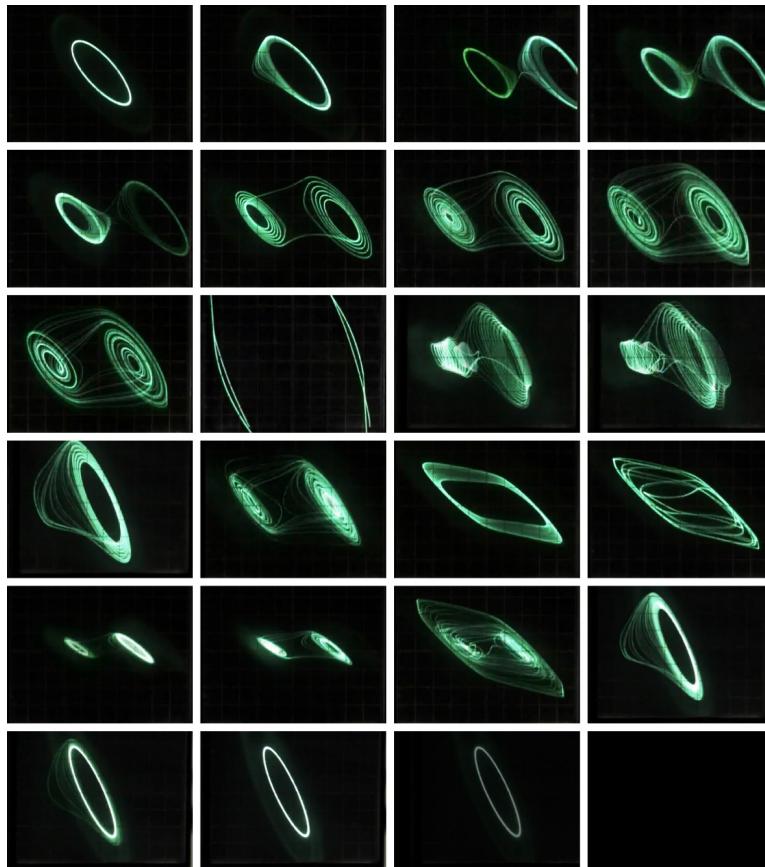


Figura 2.11: Salidas de tensión de dos variables de un circuito oscilador.

2.2. Sistemas Caóticos

Cuando se miden variables físicas, no es muy extraño encontrar que el plano de fases tiene un comportamiento similar al de la Figura 2.11. Puede verse que las trayectorias se cortan en el plano de fase, lo que indica que el sistema tiene un orden mayor que dos. Otra observación que se puede hacer es que las trayectorias no se repiten, es decir que, aunque la oscilación persiste, no aparece un ciclo con trayectoria definida. Esta última propiedad clasifica al sistema que se está midiendo como caótico.

El teorema de existencia y unicidad de las soluciones a un sistema de ecuaciones garantiza que, si una función f es continuamente diferenciable, los campos vectoriales sobre el espacio de fases son suaves y cada punto de este espacio tiene solución única. La existencia de este teorema tiene un corolario importante: trayectorias diferentes nunca se intersectan. Como consecuencia de esto, las trayectorias sobre el plano de fases quedan

restringidas a: un nodo estable o inestable, centro, foco estable o inestable y puerto. Entonces queda claro que un sistema continuo con derivada continua debe tener dimensión mayor o igual a tres para que pueda ser caótico, además, la trayectoria en el espacio de fases debe ocupar un dominio restringido.

Para describir analíticamente el comportamiento de este tipo de sistemas en torno a ciertos puntos de interés podemos hacer uso del álgebra lineal. Lo que sigue son tres ejemplos de aplicaciones para atractores caóticos.

El primer y más común ejemplo es el sistema de Lorenz, que está descrito por el siguiente sistema de ecuaciones diferenciales [26]:

$$\begin{cases} x'_1 &= \sigma(x_1 - x_2) \\ x'_2 &= -x_1x_3 + \rho x_3 - x_2 \\ x'_3 &= x_1x_2 - \beta x_3 \end{cases} \quad (2.5)$$

En donde σ , ρ y β son parámetros del sistema, la dinámica será caótica según que combinación de estos valores elijamos.

El sistema de Lorenz tiene tres puntos de equilibrio, E , E^+ y E^- : el primer punto de equilibrio E está situado en el origen $(0,0,0)$ y los otros dos tienen respectivamente como coordenadas,

$$(x_1^\pm, x_2^\pm, x_3) = (\pm\sqrt{\beta(\rho-1)}, \pm\sqrt{\beta(\rho-1)}, \rho-1)$$

Un comportamiento físico interesante de este sistema ocurre cuando variamos el parámetro de control ρ . Cuando $\rho < 1$, todas las órbitas del campo vectorial dado por la Ecuación 2.5 tienden al punto fijo situado en el origen. A medida que se va incrementando más allá de la unidad, el origen pasa a ser inestable dando lugar a dos puntos fijos, estables, y simétricos E^+ y E^- . Para todo $\rho < 1$, la geometría del comportamiento asintótico del sistema es la misma ya que todas las condiciones iniciales tienden al origen E .

Para $\rho > 1$, se observan dos comportamientos. El primero, asociado a valores de ρ menores que un cierto valor de umbral $\rho_u = (\sigma(\sigma+\beta)+3\sigma)/(\sigma-\beta-1)$, valor para el cual los puntos de equilibrio E^+ y E^- pierden su estabilidad. Dentro de este rango de valores del parámetro todas las órbitas terminan en uno de los dos puntos de equilibrio dependiendo de las condiciones iniciales.

Cuando $\rho > \rho_u$, la situación cambia drásticamente. Los dos puntos fijos pasan a ser inestables y nuevos comportamientos pueden surgir. Para estudiar estos comportamientos se

considera el análisis dinámico del sistema de Lorenz 2.5. La linealización en la proximidad del origen nos proporciona los siguientes autovalores:

$$\lambda = -\beta; \lambda_{\pm} = \frac{1}{2} \left[-(\sigma + 1) \pm \sqrt{(\sigma + 1)^2 + 4\sigma(\rho - 1)} \right]$$

asociados a la matriz Jacobiana:

$$\begin{pmatrix} -\sigma & \sigma & 0 \\ \rho & -1 & 0 \\ 0 & 0 & -\beta \end{pmatrix}$$

Los autovalores λ y λ_- son siempre negativos; el autovalor λ_+ cambia de negativo a positivo cuando ρ pasa por el valor 1.

De modo similar, la linealización del sistema 2.5 en la proximidad del punto de equilibrio E^+ nos proporciona los siguientes autovalores:

$$\lambda^3 + \lambda^2(\sigma + \beta + 1) + \lambda\beta(\sigma + \rho) + 2\sigma\beta(\rho - 1) = 0$$

asociados a la matriz Jacobiana:

$$\begin{pmatrix} -\sigma & \sigma & 0 \\ \rho - x_3 & -1 & -x_1 \\ x_2 & x_1 & -\beta \end{pmatrix}$$

tiene un autovalor real negativo λ combinado con dos autovalores imaginarios puros, $\lambda_{\pm} = \pm j\alpha$, si $\rho < \rho_h$. La linealización del sistema 2.5 en la proximidad del punto de equilibrio E^+ es un problema simétrico a este.

En la Figura 2.12 puede verse la evolución de las soluciones al sistema de Lorenz.

El sistema de Rössler está descrito por el siguiente sistema de ecuaciones diferenciales

$$\begin{cases} x'_1 &= x_2 - x_3 \\ x'_2 &= x_1 x_3 + ax_2 \\ x'_3 &= b + (x_1 + c)x_3 \end{cases} \quad (2.6)$$

en donde a , b y c son parámetros del sistema.

El sistema de Rössler tiene dos puntos de equilibrio, E^+ y E^- que existen sólo cuando

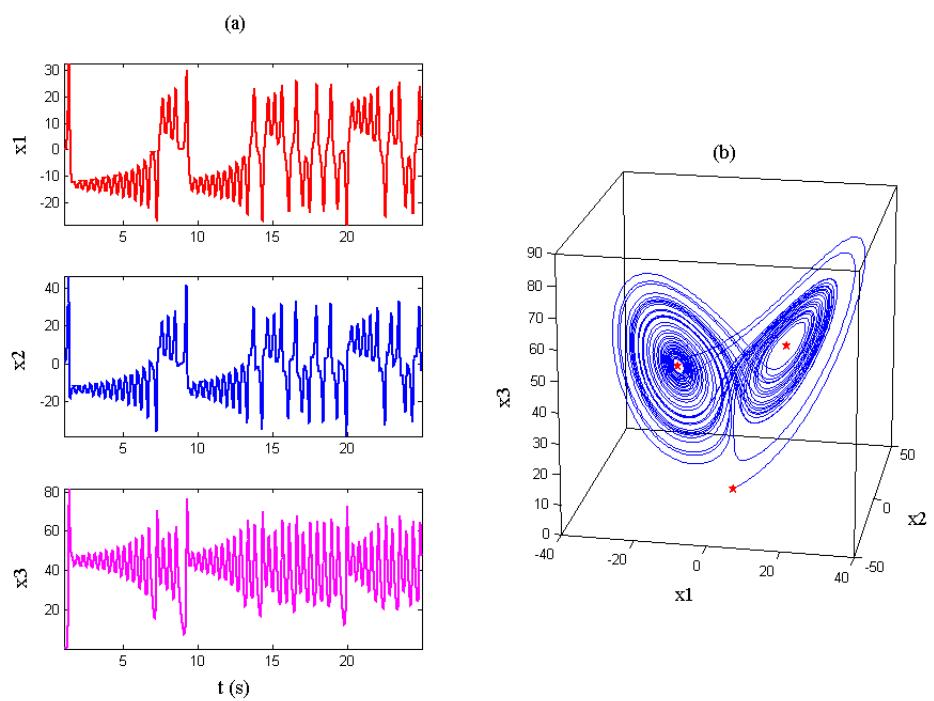


Figura 2.12: (a) Evolución temporal de las tres variables del sistema de Lorenz. (b) Disposición de sus puntos de equilibrio (estrellas rojas) con respecto a su atractor en el espacio de estados.

$\Delta = c^2 - 4ab > 0$ y cuyas coordenadas están dadas respectivamente por,

$$(x_1^\pm, x_2^\pm, x_3^\pm) = \left(\frac{1}{2}(c \pm \sqrt{\Delta}), -\frac{1}{2a}(c \pm \sqrt{\Delta}), \frac{1}{2a}(c \pm \sqrt{\Delta}) \right)$$

La linealización del sistema 2.6 en la proximidad de los puntos de equilibrio proporciona los siguientes autovalores:

$$a\lambda^3 - \lambda^2 a(x_1 - c) - \lambda(x_3 - 1) + ax_3 + (x_1 - c) = 0$$

asociados a la matriz Jacobiana

$$\begin{pmatrix} 0 & 1 & -1 \\ 1 & a & 0 \\ x_3^\pm & 0 & x_1^\pm - c \end{pmatrix}$$

de donde se deduce que fijando los parámetros a y b y variando c nos encontramos con dos escenarios diferentes, podemos tener un autovalor real negativo y dos complejos conjugados con parte real positiva, o un autovalor real negativo y dos complejos conjugados con parte real negativa. Para valores pequeños de c , el atractor de Rössler consiste en una órbita periódica o ciclo límite que tiene un sólo mínimo local. A medida que vamos incrementando el parámetro c , el ciclo límite va duplicando su período y como consecuencia, sus mínimos locales hasta alcanzar un límite en el cual las trayectorias nunca se repiten, lo que corresponde al atractor caótico de Rössler.

En la Figura 2.13 pueden verse la evolución de las variables x_1 y x_2 en el sistema de Rössler para distintos valores del parámetro c . En la Figura 2.14, se puede ver la evolución de las tres variables para un parámetro fijo.

El sistema de Chua está descrito por el siguiente sistema de ecuaciones diferenciales:

$$\begin{cases} x'_1 &= \alpha x_2 - \alpha x_1^3 - \alpha c x_1 \\ x'_2 &= x_1 + x_3 - x_2 \\ x'_3 &= -\beta x_2 \end{cases} \quad (2.7)$$

Este sistema tiene tres puntos de equilibrio, E , E^+ y E^- : el primer punto de equilibrio E está situado en el origen $(0,0,0)$ y los otros dos tienen respectivamente como coordenadas,

$$(x_1^\pm, x_2^\pm, x_3) = (\pm\sqrt{-c}, 0, \pm\sqrt{-c})$$

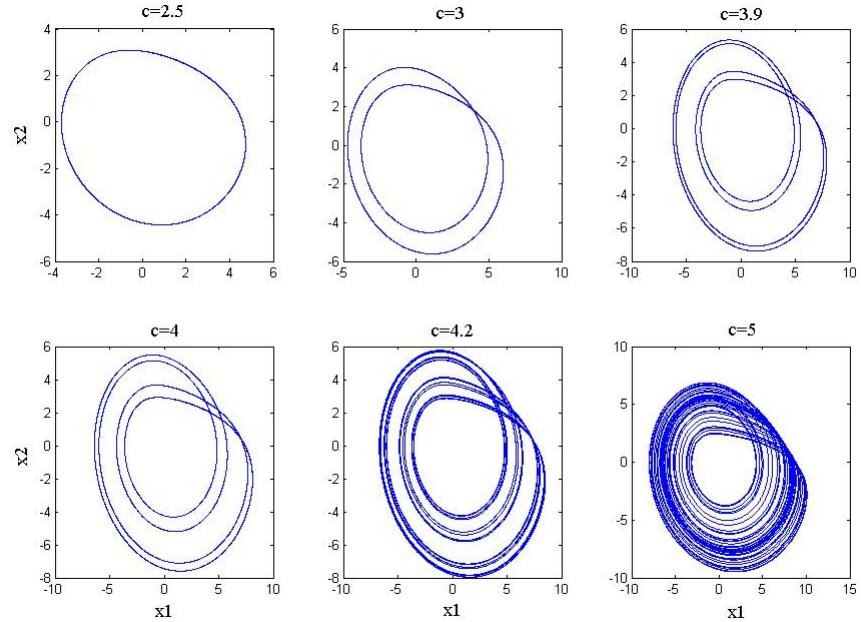


Figura 2.13: Proyecciones del atractor de Rössler en el plano para diferentes valores del parámetro c .

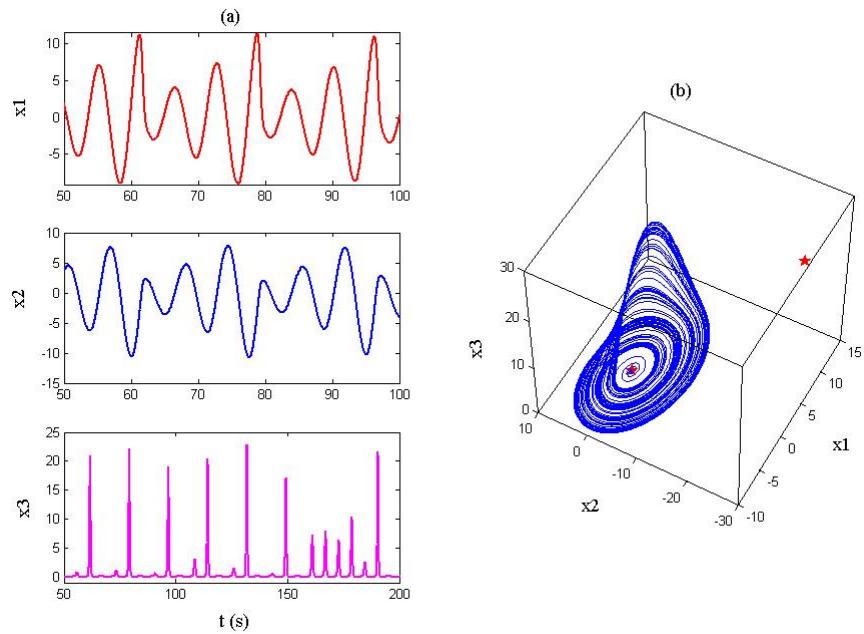


Figura 2.14: (a) Evolución temporal de las tres variables del sistema de Rössler. (b) Disposición de sus puntos de equilibrio (estrellas rojas) con respecto a su atractor en el espacio de estados.

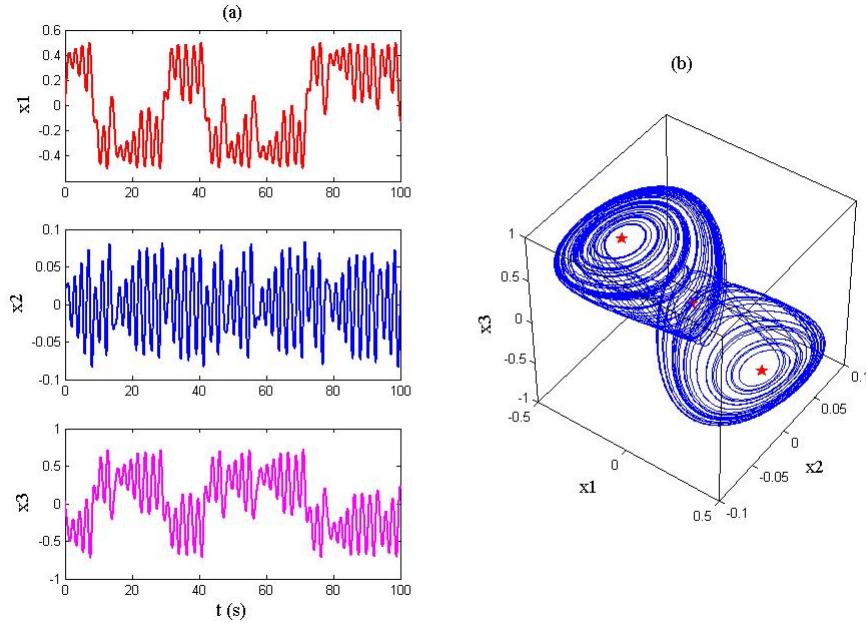


Figura 2.15: (a) Evolución temporal de las tres variables del sistema de Chua. (b) Disposición de sus puntos de equilibrio (estrellas rojas) con respecto a su atractor en el espacio de estados.

Los puntos de equilibrio existen sólo para valores positivos del parámetro c .

La linealización del sistema 2.7 en la proximidad de los puntos de equilibrio proporciona los siguientes autovalores:

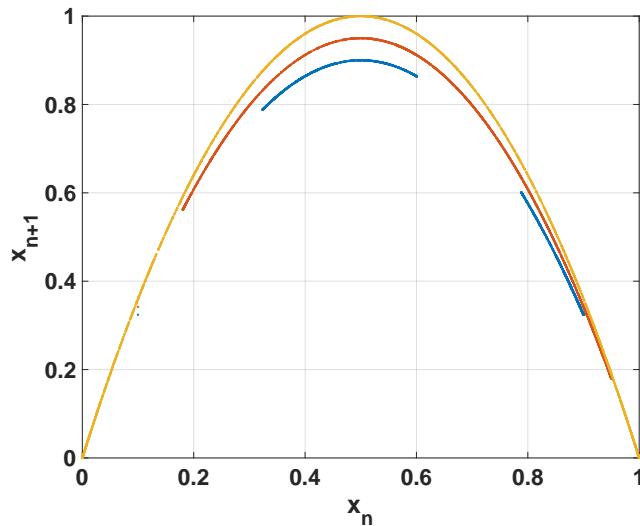
$$\lambda^3 + \lambda^2 \alpha(\alpha c + 1) + \lambda(\alpha c - \alpha + \beta) + \alpha x_3 + \alpha \beta c = 0$$

asociados a la matriz Jacobiana

$$\begin{pmatrix} -\alpha c & \alpha & 0 \\ 1 & -1 & 1 \\ 0 & -\beta & 0 \end{pmatrix}$$

donde podemos detectar un punto de bifurcación para $\alpha = 0$ en el cual el autovalor real negativo pasa a ser positivo provocando la inestabilidad del punto de equilibrio E que permanece inestable para una amplia gama de valores del parámetro α .

En la Figura 2.15, puede verse la evolución de las tres variables de estado para este sistema.

Figura 2.16: Mapa Logístico para tres parámetros distintos, $r = 2, 3, 4$

2.3. Mapas Caóticos

Los mapas son una relación de recurrencias, cuya salida puede presentar un comportamiento caótico. Dentro de la gran familia de mapas unidimensionales destacamos los mapas Tent y Logístico, que son utilizados posteriormente en esta tesis.

2.3.1. Mapa Logístico

El mapa Logístico es un claro ejemplo de cómo a partir de ecuaciones dinámicas lineales muy simples puede surgir un comportamiento caótico complejo. Su representación matemática es una relación de recurrencia polinomial de grado 2:

$$x_{n+1} = rx_n(1 - x_n) \quad (2.8)$$

en donde r es un parámetro del sistema.

La evolución de las iteraciones de este mapa exhibe una gran sensibilidad a las condiciones iniciales, dependiendo del parámetro r . Para el intervalo $3,57 \leq r \leq 4$ este mapa presenta comportamiento caótico. Su representación en el plano x_{n+1} vs. x_n se ve en la Figura 2.16 para tres valores del parámetro $r = 2, 3, 4$. También podemos ver su diagrama de bifurcaciones en 2.17, los tres valores del parámetro elegidos se resaltan en esta Figura.

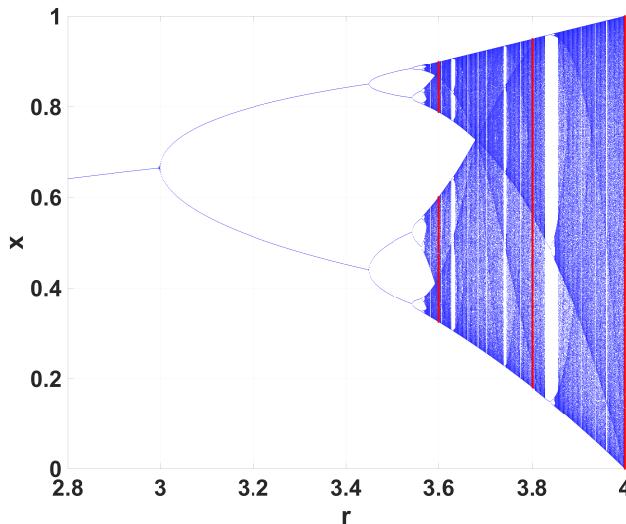


Figura 2.17: Diagrama de bifurcaciones para el mapa Logístico, con valores de parámetro $2,8 \leq r \leq 4$.

2.3.2. Mapa Tent

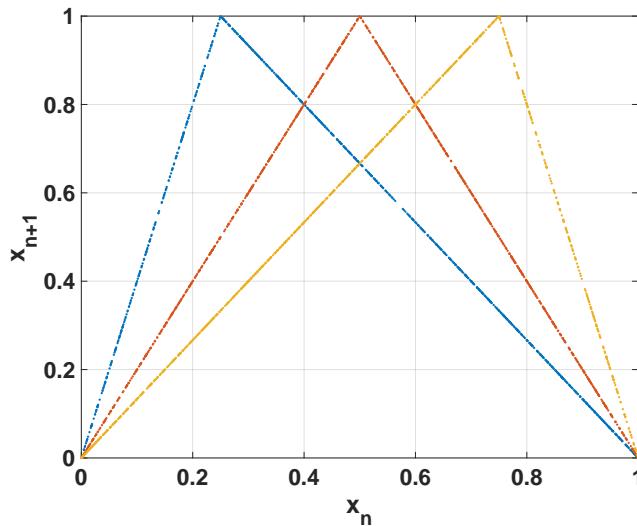
El mapa Tent está definido por la ecuación de recurrencias:

$$x_{n+1} = \begin{cases} u x_n & , \text{si } 0 \leq x_n \leq 1/u \\ \frac{u}{1-u} (1 - x_n) & , \text{si } 1/u < x_n \leq 1 \end{cases} \quad (2.9)$$

con x_n y $u \in \mathbb{R}$.

El nombre “Tent” viene de su representación en el plano x_{n+1} vs. x_n que se ve en la Figura 2.18, en donde el parámetro toma tres valores, $u = 4, 2, 4/3$. Cuando este parámetro es distinto de 2, se dice que el mapa es un *skew Tent*.

Este mapa tiene la característica de que para $r = 2$ sus propiedades estadísticas son ideales, cuando se las calcula teóricamente mediante el operador de Perron-Frobenious [27, 28], es decir analíticamente. Sin embargo, a la hora de la implementación en un medio digital, todas sus iteraciones pueden ser representadas con una operación de acarreo de bits desde la izquierda, lo que provoca que converja rápidamente a cero. Este hecho está muy bien explicado en [29].

Figura 2.18: Mapa Tent para tres parámetros distintos, $u = 4, 2, 4/3$

2.3.3. Mapas Cuadráticos Bidimensionales

La familia de mapas cuadráticos bidimensionales que estudiamos aquí es modelada por un par de ecuaciones cuadráticas acopladas:

$$\begin{cases} x_{n+1} = a_1 + a_2 x_n + a_3 x_n^2 + a_4 x_n y_n + a_5 y_n + a_6 y_n^2 \\ y_{n+1} = a_7 + a_8 x_n + a_9 x_n^2 + a_{10} x_n y_n + a_{11} y_n + a_{12} y_n^2 \end{cases} \quad (2.10)$$

donde (x, y) son las variables de estado y $A = \{a_i, i = 1, \dots, 12\}$ son los parámetros. La principal característica de este sistema es que presenta múltiples atractores caóticos en función del punto seleccionado en el espacio de parámetros. El espacio de parámetros de $12D$ generado por los coeficientes A es muy difícil de explorar.

Las razones para estudiar este sistema en particular son dos:

1. Usando la aritmética de punto flotante con un barrido automático de parámetros a_i y una gran cantidad de puntos en el espacio del parámetro (alrededor de $6 \cdot 10^{16}$), Sprott pudo detectar varios atractores en régimen caótico permanente. Es decir, este sistema tiene la característica de modificar su atractor según los valores que tomen sus 12 coeficientes reales. También encontró una relación entre la dimensión de correlación y los exponentes de Lyapunov, con su estética visual, un tema interesante para la generación automática de arte.

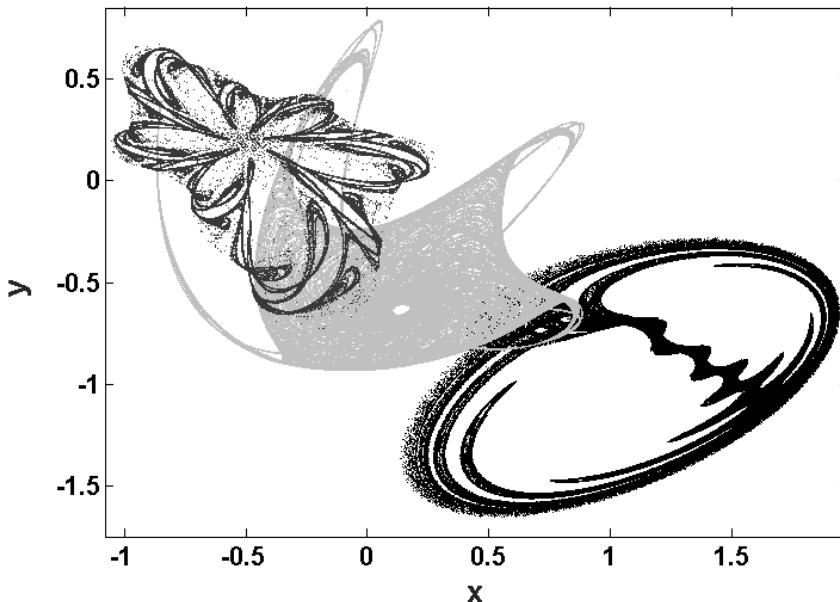


Figura 2.19: Atractores de sistema de mapas cuadráticos bidimensionales evaluado con tres juegos diferentes de parámetros.

2. Es posible emplear estos atractores en una amplia variedad de aplicaciones electrónicas, como la generación de nuevos sistemas de encriptación, ya sea reemplazando el S-box en AES [30, 31], o incluso desarrollando nuevos algoritmos [1, 2].

Tres de estos atractores caóticos se muestran juntos en la Figura 2.19. Sus juegos de parámetros A_i son:

$$\begin{aligned}
 A_1 &= \{-0.7, -0.4, 0.5, -1.0, -0.9, -0.8, 0.5, 0.5, 0.3, 0.9, -0.1, -0.9\}, \\
 A_2 &= \{-0.6, -0.1, 1.1, 0.2, -0.8, 0.6, -0.7, 0.7, 0.7, 0.3, 0.6, 0.9\}, \\
 A_3 &= \{-0.1, 0.8, -0.7, -1.1, 1.1, -0.7, -0.4, 0.6, -0.6, -0.3, 1.2, 0.6\}.
 \end{aligned} \tag{2.11}$$

Como se puede ver en la Figura, es posible obtener salidas muy diferentes simplemente modificando el valor de los parámetros y manteniendo la estructura del sistema. En una implementación electrónica, esto sería equivalente a poder variar la salida manteniendo la estructura del hardware y modificando los parámetros a través de, por ejemplo, una entrada.

Las Figuras 2.20.a a 2.20.d muestran los mismos tres atractores A_1 a A_3 de la Figura 2.19

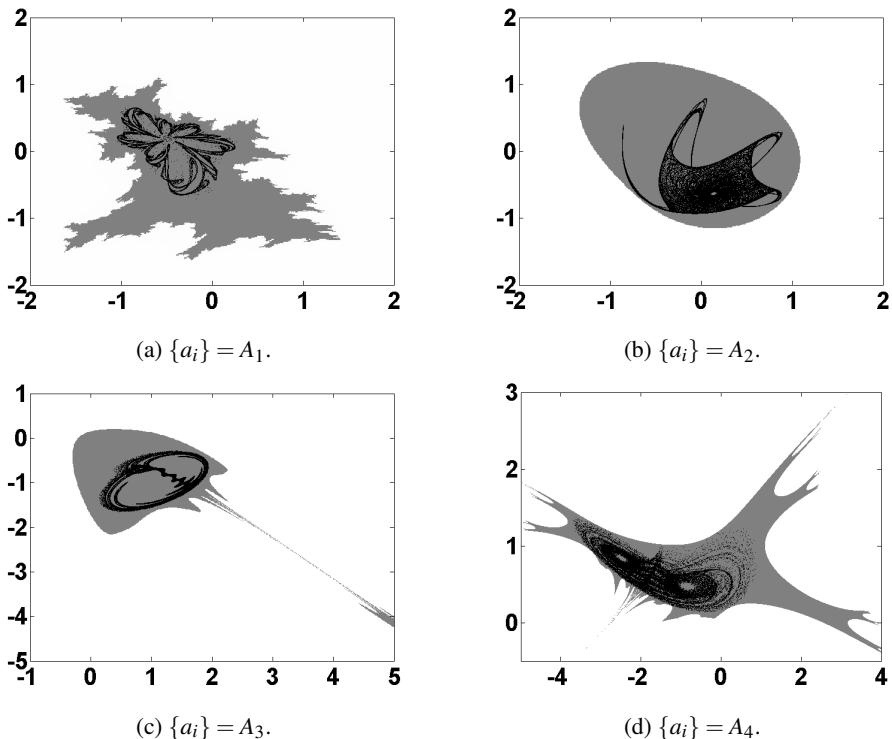


Figura 2.20: Cuatro atractores caóticos y sus respectivos dominios de atracción.

junto a un atractor con $A_4 = \{-1, 0.9, 0.4, -0.2, -0.6, -0.5, 0.4, 0.7, 0.3, -0.5, 0.7, -0.8\}$, superpuestos con sus dominios de atracción (en gris). Las áreas blancas de cada Figura corresponden a aquellas condiciones iniciales que generan trayectorias divergentes del sistema (semillas inútiles con respecto a su uso como PRNG).

2.4. El problema de la Aritmética Discreta

Cuando un sistema con MLE positivo es implementado utilizando Aritmética discreta, su comportamiento puede variar respecto del sistema original en aritmética de números reales. En un sistema como este, las pequeñas perturbaciones son amplificadas exponencialmente, y los “saltos” en las trayectorias provocados por la aritmética pueden ser considerados como tales.

Al iterar mapas caóticos, después de un transitorio que depende del parámetro de mezcla (r_{mix}), la secuencia generada limita en un punto o colección de puntos llamado atractor. Un mapa caótico puede tener uno o más atractores. Dominio de atracción se llama a todas las

condiciones iniciales (CIs) que convergen a cada atractor. Las secuencias ergódicas de los atractores, generadas por el mapa, tienen una distribución determinada llamada Función de densidad de probabilidad invariable (IPDF). Las principales características de los mapas caóticos, IPDF y r_{mix} , pueden obtenerse calculando el operador Perron-Frobenious (PFO), que depende de la estructura del mapa. Los puntos fijos de su espectro son las densidades invariables y corresponden a los vectores propios con valor propio igual a uno, la constante de mezcla corresponde al segundo mayor valor propio del PFO, [27, 28].

Cuando se utiliza precisión finita, este análisis no es válido, todos los atractores toman la forma de puntos fijos u órbitas periódicas. El PFO del mapa ya no describe las características de las secuencias. Con respecto al dominio de atracción, también cambiará cuando se digitalice, cada valor inicial será parte de, o convergerá a, un cierto punto fijo u órbita periódica. En general, aparecen muchas nuevas órbitas periódicas y cambian cuando varía el número de bits empleados.

Con el propósito de utilizar estos sistemas en aplicaciones electrónicas, se hace necesario comprender cómo evoluciona el dominio de atracción con la variación de bits empleados. Principalmente es importante saber cuál es la duración del período y el grado de aleatoriedad del ciclo en el que converge cada semilla. Por esta razón, se proponen cuantificadores de aleatoriedad que estiman indirectamente el r_{mix} e IPDF del sistema digitalizado.

Capítulo 3

Cuantificadores de Aleatoriedad

Los sistemas dinámicos son sistemas que evolucionan en el tiempo. En la práctica, sólo es posible medir alguna funcional del sistema bajo estudio, generalmente una serie de tiempo escalar $X(t)$ la cual puede ser función de las variables $V = \{v_1, v_2, \dots, v_k\}$ que describe la dinámica subyacente (por ejemplo $dV/dt = f(V)$). Tratamos de inferir propiedades de un sistema no conocido a partir del análisis de los datos guardados de variables observacionales. ¿Cuánta información revelan estos datos sobre la dinámica del sistema o procesos subyacentes?

El contenido de información de un sistema se evalúa típicamente mediante una función de distribución de probabilidades (PDF) P que describe la distribución de alguna cantidad mensurable u observable, generalmente una serie temporal $X(t)$. Podemos definir los cuantificadores de la Teoría de la Información como medidas capaces de caracterizar las propiedades relevantes de las PDFs asociadas a estas series temporales, y de esta manera debemos extraer juiciosamente información sobre el sistema dinámico en estudio. Estos cuantificadores representan métricas en el espacio de PDFs para conjuntos de datos, permitiendo comparar diferentes conjuntos y clasificarlos de acuerdo a sus propiedades de procesos subyacentes, de manera amplia, estocástica vs. determinística.

En nuestro caso, nos interesa la dinámica caótica. Por lo tanto, nos centramos en las métricas que toman en cuenta el orden temporal de las observaciones de forma explícita; es decir, el enfoque es fundamentalmente de naturaleza *causal* y *estadística*. En un enfoque puramente estadístico, las correlaciones entre los valores sucesivos de las series temporales se ignoran o simplemente se destruyen a través de la construcción de la PDF; mientras que

un enfoque causal se centra en las PDFs de secuencias de datos. Además, los exponentes de Lyapunov permiten analizar los datos de un punto de vista topológico y brindan una valiosa información acerca de la caoticidad del sistema.

En este Capítulo se presenta al Máximo Exponente de Lyapunov (MLE) como un detector de caos en la Sección 3.1, para luego presentar un caso de aplicación de un algoritmo de búsqueda de caos. Este algoritmo fue presentado en [15] y muestra la factibilidad de la búsqueda automática de caos con algoritmos eurísticos basados en el MLE. Otras publicaciones en este campo son dos implementaciones en FPGA de distintos algoritmos. En [16] se presenta una implementación en hardware de un estimador del MLE. Esta técnica precisa tener acceso a las entradas y salidas del sistema analizado, es decir, introducir las CIs del sistema. Además, forma parte del sistema presentado en [15]. En [17] se presentó otro algoritmo que estima el MLE a partir de una serie de datos, es decir que no precisa acceso al sistema. Esta última aproximación es más realista, ya que generalmente uno sólo tiene acceso a una serie temporal observable del sistema.

En la segunda Sección de este Capítulo se presentan una serie cuantificadores de aleatoriedad provenientes de la Teoría de la Información. Estos cuantificadores se utilizan luego a través del resto de esta tesis como una herramienta de análisis, con ellos se evalúa la calidad de los generadores de números aleatorios. Luego se muestran los resultados presentados en [32], en donde se implementaron estas herramientas en FPGA. La implementación de estos cuantificadores surge como una solución práctica, así es posible medir la calidad de los generadores en la misma plataforma, sin la necesidad de extraer los datos y medirlos en una computadora. Aprovechando la disponibilidad de entradas analógicas en el kit de desarrollo, al diseño se le agregó la posibilidad de medir señales analógicas externas. Cuando se midieron señales de prueba bien conocidas, los resultados mostraron ciertos corrimientos de los valores esperados debido a la contaminación con ruido aditivo (AWGN) y a la limitación en la banda de paso inherentes a todo sistema analógico. Esto abre la inquietud de caracterizar el comportamiento de los cuantificadores frente a estos dos factores, por lo que se presentó un trabajo al respecto en [33]. Los resultados de este estudio se muestran en la Sección 3.2.6.

3.1. Máximo Exponente de Lyapunov

¿Qué es lo que diferencia a un ciclo límite o una órbita cerrada de una órbita caótica? Una órbita caótica (atractor caótico) es aperiódica, es decir que nunca se repite exactamente y la oscilación persiste para $t \rightarrow \infty$. Como se vio en la Sección 2.2, el movimiento sobre un atrácto exhibe una dependencia sensible a las condiciones iniciales. Esto significa que dos trayectorias que comienzan muy cercanas, divergen rápidamente una de otra, por lo que tendrán futuros muy diferentes. La implicación práctica de esto es que la predicción a largo plazo se vuelve imposible en un sistema como este, en donde pequeñas incertezas son amplificadas rápidamente. Hagamos estas ideas un poco más precisas. Supongamos que tenemos una trayectoria sobre el atrácto y un punto $x(t_1)$ perteneciente a dicha trayectoria en un instante t_1 . Ahora consideremos un punto vecino $x(t_1) + \delta_0$, en donde δ_0 es una pequeña perturbación inicial. Ahora veamos cómo evoluciona esta separación $\delta(t)$. Encontramos que:

$$\|\delta(t)\| \sim \|\delta_0\| e^{\lambda t} \quad (3.1)$$

Por lo tanto, trayectorias vecinas se separan a un ritmo exponencial. El número λ es llamado exponente de Lyapunov. Cuando este exponente es positivo, se dice que el sistema tiene un horizonte de tiempo t_h más allá del cual la predicción falla por una tolerancia a , de modo que:

$$t_h \sim O\left(\frac{1}{\lambda} \ln \frac{a}{\|\delta_0\|}\right) \quad (3.2)$$

Como este sistema presenta un horizonte de tiempo, puede decirse que es sensible a las condiciones iniciales, su exponente de Lyapunov es positivo y resulta ser caótico.

Los exponentes de Lyapunov son cuantificadores que caracterizan cómo evoluciona la separación entre dos trayectorias [34]. En general es bien conocido que el comportamiento caótico está principalmente caracterizado por los números de Lyapunov de la dinámica del sistema.

Venimos llamando al número λ exponente de Lyapunov, sin embargo, este es un uso poco riguroso de este término, por dos razones: Primero, λ depende de la trayectoria que estamos estudiando, deberíamos promediar sobre muchos puntos sobre la misma trayectoria para obtener su verdadero valor. Segundo, realmente hay tantos exponentes de Lyapunov

como dimensiones tenga el sistema. Supongamos la evolución de una esfera infinitesimal de condiciones iniciales en el espacio de estados de tres dimensiones. Durante esta evolución la esfera se vuelve un elipsoide infinitesimal con tres ejes principales λ_1, λ_2 y λ_3 , siendo estos tres los exponentes de Lyapunov del sistema. El caos está definido por el máximo exponente de Lyapunov, a partir de ahora MLE, entonces basta que uno de los tres exponentes sea positivo para que el sistema sea caótico. Si uno o más números de Lyapunov son mayores que cero entonces el sistema se comporta caóticamente, de otra forma el sistema es estable. Esta es una condición suficiente de caoticidad, ya que un sistema divergente puede tener MLE positivo. Por lo tanto, para que un sistema sea caótico, además de tener algún exponente de Lyapunov positivo debe tener una trayectoria acotada no divergente en el plano de fase.

Entonces, el MLE caracteriza qué tan rápido se apartan dos trayectorias inicialmente vecinas. Para un sistema no divergente, si esta velocidad es exponencial, se dice que el sistema es caótico, por lo que este exponente es conocido como un detector de “caoticidad”, [5, 9, 34].

El MLE fue utilizado en diversas aplicaciones de muy distintas áreas. Sólo por mencionar algunas, en [35] el MLE es usado para medir una señal muy débil en un gas ideal utilizando criterios caóticos. En [36], se estudia la posibilidad de predecir un cambio en la probabilidad de caída para un modelo simple de caminante humano a partir del *MLE*.

Sabemos que el sistema en tiempo continuo es una idealización, por lo que se usa el exponente de Lyapunov para tiempo discreto, que para un sistema de tres dimensiones se calcula como:

$$L = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=1}^n \log_2 J_F(x, y, z) \quad (3.3)$$

en donde n es la cantidad de puntos consecutivos tomados del oscilador caótico. El vector columna de tres dimensiones L , contiene los tres exponentes de Lyapunov y $J_F(x, y, z)$ es la matriz Jacobiana de la función F para las variables (x, y, z) . El logaritmo es en base dos para estimar la velocidad de apartamiento en bits. Con estas dos consideraciones (tiempo discreto y base numérica binaria finita) la distancia entre dos trayectorias cambia en 2^{MLE} por cada iteración, en promedio. Por lo tanto, suponiendo que estamos ante un sistema no divergente tenemos tres situaciones posibles. Si el $MLE < 0$ las trayectorias se aproximan, esto puede deberse a un punto fijo. Si el $MLE = 0$ las trayectorias mantienen su distancia, esto puede deberse a un ciclo límite. Si el $MLE > 0$ la distancia entre las trayectorias es

creciente, lo que es un indicador de caos.

Calcular L con la Ecuación 3.3 tiene dos problemas: (1) requiere infinitas iteraciones para hallar los exponentes de Lyapunov y (2) trabajar con el jacobiano puede ser computacionalmente pesado, o éste puede no existir. Afortunadamente existe un algoritmo no analítico por aproximaciones sucesivas que converge al máximo exponente de Lyapunov. Las entradas y las salidas de un sistema deben ser accesibles para poder utilizarlo. El procedimiento es el siguiente: el sistema debe ser iniciado desde dos puntos cercanos en el plano de fase, digamos (x_a, y_a) y (x_b, y_b) . A medida que el sistema es iterado se mide la distancia euclídea entre las dos trayectorias (d_n en la muestra n_{esima}) (Ecuación 3.4), y la trayectoria b es relocalizada en cada iteración (Ecuación 3.6) obteniendo los puntos (x_{br}, y_{br}) para realimentar el sistema. Entonces, el MLE puede ser calculado como se muestra en la Ecuación 3.5. El proceso puede verse en la Figura 3.1.

$$\begin{aligned} d_{0(i-1)} &= \sqrt{(x_{a(i-1)} - x_{br(i-1)})^2 + (y_{a(i-1)} - y_{br(i-1)})^2} \\ d_{1(i)} &= \sqrt{(x_{a(i)} - x_{b(i)})^2 + (y_{a(i)} - y_{b(i)})^2} \end{aligned} \quad (3.4)$$

$$MLE = \frac{1}{n} \sum_{i=2}^n \log_2 \frac{d_{1(i)}}{d_{0(i-1)}} \quad (3.5)$$

$$\begin{aligned} x_{br(i)} &= x_{a(i)} + (x_{b(i)} - x_{a(i)})d_{0(i-1)}/d_{1(i)} \\ y_{br(i)} &= y_{a(i)} + (y_{b(i)} - y_{a(i)})d_{0(i-1)}/d_{1(i)} \end{aligned} \quad (3.6)$$

3.1.1. Algoritmo Evolutivo para la Búsqueda de Caos

Se propuso emplear un método heurístico para buscar parámetros del sistema implementado de tal forma que se maximice la caoticidad de su salida. La idea es utilizar al MLE como la *fitness function* de un algoritmo evolutivo, mediante el cual se busca maximizarla. Este algoritmo tiene la ventaja que realiza una búsqueda inteligente mediante el empleo de

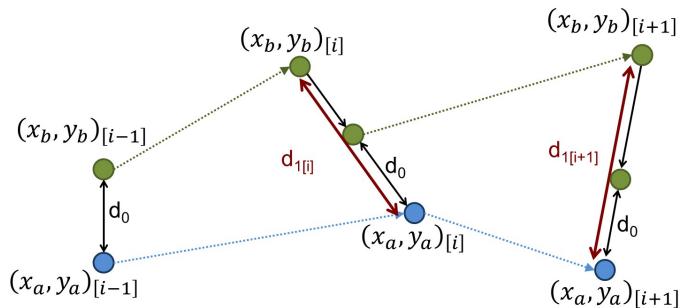


Figura 3.1: Algoritmo para calcular el MLE.

un algoritmo genético, lo que minimiza el tiempo de cómputo.

Los algoritmos evolutivos son algoritmos de optimización metaheurísticos basados en poblaciones, que utilizan mecanismos inspirados en la biología tales como mutación, cruce, selección natural y supervivencia del más apto para refinar un conjunto de soluciones candidatas en forma iterativa [37].

Las entidades que representan posibles soluciones al problema son llamados *cromosomas* y el grupo de cromosomas es llamados *población inicial*. Desde la población inicial, o los primeros padres, se genera un hijo mediante el cruce entre ellos. Luego, ellos son mutados en forma aleatoria para crear la próxima generación. Cada generación es comparada con la previa para descartar los “peor adaptados” y así los coeficientes (cromosomas) mutan hacia los “mejor adaptados”.

Cuando se aplican estos algoritmos en funciones continuas, siempre convergen hacia el máximo local. Sin embargo, si el espacio de coeficientes es fractal, existen áreas bien definidas en donde la función objetivo es positiva, negativa, cero o no existente. Este es el caso cuando la función a maximizar es el MLE y el espacio de exploración es el de los parámetros de un sistema.

Resultados

Para evaluar la viabilidad del método, se generó el siguiente algoritmo y se probó sobre el mapa Logístico.

En la Figura 3.2 podemos ver el diagrama de flujo principal. El bloque *Evolution* fue descompuesto en otro sub-diagrama para simplificar la descripción. Este segundo diagrama puede verse en la Figura 3.3, esta subrutina maneja la evolución de los parámetros.

