# MS211 - Cursão - 2010

Quinto Exercício Programa

# Random Walks

**Resumo**

Neste EP, será introduzido o conceito de seqüência (pseudo)-aleatória e serão discutidos alguns algoritmos para sua geração. São vastas as aplicações das seqüências aleatórias. Desconfio que, se pegarmos aleatoriamente qualquer simulação numérica moderna, encontraremos algum uso de seqüências aleatórias... Ilustraremos o uso de geradores de números aleatórios com duas aplicações "modernas": integração de Monte Carlo e modelos DLA (*Diffusion Limited Aggregation*), este último bastante eficiente para o entendimento do crescimento de diversas estruturas naturais, desde cristais até recifes de corais. Casualmente, é também um ótimo gerador de figuras fractais interessantes para a nossa galeria...

## 1 Números Aleatórios

*Anyone who considers arithmetical*
*methods of producing random digits*
*is, of course, in a state of sin.*
*– John von Neumann (1951)*
*(de acordo com Knuth, vol. II)*

O que significa aleatório? Como caracterizar um evento aleatório? Eles existem realmente? E uma seqüência aleatória? Estas são questões sutis, que serão abordadas aqui de uma maneira bastante superficial. Espera-se, porém, que os interessados se aprofundem no estudo destes problemas a partir do que será discutido neste EP. Como sempre, a principal referência (e inspiração) é o livro do Knuth, vol II.

Intuitivamente, associamos a idéia de aleatoriedade à de imprevisibilidade. Isto é razoável. Etimologicamente, aleatório vem de *alea*, que no Latim

significa sorte (ou azar[1]...). Um evento aleatório seria, portanto, um evento que ocorreu por acaso, sem causa evidente, sem uma lógica inerente que permitisse sua previsão. Há profundas discussões filosóficas sobre a possibilidade de existência de eventos desta natureza. (Lembrem-se do demônio de Laplace e do livre-arbítrio.) Por ora, vamos admitir que este tipo de evento possa existir e que de fato exista. Assim, podemos definir uma seqüência aleatória como uma sucessão de números escolhidos completamente ao acaso, sem nenhuma relação entre eles, sem nenhuma lógica inerente que permita cálculos antecipados. Atire um dado (honesto) e registre o valor obtido numa seqüencia, por exemplo, $\{1, 4, 6, 2, 3, 1, 5, 4, \ldots\}$. O que podemos afirmar sobre esta seqüencia? Primeiro, já que se trata de uma seqüência aleatória, não deve haver, obviamente, nenhuma lei de recorrência do tipo $S_{k+1} = f(S_k, S_{k-1}, \ldots)$. A única coisa que podemos afirmar sobre o elemento $S_k$ da seqüencia é que ele pode ser $1, 2, 3, 4, 5$ ou $6$, com igual probabilidade $1/6$. Se fizermos um histograma indicando a freqüência com que os diferentes dígitos aparecem na seqüencia para $k \to \infty$, devemos obter exatamente $1/6$ para cada um deles. É fundamental notar que apenas esta propriedade do histograma não é suficiente para se caracterizar uma seqüência aleatória. Considerem, por exemplo, a seqüência $\{1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5, 6, \ldots\}$. Ela pode ser qualquer coisa, menos aleatória! Trata-se de uma seqüência **periódica**, para a qual o histograma definido acima também indicará $1/6$ como freqüência para cada dígito. A não aleatoriedade é clara, entre outras coisas, porque há uma fórmula de recorrência, esta[2]:

$$S_{k+1} = S_k \bmod 6 + 1, \tag{1}$$

a partir da qual pode-se prever toda a seqüência usando-se como condição inicial $S_k = 1$, algo obviamente impossível para uma seqüência verdadeiramente aleatória. **Nenhuma** seqüência periódica pode ser realmente aleatória. Como, então, saber se uma seqüência candidata a ser aleatória não é, na verdade, uma seqüência com um período muito longo? Obviamente, só tem sentido afirmar que uma seqüência não é periódica com um período arbitrariamente longo se ela for *infinita*.

[1] Segundo o Houaiss, azar vem do árabe *al-zahr*, que significa literalmente "a flor". Os dados recebiam este nome pois, comumente, tinham uma das faces pintadas com uma flor. A acepção de má sorte surgiu posteriormente.

