Sobre Fractales como Atractores de Sistemas

Manuel Murillo Tsijli[1]

Resumen. En este trabajo se hace una breve introducción al tema de los Fractales, autosemejanza, sistemas iterados de funciones y sus atractores. Además, se dan algunos programas computacionales para una posible experimentación.

Introducción

Los fractales constituyen un tema matemático de actualidad, gracias a las hermosas figuras que se generan por computadora siguiendo los procedimientos recursivos, las figuras fractales se han popularizado en los últimos años y cautivan a todo tipo de público. La definición precisa de fractal tiene que ver con su dimensión (Hausdorff, topológica, fractal, y otras), sin embargo, sobre eso se ha escrito bastante, para los efectos de este trabajo, pensemos que un fractal es algo irregular, pero lo más importante es que si lo ampliamos arbitrariamente, él aún sigue irregular. En general los fractales son figuras geométricas que se caracterizan por su autosemejanza, son estructuras infinitas contenidas en una superficie finita y resultan de utilidad en áreas como la botánica, la biología, la economía, la computación, etc. en el análisis de una gran diversidad de fenómenos como: turbulencias, bolsa de valores, dispersión del humo, percolación, etc., además de sintetizar imágenes como montañas, nubes, costas rocosas, ríos y plantas entre otras. Las figuras fractales se obtienen de repetir una y otra vez el mismo procedimiento, en forma recursiva.

Sistemas iterados de funciones

Una de las formas más populares para generar fractales es usar sistemas iterados[2] de funciones, dado que estos conjuntos presentan autosemejanza, es decir, el conjunto se puede descomponer en un número finito de partes de modo que una de ellas es idéntica, salvo escala, al todo. Para definir formalmente autosemejanza de conjuntos, se darán algunas definiciones básicas.

Definición 1. *Una función* $f: S \to T$ *con* (S, ρ) *y* (T, τ) *espacios métricos, es una semejanza o aplicación afín de razón* r *si cumple*

$$\tau(f(x), f(y)) = r\rho(x, y) \quad \forall x, y \in S. \tag{1}$$

Se pueden describir estas semejanzas de $\mathbb{R}^2$ en $\mathbb{R}^2$ como traslaciones, rotaciones, reflexiones y permutación de ejes o composiciones de estos tres tipos de transformaciones, y se pueden representar conmo:

$$v_i \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e_i \\ f_i \end{pmatrix} \tag{2}$$

Definición 2. *Una lista finita de razones es una* n*-tupla ordenada* $(r_1, r_2, \ldots, r_n)$ *de números positivos; la lista se llama contractiva cuando* $r_i < 1 \ \forall i = 1, \ldots, n$. *Un sistema iterado de funciones (IFS) en un espacio métrico* S *que realiza una lista de razones, es una lista* $(f_1, f_2, \ldots, f_n)$ *de semejanzas de razón* r_i. *Un conjunto compacto no vacío* $K \subseteq S$ *es un conjunto invariante del sistema iterado de funciones* $(f_1, f_2, \ldots, f_n)$ *sii*

$$K = \bigcup_{i=1}^{n} f_i[K].$$

en cuyo caso diremos que K *es autosemejante.*

El teorema de la aplicación contractiva nos dice que si $f: S \to S$ es una aplicación contractiva y S es un espacio métrico completo existe un único punto fijo de f, $x \in S$ que lo podemos encontrar a partir de la sucesión:

$$x_0 = a \in S \tag{3}$$

$$x_n = f(x_{n-1}), \ n \geq 1. \tag{4}$$

Este resultado de análisis, se puede extender a los sistemas iterados de funciones, por medio del siguiente teorema, la demostración de este se puede encontrar en [1].

Teorema 1. *Sea* (S, ρ) *un espacio métrico completo, sea* $(r_1, r_2, \ldots, r_n)$ *una lista de razones contractiva, y sea* $(f_1, f_2, \ldots, f_n)$ *un sistema iterado de funciones en* S *que realiza esta lista de razones. Entonces existe un único conjunto compacto no vacío, invariante para el sistema iterado de funciones.*

La unicidad del punto fijo hace que no importe el conjunto que elijamos para construir una sucesión, el límite es siempre el mismo, es decir, si A_0 es cualquier conjunto compacto no vacío en S, y si $A_{k+1} = \bigcup_{i=1}^{n} f_i[A_k]$, para $k \geq 0$, entonces la sucesión (A_k) converge (en la métrica Hausdorff) al conjunto invariante del sistema iterado de funciones.