El algoritmo comienza con una inicialización general de parámetros como el número

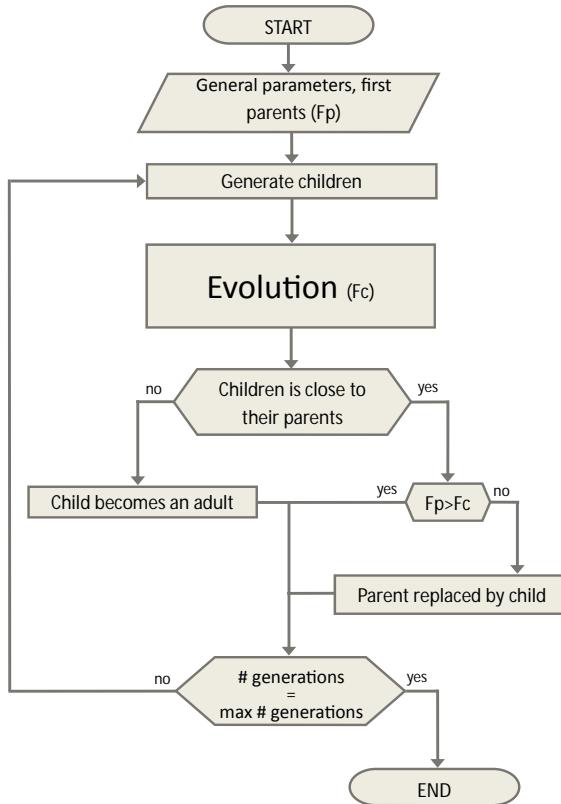
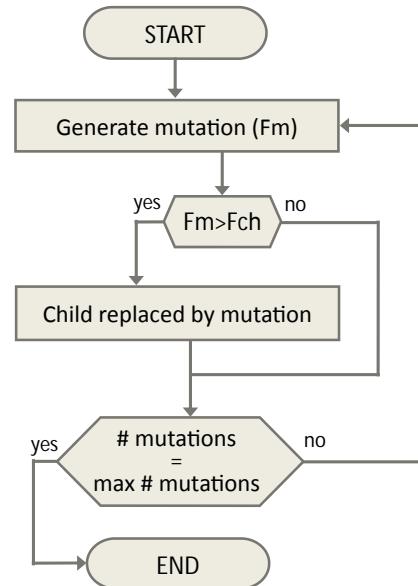


Figura 3.2: Diagrama de flujo principal.

Figura 3.3: Diagrama de flujo del bloque *Evolution*.

máximo de generaciones *max_gen*, el número máximo de mutaciones *max_mut* y el número máximo de cambios en cada mutación *max_stem*. Luego se definen los primeros dos padres, ellos definirán los márgenes de búsqueda. Además, se calcula su *fitness function* F_p . A partir de este punto se itera la segunda generación, se elige en forma aleatoria un valor de parámetro *r* con una distribución aleatoria entre los primeros dos padres, generando un nuevo hijo. Luego este hijo entra en la subrutina *Evolution* cuya salida es el valor de *r* evolucionado y su correspondiente *fitness function* F_c .

Luego se evalúa si este hijo evolucionó muy cerca de sus padres o no. Si la distancia entre ellos es más grande que el parámetro *max_hop*, entonces este hijo es considerado como adulto, en caso contrario debe competir con su parente más cercano sobreviviendo el más apto.

Este proceso se repite hasta que se llega al máximo número de generaciones *max_gen*. El grupo final de adultos es la solución al problema de buscar los máximos MLE locales.

La subrutina *Evolution* de la Figura 3.3 es un algoritmo muy simple basado en mutaciones. El primer paso es generar una mutación del hijo con una probabilidad uniformemente distribuida entre $\pm max_step$, también se calcula su *fitness function* F_m , que se compara con la del individuo original F_c . Entonces sobrevive el mejor adaptado para dar lugar a la siguiente mutación. Este procedimiento se repite hasta que se llega al máximo número de mutaciones *max_mut*.

Como resultado podemos ver el *MLE* del mapa Logístico en función de su único parámetro *r* en la Figura 3.4. La línea continua muestra el *MLE* con un paso muy fino de *r* ($\Delta r = 0,01$), mientras que los puntos destacados son el resultado del algoritmo propuesto.

El bloque que calcula el *MLE* fue sintetizado y verificado experimentalmente en un Altera CYCLONE III FPGA y los resultados de la compilación se muestran en la Figura 3.5. Los resultados del *Timing Analysis* reportan que la máxima frecuencia es de 84,95MHz. El reporte de compilación muestra que la utilización de la lógica no excede el 20 %, es decir un total de 20307 de elementos lógicos, 54 % de los bits de memoria totales y 8 % de los multiplicadores embebidos.

En la Figura 3.6 se muestra la salida del Signal Tap, esta herramienta permite tomar muestras directamente desde la placa en donde es implementado el algoritmo. La señal *salida* es la suma de los *MLE* luego de cada iteración. La segunda señal llamada *cuenta_sal* corresponde a la sumatoria actual. Finalmente, cada flanco descendente de la señal *listoD1* indica que la salida es un dato válido. La salida fue procesada con Matlab para obtener la

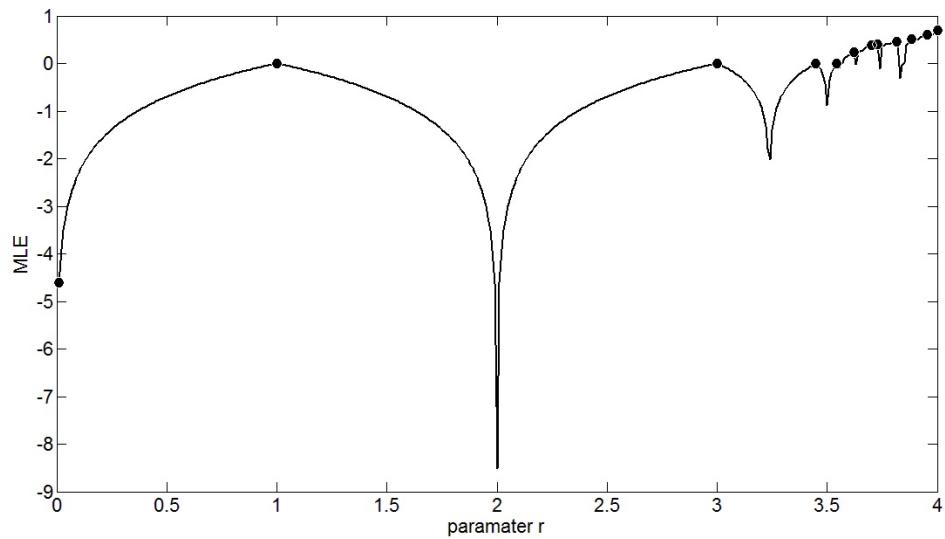


Figura 3.4: Resultados del algoritmo evolutivo para el mapa Logístico, los puntos son los resultados del algoritmo.

Flow Summary	
Flow Status	Successful - Fri Apr 19 10:20:17 2013
Quartus II 32-bit Version	12.1 Build 177 11/07/2012 SJ Web Edition
Revision Name	CalculaLyap
Top-level Entity Name	TOP
Family	Cyclone III
Device	EP3C120F780C7
Timing Models	Final
Total logic elements	29,307 / 119,088 (25 %)
Total combinational functions	26,048 / 119,088 (22 %)
Dedicated logic registers	18,014 / 119,088 (15 %)
Total registers	18014
Total pins	197 / 532 (37 %)
Total virtual pins	0
Total memory bits	2,133,356 / 3,981,312 (54 %)
Embedded Multiplier 9-bit elements	48 / 576 (8 %)
Total PLLs	1 / 4 (25 %)

Figura 3.5: Reporte de compilación de la implementación del cálculo del MLE.

Figura 3.6: Salida del Signal Tap.

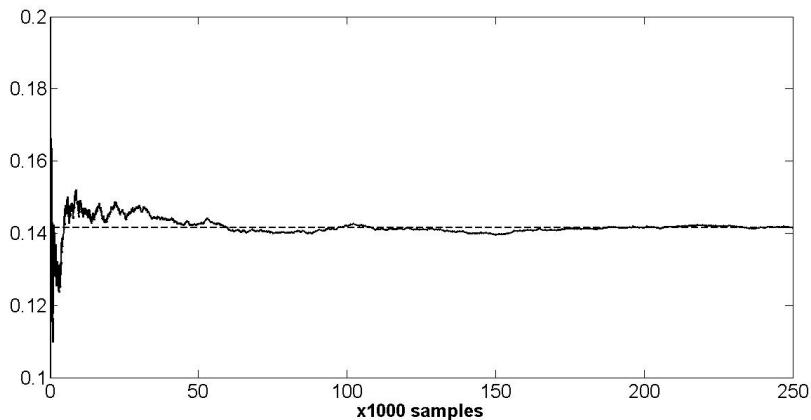


Figura 3.7: Convergencia del algoritmo que calcula el MLE.

curva mostrada en la Figura 3.7. El valor del MLE en la iteración 250 000 es 0,1415, lo que es consistente con el MLE obtenido con Matlab para el mapa Logístico.

3.2. Cuantificadores de la Teoría de la Información

Dada una fuente de símbolos cuya salida es un vector de símbolos X , existen diferentes procedimientos para obtener una PDF [38, 39, 40, 41, 42, 8]. La determinación de la mejor PDF P es un problema fundamental porque P y el espacio de muestra X están inextricablemente vinculados. Su aplicabilidad depende de las características particulares de los datos, tales como estacionariedad, longitud de la serie temporal, variación de los parámetros, nivel de contaminación de ruido, etc.

Los Cuantificadores de la Teoría de la Información (ITQs) seleccionados se basan en el recuento de símbolos y en la estadística de patrones de orden. Las métricas a utilizar pueden clasificarse de forma amplia en dos categorías: las que cuantifican el *contenido de información* de los datos en comparación con los relacionados con su *complejidad*. Obsérvese que aquí nos estamos refiriendo al espacio de funciones de densidad de probabilidad, no al espacio físico. Para clarificar y simplificar, introducimos solamente los ITQs que se definen

en PDFs discretas, ya que solo estamos tratando con datos discretos (series temporales). Sin embargo, todos los cuantificadores también tienen definiciones para el caso continuo [7].

3.2.1. Entropía de Shannon y Complejidad Estadística

La entropía es una cantidad básica que puede considerarse como una medida de la incertidumbre asociada (información) al proceso físico descripto por P . Al tratar con el contenido de la información, la entropía de Shannon se considera a menudo como la medida fundamental y más natural [7]. Es considerada como una medida de la incertidumbre y es el ejemplo más paradigmático de los cuantificadores de la información.

Sea $P = \{p_i; i = 1, \dots, N\}$ con $\sum_{i=1}^N p_i = 1$, una distribución de probabilidad discreta, con N el número de estados posibles del sistema bajo estudio. La medida de la información logarítmica de Shannon se denota como:

$$S[P] = - \sum_{i=1}^N p_i \ln [p_i] . \quad (3.7)$$

Si $S[P] = S_{\min} = 0$, estaremos en posición de predecir con total certeza cuáles de los posibles resultados i , cuyas probabilidades están dadas por p_i , tendrán lugar realmente. Nuestro conocimiento del proceso subyacente descripto por la distribución de probabilidad es máximo en este caso. Por el contrario, nuestro conocimiento es mínimo para una distribución uniforme $P_e = \{p_i = 1/N; i = 1, \dots, N\}$ dado que cada resultado exhibe la misma probabilidad de ocurrencia, y la incertidumbre es máxima, es decir, $S[P_e] = S_{\max} = \ln N$. Estas dos situaciones son casos extremos, por lo tanto, nos centramos en la entropía de Shannon "normalizada", $0 \leq H \leq 1$, dada como

$$H[P] = S[P]/S_{\max} . \quad (3.8)$$

Contrariamente a la medida de la información, no existe una definición universalmente aceptada de complejidad. Aquí, nos centramos en describir la *complejidad de las series temporales* y no nos referimos a la complejidad de los *sistemas* subyacentes. Un sistema complejo no genera necesariamente una salida compleja. De hecho, modelos "simples" pueden generar datos complejos, mientras que sistemas "complicados" pueden producir datos de salida de baja complejidad [43].

Una noción intuitiva de una complejidad cuantitativa atribuye valores bajos tanto a datos

perfectamente ordenados (es decir, con entropía de Shannon que tiende a cero) como a datos aleatorios no correlacionados (con entropía Shannon máxima). Por ejemplo, tanto la complejidad estadística de una oscilación simple o tendencia (ordenada) como la del ruido blanco no correlacionado (no ordenado) serían clasificadas como bajas. En un punto intermedio entre estos casos de mínima y máxima entropía, los datos son más difíciles de caracterizar y por lo tanto, la complejidad debe ser mayor. Buscamos alguna función $C[P]$ que cuantifique las estructuras presentes en los datos que se alejan de estos dos casos. Estas estructuras se relacionan con la organización, la estructura correlacional, la memoria, la regularidad, la simetría, los patrones y otras propiedades [44].

Asumimos que el grado de estructuras correlacionales sería capturado adecuadamente por algún funcional $C[P]$ de la misma manera que la entropía de Shannon $S[P]$ [7] “captta” la aleatoriedad. Claramente, las estructuras ordinales presentes en un proceso no son cuantificadas por medidas de aleatoriedad y, por consiguiente, son necesarias medidas de complejidad estadística o estructural para una mejor comprensión (caracterización) de la dinámica del sistema representada por sus series temporales [45].

Una medida adecuada de complejidad puede definirse como el producto de una medida de información y una medida de desequilibrio, es decir, algún tipo de distancia de la distribución equiprobable de los estados accesibles de un sistema. En este sentido, en [46] los autores introdujeron una *Medida de Complejidad Estadística* eficaz (SCM) C , que es capaz de detectar detalles esenciales de los procesos dinámicos subyacentes al conjunto de datos. Basado en el trabajo de López-Ruiz [47], esta medida de complejidad estadística [48, 46] se define a través de la forma del producto de la entropía de Shannon normalizada H (ver Ecuación (3.8)) y el desequilibrio Q_J (ver Ecuación 3.9) definido en términos de la divergencia de Jensen-Shannon $J[P, P_e]$ (ver Ecuación 3.10). En la divergencia de Jensen-Shannon mencionada arriba, Q_0 es una constante de normalización tal que $0 \leq Q_J \leq 1$ y es igual a la inversa del máximo valor posible de $J[P, P_e]$. Este valor es obtenido cuando una de las componentes de P , digamos p_m , es igual a uno y todos los p_j restantes son cero (ver Ecuación 3.11).

$$C[P] = Q_J[P, P_e] \cdot H[P] \quad (3.9)$$

$$Q_J[P, P_e] = Q_0 J[P, P_e] = Q_0 \{S[(P + P_e)/2] - S[P]/2 - S[P_e]/2\}, \quad (3.10)$$

$$Q_0 = -2 \left\{ \frac{N+1}{N} \ln(N+1) - \ln(2N) + \ln N \right\}^{-1}, \quad (3.11)$$

La divergencia de Jensen-Shannon, que cuantifica la diferencia entre las distribuciones de probabilidad, es especialmente útil para comparar la composición simbólica entre diferentes secuencias [49]. Obsérvese que la SCM anteriormente introducida depende de dos distribuciones de probabilidad diferentes: una asociada con el sistema analizado, P , y la otra con la distribución uniforme, P_e . Además, se demostró que, para un valor dado de H , el rango de valores posibles de C varía entre un mínimo C_{min} y un máximo C_{max} , restringiendo los posibles valores del SCM [50]. Por lo tanto, está claro que evaluando la medida de complejidad estadística se obtiene información adicional importante relacionada con la estructura correlacional entre los componentes del sistema físico.

3.2.2. Determinación de la Función Distribución de Probabilidades

La evaluación de los cuantificadores derivados de la Teoría de la Información supone algún conocimiento previo sobre el sistema; específicamente para aquellos introducidos previamente (entropía de Shannon y complejidad estadística). Para calcularlos, es necesario proporcionar (obtener) una función de distribución de probabilidades asociada a la serie temporal bajo análisis. La determinación de la PDF más adecuada es un problema fundamental porque la PDF P y el espacio de muestra Ω están vinculados en forma unívoca.

Las metodologías usuales asignan a cada valor de la serie $X(t)$ (o conjunto de valores consecutivos no superpuestos) un símbolo de un alfabeto finito $A = \{a_1, \dots, a_M\}$, creando así una *secuencia simbólica* que puede considerarse como una descripción de la serie cronológica en cuestión. Como consecuencia, las relaciones de orden y las escalas temporales de la dinámica se pierden por completo.

Es importante resaltar que P en sí, no es un objeto con una definición única y existen varias aproximaciones para “asociar” una dada P con una dada serie temporal. Sólo para mencionar algunos criterios de extracción utilizados frecuentemente en la literatura: *a)* histogramas de series temporales [51], *b)* dinámica simbólica binaria [40], *c)* análisis de Fourier [41], *d)* transformadas wavelet [52, 42], *e)* PDF de particiones [53], *f)* PDF de permutaciones [8, 54], *g)* PDF discreta [55], etc. Para hacer una buena elección entre ellas, se debe hacer un análisis de la aplicación específica.

Para incorporar correctamente la información causal, en la secuencia simbólica se debe incluir información sobre la dinámica pasada del sistema, es decir, los símbolos del alfabeto A se asignan a una porción del espacio de fase o trayectoria. Bandt y Pompe (BP) [8] introdujeron una metodología simbólica simple y robusta que toma en cuenta el

ordenamiento temporal de las series temporales comparando valores vecinos en una serie temporal. La propiedad de causalidad de la PDF permite que los cuantificadores (basados en esta PDF) discriminen entre sistemas determinísticos y estocásticos [56]. Los datos simbólicos son: (i) creados por la clasificación de los valores de la serie; y (ii) definidos por el reordenamiento de los datos embebidos en orden ascendente, lo que equivale a una reconstrucción del espacio de fase con dimensión de embedding (longitud de patrón) D y retardo de tiempo τ . De esta forma, es posible cuantificar la diversidad de los símbolos de ordenación (patrones) derivados de una serie temporal escalar. Obsérvese que la secuencia de símbolos apropiada surge naturalmente de la serie temporal, y no se necesitan suposiciones basadas en modelos. El procedimiento es el siguiente:

- Dada una serie $\{x_t; t = 0, \Delta t, \dots, N\Delta t\}$, se genera una secuencia de vectores de longitud D .

$$(s) \mapsto (x_{t-(d-1)\Delta t}, x_{t-(d-2)\Delta t}, \dots, x_{t-\Delta t}, x_t) \quad (3.12)$$

Cada vector resulta ser la "historia" del valor x_t . Evidentemente, cuanto más larga sea la longitud de los vectores D , mayor será la información sobre la historia de los vectores, pero se requiere un valor más alto de N para tener una estadística adecuada.

- Las permutaciones $\pi = (r_0, r_1, \dots, r_{D-1})$ de $(0, 1, \dots, D-1)$ son llamadas “patrón de orden” de tiempo t , definida por:

$$x_{t-r_{D-1}\Delta t} \leq x_{t-r_{D-2}\Delta t} \leq \dots \leq x_{t-r_1\Delta t} \leq x_{t-r_0\Delta t} \quad (3.13)$$

Para obtener un resultado único se considera $r_i < r_{i-1}$ si $x_{t-r_i\Delta t} = x_{t-r_{i-1}\Delta t}$. De esta forma podemos definir la PDF de todas las $D!$ permutaciones posibles π de orden D , como $P = \{p(\pi)\}$, en donde:

$$p(\pi) = \frac{\#\{s | s \leq N - D + 1; (s) \text{ has type } \pi\}}{N - D + 1} \quad (3.14)$$

En estas últimas expresiones, el símbolo $\#$ denota cardinalidad.

Por lo tanto, una distribución de probabilidad de patrones de orden $P = \{p(\pi_i), i = 1, \dots, D!\}$ se obtiene de la serie temporal. De esta manera, el vector definido por la Ecuación 3.14 se convierte en un símbolo único π .

La única condición para la aplicabilidad del método BP es una suposición estacionaria

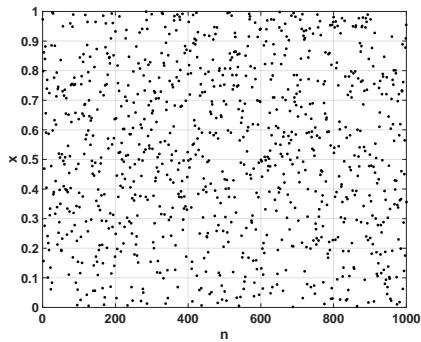
muy débil: para $k \leq D$, la probabilidad para $x_t < x_{t+k}$ no debe depender de t . Con respecto a la selección de los parámetros, Bandt y Pompe sugirieron trabajar con $3 \leq D \leq 6$ para longitudes de series de tiempo típicas, y específicamente se consideró un retraso de tiempo $\tau = 1$ en su publicación principal.

Para destacar la diferencia entre una *P causal* y una *no causal*, consideremos una serie de valores $X = \{x_i, i = 1, 2, \dots\}$ generada por la función *randn* de Matlab Consideremos también la serie $Y = \{y_i, i = 1, 2, \dots\}$ como la resultante de ordenar la serie X en forma ascendente. Esto se puede ver en el ejemplo en la Figura 3.8, en la Figura 3.8a se muestran 1000 valores sorteados con una distribución uniforme entre 0 y 1, también mostramos en la Figura 3.8b la versión ordenada de la serie de la Figura 3.8a, son los mismos valores pero, ordenados en forma ascendente. Una *P* no causal es el histograma normalizado que mostramos en las Figuras 3.8c y 3.8d, en donde puede verse que $P(X)$ es idéntica a $P(Y)$, por lo que todos los cuantificadores que se calculen a partir de ellas serán idénticos para las dos series. Una *P causal* puede ser obtenida mediante el procedimiento de Bandt & Pompe descripto arriba, en este caso $P(X)$ de la Figura 3.8e es bastante uniforme y $P(Y)$ de la Figura 3.8f tiene una forma tipo delta. En este caso, *P* registra que Y es monótonamente creciente y presenta un solo patrón de orden.

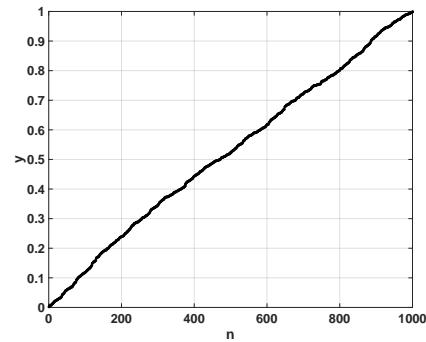
Recientemente, la entropía de permutación se amplió para incorporar también información de amplitud. Ponderar las probabilidades de patrones individuales de acuerdo a su varianza mitiga los problemas potenciales con respecto a los patrones de “alto ruido, baja señal”, porque los patrones de baja varianza que están fuertemente afectados por el ruido se ponderan en las distribuciones de patrones ordinales ponderados resultantes. Por lo tanto, una posible desventaja de las estadísticas de los patrones ordinales, es decir, la pérdida de información de amplitud, se puede abordar mediante la introducción de pesos con el fin de obtener una “entropía de permutación ponderada (WPE)” [57]. Los pesos no normalizados se calculan para cada ventana para la serie temporal X , tal que

$$w_j = \frac{1}{D} \sum_{k=1}^D \left(x_{j+k-1} - \bar{X}_j^D \right)^2. \quad (3.15)$$

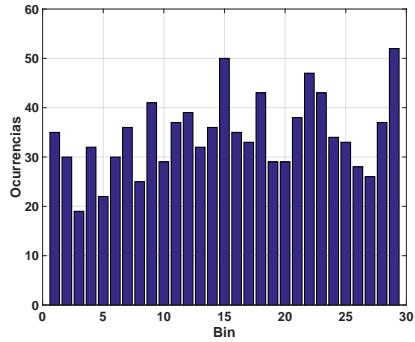
En la Ecuación anterior $x_{j+k-1} - \bar{X}_j^D$ denota la media aritmética del actual vector de embedding de longitud D y su varianza w_j se utiliza entonces para ponderar las frecuencias relativas de cada patrón ordinal p_j . Originalmente, se propuso esta técnica para discriminar patrones sumergidos en un bajo nivel de ruido. Nosotros también aprovechamos el hecho de



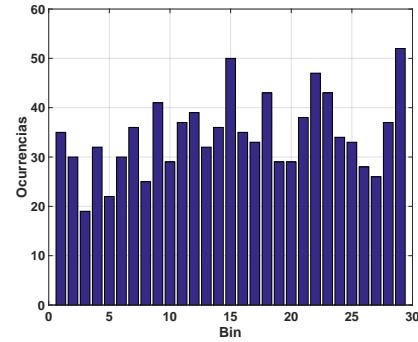
(a) 1000 puntos con distribución uniforme



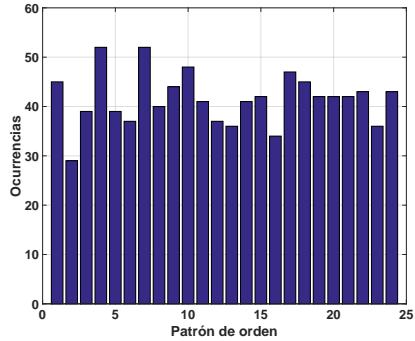
(b) Puntos ordenados



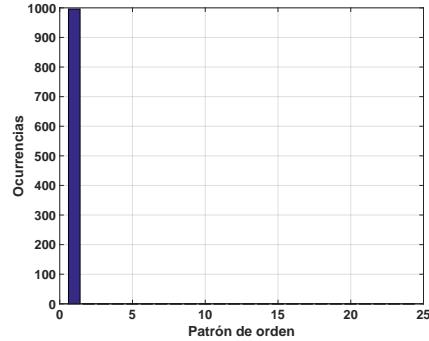
(c) Histograma de valores de los puntos sin ordenar



(d) Histograma de valores de los puntos ordenados



(e) Histograma de patrones de orden de los puntos sin ordenar



(f) Histograma de patrones de orden de los puntos ordenados

Figura 3.8: Comparación entre histogramas causal y no causal

que los puntos fijos no se computan en el WPE para detectar la existencia de trayectorias que divergen o caen a puntos fijos.

Al calcular la entropía de Shannon normalizada H y la complejidad estadística C de estas PDFs, los valores obtenidos se denotan como:

- H_{hist} , es la entropía de Shannon normalizada aplicada a una PDF no causal P_{hist}
- H_{BP} , es la entropía de Shannon normalizada aplicada a una PDF causal P_{BP}
- H_{BPW} , es la entropía de Shannon normalizada aplicada a una PDF causal con contribuciones de amplitud P_{BPW}
- C_{BP} , es la complejidad estadística normalizada aplicada a una PDF causal P_{BP}
- C_{BPW} , es la complejidad estadística normalizada aplicada a una PDF causal con contribuciones de amplitud P_{BPW}

3.2.3. Planos Doble Entropía y Entropía-Complejidad

Una visualización particularmente útil de los cuantificadores de la Teoría de la Información es su yuxtaposición en los gráficos bidimensionales. Se definen cuatro planos de información:

1. Entropía causal vs. entropía no-causal, $H_{BP} \times H_{hist}$
2. Entropía causal con contribución de amplitudes vs. entropía no-causal, $H_{BPW} \times H_{hist}$
3. Complejidad causal vs. entropía causal, $C_{BP} \times H_{BP}$
4. Complejidad causal con contribución de amplitudes vs. entropía causal con contribución de amplitudes, $C_{BPW} \times H_{BPW}$

Estas herramientas de diagnóstico demostraron ser particularmente eficientes para distinguir entre el caos determinista y la naturaleza estocástica de una serie de tiempo ya que los cuantificadores de permutación tienen comportamientos distintos para diferentes tipos de procesos [58, 59, 60, 39, 61, 62, 14].

En la Figura 3.9 se muestran los planos $H_{BP} \times H_{hist}$ y $H_{BPW} \times H_{hist}$ colapsados en un mismo plano. En este plano un valor más alto en cualquiera de las entropías, H_{BP} , H_{BPW} o H_{hist} , implica una mayor uniformidad de la PDF implicada. El punto (1,1) representa

el caso ideal con histograma uniforme y distribución uniforme de los patrones de orden. Mostramos algunos puntos relevantes como ejemplo.

El ruido aleatorio blanco ideal con distribución uniforme da un punto en $(H_{hist}, H_{BP}) = (1, 1)$ representado por un círculo azul, un círculo rojo en la misma posición muestra los resultados cuando se incluyen las contribuciones de amplitud $(H_{hist}, H_{BPW}) = (1, 1)$. Si ordenamos el vector ideal con distribución uniforme de forma ascendente, los puntos resultantes se muestran con un cuadrado azul $(H_{hist}, H_{BP}) = (1, 0)$ y un cuadrado rojo $(H_{hist}, H_{BPW}) = (1, 0)$, este ejemplo ilustra la complementariedad de H_{hist} y H_{BP} .

Las estrellas azules y rojas muestran (H_{hist}, H_{BP}) y (H_{hist}, H_{BPW}) respectivamente aplicadas a una señal de diente de sierra. Los valores están perfectamente distribuidos en todos los intervalos, pero sólo aparecen unos pocos patrones de orden, esto explica el alto H_{hist} y bajo H_{BP} . La frecuencia de aparición de patrones de baja amplitud es mayor que los patrones de alta amplitud, entonces la PDF con contribuciones de amplitud es más uniforme y H_{BPW} es un poco más alto que H_{BP} . Cuando la señal de diente de sierra está contaminada con ruido blanco, se incrementan H_{BP} y H_{BPW} como se muestra con triángulos azules y rojos. Es evidente que aparecen nuevos patrones de orden y tanto H_{BP} como H_{BPW} muestran valores más altos que los casos no contaminados, sin embargo, el incremento de H_{BPW} es menor que H_{BP} mostrando que la técnica de registrar contribuciones de amplitud añade alguna inmunidad al ruido.

Finalmente, se evaluaron los cuantificadores de una secuencia de un mapa Logístico que converge a un punto fijo, en todos los casos la longitud del vector de datos permanece constante y la longitud de transitorio es variable. Los resultados obtenidos sin las contribuciones de amplitud se representan en puntos azules, convergen a $(H_{hist}, H_{BP}) = (0, 0)$ a medida que la longitud de transitorio se hace más corta, sin embargo, H_{BPW} (puntos rojos) permanece constante para todos los casos. El último punto en $(H_{hist}, H_{BP}) = (0, 0)$ corresponde a un vector de ceros, en este caso el histograma de patrones de orden con contribuciones de amplitud es también un vector nulo y H_{BPW} no se puede calcular. A través de este último ejemplo, mostramos que la convergencia a un punto fijo puede ser detectada por la información conjunta de H_{BP} y H_{BPW} .

En la Figura 3.10 se muestra el plano causal $H_{BP} \times C_{BP}$. Podemos ver que no toda la región $0 < H_{BP} < 1, 0 < C_{BP} < 1$ es accesible, de hecho, para cualquier PDF los pares (H, C) de valores posibles caen entre dos curvas extremas en el plano $H_{BP} \times C_{BP}$ [63]. Los mapas caóticos tienen entropía intermedia H_{BP} , mientras que su complejidad C_{BP} alcanza valores

mayores, muy cercanos a los del límite de complejidad superior [59, 64]. Para procesos no aleatorios, la entropía y la complejidad tienen valores pequeños, cercanos a cero. Los procesos estocásticos no correlacionados se tienen una localización planar asociada con H_{BP} cerca de uno y C_{BP} cerca de cero. Los sistemas aleatorios ideales, que tienen una PDFs de Bandt & Pompe y de valores uniformes, están representados por el punto (1,0) [65] en el plano $H_{BP} \times C_{BP}$.

En la Figura 3.10 mostramos $H_{BP} \times C_{BP}$ con y sin contribuciones de amplitud. Se muestran los mismos puntos de muestra para ilustrar las posiciones planas para diferentes vectores de datos.

En ambos planos de información $H_{BP} \times H_{hist}$ en la Figura 3.9 y $H_{BP} \times C_{BP}$ en la Figura 3.10, los datos estocásticos, caóticos y deterministas están claramente localizados en diferentes posiciones planares.

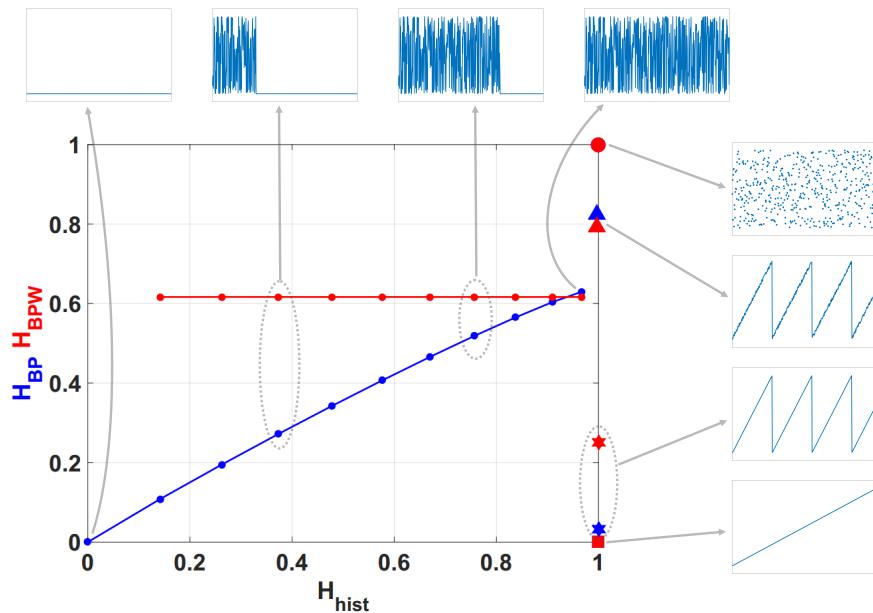


Figura 3.9: Plano de Entropías Causal y no Causal.

También usamos el número de patrones faltantes MP como cuantificador [66]. Como mostraron recientemente Amigó y colaboradores [67, 55, 68, 69], en el caso de mapas deterministas, no todos los patrones de orden posibles pueden materializarse efectivamente en órbitas. De hecho, la existencia de estos patrones de orden faltantes se convierte en un hecho persistente que puede considerarse como una nueva propiedad dinámica. Por lo

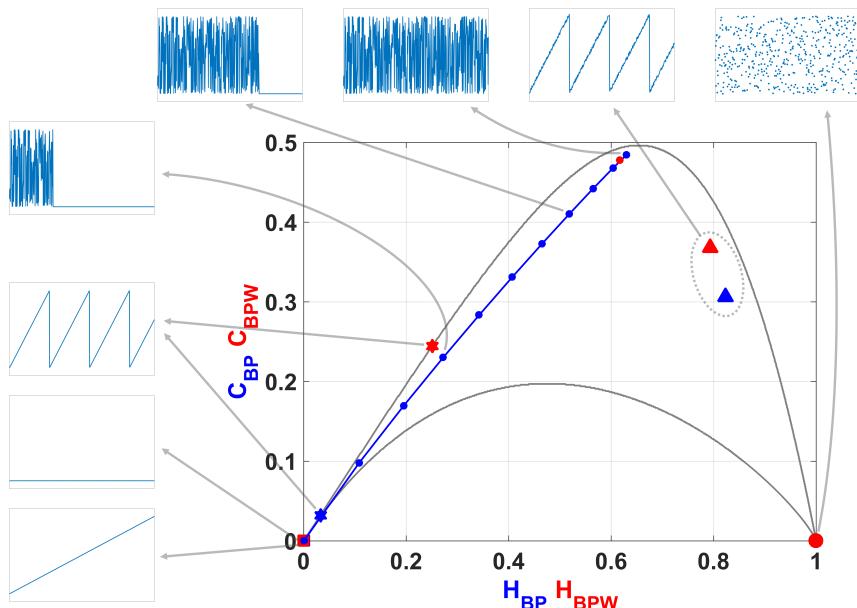


Figura 3.10: Plano de Entropía-Complejidad causales.

tanto, para una longitud de patrón fija (dimensión de embedding D) el número de patrones faltantes de una serie temporal (patrones no observados) es independiente de la longitud de la serie N . Obsérvese que esta independencia no caracteriza otras propiedades de la serie como la proximidad y la correlación [55, 69].

3.2.4. Entropías Diferenciales

Consideremos ahora el caso de las señales digitales muestreadas. La serie temporal X ahora es una señal binaria y tiene un alfabeto natural de dos símbolos $\mathfrak{A} = \{0, 1\}$. La entropía de Shannon de este alfabeto es conocida usualmente como Entropía Binaria S_2 . Supongamos que W bits consecutivos de X son agrupados en *una palabra*, que puede ser representada como un número decimal w_i entre 0 y $2^W - 1$; consideremos esos números decimales como los símbolos de un alfabeto nuevo, digamos que $Z = \{w_i, i = 1, 2, \dots\}$ es la nueva serie de símbolos. S_W es la entropía de $P_{hist}(Z)$, también es conocida como Entropía de Bloques de la serie temporal binaria X . Si además agrupamos D números consecutivos w_i y los $D!$ patrones de permutación son considerados como símbolos de un alfabeto nuevo, obtenemos $P_{BP}(Z)$, que es la entropía de patrones de orden de Z .

Enfaticemos algunas cuestiones importantes involucradas en los cálculos de las entropías binarias mencionadas anteriormente:

1. La entropía binaria S_2 es no causal, mientras que ambas, la entropía de bloque S_W y la entropía Bandt & Pompe $S_{BP}^{(D)}$, son causales.
2. La entropía de bloque S_W tiene en cuenta las correlaciones entre W bits consecutivos. La entropía Bandt & Pompe $S_{BP}^{(D)}$ tiene en cuenta las correlaciones entre D palabras consecutivas de longitud W . Ambos procedimientos de agrupación (números decimales de W bits y patrones de permutación de D números decimales) pueden realizarse con o sin superposición. La cantidad de datos requeridos para obtener buenas estadísticas es diferente dependiendo de los procedimientos de agrupación que se realicen.
3. Para S_W solo hay un proceso de agrupación (W bits agrupados para obtener una serie de números decimales Y). Definamos α como un parámetro de calidad estadística, dado por el cociente entre el número de elementos en la serie de temporal simbólica Y y la cantidad de símbolos en el alfabeto. En esta tesis se trabajó siempre con $\alpha < 10$.

Obviamente, el factor de calidad α aumenta con la longitud de la serie temporal:

- a) Si el agrupamiento de W bits está hecho con dos palabras consecutivas de longitud W comparten $W - 2$ bits. En consecuencia, comenzando con un archivo con una longitud de N bits obtenemos $N - W + 1$ palabras. Además, hay 2^W símbolos en el alfabeto y $\alpha = (N - W + 1)/(2^W)$.
- b) Si S_W se evalúa sin superposición la cantidad de palabras de longitud W es $\text{floor}\{N/W\}$ y el parámetro de calidad se calcula como $\alpha = \text{floor}\{N/W\}/(2^W)$. Si $N \gg W$ el factor de calidad estadística es W veces más bajo que el usado con superposición.
4. En el caso de $S_{BP}^{(D)}$, hay dos procesos de agrupación involucrados.
 - a) Si ambos procesos de agrupamiento se realizan con superposición obtenemos $NW - D + 2$ elementos comenzando con un archivo N bits de longitud, y el factor de calidad es $\alpha = (NW - D + 2)/D!$. En este caso $S_{BP}^{(D)}$ tiene en cuenta las correlaciones entre $W + D$ bits consecutivos.

- b) Si el proceso de agrupación de W bits se realiza sin superposición pero la agrupación de números decimales D se realiza con superposición obtenemos $\lfloor N/W \rfloor - D + 1$ elementos y el parámetro de calidad estadística es $\alpha = (\lfloor N/W \rfloor - D + 1)/D!$. En este caso $S_{BP}^{(D)}$ incluirá correlaciones entre WD bits consecutivos.
- c) Si el proceso de agrupación de W bits se realiza con superposición y la agrupación de números decimales D se realiza sin superposición, obtenemos $\lfloor (N - W + 1)/D \rfloor$ elementos a partir de un archivo de N bits. El factor de calidad estadística es $\alpha = \lfloor (N - W + 1)/D \rfloor / D!$ y $S_{BP}^{(D)}$ tiene en cuenta correlaciones de $W + D - 1$ bits.
- d) Si ambos procesos de agrupación se realizan sin superposición, obtenemos $\lfloor \lfloor N/W \rfloor / D \rfloor$ elementos a partir de un archivo de longitud de N bits. El factor de calidad estadística es $\alpha = \lfloor \lfloor N/W \rfloor / D \rfloor / D!$ y $S_{BP}^{(D)}$ tiene en cuenta las correlaciones entre WD bits consecutivos.

La entropía de Shannon $S(P)$ es el punto de partida para otros cuantificadores:

1. Entropía normalizada $H(P)$: es la entropía de Shannon dividida por su valor máximo.

Por ejemplo, si usamos S_2 (ver arriba), se obtiene la entropía máxima para equiprobabilidad entre dos símbolos. Su valor es $S_{max} = -1/2\log(1/2) - 1/2\log(1/2) = \log(2) = 1$; entonces, la entropía normalizada es $H_2 = S_2$. Si usamos S_W la equiprobabilidad entre las 2^W posibles palabras (números decimales de W -bits) produce $S_{max} = W$ y $H_W = S_W/W$. Finalmente, para $S_{BP}^{(D)}$ la equiprobabilidad entre los $D!$ patrones de orden produce $S_{max} = \log(D!)$ y $H_{BP}^{(D)} = S_{BP}^{(D)}/\log(D!)$.

2. Entropía diferencial o condicional h y h^* son:

$$h = S_{W+1} - S_W \quad (3.16)$$

$$h^* = S_{BP}^{(D+1)} - S_{BP}^{(D)} \quad (3.17)$$

En las expresiones de arriba $W = 1, 2, \dots$ y $D = 2, 3, \dots$, $S_0 = 0$ y $S_{BP}^{(1)} = 0$. Esta entropía diferencial o condicional da la cantidad promedio de información requerida para predecir el símbolo $(W + 1)$, o $(D + 1)$, dado los W , o D símbolos precedentes.

3. Finalmente, las *rate entropies* h_0 y h_0^* [53, 70] son dadas por:

$$h_0 = \lim_{W \rightarrow \infty} h = \lim_{W \rightarrow \infty} S_W/W \quad (3.18)$$

$$h_0^* = \lim_{D \rightarrow \infty} h^* = \lim_{D \rightarrow \infty} S_{BP}^{(D)} / (D - 1) \quad (3.19)$$

3.2.5. Cuantificador de Entropías Implementado en FPGA

En esta Sección se describe la implementación de un sistema de medición de entropías. El diseño fue optimizado para ser implementado en un microcontrolador simple y pequeño, conservando una precisión aceptable. El sistema permite medir entropías a señales generadas internamente por código y a señales externas analógicas muestreadas. Se utilizó la placa de desarrollo *MIAFS-embedded kit* de ACTEL. En la FPGA (*Field Programmable Gate Array*) se instanció un microcontrolador 8051 al que se programó en lenguaje C. Se detalla el diseño del *hardware* y *software* y los resultados obtenidos. Al momento hay muy poca bibliografía sobre implementaciones en *hardware* de estos cuantificadores [16].

La entropía es empleada en diversas aplicaciones, como por ejemplo, en la detección de anomalías en flujos de datos IP [71, 72, 73]. En [74] se presentó un diseño y simulación en FPGA de un cuantificador de entropía, sin embargo, actualmente no hay disponibles implementaciones en *hardware* de este cuantificador.

En esta tesis se implementó un sistema que calcula la entropía para la PDF asociada a una serie de datos. Se analizan PDFs causales y no causales. Los datos pueden tener un origen digital (generados mediante códigos), o bien provenir del muestreo de señales analógicas. Se utilizó la placa de desarrollo *MIAFS-embedded kit*, basado en el chip *MIAFS1500* que se destaca por tener un bloque analógico embebido en el mismo encapsulado de la FPGA. Luego, se verificó la exactitud numérica del cuantificador implementado comparando sus resultados con un programa patrón. A partir del máximo error detectado se determinó la exactitud numérica del sistema.

Hardware Implementado

El diseño del *hardware* se basó en el que provee ACTEL en [75], basado en el microcontrolador 8051, interfaces y periféricos. Fue realizado con el paquete de programas *Libero Soc v11.3[©]* de ACTEL. Se utilizó la placa de desarrollo *MIAFS-EMBEDDED-KIT* que contiene una FPGA *MIAFS1500* de ACTEL y periféricos [73]. El chip *MIAFS1500*

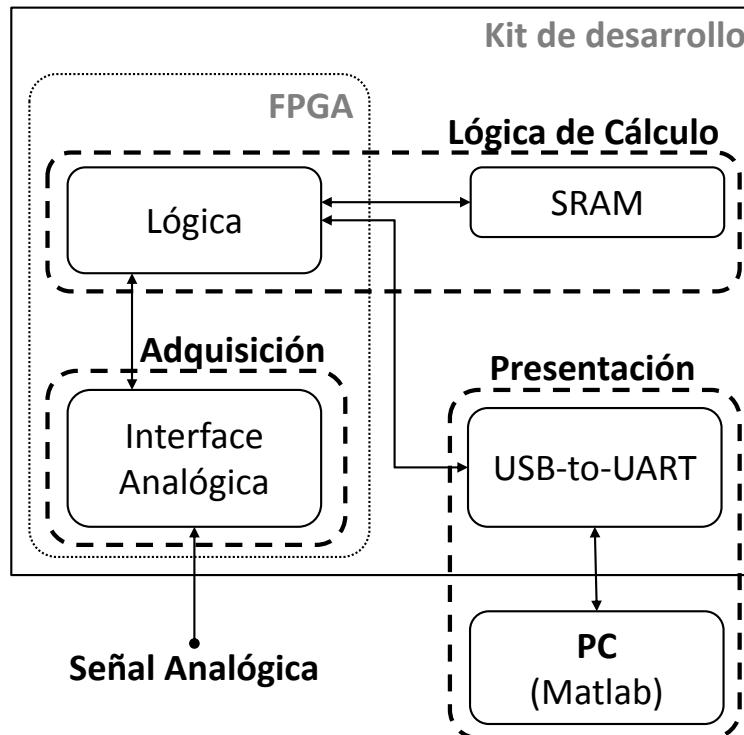


Figura 3.11: Esquema del sistema completo.

contiene embebido un bloque analógico que consiste en nueve adaptadores direccionables de cuatro entradas cada uno, un multiplexor analógico de 32 entradas y un conversor analógico-digital configurable.

El sistema implementado puede dividirse en tres etapas principales como se muestra en la Figura 3.11: una primera etapa de Adquisición de datos, que convierte a palabras digitales las señales del mundo analógico, una Lógica de Cálculo, que se vale de la memoria SRAM para llevar a cabo los cálculos y coordinar las interfaces y una etapa de Presentación de resultados, que envía los resultados de la medición a una computadora a través de la interfaz *USB-to-UART*.

1. Etapa de Adquisición:

Para ingresar los datos analógicos a ser evaluados utilizamos la entrada de tensión AV2 del *Analog Quad 2* del bloque analógico. Se encuentra direccionada en el canal siete del multiplexor analógico y fue configurada para un rango de tensiones de entrada de 0 V a 4 V. El conversor analógico-digital se configuró con una resolución

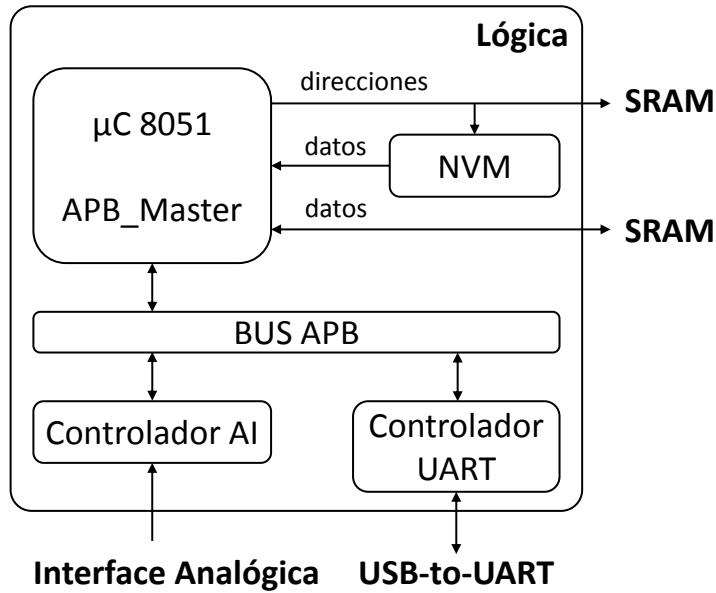


Figura 3.12: Detalle de la lógica de cálculo.

de 12 bits. En este primer prototipo la frecuencia de muestreo máxima alcanzada fue de 16 ks/s limitada por el retardo necesario en el procesamiento de la lógica.