[2] Dois inteiros $a$ e $b$ são ditos congruentes módulo $c$ se a diferença $a - b$ for um múltiplo de c, *i.e.*, $a = b \bmod c$ é equivalente a dizer que $a - b = nc$, para algum inteiro n.

Ora, mas o que é um computador se não um dispositivo que somente realiza certas operações pré-estabelecidas? E como esperar de um dispositivo desta natureza *qualquer* resposta aleatória? Não é difícil perceber que estamos diante de um paradoxo. Computadores só fazem o que são mandados. Em particular, se forem solicitados a fazer duas vezes, em instantes de tempo diferentes, a mesma tarefa, elas serão concluídas de maneira idêntica. Suponham que a tarefa fosse "crie uma seqüência aleatória". Seriam criadas duas seqüências idênticas, a partir de um certo "programa", indicando, claramente, que elas não são verdadeiramente aleatórias, acarretando o paradoxo. Esta é a essência da epígrafe desta seção, atribuída a J. von Neuman.

Rigorosamente, num computador, o máximo que podemos obter é uma seqüencia pseudo-aleatória. Há definições precisas para todos estes termos. Usaremos aqui, porém, apenas as idéias intuitivas. Uma seqüencia pseudo-aleatória é uma seqüência que, para determinados fins, se comporta como uma seqüência verdadeiramente aleatória. É comum na literatura abandonar-se o prefixo pseudo-, o que quase sempre acaba em confusão. Obviamente, não sou uma exceção... Há diversos algoritmos para a geração de seqüências pseudo-aleatórias. Os mais simples e difundidos, curiosamente, estão baseados em regras semelhantes a (1), a qual gerou nosso exemplo explícito de seqüência não-aleatória! Consideremos, por exemplo, uma seqüência do tipo congruência linear

$$S_{k+1} = (aS_k + b) \bmod c. \tag{2}$$

Analisemos a seqüência obtida com $a = 5$, $b = 3$, $c = 8$ e $S_1 = 1$:

$$\{1, 0, 3, 2, 5, 4, 7, 6, 1, 0, 3, 2, 5, 4, 7, 6, \ldots\}$$

Trata-se, claramente, de uma seqüência periódica. Os valores se repetem a cada 8 elementos. Obviamente, não se trata de uma seqüência aleatória. Porém, a maneira com que os 8 dígitos (de 0 a 7) aparecem num período, para vários efeitos práticos, pode ser considerada como imprevisível. Uma seqüência deste tipo, para ser útil como uma seqüência pseudo-aleatória, deve ter um período grande. Não é difícil perceber que o período obtido será, no máximo, igual a $c$. Numa seqüência de período máximo (chamada ciclo-completo ou *full cycle*), cada dígito entre 0 e $c - 1$ pode aparecer uma só vez em cada período, garantindo-se, portanto a mesma distribuição para todos os dígitos no limite de $k \to \infty$. Pode-se garantir que serão sempre obtidos ciclos completos e longos a partir de (2) se as constantes $a$, $b$ e $c$ forem cuidadosamente escolhidas. (Vejam Knuth, Seção 3.1). A Tabela 1 mostra algumas escolhas comuns para estas constantes.

| Implementação | $a$ | $b$ | $c$ | $S_1$ |
|---|---|---|---|---|
| IBM RANDU | 65539 | 0 | $2^{31}$ | 123456789 |
| Congruência Linear mínima | 16807 | 0 | $2^{31}-1$ | 1 |
| função `rand()` linguagem C (ANSI) | 1103515245 | 12345 | $2^{31}$ | 12345 |
| Numerical Recipes | 1664525 | 1013904223 | $2^{32}$ | 1 |
| CRAY | 44485709377909 | 0 | $2^{48}$ | 1 |
| Maple | 427419669081 | 0 | $10^{12}-11$ | 1 |

