Apéndice E

El sistema de Lindenmayer

En este tema hemos utilizado materiales procedentes de:

- La Wikipedia: `http://es.wikipedia.org/wiki/Sistema-L`, en sus ediciones española, inglesa y francesa.
- El libro "**The Algorithmic Beauty of Plants**" escrito por Przemyslaw Prusinkiewicz y Aristid Lindenmayer.

Esta lección va a tratar de la noción de **Sistema de Lindenmayer** o **Sistema–L** inventado en 1968 por el biólogo húngaro Aristid Lindenmayer. Un Sistema-L es un conjunto de reglas y símbolos que modelan el proceso de crecimiento de seres vivos como las plantas o las células. El concepto central de los Sistemas–L es la noción de re-escritura. La re-escritura es una técnica para construir objetos complejos por reemplazo a partir de un objeto inicial sencillo utilizando reglas de re-escritura.

Para ello, las células se modelizan mediante símbolos. En cada generación, las células se dividen, es decir, un símbolo se sustituye por uno o varios otros símbolos que forman una palabra.

E.1. Definición formal

Un Sistema–L es una gramática formal que comprende:

1. Un alfabeto V : conjunto de variables del Sistema–L. $V*$ es el conjunto de "palabras" que podemos generar con cualquiera de los símbolos de V, y $V+$ el conjunto de palabras con al menos un símbolo.
2. Un conjunto de valores constantes S. Algunos de estos símbolos son comunes a todos los Sistemas–L, en particular cuando los utilizamos con la tortuga).
3. Un axioma de partida ω, elegido de $V+$, es el estado inicial.

4. Un conjunto de reglas o producciones, denominado P, que definen la forma en la que las variables pueden ser reemplazadas por combinaciones de constantes y otras variables.

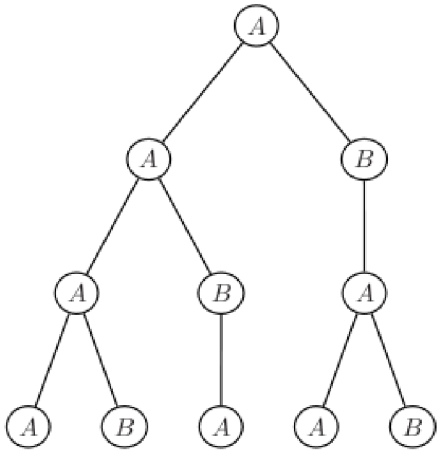
   Una producción está formada por dos cadenas. el predecesor y el sucesor.

Un Sistema–L definido así, es una tupla $\{V, S, \omega, P\}$.

Consideremos el Sistema–L siguiente:

- Alfabeto: $V = \{A, B\}$
- Constantes: $S = \{\emptyset\}$
- Axioma inicial: $\omega = A$
- Reglas : $\boxed{\begin{array}{l} A \rightarrow AB \\ B \rightarrow A \end{array}}$

Las dos reglas de producción dadas son las reglas de re-escritura del sistema. En cada etapa, A es sustituida por AB, y B es sustituido por A. Estas son las primeras iteraciones de este sistema de Lindemayer:



- Iteración 1: A
- Iteración 2: AB
- Iteración 3: ABA
- Iteración 4: $ABAAB$

Bien, bien pero ... ¿y en qué se concreta esto? Pasemos a la siguiente sección.

E.2. Interpretación por la tortuga

Este primer ejemplo ayuda a entender la noción de sistema de Lindenmayer y, posiblemente, cómo vamos a utilizarla de manera eficaz con la tortuga.

Aquí es donde se pone interesante: Cada palabra así construida no tiene ningún significado especial. Vamos a definir para cada letra de la secuencia, una acción (comando) que ejecutará la tortuga y generará dibujos en 2D o 3D.