2. Lógica de cálculo:

En esta etapa se realizan los cálculos y la sincronización entre periféricos. En la Figura 3.12 pueden verse los bloques principales que la componen.

El núcleo de la implementación es un *Core 8051* que provee ACTEL en su catálogo de librerías. Se trata de un microcontrolador que contiene la lógica principal del microprocesador 8051 de Intel, sin sus periféricos. Este micro tiene una arquitectura Von Newman con un bus de direcciones de 16 bits, lo que limita nuestro diseño a 64 KB de memoria de código y 64 KB de memoria de datos.

Sobre este microcontrolador corre el programa que realiza los cálculos presentados en la Sección 3.2. Se encarga de, a partir de los datos de entrada, obtener las PDFs (*BP* e *hist*) y de realizar los cálculos para la obtención de las entropías, según la Ecuación 3.8. El *software* implementado se describe más detalladamente en la Sección 3.2.5.

La memoria de código es una memoria no volátil (NVM) implementada con los bloques flash internos de la FPGA. Ocupa las direcciones desde 0x0000 hasta 0xFFFF y se escribe con el contenido de un archivo en formato hexadecimal durante la

compilación.

Las funcionalidades del sistema son ampliadas mediante la conexión de periféricos a través de la interfaz APB.

Para realizar la comunicación con la PC utilizamos el *Controlador UART*. La salida de este bloque es dirigida hacia afuera de la FPGA y se conecta a un chip *USB-to-UART* que se encuentra soldado a la placa del kit de desarrollo.

El bloque analógico es controlado por el *Controlador AI*, que direcciona y sincroniza sus entradas.

3. Presentación:

La etapa de Presentación de los datos involucra al chip adaptador *USB-to-UART* que se encuentra en la placa de desarrollo y es manejado tanto por el programa que corre en la FPGA como por el *software* que corre sobre la PC. El chip adaptador *USB-to-UART* es el responsable de adaptar la entrada-salida UART de la lógica a una entrada-salida USB estándar mediante la cual es posible interactuar con la PC. Por otra parte el programa que corre en la PC se encarga de la interfaz con el usuario y es descripto en detalle en la siguiente Sección.

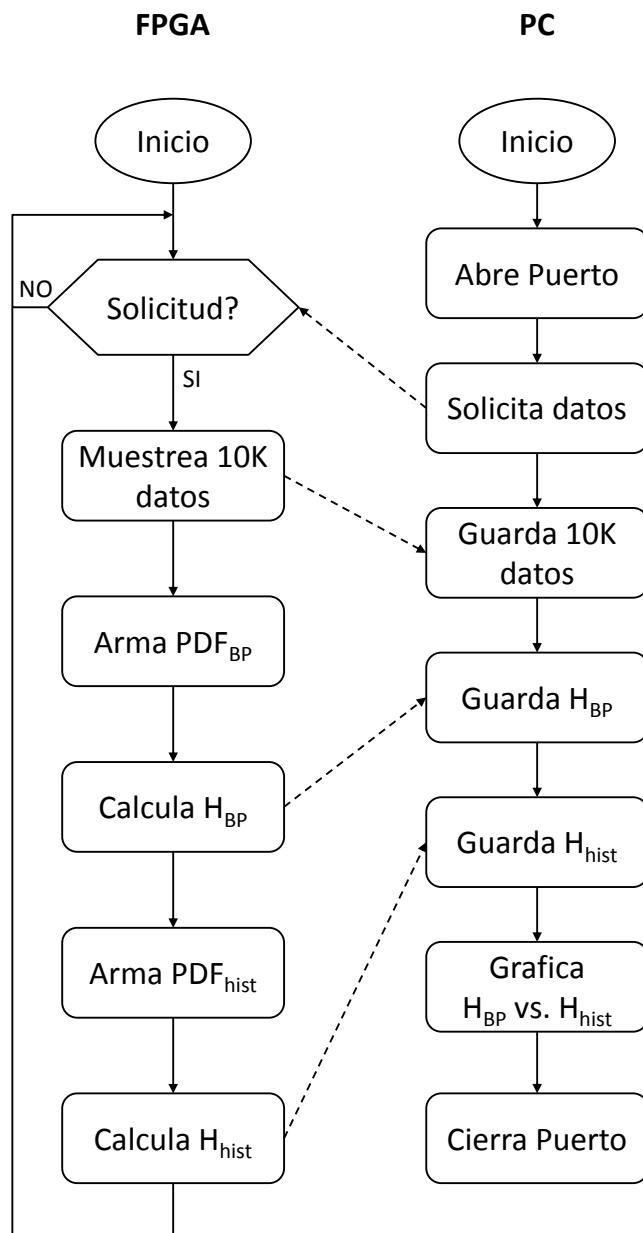
Software Implementado

El funcionamiento del sistema se logra mediante la interacción de dos programas. Uno corriendo en la PC y otro en el microcontrolador implementado en la FPGA. Puede verse un diagrama de flujo de ambos programas y la interacción entre ellos en la Figura

En la PC corre un *script* de Matlab que se encarga de abrir el puerto serie en donde se encuentra mapeado el USB, solicitar los datos, tomar los resultados del mismo puerto, graficarlos en un plano H_{BP} vs. H_{hist} y cerrar el puerto.

Sobre el microcontrolador en la FPGA corre un programa escrito en lenguaje C y compilado para el microcontrolador 8051 utilizando la herramienta *SoftConsole IDE v3.4*®. El firmware es una modificación del usado en [76]. Cuando se presenta una solicitud de datos por el puerto UART, se guardan los datos muestreados de la entrada analógica. Luego, se recorre este vector generando las PDF_{hist} y PDF_{BP} , a las que se les calcula sus respectivas entropías H_{hist} y H_{BP} . Estos resultados son enviados a la PC mediante el mismo puerto.

Con el fin de validar el sistema, el programa en la FPGA envía a Matlab el vector de datos muestreados, para que se puedan calcular en la PC sus entropías y compararlas con

Figura 3.13: Diagrama de flujo del *software* implementado.

Generador	Origen	Error H_{BP}	Error H_{hist}
Rand	Digital	$1,7421E^{-6}$	$2,6977E^{-6}$
Logístico	Digital	$0,4256E^{-6}$	$94,693E^{-6}$
Triangular	Analógico	$6,3445E^{-6}$	$2,0028EE^{-6}$
Senoidal	Analógico	$6,3151E^{-6}$	$5,6506E^{-6}$
Cuadrada	Analógico	$0,1797E^{-6}$	$1,9930EE^{-6}$
Rampa	Analógico	$245,00E^{-6}$	$1,0876E^{-6}$

Cuadro 3.1: Error de los cuantificadores evaluados en la FPGA con respecto a los resultados calculados por el programa patrón.

los resultados del sistema implementado.

Resultados

Como se dijo, para testear el sistema se compararon los resultados obtenidos por el sistema implementado y por un programa patrón que corre en la PC. Para esto, se generaron 10 000 muestras de señales con distintas formas de onda tanto externas (analógicas) como internas (digitales).

Las señales digitales fueron generadas por código en el microcontrolador, una corresponde a la función rand() de C y la otra al mapa caótico Logístico con parámetro $r=4$.

Las señales analógicas fueron generadas con el generador de funciones *HP33120A*. Tienen una amplitud de $4 V_{pp}$ y un nivel de continua de $2 V$ de forma de aprovechar todo el rango del conversor analógico-digital y aumentar la relación señal-ruido. En los cuatro casos la frecuencia de las señales fue de $100 Hz$ y la velocidad de muestreo de $16 ks/s$.

El cuadro 3.1 muestra el error absoluto entre los resultados de los cuantificadores calculados en la FPGA comparados con los resultados calculados con el programa patrón sobre los mismos datos.

La Figura 3.14 muestra los valores entregados por la FPGA en el plano H_{BP} vs. H_{hist} .

Los resultados de la compilación nos permiten conocer los recursos de la FPGA utilizados por el sistema completo y la cantidad de memoria ocupada por el *software* que corre en el microcontrolador. Recordemos que esta es una implementación de *hardware* rígida, es decir primero se arma el circuito en la FPGA (microcontrolador, periféricos, etc.) y luego se carga el *software* sobre él.

El reporte de la compilación de *hardware* devuelto por el *Place and Route* se muestra en la Figura 3.15. Podemos ver que la implementación utiliza un 19 % de los recursos lógicos de la FPGA, el 21 % de las celdas de entrada-salida y el 28 % de los bloques de memoria.

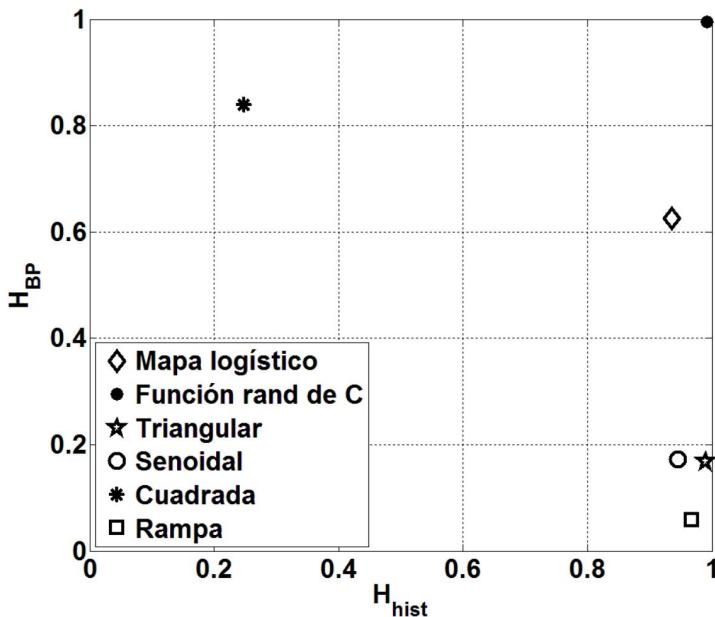


Figura 3.14: Resultados de las mediciones.

```
Core Cells : 7349 of 38400 (19%)
IO Cells : 53 of 252 (21%)
```

```
RAM/ROM Usage Summary
Block Rams : 17 of 60 (28%)
```

Figura 3.15: Recursos empleados por el *hardware* del sistema.

El reporte de la compilación de *software* se muestra en la Figura 3.16. Podemos ver que la memoria FLASH no volátil se encuentra ocupada al 15,4 %. Por otro lado, de las 47 145 direcciones de memoria SRAM ocupada, 4 096 son ocupadas por el bus APB y 43 049 por las variables del programa.

Name	Start	End	Size	Max
<hr/>				
PAGED EXT. RAM			0	256
EXTERNAL RAM	0x0000	0xb828	47145	65536
ROM/EPROM/FLASH	0x0000	0x276e	10095	65536

Figura 3.16: Recursos empleados por el *software* del sistema.

El programa debió ser adaptado al microcontrolador instanciado en la FPGA. Estas modificaciones hacen que la salida del sistema implementado no sea igual a la de un programa que corre en la PC, al cual tomamos como programa patrón. Por esto se testeó el error cometido, para tener una cota y determinar si los resultados de los cuantificadores son

correctos. El programa patrón utiliza aritmética de 64 *bits* en punto flotante norma IEEE754-64 bits y emplea la librería math.h [77]. Para el algoritmo en la FPGA se disminuyó la aritmética a 32 bits de punto flotante norma IEEE754-32 bits. También se requirió el cálculo de la función logaritmo, que se implementó mediante un algoritmo de CORDIC. En el Cuadro 3.1 se ve que el error absoluto no supera el sexto decimal, esto indica que se detecta diferencia recién a partir del quinto dígito decimal.

En la Figura 3.14 puede verse como los cuantificadores H_{BP} y H_{hist} diferencian claramente las propiedades estadísticas de las series de datos analizadas. Las señales Senoidal, Rampa y Triangular presentan un valor alto de H_{hist} porque tienen casi todos los valores que es capaz de generar el conversor Analógico-Digital. Sin embargo, la mezcla de estos datos es mala por tratarse de señales periódicas totalmente predecibles, esto se ve en el bajo valor de H_{BP} . Un caso interesante de analizar es la señal Cuadrada. El efecto del ruido aditivo es especialmente notable en las zonas en donde el valor de la señal debería ser constante. Se generan dos Gaussianas muy finas en torno a los valores ideales en la PDF_{hist} , esto no afecta demasiado el valor calculado H_{hist} , sin embargo, para la PDF_{BP} , se calcula el patrón de orden directamente a la señal ruidosa, por lo que el valor de H_{BP} es más alto que el esperado. La señal generada mediante la función rand de C, presenta las mejores propiedades estadísticas ubicándose en el punto $\sim (1, 1)$.

3.2.6. Dinámica de los ITQs con AWGN y Banda Limitada

En esta Sección exploramos la respuesta de un sistema de medición de entropías en presencia de ruido aditivo y señales filtradas. Esta inquietud surge como resultado de la implementación detallada en la Sección 3.2.5. El filtrado es inherente al ancho de banda del sistema de medición y las señales a medir siempre están contaminadas con ruido, por lo tanto, es necesario caracterizar la respuesta de nuestro sistema de medición ante estos dos procesos. Este trabajo es complementario al desarrollo de un sistema de medición de entropías implementado en FPGA.

Filtrado digital

Ya sea en la elección de un filtro como en cualquier problema de diseño en ingeniería, generalmente no es posible dar una respuesta acerca de cuál es la mejor solución. Se discute la posibilidad de la implementación de distintos filtros porque no hay un solo método de

diseño ni un sólo tipo de filtro mejor para todas las circunstancias. La elección del tipo de filtro depende de la importancia de sus ventajas aplicadas a cada problema.

Un filtro ideal es aquel en el que la respuesta en frecuencia es unitaria en el rango de las frecuencias de paso, cero en la banda de rechazo y no posee banda de transición. Dada la inherente periodicidad de la respuesta en frecuencia para tiempo discreto esta tiene la apariencia de un tren rectangular en el dominio de las frecuencias, sin embargo, en este trabajo sólo se muestra la frecuencia normalizada en el intervalo $(0, 1)$. Entonces la transferencia de un filtro pasabajos ideal en frecuencia normalizada quedaría:

$$H_{LP} = \begin{cases} 1, & |f - 0,5| > f_c \\ 0, & |f - 0,5| < f_c \end{cases} \quad (3.20)$$

Ecuación definida en el intervalo de frecuencias normalizadas $f \in (0, 1)$.

El hecho de que no podemos contar con series de valores infinitamente largas para ser filtradas, equivale a decir que disponemos de una serie de muestras enventanada. Como el producto en el dominio del tiempo equivale a una convolución en el dominio de la frecuencia, podemos estudiar el efecto que este enventanado tiene sobre la respuesta frecuencial del filtro. Consideremos la ventana más sencilla; la ventana rectangular. Supongamos que la aplicamos sobre una versión retardada de la respuesta ideal, su efecto en el dominio de la frecuencia será la convolución entre la respuesta de nuestro filtro ideal y la transformada esta ventana rectangular, es decir una función *sinc* de período $1/N$ en donde N es la cantidad de muestras que entran en la ventana.

El efecto de enventanado o truncamiento de la respuesta es doble: por una parte, la anchura del lóbulo principal está relacionada con la aparición de una banda de transición en el filtro. Por otra, la presencia de lóbulos laterales (secundarios) lleva a la aparición de un ripple u oscilaciones en la respuesta en frecuencia, en ambas bandas, (más apreciable en la banda no pasante). La aparición de los lóbulos secundarios se debe a que la ventana rectangular presenta una discontinuidad abrupta que, al pasar al dominio de la frecuencia, conlleva un reparto de la energía sobre todo el espectro a causa del aliasing.

Una opción que se plantea es generalizar el concepto de ventana y emplear ventanas más suaves que la rectangular para realizar el truncamiento de la respuesta deseada, esta técnica es una de las formas de realizar un filtro FIR. Sin embargo, si analizamos la transformada de la ventana cuadrada vemos que presenta valores nulos cada $1/N$, que son los mismos lugares en donde aparecen las componentes espectrales de la DFT. Esto significa que los

efectos de la ventana rectangular aparecen al convertir la respuesta de este filtro a tiempo continuo.

Otra opción sería diseñar un filtro analógico y transformar su respuesta a frecuencia discreta. Para esto existen fórmulas cerradas de diseño, por lo que es posible satisfacer casi cualquier especificación preestablecida. La utilización de esta técnica da como resultado un filtro IIR. Comparado con un filtro FIR, un filtro IIR requiere un orden mucho menor para cumplir las especificaciones de diseño.

En este caso se analizó la respuesta del sistema en el dominio digital. Además, es necesario filtrar las componentes espectrales de a una, lo que requiere una banda de transición muy estrecha, esto reduce el conjunto de filtros posibles. Por el lado del IIR se probó un filtro elíptico, este filtro presenta una banda de transición muy estrecha a costa de un ripple que aparece tanto en la banda de paso como en la de rechazo. Por el lado del filtro FIR se utilizó un filtro ideal con una ventana rectangular que abarca toda la serie de valores.

Resultados

Para representar la dinámica en función del filtrado, se eligieron dos señales representativas (cuadrada y senoidal) y se les calcularon los cuantificadores descriptos en la Sección 3.2 luego de ser filtrados por los filtros elegidos en la Sección anterior. Por otro lado, se calcularon los mismos cuantificadores a una señal de ruido blanco gaussiano.

En la Figura 3.17 se muestra el procedimiento utilizado. Primero se generó un vector de ruido blanco gaussiano de $N = 50\,000$ muestras, la desviación estándar σ es variable y se logra multiplicando al vector inicial de $\sigma = 1$ por la desviación estándar elegida. Luego se genera la señal determinística de $N = 50\,000$ muestras, período $T = 100$ muestras y amplitud unitaria, que se sumó al ruido para lograr la señal contaminada. La señal resultante se filtra para luego calcular cuantificadores. Como se explicó más arriba, para calcular la entropía de valores se genera el histograma de valores y se lo normaliza para calcular la Función PDF_{hist} a la que se le calcula la entropía de Shannon normalizada que da como resultado la entropía de valores normalizada H_{hist} . Para calcular la entropía de patrones de orden se utilizó el histograma de patrones de orden que cuando se normaliza se consigue la función densidad de probabilidad de patrones de orden PDF_{BP} , a la que se le calcula la entropía de Shannon normalizada para conseguir la entropía de orden H_{BP} .

Para evaluar la contribución de cada componente espectral a las entropías, se evaluaron dos filtros. Primero se aplicó un filtro elíptico de orden 10 con ripple pasabanda de $0,5\text{dB}$,

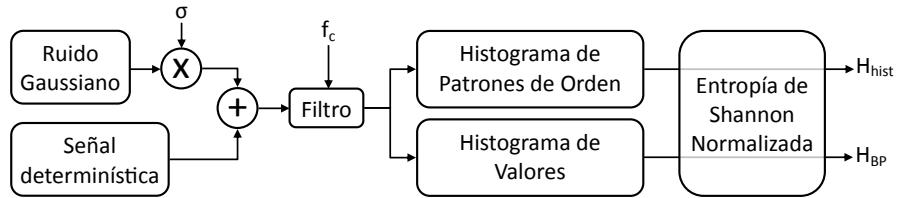


Figura 3.17: Diagrama de flujo del experimento.

ripple en la banda de rechazo de $100dB$ y frecuencia de corte variable f_c , en la Figura 3.18 se muestra su respuesta en ganancia (Figura 3.18b) y fase (Figura 3.18c) para el caso de $f_c = 0,5$. De esta forma se logra un filtrado lo suficientemente abrupto como para considerar que a medida que se barren distintas frecuencias de corte se eliminan componentes espectrales individualmente. Los resultados de este filtrado se compararon con los resultados de un filtro ideal (Figura 3.19), que consiste en una máscara aplicada a la transformada de Fourier de la señal a filtrar, de esta manera se consigue el espectro de la señal filtrada, el cual es antitransformado para recuperar la versión filtrada en las muestras. El diagrama de este filtro puede verse en la Figura 3.19a. Este procedimiento equivale a un filtrado ideal sin retardo, por lo que el bode de amplitud es $0dB$ en la banda de paso y $-\infty dB$ en la banda de rechazo (Figura 3.19a); la fase $\omega\tau = 0$ es lineal con pendiente nula (Figura 3.19c).

Primero se aplicó una señal de ruido blanco gaussiano, es decir que la señal determinística es cero y la desviación estándar de la gaussiana unitaria. En la Figura 3.20 se muestra el resultado de los cuantificadores a medida que se va barriendo la frecuencia de corte del filtro elíptico. En la Figura 3.20a se muestra la entropía del histograma de valores H_{hist} , puede verse que su valor se mantiene constante alrededor de 0,9 tanto para el filtro pasa-bajos (roja) como el pasa-altos (azul), este valor es el mismo que resulta de calcular la entropía del histograma de valores a la señal sin filtrar (resultado que se muestra con una línea punteada negra en el mismo gráfico). También puede verse que cuando la frecuencia de corte del filtro elíptico se acerca a los extremos el valor del cuantificador cae, en estas frecuencias el método numérico que calcula el vector filtrado diverge debido a la precisión finita. En la Figura 3.20b se muestra la entropía de los patrones de orden, H_{BP} se mantiene en valores bajos cuando el filtro (pasa-altos en azul y pasa-bajos en rojo) deja pasar pocas componentes espectrales. Luego, a medida que la frecuencia de corte deja pasar más componentes espectrales, el cuantificador tiende a 1, que es justamente el valor que arroja

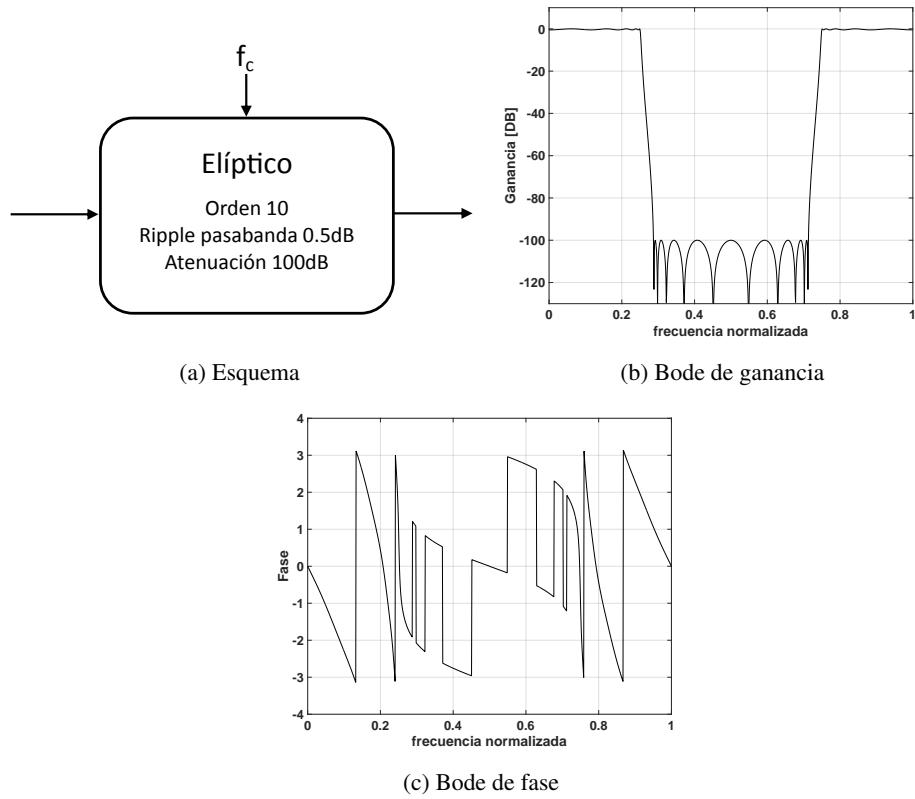


Figura 3.18: Filtro elíptico.

cuando se ingresa con la señal sin filtrar (este valor está indicado con una línea punteada negra). El cuantificador detecta los cambios en la forma de la señal a medida que es filtrada. Por último, en el plano $H_{hist} - H_{BP}$ de la Figura 3.20c se compacta la información de ambos cuantificadores, aunque se pierde la noción de la frecuencia de corte.

En la Figura 3.21 se muestran los resultados del mismo procedimiento, pero cuando se aplica un filtro ideal. El comportamiento de los cuantificadores es igual al del filtro elíptico en todos los casos con la diferencia que el método no diverge cuando $f_c \rightarrow 1$ o $f_c \rightarrow 0$. Pueden verse por lo tanto los valores que arrojan los cuantificadores en los extremos de la frecuencia de corte. La entropía no causal de la Figura 3.21a aumenta levemente en los extremos, en donde el histograma de valores deja de tener una distribución gaussiana y se aplana levemente. También puede verse en la Figura 3.21b que la entropía de valores $H_{BP} \rightarrow 0,15$ cuando $f_c \rightarrow 0$ para el pasa-bajos (rojo) y para el pasa-altos (azul) $H_{BP} \rightarrow 0,22$ cuando $f_c \rightarrow 1$. En este caso es fácil comparar la sensibilidad al filtrado de ambos cuantificadores, en el plano doble entropía de la Figura 3.21c. El círculo blanco muestra la posición en este

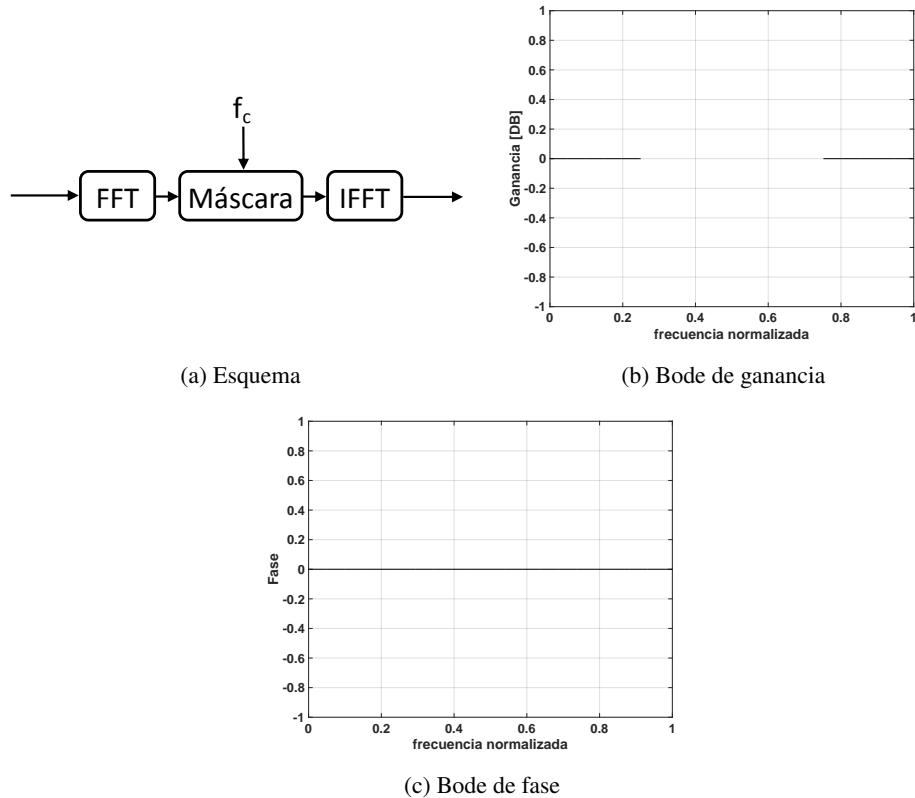


Figura 3.19: Filtro ideal.

plano cuando ningún filtro es aplicado, podemos ver que el apartamiento en el eje vertical aumenta a medida que la serie es filtrada, mientras que no se aparta en el sentido horizontal. Esto muestra que la sensibilidad al filtrado de H_{BP} es mucho mayor que la de H_{hist} .

Para el sistema planteado no se necesita volver al dominio continuo analógico, por lo que las dificultades mencionadas en la Sección 3.2.6 respecto al filtrado ideal (como ripple en las bandas de paso y rechazo) no aplican a este caso. Por este motivo para esta serie de pruebas elegimos el filtro ideal, dado que presenta mejores resultados que el elíptico.

La primera señal determinística que se muestra es una senoidal de amplitud unitaria con 100 muestras por período, los resultados pueden verse en la Figura 3.22. Mientras la única componente espectral no es filtrada, el valor de la entropía de valores es $H_{hist} \approx 0,57$ en la Figura 3.22a y la entropía de patrones de orden $H_{BP} \approx 0,16$ en la Figura 3.22b. Ambos cuantificadores caen a cero cuando la única componente espectral es filtrada, ya sea por el filtro pasa-bajos (azul) o por el pasa-altos (rojo). El plano doble entropía muestra un punto en $(0,57; 0,16)$ para la senoidal sin filtrar y otro en $(0; 0)$ cuando la única componente

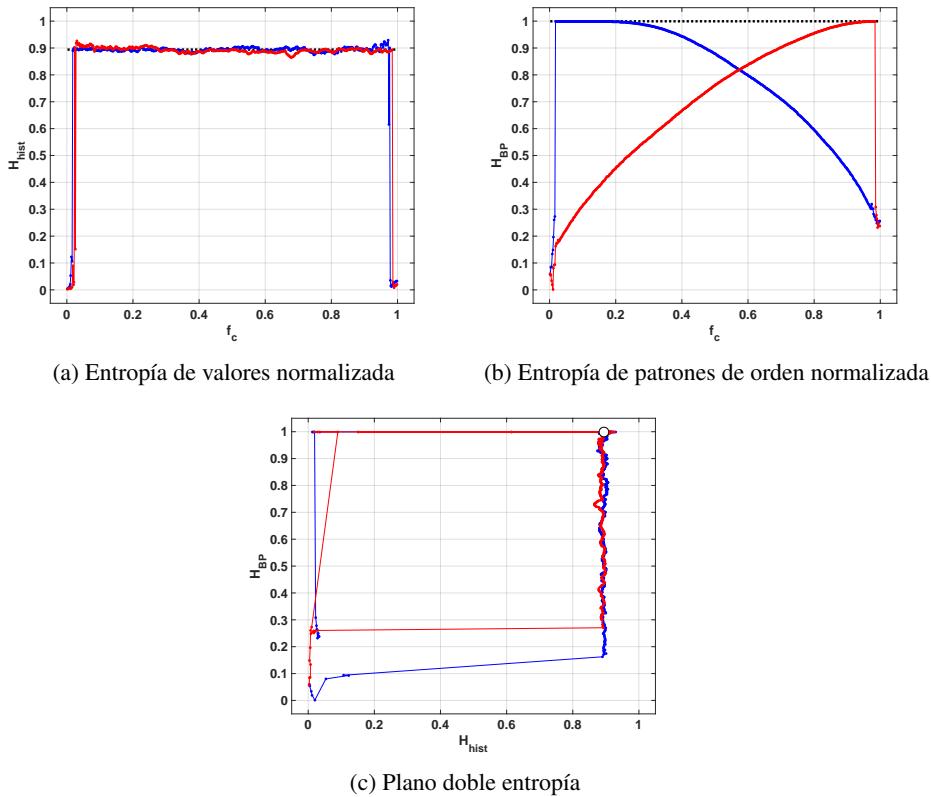


Figura 3.20: Cuantificadores calculados sobre la salida del filtro elíptico cuando se ingresa con ruido blanco gaussiano.

espectral es filtrada.

La salida de los cuantificadores cuando esta señal es contaminada con ruido gaussiano aditivo con $\sigma = 0,2$ puede verse en la Figura 3.23. Vemos en la Figura 3.23a que la entropía de valores aumenta cuando el filtrado no elimina la componente espectral, dando valores incluso por encima del valor de la entropía de la señal gaussiana. Esto se debe a que la PDF de amplitudes de la señal senoidal es complementaria con la de la señal gaussiana, entonces la PDF de amplitudes de la resultante es más parecida a la del ruido uniforme. Para los patrones de orden de la Figura 3.23b, el filtro pasa-altos no deja ver un cambio significativo debido a que la componente espectral de la senoidal es eliminada en la zona en la que su entropía es alta. El filtro pasa-bajos en cambio muestra que mientras esta componente está presente el valor de la entropía es asintótico a el valor $H_{BP} \rightarrow 0,16$ a medida que la frecuencia de corte disminuye. Recordemos que $H_{BP} \approx 0,16$ es el valor de la entropía de patrones de orden de la señal senoidal limpia. En el plano doble entropía (Figura 3.23c) se

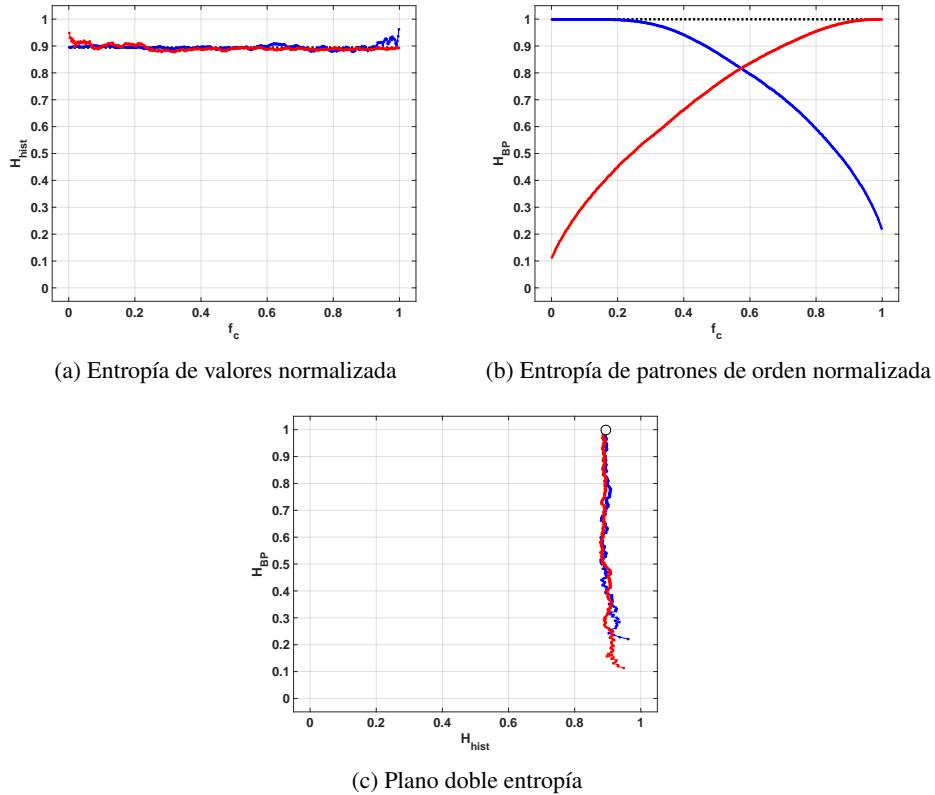


Figura 3.21: Cuantificadores calculados sobre la salida del filtro ideal cuando se ingresa con ruido blanco gaussiano.

ve que ambos cuantificadores son complementarios, en el sentido que la entropía de valores detecta la presencia o no de la señal determinística mientras que la entropía de patrones de orden detecta el efecto del filtrado sobre la señal de ruido.

En la Figura 3.24 se muestran los resultados cuando la señal determinística es una cuadrada de amplitud unitaria sin ruido y 100 muestras por período. Tanto la entropía de valores H_{hist} como la entropía de patrones de orden H_{BP} presentan una forma escalonada, sus valores se mantienen constantes a medida que se barre la frecuencia de corte de los filtros hasta que la siguiente componente espectral es filtrada. También se ve que en ambos casos los valores resultantes se mantienen bastante lejos del valor sin filtrar, el cual es indicado con una línea negra punteada.

El caso contaminado con ruido (Figura 3.25) cambia respecto del caso sin contaminar. En la Figura 3.25a se ve que para el filtro pasabajos (rojo) H_{hist} se mantiene alrededor del valor sin filtrar (línea punteada), excepto con las tres frecuencias más bajas, en donde

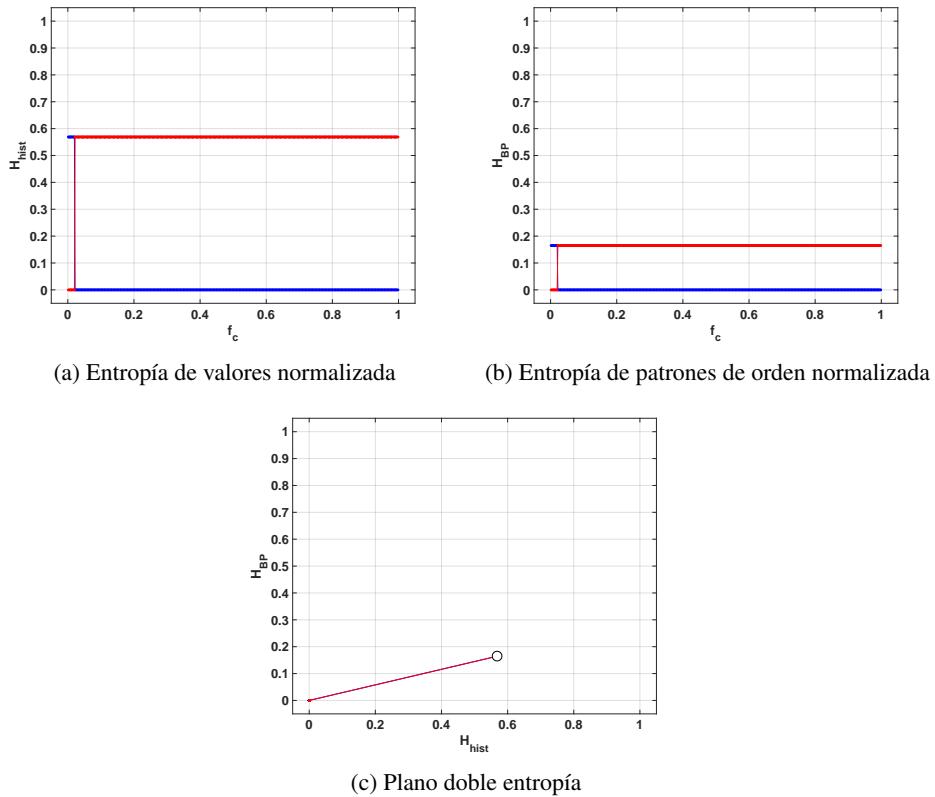


Figura 3.22: Cuantificadores calculados sobre la salida del filtro ideal cuando se ingresa con una señal senoidal limpia.

su valor aumenta un levemente por las mismas razones que aumentaba con la senoidal contaminada. Algo parecido sucede con H_{BP} en la Figura 3.25b. Cuando la señal cuadrada se contamina con ruido su valor se mantiene cercano al del ruido gaussiano, esto es porque en las regiones en las que la señal cuadrada es plana su contribución al patrón de orden es nula. Para el filtro pasa bajos se ve un escalonado en la posición de cada componente espectral que se hace más notorio para las frecuencias más bajas, en donde la contribución del ruido ya es bastante baja y a la vez se encuentran las componentes espectrales de mayor peso. Esto no es tan notorio en el filtro pasa-altos, en este caso cuando la contribución del ruido es de baja amplitud también lo es la de la señal determinística, enmascarando este fenómeno.

Para caracterizar el comportamiento de los cuantificadores frente a la amplitud de ruido, se generaron señales cuadradas contaminadas con AWGN de dos amplitudes y se filtraron para calcular los cuantificadores. En la Figura 3.26 se muestran ambos cuantificadores

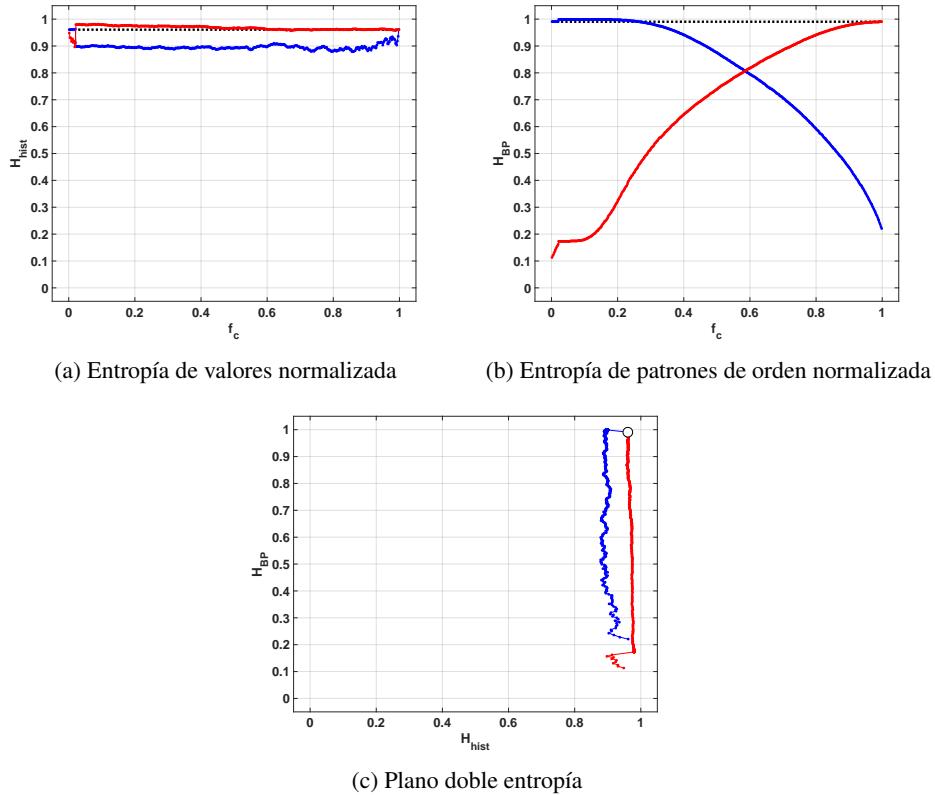


Figura 3.23: Cuantificadores calculados sobre la salida del filtro ideal cuando se ingresa con una señal senoidal ruidosa.

cuando se hace variar el ruido con valores de la desviación estándar $\sigma = [0 \ 0,1 \ 1]$. Cuando comparamos las Figuras 3.26a, 3.26b y 3.26c vemos un cambio significativo cuando pasamos de la señal limpia de 3.26a a la contaminada con bajos niveles de ruido de la 3.26b, sin embargo, cuando pasamos del bajo nivel de ruido de la Figura 3.26b al de la Figura 3.26c el cambio es mucho más sutil. De modo similar, entre las Figuras 3.26d y 3.26e hay muy pocas similitudes, mientras que las Figuras 3.26e y 3.26f son bastante similares. En este segundo caso es más evidente la diferencia cuando cambia el nivel de ruido, con bajos niveles puede verse el escalonado que aparece cada vez que una frecuencia es filtrada, mientras que cuando la amplitud de ruido es mayor este escalonado aparece solo en el pasabajos para las tres primeras frecuencias, que resultan ser las de mayor peso.

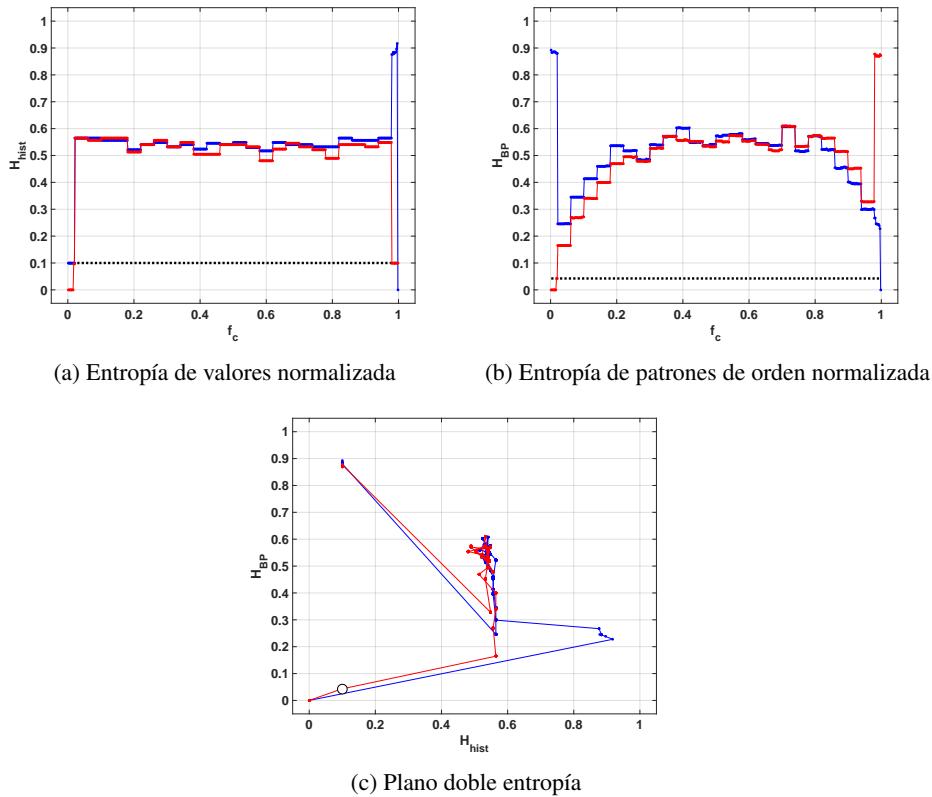


Figura 3.24: Cuantificadores calculados sobre la salida del filtro ideal cuando se ingresa con una cuadrada limpia.

3.3. Conclusiones

En este Capítulo se presentaron las principales herramientas utilizadas para detectar caos y cuantificar la calidad estadística de los generadores de números aleatorios. Junto con la introducción teórica, se mostraron algunos avances en la implementación de dichas herramientas.

El algoritmo evolutivo desarrollado detecta con precisión el máximo MLE del sistema en cada región en el espacio de parámetros del conocido oscilador Logístico. La búsqueda exhaustiva del MLE barriendo todos los valores de parámetros se vuelve muy complicada cuando aumenta el número de parámetros. Esta es la razón por la cual se empleó un algoritmo genético en este trabajo. Este algoritmo heurístico permite encontrar las áreas de interés, $MLE > 0$, de una manera más rápida y simple. En la implementación de hardware del cálculo MLE, hemos explotado la naturaleza paralela de las ecuaciones de cálculo

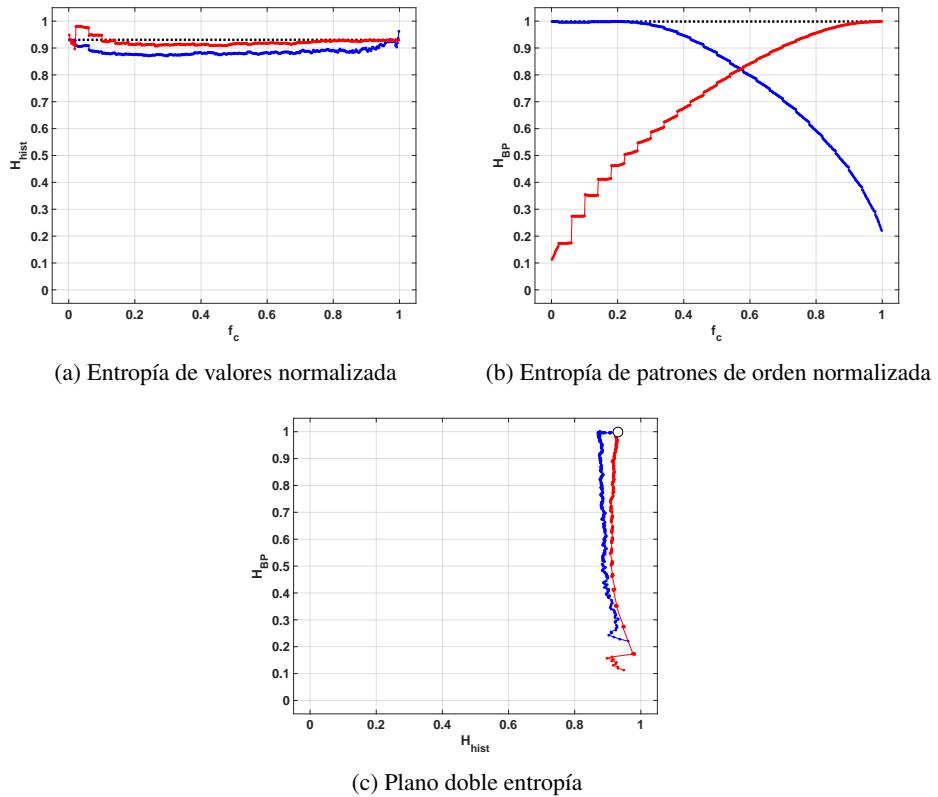


Figura 3.25: Cuantificadores calculados sobre la salida del filtro ideal cuando se ingresa con una cuadrada ruidosa.

del MLE con el objetivo de optimizar el diseño de arquitectura propuesto, permitiendo su implementación concurrente basada en tecnología FPGA.

