UACM

Universidad Autónoma de la Ciudad de México

NADA HUMANO ME ES AJENO

COLEGIO DE CIENCIA Y TECNOLOGÍA

LICENCIATURA EN MODELACIÓN MATEMÁTICA

El mapeo logístico modificado y su aplicación a la encriptación de imágenes en tonos de gris

T E S I S

QUE PARA OPTAR POR EL TÍTULO DE

LICENCIADA EN MODELACIÓN MATEMÁTICA

P R E S E N T A :

SARA APOLINAR CRISANTO

D I R E C T O R

DR. DANIEL ROBERTO JARDÓN ARCOS

Ciudad de México, abril de 2025.

UACM
Universidad Autónoma de la Ciudad de México
Nada humano me es ajeno

CA Coordinación Académica

SISTEMA BIBLIOTECARIO DE INFORMACIÓN Y DOCUMENTACIÓN

UNIVERSIDAD AUTÓNOMA DE LA CIUDAD DE MÉXICO
COORDINACIÓN ACADÉMICA

RESTRICCIONES DE USO PARA LAS TESIS DIGITALES

DERECHOS RESERVADOS©

Índice general

Índice de tablas

Índice de figuras

Agradecimientos

Expreso mi más sincero agradecimiento a la Universidad Autónoma de la Ciudad de México por el apoyo brindado a través de la beca para la elaboración de trabajo recepcional o tesis. También a los dos planteles Casa Libertad y San Lorenzo Tezonco por brindarme los conocimientos y habilidades que me han permitido desarrollarme de manera profesional. Agradezco a todos los profesores que fueron parte de mi trayectoria academica, en particular a los profesores Isaías López, Felipe Alfaro, Francisco Portillo, Igor Peña, a las profesoras Erendira Huerta y Erika Álvarez. Su pasión por la enseñanza, su dedicación me inspiraron a alcanzar nuevas metas. En especial agradezco a los maestros Isaías López y Francisco Portillo por sus valiosas enseñanzas en el campo de las matemáticas, y a la profesora Erendira por su valioso apoyo en el desarrollo de los nuevos proyectos que se presentaron durante la culminación de mis créditos.

Dedicatoria

Dedico esta tesis a mi madre, mi heroína, quien me enseñó a enfrentar la vida con valentía y a nunca rendirme. A mi padre, mi confidente, por sus sabios consejos y su amor incondicional. A mis hermanas y hermanos, mi familia, mi refugio. A Sofía y Julia, mis compañeras de vida, por su apoyo inquebrantable y por ser la razón de mi lucha.

A Alejandra García y Guadalupe Riaño, mis queridas amigas. Alejandra, mi cómplice en las aventuras más locas. Sus mensajes llenos de aliento en mis días más grises y su disposición para compartir mis tristezas me han hecho sentir profundamente acompañada, por todos los momentos felices que hemos vivido juntas y por ayudarme a encontrar la valentía para superar mis miedos. A Guadalupe, por su sabiduría y paciencia. Ha sido una guía invaluable en mi camino, y su influencia en mi vida ha sido profunda y duradera.

A mi tía, quien siempre me hizo preguntas que me obligaban a pensar más allá de las respuestas obvias. Su ausencia deja un vacío inmenso, pero su recuerdo me acompaña en cada paso que doy. Dedico este trabajo a ella, con la esperanza de que se sienta orgullosa de mi logro.

A mi sombra inseparable mi amigo perruno, cuya compañía fue mi refugio en las noches más oscuras y mi mayor apoyo durante este camino, que escuchó mis divagaciones sobre esta tesis y me ofreció su incondicional compañía. También quiero agradecer a mi querido tío Edgar, quien cuidó de nosotros en las noches en que nos quedábamos hasta tarde. A pesar de estar ocupado con su propio trabajo, siempre compartió su espacio y me brindó su apoyo incondicional.

A Luis, por ser mi confidente en momentos difíciles. A B. y B., quienes llenaron mi vida de felicidad y me recordaron que siempre puedo superar cualquier obstáculo. A mis compañeros de clase y amigos, por compartir conmigo este camino. Y a aquellos que ya no están, por su huella imborrable en mi vida. A todos ustedes, dedico este logro.

Por último y no menos importante al Dr. Daniel Jardón, mi tutor y mentor. Su apoyo incondicional y su confianza en mí fueron clave para la culminación de este trabajo. Más allá de su conocimiento y experiencia, agradezco su paciencia y su capacidad para motivarme en cada etapa del proceso. Gracias a él, no solo logré finalizar esta tesis, sino que también descubrí nuevas pasiones y oportunidades. Este logro es también suyo.

Con este trabajo doy por concluida una etapa, pero también inicio un nuevo camino, siempre agradecido por las enseñanzas recibidas a todos ustedes gracias.

Objetivos

General

- Proponer una modificación del mapeo logístico capaz de generar órbitas seudoaletorias para encriptar imágenes en tonos de gris, con buenas propiedades estadísticas.

Específicos

- Diseñar una modificación al mapeo logístico, considerando sus limitaciones para la generación de secuencias seudoaleatorias.
- Evaluar las órbitas generadas mediante pruebas estadísticas.
- Presentar ejemplos de aplicación de números seudoaleatorios (órbitas seudoaleatorias) generados por los mapeos mencionados.
- Desarrollar un método de encriptación de imágenes en tonos de gris.
- Aplicar el esquema a imágenes de prueba.
- Análisis estadístico de las imágenes encriptadas

Introducción

A partir de la década de los años $60'$, del siglo XX se han investigado, de manera sistemática, propiedades relativas al caos tales como, la sensibilidad a las condiciones iniciales, entropía y transitividad. Entre la literatura existente resalta el libro *An Introduction to Chaotic Dynamical Systems* [14], de Robert Devaney así como de Richard A. Holmgren [16]. En años recientes se ha publicado en México el libro *Sistemas Dinámicos Discretos* [8] de los autores Jefferson King y Héctor Méndez. Estos libros sirven como referencia obligada para entender diversos tópicos de la teoría relativa a los sistemas dinámicos discretos. Adicional a lo anterior, existen publicaciones de artículos recientes que muestran la importancia del caos y los sistemas dinámicos discretos.

Diversos autores han publicado artículos en los que utilizan las propiedades caóticas de algunos mapeos para encriptar imágenes definidas por píxeles, en tonos de grises o de color. La metodología usada por estos autores consiste en modificar los valores de los píxeles con ayuda de operaciones, como por ejemplo XOR (or exclusive por sus siglas en inglés), véase por ejemplo el artículo *A bit shift image Encryption Algorithm Based on Double Chaotic Systems* [24].

En los artículos de *Review of image encryption using different techniques* [1], *Modification of the logistic map using fuzzy numbers with application to pseudorandom number generation and image encryption* [13], *Chaos-based image encryption: review, application, and challenges* [6], entre otros se puede conocer la historia, muy reciente, de la encriptación de imágenes de píxeles con ayuda de mapeos caóticos. Adicional a lo anterior resalta el libro clásico *Digital Image Processing* de Rafael Gonzalez [10], al que cubre con creces el material necesario para trabajar con imágenes a color así como en tonos de gris.

En este trabajo de tesis,se usan los conocimientos y habilidades adquiridas en distintas materias del plan de estudios de la Licenciatura en Mo-

delación Matemática de la UACM. Respecto a la parte computacional se utilizan los software de programación R y Python que son de uso libre. Se usan imágenes en formato PNG de autoría propia.

La idea principal de este trabajo es utilizar modificaciones del mapeo logístico $f(x) = \lambda x(1-x)$ para generar secuencias seudoaleatorias con mejores resultados estadísticos a los obtenidos con el mapeo logístico normal. Lo anterior con el objetivo de encriptar imágenes de píxeles en tonos de grises.

En el Capítulo 1 se aplica el análisis del sistema dinámico, con enfoque particular en el mapeo logístico, así como algunos de los elementos por los que está conformado. También se presenta la composición del mapeo logístico y otros importantes mapeos. Además, se presenta una de las principales características de los sistemas dinámicos caóticos como es la sensibilidad a las condiciones iniciales.

En el Capítulo 2 se presentarán algunas pruebas estadísticas que determinarán si las órbitas generan números seudoaletorios con uniformidad dentro del intervalo $[0, 1]$ y si tienen un comportamiento aleatorio.

El Capítulo 3 demostrará la versatilidad de los números seudoaleatorios generados por el mapeo logístico y por el mapeo logístico modificado, a través de dos aplicaciones: el método de Montecarlo y la creación del triángulo de Sierpinski.

Finalmente, en el Capítulo 4, con el objetivo de encriptar imágenes de píxeles en tonos de gris, se usaron dos métodos distintos para sumar los valores de los píxeles de la imagen original con los valores obtenidos por el mapeo logístico modificado. Además se describe el proceso que permite desencriptar para obtener la imagen original. Se realiza las respectivas pruebas estadísticas para asegurar la confianza del método. Se usan familias de funciones del tipo $f_\lambda : [0, 1] \to [0, 1]$, con dos valores uno de ellos es el valor inicial $x_0 \in [0, 1]$, y otro es un parámetro $\lambda \in (3.99, 4]$. Por lo anterior, se utilizan dos llaves para encriptar los píxeles de una imagen, para desencriptar se utilizan las mismas llaves.

El método utilizado es notablemente sensible a los cambios de los valores de las llaves, se realizan pruebas con pequeñas modificaciones a las llaves, del orden de 10^{-14}, con lo cual no se recuperan las imágenes correctamente. También se resalta la gran cantidad de llaves distintas, lo cual, junto con la gran sensibilidad permite resistir a los llamados ataques por fuerza bruta.

Como complemento se realizaron algunas pruebas del Instituto Nacional de Estándares y Tecnología de Estados Unidos (NIST) para el análisis de generadores de números seudoaletorios.

Capítulo 1

Sistemas dinámicos discretos

La teoría relativa de los sistemas dinámicos discretos estudia el comportamiento de las órbitas generadas por las composiciones de una función $f : X \to X$, donde X es un espacio métrico.

Un espacio métrico es un conjunto que viene acompañado de una función que mide distancias entre cualesquiera dos de sus puntos.

Definición 1.0.1 *Un conjunto X se denomina espacio métrico si existe una función $d : X \times X \to [0, \infty)$, llamada métrica, tal que para $x, y, z \in X$ se tiene:*

i) $d(x, y) = 0$ *si y sólo si* $x = y$
ii) $d(x, y) = d(y, x)$
iii) $d(x, y) \leq d(x, z) + d(z, y)$ *(desigualdad del triángulo).*

El número $d(x, y)$, de la definición anterior, recibe el nombre de distancia entre x y y. El conjunto de los números reales $\mathbb{R}$, el plano $\mathbb{R}^2$, el conjunto de los números complejos $\mathbb{C}$, y el intervalo cerrado $[0, 1]$ son ejemplos de espacios métricos. Particularmente se usará principalmente en $\mathbb{R}$ y $[0, 1]$ generalmente se usa la distancia definida por $d(x, y) = |x - y|$

A continuación se define el concepto de órbita de un punto x, definida por una función $f : X \to X$. La órbita de un punto $x \in X$, bajo f se define por:

$$O(x, f) = \left\{x, f(x), f^2(x), f^3(x), \ldots\right\}$$

donde $f^n(x)$ es composición de f consigo misma $n - 1$ veces, es decir $\underbrace{f \circ f \circ f \circ \cdots \circ f}_{n \ f's}$.

El mapeo identidad en X se denota por $f^0 : X \to X$. Así las iteraciones de f se definen por:

$$\begin{array}{lcl} f^0(x) & = & x \\ f^1(x) & = & f(x) \\ f^2(x) & = & f(f(x)) \\ f^3(x) & = & f(f^2(x)) \\ \vdots & & \vdots \\ f^{n+1}(x) & = & f(f^n(x)) \\ \vdots & & \vdots \end{array}$$

Se utiliza la notación $x_k = f^k(x)$, así la órbita de $\{x\}$ es el conjunto:
$\{x_0 = x, x_1 = f(x), x_2 = f^2(x), x_3 = f^3(x), \ldots, x_{n-1} = f^{n-1}(x), f^n(x), \ldots\}$

Las órbitas pueden ser finitas o infinitas. Un caso particular son las órbitas de los puntos fijos. Se dice que $x_0 \in X$ es punto fijo de la función $f : X \to X$, si $f(x_0) = x_0$. La órbita de un punto fijo x_0 es por tanto:

$$O(x_0, f) = \{x_0, x_0, x_0, x_0, x_0, \ldots\} = \{x_0\}$$

ya que $f^n(x_0) = x_0$ para cualquier n-ésima iteración, por lo cual la órbita $O(x_0, f)$ es el conjunto unipuntual $\{x_0\}$.

En la proposición siguiente, extraída del libro *Sistemas Dinámicos Discretos* la prueba se puede verse en [8, pp. 12-13], se garantiza la existencia de un punto fijo bajo ciertas condiciones.

Proposición 1.0.1 *Sean A un intervalo en $\mathbb{R}$ y $f : A \to A$ una función continua en A. Sea $[a, b]$ un intervalo contenido en A.*

1. *Si $f([a, b]) \subset [a, b]$, entonces f tiene un punto fijo en $[a, b]$.*
2. *Si $[a, b] \subset f([a, b])$, entonces f tiene un punto fijo en $[a, b]$.*

Los puntos 1 y 2 son relevantes ya que se usarán principalmente funciones de $[0, 1]$ en $[0, 1]$.

1.1. Puntos fijos atractores y repulsores

- **Puntos fijos atractores**

  Sean A un intervalo en $\mathbb{R}$ y $f : A \to A$ una función continua. Se dice que un $x_0 \in A$, tal que $f(x_0) = x_0$, es punto fijo atractor, si existe un intervalo (a,b), tal que $x_0 \in (a, b)$

  $$f((a, b) \cap A) \subset (a, b) \cap A$$

  y además para todo $x \in (a, b) \cap A$ se tiene que $\lim_{n\to\infty} f^n(x) = x_0$, lo cual indica que las órbitas de puntos cercanos convergen al punto fijo x_0.

Figura 1.1: Punto fijo atractor

Cuenca atractora

Sea x_0 un punto fijo atractor bajo una función $f : \mathbb{R} \to \mathbb{R}$. La cuenca de atracción de x_0 se define como el conjunto de todos los puntos en X cuyas órbitas convergen a x_0.

- **Puntos fijos repulsores**

  Sea A un intervalo en $\mathbb{R}$, y $f : A \to A$ una función continua en A

  Se dice que x_0 es un punto fijo repulsor, si existe (a, b) tal que $x_0 \in (a, b)$ y además para cada $x \in (a, b) \cap A$, $x \neq x_0$, existe $n \in \mathbb{N}$, $n = n(x)$, tal que

  $$f^n(x) \notin (a, b) \cap A$$

  es decir las órbitas de puntos cercanos escapan, en un tiempo finito que depende de cada punto.

Figura 1.2: Punto repulsor

Para determinar si estos puntos son repulsores o atractores se emplea la proposición 2 mencionada [8, p. 15].

Proposición 1.1.1 *Suponga que f es continua en A, derivable en $x_0 \in A$ y $f(x_0) = x_0$:*

i) si $|f'(x_0)| < 1$, entonces x_0 es punto fijo atractor;

ii) si $|f'(x_0)| > 1$, entonces x_0 es punto fijo repulsor.

A continuación se realiza la demostración.

Dado que $f(x)$ es diferenciable en $x_0 \in (a, b)$ se tiene que también es continua en x_0. Se supone que $|f'(x_0)| < 1$. La función $g(x)$ definida por

$$g(x) = \begin{cases} \frac{f(x)-f(x_0)}{x-x_0} & \text{si } x \in (a,b) \setminus \{x_0\} \\ f'(x_0) & \text{si } x = \{x_0\} \end{cases}$$

es continua en (a, b), también es continua. También es continua $|g(x)|$. Por lo anterior para

$\epsilon = \frac{1-|f'(x_0)|}{2} > 0$ existe δ_0 tal que $0 < |x - x_0| < \delta_0$ implica que

$|g(x) - g(x_0)| = |\frac{f(x)-f(x_0)}{x-x_0} - f'(x_0)| = |\frac{f(x)-x_0}{x-x_0} - f'(x_0)| < \epsilon = \frac{1-|f'(x_0)|}{2}$.

Se define $\delta = \frac{1}{2}\min\{\delta_0, \epsilon\}$. Si $0 < |x - x_0| < \delta$, entonces se cumple lo anterior y, por lo tanto

$|\frac{f(x)-x_0}{x-x_0}| - |f'(x_0)| \leq |\frac{f(x)-x_0}{x-x_0} - f'(x_0)| < \frac{1-|f'(x_0)|}{2}$

$|\frac{f(x)-x_0}{x-x_0}| < \frac{1-|f'(x_0)|}{2} + |f'(x_0)| = \frac{1-|f'(x_0)|+2|f'(x_0)|}{2} = \frac{1+|f'(x_0)|}{2}$.

Dado que $|f'(x_0)| < 1$, se tiene que:

$\frac{1+|f'(x_0)|}{2} < 1$, se denota $\lambda = \frac{1+|f'(x_0)|}{2}$.

Note que $\frac{1}{2} \leq \lambda < 1$ y además

$|f(x) - x_0| < \lambda|x - x_0|$

Por la elección de δ es fácil ver que

$|f^2(x) - x_0| < \lambda|f(x) - x_0| < \lambda^2|x - x_0|$

y en general para cada $n \in \mathbb{N}$

$|f^n(x) - x_0| < \lambda^n|x - x_0|$

Dado que $|x - x_0|$ es una constante, para una x fija, se tiene que

$\lim_{n\to\infty} |f^n(x) - x_0| = 0,$

lo cual equivale a $\lim_{n\to\infty} f^n(x) = x_0$. Con lo anterior, se concluye la demostración de i). De manera análoga, se muestra la parte ii). □

Un punto fijo atractor x_0 para el cual se tiene $|f'(x_0)| = 0$ se denomina punto súper atractor.

En contraste si $|f'(x_0)| = 1$, se tiene otro tipo de comportamiento para el punto fijo x_0 (punto fijo neutral).

Para ilustrar la pertinencia del llamado punto fijo neutral, se puede considerar, por ejemplo, la función $f : \mathbb{R} \to \mathbb{R}$, definida por $f(x) = -x$, sólo tiene un punto fijo $x_0 = 0$. Al derivar y evaluar en x_0 se tiene

$f'(x) = -1$ y $|f'(x_0)| = |f'(0)| = |-1| = 1$

es decir $x_0 = 0$ es un punto fijo neutral. Cabe señalar que para todo $x \in \mathbb{R}$ distinto de 0 se tiene $f(x) = -x \neq x$ y $f^2(x) = f(-x) = x$, con lo cual se obtienen órbitas con sólo dos elementos, es decir $\{-x, x\}$, mismas que ya no se pueden acercar ni alejar al punto fijo $x_0 = 0$.

1.2. Puntos periódicos

Sea $f : X \to X$ una función continua y $x_0 \in X$. Se dice que x_0 es un punto periódico de f, si existe $n \in \mathbb{N}$ tal que $f^n(x_0) = x_0$. Se denota $Per(f) = \{x : x \text{ es periódico}\}$. Si $x_0 \in Per(f)$, se dice que $O(x_0, f)$ es una órbita periódica, además se define el periodo de x_0 como:

$$\text{mín}\,\{n \in \mathbb{N} : f^n(x_0) = x_0\}$$

1. Si x_0 es punto fijo de f, entonces x_0 tiene periodo 1 lo cual implica que $x_0 \in Per(f)$.

2. Si x_0 un punto periódico bajo f, de periodo k, con $k \geq 2$, entonces para cada $1 \leq j < k$ se tiene que $f^j(x_0)$ es distinto de x_0; En general si $i \neq j$ y $0 \leq i, j < k$ se tiene que $f^i(x_0) \neq f^j(x_0)$.

Para mejor referencia consulte [8, pp. 17-18].

Uno de los resultados más importantes en la teoría de los sistemas dinámicos discretos es el Teorema de Sharkovskii. Este teorema establece una relación entre los posibles periodos de un mapeo. Se utiliza por comodidad los números naturales en orden con el símbolo $\triangleright$.

$$\begin{array}{ccccccccc}
3 & \triangleright & 5 & \triangleright & 7 & \triangleright & 9 & \triangleright & \cdots \\
2\cdot 3 & \triangleright & 2\cdot 5 & \triangleright & 2\cdot 7 & \triangleright & 2\cdot 9 & \triangleright & \cdots \\
2^2\cdot 3 & \triangleright & 2^2\cdot 5 & \triangleright & 2^2\cdot 7 & \triangleright & 2^2\cdot 9 & \triangleright & \cdots \\
2^3\cdot 3 & \triangleright & 2^3\cdot 5 & \triangleright & 2^3\cdot 7 & \triangleright & 2^3\cdot 9 & \triangleright & \cdots \\
\vdots & & & & & & & & \\
\cdots & \triangleright & 2^3 & \triangleright & 2^2 & \triangleright & 2 & \triangleright & 1
\end{array}$$

Teorema 1.2.1 ***Sharkovskii***

Sea A un intervalo en $\mathbb{R}$. Sean n y m números naturales.

- *Si una función continua $f : A \to A$ tiene un punto de periodo n y $n \triangleright m$, entonces f tiene un punto periódico de periodo m.*
- *Si $m \triangleright n$, entonces existe una función continua $f : A \to A$ que tiene un punto de periodo n, pero no tiene puntos de periodo m.*
- *Existe una función continua $f : A \to A$ que tiene puntos periódicos de periodo 2^k para todo $k \in \mathbb{N} \cup \{0\}$, y no tiene puntos periódicos de ningún otro periodo.*

Para la demostración del Teorema de Sharkovskii consulte el libro [8, pp. 45-57]

1.3. Mapeos tienda, logístico y pastelero

El mapeo tienda $T : \mathbb{R} \to \mathbb{R}$ es considerado uno de los mapeos más importantes en la teoría de los sistemas dinámicos discretos. T se define por:

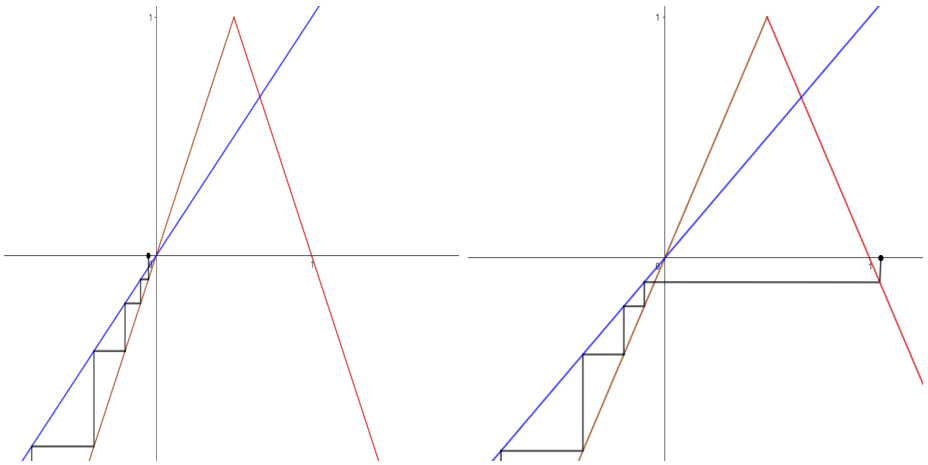
$$T(x) = \begin{cases} 2x & \text{si } x \leq 1/2 \\ 2(1-x) & \text{si } x > 1/2 \end{cases}$$

1. Si $x < 0$, entonces $O(x, T)$ es una sucesión decreciente que tiende a $-\infty$.
2. Si $x > 1$, entonces $T(x) < 0$; la órbita tiende a $-\infty$.