El conjunto invariante de un sistema iterado de funciones contractivo es llamado el **atractor** del sistema.

Algunos ejemplos de atractores

Subsecciones

El Conjunto de Cantor

Como primer ejemplo consideremos el clásico Conjunto de Cantor, para su construcción tome el intervalo $C_0 = [0,1]$ y divídalo en tres partes iguales. Al remover el intervalo abierto que corresponde al tercio central, obtenemos $C_1 = \left[0, \frac{1}{3}\right] \cup \left[\frac{2}{3}, 1\right]$. El siguiente conjunto C_2 lo obtenemos removiendo el tercio central a cada intervalo de C_1, de manera que $C_2 = \left[0, \frac{1}{9}\right] \cup \left[\frac{2}{9}, \frac{1}{3}\right] \cup \left[\frac{2}{3}, \frac{7}{9}\right] \cup \left[\frac{8}{9}, 1\right]$ y siguiendo de la misma forma obtenemos una sucesión anidada de conjuntos cerrados $C_0 \supseteq C_1 \supseteq C_2 \supseteq \ldots$ cuyo límite llamamos conjunto de los tercios centrales omitidos o Conjunto de Cantor, y lo denotamos por **C**, Figura 1.

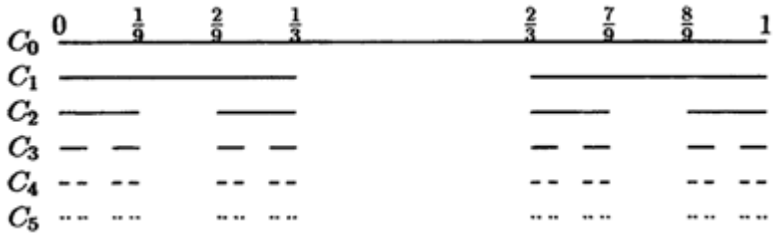

Figura: $\mathbf{C} = \lim_{k\to\infty} C_k = \bigcap_{k=0}^{\infty} C_k$

Si consideramos f_0, $f_1 : \mathbb{R} \to \mathbb{R}$ tal que $f_0(x) = \frac{x}{3}$, $f_1(x) = \frac{x+2}{3}$, y con esto definimos un sistema iterado de funciones (IFS) que realizan la lista contractiva de razones $\left(\frac{1}{3}, \frac{1}{3}\right)$. El atractor de este IFS es el Conjunto de Cantor, [10].

El Dragón de Heighway

Sean f_1, $f_2 : \mathbb{R}^2 \to \mathbb{R}^2$ tal que $f_1((x,y)) = \frac{1}{2}(1 - x - y, x - y)$, $f_2((x,y)) = \frac{1}{2}(x - y, x + y + 1)$. Con esto definimos un sistema iterado de funciones (IFS) que realizan la lista contractiva de razones $\left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$, Figura 2. El atractor de este IFS es el Dragón de Heighway, Figura 3.

