



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

CENTRO DE CIENCIAS MATEMÁTICAS
UNIDAD MORELIA

REPRESENTACIÓN DEL ADN MEDIANTE EL JUEGO DEL
CAOS: CLASIFICACIÓN DE IMÁGENES Y DESARROLLO
DE UNA APP DE DIVULGACIÓN

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

MAESTRA EN CIENCIAS MATEMÁTICAS

PRESENTA:

MARIEL GUADALUPE GUTIÉRREZ CHAVESTE
mchaveste@matmor.unam.mx

DIRECTORA:

DRA. NELLY SÉLEM MOJICA
nselem@matmor.unam.mx



Morelia, Michoacán, 2024

Dedicada a mí

Agradecimientos

Quiero agradecer a todos los seres que me apoyaron de muchas maneras para que fuera posible este proyecto, desde sus inicios, la parte más compleja que fue su desarrollo y ahora su conclusión, quiero comenzar con expresar mi profunda gratitud hacia mi amado Berserker que, si bien, nunca entenderá este trabajo, su participación y absoluto amor durante el proceso fueron cruciales para tener la fuerza de continuar cada día. A mi asesora Nelly, destacable investigadora, cuya disciplina, trabajo y humanidad admiraré siempre pues me fueron de gran inspiración, al doctor Pedro por haber plantado esa semilla de curiosidad sobre un tema que se volvería tan apasionante para mí así como retador. A mi gente que son mi madre y mi hermano, por todo su apoyo y compañía, ustedes son mi origen y siempre estaré orgullosa de cada uno de nosotros. A mi amada amiga Dany que estuviste conmigo y confiaste en mí desde antes que pudiéramos vernos de frente. A Migue y a todo el 'club de señoritas' que me recibieron con los brazos abiertos y me permitieron conocer la experiencia de vivir con perspectivas más que diversas. A ti, Luis, pues nuestra amistad ha tenido una larga trayectoria y ambos hemos sido testigos de nuestros peores y mejores momentos de evolución, gracias por compartir conmigo también el estudiar una maestría. A Miguel Magaña y Luis Gerardo por su infinita paciencia y apoyo en muchos de los aspectos más técnicos de este trabajo, gracias por compartirme sus amplios conocimientos.

Alnitak

Betterlab

Me disculpo con todas aquellas personas, cuyos nombres no mencioné, ya sea que hayan sido un capítulo en mi vida, o bien, sean amistades que tienen un lugar especial en mi corazón, sin embargo, sepan que les extiendo mis agradecimientos ya que, sus participaciones en mi cotidianidad han ayudado a forjar la persona que hoy soy.

Este trabajo se realizó gracias al apoyo de la DGAPA-UNAM a través del financiamiento del

proyecto PAPIIT IN114323.

Índice general

Agradecimientos	II
1 Introducción	1
§1.1 Fractal	1
§1.2 El Juego del Caos	2
§1.3 Juego del Caos Restringido	6
§1.4 Genoma	10
§1.5 Representación del Juego del Caos para Secuencias de ADN (CGR por sus siglas en inglés)	13
2 Objetivo General	16
§2.1 Objetivos Específicos	16
§2.1.1 Objetivo 1. Representar geométricamente propiedades de secuencias de ADN.	16
§2.1.2 Objetivo 2. Crear una herramienta de visualización para la comunicación científica del juego del caos.	17
3 Resultados	18
§3.1 Representación geométrica de propiedades de secuencias de ADN.	18
§3.1.1 Descarga de muestras de bacterias clínicamente relevantes.	18
§3.1.2 Generación del código del juego del caos	19
§3.1.3 Generación de simulaciones de secuencias específicas con distintos sesgos y distribuciones de nucleótidos.	20

§3.1.4 Representación visual de las frecuencias de k-meros en secuencias de ADN.	25
§3.2 Obtención de propiedades de las representaciones e investigación de su capacidad de clasificación en distintas categorías biológicas.	26
§3.2.1 Contenido de GC en las secuencias a partir de las CGR's.	26
§3.2.2 Dimensión fractal de correlación de las imágenes.	27
§3.2.3 Uso de las características obtenidas para implementar un algoritmo de clasificación.	29
§3.3 Creación de una herramienta de visualización para la comunicación científica del juego del caos.	29
§3.3.1 Programación del algoritmo del juego del caos para polígonos de n lados con el framework AngularJS.	30
§3.3.2 Adaptación del algoritmo para generar restricciones en el juego del caos. .	31
§3.3.3 Programación de la versión del juego del caos para secuencias de ADN. .	32
§3.3.4 Desarrollo de la interfaz de usuario para manipulación de parámetros mediante la App.	33
§3.3.5 Desarrollo de un programa en Python que genere la CGR de una secuencia dada.	34
4 Materiales y métodos	35
§4.1 Metodología 1. Representación visual de las frecuencias de k -meros en secuencias de ADN.	36
§4.1.1 Descarga de muestras de bacterias clínicamente relevantes.	36
§4.1.2 Generación del código del juego del caos	36
§4.1.3 Generación de simulaciones de secuencias específicas con distintos sesgos y distribuciones de nucleótidos.	38
§4.1.4 Representación visual de las frecuencias de k -meros en secuencias de ADN.	40
§4.2 Metodología 2. Obtención de propiedades de las representaciones.	41
§4.2.1 Dimensión fractal de correlación de las imágenes.	41

§4.2.2 Utilizar las características obtenidas para implementar un algoritmo de clasificación.	41
§4.3 Metodología 3. Creación de una herramienta de visualización para la comunicación científica del juego del caos.	41
§4.3.1 Programa del algoritmo del juego del caos para polígonos de n lados con el framework AngularJS.	41
§4.3.2 Adaptación del algoritmo para generar restricciones en el juego del caos.	42
§4.3.3 Programa de la versión del juego del caos para secuencias de ADN.	43
§4.3.4 Desarrollo de la interfaz de usuario para su interacción con la App.	43
5 Perspectiva/Discusión	46
§5.1 Representaciones Fractales y Clasificación	46
§5.2 Aplicación y Desarrollo del Software	47

Índice de figuras

1.1	Interpretación geométrica del algoritmo del juego del caos 1.2.1.	4
1.2	Una representación 'decente' del <i>triángulo de Sierpinski</i> se obtiene al iterar el algoritmo, un total de 2000 veces	4
1.3	En esta figura se ilustra lo que fue descrito anteriormente, es decir, el Juego del Caos en un cuadrado con $f = \frac{1}{2}$, visualmente se aprecia lo que parece ser sólo 'ruido' y esto se debe a que los puntos de distribuyen de manera aleatoria y uniforme sin ninguna tendencia o restricción en particular. Como veremos más adelante, se pueden aplicar ciertas restricciones que generarán patrones fractales para ciertos casos.	5
1.4	El Juego del Caos al ser aplicado en un polígono de regular de cinco lados y un valor de $f = 0.618$ obtenemos este fractal, por lo que es fácil deducir que el valor de f juega un papel muy importante en la distribución de los puntos en el polígono.	5
1.5	Este fractal se obtiene aplicando el algoritmo a un polígono regular de once lados, un factor de $r = 0.777$ y sin restricciones.	6
1.6	Esta representación tiene un comportamiento fractal, se obtiene con la restricción antes mencionada con los parámetros: número de lados = 4 y factor $f = 0.5$. . .	7
1.7	Este fractal se obtiene aplicando el algoritmo a un polígono regular de cinco lados, un factor de $f = 0.5$ y la restricción de no repetir el mismo vértice dos iteraciones consecutivas.	7
1.8	Este fractal se obtiene aplicando el algoritmo a un polígono regular de siete lados, un factor de $f = \frac{2}{3} \sim 0.6667$ y la restricción de no repetir el mismo vértice dos iteraciones consecutivas.	7

1.9 Este fractal se obtiene al aplicar el juego del caos cuadrado con $f = \frac{1}{2}$ y la restricción 1.3.2 que, parafraseando, consiste en evitar que el vértice que se seleccionará se encuentre a una rotación (en sentido horario) del vértice seleccionado en la iteración anterior	8
1.10 Este patrón se obtiene al aplicar el juego del caos con $f = 0.7$ y la restricción 1.3.2..	8
1.11 Esta figura muestra unos elegantes patrones de autosemejanza cuando consideramos la restricción 1.3.3, que básicamente consiste en evitar que el vértice actual se encuentre a dos rotaciones (en sentido horario) del vértice elegido en la iteración anterior que para el caso del cuadrado que sea el vértice de la diagonal opuesta.	9
1.12 El resultado de la restricción 1.3.4 es una representación del fractal de Vicsek .	10
1.13 En esta figura vemos el comportamiento del algoritmo con la restricción 1.3.4 en un polígono de 16 lados y un $factor = 0.666$	10
1.14 Representación molecular del ADN (imagen «Nucleotides and Bases - Genetics Generation — knowgenetics.org», s.f.)	11
1.15 Ejemplo de archivo 'crudo' de una muestra de genoma secuenciado. En la primer linea encontramos información acerca del read, en la segunda el read, la tercera el símbolo '+' que funge como conector de la segunda y cuarta linea, en esta última los símbolos representan las calidades del read.	12
1.16 Computacionalmente, un genoma secuenciado se puede manejar como una cadena de caracteres y suele encontrarse con formato .fasta. En esta figura podemos visualizar las primeras 16 lineas de la secuencia de ADN ya procesada de una muestra de genoma de bacteria.	12
1.17 Representación conceptual del Juego del Caos aplicado a secuencias de ADN con la secuencia de juguete 'AGGTCG' que indica la posición en que se dibujará el punto de esa iteración, es decir, la posición del punto p_0 es el centro del cuadrado, la posición de p_1 es a mitad del 'camino' entre p_0 y el vértice A que es la letra en la posición 1 de la secuencia, luego p_2 se ubica a mitad de camino entre p_1 y la letra de la posición 2 en la secuencia, que es 'G', y así sucesivamente.	13

1.18	Esta figura se obtuvo al correr el algoritmo del juego del caos para los primeros 10000 pares de bases del genoma completo de mitocondria NS06 aislada de homo sapiens (GenBank: GU170820.1) en una de las primeras versiones de la App de divulgación acerca del juego del caos.	14
1.19	El algoritmo de generación de las imágenes se programó para que una densidad mayor de puntos en determinada zona, se tradujera en una tonalidad más clara en el color, fenómeno que podemos observar en estas representaciones con cantidades de puntos (longitudes de secuencia) muy distintas. Izquierda con casi 250 millones de pares de bases y en la derecha con más de 2 millones de pares de bases.	15
3.1	Se creó una base de datos que recopila la información relevante para este estudio de las 5,554 muestras y sus representaciones.	19
3.2	Aquí podemos observar la representación del juego del caos para ocho géneros de bacterias y notamos variaciones sutiles y no tan sutiles entre los patrones obtenidos para los distintos géneros	20
3.3	Las secuencias fabricadas que generan estas representaciones constan de una aleatoriedad con una distribución uniforme (izquierda) y una con pesos de probabilidad asignados que son 0.1, 0.2, 0.3 y 0.4 para los nucleótidos A, C, G y T, respectivamente (derecha).	21
3.4	Las probabilidades asignadas a los nucleótidos en las secuencias generadas para obtener estas representaciones fueron 'A' ~ 0.1, 'C' ~ 0.1, 'G' ~ 0.1 y 'T' ~ 0.7 (izquierda) y 'A' ~ 0.1, 'C' ~ 0.1, 'G' ~ 0.7 y 'T' ~ 0.1 (derecha).	21
3.5	De manera similar a la Figura 3.4, aquí tenemos que el nucleótido con el peso de probabilidad más alto causa una mayor acumulación de puntos en la representación hacia el vértice con el que se identifica.	22
3.6	En ambas representaciones visualizamos secuencias generadas que sólo contienen las bases 'A', 'G' y 'T', sin embargo, las distribuciones de probabilidades al variar presentan diferencias observables en cuanto a la 'iluminación' del patrón.	22

3.7 Las secuencias con presencia de tanto dos nucleótidos (izquierda), como de uno (derecha), se generaron con una distribución uniforme para una mejor visualización de los patrones y en la de uno se ilustra un acercamiento para posibilitar la observación de la sucesión de puntos convergente al vértice correspondiente, en este caso el vértice 'T'	23
3.8 En esta representación se vuelve muy clara la visibilidad de la alta presencia de los monómeros 'A' y 'T', ambos con probabilidad 0.3 en comparación de 'C' y 'G' con probabilidad 0.15 cada uno. Tal representación cuenta con cierta semejanza a la CGR de una muestra de <i>Acinetobacter</i>	24
3.9 Al realizar un intercambio en los pesos de probabilidad entre los pares de bases mencionados en la figura anterior, obtenemos un patrón muy similar al de la Figura 3.8 (izquierda) con una dirección diferente en cuanto a la diagonal que se ilumina, y cierto parecido con la representación de la muestra de <i>Pseudomonas</i>	24
3.10 k-meros más frecuentes en muestra de <i>Pseudomonas</i>	26
3.11 Comparación de contenido de GC entre <i>Actinobacteria</i> y <i>Enterobacter</i>	27
3.12 En este boxplot podemos observar el comportamiento y distribución de las dimensiones de los fractales para muestras resistentes mostrando un comportamiento muy similar entre organismos susceptibles y organismos resistentes.	28
3.13 En este heatmap observamos que, en efecto, de existir una correlación entre la dimensión fractal y el fenotipo es muy baja y se conjectura que se debe a una correlación "menos directa"	28
3.14 Matriz de confusión de la estimación de la CNN cuya arquitectura está basada en la descrita por Zhao et al., 2023	29
3.15 Diseño de la página 1 de la App destacando las componentes de la misma	30
3.16 En esta figura se presenta la estructura de la página 2 de la App destacando sus componentes.	31
3.17 Restricciones en cuadrados disponibles en la App se pueden vizualizar en esta figura.	32

3.18 La página 3 de la App muestra la representación de un fragmento de la secuencia de Homo Sapiens Chromosome 1 (CM000663.2)	33
3.19 Aquí se observa la estructura de la página 4 de la App con los botones de las interacciones correspondientes.	34

Capítulo 1

Introducción

El juego del caos es un algoritmo que permite la generación de fractales a partir de reglas iterativas simples. Este capítulo introduce los conceptos fundamentales que sustentan este método, comenzando con la noción de fractal. Posteriormente, se describe el juego del caos en su versión clásica, así como sus variaciones, incluyendo el juego del caos restringido, que introduce restricciones en la selección de puntos para modificar los patrones generados.