3. Si $x \in [0,1]$, entonces $T(x) \in [0,1]$ y además para todo $n \in \mathbb{N}$, $T^n(x) \in [0,1]$, es decir la órbita se queda en $[0,1]$, particularmente $x_1 = 0$ y $x_2 = \frac{2}{3}$ son puntos fijos.

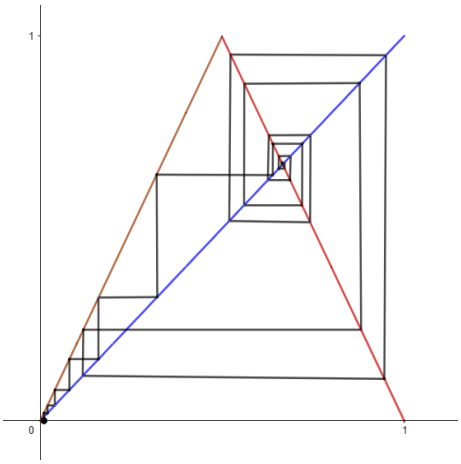
El resultado anterior puede verse con más detalle en [16, pp. 77-76].

Para ilustrar gráficamente el comportamiento de las órbitas de una función, generalmente se trazan gráficas de tipo telaraña; a continuación se ilustra el comportamiento de las órbitas del mapeo tienda.



(a) La órbita sucesión decreciente que tiende a $-\infty$.

(b) La órbita tiende a $-\infty$.



(c) La órbita se queda en el intervalo.

Figura 1.3: Mapeo Tienda.

Otros mapeos importantes son los relacionados con los modelos de crecimiento poblacional, que sirven para conocer la evolución de una determinada población. Entre estos mapeos se tiene el mapeo logístico

$$f(x) = \lambda x(1-x),$$

el cual surge de la ecuación diferencial propuesta por Pierre Francois Verhulst, que modela el crecimiento logístico de ciertas poblaciones:

$$\frac{dx}{dt} = \lambda x(1-x)$$

De manera sistemática ha sido estudiado el mapeo $f(x) = \lambda x(1-x)$ principalmente en $[0,1]$ y con parámetro $\lambda \in (0,4]$.

Por último, uno de los sistemas dinámicos discretos más clásicos e importantes es el mapeo del pastelero con factor 2, también conocido como mapeo peine como se menciona en [17, p. 67].

$$s(x) = \begin{cases} 2x & \text{si } x \in [0, \frac{1}{2}) \\ 2x-1 & \text{si } x \in [\frac{1}{2}, 1] \end{cases}$$

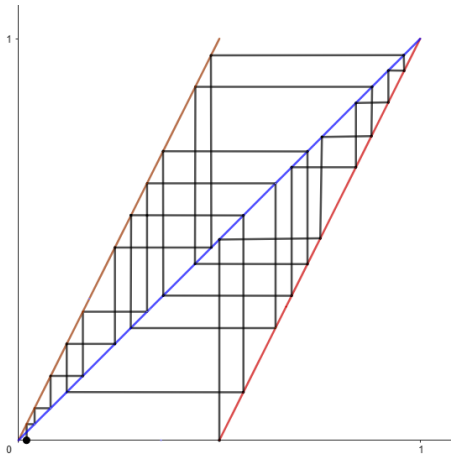


Figura 1.4: Mapeo Pastelero.

Este sencillo modelo, a pesar de su aparente simplicidad, sirve como punto de partida para comprender fenómenos más complejos en diversos campos. Su capacidad para exhibir un crecimiento exponencial rápido lo convierte en una herramienta valiosa para estudiar procesos de bifurcación y caos, conceptos fundamentales en la teoría de sistemas dinámicos.

En este trabajo serán importante las variantes del mapeo pastelero con factores 10 y 10,000, así como las composiciones de éstos con el mapeo logístico.

1.4. Órbitas estables y sensibilidad

En el apartado 1 se comentó que el estudio del comportamiento de las órbitas es el objeto de estudio de la teoría relativa a los sistemas dinámicos discretos. De manera particular son de interés las propiedades caóticas de las órbitas. Aunque no existe una definición universal de caos, se considera que un sistema es caótico si presenta desorden de alguna manera. Algunos de los conceptos ligados al caos son la sensibilidad a las condiciones iniciales, la entropía y la aleatoriedad.

Igual que antes X denota a un espacio métrico con métrica d. Particularmente se utilizan los espacio métricos $X = \mathbb{R}$ y $X = [0, 1]$, con la métrica definida por $d(a, b) = |a - b|$.

Dados $f : X \to X$ una función y x_0 un punto periódico con periodo $n \in \mathbb{N}$, se nota que x_0 se traslada de manera ordenada a través de un conjunto de puntos n.

$$\{x_0, f(x_0), f^2(x_0), ..., f^{n-1}(x_0)\}$$

Una tarea importante es conocer el comportamiento de los puntos cercanos al punto x_0. Particularmente se estudian las órbitas de puntos cercanos a x_0 y se compara con la órbita de x_0.

Los puntos cercanos a x_0 no siempre tienen órbitas convergentes a x_0. Se dice que un punto fijo x_0 no es estable, si existe $\epsilon > 0$ tal que para toda $\delta > 0$ existen x y $n \in \mathbb{N}$ tales que $d(x, x_0) < \delta$ y

$$d(f^n(x), x_0) \geq \epsilon.$$

Para describir órbitas estables se define en primer instancia la existencia de la bola.

Sea $X = (X, d)$ un espacio métrico. Dados $x \in X$ y $\epsilon > 0$, se define la bola de radio ϵ con centro en x por:

$$B(x, \epsilon) = \{y \in X : d(x, y) < \epsilon\}$$

Definición 1.4.1 *[8, p. 127] Sea $f : X \to X$ una función continua en un espacio métrico X. Se dice que un punto x_0 de X tiene órbita estable, o tiene órbita Lyapunov estable, si para toda $\epsilon > 0$, existe $\delta > 0$, tal que para toda $x \in B(x_0, \delta)$ y para toda $m \geq 0$ se tiene que:*

$$d(f^m(x), f^m(x_0)) < \epsilon.$$

Por otro lado, si existe $\epsilon_0 > 0$ tal que para todo $\delta > 0$, para el cual es posible encontrar un punto $y \in B(x_0, \delta)$ y un número natural n, que depende de y, tales que

$$d(f^n(x_0), f^n(y)) \geq \epsilon_0$$

se dice que la órbita de $x_0 \in X$ no es estable.

La estabilidad está relacionada estrechamente con el concepto de sensibilidad; así se denota que la sensibilidad se interpreta como la no estabilidad (ausencia de estabilidad) en cada punto $x_0 \in X$. En la definición siguiente se plasma esta idea.

Definición 1.4.2 *[8, p. 131] Sea $f : X \to X$ una función. Se dice que f es sensible a las condiciones iniciales en X, si existe un valor $\epsilon > 0$, fijo, tal que para toda $x \in X$, y para toda $\delta > 0$, existen $y \in B(x, \delta)$ y $m \in \mathbb{N}$ tales que:*

$$d(f^m(x), f^m(y)) \geq \epsilon$$

Al número ϵ se le llama constante de sensibilidad de f. En la Figura 1.5 se puede ver su comportamiento geométricamente.

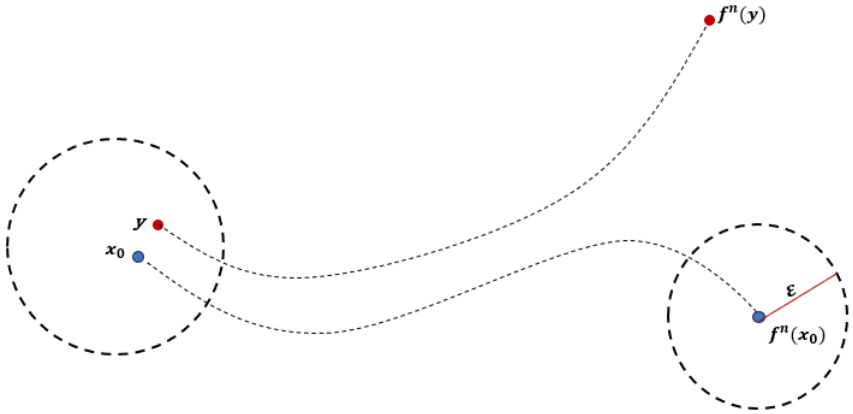


Figura 1.5: Sensibilidad a las condiciones iniciales.

La transitividad es otro concepto fundamental en el estudio de los sistemas dinámicos discretos. Este concepto se refiere a la capacidad de un sistema para transferir el estado de un elemento a las cercanías de otro estado dentro del sistema.

Definición 1.4.3 *[14] Sean X un espacio métrico y $f : X \to X$ una función continua en X. Se dice que f es topológicamente transitiva en X (o transitiva en X) si para todo par de conjuntos abiertos no vacíos de X, por ejemplo U y V, existe $n \in \mathbb{N}$ tal que*

$$f^n(U) \cap V \neq \emptyset$$

.

En la Figura 1.4 se muestra geométricamente el comportamiento de un punto de U que llega hasta V en la iteración f^n.

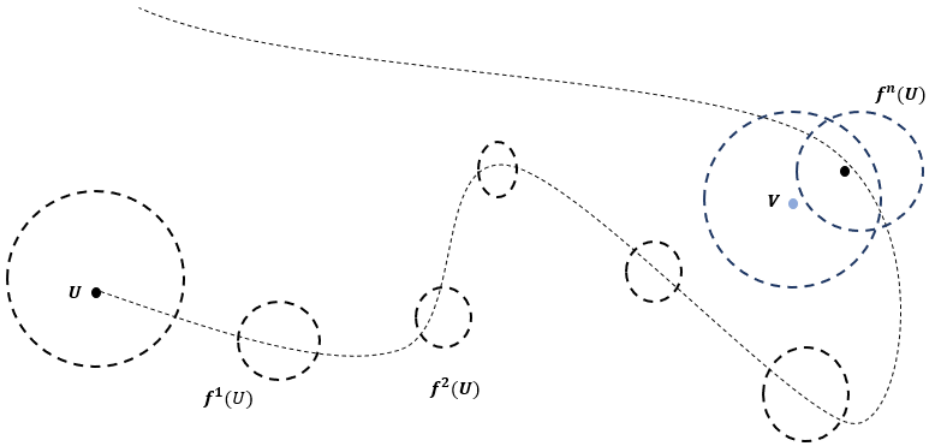


Figura 1.6: Transitividad.

Uno de los conceptos más estudiados referentes al caos es del célebre matemático Robert Luke Devaney, que describe la presencia de gran cantidad de órbitas cuyos puntos generan movimientos complejos, a continuación se dará algunos resultados. En este trabajo se busca indicar las propiedades caóticas descritas por Devaney.

Antes de abordar la definición de una función Devaney caótica, es necesario introducir la definición de conjunto denso, el cual establece que:

Proposición 1.4.1 *Sea (X, d) un espacio métrico, y $D \subseteq X$, entonces*

$$D \subseteq X \text{ denso} \Leftrightarrow \forall \mathcal{U} \in \Im \setminus \{\emptyset\}$$

Es decir, un conjunto denso "encuentra" todos los conjuntos abiertos no vacíos.

Véase la demostración en [7, p. 75]

Definición 1.4.4 *[14] Se dice que $f : X \to X$ es una función Devaney caótica, o que genera un sistema dinámico caótico, en X, si se cumplen las siguientes tres condiciones:*

- *El conjunto $Per(f)$ forma un conjunto denso en X*
- *f es transitiva en X*
- *f es sensible a las condiciones iniciales en X*

Se han retomado los resultados más relevantes de las referencias correspondientes. Si bien no se incluyen las demostraciones de dichos resultados, se cita la fuente original en cada caso.

El resultado siguiente muestra que la tercera condición en la definición del caos de Devaney es consecuencia de las dos primeras.

Teorema 1.4.1 *[4] Si la función continua $f : X \to X$ es transitiva en X y el conjunto $Per(f)$ es denso en X, entonces f es sensible a las condiciones iniciales en X.*

En este trabajo se utilizan las funciones de $[0, 1]$ en $[0, 1]$ por lo cual los siguientes dos resultados cobran importancia. El Teorema 3 indica que para funciones en intervalos en $\mathbb{R}$ es suficiente la transitividad para obtener el caos de Devaney.

Teorema 1.4.2 *[22] Sea A un intervalo cerrado en $\mathbb{R}$.*
Sea $f : A \to A$ una función transitiva en A. Entonces $Per(f)$ es un conjunto denso en A.

Es interesante saber que para intervalos cerrados en $\mathbb{R}$ es suficiente la sensibilidad a las condiciones iniciales para obtener el caos, como se indica en el teorema siguiente, sin embargo es fundamental aclarar a que se refiere con $int(B)$:

El interior de $B \subset A$ está formado por los $x \in A$ para los cuales existe $\delta > 0$ tal que $(x - \delta, x + \delta$ está contenido en B.

Teorema 1.4.3 *[15]*
Sea $A = [a, b]$ un intervalo cerrado en $\mathbb{R}$. Si $f : A \to A$ es sensible a las condiciones iniciales en A, entonces existen $B \subset A$, conjunto cerrado con $int(B) \neq \emptyset$, y $N \in \mathbb{N}$ tales que $f^N(B) = B$ y f^N, restringida a B, es caótica en B.

Capítulo 2

El mapeo logístico

En esta sección se analizará los puntos fijos generados por el mapeo logístico, así como una modificación de éste. Posteriormente se realiza algunas pruebas estadísticas a las órbitas generadas por los mapeos logísticos mencionados en el Apartado 1.3.

Para iniciar el análisis se buscan los puntos que determinen el estado de equilibrio del sistema, es decir, un valor que, una vez alcanzado, se mantiene en todas las iteraciones posteriores, mejor conocidos como puntos fijos. Para el mapeo logístico

$$f(x) = \lambda x(1-x), \text{ con } 1 \leq \lambda \leq 4, \text{ y } x \in [0,1].$$

se tiene que $x_1 = 0$ es un punto fijo ya que $f(0) = \lambda(0)(1-0) = 0$.

Del mismo modo para el caso $x \neq 0$ al igualar la función a x se tiene

$f(x) = \lambda x(1-x) = x,$

multiplicando ambos lados de la igualdad por x^{-1} se tiene que $\lambda(1-x) = 1$, al despejar se obtiene $x = 1 - \frac{1}{\lambda}$ por lo cual, el otro punto fijo corresponde a $x_2 = 1 - \frac{1}{\lambda}$, el cual está en $(0,1)$, si $\lambda > 1$. Por lo anterior se usa a $\lambda \geq 1$ con la igualdad es cerrado por la derecha.

Ahora se analiza el comportamiento de los puntos fijos x_1, x_2, es decir si corresponden a un punto repulsor o atractor, aplicando lo mencionado en el Capítulo 1 respecto a $|f'(x)|$. Se denota que

$f(x) = \lambda x(1-x) = \lambda x - \lambda x^2,$ y
$f'(x) = (\lambda x - \lambda x^2)' = \lambda - 2\lambda x$

Con $x_1 = 0$ se obtiene:

$|f'(x_1)| = |\lambda - 2\lambda x_1| = \mid \lambda - 2\lambda(0) \mid = |\lambda| = \lambda$

$|f'(0)| = |\lambda| = \lambda > 1$, para $\lambda > 1$.

Con el punto fijo $x_2 = 1 - \frac{1}{\lambda}$ se tiene

$|f'(x_2)| = |f'(1 - \frac{1}{\lambda})| = |\lambda - 2\lambda(1 - \frac{1}{\lambda})| = |\lambda - 2\lambda + 2|$

$|f'(1 - \frac{1}{\lambda})| = |\lambda - 2| > 1$, si $\lambda > 3$ y

$|f'(1 - \frac{1}{\lambda})| < 1$, si $\lambda \in (1, 3)$

Se concluye que $x_1 = 0$ es repulsor si $\lambda \in (1, 4] < 1$, por otra parte x_2 es atractor si $\lambda \in (1, 3)$ y repulsor si $3 < \lambda \leq 4$. Con lo anterior se muestra que $\lambda \in (1, 4]$, entonces $x_1 = 0$ es repulsor. Por otra parte x_2 es atractor si $\lambda \in (1, 3)$ y repulsor si $\lambda \in (3, 4]$ □

Con los puntos fijos $x_1 = 0$, $x_2 = 1 - \frac{1}{\lambda}$, y $\lambda = 3.9999$, se tiene que:
$x_1 = 0$, $x_2 = 1 - \frac{1}{3.9999} \approx 0.749993$, entonces

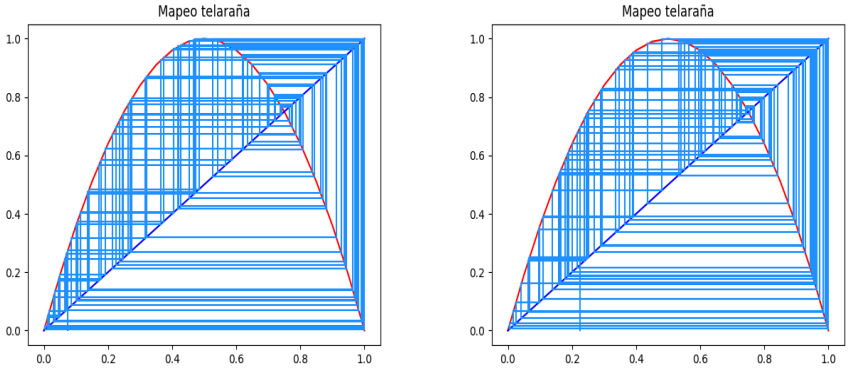
$|f'(x)| = |\lambda - 2\lambda x|$

$|f'(x_1)| = |3.9999 - 2(3.9999)(0)| = 3.9999 > 1$

$|f'(x_2)| = |3.9999 - 2(3.9999)(0.749993)| = 1.9999 > 1$

lo que confirma lo anterior.

Se puede analizar el comportamiento de las órbitas de los puntos mencionados anteriormente mediante un diagrama de telaraña. En las siguientes gráficas se nota, en ambos casos, $x_1 = 0$ y $x_2 = 1 - \frac{1}{3.9999}$ claramente que son repulsores.



(a) $x_0 = 0.074135$ (b) $x_0 = 0.22559436$

Figura 2.1: Diagrama de telaraña del mapeo logístico con $\lambda = 3.9999$ y $n = 100$ iteraciones.

Ahora se analizan las propiedades de los puntos fijos y de los puntos con periodo 2. Para encontrar puntos de periodo 2 es necesario analizar, en $\mathbb{R}$, la ecuación

$$f^2(x) = f(f(x)) = x$$

Para la función logística $f(x) = \lambda x(1-x)$, la segunda iteración corresponde a:

$$f(f(x)) = f\left(\lambda x(1-x)\right) = \lambda\lambda x(1-x)[1-\lambda x(1-x)].$$

Dado que $x = 0$ es punto fijo, se considera $x \neq 0$, así la ecuación

$$f(f(x)) = \lambda\lambda x(1-x)[1-\lambda x(1-x)] = x$$

implica, después de eliminar $x \neq 0$,

$$\lambda^2(1-x)[1-\lambda x(1-x)] - 1 = 0 \Longrightarrow \lambda^2(1-x) - \lambda^3 x(1-x)^2 - 1 = 0.$$

Al desarrollar y simplificar se tiene:

$$\lambda^2 - \lambda^2 x - \lambda^3 x(1 - 2x + x^2) - 1 = 0$$

$$\lambda^2 - \lambda^2 x - \lambda^3 x + 2\lambda^3 x^2 - \lambda^3 x^3 - 1 = 0$$

$$-\lambda^3 x^3 + 2\lambda^3 x^2 - \lambda^2 x - \lambda^3 x + \lambda^2 - 1 = 0.$$

Puesto que $x_2 = 1 - \frac{1}{\lambda}$ es solución de la ecuación, se tiene que existe un polinomio cuadrático $Q(x)$ tal que

$$-\lambda^3 x^3 + 2\lambda^3 x^2 - \lambda^2 x - \lambda^3 x + \lambda^2 - 1 = [x - (1 - \tfrac{1}{\lambda})]Q(x) = [x - 1 + \tfrac{1}{\lambda}]Q(x)$$

es claro que

$$Q(x) = \frac{-\lambda^3 x^3 + 2\lambda^3 x^2 - \lambda^2 x - \lambda^3 x + \lambda^2 - 1}{x - 1 + \frac{1}{\lambda}}.$$

Obteniendo la forma de $Q(x)$ mediante la división sintética, se obtiene:

$$\begin{array}{r|l}
 & \begin{array}{lll} -\lambda^3x^2 & +(\lambda^3+\lambda^2)x & -(\lambda^2+\lambda) \end{array} \\
x-1+\frac{1}{\lambda} & \begin{array}{lll} |-\lambda^3x^3 & +2\lambda^3x^2 & -\lambda^2x \\ \lambda^3x^3 & -\lambda^3x^2 & +\lambda^2x^2 \end{array} \quad \begin{array}{lll} -\lambda^3x & +\lambda^2 & -1 \end{array} \\
\hline
 & \begin{array}{lll} \lambda^3x^2 & +\lambda^2x^2 & -\lambda^2x \\ -\lambda^3x^2 & -\lambda^2x^2 & +\lambda^3x \end{array} \quad \begin{array}{lll} -\lambda^3x & +\lambda^2-1 & \\ +\lambda^2x & -\lambda^2x & -\lambda x \end{array} \\
\hline
 & -\lambda^2x \quad -\lambda x \quad +\lambda^2 \quad -1+\lambda^2x+\lambda x-\lambda^2-\lambda+\lambda+1=0 \quad .
\end{array}$$

Al resolver, en $\mathbb{R}$, la ecuación

$$Q(x) = -\lambda^3 x^2 + (\lambda^3 + \lambda^2)x - (\lambda^2 + \lambda) = 0$$

se obtienen los puntos con periodo 2. Primero se nota que si $1 < \lambda < 3$ se concluye que las soluciones no son reales, en efecto:

$$x_{3,4} = \frac{-\lambda^3 - \lambda^2 \pm \sqrt{\lambda^6 + 2\lambda^5 + \lambda^4 - 4\lambda^5 + 4\lambda^4}}{-2\lambda^3}$$

$$x_{3,4} = \frac{-\lambda^3-\lambda^2\pm\sqrt{\lambda^6-2\lambda^5-3\lambda^4}}{-2\lambda^3} = \frac{-\lambda^3-\lambda^2\pm\sqrt{\lambda^2-2\lambda-3}}{-2\lambda^3}$$

$$x_{3,4} = \frac{-\lambda-1\pm\sqrt{(\lambda+1)(\lambda-3)}}{-2\lambda}.$$

Nótese que $\lambda^2-2\lambda-3=(\lambda-3)(\lambda+1)>0$, si $\lambda\in(-\infty,-1]\cup[3,\infty)$, por lo cual las soluciones son reales, si $\lambda\in(-\infty,-1]\cup[3,\infty)$. En contraste, si $\lambda\in(-1,3)$, no tiene soluciones en $\mathbb{R}$. Dado que se considera $\lambda\in[1,4]$, se concluye que $\{x_3,x_4\}$ define una órbita de tamaño 2, si $\lambda\in(3,4]$. Lo anterior debido a que x_3 y x_4 no son puntos fijos, es decir $f(x_3)\neq x_3$ y $f(x_4)\neq x_4$.Además $f^2(x_3)=x_3$ y $f^2(x_4)=x_4$, lo cual sólo da una opción $f(x_3)=x_4$ y $f(x_4)=x_3$.