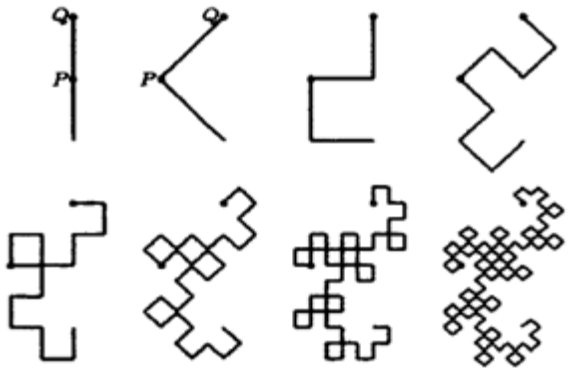

Figura 2: Construcción del Dragón de Heighway

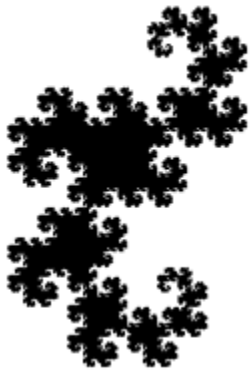

Figura 3: Dragón de Heighway

El triángulo de Sierpinski

Sean f_0, f_1, $f_2 : \mathbb{R}^2 \to \mathbb{R}^2$ tal que $f_0((x,y)) = \frac{1}{2}(x,y)$, $f_1((x,y)) = \frac{1}{2}(x,y) + \frac{1}{2}(1,0)$ y $f_2((x,y)) = \frac{1}{2}(x,y) + \frac{1}{2}(0,1)$. Con esto definimos un sistema iterado de funciones (IFS) que realizan la lista contractiva de razones $\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$. El atractor de este IFS es el triángulo de Sierpinski. En la Figura 4, algunas aproximaciones de S usando la sucesión anterior con $k = 2, 3, 4, 5, 6$ y 7.

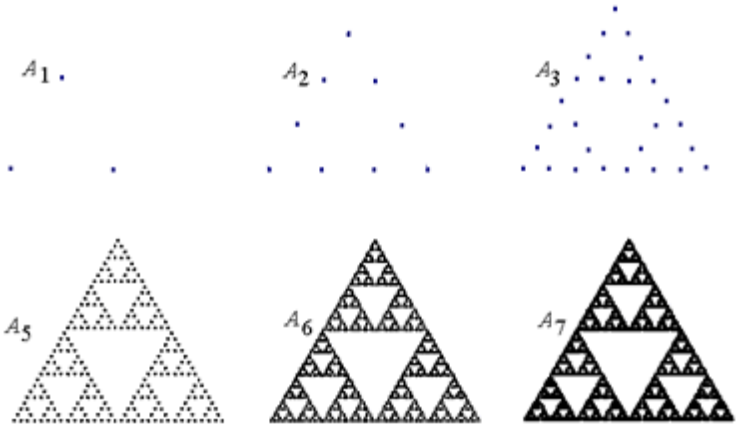

Figura 4: Vértices de S_k

Podemos pensar en este sistema iterado de funciones como una máquina de fotocopiar con tres lentes, la primera lente forma una copia reducida al 50% en la parte izquierda de la figura, la segunda lente forma una copia reducida al 50% en la parte derecha de la figura y la tercera lente forma una copia reducida al 50% en la parte superior centrada de la figura. Ahora tomamos una figura cualquiera, le sacamos la primera copia en nuestra máquina y ésta la volvemos a copiar, hacemos esto varias veces y debemos obtener una figura similar al atractor del sistema. Una exposición muy detallada se puede encontrar en [1], [12].

Esta máquina de copiado de reducción múltiple (MCRM), se puede representar como en la Figura 5, donde se indica además la orientación de la copia. El atractor de este MCRM es el triángulo de Sierpinski, con un cambio de ejes.

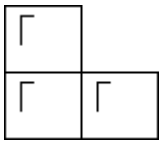

Figura 5: MCRM cuyo atractor
es el Triángulo de Sierpinski.

En total hay 8^3 posibles MCRM, tomando en cuenta las posibles rotaciones, reflexiones, traslaciones y contracciones a escala $1/2$, análogas a la Figura 5.

Como un ejemplo interesante, consideremos como base al número complejo $b = -1 + i$ que provee una representación binaria de todos los números complejos. Al conjunto de todas estas fracciones se le llama usualmente Twindragon, Figura 6 izquierda, pues está formado por dos copias del Dragón de Heighway. Se puede hacer una construcción geométrica de éste, sin embargo, para los efectos de este artículo, este conjunto es el atractor para el sistema iterado de funciones definido por $f_0(x) = b^{-1}x$ y $f_1(x) = b^{-1} + b^{-1}x$.

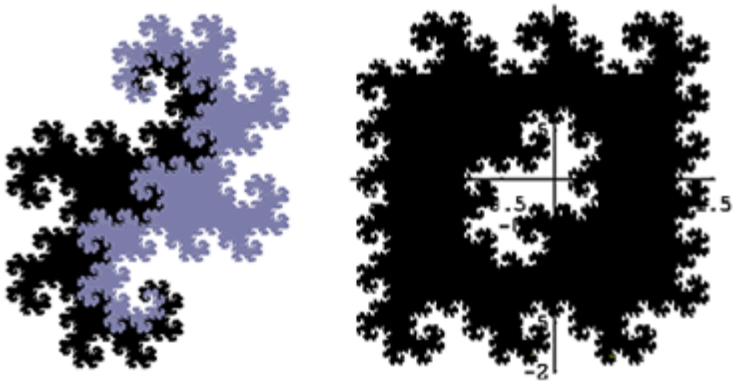

Figura 6: Twindragon

Se puede cubrir el plano con una cantidad numerable de conjuntos idénticos a este conjunto, de la forma $T + w$ donde w es un entero gaussiano[3], y como todos los enteros gaussianos se pueden representar en esta base, se puede obtener un mosaico o ``teselación" fractal del plano, Figura 6 derecha.