Se desarrolló e implementó un sistema que permite medir con buena precisión las entropías causales y no-causales de señales analógicas provenientes del exterior de la FPGA y también internas generadas por código. Se logró medir señales y realizar cálculos complejos con un microcontrolador modesto como el 8051 instanciado en la FPGA AFS1500 de ACTEL. Este prototipo cumple con las especificaciones de precisión y cantidad de recursos requeridos establecidas en el diseño. El código de este sistema ocupa el 15,4% del total de la memoria flash del microprocesador instanciado. En cuanto a los recursos disponibles en la FPGA se utilizaron 7 349 celdas lógicas, quedando casi el 80% de los recursos de *hardware* disponibles para implementar los sistemas bajo prueba en forma concurrente.

También se exploraron las fuentes de error del medidor de entropías implementado en

FPGA. Para este primer análisis se evaluó qué sucede al aplicar un filtro abrupto, es por esto que elegimos para comparar un filtro elíptico y uno ideal. Las respuestas del filtro elíptico y del ideal fueron muy similares en el rango de frecuencias en los que el elíptico tiene un buen comportamiento, sin embargo cuando la frecuencia de corte del elíptico se acerca a los extremos (es decir cuando $f_c \rightarrow 0$ o $f_c \rightarrow 1$) la salida del filtro diverge. El problema se debe a que el método numérico utilizado para calcular la salida del filtro diverge por la precisión finita utilizada. Como todo el análisis se realiza en el dominio digital, no necesitamos volver a la frecuencia continua, por lo que nos quedamos con los resultados del filtro ideal para hacer las pruebas, sin tener que preocuparnos por el ripple que aparece en las bandas de paso y rechazo cuando pasamos al mundo analógico. Cuando se compararon las respuestas de los cuantificadores con y sin ruido, se vio que las señales limpias tienen mesetas, es decir que se mantienen constantes hasta que el filtrado elimina la siguiente componente espectral. Sin embargo, cuando están contaminadas con ruido los cuantificadores cambian para parecerse más a los resultados que arroja el ruido blanco gaussiano sin ninguna señal determinística. En todos los casos se vio que estos cuantificadores son muy sensibles a la presencia de ruido, lo que nos permitió vincular a este hecho los errores en la medición. También se vio que los valores cambian a medida que se filtra la señal sin contaminar, lo que agrega una segunda fuente de error dada por el ancho de banda finito del sistema.

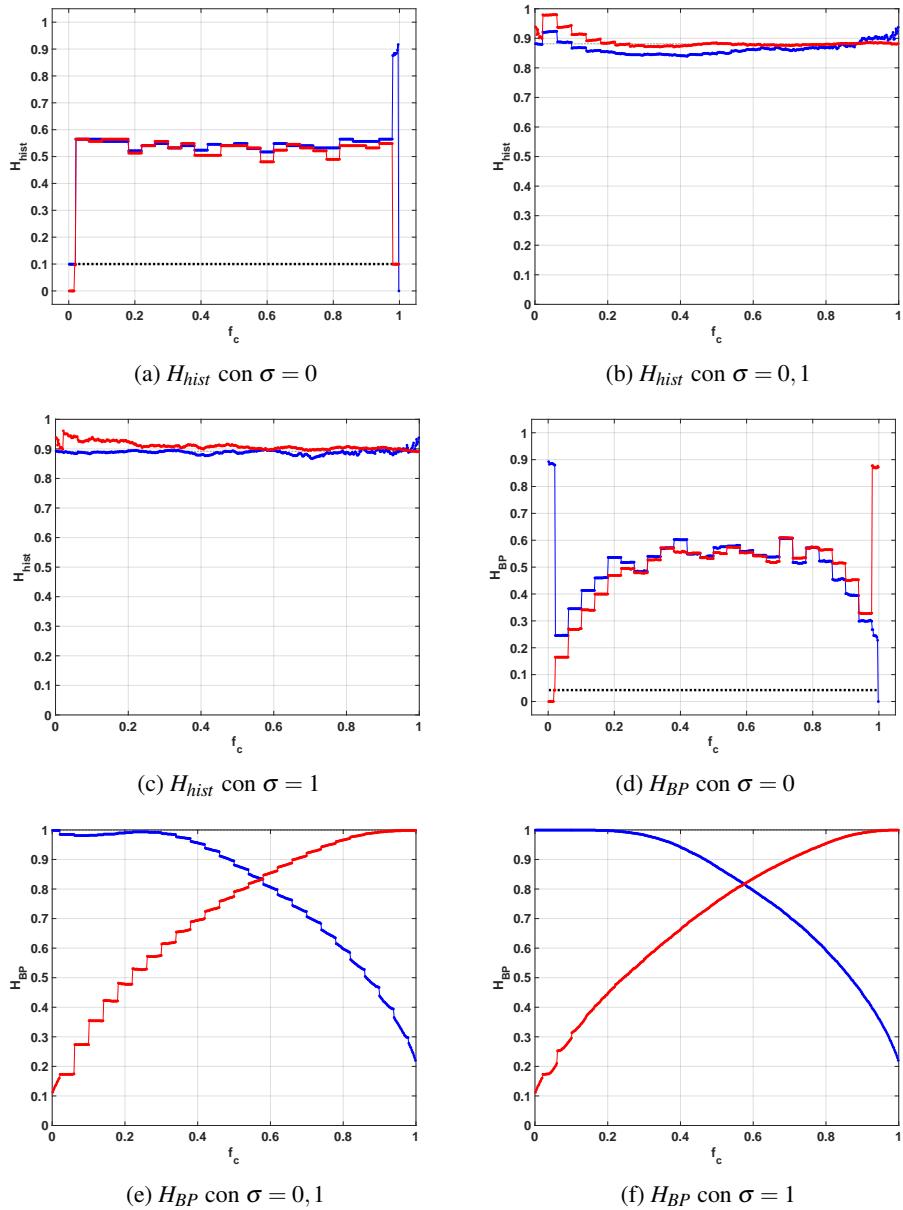


Figura 3.26: Cuantificadores calculados sobre la salida del filtro ideal cuando se ingresa con cuadradas contaminadas con AWGN con amplitudes de ruido $\sigma = [0 \ 0,1 \ 1]$.

Capítulo 4

Generadores de Números Aleatorios Usando Caos

En los últimos treinta años, los sistemas caóticos han producido una revolución en nuestra visión de la naturaleza ya que tienen dos características contrastantes: (1) son deterministas ya que su dinámica está determinada por un modelo matemático, pero (2) debido a su sensibilidad a las condiciones iniciales, se pierde la predicción a largo plazo y, en consecuencia, pueden incluirse en la clase de sistemas estocásticos que se estudian mediante herramientas estadísticas. Estos sistemas pueden generar señales estocásticas a partir de modelos simples que son fáciles de implementar a través del software o hardware apropiado.

Esta *dualidad determinista-estocástica* hace que los sistemas caóticos sean especialmente interesantes para las aplicaciones de ingeniería, en la medida en que las señales generadas pueden usarse como ruidos controlados en una amplia gama de aplicaciones. Por lo general, se requiere una manipulación adecuada de las series temporales que generan estos modelos para mejorar sus propiedades estadísticas. Esto se debe a que las secuencias caóticas presentan correlaciones internas no lineales, por lo tanto, es necesario utilizar técnicas de aleatorización romper estas correlaciones y para mejorar la aleatoriedad de la serie [39]. La determinación del grado de estocasticidad tiene como objetivo proporcionar una metodología de diseño optimizada para una aplicación particular.

Así, por ejemplo, hay aplicaciones que requieren que el sistema caótico reemplace un sistema estocástico (criptografía [78], generadores de secuencia para espaciar el espectro en

comunicaciones de espectro esparcido [79, 24], generadores de números pseudoaleatorios [80, 81, 58], reducción de la interferencia electromagnética [82], etc.). Por otro lado, algunas aplicaciones requieren previsibilidad a largo plazo, por ejemplo, para reproducir el sistema caótico de la manera más precisa posible, este es el caso de las comunicaciones analógicas que usan señales caóticas como de portadoras [83, 84].

Un problema es la determinación exacta del período de la secuencia pseudoaleatoria. Para generadores que se basan en operaciones lineales, el problema se ha estudiado en profundidad y existen criterios de diseño bien conocidos para obtener dispositivos que generen secuencias de máximo período. Ejemplos de algoritmos lineales son: el algoritmo Mersenne Twister que es un generador de números aleatorios muy rápido de período $T = 2^{19937} - 1$ [85] y Multiply-With-Carry (MWC), que es un método inventado por George Marsaglia para la generación de secuencias aleatorias de números enteros basados en un conjunto inicial de dos a miles de valores de semilla elegidos al azar, presenta períodos inmensos, que van desde alrededor de 2^{60} a $2^{2000000}$ [86]. Sin embargo, desde el punto de vista criptográfico son débiles.

Cuando se trata de aplicaciones criptográficas, no se recomienda utilizar métodos lineales para generar secuencias pseudoaleatorias (como LFSR, LCG o sus combinaciones adecuadas), ya que hay disponibles algoritmos eficientes para predecir la secuencia a partir de observaciones relativamente cortas [87]. Por otro lado, para la mayoría de las familias de generadores no lineales, el problema parece ser intratable y, con pocas excepciones, no existe un análisis analítico de sus períodos [88].

En realidad, si los sistemas caóticos pudieran implementarse con una precisión infinita, serían deterministas en sentido estricto. Sin embargo, solo disponemos de computadoras y dispositivos digitales, que pueden representar internamente las señales con una cantidad finita de bits, esto significa que los valores se describen usando aritmética de precisión finita. Esta restricción es crítica para un sistema caótico ya que es extremadamente sensible a la aritmética empleada, estos dispositivos solo pueden generar atractores *pseudocaóticos*, en el mejor de los casos. En consecuencia la discretización es un proceso no trivial ya que puede destruir el comportamiento pseudocaótico del sistema original.

Otro de los problemas fundamentales, desde el punto de vista de la implementación en hardware, es la optimización de recursos. Continuamente aparecen nuevas formas de implementar los sistemas para reducir área y potencia.

En este Capítulo se resumen varios trabajos propios orientados a la implementación

de sistemas caóticos en hardware. Primero, en la Sección 4.1, se explora la la posibilidad de implementar redes neuronales con comportamiento caótico. Este tipo de sistemas es interesante por presentar un comportamiento autónomo, que puede ser implementado de forma independiente al resto del circuito. Luego, en la Sección 4.2 se propone un nuevo esquema de codificación basado en los mapas cuadráticos bidimensionales presentados en la Sección 2.3.3. Estos mapas presentan distintos atractores con propiedades muy diferentes según sus 12 parámetros, que pueden ser usados como llave, lo que permite mantener la estructura del circuito y generar salidas pseudoaleatorias muy distintas según sea la llave utilizada. Como se dijo más arriba, la precisión numérica es un factor que puede determinar la caoticidad y la estocasticidad de los sistemas caóticos digitalizados. Estos resultados fueron presentados en [18]. En las Secciones 4.3 y 4.4 se explora la degradación de las propiedades estadísticas de los sistemas caóticos cuando son implementados en hardware digital. Primero, en la Sección 4.3 se estudió el comportamiento del sistema de Lorenz utilizando distintos tipos de representación numérica. Este análisis fue publicado en [19]. Se utilizaron estrategias de aleatorización ya que, en este caso, el sistema está orientado a la generación de números pseudoaleatorios. En este caso se utilizaron herramientas estándar de uso libre para evaluar la estocasticidad del sistema resultante. Luego, en 4.4, se realizó un análisis exhaustivo de los atractores y regiones de atracción para los mapas cuadráticos bidimensionales presentados en la Sección 2.3.3. Estos resultados fueron publicados en [20].

4.1. Caos en Redes Neuronales

El problema del caos en las redes neuronales ha recibido mucha atención recientemente [89]. Las actividades en este campo pueden ser divididas en tres categorías:

- Estudio experimental del comportamiento aperiódico observado en una sola neurona perturbada o en un pequeño ensamble de neuronas.
- Estudio del comportamiento temporal complejo del cerebro y los posibles roles del caos en el procesamiento de la información.
- Estudio de las rutas al caos y las propiedades de los atractores caóticos en en modelos de redes neuronales.

Las redes neuronales artificiales proveen soluciones efectivas a problemas en diversos campos, en particular, pueden servir como generadores de señales caóticas. Las aplicaciones de señales caóticas son muy diversas, pero en este caso son especialmente atractivas ya que en los algoritmos de aprendizaje se realiza una búsqueda aleatoria, entonces un generador neuronal de caos puede ser parte de la red neuronal determinística que se está entrenando.

4.1.1. El Modelo de Hopfield

Una de las piedras fundamentales para el reciente renacimiento en el campo de las redes neuronales fue el modelo asociativo propuesto por Hopfield en 1982. La aproximación de Hopfield es un enfoque teórico para pensar ensambles entre unidades de cómputo [37].

El perceptrón multicapa es una Red Neuronal Artificial (RNA) formada por capas de neuronas. Las neuronas pueden pertenecer a la capa de entrada, capas ocultas o capa de salida. Estas neuronas no poseen memoria, por lo que su salida depende del estado de sus entradas en el instante actual (no tienen retardo), además, como el nombre de sus capas lo sugiere, las conexiones son unidireccionales y jerárquicas. Es por esto que la matriz de pesos tiene solo algunos valores distintos de cero, no hay conexiones hacia atrás, ni en la misma capa, ni sobre la misma neurona, ni saltándose capas. En la Figura 4.1 se ve un ejemplo para un perceptrón pequeño y su matriz de pesos.

Al contrario de los perceptrones multicapa, los sistemas adaptativos y los mapas auto-organizados, las redes de Hopfield sí tienen realimentación entre neuronas. Este tipo de arquitectura tiene como campo principal de aplicación la optimización de procesos. Se basa en el planteamiento de una memoria asociativa, es decir que el estado actual de una neurona depende de su historia y de la de las neuronas asociadas a ella; se hace necesario entonces definir una función de energía. Además, se destaca la facilidad de implementación en FPGA y VLSI.

Esta red recurrente se basa en almacenar información en un sistema que presenta una configuración dinámica estable, es decir, se plantea como una memoria asociativa o memoria direccionable por contenido. Intuitivamente, la idea de Hopfield es localizar cada patrón que se requiere almacenar a la red en el fondo de un valle de la función de energía. Se parte de un determinado estado inicial (información de partida) tras lo cual se deja evolucionar el sistema hasta llegar a un estado estable. Este estado estable será el patrón que se corresponde con el estado inicial (reconocimiento de patrones).

Hopfield, en su trabajo destaca tres diferencias con el perceptrón multicapa:

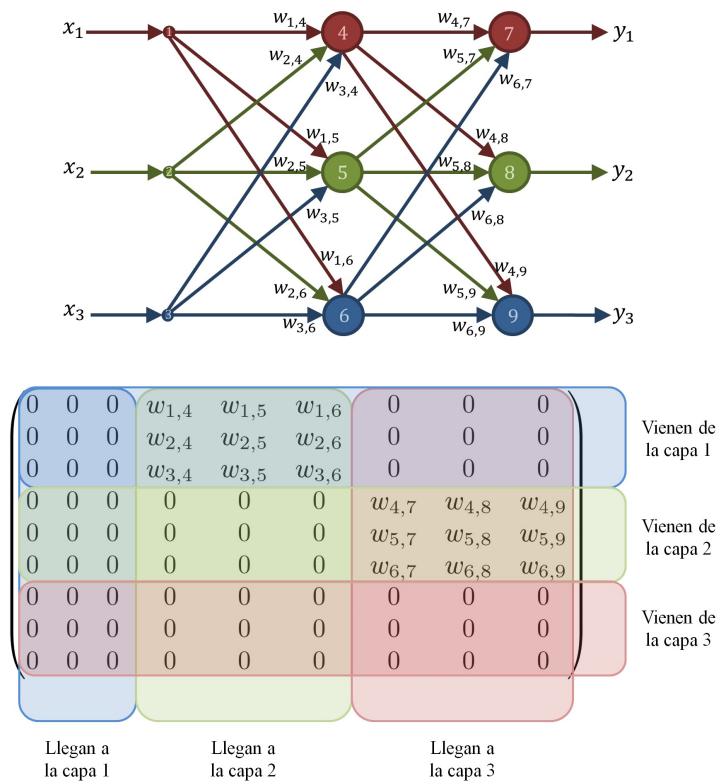


Figura 4.1: Perceptrón multicapa y matriz de pesos asociada. Puede verse que la topología de la red y la configuración de la matriz de pesos son biunívocas.

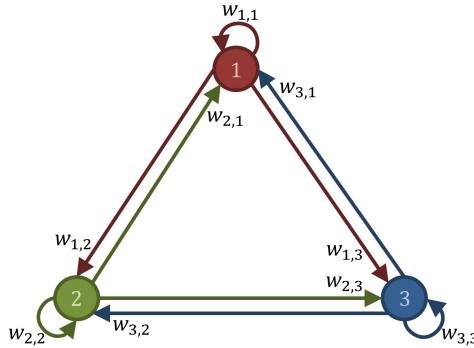


Figura 4.2: Red de Hopfield. Ahora, la matriz de pesos tiene todos sus valores permitidos.

- Su modelo incluye realimentaciones, que son fundamentales en su modo de funcionamiento.
- La elección de la arquitectura del perceptrón multicapa se realiza en forma arbitraria.
- El perceptrón multicapa funciona de manera síncrona, es decir, todas las neuronas cambian al mismo tiempo. La red de Hopfield permite un funcionamiento tanto síncrono como asíncrono, aunque el funcionamiento asíncrono es el más habitual en las neuronas biológicas.

El grafo de la red cambia con respecto al perceptrón multicapa, la representación no es la de un grafo separable por capas con conexiones hacia adelante, sino la de un grafo completo como se ve en la Figura 4.2.

4.1.2. Estudio de la RNA en Función de un Parámetro

La red neuronal usada tiene siguiente el modelo de tiempo continuo:

$$\dot{u} = -u + W \cdot f(u); u \in \mathbb{R}^3 \quad (4.1)$$

en donde u es un vector de tres dimensiones, W es la matriz de pesos y f es la función de activación

$$u = \begin{pmatrix} x \\ y \\ z \end{pmatrix}; W = \begin{pmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ w_{2,1} & w_{2,2} & w_{2,3} \\ w_{3,1} & w_{3,2} & w_{3,3} \end{pmatrix}; f = \begin{pmatrix} \arctan x \\ \arctan y \\ \arctan z \end{pmatrix} \quad (4.2)$$

Las Ecuaciones 4.2 se corresponden con el diagrama de la Figura 4.2. De la ecuación que se corresponde con una red de Hopfield de memoria diferencial. No disponemos de computadoras analógicas, por lo que el sistema debe ser convertido a tiempo discreto. Aunque el paquete de programas Matlab incluye rutinas para el cálculo de ecuaciones diferenciales, perdemos el control de paso de tiempo necesario para calcular el exponente de Lyapunov con el método descripto en la Sección 3.1. Por lo tanto, se emplea la aproximación de Euler de primer orden, en donde la derivada se aproxima con un trapecio de base Δt .

$$\begin{aligned} \frac{u_{n+1} - u_n}{\Delta t} \approx \dot{u}_n &= -u + W \cdot f(u_n) \Rightarrow \\ \Rightarrow u_{n+1} &= (1 - \Delta t)u_n + \Delta t W \cdot f(u_n) \\ &= Gu_n + \Omega f(u_n) \end{aligned} \quad (4.3)$$

En la Figura 4.3 se muestra el diagrama de la red neuronal diseñada en tiempo discreto. Sus coeficientes dependen del paso de tiempo. Este sistema se aproxima al de tiempo continuo en el límite $\Delta t \rightarrow 0$, se verificó que el sistema converge al planteado. Pudo verse que con $\Delta t = 1$ y $\Delta t = 0,1$ las soluciones en el espacio de fases fueron las mismas, para hacer los cálculos se empleó $\Delta t = 0,01$.

Se barrió un parámetro (peso de un axón) para identificar la existencia de caos en función de éste. Siguiendo a [89] en donde se reporta una transición al caos en torno a un juego de parámetros, utilizamos la siguiente matriz de pesos:

$$W = \begin{pmatrix} 2 & -1,2 & 0 \\ 1,9 + p & 1,71 & 1,15 \\ -4,75 & 0 & 1,1 \end{pmatrix} \quad (4.4)$$

en donde p es el parámetro a barrer entre $-0,35$ y $0,55$ en pasos de 9×10^{-5} .

Para cada valor del parámetro se le da condiciones iniciales al sistema $[1,68; -0,292; -3,47]$ y se lo deja evolucionar 800s, esto es 200s más que el transitorio más largo reportado en [89], con esto nos aseguramos de descartar el transitorio y que el sistema se encuentra en régimen permanente. Se calcula el MLE para $t \in (800; 1\,000]$.

De esta forma se genera la Figura 4.4 en donde se muestra el MLE en función del parámetro. Como es usual, el MLE no es una función suave, sino que es una función discontinua que presenta saltos abruptos en todo el dominio, sin embargo, se encontraron zonas de caos robusto frente al parámetro p en algunos intervalos, especialmente en $p \in$

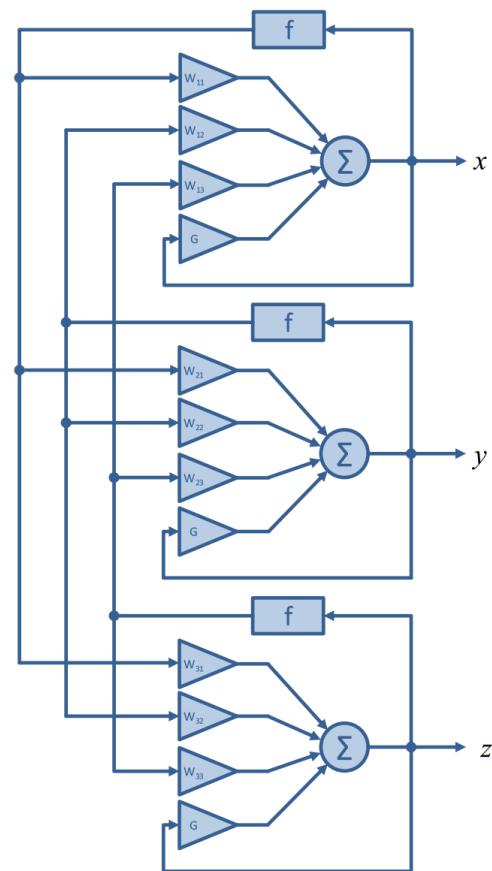


Figura 4.3: Red utilizada. Se trata de una red de Hopfield tridimensional de memoria diferencial, el diseño está orientado a una posterior implementación en FPGA.

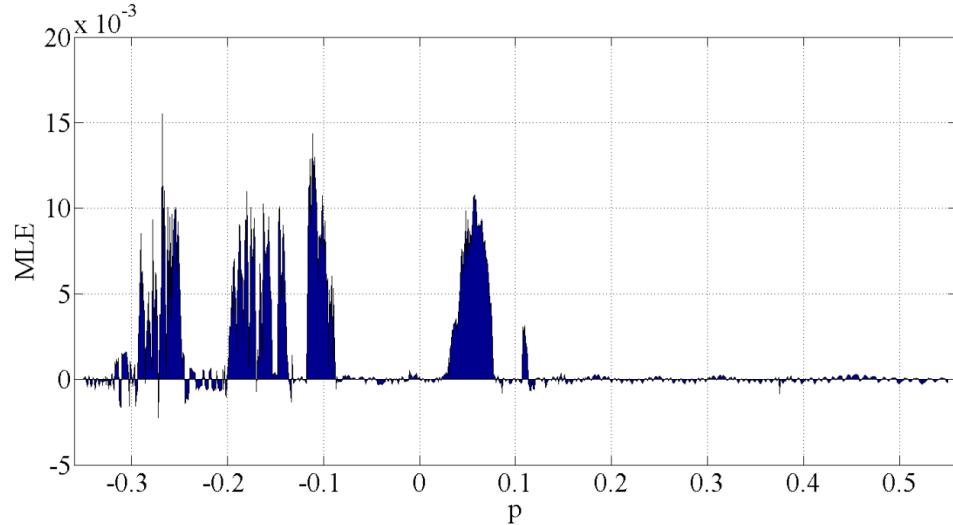


Figura 4.4: MLE en función del parámetro p . Existe caos en toda la zona en la que es positiva.

(0,0223; 0,0791). Esto significa que el caos persiste con una variación no infinitesimal del parámetro, esta zona es muy útil para implementaciones prácticas.

Para mostrar la transición al caos y la relación entre el MLE y el espacio de fases, se eligieron dos parámetros $p_1 = -0,2725$ y $p_2 = 0,268$, para p_1 el $MLE = -2,2 \times 10^{-3}$, para p_2 el $MLE = 1,55 \times 10^{-2}$. Se muestra la trayectoria resultante para cada uno en la Figura 4.5.

Para el atractor caótico, dos trayectorias generadas a partir de condiciones iniciales muy cercanas deben, al cabo de un tiempo, separarse y oscilar en trayectorias distintas. En la Figura 4.6 puede verse este efecto.

4.2. Cripto-Codificación Caótica Variante en el Tiempo

En esta Sección se presenta una nueva técnica para la criptocodificación de datos mediante una familia de mapas caóticos. El diseño se basa en los mapas cuadráticos bidimensionales presentados en la Sección 2.3.3, aprovechando su característica de modificar su atractor según los valores que tomen sus 12 coeficientes reales. Para la implementación se utilizó aritmética de punto fijo con 19 bits de parte fraccionaria. Se realizaron simulaciones y el diseño en VHDL mediante el programa Quartus II v8.0 de ALTERA, para su posterior implementación en FPGA.

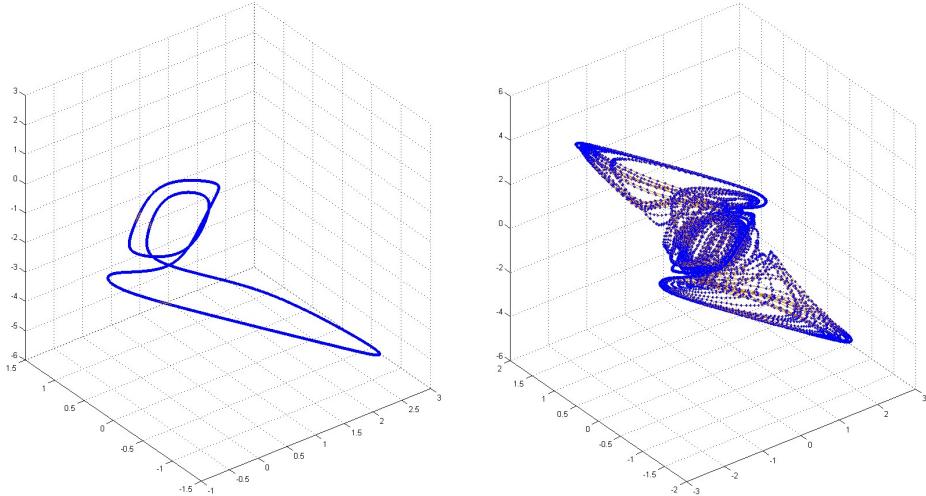


Figura 4.5: Dos trayectorias características del sistema en el espacio de fases. La trayectoria de la izquierda se corresponde con un $MLE < 0$ y la positiva con un $MLE > 0$.

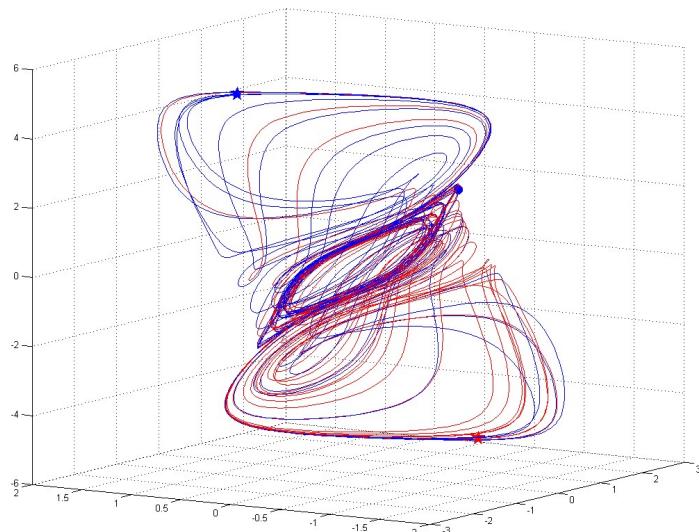


Figura 4.6: Dos trayectorias de la red de Hopfield para condiciones iniciales próximas. Las condiciones iniciales están marcadas con dos puntos grandes cerca del centro del atractor y los valores después de $\Delta t = 3s$ con estrellas en ambos extremos de la figura.

En los sistemas de comunicaciones y particularmente en los dedicados a la codificación para el control de error y encriptamiento de datos se usan técnicas derivadas de la teoría de señales. Estas técnicas se aplican típicamente en la forma lineal debido a la simplicidad que esto trae aparejado. Además, cada una se implementa algorítmica o físicamente como una entidad independiente. Para cada sistema en particular se las elige con criterios de conveniencia práctica, y se las aplica en forma consecutiva o encadenada. La teoría de los sistemas no lineales [5, 27] aparece como un marco de trabajo ideal para ser utilizado en el contexto anteriormente mencionado. La existencia de los sistemas caóticos, y la relación de estos con la aleatoriedad, o pseudo aleatoriedad, otorga una plataforma de diseño que hasta hoy se encuentra poco explotada.

En los últimos veinte años se han presentado diversos trabajos que emplean caos en los sistemas de comunicaciones, como por ejemplo el empleo de portadoras caóticas sincronizadas en las transmisiones analógicas [83, 84]. Si nos centramos en la representación discreta un referente muy importante es el excelente trabajo de Kozic *et al.* [90, 91] en el que se presenta una técnica de modulación empleando mapas caóticos unidimensionales lineales por tramos, la técnica consiste en la introducción del mensaje a codificar en el bit menos significativo de la secuencia generada. Así, se obtiene una secuencia levemente alterada lo que impide que el sistema entre en ciclos periódicos.

En este trabajo se propone un grupo de atractores como generadores de señales pseudoaleatorias para realizar el proceso de codificación y encriptamiento. El esquema de codificación se basa en la familia de mapas cuadráticos bidimensionales, cuyas salidas presentan comportamiento caótico, con distintos atractores conforme a los coeficientes que se empleen. La idea es que cada palabra a codificar sea unívoca con un juego de coeficientes que serán parámetros de un mapa cuadrático bidimensional. Como resultado de este procedimiento, la señal de salida son puntos pertenecientes a distintos atractores elegidos por la información a transmitir.

La ventaja de este método reside en que la estructura de toda la familia de mapas es única y común. Modificándose solamente los coeficientes se consiguen atractores distintos. Esta propiedad reduce y facilita la implementación en hardware. Resultados preliminares obtenidos mediante simulaciones muestran que el sistema presenta una performance comparable a la obtenida en sistemas clásicos de encriptamiento, en cuanto a probabilidad de error y distancia mínima.

4.2.1. Implementación

Desde el punto de vista del esquema de codificación propuesto, estos mapas son muy atractivos por el hecho de contar con 12 coeficientes para generar cada atractor. Por lo tanto, las combinaciones posibles serán N^{12} , en donde N es la cantidad de símbolos posibles según la aritmética utilizada. En nuestro caso empleamos una aritmética de 19 bits expresados en complemento a 2 con aritmética de punto fijo, con 1 bit de signo, 3 bits de parte entera y 15 bits de parte decimal. Esta aritmética limita y discretiza el plano xy que queda delimitado por $\Delta x = 4$, $-\Delta x = -4$, $\Delta y = 4$, $-\Delta y = -4$. Estas limitaciones al plano de atracción tienen como consecuencia dos cuestiones a tener en cuenta:

- Debido a que los coeficientes se generan con la misma aritmética que las variables, nos encontramos con $N = 2^{19}$ valores posibles para cada coeficiente, lo que arroja $(2^{19})^{12} \cong 4,3^{68}$ combinaciones posibles de coeficientes para generar distintos atractores.
- En cuanto a las trayectorias de los atractores sobre el plano discretizado, éstas se tornan periódicas debido a la discretización.

No todos los juegos de coeficientes generan atractores caóticos contenidos en el plano dado por la aritmética utilizada. Aunque esto no sería problema para la codificación/decodificación, se eligieron los coeficientes de modo que se generen atractores contenidos en el plano a modo de validación visual.

Dada la naturaleza de los mapas caóticos, un punto muy lejano a la zona de atracción puede hacer que el punto calculado para la próxima iteración diverja, por lo tanto, las condiciones iniciales deben ser normalizadas antes de cambiar al siguiente mapa. Para solucionar este problema se utiliza la siguiente estrategia:

- Primero se define el plano mínimo que contiene al atractor. Para identificarlo se simularon los mapas mediante Quartus generando secuencias de salida lo suficientemente largas como para verificar la periodicidad. Luego se analizó este vector de datos con Matlab buscando los valores extremos en cada una de las variables: $X_{1\max}$, $X_{1\min}$, $Y_{1\max}$, $Y_{1\min}$. Estos límites delimitan al plano mínimo que contiene al atractor. La normalización dada por la Ecuación 4.5 se aplica a la salida (x,y) para mapear este plano mínimo a todo el plano delimitado por la aritmética utilizada de dimensiones Δx , $-\Delta x$, Δy , $-\Delta y$.

- Segundo, se halla el plano máximo que contiene las condiciones iniciales que hacen que no diverja la solución sino que genere el atractor. Para esto se realizó un programa en Matlab que genera los atractores desde todas las condiciones iniciales del plano delimitado y discretizado por la aritmética utilizada, a continuación se marcan todos los puntos que generan trayectorias divergentes o bien convergentes a un punto fijo. Este proceso genera la zona de condiciones iniciales factible para generar atractores, nuevamente se identificaron los valores máximos y mínimos del área rectangular máxima que contenga todos sus puntos como condiciones iniciales factibles $X2_{max}$, $X2_{min}$, $Y2_{max}$, $Y2_{min}$. La normalización dada por la Ecuación 4.6 se aplica a la entrada de condiciones iniciales (x_{n-1}, y_{n-1}) para mapear todo el plano de dimensiones Δx , $-\Delta x$, Δy y $-\Delta y$ al de condiciones iniciales factibles.

$$\begin{aligned}
 x_{1norm} &= a_{1x}x + b_{1x} \\
 y_{1norm} &= a_{1y}y + b_{1x} \\
 a_{1x} &= \frac{2\Delta x}{x_{1max} - x_{1min}} \\
 a_{1y} &= \frac{2\Delta y}{y_{1max} - y_{1min}} \\
 b_{1x} &= -\frac{x_{1max} - x_{1min}}{2} \\
 b_{1x} &= -\frac{y_{1max} - y_{1min}}{2}
 \end{aligned} \tag{4.5}$$

$$\begin{aligned}
 x_{1norm} &= a_{2x}x + b_{2x} \\
 y_{1norm} &= a_{2y}y + b_{2x} \\
 a_{2x} &= \frac{x_{2max} - x_{2min}}{2\Delta x} \\
 a_{2y} &= \frac{y_{2max} - y_{2min}}{2\Delta y} \\
 b_{2x} &= \frac{x_{2max} - x_{2min}}{2} \\
 b_{2x} &= \frac{y_{2max} - y_{2min}}{2}
 \end{aligned} \tag{4.6}$$

El problema de la existencia de puntos fijos para cierto conjunto de coeficientes y condiciones iniciales queda salvado al perturbar continuamente al atractor actual con valores afectados por la información.

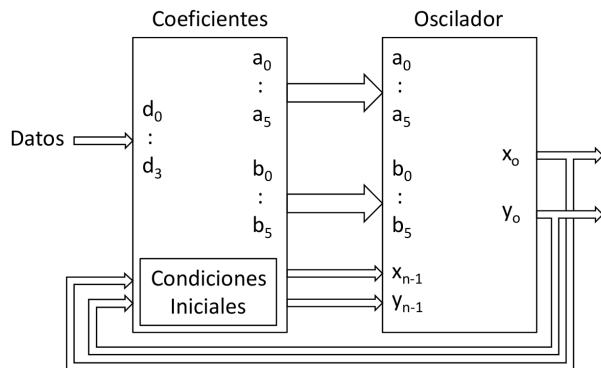


Figura 4.7: Generador de atractores.

Se generó un circuito en VHDL con un total de 16 juegos de parámetros seleccionables con la palabra de entrada de 4 bits que se desea encriptar. Esta palabra multiplexa estos coeficientes y alimenta un oscilador que calcula la próxima iteración de datos, además, este circuito almacena la salida del oscilador y la realimenta como “condición inicial” para calcular la iteración siguiente (Figura 4.7). Como resultado de este proceso, la salida encriptada resulta ser el oscilador actual seleccionado por la palabra de entrada perturbado por la historia de los mapas seleccionados por las entradas anteriores. Este circuito de dos bloques se encarga de generar los atractores, por lo que se lo llama “generador de atractores”.

Para la primer iteración, las condiciones iniciales son $(x; y) = (0,1; 0,1)$ para cualquiera de los atractores.

Codificador

El bloque del Codificador consiste en circuito generador y acondicionamiento de la salida. Para codificar una palabra de cuatro bits de entrada se generan los valores de x e y con el circuito generador correspondiente a esta palabra y se los concatena en un circuito posterior formando un vector $[x : y]$ (Figura 4.8). De esta forma cada palabra de información a ser enviada será representada por la salida $\{xy\}$ del oscilador del atractor correspondiente. Por lo tanto, una palabra a codificar no se corresponderá con una palabra codificada, dos palabras iguales generarán dos salidas distintas.

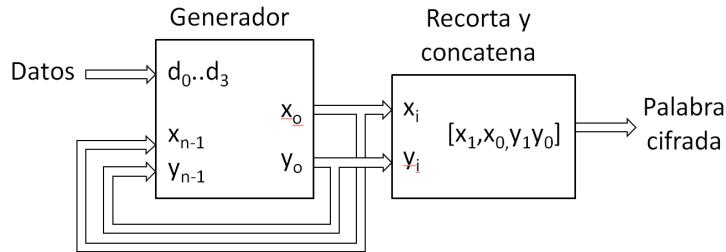


Figura 4.8: Codificador.

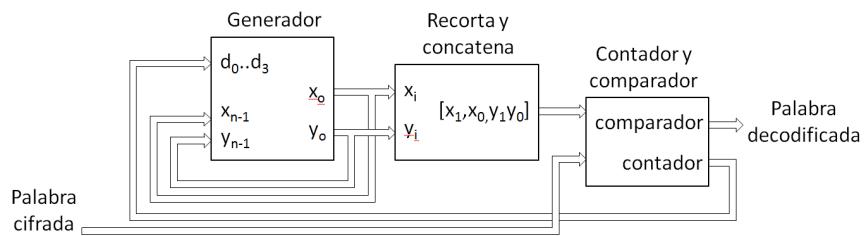


Figura 4.9: Decodificador.

Decodificador

Un segundo circuito generador de atractores funciona en el decodificador generando las 16 palabras posibles para la próxima iteración. Luego, se ingresan todas estas posibles palabras cifradas junto con la que se desea decodificar a un comparador que aplica una XOR a la palabra ingresada contra todas las palabras posibles generadas localmente para decodificarla. La salida de este circuito será la palabra decodificada (Figura 4.9).

4.2.2. Resultados

Se realizó un primer esquema del diseño mediante la herramienta Quartus II v8.0 de ALTERA, para implementar el sistema en una FPGA *Altera Cyclone III EP3C120*.

Se obtuvieron resultados preliminares de simulaciones realizadas mediante el programa Matlab y mediante simulaciones con el programa Quartus de Altera, estas últimas tienen en cuenta el empleo de la precisión finita elegida para representar los valores.

En la Figura 4.10 se pueden ver las salidas del bloque generador para una transmisión de los datos [1,2,3,2,3,3,1,3,1,3,1]. En este caso se mantiene el dato a enviar durante 100 ciclos con el objetivo de que sea visible en la Figura, en el sistema real cada oscilador codifica

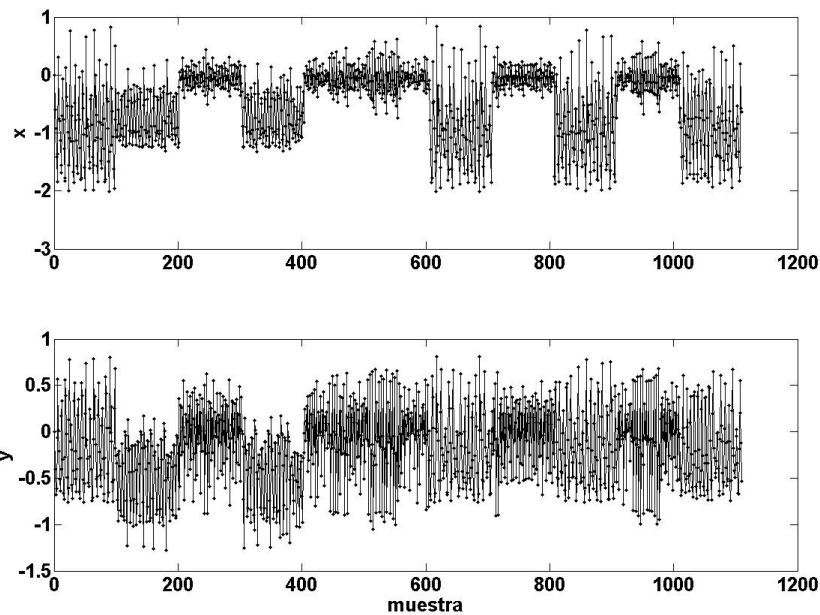


Figura 4.10: Señales a transmitir.

una palabra de información en cada iteración. Aquí puede observarse que el sistema cambia el atractor generado según los coeficientes que dependen de la entrada de información a transmitir.

4.3. Implementación de Atractor Determinístico - Estocástico

En aplicaciones digitales, el tiempo y la variable de estado tienen valores discretos. La discretización de tiempo impone el uso de un algoritmo para aproximar las ecuaciones diferenciales de tiempo continuo que modelan el sistema. El algoritmo más simple es el método de Euler de primer orden en el que los diferenciales se reemplazan directamente por incrementos finitos. Los algoritmos más elaborados, como los algoritmos de paso variable Runge-Kutta de cuarto orden (RK4) (o superior), hacen que el sistema discreto evolucione más cerca del sistema continuo, pero con mayores requisitos de recursos de hardware y tiempos de cálculo. En consecuencia, deben usarse solo si la exactitud es un requisito de la aplicación específica. Este no es el caso en PRNG, donde la aleatoriedad es la característica

principal que debe garantizarse.

Como se dijo anteriormente la cantidad de bits a emplearse es un tema crítico en la implementación de sistemas caóticos. Se han propuesto varias estrategias para una selección correcta del número óptimo de bits en las implementaciones en hardware. Sin embargo, la mayoría de estos procedimientos están limitados a sistemas lineales [92, 93]. En los sistemas digitales caóticos, se puede obtener un comportamiento completamente diferente al variar la precisión. Este problema ha ganado interés recientemente, y se han propuesto varios esquemas nuevos [94, 95, 96].

En resumen, a pesar de la aritmética utilizada, ya sea de punto fijo o punto flotante, el conjunto de números que se pueden representar es limitado. Incluso utilizando una precisión extremadamente alta como lo hacen Liao y Wang [10], las secuencias generadas por un sistema caótico que usa hardware digital siempre serán periódicas.

En esta Sección, se implementó un oscilador discreto de Lorenz obtenido mediante el uso del algoritmo de primer orden de Euler y tres estándares de representación diferentes. Además, se aplicaron técnicas de aleatorización a las variables de estado para obtener un PRNG en tiempo real. El objetivo de este trabajo es estudiar la influencia del procedimiento de discretización en la dinámica del sistema. En [97] se estudió el grado de estocasticidad de un sistema caótico determinista mediante su implementación en coma flotante de precisión simple (32 bits estándar IEEE 754).

Se eligió el sistema caótico de Lorenz porque es un sistema ampliamente estudiado y también ha sido implementado por otros autores con diferentes metodologías [95, 96, 98]. Por ejemplo, en [95], se implementó un oscilador caótico de Lorenz mediante una toolbox del Generador de Sistemas Xilinx que funciona bajo MATLAB-Simulink. Esta toolbox convierte el modelo MATLAB-Simulink en el modelo de Xilinx System Generator y luego se obtiene el código VHDL. Esta herramienta, si bien facilita la implementación, al realizar una operación automática no permite ciertos cambios específicos y presenta algunas limitaciones. La operación de integración se aproximó con el algoritmo de Euler, utilizando bloques de suma y retardo. La implementación propuesta en [96] y [98] usa RK4 en una arquitectura de punto fijo de 32 bits.

Se utilizó el software QuartusII 7.2 para generar el lenguaje de descripción de hardware VHDL y la implementación física se realizó en la placa de desarrollo Altera Cyclone III EP3C12. Se estudiaron tres representaciones numéricas: 1) punto flotante estándar IEEE 754, 2) punto fijo decimal, y 3) aritmética entera [99]. Cada representación implica una

cantidad diferente de bits. Consideramos dos representaciones de coma flotante para los estándares simple o doble, con 32 y 64 bits respectivamente para representar el signo, el exponente y la mantisa. La aritmética decimal de punto fijo usa p bits para representar la parte entera y m bits para representar la parte fraccional. Se consideraron representaciones de punto fijo de 32 y 64 bits con 9 bits para la parte entera y 1 bit para el signo y los 22 o 54 bits restantes para la parte fraccionaria. En aritmética de enteros k bits el alfabeto tiene 2^k símbolos. Consideramos $k = 54$.

Para cuantificar la aleatoriedad del sistema, se utilizó el conjunto de pruebas DIEHARD de Marsaglia [6]. Estas pruebas han sido ampliamente utilizadas en la literatura abierta y son muy efectivas para clasificar sistemas determinísticos y estocásticos.

4.3.1. Discretización Temporal del Oscilador de Lorenz

El sistema de Lorenz se define mediante el siguiente conjunto de ecuaciones diferenciales ordinarias acopladas:

$$\begin{aligned}\frac{dx}{dt} &= -\delta(x - y), \\ \frac{dy}{dt} &= \Gamma x - y - xz, \\ \frac{dz}{dt} &= -bz + xy,\end{aligned}\tag{4.7}$$

donde δ , Γ y b son parámetros constructivos del sistema. Para ciertos valores de estos parámetros, el sistema tiene un comportamiento caótico. Para convertir un sistema dinámico continuo en un sistema dinámico de tiempo discreto se requiere emplear algún algoritmo. El algoritmo más simple fue propuesto por Euler y, para las Ecuaciones 4.7 surge el siguiente modelo discreto:

$$\begin{aligned}\tilde{X}_{t+\Delta t} &= \tilde{X}_t + \Delta t \left[-\delta (\tilde{X}_t - \tilde{Y}_t) \right], \\ \tilde{Y}_{t+\Delta t} &= \tilde{Y}_t + \Delta t \left[-\tilde{X}_t \tilde{Z}_t + \Gamma \tilde{X}_t - \tilde{Y}_t \right], \\ \tilde{Z}_{t+\Delta t} &= \tilde{Z}_t + \Delta t \left[\tilde{X}_t \tilde{Y}_t - b \tilde{Z}_t \right],\end{aligned}\tag{4.8}$$

donde Δt es el tamaño de paso de tiempo y \tilde{X} , \tilde{Y} y \tilde{Z} son variables de estado de tiempo discreto que toman valores reales.

