Sistema de funciones iteradas con gráficas dirigidas

Geometría fractal

Vazquez Arriaga Jorge

1. Introducción

Durante el curso trabajo con Sistema de Funciones Iterados (SFI), sus propiedades y atractores, este proyecto se pretende extender este tema a los Sistemas de Funciones Iterados con Gráficas Dirigidas (DGIFS). Hablamos de una extensión ya que teniendo gráfica simple con un solo vértice obtenemos la teoría ya conocida de un SFI estándar, este hecho se abordara más a fondo posteriormente. Este proyecto tiene como propósito principal la construcción y visualización de DGFIS, para lo cual se hará un breve desarrollo teórico, en especial de Gráficas Dirigidas.
La teoría de gráficas es una rama de las matemáticas discretas que en los últimos años ha tenido un gran desarrollo, probablemente debido a la enorme cantidad de aplicaciones que estas tienen. Se considera que tiene su inicio en 1736 cuando Leonhard Euler [6] publicó *Solutio problematis and geometrian situs pertinentis 8* en donde aparece la solución al famoso Problema de los Puentes de Königsberg. Durante el siglo XIX la teoría de gráficas fue redescubierta a través del estudio de diversos problemas obteniendo así nuevos y más resultados importantes. En particular las gráficas dirigidas son una rama de la teoría de gráficas.
La terminología IFS, fue introducida por Barnsley [2] para describir un proceso de representación de imágenes de tales conjuntos. Los DGIFS, son una de las categorías más generales de sistemas dinámicos en los que se utiliza una gráfica dirigida para imponer un orden en el que se pueden aplicar las funciones asociadas. Si el grafo dirigido tiene n vértices, la dinámica de un DGIFS siempre produce un único conjunto de n componentes que se conoce como atractor.
El código utilizado se encuentra en github.com/arriagajorge/DGIFS.

2. Definiciones y notación

Las definiciones estándar se asumen siempre que no se indiquen explícitamente, por lo que, por ejemplo, tomamos como dadas las definiciones y la notación básicas de la teoría de conjuntos, las definiciones del conjunto de enteros positivos $\mathbb{N}$, el conjunto de números reales $\mathbb{R}$, la definición de un espacio métrico, SFI y sus propiedades, en general los temas tratados en el curso. Una selección de libros que se podrían consultar para las definiciones que no se dan en el texto son [1], [2] y [4]. Cuando hagamos una definición pondremos el objeto que se define en cursiva. Se incluyen un breve repaso a la Métrica de Hausdorff en $\mathbb{R}^n$ tanto por su notación como por su significado.

2.1. Métrica de Hausdorff

Sea A un conjunto no vacío de $\mathbb{R}^n$, entonces para $r > 0$, la *vecindad r-cerrada* es definida como

$$A(r) = \{x : x \in \mathbb{R}^n, \text{dist}(x, A) \leq r\},$$

donde dist es una función de distacia.
Sea $K(\mathbb{R}^n)$ que denota el conjunto formado por todos los subconjuntos compactos no vacíos de $\mathbb{R}^n$, entonces la *métrica de Hausdorff* es definida para cualquier $K(\mathbb{R}^n)$ como

$$d_H(A,B)) = \text{ínf}\{r : A \subset B(r), B \subset A(r)\}.$$

Una definición equivalente de la métrica de Hausdorff, dada por

$$d_H(A,B) = \text{máx}\{\text{dist}(a,B), \text{dist}(A,b) : a \in A, b \in B\},$$

véase [10]. Igualmente se puede demostrar que $(K(\mathbb{R}^n), d_H)$ es un espacio métrico completo, veáse Teorema 2.5.3 en [4].
Para un conjunto finito V, que contiene $\#V$ donde $\#V$ representa la cardinalidad del conjunto V, utilizaremos la siguiente notación para el $\#V$-veces producto cartesiano de $K(R^n)$,

$$(K(\mathbb{R}^n))^{\#V} = \underbrace{K(\mathbb{R}^n) \times K(\mathbb{R}^n) \times \cdots \times K(\mathbb{R}^n)}_{\#V \text{ veces}}$$

Para cualquier pareja ordenada, $(A_u)_{u\in V}, (B_u)_{u\in V} \in (K(\mathbb{R}^n))^{\#V}$, podemos definir la métrica D_H como

$$D_H((A_u)_{u\in V}, (B_u)_{u\in V} = \text{máx}\{d_H(A_u, B_u) : u \in V\} \tag{1}$$

Utilizando la completitud del espacio $(K(\mathbb{R}^n), d_H)$, se puede demostrar que el espacio producto $((K(\mathbb{R}^n))^{\#V}, D_H)$ es también un espacio métrico completo.

