

Geometria Computacional da Natureza*

1.	Introdução	4
2.	A Geometria Fractal da Natureza	5
2.1.	Auto-Similaridade.....	7
2.2.	Alguns Fractais Pioneiros	9
2.3.	Dimensão e Dimensão Fractal	18
3.	Autômatos Celulares e Geometria Fractal	25
3.1.	Definição Formal.....	26
4.	Sistemas L (L-Systems)	30
4.1.	Conceitos sobre Sistemas de Produção e Gramáticas	30
4.2.	Sistemas DOL.....	32
4.3.	Gráfico Tartaruga (Turtle Graphics).....	35
4.4.	Modelos de Arquiteturas de Plantas.....	41
4.5.	Escopo dos Sistemas-L	43

* Material baseado em notas de aula gentilmente cedidas pelo Prof. Leandro Nunes de Castro (Mackenzie/SP)

5.	Sistemas de Funções Iterativas	48
5.1.	Conceitos básicos.....	48
5.2.	Mapeamentos e aplicações	51
5.3.	Auto-Similaridade e Auto-Afinidade Revisitadas	58
6.	Movimento Browniano	59
6.1.	Fractais Aleatórios na Natureza e Movimento Browniano.....	60
6.2.	Movimento Browniano Fracionário	68
6.3.	Escopo do MBF	73
7.	Sistemas de Partículas	75
7.1.	Conceitos Básicos	76
7.2.	Modelo Básico	78
7.3.	Simulando Fogos de Artifício	80
7.4.	Escopo dos Sistemas de Partículas	82
8.	Da Geometria Natural à Geometria Fractal.....	82
9.	Escopo da Geometria Fractal.....	84
10.	Ask nature.....	86

*“Why is geometry often described as ‘cold’ and ‘dry’? One reason lies in its inability to describe the shape of a cloud, a mountain, a coastline, or a tree. Clouds are not spheres, mountains are not cones, coastlines are not circles, and bark is not smooth, nor does lightning travel in a straight line. ... The existence of these patterns challenges us to study those forms that Euclid leaves aside as being ‘formless’, **to investigate the morphology of the ‘amorphous’.**” (Mandelbrot, 1983, p.1)*

*“In the past, mathematics has been concerned largely with sets and functions to which the methods of classical calculus can be applied. Sets or functions that are not sufficiently smooth or regular have tended to be ignored as ‘pathological’ and not worthy of study. Certainly, they were regarded as individual curiosities and only rarely were thought of as a class to which a general theory might be applicable. In recent years this attitude has changed. It has been realized that a great deal can be said, and is worth saying, about the **mathematics of non-smooth objects**. Moreover, irregular sets provide a much better representation of many natural phenomena than do the figures of classical geometry. Fractal geometry provides a general framework for the study of such irregular sets. (Falconer, 2003, p. xvii)*

1. Introdução

- O grande avanço da computação gráfica tornou possível a visualização de diversos modelos e estruturas de fenômenos naturais com grande realismo.
- As imagens, animações e sistemas resultantes são úteis como ferramentas científicas, de pesquisa e educacionais em diversas áreas.
- As aplicações incluem o projeto e criação de paisagens naturais, plantas, predição de fenômenos naturais (colheita), estudo de processos de desenvolvimento e crescimento, modelagem e síntese de diversos padrões e formas naturais.
- Existem diversas técnicas de modelagem e síntese de padrões naturais. Por exemplo: modelos de reação-difusão (*reaction-diffusion models*), modelos de agregação de difusão limitados (*diffusion-limited aggregation*), autômatos celulares, sistemas de Lindenmeyer (*L-systems*), sistemas de funções iterativas (*iterated function systems*), sistemas de partículas (*particle systems*).

- Uma etapa marcante no processo de modelagem e síntese de padrões naturais foi o reconhecimento de que a natureza é *fractal*, juntamente com o desenvolvimento da *geometria fractal*.
- De forma simplificada, a geometria fractal pode ser vista como a geometria da natureza, com toda a sua irregularidade e estruturas complexas e fragmentadas.

2. A Geometria Fractal da Natureza

- A geometria euclidiana descreve formas ideais, como pontos, círculos, retas, esferas, quadrados, cubos, etc.
- Entretanto, estas formas euclidianas são geralmente encontradas apenas em objetos produzidos por seres humanos.
- Na natureza predominam formas não-suaves e não-uniformes e muitos padrões são irregulares e fragmentados.

- Qual é a forma de um floco de neve? E de uma montanha? E de uma encosta, nuvem, árvore e diversas outras formas da natureza?
- O termo *fractal* foi cunhado por MANDELBROT (1983) para identificar uma família de formas que apresenta padrões irregulares e fragmentados.
- A *geometria fractal* é a geometria das formas irregulares encontradas na natureza.
 - Genericamente, os fractais são caracterizados por detalhes infinitos, comprimento infinito, *auto-similaridade*, *dimensões fractais*, e a ausência de suavidade ou derivadas.
- Portanto os fractais são irregulares; eles possuem o mesmo grau de irregularidade em todas as escalas.
- Os fractais parecem os mesmos quando observados à distância ou de muito perto (auto-similaridade).
- Alguns exemplos de fractais na natureza:

- Arbustos, costas marítimas, montanhas, couve-flor, brócolis, pulmões, cérebro, rins, nuvens etc.

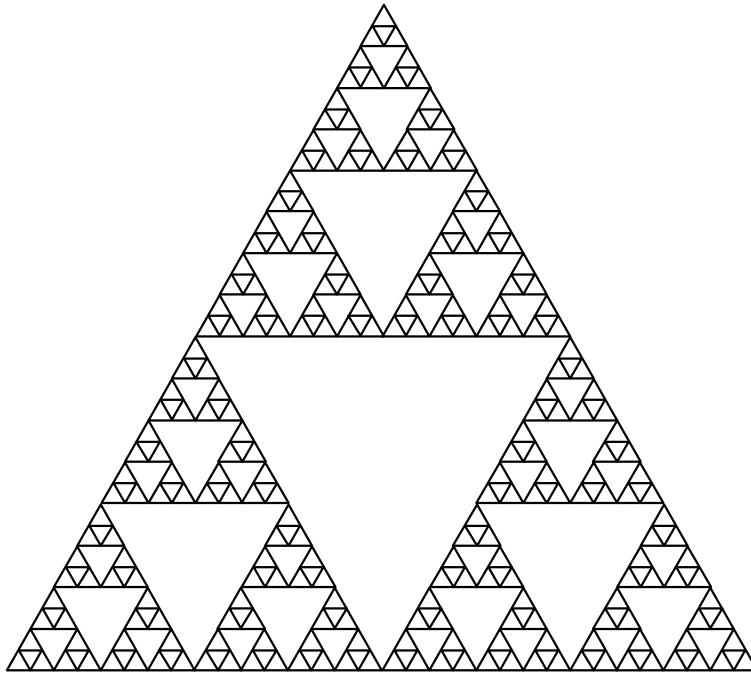


2.1. Auto-Similaridade

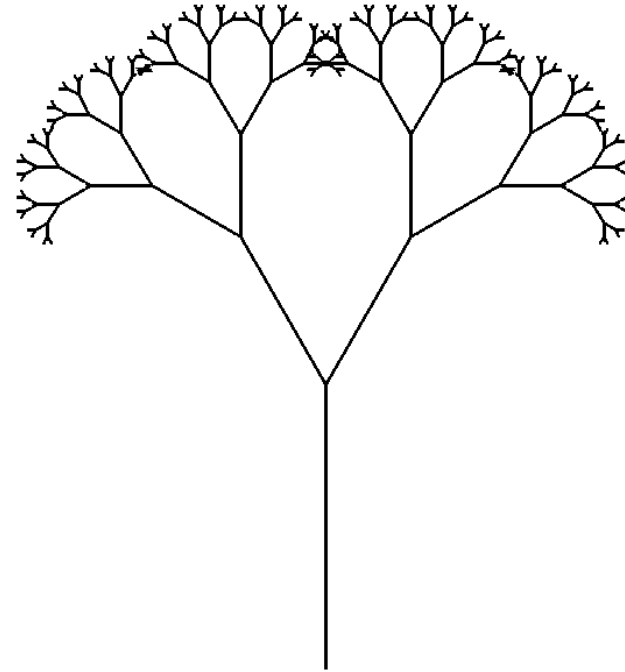
- Uma interpretação intuitiva de auto-similaridade:



- Auto-similaridade estatística: quando as cópias (partes) menores são pequenas variações de toda a estrutura.
- Auto-similaridade estrita $\times$ Auto-similaridade estatística (Auto-afinidade):



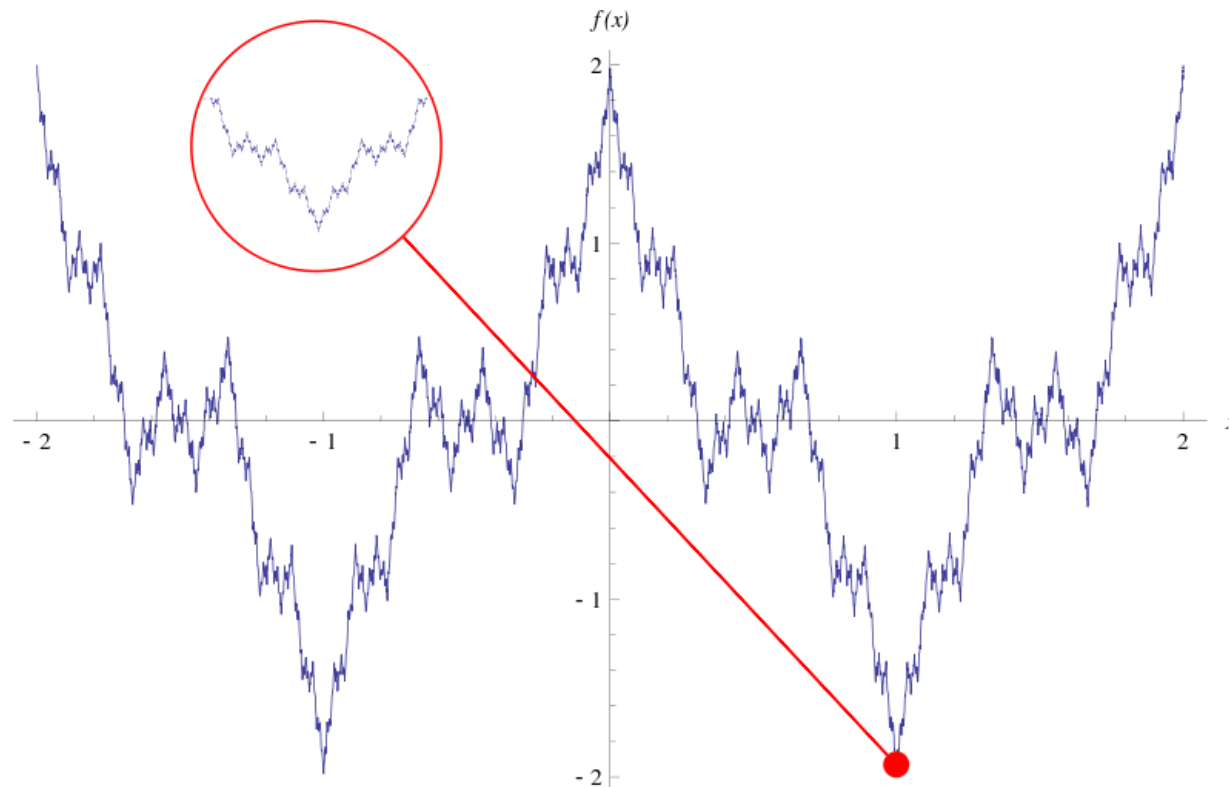
Auto-similaridade estrita



Auto-afinidade

2.2. Alguns Fractais Pioneiros

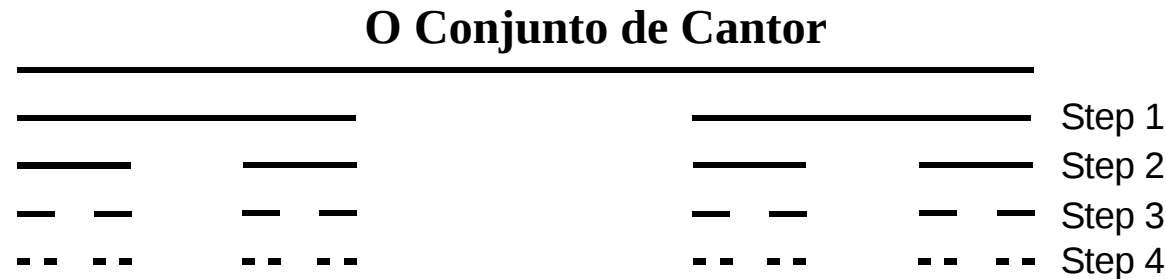
- O primeiro fractal foi descoberto por K. Weierstrass em 1861. Ele descobriu uma função contínua que não é diferenciável em ponto algum, ou seja, uma curva constituída somente por “cantos”.



Poucas iterações da função de Weierstrass e auto-similaridade (zoom)

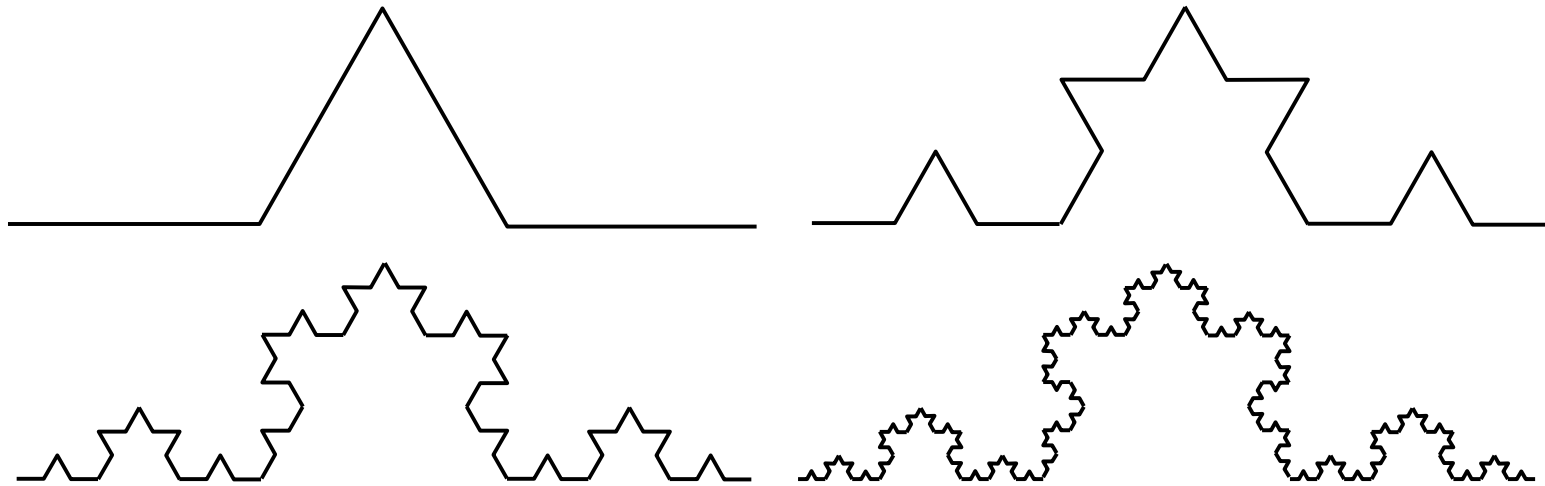
- Outros fractais pioneiros foram descobertos por G. Cantor, H. von Koch, W. Sierpinski e outros.

- Estes fractais foram considerados “monstros matemáticos” devido a algumas características não-intuitivas que eles apresentavam.