Más allá de su aplicación en la generación de fractales geométricos, este algoritmo ha encontrado usos en áreas como la bioinformática, en particular en la representación de secuencias de ADN. Se explorará el concepto de genoma y cómo la información genética puede transformarse en patrones gráficos mediante la representación del juego del caos para secuencias de ADN. Este enfoque permite analizar visualmente características de las secuencias biológicas, proporcionando herramientas para la clasificación y el reconocimiento de estructuras genómicas.

A lo largo de este capítulo, se establecerán las bases teóricas necesarias para comprender tanto el funcionamiento del algoritmo como su adaptación a la biología, sentando las bases para los desarrollos presentados en los capítulos siguientes.

1.1. Fractal

Antes de comenzar a hablar de los fractales, que en ciertos círculos podrían ser bastante populares por el atractivo visual que llegan a poseer, es importante resaltar que existen muchas disciplinas en Matemáticas que abordan este concepto de maneras muy distintas, lamentablemente,

pese a los deseos ambiciosos de la autora de mencionar más de uno, abordaremos únicamente el que menciona BARNSLEY, 1993 en su libro: *Fractals Everywhere* 'En Matemáticas, un fractal es una forma geométrica que contiene una estructura detallada en escalas arbitrariamente pequeñas, y que por lo general tiene una dimensión fractal que excede estrictamente la dimensión topológica. Muchos fractales presentan autosemejanzas en distintas escalas.'

1.2. El Juego del Caos

BARNESLEY, 1993 propuso un algoritmo determinista que posteriormente Jeffrey, 1990 rescataría e interpretaría para su artículo en el que propone una representación novedosa, al menos para el año de 1990, que proporciona un enfoque "geométrico" de visualización de secuencias de ADN, permitiendo investigar patrones, en dichas secuencias, previamente desconocidos. Tal representación se basa en el método extraído de dinámica caótica conocido como Chaos Game Representation (CGR)", o simplemente Juego del Caos; este genera una imagen de la secuencia genética que muestra patrones tanto locales como globales. Las representaciones tienen una estructura compleja tipo fractal que varía según la secuencia.

El ejemplo más sencillo del Juego del Caos descrito en dicho artículo consiste en seguir los siguientes pasos:

- (I) Ubicar tres puntos (preferentemente no co-lineales) en un espacio euclíadiano, a estos les llamaremos **vértices**.
- (II) Etiquetar uno de estos vértices con los números '1' y '2', a otro con los números '3' y '4' y al último vértice con los números '5' y '6'
- (III) Seleccionar un punto aleatoriamente en el espacio (preferentemente el delimitado por los tres vértices elegidos inicialmente) y etiquetarlo, lo ubicaremos como el punto inicial.
- (IV) Se lanza un dado nivelado de seis lados (figurativamente hablando, o mejor dicho, escribiendo) y ya que los vértices fueron etiquetados en el paso dos, el resultado de lanzar el número del dado que salga nos señalará a uno de los vértices. En el plano se coloca un punto a la misma distancia del vértice indicado por el dado y el punto inicial (o punto anterior para el primer lanzamiento del dado) sobre la geodésica que los une.

- (v) Se repite el lanzamiento del dado y se coloca el punto nuevo entre el punto anterior y el vértice señalado por el dado indefinidamente.

Si bien, la descripción algorítmica anterior es bastante clara e intuitiva, en esta tesis se usará la siguiente definición para el Juego del Caos, ya que ofrece una visión Matemática más rigurosa y más general.

1.2.1 Definición. El **Juego del Caos** es un algoritmo iterativo que genera una sucesión de puntos en un espacio euclíadiano, el proceso se describe de la siguiente manera:

Dados un polígono relleno P , $f \in (0, 1)$ y $\{a_n\}_{n \in \mathbb{N}}$ una sucesión aleatoria de elementos en el conjunto de vértices de P . Sea p_0 un punto seleccionado aleatoriamente en el interior de P , cada punto obtenido con el Juego del Caos se calcula con la fórmula

$$p_{n+1} = (p_n + a_n) * f \quad (1.2.2)$$

f recibirá el nombre de **factor**

En la figura 1.1 (I) se selecciona p_0 de manera aleatoria en el interior del triángulo, tal como se describe en la definición 1.2.1, en el diagrama (II) el vértice que indica la sucesión aleatoria para la iteración 1 es $a_1 = V_3$ y se marca p_1 sobre la recta que une a p_0 y V_3 , posteriormente para la iteración 2 (III) el vértice indicado por la sucesión es V_1 , se coloca el respectivo p_2 y se continua el proceso iterativo indefinidamente. Cabe destacar que, para programar el algoritmo en la computadora, sí se requiere establecer una cantidad conveniente de puntos

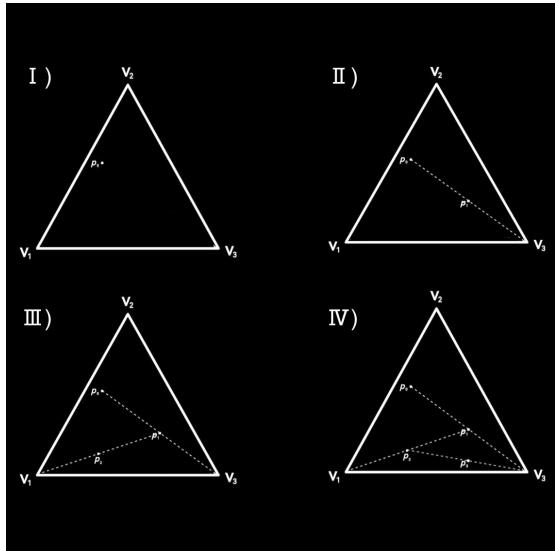


Figura 1.1: Interpretación geométrica del algoritmo del juego del caos 1.2.1.

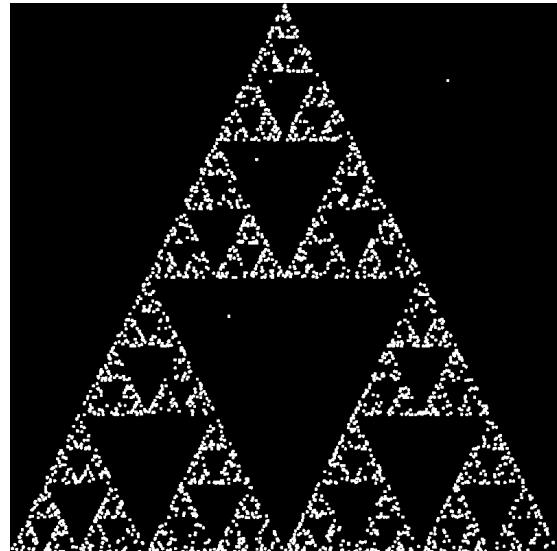


Figura 1.2: Una representación 'decente' del *triángulo de Sierpinski* se obtiene al iterar el algoritmo, un total de 2000 veces

Resulta que al aplicar este proceso en un polígono de tres lados (triángulo) como se ilustra en la Figura 1.1 con $f = \frac{1}{2}$, se obtiene un patrón, que aunque podría no atender a la intuición, ya que la sucesión de puntos es uniformemente aleatoria y, en principio, no posee ningún sesgo; dicho patrón es muy particular y también muy conocido por el nombre de *triángulo de Sierpinski* (Figura 1.2), cuyo nombre se debe al matemático que lo describió por primera vez en 1915, Wacław Sierpiński (1882-1969).

Como se menciona en la definición del juego del caos, este algoritmo se puede aplicar a cualquier polígono. Sin embargo, en esta tesis nos centraremos en casos de polígonos regulares, a continuación, en la Figura 1.3 vemos el algoritmo aplicado a un cuadrado con $f = \frac{1}{2}$ y en la Figura 1.4 lo vemos para un pentágono y con $f = 0.618$.

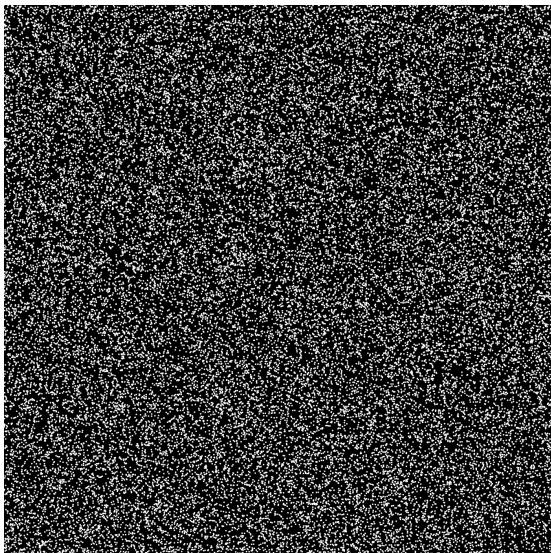


Figura 1.3: En esta figura se ilustra lo que fue descrito anteriormente, es decir, el Juego del Caos en un cuadrado con $f = \frac{1}{2}$, visualmente se aprecia lo que parece ser sólo 'ruido' y esto se debe a que los puntos de distribuyen de manera aleatoria y uniforme sin ninguna tendencia o restricción en particular. Como veremos más adelante, se pueden aplicar ciertas restricciones que generarán patrones fractales para ciertos casos.

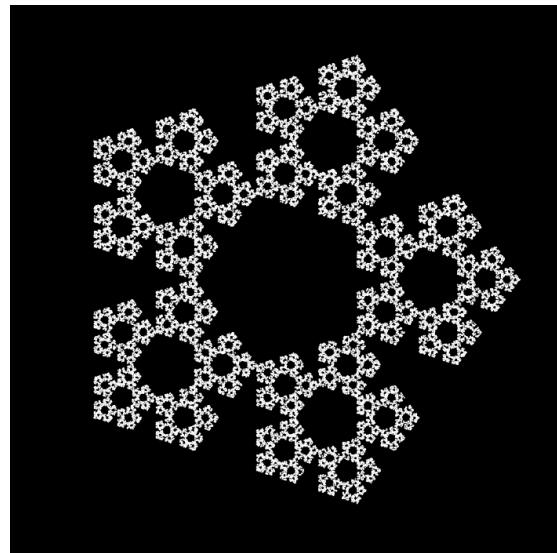


Figura 1.4: El Juego del Caos al ser aplicado en un polígono de regular de cinco lados y un valor de $f = 0.618$ obtenemos este fractal, por lo que es fácil deducir que el valor de f juega un papel muy importante en la distribución de los puntos en el polígono.

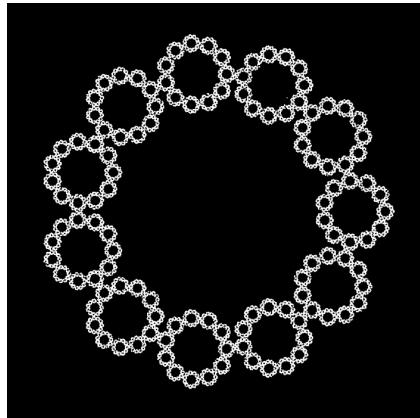


Figura 1.5: Este fractal se obtiene aplicando el algoritmo a un polígono regular de once lados, un factor de $r = 0.777$ y sin restricciones.

1.3. Juego del Caos Restringido

En la sección anterior pudimos contemplar el comportamiento del Juego del Caos en triángulos y cuadrados sin establecer condiciones (o restricciones), a la sucesión $\{a_n\}_{n \in \mathbb{N}}$ de vértices del polígono en cuestión, más que ser aleatoria con una distribución uniforme. Ahora observaremos el comportamiento de dicho algoritmo en cuadrados para sucesiones con comportamientos un poco más específicos con y sin modificar el valor de $f = \frac{1}{2}$.

Un ejemplo de dichas restricciones es la que genera el patrón de la Figura 1.6 que, en términos matemáticos, se describe de la siguiente manera:

$$\textbf{Restricción 1.} \text{ Para cualquier } i \in \mathbb{N}, a_i \neq a_{i+1} \text{ en } \{a_n\}_{n \in \mathbb{N}}. \quad (1.3.1)$$

Lo que, en términos simples, se puede describir como que no se repite el mismo vértice en dos iteraciones consecutivas, esta restricción y las otras descritas en esta sección podrán ser exploradas para otros polígonos regulares de más o de menos lados y con distintos valores para el factor en la App desarrollada como parte de este proyecto de tesis para el Museo Virtual de Matemáticas.

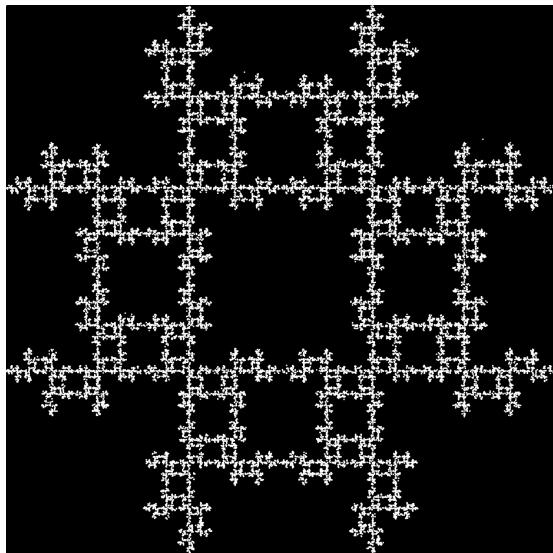


Figura 1.6: Esta representación tiene un comportamiento fractal, se obtiene con la restricción antes mencionada con los parámetros: número de lados = 4 y factor $f = 0.5$.