3. Gráficas Dirigidas

Un *vértice* es un nodo o punto. Si u y v representan vértices (no necesariamente distintos) entonces una *arista (dirigida)*, e, de u a v es un arco o flecha dirigida que conecta u con v. Las funciones de vértice inicial y terminal i y t (definidas a continuación) dan el valor inicial (inicio) y vértices terminales (final) de una arista, por lo que para una arista e de u a v, $i(e) = u$ y $t(e) = v$. Un *camino finito (dirigido)*, $\mathbf{e}$, que tiene una longitud $|\mathbf{e}| = k \in \mathbb{N}$, es una cadena finita de aristas consecutivas, y puede escribirse como $e = e_1 \cdots e_k$ para algunas aristas ei, $1 \leq i \leq k$, donde $t(e_j) = i(e_j + 1)$ para $1 \leq j \leq k-1$. Los *vértices de un camino* son los vértices iniciales y terminales de sus aristas constituyentes, por lo que el conjunto de vértices de un camino, $e = e_1 \cdots e_k$ es el conjunto $i(e_1), t(e_i) : 1 \leq i \leq k$. Definimos la *lista de vértices* de $\mathbf{e}$ como la cadena finita de vértices consecutivos, $v_1, v_2, v_3 \cdots v_{k+1} = i(e_1)t(e_1)t(e_2)\cdots t(ek)$, que muestra el orden en que el camino $\mathbf{e}$ visita sus vértices.
Una *gráfica dirigida* (V, E^*, i, t) consiste del conjunto de todos los vértices V y el conjunto de todos los caminos finitos (dirigidos) E^*, junto con las funciones de vértice inicial y final $i : E^* \to V$ y $t : E^*$. E^1 denota el conjunto de todas las aristas (dirigidas) de la gráfica, es decir, el conjunto de todos los caminos de longitud uno, con $E^1 \subset E^*$. V y E^1 se suponen siempre conjuntos finitos. Observe que puede haber más de una arista que conecte un determinado par de vértices, por lo que puede considerarse como una multigráfica. En la Figura 1 se muestran ejemplos de representación de gráficas dirigidas.
Las *funciones de vértice inicial y terminal* se definen como sigue. Tomemos $e \in E$, sea cualquier camino finito, entonces podemos escribir $e = e_1 \cdots e_k$ para algunas aristas $e_i \in E^1$, $1 \leq i \leq k$.

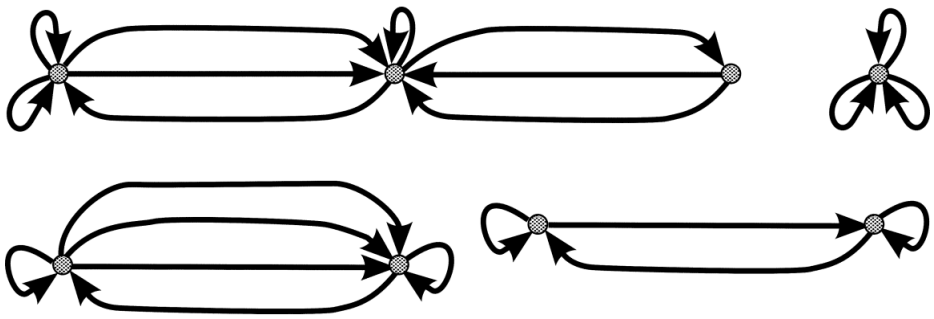


Figura 1: Ejemplos de Gráficas Dirigidas

El vértice inicial de **e** es el vértice inicial de su primera arista, por lo que $i(e) = i(e_1)$ y de forma similar para el vértice terminal $t(e) = t(e_k)$.
Para $\mathbf{f}, \mathbf{g} \in E^*$, **f** es un *subcamino* de g si y solo si $\mathbf{g} = \mathbf{sft}$ para algún $\mathbf{s}, \mathbf{t} \in E^*$, donde suponemos que el camino vacío es un elemento de E^*. Utilizamos la notación $\mathbf{f} \subset \mathbf{g}$ para indicar que **f** es un subcamino de **g**.
Para $\mathbf{f}, \mathbf{g} \in E^*$, **f** *no es un subcamino* de g si y solo si $\mathbf{g} \neq \mathbf{sft}$ para algún $\mathbf{s}, \mathbf{t} \in E^*$, donde suponemos que el camino vacío es un elemento de E^*. Utilizamos la notación $\mathbf{f} \not\subset \mathbf{g}$ para indicar que **f** no es un subcamino de **g**.
Un *camino simple* no visita ningún vértice más de una vez, por lo que un camino $e = e_1, \cdots, e_k \in E^*$ es simple si su lista de vértices contiene exactamente $k + 1$ vértices diferentes. Para las caminos simples utilizamos la notación **p**.
Un *ciclo* es un camino que tiene los mismos vértices iniciales y terminales, por lo que el camino $\mathbf{e} \in E^*$ es un ciclo si $i(e) = t(e)$. Un ciclo es independiente de su vértice inicial y terminal, es decir, si $e = e_1 e_2 \cdots e_k$ es un ciclo entonces también podemos escribir $e = e_k e_1 e_2 \cdots e_{k-1}$.
Un *ciclo simple* es un ciclo que no visita ningún vértice más de una vez aparte de los vértices iniciales y terminales que son los mismos, por lo que si $e = e_1 \cdots e_k \in E^*$ es un ciclo entonces $i(e) = t(e)$ y su lista de vértices contiene exactamente k vértices diferentes. Para ciclos simples reservamos la letra **c** en negrita y minúscula. Un ciclo simple de longitud uno se suele llamar bucle.
Escribimos E^k para el conjunto de todos los caminos de longitud k, E^k_u para el conjunto de todos los caminos de longitud k que parten del vértice u y E^k_{uv} para el conjunto de todos los caminos de longitud k que empiezan en el vértice u y que terminan en v. De forma similar, E^*_u denota el conjunto de todos los caminos finitos que comienzan en el vértice u y E^*_{uv} el conjunto de todos los caminos finitos que van del vértice u a v. E^*_{uu} denota el conjunto de caminos finitos o ciclos que comienzan y terminan en u. Podemos escribir $E^{\leq k}_{uv}$ para el conjunto de todos los caminos, de u a v, de longitud menor o igual a k. Utilizamos D^*_{uv} para el conjunto de todos los caminos simples finitos desde el vértice u al vértice v, con $D^*_{uv} \subset E^*_{uv}$.
En la Figura 2 se muestra un ejemplo concreto de una gráfica dirigida G, en este caso tenemos $V = \{\texttt{S, T}\}$ y $E = \{\texttt{a,b,c,d,e,g}\}$ y por ejemplo $i(\texttt{c}) = \texttt{S}, t(\texttt{c}) = \texttt{T}, i(\texttt{f}) = t(\texttt{f}) = \texttt{T}$.