El algoritmo de Euler es un algoritmo de un sólo paso porque para calcular las variables

en el momento $t + \Delta t$, sólo es necesario conocer los valores en el instante anterior. Calculando iterativamente con un paso Δt apropiado, es posible obtener la evolución del sistema discreto. Es razonable esperar que cuanto menor sea el valor de Δt , más exactos serán los valores obtenidos. Sin embargo, se debe tener en cuenta que al reducir el valor de Δt se incrementa la cantidad de cálculos y esto genera más errores de redondeo.

En aplicaciones que requieren una reproducción exacta de la dinámica del sistema continuo, los algoritmos más exactos son obligatorios, pero en el caso de los PRNG, solo las propiedades estadísticas y la aleatoriedad de las series temporales son importantes y, por consiguiente, en este caso el algoritmo de Euler es lo suficientemente bueno.

4.3.2. Discretización de las Variables de Estado

Como se señaló anteriormente, se utilizan tres representaciones numéricas diferentes. Cada una se describe en las siguientes Subsecciones.

Estándar IEEE 754

La representación en punto flotante es uno de los métodos para representar números reales con precisión finita. La ventaja de la representación de punto flotante sobre las representaciones de punto fijo y entero es que puede admitir un rango de valores mucho más amplio porque escala automáticamente cada número para usar la longitud de palabra completa para la mantisa; esto se hace moviendo el punto decimal (este procedimiento implica un cambio en el valor del exponente) hacia la posición del bit más significativo. En consecuencia, la precisión total se conserva incluso para números pequeños. La aritmética de punto flotante binaria es más adecuada para trabajar con cantidades del mundo real en una amplia gama de escalas.

El estándar de precisión simple IEEE 754 asigna 23 bits a la mantisa (bit 0 a 22), el exponente ocupa los siguientes 8 bits (23 a 30) y el bit 31 está asignado al signo. El estándar de precisión doble IEEE 754 asigna 52 bits a la mantisa (bit 0 a 51), el exponente ocupa los siguientes 11 bits (52 a 62) y el bit 63 está asignado al signo.

Las operaciones aritméticas de punto flotante son más complicadas que las de punto fijo. Su ejecución requiere más ciclos de reloj y hardware complejo. Sin embargo, gracias al avance tecnológico y al desarrollo de nuevos materiales, la cantidad de recursos, memoria y frecuencia de operación de los dispositivos digitales se incrementa constantemente. Hoy en

día existen FPGAs con más memoria y recursos, capaces de trabajar a altas frecuencias en estos estándares.

Implementación en Punto Fijo

Cuando todos los valores a utilizar se encuentran dentro de un rango conocido, es posible lograr una mayor precisión utilizando la denominada representación de punto fijo en lugar de la representación de punto flotante. El hardware requerido para manipular estas representaciones es el mismo comúnmente utilizado para realizar operaciones enteras y es menos costoso que el requerido para el caso de punto flotante.

Para evitar el desbordamiento, inicialmente es necesario realizar un análisis para determinar los valores extremos involucrados en el cálculo, incluidas las operaciones intermedias. Con esta información, se determina el número mínimo de bits que se emplearán. Una vez establecido el número de bits necesarios para representar la parte entera, se usa un bit adicional para representar números negativos basados en complemento a 2 (CA2). Los bits restantes se utilizan para mejorar la precisión ya que representan la parte fraccionaria.

Las operaciones de suma, resta y multiplicación se implementan de la misma manera que en la aritmética de enteros. Solo es necesario cuidar la posición del punto que separa la parte entera de la fraccionaria.

Aquí consideramos dos casos, 32 y 64 bits por cada número entero. En ambos casos se usaron 9 bits para la parte entera, más 1 bit para el signo, dejando los bits restantes, 22 o 54 respectivamente, para la parte decimal.

Implementación en Aritmética Entera

En aritmética de enteros, los circuitos se pueden reducir significativamente si se adoptan divisores con una potencia de 2. Para obtener la versión entera para el sistema Lorenz, se realizaron las siguientes transformaciones de polarización y escalado. La polarización consiste en un corrimiento respecto de los ejes coordenados y el escalado en una contracción o expansión de estos ejes [99]:

$$\begin{aligned} X_t &= (\tilde{X}_t + B) S, \\ Y_t &= (\tilde{Y}_t + B) S, \\ Z_t &= (\tilde{Z}_t + B) S, \end{aligned} \tag{4.9}$$

donde B y S son los parámetros de polarización y escala, respectivamente. Reemplazando la Ecuación 4.9 en 4.8 se obtiene:

$$\begin{aligned} X_{t+\Delta t} &= X_t + \Delta t \delta (Y_t - X_t) , \\ Y_{t+\Delta t} &= (1 - \Delta t) Y_t + \Delta t (B + \Gamma) X_t + \Delta t B Z_t \\ &\quad - \frac{\Delta t}{S} X_t Z_t + \Delta t B S (1 - \Gamma - B) , \\ Z_{t+\Delta t} &= (1 - \Delta t b) Z_t - \Delta t B (X_t + Y_t) \\ &\quad + \frac{\Delta t}{S} X_t Y_t + \Delta t B S (B - b) . \end{aligned} \quad (4.10)$$

En este caso, fue adoptado: $\delta = 8$, $\Gamma = 24$, $b = 2$, $\Delta t = 2^{-n}$, $B = 40$, $S = 512$. La variable n es un número entero que dejamos libre para poder explorar el comportamiento del sistema con distintos pasos de tiempo.

Debe tenerse cuidado cuando se eligen los parámetros del sistema, en este caso se seleccionaron coeficientes enteros y se realizó un análisis de estabilidad para garantizar que el sistema no converja a un punto fijo o a una órbita de período bajo.

El sistema final es:

$$\begin{aligned} X_{t+\Delta t} &= X_t + \text{floor} \left[\frac{Y_t}{2^{n-3}} \right] - \text{floor} \left[\frac{X_t}{2^{n-3}} \right] , \\ Y_{t+\Delta t} &= Y_t - \text{floor} \left[\frac{Y_t}{2^n} \right] + \text{floor} \left[\frac{X_t}{2^{n-6}} \right] \\ &\quad + \text{floor} \left[\frac{Z_t}{2^{n-3}} \right] + \text{floor} \left[\frac{Z_t}{2^{n-5}} \right] \\ &\quad - \text{floor} \left[\frac{X_t}{2^{(22+\text{floor}[\frac{n}{2}+1])}} \right] \\ &\quad \text{floor} \left[\frac{Z_t}{2^{(22+\text{floor}[\frac{n}{2}])}} \right] - 2^{(44-n)} 2520 , \\ Z_{t+\Delta t} &= Z_t - \text{floor} \left[\frac{Z_t}{2^{n-1}} \right] - \text{floor} \left[\frac{(X_t + Y_t)}{2^{n-3}} \right] \\ &\quad - \text{floor} \left[\frac{(X_t + Y_t)}{2^{n-5}} \right] + \text{floor} \left[\frac{X_t}{2^{(22+\text{floor}[\frac{n}{2}+1])}} \right] \\ &\quad \text{floor} \left[\frac{Y_t}{2^{(22+\text{floor}[\frac{n}{2}])}} \right] + 2^{44-n} 1680 . \end{aligned} \quad (4.11)$$

Este sistema discreto tiene un comportamiento caótico (de hecho, pseudocaótico) y todos los divisores están en potencia de 2. Todo el procedimiento de preprocesamiento de las ecuaciones minimiza los recursos de hardware necesarios (como se mostrará más adelante).

4.3.3. Post-procesamiento de Aleatorización

Para eliminar o mitigar las estructuras de correlación internas que no son deseables, se analizan dos procedimientos de aleatorización de las secuencias de salida, que no requieren incrementar los recursos de hardware utilizados:

1. *descartar*: se forma una nueva secuencia cuyos elementos son números enteros formados por los 32 bits menos significativos de cada elemento de datos (llamado x_{disc} , y_{disc} y z_{disc} respectivamente);
2. *concatenar*: se forman nuevos enteros de 32 bits al concatenar los bits menos significativos de cada variable (11 bits de z , 10 bits de y y 10 bits de x , se debe tener en cuenta esta es una de las muchas posibilidades), llamado zyx .

Definimos los valores x_{disc} , (y_{disc}, z_{disc}) como la serie temporal x (y, z) después de aplicar la técnica de randomización por *descarte*. La variable xyz es la serie de tiempo obtenida mediante la técnica de aleatorización de *concatenado*. Estos procedimientos se aplicaron a las secuencias de salida generadas por todas las implementaciones descritas en las Secciones anteriores. Además, se varió Δt para encontrar su valor óptimo.

Hay varias propiedades básicas que satisfacer un PRNG para ser considerado bueno: longitud de ciclo larga, aleatoriedad, velocidad, reproducibilidad y portabilidad. Existen test de prueba disponibles para analizar PRNGs [100]. Algunas suites de pruebas de propósito general son DIEHARD de George Marsaglia [6], Crypt-XS de Helen Gustafson de la Universidad Tecnológica de Queensland [101], la suite de pruebas estadísticas del Instituto Nacional de Estándares y Tecnología (NIST) [102], el Test U01 por L'Ecuyer y R. Simard [103] y DIEHARDER [104]. En este documento usamos las 15 pruebas más estrictas de DIEHARD [6] para medir la estocasticidad de cada implementación, pero si la aplicación específica es un PRNG, se recomienda el uso de todas las pruebas mencionadas anteriormente, especialmente NIST 800/22 y Prueba U01.

4.3.4. Resultados

Para cada PRNG se requiere un archivo con más de 80×10^6 bits. Cada ejecución de cada prueba en DIEHARD devuelve un valor p , que debe ser uniforme en $[0, 1)$ si el archivo de entrada contiene bits aleatorios verdaderamente independientes. Esos valores p deben ser $p < 0,025$ ó $p > 0,975$ para considerar que la prueba ha sido aprobada. Cada prueba se

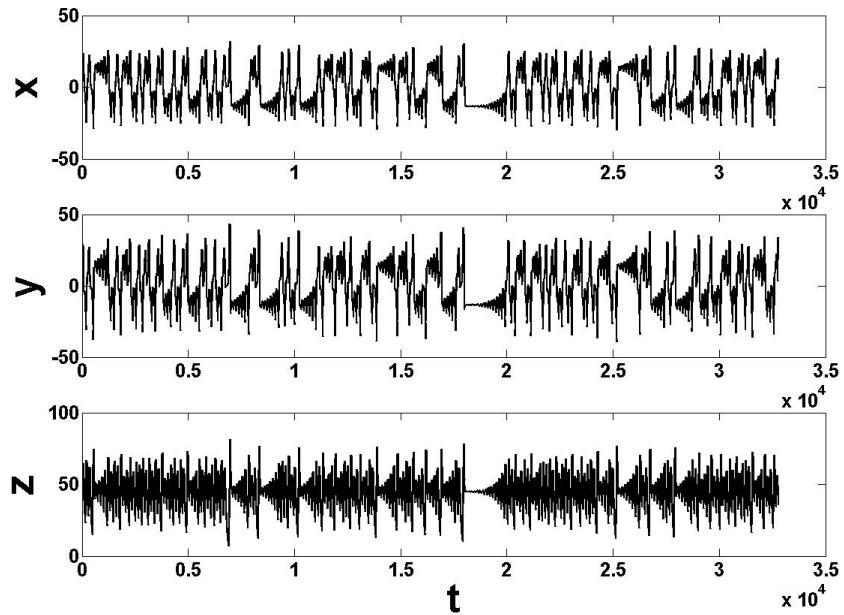


Figura 4.11: Series temporales del atractor de Lorenz.

ejecuta varias veces dando un valor p por cada ejecución. Combinando todos estos p -valores (229 por cada PRNG) se obtuvo un valor general de p por medio de KStest. Sólo si todos los p individuales y también los p globales están en el rango apuntado arriba, se coloca “sí” en el Cuadro 4.2.

Las implementaciones presentadas aquí se desarrollaron completamente con el software Quartus II 7.2. Las implementaciones físicas se realizaron en el kit de desarrollo Altera Cyclone III EP3C120.

Se utilizó SignalTap II Embedded Logic Analyzer para realizar la evaluación de hardware para cada diseño. Esta es una herramienta de depuración a nivel de sistema, proporcionada por *Altera*, que captura y muestra el comportamiento de la señal en tiempo real. Permite observar las interacciones entre el hardware y el software en los diseños del sistema. Después de capturar los datos y guardarlos en un archivo SignalTap II, se pueden analizar y visualizar como una forma de onda [105].

Las Figuras 4.11 y 4.12 muestran respectivamente la serie temporal y el atractor, obtenidos por la implementación del hardware con $\Delta t = 0,0045$ y precisión simple de coma flotante (las cifras con la otra representación numérica son muy similares).

El hardware requerido se muestra en el Cuadro 4.1. Allí se muestra una comparación

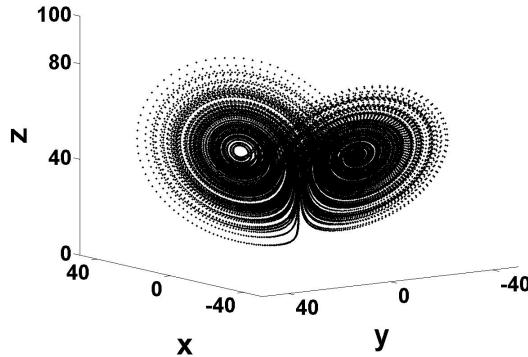


Figura 4.12: Atractor de Lorenz.

Cuadro 4.1: Resultados de compilación en CYCLONE III EP3C120F780C7.

	Punto fijo (32bits)	Punto fijo (64bits)	Flotante (32bits)	Flotante (64bits)	Enteros (54bits)
Elementos lógicos [Total]	2392	6104	8176	17532	1297
Elementos lógicos [%]	2,00	5,12	6,86	14,72	1,08
Total de registros	1658	1754	4753	8532	159
Reloj f_{max} [MHz]	37,82	20,51	7,48	5,42	55,38
Throughput [Mbs]	1210,24	656,32	14,96	173,44	1772,16

entre los resultados de la compilación en las tres representaciones numéricas estudiadas:

- *aritmética de punto flotante*: los dos casos, precisión simple y doble (Flotante(32bits) y Flotante(64bits) respectivamente),
- *aritmética entera* (Enteros(54bits)) y
- *aritmética de punto fijo*: los dos casos, ambos con 9 bits de parte entera más 1 bit de signo más los bits para la parte decimal, 22 bits para punto fijo (32bits) o 54 bits para punto fijo (64bits) respectivamente.

La implementación aritmética de enteros es la que emplea recursos mínimos y admite un f_{max} más alto, la razón de esto es que las ecuaciones implementadas fueron optimizadas previamente para esta aplicación en particular. En el caso de la representación de punto flotante, la optimización se realizó para disminuir el área, pero se pueden aplicar otras técnicas de optimización para mejorar la frecuencia o el consumo de energía [106, 107].

Para utilizar este sistema como PRNG, los datos de salida se procesan con las técnicas *descarte* y *concatenado*. Ambas técnicas mantienen los bits menos significativos porque

Cuadro 4.2: Resultados del test DIEHARD. $\Delta t = 2^{-n}$.

	n	6	7	8	9	10
Flotante (64 bits)	x_{disc}	no	no	si	si	no
	y_{disc}	no	no	si	si	no
	z_{disc}	si	si	si	no	no
	zyx	no	no	no	no	si
Punto fijo (64 bits)	x_{disc}	si	si	no	no	no
	y_{disc}	si	si	no	no	no
	z_{disc}	si	si	no	no	no
	zyx	si	si	si	no	no
Enteros (54 bits)	x_{disc}	si	si	no	no	no
	y_{disc}	si	si	no	no	no
	z_{disc}	si	si	no	no	no
	zyx	si	si	si	si	si

presentan el comportamiento más variable. En el caso de la técnica de *concatenado*, la parte más ruidosa de cada variable de estado se mantiene y se recombinan, y se obtiene una salida de secuencia de 32-bits en cada iteración.

Se generaron archivos de datos de análisis estocástico con 3 000 000 palabras de 32-bits cada uno, para $\Delta t = 2^{-n}$, con $n = 6, 7, 8, 9$ y 10 . Se calcularon las pruebas DIEHARD para todos los archivos generados. En el Cuadro 4.2 se informan algunos de los resultados más relevantes.

Este Cuadro muestra que en los casos de implementaciones de enteros y punto fijo la técnica de aleatorización de *descarte* funciona mejor a medida que Δt aumenta porque, para elementos Δt menores, las series temporales son más correlacionadas y esta técnica de aleatorización no los mezcla lo suficiente. Para utilizar valores de Δt más bajos, más bits deben descartarse para obtener buenos PRNG. Por otro lado, la técnica de aleatorización de *concatenación* funciona bien independientemente del valor de Δt (dentro del rango analizado).

4.4. Mapas Cuadráticos 2-D Implementados en Punto Fijo

En esta Sección se emuló el comportamiento de una implementación en hardware digital, como FPGA, dispositivo lógico programable complejo (CLPD) o circuito integrado de aplicaciones específicas (ASIC), para replicar exactamente el funcionamiento del dispositivo.

Nuestro interés es medir cómo los dominios de atracción se degradan en función del número de bits n empleado, y así encontrar el valor umbral n_{min} para el cual se conservan ciertas propiedades estadísticas requeridas.

El trabajo de Grebogi [108] mostró que la longitud promedio de las órbitas periódicas T de un sistema dinámico implementado en una computadora, escala en función de la precisión de la computadora ξ y la dimensión de correlación D del atractor caótico, como $T \sim \xi^{-D/2}$. En este caso el objetivo es investigar las características de cada precisión para que el diseñador tenga una visión general completa de las opciones que se utilizarán en su implementación. De esta manera, se podrán establecer las propiedades a rescindir según los requerimientos y los recursos disponibles al momento del diseño.

Además de analizar los cambios en las duraciones de los períodos, las propiedades estadísticas de las secuencias serán diferentes de las del sistema real, por lo que también deberían analizarse. En [109] se desarrolló un excelente trabajo sobre las consecuencias que tiene la precisión finita sobre la periodicidad de un PRNG basado en el mapa Logístico. Allí, se determinaron el número, retardo y período de las órbitas del mapa Logístico con diversos grados de precisión, sin embargo, este trabajo carece de un análisis estadístico. La investigación realizada en este Capítulo complementa su trabajo al agregar cuantificadores estadísticos. Además aquí se estudia la implementación en arquitectura de punto fijo (en cambio en [109] se utilizó punto flotante) ya que es la arquitectura que menos recursos emplea en las implementaciones en hardware. Como consecuencia, el consumo de energía también se ve disminuido.

Entre muchos sistemas caóticos disponibles en la literatura, se utilizó la familia de mapas cuadráticos bidimensionales descriptos en la Sección 2.3.3. Solo los resultados para la representación analítica de los mapas en [110] han sido publicados en la literatura abierta.

El objetivo en esta Sección es ampliar el análisis a la versión digital para posibilitar la implementación del hardware en aritmética de punto fijo, para lo cual es imprescindible conocer la duración del período y el grado de aleatoriedad de las secuencias generadas por el sistema discretizado. Se desarrolló un análisis detallado de la *degradación* del sistema caótico multiatractor a medida que varía la precisión empleada en una implementación de punto fijo. Por *degradación* se entiende: (a) la aparición de puntos fijos estables y órbitas periódicas estables con períodos cortos, dentro de un dominio de atracción de coma flotante sin órbitas estables; (b) el mismo atractor se vuelve periódico y sus características estadísticas cambian, lo que hace que el sistema sea más determinista.

Las principales contribuciones de esta Sección son:

- el análisis de los dominios de atracción de los atractores caóticos para un conjunto dado de parámetros a medida que aumenta el número de bits; en términos de la duración del período y la aparición de puntos fijos estables y órbitas periódicas con períodos cortos;
- la determinación del consecuente umbral para el ancho del bus, para hacer que las propiedades estadísticas de la implementación digital sean cercanas a las de la implementación de coma flotante;
- se asignan dos PDFs distintas para evaluar la estocasticidad de la serie temporal para diferentes anchos de bus. Cada PDF P se mide por la correspondiente entropía de Shannon normalizada $H(P)$. Estas entropías tienen cambios abruptos en anchos de bus específicos. Las duraciones de los períodos y el *MLE* también se evalúan y los resultados se comparan con las $H(P)$.

4.4.1. Resultados

Un código ANSI C que simula un sistema no lineal iterando (el mapa cuadrático) en un dispositivo electrónico digital fue desarrollado con el fin de generar secuencias que luego fueron analizadas.

Este código itera el mapa cuadrático-2D 10^5 veces, en este caso los coeficientes a_0 a a_{11} tienen los valores: $\{a_i\} = \{-1,0, 0,9, 0,4, -0,2, -0,6, -0,5, 0,4, 0,7, 0,3, -0,5, 0,7, -0,8\}$. El sistema fue diseñado para trabajar en arquitectura fraccionaria de punto fijo con n bits, donde $n = n_i + n_f$, en representación de complemento a 2 (Ca_2). En este caso, se empleó $n_i = 4$ bits para representar la parte entera, y el código varía automáticamente la cantidad de bits que representan la parte fraccionaria del número, n_f , para analizar cómo reacciona el sistema cuando cambia la precisión. El código se ejecuta a partir de todas las CIs dentro del intervalo $[-2, 2]$ en pasos determinados por n_f , por lo tanto, la grilla tendrá un paso de:

$$step_grid = \frac{1}{n_f, 2^{n_f}}. \quad (4.12)$$

En cada caso se determinó si los sistemas evolucionan a un punto fijo, divergen o van hacia un ciclo periódico, también se generaron secuencias para esa misma CI usando diferentes n_f bits de precisión. Estos datos fueron luego evaluados utilizando los cuantificadores

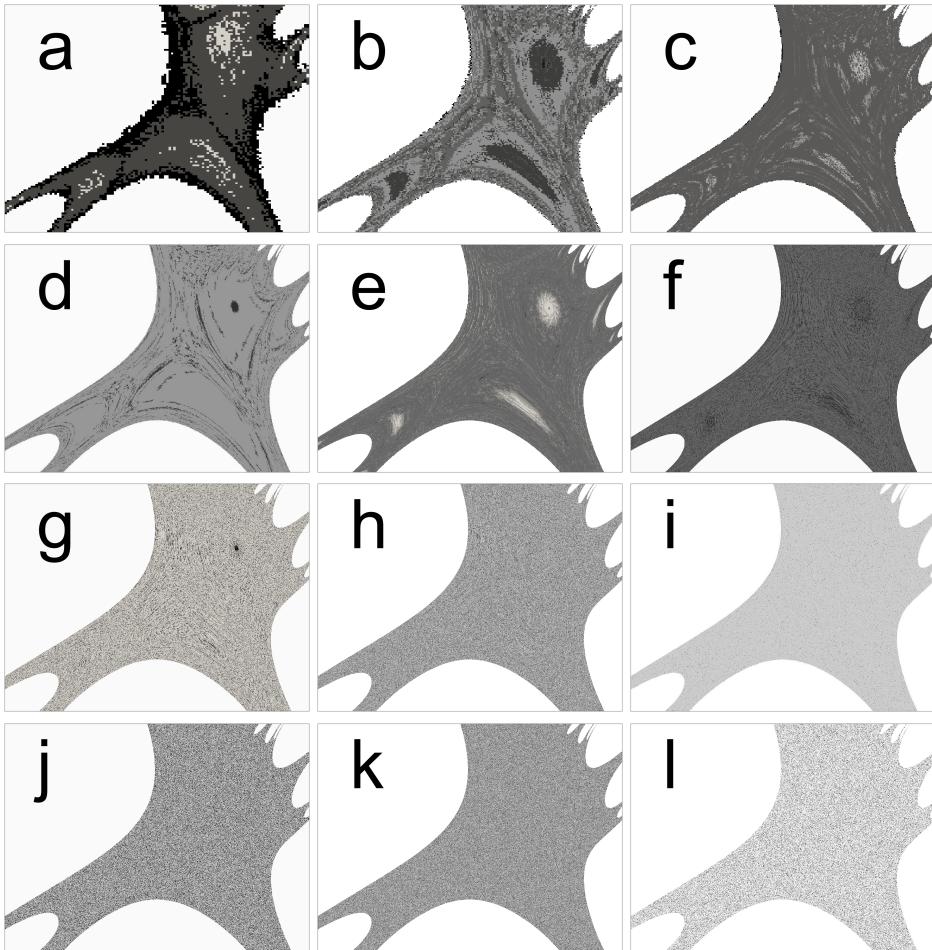


Figura 4.13: Áreas coexistentes en el dominio de atracción para: (a) $n_f = 5$, (b) $n_f = 6$, (c) $n_f = 7$, (d) $n_f = 8$, (e) $n_f = 9$, (f) $n_f = 10$, (g) $n_f = 11$, (h) $n_f = 12$, (i) $n_f = 13$, (j) $n_f = 14$, (k) $n_f = 17$, (l) $n_f = 18$.

de aleatoriedad introducidos en la Sección 3.

La Figura 4.13 muestra los dominios de atracción obtenidos para $n_i = 4$ y diferentes valores de n_f . Los ejes de abscisas y ordenadas corresponden a valores iniciales de x e y respectivamente. Cada punto representa una CI y el color está asociado a su estado final, mientras más oscuro es el tono de gris, más corto es el ciclo al que converge, los puntos fijos están en negro y los puntos divergentes en blanco. Entonces, se pueden ver los diferentes dominios de atracción (incluyendo los atractores) que coexisten en el sistema para cada precisión.

Con el fin de poder distinguir las diferentes áreas coexistentes, se ha utilizado una amplia

gama de tonos grises en cada Figura. Se debe tener en cuenta que cada figura tiene su propio rango de grises, esto significa que, por ejemplo, un área casi blanca cuando $n_f = 5$ (Figura 4.13.a) corresponde a un período de 6, mientras que un área más oscura en una Figura con mayor n_f puede corresponder a un período superior a mil (Figura 4.13.e). Estas cifras permiten reflejar los complejos dominios de atracción que aparecen al digitalizar.

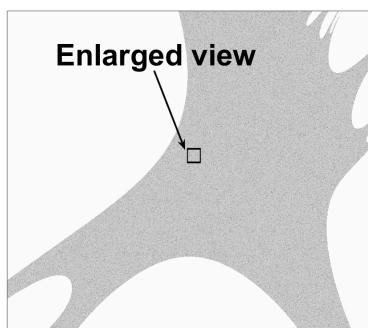
Se puede ver en la Figura 4.13 cuánto menor es el valor de n_f , mayor es el área de ICs que tiende a diverger y/o converger a puntos fijos. A medida que n_f aumenta, el área de puntos divergentes y fijos disminuye. Estas Figuras junto con el Cuadro 4.3 permiten interpretar fácilmente el comportamiento del sistema. En el Cuadro 4.3 las longitudes de las secuencias que aparecen en el dominio de atracción para cada n_f se ordenan por el número de circuitos integrados menos numerosos que convergen en ese ciclo. Además, entre paréntesis se presenta la tasa de ocurrencia. De hecho, las Figuras con valores más bajos de n_f presentan superficies irregulares o rugosas, señalando que los ciclos de diferentes longitudes coexisten. Por ejemplo, para $n_f = 5$ hay una prevalencia de ciclos de períodos cortos. En ese caso, existen solo dos ciclos límite, la zona gris más clara corresponde al dominio de atracción de los ciclos límite de longitud seis, que es el ciclo menos numeroso, de acuerdo con el Cuadro 4.3, y la zona más oscura corresponde al dominio de atracción de longitud de ciclo dos.

Aunque para $n_f \geq 13$ (Figuras 4.13.i a 4.13.l), el dominio de atracción parece ser suave y uniforme, todavía hay ciclos con diferentes períodos que coexisten en el atractor por $n_f = 14, 17$ y 18 . Esto se puede ver si ampliamos una Sección de las figuras (Figura 4.14).

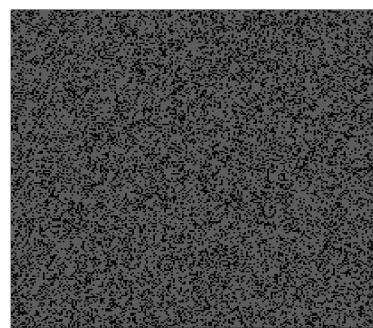
Sin embargo, cuando queremos hacer una comparación general de lo que ocurre con los períodos cuando varían las precisiones, se requiere una escala de colores, ver Figura 4.15.

La Figura 4.15 muestra que a medida que aumenta el valor de n_f el color del área se vuelve más uniforme y claro, lo que indica que las CIs convergen a ciclos de períodos más largos. Esto también se puede ver en el Cuadro 4.3, donde a medida que n_f aumenta, la longitud del ciclo límite predominante también aumenta.

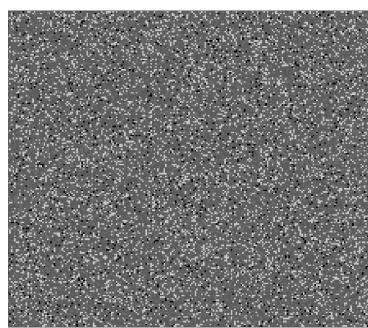
Para comparar los valores obtenidos con las secuencias reales, se iteraron los atractores en punto flotante con mantisa de 236 bits (IEEE754 de punto flotante binario de precisión octuple) lo que aquí se llamó *punto flotante* o simplemente *flotante*, es la aritmética más cercana a los números reales a la que podemos acceder con tiempos de cómputo razonables. Se puede observar que cuando se empleó la precisión mencionada, todos los períodos resultaron ser superiores a 10^5 y convergieron al atractor caótico que se ve en la Figura



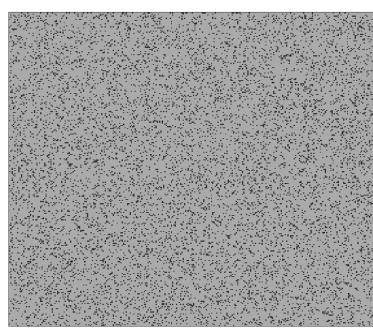
(a) Sección rectangular del dominio de atracción,
sección $[0,3 \pm 0,001, -0,2 \pm 0,001]$.



(b) $n_f = 14$.



(c) $n_f = 17$.



(d) $n_f = 18$.

Figura 4.14: Vistas ampliadas del dominio de atracción para distintos valores de n_f .

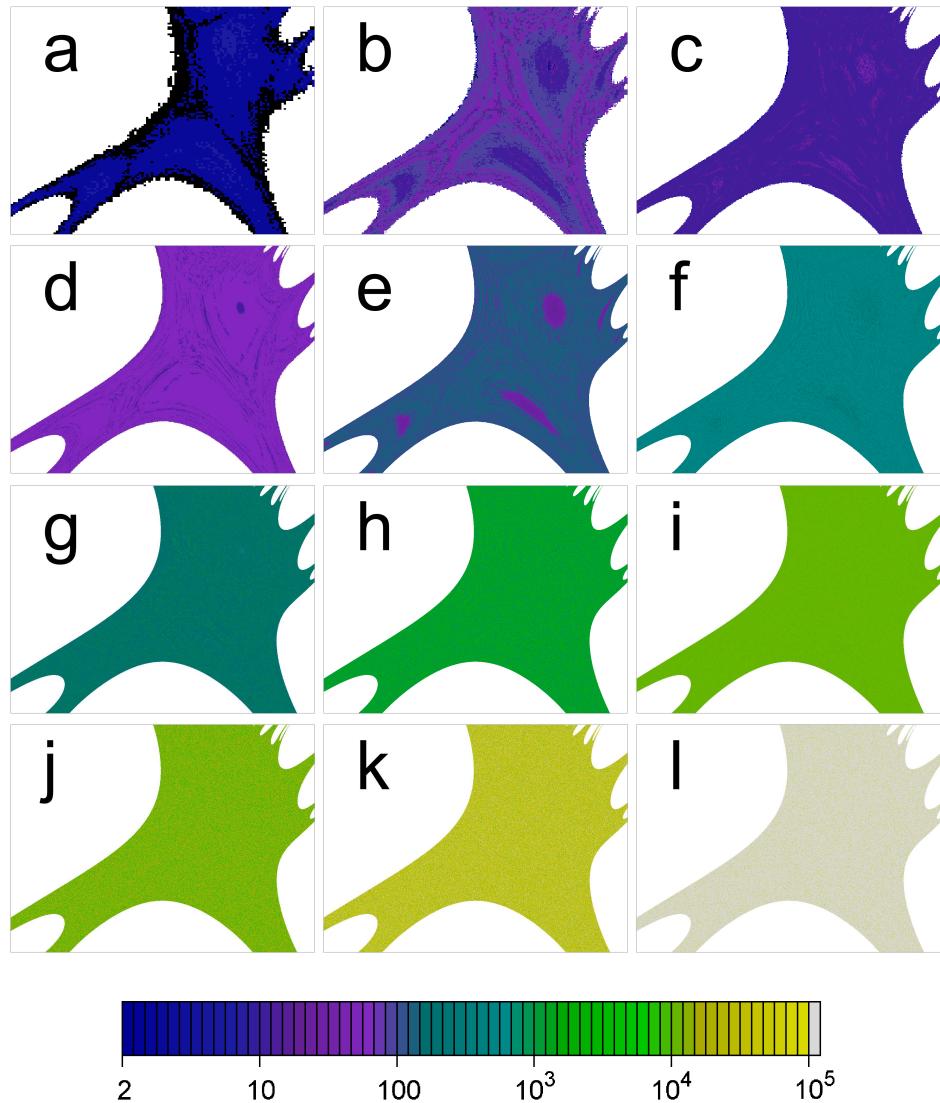


Figura 4.15: Evolución de las longitudes de período de los dominios de atracción para: (a) $n_f = 5$, (b) $n_f = 6$, (c) $n_f = 7$, (d) $n_f = 8$, (e) $n_f = 9$, (f) $n_f = 10$, (g) $n_f = 11$, (h) $n_f = 12$, (i) $n_f = 13$, (j) $n_f = 14$, (k) $n_f = 17$, (l) $n_f = 18$.

Cuadro 4.3: Longitudes de períodos dentro del dominio de atracción x e ε $[-2, 2]$.

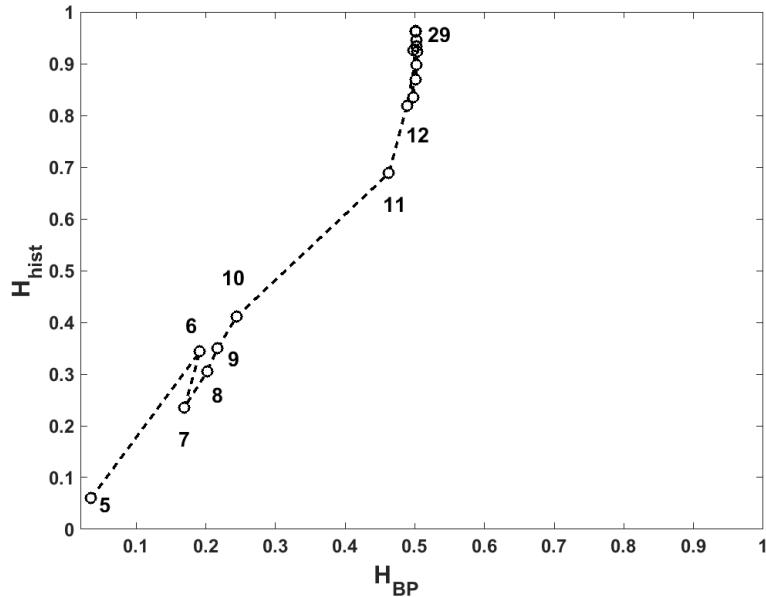
n_f	T (Porcentaje de CIs que convergen a esta longitud de período)
5	2 (92,7%); 6 (7,3%)
6	88 (41,6%); 44 (36,7%); 12 (13,8%); 16 (6,2%); 2 (0,8%); 24 (0,6%); 26 (0,2%)
7	12 (83,5%); 14 (8,9%); 24 (5,2%); 34 (1,8%); 2 (0,6%)
8	68 (91,7%); 14 (6,2%); 12 (1,8%); 17 (0,2%); 15 (0,1%)
9	140 (54,5%); 123 (25,4%); 34 (8,6%); 44 (4,3%); 38 (3,9%); 22 (2,9%); 48; 2; 12; 4 (< 0,1%)
10	655 (78,2%); 212 (21,1%); 143 (0,5%); 12 (0,1%); 2; 36; 13; 20; 10; 4 (< 0,1%)
11	153 (78,1%); 461 (10,8%); 1381 (8,7%); 434 (2,3%); 18; 30; 53; 32; 34; 10; 2 (< 0,1%)
12	2, 278 (64,4%); 438 (22,4%); 598 (7,6%); 886 (4,7%); 12 (0,7%); 87; 2; 42; 23; 32; 10 (< 0,1%)
13	11, 510 (98,9%); 1052 (1%); 12; 26; 2; 10 (< 0,1%)
14	21, 333 (69,2%); 5, 804 (16,5%); 4, 795 (7,9%); 1, 264 (5,8%); 2, 429 (0,5%) 46; 23; 21; 10; 12; 17 (< 0,1%)
15	10, 099 (58,6%); 1, 762 (19,4%); 14, 887 (18,3%); 1, 598 (3,4%); 750; 105; 23; 14; 2; 10 (< 0,1%)
16	54, 718 (87,5%); 5, 017 (4,7%); > 10 ⁵ (3,7%); 5, 367 (2,5%); 703 (0,9%) 1, 159; 1, 802 (0,2%); 377; 75; 10 (< 0,1%)
17	37, 812 (53,1%); 38, 456 (24,1%); > 10 ⁵ (16,0%); 34, 749 (3,0%); 3, 362; 718 (1,5%) 3, 006, 5, 222 (0,1%); 15 (< 0,1%)
18	> 10 ⁵ (87,4%); 52, 069 (12,5%); 2, 471 (0,1%); 146; 51 (< 0,1%)
float	> 10 ⁵ (100%)

2.20.d.

Sin embargo, alcanzar períodos largos no garantiza que los sistemas exhiban buenas propiedades con respecto a la aleatoriedad. Por lo que se decidió estudiar más a fondo los datos obtenidos mediante el empleo de cuantificadores estadísticos.

Como se dijo, en la Figura 4.13.a las dos zonas grises corresponden a las condiciones iniciales que convergen a los dos ciclos coexistentes de período dos y seis, respectivamente. Entonces, estos dos ciclos tendrán un valor determinado de H_{BP} , $H_{hist}|_{T=2} = 0,0625$, $H_{hist}|_{T=6} = 0,1199$, $H_{BP}|_{T=2} = 0,1053$ y $H_{BP}|_{T=6} = 0,2723$. Sin embargo, el valor reportado de estos cuantificadores no puede ser el promedio de ambos, ya que la tasa de ocurrencia del ciclo dos es mucho mayor que la del ciclo seis (el período dos aparece 92,7 % veces mientras que el período seis solo 7,3 %, ver Cuadro 4.3). Por lo tanto, se calcularon los cuantificadores promedios ponderando cada cuantificador con su tasa de ocurrencia.

El plano H_{hist} vs H_{BP} , que se muestra en la Figura 4.16, permite una visualización rápida del comportamiento en términos de aleatoriedad del sistema. Como se dijo, en este plano el punto “ideal”, desde el punto de vista estadístico, es (1, 1). Aquí, el sistema parece estabilizarse para n_f superior a 12. Se puede observar que mientras H_{hist} se estabiliza cerca del valor máximo ($H_{hist} = 1$), el H_{BP} tiende a estabilizarse a 0,5. Este valor de H_{BP} es característico de los sistemas caóticos y se debe a las ya mencionadas estructuras internas

Figura 4.16: Plano H_{hist} - H_{BP} para diferentes números de bits.

de sus atractores.

Un resumen del análisis observado de estos resultados se puede ver en la Figura 4.17. La Figura 4.17.a y 4.17.b muestran el número de puntos que divergen y convergen en puntos fijos respectivamente a medida que aumenta el valor de n_f , en ambos casos, el valor final tiende al que se obtiene en implementación en punto flotante. De estas Figuras se desprende que para $n_f \sim 12$ el sistema parece haberse estabilizado. La Figura 4.17.c muestra que el período promediado aumenta a una velocidad logarítmica. Finalmente, la Figura 4.17.d muestra el número de condiciones iniciales que presentan períodos T más altos y más bajos que 1 000. De nuevo, un valor de 12 para n_f parece ser el límite para obtener una buena aproximación del sistema.

La Figura 4.18 muestra el promedio ponderado de los cuantificadores H_{hist} , H_{BP} y MLE . En la Figura se puede ver que los tres cuantificadores tienden al valor calculado usando aritmética de punto flotante. Mientras H_{BP} y MLE se estabilizan para $n_f \sim 12$ o 13, H_{hist} alcanza el valor en coma flotante de $n_f \sim 19$, mostrando que hay propiedades de las secuencias de salida que sólo este cuantificador puede detectar. Esto confirma la necesidad de usar ambos cuantificadores para caracterizar la aleatoriedad de las secuencias.

Como puede verse en el análisis anterior, el número mínimo de bits está determinado por H_{hist} y los resultados son $n_f = 19$ más la cantidad de bits utilizados para representar la

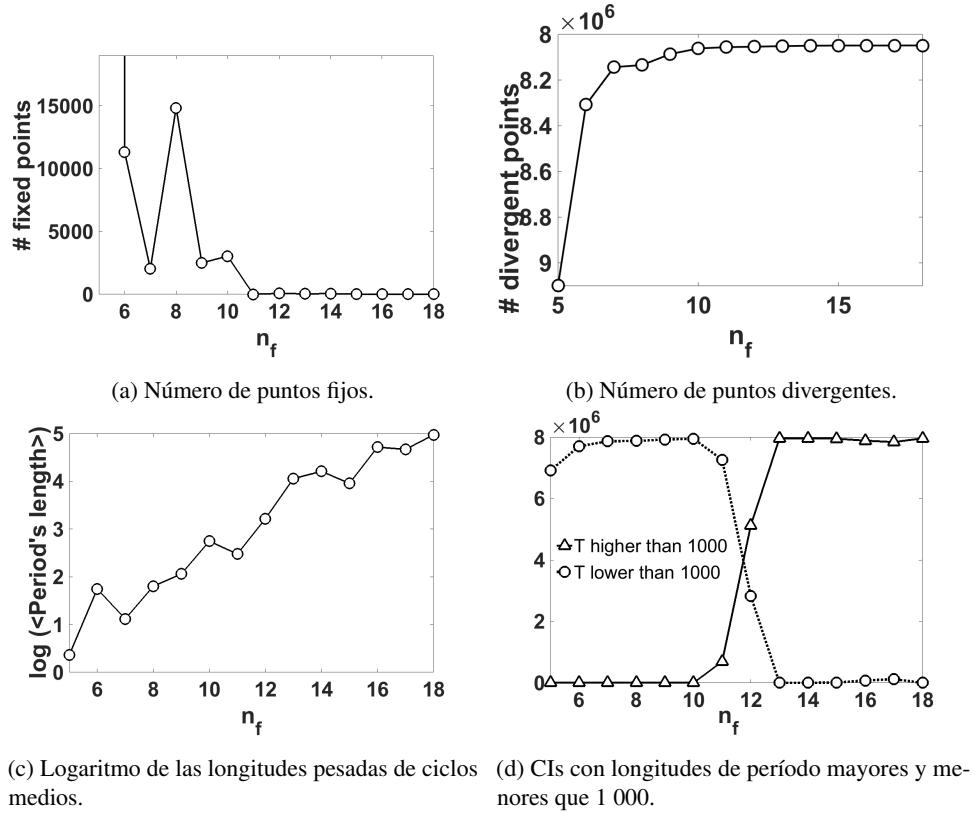
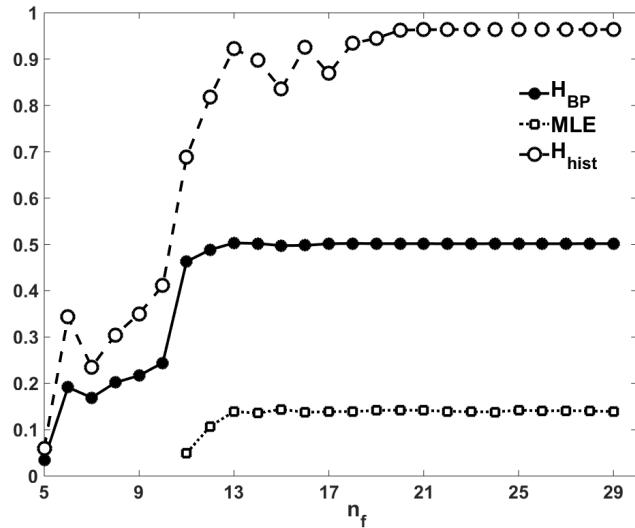


Figura 4.17: Comportamiento de las condiciones iniciales.


 Figura 4.18: Promedio pesado de cuantificadores H_{BP} , H_{hist} y MLE en función del número de bits.

parte entera $n_i = 4$, por lo tanto n_{min} resulta ser igual a 23.

4.5. Conclusiones

En este Capítulo se resumen los resultados de cuatro trabajos orientados a la implementación de sistemas caóticos. Se propusieron y analizaron distintos sistemas y aplicaciones desde el punto de vista estadístico, siempre contemplando la implementación orientada a la ingeniería.

Con respecto a la exploración de caos en RNAs, encontramos que es posible conseguir caos robusto en una zona del espacio de los parámetros utilizando un sistema de tres neuronas. Se obtuvo un MLE positivo en toda una zona del parámetro P , además la trayectoria del atractor no fue divergente, lo que confirma la existencia de caos para todo el rango.

Se desarrolló un nuevo método para el diseño de sistemas de criptocodificación mediante el empleo de mapas caóticos cuadráticos bidimensionales. Se obtuvieron resultados muy prometedores hasta el momento mediante las simulaciones realizadas. También se obtuvieron buenos resultados en la implementación en VHDL del sistema codificador, en la que se verificó que la salida generada por el sistema no cayera en ciclos periódicos debido a la utilización de precisión finita.

A partir de los resultados presentados para PRNGs determinístico-estocásticos, es posible concluir que los mejores PRNGs en cuanto a recursos y frecuencia de operación, resultaron ser los representados por aritmética entera. La técnica de aleatorización de *concatenado* hace que la calidad sea independiente de Δt para esta representación numérica. Para la técnica de aleatorización de *descarte* se obtienen buenos resultados sólo para valores grandes de Δt . Lo mismo ocurre con las implementaciones de punto fijo con ambas técnicas de aleatorización. En términos de uso de recursos y limitaciones de frecuencia, el rendimiento en aritmética de enteros es considerablemente mejor que en punto flotante y punto fijo. Se observó que, para minimizar los recursos, se requiere un preprocesamiento de las ecuaciones del sistema caótico (escalado y polarización) para obtener divisores que sean potencia de 2, tal como se explica en la Subsección 4.3.2. Por otro lado, para el caso de la aritmética de punto flotante, el exponente que indica la posición de la coma se descarta en todas las técnicas de aleatorización utilizadas y, en consecuencia, la dinámica del sistema se ve muy perturbada por el proceso de aleatorización. Entonces, como se muestra en el Cuadro 4.2, Δt no es la variable relevante para predecir si un PRNG será bueno o malo.