Si ahora consideramos como dígitos, números complejos, por ejemplo $D = \{0, 1, \omega, \omega^2\}$ donde $\omega = \frac{1}{2}(-1 + i\sqrt{3})$ y como base $b = -2$ da una representación de todos los números complejos. Al conjunto de todas las fracciones se le llama Fracciones de Eisenstein, Figura 7.

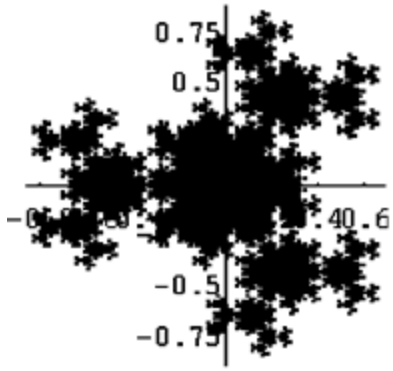

Figura 7: Fracciones de Eisenstein

Este conjunto es el atractor para el IFS definido por $f_0(x) = -\frac{1}{2}x$, $f_1(x) = -\frac{1}{2}(1 + x)$, $f_2(x) = -\frac{1}{2}(\omega + x)$ y $f_3(x) = -\frac{1}{2}(\omega^2 + x)$.

Comprimiendo y descomprimiendo imágenes con IFS's

Sobre cómo se guarda una imagen en el computador, los diversos métodos de compresión[4] de imágenes y algunos ejemplos, puede consultar [1], [11]. Pensando en un sentido inverso, acorde con el tema tratado en este artículo, dado un conjunto (figura, foto, etc.), nos podemos preguntar si será posible determinar un sistema iterado de funciones de manera que el atractor de éste sea el conjunto dado? La compresión de imágenes juega un papel importante en el almacenamiento de imágenes con un menor costo, así como en la rápida transmisión de datos. Claro que al comprimir y luego descomprimir estos datos se puede perder información, esta información perdida muchas veces no se percibe por el ojo humano o es redundante, y los métodos de compresión eficientes son los que logran extraer el ``espíritu" de la imagen, para después de descomprimir, reproducir una imagen muy cercana a la original. En la naturaleza muchos objetos mantienen una autosemejanza

que se puede representar como el atractor de un sistema iterado de funciones, tal es el caso de las nubes, helechos, plantas, árboles, arbustos, etc. La teoría de los fractales tiene aplicaciones en la compresión de imágenes, en el sentido que podemos pensar que un objeto de esta naturaleza se puede transmitir o guardar eficientemente, reproduciéndolo en el momento que se desee.

El siguiente teorema garantiza que el conjunto invariante puede ser una buena aproximación del conjunto inicial si la unión de copias pequeñas están cerca de éste, pues en este caso tendríamos que $D(E,F)$ se acercaría a cero. En este sentido y como consecuencia de este teorema, el siguiente corolario nos dice que un conjunto compacto se puede aproximar tanto como se quiera, bajo la métrica Hausdorff, por un conjunto autosemejante e invariante de un sistema iterado de funciones. La demostración de ellos se puede encontrar en [6, p. 134]. Si el conjunto es más complicado, este se puede ver como la superposición de varios conjuntos invariantes de varios sistemas de funciones.