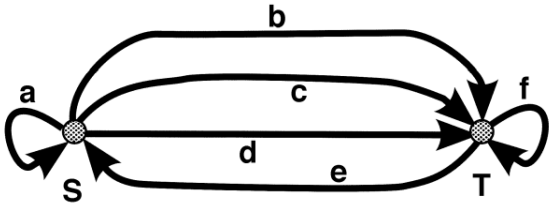


Figura 2: Ejemplo particular de una Gráfica dirigida

4. DGIFS

$(V, E^*, i, t, r, ((X_v, d_v))_{v\in V}, (S_e)_{e\in E^1})$ denota una *Sistema de funciones iterados con gráficas dirigidas* (DGIFS) y siempre asumimos que la gráfica dirigida está *fuertemente conectada*, por lo que hay al menos un camino que conecta a dos vértices cualesquiera. También suponemos que cada vértice de la gráfica dirigida tiene al menos dos aristas que lo abandonan, esto es para evitar conjuntos autosimilares que consisten en conjuntos de un solo punto, y atractores que son sólo copias escalares de los de otros vértices (véase [5]). La función $r : E^* \to (0, 1)$ asigna una factor de contracción a los caminos finitos en la gráfica. A cada vértice $v \in V$, se le asocia un espacio métrico completo (X_v, d_v) y a cada arista dirigida $e \in E^1$ se le asigna una contracción $S_e : X_{t(e)} \to X_{i(e)}$, que tiene factor de contracción dado por la función $r(e) = r_e$. Seguimos la convención ya establecida en la literatura, ver [4] o [5], que una similitud se dibuje en la dirección opuesta a la de la arista a la que está asociada en la gráfica.
La función del factor de contracción $r : E^* \to (0, 1)$, el *factor de contracción* a lo largo de un camino $e = e_1e_2 \cdots e_k \in E^*$ es definido como $r(e) = r_{\mathbf{e}} = r_{e_1}r_{e_2} \cdots r_{e_k}$. El factor $r_{\mathbf{e}}$ es el factor para la contracción $S_e : X_{t(e)} \to X_{i(e)}$ a través del camino $\mathbf{e}$, donde $S_{\mathbf{e}} = S_{e_1} \circ S_{e_2} \circ \cdots \circ Se_k$.
En este trabajo sólo vamos a ocuparnos de los IFS con graficas dirigidas definidos en un espacio euclidiano de n dimensiones y métrica euclidiana, con $((X_v, d_v))_{v\in V} = ((\mathbb{R}^n, |\quad|))_{v\in V}$ y donde $(S_e)_{e\in E^1}$ son contracciones autosimiliares y no sólo contracciones. Se puede asumir entonces, siempre que la notación $(V, E^*, i, t, r, ((\mathbb{R}^n, |\quad|))_{v\in V}, (S_e)_{e\in E^1})$ aparece en el texto, que para cada $e \in E^1$, $S_e : \mathbb{R}^n \to \mathbb{R}^n$ es una contracción autosimilar con r_e, $0 < r_e < 1$ el factor de contracción.

4.1. Condiciones de Separación

Un SFI con gráficas dirigidas $(V, E^*, i, t, r, ((\mathbb{R}^n, |\quad|))_{v\in V}, (S_e)_{e\in E^1})$, determina un única lista de conjuntos compactos no vacíos, $(F_u)_{u\in V} \in (K(\mathbb{R}^n))^{\#V}$, esta propiedad se puede ver reflejada en el Teorema 1. Los conjuntos $(F_u)_{u\in V}$ suelen denominarse lista de atractores o conjuntos autosimilares del sistema. Para obtener resultados significativos sobre tales atractores, a menudo es necesario restringir el gráfico dirigido IFS a un sistema que satisface una o más de las condiciones de separación que se definen a continuación. Como un atractor F_u es un conjunto compacto se deduce que la envoltura convexa $C(F_u)$ es también un compacto para cada $u \in V$. Escribimos el $\#V$-veces producto cartesiano de $\mathbb{R}^n$ como

$$(\mathbb{R}^n)^{\#V} = \mathbb{R}^n \times R^n \times \cdots \times \mathbb{R}^n$$