Tabela 1: Escolhas típicas para geradores de números aleatórios baseados em congruência linear, veja Eq. (2). Notem que o maior número que pode ser armazenado num `unsigned int` da linguagem C é $2^{32}-1 = 4294967295$. O algoritmo da Congruência Linear mínima (Minimal Standard LCG), foi introduzido em S.K. Park e K.W. Miller, *Random Number Generators: Good Ones Are Hard To Find*, Communications of the ACM, **31**, 1192 (1988) e é usada com padrão mínimo de qualidade para geradores pseudo-aleatórios. A implementação RANDU da IBM foi muito popular nos anos 60. Algum tempo após seu lançamento, descobriu-se que ela continha uma falha quase elementar. Muitos trabalhos da época estão ainda sob suspeita. De acordo com D. Knuth, *...its very name RANDU is enough to bring dismay into the eyes and stomachs of many computer scientists!*

Muitas aplicações práticas requerem seqüências pseudo-aleatórias reais $U_k$, com $0 \leq U_k < 1$ e distribuição uniforme, o que significa que a probabilidade de $U_k \in [x, x+\Delta x] \in [0,1)$ é dada por $\Delta x$. Uma seqüência deste tipo pode ser facilmente obtida de (2) escolhendo-se $U_k = S_k/c$. Há diversos testes estatísticos que podem ser feitos para caracterizar quantitativamente o "grau de aleatoriedade" de uma seqüência pseudo-aleatória. Os interessados devem consultar o livro do Knuth e também os Numerical Recipes. Uma vez mais, ficaremos apenas com algumas idéias intuitivas. Suponham que usássemos o gerador de seqüências pseudo-aleatórias para escolher, aleatoriamente, pontos $(x_k, y_k)$ tais que $0 \leq x_k, y_k < 1$. Que tipo de imagem esperaríamos encontrar quando marcássemos estes pontos no plano $(x, y)$? A resposta está na Figura 1. Talvez seja instrutivo também darmos uma olhada em um caso em que as coisas não funcionam tão bem. Vejam a Fig. 2.

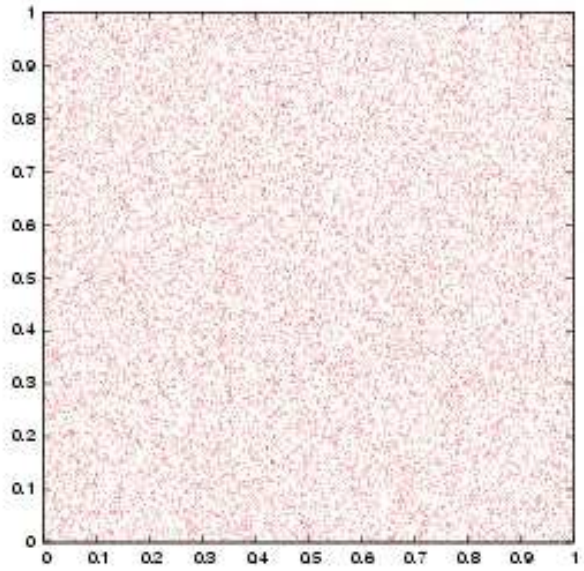


Figura 1: Imagem correspondente a 50.000 pontos gerados a partir da congruência linear mínima (veja Tabela 1). A ausência de qualquer estrutura evidente é uma indicação de que os pontos se comportaram de uma maneira "bastante aleatória". Os pontos foram gerados pelo programa `lcg.c`, disponível no repositório. O gráfico foi feito com o gnuplot e os comandos usados estão anexados no final do programa.

O caso do "desastre" da implementação RANDU da IBM dos anos 60 merece um destaque especial. Esta implementação foi usada nos pacotes científicos preparados pela IBM para o histórico e notável[3] System/360. Este sistema foi muito difundido, tornando-se um padrão no mundo das ciências e dos negócios. Desgraçadamente, o gerador RANDU também se difundiu com o sistema. Podemos repetir o teste do plano para o RANDU, usando os dados da Tabela 1. A imagem obtida será como a da Fig. 1, sem nenhuma estrutura evidente. Obviamente, este teste, dentre outros, foi feito pelo pessoal da IBM. Porém, podemos também questionar o que deveríamos esperar de uma seqüência aleatória se a usássemos para escolher pontos não no plano, mas no espaço. Em outras palavras, que imagem deve-se esperar de pontos

[3]Eu vi (e mexi) num!