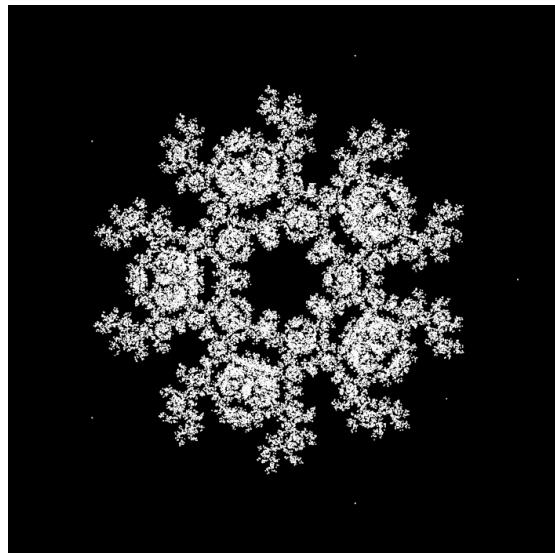


Figura 1.7: Este fractal se obtiene aplicando el algoritmo a un polígono regular de cinco lados, un factor de $f = 0.5$ y la restricción de no repetir el mismo vértice dos iteraciones consecutivas.

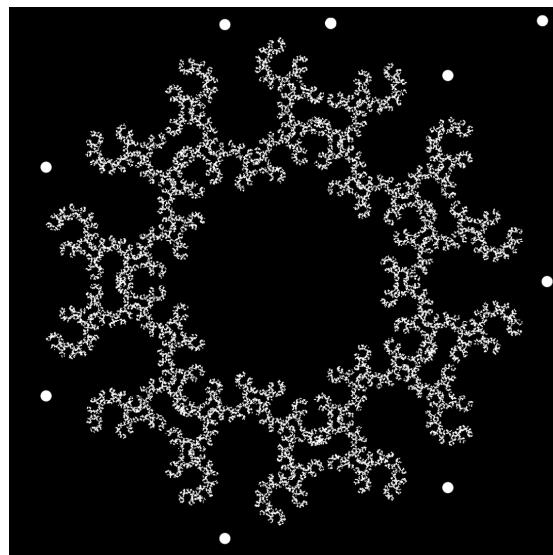


Figura 1.8: Este fractal se obtiene aplicando el algoritmo a un polígono regular de siete lados, un factor de $f = \frac{2}{3} \sim 0.6667$ y la restricción de no repetir el mismo vértice dos iteraciones consecutivas.

Observemos otro tipo de restricción de las tantas que se pueden considerar, aquí '*el límite es la imaginación*' (y quizás también el tiempo, a veces).

Restricción 2. En la sucesión $\{a_n\}_{n \in \mathbb{N}}$ si $a_i = V_s$ para algún vértice V_s , entonces $a_{i+1} \neq V_{s+1}$

(1.3.2)

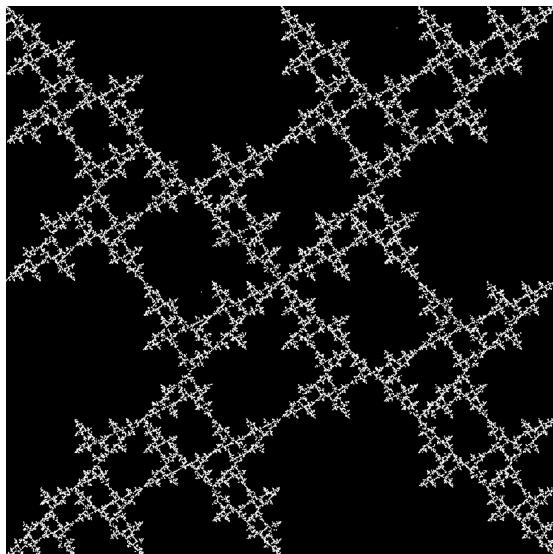


Figura 1.9: Este fractal se obtiene al aplicar el juego del caos cuadrado con $f = \frac{1}{2}$ y la restricción 1.3.2 que, parafraseando, consiste en evitar que el vértice que se seleccionará se encuentre a una rotación (en sentido horario) del vértice seleccionado en la iteración anterior

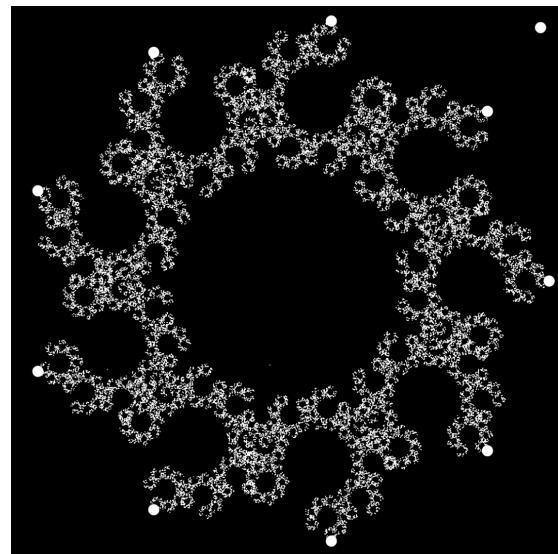


Figura 1.10: Este patrón se obtiene al aplicar el juego del caos con $f = 0.7$ y la restricción 1.3.2..

Describamos una restricción más:

Restricción 3. Para la sucesión $\{a_n\}_{n \in \mathbb{N}}$ si $a_i = V_s$ para algún vértice V_s , entonces $a_{i+1} \neq V_{s+2}$

(1.3.3)

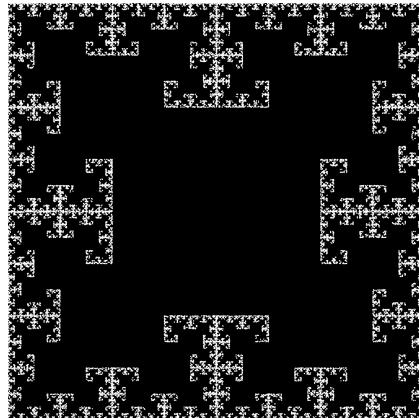


Figura 1.11: Esta figura muestra unos elegantes patrones de autosemejanza cuando consideramos la restricción 1.3.3, que básicamente consiste en evitar que el vértice actual se encuentre a dos rotaciones (en sentido horario) del vértice elegido en la iteración anterior que para el caso del cuadrado que sea el vértice de la diagonal opuesta.

Para la siguiente restricción no nos limitamos a modificar la sucesión $\{a_n\}_{n \in \mathbb{N}}$, si no también el valor de f .

Restricción 4. La sucesión sigue siendo aleatoria, pero ahora también toma el valor del punto central del polígono y $f = \frac{2}{3}$

(1.3.4)

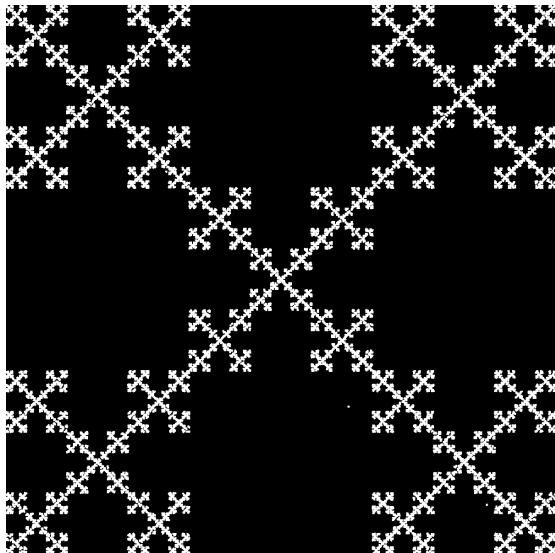


Figura 1.12: El resultado de la restricción 1.3.4 es una representación del fractal de Vicsek

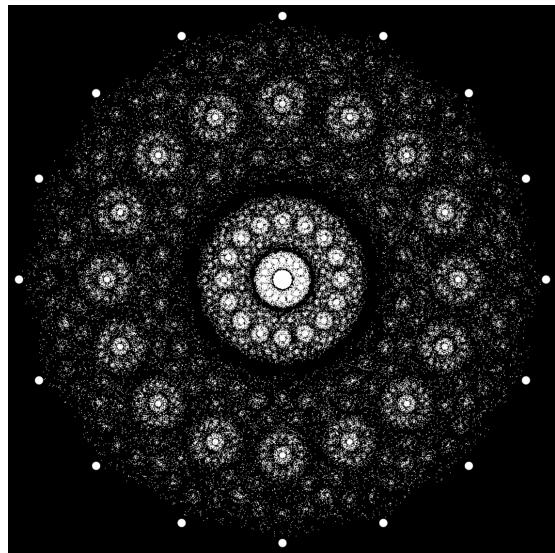


Figura 1.13: En esta figura vemos el comportamiento del algoritmo con la restricción 1.3.4 en un polígono de 16 lados y un $factor = 0.666$

En las secciones siguientes exploraremos la propuesta de Jeffrey, 1990 de sustituir la sucesión $\{a_n\}_{n \in \mathbb{N}}$ por una secuencia de ADN, entendiendo a esta última como una restricción más para el cuadrado al considerar ciertas condiciones para el polígono y algunos conceptos básicos que nos permitan entrar en contexto.

1.4. Genoma

En términos simples, el genoma se define como el depósito de información de un organismo puesto que es el conjunto completo de información genética que procura el correcto funcionamiento del ser vivo a lo largo de su vida (AVS et al., 2022).

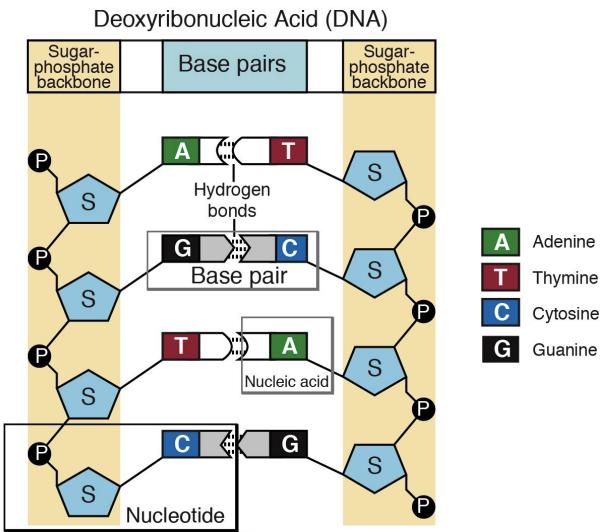


Figura 1.14: Representación molecular del ADN (imagen «Nucleotides and Bases - Genetics Generation — knowgenetics.org», s.f.)

'Las macromoléculas de la célula que llevan la información genética son el ácido desoxirribonucleico (ADN) y el ácido ribonucleico (ARN), estas son las que conforman el genoma. Los ácidos nucleicos son polímeros de unidades repetidas llamadas nucleótidos. Un nucleótido está formado por un azúcar, un fosfato y una base nitrogenada. Las bases nitrogenadas son de cuatro tipos: adenina (A), citosina (C), guanina (G) y timina (T). En el ARN la timina es reemplazada por uracilo (U). La secuencia de estas bases determina la secuencia de ADN o la información genética de un organismo.' Panchal, 2022.

Ahora, para estudiar y analizar el genoma y su estructura, es necesario extraer la información de la distribución de los nucleótidos a lo largo del ADN perteneciente al organismo (o el micro-bioma) a estudiar. Tal extracción suele realizarse a través de un proceso llamado secuenciación que consiste en una serie de procedimientos bioquímicos los cuales a su vez constan, esencialmente, de 7 fases:

1. Extracción y aislamiento de la molécula de ADN
2. Fragmentación de la molécula en trozos más pequeños
3. Adición de secuencia de etiquetas
4. Amplificación por medio de replicación

5. Reacción de secuenciación

6. Detección de indicadores de nucleótidos

7. Evaluación y codificación de la información detectada

Esta lista es una forma muy simplificada de lo que conlleva un proceso de secuenciación, ya que, actualmente, existen diversos métodos de secuenciación en los cuales algunos pasos pueden variar, por ejemplo las muestras que se usaron para este análisis son genomas de bacterias secuenciados con el método Illumina.

Posterior a su secuenciación, se obtienen archivos con la información de los pedazos de ADN secuenciados que se conocen como 'reads' tal como se muestra en la figura 1.15 que, por medio de técnicas de Bioinformática, se procesa midiendo (y posteriormente mejorando) la calidad, ensamblando los reads, refinando, anotando, analizando, validando con genomas de referencias y finalmente almacenando.

Tal proceso se llevó a cabo con las ya mencionadas muestras de bacterias para obtener archivos como el de la figura 1.16 que serían la información a la cual se le aplicaría el algoritmo del juego del caos como se describe en la sección siguiente.

```
>assembly_SRR5901132_contig_1
CCGGTAAAGGCTAACCATCGGCAAGAGCAGCATTAAATCAAAAGATTAGCTGACGCTT
TTTATCTTATAAACAGCTTAGTGCACGAGCTGCTGGCAATACCTCGGAAACGGCTGTAC
AAAAGTGTTGTGATACTACACACCCACTCAGGTGAACTGACTCTATCTGACTTGA
ATGATACAGGGCTTCAGTAACGAGCATCAGCGAAAGATACTTAACTTAACTTAAAGC
TTGAAGGACAGGAAACTGTAGCGGAGTAACTATTAGTTTCAACTGTAGAAGGGAAAAA
CTTGGCAGGAAACCAGGGTAGTCCAAAAAGATTGGCTGACGTTTATTAATATAAG
CTGTAGTGCAGACGGCTGCTGGAACTACCTCGGAAACGGCTGTACAAAAGTTGGTTG
ATACTACCAACACCAAGCAGGTGAACTGACTTATCTGACTTGAATGATACAGGGCTT
GACCCAGATGATCAGATTGCGAAGATAACAACTTCAATTAAAGCTTGAAGGACAGGAAA
CTGGTAGCCGGAACTATTTAGTTCCACCGGTGAAAGGGAAAAACGTGGCAGGAAACCA
GGTAGGCCCAAAGGATTGGCTGATGTTGTTAACAAATAAGCTGTAGTGACAGACG
CTGGCCGAACTTCTCGGAAACGGCTGTACAAAAGGTGTTGGTGAATACCGCAGGCG
AAGCAGGTAAACTACTTATCTGATTGGAATGATACAGGTGTTTCAGGAAACGATCAGA
TTACGCGAAGATACTTACCTTAAAGCTGTAGTGGCGGATTGTGTTGGGAAACAGA
CGCTTACTAGACCATCTAGAAGTCTAAAAGTGAAGGAAAGGAACTTGGCAAGAGCAA
```

Figura 1.15: Ejemplo de archivo 'crudo' de una muestra de genoma secuenciado. En la primer linea encontramos informacin acerca del read, en la segunda el read, la tercera el smbolo '+' que funge como conector de la segunda y cuarta linea, en esta ´ltima los smbolos representan las calidades del read.