Teorema 2 (Collage). *Suponga que se tienen* $S_i : \mathbb{R}^n \to \mathbb{R}^n$ *funciones de crecimiento acotado* $\forall i = 1, \ldots, m$, *y suponga que para todo* i *se tiene* $|S_i(x) - S_i(y)| \leq c|x-y|$ *para todo* $x, y \in \mathbb{R}^n$, *con* $c < 1$. *Sea* $E \subset \mathbb{R}^n$ *compacto no vacío, entonces*

$$D(E,F) \leq D\left(E, \bigcup_{i=1}^{m} S_i(E)\right) \frac{1}{1-c} \tag{5}$$

donde F *es el conjunto invariante de los* S_i *y* D *la métrica Hausdorff.*

Corolario 1. *Sea* $E \subset \mathbb{R}^n$ *compacto no vacío. Para todo* $\delta > 0$ *existe un sistema iterado de funciones de crecimiento acotado* $S_1, \ldots, S_n$ *con conjunto invariante* F *tal que* $D(E,F) < \delta$.

A estas MCRM, se les puede permitir algún estiramiento de la copia, así obtenemos gran variedad de sistemas iterados, con atractores muy variados: Helecho de Barnsley, Figura 9 parte (b); árbol, Figura 10. Estos atractores, a pesar de ser tan naturales, se obtienen de la misma forma que la curva de Koch, el triángulo de Sierpinski, el Twindragon, etc. Sobre estos últimos se dan algunos programas al final, en un lenguaje sencillo, para que el lector interesado pueda experimentar.

Tabla 1: Funciones generadoras.

		a	b	c	d	e	f
Figura	f_1	0.849	0.037	−0.037	0.849	0.075	0.183
8	f_2	0.197	−0.226	0.226	0.197	0.4	0.049
	f_3	−0.15	0.283	0.26	0.237	0.575	−0.084
	f_4	0	0	0	0.16	0.5	0
Figura	f_1	0.195	−0.488	0.344	0.443	0.4431	0.2452
10	f_2	0.462	0.414	−0.252	0.361	0.2511	0.5692

	f_3	-0.058	-0.07	0.453	-0.111	0.5976	0.0969
	f_4	-0.035	0.07	-0.469	-0.022	0.4884	0.5069
	f_5	-0.637	0	0	0.501	0.8562	0.2513

Con este esquema de guardar solamente el sistema iterado de funciones y reproducir una aproximación de su atractor en el momento que deseamos, pueden presentarse algunos problemas en cuanto al tiempo de descompresión, aunque la razón de compresión es muy buena, en algunos de estos el tiempo de descompresión no lo es.

Para la descompresión del helecho de Barnsley se usó el sistema de funciones descrito en la Tabla 1. Aplicando este sistema iterado de funciones a x_0 obtenemos 4 imágenes, es decir, $\{f_1(x_0), f_2(x_0), f_3(x_0), f_4(x_0)\}$, y al aplicar de nuevo el sistema obtenemos 16 imágenes, así sucesivamente. En este procedimiento el programa que se usó, que se anexa al final del trabajo, representó 4^7 puntos, pero muchos de ellos están cerca de su imagen, sobre todo los que se obtienen al aplicar f_4 y la resolución que se obtiene no es muy buena, Figura 8.

Figura: Helecho de Barnsley aplicando el IFS descrito en Tabla 1.

Figura 9: (a) Juego del Caos.
(b) Juego del Caos con probabilidades diferentes.

Pensando en este problema, podemos modificar la aplicación del sistema usando el conocido *Juego del Caos*, en donde en cada paso no aplicamos todas las funciones sino solamente una función de manera aleatoria, todas con

la misma probabilidad. Esto quiere decir que si x_0 es el punto inicial, tenemos que $x_n = f_{i_k}(x_{n-1})$ escogiendo i_k con igual probabilidad del conjunto $\{1,2,3,4\}$. Con este procedimiento se obtuvo la Figura 9 parte (a). Aún así, el Juego del Caos descrito aquí se puede modificar tomando diferentes probabilidades p_i para cada función f_i, este proceso recibe el nombre de **sistema iterado recurrente de funciones** (RIFS); una excelente exposición sobre la existencia, unicidad, convergencia y caracterización del atractor se puede encontrar en [3]. La aplicación a compresión de imágenes lo ilustra John Hart en [8], con algunos ejemplos. Un buen índice, para δ pequeño, está dado por

$$p_i = \frac{\max(\delta, |C_i|)}{\sum_{i=1}^{n} \max(\delta, |C_i|)}, \tag{6}$$

donde $|C_i| = \det C_i = \begin{vmatrix} a_i & b_i \\ c_i & d_i \end{vmatrix}$ y se tiene que $\sum p_i = 1$. Con $\delta = 0.01$, con el programa dado al final del trabajo se obtuvo la Figura 9 parte (b), que tiene una mejor resolución con la misma cantidad de puntos que las Figuras 8 y 9 parte (b).