- La *condición de conjunto abierto* (OSC) se cumple si y sólo si existen conjuntos abiertos

acotados no vacíos, $(U_u)_{u\in V} \subset (\mathbb{R}^n)^{\#V}$, donde para cada $u \in V$

$$S_e(U_{t(e)}) \subset U_n \text{ para todo } e \in E_u^1,$$

y

$$S_e(U_{t(e)}) \cap S_f(U_{t(f)}) = \emptyset \text{ para todo } e, f \in E_u^1 \text{ con } e \neq f.$$

- La *condición de conjunto abierto fuerte* (SOSC) se satisface si y sólo si la OSC se satisface para los conjuntos abiertos acotados no vacíos $(U_u)u \in V \subset (\mathbb{R}^n)^{\#V}$ donde para cada $u \in V$

$$F_u \cap U_u \neq \emptyset.$$

- La *condición de conjunto abierto convexo* (COSC) se satisface si y sólo si la OSC se satisface para los conjuntos abiertos acotados no vacíos $(U_u)u \in V \subset (\mathbb{R}^n)^{\#V}$ donde estos conjuntos son también convexos.

- La *condición de separación fuerte* (SSC) se cumple si y sólo si para cada $u \in V$

$$S_e(F_{t(e)}) \cap S_f(F_{t(f)}) = \emptyset \text{ para todo } e, f \in E_u^1 \text{ con } e \neq f.$$

- La *condición de separación fuerte convexa* (CSSC) se cumple si y sólo si para cada $u \in V$,

$$S_e(C(F_{t(e)})) \cap S_f(C(F_{t(f)})) = \emptyset \text{ para todo } e, f \in E_u^1 \text{ con } e \neq f.$$

Ver [4], [7], [8], [9], [11].

4.2. Conjuntos Autosimilares

En esta subsección $((K(\mathbb{R}^n))^{\#V}, D_H)$ es un espacio métrico completo, donde cada componente $K(\mathbb{R}^n)$ es el conjunto de subconjuntos compactos no vacíos de R^n y D_H es la métrica definida en la ecuación (1), derivada de la métrica de Hausdorff, d_H.
A continuación se presentan dos de los teoremas más importantes en la teoría de DGIFS.

Teorema 1. *Sea $((A_{m,u})_{u\in V}$ una sucesión de conjuntos compactos no vacíos con $(A_{m,u})_{u\in V} \in (K(\mathbb{R}^n))^{\#V}$, para cada $m \in \mathbb{N} \cup \{0\}$, supongamos que estos decrecen,*

$$(A_{0,u})_{u\in V} \supset (A_{1,u})_{u\in V} \supset (A_{1,u})_{u\in V} \supset \cdots,$$

entonces $(A_{m,u})_{u\in V})$ converge, con respecto a la métrica de Haursdorff D_H, al compacto no vacío $(A_u)_{u\in V} \in (K(\mathbb{R}^n))^{\#V}$ donde

$$(A_u)_{u\in V} = \bigcap_{m=0}^{\infty} (A_{m,u})_{u\in V} = \left(\bigcap_{m=0}^{\infty} A_{m,u} \right)_{u\in V}$$

Prueba. La idea de la demostración se basa en probar que si (A_n), $n \in N \cup \{0\}$, es una sucesión decreciente de conjuntos compactos no vacíos en $K(\mathbb{R}^n)$ entonces converge con respecto a la métrica de Hausdorff d_H, a un conjunto compacto no vacío $A \in K(\mathbb{R}^n)$ donde $A = \bigcap_{n=0}^{\infty} A_n$ Así, para cada $u \in V$ el componente de la sucesión (A_n, u) converge a un conjunto compacto no vacío A_u con respecto a la métrica de Hausdorff d_H, donde $A_u = \bigcap_{n=0}^{\infty} A_{n,u}$ y el teorema se deduce de la definición de la métrica D_H.

Teorema 2. *Sea* $(V \in E^*, i, t, r, ((\mathbb{R}^n, |\quad|))_{u\in V}, (S_e)_{e\in E^1})$ *un DGIFS, y tomemos la función* $f : (K(\mathbb{R}^n))^{\#V} \to (K(\mathbb{R}^n))^{\#V}$ *dada por*

$$f\left((A_u)_{u\in V}\right) = \left(\bigcup_{e\in E_u^1} S_e(A_{t(e)})\right)_{u\in V}$$

para cada $(A_u)_{u\in V} \in (K(\mathbb{R}^n))^{\#V}$.
Entonces existe a una única lista de conjuntos compactos no vacíos $(F_u)u \in V \in (K(\mathbb{R}^n))^{\#V}$ *tal que*

$$f\left((F_u)\right) = \left(\bigcup_{e\in E^1} S_e(F_{t(e)})\right)_{u\in V} = (F_u)_{u\in V}$$

y donde, para cualquier $(A_u) \in (K(\mathbb{R}^n))^{\#V}$

$$D_H\left(f^k\left((A_u)_{u\in V}\right), (F_u)_{u\in V}\right) \to 0, \text{conforme } k \to \infty$$

Para cualquier $(B_u)_{u\in V} \in (K(\mathbb{R}^n))^{\#V}$, *tal que* $f\left((B_u)_{u\in V}\right)$ *donde hacemos* $f^0\left((B_u)_{u\in V}\right) = (B_u)_{u\in V}$

$$(F_u)_{u\in V} = \bigcap_{k=0}^{\infty} f^k((B_u)_{u\in V}).$$

La demostración excede los alcances de este proyecto se recomienda consultar [3].