Para conocer el comportamiento de los puntos de periodo 2, y de sus órbitas, se consideran las raíces de la ecuación polinomial de grado 4 calculadas anteriormente

$$x_3 = \frac{-\lambda-1+\sqrt{(\lambda+1)(\lambda-3)}}{-2\lambda},\ x_4 = \frac{-\lambda-1-\sqrt{(\lambda+1)(\lambda-3)}}{-2\lambda}$$

al evaluar la derivada de $f(x)$ en x_3 y x_4 se tiene

$$f'(x)=\lambda-2\lambda x$$

$$f'(x_3)=\lambda-2\lambda\frac{-\lambda-1+\sqrt{(\lambda+1)(\lambda-3)}}{-2\lambda}$$

$$f'(x_3)=\lambda-\lambda-1+\sqrt{(\lambda+1)(\lambda-3)}$$

análogamente

$$f'(x_3)=-1+\sqrt{(\lambda+1)(\lambda-3)}$$

$$f'(x_3)=-1-\sqrt{(\lambda+1)(\lambda-3)}.$$

Por la regla de la cadena, al derivar $f_\lambda^2=f_\lambda\circ f_\lambda$, se tiene

$$[f_\lambda(f_\lambda(x_3))]'=f_\lambda'(f_\lambda(x_3))f_\lambda'(x_3)=f'(x_4)f'(x_3)$$

$$\begin{aligned} f'(x_4)f'(x_3) &= \left(-1+\sqrt{(\lambda+1)(\lambda-3)}\right)\left(-1-\sqrt{(\lambda+1)(\lambda-3)}\right)\\ &= 1-(\lambda^2-2\lambda-3)=1-\lambda^2+2\lambda+3\\ &= 4-\lambda^2+2\lambda=-(\lambda^2-2\lambda-4)\end{aligned}$$

al igualar a 0 y resolver la ecuación equivalente $\lambda^2-2\lambda-4=0$

se tiene $\lambda_{1,2}=\frac{2\pm\sqrt{4+16}}{2}=\frac{2\pm2\sqrt{5}}{2}=1\pm\sqrt{5}$

Si $\lambda=3$, entonces

$$|f_3'(x_3)|=|\lambda^2-2\lambda-4|=|3^2-2(3)-4|=1,$$

y para $\lambda=1+\sqrt{5}$

$|f'_{1+\sqrt{5}}(x_3)| = |(1+\sqrt{5})^2 - 2(1+\sqrt{5}-4| = |1+2\sqrt{5}+\sqrt{5}^2 - 2 - 2\sqrt{5} - 4| = 0$
se obtiene un punto súper atractor.

Proposición 2.0.1 *[8, p. 226] La órbita $\{x_3, x_4\}$, de periodo 2, es atractora en el intervalo abierto (λ_1, λ_2) y repulsora si $\lambda > \lambda_2$, donde $\lambda_1 = 3$ y $\lambda_2 = 1 + \sqrt{6}$.*

Es necesario analizar

$|[f^2(x_{3,4})]'| = | - (\lambda^2 - 2\lambda - 4)| = |\lambda^2 - 2\lambda - 4|$ para esto se define

la función $g(\lambda) = \lambda^2 - 2\lambda - 4$, cuya gráfica es una parábola que abre hacia la ordenada positiva con vértice en $V = (\frac{-b}{2a}, g(\frac{-b}{2a})) = (1, g(1)) = (1, -5)$. Es fácil ver que $g(\lambda)$ es creciente en $(3, 4]$. Note que el interés es sobre el comportamiento de la función cuando $\lambda \in (3, 4]$. Se ve dónde $|g(\lambda)| < 1$ y dónde $|g(\lambda)| > 1$. Se nota que $g(3) = 3^2 - 2(3) + 4 = 9 - 6 - 4 = -1$.

Resolviendo la ecuación $g(\lambda) = \lambda^2 - 2\lambda - 4 = 1$, la cual es equivalente a $\lambda^2 - 2\lambda - 4 - 1 = \lambda^2 - 2\lambda - 5 = 0$

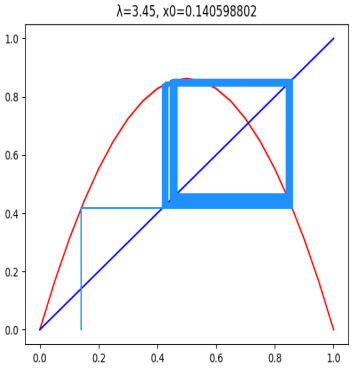
al aplicar la fórmula general se tiene

$\lambda = \frac{2 \pm \sqrt{4+20}}{2} = \frac{2 \pm 2\sqrt{6}}{2} = \frac{2(1 \pm \sqrt{6})}{2} = 1 \pm \sqrt{6}$

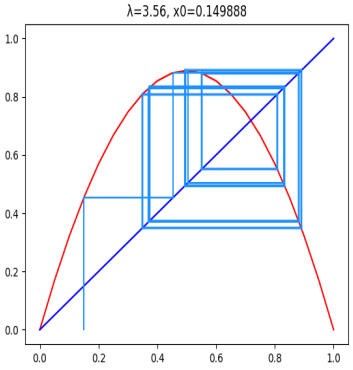
al separar las raíces $\lambda = 1 - \sqrt{6} < 0$, $\lambda = 1 + \sqrt{6} \approx 3.44949 \in (3, 4]$ □

Por lo tanto para el intervalo $(3, 1 + \sqrt{6})$, la órbita $\{x_3, x_4\}$, con periodo 2, es atractora.

Al aumentar el valor del parámetro λ dentro del intervalo $(3, 4]$, se observa la aparición de una órbita atractora de orden $2^2 = 4$. Simultáneamente, la órbita de orden 2 $\{x_3, x_4\}$, que previamente era atractora, se transforma en repulsora. Este cambio marca el inicio de una serie de bifurcaciones sucesivas, donde la órbita atractora duplica su tamaño en cada bifurcación. Las siguientes imágenes ilustran un fenómeno, mostrando órbitas atractoras de tamaño 2^2, 2^3, 2^4 y 2^5 para valores específicos de λ.

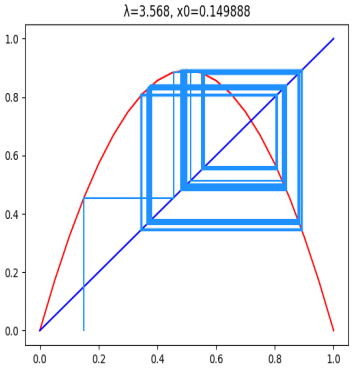


(a) Órbita expulsora de periodo 2 y una atractora de periodo 4

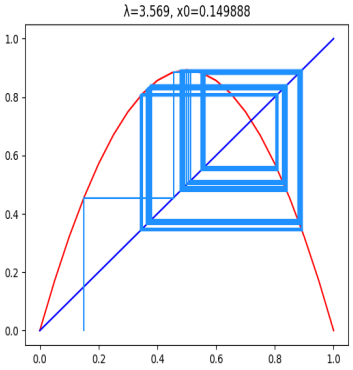


(b) Órbitas expulsoras de periodos 2 y 4 con una atractora de periodo 8

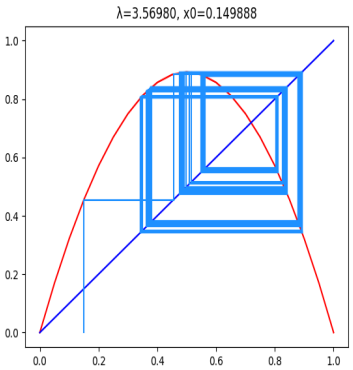
Figura 2.2: Órbitas expulsoras y atractoras de orden 2^n



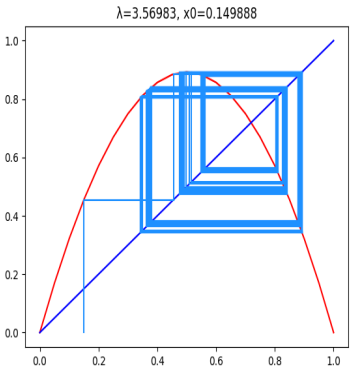
(a) Órbita atractorade periodo 16



(b) Órbita atractorade periodo 32



(c) Órbita atractora deperiodo 64



(d) Órbita atractorade periodo 128

Figura 2.3: Órbitas atractoras de orden 2^n

Después de que se agotan todas las órbitas de orden 2^n para todo $n \in \mathbb{N}$, aparece una órbita atractora de periodo 3. Por el Teorema de Sharkovskii se concluye que existirán órbitas de orden k para todo $k \in \mathbb{N} \setminus \{3\}$, las cuales serán repulsoras. También aparecerán una serie de bifurcaciones a partir de la órbita atractora de orden 3. Lo anterior hace que el comportamiento de las órbitas sea cada vez más caótico.

2.1. Mapeo logístico modificado

Se han propuesto diversas modificaciones del mapeo logístico, principalmente componiendo con funciones trigonométricas, mapeos de Gauss, así como algunos módulos, lo anterior con el fin de obtener mapeos con mejores propiedades caóticas. En esta sección se describe una propuesta de modificación, con su respectivo análisis.

El mapeo del pastelero, con factor 10, es la función $h : [0,1] \to [0,1]$ que se define por $h(x) = 10x - [10x]$, donde $[a]$ es la parte entera de a. Es fácil ver que $h(1) = 0$, y además

$h(x) = 10x - k$, si $x \in [\frac{k}{10}, \frac{k+1}{10})$ para $k = 0, 1, 2, 3, \ldots, 9$.

Los puntos fijos de $h(x)$ son las soluciones de la ecuación $h(x) = x$

$10x - k = x \Rightarrow 9x = k \Rightarrow x = \frac{k}{9}$, para $k = 0, 1, 2, , \ldots, 9$.

Se considera ahora el mapeo logístico modificado con factor 10, mismo que es la composición del mapeo logístico $g(x) = 3.9x(1-x)$ con el mapeo del pastelero con factor 10

$f(x) = h(f(x)) = h(3.9x(1-x)) = 39x(1-x) - [39x(1-x)]$.

Obteniendo los puntos fijos de $f(x)$, mismos que son soluciones de la ecuación $f(x) = x$, es decir soluciones de las ecuaciones

$39x(1-x) - k = x \Rightarrow 39x^2 - 38x + k = 0$, para $k = 0, 1, 2, , \ldots, 9$.

Para cada k se resuelve la ecuación cuadrática correspondiente

$x^k_{1,2} = \frac{38 \pm \sqrt{(38)^2 - 4(39)k}}{78} = \frac{38 \pm \sqrt{1444 - 156k}}{78}$.

El discriminante $1444 - 156k$ es mayor que 0, para cada k, por lo cual se obtienen 20 soluciones reales distintas, las cuales están en el intervalo $[0, 1)$.

$39999x(1-x)-k=x \Rightarrow 39999x^2-39998x+k=0$, para $k=0,1,2,,\ldots,9999$

$x_{1,2}^k=\frac{39998\pm\sqrt{(39998)^2-4(39999)k}}{79998}=\frac{39998\pm\sqrt{159984004-159996k}}{79998}$.

El discriminante $15984004-15996k$ es mayor que 0, para cada k, por lo cual se obtienen 20000 soluciones reales distintas, las cuales están en el intervalo $[0,1)$.

$f(x)=10\lambda x(1-x)-[k]$, donde [k] es la parte entera.

Con $\lambda=3.9$ y aplicando el factor 10

$39x(1-x)-k=x \Rightarrow 39x^2-38x+k=0$, para $k=0,1,2,,\ldots,9$

los puntos fijos determinados por:

$x_{1,2}^k=\frac{38\pm\sqrt{(38)^2-4(39)k}}{78}=\frac{38\pm\sqrt{1444-156k}}{78}$, al sustituir k

$x_{1,2}^0=[x_1^0=0, x_2^0=0.97435897435897]$

$x_{1,2}^1=[x_1^1=0.027067732181375, x_2^1=0.9472912421776]$

$x_{1,2}^2=[x_1^2=0.055830670736905, x_2^2=0.91852830362207]$

$x_{1,2}^3=[x_1^3=0.086653862774018, x_2^3=0.88770511158496]$

$x_{1,2}^4=[x_1^4=0.1200558701724, x_2^4=0.85430310418657]$

$x_{1,2}^5=[x_1^5=0.15681798136602, x_2^5=0.81754099299295]$

$x_{1,2}^6=[x_1^6=0.19821980334398, x_2^6=0.77613917101499]$

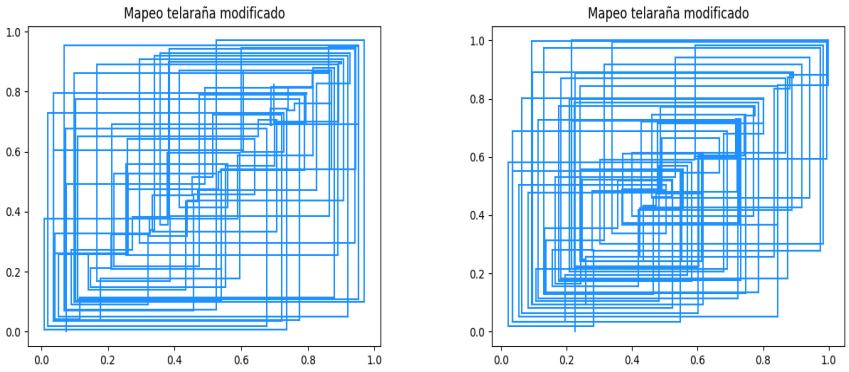
$x_{1,2}^7=[x_1^7=0.24664534565008, x_2^7=0.72771362870889]$

$x_{1,2}^8=[x_1^8=0.30769230769231, x_2^8=0.66666666666667]$

$x_{1,2}^9=[x_1^9=0.40609544461107, x_2^9=0.56826352974791]$

Los 20 puntos fijos se determinan con el programa Máxima[1]. A continuación el diagrama de telaraña del mapeo modificado.

[1]Sistema algebraico computacional con capacidad para cálculo simbólico y numérico de alta precisión, álgebra lineal y visualización gráficas.



(a) $x_0 = 0.074135$ (b) $x_0 = 0.22559436$

Figura 2.4: Diagrama de telaraña del mapeo logístico modificado con $\lambda = 3.9999$ y $n = 100$ iteraciones.

En contraste con lo obtenido con el mapeo logístico normal, en el diagrama de telaraña no se identifica a simple vista el comportamiento en torno a los puntos fijos. Lo que claramente se muestra un flujo más desordenado. Como ya se mencionó el análisis de los puntos fijos del mapeo modificado con el mapeo del pastelero sería muy largo, ya que se obtiene 20,000 puntos fijos, por lo que se realizó el estudio con factor 10.

2.2. Pruebas estadísticas

El análisis de la aleatoriedad y la uniformidad de las órbitas generadas por los mapeos logísticos mediante la aplicación de pruebas paramétricas y no paramétricas. Con el objetivo de determinar si las secuencias son uniformes y aleatorias en el intervalo $[0, 1]$.

- **Prueba de bondad de ajuste χ^2 (ji-cuadrada)**

  En el libro [23, p. 714] se establece que la prueba de ji-cuadrada busca determinar si los elementos de una población Q se distribuyen de acuerdo a cierta distribución, lo anterior a partir de un conjunto de n datos $\{r_i : i = 1, ..., n\} \subset Q$. En este trabajo se considera la distribución uniforme en el intervalo $[0, 1]$. Para esta prueba en principio se debe dividir el intervalo $[0, 1]$ en m sub-intervalos, con ayuda del número entero $m \approx \sqrt{n}$. Posteriormente se determina a cuál sub-intervalo pertenece cada r_i.

  Se considero a continuación las hipótesis nula H_0 y alterna H_1:

  $H_0 =$ Los datos están distribuidos uniformemente en $[0, 1]$.

  $H_1 =$ Los datos no están distribuidos uniformemente en $[0, 1]$.

  El estadístico de prueba a utilizar es "ji cuadrada".

  $\chi^2 = \sum_{i=1}^{m} \frac{(O_i - e_i)^2}{e_i}$,

  donde O_i y e_i representan, respectivamente, la frecuencias observada y esperada en la clase i. El rechazo o no rechazo de la hipótesis, como es usual, depende del nivel de significancia y, por lo tanto de la localización del estadístico de prueba respecto al valor crítico o valores críticos, mismo que depende de la significancia.

  Con el uso, cada vez más frecuente de las computadoras el valor-p, (p-value) cobra relevancia. Si el valor-p es menor que α, con α determinada previamente (generalmente con $\alpha = 0.05$ o $\alpha = 0.01$), se rechaza la hipótesis nula a favor de la alterna, en caso contrario no se rechaza la hipótesis nula, que establece, en esta prueba, la uniformidad en $[0, 1]$.

- **Prueba de rachas (Aleatoriedad)**

  Se considera racha a la secuencia de valores con una característica común precedida y seguida por valores que no presentan esa característica, generalmente se omiten los valores iguales a la mediana. Por ejemplo en la secuencia 11110001011 existen 5 rachas, 1111, la primera, 000, la segunda, 1, la tercera, 0, la cuarta y, finalmente la racha 11.

  La prueba de rachas determina, con cierto nivel de confianza, si es aleatorio el orden de aparición de los valores de una variable, en dos clases. En este trabajo se utilizan números en $[0, 1]$, por lo cuál se dividen los valores en dos clases, determinadas por ser mayor menor que la mediana. Cabe mencionar que lo anterior se puede aplicar a la aparición de valores respecto a cualquier otro punto de referencia. Finalmente se enlistan los valores de la muestra de acuerdo con el orden de aparición y se cuentan las rachas (véase el libro [23, p. 777]).

  Ahora se considero un número natural n y una secuencia o serie $\{x_k : k = 1, 2, \ldots, n\}$ de elementos en $[0, 1]$. Se define otra secuencia $y_1, y_2, \ldots, y_n$ donde cada y_k toma valores 0 o 1, de acuerdo a la regla siguiente. Para cada $k = 1, 2, \ldots, n$ se define $y_k = 0$, si $x_k \in [0, mediana)$, por otra parte $y_k = 1$, si $x_k \in [mediana, 1]$. En esta definición es claro que se considera la mediana de la serie original. Se define R como de número de rachas, de 0's y 1's, en la secuencia $\{y_k : k = 1, 2, \ldots, n\}$, Es importante comentar que no se puede alterar el orden de aparición de las secuencias $\{x_k : k = 1, 2, \ldots, n\}$ y $\{y_k : k = 1, 2, \ldots, n\}$.

  En esta prueba las hipótesis nula H_0 y alterna H_1 son:

  $H_0 =$ Existe aleatoriedad.

  $H_1 =$ No existe aleatoriedad.

  El estadístico de prueba a usar es

  $z = \frac{R - E(R)}{\sqrt{var(R)}}$, se usa la distribución normal estándar Z.

  $E(R) = \frac{2n_1n_2}{n_1+n_2} + 1$

  $n_1 =$ número de observaciones en el intervalo $[0, mediana) =$ cantidad de 0's

n_2 = número de observaciones en el intervalo $[mediana, 1]$ = cantidad de 1's

$var(R) = \frac{2n_1n_2(2n_1n_2-n_1-n_2)}{(n1+n2)^2(n_1+n_2-1)}$ (varianza).

Normalmente las pruebas se determinan mediante el

valor-$p = 2P(Z > z)$, donde $Z \sim N(0, 1)$.

Aunque también se pueden utilizar los valores críticos $\pm z_c$.

Para ampliar la información sobre el tema, se recomienda consultar la referencia [23, p. 778]. Con el fin de facilitar el análisis estadístico de la gran cantidad de datos, se ha empleado el programa R. Un aspecto fundamental en este análisis es la determinación del valor-p, el cual permite evaluar la hipótesis nula en función de un nivel de insignificancia α preestablecido. Para evaluar la hipótesis nula, se considera una significancia con valor fijo α. Generalmente se considera $\alpha = 0.05$ o $\alpha = 0.01$ como valores de referencia para las pruebas de hipótesis. Se rechaza la hipótesis nula si valor-$p < \alpha$, en caso contrario no se rechaza la hipótesis nula.

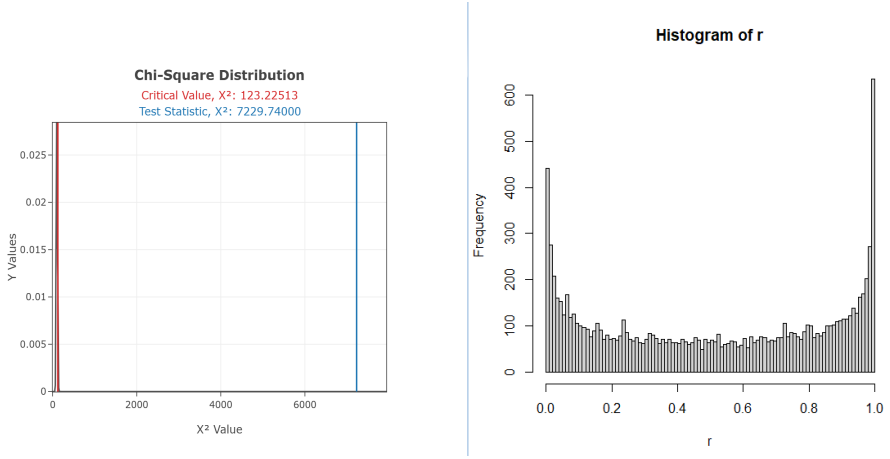
A continuación se ilustra, con un ejemplo, los resultados obtenidos al aplicar el mapeo logístico. Cabe mencionar que las pruebas fueron realizadas a 100 puntos semilla, o iniciales, diferentes con el mapeo logístico

$$f(x) = \lambda x(1 - x),\ x_0 \in (0, 1) \text{ y } \lambda = 3.9999.$$

Para cada uno de los puntos semilla, se obtuvo el rechazo de la hipótesis nula H_0, en favor de hipótesis alterna que establece la no uniformidad. Este resultado se obtuvo con un valor-p de 2.2×10^{-16}, lo que implica una probabilidad de error al rechazar la hipótesis nula, siendo esta verdadera, inferior a 10^{-15}.

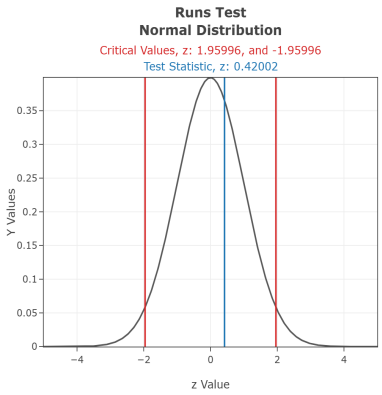
Por otro lado para la hipótesis nula H_0, sobre la existencia de aleatoriedad, los valores-p fueron mayores a 0.01, con lo cual se dictamina el no rechazó de la hipótesis H_0.