E.2.1. Simbología

- F : Avanzar un paso (unitario) ($\in V$)
- $+$: Girar un cierto ángulo α a la izquierda ($\in S$).
- $-$: Girar un cierto ángulo α a la derecha ($\in S$).
- & : Girar un cierto ángulo α hacia abajo ($\in S$).
- ^: Girar un cierto ángulo α hacia arriba ($\in S$).
- \: Girar sobre sí mismo hacia la izquierda un ángulo α ($\in S$).
- / : Girar sobre sí mismo hacia la derecha un ángulo α ($\in S$).
- | : Efectuar una media vuelta. Con xLogo: `giraderecha 180`

Por ejemplo, con $\alpha = 90°$ y un desplazamiento unitario de 10 pasos de tortuga, obtennemos:

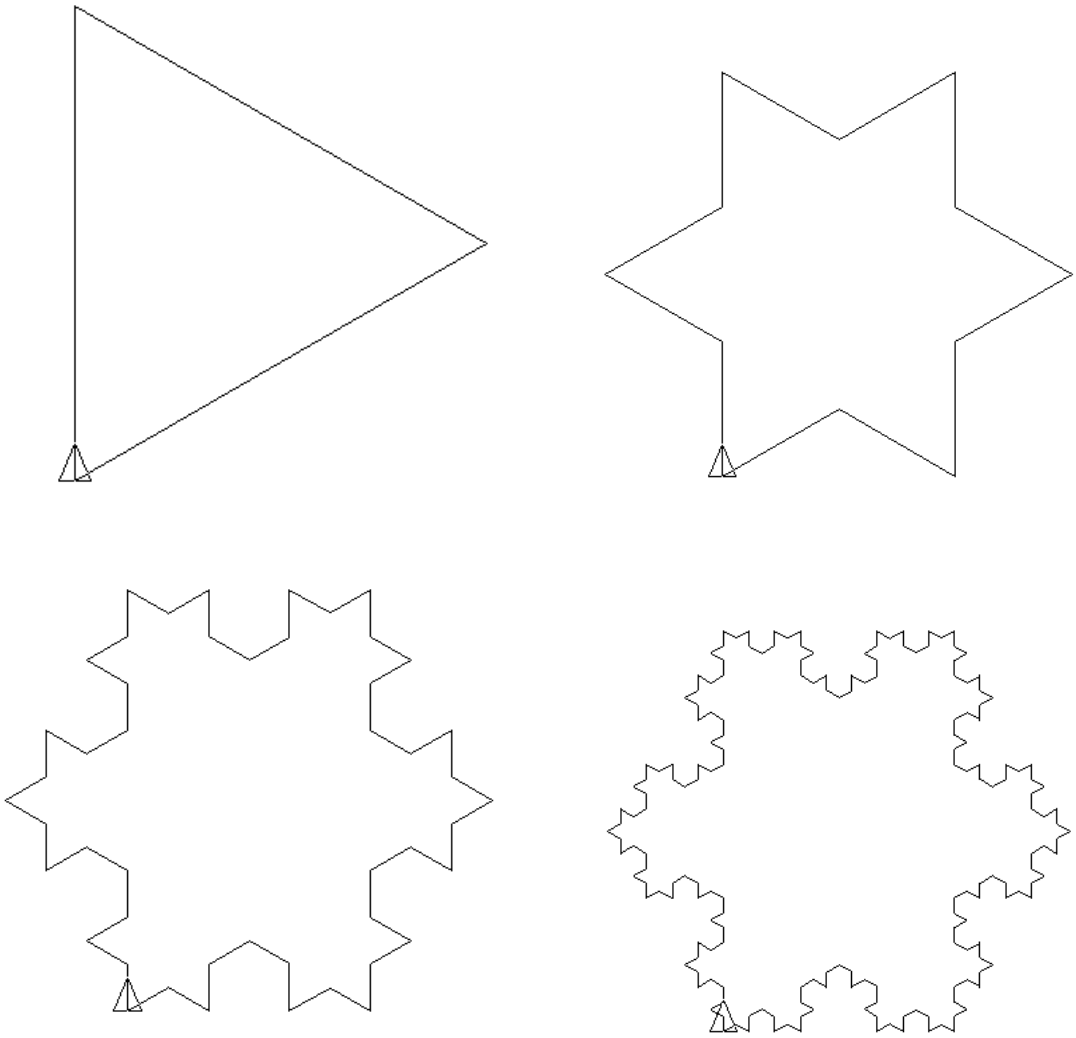
Símbolo	F	$+$	$-$	&	^	\	/	|
Comando xLogo	`av 10`	`gi 90`	`gd 90`	`sn 90`	`bn 90`	`bi 90`	`bd 90`	`gd 180`