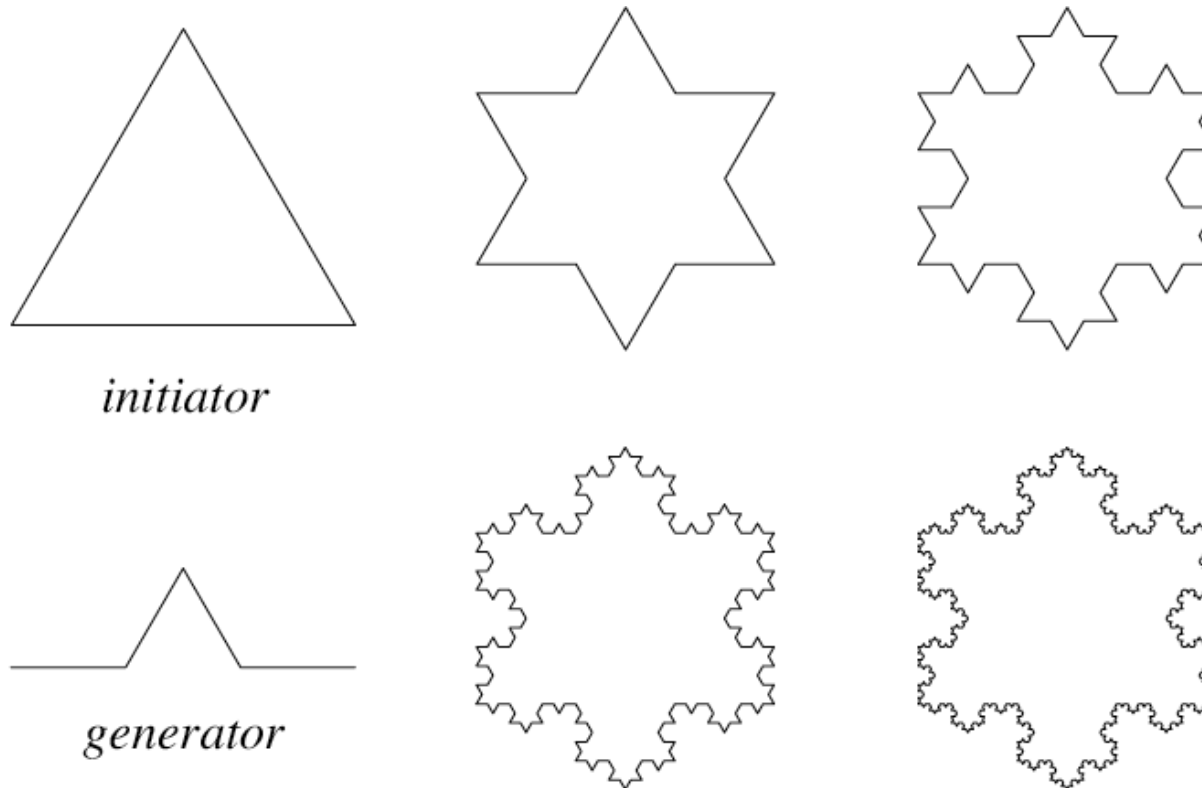
- Propriedades interessantes:
 - Não possui comprimento algum ou interior
 - Cada uma de suas partes é constituída basicamente de buracos
 - É totalmente formado por pontos desconexos
 - Contém a mesma quantidade de pontos que a curva da qual ele é derivado
 - Cada um de seus pontos é um ponto limite, ou seja, existe uma quantidade infinita de outros pontos do conjunto na sua vizinhança.

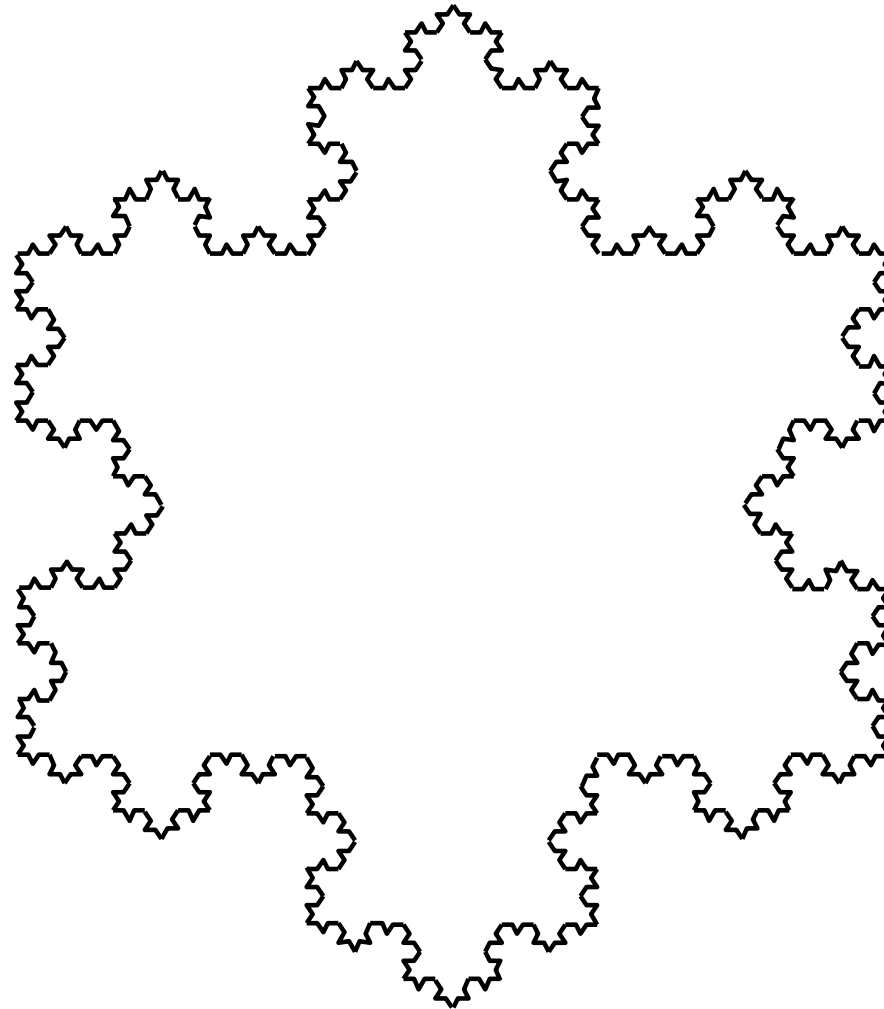
A Curva de Koch (Floco de neve)



- Propriedades interessantes:
 - No limite, a curva de Koch não possui segmento algum de reta; a curva é inteiramente constituída por cantos.
 - Portanto a curva não apresenta derivada (tangente) em ponto algum.
 - Embora ela se inicie a partir de uma reta de comprimento L , seu comprimento é infinito.

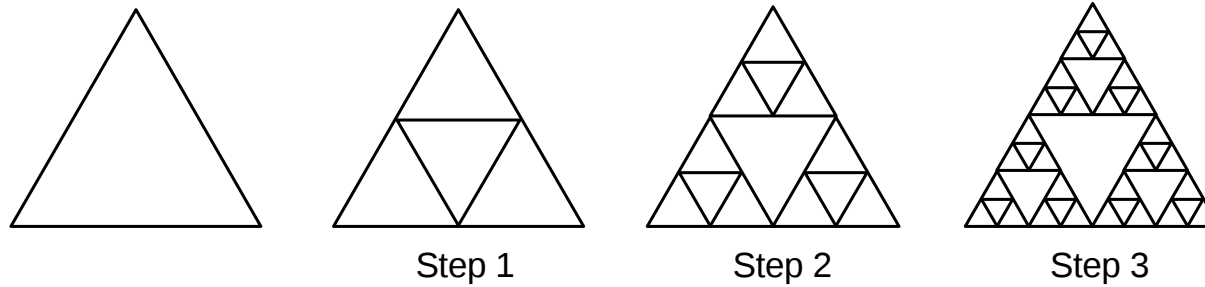
- No passo t a curva possui 4^t segmentos, cada qual com comprimento $1/3^t$.
Portanto, o comprimento total da curva é $(4/3)^t$.
- Note que uma curva de comprimento infinito pode ser colocada em uma área finita.



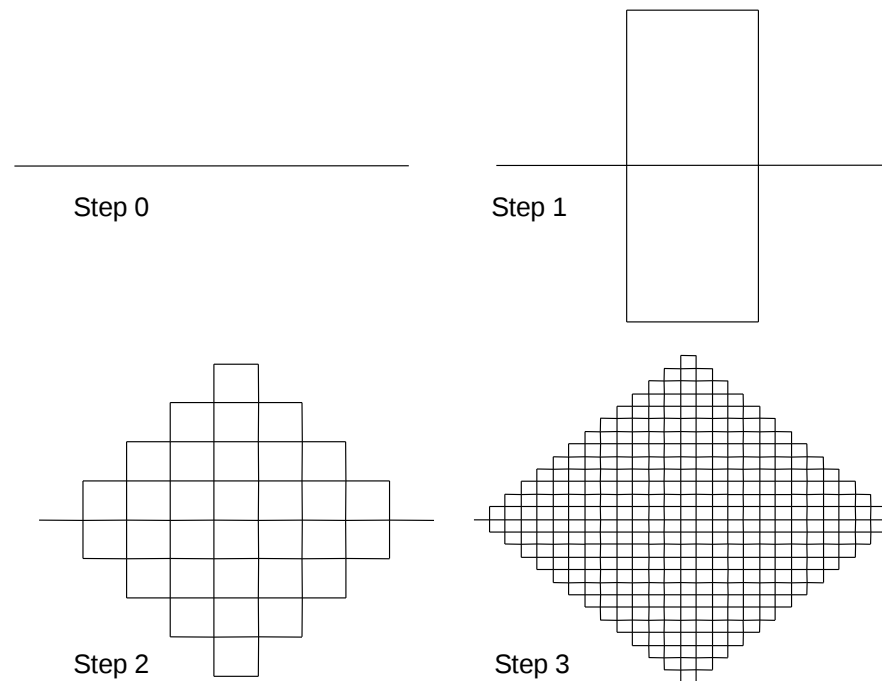


Poucas iterações da curva de Koch, tendo um triângulo como iniciador

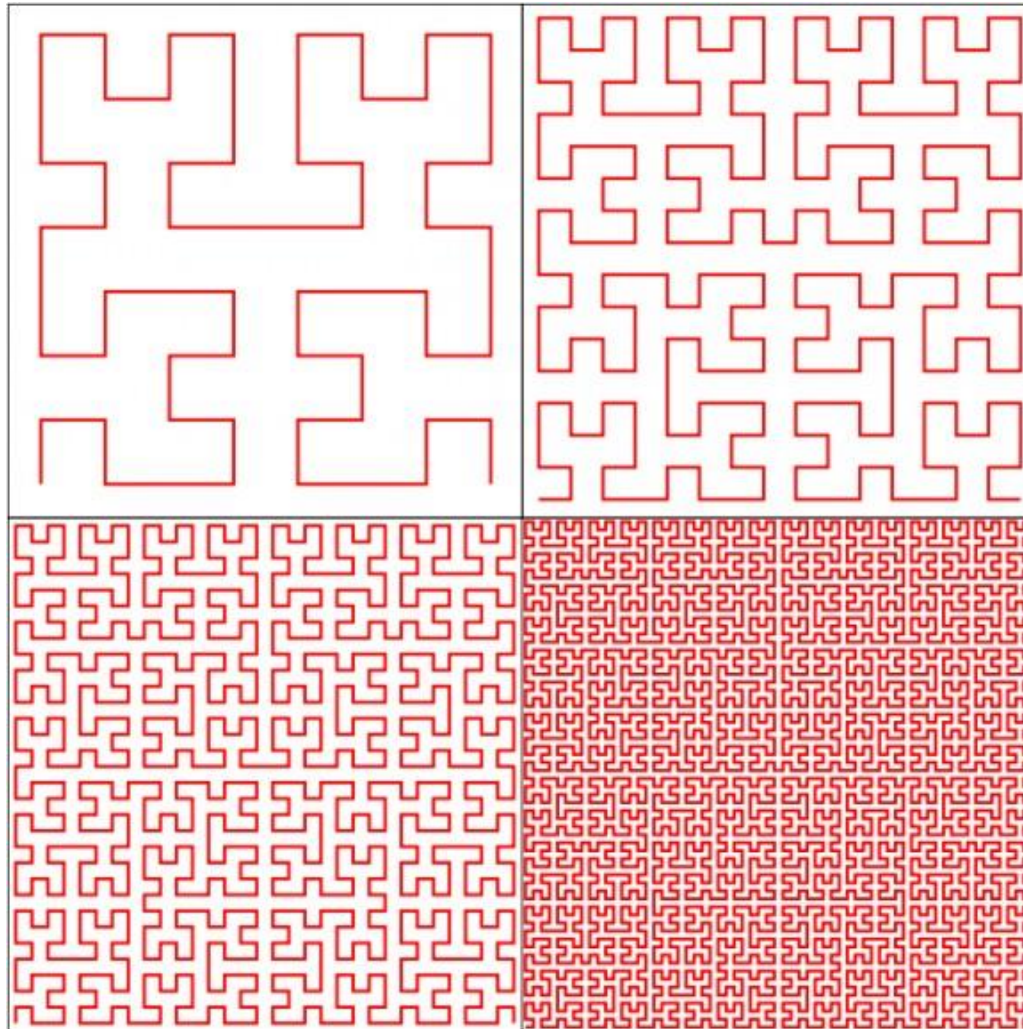
- Outro fractal pioneiro: o triângulo de Sierpinski.



- Curvas que, no limite, preenchem o espaço 2D: curva de Peano.



- Em 1890, Giuseppe Peano (1858–1932) descobriu uma curva de auto-interseção que passa por todos os pontos de um quadrado unitário. Seu propósito era construir um mapeamento contínuo do intervalo unitário para o quadrado unitário. A busca de Peano foi motivada pelo resultado contra-intuitivo de Georg Cantor: o número infinito de pontos em um intervalo unitário é de mesma cardinalidade que o número infinito de pontos em qualquer variedade de dimensão finita, como o quadrado unitário. O problema abordado e resolvido por Peano foi mostrar se esse mapeamento pode ser contínuo, ou seja, se uma curva pode preencher um espaço.
- Um ano depois, David Hilbert (1862-1943) publicou no mesmo periódico uma variação da proposta de Peano. O trabalho de Hilbert foi o primeiro a incluir uma imagem permitindo a visualização da técnica de construção. A fórmula da curva analítica de Hilbert, no entanto, é mais complicada do que a de Peano.



Poucas iterações (3ª a 6ª iteração) da curva de Hilbert

2.3. Dimensão e Dimensão Fractal

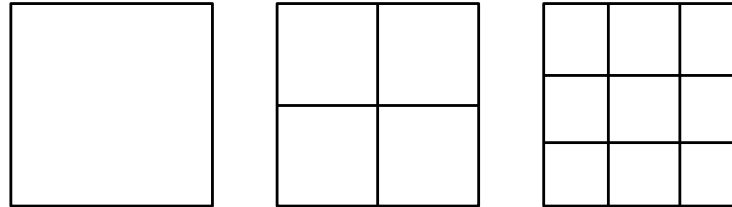
- Estruturas como a curva de Peano desafiavam o conceito de dimensão, pois curvas são sabidas ter dimensão 1, mas para preencher o espaço elas deveriam ser percebidas como tendo dimensão 2.
- Pontos possuem dimensão 0, linhas e curvas possuem dimensão 1, planos e superfícies possuem dimensão 2, sólidos possuem dimensão 3.
- De forma simplificada, um conjunto possui dimensão d se d variáveis independentes (coordenadas) são necessárias para descrever a vizinhança de cada ponto. Esta noção de dimensão é denominada de *dimensão topológica*.
- No final do século 19, alguns matemáticos perceberam que um bom entendimento da irregularidade ou fragmentação de algumas formas não pode ser alcançado definindo-se dimensão como sendo um número de coordenadas.
- Por exemplo, a curva de Koch possui dimensão topológica 1, mas não pode ser considerada uma curva sob a perspectiva da geometria euclidiana: o comprimento

entre quaisquer dois pontos da curva é infinito; nenhuma de suas partes é uma linha ou um plano. De certa forma, é possível dizer que ela é muito grande para ser unidimensional e, ao mesmo tempo, muito pequena para ser bidimensional. Logo, sua dimensão deve ser um número entre 1 e 2.

- A dificuldade em se definir a dimensão de objetos como a curva de Koch, o conjunto de Cantor e o triângulo de Sierpinski não é apenas um problema dos fractais exemplificados aqui.
- Um fenômeno similar foi identificado pelo meteorologista inglês L. Richardson em 1961 em sua tentativa de medir o comprimento de várias costas marítimas, incluindo a costa da Inglaterra.
- Ele percebeu que o comprimento aparente da costa parecia crescer sempre que o comprimento do instrumento de medida era reduzido.
- Isso ocorria, pois quanto menor o comprimento do medidor maior a amplificação dos detalhes.

- Richardson concluiu que o comprimento da costa não é bem definido, e ele também propôs uma lei empírica relacionando este aumento no comprimento da unidade de medida com a quantidade de detalhes percebidos.
- Ele notou que, quando o logaritmo do comprimento do instrumento de medida era plotado em função do logaritmo do comprimento total da costa, os pontos tendiam a se distribuir em torno de uma linha reta.
- A inclinação da reta resultante indicava, de alguma forma, o grau de dobramento ou fragmentação da costa.
- MANDELBROT (1983) encontrou o trabalho de Richardson e verificou que os fractais poderiam ser classificados de forma similar.
- Como então medir a dimensão de um objeto?
- Para o caso de formas regulares, uma forma com dimensão d é composta por N cópias de tamanho $1/m$ em relação ao tamanho original, onde $N = m^{-d}$.
 - m é conhecido como fator de redução.

- Exemplo: quadrado.



$$m = 1/2 \Rightarrow N = 4; m = 1/3 \Rightarrow N = 9.$$

- A ideia da relação entre o logaritmo do número de cópias de um objeto e o logaritmo do tamanho de suas cópias sugeriu uma generalização do conceito de dimensão que permite valores fracionários.