En el ejemplo con punto semilla $x_0 = 0.074135$ la prueba de uniformidad arrojó un valor-$p = 2.2 \times 10^{-16}$ (véase 2.5(a)); de acuerdo a los parámetros ya mencionados se rechaza la hipótesis nula en favor de la alterna que establece la no uniformidad en $[0, 1]$.



(a) La línea de aceptación se encuentra muy lejos de la zona de no rechazo

(b) Histograma de distribución con 100 clases



(c) Dentro del intervalo de no rechazo se encuentra la línea de aceptación

Figura 2.5: Simulaciones obtenidas en *Statdisk* y en *R*.

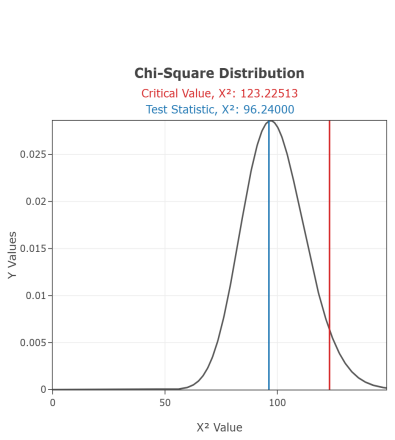
Como referencia del comportamiento de la distribución de las clases en la Figura 2.5(b), se observa un acumulado de puntos en las clases de los extremos.

Por otra lado, la hipótesis nula H_0, que postula la aleatoriedad de los datos, no se rechaza al obtener un valor-$p = 0.6745$. Este resultado indica que no hay evidencia suficiente para descartar la aleatoriedad en relación a la mediana. Véase la Figura 2.5(c).

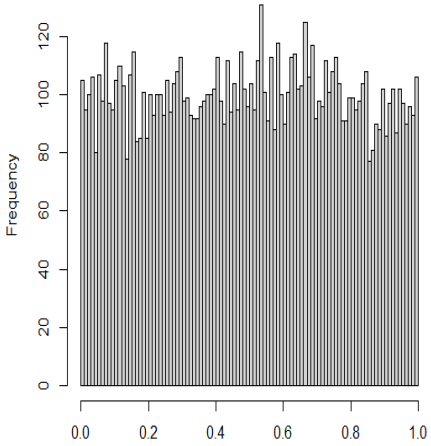
Por el contrario para el mapeo logístico modificado factor 10,000, con los mismos puntos semillas,

$$f(x) = 10,000\lambda x(1-x) - [10,000\lambda x(1-x)] \; x_0 \in (0,1) \text{ y } \lambda = 3.9999$$

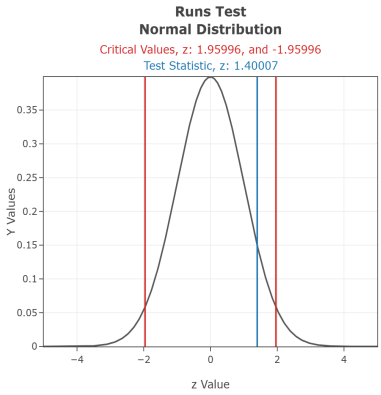
en ambas pruebas se obtuvieron valores-p favorables, es decir muy pequeños. Lo anterior implica el no rechazo a la pareja de ambas hipótesis H_0 (uniformidad-aleatoriedad). se nota un histograma totalmente distinto a lo anterior con una uniformidad más clara.



(a) valor-p = 0.5598 dentro de la zona de no rechazo



(b) Histograma de distribución con 100 clases



(c) valor-p= 0.1615 dentro de la zona de no rechazo

Figura 2.6: Simulaciones en *Statdisk* y *R*.

Nota: En este trabajo se usa el software *Statdisk* el cual es de uso libre para los usuarios del libro *Estadística* de Mario Triola [19, p. 53, 721].

2.3. Exponentes de Lyapunov

Una prueba estadística paramétrica de uso común para el análisis de las series de tiempo son los exponentes de Lyapunov, que determina si una

serie es sensible a las condiciones iniciales.

Sea una función $f : I \to I$, donde I es un intervalo. Los exponentes de Lyapunov definen a través de sus órbitas. Para cada $x_0 \in I$ se considera la órbita así como la derivada de $f(x)$ evaluada en cada punto de la órbita.

Para mayor información revise el trabajo The Lyapunov Exponent Test and the 0-1 Test for Chaos Compared [12]. Para determinar el exponente de Lyapunov de x_0 se construye la órbita.

$x_0, x_1 = f(x_0), x_2 = f(x_1), \cdots, x_n = f(x_{n-1}) = f^n(x_0)$

posteriormente se calcula la derivada en cada punto de la órbita, su valor absoluto, así como el logaritmo natural de lo anterior

$f'(x_k) \to |f'(x_k)| \to \ln(|f'(x_k)|)$

el exponente de Lyapunov x_0 se define por

$Lya(x_0) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} \ln |f'(x_k)|$

Los exponentes de Lyapunov reflejan la sensibilidad a las condiciones iniciales.

- Si $Lya(x_0) > 0$, con $x_0 \in I$, se tiene sensibilidad a condiciones iniciales.
- Si $Lya(x_0) < 0$, con $x_0 \in I$, no se tiene sensibilidad a condiciones iniciales.

Análisis de los puntos fijos.

Con anterioridad se establecieron los puntos fijos del mapeo logístico denotados por x_1 y x_2. A continuación se realiza el análisis de los exponentes de Lyapunov.

Para $x_1 = 0$, $x_2 = 1 - \frac{1}{\lambda}$ y $\lambda = 3.9999$ se tiene que:

$f(x) = 3.9999x - 3.9999x^2$

$f'(x) = 3.9999 - 7.9998x$

$f'(x_1) = 3.9999 - 7.9998(0)$

$f'(x_1) = 3.9999$

$\frac{1}{n} \sum_{k=0}^{n-1} |f'(x_k)| = \frac{1}{n} \sum_{k=0}^{n-1} \ln |3.9999| = \frac{1}{n} n \ln 3.9999 = \ln 3.9999$

al aplicar el límite de una constante,

$\lim_{n\to\infty} \ln 3.9999 = \ln 3.9999 \approx 1.386269$

Con $x_2 = 1 - \frac{1}{\lambda} \approx 0.749993$

$f'(x_2) = 3.9999 - 7.9998(0.749993)$

$f'(x_2) = -1.999390014$

$\lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} \ln |f'(x_k)| = \frac{1}{n} \sum_{k=0}^{n-1} \ln | - 1.999390014| = \ln 1.999390014 =$

$\lim_{n\to\infty} \ln 1.999390014 = \ln 1.999390014 \approx 0.692842$

La simulación mostrada en la Figura 2.7, recrea lo obtenido para 20000 diferentes $\lambda \in (1, 4)$. Para cada λ se aplican 100 iteraciones, con puntos iniciales aleatorios entre $x_0 \in (0, 1)$. En el apéndice se muestra el script del programa utilizado.

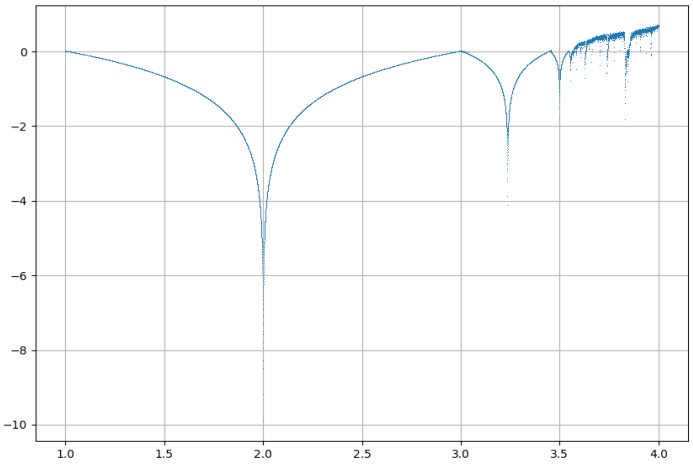


Figura 2.7: Coeficiente λ vs Exponentes de Lyapunov.

El teorema siguiente permite usar en calidad de punto inicial, $x = \frac{1}{2}$ sin importar el valor de λ.

Teorema 2.3.1 *Si la función cuadrática $f(x) = ax^2 + bx + c$ tiene una órbita periódica atractiva, entonces el punto crítico $-\frac{b}{2a}$ está en el conjunto estable de uno de los puntos de la órbita.*

$f(x) = \lambda x(1 - x) = \lambda x - \lambda x^2 = -\lambda x^2 + \lambda x$
$a = -\lambda$, $b = \lambda$

$$-\frac{b}{2a} = -\frac{\lambda}{2(-\lambda)} = \frac{1}{2}.$$

Note que para $\lambda > 3.5$ aún se observan regiones con valores Lya menores a 0; es decir, que para algunas $\lambda \in (3.5, 4)$ existen regiones o islas donde no hay sensibilidad a las condiciones iniciales. Lo anterior se ve mejor reflejado en el diagrama de bifurcación, ya que permite observar el cambio cualitativo que ocurre al cruzar valor o valores de los parámetros que aparecen en el sistema dinámico discreto.

La recreación del diagrama de bifurcación se muestra en la siguiente Figura 2.8, utilizando el mapeo logístico $f(x) = \lambda x(1 - x)$.

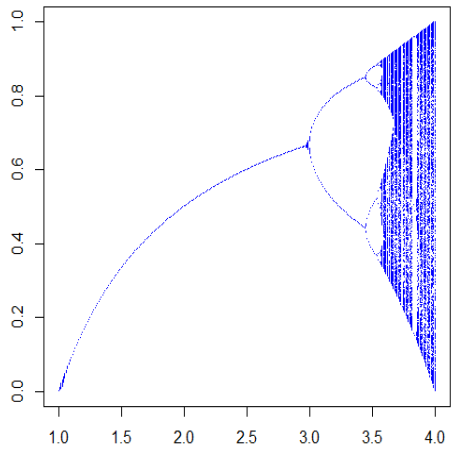


Figura 2.8: Bifurcación del mapeo logístico.

Se observa en el eje de las abscisas que $\lambda \in (1, 4)$.

En el primer segmento de $1 < \lambda < 3$ se puede ver que existe sólo un punto atractor, la proposición siguiente garantiza la existencia de éste:

Proposición 2.3.1 *[8] Para* $1 < \lambda < 3$ *el punto fijo* $x_\lambda = 1 - \frac{1}{\lambda}$ *es atractor. En este caso la cuenca de atracción de* x_λ *es todo el intervalo abierto* $(0, 1)$. *Para* $\lambda > 3$ *el punto fijo* x_λ *es repulsor.*

Lo que se observa de manera sencilla: Como $f'(x_\lambda) = 2 - \lambda$, mediante la desigualdad

$1 < \lambda < 3$

$-1 > -\lambda > -3$

$2 - 1 > 2 - \lambda > 2 - 3$

$1 > 2 - \lambda > -1$

$-1 < 2 - \lambda < 1 \Rightarrow |2 - \lambda| < 1,$

entonces $|f'(x_\lambda)| = |2 - \lambda| < 1$, para $\lambda \in (1, 3)$

Se puede concluir a partir de la gáfica que si $\lambda > 3$, entonces se genera una nueva órbita de periodo 2, la cual es atractora, así lo indica la siguiente proposición.

Proposición 2.3.2 *[8, pp. 220-221] Para* $\lambda > 3$ *nace una nueva órbita de periodo* 2

A partir de $\lambda = 1 + \sqrt{6}$ la órbita de periodo 2 se vuelve repulsora, en la Figura 2.8 se puede observar puntos atractores de diferentes periodos, lo cual hace recordar el Teorema de Sharkovskii, mencionado en el Capítulo 1.

La simulación del mapeo logístico modificado

$$f(x) = 10,000\lambda x(1 - x) - [10,000\lambda x(1 - x)]$$

se observa en la Figura 2.9, que para $\lambda \in (1, 4)$, no hay valores de Lyapunov negativos, a diferencia del mapeo original. De manera que la modificación del mapeo es más sensible a las condiciones iniciales.

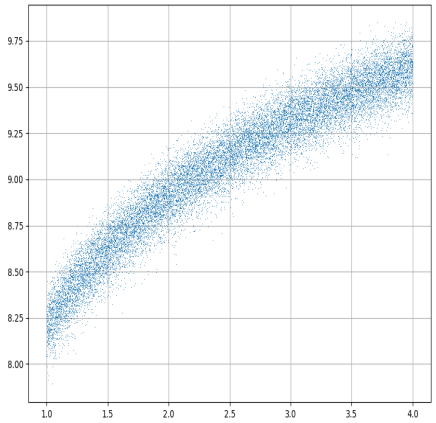


Figura 2.9: Coeficiente λ vs Exponentes de Lyapunov.

En la Figura 2.10, se observa el comportamiento del diagrama de bifurcación con los mismos parámetros que el mapeo logístico normal, mencionado anteriormente. Se observa que el rango de parámetros del sistema mejorado es mucho mayor que el del mapeo logístico original.

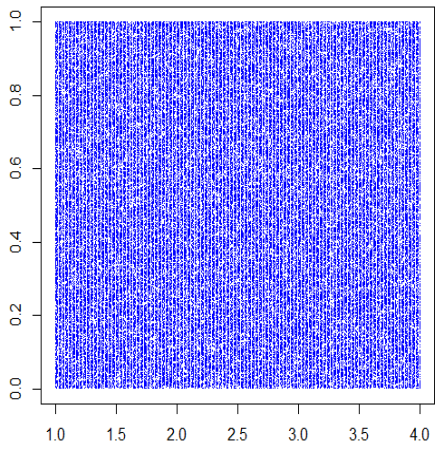


Figura 2.10: Bifurcación mapeo logístico modificado.

Capítulo 3

Ejemplo de aplicación de números seudoaleatorios

Los números seudoaletorios se utilizan para distintos fines. A continuación se muestran dos ejemplos de la aplicación de los números seudoaleatorios, el Método de Montecarlo y la aproximación de un fractal; en particular, el triángulo de Sierpinsky. En este trabajo de tesis particularmente se usan órbitas generadas por el mapeo logístico usual y el logístico modificado. Para sustentar la utilidad de los números generados se realizan pruebas estadísticas. Cabe mencionar que en el capítulo se utilizan los números generados para construir secuencias de bits que se pretende tengan propiedades seudolaeatorias.

3.1. Método de Montecarlo

Uno de los usos del Método de Montecarlo es la aproximación del valor de una integral definida con el uso de la probabilidad geométrica.

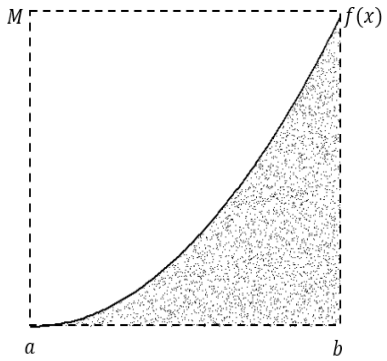
Para iniciar, suponga que la función continua $f : [a, b] \to \mathbb{R}$ es no negativa, es decir $f(x) \geq 0$ para todo $x \in [a, b]$. Se supone que M es el valor máximo de $f(x)$ en $[a, b]$ y D el rectángulo. También se puede usar un valor mayor al máximo de $f(x)$ en $[a, b]$.

$D = \{(x, y) : a \leq x \leq b,\ 0 \leq y \leq M\}$

Se realizan simulaciones con el método de Montecarlo para una alta gama de problemas complejos de diversas áreas; principalmente en las simulaciones de escenarios aleatorios para obtener soluciones numéricas aproximadas, consulte sobre el método de Montecarlo en el libro A First Course in Mathematical Modeling [9, p. 187].

El método nos permite obtener una aproximación de la integral $\int_a^b f(x)dx$, la cual es igual al área de la región:

$A = \{(x, y) : a \leq x \leq b,\ 0 \leq y \leq f(x)\}$



Para un valor $n \in \mathbb{N}$, el algoritmo se define por los pasos siguientes:

1) Se eligen aleatoriamente n puntos en el rectángulo D, para lograr lo anterior se generan:

a) n números aleatorios en $[a, b]$ (para x) ,

b) n números aleatorios en $[0, M]$ (para y).

2) Para cada punto $(x_k, y_k) \in D$ se determina si éste está o no en la región A.

3) Si r es el número de puntos que sí están en A, entonces el cociente $\frac{r}{n}$ es una estimación de la probabilidad de que el punto esté en la región A, es decir

$$P(A) = \frac{\text{área}(A)}{\text{área}(D)} = \frac{\int_a^b f(x)dx}{(b-a)M}$$

se tiene, por tanto

$$\frac{r}{n} \approx \frac{\int_a^b f(x)dx}{(b-a)M} \Rightarrow \int_a^b f(x)dx \approx \frac{r(b-a)M}{n}$$

A continuación se replica el método descrito para lo cual se utilizarán los números generados por los mapeos involucrados en este trabajo de tesis. Lo anterior para analizar qué tan aleatorios o seudoaleatorios son tales secuencias de números. Para este análisis se estimará la integral:

$$\int_0^1 x^2 dx.$$

Para la aproximación del valor de la integral mediante la simulación se usa, en primer lugar, los números generados por el mapeo logístico normal y, posteriormente por el logístico modificado. Es importante aclarar que se ocuparon distintos puntos semilla para las simulaciones, pero sólo se desarrolla un ejemplo para ilustrar.

Para el mapeo logístico $f(x) = 3.9999x(1-x)$ con dos puntos semilla distintos y 100,000 iteraciones se tiene el valor máximo de $f(x)$ en $[0,1]$ es $f(\frac{1}{2}) = \frac{3.9999}{4} < 1$ y además

$$\int_0^1 x^2 dx \approx \frac{r(b-a)M}{n} = \frac{38673(1)(1)}{100,000} = 0.38673$$

el cual está relativamente alejado del valor teórico $\frac{1}{3}$. Si aumentan el número de puntos a la simulación, el valor aproximado sigue siendo significativamente distinto al valor esperado.

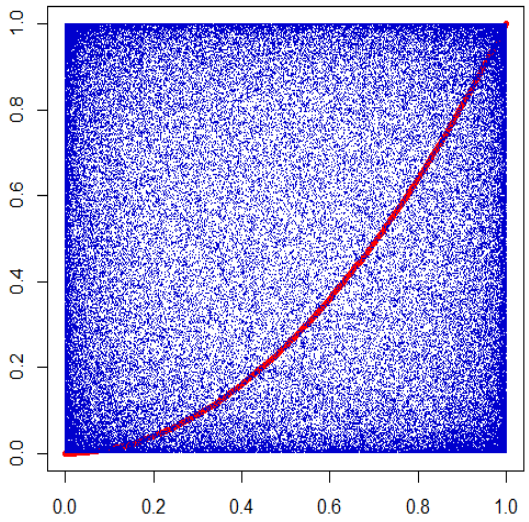


Figura 3.1: Área bajo la curva estimada por Montecarlo usando las órbitas seudoaletorias del mapeo logístico con $n = 100,000$ y punto semilla $(0.613764, 0.576897)$.

En la Figura 3.1, se puede apreciar que los puntos se encuentran acumulados, con una densidad mayor, en las cercanías a los vértices, mientras que en la zona central hay una notable escasez de puntos. Además, el gráfico nos permite ver que la distribución de los puntos no es uniforme.

En contraste con lo anterior, para el mapeo logístico modificado

$$f(x) = 39999x(1-x) - [39999x(1-x)]$$

con los mismos puntos semilla, ocupados para el mapeo normal y 100 mil iteraciones, se obtiene que

$$\int_0^1 x^2 dx \approx 0.33361$$

lo cual es un valor más cercano al valor teórico de $\frac{1}{3}$, mismo que mejora lo obtenido en la primera simulación. La distribución de los puntos es más uniforme como se muestra en el la Figura 3.2.

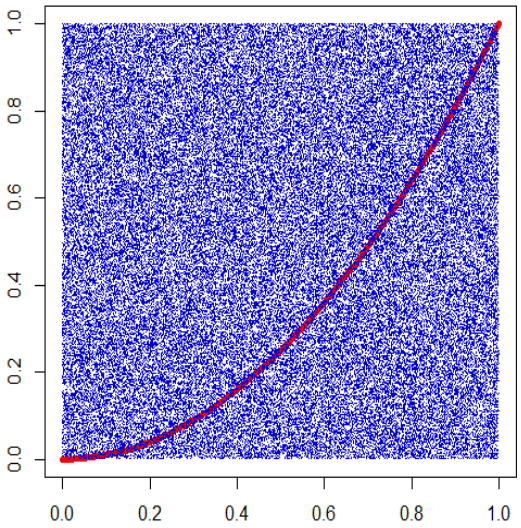


Figura 3.2: Área bajo la curva estimada por Montecarlo usando las órbitas seudoaletorias del mapeo logístico modificado con $n = 100,000$ y punto semilla $(0.613764, 0.576897)$.

3.2. Triángulo de Sierpinski

Diversos fractales conocidos se definen por la partición de algún objeto geométrico con distintas escalas y posiciones, que al mirarlo a cualquier nivel de escala será similar que el conjunto total, véase por ejemplo [18, p. 9].

El triángulo de Sierpinski se obtiene después de dividir infinitamente un triángulo en cuatro triángulos iguales y eliminar el triángulo central, es decir permanecen los tres triángulos de los vértices.

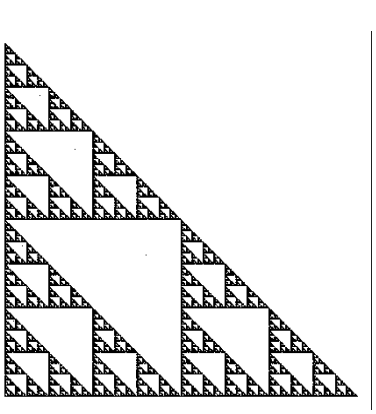


Figura 3.3: Recreación en geogebra del triángulo de Sierpinski.

El juego caótico es un procedimiento que funciona muy bien para aproximar fractales, por ejemplo el triángulo de Sierpinski que se describe de la siguiente manera:

Se inicia el procedimiento definiendo el triángulo $T = \{(x, y) : x + y \leq 1, x \geq 0, y \geq 0\}$, con vértices en los puntos A=$(0, 0)$, B=$(1, 0)$ y C=$(0, 1)$ y un punto semilla $(x_0, y_0) \in \mathbb{R}^2$, generalmente se toma $(x_0, y_0) \in [0, 1] \times [0, 1]$.

Los puntos medios de los segmentos AB, BC y AC definen cuatro triángulos semejantes al original, aunque con lados cuyas longitudes son iguales a la mitad de los lados del triángulo original.

$T_1^1 = \{(x, y) : x + y \leq \frac{1}{2}, x \geq 0, y \geq 0\}$,

$T_1^2 = \{(x, y) : x + y \leq 1, x \geq \frac{1}{2}, y \geq 0\}$,

$T_1^3 = \{(x, y) : x + y \leq 1, x \geq 0, y \geq \frac{1}{2}\}$.

Se considera un punto $P = (x, y)$. Las funciones siguientes nos ayudan a definir el triángulo de Sierpinski, si son elegidas aleatoriamente.