Otros ejemplos de descompresión de un RIFS, aplicado a un punto como imagen inicial, son el árbol de la Figura 10 izquierda, para el sistema de funciones descrito en la Tabla 1 y la rama de la Figura 10 derecha.

Figura 10: Un árbol y una rama como atractores.

Sobre algunos aspectos adicionales a compresión de imágenes, consulte [1], [7], sobre algunos de los métodos de partición de imágenes: Quadtree, partición HV y la partición triangular, [7], [11]. En [7, Apéndice A] se da un programa, en lenguage C, para comprimir y descomprimir imágenes usando el esquema HV. Existen además otros esquemas de compresión de imágenes usando la teoría fractal, [7, Caps. 9,13], [9].

Programas

Estos programas computacionales, se adjuntan con la finalidad de que usted pueda experimentar con ellos, los dos primeros en *Mathematica* y otros tres en *Logo*. Sobre el tema de recursividad en logo puede consultar [2], en

donde se dan varios programas para la curva de Koch, el copo de nieve, triángulo de Sierpinski, curvas dragones, etc. Además, usted puede descargar el programa *Fract.exe* que luego de descomprimirlo, obtiene el *Fractint*, en este puede obtener los conjuntos de Julia, Mandelbrot, etc. y jugar a construir sus propias figuras fractales. Por último, en la dirección de Internet www.iterated.com usted puede navegar por la librería, y conseguir el decodificador de imágenes, de uso libre, y el codificador de imágenes, que requiere de licencia, para los formatos .FIF.

1.
```
A=Table[1,{i,1,2},{j,1,2}];
A[[1,1]]=.849;A[[1,2]]=-.037;A[[2,1]]=.037;A[[2,2]]=.849;
B=Table[1,{i,1,2},{j,1,2}];
B[[1,1]]=0.197;B[[1,2]]=0.226;B[[2,1]]=-0.226;B[[2,2]]=0.197;
Ci=Table[1,{i,1,2},{j,1,2}];
Ci[[1,1]]=-.15;Ci[[1,2]]=.26;Ci[[2,1]]=.283;Ci[[2,2]]=.237;
Da=Table[1,{i,1,2},{j,1,2}];
Da[[1,1]]=0;Da[[1,2]]=0;Da[[2,1]]=0;Da[[2,2]]=.16;
F1[{x_,y_}]={x,y}.A+{.075,.183};
F2[{x_,y_}]={x,y}.B+{0.4,.049};
F3[{x_,y_}]={x,y}.Ci+{.575,-.084};
F4[{x_,y_}]={x,y}.Da+{.5,0};k=6;
X=Table[{0,0},{i,1,4^(k+1)}];
X[[1]]=F1[{0,0.5}];
X[[2]]=F2[{0,0.5}];
X[[3]]=F3[{0,0.5}];
X[[4]]=F4[{0,0.5}];
i=4; j=5;
While[j< 4^(k+1)+1,
X[[j]]=F1[X[[Floor[j/4]+1]]];
X[[j+1]]=F2[X[[Floor[j/4]+1]]];
X[[j+2]]=F3[X[[Floor[j/4]+1]]];
X[[j+3]]=F4[X[[Floor[j/4]+1]]];
j=j+4]; ListPlot[X,
Axes->False,AspectRatio->2/1,Prolog->AbsolutePointSize[1]]
```