5. Visualización DGIFS

Para visualizar DGIFS empezaremos con gráficas con un solo vértice.
Consideremos $(V, E^*, i, t, r, ((X_v, d_v))_{v\in V}, (S_d)_{d\in E^1})$ un con $V = \{v\}$, $E^1 = \{e, f, g\}$
$i(e) = i(f) = i(g) = v$, $t(e) = t(f) = t(g) = v$, (veáse Figura 3),
$r = 1/2$, $S_e(x, y) = r(x, y)$, $S_f(x, y) = r(x + 1, y)$, $S_g(x, y) = r(x + 1/2, y + 1)$, con $((X_v, d_v)) = ((\mathbb{R}^2, |\quad|))$.

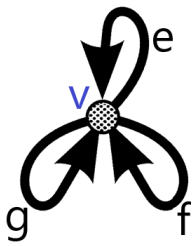


Figura 3: DGIFS con un único vértice

Como lo mencionamos un DGIFS con un solo vértice conlleva a los SFI ya conocidos ya que al haber solo un vértice la lista de de atractatores es un solo atractor y se encuentra en $(\mathbb{R}^2)^{\#V} = \mathbb{R}^2$.

Empezamos con un compacto $T0 \in \mathbb{R}^2$ a este compacto se le aplican las contracciones $(S_d)_{d\in E^1}$ tales que la arista d incide en v, en este ejemplo solo tenemos un vértice esto implica que todas las aristas inciden en v, así obtenemos un nuevo compacto a quien por simplicidad lo colocaremos el nombre de $S(T0)$, repetimos este proceso y obtenemos un nuevo compacto $S(S(T0))$, así repitiendo este proceso n veces, obtenemos una aproximación de la visualización del DGIFS con un vértice, y por tener un solo vértice obtenemos un SFI ya conocido. (Véase Figura 4)
El ejemplo de un vértice es ilustrativo y pretende dar una idea de como se construye un DGIFS, a continuación se presenta un ejemplo con cuatro vértices.

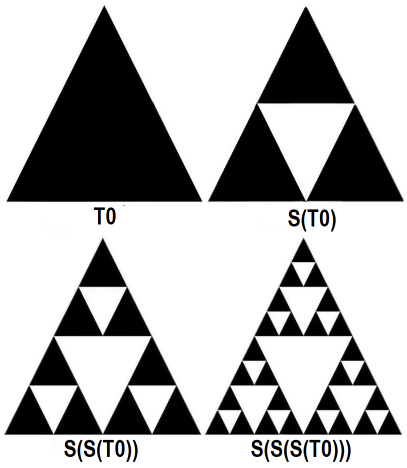


Figura 4: Iteraciones DGIFS con un único vértice

Consideremos $(V, E^*, i, t, r, ((X_v, d_v))_{v\in V}, (S_d)_{d\in E^1})$ un con
$V = \{v_1, v_2, v_3, v_4\}$, $E^1 = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$,
$i(e_1) = i(e_2) = i(e_3) = v_1, \quad i(e_4) = i(e_5) = i(e_6) = v_2,$
$i(e_7) = i(e_8) = i(e_9) = v_3, \quad i(e_{10}) = i(e_{11}) = v_4,$
$t(e_6) = t(e_7) = v_1, \quad t(e_2) = t(e_9) = t(e_{10}) = v_2,$
$t(e_3) = t(e_5) = t(e_{11}) = v_3, \quad t(e_1) = t(e_4) = t(e_8) = v_4$, (veáse Figura 5),
$r = 1/2$, $S_{e_1}(x, y) = S_{e_4}(x, y) = S_{e_7}(x, y) = S_{e_{10}}(x, y) = r(x, y)$,
$S_{e_2}(x, y) = S_{e_5}(x, y) = S_{e_8}(x, y) = S_{e_{11}}(x, y) = r(x + 1, y)$,
$S_{e_3}(x, y) = S_{e_6}(x, y) = S_{e_9}(x, y) = r(x + 1/2, y + 1)$, con $((X_v, d_v))_{v\in V} = ((\mathbb{R}^2, |\quad|))$.

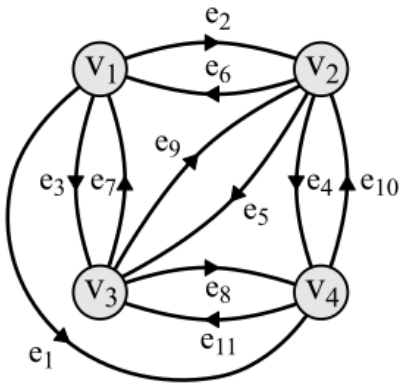


Figura 5: Gráfica asociada a un DGIFS con cuatro vértices

Empezamos asociando un compacto en el espacio $(\mathbb{R}^2, |\quad|)$ a cada vértice, para este ejemplo se toma un triángulo donde sus vértices estan en [0,0], [0,1], [1/2, 1/2], véase Figura 6.