$Q_1(x, y) = (f_1(x), g_1(y)) = (\frac{x}{2}, \frac{y}{2})$, (punto medio entre P y A)

$Q_2(x, y) = (f_2(x), g_2(y)) = (\frac{x+1}{2}, \frac{y}{2})$, (punto medio entre P y B)

$Q_3(x, y) = (f_3(x), g_3(y)) = (\frac{x}{2}, \frac{y+1}{2})$, (punto medio entre P y C)

No es difícil ver que si $(x, y) \in T$, entonces $Q_1(x, y) \in T_1^1$, $Q_2(x, y) \in T_1^2$ y $Q_3(x, y) \in T_1^3$.

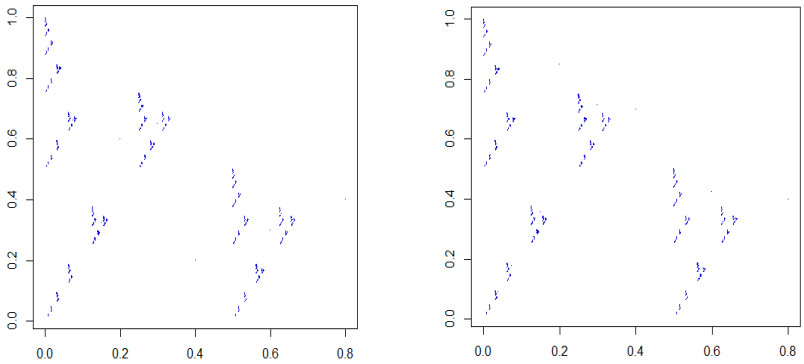
Las funciones se utilizan de acuerdo a lo ya establecido desde un inicio, es decir; que para los números $\{r_1, r_2, r_3, \ldots, r_n\} \subset [0, 1]$, generados por el mapeo logístico y el mapeo logístico modificado, se aplica $Q_1(x, y)$, $Q_2(x, y)$ o $Q_3(x, y)$ de acuerdo a la regla

Q_1, si $r_k \in [0, \frac{1}{3})$,

Q_2, si $r_k \in [\frac{1}{3}, \frac{2}{3})$,
Q_3, si $r_k \in [\frac{2}{3}, 1]$.

En la Figura 3.4 se puede observar la aproximación del triángulo con los números generados por el mapeo logístico

$$f(x) = 3.9999x(1-x).$$



(a) $x_0 = 0.074135$ (b) $x_0 = 0.22559436$

Figura 3.4: Aproximación generado por la órbita seudoaletoria del mapeo logístico con $n = 100,000$ valores generados y punto semilla en $(0.4, 0.4)$.

Como se aprecia en la Figura 3.4, la distribución de los puntos no es uniforme, la aproximación sólo muestra alguna regiones cargadas de puntos.

Por otra parte se puede observar la aproximación del triángulo de Sierpinski con la modificación del mapeo logístico factor con 10 y parámetro $\lambda = 3.9999$

$$f(x) = 39.999x(1-x) - [39.999x(1-x)].$$

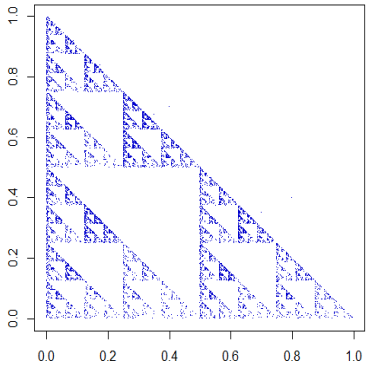


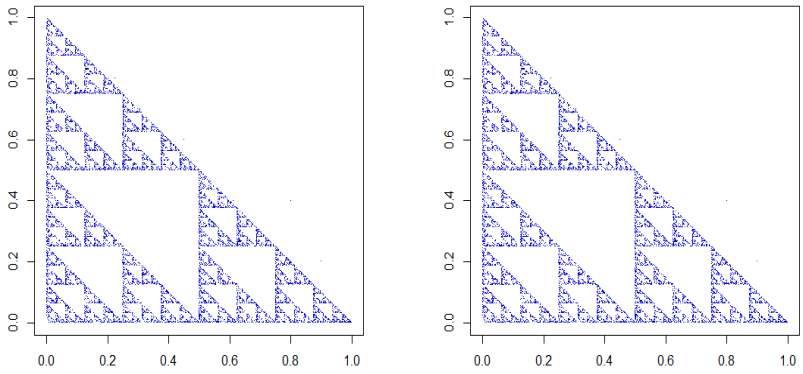
Figura 3.5: 10,000 punto semilla en $(0.4, 0.8)$.

Es posible notar en la Figura 3.5 que con la modificación, factor 10 se recrea una mejor aproximación al triángulo de Sierpinski. Sin embargo aún se observan en las regiones inferiores con poca acumulación de puntos.

Finalmente se analiza qué pasa si se usan números seudoaleatorios generados por el mapeo logístico factor 10,000, es decir;

$$f(x) = 39999x(1-x) - [39999x(1-x)].$$

En la figura 3.6 se observa una mejor aproximación del triángulo, con una distribución de puntos en todas las regiones.



(a) $x_0 = 0.074135$ (b) $x_0 = 0.22559436$

Figura 3.6: Aproximación con $n = 100,000$ valores generados y punto semilla en $(0.4, 0.4)$.

Capítulo 4

Encriptación

Una imagen digital rectangular en tonos de gris se obtiene al dividir una imagen en, por ejemplo, $M \times N$ subrectángulos de igual tamaño. A cada subrectángulo se denomina píxel y le asigna un tono de gris. En este trabajo se usan imágenes en tonos de gris de 8 bits lo cual genera $2^8 = 256$ tonos, desde negro hasta blanco.

En este capítulo se utilizarán imágenes digitales modeladas por una matriz de píxeles en tonos de gris, las cuales serán transformadas con ayuda de la secuencia generada por el mapeo logístico modificado, donde se espera sea seudoaleatoria. Lo anterior con el fin de encriptar píxeles y obtener una imagen encriptada.

4.1. Imágenes digitales

Se modela a las imágenes digitales o imágenes bidimensionales en la forma $f(x, y)$, donde el valor de f en las coordenadas (x, y) está determinada por la imagen.

En el caso de las imágenes digitales en tonos de gris la función $f(x, y)$ asigna a cada coordenada (x, y) un tono de gris, y su representación se da mediante un arreglo (matriz) compuesto por los valores numéricos $f(x, y)$.

Para la digitalización de una imagen en principio se requiere discretizar, a cada subrectángulo se le asigna un tono de gris; con lo cual se obtiene una matriz de $M \times N$ filas y columnas. Cada entrada de la matriz equivale a un píxel o un elemento de la imagen.

$$[f(x,y)] = \begin{bmatrix} f(1,1) & f(1,2) & \dots & f(1,N) \\ f(2,1) & f(2,2) & \dots & f(2,N) \\ \vdots & & & \\ f(M,1) & f(M,2) & \dots & f(M,N) \end{bmatrix}$$

Generalmente las coordenadas toman valores enteros entre 0 y $2^k - 1$, donde k indica el número de bits, lo cual determina la resolución de intensidad de la imagen, por último su almacenamiento y la cuantificación se determinará considerando el tipo de software.

La resolución de intensidad mejor conocida como niveles de intensidad o saturación de una la imagen digital se obtiene mediante el cálculo de una potencia entera de dos:

$$L = 2^k,$$

donde k = número de bits, suponiendo que los niveles discretos están igualmente espaciados y que son números enteros en el rango $[0, L-1]$.

El número necesario de bits para almacenar una imagen digital se define $b = MNk$, en el caso $M = N$, esta cantidad se convierte en $b = N^2k$. Cabe aclarar que un byte equivale a 8 bits y un megabyte equivale a 2^{20} bytes lo cual es aproximadamente igual a 10^6 bytes.

Cuando una imagen puede tener 2^k posibles niveles de intensidad, es una práctica común referirse a ella como una "imagen de k bits" (por ejemplo, una imagen de 256 niveles se denomina imagen de 8 bits). Cabe mencionar que los requisitos de almacenamiento para imágenes grandes, por ejemplo de 8 bits, para $16384 \times 16384 = 2^{14} \times 2^{14} = 2^{28}$ píxeles. Se necesitan $8(2^{28})$ bits, es decir 2^{28} bytes. Por lo cual se requieren $\frac{2^{28}}{2^{20}} = 2^8 = 256$ megabytes, lo cual no es insignificante.

Para mayor información respecto a las imágenes se recomienda consultar el libro de Rafael Gonzalez [10, pp. 61-70].

4.2. Escala de gris de 8 bits. Operación XOR (o exclusivo). Suma módulo 1

En esta sección se definen dos operaciones que serán utilizadas para las siguientes secciones, se recomienda [10, pp. 83-90] para la aplicación de otras posibles operaciones.

En las imágenes en escala de grises, la resolución determina la cantidad de niveles de gris que se pueden representar. Esta cantidad suele estar definida por el software utilizado durante la digitalización, siendo común el uso de 8 bits para definir cada nivel de gris. Normalmente se consideran valores enteros entre 0 y 255, aunque, por ejemplo, en el programa R se normaliza los valores al dividir entre 255. Cabe señalar que el paquete readPNG de R arroja valoren en el intervalo $[0, 1]$, el cuál se divide en 256 subintervalos de igual longitud; cada subintervalos representa un tono de gris.

En la Figura 4.1 se muestran las 256 variaciones de tonos de gris producidas por R, mismas que van de 0 a 255 tonos, en el cual 0 representa el color negro y 255 el tono blanco.

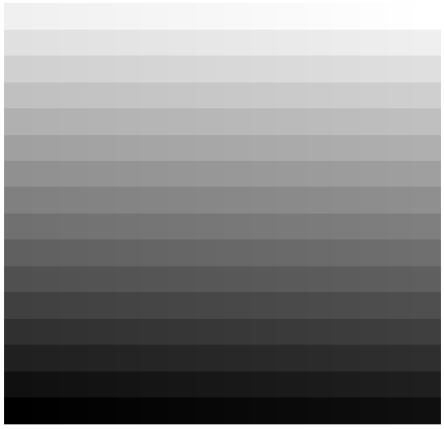


Figura 4.1: Escala de gris en R de 0a 256.

Suma de matrices y suma XOR.

- Suma de matrices:

  Si $A = [a_{ij}]$ y $B = [b_{ij}]$ son matrices de tamaño $m \times n$ con entradas en $\mathbb{R}$ se define la suma igual a la suma de los elementos de cada entrada. Así la entrada (i, j) de $A + B$ es $a_{i,j} + b_{i,j}$. Por ejemplo las matrices de 2×2 se tiene:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

  La suma elemental (se indica con el símbolo $\oplus$) de las dos matrices es:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \oplus \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} \\ a_{21} + b_{21} & a_{22} + b_{22} \end{bmatrix}$$

- Operación lógica "o exclusivo", o su expresión en inglés XOR (exclusive or):

  Una de las operaciones más utilizadas en la encriptación de imágenes es la operación lógica XOR, vinculada con el concepto de grupo. Es necesario recordar que un grupo es un conjunto G equipado con una operación $(+)$ tal que se cumplen las propiedades i) a iv):

  - i) $a, b \in G \Rightarrow a + b \in G$
  - ii) $a + (b + c) = (a + b) + c$, para todo $a, b, c \in G$ (asociatividad)
  - iii) Existe un elemento $e \in G$ tal que $a + e = e + a = a$. Existencia de elemento neutro.
  - iv) Para toda $a \in G$, existe $b \in G$ tal que $a + b = b + a = e$. Existencia de elemento inverso.

  Si además, se tiene la propiedad

  - v) Para cualesquiera $a, b \in G$ se tiene que $a + b = b + a$ se dice que el grupo es abeliano o conmutativo. Para mayor información consulte el [20, p. 2].

  Para el conjunto $\mathbb{Z}$, de los números enteros con la suma usual $+$ es un grupo. Otro ejemplo relevante es $\mathbb{Z}_2 = \{0, 1\}$, el grupo de los residuos módulo 2, con la operación

$\oplus$	0	1
0	0	1
1	1	0

  Los operadores lógicos se pueden definir en términos de tablas de verdad. Si define verdadero= 1 y falso= 0, en el caso de la operación XOR (o exclusivo) se obtiene la tabla siguiente:

XOR	F	V
F	F	V
V	V	F

lo cual coincide con

$XOR = \oplus$	0	1
0	0	1
1	1	0

Se nota que la operación XOR está definida en $\{0, 1\}$ y coincide con el grupo $(\mathbb{Z}_2, \oplus)$, donde $\oplus$ denota la suma módulo 2.

Ahora se define la operación XOR entre números enteros no negativos de k bits, con $k \in \mathbb{N}$.

Para cada $k \in \mathbb{N}$, el conjunto $\mathbb{Z}_2^k$ con la operación

$(a_1, a_2, \ldots, a_k) \oplus (b_1, b_2, \ldots, b_k) = (a_1 \oplus b_1, a_2 \oplus b_2, \ldots, a_k \oplus b_k)$

es un grupo. Si x y y son dos números enteros no negativos menores que 2^k (enteros de k bits), la operación $XOR(x, y)$ se define de la forma siguiente:

a) x y y se expresan en base 2, si es necesario se completa con ceros por la parte izquierda hasta tener k cifras. Si se supone que en base 2 se tiene

$x = \sum_{i=0}^{k-1} a_{k-i} 2^i = (a_1 a_2 \cdots a_k)_2$ y

$y = \sum_{i=0}^{k-1} b_{k-i} 2^i = (b_1 b_2 \cdots b_k)_2$.

ahora se realiza la correspondencia

$x = \sum_{i=0}^{k-1} a_{k-i} 2^i = a_1 a_2 \cdots a_k \to (a_1, a_2, \cdots, a_k) \in \mathbb{Z}_2^k$ y

$y = \sum_{i=0}^{k-1} b_{k-i} 2^i = b_1 b_2 \cdots b_k \to (b_1, b_2, \cdots, b_k) \in \mathbb{Z}_2^k$.

b) Para la suma en el grupo $\mathbb{Z}_2^k$

$x \oplus y = (a_1, a_2, \ldots, a_k) \oplus (b_1, b_2, \ldots, b_k)$

$x \oplus y = (a_1 \oplus b_1, a_2 \oplus b_2, \ldots, a_k \oplus b_k)$.

c) Finalmente si $c_{k-i} = a_{k-i} \oplus b_{k-i}$ para cada $i = 0, 1, 2, \ldots, k-1$, se define

$XOR(x, y) = \sum_{i=0}^{k-1} c_{k-i} 2^i = (c_1 c_2 \cdots c_k)_2$, el cual es un número entero no negativo de k bits, expresado en base 2.

Es importante destacar que 1 es el inverso aditivo de 1 en $\mathbb{Z}_2$, en efecto $1 + 1 = 2 \equiv 0 mod 2 = 0$. Por otra parte el inverso aditivo de 0 es él mismo, ya que $0 + 0 = 0$. Por lo anterior el inverso aditivo del número entero de k bits x es x mismo, es decir $XOR(x, x) = 0$

Proposición 4.2.1 *El conjunto G formado por los enteros no negativos de k bits, con $k \in \mathbb{N}$, es un grupo abeliano con la operación XOR.*

Demostración. Sean $a = \sum_{i=0}^{k-1} a_{k-i} 2^i$, $b = \sum_{i=0}^{k-1} b_{k-i} 2^i$ y $c = \sum_{i=0}^{k-1} c_{k-i} 2^i$ tres números enteros no negativos de k bits, tales que los coeficientes corresponden, respectivamente, a las expansiones de a, b y c en base 2.

i) Es consecuencia de c) que si $a, b \in G$, entonces $XOR(a, b) \in G$

ii) Para a y b se tiene

$XOR(a, XOR(b, c)) = XOR(\sum_{i=0}^{k-1} a_{k-i} 2^i, \sum_{i=0}^{k-1} (b_{k-i} \oplus c_{k-i}) 2^i) =$

$\sum_{i=0}^{k-1} [a_{k-i} \oplus (b_{k-i} \oplus c_{k-i})] 2^i,$

donde $r \oplus s$ denota la suma en $\mathbb{Z}_2$. Dado que la suma $\oplus$ es asociativa se tiene que

$XOR(a, XOR(b, c)) = \sum_{i=0}^{k-1} [a_{k-i} \oplus (b_{k-i} \oplus c_{k-i})] 2^i = \sum_{i=0}^{k-1} [(a_{k-i} \oplus b_{k-i}) \oplus c_{k-i}] 2^i,$

lo cual es igual a $XOR(XOR(a, b), c)$. Lo anterior muestra que la operación XOR es asociativa.

iii) El elemento neutro es $0 = \sum_{i=0}^{k-1} (0) 2^i \in G$

iv) El elemento inverso de $a = \sum_{i=0}^{k-1} a_{k-i} 2^i \in G$ es a, ya que
$a_{k-i} \oplus a_{k-i} = 0$ para todo $i = 0, 1, \ldots, k-1$. En efecto

$XOR(a, a) = XOR(\sum_{i=0}^{k-1} a_{k-i} 2^i, \sum_{i=0}^{k-1} a_{k-i} 2^i) = \sum_{i=0}^{k-1} (a_{k-i} \oplus a_{k-i}) 2^i = \sum_{i=0}^{k-1} (0) 2^i = 0$

v) Para a y b se tiene
$XOR(a, b) = XOR(\sum_{i=0}^{k-1} a_{k-i} 2^i, \sum_{i=0}^{k-1} b_{k-i} 2^i) = \sum_{i=0}^{k-1} (a_{k-i} \oplus b_{k-i}) 2^i =$
$\sum_{i=0}^{k-1} (b_{k-i} \oplus a_{k-i}) 2^i = XOR(b, a)$

por lo tanto XOR es conmutativa. Con lo que se demuestra que G con la operación XOR es un grupo abeliano.□

De lo anterior se desprende la proposición siguiente, misma que es evidente ya que se tiene la estructura de grupo. Se incluye la demostración para dar claridad a la notación XOR.

Proposición 4.2.2 *Si $a, b \in G$, entonces la ecuación $XOR(x, a) = b$ tiene solución única, la cual se obtiene por $x = XOR(b, a)$.*

Demostración Primero es necesario notar que de la proposición anterior se sigue $XOR(XOR(b, a), a) = XOR(b, XOR(a, a)) = XOR(b, 0) = b$

lo cual implica que $XOR(b, a)$ es solución de la ecuación $XOR(x, a) = b$.

Si se supone que y es solución de la ecuación $XOR(x, a) = b$, entonces $XOR(y, a) = b$. Del resultado anterior se tiene que

$XOR(y, a) = b$ implica, al aplicar $XOR(\ , a)$ en ambos lados, que

$XOR(XOR(y, a), a) = XOR(b, a)$

por lo cual

$XOR(XOR(y, a), a) = XOR(y, XOR(a, a)) = XOR(y, 0) = y = XOR(b, a)$.

Lo anterior muestra que la ecuación $XOR(x, a) = b$ tiene solución única.□

Ejemplo.

Las expresiones en binario de 14 y 11, enteros de 4 bits, son

$14 = 1110_2$ y $11 = 1011_2$, la suma común es

$14+11 = 1110_2+1011_2 = 11001_2 = 1(2^4)+1(2^3)+0(2^2)+0(2^1)+1(2^0) = 16 + 8 + 1 = 25$ (entero de 5 bits).

Por otra parte la suma $XOR(14, 11)$ se obtiene a partir de la suma en $\mathbb{Z}_2^4$ de los vectores obtenidos por

$14 = 1110_2 \to (1, 1, 1, 0) \in \mathbb{Z}_2^4$

$11 = 1011_2 \to (1, 0, 1, 1) \in \mathbb{Z}_2^4$

al sumar en $\mathbb{Z}_2^4$ se tiene

$(1, 1, 1, 0) \oplus (1, 0, 1, 1) = (1 \oplus 1, 1 \oplus 0, 1 \oplus 1, 0 \oplus 1) = (0, 1, 0, 1)$,

pasando posteriormente a un entero expresado en base 2,

$XOR(14, 11) = 0101_2 = 0(2^3) + 1(2^2) + 0(2^1) + 1(2^0) = 4 + 1 = 5$

Es fácil ver que $XOR(14,5) = 11$ y $XOR(11,5) = 14$, debido a que

$(1,1,1,0) \oplus (0,1,0,1) = (1,0,1,1) \rightarrow 1011_2 = 2^3 + 0(2^2) + 1(2^1) + 1(2^0) = 8 + 2 + 1 = 11$

$(1,0,1,1) \oplus (0,1,0,1) = (1,1,1,0) \rightarrow 1110_2 = 2^3 + (2^2) + 1(2^1) + 0(2^0) = 8 + 4 + 2 = 14$

Lo anterior debe cumplirse debido a que finalmente se está utilizando la operación del grupo $\mathbb{Z}_2^4$. En otras palabras se puede decir que la ecuación

$XOR(a, 11) = 5$

tienen solución única, la cual se obtiene por

$XOR(XOR(a, 11), 11) = XOR(5, 11)$

$XOR(a, XOR(11, 11)) = XOR(a, 0) = a = XOR(5, 11) = XOR(11, 5) = 14.$

4.3. Codificación de imágenes en tonos de gris con el mapeo logístico

A continuación, se presentan los pasos del algoritmo de encriptación de acuerdo al método, Módulo 1 o XOR. Con el objeto de mejorar la resistencia a los ataques y obtener mejores resultados estadísticas en el análisis, se realiza una modificación, especificada al final de la descripción.

4.3.1. Suma XOR

Para evitar problemas con las regiones completamente negras o blancas, al aplicar los métodos, en la imagen original se eliminaron ambos colores sustituyendo el 1 y 0 por valores cercanos a ellos. Esto por supuesto fue realizado antes de iniciar los métodos de encriptación.

Procedimiento con una órbita:

1: Se lee una matriz $[A_{i,j}]$ de tamaño $m \times n$, la cual está definida por los tonos de gris de los píxeles de la imagen. En el programa R se usa valores en el intervalo $[0, 1]$.

2: La matriz $[A_{i,j}]$, se transforma en vector B de $\mathbb{R}^{mn}$, por columnas.

3: Se utiliza el mapeo logístico modificado con factor 10,000

$f(x) = 10000\lambda x(1-x) - [10000\lambda x(1-x)]$ dadas las llaves x_1 y λ ya definidas en primer instancia, donde el valor inicial x_1 genera mn valores x=$(x_1, x_2, \ldots, x_{mn}) \in \mathbb{R}^{mn}$.

- Para i de 2 a mn

$$x_i = 10000\lambda x_{i-1}(1-x_{i-1}) - [10000\lambda x_{i-1}(1-x_{i-1})]$$

considerando como valor inicial $x_1 = x_0$ o punto semilla $x_0 \in (0,1)$ y $\lambda \in (3.9999, 4)$.

4: Se realiza el siguiente procedimiento se definen ambos vectores B y x:

- Sea C un vector con mn entradas, en cada entrada una la suma XOR, con lo cual se obtiene un vector encriptado.

  El programa R, versión 4.2.3 de 64 bits, la función "bitXor" sólo acepta números enteros no negativos de 32 bits, es decir desde 0 hasta $2^{32}-1$. Por otra parte, los valores de los vectores A, x, correspondientes a la imagen y al mapeo logístico modificado, se encuentran dentro del intervalo $[0,1]$. Para poder hacer uso de bitXor se podría aplicar la siguiente transformación:
  $[(2^{32}-1)\cdot B], [(2^{32}-1)\cdot x]$.

  Dado que en esta tesis se manejan imágenes de píxeles de 8 bits, se considera
  $C = XOR((2^8)B, (2^8)x) = XOR[(256)B], [(256)\cdot x]$.