- Essa dimensão é conhecida como *dimensão de auto-similaridade*:

$$N = (1/m)^d \Rightarrow \log(N) = \log((1/m)^d) \Rightarrow \log(N) = d \cdot \log(1/m) \Rightarrow d = \frac{\log N}{\log 1/m}$$

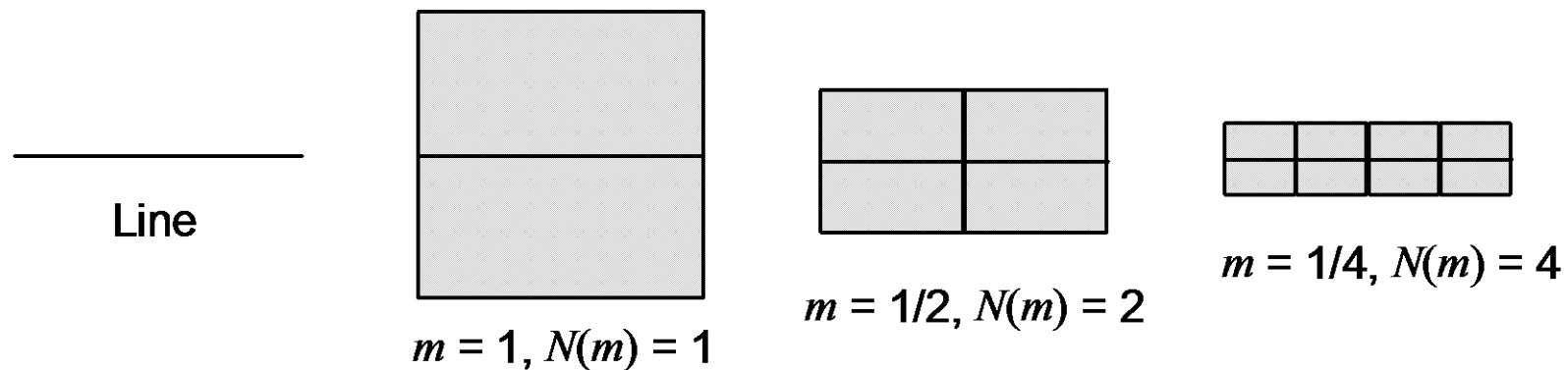
- Portanto, a dimensão d de uma forma auto-similar pode então ser dada por:

$$d = \frac{\log N}{\log 1/m},$$

onde N é a quantidade de cópias do objeto e m é o fator de redução.

- Em 1919, Hausdorff estendeu a noção de dimensão de similaridade para que ela se aplicasse a todo tipo de formas, além das formas auto-similares.
- A dimensão fractal serve para descrever a complexidade (fractal) de um objeto.
 - Exemplos:
 - ✓ A costa Britânica e a curva de Koch possuem dimensão fractal aproximada de 1,26;
 - ✓ Uma nuvem típica, quando projetada num plano, possui dimensão 1,35;
 - ✓ As curvas de Peano e Hilbert possuem dimensão 2,0, justamente por possuíram a propriedade de preencherem o quadrado unitário;
 - ✓ O conjunto de Cantor possui dimensão fractal $\log_3 2 = 0,6309$.
- Embora o escopo da dimensão de Hausdorff seja geral, ela é difícil de ser calculada na prática. Uma forma mais simples de medir a dimensão fractal de uma curva é denominada de *método da contagem de quadrados*:

- Cubra com quadrados a forma cuja dimensão você pretende medir e verifique como o número de quadrados varia em relação ao tamanho dos quadrados.
- No limite, para uma forma fractal, a taxa com a qual a proporção de quadrados cheios decresce fornece a dimensão do método da contagem de quadrados.



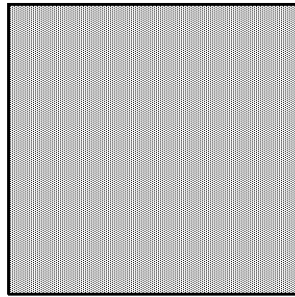
$$N(1) = 1$$

$$N(1/2) = 2 = 1/(1/2)$$

$$N(1/4) = 4 = 1/(1/4)$$

...

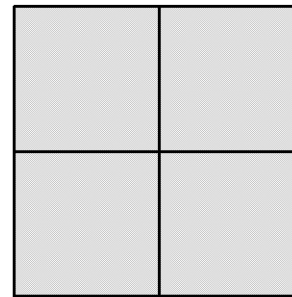
$$N(m) = 1/m$$



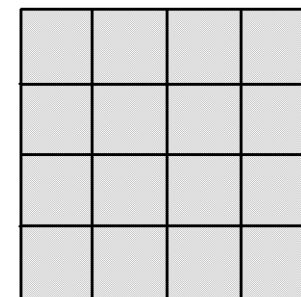
Filled-in
square



$m = 1, N(m) = 1$



$m = 1/2, N(m) = 4$



$m = 1/4, N(m) = 16$

$$N(1) = 1$$

$$N(1/2) = 4 = (1/(1/2))^2$$

$$N(1/4) = 16 = (1/(1/4))^2$$

$$\dots$$
$$N(m) = (1/m)^2$$

- Para objetos mais complicados, como fractais, a relação entre $N(m)$ e $1/m$ pode ser uma potência: $N(m) = k(1/m)^d$, resultando na definição da dimensão do método da contagem de quadrados:

$$d_b = \frac{\log(N(m)) - \log(k)}{\log(1/m)}.$$

- Note que a equação acima define a dimensão como sendo o coeficiente angular da reta formada entre $\log(N(m))$ e $\log(1/m)$ e que intercepta o eixo y no ponto $\log(k)$.
- No limite para $m \rightarrow 0$:

$$d_b = \lim_{m \rightarrow 0} \frac{\log(N(m))}{\log(1/m)} .$$

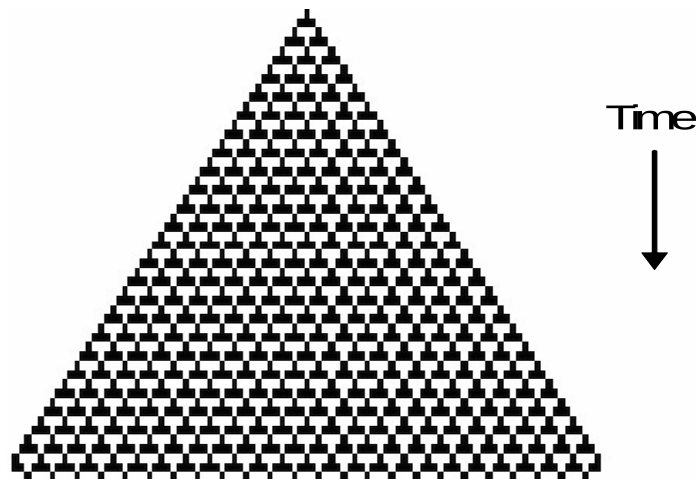
3. Autômatos Celulares e Geometria Fractal

- O estado s_i de uma célula i (i é o índice que indica a posição da célula) é atualizado em tempos discretos de acordo com regras determinísticas que dependem da vizinhança da célula e de seu estado atual. Exemplo:

$$s_i(t + 1) = f(s_{i-1}(t), s_i(t), s_{i+1}(t))$$



$0\underline{0}0 \rightarrow 0$	$1\underline{0}0 \rightarrow 1$
$00\underline{1} \rightarrow 1$	$10\underline{1} \rightarrow 1$
$0\underline{1}0 \rightarrow 1$	$1\underline{1}0 \rightarrow 0$
$0\underline{1}1 \rightarrow 0$	$1\underline{1}1 \rightarrow 0$



3.1. Definição Formal

- Um autômato celular d -dimensional consiste em uma grade finita de dimensão d com células que podem assumir um entre um conjunto finito de valores (estados).

- O estado de cada célula no tempo $t + 1$ é função do estado de um conjunto de células vizinhas no instante t .

- Formalmente, um CA pode ser definido como uma quintupla:

$$C = (S, s_0, G, d, f),$$

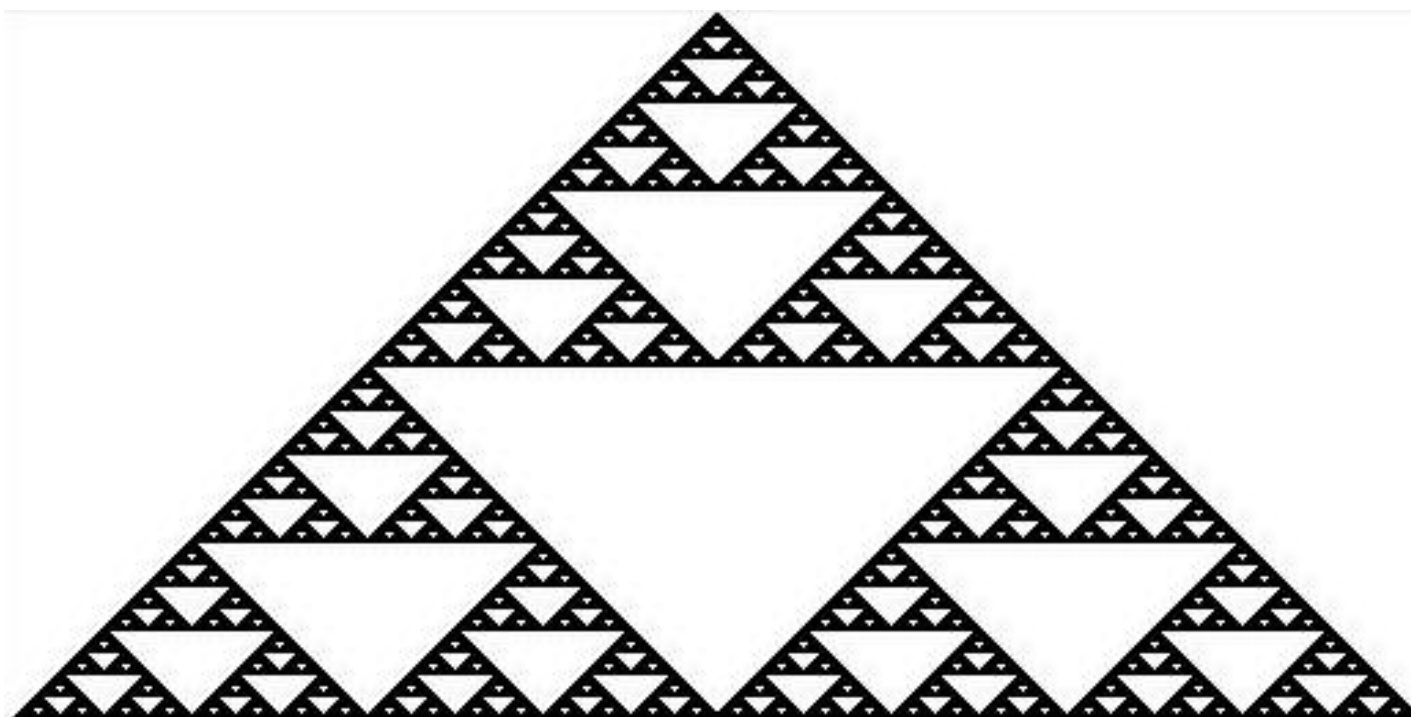
onde S é um conjunto finito de estados, $s_0 \in S$ são os estados iniciais, G é a vizinhança da célula, $d \in \mathbb{Z}^+$ é a dimensão de C , e f é o conjunto de regras locais de interação, também conhecidas como regras ou funções de transição.

- Outros conceitos:

- Vizinhança: $G_i = \{\mathbf{i}, \mathbf{i} + \mathbf{r}_1, \mathbf{i} + \mathbf{r}_2, \dots, \mathbf{i} + \mathbf{r}_n\}$, onde n é o tamanho da vizinhança.
- Regra de transição: $f: S^n \rightarrow S$
- Configuração da grade: $C(t) = (s_0(t), s_1(t), \dots, s_{N-1}(t))$, onde s_i é o estado da célula $\mathbf{i}$ no instante t .
- Progressão: $F: C(t) \rightarrow C(t + 1), \quad t = 0, 1, \dots$

- Exemplo de aplicação: geração de formas fractais.

$0\underline{0}0 \rightarrow 0$	$1\underline{0}0 \rightarrow 1$
$00\underline{1} \rightarrow 1$	$10\underline{1} \rightarrow 1$
$0\underline{1}0 \rightarrow 1$	$1\underline{1}0 \rightarrow 1$
$0\underline{1}1 \rightarrow 1$	$1\underline{1}1 \rightarrow 0$





A próxima seção está fundamentada na versão digital do livro de Prusinkiewicz & Lindenmayer (1990), editada em 2004.

4. Sistemas L (L-Systems)

- A. Lindenmayer introduziu em 1968 um formalismo para simular o desenvolvimento de organismos multicelulares.
- Estes sistemas foram posteriormente denominados sistemas-L (*L-systems*) ou algoritmos de desenvolvimento (*developmental algorithms*).
- O desenvolvimento multicelular consiste na geração de estruturas através da divisão, crescimento, diferenciação e morte celular. Ele corresponde a uma série de mudanças que os organismos sofrem durante sua passagem do estado embrionário para a maturidade.

4.1. Conceitos sobre Sistemas de Produção e Gramáticas

- O formalismo por trás dos sistemas-L é baseado em *sistemas de produção* e uma *gramática* específica.

- Um *sistema de produção*, também conhecido como *sistema baseado em regras*, emprega *implicações* como sua representação primária.
- Tipicamente seu elemento mais importante é um conjunto de *regras de produção* ou simplesmente *produções*. Ex. **Se a então b .**

Gramática

- Um *alfabeto* V é um conjunto finito de símbolos. O conjunto de todas as palavras finitas formadas pelo alfabeto V é denominado por V^* .
- Uma *palavra* ou *string* w é uma sequência de elementos de V .
- A string vazia ou de comprimento zero, e , também faz parte de V^* ; $V^+ = V^* - \{e\}$.
- Uma *linguagem* L sobre um alfabeto V é qualquer subconjunto de V^* ; $L \subseteq V^*$.
- Uma *gramática* é uma quintupla $G = \langle V, T, N, P, S \rangle$, onde V é um conjunto finito (alfabeto); T é o conjunto de símbolos terminais; N é o conjunto de símbolos não-terminais; P é o conjunto de produções; e S é um símbolo não-terminal conhecido como *axioma* (ponto de partida).

4.2. Sistemas DOL

- A ideia básica de um sistema-L está contida na natureza das linguagens formais.
- As formas geométricas (fractais) a serem estudadas são palavras em uma linguagem formal paralela.
- As gramáticas em sistemas-L são similares às gramáticas formais apresentadas anteriormente, porém as produções são aplicadas simultaneamente (paralelamente) e não existe distinção entre símbolos terminais e não-terminais.
- Exemplo de um sistema-L:
 - Seja o alfabeto $G = \{a,b,c\}$, onde c é o axioma, e as seguintes produções:
 1. $a \rightarrow c$
 2. $b \rightarrow ac$
 3. $c \rightarrow b$
 - O processo de aplicação das regras é denominado de *processo de derivação* ou de *reescrita*.
- A introdução dos sistemas-L fez reavivar o interesse na representação de imagens utilizando sequências de caracteres.

Iteration	Word
-----------	------

0	<i>c</i>
1	<i>b</i>
2	<i>ac</i>
3	<i>cb</i>
4	<i>bac</i>
5	<i>accb</i>
6	<i>cbbac</i>
7	<i>bacaccb</i>
8	<i>accbcbbac</i>
9	<i>cbbacbacaccb</i>
10	<i>bacaccbaccbcbac</i>

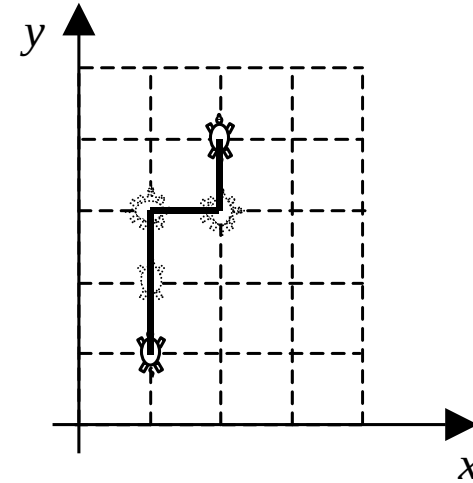
- Um sistema-OL é definido como sendo a tripla ordenada $G = \langle V, \omega, P \rangle$, onde V é o alfabeto do sistema, $\omega \in V^+$ é uma palavra de comprimento não-nulo denominada de axioma, e $P \subset V \times V^*$ é um conjunto finito de produções.
- A notação OL se refere a classes de linguagens geradas por sistemas-L livres de contexto.

- Uma produção $(a, \chi) \in P$ é escrita como $a \rightarrow \chi$. A letra a e a palavra χ são denominadas de *predecessor* e de *sucessor* da produção, respectivamente.
- É considerado que para cada letra do alfabeto existe pelo menos uma palavra do conjunto V^* tal que $a \rightarrow \chi$, ou seja, $\exists \chi \in V^* \mid a \rightarrow \chi, \forall a \in V$.
- Caso nenhuma produção for especificada para um determinado predecessor a , $a \rightarrow a$ é tomado por *default*.
- Um sistema-OL é dito determinístico, sistema-DOL, se e somente se para cada $a \in V$ existe um único $\chi \in V^*$ tal que $a \rightarrow \chi$.
- Seja $\mu = a_1 \dots a_m$ uma palavra arbitrária em V . A palavra $v = \chi_1 \dots \chi_m \in V^*$ é diretamente derivável (ou gerada a partir) de μ , $\mu \Rightarrow v$, se e somente se $a_i \rightarrow \chi_i$, $i = 1, \dots, m$.
- Uma palavra v é gerada por G em uma derivação de comprimento n se existe uma *sequência de desenvolvimento* de palavras $\mu_0, \mu_1, \dots, \mu_n$ tais que $\mu_0 = \omega$, $\mu_n = v$ e $\mu_0 \Rightarrow \mu_1 \Rightarrow \dots \Rightarrow \mu_n$.