E.2.2. Copo de Koch

Consideremos el sistema–L:

- Estado inicial: $F--F--F--$
- Regla de producción: $F \to F+F--F+F$
- Ángulo $\alpha = 60°$, el paso unitario se divide por 3 en cada iteración.

Primeras iteraciones:



cuyo programa en xLogo es:

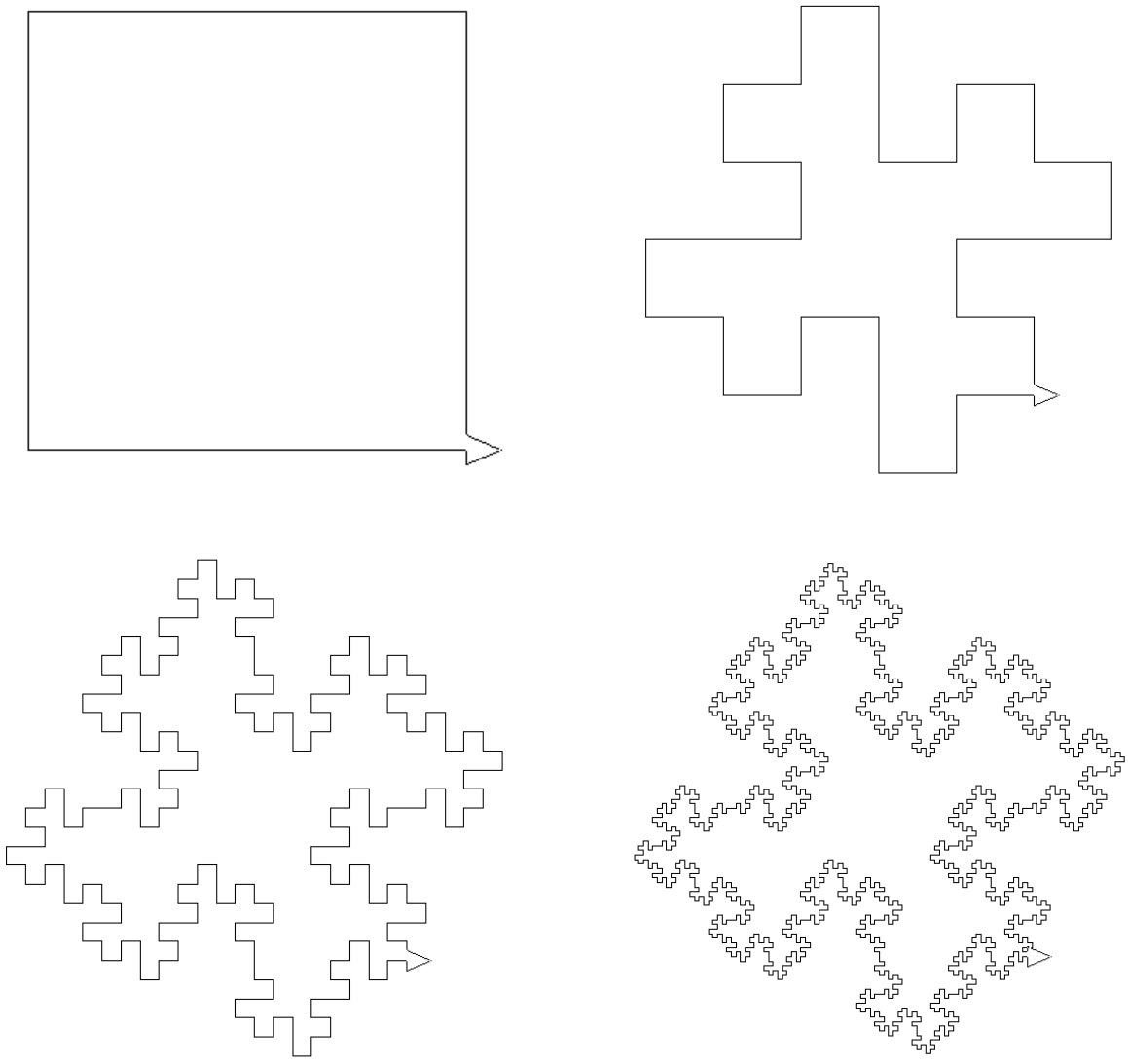
```
para coponieve :p
  borrapantalla
  haz "unidad 300/potencia 3 :p-1
  repite 3 [f :p-1 giraderecha 120]
fin

para f :p
  si :p=0 [avanza :unidad alto]
  f :p-1 giraizquierda 60
  f :p-1 giraderecha 120
  f :p-1 giraizquierda 60
  f :p-1
fin
```