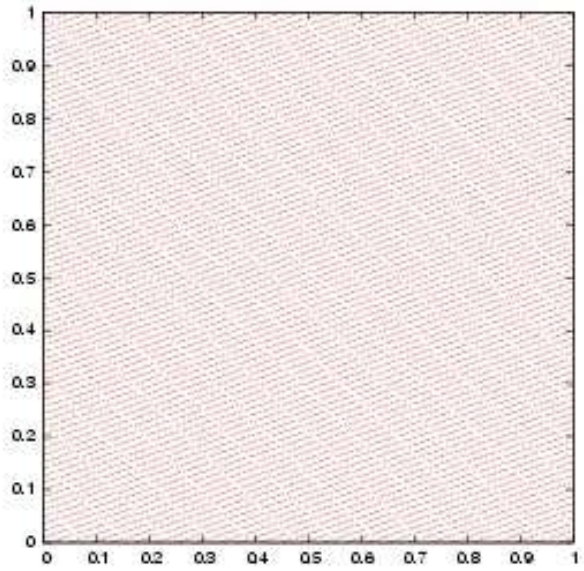


Figura 2: Imagem correspondente a 50.000 pontos gerados a partir de uma (péssima) congruência linear com $a = 1277$, $b = 0$ e c=131072. Ao contrário da Fig. 1, há aqui uma estrutura evidente. Os pontos dispõem-se ao longo de certas retas, claramente indicando algum efeito periódico, alguma correlação entre pontos subseqüentes, mesmo tratando-se de "apenas" 50.000 pontos, menos da metade do valor de $c$.

$(x_k, y_k, z_k)$ tais que $0 \leq x_k, y_k, z_k < 1$, escolhidos aleatóriamente? Por analogia ao que ocorreu no plano, se os pontos forem escolhidos aleatóriamente, deveríamos obter uma ocupação homogênea de um cubo de aresta 1, sem nenhuma estrutura aparente, certo? Vejam, agora, a Fig. 3. Há uma estrutura óbvia, ligada a uma forte (e inesperada) correlação que ocorre acada 3 pontos dada por

$$S_{k+2} = 6S_{k+1} - 9S_k.$$

Vejam algumas citações curiosas sobre este fato no artigo correspondente da Wikipedia.

Já temos condições de implementar um gerador de números pseudo-aleatórios. Porém, fica a impressão que somente é possível gerar seqüências periódicas, mesmo que seus períodos sejam arbitrariamente longos, limitados pelo valor de $c$. Como já sabemos utilizar a aritmética de precisão ar-

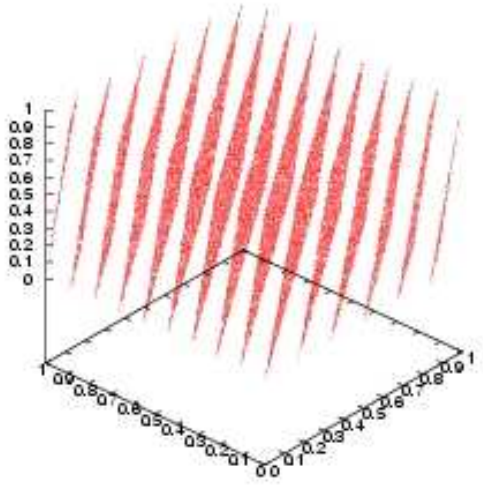


Figura 3: Imagem tri-dimensional correspondente a 50.000 pontos gerados a partir da implementação RANDU. Os pontos ocupam apenas 15 planos contidos no cubo! Um péssimo comportamento "aleatório". Novamente, de acordo com D. Knuth, *...its very name RANDU is enough to bring dismay into the eyes and stomachs of many computer scientists!*

bitrária (MPA), podemos, em princípio, gerar seqüências com período arbitrariamente grandes usando congruência linear. Contudo, podemos fazer algo ainda melhor e, de certa forma, surpreendente. É possível gerar seqüências não-periódicas (ou de período infinito) explorando-se os sistemas caóticos. Podemos, por exemplo, considerar o nosso conhecido mapa logístico

$$x_{k+1} = 4\lambda x_k(1 - x_k), \tag{3}$$

$x_k \in [0, 1]$ e $\lambda \in (0, 1]$. Este mapa do intervalo $[0, 1]$ nele mesmo é um protótipo de sistema caótico (Mais detalhes, no livro M. Tabor, *Chaos and Integrability in Non-Linear Dynamics*). O comportamento deste sistema é determinado pelo valor da constante $\lambda$. A dinâmica caótica aparece para $\lambda = 1$. Para este valor de $\lambda$, introduzindo-se a nova variável $x_k = (y_k + 1)/2$, temos a partir de (3)