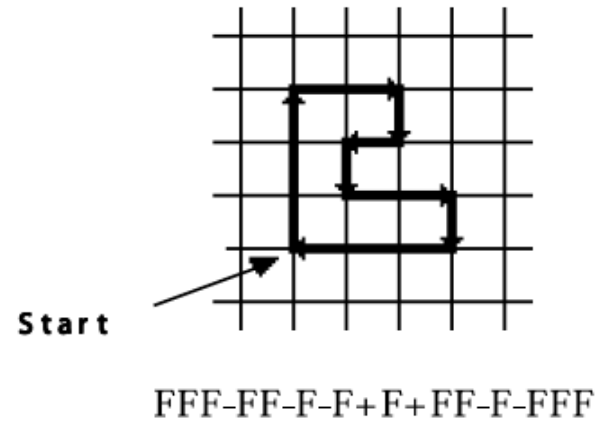
4.3. Gráfico Tartaruga (Turtle Graphics)

- Na descrição apresentada acima, o sistema-L foi utilizado para gerar uma sequência de palavras.
- A interpretação geométrica destas palavras pode ser utilizada para gerar imagens de diversos padrões naturais.
- Uma linguagem do tipo *turtle graphics* pode ser usada para fazer a interpretação geométrica das palavras geradas pelo sistema-L.
- Ideia básica em 2-D:
 - O estado da tartaruga é definido como sendo a tripla (x,y,α) , onde as coordenadas cartesianas (x,y) correspondem à posição da tartaruga, e o ângulo α , denominado de *direção (heading)*, é interpretado como sendo a direção para a qual a tartaruga aponta.
 - Dado o tamanho do passo d e o incremento de ângulo δ , a tartaruga pode responder a diversos comandos:

- F Mova adiante com um passo de comprimento d . O estado da tartaruga muda para (x', y', α) , onde $x' = x + d \cdot \cos \alpha$ e $y' = y + d \cdot \sin \alpha$. Um segmento de reta entre os pontos (x, y) e (x', y') é desenhado.
- f Mova adiante com um passo de comprimento d , sem desenhar o segmento de reta.
- + Rotacione no sentido horário com um ângulo δ . O próximo estado da tartaruga é $(x, y, \alpha + \delta)$.
- Rotacione no sentido anti-horário com um ângulo δ . O próximo estado da tartaruga é $(x, y, \alpha - \delta)$.
- Dado um axioma v , o estado inicial da tartaruga (x_0, y_0, α_0) e os parâmetros d e δ , a *turtle interpretation* de v é a figura (conjunto de linhas) desenhada pela tartaruga em resposta a v .



b

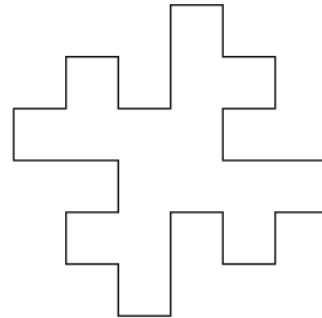


a



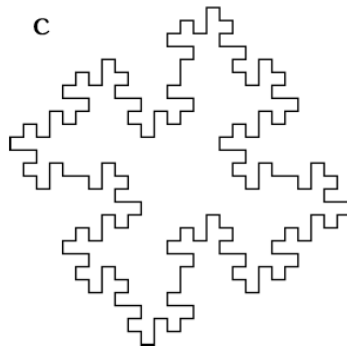
n = 0

b



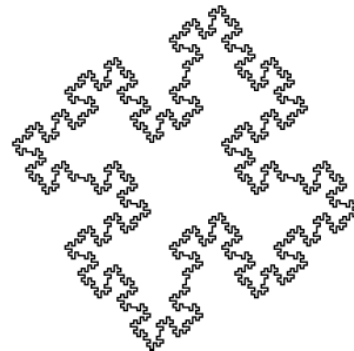
n = 1

c



n = 2

d



n = 3

Generating a quadratic Koch island

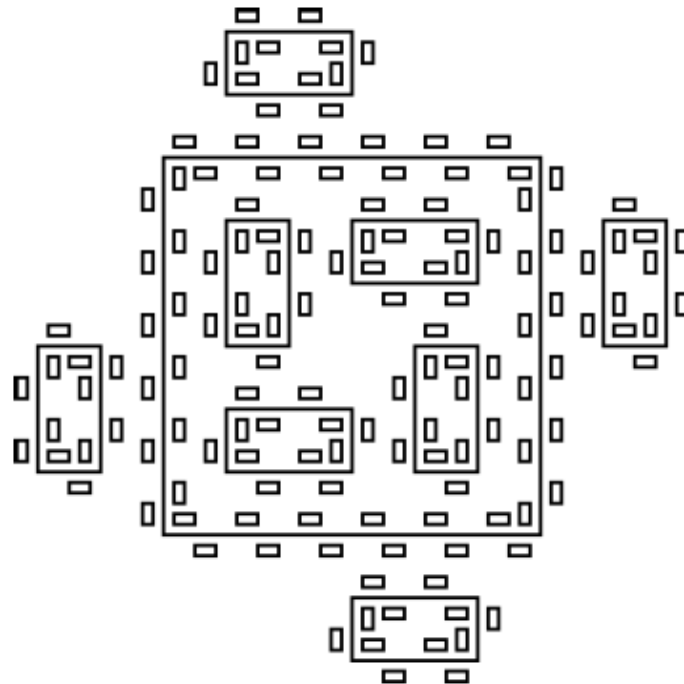
These figures were obtained by interpreting strings generated by the following L-system:

$$\omega : F - F - F - F$$

$$p : F \rightarrow F - F + F + FF - F - F + F$$

The images correspond to the strings obtained in derivations of length 0 to 3. The angle increment δ is equal to 90° . The step length d is decreased four times between subsequent images, making the distance between the endpoints of the successor polygon equal to the length of the predecessor segment.

Ilhas e lagos



$$n = 2, \delta = 90^\circ$$

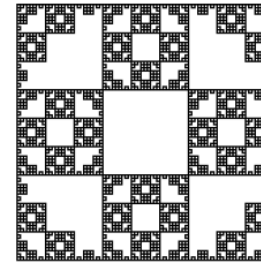
$$F + F + F + F$$

$$F \rightarrow F + f - FF + F + FF + Ff + FF - f + FF - F - FF - Ff - FFF$$

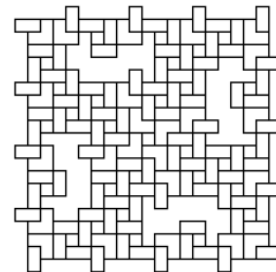
$$f \rightarrow ffffff$$



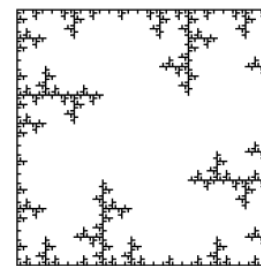
a $n = 4, \delta = 90^\circ$
F-F-F-F
 $F \rightarrow FF-F-F-F-F+F$



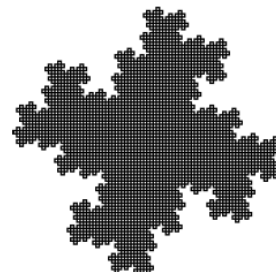
b $n = 4, \delta = 90^\circ$
F-F-F-F
 $F \rightarrow FF-F-F-FF$



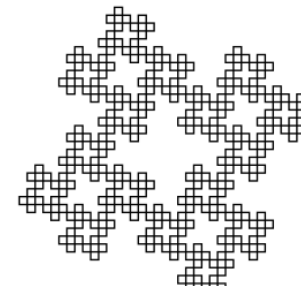
c $n = 3, \delta = 90^\circ$
F-F-F-F
 $F \rightarrow FF-F+F-F-FF$



d $n = 4, \delta = 90^\circ$
F-F-F-F
 $F \rightarrow FF-F--F-F$



e $n = 5, \delta = 90^\circ$
F-F-F-F
 $F \rightarrow F-FF--F-F$



f $n = 4, \delta = 90^\circ$
F-F-F-F
 $F \rightarrow F-F+F-F-F$

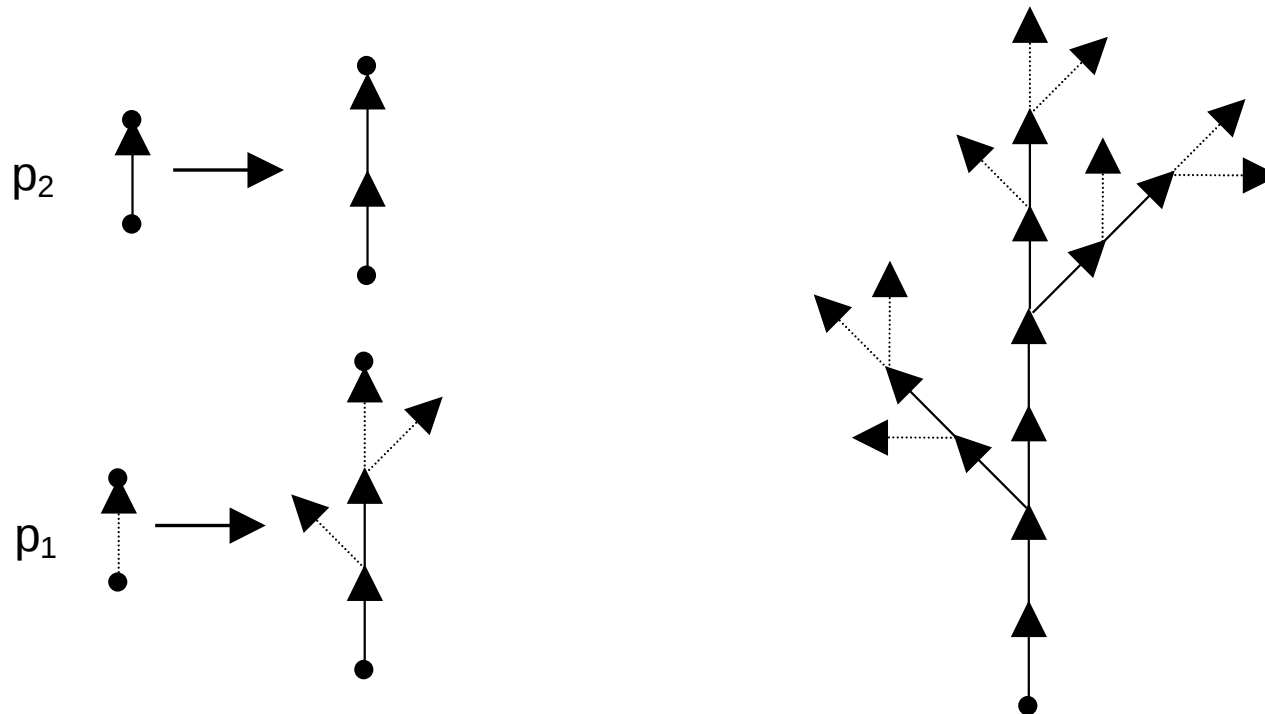
A sequence of Koch curves obtained by successive modification
of the production successor

4.4. Modelos de Arquiteturas de Plantas

- Em 1968, Lindenmayer estendeu os sistemas-L incluindo os colchetes $\{[,]\}$ no alfabeto dos sistemas-L, criando os *bracketed L-systems*. A motivação foi a de descrever estruturas ramificadas observadas em plantas, algas, árvores.
- Os dois novos símbolos '[' e ']' são interpretados pela tartaruga como a seguir:
 - [Armazene o estado corrente (x,y,α) da tartaruga para uso futuro, usando pilha.
 -] Remova o último estado salvo da pilha e o utilize para restaurar o último estado da tartaruga. Nenhuma linha é desenhada, embora em geral a posição da tartaruga mude.
- Exemplo: $V = \{F, G, [,], +, -\}$, axioma $\omega = F$, $\delta = 45^\circ$, produções:
 - $p_1: F \rightarrow G[-F]G[+F]F$
 - $p_2: G \rightarrow GG$
 - $p_3: [\rightarrow [$
 - $p_4:] \rightarrow]$

Iteration	Word
0	F
1	$G[-F]G[+F]F$
2	$GG[-G[-F]G[+F]F]GG[+G[-F]G[+F]F] G[-F]G[+F]F$

- F : seta tracejada; G : seta sólida.



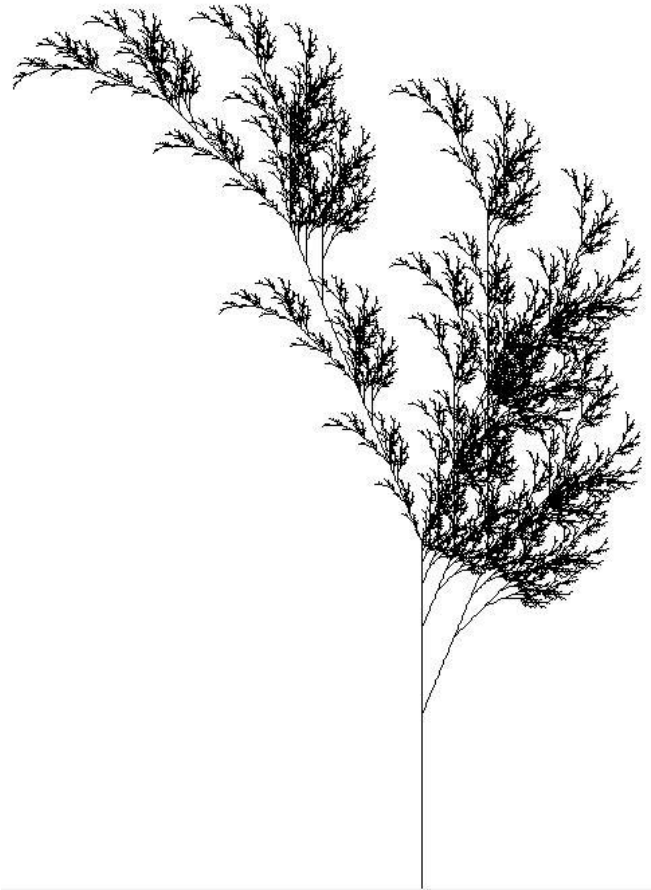
```
procedure [word] = DOL_turtle(max_it,  $\omega$ , P, d,  $\delta$ )  
  word  $\leftarrow$   $\omega$   
  t  $\leftarrow$  1  
  while t < max_it do,  
    word  $\leftarrow$  rewrite(word, P)  
    t  $\leftarrow$  t + 1  
  end while  
  turtle(word, d,  $\delta$ );  
end procedure
```

Algoritmo 7.1: Um procedimento de interpretação de um sistema-DOL

4.5. Escopo dos Sistemas-L

- Projeto de paisagens, ornamentações, ilustrações botânicas, desenvolvimento de organismos, reconstrução de plantas extintas, modelos estruturais de plantas, projeto de novas variedades de plantas, predição de colheita, descrição de florescência, simulação de crescimento de fungos, etc.

- Exemplos de plantas geradas com sistemas-L.

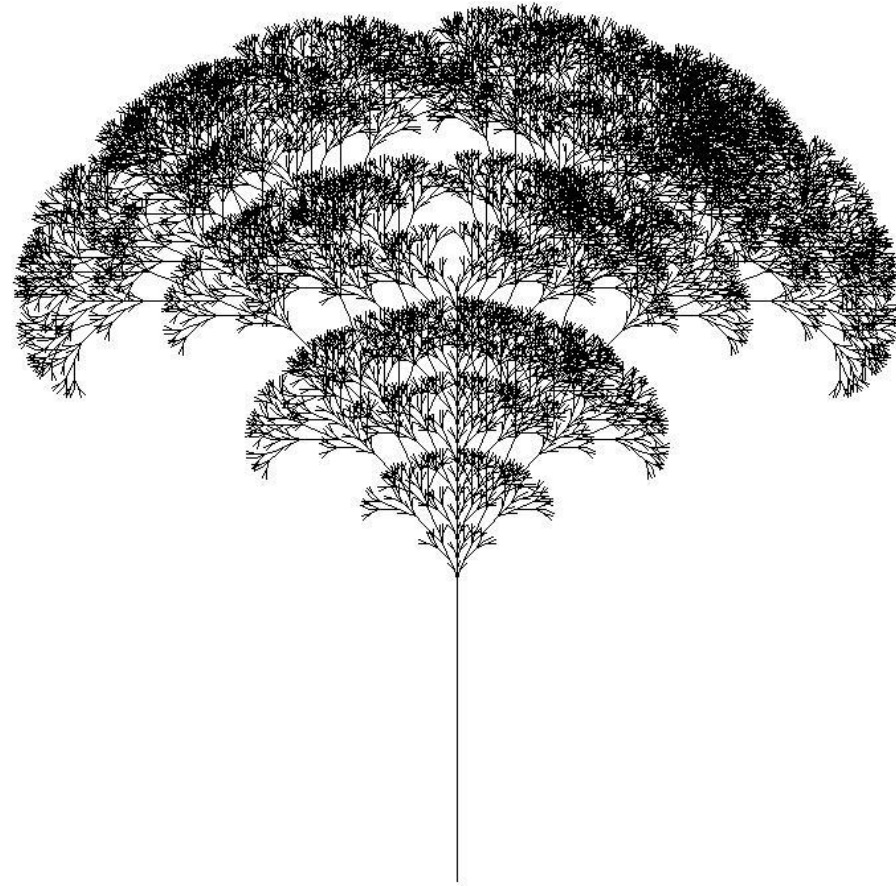


$t = 8, \delta = 22.5^\circ$

$\omega: G$

$G \rightarrow F+[[G]-G]-F[-FG]+G$

$F \rightarrow FF$



$t = 5, \delta = 22.5^\circ$

$\omega: G$

$G \rightarrow FG[-F[G]-G][G+G][+F[G]+G]$

$F \rightarrow FF$



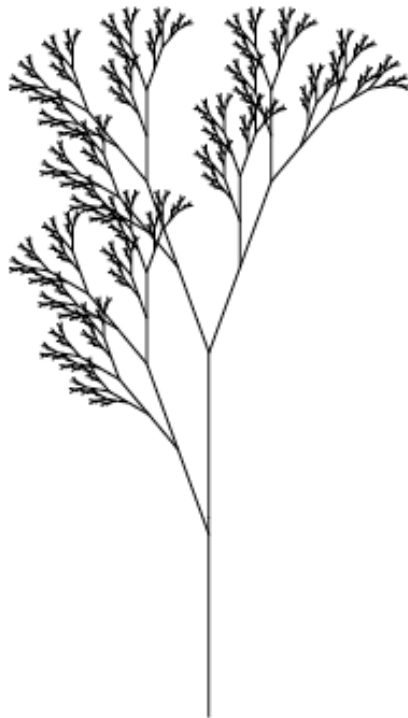
a
 $n=5, \delta=25.7^\circ$
 F
 $F \rightarrow F[+F]F[-F]F$