E.2.3. Curva de Koch de orden 2

Fijémonos ahora en el sistema–L siguiente:

- Estado inicial: $F - F - F - F$
- Regla de producción: $F \rightarrow F - F + F + FF - F - F + F$

Las primeras representaciones utilizando $\alpha = 90$ y ajustando el paso unitario para que la figura tenga un tamaño constante:



Ahora es muy fácil crear el programa Logo para generar esas figuras:

```
# p indica la iteracion
```

```
para koch :p
  # entre cada iteracion, la distancia unitaria se divide entre 4
  # asi, la figura final tendra unas dimensiones maximas de 600x600
  haz "unidad 300/potencia 4 :p-1
  repite 3 [ f :p-1 giraizquierda 90]
  f :p-1
fin

# la cadena de re-escritura
para f :p
  si :p=0 [avanza :unidad alto]
  f :p-1 giraizquierda 90
  f :p-1 giraderecha 90
  f :p-1 giraderecha 90
  f :p-1 f :p-1 giraizquierda 90
  f :p-1 giraizquierda 90
  f :p-1 giraderecha 90 f :p-1
fin
```

E.2.4. Curva del dragón

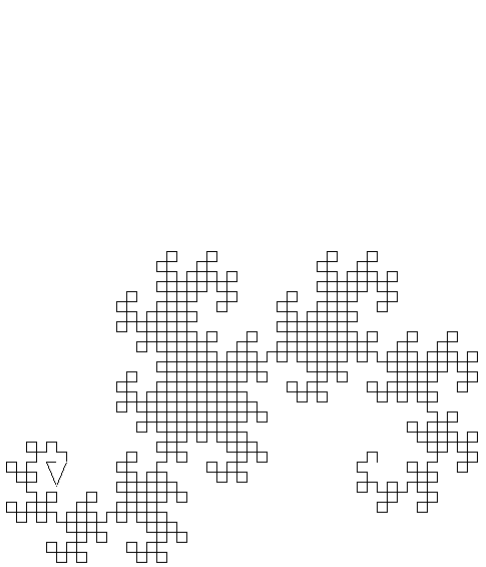
Terminamos esta serie de ejemplos con la **curva dragón**, cuyas condiciones son:

- Estado inicial: F
- Regla de producción: $\boxed{\begin{array}{l} A \rightarrow A + B+ \\ B \rightarrow -A - B \end{array}}$

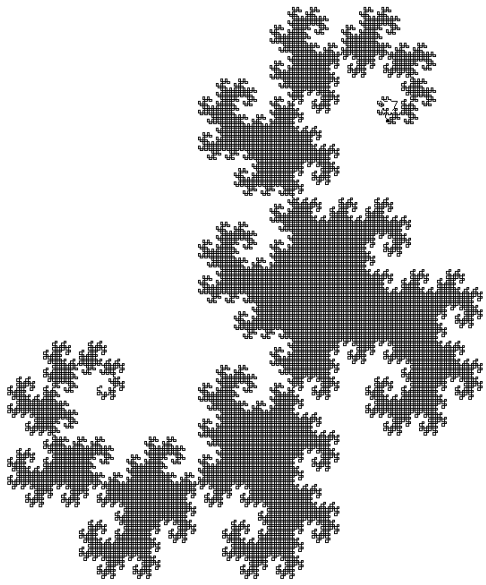
El programa resulta:

```
para dragon :p
  haz "unidad 300/8/ :p
  a :p
fin

para a :p
  si :p=0 [avanza :unidad alto]
  a :p-1 giraizquierda 90 b :p-1 giraizquierda 90
fin

para b :p
  si :p=0 [avanza :unidad alto]
  giraderecha 90 a :p-1 giraderecha 90 b :p-1
fin
```