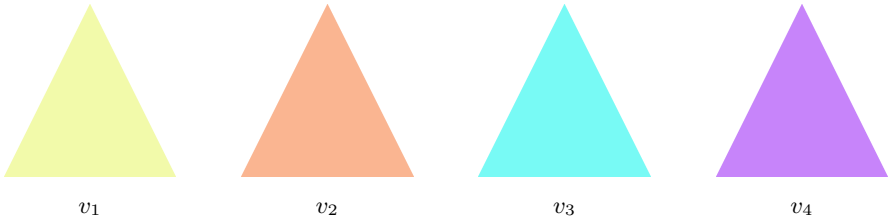


Figura 6: Compactos iniciales asociados a cada vértice

Una vez que contamos con los compactos iniciales, al espacio de cada vértice se le realizan las transformaciones correspondientes, y se actualizan los espacios como se puede ver en la Figura 7. Intuitivamente se pude entender como que cada vértice se le aplica una transformación que lo manda al espacio de otro vértice. Por ejemplo al vértice 4 tiene dos aristas e tal que $i(e) = v_4$, esta aristas son e_{10}, e_{11} por lo tanto va a recibir las transformaciones $S_{e_{10}} = r(x, y)$ del espacio del vértice $t(e_{10}) = v_2$, igualmente recibe la transomación $S_{e_{11}} = r(x, y + 1/2)$ del espacio del vértice $t(e_{11}) = v_3$. Así para cada Vértice

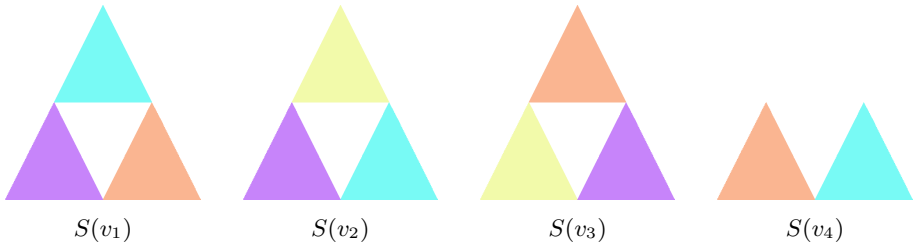


Figura 7: Primera iteración

En la Figura 8 se hace de nueva cuenta una iteración, esto corresponde a la segunda iteración del sistema, por simplicidad se usa $S^n(v_i)$ para indicar la $n-$ésima iteración del espacio asociado al vértice v_i.
En la Figura 9 se hace una aproximación al sistema, observé que esto produce resultados satisfactorios, esto se logro con solo 10 iteraciones.

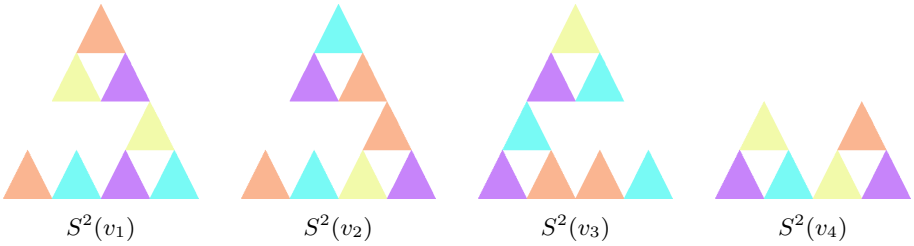


Figura 8: Segunda iteración

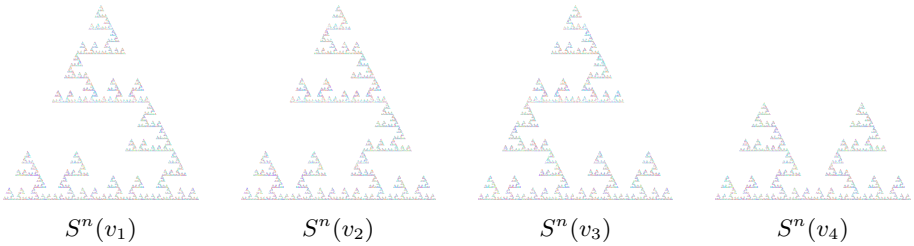


Figura 9: Compactos iniciales asociados a cada vértice

6. Implementación DGFIS

A continuación se presenta el algoritmo usado para dibujar DGIFS, con n iteraciones.

contador de iteraciones = 0;
número de iteraciones = n;
para v_i *en los vértices del DGIFS* **hacer**
  Dibujar una imagen que represente al compacto en el espacio asociado a v_i;
  Llamar a este espacio w_i
fin
mientras *contador de iteraciones sea menor que el numero de iteraciones* **hacer**
  para v_i *en los vértices del DGIFS* **hacer**
    Dibuar una imagen en blanco llamandole $\widehat{w_i}$;
    para v_j *que incide en el vértice* v_i **hacer**
      Aplicar la transformación asociada S_e con $i(e) = v_j$, $t(e) = v_i$, del espacio w_j al espacio $\widehat{w_i}$
    fin
  fin
  para w_i *espacio asociado al vertice* v_i *en los vértices del DGIFS* **hacer**
    Actualizar el espacio w_i por $\widehat{w_i}$
  fin
  contador de iteraciones += 1
fin

Algoritmo 1: Dibujo de DGIFS, n iteraciones.