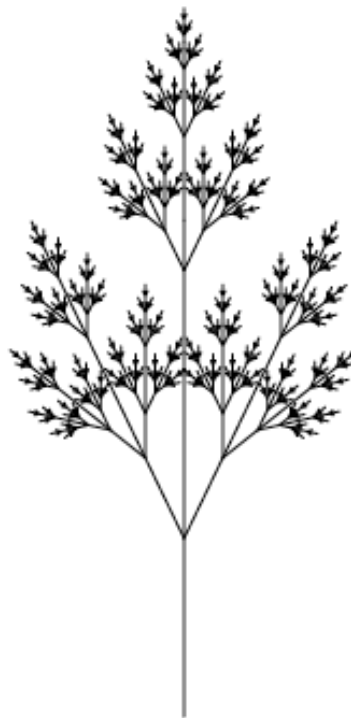
b
 $n=5, \delta=20^\circ$
 F
 $F \rightarrow F[+F]F[-F][F]$



c
 $n=4, \delta=22.5^\circ$
 F
 $F \rightarrow FF[-F+F+F] + [F-F-F]$



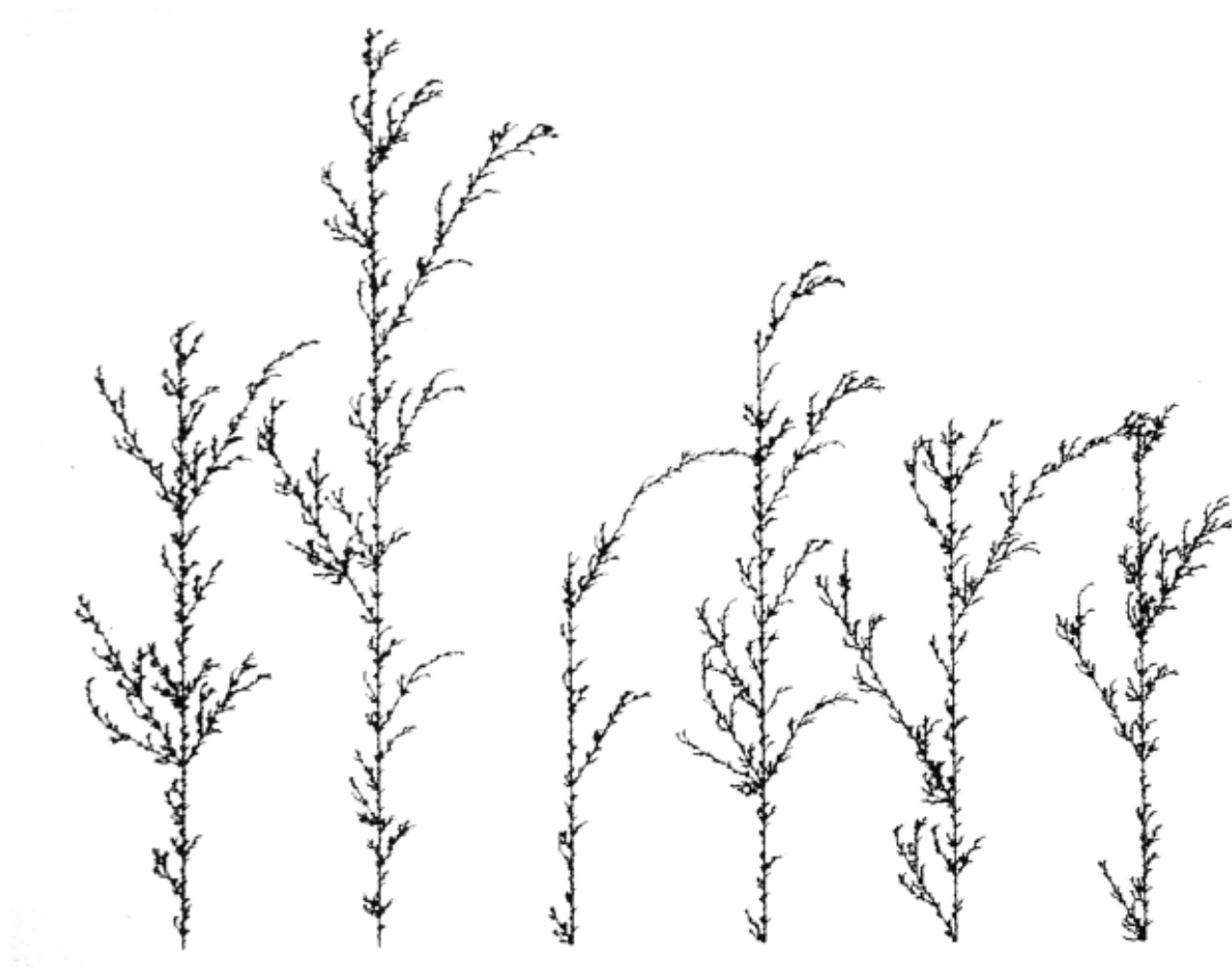
d
 $n=7, \delta=20^\circ$
 X
 $X \rightarrow F[+X]F[-X]+X$
 $F \rightarrow FF$



e
 $n=7, \delta=25.7^\circ$
 X
 $X \rightarrow F[+X][-X]FX$
 $F \rightarrow FF$



f
 $n=5, \delta=22.5^\circ$
 X
 $X \rightarrow F-[[X]+X]+F[+FX]-X$
 $F \rightarrow FF$



Stochastic branching structures

5. Sistemas de Funções Iterativas

- Os sistemas de funções iterativas (IFSs – *iterated function systems*) foram desenvolvidos por J. Hutchinson, M. Barnsley e S. Demko como uma ferramenta para a geração de fractais através do uso de um conjunto de transformações, também denominadas de *mapeamentos contrativos*, de uma imagem sobre si própria.
- Os IFSs consistem basicamente da aplicação recursiva de um conjunto de *transformações afins* a um conjunto de pontos iniciais (imagem).
- Após um determinado número de iterações, o conjunto final, ou limite, irá definir uma certa configuração geométrica.

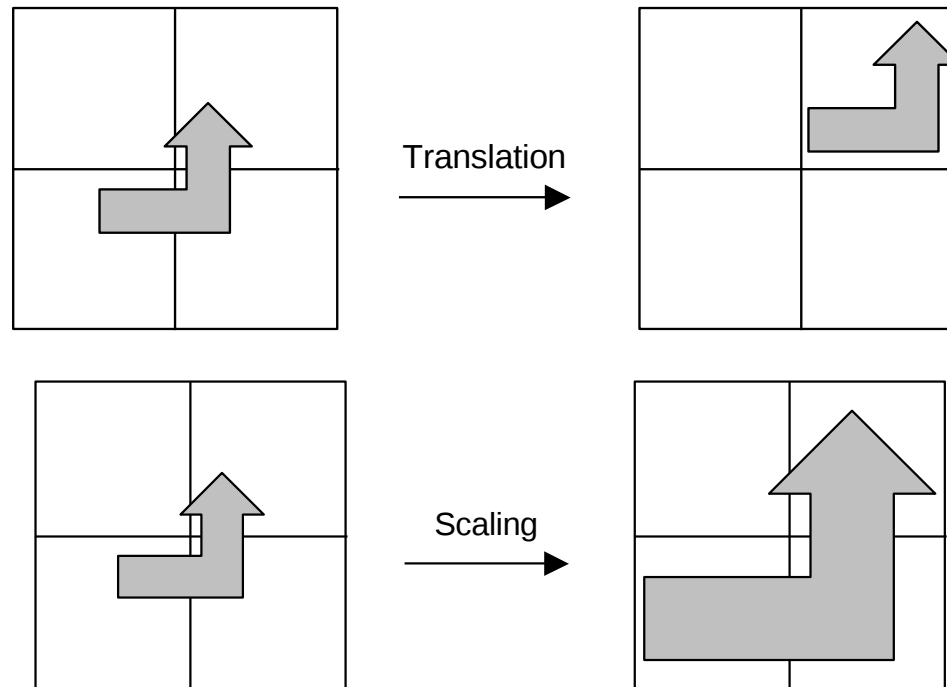
5.1. Conceitos básicos

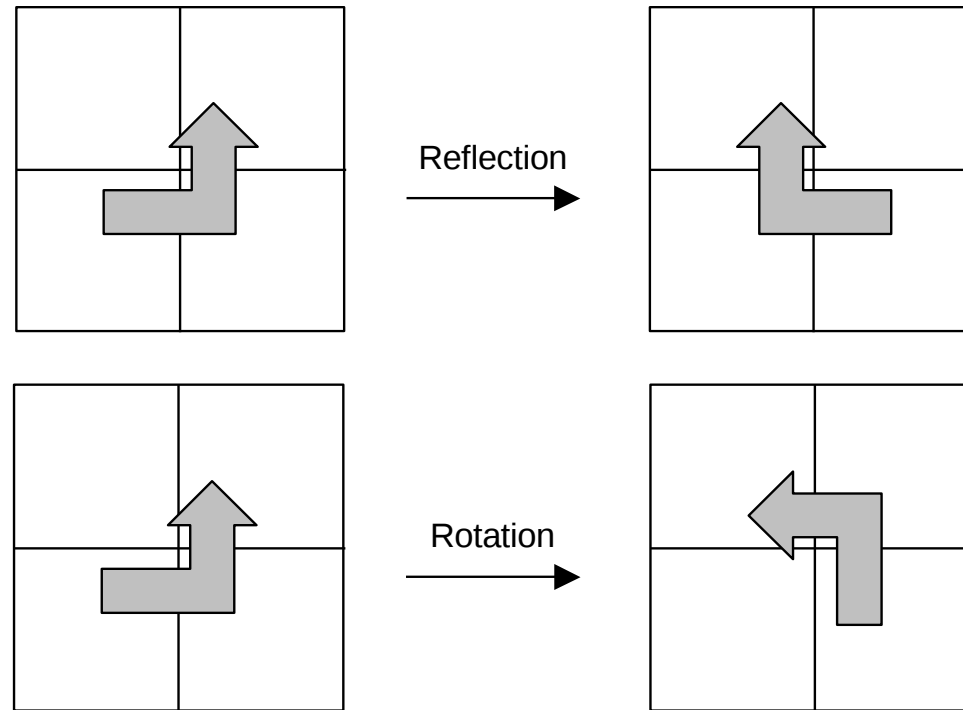
- Um *espaço X* é um conjunto, e os *pontos* do espaço são elementos do conjunto.

- Um espaço métrico $(\mathbf{X}, d)$ é um espaço $\mathbf{X}$ juntamente com uma função real $d : \mathbf{X} \times \mathbf{X} \rightarrow \mathbb{R}$ que mede a distância entre pares de pontos $x, y \in \mathbf{X}$.
- A função d , denominada de *métrica*, deve obedecer às seguintes propriedades:
 1. $d(x, y) = d(y, x) \quad \forall x, y \in \mathbf{X}$
 2. $0 < d(x, y) < \infty \quad \forall x, y \in \mathbf{X}, x \neq y$
 3. $d(x, x) = 0 \quad \forall x \in \mathbf{X}$
 4. $d(x, y) \leq d(x, z) + d(z, y) \quad \forall x, y, z \in \mathbf{X}$
- Seja $(\mathbf{X}, d)$ um espaço métrico. Uma transformação sobre $\mathbf{X}$ é uma função $f : \mathbf{X} \rightarrow \mathbf{X}$ que especifica exatamente um ponto $f(x) \in \mathbf{X}$ a cada ponto $x \in \mathbf{X}$.
- Uma transformação $w : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ da forma $w(x_1, x_2) = (ax_1 + bx_2 + e, cx_1 + dx_2 + f)$, onde a, b, c, d, e , e f são números reais, é denominada de *transformação afim* (*bi-dimensional*).
- Em notação matricial:

$$w(\mathbf{x}) = w\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} = \mathbf{Ax} + \mathbf{t}$$

- Estas transformações possuem propriedades algébricas e geométricas importantes.
- As quatro principais transformações afins são:
 - Translação, escalonamento, reflexão, e rotação.





5.2. Mapeamentos e aplicações

- Uma transformação $f: \mathbf{X} \rightarrow \mathbf{X}$ sobre um espaço métrico $(\mathbf{X}, d)$ é denominada *contrativa*, ou *mapeamento contrativo*, se existe uma constante $0 \leq s < 1$ tal que:

$$d(f(x), f(y)) \leq s \cdot d(x, y), \quad \forall x, y \in \mathbf{X},$$

onde s é chamado de *fator de contratividade* para f .

- Uma propriedade importante de contrações é que, independentemente do ponto inicial, a aplicação iterativa do mapeamento contrativo resulta sempre na convergência para o mesmo ponto, denominado de *atrator*.
- Um *sistema de funções iterativas* consiste em um espaço métrico completo $(\mathbf{X}, d)$ juntamente com um conjunto finito de mapeamentos contrativos $w_n : \mathbf{X} \rightarrow \mathbf{X}$ (com os respectivos fatores de contratividade s_n , $n = 1, 2, \dots, N$).
- Vamos nos restringir ao caso de IFS da forma $\{\mathbb{R}^2; w_n : n = 1, 2, \dots, N\}$, onde cada mapeamento contrativo corresponde a uma transformação linear.

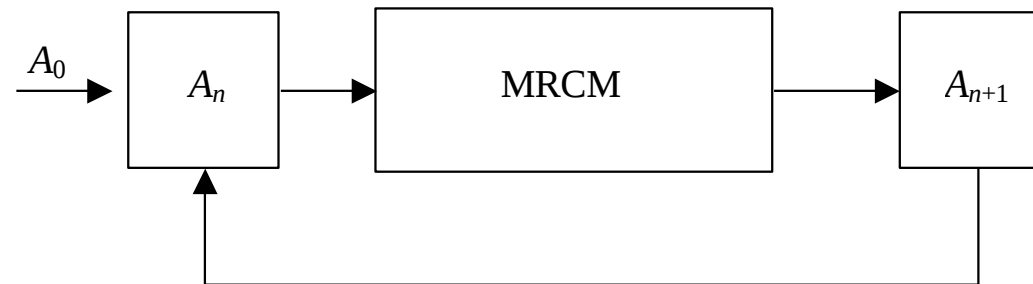
Sistema de funções iterativas determinístico (*deterministic IFS – DIFS*).