Figura 1.16: Computacionalmente, un genoma secuenciado se puede manejar como una cadena de caracteres y suele encontrarse con formato .fasta. En esta figura podemos visualizar las primeras 16 líneas de la secuencia de ADN ya procesada de una muestra de genoma de bacteria.

1.5. Representación del Juego del Caos para Secuencias de ADN (CGR por sus siglas en inglés)

Para aplicar el algoritmo descrito en 1.2.1 a secuencias de ADN, hacemos una pequeña modificación, identificando cada vértice del cuadrado con un único nucleótido para después sustituir la sucesión $\{a_n\}_{n \in \mathbb{N}}$ por una secuencia de ADN tal como se muestra en la figura 1.17. El algoritmo para secuencias de ADN se define de la siguiente manera.

1.5.1 Definición. El Juego del Caos para secuencias de ADN maneja la información del genoma como una sucesión que toma valores en $\{A, C, G, T\}$ (cada letra corresponde a la inicial de un nucleótido) y cada nucleótido está asociado a cada vértice de un cuadrado relleno como se muestra en la figura 1.17. Y con ello los pasos para correr el algoritmo con secuencias de ADN es el mismo que de la definición 1.2.1.

Para asignar los vértices del cuadrado a cada nucleótido, durante este trabajo, se utilizó la configuración RY, como fue presentada inicialmente por Jeffrey en Jeffrey, 1990.

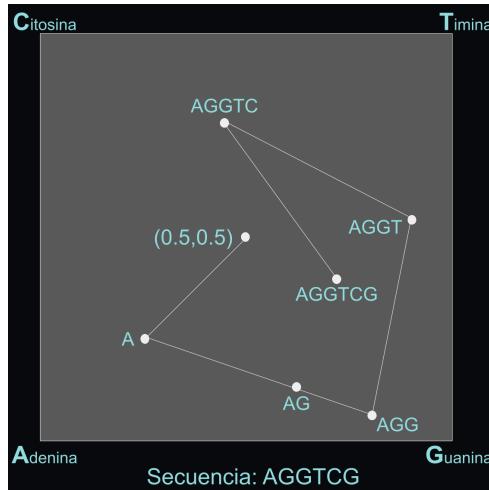


Figura 1.17: Representación conceptual del Juego del Caos aplicado a secuencias de ADN con la secuencia de juguete 'AGGTCG' que indica la posición en que se dibujará el punto de esa iteración, es decir, la posición del punto p_0 es el centro del cuadrado, la posición de p_1 es a mitad del 'camino' entre p_0 y el vértice A que es la letra en la posición 1 de la secuencia, luego p_2 se ubica a mitad de camino entre p_1 y la letra de la posición 2 en la secuencia, que es 'G', y así sucesivamente.

Una vez que contamos con los genomas limpios y ensamblados, procedemos a correr el juego del caos con ellos y obtenemos patrones como los que observamos en las figuras 1.18, 1.19 y 3.2. Esta representación nos permite observar que la estructura de ADN **no** tiene un comportamiento meramente aleatorio como el que observamos en la figura 1.3, además de presentar autosemejanzas a distintas escalas, tal como en un fractal según el concepto de BARNSLEY, 1993.

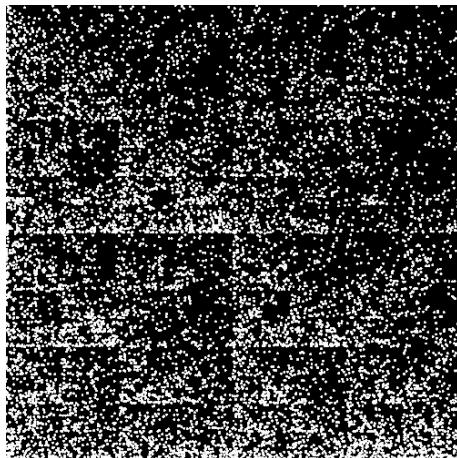


Figura 1.18: Esta figura se obtuvo al correr el algoritmo del juego del caos para los primeros 10000 pares de bases del genoma completo de mitocondria NS06 aislada de homo sapiens (GenBank: GU170820.1) en una de las primeras versiones de la App de divulgación acerca del juego del caos.

En las siguientes figuras tenemos dos CGR's que fueron obtenidas con el algoritmo programado en Python usando paquetes especializados para graficar enormes cantidades de datos, en este caso puntos en el espacio acotado por un cuadrado, lo que mejora enormemente la calidad de las representaciones, así como su análisis. Las coordenadas de tales puntos fueron almacenadas en archivos .txt para un posterior cálculo de la dimensión fractal de correlación de cada representación (imagen).

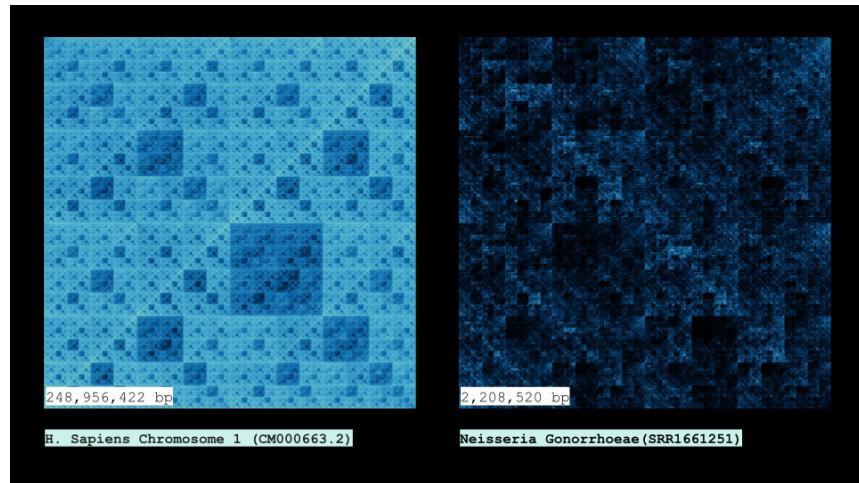


Figura 1.19: El algoritmo de generación de las imágenes se programó para que una densidad mayor de puntos en determinada zona, se tradujera en una tonalidad más clara en el color, fenómeno que podemos observar en estas representaciones con cantidades de puntos (longitudes de secuencia) muy distintas. Izquierda con casi 250 millones de pares de bases y en la derecha con más de 2 millones de pares de bases.

Capítulo 2

Objetivo General

Explorar características de la representación del juego del caos para secuencias de ADN en diferentes linajes taxonómicos, así como investigar el posible uso de dichas imágenes para clasificación de microorganismos. Por otro lado, utilizar tales representaciones visuales como herramienta de comunicación tanto para la comunidad científica como para el público general.

2.1. Objetivos Específicos

2.1.1. Objetivo 1.

Representar geométricamente propiedades de secuencias de ADN.

Adaptar del algoritmo del juego del caos para genomas de bacterias (ver 1.5.1).
Resaltar la presencia de los patrones fractales.

Objetivo 1.1. Representar visualmente las frecuencias de k-meros en secuencias de ADN.

1. Descargar muestras de bacterias clínicamente relevantes.
2. Generar el código del juego del caos
3. Realizar simulaciones de secuencias específicas con distintos sesgos y distribuciones de nucleótidos.
4. Representar visualmente las frecuencias de k-meros en secuencias de ADN.

Objetivo 1.2. Obtener propiedades de las representaciones.

1. Contenido de GC en las secuencias a partir de las representaciones. Comparar la medición por pixelaciones con la medición directa en la secuencia.
2. Dimensión fractal de correlación de las imágenes.

Objetivo 1.3. Investigar la capacidad de clasificación en distintas categorías biológicas.

Utilizar las características obtenidas para implementar un algoritmo de clasificación.

2.1.2. Objetivo 2.**Crear una herramienta de visualización para la comunicación científica del juego del caos.**

Objetivo 2.1. Programar el algoritmo del juego del caos para polígonos de n lados con el framework AngularJS.

Objetivo 2.2. Adaptar el algoritmo para generar restricciones en el juego del caos.

Objetivo 2.3. Programar la versión del juego del caos para secuencias de ADN.

Objetivo 2.4. Desarrollar la interfaz de usuario para su interacción con la App.

Objetivo 2.5. Generar un programa en Python que genere la CGR de una secuencia dada.

Capítulo 3

Resultados

En esta tesis, reportamos la diferenciación entre comunidades microbianas, posibilidad de diagnóstico de condiciones como caries y capacidad de identificación de resistencia antimicrobiana, por medio de análisis estadísticos descriptivos y algoritmos de clasificación aplicados a imágenes generadas por la CGR.

La aplicación, a diferencia de otras propuestas interactivas en línea, ofrece la selección de parámetros de una manera más concreta y clara, además de introducir la adaptación del algoritmo del juego del caos para generar las CGR de secuencias de ADN.

3.1. Representación geométrica de propiedades de secuencias de ADN.

3.1.1. Descarga de muestras de bacterias clínicamente relevantes.

Como parte de un proyecto de investigación derivado del reto de resistencia antimicrobiana de la conferencia internacional anual Critical Assessment of Massive Data Analysis edición 2024 (CAMDA 2024) en la que la autora de esta tesis ha tomado parte, se obtuvieron las muestras designadas a dicho reto para también desarrollar este análisis, tales muestras pertenecen a 8 géneros de bacterias ampliamente estudiadas. La información obtenida de tales muestras y de sus respectivas representaciones se puede visualizar en la Figura 3.1.

	id	correlation_dimension	antibiotic	genus	phenotype	mic
0	Acinetobacter_baumannii_SRR3223466	1.043075	meropenem	Acinetobacter	Susceptible	2.00
1	Acinetobacter_baumannii_SRR5862467	0.766278	meropenem	Acinetobacter	Resistant	16.00
2	Acinetobacter_baumannii_SRR3223164	1.553864	meropenem	Acinetobacter	Susceptible	0.25
3	Acinetobacter_baumannii_SRR4417603	1.219716	meropenem	Acinetobacter	Resistant	8.00
4	Acinetobacter_baumannii_SRR3217322	1.151460	meropenem	Acinetobacter	Susceptible	1.00

Figura 3.1: Se creó una base de datos que recopila la información relevante para este estudio de las 5,554 muestras y sus representaciones.

3.1.2. Generación del código del juego del caos

A partir del algoritmo del juego del caos para genomas descrito por Jeffrey, 1990, en nuestro caso genomas de bacterias, se obtuvieron las CGR para las 5,554 muestras que nos permitieron identificar similitudes entre CGR's de un mismo género, a su vez, identificar diferenciación entre algunas representaciones de distintos géneros de bacterias como se puede notar en la Figura 3.2. Por otro lado, las representaciones poseen patrones con un evidente comportamiento fractal, en el sentido de auto-semejanza a distintas escalas (ver 1.1).

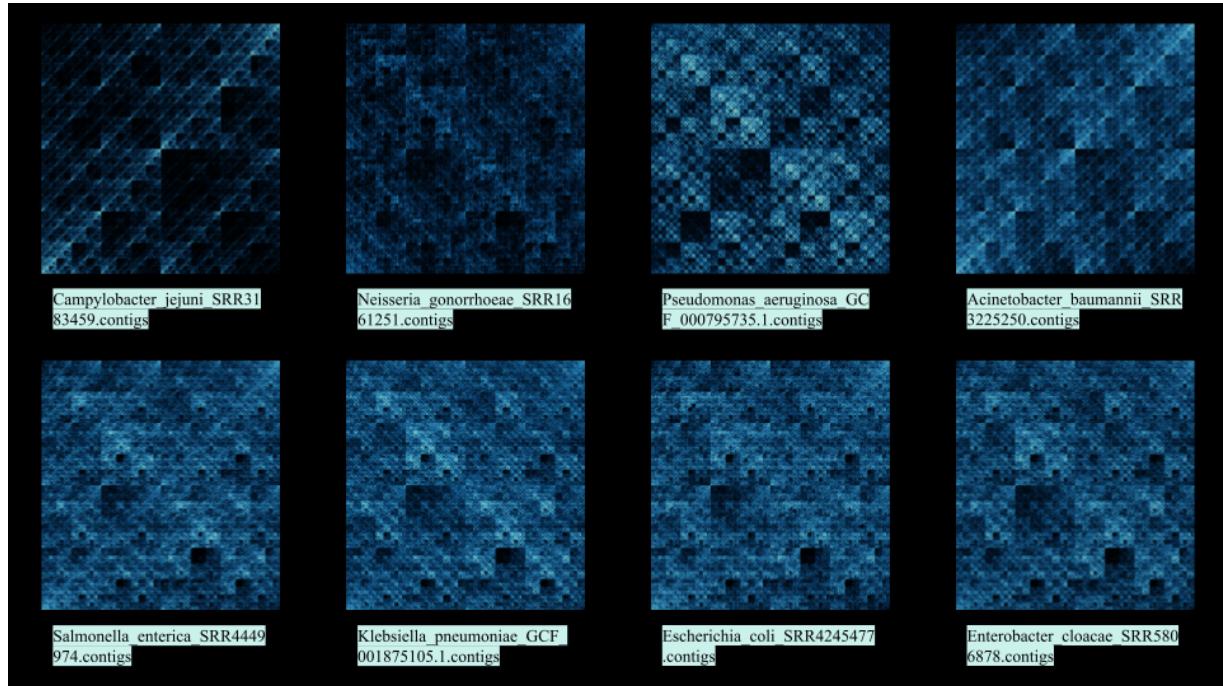


Figura 3.2: Aquí podemos observar la representación del juego del caos para ocho géneros de bacterias y notamos variaciones sutiles y no tan sutiles entre los patrones obtenidos para los distintos géneros

3.1.3. Generación de simulaciones de secuencias específicas con distintos sesgos y distribuciones de nucleótidos.

Como parte del proceso de validación del algoritmo programado en Python y las imágenes obtenidas, se generaron representaciones para secuencias artificiales de 3'000,000 de nucleótidos, para tener un punto de comparación contra las representaciones de las muestras reales, entre las secuencias simuladas que se generaron se cuenta con una generada aleatoriamente la cual presenta un comportamiento que se corresponde al juego del caos sin restricciones en el cuadrado (ver Figura 1.3) como se puede ver en la Figura 3.3 (izquierda), mientras que para la segunda (derecha) se generó una secuencia con distintas probabilidades asignadas a cada nucleótido.