y los resultados son:



dragon 10



dragon 15

E.2.5. Curva de Hilbert en 3D

El ejemplo siguiente generará la curva de Hilbert en el espacio. Esta es una curva que presenta la propiedad de reemplazar perfectamente un cubo al aumentar el número de iteraciones.

El sistema–L asociado es:

- Estado inicial: A
- Ángulo $\alpha = 90°$, la longitud unitaria se divide entre dos en cada iteración
- Regla de produción:

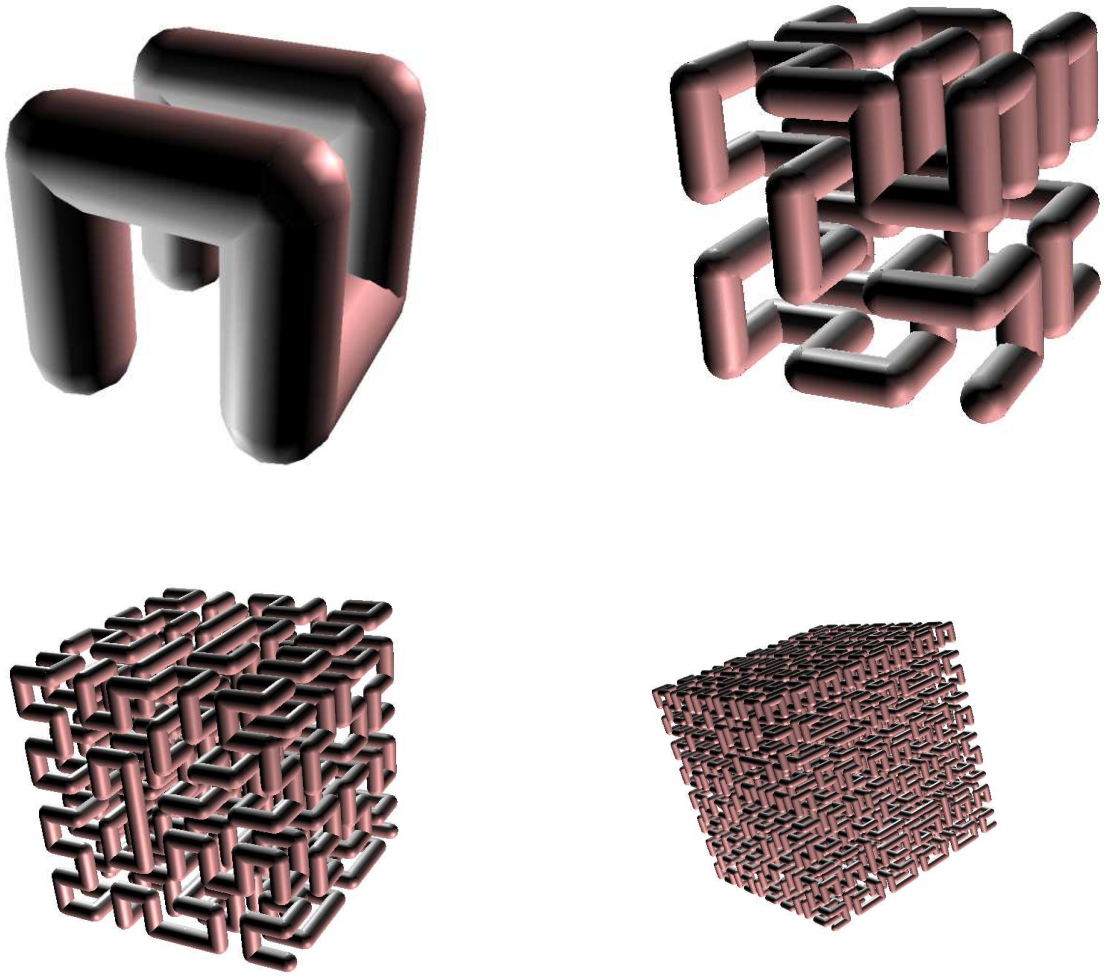
$$\begin{array}{l} A \rightarrow B - F + CFC + F - D\&F\hat{}D - F + \&\&CFC + F + B// \\ B \rightarrow A\&F\hat{}CFB\hat{}F\hat{}D\hat{}\hat{} - F - D\hat{}|F\hat{}B|FC\hat{}F\hat{}A// \\ C \rightarrow |D\hat{}|F\hat{}B - F + C\hat{}F\hat{}A\&\&FA\&F\hat{}C + F + B\hat{}F\hat{}D// \\ D \rightarrow |CFB - F + B|FA\&F\hat{}A\&\&FB - F + B|FC// \end{array}$$

```
para hilbert :p
  borrapantalla perspectiva
  haz "unidad 400/potencia 2 :p
  empiezalinea
    pongrosor :unidad/2 a :p
  finlinea
  vistapoligono
fin
```

```
para a :p
  si :p=0 [alto]
  b :p-1 giraderecha 90 avanza :unidad giraizquierda 90
  c :p-1 avanza :unidad
  c :p-1 giraizquierda 90 avanza :unidad giraderecha 90
  d :p-1 cabeceaabajo 90 avanza :unidad cabeceaarriba 90
  d :p-1 giraderecha 90 avanza :unidad giraizquierda 90 cabeceaabajo 180
  c :p-1 avanza :unidad
  c :p-1 giraizquierda 90 avanza :unidad giraizquierda 90
  b :p-1 balanceaderecha 180
fin