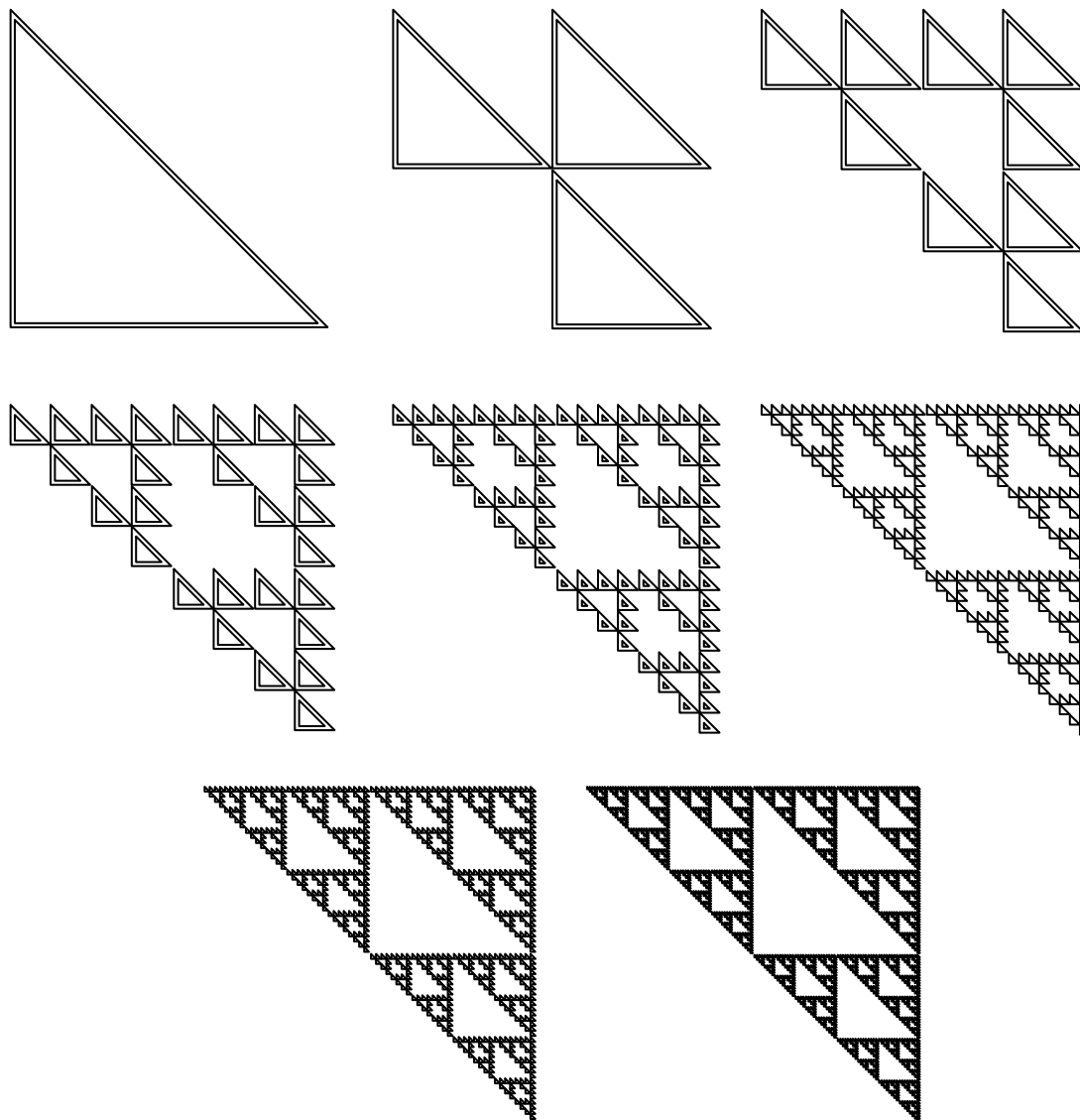
- Este algoritmo é baseado na ideia de calcular diretamente uma sequência de conjuntos $\{A_n = W^{on}(A)\}$ a partir de um conjunto inicial A_0 .

- Seja $\{\mathbf{X}; w_1, w_2, \dots, w_N\}$ um IFS. Escolha um conjunto compacto $A_0 \subset \mathbb{R}^2$ que servirá de condição inicial para o DIFS. Em seguida, calcule sucessivamente $A_n = W^{\text{on}}(A)$ de acordo com:

$$A_{n+1} = \cup_{j=1}^N w_j(A_n), \quad n = 1, 2, \dots$$

- Assim, uma sequência $\{A_n : n = 0, 1, 2, \dots\}$ será construída e $\{A_n\}$ convergirá para o atrator do IFS de acordo com o teorema do mapeamento contrativo.
- Metáfora para o DIFS: *multiple reduction copy machine* (MRCM).





Sistema de funções iterativas aleatório (*random iterated function system – RIFS*).

- Seja $\{X; w_1, w_2, \dots, w_N\}$ um IFS onde uma probabilidade $p_i > 0$ é atribuída a cada mapeamento w_i , $i = 1, \dots, N$, $\sum_i p_i = 1$.
- Escolha $x_0 \in X$ e, em seguida, escolha recursivamente e independentemente $x_n \in \{w_1(x_{n-1}), w_2(x_{n-1}), \dots, w_N(x_{n-1})\}$, $\forall n$
onde a probabilidade de um evento $x_n = w_i(x_{n-1})$ ocorrer é p_i .
- Portanto, construa uma sequência $\{x_n : n = 0, 1, 2, \dots\} \subset X$.

```
procedure [] = RIFS(max_it, x0, W, p)
  x ← x0;
  t ← 1
  while t < max_it do,
    j ← select(p)      //select a mapping j with probability pj
    x ← wj(x)          //apply mapping j to x
    draw(x)            //plot point x on the screen
    t ← t + 1
  end while
end procedure
```

Algoritmo 7.2: Implementação do sistema de funções iterativas aleatório (RIFS).

- Exemplo: Arbusto de Barnsley

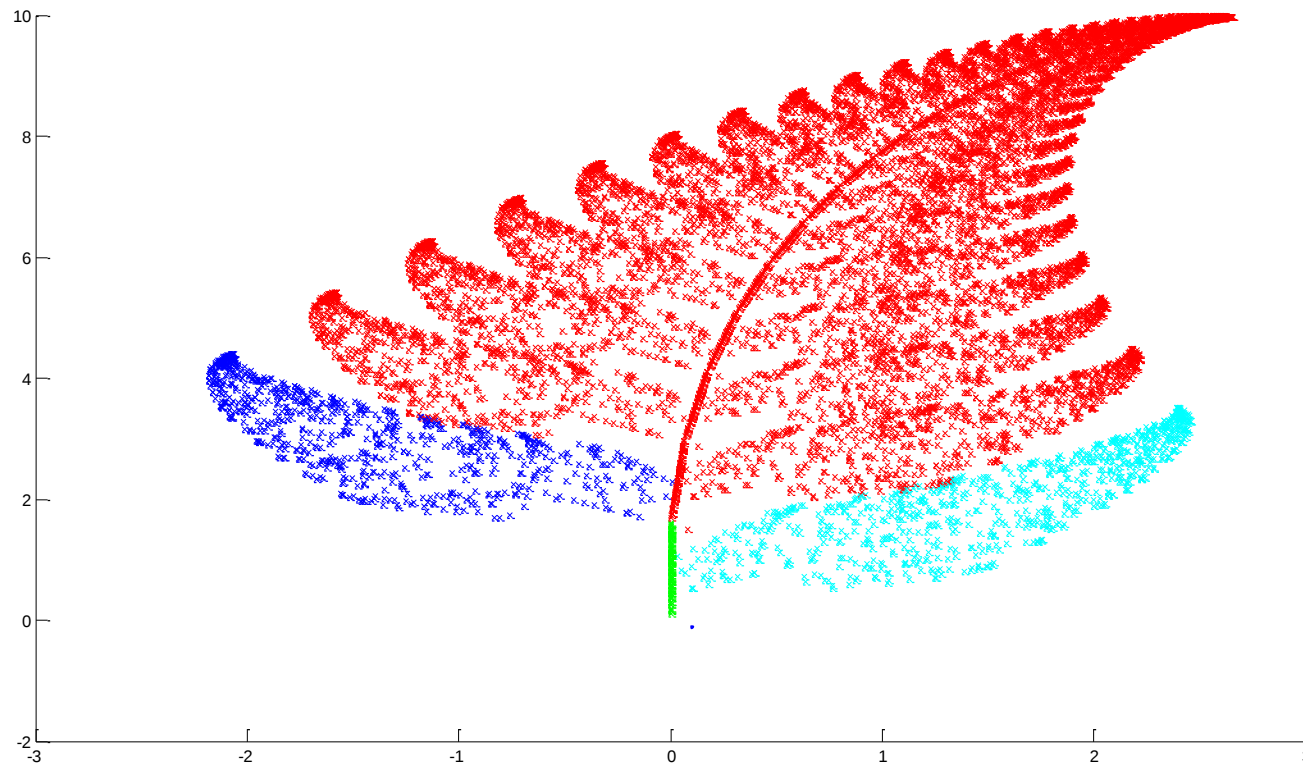
w	a	b	c	d	e	f	p
1	0	0	0	0.16	0	0	0.01
2	0.85	0.04	-0.04	0.85	0	1.6	0.85
3	0.2	-0.26	0.23	0.22	0	1.6	0.07
4	-0.15	0.28	0.26	0.24	0	0.44	0.07

$$w_1(x_1, x_2) = (0, 0.16 * x_2) \text{ (green)}$$

$$w_2(x_1, x_2) = (0.85 * x_1 + 0.04 * x_2, -0.04 * x_1 + 0.85 * x_2 + 1.6) \text{ (red)}$$

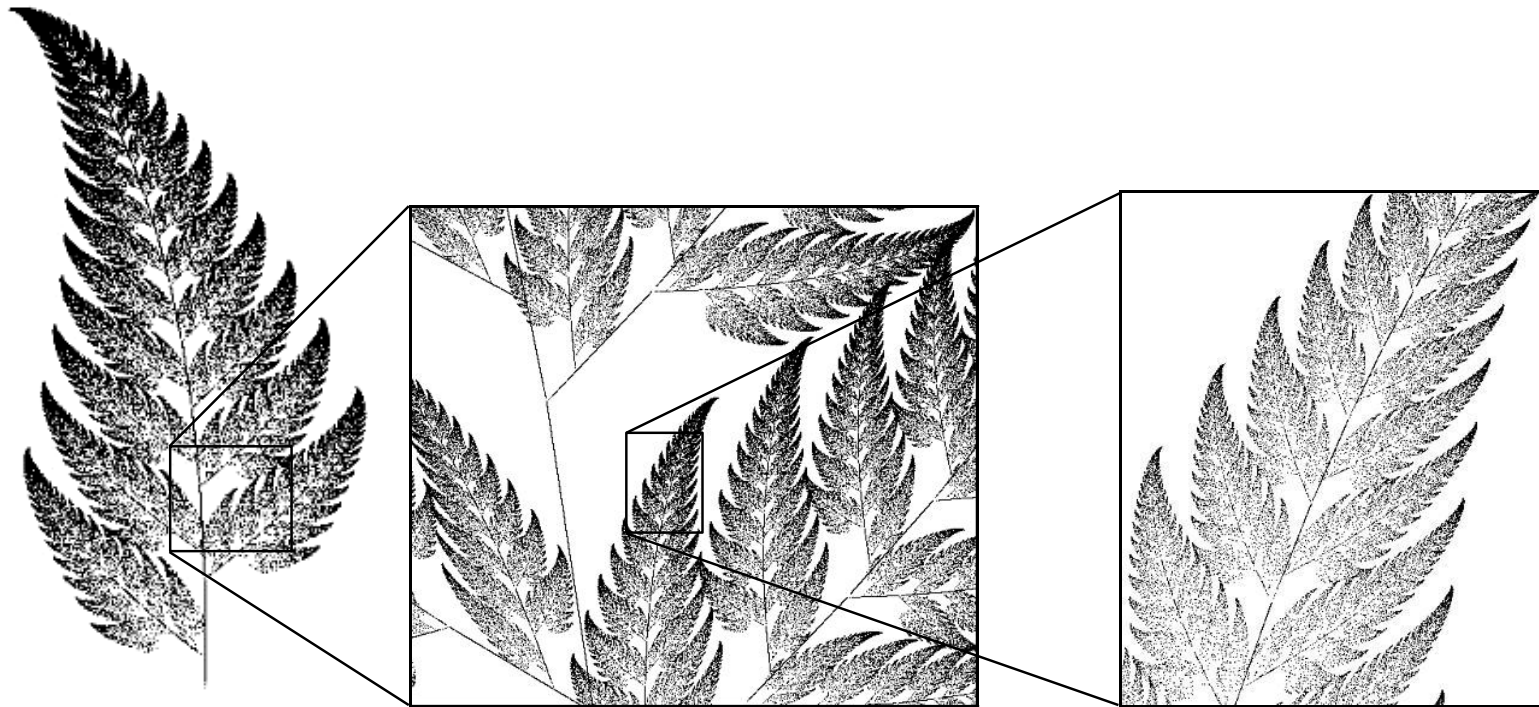
$$w_3(x_1, x_2) = (0.2 * x_1 - 0.26 * x_2, 0.23 * x_1 + 0.22 * x_2 + 1.6) \text{ (blue)}$$

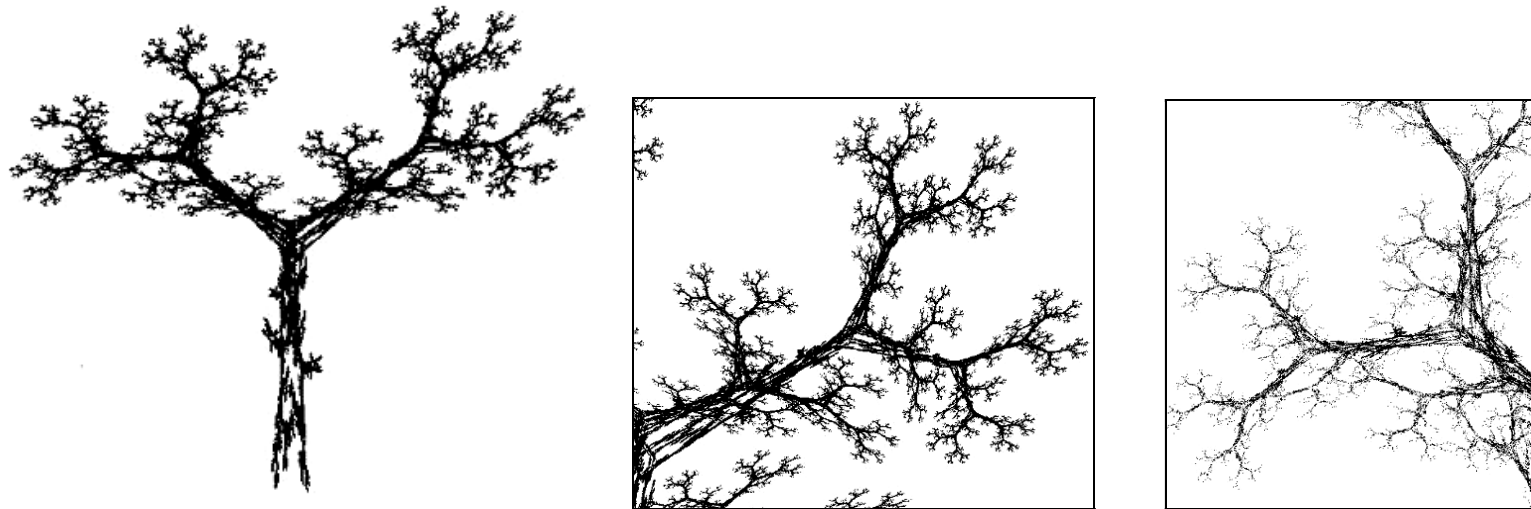
$$w_4(x_1, x_2) = (-0.15 * x_1 + 0.28 * x_2, 0.26 * x_1 + 0.24 * x_2 + 0.44) \text{ (cyan)}$$



Arbusto de Barnsley com 50.000 pontos (as cores indicam a transformação que gerou cada ponto)

5.3. Auto-Similaridade e Auto-Afinidade Revisitadas





6. Movimento Browniano

- Para modelar regiões costeiras e montanhas, são necessárias curvas que parecem diferentes quando amplificadas, mas que ainda apresentam a mesma impressão predominante.

6.1. Fractais Aleatórios na Natureza e Movimento Browniano

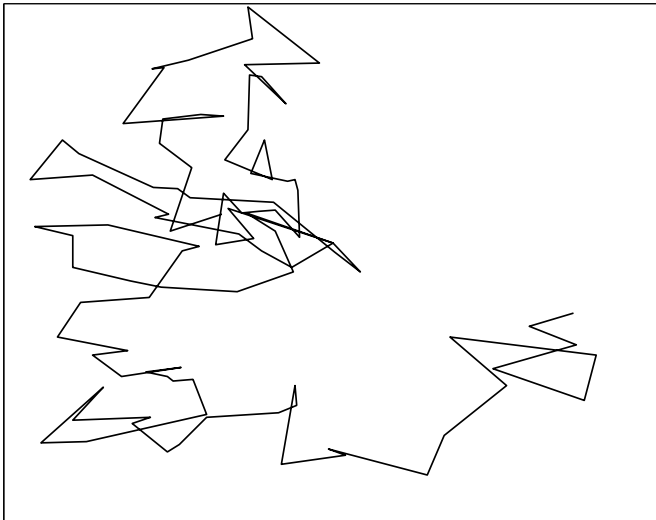
- Uma *caminhada aleatória* (*random walk*) é um caminho que pode ser gerado por um processo aleatório.

$$x(t+1) = x(t) + \Delta x$$

$$y(t+1) = y(t) + \Delta y$$

onde Δx e Δy podem ser distribuições gaussianas de média zero e desvio padrão 1.