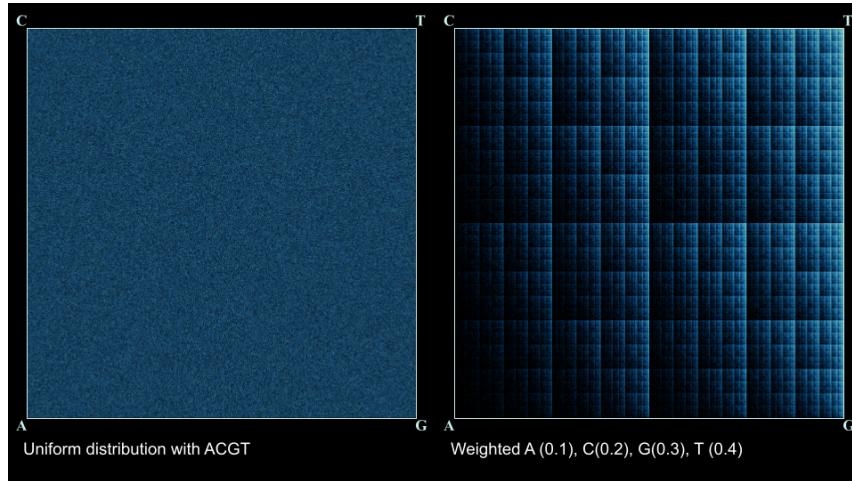


Figura 3.3: Las secuencias fabricadas que generan estas representaciones constan de una aleatoriedad con una distribución uniforme (izquierda) y una con pesos de probabilidad asignados que son 0.1, 0.2, 0.3 y 0.4 para los nucleótidos A, C, G y T, respectivamente (derecha).

En la Figura 3.4 las CGR presentadas permiten observar el comportamiento del algoritmo para secuencias en las que se maximizaron las presencias de los nucleótidos 'T' y 'G' respectivamente y de la misma manera, en la Figura 3.5 se tienen las CGR's de muestras con las presencias de los nucleótidos 'A' y 'C' maximizadas en proporción a los otros.

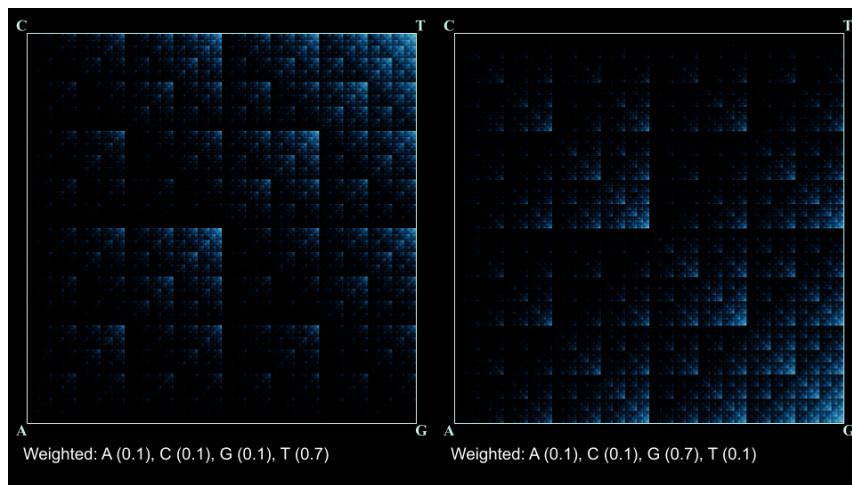


Figura 3.4: Las probabilidades asignadas a los nucleótidos en las secuencias generadas para obtener estas representaciones fueron ' $A \sim 0.1$ ', ' $C \sim 0.1$ ', ' $G \sim 0.1$ ' y ' $T \sim 0.7$ ' (izquierda) y ' $A \sim 0.1$ ', ' $C \sim 0.1$ ', ' $G \sim 0.7$ ' y ' $T \sim 0.1$ ' (derecha).

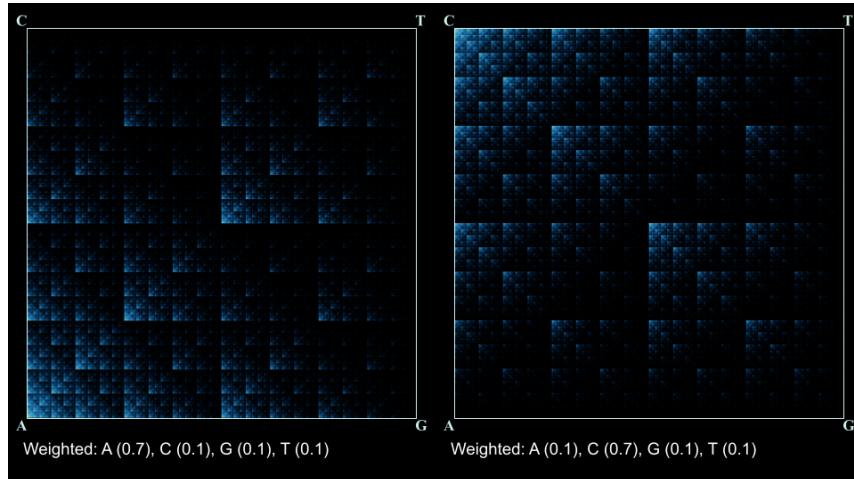


Figura 3.5: De manera similar a la Figura 3.4, aquí tenemos que el nucleótido con el peso de probabilidad más alto causa una mayor acumulación de puntos en la representación hacia el vértice con el que se identifica.

Al generar secuencias que se limitan a tres de los cuatro nucleótidos, obtenemos patrones similares al triángulo de sierpinski (salvo alguna 'deformación') como en las CGR's de la Figura 3.6.

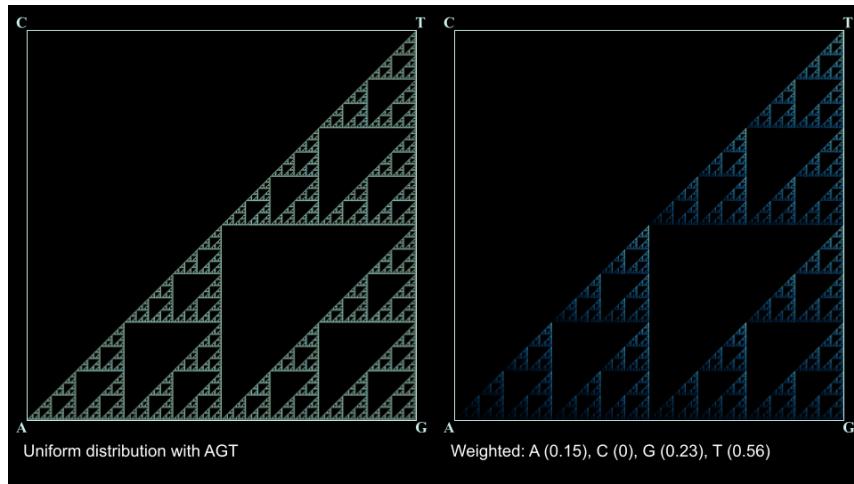


Figura 3.6: En ambas representaciones visualizamos secuencias generadas que sólo contienen las bases 'A', 'G' y 'T', sin embargo, las distribuciones de probabilidades al variar presentan diferencias observables en cuanto a la 'iluminación' del patrón.

En el caso anterior, al quedarnos sólo con tres bases, tenemos que el triángulo de Sierpinski

vuelve a aparecer, ahora, si sólo generamos secuencias con dos o un sólo nucleótido tenemos comportamientos como los presentes en las representaciones de la Figura 3.7 en los cuales, como la intuición nos indicaría, sólo se visualiza una línea que conecta los vértices de los nucleótidos presentes en la secuencia para el caso de dos bases nitrogenadas y una sucesión de puntos que convergen al vértice del nucleótido que conforma la secuencia.

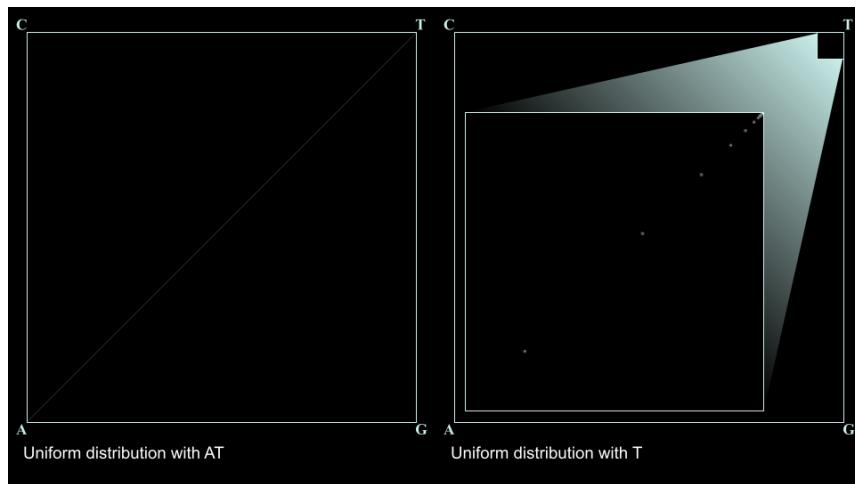


Figura 3.7: Las secuencias con presencia de tanto dos nucleótidos (izquierda), como de uno (derecha), se generaron con una distribución uniforme para una mejor visualización de los patrones y en la de uno se ilustra un acercamiento para posibilitar la observación de la sucesión de puntos convergente al vértice correspondiente, en este caso el vértice 'T'.

Como en secciones anteriores se menciona, la posibilidad de interpretar una mayor presencia de cierto dímero conocido en una secuencia a partir de su CGR, se confirma al realizar las simulaciones en las siguientes Figuras (3.8 y 3.9) cuya intención es aproximar los patrones formados por secuencias reales de muestras de bacterias al modificar los pesos de probabilidades para cada nucleótido.

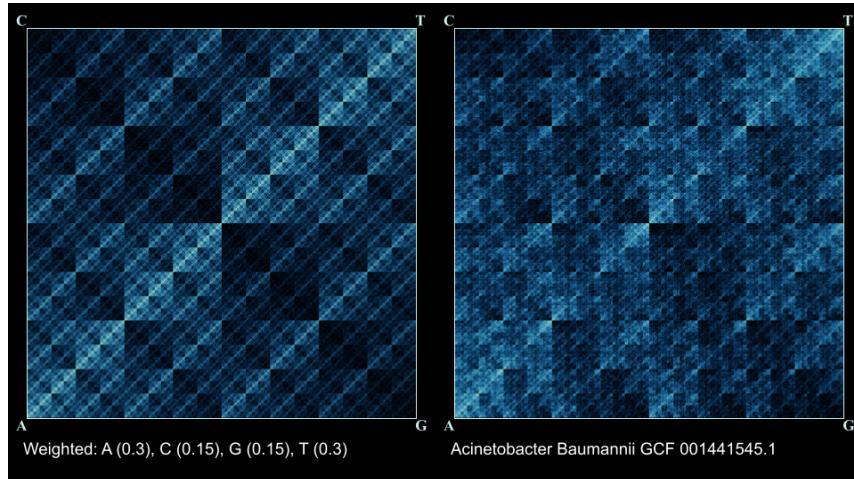


Figura 3.8: En esta representación se vuelve muy clara la visibilidad de la alta presencia de los monómeros 'A' y 'T', ambos con probabilidad 0.3 en comparación de 'C' y 'G' con probabilidad 0.15 cada uno. Tal representación cuenta con cierta semejanza a la CGR de una muestra de *Acinetobacter*.

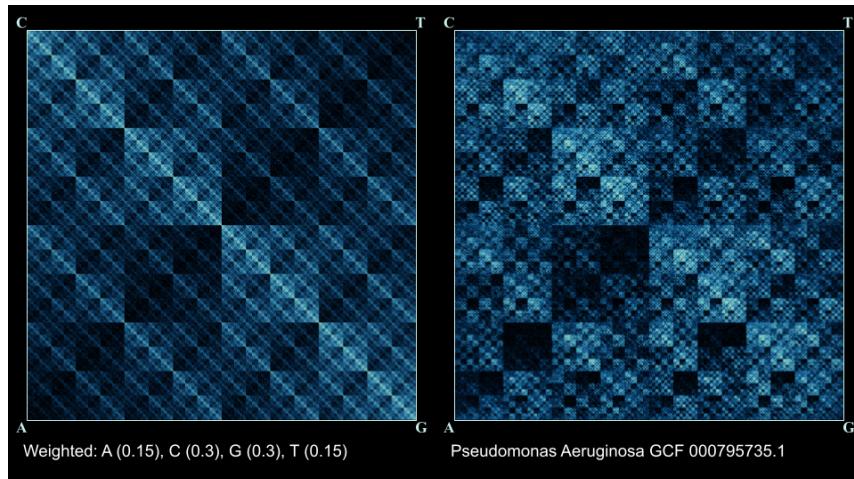


Figura 3.9: Al realizar un intercambio en los pesos de probabilidad entre los pares de bases mencionados en la figura anterior, obtenemos un patrón muy similar al de la Figura 3.8 (izquierda) con una dirección diferente en cuanto a la diagonal que se ilumina, y cierto parecido con la representación de la muestra de *Pseudomonas*.