para b :p
  si :p=0 [alto]
  a :p-1 cabeceaabajo 90 avanza :unidad cabeceaarriba 90
  c :p-1 avanza :unidad
  b :p-1 cabeceaarriba 90 avanza :unidad cabeceaarriba 90
  d :p-1 cabeceaarriba 180 giraderecha 90 avanza :unidad giraderecha 90
  d :p-1 cabeceaarriba 90 giraderecha 180 avanza :unidad cabeceaarriba 90
  b :p-1 giraderecha 180 avanza :unidad
  c :p-1 cabeceaarriba 90 avanza :unidad cabeceaarriba 90
  a :p-1 balanceaderecha 180
fin

para c :p
  si :p=0 [alto]
  giraderecha 180
  d :p-1 cabeceaarriba 90 giraderecha 180 avanza :unidad cabeceaarriba 90
  b :p-1 giraderecha 90 avanza :unidad giraizquierda 90
  c :p-1 cabeceaarriba 90 avanza :unidad cabeceaarriba 90
  a :p-1 cabeceaabajo 180 avanza :unidad
  a :p-1 cabeceaabajo 90 avanza :unidad cabeceaarriba 90
  c :p-1 giraizquierda 90 avanza :unidad giraizquierda 90
  b :p-1 cabeceaarriba 90 avanza :unidad cabeceaarriba 90
  d :p-1 balanceaderecha 180
fin

para d :p
  si :p=0 [alto]
  giraderecha 180
  c :p-1 avanza :unidad
  b :p-1 giraderecha 90 avanza :unidad giraizquierda 90
```

```
  b :p-1 giraderecha 180 avanza :unidad
  a :p-1 cabeceaabajo 90 avanza :unidad cabeceaarriba 90
  a :p-1 cabeceaabajo 180 avanza :unidad
  b :p-1 giraderecha 90 avanza :unidad giraizquierda 90
  b :p-1 giraderecha 180 avanza :unidad
  c :p-1 balanceaderecha 180
fin
```

En las primeras iteraciones obtenemos:



Hermoso, ¿verdad?