El Algoritmo 1 presenta una forma en la que se pude dibujar DGIFS, sin embargo existen detalles como el dibujar compactos o aplicar transformaciones de una imagen que representa un espacio a otra imagen que representa otro espacio, que dependen del lenguaje de programación, y de la propia forma de programarlo, en la siguiente subsección se presenta un desarrollo en el lenguaje de programación `Julia`

6.1. Implementación en `Julia`

Empezamos con el dibujo de un compacto, en la Figura 10 se puede ver como se implementa, donde los píxeles blancos representan el complemento del conjunto.

```
d = RectangularRegion(0, 1, 0, 1, width=500, height=500)
W, H = size(draw(d))
# dibuar un triangulo de la manera adecuada
# colorear los pixeles que pertenecen al conjunto
color_ = RGB(0,0,0)
for h in 1:500
    if h%2==0
        j=h÷2-1
    else
        j=h÷2
    end
    for w in 1+j:500-j
        pixel!(d, (w, 501-h), color_)
    end
end
draw(d)
```

Figura 10: **Listing 1.** Dibujo de un conjunto compacto en una imagen donde los píxeles de color blanco representan complemento del conjunto

En la Figura 11 se muestra como implementar una transformación (`wi`) de un espacio (`RR_ant`) a

otro (`newRR`).

```
# función que aplica una transformación de un RectangularRegion a otra

# argumentos RR_ant: RectangularRegion inicial; wi: Transformación a aplicar;
# newRR: RectangularRegion donde se aplica la transormación, colormew: color, con este se colorea la transormación
function RR_wi_color(RR_ant, wi; newRR=RectangularRegion(0, 1, 0, 1), colornew=RGB(0,0,0))
    # obtener el Weight and Height
    W, H = size(draw(RR_ant))
    for w in 1:W
        for h in 1:H # Recorrer los pixeles de la imagen inicial
            if RR_ant.img[h,w] != RGB(1,1,1) # si el pixel no es blanco (pertenece al conjunto)
                point_ = pixeltopoint(RR_ant, (w,h)) #obtener el punto
                pixel_ = pointtopixel(RR_ant, wi(point_)) #aplicarle la transormación y convertirlo a pixel
                if containspixel(newRR, pixel_) # si la nueva RR contiene el punto , dibujarlo con un color
                    pixel!(newRR, pixel_, colornew)
                end #end contain pixel
            end #end pixel = RGB(1,1,1)
        end # end H
    end #end W
    return newRR #regresar la nueva región
end
```

Figura 11: **Listing 2.** Transformación a una `RectangularRegion`

Para la siguiente parte del código introducimos algo que llamamos **Matriz Asociada a DGIFS**, la cual es una matriz de tamaño $\#V, \#V$, es decir una matriz cuadrada de la cardinalidad del número de vértices.
Empezamos rellenando la matriz con ceros, luego para cada arista $e \in E^1$, hacemos la entrada $[t(e), i(e)]$ de la matriz sea igual a S_e. Al momento de implementar esta función podemos enumerar las contracciones S_e y en la matriz colocar el número correspondiente a la contracción.
Una vez teniendo la Matriz Asociada a DGIFS podemos implementar una función que dado un vértice `vi` le aplique todas sus transformaciones correspondientes para una iteración. En la Figura 12 se presenta la implementación.

```
# Función que le aplica las transformaciones a un vértice

# argumentos M:Matriz asociada al DGIFS; space_vis:Compactos iniciales asociados a cada vértice
# rr_ = RectangularRegion donde se encuentra el space_vis; ifs: el
function iterateOne(M, space_vis, vertex,;rr_ = RectangularRegion(0, 1, 0, 1), ifs=trs_ifs)
    iter_ = rr_
    for i in 1:size(M)[1] #recorre todos los vértices
        if  M[vertex,:][i] != 0 #revisa cuales inciden en el vértice
            iter_ = RR_wi(space_vis[i], ifs[M[vertex,:][i]], newRR=iter_) #lsi inciden se le aplica la transformación
        end
    end
    return iter_ # regresa el espacio con las transformaciones
end
```

Figura 12: **Listing 3.** Transformaciones al espacio asociado a un vértice

Una vez aplicadas las transformaciones para un vértice lo hacemos para todos los vértices. Aquí tenemos una iteración del sistema. Véase Figura 13 para la implementación.

```julia
# Función que le aplica las transformaciones a todos los vértice
# arumentos, initial_vis: espacio de vértices iniciales, M Matriz asociada al DGIFS
function iterateVis(initial_vis, M)
    vis = initial_vis
    nvis = copy(vis)
    n = size(M)[1]
    for i in 1:n #aplicamos la función que hace transformaciones a todos los vertices
        nvis[i] = iterateOne(M, vis, i)
    end
    return nvis
end
```

Figura 13: **Listing 4.** Transformaciones a cada uno de los espacios asociados a los vértices

Una vez que tenemos la una iteración, lo hacemos el número de iteraciones que deseemos. Adicionalmente se guarda cada iteración. En la Figura 14 se muestra la implementación, igualmente puede consultarse el notebook que se anexa y revisar el código fuente.