3.1.4. Representación visual de las frecuencias de k-meros en secuencias de ADN.

Las CGR's nos permiten visualizar frecuencias de k-meros, pues al realizar simulaciones de secuencias específicas con distintos sesgos y distribuciones de nucleótidos, pudimos observar los diversos comportamientos de los patrones en las representaciones, desde las Figuras 3.3 sin un patrón determinado, 3.5 que evidencian mayor presencia de algún monómero, hasta 3.7 que destacan cómo la presencia o ausencia de ciertos nucleótidos define casi completamente el patrón presente en la CGR correspondiente. Por lo anterior, es natural preguntarnos qué patrones se forman si garantizamos mayor presencia de k -meros específicos cuya abundancia ya es conocida en genomas de bacterias, y de manera inversa, qué patrones en las representaciones nos arrojan información a simple vista acerca de la presencia de k -meros específicos en la secuencia.

Previo a generar las simulaciones de presencias maximizadas de k -meros en secuencias, cuyas abundancias están basadas en genomas reales, se realizaron cálculos de frecuencias de k -meros para $k \in \{1, 2, 3\}$ en muestras seleccionadas de manera aleatoria de cada género.

k-mer	count	weight
C	2'119,741	0.3
G	2'112,571	0.3
A	1'115,774	0.15
T	1'111,842	0.15
GC	813,447	0.13
CG	757,351	0.11
CC	591,933	0.1
GG	587,970	0.1
CGC	270,490	0.045
GCG	269,188	0.045
GCC	262,669	0.043
GGC	261,488	0.043

Tabla 3.1: Frecuencias de k-mers y sus conteos.

En base a la información obtenida en la Tabla 3.1 se asignaron los pesos con lo que se determinó la frecuencia de los k-meros especificados en la tabla para generar secuencias sintéticas y posteriormente sus representaciones, las cuales de observan en la Figura 3.10.

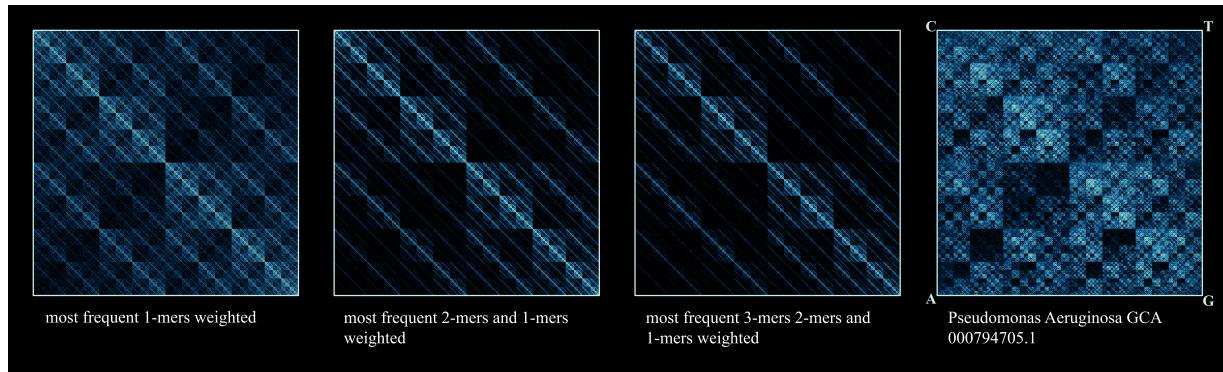


Figura 3.10: k-meros más frecuentes en muestra de Pseudomonas

3.2. Obtención de propiedades de las representaciones e investigación de su capacidad de clasificación en distintas categorías biológicas.

A partir de la información resultante de calcular y analizar la dimensión fractal obtenemos que el género de *Campylobacter* se diferencia de los demás puesto que presenta una dimensión fractal notablemente mayor a la de otros géneros.

3.2.1. Contenido de GC en las secuencias a partir de las CGR's.

En la Figura 3.11 se puede observar que, a simple vista, la representación ya permite visualizar una característica diferencial entre dos géneros de bacterias, como lo son *Actinobacteria* y *Enterobacter*, que es el contenido de GC en una y otra por medio de una línea muy marcada en la diagonal que conecta los dos vértices identificados con los nucleótidos G y C presente en la CGR de la muestra de *Actinobacteria* a diferencia de la representación de la muestra de *Enterobacter*. Cabe mencionar que, si bien, *Actinobacteria* no es un género de relevancia para este estudio, la autora consideró una muestra únicamente para ilustrar la posibilidad de comparar el contenido

del dímero GC a partir de la CGR.

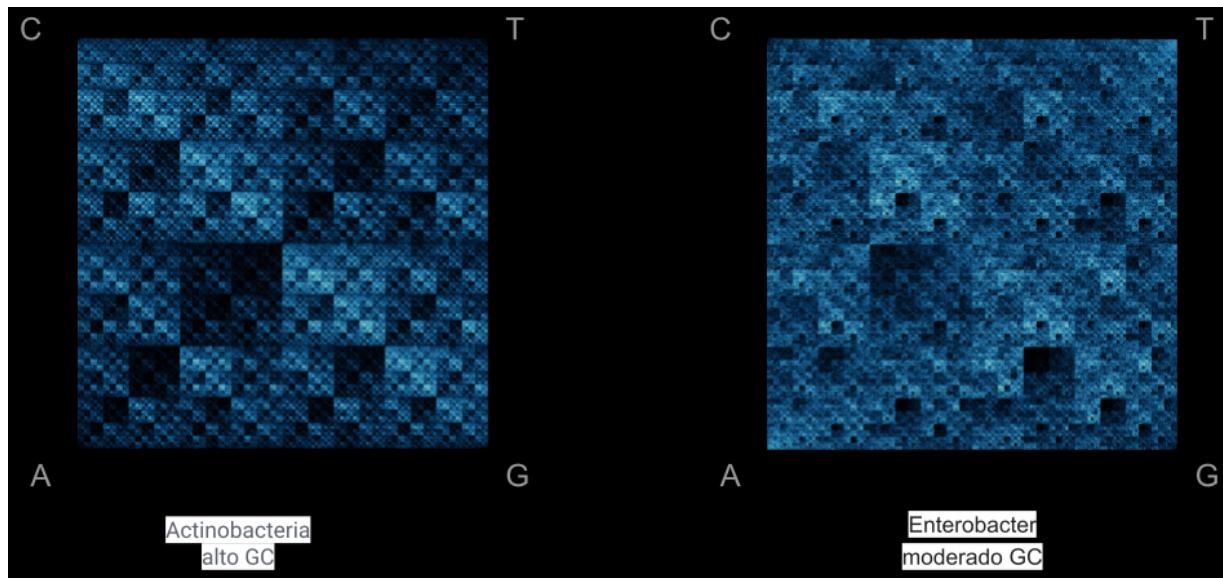


Figura 3.11: Comparación de contenido de GC entre Actinobacteria y Enterobacter

3.2.2. Dimensión fractal de correlación de las imágenes.

Al realizar los análisis estadísticos pertinentes a las dimensiones fractales obtenidas para las representaciones de cada muestra se obtuvieron los siguientes gráficos. En el primero (Figura 3.12) notamos que no hay una diferencia lo suficientemente marcada entre las dimensiones fractales de las muestras resistentes y las susceptibles de un mismo género, sin embargo, podemos observar que las muestras pertenecientes al género *Campylobacter* se diferencian por tener dimensiones fractales generalmente mayores a las de los otros géneros.

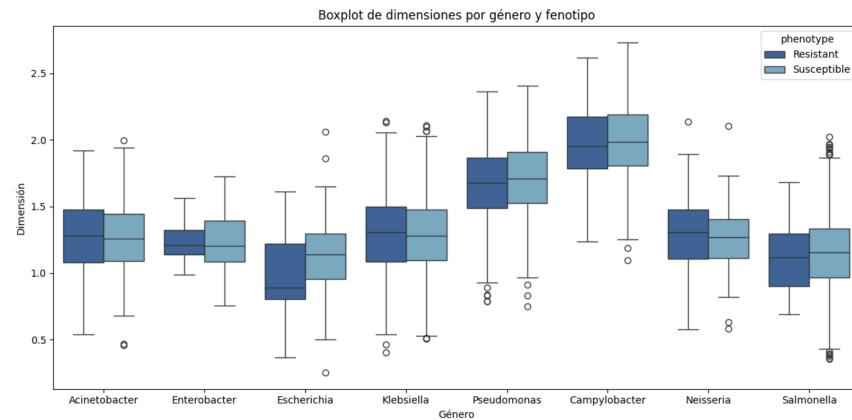


Figura 3.12: En este boxplot podemos observar el comportamiento y distribución de las dimensiones de los fractales para muestras resistentes mostrando un comportamiento muy similar entre organismos susceptibles y organismos resistentes.

Por los resultados obtenidos en el gráfico anterior procedemos a generar un "heatmap" de correlación (Figura 3.13) entre el fenotipo (resistente o susceptible), el valor de las dimensiones fractales de todas las muestras y la concentración inhibitoria mínima (mic) de antibiótico, para terminar de confirmar que la correlación entre las variables fenotipo y dimensión de correlación es relativamente baja.

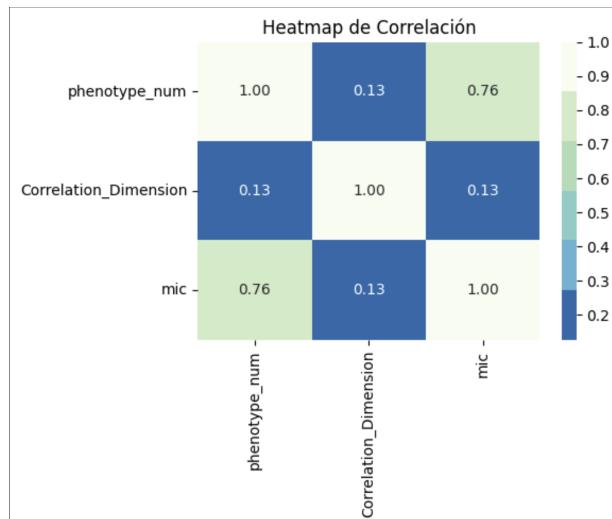


Figura 3.13: En este heatmap observamos que, en efecto, de existir una correlación entre la dimensión fractal y el fenotipo es muy baja y se conjectura que se debe a una correlación "menos directa"

3.2.3. Uso de las características obtenidas para implementar un algoritmo de clasificación.

Para implementar algoritmos de clasificación de deep learning se implementó la arquitectura de la red neuronal convolucional descrita en Jeffrey, 1990 y se seleccionaron dos géneros de bacterias cuyas representaciones fueran "fáciles de diferenciar" para el ojo humano: *Campylobacter* y *Pseudomonas*. Se usaron 150 imágenes (una imagen por muestra) divididas en 70%, 15%, 15% para entrenamiento, prueba y validación, respectivamente, obteniendo un valor de accuracy de 0.56 como se puede visualizar en la Figura 3.14 que muestra una matriz de confusión de algún conjunto identificado de muestras, ya sea de géneros de bacterias o su resistencia a meropenem y ciprofloxacin.

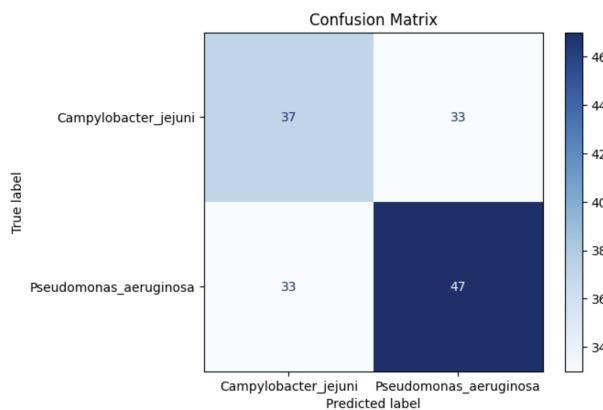


Figura 3.14: Matriz de confusión de la estimación de la CNN cuya arquitectura está basada en la descrita por Zhao et al., 2023

3.3. Creación de una herramienta de visualización para la comunicación científica del juego del caos.

Disponibilidad de la App en la página <https://132.248.41.65/biomat/caos/ch-g-app/browser/page1>

Disponibilidad del código (enlace del repo)

Desarrollo de una aplicación de fácil uso para el público que, desde el triángulo de Sierpinski hasta las representaciones basadas en secuencias de ADN, la interfaz incluye controles intuitivos

y explicaciones detalladas que fomentan la exploración interactiva y el aprendizaje.

3.3.1. Programación del algoritmo del juego del caos para polígonos de n lados con el framework AngularJS.

La aplicación fue diseñada con una estructura "progresiva", es decir, comienza desarrollando los temas de manera simple y agregando complejidad. La aplicación comienza con una implementación básica del algoritmo del juego del caos en polígonos regulares, utilizando AngularJS para facilitar su modularidad y escalabilidad. En la primera página, el triángulo de Sierpinski se presenta como introducción visual al concepto, limitando los parámetros f y $sides$ para mantener la simplicidad inicial. Los botones de navegación e información permiten al usuario explorar los fundamentos del juego del caos de manera interactiva.



Figura 3.15: Diseño de la página 1 de la App destacando las componentes de la misma

3.3.2. Adaptación del algoritmo para generar restricciones en el juego del caos.

La segunda página (Figura 3.16) amplía las posibilidades del algoritmo al incorporar el cuadrado como polígono base y al introducir la modificación del parámetro f . Se introducen restricciones específicas, que pueden ser observadas en la Figura 3.17, como la que genera al fractal de Vicsek, entre otras, destacando cómo afectan visualmente el resultado. Botones dedicados explican cada restricción, destacando el impacto de los ajustes realizados.



Figura 3.16: En esta figura se presenta la estructura de la página 2 de la App destacando sus componentes.