Finalmente, se desarrolló un análisis detallado de los cambios en el comportamiento de una implementación en punto fijo de un mapa cuadrático bidimensional. El objetivo fue reportar la tasa de degradación de las propiedades de cada sistema, para que para que puedan ser utilizadas en el diseño de aplicaciones particulares. Los resultados muestran que es posible determinar un umbral para el número de bits empleados en la representación de punto fijo del sistema, mientras que el dominio de atracción conserva su integridad y se mantienen las características de las secuencias generadas. Con la ayuda de los cuantificadores de aleatoriedad introducidos fue posible determinar ese límite, en el caso del estudio fue de 23 bits. El mismo procedimiento debe repetirse para cualquier otro sistema si se desea utilizarlo en una aplicación electrónica digital, como generadores de ruido controlados o para desarrollar nuevos sistemas de encriptación.

Capítulo 5

Mapas Conmutados en Precisión Finita

5.1. Introducción

En los últimos años, los sistemas digitales se convirtieron en el estándar en todas las ciencias experimentales. Mediante el uso de nuevos dispositivos electrónicos programables como DSP y electrónica reconfigurable como FPGA o ASIC, los experimentadores pueden diseñar y modificar sus propios generadores de señales, sistemas de medición, modelos de simulación, etc.

En estos nuevos dispositivos, el punto flotante y el punto fijo son las aritméticas más comunes. El punto flotante es la solución más precisa cuando no se conoce el rango de valores que se implementará, pero no siempre se recomienda cuando se requieren velocidad, baja potencia y/o área de circuito pequeño, una solución de punto fijo es mejor en estos casos.

El efecto de la discretización numérica sobre un mapa caótico fue abordado recientemente en [20], en donde se explora la degradación estadística del espacio de fases para una familia de mapas cuadráticos 2D. Estos mapas presentan una dinámica multiatractor que los hace muy atractivos como generadores de números aleatorios en campos como criptografía, codificación, etc. En [111] y [112], los autores propusieron usar el valor de la entropía para elegir el número de bits en la parte fraccionaria, cuando se implementan mapas en aritmética entera.

En [108], Grebogi y colaboradores estudiaron este tema y vieron que el período T escala con el redondeo ε como $T \sim \varepsilon^{-d/2}$ donde d es la dimensión de correlación del atractor caótico. Conseguir un período grande T es una propiedad importante de los mapas caóticos, en [113] Nagaraj *et al.* estudiaron el efecto de cambiar las longitudes de período promedio de los mapas caóticos en precisión finita. Vieron que el período T del mapa compuesto obtenido al conmutar entre dos mapas caóticos es mayor que el período de cada mapa. Liu *et al.* [114] estudiaron diferentes reglas de conmutación aplicadas a sistemas lineales para generar caos. El problema de la conmutación también se trató en [115], el autor consideró algunos aspectos matemáticos, físicos y de ingeniería relacionados con sistemas singulares, principalmente de conmutación. Los sistemas conmutados surgen naturalmente en la electrónica de potencia y en muchas otras áreas de la electrónica digital.

La estocasticidad y la mezcla también son relevantes para caracterizar un comportamiento caótico. Para investigar estas propiedades, se estudiaron varios cuantificadores [58]. La entropía y la complejidad de la teoría de la información se aplicaron para dar una medida de la entropía causal y no causal y la complejidad causal.

Los resultados presentados en este Capítulo corresponden al artículo publicado en [13], en donde se investigan distintas estrategias de conmutación para mejorar la estocasticidad, mezcla y período de mapas pseudocaóticos.

5.2. Herramientas de Análisis Estadístico

Existe un amplio abanico de test estadísticos, aquí se consideró la PDF tradicional no causal obtenida al normalizar el histograma de la serie temporal. Su cuantificador estadístico es la entropía normalizada H_{hist} , que es una medida de equiprobabilidad entre todos los valores permitidos. También se consideró una PDF causal que se obtiene asignando patrones de orden a segmentos de trayectoria de longitud D .

La entropía correspondiente H_{BP} también fue propuesta como un cuantificador por Bandt & Pompe, en [59] los autores aplicaron la complejidad causal C_{BP} para detectar el caos. Entre ellos, merece una consideración especial el uso de una representación planar de complejidad y entropía (plano $H_{hist} \times C_{BP}$) y el plano entropía causal vs. no causal (plano $H_{BP} \times H_{hist}$). Para poder detectar caídas a puntos fijos o trayectorias que divergen utilizamos la entropía causal con contribución de amplitudes H_{BPW} . Como se mencionó en la Sección 3.2, esta entropía se basa en un histograma de patrones de orden en donde cada símbolo está

pesado por su frecuencia de aparición y su varianza. Un aspecto a tomar en cuenta es que no existe PDF con contribución de amplitud para un vector con todos sus valores iguales, ya que todas las veces aparece el mismo símbolo pesado por 0, lo que da como resultado una PDF vacía. También utilizamos la cantidad de patrones faltantes (MP) para calcular un límite superior para los cuantificadores causales.

Siguiendo el análisis realizado en [113], en este caso se estudian las características estadísticas de cinco mapas, dos mapas bien conocidos: (1) los mapas Tent (TENT) y (2) Logístico (LOG), y tres mapas adicionales generados a partir de ellos: (3) SWITCH, generado al comutar entre TENT y LOG; (4) EVEN, generado al omitir todos los elementos en posiciones impares de la serie temporal SWITCH y (5) ODD, generados descartando todos los elementos en posiciones pares en la serie de tiempo SWITCH. Se emplearon números binarios flotantes y de punto fijo, estos sistemas numéricos específicos pueden implementarse en dispositivos modernos lógicos programables.

5.3. Resultados

Se estudiaron cinco mapas pseudocaóticos: dos mapas simples y tres combinaciones de ellos. Para cada uno se emplearon números representados por coma flotante (80 bits de mantisa) y números de punto fijo con $1 \leq B \leq 53$, donde B es el número de bits que representa la parte fraccionaria. Las series de tiempo se generaron usando 100 condiciones iniciales elegidas al azar dentro de su dominio de atracción (intervalo $[0, 1]$), para cada una de estas 54 precisiones numéricas. Los mapas estudiados son: LOG, TENT, SWITCH, EVEN y ODD.

El mapa Logístico fue elegido por ser representativo de la gran familia de mapas cuadráticos, fue presentado den la Sección 2.2. Para trabajar en una representación dada, es necesario cambiar la expresión del mapa de la Ecuación 2.8 para realizar todas las operaciones en los números de representación elegidos. Por ejemplo, en el caso de LOG, la expresión en números binarios de punto fijo es:

$$x_{n+1} = 4\epsilon \text{ floor}\left\{\frac{x_n(1-x_n)}{\epsilon}\right\} \quad (5.1)$$

con $\epsilon = 2^{-B}$ donde B es la cantidad de bits que representa la parte fraccionaria.

En la ecuación anterior, la función floor() descarta la parte fraccionaria del argumento.

Ésta técnica de redondeo es la misma que la utilizada en [18, 108, 113] y tiene algunas ventajas, ya que es algorítmicamente fácil de implementar y es independiente de la plataforma en donde es utilizada, siempre y cuando B sea menor que la mantisa de la unidad aritmética lógica de la máquina local. Para obtener los resultados, se trabajó con una PC Intel i7, que cuenta con una ALU con estándar IEEE-754 de punto fijo de doble precisión, lo cual limita el método a $B \leq 53$ bits.

El mapa TENT ha sido ampliamente estudiado en la literatura porque teóricamente tiene buenas propiedades estadísticas que pueden obtenerse analíticamente. Por ejemplo, es fácil probar que tiene un histograma uniforme y, en consecuencia, un $H_{hist} = 1$ ideal. El operador Perron-Frobenius y sus autovalores y autovectores correspondientes se pueden obtener analíticamente para este mapa [27].

En el redondeo de números fraccionarios base-2, la Ecuación 2.9 se convierte en:

$$x_{n+1} = \begin{cases} \varepsilon \text{ floor}\left\{\frac{1}{\varepsilon}\mu(x_n)\right\} & , \text{if } 0 \leq x_n \leq \mu^- \\ \varepsilon \text{ floor}\left\{\frac{1}{\varepsilon}\rho(1-x_n)\right\} & , \text{if } \mu^- < x_n \leq 1 \end{cases} \quad (5.2)$$

con $\varepsilon = 2^{-B}$, $\mu = \varepsilon \text{ floor}\left\{\frac{1}{\varepsilon}u\right\}$, $\mu^- = \varepsilon \text{ floor}\left\{\frac{1}{\varepsilon}(1/\mu)\right\}$ y $\rho = \varepsilon \text{ floor}\left\{\frac{1}{\varepsilon}(\mu/(1-\mu))\right\}$.

En [112], los autores mostraron la evolución de la entropía de valores H_{hist} con respecto a la precisión binaria. Ellos caracterizaron la evolución del mapa TENT como función de la precisión binaria en aritmética de punto fijo. En su esquema de generación de números aleatorios usaron dos etapas de postprocesamiento, primero binarizaron los datos detectando el cruce por un umbral y luego estos datos fueron procesados por una compuerta XOR. Aquí se utilizó la salida de los mapas caóticos sin ningún postprocesamiento de aleatorización, sin embargo, sus resultados son muy interesantes y permiten obtener un criterio acerca de cuáles parámetros son útiles para implementar. Aquí se utilizaron dos valores de u por dos distintas razones, siguiendo a [112] un valor interesante es $u = 1,96$ o el valor más cercano en la aritmética utilizada, por otro lado, el valor $u = 2$ es muy atractivo dado su extremadamente bajo costo de implementación.

En la Figura 5.1 se muestran los procedimientos de conmutación, skipping par y skipping impar.

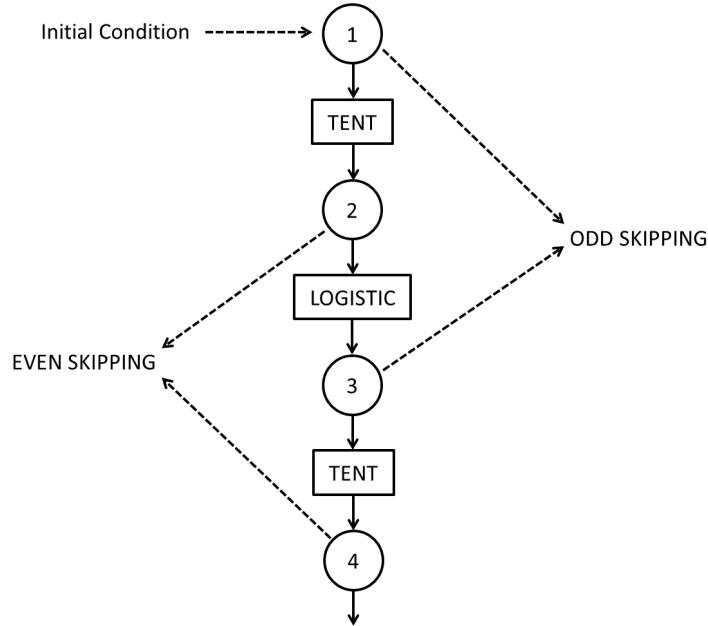


Figura 5.1: Comutación secuencial entre TENT y LOG. En la figura también se muestran las estrategias de skipping par e impar.

El mapa SWITCH se expresa como:

$$\begin{cases} x_{n+1} = \begin{cases} ux_n & , \text{if } 0 \leq x_n \leq 1/u \\ \frac{u}{1-u}(1-x_n) & , \text{if } 1/u < x_n \leq 1 \end{cases} \\ x_{n+2} = 4x_{n+1}(1-x_{n+1}) \end{cases} \quad (5.3)$$

con $x_n \in \mathbb{R}$ y n un número par.

Sin embargo, como en el resto de los casos, se utiliza su contraparte pseudocaótica, que puede ser expresada como:

$$\begin{cases} x_{n+1} = \begin{cases} \varepsilon \text{ floor}\left\{\frac{1}{\varepsilon}\mu(x_n)\right\} & , \text{if } 0 \leq x_n \leq \mu^- \\ \varepsilon \text{ floor}\left\{\frac{1}{\varepsilon}\rho(1-x_n)\right\} & , \text{if } \mu^- < x_n \leq 1 \end{cases} \\ x_{n+2} = 4\varepsilon \text{ floor}\left\{\frac{x_n(1-x_n)}{\varepsilon}\right\} \end{cases} \quad (5.4)$$

El skipping es una técnica habitual de aleatorización que sólo aumenta la calidad de mezcla de un mapa y, por consiguiente, aumenta el valor de H_{BP} y disminuye C_{BP} de la serie temporal. El skipping no cambia los valores de H_{hist} para los mapas ergódicos porque

se evalúan utilizando el PDF no causal, como se describió en la Sección 3.2.

En el caso bajo consideración, se estudiaron saltos pares e impares de la conmutación secuencial de los mapas de Tent y de Logístico:

1. Skipping par de la conmutación secuencial de mapas TENT y LOG (EVEN). Si $\{x_n; n = 1, \dots, \infty\}$ es la serie de tiempo generada por la Ecuación 5.3, descarta todos los valores en posiciones impares y conserva los valores en posiciones pares.
2. Skipping impar de la conmutación secuencial de mapas TENT y LOG. Si $\{x_n; n = 1, \dots, \infty\}$ es la serie de tiempo generada por la Ecuación 5.3, descarta todos los valores en posiciones pares y conserva todos los valores en posiciones impares.

El skipping par se puede expresar como la función de composición $\text{TENT} \circ \text{LOG}$ mientras que el skipping impar se puede expresar como $\text{LOG} \circ \text{TENT}$.

5.3.1. Período T en Función de B

Según [108], el período T se relaciona con la precisión ε como $T \sim \varepsilon^{-d/2}$ donde d es la dimensión de correlación del atractor caótico. La evolución del período como función de la precisión para los mapas que se muestran aquí se informó en [113]. Allí mostraron que el período T del mapa compuesto obtenido al conmutar entre dos mapas caóticos es más alto que el período de cada mapa y encontraron que una conmutación aleatoria mejora los resultados. Aquí hemos considerado el solo la conmutación secuencial para evitar el uso de otra variable aleatoria, ya que esta puede introducir sus propias propiedades estadísticas en la serie temporal.

La Figura 5.2 muestra T vs. B en escala semi logarítmica para el mapa LOG. Para cada precisión, se generaron 100 surrogados diferentes, cada uno con una condición inicial generada aleatoriamente. La Figura muestra 100 puntos rojos por cada precisión de punto fijo ($1 \geq B \geq 53$) y los resultados promediados (puntos negros conectados con líneas entrecortadas negras). Los puntos promediados experimentales se pueden ajustar por una línea recta expresada como $\log_2 T = mB + b$ donde m es la pendiente y b es la ordenada al origen.

Los resultados para todos los mapas considerados se resumen en el Cuadro 5.1. Pudimos detectar que el período promediado fue el mismo usando $u = 2$ y $u = 1,96$ cuando se utiliza la estrategia de switching. Por lo tanto, para calcular los resultados mostrados en el Cuadro para SWITCH, EVEN y ODD se iteró con $u = 2$.

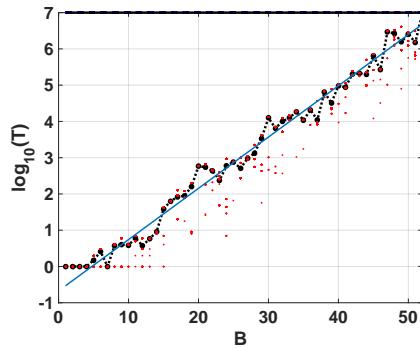


Figura 5.2: Período T en función de la precisión B en números binarios para el mapa LOG.

Cuadro 5.1: Período T en función de la precisión B para todos los mapas considerados. SWITCH, EVEN y ODD fueron calculados con $u = 2$.

mapa	m	b
TENT $u = 2$	0	0
TENT $u = 1,96$	0.1487	-0.01177
LOG	0.139	-0.6188
SWITCH	0.1462	-0.5115
EVEN	0.1447	-0.7783
ODD	0.1444	-0.7683

Los resultados son compatibles con los obtenidos en [113]. La conmutación entre mapas aumenta el período T pero el procedimiento de skipping lo disminuye casi a la mitad. Se puede observar que, los resultados para el mapa TENT con $u = 1,96$ exhibe los mejores resultados.

5.3.2. Cuantificadores de Mapas Simples

A continuación se analizan los resultados obtenidos para los mapas simples LOG y TENT.

Mapa LOG

Las Figuras 5.3a a 5.3f muestran las propiedades estadísticas del mapa LOG en representación de coma flotante y punto fijo. Todas estas Figuras muestran: 100 puntos rojos (surrogados) por cada precisión de punto fijo ($1 \geq B \geq 53$) y en negro su promedio (línea negra discontinua que conecta puntos negros), 100 líneas discontinuas horizontales azules que son el resultado de cada surrogado en punto flotante y una línea continua negra en su promedio. se debe tener en cuenta que estas líneas son independientes del eje x. En este caso, todas las líneas del punto flotante se superponen.

Según B crece, las propiedades estadísticas varían hasta que se estabilizan. Para $B \geq 30$, el valor de H_{hist} permanece casi idéntico al valor de la representación en coma flotante, mientras que H_{BP} y C_{BP} se estabilizan a $B > 21$. Sus valores son: $\langle H_{hist} \rangle = 0,9669$; $\langle H_{BP} \rangle = 0,6269$; $\langle C_{BP} \rangle = 0,4843$. Cabe destacar que el valor estable de los patrones faltantes $MP = 645$ hace que el valor óptimo sea $H_{BP} \leq \ln(75)/\ln(720) \simeq 0,65$. Entonces, $B = 30$ es la opción más conveniente para la implementación en hardware porque un aumento en el número de dígitos fraccionarios no mejora las propiedades estadísticas.

Se pueden sacar algunas conclusiones comparando los cuantificadores de BP y BPW. Para $B = 1, 2, 3$ y 4 , los cuantificadores de BP promediados toman valores cercanos a 0 mientras que no existe solución para los cuantificadores de BPW promediados (ver en las Figuras 5.3c y 5.3e la línea punteada negra no aparece entre $B = 12345$). Esto se debe a que para esas secuencias la condición inicial fue cero, todas las iteraciones resultan ser una secuencia de ceros (el punto fijo del mapa), esto es más probable que ocurra cuando se usan pequeñas precisiones debido al redondeo.

Cuando B aumenta las condiciones iniciales se redondean a cero con menos frecuencia, esto se puede ver para $B > 6$. En este caso, las secuencias generadas que comienzan desde

un valor no nulo caen a cero después de un transitorio corto muy frecuentemente. Un tema interesante en las Figuras 5.3c y 5.3e, es que los cuantificadores de BPW muestran una alta dispersión a diferencia de los cuantificadores de BP. Esto se debe a que el procedimiento BPW tiene en cuenta transitorios y descarta los puntos fijos, a diferencia del procedimiento BP, que considera todos los valores de la secuencia. Podemos ver en las Figuras 5.3c y 5.3e para $1 < B < 10$ líneas horizontales de puntos rojos que no aparecen en las Figuras 5.3b y 5.3d, esto evidencia que las diferentes condiciones iniciales caen en las mismas órbitas, incluso para las precisiones adyacentes.

Los mismos resultados se muestran en planos de doble entropía con la precisión como parámetro (Figuras 5.4a sin contribuciones de amplitud y Figura 5.4b con contribuciones de amplitud). Estas Figuras muestran: 100 puntos rojos por cada precisión de punto fijo (B) y su promedio en negro (línea negra discontinua que conecta puntos negros), 100 puntos azules que son los resultados de cada surrogado en coma flotante y la estrella negra es su promedio. Aquí, los 100 puntos azules y su promedio se superponen.

Como se esperaba, la implementación de la arquitectura de punto fijo converge al valor de coma flotante a medida que aumenta B . Para ambos planos, $H_{hist} \times H_{BP}$ y $H_{hist} \times H_{BPW}$, desde $B = 20$, H_{hist} aumenta, pero H_{BP} y H_{BPW} permanecen constantes. Se puede ver que la entropía de distribución de valores es alta ($\langle H_{hist} \rangle = 0,9669$) pero su mezcla es pobre ($\langle H_{BP} \rangle = 0,6269$).

En la Figura 5.5a y 5.5b, se muestran los planos entropía-complejidad. Las líneas grises punteadas son los márgenes superior e inferior, se espera que un sistema caótico permanezca cerca del margen superior. Estos resultados caracterizan un comportamiento caótico, en el plano $H_{BP} \times C_{BP}$ podemos ver una baja entropía y alta complejidad.

Mapa TENT

La ecuación que representa la implementación para $u = 2$ es:

$$x_{n+1} = \begin{cases} 2x_n & , \text{if } 0 \leq x_n \leq 0,5 \\ \varepsilon \text{ floor}\left\{\frac{1}{\varepsilon}2(1-x_n)\right\} & , \text{if } 0,5 < x_n \leq 1 \end{cases} \quad (5.5)$$

En este caso, el redondeo es necesario sólo en la segunda multiplicación porque la multiplicación por dos es equivalente a un desplazamiento hacia la izquierda.

Cuando este mapa se implementa con $u = 2$ en una computadora que utiliza cualquier

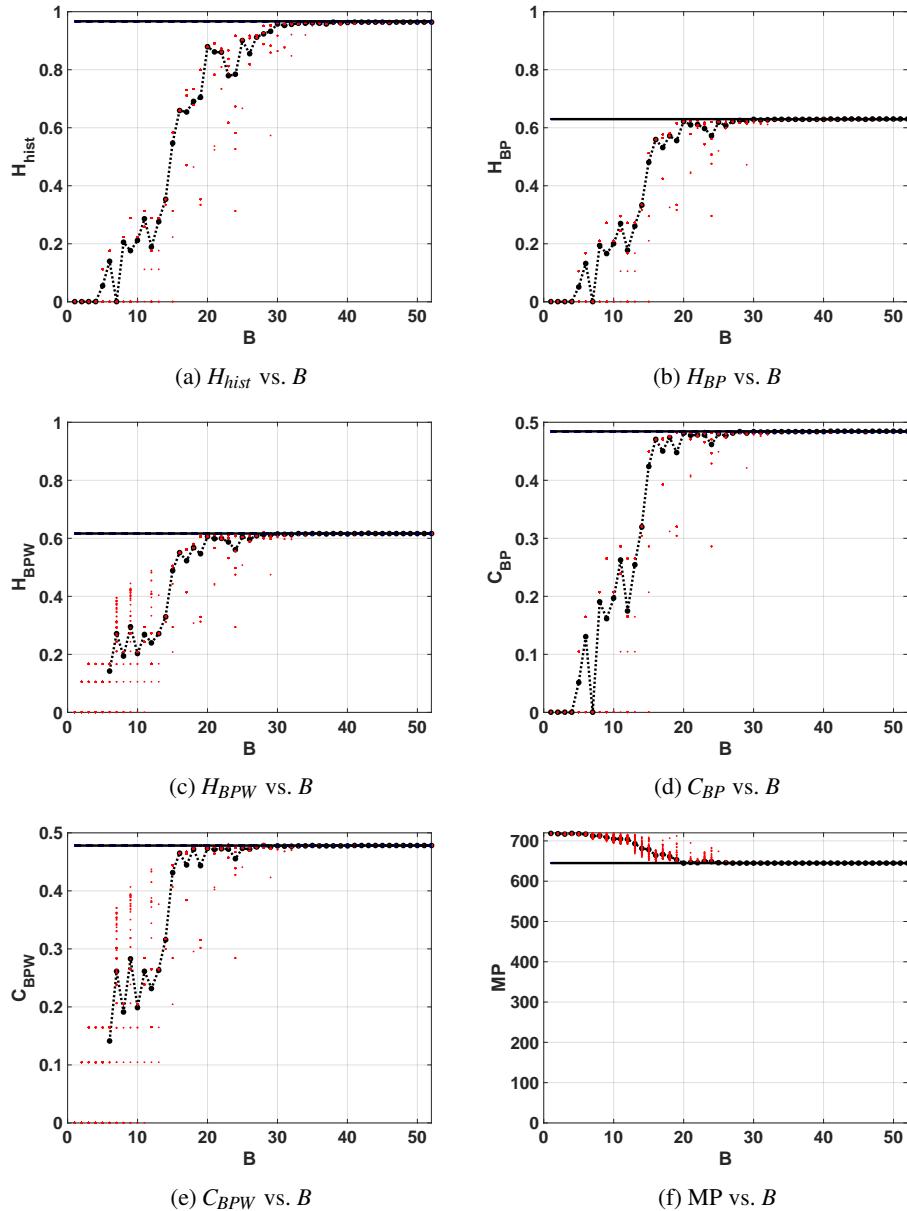


Figura 5.3: Propiedades estadísticas para el mapa LOG en función de B .

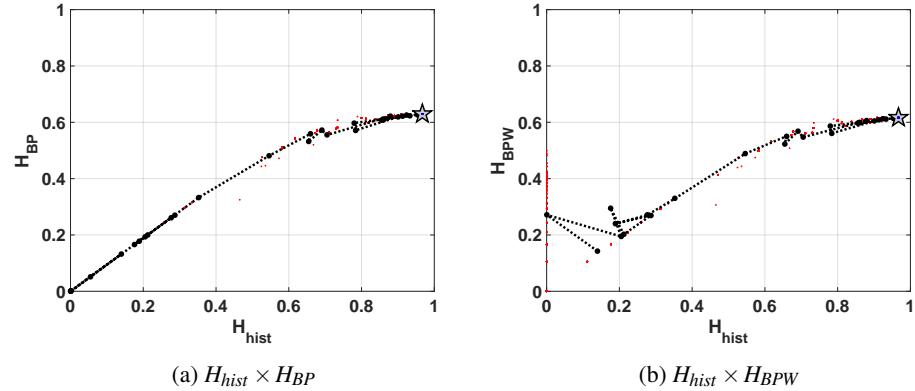


Figura 5.4: Evolución de las propiedades estadísticas en el plano de doble entropía para el mapa LOG.

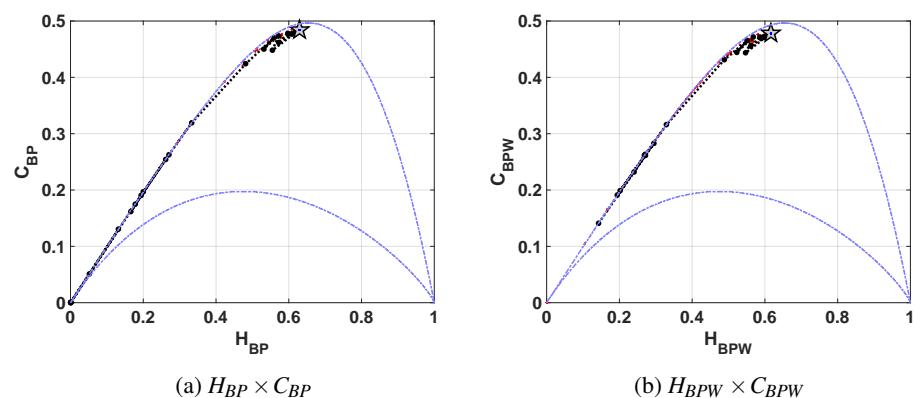


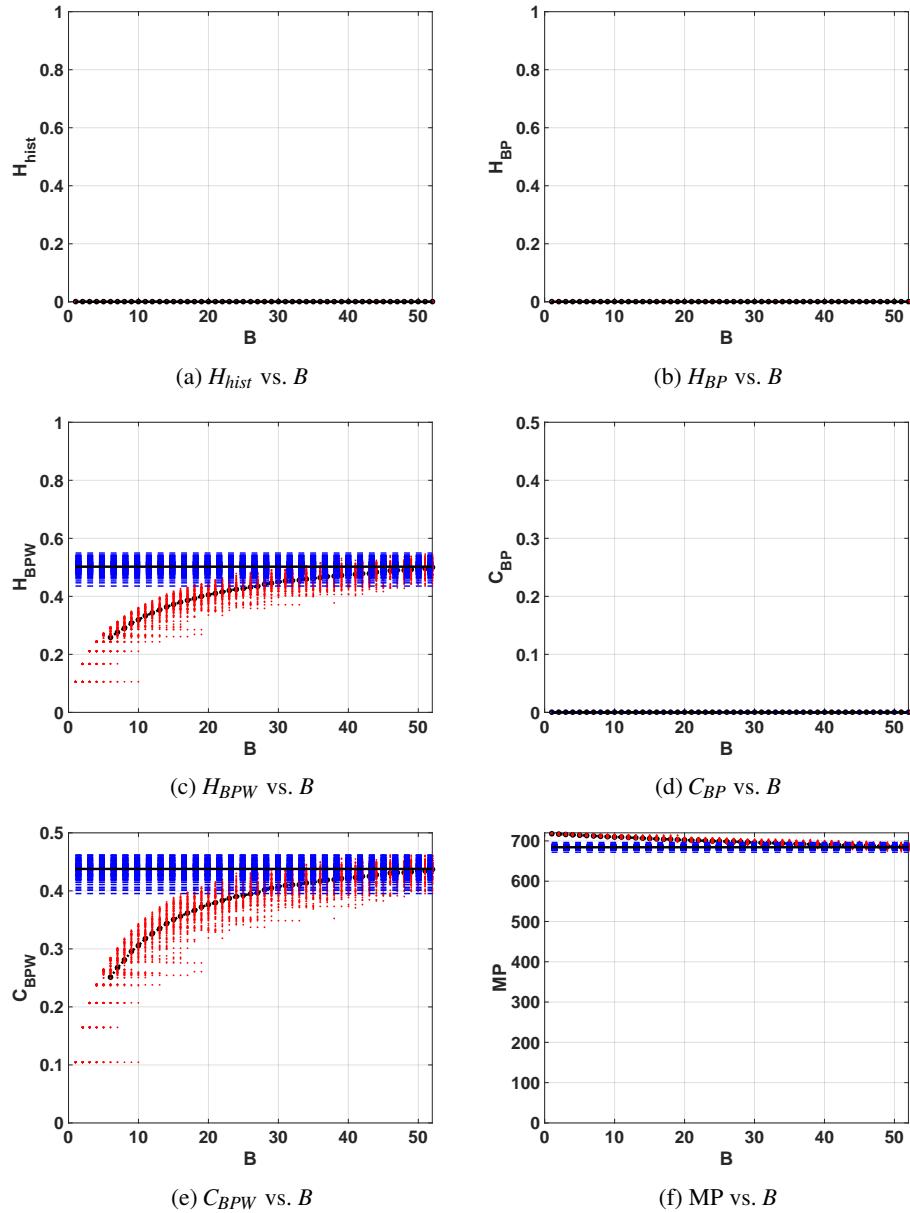
Figura 5.5: Evolución de las propiedades estadísticas en el plano causal entropía-complejidad para el mapa LOG.

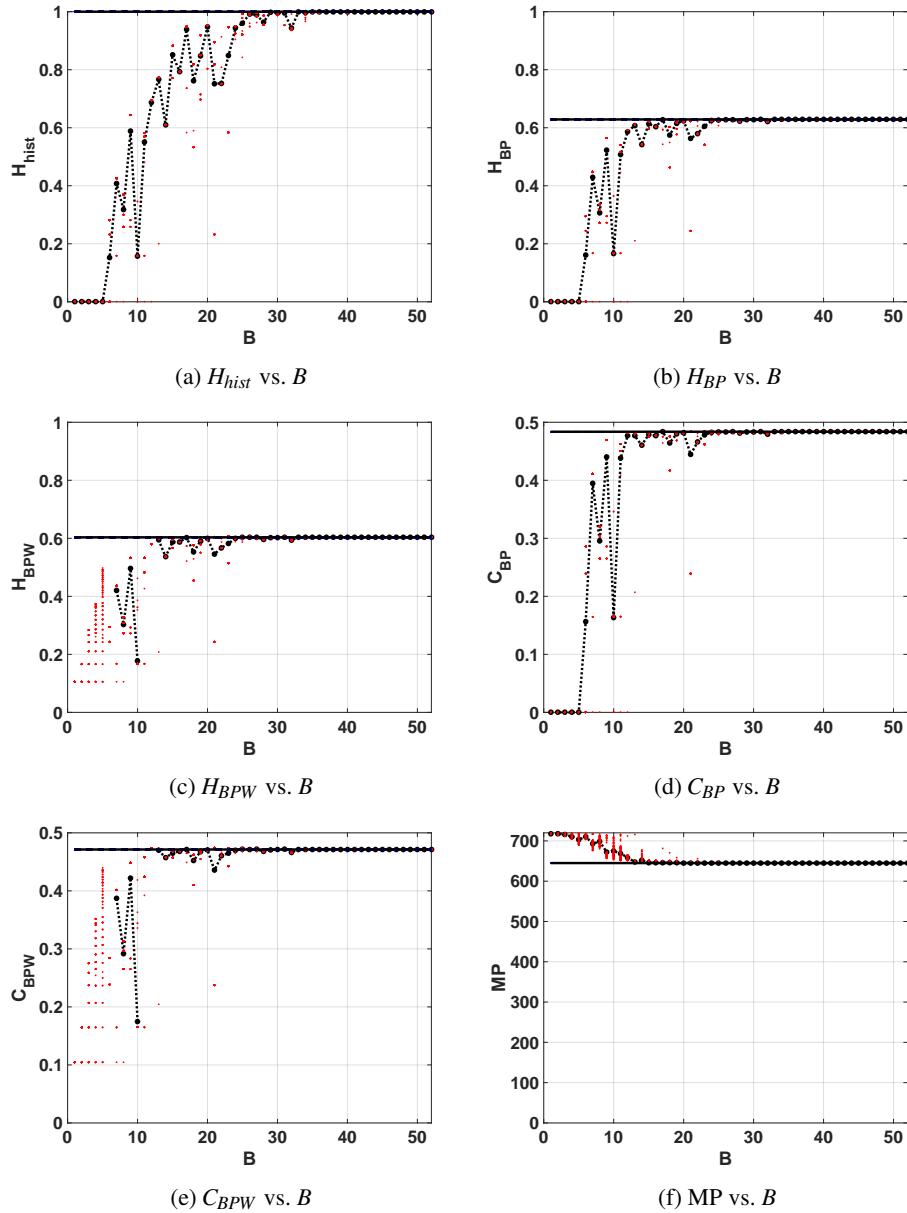
sistema de representación numérica binaria (¡Incluso punto flotante!), los errores de truncamiento aumentan rápidamente y hacen que el punto fijo inestable en $x^* = 0$ se estabilice. Las secuencias dentro del dominio de atracción de este punto fijo tendrán un transitorio corto de longitud entre 0 y B seguido de un número infinito de 0s [116, 117]. Este problema se explica de forma muy sencilla en [29], el problema aparece porque todas las iteraciones tienen una operación de desplazamiento a la izquierda que arrastra los ceros del lado derecho del número a las posiciones más significativas.

Las Figuras 5.6a a 5.6f muestran los cuantificadores para representaciones numéricas de coma flotante y fija. Los cuantificadores H_{hist} , H_{BP} y C_{BP} son iguales a cero para todas las precisiones, esto refleja que las series convergen rápidamente hacia un punto fijo para cada condición inicial. En el caso de H_{BPW} y C_{BPW} los cuantificadores son diferentes a cero porque el procedimiento BPW descarta los elementos una vez que se alcanza el punto fijo. Las altas dispersiones en H_{BPW} , C_{BPW} y MP están relacionadas con la corta duración del transitorio de la serie. Estos transitorios que convergen a un punto fijo tienen una longitud máxima de B elementos (iteraciones) para aritmética de punto fijo y 80 para punto flotante (precisión long-double). Resumiendo, no importa la cantidad de bits que se utilicen (con cualquier representación numérica en base 2) para representar el mapa TENT digitalizado, éste siempre converge rápidamente al punto fijo en $(x_n, x_{n+1}) = (0, 0)$.

Cuando este mapa se calcula con un valor distinto de u los resultados son completamente distintos, como se ve en las Figuras 5.7, en donde se muestran los resultados tomando $u = 1,96$. Los resultados son similares a los del mapa LOG, en cuanto a que el valor promediado de los cuantificadores tiende no monótonamente al valor encontrado en punto flotante y se estabiliza a partir de cierto valor de B . Sin embargo, fueron necesarios más bits de precisión para alcanzar estas asíntotas. Por otro lado, se puede ver que el valor de H_{hist} es mejor que el encontrado para LOG y que H_{BP} es similar. Mediante los cuantificadores H_{BPW} y C_{BPW} se puede detectar que, con una precisión por debajo de 13 bits, algunas condiciones iniciales convergen a puntos fijos o divergen, por lo que no es posible utilizar este mapa con $B < 13$.

Las posiciones en los planos doble entropía (Figura 5.8) y entropía-complejidad (Figura 5.8) son levemente mejores que los que se obtienen para el mapa LOG y son característicos de los mapas pseudocaóticos.

Figura 5.6: Propiedades estadísticas del mapa TENT con $u = 2$.

Figura 5.7: Propiedades estadísticas del mapa TENT con $u = 1.96$.

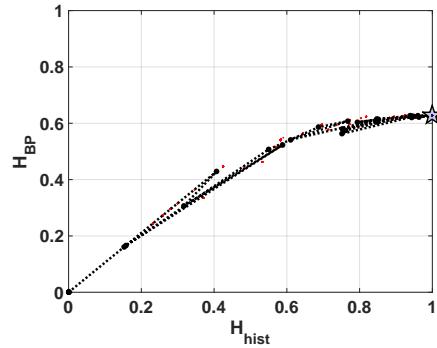


Figura 5.8: Evolución de las propiedades estadísticas en el plano causal entropía-complejidad para el mapa TENT con $u = 1,96$.

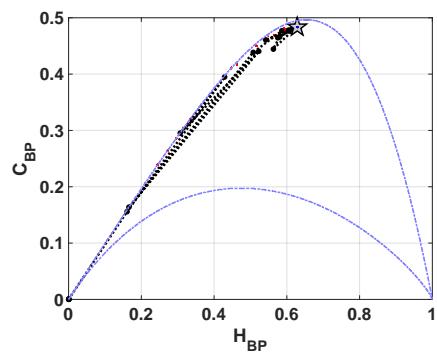


Figura 5.9: $H_{BP} \times C_{BP}$

Figura 5.10: Evolución de las propiedades estadísticas en el plano causal entropía-complejidad para el mapa TENT con $u = 1,96$.

5.3.3. Cuantificadores de Mapas Combinados

A continuación, se analizan los resultados obtenidos para los mapas generados a partir de las combinaciones de los mapas simples, SWITCH, EVEN y ODD.

Mapa SWITCH

Entre los dos parámetros analizados para el mapa TENT, se encontró que cuando se utilizan dentro del esquema switcheado, la mezcla y la estocasticidad convergen a los mismos valores ya sea con $u = 2$ como con $u = 1,96$. Entonces, elegimos $u = 2$ dada su simplicidad para ser utilizado tanto en simulaciones de software como en la implementación en hardware.

Los resultados con conmutación secuencial se muestran en las Figuras 5.11a a 5.11f. El valor de entropía calculado para la implementación en punto flotante es $\langle H_{hist} \rangle = 0,9722$, este valor es ligeramente más alto que el obtenido para el mapa LOG. Para la aritmética de punto fijo, este valor se alcanza en $B = 24$, pero se estabiliza desde $B = 28$. En cuanto a los patrones faltantes, el número de MP disminuye a 586, este valor es menor que el obtenido para el mapa LOG. Significa que la entropía H_{BP} puede aumentar hasta $\ln(134)/\ln(720) \simeq 0,74$. Los cuantificadores BP y BPW alcanzan su máximo de $\langle H_{BP} \rangle = 0,6546$ y $\langle H_{BPW} \rangle = 0,6313$ a $B = 16$, pero se estabilizan desde $B = 24$. Las complejidades son menores que para LOG, $\langle C_{BP} \rangle = 0,4580$ y $\langle C_{BPW} \rangle = 0,4578$, estos valores se alcanzan para $B \geq 15$ pero se estabilizan en $B \geq 23$. Comparado con LOG, las propiedades estadísticas son mejores con menos cantidad de bits, para $B \geq 24$ este mapa alcanza mejores características en el sentido de generador aleatorio.

Además, se encontró una condición inicial con un comportamiento anómalo en doble precisión de coma flotante. Las Figuras 5.11a, 5.11b y 5.11d muestran una línea discontinua azul horizontal que está lejos del valor promedio, esto no es detectado por los cuantificadores basados en el procedimiento con contribuciones de amplitud BPW de las Figuras 5.11c y 5.11e. Sin embargo, al comparar ambos procedimientos (BP y BPW) pudimos detectar una caída a un punto fijo después de un transitorio largo, el procedimiento BPW descarta los valores constantes (que corresponden a un punto fijo) y sólo calcula sobre los valores transitorios.

El plano de doble entropía $H_{hist} \times H_{BP}$ se muestra en la Figura 5.12. El punto alcanzado en este plano para el mapa SWITCH es similar al alcanzado para el mapa LOG, y se indica

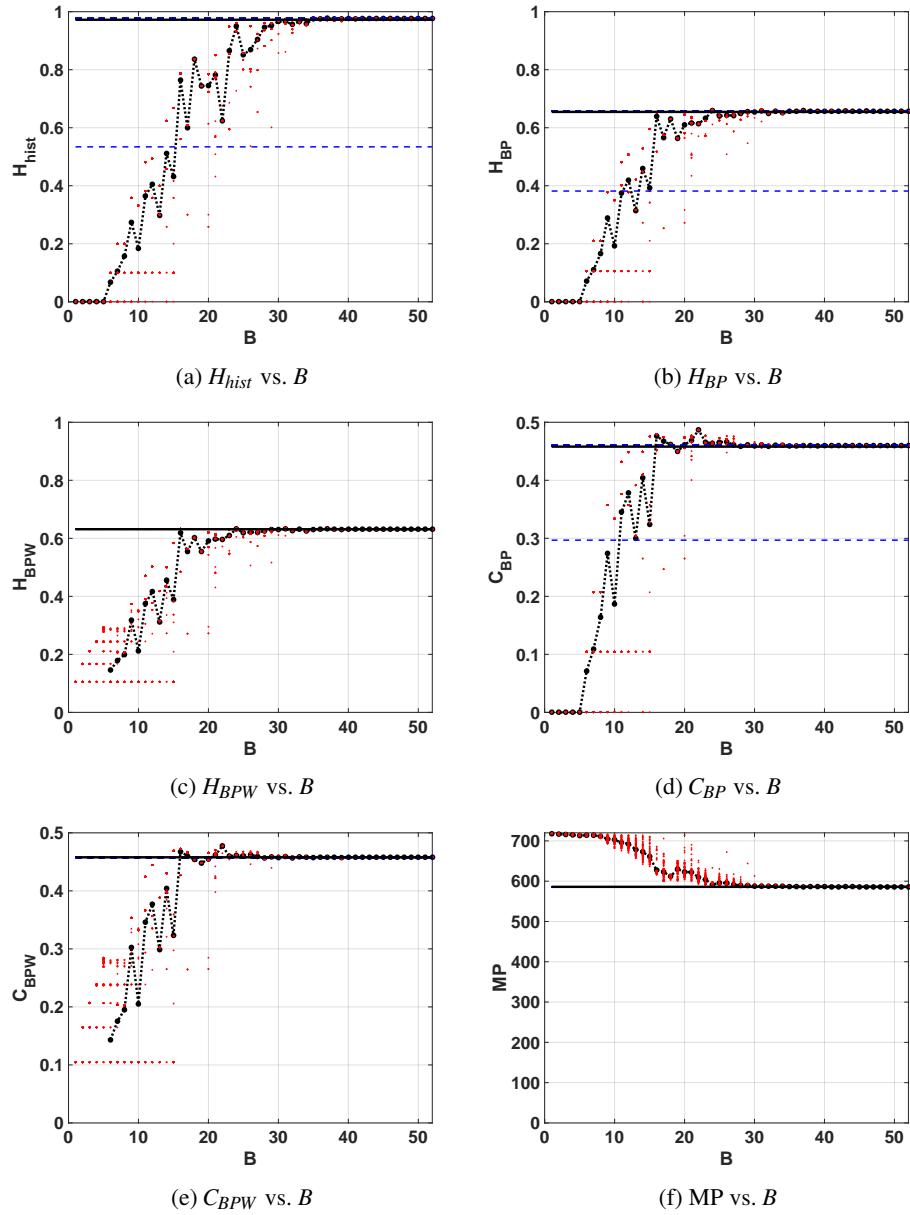


Figura 5.11: Propiedades estadísticas del mapa SWITCH

con una estrella en la Figura. La mezcla es levemente mejor en este caso.

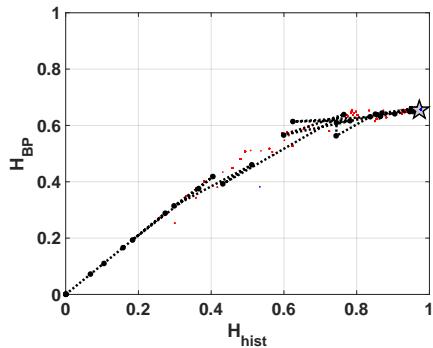


Figura 5.12: Evolución de las propiedades estadísticas en el plano doble entropía para el mapa SWITCH $H_{hist} \times H_{BP}$.

El plano de entropía - complejidad $H_{BP} \times C_{BP}$ se muestra en la Figura 5.13. Si comparamos con el mismo plano en el caso de LOG (Figura 5.5a), C_{BP} es menor para SWITCH, este hecho muestra un comportamiento más aleatorio.

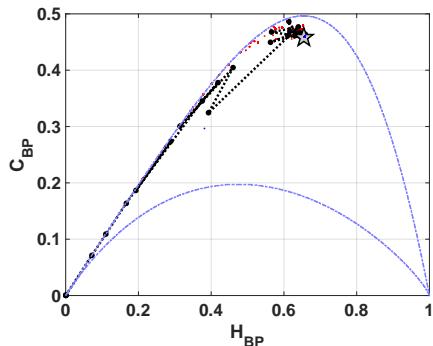


Figura 5.13: Evolución de las propiedades estadísticas en el plano entropía-complejidad para el mapa SWITCH $H_{BP} \times C_{BP}$.

EVEN y ODD

En las Figuras 5.14a y 5.15a se puede ver que los cuantificadores relacionados con el histograma de valores normalizado se degradan ligeramente con el procedimiento skipping. Por ejemplo, $\langle H_{hist} \rangle$ se reduce de 0,9722 sin skipping a 0,9459 para EVEN y 0,9706 para ODD. Esta diferencia entre los cuantificadores de las secuencias generadas por EVEN y ODD en coma flotante se debe a que se obtuvo una alta dispersión para H_{hist} , H_{BP} y C_{BP} pero no para H_{BPW} o C_{BPW} .

Las Figuras 5.14b a 5.14f y las Figuras 5.15b a 5.15f muestran los resultados de los cuantificadores BP y BPW para EVEN y ODD, respectivamente. Se requiere una mayor precisión para lograr una complejidad menor, a diferencia de los casos sin skipping que convergen a valores altos. Desde el punto de vista de MP, se obtiene una gran mejora utilizando cualquiera de las estrategias de post-procesamiento, pero el ODD es ligeramente mejor que EVEN. Los patrones faltantes se reducen a $MP = 118$ para EVEN y ODD, lo que aumenta la entropía Bandt & Pompe máxima permitida que alcanza el valor medio $\langle H_{BP} \rangle = 0,8381$ para EVEN, y $\langle H_{BP} \rangle = 0,9094$. La complejidad se reduce a $\langle C_{BP} \rangle = 0,224$ para EVEN y $\langle C_{BP} \rangle = 0,282$ para ODD. El número mínimo de bits para converger a este valor es de $B > 40$ para los mapas EVEN y ODD.

La mejora mostrada en las Figuras 5.14 y 5.15 se refleja en la posición del punto asintótico en los planos 5.16, y 5.17. En ambos casos, esta posición es la más cercana al punto ideal $(H_{hist}, H_{BP}) = (1, 1)$, porque las secuencias resultantes presentan una mejor mezcla.

Los resultados que se muestran en las Figuras 5.18 y 5.19 son compatibles, la posición del punto asintótico es más cercana al punto ideal $(H_{hist}, H_{BP}) = (1, 0)$. Este resultado refleja que la mezcla es mejor porque la complejidad del sistema resultante es menor. Este plano detecta que en las secuencias generadas por skipping, la mezcla de ODD es levemente mejor que EVEN.