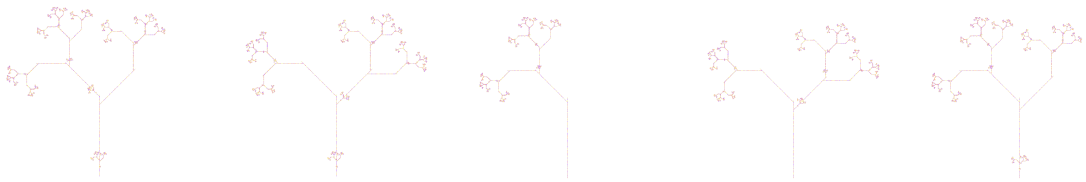
```julia
# función que aplica transformaciones a los vértices nrep-veces

# argumentos nrep; número de iteraciones, initial_vis: espacio de vértices iniciales, M Matriz asociada al DGIFS
function replicateIterSave(nrep, initial_vis, M)
    new_vis = iterateVis(initial_vis, M)
    lista = [new_vis] #lista para guardar cada una de las iteraciones
    for i in 1:nrep-1 #hacemos nrep iteraciones
        new_vis = iterateVis(new_vis, M) #aplicar las debidas transformaciones a todos los vértices
        push!(lista, new_vis) #guardamos esta iteración
    end
    return lista #regresamos todas las iteraciones
end
```

Figura 14: **Listing 5.** Iteraciones del sistema

Una vez con este algoritmo podemos dibujar otros DGIFS tomemos como ejemplo un DGIFS con $S_e^{(1)}(p) = \begin{bmatrix} 0 & 0 \\ 0 & 0.5 \end{bmatrix} p$, $S_e^{(2)} = \begin{bmatrix} 0.42 & -0.42 \\ 0.42 & 0.42 \end{bmatrix} p + [0, 2.1]$, $S_e^{(3)} = \begin{bmatrix} 0.42 & 0.42 \\ -0.42 & 0.42 \end{bmatrix} p + [0, 1.9]$ $S_e^{(4)} = \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix} p + [0, 0.2]$ donde $S_e^{(n)}$ representa la númeración de la n-ésima contracción, a estas contracciones las llamaremos `tree_ifs`. La matriz Asociada a este DGIFS, $M_2 = \begin{bmatrix} 0 & 2 & 3 & 1 & 4 \\ 4 & 0 & 2 & 1 & 3 \\ 1 & 0 & 0 & 2 & 0 \\ 3 & 1 & 2 & 0 & 0 \\ 1 & 4 & 3 & 2 & 0 \end{bmatrix}$. En la Figura 15 se presenta un representación de este DGIFS, donde se apoxima con 15 iteraciones.

7. Conclusiones

En este trabajo se presento una extensión de los sistema de funciones iterados, auxiliado de gráficas, esto resulto ser una gran ayuda ya que la representación visual de una gráfica permite darse una intuición de como se comportará el sistema, aunado a eso la teoría resulta ser amplia ya que como mencionamos se trata de una extensión de los SFI ya conocidos. Por otro lado la programación para visualizar es aceptable ya que realizar 15 iteraciones de 5 espacios tomaba poco más de un minuto pero es claro que aun se puede mejorar, tanto para disminuir tiempos de ejecución y espacios de



$S^n(v_1)$ $S^n(v_2)$ $S^n(v_3)$ $S^n(v_4)$ $S^n(v_5)$

Figura 15: Aproxmación DGIFS, con Matriz asociada M_2 y contraciones `tree_ifs`

memoria, igualmente para aumentar el número de píxeles y obtener mejores resultados. Por útlimo la referencia principal fue Boore [3] y el sitio web
`https://gcboore.com/pages/directed-graph-iterated-function-systems.html`.

Referencias

[1] T. M. Apostol, *Mathematical Analysis*, Addison-Wesley, Massachusetts, 1978.

[2] Barnsley, Michael F., *Fractals Everywhere*, Morgan Kauffman, 2nd Edición, 1993.

[3] G. C. Boore, *Directed Graph Iterated Function Systems*, Ph.D. thesis, School of Mathematics and Statistics, St Andrews University, 2011.

[4] G. A. Edgar, *Measure, Topology, and Fractal Geometry*, Springer-Verlag, New York, 2000

[5] G. A. Edgar and R. D. Mauldin, *Multifractal decompositions of digraph recursive fractals*, Proc. London Math. Soc. (3) 65, 1992

[6] L. Euler, *Solutio problematis and geometrian situs pertinentis, Commentarii Academiae Scientiarumhi Imperialis Petropolitanae 8*, 128–140, 1736

[7] K. J. Falconer, *Fractal Geometry, Mathematical Foundations and Applications*, John Wiley, Chichester, 2nd Ed. 2003.

[8] De-J. Feng and Y. Wang, *On the structures of generating iterated function systems of Cantor sets*, Adv. Math. 222, 2009

[9] J. Hutchinson, *Fractals and self-similarity*, Indiana Univ. Math. J. 30, 1981.

[10] P. Mattila, *Geometry of Sets and Measures in Euclidean Spaces*, Cambridge University Press, Cambridge, 1999

[11] J. Wang, *The open set conditions for graph directed self-similar sets*, Random Comput. Dynam. 5 4, 1997.