5: El vector C, que tiene mn entradas, se transforma en matriz de $m \times n$, que se denota por $D = [D_{i,j}]$. La matriz D define a la imagen encriptada, al graficar se obtiene una imagen en tonos de gris con los valores nuevos.

Con la imagen encriptada, es claro, tener en mente el recuperar la imagen original. Para lograrlo es necesario aplicar el proceso inverso, mismo que se describe a continuación.

Algoritmo de desencriptado:

1: Se lee el vector C. Se cuenta con la matriz D, entonces se convierte a ésta en vector, a través de sus columnas, $C = c(D)$ (en R, opr supuesto).

2: Se genera el vector x, de longitud mn con el mapeo logístico modificado mediante las llaves ya proporcionadas, es decir $x_1 = x_0$ (punto semilla) y λ.

3: Para cada i se consideran la entradas x_i, C_i y posteriormente se aplica la función XOR.

Por la definición de $XOR(B_i, x_i)$, si tiene que si $C_i = XOR(B_i, x_i)$, entonces

$XOR(B_i, C_i) = x_i$ y $XOR(x_i, C_i) = B_i$, lo anterior obedece a que $\mathbb{Z}_2^n$ es un grupo y, además el inverso aditivo de un elemento es él mismo.

- Sea $E \in \mathbb{R}^{nm}$ el vector descifrado. A partir de los valores de los píxeles encriptados C y de los valores del vector $B \in \mathbb{R}^{nm}$, generados por el mapeo logístico modificado, con las llaves correctas se tiene que para cada i entero en $1 : nm$

$$E_i = XOR(C_i, 256B_i)$$

en este caso como los valores de la matriz encriptada son enteros no negativos de 8 bits, de 0 a $2^8 = 256$, no es necesario hacer de nuevo la multiplicación por 256.

Cabe señalar que no se recuperan los números originales de la matriz A, pero sí corresponden a los mismos tonos de grises.

4: El vector E se transforma en matriz de $m \times n$, que se denota por $F_{i,j}$. Con los tonos de gris definidos en cada píxel por el número correspondiente, se recupera la imagen original.

4.3.2. Suma módulo 1.

La parte inicial coincide, lo que marca la diferencia es la suma.

Procedimiento para encriptación:

1: Se lee la matriz de píxeles $[A_{i,j}]$, la cual está definida por los tonos de gris de la imagen, en R se tiene valores en el intervalo $[0, 1]$.

2: Se utiliza el mapeo logístico modificado, con factor 10,000.

$f(x) = 10000\lambda x(1-x) - [10000\lambda x(1-x)]$

con las llaves $x_1 \in (0,1)$ y coeficiente $\lambda \in (3.9999, 4)$, ya definidas en primera instancia. Con el valor inicial x_1 se genera en total nm valores x=$\{x_1, x_2, \ldots, x_{nm}\}$.

- Para cada entero i desde 2 hasta nm se usa el valor anterior x_{i-1} y se itera:

$$x_i = 10000\lambda x_{i-1}(1-x_{i-1}) - [10000\lambda x_{i-1}(1-x_{i-1})].$$

3: El vector $V = (x_1, x_2, \ldots, x_{nm})$, generado por del mapeo logístico, se convierte a matriz con n filas y m columnas, se denota como $B = [B_{i,j}]$.

4: Se realiza el siguiente procedimiento utilizando los valores de ambas matrices:

- Se define la matriz $C = [C_{i,j}]$, la cual es la nueva matriz, que contiene a los píxeles encriptados por el mapeo logístico modificado.

  Posteriormente, se realiza la suma $C_{i,j} = A_{i,j} \oplus B_{i,j} = A_{i,j} + B_{i,j} - [A_{i,j} + B_{i,j}]$, donde $[a]$ =parte entera de a.

Algoritmo de descifrado:

Para recuperar la imagen original, es necesario construir el camino inverso que genera los píxeles iniciales. Como se indica a continuación.

1: Se lee la matriz encriptada $C = [C_{i,j}]$.

2: Se genera el vector V con ayuda del mapeo logístico modificado, las llaves $x_1 \in (0,1)$ (punto semilla) y $\lambda \in (3.9999, 4)$. Posteriormente, el vector V se convierte en una matriz $[B_{i,j}]$ de $n \times m$ (así como se realizó en el proceso de encriptación).

3: Se define la matriz E de $n \times m$, que corresponde a la matriz desencriptada.

La operación inversa define $E = [E_{i,j}]$ por cada una de sus entradas a través de:

$$E_{i,j} = \begin{cases} E_{i,j} = C_{i,j} - B_{i,j} & , \text{si } C_{i,j} > B_{i,j} \\ E_{i,j} = C_{i,j} - B_{i,j} + 1 & , \text{si } C_{i,j} \leq B_{i,j}. \end{cases}$$

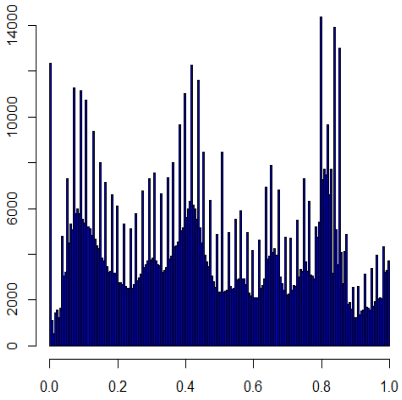
4: Finalmente la matriz de píxeles E nos da la imagen desencriptada.

Se recomienda ver los artículos de referencia [1],[2] y [21]

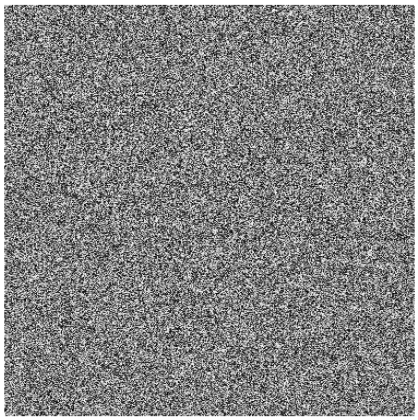
4.4. Análisis experimental

$\lambda = 3.999916$ puntos semilla $x_0 = 0.125148$ y $x_0 = 0.160126$

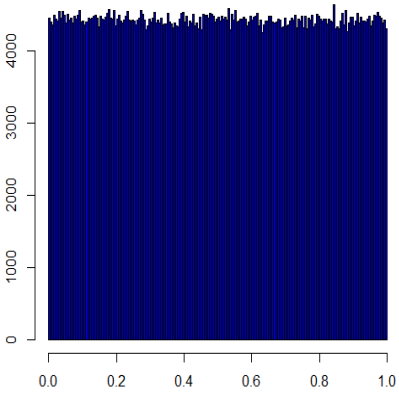
(a) Imagen original



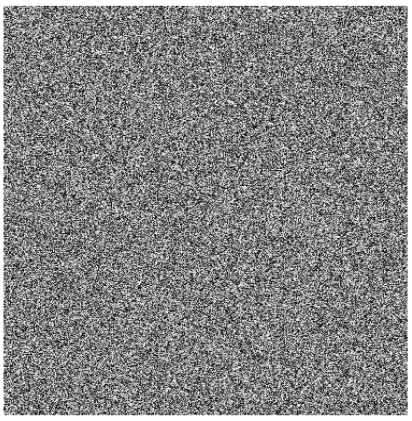
(b) Histograma imagen original



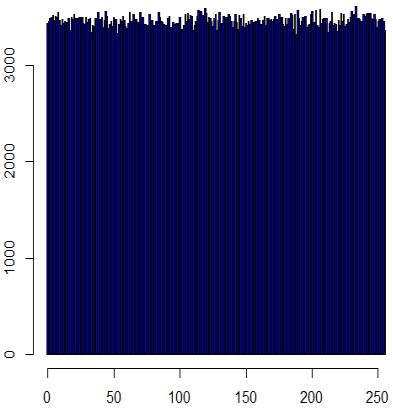
(c) Imagen encriptada método suma módulo 1



(d) Histograma encriptado módulo 1



(e) Imagen encriptada método XOR

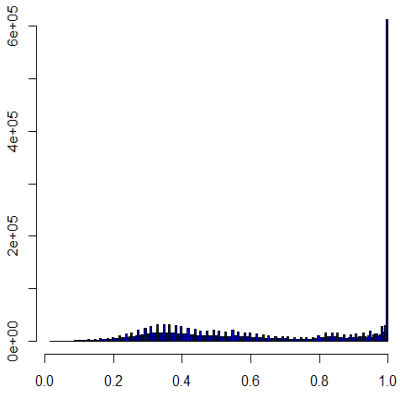


(f) Histograma método XOR

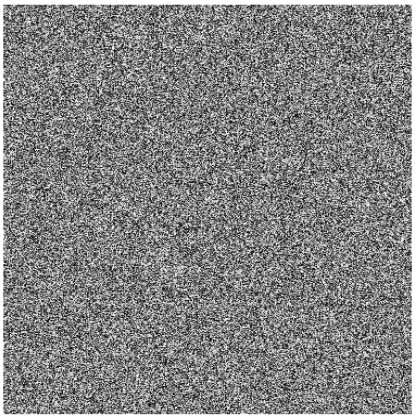
Figura 4.2: Imagen digital de 820×1082.

Imagen 4.2(a) extraída del sitio oficial de la Universidad Autónoma de la Ciudad de México.
$\lambda = 3.999916$ puntos semilla $x_0 = 0.125148$ y $x_0 = 0.160126$.

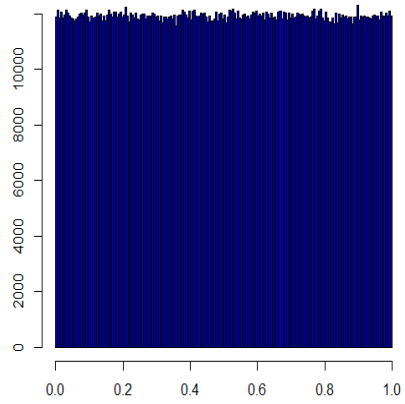
(a) Imagen original



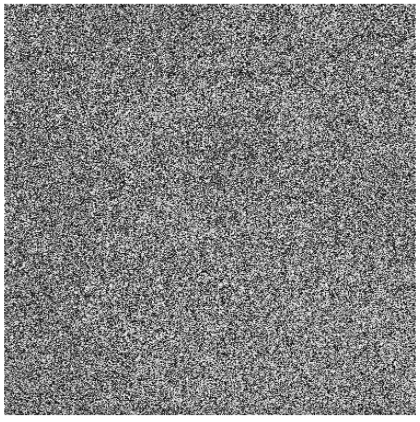
(b) Histograma imagen original



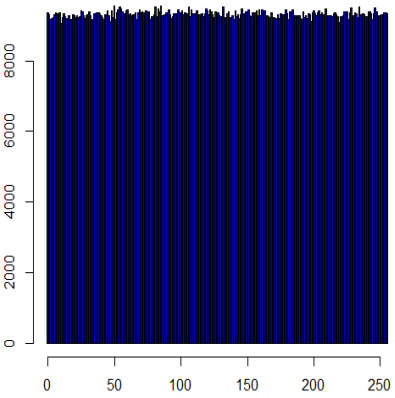
(c) Imagen encriptada método suma módulo 1



(d) Histograma encriptado módulo 1



(e) Imagen encriptada método XOR



(f) Histograma método XOR

Figura 4.3: Imagen digital de 1544×1544.

Fotografía 4.3(a) tomada en San Pedro Cajonos Oaxaca.

En las páginas siguientes se muestran las imágenes desencriptadas.

Correlación de Pearson

El coeficiente de correlación de Pearson, es un estadístico paramétrico cuya aplicación es para variables cuantitativas, es un índice que mide el grado de correlación entre dos variables relacionadas. El coeficiente de correlación toma valores entre -1 y 1.

Donde:

- -1 implica una correlación lineal perfecta negativa entre dos variables,
- 0 no existe correlación lineal entre dos variables,
- 1 la correlación lineal es perfecta positiva entre dos variables,
- valores cercanos a -1, 1 indica correlación lineal alta,
- valores cercanos a 0, indica la correlación lineal baja, es decir los cambios en una variable no estás asociados de manera sistemática con cambios en otra.

Para calcular el coeficiente de correlación de Pearson véase el [24] entre dos variables x, y, de tamaño n sin asumir una relación funcional de independencia entre ellas; se emplean las siguientes relaciones:

$$p = \frac{Cov(x,y)}{\sigma_x \sigma y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n\sigma_x \sigma y}$$

$$Cov(x,y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{n}$$

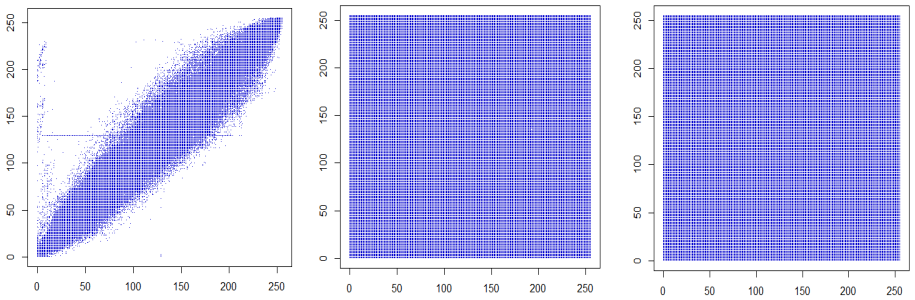
$$p = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

La prueba de hipótesis a considerar es:

H_0 = No hay correlación lineal.

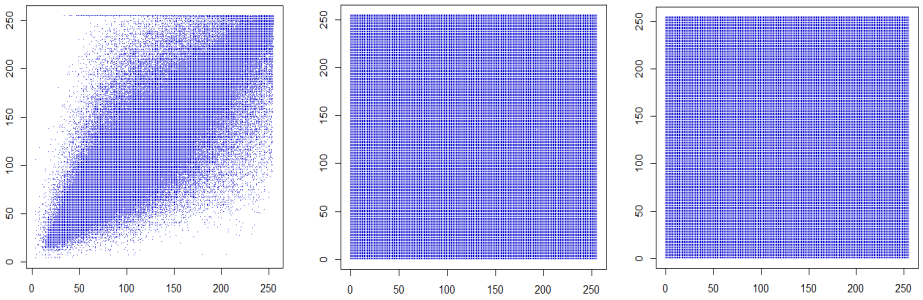
H_1 = Hay correlación lineal.

La prueba indicara si existe correlación entre los píxeles cercanos, dentro de la matriz que define a la imagen por estudiar, las Imágenes siguientes se ilustra la correlación dentro de las matrices.



(a) Correlación de la imagen 4.2(a) (b) Correlación de la imagen 4.2(c) (Módulo 1) (c) Correlación de la imagen 4.2(e) (XOR)

Figura 4.4: Correlación de pixeles vecinos



(a) Correlación de la imagen 4.3(a). (b) Correlación de la imagen 4.3(c) (Módulo 1). (c) Correlación de la imagen 4.3(e) (XOR).

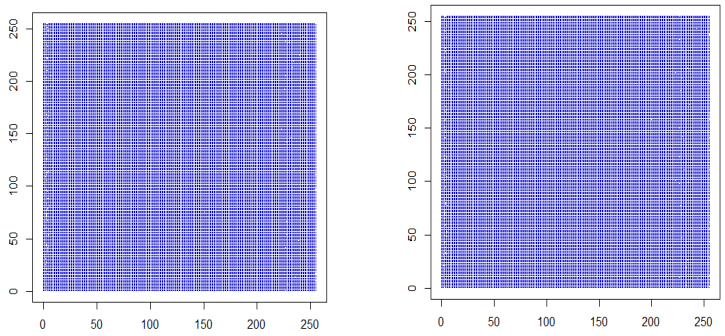
Figura 4.5: Correlación dentro de la matriz imagen 2

Se observa una alta correlación entre los píxeles de las imágenes originales a comparación de las imágenes encriptadas. Aplicando la prueba de correlación de Pearson en la Tabla 4.1, se puede afirmar el rechazo de la H_0, para la correlación entre los píxeles de la Imagen 4.2(a) y 4.3(a). En cambio, para las Imágenes 4.2(c), 4.3(c), 4.3(e) y 4.2(e), se obtuvo un no rechazo a la H_0, además se obtienen valores cercanos a 0, es decir, existe una baja correlación lineal entre los píxeles.

	Imagen		Módulo 1		XOR	
	Correlación	valor-p	Correlación	valor-p	Correlación	valor-p
4.2(a)	0.986186	2.2×10^{-16}	-0.001032578	0.3307	0.000640028	0.5379
4.3(a)	0.988363	2.2×10^{-16}	0.001247606	0.05407	0.0006212448	0.3375

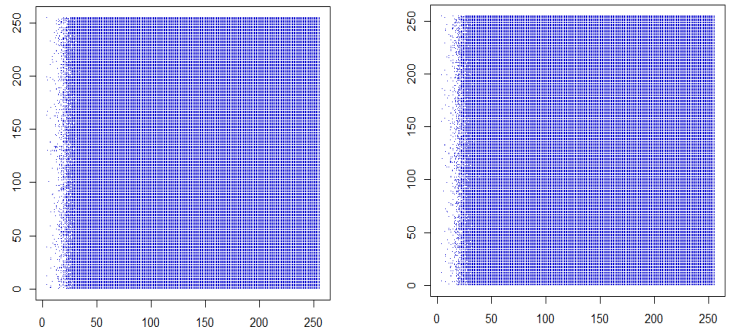
Tabla 4.1: Análisis de correlación dentro de la matriz misma.

En el análisis de los píxeles correspondientes a la imagen original y los dos métodos, se puede observar en las Figuras siguientes que no existe correlación lineal entre los píxeles.



(a) 4.2(a) y 4.2(c). (b) 4.2(a) y 4.2(e).

Figura 4.6: Correlación entre imagen original-encriptada.



(a) 4.3(a) y 4.3(c). (b) 4.3(a) y 4.3(e).

Figura 4.7: Correlación entre imagen original-encriptada.

Además en la Tabla 4.2 se puede observar que para ambos métodos no se rechaza la hipótesis H_0 y los valores de correlación son cercanos a 0, es decir no existe correlación lineal entre los píxeles.

Imagen	Original vs Módulo 1		Original vs XOR	
	valor-p	Correlación	valor-p	Correlación
4.2(a)	0.282	-0.001130672	0.4066	-0.0008809831
4.3(a)	0.4847	-0.000452763	0.05589	-0.001238293

Tabla 4.2: Correlación entre la imagen encriptada y la original.

Nota: En este escrito sólo se ilustró con 2 imágenes distintas y dos puntos iniciales fijos los resultados obtenidos, no obstante se hicieron 100 simulaciones con distintos puntos semilla aleatorios $x_0 \in (0, 1)$ y diferentes imágenes en donde se obtuvieron resultados similares.

Uniformidad

Como se describió anteriormente la prueba de χ^2 determina la posible distribución uniforme dentro de un intervalo, por lo que, mediante la prueba se buscará la uniformidad de las 256 clases obtenidas por cada tono de gris en la secuencia seudoaletoria generada por la imagen encriptada. Cabe mencionar que en esta sección se muestra otro método eficaz para determinar sí existe un distribución uniforme dentro de una secuencia, la cual se aplica para distribuciones continuas.

En los resultados de la Tabla 4.3 se tiene la prueba de χ^2 aplicada a la secuencia seudoaleatoria generada por la imagen encriptada mediante el método XOR

Imagen	XOR
4.2(e)	valor-$p = 0.1887$
4.3(e)	valor-$p = 0.0724$

Tabla 4.3: Uniformidad χ^2

Los resultados que se obtienen muestran que no hay evidencia estadística de rechazar la hipótesis nula, con una insignificancia de $\alpha = 0.01$ al obtener valores-$p > \alpha$, y 255 grados de libertad.

Al igual que la prueba de χ^2, el método Kolmogorov-Smirnov [11] determina si existe uniformidad dentro de un intervalo. Sin embargo para esté método se debe considerar que en la secuencia seudoaleatoria obtenida por la imagen encriptada mediante el método Modulo 1, no se obtuvieron valores repetidos, por lo que se aplica la prueba que consiste en el procedimiento siguiente:

$D = sup \mid F_n(x_i) - F_0(x_i) \mid$

x_i es el $i-$ésimo valor observado en la muestra,

$F_n(x_i)$ es un estimador de la probabilidad de observar valores menores o iguales que x_i,

$F_0(x_i)$ es la probabilidad de observar valores menores o iguales que x_i cuando H_0 es cierta.

Si D es la mayor diferencia absoluta observada entre la frecuencia acumulada observada $F_n(x_i)$ y la frecuencia acumulada teórica $F_0(x_i)$, entonces:

- $D \leq D_\alpha \rightarrow$ se acepta H_0.

- $D \geq D_\alpha \rightarrow$ se rechaza H_0.

Donde D se obtiene:

$D^+ = max_{1 \leq i \leq n}\{\frac{i}{n} - F_0(x_i)\}$, $D^- = max_{1 \leq i \leq n}\{F_0(x_i) - \frac{i-1}{n}\}$,

$D = max\{D^+, D^-\}$.

El valor de D_α depende del tipo de distribución a probar y se encuentra realizando una tabulación. En general es de la forma:

$D_\alpha = \frac{C_\alpha}{k(n)}$, donde C_α y $k(n)$, se obtienen de la tabla siguiente, extraída de [11]:

α	0.1	0.05	0.01
C_α	1.224	1.358	1.628

Distribución que se contrasta:

$k(n) = \sqrt{n} + 0.12 + \frac{0.11}{\sqrt{n}}$.

H_0: Los datos analizados siguen una distribución M.

H_1: Los datos analizados no siguen una distribución M.

Si valor-$p \geqslant \alpha \rightarrow$ se acepta H_0.

Si valor-$p < \alpha \rightarrow$ se rechazar H_0, con una significancia de $\alpha = 0.01$, ya que se reduce la probabilidad de cometer errores de tipo 1 en comparación con niveles de significancia más altos, como $\alpha = 0.05$.

En la Tabla 4.4, se puede observar los resultados obtenidos al aplicar la prueba de Kolmogorov-Smirnov para el método Módulo 1 y para las imágenes originales:

Imagen	Original	Imagen encriptada	Módulo 1
4.2(a)	valor-$p = 2.2 \times 10^{-16}$	4.2(c)	valor-$p = 0.27$
4.3(a)	valor-$p = 2.2 \times 10^{-16}$	4.3(c)	valor-$p = 0.27$

Tabla 4.4: Uniformidad Kolmogorov-Smirnov

Los resultados de las pruebas de Kolmogorov-Smirnov y χ^2 indican el rechazo de la hipótesis nula (H_0) de uniformidad, lo que sugiere que los píxeles no se distribuyen uniformemente en el intervalo $[0, 1]$ para la Figura 4.2(b). Sin embargo, al analizar los histogramas presentados en las Figuras 4.2(d) y 4.2(f), se observa una distribución más uniforme entre las clases. Esta observación se confirma con los valores-p obtenidos, superiores a 0.01, lo que implica que no se rechaza la hipótesis nula en estos casos.

Aleatoriedad

Para esta prueba se proponen las hipótesis siguientes:

H_0 = Existe aleatoriedad.

H_1 = No existe aleatoriedad.