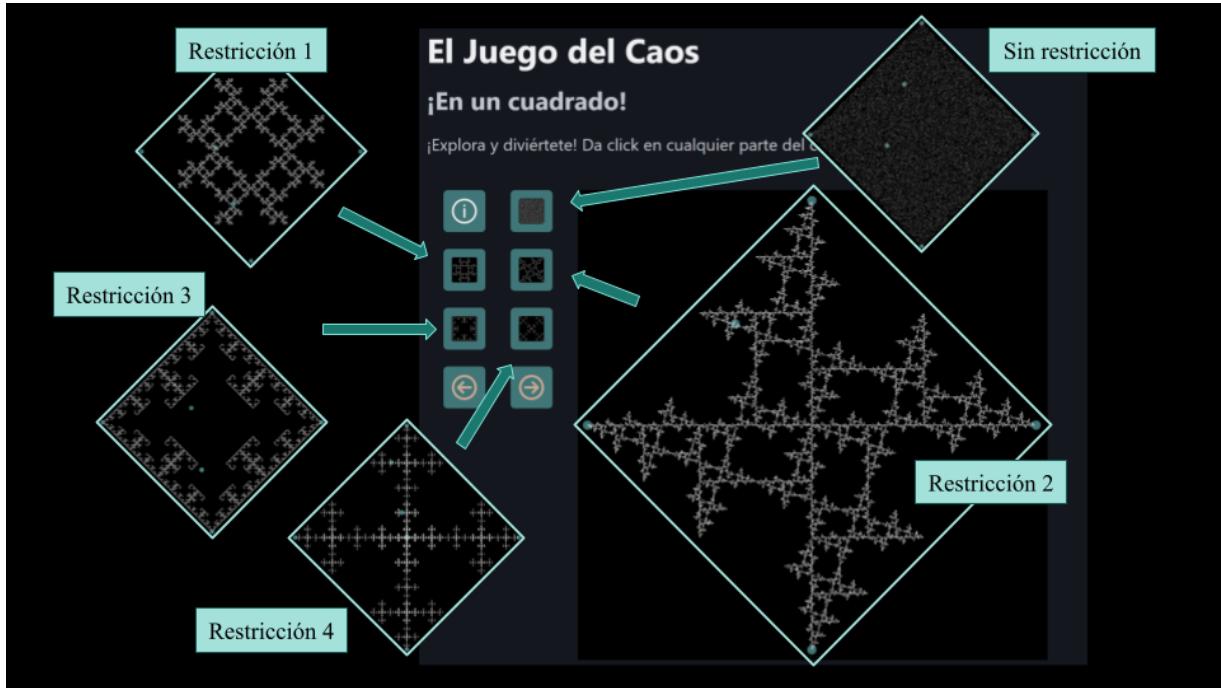


Figura 3.17: Restricciones en cuadrados disponibles en la App se pueden vizualizar en esta figura.

3.3.3. Programación de la versión del juego del caos para secuencias de ADN.

En la tercera página, el juego del caos se adapta para generar las representaciones de secuencias de ADN. Esto incluye un mapeo directo de las bases nitrogenadas (A, C, T, G) a los vértices de un cuadrado, permitiendo observar patrones genéticos. Los botones informativos explican el algoritmo y presentan una secuencia predefinida que corresponde a los primeros 100,000 pares de bases del Homo Sapiens Chromosome 1 (CM000663.2), ayudando al usuario a comprender cómo las propiedades de las secuencias de ADN pueden visualizarse de manera fractal.



Figura 3.18: La página 3 de la App muestra la representación de un fragmento de la secuencia de Homo Sapiens Chromosome 1 (CM000663.2)

3.3.4. Desarrollo de la interfaz de usuario para manipulación de parámetros mediante la App.

La cuarta página reúne todas las funcionalidades de las páginas 1 y 2, brindando al usuario la capacidad de experimentar con combinaciones de restricciones y parámetros en tiempo real, a través de permitir cambios en los parámetros *"factor"* (la proporción distancia), *"lados"* (número de lados del polígono), a lo largo de las cuatro distintas restricciones disponibles que corresponden a las mencionadas en el Capítulo 1 de este trabajo de tesis, así como la consulta de información referente al trasfondo teórico y conceptual del juego del caos.

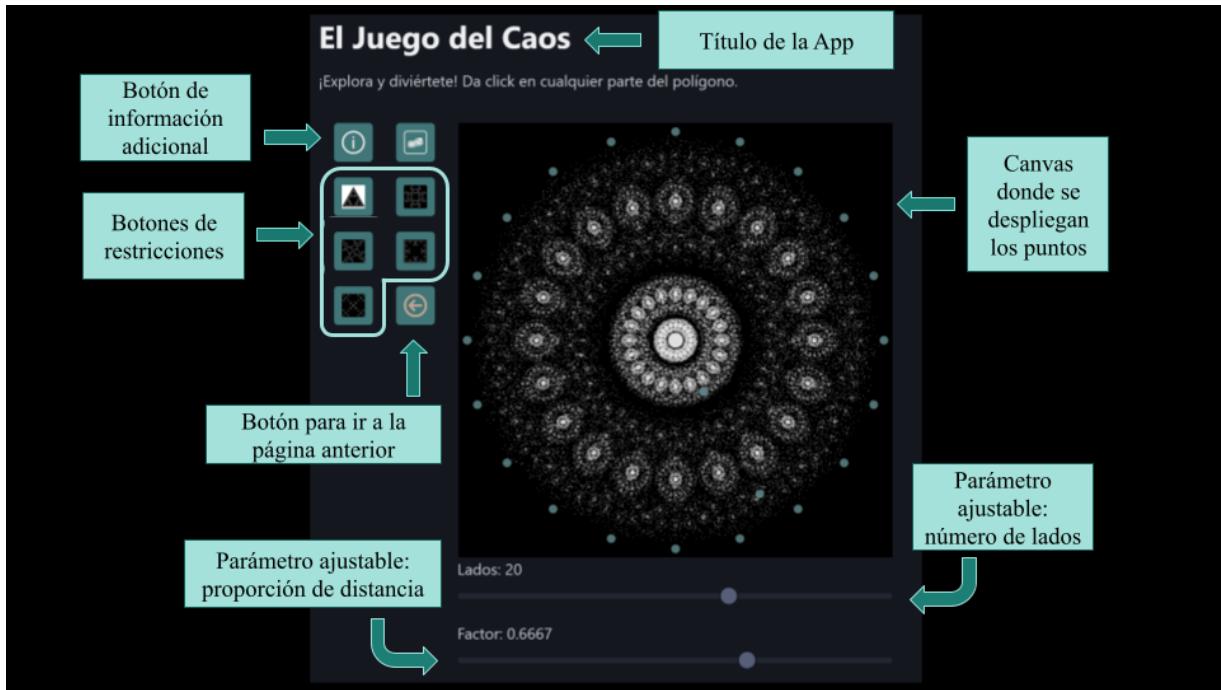


Figura 3.19: Aquí se observa la estructura de la página 4 de la App con los botones de las interacciones correspondientes.

3.3.5. Desarrollo de un programa en Python que genere la CGR de una secuencia dada.

link a un cuaderno de jupyter notebook en un repo de la tesis

Capítulo 4

Materiales y métodos

Poner referencias

Como parte de este proyecto de tesis se desarrollaron principalmente dos proyectos, el primero consistió en la generación de las DNA-CGR, es decir, la representación del juego del caos para secuencias de ADN las cuales fueron generadas con el algoritmo ya mencionado programado en Python aplicado a secuencias de ADN que previamente fueron pre-procesadas para obtener una mayor fiabilidad de la información; mientras que el segundo se enfoca en el desarrollo de una aplicación que sirve de herramienta digital para dos fines, el primero, divulgar conceptos matemáticos y biológicos de una manera interactiva y atractiva visualmente a través del juego del caos para formar parte del museo virtual de Matemáticas de la UNAM y la Sociedad Matemática Mexicana y, el segundo, representar geométricamente secuencias de ADN utilizando la técnica de la representación del juego del caos en el ADN (DNA Chaos Game Representation o DNA-CGR), lo que facilita la visualización de patrones genómicos complejos en forma de fractales, tales como frecuencias de k -meros, en particular, contenido de GC.

Una vez introducido el concepto y algoritmo del juego del caos, así como la adaptación de este para representar secuencias de ADN de manera geométrica, procedemos a programar ambas versiones con el lenguaje de programación Python para tener un código especializado de generación de imágenes.

4.1. Metodología 1. Representación visual de las frecuencias de *k*-meros en secuencias de ADN.

Para esta etapa del estudio se programó el algoritmo con Python para la generación masiva de las imágenes de CGR y su almacenamiento en una base de datos para su posterior análisis. El cual se resume en obtener las secuencias de NCBI, pre-procesarlas y obtener los archivos con la información de las secuencias de ADN en formato .fasta a los que posteriormente se aplica el algoritmo del juego del caos, y se almacenan las imágenes resultantes en un formato visual atractivo que captura patrones genómicos en formato fractal.

4.1.1. Descarga de muestras de bacterias clínicamente relevantes.

1. Descarga y almacenamiento de los datos crudos obtenidos de NCBI por medio de Bash, provistos por el concurso CAMDA 2024 para el reto de resistencia, considerando 5,554 muestras de 8 géneros de bacterias *Acinetobacter Baumannii* (214), *Campylobacter Je-juni* (468), *Enterobacter Cloacae* (34), *Escherichia Coli* (210), *Klebsiella Pneumoniae* (1,755), *Neisseria Gonorrhoeae* (178), *Pseudomonas Aeruginosa* (536) y *Salmonella Enterica* (2,159).
2. Se realiza el control de calidad, ensamblado y anotación de dichas muestras con recursos provistos por los servidores Patric, Alnitak, Betterlab y Geomtop. Obteniendo la información de las secuencias en archivos de contigs con formato .fasta.

4.1.2. Generación del código del juego del caos

Se programa el algoritmo con el lenguaje de programación Python haciendo uso de paqueteterías especializadas para graficar datos masivos con el fin de obtener y almacenar las imágenes y coordenadas de los puntos calculados.

El código se estructuró de la siguiente manera:

- a) Lectura de secuencias en formato FASTA.

Se desarrolló una función para procesar archivos FASTA, donde se extraen secuencias de

ADN al ignorar las líneas de encabezado y concatenar las subsecuencias en una única cadena. Este procedimiento garantiza que la información genómica se represente de manera continua para su posterior análisis.

b) Cálculo de puntos mediante el Juego del Caos.

A partir de las secuencias leídas, se utiliza el algoritmo del Juego del Caos para mapear cada nucleótido (A, C, G, T) a un vértice fijo de un lienzo cuadrado. Cada posición se calcula como el promedio entre la coordenada actual y la del vértice correspondiente al nucleótido (ver 1.5.1). Este método genera un conjunto de coordenadas que captura propiedades fractales inherentes a la secuencia.

c) Almacenamiento de coordenadas generadas.

Se implementó un procedimiento para exportar las coordenadas calculadas a archivos de texto con formato (x,y) . Esto permite que los puntos generados sean reutilizados o analizados en otros contextos bioinformáticos.

d) Generación de imágenes basadas en coordenadas.

Mediante el paquete Datashader, se mapean las coordenadas generadas a un lienzo de alta resolución ($1800 \times 1800\text{px}$). Se aplican transformaciones logarítmicas u otras configuraciones personalizables para optimizar la representación visual. Los colores utilizados se definen a través de un colormap, como GnBu9, lo que produce imágenes detalladas y claras.

e) Exportación de visualizaciones.

Las imágenes generadas se guardan en formato PNG con fondo negro, lo que resalta los patrones fractales y asegura su fácil interpretación. Este paso es clave para integrar los resultados en presentaciones, análisis bioinformáticos y comunicación científica.

f) Flexibilidad y personalización.

El programa admite configuraciones personalizables como el tamaño del lienzo, el colormap, y el directorio de salida para las coordenadas e imágenes. Esto permite adaptar el análisis a diversas necesidades y escalas de trabajo.

Esta automatización permite procesar grandes volúmenes de secuencias de ADN de manera eficiente y uniforme y, a su vez, representar visualmente las características fractales de secuencias

genómicas, facilitando su integración en estudios de clasificación, análisis comparativo y difusión científica sobre el Juego del Caos.

4.1.3. Generación de simulaciones de secuencias específicas con distintos sesgos y distribuciones de nucleótidos.

Con el objetivo de validar los patrones observados en secuencias reales maximizando sus frecuencias conocidas, se implementaron una serie de programas en Python que permiten obtener las frecuencias de los distintos k -meros en secuencias reales e implementar la información obtenida para generar secuencias de nucleótidos artificiales. Los programas clave para este fin son dos, el primero consta de una calculadora de frecuencias de k -meros que permite visualizar una lista de los más abundantes y sus frecuencias a lo largo de la secuencia, el segundo consta de un generador de archivos FASTA que permite experimentar con cadenas (k -meros) personalizadas y sus distintas distribuciones, así como ponderaciones. Cada uno de estos programas sigue un enfoque modular dividido en etapas como se describe a continuación:

1. Calculadora de frecuencias de k -meros.

- a) Lectura de secuencias en formato FASTA.

Se desarrolló una función para extraer secuencias de ADN de archivos FASTA, ignorando encabezados y concatenando las líneas en una sola cadena continua. Esto asegura la correcta manipulación de los datos biológicos en pasos posteriores.

- b) Extracción de k -meros.

A partir de la secuencia, se descompone en segmentos consecutivos de longitud k (k -meros), preservando la posición relativa de cada subsecuencia. Este paso es esencial para analizar patrones repetitivos en diversos tamaños.

- c) Conteo de k -meros y generación de frecuencias.

Se calculan las frecuencias absolutas de cada k -mero en la secuencia, y éstas se pueden normalizar para obtener distribuciones probabilísticas. Esto facilita la identificación de k -meros predominantes que reflejan características específicas de las secuencias reales.

- d) Identificación de los k -meros más frecuentes.

Se seleccionan los k -meros con mayor ocurrencia mediante un método que ordena las frecuencias en orden descendente. Este enfoque permite priorizar los patrones más representativos de la muestra analizada.

e) Visualización y análisis de patrones recurrentes.

Los resultados obtenidos pueden exportarse en un formato tabular (e.g., DataFrame de pandas) para facilitar su interpretación y almacenamiento. Esto habilita su integración en análisis bioinformáticos posteriores o su uso como referencia en simulaciones.

f) Aplicación en simulaciones.