Random Walk – First 100 Steps

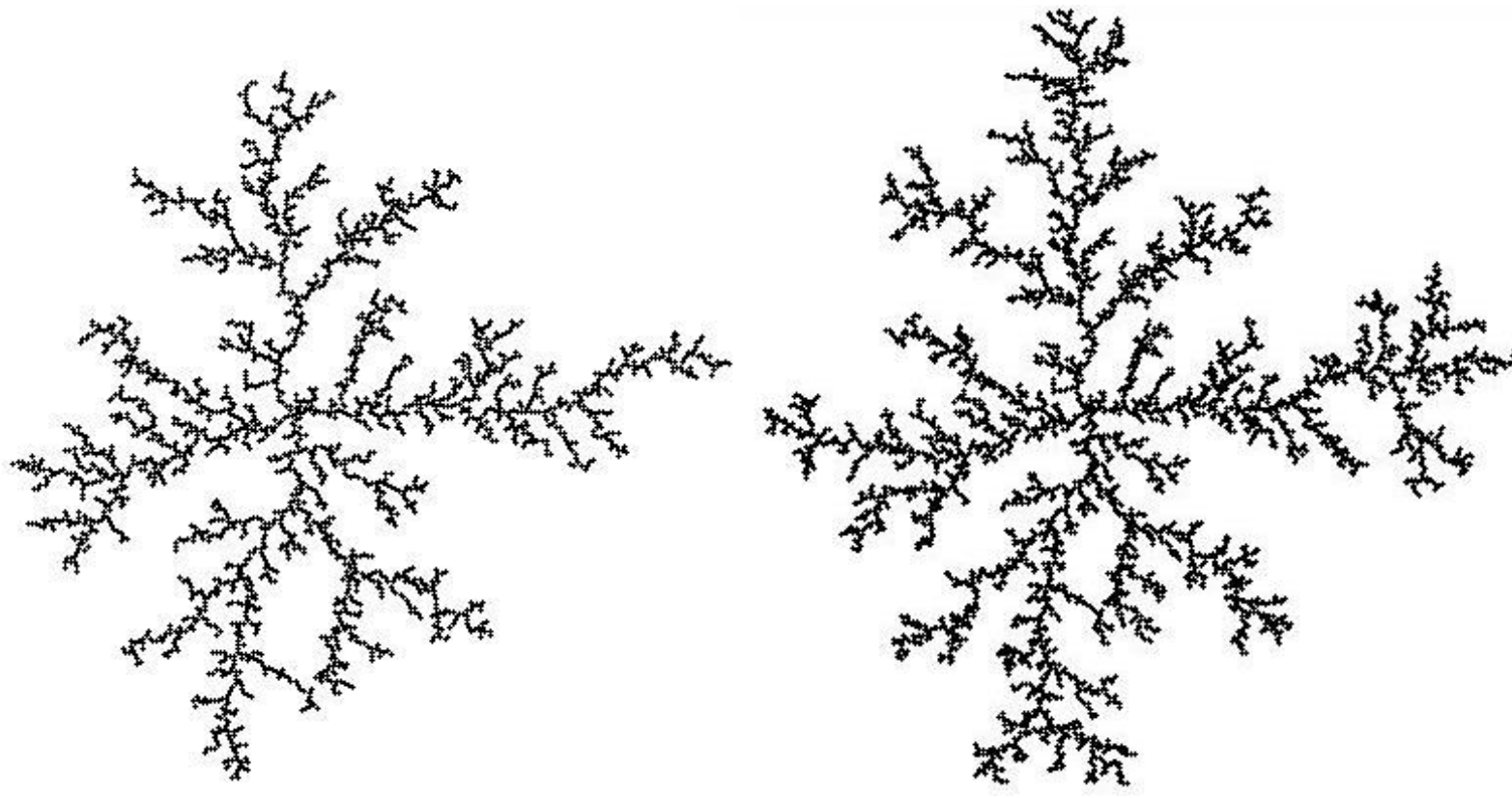


Random Walk – 10,000 Steps



- Este tipo de caminhada aleatória está intimamente relacionada ao movimento browniano, que é encontrado no movimento de partículas em líquidos e gases, e em ruído branco, comumente usado para descrever outros fenômenos gerados por caminhadas aleatórias.
- Depósitos eletroquímicos constituem um exemplo típico de movimento browniano.
 - Por exemplo, colocando-se uma solução de sulfato de zinco coberta com uma camada fina de n-butil-acetato em uma placa de Petri e aplicando-se uma corrente contínua ao conjunto, é possível investigar o crescimento de estruturas fractais de eletro-depósitos e suas mudanças morfológicas.
 - Experimentos desta natureza são importantes em ciências de polímeros, ciência dos materiais, imunologia e várias outras áreas.
- Um modelo matemático de depósito de zinco que apresenta um comportamento do tipo movimento browniano pode ser simulado com uma técnica denominada de DLA (*diffusion limited aggregation*):

- Gere uma grade quadrada de células;
- Fixe uma única célula, chamada semente, em uma dada posição da grade (p. ex. no centro);
- Selecione uma vizinhança de interesse centrada em torno da semente (p. ex. usando uma vizinhança de Moore) e introduza uma nova partícula em movimento nesta vizinhança;
- Se a partícula em movimento sair da vizinhança, então ela é substituída por uma nova aleatoriamente gerada; senão, se a partícula em movimento encontrar a semente ou alguma partícula agregada à semente, então ela se une ao cluster tornando-se uma partícula estática.



Resultados da aplicação de DLA (*diffusion limited aggregation*)

```
procedure [] = DLA(seed)
  C ← generate_C          //generate Grid
  Cr ← select_region(seed) //region around seed
  p ← new_particle(Cr)     //new particle within Cr
  cs ← 1                   //cluster size
  while not_stopping_criterion do,
    p ← move_particle      //moving particle
    if p meets cluster,
      then attach p to cluster
        cs ← cs + 1
        p ← new_particle(Cr)
    else
      if p leaves Cr
        p ← new_particle(Cr)
      end if
    end if
  end while
end procedure
```

Algoritmo 7.3: Um algoritmo simplificado de agregação por difusão limitada.

- Em uma dimensão, o movimento browniano é caracterizado por um processo aleatório $X(t)$, que corresponde a uma função X de uma variável real t (tempo), cujos valores são variáveis aleatórias $X(t_1)$, $X(t_2)$, ... , onde o incremento $X(t_2) - X(t_1)$ possui uma distribuição gaussiana e os incrementos quadráticos médios têm uma variância proporcional à diferença entre os tempos:

$$E[| X(t_2) - X(t_1) |^2] \propto |t_2 - t_1|.$$

- Os incrementos de X são estatisticamente auto-similares no sentido que:

$$X(t_0 + t) - X(t_0) \text{ e } \frac{1}{\sqrt{r}}(X(t_0 + rt) - X(t_0)),$$

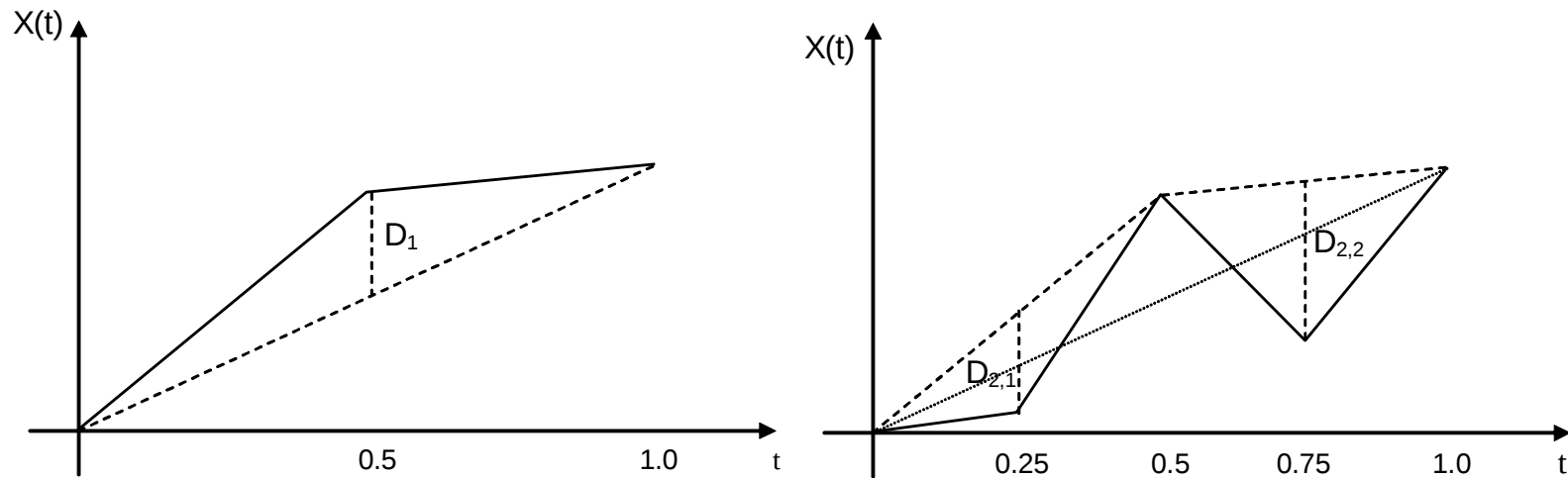
possuem as mesmas funções de distribuição conjuntas para quaisquer t_0 e $r > 0$.

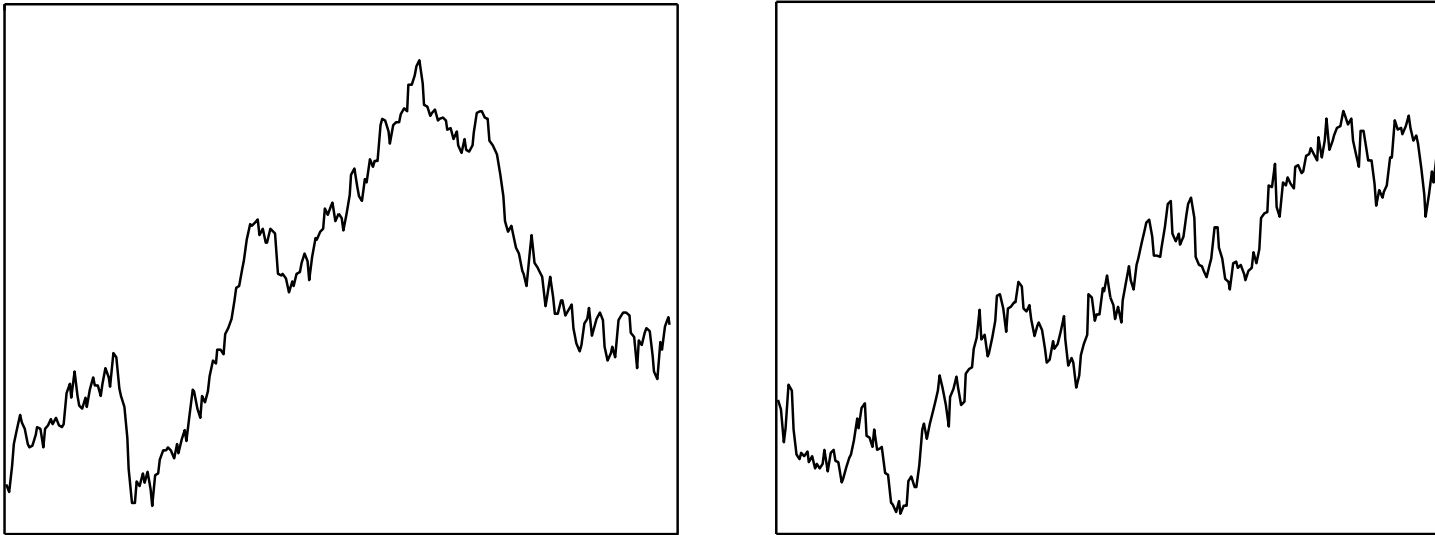
- Tomando, por exemplo, $t_0 = 0$ e $X(t_0) = 0$, ambas as funções aleatórias $X(t)$ e $\frac{1}{\sqrt{r}}(X(rt))$ são estatisticamente indistinguíveis; a segunda sendo uma versão apropriadamente re-escalada da primeira.

- Um método popular para gerar movimento browniano é conhecido como *algoritmo recursivo da subdivisão (recursive subdivision algorithm)*, também conhecido como *algoritmo do deslocamento aleatório do ponto médio (random midpoint displacement algorithm – RMD)*.
- O RMD opera como a seguir:
 - Se o processo $X(t)$ deve ser computado para o tempo $t \in [0, 1]$, então comece definindo $X(0) = 0$ e escolhendo $X(1)$ como uma amostra de um valor gaussiano de média 0 e variância σ^2 .
 - No primeiro passo, o ponto médio entre $t = 0$ e $t = 1$ é dado pela média entre $X(0)$ e $X(1)$, mais um desvio D_1 de média zero e variância Δ_1^2 :
$$X(1/2) - X(0) = \frac{1}{2}(X(1) - X(0)) + D_1.$$
 - Como uma amostra de uma variável aleatória gaussiana possui média 0 e variância σ^2 , é esperado que:
$$\text{var}(X(t_2) - X(t_1)) = |t_2 - t_1|\sigma^2.$$

- Além disso, a cada iteração o número de fragmentos dobra, e o desvio D_n no tempo n deve ter variância:

$$\Delta_n^2 = \frac{1}{2^{n+1}} \sigma^2.$$





6.2. Movimento Browniano Fracionário

- O termo *movimento browniano fracionário* (MBF) foi introduzido para se referir a uma família de funções gaussianas capazes de gerar modelos de séries temporais naturais.
- Muitas extensões e variações foram propostas para modelar uma vasta gama de fenômenos naturais, de paisagens montanhosas a nuvens.

- O MBF é uma generalização do movimento browniano definido como um processo aleatório $X(t)$ com incrementos gaussianos e $\text{var}(X(t_2) - X(t_1)) \propto |t_2 - t_1|^{2H}$, onde $0 < H < 1$.

- Neste caso genérico, os incrementos de X são estatisticamente auto-similares, com parâmetro H , no sentido que:

$$X(t_0 + t) - X(t_0) \text{ e } \frac{X(t_0 + rt) - X(t_0)}{r^H}$$

possuem as mesmas funções de distribuição conjuntas para quaisquer t_0 e $r > 0$.

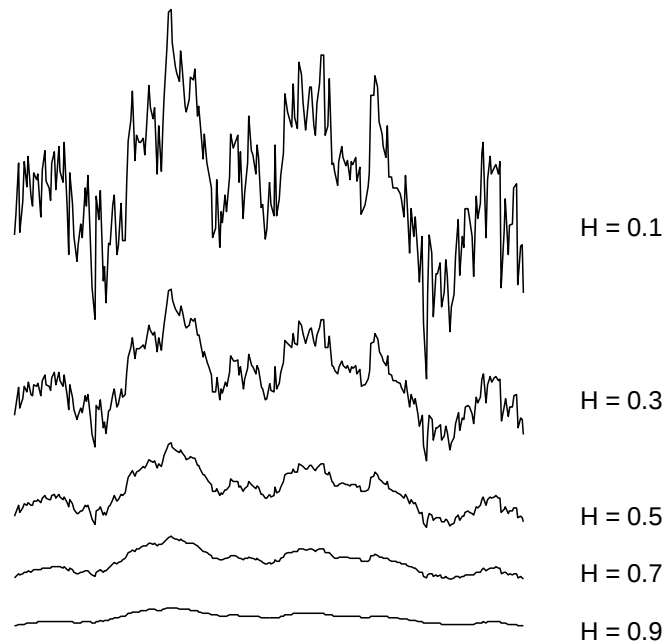
- Tomando-se, por exemplo, $t_0 = 0$ e $X(t_0) = 0$, ambas as funções aleatórias $X(t)$ e $\frac{1}{r^H}(X(rt))$ são estatisticamente indistinguíveis; a segunda sendo uma versão apropriadamente re-escalada da primeira.
- Para aplicar o algoritmo RMD ao caso mais geral de MBF é necessário que:

$$\text{var}(X(t_2) - X(t_1)) = |t_2 - t_1|^{2H} \sigma^2.$$

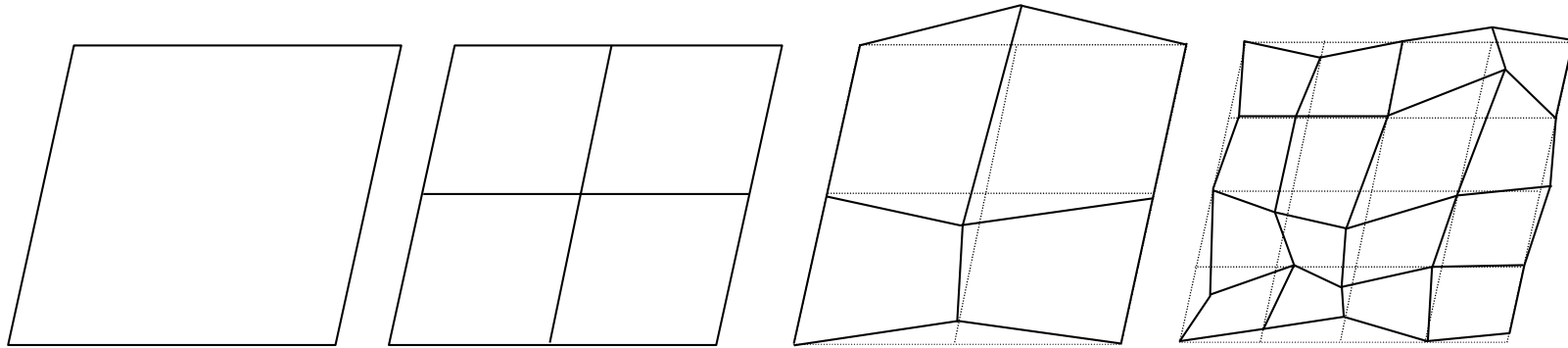
- O deslocamento do ponto médio torna-se então