5.4. Conclusiones

Se exploró la degradación estadística debido al error inherente de los sistemas en base dos para los mapas caóticos simples, conmutados y con skipping. Se evaluaron las distribuciones de mezcla y amplitud desde un punto de vista estadístico.

Este trabajo complementa los resultados anteriores dados en [113], donde se investigaron las longitudes de los períodos. En ese sentido, los resultados obtenidos aquí fueron compatibles. Es posible observar que la conmutación entre dos mapas aumenta la dependencia del período en función de la precisión, esto se debe a que la longitud de correlación también se incrementa. Sin embargo, el procedimiento estándar de skipping reduce la duración del período a casi la mitad.

Todas las estadísticas de los mapas representados en punto fijo producen una evolución no monótona hacia los resultados de punto flotante. Este resultado es relevante ya que

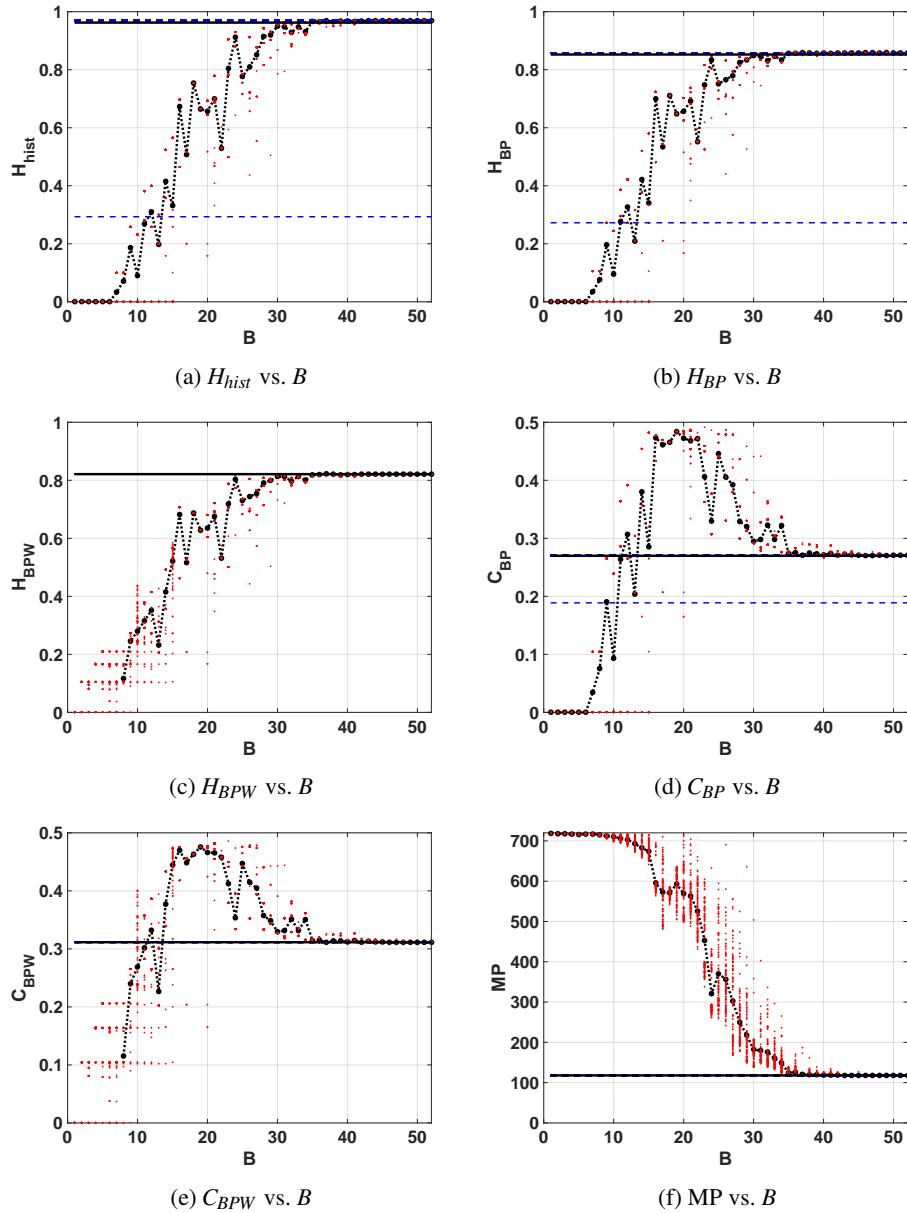


Figura 5.14: Propiedades estadísticas para el mapa EVEN

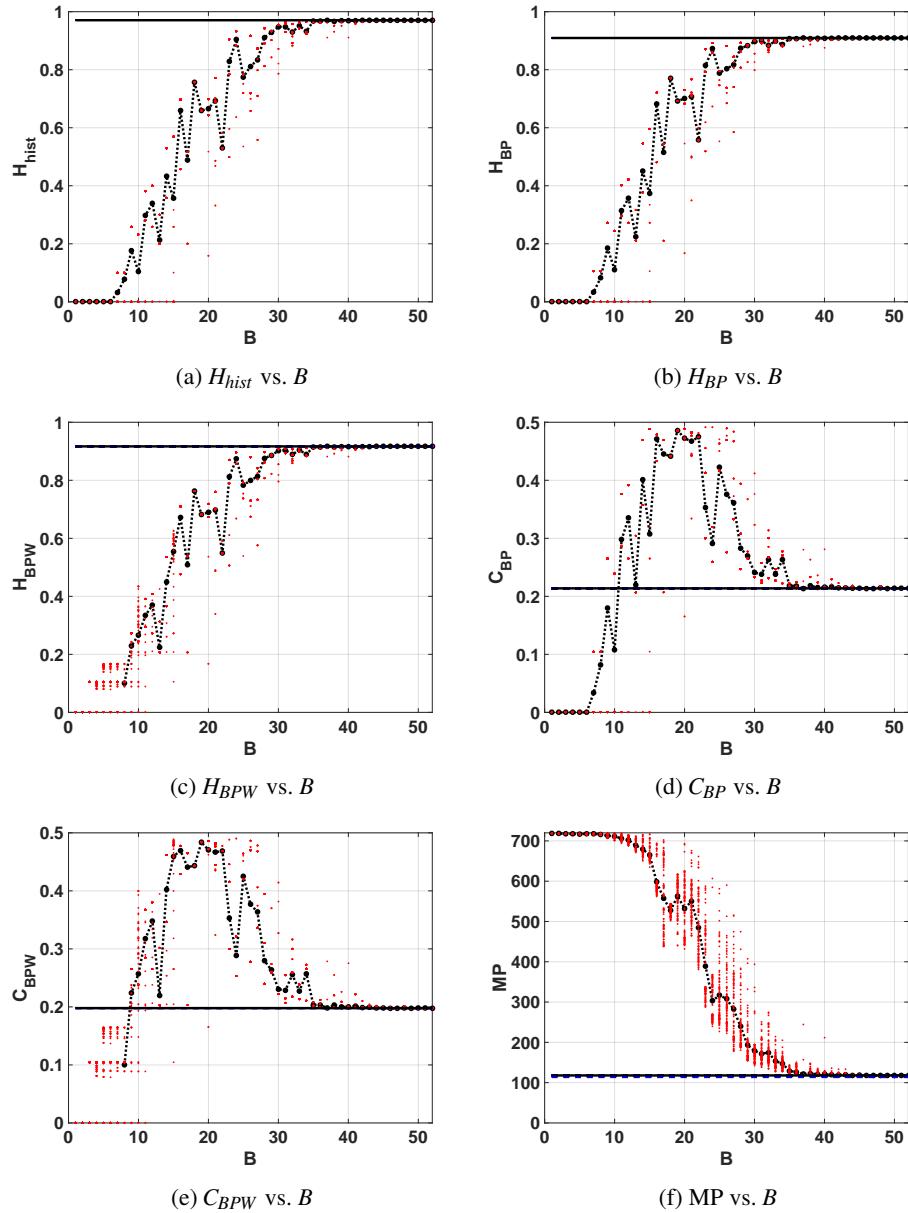


Figura 5.15: Propiedades estadísticas para el mapa ODD

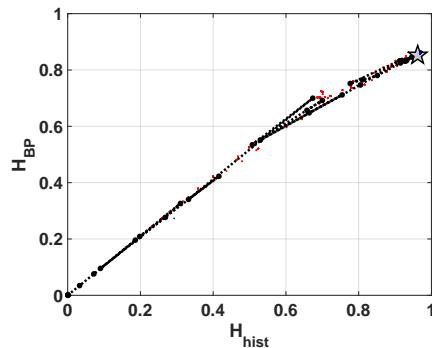


Figura 5.16: Evolución de las propiedades estadísticas en el plano doble entropía para el mapa EVEN $H_{hist} \times H_{BP}$.

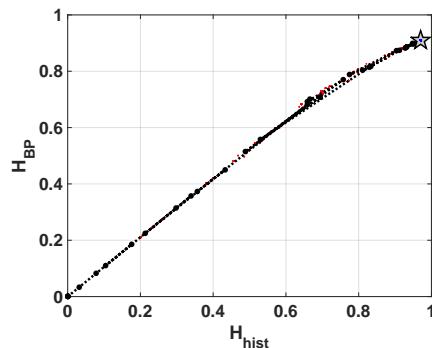


Figura 5.17: Evolución de las propiedades estadísticas en el plano doble entropía para el mapa ODD $H_{hist} \times H_{BP}$.

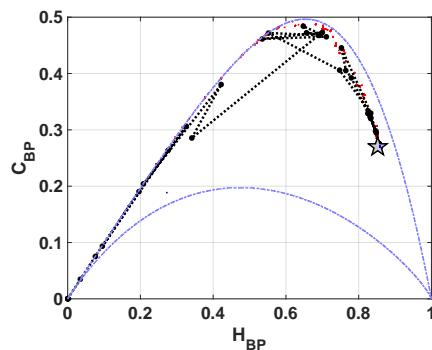


Figura 5.18: Evolución de las propiedades estadísticas en el plano entropía - complejidad para el mapa EVEN $H_{BP} \times C_{BP}$.

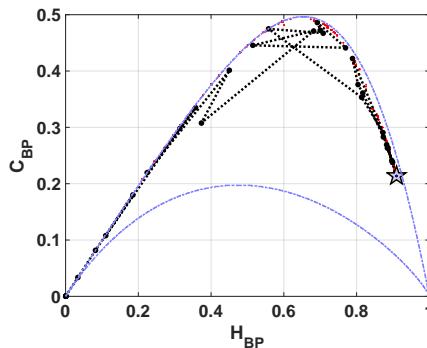


Figura 5.19: Evolución de las propiedades estadísticas en el plano entropía - complejidad para el mapa ODD $H_{BP} \times C_{BP}$.

muestra que no siempre es conveniente aumentar la precisión empleada.

Es especialmente interesante observar que algunos sistemas (TENT) con muy buenas propiedades estadísticas en el mundo de los números reales, se vuelven “patológicos” cuando se usan representaciones numéricas binarias. Como regla general, si un mapa se genera sólo por operaciones de desplazamiento (esto depende de la base de la unidad lógica aritmética y del mapa en sí), todas las condiciones iniciales convergerán a un punto fijo con un transitorio no mayor que la longitud de la mantisa que está siendo utilizada.

Al comparar los cuantificadores BP y BPW, se pudo detectar caídas a puntos fijos y se pudo estimar la longitud relativa de sus transitorios. Esto puede verse en todas las implementaciones del mapa TENT y en una condición inicial de los mapas SWITCH y EVEN para la implementación en punto flotante.

En relación con el comportamiento estadístico, los resultados obtenidos muestran que el mapa SWITCH es levemente mejor en la mezcla con respecto al mapa LOG (y TENT, por supuesto). Sin embargo, la mayor mejora se obtiene cuando se aplica el procedimiento de skipping, es posible ver que las entropías de BP y BPW crecen y las complejidades de BP y BPW disminuyen, para una dada representación numérica. Este resultado es relevante ya que evidencia de que un período largo no es sinónimo de buenas propiedades estadísticas, los mapas con skipping EVEN y ODD tienen longitudes de período de la mitad que los de SWITCH, pero su mezcla es mejor y sus distribuciones de amplitud se mantienen casi iguales. Como contrapartida, se necesita más precisión para alcanzar estas mejoras que ofrece el método de skipping.

Resultó muy interesante el hecho de que el mapa TENT con $u = 2$ (el cual produce

salidas que convergen rápidamente a cero) y $u = 1,96$ (con propiedades estadísticas mejores que LOG) produzcan salidas con los mismos resultados en el esquema switcheado.

Capítulo 6

Generadores de TRNG usando ROs en FPGA

6.1. Introducción

El *Jitter* es cualquier desviación leve del período medio de una señal presuntamente periódica. Hay muchos ejemplos físicos donde esta inestabilidad es relevante. Algunos ejemplos de diferentes áreas son: (a) Stalberg *et al.* [118] encontraron que el intervalo de tiempo entre los dos potenciales de acción de dos fibras musculares, que pertenecen a la misma unidad motora en los músculos humanos normales, muestra una variabilidad o inestabilidad; (b) Mecozzi *et al.* [119] detectaron jitter temporal y variaciones de amplitud en enlaces ópticos utilizando transmisión de pulso altamente disperso; (c) Derickson *et al.* [120] realizaron una comparación completa de la fluctuación de tiempo en el caso de los láseres semiconductores en modo bloqueado; (d) el California y Carnegie Planet Search en el Observatorio Keck [121] informó la inestabilidad de las estrellas en las velocidades radiales; (e) Roberts & Guillemin estudiaron los retardos debidos a las colas en etapas de *upstream multiplexing*, en una red de Modo de transferencia asíncrono (ATM); (f) Baron *et al.* [11] consideraron la calidad de la señal del *bunch clock* del *Large Hadron Collider* (LHC), en términos de inestabilidad, un problema fundamental porque sincroniza todos los sistemas electrónicos en el detector; (g) Marsalek *et al.* analizaron la relación entre la entrada sináptica y la fluctuación de fase de salida pico en neuronas individuales [122], etc.

Además, los instrumentos digitales se utilizan en cualquier experimento moderno y la

inestabilidad inevitable en los sistemas de adquisición de datos produce incertidumbres en el tiempo y, por consiguiente, en cualquier determinación del espectro.

En esta aplicación particular de los ROs, el *jitter* no siempre es indeseable. El *jitter* no es deseado en aplicaciones que usan un RO como generador de reloj [123, 124, 125, 126, 127]. Por el contrario, los generadores de números aleatorios RNG basados en *ROs*, usan el *jitter* como fuente de aleatoriedad, [12, 128]. El *jitter* también puede ser utilizado para mejorar la compatibilidad electromagnética (EMC) en un circuito digital para distribuir la frecuencia del reloj sobre una banda [61].

La determinación del *jitter* de fase en ROs se ha estudiado en varios artículos: en [129] se presentó el estudio de tres medidas relevantes del *jitter* en el dominio del tiempo. En [130] se propuso un modelo para la generación y distribución del *jitter* en ROs. En este artículo, los autores separan las fuentes de inestabilidad en deterministas y aleatorias (gaussianas); además, cada fuente se clasifica adicionalmente en local o global. Demuestran que las contribuciones más importantes son la inestabilidad gaussiana local y la inestabilidad determinística global y sólo la primera debe usarse como una fuente de aleatoriedad de generadores de TRNG. El mismo enfoque se usó en [131, 132, 133, 134]. En [135] Lubicz *et al.* describen un método práctico y eficiente para estimar la tasa de entropía de un *TRNG* basado en osciladores libres; enfatizaron que su método no requiere extraer las señales del dispositivo y analizarlas con equipos externos (una metodología que introduce fluctuación y distorsión extra en la señal medida debido a la cadena de adquisición de datos).

Por lo general, *jitter determinista* es el nombre que se le da a cualquier *jitter no gaussiano*. Este *jitter* está limitado y se caracteriza por su valor máximo de Δ_{pp} . *Jitter aleatorio* es el nombre utilizado para la *jitter gaussiano* y se caracteriza por su valor RMS. También frecuentemente las señales presentan *jitter* periódico determinístico, en este caso el tiempo entre flancos consecutivos se incrementa y decrementa a intervalos regulares. Estos intervalos entre dos tiempos el efecto máximo son el período del *jitter*; el inverso de este período es la frecuencia del *jitter*. El *jitter* periódico con frecuencia de fluctuación inferior a 10Hz usualmente se denomina *wander* y el nombre *jitter* se refiere sólo a las fluctuaciones periódicas con frecuencias en o por encima de 10Hz. En comunicaciones, *jitter total* es $T = \Delta_{pp} + 2nR_{rms}$ donde n es un número entre 6 y 8 relacionado con la tasa de error binario (*BER*).

Los *ROs* son uno de los principales componentes de los circuitos integrados analógicos y digitales y se han utilizado ampliamente como osciladores *on-chip* para generar relojes

en circuitos de alta velocidad. Además, los *ROs* se pueden implementar fácilmente en circuitos digitales programables como *FPGAs*. Las principales ventajas de los osciladores integrados *RO* sobre los clásicos circuitos resonantes Bobina-Capacitor son su área de chip más pequeña, su rango de funcionamiento más amplio (que puede ser sintonizado eléctricamente) y su menor consumo de energía.

Ya sea que se quiera usarlo o eliminarlo, el *jitter* en *ROs* debe medirse, lo que no es una tarea simple. La principal contribución de este trabajo es proporcionar una técnica de medición del *jitter* basada en cuantificadores de la teoría de la información (*ITQ*). Se utilizó un modelo estocástico cuya aleatoriedad está relacionada con la amplitud de la inestabilidad. Cada *ITQ* propuesto utilizado en este trabajo se basa en una entropía, es decir, una función de Shannon de la *PDF* asignada a la serie de tiempo del proceso estocástico. También se pueden usar desequilibrios y complejidades [70, 56], pero no representan una mejora en este caso. En este caso utilizamos dos opciones para *PDF*: el *histograma normalizado* y el *histograma de patrones de orden*. Se usa un plano de representación para comparar diferentes situaciones. Una vez que se elige la *PDF*, la Entropía de Shannon es la función básica que cuantifica la uniformidad de la *PDF*. Las *entropías normalizadas, entropías diferenciales y tasa entropía* son las otras *ITQs* evaluadas. En este caso las *entropías diferenciales* obtienen los mejores resultados y se utiliza un *plano de entropías diferenciales* para comparar su sensibilidad como medida de *jitter*.

En este Capítulo se muestran los resultados de dos publicaciones. Primero, en 6.2 se presentan los datos publicados en [14], en donde se propone utilizar *ITQs* para medir la amplitud de *jitter* y constituye el principal aporte de este capítulo. Luego, en 6.3 se muestran los aportes publicados en [21], en donde se hace uso de *ROs* como bloques de base para construir un *TRNG*.

6.2. Determinación del *jitter* en *ROs*

Hay dos situaciones diferentes en lo que concierne al *jitter* en *ROs*: (a) en algunas aplicaciones es suficiente con asegurar que el *jitter* no perturba a la señal por encima de un límite aceptable. En este caso la señal se observa en un osciloscopio con una máscara sobre la pantalla, lo que es suficiente para verificar que la señal se mantiene dentro de los márgenes de tolerancia; (b) en otros casos se precisa una determinación exacta del *jitter*. Entre esos casos está la caracterización de *ROs* considerada en este trabajo.

Los *ROs* ideales están compuestos por un número impar de inversores. Cada inversor tiene un tiempo de propagación y por lo tanto los flancos de subida y bajada separados por medio período viajan a través de los inversores. Si todos los tiempos de propagación son constantes, la salida de este *RO* ideal es una señal cuadrada con un espectro de frecuencia discreto. Pero como los tiempos de propagación no son constantes, el *jitter* distorsiona el espectro de potencia ensanchando cada delta.

Supongamos que $T/2$ es el tiempo de medio período de un *RO* ideal. Entonces está dado por:

$$\frac{T}{2} = k \sum_{i=1}^k d_i \quad (6.1)$$

En donde k es el número de inversores y d_i es el tiempo de propagación a través del i -ésimo inversor. Cuando hay *jitter*, d_i es una variable aleatoria que modelamos como:

$$d_i = D_i + \Delta d_i \quad (6.2)$$

donde D_i es el valor medio de d_i con el nivel nominal de voltaje de fuente y la temperatura normal, y Δd_i es la variación del retardo producida por los eventos físicos locales y los cambios globales en las condiciones de trabajo del dispositivo (como V_{CC} , temperatura , etc.). Entonces, el *jitter* en *ROs* se evidencia por el desplazamiento aleatorio de la ubicación de los flancos ascendentes y descendentes, con respecto a la ubicación perfectamente periódica. La medición directa de este desplazamiento tiene dos problemas principales: (a) requiere un instrumento de muy alta frecuencia, porque la resolución del tiempo está limitada por el período de muestreo T_s ; (b) esta técnica introduce fluctuaciones y distorsiones adicionales en la señal medida proveniente de la cadena de adquisición de datos. Entonces, para medir el *jitter* con una perturbación mínima, es más conveniente usar *medidas indirectas* por medio de variables aleatorias auxiliares relacionadas con las propiedades estadísticas involucradas en el *jitter* [135]. El procedimiento general es el siguiente:

1. Se muestrea la salida con el período de muestreo T_s para obtener una serie de tiempo binaria. En el caso ideal de *no-jitter*, la salida es una *onda cuadrada continua y perfectamente periódica* con un período T . Entonces es posible ajustar T_s para hacer $T/2 = mT_s$ con $m \in N^+$. La serie de tiempo binaria será periódica con m unos seguidos de m ceros. Cuando el *jitter* está presente, la serie binaria no es periódica sino

estocástica. Este modelo estocástico se conoce como *proceso de renovación alterna*.

2. Se pueden usar muchos cuantificadores de aleatoriedad diferentes para caracterizar el modelo estocástico asociado con el *jitter* medido. En este trabajo se utilizaron cuantificadores de la teoría de la información.

Tengamos en cuenta que el *jitter* es acumulativo y surgen dos situaciones básicas: (a) si se supone que el *jitter* introducido por cada etapa es totalmente independiente del *jitter* introducido por otras etapas, significa que $\sigma_T^2 = m\sigma_s^2$, en donde σ_s es el *jitter* de cada muestra, y se supone que todas las muestras tienen fluctuaciones con la misma distribución normal; (b) si las fuentes de *jitter* están totalmente correlacionadas entre sí, entonces $\sigma_T = m\sigma_s$.

6.2.1. Resultados

Se simuló con Matlab una salida de un *RO* muestreada uniformemente sin *jitter* y se generó un archivo de salida con una longitud de $N_b = 7\,000\,000$ de bits. Se exploró un conjunto de cien valores con relación de muestreo $r = T_s/T \in [6,5;9,5]$ (donde T_s es el período de muestreo y T es el período de salida *RO*). Se agregó *Jitter* con una distribución normal en las secuencias generadas con diferentes valores de varianza σ_s (ver a continuación) y se generaron nuevos archivos de igual longitud. El método propuesto emula el verdadero proceso de muestreo de la salida ruidosa de un *RO* real; el código detallado se publicó en Mathworks [136].

Para cada valor de σ_s , se generaron diez surrogados (cada uno con una condición inicial aleatoria diferente) y se almacenaron nuevos archivos con N_b bits cada uno. Se asumió que el *jitter* de las muestras individuales es independiente, con variables aleatorias distribuidas normales y un valor medio cero y varianza $\sigma_i = \sigma_s$. En consecuencia, la varianza del *jitter* acumulada durante un período de T está dada por $\sigma_T^2 = r\sigma_s^2$ [130]. Los valores considerados son $\sigma_T = \{0, 0,001, 0,002, 0,003, 0,004, 0,005, 0,007, 0,01, 0,02, 0,02, 0,04, 0,05, 0,07, 0,1\}$.

Por cada archivo se evaluaron todos los cuantificadores definidos en la Sección 3 para $D \in [2, 10]$ y $W \in [1, 26]$. Los detalles sobre la evaluación, las ventajas y los inconvenientes de cada cuantificador se informan en la Sección 3: ellos son S_W , $S_{BP}^{(D)}$, H_W , $H_{BP}^{(D)}$, h y h^* . Aquí mostraremos sólo los resultados más relevantes para justificar la elección los dos últimos cuantificadores (h y h^*).

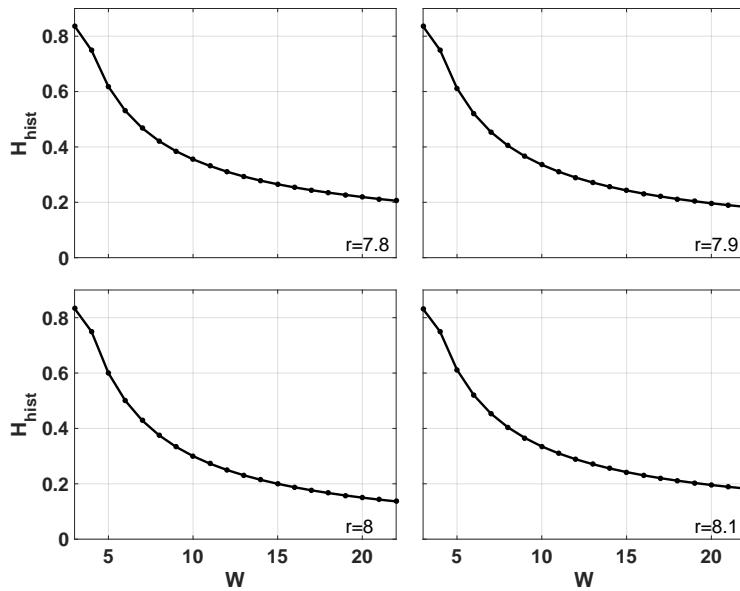


Figura 6.1: Entropía normalizada H_W en función de W para un *RO* sin *jitter* muestreado con diferentes valores de r .

- En el caso de entropía normalizada H_W , se observó que tiene una fuerte dependencia con el W empleado. Además, el análisis de H_W en función de r muestra que este cuantificador no permite determinar un valor óptimo de la relación de muestreo r (ver Figura 6.1). Este es un problema no trivial para el caso en el que se trabaje con configuraciones experimentales.
- En el caso de la entropía de Bandt & Pompe normalizada $H_{BP}^{(D)}$, también está presente una fuerte dependencia con la dimensión de embedding D empleada. Nuevamente, no es fácil determinar el valor óptimo de r del análisis de este parámetro en función de r (ver Figura 6.2).
- Un comportamiento similar aparece en todos los otros funcionales relacionados con estas dos entropías. En resumen, estos resultados muestran que tanto h y h^* son independientes de cualquier parámetro arbitrario utilizado para su cálculo.

Estos resultados muestran que los dos cuantificadores, h y h^* , son apropiados para ser usados como medidores de *jitter* debido a que:

- (a) Para $\sigma_T = 0$ (salida sin *jitter*) se acercan rápidamente a un valor límite constante ya que tanto D como W tienden a ∞ y este valor es independiente de D y W ;

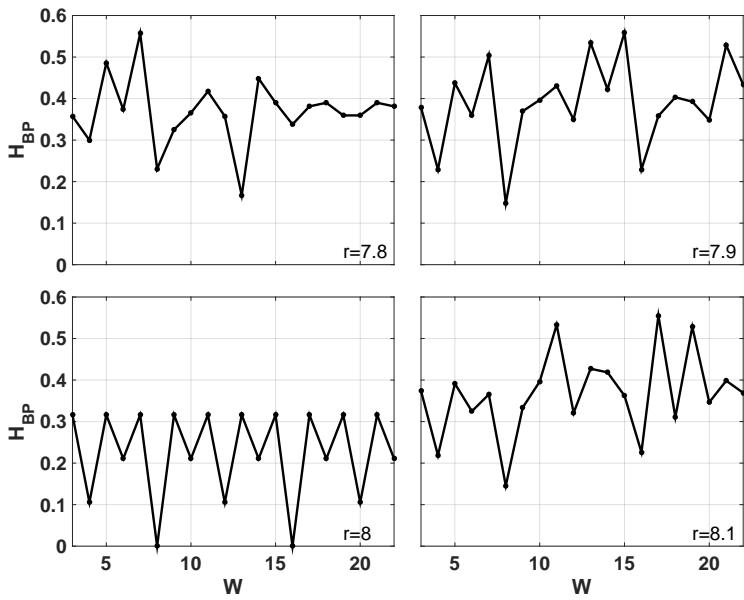


Figura 6.2: $H_{BP}^{(D)}$ en función de W para un RO sin jitter muestreado con diferentes valores de r . Los cálculos fueron hechos con superposición de palabras

- (b) Son funciones monótonas y proporcionales de σ_T .
- (c) A partir de su análisis, es posible detectar el valor óptimo de la relación de muestreo r . En las siguientes Figuras se muestran estas afirmaciones que son representativas de todos los resultados presentados.

La Figura 6.3 muestra la entropía diferencial de Bandt & Pompe h^* , como función de D , con W como parámetro, para un RO sin jitter. Se puede ver que existe un valor umbral $W = 4$ sobre el cual todas las curvas colapsan independientemente del valor de D . Además, la Figura 6.3 también muestra que para $D \geq 8$ todas las curvas colapsan, independientemente del valor de W . En conclusión, si $D \geq 8$ y $W \geq 4$ se obtiene un cuantificador independiente de D y W .

La influencia del jitter en este cuantificador se muestra en la Figura 6.4, donde h^* se representa como una función de D con σ_T como parámetro. Los valores considerados son $\sigma_T = \{0(\sin \text{ jitter}), 0,001, 0,002, 0,003, 0,004, 0,005, 0,007, 0,01, 0,02, 0,02, 0,04, 0,05, 0,07, 0,1\}$. El recuadro de la Figura 6.4 muestra h^* como función de σ_T para $D = 8$. Este recuadro muestra que h^* es una función monótona creciente de σ_T .

Finalmente, la Figura 6.5 muestra h^* como una función de la relación de muestreo r . En esta Figura, se muestra que hay un mínimo para el r correcto (en este caso $r = 8$). Además,

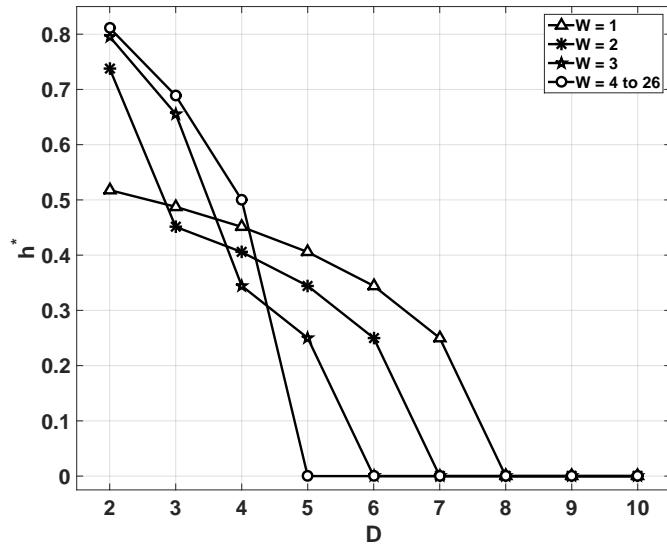


Figura 6.3: h^* en función de D para un RO sin *jitter* muestreado con $r = 8$.

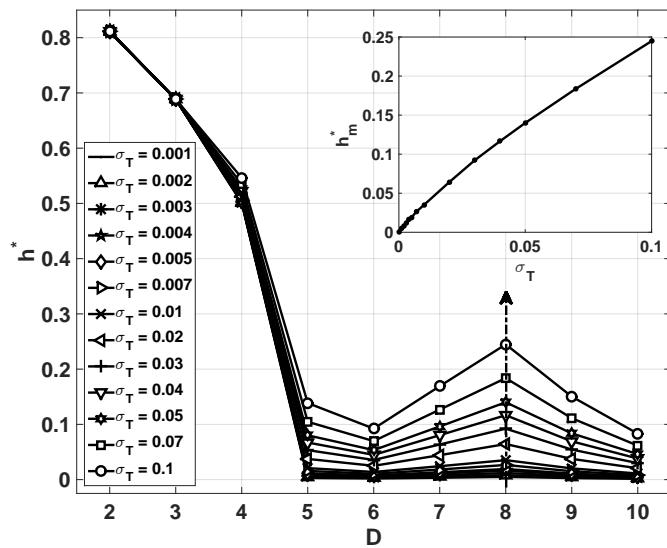


Figura 6.4: h^* en función de D para un RO muestreado con $r = 8$ con longitud de palabra $W = 6$ para *jitter* con diferentes varianzas. El recuadro muestra h^* en función de σ_T para $r = 8$, $W = 6$ y $D = 8$.

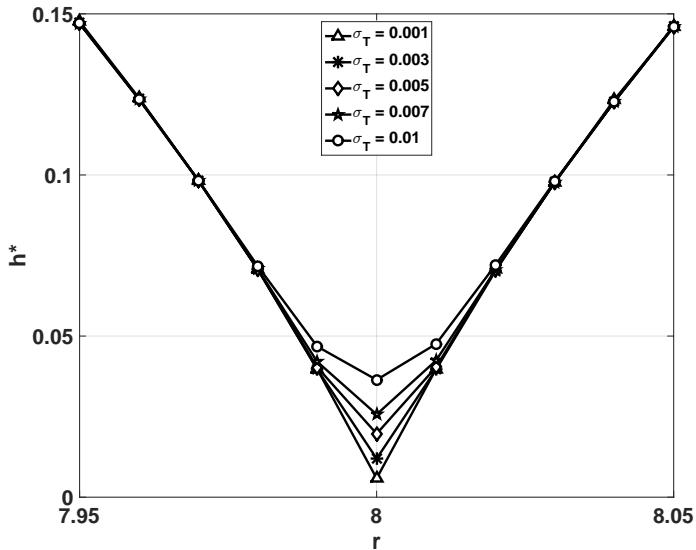


Figura 6.5: h^* en función de r para $r \in [7.95, 8.05]$, con algunos σ_T , $W = 6$ y $D = 8$. La curva tiene un mínimo en el valor correcto de $r = 8$.

la sensibilidad de h^* en función del *jitter* es máxima para este mismo valor ideal de r .

En cuanto al segundo cuantificador h , se observa que sólo depende de W ya que el parámetro D no se usa para definir la *PDF* asignada a la serie de datos. La Figura 6.6 muestra un caso sin *jitter*, h es independiente de W para $W \geq 4$. Para lo siguiente se adoptó $W = 6$.

La Figura 6.7 muestra la influencia del *jitter* sobre este cuantificador. Queda claro en el recuadro de esta Figura que, para el valor seleccionado $W = 6$, h es una función monótona creciente de la varianza del *jitter* σ_T .

La Figura 6.8 muestra que h tiene un mínimo cuando r toma su valor óptimo ($r = 8$). Cabe destacar que este mínimo es robusto también en presencia de *jitter*.

Se debe realizar un análisis adicional para asegurar que al utilizar los valores seleccionados $W = 6$ y $D = 8$ se esté trabajando con una cantidad de datos suficiente para tener una buena estadística. De acuerdo con la Sección 3, para un alfabeto dado \mathcal{A} con m elementos, y un archivo simbólico dado de longitud n , se define el parámetro de calidad $\alpha = n/m$. La calidad es mejor a medida que α aumenta y para esta aplicación se acepta un valor mínimo $\alpha = 10$. Los valores seleccionados $W = 6$ y $D = 8$ proporcionan $\alpha_h \simeq 10^5$, $\alpha_{h^*} \simeq 175$ si las w_i palabras son tomadas con superposición. Si las palabras son tomadas con superposición, resulta $\alpha_{h^*} \simeq 29$. En todos los casos se trabajó con un valor $\alpha > 10$ como es requerido.

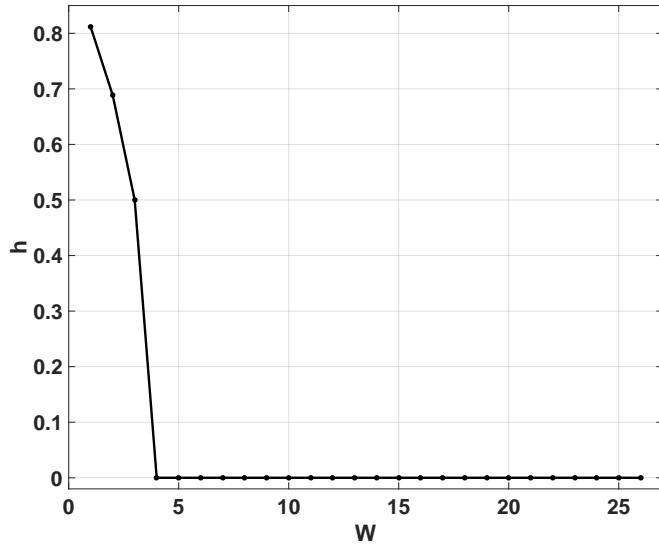


Figura 6.6: h en función de W para un RO sin *jitter* muestreado con $r = 8$.

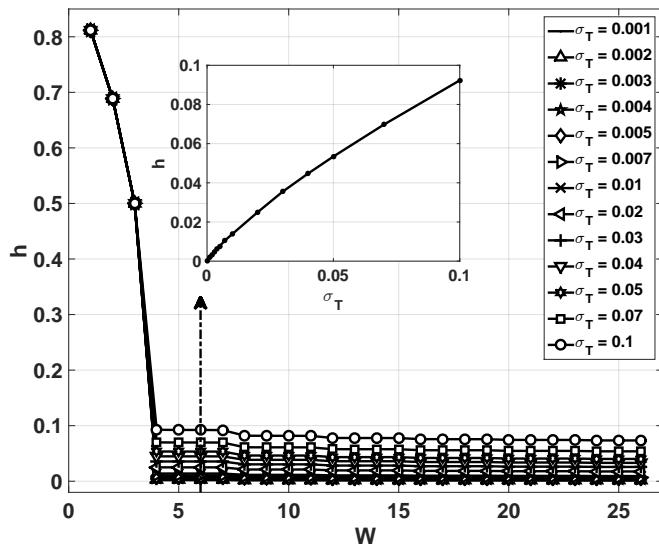


Figura 6.7: h en función de W para un RO muestreado con $r = 8$, con *jitter* con distintas varianzas. El recuadro muestra h en función de σ_T con $r = 8$ y $W = 6$.

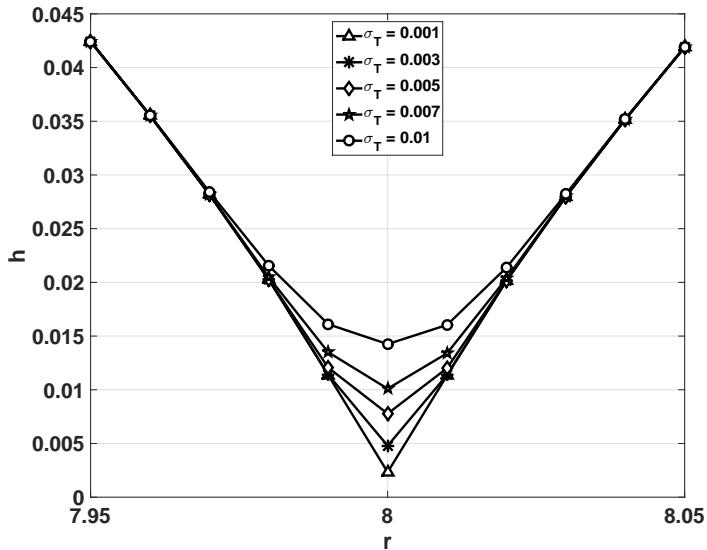


Figura 6.8: h en función de r con $r \in [7,95,8,05]$, para distintos σ_T y $W = 6$. La curva tiene un mínimo en $r = 8$.

La Figura 6.9 muestra el plano $h_m^* \times h$. Los cuantificadores se calcularon barriendo los valores de D de 2 a 11 y W de 2 a 26 (se barrió ambos para h^* y sólo W en el caso de h_{hist}). Se obtuvo una mejor diferenciación para valores más altos de los parámetros, esto se debe a que ambos cuantificadores tienden a cuantificar la entropía de la fuente cuando D y W tienden a infinito. Sin embargo, esto es imposible en la práctica real, la cantidad de datos disponibles limita los valores de los parámetros para lograr buenas estadísticas. Por lo tanto, se buscaron los valores mínimos (valor umbral) de los parámetros que distinguieran suficientemente bien el jitter.

Una comparación entre ambos cuantificadores se muestra en la Figura 6.9. Los marcadores corresponden a varianzas $\sigma_T = \{0, 0,001, 0,002, 0,003, 0,004, 0,005, 0,007, 0,01, 0,02, 0,03, 0,04, 0,05, 0,07, 0,1\}$. Hay que tener en cuenta que la pendiente de cualquiera de estas curvas es dh^*/dh y es igual al cociente entre las pendientes de curvas en los recuadros de las Figuras 6.4, y 6.7. Si $dh^*/dh \rightarrow 1$, h^* es más sensible que h para medir el jitter. La pendiente aumenta ligeramente de $\sim 2,47$ para $W = 5$ a $\sim 5,54$ para $W = 19$, esto muestra que h^* se vuelve más sensible a medida que aumenta W .

También se evaluó h^* sin la superposición de bits entre números naturales consecutivos, pero manteniendo la superposición de los $D - 1$ números naturales entre patrones de orden (en todos los casos h se evaluó con la superposición de $W - 1$ bits consecutivos). Los

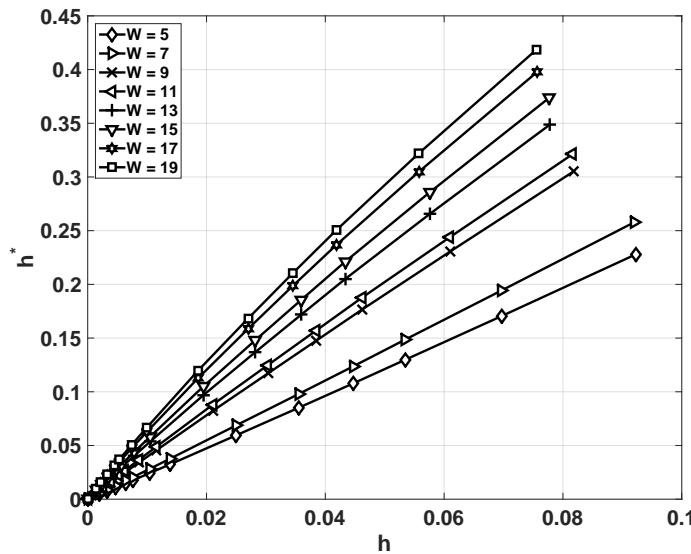


Figura 6.9: h^* como función de h para $r = 8$, $D = 8$ y diferentes valores de W .

resultados se representan en la Figura 6.10 donde se muestra que al eliminar la superposición aumenta la sensibilidad de este cuantificador. Por supuesto, se obtiene una cantidad menor de W bits en números naturales del archivo original de siete millones de datos binarios, y en consecuencia, la calidad estadística es menor que la del cálculo original con superposición. Para aumentar el valor de α y llegar al mínimo nivel requerido y asegurar una buena estadística, se requieren archivos binarios más largos.

En la Figura 6.11 se muestra el plano doble entropía diferencial con todas las curvas obtenidas para todas las longitudes de palabra W y de embedding D utilizados en esta Sección. Sobre el eje horizontal es posible separar cuatro grupos de curvas ($W = 1$, $W = 2$, $W = 3$ y $W \geq 4$). Sobre el eje vertical puede verse que a medida que D crece, las curvas tienen origen en valores cada vez menores. El grupo de curvas que se encuentran a la izquierda van del azul al rojo según el valor de W . El área inferior izquierda del plano es la que tiene un mejor rendimiento de ambos cuantificadores, sin embargo, es la que precisa un mayor esfuerzo de cómputo y una mejor estadística. Este detalle es fácil de ver en la Figura 6.12, allí se muestra cómo la sensibilidad al jitter aumenta cuando W aumenta.

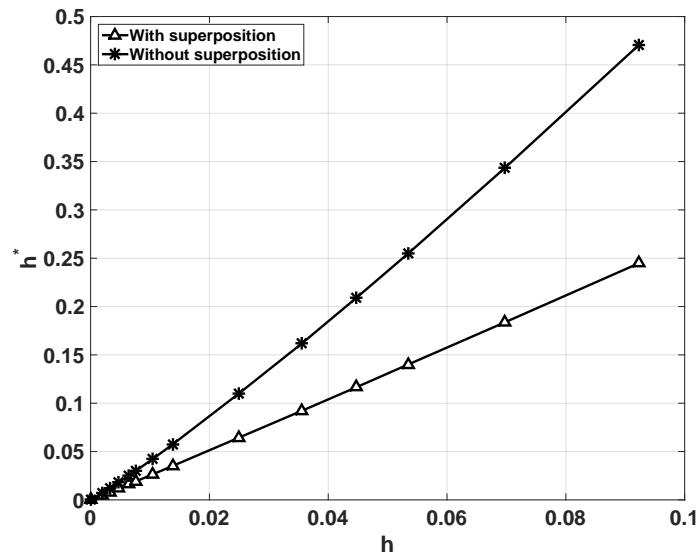


Figura 6.10: h^* en función de h para $r = 8$, $W = 6$ y $D = 8$. Son considerados los dos procedimientos para obtener números naturales de W -bits: con y sin superposición (ver texto).

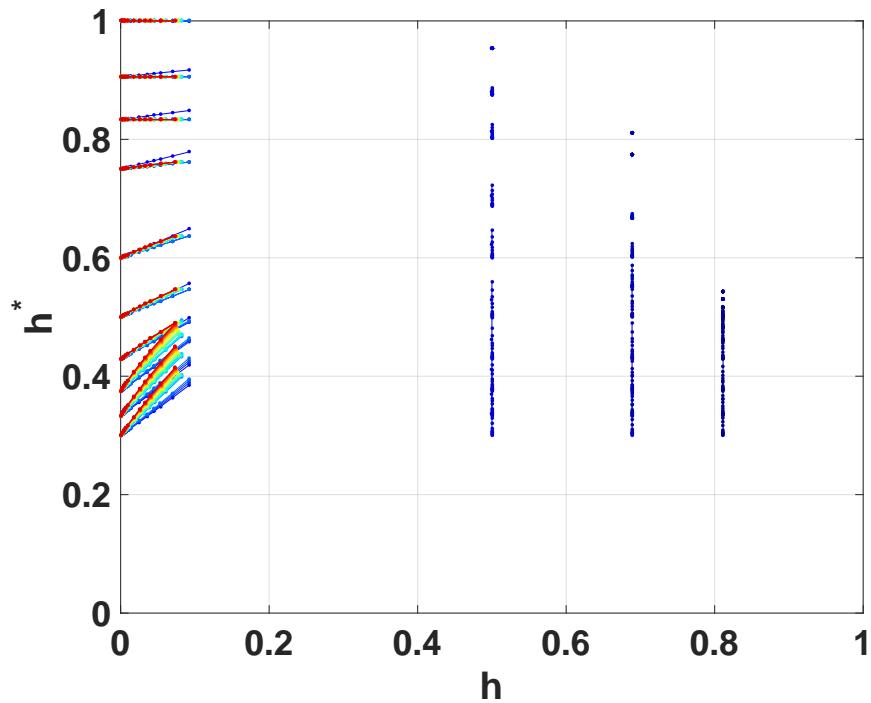


Figura 6.11: Plano doble entropía diferencial con D y W como parámetro.

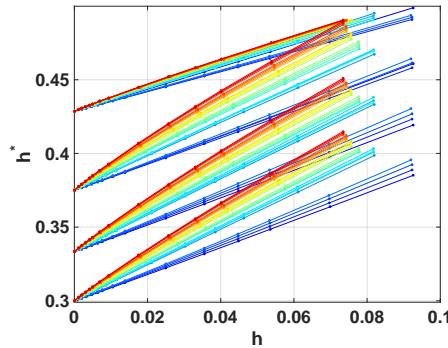


Figura 6.12: Detalle del plano doble entropía, en donde puede verse la sensibilidad como funcional de W y D .