2.
```
A=Table[1,{i,1,2},{j,1,2}];
A[[1,1]]=.849;A[[1,2]]=-.037;A[[2,1]]=.037;A[[2,2]]=.849;
B=Table[1,{i,1,2},{j,1,2}];
B[[1,1]]=0.197;B[[1,2]]=0.226;B[[2,1]]=-0.226;B[[2,2]]=0.197;
Ci=Table[1,{i,1,2},{j,1,2}];
Ci[[1,1]]=-.15;Ci[[1,2]]=.26;Ci[[2,1]]=.283;Ci[[2,2]]=.237;
Da=Table[1,{i,1,2},{j,1,2}];
Da[[1,1]]=0;Da[[1,2]]=0;Da[[2,1]]=0;Da[[2,2]]=.16;
F1[{x_,y_}]={x,y}.A+{.075,.183};
F2[{x_,y_}]={x,y}.B+{0.4,.049};
F3[{x_,y_}]={x,y}.Ci+{.575,-.084};
F4[{x_,y_}]={x,y}.Da+{.5,0};
k=13000;        (*Numero de puntos que plotea*)
X=Table[{0,0},{i,1,k}];
X[[1]]=F1[{.5,0.2}];
i=4; j=2;
While[j< k+1, RAN=Random[Integer,{0,100}];
    If       (* Los p_i son las probabilidades en el RIFS*)
[RAN<p_1,
    X[[j]]=F1[X[[j-1]]],
    If[RAN<p_2,
       X[[j]]=F2[X[[j-1]]],
         If[RAN<p_3,
        X[[j]]=F3[X[[j-1]]],
  X[[j]]=F4[X[[j-1]]]]]
];
j=j+1;
If[Mod[j-1,k/4]==0,X[[1]]=F1[{.5,0.2}]]
];
ListPlot[X,
```

```
Axes->False,AspectRatio->2/1,Prolog->
AbsolutePointSize[1]]
```

3. ```
   to Koch :depth :size
       if :depth=0 [forward :size stop]
       Koch :depth-1 :size/3
       left 60
       Koch :depth-1 :size/3
       right 120
       Koch :depth-1 :size/3
       left 60
       Koch :depth-1 :size/3
   end
   ```

4. ```
   to heighway :depth :size :parity
       if :depth=0 [forward :size stop]
       left :parity*45
       heighway :depth-1 :size*:factor 1
       right :parity*90
       heighway :depth-1 :size*:factor (-1)
       left :parity*45
   end
   ```

5. Triangulo de Sierpinski
   ```
   to ens :d :s
   pu
   left 120
   forward :s
   right 120
   pd
   make "p 1
   repeat 6 [ sd :d :s (-:p) right 60 make "p :p*-1  ]
   end
   to SD :d :s :p
       if :d=0*:p [forward :s stop]
       left 60*:p
       SD :d-1 :s/2 (:p)
       right 60*:p
       SD :d-1 :s/2 (-:p)
       right 60*:p
       SD :d-1 :s/2 (:p)
       left 60*:p
   end
   ```

Bibliografía

1

Alfaro, M.; Murillo, M.& Soto A. *Tesis de Licenciatura*, Universidad de Costa Rica, 1997.

2

Arce, Carlos, *Elementos de Matematica con LogoWriter*, Editorial UNED, págs. 55-67, 1994.

3

Barnsley, Michael F.; Elton John H. & Hardin Douglas P., *Recurrent Iterated Function Systems*, Constructive Approximation, Vol. 5, págs. 3-31, 1989.

4

Brenes, Gerardo; Corrales Víctor, *Fractales*, Revista Tecnología en Marcha, ITCR, Vol 11, Num. 1, 1991.

5

Edgar, Gerald A., *Measure, Topology, and Fractal Geometry*, Springer-Verlag, New York, 1990.

6

Falconer, Kenneth, *Fractal Geometry*, John Wiley & Sons, England, 1990.

7

Fisher, Yuval, *Fractal Image Compression*, Springer-Verlag, New York, 1994.

8

Hart, John C., *Fractal Image Compression and Recurrent Iterated Function Systems*, IEEE Computer Graphics, Vol. 16, Num. 4, 1996.

9

Jacquin, Arnaud E., *Image Coding Based on a Fractal Theory of Iterated Contractive Image Transformations*, IEEE Transactions on Image Processing, Vol. 1, Num. 1, págs. 18-30, 1992.

10

Murillo T. Manuel & Soto A. Alberto, *Sobre intersecciones, tamaño y prisioneros de los Conjuntos de Cantor*, Revista del Profesor de Matemática, año 3, Num. 1, pág. 21-30, Chile, 1998.

11

Murillo T. Manuel, *Sobre la compresión de imágenes y atractores de sistemas*, Revista Tecnología en Marcha, ITCR, Vol 15, Num. 1, 2002.

12

Soto A. Alberto. *El Fascinante Triángulo de Sierpinski*, Memorias del V ECADIM, Liberia, Costa Rica. 1997.