Imagen	Original	Módulo 1	XOR
4.2(a)	valor-$p = 2.2 \times 10^{-16}$	valor-$p = 0.27$	valor-$p = 0.929$
4.3(a)	valor-$p = 2.2 \times 10^{-16}$	valor-$p = 0.27$	valor-$p = 0.729$

Tabla 4.5: Aleatoriedad

Como se puede ver en la tabla 4.5 no se puede considerar que el orden es aleatorio de los píxeles de las imágenes originales es aleatorio. Sin embargo al aplicarse la prueba a las sucesiones obtenidas mediante los dos métodos de encriptación Módulo 1 y XOR se obtienen para ambos métodos valor-p > 0.01, dando así el no rechazo a la H_0, es decir; la serie de píxeles se generan aleatoriamente.

4.4.1. Análisis de sensibilidad en las llaves

Para tener un buen cifrado se debe contar con una cantidad grande de llaves para que no sea fácil mediante ataques que utilizan "la fuerza bruta" obtener la llave correcta.

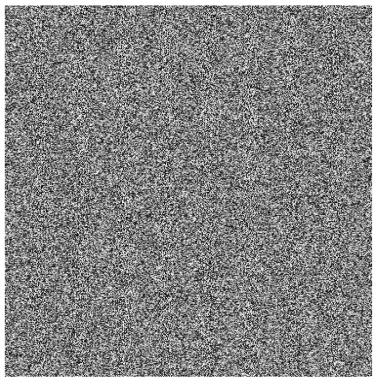
Con los métodos Módulo 1 y XOR se pueden obtener 4 llaves; los puntos iniciales x_0 y dos $\lambda \in (3.9999, 4)$, es decir:

$$10^{15} \times 10^{15} \times 10^{11} \times 10^{11} = 10^{52} \approx 2^{172}.$$

En el ejemplo siguiente se muestra que al realizar un pequeño cambio en las llaves, la imagen no se logra desincriptar correctamente.

Es importante aclarar que en los siguientes ejemplos se toman dos puntos iniciales x_{0_1}, x_{0_2} y un $\lambda \in (3.9999, 4)$, por lo que en principio se genera la modificación de los puntos semilla dejando a $\lambda = 3.9999$, figuras 4.8(a) y 4.10(a), para después generar la modificación a λ y fijar a los puntos iniciales como se observa en las figuras 4.9(a) 4.11(a).

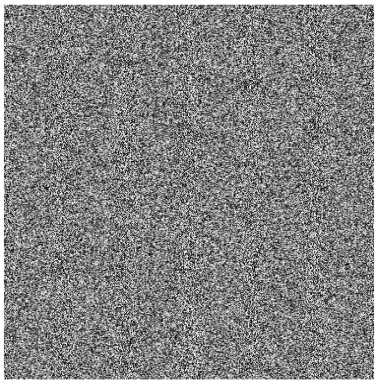
$$f(x) = 10,000\lambda x(1-x) - [10,000\lambda x(1-x)]$$



(a) Imagen desencriptada con $x_0 = 0.125148 + 10^{-14}$, $x_0 = 0.160126 + 10^{-14}$.

(b) Imagen desencriptada $x_0 = 0.125148$, $x_0 = 0.160126$.

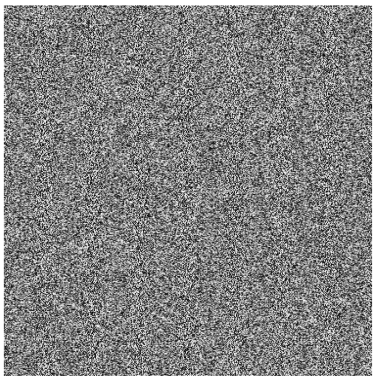
Figura 4.8: Módulo 1, con puntos semilla modificado.



(a) Imagen desencriptada $\lambda = 3.999916 + 10^{-14}$.

(b) Imagen desencriptada $\lambda = 3.999916$.

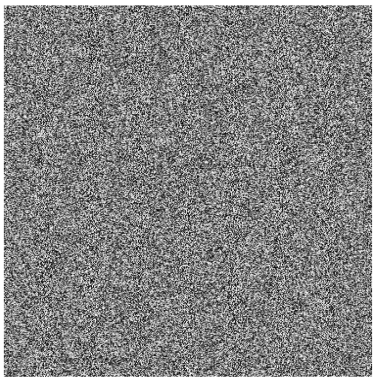
Figura 4.9: Módulo 1, con λ modificado.



(a) Imagen desencriptada con $x_0 = 0.125148 + 10^{-14}$, $x_0 = 0.160126 + 10^{-14}$.

(b) Imagen desencriptada $x_0 = 0.125148$, $x_0 = 0.160126$.

Figura 4.10: XOR, con puntos semilla modificado.



(a) Imagen desencriptada $\lambda = 3.999916 + 10^{-14}$.

(b) Imagen desencriptada $\lambda = 3.999916$.

Figura 4.11: XOR, con λ modificado.

Dadas las llaves correctas se logra la desencriptación de las imágenes como se visualiza en 4.8(b), 4.9(b), 4.10(b), y 4.11(b), sin embargo no se obtiene una desencriptación correcta aún cuando el cambio en las llaves es muy pequeño.

4.4.2. Entropía de Shannon

La entropía de Shannon es utilizada para medir la cantidad de información contenida en un mensaje, mientras que para una imagen estima los datos normales para cada bit de la imagen, véase el artículo [3].

- La entropía debe ser mínima, si no existe aleatoriedad.
- La entropía debe ser máxima, si la secuencia es aleatoria.

En la teoría, un vector de probabilidad p es una secuencia de un número finito de números no negativos $\{p_1, p_2, p_3, ..., p_n\}$ que cumple $\sum_{i=1}^{n} p_i = 1$. Si log_2 denota el logaritmo en base 2, se define la entropía de Shannon de p como:

$$H(p) = -\sum_{i=1}^{n}(p_i)log_2(p_i)$$

.

En la fórmula anterior, es una convención usual considerar que $0log_20 = 0$. En este trabajo la entropía debe de ser cercana a 8, ya que se tienen imágenes de 8 bits. Lo anterior debido a que los 256 números están al estar igualmente representados. Por ser justamente un vector aleatorio, se tendrá

una probabilidad de $\frac{1}{2^8}$ para cada elemento; Calculando la la entropía, se tiene:

$$H = -\sum_{i=0}^{255} \frac{1}{2^8} log_2(\frac{1}{2^8}) = -256(\frac{1}{2^8})(-8) = -(-8) = 8.$$

En la Tabla 4.6 se muestran los cálculos realizados en el programa R. Se nota que la entropía para las imágenes encriptadas con ambos métodos (Módulo 1, XOR) es muy cercana al valor esperado de 8, con una diferencia menor a 0.01 mientras que para las imágenes 4.2(a) y 4.3(a) el valor de la entropía se encuentra más alejado; Por lo tanto, la sucesión de píxeles correspondientes a las imágenes encriptadas es una secuencia aleatoria.

Imagen	Original	Módulo 1	XOR
4.2(a)	7.870195	7.999795	7.992046
4.3(a)	6.522237	7.99991	7.992091

Tabla 4.6: Entropía.

4.4.3. Tasa de cambio de píxeles

La tasa de cambio de número de píxeles o NPCR de Number of Pixels Change Read (por sus siglas en inglés), indica el % de píxeles que presentan un cambio en el tono de gris.

Dado que se compararon tonos de gris, se aplica la transformación

$$[256x_i] \text{ donde} [a] = \text{parte entera de a.}$$

Con lo anterior se obtienen números enteros entre 0 y 255, mismos que codifican los 256 tonos de gris. Se aplica el mismo procedimiento a la matriz encriptada definiendo:

$$d(i,j) = \begin{cases} 1 & \text{si } C_1(i,j) \neq C_2(i,j), \\ 0 & \text{si } C_1(i,j) = C_2(i,j) \end{cases}$$

donde C_1 y C_2 son las matrices de la imagen original y encriptada, respectivamente después de aplicar $[256x_i]$. Para mayor detalle revisar [6]. La tasa de cambio es:

$$NPCR = \sum_{i,j} \frac{d(i,j)}{S} \times 100\,\%,$$

donde S es el número total de píxeles de la imagen original.

Para las imágenes 4.2(a) y 4.3(a), se tiene que el cambio corresponde casi al 100 %, como se ve en la tabla 4.8.

Imagen	Módulo 1	XOR
4.2(a)	99.60022 %	99.60304 %
4.3(a)	99.60444 %	99.60943 %

Tabla 4.7: NPCR.

Mientras que, para la imagen original y la imagen desencriptada el porcentaje de cambio es menor, con las llaves correctas.

Imagen	Módulo 1	XOR
4.2(a)	0.0001127091 %	0.0 %
4.3(a)	0.0 %	0.0 %

Tabla 4.8: NPCR entre imagen original y la imagen desencriptada.

Como complemento en la Tabla 4.9 se muestra el tiempo que ocupa el programa R en generar distintas órbitas seudoaletorias con ambos mapeos, correspondientes al tamaño de las órbitas generadas para las simulaciones de números seudoaletorios, así como la encriptación de las imágenes presentadas en está tesis.

Iteraciones	Mapeo Logístico	Mapeo Logístico Modificado
10,000	0.16s	0.14s
100,000	0.15s	0.19s
908880	0.31s	0.36s
2383936	0.67s	1.25s

Tabla 4.9: Tiempo órbita seudoaletoria

Se obtienen que el tiempo de ejecución es pequeño, aun cuando se aumente el número de iteraciones.

También se obtuvo el tiempo de ejecución de los métodos Módulo 1 y XOR utilizando las imágenes propuestas en este trabajo:

Imagen	Módulo 1	XOR
4.2(a)	8.17s	10.97s
4.3(a)	21.36s	26.24s

Tabla 4.10: Tiempo de encriptación

Los tiempos de ejecución obtenidos para ambos métodos fueron relativamente pequeños, aunque ligeramente superiores a los reportados en el artículo [2]. A pesar de utilizar un sistema de cómputo con recursos limitados, los tiempos se mantuvieron dentro de rangos aceptables, lo que sugiere un rendimiento óptimo de los algoritmos.

4.5. Pruebas estadísticas NIST

Otra de las aplicaciones de las secuencias seudoaleatorias es la generación secuencias de bits. Para establecer la seudolaleatoriedad de las secuencias de bits generadas el Instituto Nacional de Estándares y Tecnología de Estados Unidos NIST, por sus siglas en inglés National Institute of Standards and Technology cuenta con 15 pruebas estadísticas para el análisis de generadores de números seudoaletorios. Debido a la extensión requerida para la descripción y análisis de las 15 pruebas en su totalidad, en este trabajo se presentan únicamente los resultados de 4 de ellas. Se deja al lector la referencia [5] para consultar las pruebas restantes.

- **Prueba de frecuencia**

  Esta prueba determina si las proporciones de ceros y unos de una secuencia son aproximadamente iguales a lo esperado para una secuencia aleatoria. En la prueba se evalúa la cercanía de la proporción de unos a un 50 %. En otras palabras, se espera obtener aproximadamente el mismo número de ceros y unos.

  Sea $\epsilon = \{\epsilon_1, \epsilon_2, ..., \epsilon_n\}$ una secuencia binaria de ceros y unos de tamaño n, se usa una conversión de ceros y unos a -1 y 1, lo cual se logra mediante lo siguiente: $X_i = 2\epsilon_i - 1$. El estadístico de prueba a considerar es

$$S_{obs} = \frac{|S_n|}{\sqrt{n}}, \text{ donde} S_n = \sum_{i=1}^{n} X_i$$

donde la distribución a utilizar es la normal estándar.

Para determinar el valor-p se utiliza la función de error complementaria erfc(x), descrita de la siguiente forma:

$$erfc(\frac{S_{obs}}{\sqrt{2}}) = \frac{2}{\sqrt{\pi}} \int_{\frac{S_{obs}}{\sqrt{2}}}^{\infty} e^{-u^2} du$$

- **Prueba de frecuencia dentro de un bloque**

  Dada una cadena binaria $\epsilon = \{\epsilon_1, \epsilon_2, ..., \epsilon_n\}$, se consideraran bloques de tamaño M, donde M es un número natural. Para determinar si la frecuencia de unos en un bloque de M bits es próximo a $\frac{M}{2}$ (como se esperaría en una secuencia aleatoria).

  Se define $N = [\frac{n}{M}]$, donde $[a]$ es la parte entera de a.

  Se determina la proporción π_i de unos en cada bloque de M-bits
  $\pi_i = \frac{\sum_{j=1}^{M} \epsilon_{(i-1)M+j}}{M}$, así como el estadístico de prueba
  $\chi^2(Obs) = 4M \sum_{i=1}^{N} (\pi_i - \frac{1}{2})^2$, para esta prueba la distribución a utilizar χ^2.

  El valor-p se obtiene por:
  $igamc(\frac{N}{2}, \frac{\chi^2(Obs)}{2})$ donde

$$\frac{\int_{\chi^2(Obs)}^{\infty} e^{-u/2} u^{N/2-1} du}{\Gamma(N/2) 2^{N/2}} = \frac{\int_{\chi^2(Obs)/2}^{\infty} e^{-u} u^{N/2-1} du}{\Gamma(N/2)} = igamc(\frac{N}{2}, \frac{\chi^2(Obs)}{2})$$

  es la función Gama incompleta.

- **Prueba de rachas**

  Esta prueba determina si la oscilación de unos y ceros es demasiado rápida o lenta, es decir; si el número de ceros y unos de distintas distancias es el esperado para una secuencia aleatoria. Puesto que se considera una racha como una secuencia de k bits idénticos que está limitada antes y después por el bit opuesto. Por ejemplo, en la secuencia $\cdots 10001 \cdots$, la racha sería 000.

Para esta prueba en principio se determina $\pi = \frac{\sum_{j=1}^{n} \epsilon_j}{n}$ (descrito anteriormente) para una cadena de bits $\epsilon = \{\epsilon_1, \epsilon_2, ..., \epsilon_n\}$ de tamaño n, donde se pretende determinar si, $|\pi - \frac{1}{2}| \geq \tau$, ya que se tiene: $4 = \frac{|X - Np_0|}{\sqrt{Np_0q_0}} = \frac{|\hat{p} - p_0|}{\sqrt{\frac{p_0q_0}{N}}} = \frac{|\hat{p} - \frac{1}{2}|}{\sqrt{\frac{1}{4N}}} = \frac{|\hat{p} - \frac{1}{2}|}{\frac{1}{2\sqrt{N}}} = |\hat{p} - \frac{1}{2}| = \frac{4}{2\sqrt{N}} = \frac{2}{\sqrt{N}}$, entonces $|\hat{p} - \frac{1}{2}| \geq \tau$ y $\tau = \frac{2}{\sqrt{N}}$ de ser así no será necesario continuar con la prueba, en caso contrario se determina $\lim_{n\to\infty} P(\frac{V_n - 2n\pi(1-\pi)}{2\sqrt{n}\pi(1-\pi)} \leq z) \approx \phi(z)$, donde $\phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-\frac{u^2}{2}} dt$, se define el estadístico de prueba V_n cómo:

$V_n(obs) = \sum_{k=1}^{n-1} r(k) + 1$, con distribución χ^2

Para los casos:

$$r(k) = \begin{cases} 0 & \text{si } \epsilon_k = \epsilon_{k+1} \\ 1 & \text{si } \epsilon_k \neq \epsilon_{k+1} \end{cases}$$

valor-$p = erfc(\frac{|V_n(obs) - 2n\pi(1-\pi)|}{2\sqrt{2n}\pi(1-\pi)})$

- **Prueba de la racha más larga en un bloque**

  Para una secuencia binaria $\epsilon = \{\epsilon_1, \epsilon_2, ..., \epsilon_n\}$, de longitud n se busca determinar, si la longitud de la serie más larga de unos dentro de un bloque de M-bits de una secuencia probada es consistente con la longitud de la serie más larga de unos que se esperaría de una secuencia aleatoria.

  Para esta prueba la longitud M de cada bloque estará determinado por la tabla siguiente:

Mínimo n	M
128	8
6272	128
$570,000$	10^4

  Las frecuencias de las rachas más largas de unos se determinan como v_i, las cuales están categorizadas de acuerdo a la siguiente tabla, teniendo en cuenta la longitud de $M - bits$.

v_i	$M = 8$	$M = 128$	$M = 10^4$
v_0	≤ 1	≤ 4	≤ 10
v_1	2	5	11
v_2	3	6	12
v_3	≥ 4	7	13
v_4		8	14
v_5		≥ 9	15
v_6			≥ 16

Donde el estadístico de prueba $\chi^2(obs) = \sum_{i=0}^{k} \frac{(v_i - N\pi_i)^2}{N\pi_i}$. Los valores de K y N están determinados por la longitud M, de acuerdo con la información mostrada a continuación:

M	K	N
8	3	16
128	5	49
10^4	6	75

y π_i se determina mediante lo siguiente:

Clases	π_i
$\{v \leq 4\}$	$\pi_0 = 0.1174$
$\{v = 5\}$	$\pi_1 = 0.2430$
$\{v = 6\}$	$\pi_2 = 0.2493$
$\{v = 7\}$	$\pi_3 = 0.1752$
$\{v = 9\}$	$\pi_4 = 0.1027$
$\{v \geq 10\}$	$\pi_5 = 0.1124$

por último, el valor-$p = igamc(\frac{K}{2}, \frac{\chi^2(obs)}{2})$.

Cabe mencionar que la referencia para todas las tablas es el manual [5].

Estudio de la secuencia generadora propuesta

Los generadores de las órbitas seudoaleatoria d_i estudiados en este trabajo, están determinadas por el mapeo logístico, el mapeo logístico modificado y la secuencia seudoaleatoria generada por la encriptación

de las imágenes propuestas. Para la conversión de las secuencias se aplica el criterio descrito en [13], donde:

$$\epsilon_i = \begin{cases} 1 & \text{si } d_i \geq 0.5 \\ 0 & \text{si } d_i < 0.5 \end{cases}$$

A continuación se muestran los resultados obtenidos mediante el entorno y lenguaje de programación R. Para las pruebas descritas, se utilizaron las funciones de distribución disponibles en el software, y que pueden ser consultadas en apéndice de este trabajo.

El criterio para determinar que una secuencia sea seudoaletoria es:

$$\text{valor-}p > 0.01$$

.

Al aplicar las diferentes pruebas se obtienen los siguientes resultados:

Prueba	Mapeo logístico	Mapeo logístico modificado
Frecuencia	valor-$p = 0.1675866$	valor-$p = 0.9680931$
Frecuencia de bloques	valor-$p = 0.8035708$	valor-$p = 0.7872196$
Rachas	valor-$p = 0.5747547$	valor-$p = 0.8103426$
Racha más larga en un bloque	valor-$p = = 0.8233483$	valor-$p = = 0.8348184$

Tabla 4.11: Órbitas de tamaño $n = 10,000$ y punto semilla 0.1642925

La tabla 4.11 muestra que las pruebas aplicadas validan que la secuencia es seudoaletoria, pues se obtuvieron valores-p mayores a 0.01. Además se realizaron múltiples corridas con diferentes puntos semillas y diferentes tamaños de longitud n.

En la Tabla 4.12 se pueden observar los valores-p obtenidos al realizar las pruebas a la secuencias de encriptación.

Prueba	Módulo 1 4.2(c)	XOR 4.2(e)	Módulo 1 4.3(c)	XOR 4.3(e)
Frecuencia	0.1542358	0.4941649	0.8035892	0.2686323
Frecuencia de bloques	0.3406962	0.420667	0.6145382	0.6663809
Rachas	0.2165368	0.9242736	0.050001864	0.9830636
Racha más larga en un bloque	0.781948	0.6818323	0.4329842	0.2652996

Tabla 4.12: Secuencia aleatoria generada por las imágenes encriptadas

Se nota que se obtuvieron valores-p favorables, es decir la secuencia obtenida al encriptar una imagen por el método Módulo 1 y XOR es una secuencia seudoaletoria que pasa, al menos, las cinco pruebas, del El Instituto Nacional de Estándares y Tecnología de Estados Unidos, usadas por la autora de esta tesis.

Resultados

Las pruebas estadísticas realizadas a las órbitas pseudoaleatorias generadas por el mapeo logístico y el mapeo logístico modificado revelaron diferencias significativas en su calidad. La prueba de bondad de ajuste χ^2 demostró que el mapeo logístico modificado produce números pseudoaleatorios con una distribución más uniforme en el intervalo $[0, 1]$ en comparación con el mapeo logístico original.

Adicionalmente, la prueba de rachas confirmó la aleatoriedad de las secuencias generadas para cada punto semilla, al obtener valores-p superiores a 0.01, lo que indica que no se puede rechazar la hipótesis nula de aleatoriedad. Finalmente, el análisis de los exponentes de Lyapunov validó la propiedad de sensibilidad a las condiciones iniciales en ambos mapeos, un criterio fundamental para sistemas caóticos.

Las simulaciones realizadas con el método de Montecarlo y el triángulo de Sierpinski mostraron una mejora significativa en los resultados esperados para ambas metodologías.

Los resultados obtenidos en la encriptación de imágenes revelaron una distribución uniforme y aleatoria de los píxeles en la imagen encriptada. Los métodos empleados demostraron una alta sensibilidad a las condiciones iniciales, lo que los hace resistentes a los ataques de fuerza bruta, como se confirmó mediante cambios mínimos en las llaves. El análisis de correlación confirmó la independencia estadística entre los píxeles, y la prueba de la taza de cambio de píxeles reveló un cambio cercano al 100 % entre la imagen original y la encriptada.

Para mejorar la seguridad al encriptar la imagen propuesta, en este trabajo, se usan dos órbitas con diferentes puntos semilla x_0 y λ fijo con fin de hacer la combinación de las órbitas. Se dividieron los vectores involucrados en dos partes, coordenadas impares y coordenadas pares. Se encriptaron las coordenadas impares con ayuda de la primera órbita y, las coordenadas

pares con ayuda de la segunda órbita. El proceso inverso, que corresponde a desencriptar la imagen, repetimos a cada grupo lo descrito arriba.

Para evaluar la aleatoriedad de los números pseudoaleatorios generados, se implementaron 4 de las 15 pruebas propuestas por el El Instituto Nacional de Estándares y Tecnología (por sus siglas en inglés NIST), debido a las extensión que implicaría realizar todas las pruebas. En particular, para la prueba de bloques, se utilizaron bloques de 32 bits, tamaño determinado por la capacidad del programa R. La implementación de la prueba se realizó utilizando la función gamma del software.

Si bien se realizaron 4 de las 15 pruebas propuestas por el NIST, los resultados obtenidos sugieren que las secuencias binarias generadas por el mapeo logístico, el mapeo logístico modificado y las imágenes encriptadas exhiben un comportamiento aleatorio favorable en las pruebas realizadas. Sin embargo, se recomienda realizar la batería completa de pruebas del NIST para obtener una conclusión más sólida sobre la aleatoriedad de estas secuencias.

Una parte significativa de esta investigación fue presentada en el evento nacional ENOAN 2024, evento organizado por la Sociedad Mexicana de Cómputo Científico y sus Aplicaciones, donde se me otorgó el primer lugar en la categoría de licenciatura por el cartel titulado “Números pseudoaleatorios generados por el mapeo logístico modificado y su aplicación en la encriptación de imágenes ”.