Los k -meros más frecuentes, junto con sus frecuencias relativas, se utilizan para definir distribuciones ponderadas en simulaciones de ADN. De este modo, las secuencias simuladas reproducen características observadas en las muestras reales, permitiendo validar y replicar patrones específicos.

Este enfoque flexible y personalizable fue clave para analizar las distribuciones de k -meros y para ajustar simulaciones de ADN que imitan fielmente las propiedades estadísticas de las secuencias reales estudiadas.

2. Generador de secuencias artificiales.

a) Definición de conjuntos de caracteres o cadenas.

Se definieron alfabetos personalizados que incluyen nucleótidos individuales (A, C, G, T), dímeros, trímeros y k -meros más largos. Adicionalmente, se diseñaron alfabetos basados en patrones obtenidos de análisis previos en secuencias reales. Esta flexibilidad permite ajustar las simulaciones para maximizar la similitud con distribuciones reales o explorar configuraciones hipotéticas.

b) Generación de secuencias aleatorias.

Se emplearon diferentes distribuciones probabilísticas para modelar la ocurrencia de los elementos del alfabeto seleccionado:

- Uniforme: Todos los elementos tienen la misma probabilidad.
- Ponderada: Se asignaron pesos específicos a los elementos para reflejar distribuciones conocidas de frecuencias.

- Normal: Modela una distribución centrada en un valor medio con cierta desviación estándar.
- Personalizada: Define explícitamente las probabilidades para cada elemento del alfabeto.

En cada caso, las probabilidades se normalizan para garantizar que sumen uno, y las secuencias se generan seleccionando elementos de acuerdo con la distribución especificada.

c) Formato FASTA y almacenamiento.

Las secuencias generadas se guardaron en archivos con formato FASTA, lo que facilita su uso posterior en análisis bioinformáticos. Para ello, el programa formatea las secuencias en líneas de 80 caracteres e incluye encabezados descriptivos.

d) Validación de patrones en secuencias simuladas.

Se generaron simulaciones controladas con distribuciones ponderadas con alfabetos personalizados para maximizar la representación de patrones previamente identificados en secuencias reales.

El programa se diseñó para ser altamente personalizable, permitiendo simulaciones desde distribuciones simples hasta configuraciones más complejas. Este enfoque fue clave para evaluar la relación entre los patrones observados y las distribuciones esperadas en el contexto de las secuencias reales analizadas.

4.1.4. Representación visual de las frecuencias de k -meros en secuencias de ADN.

Las secuencias obtenidas con los métodos descritos en 4.1.3 fueron utilizadas para generar sus respectivas CGR's, así como almacenar información relevante para futuros análisis por medio del código descrito en 4.1.2.

4.2. Metodología 2. Obtención de propiedades de las representaciones.

4.2.1. Dimensión fractal de correlación de las imágenes.

A partir de los archivos de texto con la información de las coordenadas calculadas con el juego del caos para secuencias de ADN, se genera un código que realiza el cálculo estimado de la dimensión de correlación de las representaciones, la selección de esta dimensión es motivada por la naturaleza de los datos y el comportamiento fractal en la distribución de los mismos, sin embargo, debido a la masividad de información no fue posible hacer el cálculo preciso de las dimensiones fractales, por lo que, se realiza una estimación por medio de muestreo incremental.

4.2.2. Utilizar las características obtenidas para implementar un algoritmo de clasificación.

Dimensión fractal de correlación de las imágenes.

Una vez generada la base de datos de las dimensiones fractales para cada muestra, se realiza un análisis estadístico descriptivo por medio de gráficos de box-plot y cálculo de correlación entre la dimensión fractal por género de bacteria con respecto a su fenotipo en cuanto a resistencia y susceptibilidad a los antibióticos meropenen y ciprofloxacin.

4.3. Metodología 3. Creación de una herramienta de visualización para la comunicación científica del juego del caos.

4.3.1. Programa del algoritmo del juego del caos para polígonos de n lados con el framework AngularJS.

Descripción

Se implementó un algoritmo generalizado del juego del caos para polígonos de n lados, utilizando un enfoque modular y escalable. El algoritmo permite generar patrones fractales caracte-

rísticos, variando parámetros como el número de lados, el factor de interpolación y restricciones en la selección de vértices.

Arquitectura del Software

La aplicación sigue un patrón de arquitectura MVC (Modelo-Vista-Controlador) que facilita la separación de la lógica del algoritmo del juego del caos tanto en polígonos como para secuencias de ADN, y la presentación gráfica, permitiendo una mayor escalabilidad y facilidad de mantenimiento.

Lenguaje de Programación y Herramientas Utilizadas

La aplicación fue desarrollada utilizando el framework AngularJS de JavaScript, con el apoyo de la biblioteca p5.js para la visualización gráfica de los patrones generados por el juego del caos. Esta biblioteca permite crear gráficos y visualizaciones interactivas, y se utilizó específicamente para generar representaciones visuales de los fractales creados a partir del juego del caos y su aplicación a secuencias de ADN. Para la gestión de datos y la implementación de los motores que simulan los distintos casos del juego del caos, se empleó el lenguaje TypeScript, lo que permitió una estructura más robusta y tipada en el manejo del código. En particular, se importa el decorador Component y la interfaz OnInit de Angular para definir la estructura y el comportamiento de los componentes de la aplicación, así como el Router de Angular para gestionar la navegación entre diferentes componentes, asegurando una experiencia de usuario fluida.

4.3.2. Adaptación del algoritmo para generar restricciones en el juego del caos.

Descripción

Se incluyeron variantes al algoritmo original que permiten generar restricciones específicas, como prohibir la selección de vértices consecutivos o modificar el parámetro de proporción de distancia. Estas adaptaciones permiten explorar patrones fractales más complejos y diversos.

Validación y Pruebas

La aplicación fue probada para el juego del caos aplicado a polígonos de tres, cuatro, cinco, siete y once lados, y los primeros diezmil caracteres del genoma de homo sapiens cromosoma 1 para validar la precisión de la CGR, obteniendo resultados consistentes con estudios previos.

4.3.3. Programa de la versión del juego del caos para secuencias de ADN.

Descripción

El algoritmo fue adaptado para representar secuencias de ADN mediante el mapeo de las bases nitrogenadas (A, T, C, G) a vértices específicos de un cuadrado. Esto permite generar representaciones visuales que capturan patrones significativos en las secuencias biológicas.

Limitaciones del Proceso y el Software

Una limitación encontrada fue la capacidad de procesamiento de grandes volúmenes de datos, lo cual afecta la velocidad de visualización de secuencias mayores a 10,000 pares de bases, principalmente por la traducción mediante el uso de diccionarios en el proceso del algoritmo.

4.3.4. Desarrollo de la interfaz de usuario para su interacción con la App.

Descripción

La interfaz fue diseñada para proporcionar una experiencia de usuario intuitiva y educativa. Se implementaron cuatro páginas principales que agrupan los siete componentes desarrollados, con la siguiente estructura:

1. Introducción al Triángulo de Sierpinski y el Algoritmo Básico

- Propósito: Presentar los fundamentos del Juego del Caos a través del Triángulo de Sierpinski.
- Contenido:
 - Una explicación breve del algoritmo básico.

- Parámetros limitados que no permiten modificar el factor de distancia.
- Un botón de información que detalla el proceso detrás de la generación del fractal.

2. Exploración de Variantes en Cuadrados

- Propósito: Mostrar cómo el algoritmo del Juego del Caos puede adaptarse a formas cuadradas.
- Contenido:
 - Variantes del algoritmo que incluyen restricciones adicionales, como "Snowflake" (prohibir ciertos vértices consecutivos).
 - Parámetros ajustables para incluir ruido o cambios en el factor de distancia.
 - Información explicativa sobre las restricciones utilizadas y su impacto en los patrones generados.

3. Aplicaciones en ADN

- Propósito: Conectar el Juego del Caos con su uso en bioinformática, específicamente en la representación de secuencias de ADN.
- Contenido:
 - Una descripción del mapeo de bases de ADN (A, T, C, G) a vértices específicos del caos cuadrado.
 - Una muestra visual generada a partir de una secuencia real.
 - Información detallada sobre cómo el algoritmo se adapta para trabajar con secuencias biológicas y qué patrones son relevantes en este contexto.

4. Exploración Interactiva de Algoritmos

- Propósito: Proporcionar un espacio para experimentar con variantes del algoritmo.
- Contenido:

- Una interfaz interactiva que permite seleccionar el número de lados del polígono, el factor de distancia y restricciones adicionales.
- Parámetros personalizables para simular escenarios con diferentes reglas.
- Explicaciones en tiempo real sobre las configuraciones elegidas y cómo afectan al patrón fractal.

Esta estructura asegura una experiencia educativa y práctica, adecuada tanto para principiantes como para usuarios avanzados interesados en algoritmos fractales y sus aplicaciones.

Mejoras Futuras:

- Preprocesamiento de la información de secuencias para suprimir el uso de diccionarios en el código.
- En futuras versiones se planea incorporar técnicas de preprocesamiento y codificación de las secuencias para mejorar el rendimiento en el procesamiento de datos con grandes longitudes.

Capítulo 5

Perspectiva/Discusión

Este trabajo explora el potencial de las representaciones fractales generadas mediante el juego del caos en el análisis y clasificación de datos genómicos, destacando tanto los retos como las oportunidades de mejora. A continuación, se analizan las áreas clave:

5.1. Representaciones Fractales y Clasificación

Las CGR's (Chaos Game Representations) ofrecen una manera innovadora de transformar secuencias genómicas en patrones visuales. Este enfoque tiene aplicaciones en bioinformática, como la clasificación de organismos y la detección de características biológicas relevantes. Inspirados en el trabajo de Zhao et al., 2023, quienes reportaron una precisión del 80% en la clasificación de secuencias del gen 16S para el diagnóstico temprano de caries, se implementó una CNN para clasificar géneros bacterianos a partir de las CGR.

Sin embargo, los resultados obtenidos en este trabajo alcanzaron solo un 56% de precisión, lo que refleja la necesidad de optimizar tanto el preprocesamiento de las imágenes como la arquitectura de la red neuronal. A pesar de estos resultados iniciales, se identificaron áreas prometedoras para mejorar el rendimiento del modelo, entre ellas:

Mejoras en la arquitectura de la CNN: Ajustar hiperparámetros, añadir más capas convolucionales o implementar técnicas avanzadas como redes preentrenadas.

Enriquecimiento del conjunto de datos: Incorporar más ejemplos y diversificar las secuencias genómicas para robustecer el aprendizaje.

Focalización en resistencia antimicrobiana: Extender la clasificación hacia la identificación de bacterias resistentes o susceptibles a antibióticos como meropenem y ciprofloxacino.

Adicionalmente, el análisis de estas representaciones fractales se complementa con cálculos de:

Frecuencias de k-meros: Que destacan patrones únicos en las secuencias, aportando información útil sobre características funcionales o evolutivas.

Dimensiones fractales: Que cuantifican la complejidad de los patrones generados, ayudando a extraer características relevantes para la clasificación.

Estos métodos no solo permiten un análisis matemático más profundo, sino que también abren la puerta a explorar nuevas correlaciones biológicas entre los sujetos de estudio, en este caso, bacterias y características cuantificables de sus respectivas representaciones.

5.2. Aplicación y Desarrollo del Software

La aplicación desarrollada en este trabajo proporciona un entorno interactivo y educativo para explorar el juego del caos y sus aplicaciones en bioinformática. Comparada con otras herramientas existentes, esta aplicación destaca por su enfoque en la interpretabilidad, permitiendo al usuario comprender fácilmente cómo los parámetros del algoritmo afectan los patrones generados. Esta característica es crucial para democratizar el uso de estas técnicas tanto en educación como en investigación.

En términos de desarrollo y potencial, se identifican las siguientes perspectivas:

Rendimiento y escalabilidad: Optimizar el procesamiento para manejar secuencias más largas permitirá ampliar su aplicabilidad.

Personalización y análisis avanzado: Incorporar herramientas que faciliten simulaciones más complejas y exportación de datos para análisis externos.

Visualizaciones tridimensionales: Explorar representaciones fractales 3D podría revelar patrones ocultos en los datos genómicos.

Finalmente, este trabajo resalta cómo la teoría del juego del caos puede integrarse exitosamente en bioinformática, ofreciendo no solo una herramienta visual poderosa, sino también una base teórica sólida para entender mejor los patrones genómicos y su relación con características

biológicas relevantes. Esto tiene el potencial de impulsar descubrimientos en áreas clave como la clasificación de organismos y la investigación en resistencia antimicrobiana.

Bibliografía

- AVS, S. K., Patle, S., Kaur, P., Omkumar, S., & Sharma, A. (2022). Genomics. En D. Kar & S. Sarkar (Eds.), *Genetics Fundamentals Notes* (pp. 699-760). Springer Nature Singapore. https://doi.org/10.1007/978-981-16-7041-1_15
- BARNSLEY, M. F. (1993). Chapter III - Transformations on Metric Spaces; Contraction Mappings; and the Construction of Fractals. En M. F. BARNSLEY (Ed.), *Fractals Everywhere (Second Edition)* (Second Edition, 42-CP8). Academic Press. <https://doi.org/https://doi.org/10.1016/B978-0-12-079061-6.50008-5>
- Jeffrey, H. (1990). Chaos game representation of gene structure. *Nucleic Acids Research*, 18(8), 2163-2170. <https://doi.org/10.1093/nar/18.8.2163>
- Nucleotides and Bases - Genetics Generation — knowgenetics.org [[Accessed 23-01-2025]]. (s.f.).
- Panchal, S. (2022). Fundamentals of Genetics. En D. Kar & S. Sarkar (Eds.), *Genetics Fundamentals Notes* (pp. 3-51). Springer Nature Singapore. https://doi.org/10.1007/978-981-16-7041-1_1
- Zhao, C., Chen, E., & Chen, T. (2023). Deep Learning Classification of Caries Based on DNA Sequences Transformed by Chaos Game Representation. *2023 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 4996-4998. <https://doi.org/10.1109/BIBM58861.2023.10385889>