$$\Delta_n^2 = \frac{\sigma^2}{(2^n)^{2H}} (1 - 2^{2H-2}).$$

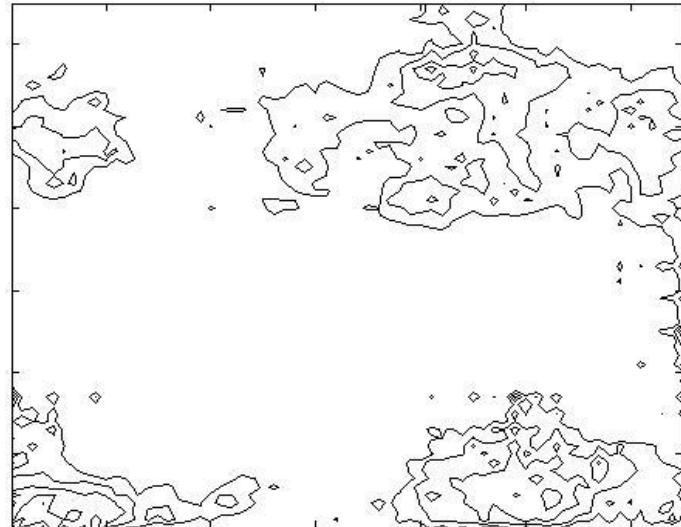
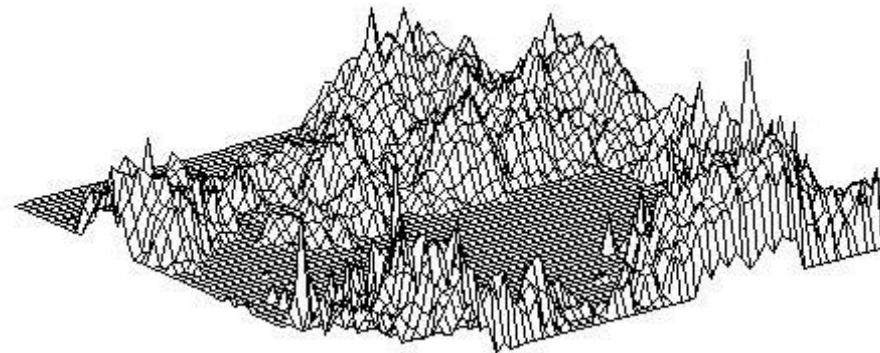
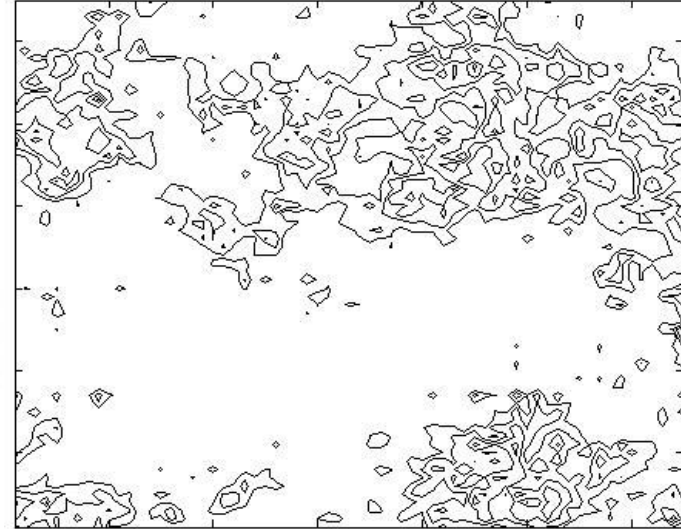
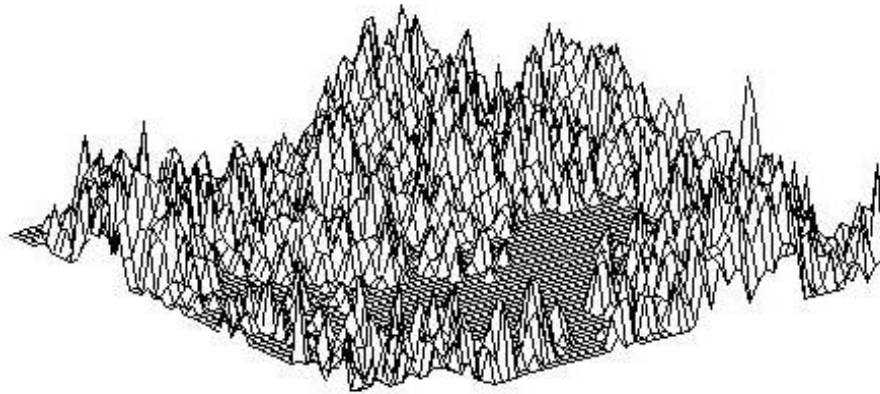
- O parâmetro H , denominado coeficiente de Hurst, descreve a ‘rugosidade’ da fração em pequenas escalas.



- A extensão deste algoritmo (RMD) para três ou mais dimensões é direta e resulta em algoritmos capazes de gerar paisagens montanhosas realistas.
- A ideia envolve aplicar o algoritmo RMD a uma grade até que uma determinada granularidade seja obtida.



- O algoritmo opera como a seguir:
 - Determine os pontos centrais da grade atual;
 - Perturbe verticalmente cada um dos novos vértices usando uma distribuição gaussiana apropriada;
 - Repita para cada novo quadrado reduzindo a perturbação a cada passo.



6.3. Escopo do MBF

- Muitos processos dinâmicos na natureza envolvem movimento browniano, do movimento das borboletas ao movimento de partículas em um líquido ou meio gasoso.
- Aplicações também são encontradas em economia (por exemplo, na simulação do comportamento dos preços de ações e outros bens), ecologia, ciência dos materiais, imunologia, etc.
- Em se tratando de computação gráfica, MBF pode ser usado para simular o leito de rios, nuvens, costas marítimas, dobramento e desdobramento de materiais (p. ex. papel), rachaduras em concretos e pavimentos, projeto de simuladores de vôo, geração de tráfego e animação comportamental em geral.



© K. Musgrave, Pandromeda.com

7. Sistemas de Partículas

- Alguns fenômenos naturais envolvem a participação de diversos elementos básicos (partículas) como, por exemplo, o movimento de líquidos, gases, nuvens, explosões e fogos.
- Os *sistemas de partículas* (*particle systems*) podem ser usados para modelar uma grande variedade de fenômenos desta natureza.
- Um sistema de partículas consiste basicamente de uma coleção de partículas (objetos) com algumas propriedades fundamentais e algumas regras de comportamento que elas devem seguir.
- A definição precisa destas propriedades e leis vai depender do fenômeno que se deseja modelar.

7.1. Conceitos Básicos

- Uma partícula é um ponto no espaço, geralmente tri-dimensional, que possui uma série de atributos (propriedades) como posição, velocidade, cor, tempo de vida, tamanho e transparência.
- Um sistema de partículas (PS) é uma coleção de partículas que, em conjunto, representam um objeto.
- As partículas mudam dinamicamente com o tempo em função de forças externas ou outros processos.
- Os sistemas de partículas foram introduzidos por Reeves em 1983 para modelar o que ele denominou de objetos *fuzzy*, como nuvens, fumaça, água e fogo.
- A representação de sistemas de partículas difere em três aspectos básicos da representação geralmente utilizada em síntese de imagens:

- Um padrão ou forma é representado por uma coleção de partículas (objetos primitivos) que irão definir seu volume ao invés de serem representados por um conjunto de elementos primitivos de superfície;
- As partículas são dinâmicas, ou seja, elas podem se mover, nascer e morrer;
- Um padrão representado por partículas não é determinístico, ou seja, sua forma não é completamente pré-especificada.
- Algumas propriedades interessantes:
 - Uma partícula é uma primitiva muito simples;
 - A definição do modelo é procedural e envolve processos estocásticos;
 - Os sistemas de partículas modelam padrões e formas dinâmicas, como queimadas e nuvens em movimento;
 - O nível de detalhes pode ser facilmente controlado regulando, por exemplo, a quantidade de partículas.

7.2. Modelo Básico

- Em um determinado período de tempo, partículas são geradas e inseridas no sistema, se movem e mudam dentro do sistema, morrem e são removidas do sistema.
- Os atributos das partículas e suas regras comportamentais dependem da aplicação.
- Na proposta original de PS, Reeves aplicou um sistema de partículas para gerar uma parede de fogo que foi utilizada no filme *Star Trek II: The Wrath of Khan*.
- Neste caso, as partículas possuíam os seguintes atributos:
 - Posição e Velocidade
 - Cor e Transparência
 - Tempo de vida ou idade
 - Forma
 - Tamanho

- O procedimento para computar cada frame (intervalo de tempo) em uma sequência envolve os seguintes passos:
 - Geração de partículas
 - Determinação dos atributos das partículas
 - Morte de partículas
 - Atualização das partículas
 - Renderização das partículas

```
procedure [] = PS(max_it, d, o)
  initialize X; //generate particles and assign their attributes
  t ← 1
  while t < max_it do,
    X ← destroy(X, d)           //destroy all particles older than d
    X ← dynamics(X, o)         //change each remaining particle
    render(X)                  //draw (render) particles X and plot
    initialize X'               //generate new particles
    X ← insert(X, X')          //insert the new particles into X
    t ← t + 1
  end while
end procedure
```

Algoritmo 7.4: Procedimento para implementar um sistema de partículas convencional. (Nota: Muitos parâmetros são necessários em funções como *initialize* e *render*.)

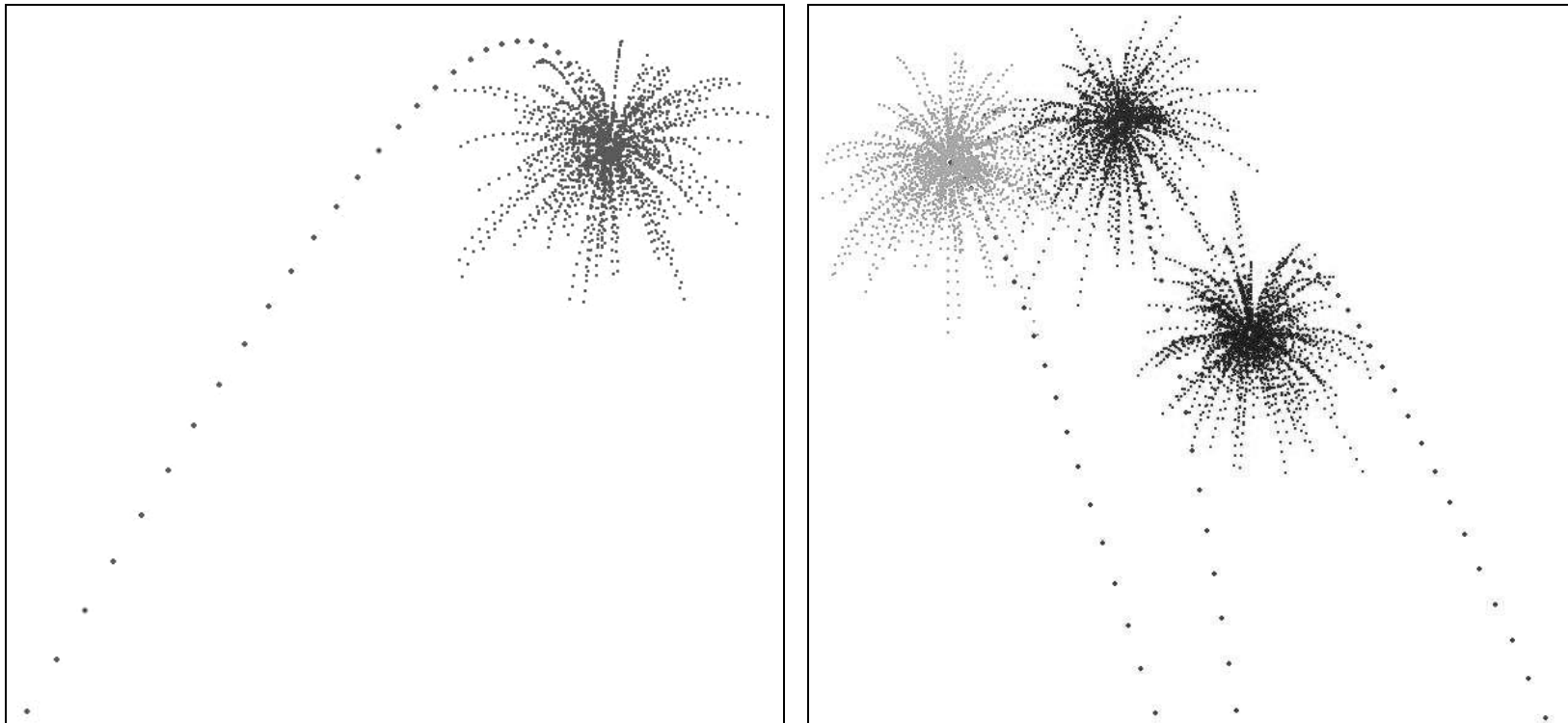
7.3. Simulando Fogos de Artifício

- Uma partícula será lançada e explodirá após um determinado tempo de voo.
 - A explosão gera um conjunto de novas partículas.
 - Todas as partículas estão sujeitas à força da gravidade.
- Atributos das partículas:
 - Posição: $\mathbf{p} = (x,y)$.
 - Velocidade: (v,θ) , v = velocidade e θ = ângulo de disparo.
 - Cor: a cor inicial pode ser diferente da cor após a explosão e durante a queda.
 - Tempo de vida: a partícula inicial ‘vive’ até que ela exploda, momento em que novas partículas são geradas e têm seus períodos de vida definidos.
 - Forma: todas as partículas podem ter a mesma forma.
 - Tamanho: após a explosão, as partículas terão a metade do tamanho da partícula antes da explosão.
 - Transparência: ‘ao gosto do freguês’.

$$\mathbf{v} = \mathbf{v} + \mathbf{a}$$

$$\mathbf{p} = \mathbf{p} + \mathbf{v}$$

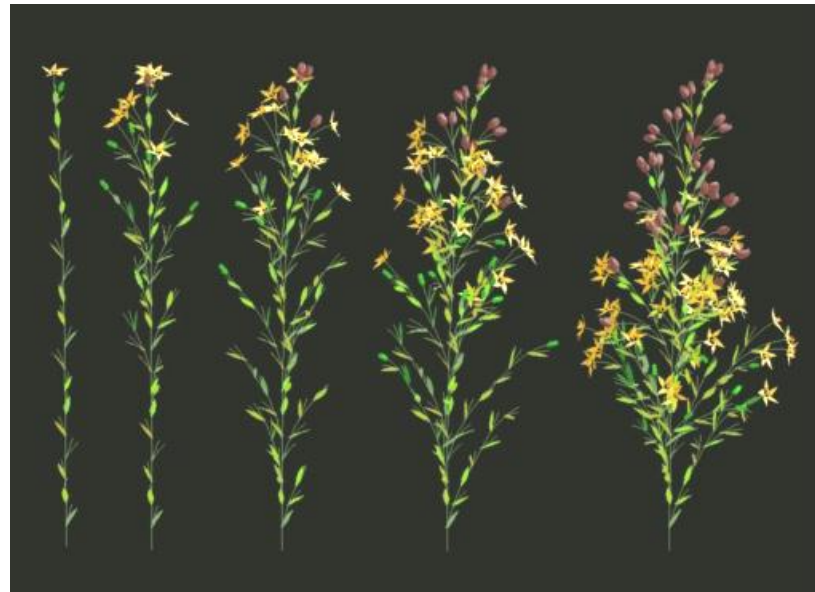
onde $\mathbf{v}$ é a velocidade da partícula e $\mathbf{a} = (0, -a)$ é a força atuando sobre a partícula.



7.4. Escopo dos Sistemas de Partículas

- Sistemas de partículas têm sido usados para modelar quedas d'água, ondas, fogo, nuvens, fontes d'água, fogos de artifício, explosões, cardumes de peixes, estrelas, arco-íris, dinâmica de fluidos, plantas, árvores, florestas, grama, furacões, tempestades de areia, avalanches.

8. Da Geometria Natural à Geometria Fractal





9. Escopo da Geometria Fractal

- Já foi discutido neste curso que a natureza é cheia de “regras”, e agora verificamos que estas regras envolvem irregularidade e auto-similaridade: a natureza possui uma geometria fractal.
- Curvas como o conjunto de Cantor e a curva de Koch são comuns na natureza e no nosso dia-a-dia.
- Variações do conjunto de Cantor ocorrem, por exemplo, em frequências de palavras e letras em linguagem e em ruídos em linhas telefônicas.
- As curvas de Koch servem de modelo para nuvens, costas marítimas, etc.
- Organismos também são fractais. Nossos pulmões, sistema circulatório, cérebro, rins e vários outros sistemas e órgãos são fractais.
 - O escopo e a importância dos fractais e da geometria fractal vão mais além: Incêndios em florestas possuem fronteiras fractais; fractais estão sendo

utilizados para modelar a dinâmica do HIV; fractais vêm sendo usados em Economia.

- Sistemas-L e sistemas de funções iterativas são úteis para o desenho e modelagem de paisagens naturais, ornamentais e ilustrações botânicas.
- Outras aplicações:
 - Reconstrução de espécies extintas de plantas
 - Identificação de respostas de plantas a ataques
 - Desenvolvimento de modelos estruturais de plantas integrados com ecossistemas complexos
 - Síntese de conchas e outros padrões naturais
 - Modelagem de processos de desenvolvimento e crescimento (sistemas-L)
 - Classificação de padrões de ramificação em plantas e animais
 - Predição de colheitas
 - Modelagem de fractais

- Descrição de inflorescências
- Aplicações rurais diversas
- Em particular, os sistemas de funções iterativas têm tido muito sucesso no processo de compressão de dados (imagens): compressão fractal.
- Sistemas de partículas têm sido usados também como descanso de tela de computador e em diversos filmes e desenhos animados, como *Tornado* e o *Príncipe do Egito*.

10. Ask nature

- Criado pelo Biomimicry Institute (<http://biomimicry.net/>), o site a seguir é um banco de dados de soluções inspiradas na natureza (de estratégias naturais para otimização de recursos a produtos bio-inspirados):

<http://www.asknature.org>