6.3. Implementación de TRNG basado en ROs

En esta Sección se estudia el uso de los ROs como generadores de números aleatorios (TRNG). Se explica el diseño, hecho para ALTERA Cyclone III, usando primitivas de bajo nivel. Se consideran dos características relevantes de un RNG para validar el diseño: 1) la equiprobabilidad de todos los resultados posibles y 2) la estadística independencia de valores consecutivos. En este trabajo, estas propiedades se miden a través de Cuantificadores de teoría de la información descriptos en la Sección 3.2. Se utiliza el plano de doble entropía para representar las series de tiempo y visualizar fácilmente los resultados obtenidos con diferentes configuraciones. La calidad estadística también se compara con otros RNG disponibles por medio de este plano. Nuestro método constituye una reducción efectiva del análisis completo realizado con test estadísticos estándar como DIEHARD o NIST.

6.3.1. Introducción

Como se dijo en la Sección anterior, el jitter y los ruidos de fase presentes en los osciladores en anillo no son convenientes en muchas aplicaciones de ROs, por ejemplo en la implementación de osciladores en el chip para generar relojes en circuitos de alta velocidad [125, 126, 127]. Sin embargo, son la fuente de aleatoriedad para un TRNG basado en ROs [12, 128]. Además, un RO se puede implementar en un circuito totalmente digital como FPGAs ya que básicamente son solo una serie de inversores.

En [12], Sunar et al. presentaron un RNG usando *jitter* estocástico combinando varios ROs. Ellos requerían un procesamiento posterior del flujo de bits, para enmascarar imper-

fecciones en la fuente de entropía y para aumentar la inmunidad contra los cambios en las condiciones ambientales.

Wold et al. [128] propusieron una versión con mejores características aleatorias y que no requieren un procesamiento posterior. Ellos sólo agregaron un flip-flop D adicional en cada salida de anillo. La efectividad de su propuesta fue probada por medio de pruebas estadísticas disponibles en la literatura abierta [137, 6, 138].

En este Capítulo se realiza una descripción detallada de una implementación de hardware muy compacta de TRNGs basados en ROs propuesto en [128]. Para validar la aleatoriedad de las secuencias de ruido generadas, se emplearon dos cuantificadores derivados de la teoría de la información y el plano de doble entropía $H_{BP} \times H_{hist}$ propuestos en la Sección 3. Según lo explicado arriba, H_{hist} es una medida de la equiprobabilidad entre todos los valores posibles y H_{BP} es una medida de la independencia entre valores consecutivos.

6.3.2. Implementación en Hardware

Los TRNG implementados consisten en varios ROs con sus salidas XOR conectadas y muestreadas por un flip-flop *D*. El flip-flop latcea la salida a una frecuencia seleccionada (aquí 100 MHz) [128]. La implementación física se realizó en el kit de desarrollo EP3C120F780C7N FPGA cuyo dispositivo principal es una ALTERA Cyclone III EP3C120. El diseño está hecho con el software Quartus II 13.1.

Reseña del Chip

Las FPGAs consisten en una gran cantidad de bloques de matriz lógica (LABs), con grupos de elementos lógicos (LEs) para implementar circuitos tanto secuenciales como combinacionales. En la arquitectura de la familia Cyclone III cada LAB contiene 16 LEs. Básicamente, cada LE consiste en un flip-flop (FF) con una Look up table (LUT) de cuatro entradas (ver Figura 6.13). Cada LUT puede implementar cualquier función de cuatro variables. El FF y la LUT se pueden usar juntos o independientemente, [139].

Por lo general, el software de síntesis asigna recursos sin la intervención del diseñador. Pero en el diseño de TRNGs basados en ROs es necesario controlar la ubicación exacta de cada componente individual para evitar la simplificación de los inversores realizada por la herramienta de síntesis. En Altera el uso de primitivas de bajo nivel permite controlar la implementación. Por consiguiente, estas primitivas y asignaciones de bajo nivel se emplean

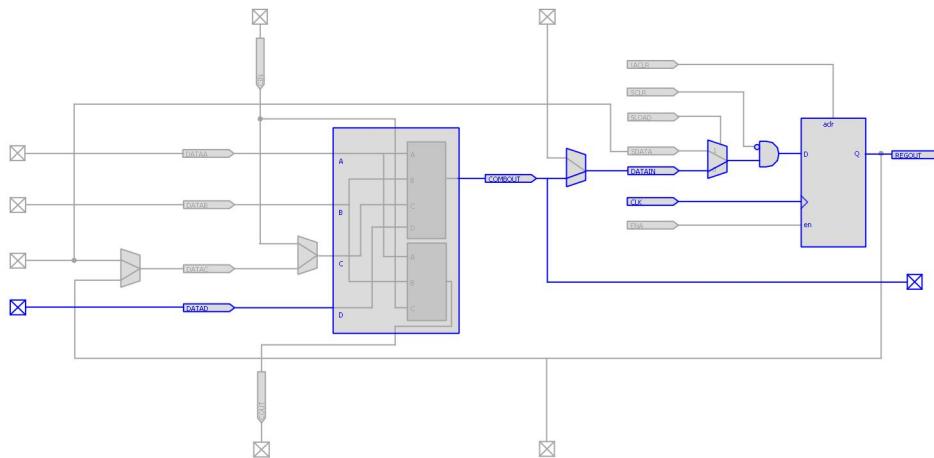


Figura 6.13: Imagen del Chip Planner que muestra la implementación de un inversor y un Flip Flop.

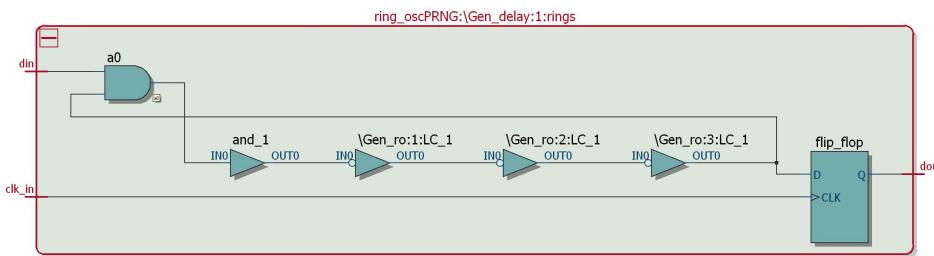


Figura 6.14: Vista RTL de un ring con 3 inversores.

dentro del código HDL empleado en el diseño desarrollado. Además, se debe configurar la herramienta de síntesis para evitar que elimine los buffers redundantes.

Las cadenas de ROs se pueden implementar en el chip programando las LUTs como inversores. Es necesario evitar que el motor de síntesis Quartus II fusione dos compuertas NOT en serie, utilizando una primitiva llamada LCELL. Una LCELL consume una celda lógica y no es eliminada del proyecto durante la síntesis lógica. Para crear un RO, se programan LCELLs como buffers de inversores. Las Figuras 6.14 y 6.15 muestran como esta primitiva es implementada por el compilador Quartus II.

Para lograr que cada RO tenga distintos comportamientos, cada uno debe ser ubicado en distintas posiciones del chip. Para esto se lo debe asignar a una región previamente definida (*LogicLock*).

La Figura 6.16 muestra las 50 regiones *LogicLocks* utilizadas en este trabajo. Se asigna

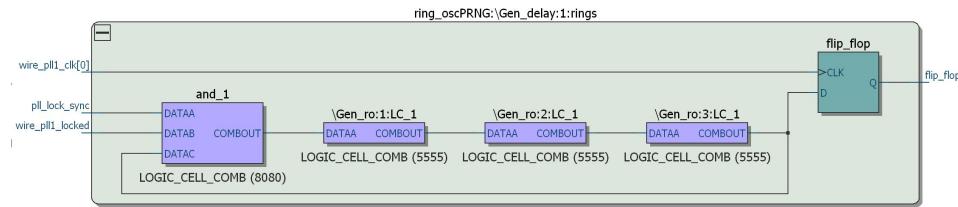


Figura 6.15: *Technology map viewer* (post mapeo) de un ring con 3 inversores.

un RO a cada región. Las regiones se distribuyen sobre el chip para un análisis futuro de la importancia de la ubicación. Cada región tiene 16 LABs, lo que permite aumentar el número de inversores de cada anillo.

Hay muchos factores que determinan la frecuencia de cada RO, y contribuyen a la imprevisibilidad de la salida:

1. Ubicación dentro de LAB: las diferentes ubicaciones entre los anillos pueden dar como resultado diferencias de tiempo.
2. Conexiones: incluso teniendo exactamente colocación idéntica de una LUT con respecto a otra en un anillo dado, no es posible tener exactamente el mismo uso de recursos de enrutamiento en las conexiones.
3. Selección de entrada: durante la etapa de enrutamiento, el *fitter* elegirá qué entrada de la LUT se utiliza. Como el retraso a través de la LUT depende de cuál de las cuatro entradas se utiliza, los anillos tienen diferentes retardos.
4. Vecindad: incluso si los ROs pudieran ser exactamente idénticos en los items anteriores, el retraso puede cambiar dependiendo de lo que se coloque y enrute alrededor del anillo.

En la Figura 6.17 (vista RTL) se muestra un TRNG usando 3 ROs seguido por una compuerta XOR.

Para tener una idea de los recursos empleados para este diseño, la Tabla 6.1 muestra el informe de compilación de un TRNG usando 15 ROs cada uno con 3 inversores.

6.3.3. Resultados

La herramienta *Emmbedded Logic Analyzer* se utilizó para recopilar las secuencias aleatorias generadas. Esta es una herramienta de depuración a nivel de sistema, proporcionada

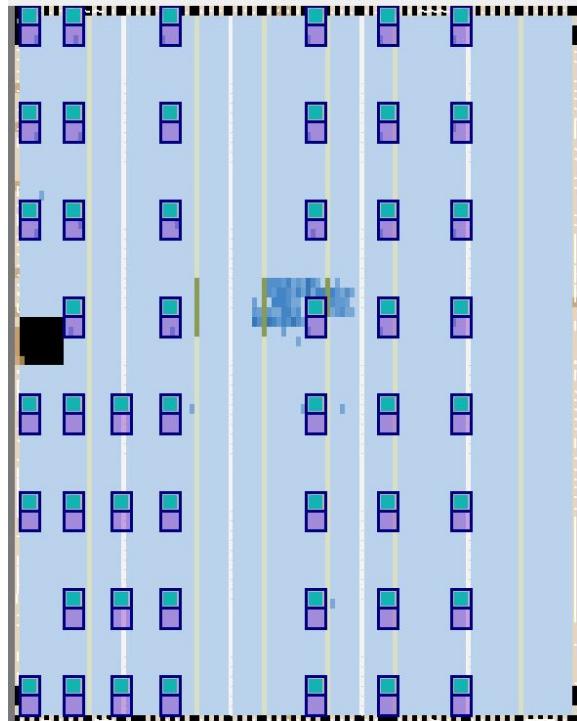


Figura 6.16: Vista de las regiones *LogicLock* del *Chip Planner*.

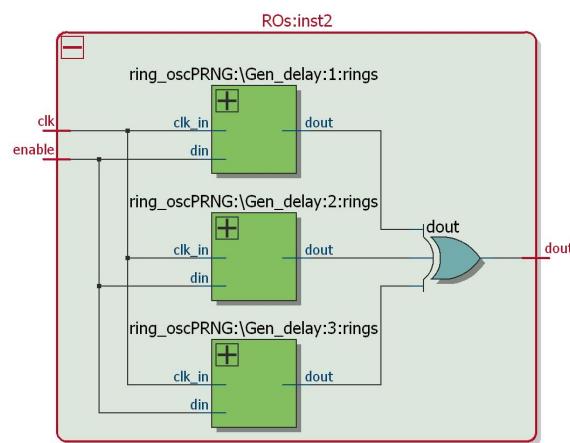


Figura 6.17: RTL de TRNG con 3 ROs.

Elementos lógicos totales	847/119,088	(< 1 %)
Funciones combinacionales totales	629/119,088	(< 1 %)
Registros lógicos dedicados	617/119,088	(< 1 %)
Registros totales	617	
Bits de memoria totales	131,072/3,981,312	(3 %)

Cuadro 6.1: Reporte de compilación de un RO basado en TRNG, utiliza 15 ROs de 3 inversores cada uno.

por *Altera* [105], que captura y almacena el comportamiento de la señal en tiempo real y permite observar las interacciones entre el hardware y el software en los diseños del sistema. Después de adquirir los datos y guardarlos en un archivo SignalTap II, pueden ser analizado o visto como una forma de onda. Este procedimiento no introduce *jitter* ni distorsión en la señal medida.

Se utilizaron archivos de datos con 917504bits cada uno para cada TRNG. Consideramos conjuntos de N_{RO} anillos, cada uno con 3 inversores; $N_{RO} = 2, 3, 4, 5, 6, 7, 15, 25$ y 50.

Los datos de SignalTap se procesaron usando Matlab. Se agruparon los datos en palabras de 6 bits sin superposición, por lo que se generaron archivos con 152917 datos cada uno. Se calcularon los cuantificadores descriptos en la Sección 3 para todos los archivos generados.

También se evaluaron otros generadores de ruido conocidos para comparar su calidad con la de los TRNG basados en RO. Los ruidos analizados fueron:

- Mersenne Twister pseudo-random number generator, [85].
- Dos algoritmos empleados para generar datos aleatorios por *Matlab* (método congruente multiplicativo) y *Excel* [140].
- Dos ruidos físicos: ruido de decaimiento radiactivo [141] y ruido atmosférico [142]. Los archivos de datos para estos ruidos están disponibles en los online en los sitios referidos.
- Dos mapas caóticos M^1 y sus versiones iteradas M^2 a M^8 [39] para el mapa Logístico y el Three Way Bernoulli Map (TWBM).

La Figura 6.18 muestra los resultados en el plano de doble entropía $H_{BP} \times H_{hist}$ para todos los ruidos estudiados. Puede verse que los ruidos físicos, el algoritmo *Mersenne Twister* y los PRNGs utilizados en *Matlab*(función rand) y en *Excel* (función RAND), tienen el valor máximo para H_{BP} , lo que indica que todos los patrones de orden aparecen casi el

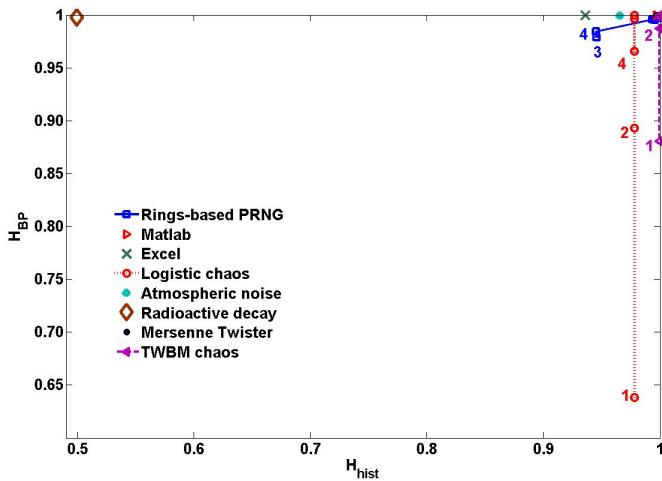


Figura 6.18: Plano $H_{hist} \times H_{BP}$ para distintos RNGs. Los números que siguen a cada cuadrado indica la cantidad de ROs utilizado en cada TRNG. Los números al lado de cada punto en los resultados de los mapas Logístico y TWBM indican el número de iteración.

el mismo número de veces. Sin embargo, estos cinco ruidos presentan un comportamiento muy diferente con respecto al cuantificador H_{hist} . El decaimiento radiactivo es el peor, con $H_{hist} \sim 0,5$, lo que indica que esta secuencia no muestra todos los valores posibles en la misma proporción. Los números al lado de cada marcador para las secuencias caóticas indican el número de iteración. Los mapas iterados tienen mayor H_{BP} debido a su propiedad de mezcla [39]. El plano de entropía dual muestra que un aumento en el número de ROs mejora tanto H_{BP} como H_{hist} .

La Figura 6.19 es una vista con más detalle de la Figura 6.18 alrededor del punto ideal $(1, 1)$. Allí, se muestra la evolución de las secuencias cuando la cantidad de ROs aumenta de 5 a 50. Se puede observar que a medida que el número de anillos aumenta los datos aumentan su mezcla y también el histograma tiende a ser más uniforme, por lo tanto, ambas propiedades mejoran. Se puede determinar un umbral en el número de anillos, ya que los puntos saturan alrededor de $(0,997, 1)$, por lo que este es el mejor TRNG posible, usar más de 15 ROs no presenta ninguna mejora. Como se dijo anteriormente, el cuantificador H_{hist} detecta la variación del histograma de la secuencia, y el cuantificador H_{BP} refleja la mejora en la mezcla de datos. Finalmente, las secuencias de Mersenne Twister y Matlab presentan un valor idéntico, ideal H_{BP} y un valor alto de H_{hist} ; no obstante, el histograma no es perfectamente uniforme (los valores no son equiprobables).

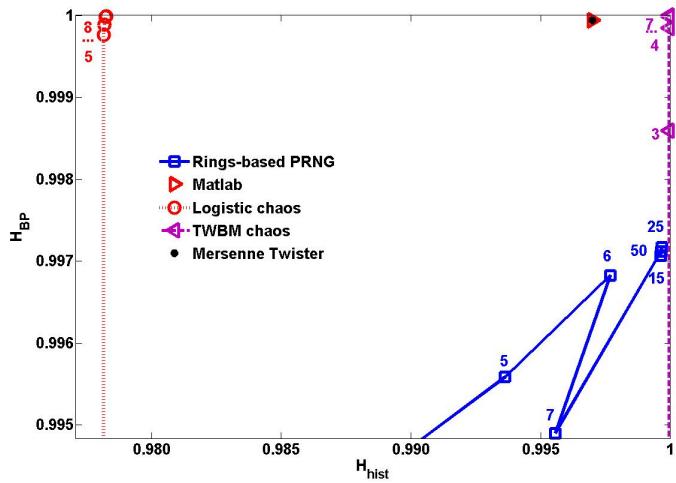


Figura 6.19: Detalle de la Figura 6.18 alrededor del punto ideal (1, 1).

6.4. Conclusiones

El *jitter* es inevitable en ROs, y en consecuencia, necesita ser caracterizado. La mezcla y la distribución de valores son las principales propiedades a considerar. Varios ITQ fueron evaluados aquí. S_W , $S_{BP}^{(D)}$, H_W y $H_{BP}^{(D)}$ resultan dependientes de los parámetros W y D . Esto es un inconveniente si se deben emplear como cuantificadores de aleatoriedad, ya que dependen de parámetros externos a la fuente generadora de símbolos. Por otro lado, no es posible calcular *rate entropies*, h_0^* y h_0 , ya que se necesita una cantidad infinita de datos para su cálculo. Las dos entropías diferenciales, h^* y h , en cambio, son independientes de los parámetros utilizados para su determinación y son estimadores de la *rate entropy*. Se demostró en la Sección 6.2.1 que, en el caso de ROs muestreados, presentan un mínimo para la tasa de muestreo correcta, lo que los convierte en una buena medida de la calidad tanto de los ROs como de los PRNGs derivados de ellos.

El plano de entropía dual determinado por estos cuantificadores ha demostrado discernir satisfactoriamente entre las dos principales propiedades deseadas de PRNG, la equiprobabilidad entre todos los valores posibles y la independencia estadística entre valores consecutivos. Por lo tanto, permite ver claramente lo que debe mejorarse en una secuencia determinada para obtener una buena PRNG. Los ejemplos presentados aquí han demostrado la necesidad de utilizar ambos histogramas para caracterizar secuencias.

En cuanto a la implementación en hardware, los TRNGs basados en RO implementados

aquí han demostrado satisfacer las propiedades estadísticas deseadas para un RNG. Son comparables a otros RNGs utilizados y en algunos casos presentan mejores características. Emplean pocos recursos del dispositivo y se implementan de forma muy simple en una plataforma digital.

Se demostró que para esta arquitectura la cantidad de ROs establece las propiedades estadísticas del TRNG. Se vio que para 15 ROs el histograma y la mezcla, eran casi ideales, haciendo innecesario el aumento de la cantidad de anillos.

Bibliografía

- [1] Romuel F Machado, Murilo S Baptista, and Celso Grebogi. Cryptography with chaos at the physical level. *Chaos, Solitons & Fractals*, 21(5):1265–1269, 2004.
- [2] Nejib Smaoui and Ali Kanso. Cryptography with chaos and shadowing. *Chaos, Solitons & Fractals*, 42(4):2312–2321, 2009.
- [3] Raymond. Kapral and Kenneth. Showalter. *Chemical Waves and Patterns*. Springer Netherlands, 1995.
- [4] Jan Awrejcewicz and C.-H. Lamarque. *Bifurcation and Chaos in Nonsmooth Mechanical Systems*, volume 45 of *World Scientific Series on Nonlinear Science Series A*. WORLD SCIENTIFIC, jul 2003.
- [5] Steven Strogatz. Nonlinear Dynamics and Chaos_ With Applications to Physics, jan 2018.
- [6] G. Marsaglia. The marsaglia random number cdrom including the diehard battery of tests of randomness. <http://www.stat.fsu.edu/pub/diehard/>, 1995.
- [7] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423 y 623–656., 1948.
- [8] Christoph Bandt and Bernd Pompe. Permutation entropy: a natural complexity measure for time series. *Physical Review Letters*, 88(17):174102, apr 2002.
- [9] A. Kantz. A robust method to estimate the maximal lyapunov exponent of a time series. *Phys. Lett. A*, 185(77), 1994.
- [10] ShiJun Liao and PengFei Wang. On the mathematically reliable long-term simulation of chaos of lorenz equation in the interval [0, 10000]. *arXiv preprint arXiv:1305.4222*, 2013.

- [11] S. Baron, T. Mastoridis, J. Troska, and P. Baudrenghien. Jitter impact on clock distribution in lhc experiments. In *TOPICAL WORKSHOP ON ELECTRONICS FOR PARTICLE PHYSICS 2012*. IOP for SISSA Media Lab, 2012.
- [12] Berk Sunar, William J. Martin, and Douglas R. Stinson. A provably secure true random number generator with built-in tolerance to active attacks. *IEEE Transactions on Computers*, 56(1):109–119, 2007.
- [13] M. Antonelli, L. De Micco, H. A. Larrondo, and O. A. Rosso. Complexity of Simple , Switched and Skipped Chaotic Maps in Finite Precision. *Entropy*, 20(2):1–24, feb 2018.
- [14] M. Antonelli, L. De Micco, and H. A. Larrondo. Measuring the jitter of ring oscillators by means of information theory quantifiers. *Communications in Nonlinear Science and Numerical Simulation*, 43:139–150, 2017.
- [15] Maximiliano Antonelli and Luciana De Micco. Implementación de algoritmo genético para la búsqueda automática de caos en sistemas multiatractores. volume 3, page 51. UBA, 2013.
- [16] L. De Micco, M. Antonelli, C.M. Gonzalez, and H.A Larrondo. Hardware implementation of maximum lyapunov exponent. In *Embedded Systems (SASE/CASE), 2013 Fourth Argentine Symposium and Conference on*, pages 1–4, Aug 2013.
- [17] L. De Micco, M. Antonelli, M. L. Crespo, and A. Cicuttin. Hw/sw codesign of maximum lyapunov exponent estimator. In *2017 IEEE 8th Latin American Symposium on Circuits Systems (LASCAS)*, pages 1–4, Feb 2017.
- [18] M. Antonelli and L. De Micco. Cripto-codificación caótica variante en el tiempo. 2012.
- [19] M. Antonelli, L. De Micco, C. M. Gonzalez, and H. A. Larrondo. Analysis of the digital implementation of a chaotic deterministic-stochastic attractor. In *2012 Argentine School of Micro-Nanoelectronics, Technology and Applications (EAMTA)*, pages 73–78, Aug 2012.
- [20] L De Micco, M Antonelli, and H.A. Larrondo. Stochastic degradation of the fixed-point version of 2D-chaotic maps. *Chaos, Solitons and Fractals*, 104:477–484, 2017.

- [21] L. De Micco, M. Antonelli, H. A. Larrondo, and E. Boemo. Ro-based prng: Fpga implementation and stochastic analysis. In *2014 IX Southern Conference on Programmable Logic (SPL)*, pages 1–6, Nov 2014.
- [22] L. De Micco, R. A. Petrocelli, D. O. Carrica, and H. A. Larrondo. Muestreo caótico para la adquisición de señales de baja frecuencia con ruido de alta frecuencia. *Proceedings de la XII Reunión de Trabajo en Procesamiento de la Información y Control*, 2007.
- [23] L. De Micco, R. A. Petrocelli, and H. A. Larrondo. Constant envelope wideband signals using arbitrary chaotic maps. *Proceedings of XII Reunión de Trabajo en Procesamiento de la Información y Control (RPIC 2007)*, 2007.
- [24] L. De Micco, C. M. Arizmendi, and H. A. Larrondo. Zipping characterization of chaotic sequences used in spread spectrum communication systems. *Institute of Physics Conference Proceedings 913*, pages 139–144, 2007.
- [25] Eckehard Schöll. *Nonlinear Spatio-Temporal Dynamics and Chaos in Semiconductors*. Number 1992. Cambridge University Press, 2001.
- [26] E. N. Lorenz. Deterministic non periodic flow. *Journal of the Atmospheric Sciences*, 20:130 – 141, 1963.
- [27] A. Lasota and M. C. Mackey. *Chaos, Fractals, and Noise: Stochastic Aspects of Dynamics*. Applied Mathematical Sciences 97. Springer Verlag, 2nd. edition. edition, 1994.
- [28] A. Lasota and J. A. Yorke. On the existence of invariant measure for piecewise monotonic transformations. *Trans. Amer. Math. Soc.*, 186:481–488, 1973.
- [29] Shujun Li. *When Chaos Meets Computers*, 2004.
- [30] Musheer Ahmad, Hitesh Chugh, Avish Goel, and Prateek Singla. *A Chaos Based Method for Efficient Cryptographic S-box Design*, pages 130–137. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [31] Iqtadar Hussain, Tariq Shah, Muhammad Asif Gondal, and Hasan Mahmood. Efficient method for designing chaotic s-boxes based on generalized baker’s map and tderc chaotic sequence. *Nonlinear Dynamics*, 74(1):271–275, 2013.

- [32] M. Antonelli, L. De Micco, and H. Larrondo. Causal and non-causal entropy quantifiers implemented in fpga. In *2016 IEEE Biennial Congress of Argentina (ARGEN-CON)*, pages 1–5. IEEE, jun 2016.
- [33] M. Antonelli, L. De Micco, and H. Larrondo. Degradación de las propiedades estadísticas de señales durante el proceso de medición. In *2016 Taller de no se que cosa, TREFEMAC 2016*, pages 38–39, may 2016.
- [34] J Sprott. *Chaos and Time-Series Analysis*. Oxford University Press, 2003.
- [35] Ma Xian-Min. Detecting of coal gas weak signals using lyapunov exponent under strong noise background. *Intelligent System Design and Engineering Applications (ISDEA), 2013 Third International Conference on*, 2013.
- [36] S. M. Bruijna, D. J.J. Bregmanc, O. G. Meijerb, P. J. Beekb, and J. H. van Dieën. Maximum lyapunov exponents as predictors of global gait stability: A modelling approach. *Medical Engineering and Physics*, 2011.
- [37] Thomas Weise. *Global Optimization Algorithms*. 2009.
- [38] O. A. Rosso, L. De Micco, H. A. Larrondo, M. T. Martin, and A. Plastino. Generalized statistical complexity measure: a new tool for dynamical systems. *International Journal of Bifurcation and Chaos*, Vol. , No. 3 (2010) ., 20(3):775–785.
- [39] L. De Micco, C. M. González, H. A. Larrondo, M. T. Martin, A. Plastino, and O. A. Rosso. Randomizing nonlinear maps via symbolic dynamics. *Physica A*, 387:3373–3383, 2008.
- [40] K. Mischaikow, M. Mrozek, J. Reiss, and A. Szymczak. Construction of symbolic dynamics from experimental time series. *Phys. Rev. Lett.*, 82:1114–1147, 1999.
- [41] G. E. Powell and I. C. Percival. A spectral entropy method for distinguishing regular and irregular motion of hamiltonian systems. *J. Phys. A: Math. Gen.*, 12:2053–2071, 1979.
- [42] O. A. Rosso, S. Blanco, J. Jordanova, V. Kolev, A. Figliola, M. Schürmann, and E. Başar. Wavelet entropy: a new tool for analysis of short duration brain electrical signals. *Journal of Neuroscience Methods*, 105:65–75, 2001.

- [43] Holger Kantz, J. (Jürgen) Kurths, and G. (Gottfried) Mayer-Kress. *Nonlinear analysis of physiological data*. Springer, 1998.
- [44] D. P. Feldman, C. S. McTague, and P. Crutchfield. The organization of intrinsic computation: complexity-entropy diagrams and the diversity of natural information processing. *arxiv.org:0866.4789[nlin.CD]*, pages 1–18, junio 2008.
- [45] D. P. Feldman and J. P. Crutchfield. Measures of statistical complexity: why? *Physics Letters A*, 238:244–252, 1998.
- [46] P. W. Lamberti, M. T. Martín, A. Plastino, and O. A. Rosso. Intensive entropic non-triviality measure. *Physica A*, 334:119–131, 2004.
- [47] R. López-Ruiz, H. L. Mancini, and X. Calbet. A statistical measure of complexity. *Phys. Lett. A*, 209:321–326, 1995.
- [48] M. T. Martín, A. Plastino, and O. A. Rosso. Statistical complexity and disequilibrium. *Phys. Lett. A*, 311:126–132, 2003.
- [49] Ivo Grosse, Pedro Bernaola-Galván, Pedro Carpena, Ramón Román-Roldán, Jose Oliver, and H. Eugene Stanley. Analysis of symbolic sequences using the Jensen-Shannon divergence. *Physical Review E*, 65(4):041905, mar 2002.
- [50] M. T. Martín and A. Plastino. Generalized statistical complexity measures: Geometrical and analytical properties. *Physica A*, 369:439–462, 2006.
- [51] M. T. Martin. *Ph.D. Thesis, Department of Mathematics*,. PhD thesis, Faculty of Sciences, University of La Plata, 2004.
- [52] S. Blanco, A. Figliola, R. Quian Quiroga, O. A. Rosso, and E. Serrano. Time-frequency analysis of electroencephalogram series (iii): Wavelet packets and information cost function. *Phys. Rev. E*, 57:932–940., 1998.
- [53] W. Ebeling and R. Steuer. Partition-based entropies of deterministic and stochastic maps. *Stochastics and Dynamics*, 1(1):1–17, 2001.
- [54] K. Keller and M. Sinn. Ordinal analysis of time series. *Physica A*, 356:114–120, 2005.
- [55] J. M. Amigó, L. Kocarev, and I. Tomovski. Discrete entropy. *Physica D*, 228:77–85., 2007.

- [56] O. A. Rosso, L. Zunino, D. G. Pérez, A. Figliola, H. A. Larrondo, M. Garavaglia, Martín M. T., and A. Plastino. Extracting features of gaussian selfsimilar stochastic processes via the bandt & pompe approach. *Phys. Rev. E*, 76(6):061114, 2007.
- [57] Bilal Fadlallah, Badong Chen, Andreas Keil, and José Príncipe. Weighted-permutation entropy: A complexity measure for time series incorporating amplitude information. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 87(2):1–7, 2013.
- [58] L. De Micco, H. A. Larrondo, A. Plastino, and O. A. Rosso. Quantifiers for randomness of chaotic pseudo random number generators. *Philosophical Transactions of the Royal Society A*, 367:3281–3296, 2009.
- [59] O. A. Rosso, H. A. Larrondo, M. T. Martín, A. Plastino, and M. A. Fuentes. Distinguishing Noise from Chaos. *Physical Review Letters*, 99(15):154102, oct 2007.
- [60] J. S.Armand Eyebe Fouda, Wolfram Koepf, and Sabir Jacquir. The ordinal Kolmogorov-Sinai entropy: A generalized approximation. *Commun. Nonlinear Sci. Numer. Simul.*, 46:103–115, 2017.
- [61] L. De Micco, R. A. Petrocelli, O. A. Rosso, A. Plastino, and H. A. Larrondo. Mixing chaotic maps and electromagnetic interference reduction. *International Journal of Applied Mathematics and Statistics*, 26:106–120, 2012.
- [62] Osvaldo A Rosso, Luciana De Micco, Hilda A Larrondo, Maria T Martín, and Angelo Plastino. Generalized Statistical Complexity Measure. *International Journal Of Bifurcation And Chaos*, 20(03):775, mar 2010.
- [63] C. Anteneodo and A. R. Plastino. Some features of the López-ruiz-mancini-calbet (lmc) statistical measure of complexity. *Phys. Lett. A*, 223(5):348–354, Diciembre 1996.
- [64] Felipe Olivares, Angelo Plastino, and Osvaldo A. Rosso. Contrasting chaos with noise via local versus global information quantifiers. *Physics Letters A*, 376(19):1577–1583, apr 2012.
- [65] C. M. González, H. A. Larrondo, and O. A. Rosso. Statistical complexity measure of pseudorandom bit generators. *Physica A*, 354:281–300, August 2005.

- [66] O. A. Rosso, L. C. Carpi, P. M. Saco, M. Gómez Ravetti, A. Plastino, and H. A. Larrondo. Causality and the entropy complexity plane: Robustness and missing ordinal patterns. *Physica A*, 391:42–55, 2012.
- [67] J. M. Amigó, L. Kocarev, and J. Szczepanski. Order patterns and chaos. *Physics Letters A*, 355:27–31, 2006.
- [68] J. M. Amigó, S. Zambrano, and M. A. F. Sanjuán. Combinatorial detection of determinism in noisy time series. *Europhysics Letters*, 83:60005, 2008.
- [69] J. M. Amigó. *Permutation complexity in dynamical systems*. Springer-Verlag, Berlin, Germany, 2010.
- [70] J. M. Amigó, B. Kennel, and L. Kocarev. The permutations entropy rate equals the metric entropy rate for ergodic information sources and ergodic dynamical systems. *Physica D*, 210:77–95, 2005.
- [71] Yu Gu, Andrew McCallum, and Don Towsley. Detecting anomalies in network traffic using maximum entropy estimation. In *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, IMC ’05, pages 32–32, Berkeley, CA, USA, 2005. USENIX Association.
- [72] Arno Wagner and Bernhard Plattner. Entropy Based Worm and Anomaly Detection in Fast IP Networks. In *Proceedings of the 14th IEEE International Workshops on Enabling Technologies Infrastructure for Collaborative Enterprise*, 2005.
- [73] Actel Corporation. *Fusion Embedded Development Kit*, 2009.
- [74] Subramanya Nagalakshmi. *Study of FPGA implementation of entropy norm computation for IP data streams*. PhD thesis, University of South Florida Scholar Commons, 2008.
- [75] Actel Corporation. *Core8051s Embedded Processor Hardware Development Tutorial, for Fusion Mixed-Signal FPGAs*, 2009.
- [76] Actel Corporation. *Core8051s Embedded Processor Software Development Tutorial, for Fusion Mixed-Signal FPGAs*, 2009.
- [77] Issue 7 The Open Group Base Specifications. math.h: mathematical declarations, base definitions reference.

- [78] J.G. Fernández, H. A. Larrondo, H. A. Slavin, D. G. Levin, R. M. Hidalgo, and R. R. Rivera. Masking properties of apd communication systems. *Physica A*, 354:281–300, 2005.
- [79] G. Setti, R. Rovatti, and G. Mazzini. Performance of chaos-based asynchronous ds-cdma with different pulse shapes. *IEEE Communications Letters*, 8(7):416–418, July 2004.
- [80] L. Kocarev and G. Jakimoski. Pseudorandom bits generated by chaotic maps. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 50(1):123–126, January 2003.
- [81] H. A. Larrondo, M. T. Martin, C.M. González, A. Plastino, and O. A. Rosso. Random number generators and causality. *Phys. Lett. A*, 352(4- 5):421–425, April 2006.
- [82] S. Callegari, R. Rovatti, and G. Setti. Chaos-based fm signals: application and implementation issues. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 50(8):1141–1147, August 2003.
- [83] L. Kocarev and U. Parlitz. General approach for chaotic synchronization with applications to communication. *Physical Review Letters*, 74(25):5028–5031, 1995.
- [84] R. M. Hidalgo, J. G. Fernández, R. R. Rivera, and H. A. Larrondo. Versatile dsp-based chaotic communication system. *Electronic Letters*, 37:1204–1205, 2001.
- [85] M. Matsumoto and T. Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. on Modeling and Computer Simulation*, 8(1):3–30, January 1998.
- [86] G. Marsaglia and A. Zaman. A new class of random number generators george marsaglia and arif zaman the annals of applied probability. *Institute of Mathematical Statistics Stable*, 1(3):462–480, August 1991.
- [87] J. Boyar. Inferring sequences produced by pseudo-random number generators. *Journal of the ACM*, 36(1):129–141, 1989.
- [88] L. Kocarev and S. Lian. *Chaos-Based Cryptography: Theory, Algorithms and Applications*. Studies in Computational Intelligence. Springer, 2011.

- [89] Xiao Song Yang and Quan Yuan. Chaos and transient chaos in simple Hopfield neural networks. *Neurocomputing*, 69(1-3):232–241, dec 2005.
- [90] Slobodan Kozic, Thomas Schimming, and Martin Hasler. Controlled One- and Multidimensional Modulations Using Chaotic Maps. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 53(9):2048– 2059, 2006.
- [91] Slobodan Kozic and Martin Hasler. Belief Propagation Decoding for Codes Based on Discretized Chaotic Maps. In *International Symposium on Circuits and Systems (ISCAS)*, 2006.
- [92] G.A. Constantinides, P.Y.K. Cheung, and W. Luk. Optimum wordlength allocation. In *Field-Programmable Custom Computing Machines, 2002. Proceedings. 10th Annual IEEE Symposium on*, pages 219–228, 2002.
- [93] George A. Constantinides, Peter Y. K. Cheung, and Wayne Luk. Wordlength optimization for linear digital signal processing. *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, 22:1432–1442, 2003.
- [94] Qun Ding, Jing Pang, Jinqing Fang, and Xiyuan Peng. Designing of chaotic system output sequence circuit based on fpga and its applications in network encryption card. *International Journal of Innovative Computing, Information and Control*, 3:1 – 6, 2007.
- [95] M. A. Asseri, M. I. Sobhy, and P. Lee. Lorenz chaotic model using field programmable gate array. *The 2002 45th Midwest Symposium on Circuits and Systems, 2002. MWSCAS-2002*, 1:I – 527–30, 2002.
- [96] M.S. Azzaz, C. Tanougast, S. Sadoudi, A. Bouridane, and A. Dandache. Fpga implementation of new real-time image encryption based switching chaotic systems. *Signals and Systems Conference (ISSC 2009)*, pages 1 – 6, 2009.
- [97] L. De Micco, O. G. Zabaleta, C. M. González, C. M. Arizmendi, and H. A. Larrondo, editors. *Estocasticidad de un atractor caótico determinista implementado en FPGA*, 2010.
- [98] M.S. Azzaz, S. Tanougast, C. and Sadoudi, A. Bouridane, and A. Dandache. An fpga implementation of a feed-back chaotic synchronization for secure communications.

- International Symposium on Communication Systems Networks and Digital Signal Processing (CSNDSP)*, pages 239 – 243, 2010.
- [99] C. M. González, H. A. Larrondo, C. A Gayoso, and L. J. Arnone. Generación de secuencias binarias pseudo aleatorias por medio de un mapa caótico 3d. In *Proceedings del IX Workshop de IBERCHIP*, 2003.
 - [100] J. Soto. Statistical testing of random number generators. *available online at <http://csrc.nist.gov/groups/ST/toolkit/rng/documents/nissc-paper.pdf>.*
 - [101] H. M. Gustafson, E. P. Dawson, L. Nielsen, and W. J. Caelli. A computer package for measuring the strength of ciphers. *Computers and Security*, 13(8):687–697, 1994.
 - [102] A.L. Rukhin. Testing randomness: a suite of statistical procedures. *Theory Probab. Appl.*, 45:111, 2000.
 - [103] P. L'Ecuyer and R. Simard. Testu01: A c library for empirical testing of random number generators. *ACM Transactions on Mathematical Software*, 33(22), 2007.
 - [104] R. G. Brown. dieharder: A random number test suite. <http://www.phy.duke.edu/~rgb/General/dieharder.php>, 2012.
 - [105] ALTERA. *Quartus II Handbook Version 9.1*, 2009.
 - [106] Rathindra Nath Giri and M. K. Pandit. Pipelined floating-point arithmetic unit (fpu) for advanced computing systems using fpga. *International Journal of Engineering and Advanced Technology*, 1(4):168–174, 2012.
 - [107] Gokul Govindu, Ling Zhuo, Seonil Choi, and Viktor Prasanna. Analysis of high-performance floating-point arithmetic on fpgas. 2004.
 - [108] Celso Grebogi, Edward Ott, and James A. Yorke. Roundoff-induced periodicity and the correlation dimension of chaotic attractors. *Phys. Rev. A*, 38:3688–3692, Oct 1988.
 - [109] KJ Persohn and Richard J Povinelli. Analyzing logistic map pseudorandom number generators for periodicity induced by finite precision floating-point representation. *Chaos, Solitons & Fractals*, 45(3):238–245, 2012.
 - [110] Julien Clinton Sprott. Automatic generation of strange attractors. *Computers & Graphics*, 17(3):325–332, 1993.

- [111] Esteban Tlelo-Cuautle, Antonio de Jesus Quintas-Valles, Luis Gerardo De La Fraga, and Jose de Jesus Rangel-Magdaleno. VHDL descriptions for the FPGA implementation of PWL-function-based multi-scroll chaotic oscillators. *PLoS One*, 11(12):e0168300, dec 2016.
- [112] Luis Gerardo de la Fraga, Esteban Torres-Pérez, Esteban Tlelo-Cuautle, and Cuauhtemoc Mancillas-López. Hardware implementation of pseudo-random number generators based on chaotic maps. *Nonlinear Dyn.*, 90(3):1661–1670, nov 2017.
- [113] N. Nagaraj, M. C. Shastry, and P. G. Vaidya. Increasing average period lengths by switching of robust chaos maps in finite precision. *The European Physical Journal Special Topics*, 165:73–83, 2008.
- [114] Xinzhi Liu, Kok-Lay Teo, Hongtao Zhang, and Guanrong Chen. Switching control of linear systems for generating chaos. *Chaos, Solitons and Fractals*, 30:725–733, 2006.
- [115] E. Gluskin. The nonlinear-by-switching systems (a heuristic discussion of some basic singular systems). <http://arxiv.org/abs/0801.3652>, 2008.
- [116] M. Jessa. The period of sequences generated by tent-like maps. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 49(1):84–89, 2002.
- [117] Sergio Callegari, Gianluca Setti, and P.J. Langlois. A CMOS tailed tent map for the generation of uniformly distributed chaotic sequences. In *Proceedings of 1997 IEEE International Symposium on Circuits and Systems. Circuits and Systems in the Information Age ISCAS '97*, volume 2, pages 781–784. IEEE, 2013.
- [118] Kim Beomsup, D. N. Helman, and P. R. Gray. The electromyographic jitter in normal human muscles. *Electroencephalography and Clinical Neurophysiology*, 31:429–438, 1971.
- [119] A Mecozzi, C. B. Clausen, Sang-Gyu Park, and A. H. Gnauck. Cancellation of timing and amplitude jitter in symmetric links using highly dispersed pulses. *IEEE PHOTONICS TECHNOLOGY LETTERS*, 13(5):445–447, 2001.
- [120] D. J. Derickson, P. A. Morton, J. E. Bowers, and R. L. Thornton. Comparison of timing jitter in external and monolithic cavity mode-locked semiconductor lasers. *Applied Physics Letters*, 59:3372–3374, 1991.

- [121] J. T. Wright. Radial velocity jitter in stars from the california and carnegie planet search at keck observatory. *Publications of the Astronomical Society of the Pacific*, 117:657–664, 2005.
- [122] C. K. Marsalek and J. MAUNSELL. On the relationship between synaptic input and spike output jitter in individual neurons. *Neurobiology*, 94:735–740, 1997.
- [123] Sergio Lopez-Buedo Javier, Javier Garrido, and Eduardo Boemo. Thermal testing on programmable logic devices. *IEEE Int'l Symp. Circuits and Systems*, 2:240–243, 1998.
- [124] Kim Beomsup, D. N. Helman, and P. R. Gray. A 30-mhz hybrid analog/digital clock recovery circuit in 2-?m cmos. *Journal of Solid-State Circuits*, 25(6):1385–1394, 1990.
- [125] Ali Hajimiri, Sotirios Limotyrakis, and Thomas H. Lee. Jitter and phase noise in ring oscillators. *IEEE Journal of Solid-state Circuits*, 1999.
- [126] M. K. Mandal and B. C. Sarkar. Ring oscillators: Characteristics and applications. *Indian Journal of Pure & Applied Physics*, 48:136–145, feb 2010.
- [127] Nisha Gupta. Article: Voltage-controlled ring oscillator for low phase noise application. *International Journal of Computer Applications*, 14(5):23–27, January 2011. Published by Foundation of Computer Science.
- [128] Knut Wold and Chik How Tan. Analysis and enhancement of random number generator in fpga based on oscillator rings. *Int. J. Reconfig. Comput.*, 2009:4:1–4:8, January 2009.
- [129] J. A. McNeill. Jitter in ring oscillators. *IEEE Journal of Solid State Circuits*, 32(6):870–879, 1997.
- [130] Boyan Valtchanov, Alain Aubert, Florent Bernard, and Viktor Fischer. Modeling and observing the jitter in ring oscillators implemented in fpgas. In Bernd Straube, Milos Drutarovsky, Michel Renovell, Peter Gramata, and Mária Fischerová, editors, *DDECS*, pages 158–163. IEEE Computer Society, 2008.
- [131] V. Fischer, F. Bernard, N. Bochard, and M. Varchola. Enhancing security of ring oscillator-based trng implemented in fpga. In *FPL*, pages 245–250, 2008.

- [132] B. Valtchanov, V. Fischer, A. Aubert, and F. Bernard. Characterization of randomness sources in ring oscillator-based true random number generators in fpgas. *Design and Diagnostics of Electronic Circuits and Systems (DDECS), 2010 IEEE 13th International Symposium on*, pages 48 – 53, 2010.
- [133] Mathieu Baudet, David Lubicz, Julien Micolod, and André Tassiaux. On the security of oscillator-based random number generators. *J. Cryptology*, 24(2):398–425, 2011.
- [134] M. Jessa and L. Matuszewski. Enhancing the randomness of a combined true random number generator based on the ring oscillator sampling method. *Reconfigurable Computing and FPGAs (ReConFig), International Conference on*, pages 274 – 279, 2011.
- [135] D. Lubicz and N. Bochard. Towards an oscillator based trng with a certified entropy rate. *Computers, IEEE Transactions on*, PP(99):1–1, 2014.
- [136] M. Antonelli. Emulating a ring oscillator with jitter. www.mathworks.com/matlabcentral/fileexchange/54021-jitter-samples-n-r-sigma-, 2015.
- [137] NIST. Guidelines for evaluating and expressing the uncertainty of nist measurement results; appendix d. Technical report, 2000.
- [138] NIST. *Analysis of repeatability*, 2000.
- [139] ALTERA. ip-basesuite.html. <http://www.altera.com/products/ip/design/basesuite/ip-basesuite.html.>, 2008.
- [140] A. Ian McLeod. Remark AS R58: A Remark on Algorithm AS 183. An Efficient and Portable Pseudo-Random Number Generator. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 34(2), 1985.
- [141] John Walker. HotBits: Genuine random numbers, generated by radioactive decay. *online at www.fourmilab.ch/hotbits*, 2001.
- [142] Mads Haahr. Random.org.