Adicionalmente, la autora de esta tesis presentó parte de este trabajo de tesis en el Seminario de Investigación de la Maestría de Ciencias de la Complejidad así como, una ponencia titulada “El mapeo logístico modificado y su aplicación a la generación de números seudoaleatorios y encriptado de imágenes” en las XVI Jornadas de Modelación Matemática, realizadas del 21 al 24 de noviembre de 2024 en la UACM.

Conclusiones

La modificación del mapeo logístico permitió generar órbitas seudoaletorias capaces de ejemplificar mediante el método de Montecarlo, el triángulo de Sierpinski y la encriptación de imágenes cómo los conceptos teóricos se traducen en aplicaciones prácticas.

A través del análisis detallado de las órbitas seudoaletorias, se demuestra que tanto el mapeo logístico original como la modificación propuesta ofrecen una base sólida para la generación de números seudoaleatorios, que son cruciales en aplicaciones prácticas de encriptación.

Los ejemplos presentados evidencian la efectividad de estos mapeos en la encriptación de imágenes, lo cual se complementa con un extenso análisis estadístico de las imágenes resultantes. Las 4 pruebas de NIST, así como el estudio de la correlación y la tasa de cambio de píxeles, confirman que las imágenes encriptadas cumplen con estándares de seguridad elevados, garantizando así una robustez en la protección de la información.

Apéndice

A. Análisis Estadístico

La ejecución de los programas para el análisis estadístico fueronrea realizados en una LAPTOP-S2GHD9SQ con procesador Intel(R) Celeron(R) N4000 CPU 1.10GHz, RAM de 4.00 GB (3.82 GB utilizable) y sistema operativo de 64 bits (procesador $x64$). Las versiones del lenguaje de programación R y Python son:

- R para Windows 4.2.2
- Python Visual Studio Code Setp 1.76.0, empleando las librerías de numpy y matplotlib.pyplot.

A.1. Generador de números seudoaletorios

En el siguiente recuadro se muestra el código para el mapeo logístico modificado factor 10,000 como generador de números aleatorios en R.

```
LO <- vector(mode="numeric",length=10000)
r <- 3.9999

LO[1] <- 0.074135
for(i in 2:10000){LO[i] <- 10000*r*LO[i-1]*(1-LO[i-1])
-floor(10000*r*LO[i-1]*(1-LO[i-1]))}
```

Diagrama de telaraña

En el siguiente recuadro, se muestra el código del diagrama de telaraña utilizando el mapeo logístico $f(x) = \lambda x(1-x)$ en el software Python.

```
import numpy as np
import matplotlib.pyplot as plt

def fun(x):
    return(3.999*x*(1-x))

x=np.arange(0,1.05,0.05)
y=3.999*x*(1-x)
x0=0.6
x1=fun(x0)
n=100

plt.figure()
plt.plot(x,y,c="r")
plt.plot(x,x,c="b")
plt.plot([x0,x0],[0,x1],c="dodgerblue")
for i in range(n):
    x2=fun(x1)
    plt.plot([x0,x1],[x1,x1],c="dodgerblue")
    plt.plot([x1,x1],[x1,x2],c="dodgerblue")
    x0, x1= x1, x2
plt.show()
```

A.2. Uniformidad y Aleatoriedad

Código para la prueba de Uniformidad y Aletoriedad para órbitas de longitud 10,000; generadas por el mapeo logístico en el programa *R*, empleando la función "chisq.test" y el paquete "randtests".

```
LO <- vector(mode="numeric",length=10000)
r <- 3.9999

LO[1] <- 0.074135
for(i in 2:10000){LO[i] <- 10000*r*LO[i-1]*(1-LO[i-1])
-floor(10000*r*LO[i-1]*(1-LO[i-1]))}

# vector de fronteras de las 100 clases
c <- vector(mode="numeric",101)

for(i in 1:101){c[i] <- (i-1)/100 }
h <- hist(LO, breaks=c, right=TRUE, plot=TRUE) #histograma

# h[2] contiene la tabla de las frecuencias

h[2]

clases <- unlist(h[2], use.names = FALSE)

chisq.test(clases)
chisq.test(clases)$expected$

#Prueba de aleatoriedad

library(randtests)

runs.test(LO)  #Mapeo modificado
```

A.3. Exponentes de Lyapunov para puntos fijos

En el siguiente recuadro se observa el código empleando para obtener los exponentes de Lyapunov usando el programa Python.

```
#Como entrada, se ingresa la funcion, la derivada de la
#funcion, el valor inicial x y lambda el numero de
#iteraciones y por ultimo se calcula el promedio de
#los valores obtenidos por la derivada.

 r = 3.9999
 x = 1-(1/r)
 L = np.log(abs(r-2*r*x)))
 print(L)
```

Exponentes de Lyapunov

En el código siguiente se muestra el mapeo logístico para 100 iteraciones con 20,000 $\lambda \in (1, 4)$, y puntos semilla $x \in (0, 1)$ simulación recreada en Python.

```
a = 1
b = 4
n = 20000
L = np.linspace(a,b,n)
Arr = []
result = []

for r in L
    x = np.random.random()
    for n in range(100):
        x = r*x*(1-x)

    result = []

    for n in range(100):
        x = r*x*(1-x)
        result.append(np.log(abs(r-2*r*x)))
    Arr.append(np.mean(result))
```

B. Ejemplos de aplicación

B.1. Método de Montecarlo

Código para calcular la aproximación de la integral $\int_0^1 x^2 dx$, con los números seudoaleatorios generados por dos puntos semillas correspondien-

tes al eje x y al eje y, utilizando el mapeo logístico factor 10,000.

```
n <- 100000
a <- 0
b <- 1
M <- 1

LO <- vector(mode="numeric",length=n)
r <- 3.9999
LO[1] <- 0.613764

for(i in 2:n){LO[i] <- 10000*r*LO[i-1]*(1-LO[i-1])
-floor(10000*r*LO[i-1]*(1-LO[i-1]))}

t <- vector(mode="numeric",length=n)
t[1] <-  0.576897

for(i in 2:n){t[i] <- 10000*r*t[i-1]*(1-t[i-1])
-floor(10000*r*t[i-1]*(1-t[i-1]))}

xp <- a+(b-a)*LO
yp <- M*t
y <- function(x) {
  f <- x^2
   return(f)
}
fpi=y(xp)
u <- yp <= fpi
e <- vector(mode="numeric", length=n)
for(i in 1:n){
ifelse (yp[i] <= fpi[i], e[i] <-1, e[i]<-0)}

pb <- sum(sum(e))
area <- (M*(b-a)*pb)/n
plot(y,type="l", lwd=3, col="red")
points(xp,yp,pch=".")
```

B.2. Triángulo de Sierpinski

Código para obtener el triángulo de Sierpinski con una órbita de tamaño 100,000 generada por el mapeo logístico modificado, aplicando las funciones mencionadas en la sección 3.2.

```
LO <- vector(mode="numeric",length=100000)
r <- 3.9999
LO[1] <- 0.074135

for(i in 2:100000){LO[i] <- 10000*r*LO[i-1]*(1-LO[i-1])
-floor(10000*r*LO[i-1]*(1-LO[i-1]))
}

a <- vector(mode="numeric",length=100000)
a[1] <- 0.4

b <- vector(mode="numeric",length=100000)
b[1] <- 0.4

for(i in 2:100000){ifelse (LO[i] <1/3, a[i] <- a[i-1]/2,
ifelse(O[i] <2/3, a[i] <- a[i-1]/2  + 1/2, a[i] <- a[i-1]/2
)
)
}

for(i in 2:100000){ifelse (LO[i] <1/3, b[i] <- b[i-1]/2,
ifelse(O[i] <2/3, b[i] <- b[i-1]/2  , b[i] <- b[i-1]/2 + 1/2
)
)
}

plot(a,b,pch=".")
```

C. Encriptación y análisis estadístico

C.1. Encriptación, método Módulo 1

Código para aplicar el método de encriptación utilizando una sola órbita generada por el mapeo logístico y una imagen de 820×1082 píxeles en tonos de gris, mediante la suma Módulo 1, empleando el software *R*.

```
#En R se utiliza la funcion "floor" para quitar
la parte entera.

file.choose()
library("png") #para abrir png'sr

A <- readPNG( "C:\\Users\\shara\\Downloads\\
Imagen1gray (1).png")

class(A)

B <- A[, ,1]
dim(B)

# Generador del mapeo

n <-  820*1082
seed <- 0.440629

r1 <-3.999916
C <- vector(mode="numeric",n)

C[1] <- seed
for(i in 2:n){
       C[i] <- 10000*r1*C[i-1]*(1-C[i-1])-
       floor(10000*r1*C[i-1]*(1-C[i-1]))
}

D <- matrix(C,820,) # se convierte en matriz 360x250
dim(D)

F <- c(B)
Z <- C+F-floor(C+F)
z1 <- c(Z)
Q <- matrix(Z,820,)
```

Desencriptado, Módulo 1

Código para la desencriptación factor 10,000 utilizando las llaves proporcionadas.

```
seed1 <- 0.440629
r2 <-3.999916

C <- vector(mode="numeric",n)
C[1] <- seed1
for(i in 2:n){
       C[i] <- 10000*r2*C[i-1]*(1-C[i-1])
       -floor(10000*r2*C[i-1]*(1-C[i-1]))
}

D <- matrix(C,820,)

#Para desencriptar
E <- vector(mode="numeric",n)
for(i in 1:n){ifelse(z1[i]>C[i], E[i] <- z1[i]-C[i],
               E[i] <- z1[i]-C[i]+1 )}
```

C.2. Encriptación, método XOR

Código del método de encriptación utilizando una sola órbita generada por el mapeo logístico y una imagen de 820×1082 píxeles en tonos de gris empleando el paquete ”bitops”del programa R.

```
file.choose()
library("png") #para abrir png'sr

A <- readPNG( "C:\\Users\\shara\\Downloads
\\Imagen1gray (1).png")

class(A)

B <- A[, ,1]
dim(B)

F <- c(B)

# Generador del mapeo

n <-  820*1082
seed <- 0.440629
```

```
r1 <-3.999916
C <- vector(mode="numeric",n)

C[1] <- seed
for(i in 2:n){
       C[i] <- 10000*r1*C[i-1]*(1-C[i-1])-
       floor(10000*r1*C[i-1]*(1-C[i-1]))
}

# Encriptar
library(bitops)

K  <- vector(mode="numeric",n)
for(i in 1:n){K[i] <- bitXor((2^8)*B[i],(2^8)*C[i] )}
```

Desencriptado método, XOR

Código para realizar el desencriptado imágenes, aplicando al nuevo vector generado por el mapeo logístico modificado.

```
seed1 <- 0.440629
r2 <-3.999916

C <- vector(mode="numeric",n)
C[1] <- seed1
for(i in 2:n){
       C[i] <- 10000*r2*C[i-1]*(1-C[i-1])
       -floor(10000*r2*C[i-1]*(1-C[i-1]))
}


#Para desencriptar

L  <- vector(mode="numeric",n)

for(i in 1:n){L[i] <- bitXor(K[i],(2^8)*C[i] )}

Uk1 <- L/max(L)
```

D. Análisis estadístico de la encriptación

En esta sección se muestra los códigos de las pruebas estadísticas aplicadas a la Figura 4.2(a) en el software R.

D.1. Aleatoriedad y Uniformidad Kolmogorov-Smirnov

```
# Kolmogorov-Smirnov funcion ks.test() en R
# Aleatoriedad
library(randtests)

runs.test(C)  #Mapeo
runs.test(K) #Imagen encriptada

#Uniformidad

#Elimina valores repetidos
Uk <- unique(K)


ks.test(Uk,punif)
```

D.2. Uniformidad χ^2

```
# Uniformidad utilizando la funcion chisq.test de R
# vector de fronteras de las 100 clases

ji1 <- vector(mode="numeric",256)

for(i in 1:256){ji1[i] <- (i-1)/256 }
h <- hist(K, breaks=seq(-0.5,255.5,by=1),
right=TRUE, plot=TRUE)

# h[2] contiene la tabla de las frecuencias

h[2]

clases <- unlist(h[2], use.names = FALSE)

chisq.test(clases)
```

D.3. Correlación

```
n2 <- n-1
R1 <- u[1:n2]
S1 <- C[1:n2]

# Matriz encriptada
```

```
R <- vector(mode="numeric",n2)
for(i in 1:n2){R[i] <- Uk[i+1]}

cor.test(R1,R) #sobre la misma matriz normal

# Imagen original
S <- vector(mode="numeric",n2)
for(i in 1:n2){S[i] <- C[i+1]}

cor.test(S1,S) #sobre la misma matriz

# Imagen original-Imagen encriptada
cor.test(C,Uk)

# Diagramas

SS1 <- floor(S1*256)
SS <- floor(S*256)
RR1 <- floor(R1*256)
RR <- floor(R*256)
NP1 <- floor(C*256)
NP <- floor(Uk*256)

plot(SS1,SS,pch=".") #sobre la matriz Original
plot(RR1,RR,pch=".") #sobre la matriz Encriptada
plot(NP1,NP,pch=".") #sobre la matriz Original-Encriptada
```

D.4. Entropía

```
# Para el mapeo generado
hisIm <- hist(C, breaks=0:256/256, right=TRUE, plot=TRUE,
freq = FALSE)$counts$ # histograma

hisI <-hisIm/length(C)

# Manualmente
entoI <- sum((hisI)*log(hisI, base=2))
-entoI

#Imagen encriptada
his <- hist(Uk, breaks=0:256/256, right=TRUE, plot=TRUE,
freq = FALSE)$counts$ #histograma
his1 <-his/length(Uk)
ento <- sum((his1)*log(his1, base=2))
-ento
```

```
# Ocupando el paquete
library(entropy)
entropy(his1,unit="log2")
entropy(hisI,unit="log2")
```

D.5. Tasa de cambio de píxeles

```
NP1 <- floor(C*256) # Mapeo modificado
NP <- floor(Uk*256) # Imagen encriptada
NPen <- floor(Uk1*256) # Imagen desencriptada

NPC <- vector(mode="numeric", length=n)
for(i in 1:n){
ifelse (NP1[i] == NP[i] , NPC[i] <-0, NPC[i]<-1)}
NPCR <- (sum(NPC)/length(C))*100
NPCR

NPCE <- vector(mode="numeric", length=n)
for(i in 1:n){
ifelse (NP1[i] == NPen[i] , NPCE[i] <-0, NPCE[i]<-1)}
NPCR1 <- (sum(NPCE)/length(NP1))*100
NPCR1
```

E. NIST

En los siguientes apartados se muestran los códigos correspondientes a la prueba de frecuencia, frecuencia de bloques, racha y la racha más larga descritas en la sección 4.5.

Se generó la secuencia seudoaleatoria con el mapeo logístico $f(x) = \lambda x(1-x)$ y se transforma para obtener la nueva secuencia binaria.

```
nt <- 10000
r <- 3.999916
PRNG <- vector(mode="numeric",nt)
PRNG[1] <- 0.16558
for(i in 2:nt){PRNG[i] <- r*PRNG[i-1]*(1-PRNG[i-1])
}

eps <- vector(mode="numeric", length=nt)
for(i in 1:nt){
ifelse (PRNG[i] >= 0.5, eps[i] <-1, eps[i]<-0)}
```

E.1. Frecuencia

Código para calcular la frecuencia de la secuencia binaria generada por el mapeo logístico.

```
#####  Frecuencia  #######
# Paquete pracma para llamar a la
#funcion de error complementaria (erfc)

n <- length(eps)

Sn <- vector(mode="numeric", n)
for(i in 1:n){
ifelse (eps[i] == 0, Sn[i] <- -1, Sn[i]<- eps[i])}

SSn <- sum(Sn)
Sob <- abs(SSn)/sqrt(n)

library(pracma)
erfc(Sob/sqrt(2))
```

E.2. Frecuencia dentro de un bloque

Código para calcular la frecuencia dentro de un bloque de la secuencia binaria generada por el mapeo logístico descrita en la sección 4.5.

```
## Frecuencia de Bloques ##

M1 <- 32
N1 <- floor(n/M1)

pii1 <- vector(mode="numeric",N1)
for(i in 1:N1){pii1[i] <- sum(eps[32*i-31],eps[32*i-30],
eps[32*i-29],eps[32*i-28],eps[32*i-27],eps[32*i-26],
eps[32*i-25],eps[32*i-24],eps[32*i-23],eps[32*i-22],
eps[32*i-21],eps[32*i-20],eps[32*i-19],eps[32*i-18],
eps[32*i-17],eps[32*i-16],eps[32*i-15],eps[32*i-14],
eps[32*i-13],eps[32*i-12],eps[32*i-11],eps[32*i-10],
eps[32*i-9],eps[32*i-8],eps[32*i-7],eps[32*i-6],
eps[32*i-5],eps[32*i-4],eps[32*i-3],eps[32*i-2],
eps[32*i-1],eps[32*i])
}

X3obs <- 4*M*sum(((pii1/M1)-(1/2))^2)
T3p <- X3obs/2
T3 <- N1/2
```

```
1-pgamma(T3p,T3)
```

E.3. Rachas

Código para la prueba de rachas fueron utilizados algunos de los elementos de la prueba de frecuencia, como se observa en el script.

```
######## Rachas  #########

pi1 <- sum(eps)/n
pre <- abs(pi1-1/2)
pre < Sob

Vobs <- vector(mode="numeric",n)
for(i in 1:n)
{ifelse(eps[i] == eps[i+1], Vobs[i]<-0, Vobs[i]<-1)
}

Vobsn <- sum(Vobs)+1
er <- abs(Vobsn-(2*n*pi1*(1-pi1)))/(2*sqrt(2*n)*pi1*(1-pi1))

erfc(er)
```

E.4. Racha más larga en un bloque

Código para calcular la racha más larga; la prueba se determinó de acuerdo al manual [13], al igual que los π_i utilizados en este script.

```
nt <- length(eps)
ML <- 128
epsL <- eps[1:6272]
length(epsL)
kL <- 5
NL <- 49

B <- matrix(epsL,ML,NL)
B1 <- t(B)

dim(B1)
C <- matrix(0,NL,ML)
dim(C)
for(i in 1:NL){C[i,1] <- B1[i,1]  }
```

```
for(i in 1:NL){for(j in 2:ML){ifelse(B1[i,j] == 1,
C[i,j] <- C[i,j-1]+1, C[i,j] <- 0 ) }}

MAX <- vector(mode="numeric",NL)

for(i in 1:NL){MAX[i] <- max(C[i,])}
MAX
length(MAX)
V1 <- as.numeric(MAX<=4)
v1s <-sum(V1)
V2 <- as.numeric(MAX==5)
v2s <- sum(V2)
V3 <- as.numeric(MAX==6)
v3s <- sum(V3)
V4 <- as.numeric(MAX==7)
v4s <- sum(V4)
V5 <- as.numeric(MAX==8)
v5s <- sum(V5)
V6 <- as.numeric(MAX>=9)
v6s <- sum(V6)
p1 <- 0.1174
p2 <- 0.2430
p3 <- 0.2493
p4 <- 0.1752
p5 <- 0.1027
p6 <- 0.1124

sum1 <- (v1s-NL*p1)^2/(NL*p1)
sum2 <- (v2s-NL*p2)^2/(NL*p2)
sum3 <- (v3s-NL*p3)^2/(NL*p3)
sum4 <- (v4s-NL*p4)^2/(NL*p4)
sum5 <- (v5s-NL*p5)^2/(NL*p5)
sum6 <- (v6s-NL*p6)^2/(NL*p6)

obs <- sum(sum1,sum2,sum3,sum4,sum5,sum6)
obs
TL <- obs/2
TL1 <- kL/2
TL1
1-pgamma(TL,TL1)
```

Bibliografía

[1] Abdullah, R. M., y Abrahim, A. R. Review of image encryption using different techniques. *Academic Journal of Nawroz University*, 11(3):170–177, 2022.

[2] Askar, S. S., Karawia, A. A., Al-Khedhairi, A., y Al-Ammar, F. S. An algorithm of image encryption using logistic and two-dimensional chaotic economic maps. *Entropy*, 21(1):44, 2019.

[3] Balibrea, F. On clausius, boltzmann and shannon notions of entropy. *Journal of Modern Physics*, 7(02):219, 2016.

[4] Banks, J. , Books, J.,Cairns, G., Davis, G., yStacey, P. On devaney's definition of chaos. *The American mathematical monthly*, 99(4):332–334, 1992.

[5] Bassham, E. L, Rukhin, A. L., Soto, J., Nechcatal, J. R.,Smid, E. M.Leigh, S. D., Levenson, M., Vangel, M., Heckert, A. N.Banks, L. D. *A statistical test suite for random and pseudorandom number generators for cryptographic applications*. US Department of Commerce, Technology Administration, National Institute of . . . , 2001.

[6] Bowen,L. Z., Lingfeng, L. Chaos-based image encryption: review, application, and challenges. *Mathematics*, 11(11):2585, 2023.

[7] Daniilidis, A. *Espacios Métricos*. EDUNI, 2020.

[8] Dávalos J. E. K., y Lango, H. M. *Sistemas Dinámicos Discretos*. UNAM, Facultad de Ciencias, 2014.

[9] Giordano, R. F., William, P. F., y Horton, B. S. *A First Course in Mathematical Modeling*. Cengage Learning, 2013.

[10] Gonzalez, R. C., Woods, R. E. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., 1987.

[11] Gonzáles, J. J., Guerra, N., Quintana, M.P., y Santana, A. Estadística. https://www2.ulpgc.es/hege/almacen/downloand/30/30806/ ftemallcontrastesnoparametricos.pdf. Use la fecha de acceso.

[12] Lok K. *The Lyapunov Exponent Test and the 0-1 Test for Chaos Compared*. PhD thesis, Faculty of Science and Engineering, 2016.

[13] Lazaros, M. V., Jafari, C., Munoz-Pacheco, S., Kengne, M. J., Rajagopal, J, y Karthikeyan-Stouboulos, Ioannis. Modification of the logistic map using fuzzy numbers with application to pseudorandom number generation and image encryption. *Entropy*, 22(4):474, 2020.

[14] Devaney R. L. *An Introduction to Chaotic Dynamical Systems*. CRC press, 2021.

[15] Méndez-Lango, H. On intervals, sensitivity implies chaos. *Revista Integración, temas de matemáticas*, 21(1 y 2):15–23, 2003.

[16] Holmgren R. *A First Course in Discrete Dynamical Systems*. Springer Science and Business Media, 2000.

[17] Sternberg S. *Dynamical Systems*. Courier Corporation, 2010.

[18] Talanquer, V. *Fractus, Fracta, Fractal*. FCE-Fondo de Cultura Económica, 1996.

[19] Triola, M. F. *Probabilidad y estadística*. Pearson educación, 2004.

[20] Vargas, J. A. Algebra clásica. *Publicaciones electrónicas, Sociedad Matemática Mexicana, Serie: Textos*, 7, 2013.

[21] Varshney, H., Gupta, H., y Kushwaha, M. Image encryption using chaotic logistic map. *International Journal of Electrical y Computer Science Engineering*, 4(4):40–45, 2017.

[22] Vellekoop, M., Berglund, R. On intervals, transitivity = chaos. *The American Mathematical Monthly*, 101(4):353–355, 1994.

[23] Wackerly, D. D., Mendehall III, W. S., y Scheaffer, R. L. *Mathematical Statistics with Applications*. Cengage Learning, 2014.

[24] Zhao, Y., Liu, L. A bit shift image encryption algorithm based on double chaotic systems. *Entropy*, 23(9):1127, 2021.