$$y_{k+1} = 1 - 2y_k^2, \tag{4}$$

com $y_k \in [-1, 1]$. Introduzindo uma outra variável $y_k = \sin\theta_k$, obtemos o seguinte mapa, equivalente a (3) para $\lambda = 1$,

$$\sin\theta_{k+1} = \cos 2\theta_k,$$

com $\theta_k \in [0, 2\pi]$. Este mapa, porem, nada mais é do que uma transformação linear

$$\theta_{k+1} = \left(2\theta_k + \frac{\pi}{2}\right) \bmod 2\pi. \tag{5}$$

Podemos agora apresentar um argumento sugerindo que a seqüência definida por (5) é essencialmente não-periódica. De fato, mostraremos que a seqüência, salvo para condições iniciais especiais, tem as mesmas propriedades estatísticas dos dígitos de $\pi$. A equação (5) tem como solução[4]

$$\theta_k = \left(2^k\theta_0 + (2^k - 1)\frac{\pi}{2}\right) \bmod 2\pi,$$

que para $k > 2$ pode ser escrita como

$$\theta_k = \left(2^k\theta_0 - \frac{\pi}{2}\right) \bmod 2\pi = 2\pi \,\mathrm{Frac}\, \left(\frac{2^{k-1}\theta_0}{\pi} - \frac{1}{4}\right),$$

sendo Frac a parte fracionária. Qualquer periodicidade da seqüência deve vir da parte fracionária. Supondo, por exemplo, que $\theta_0$ seja racional, é facíl, graças ao fato de $\pi$ ser irracional, perceber que não haverá periodicidade. De fato, só haverá periodicidade se $\theta_0$ for uma fração racional de $\pi$. Para a imensa maioria de condições iniciais teremos, portanto, um comportamento não periódico que, como veremos, pode dar origem a um ótimo gerador de números pseudo-aleatórios.

Há uma sutileza quanto à distribuição dos números pseudo-aleatórios gerados a partir do mapa logístico. É claro que a seqüência $\theta_k$ dada por (5) está uniformemente distribuída no intervalo $[0, 2\pi]$, mass esse não é o caso da seqüencia original $x_k$ dada por (3). Tudo pode ser resolvido, porém, usando-se

$$\begin{aligned} x_{k+1} &= 4x_k(1 - x_k), \\ u_{k+1} &= \frac{1}{\pi}\cos^{-1}(1 - 2x_{k+1}). \end{aligned} \tag{6}$$

[4]Prove!

Para qualquer $x_0$ diferente de 0, 1/4, 1/2, 3/4 ou 1, a seqüência $u_k$ será não periódica e homogênea[5] no intervalo $[0, 1]$. A seqüência $x_k$ tende a se aglomerar nos limites do intervalo. A probabilidade de $x_k$ estar próximo de 0 ou próximo de 1 é maior do que estar próximo de 1/2, por exemplo. Vejam as Fig. 4 e 5, geradas a partir do programa `lm.c`. **Leiam** com atenção os

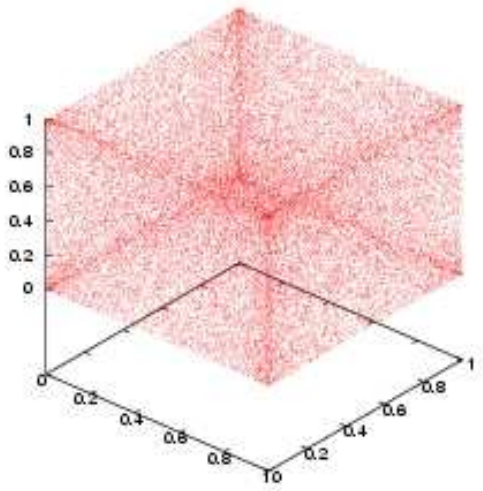


Figura 4: Imagem tri-dimensional correspondente a 50.000 pontos gerados a partir da iteração do mapa logístico (3). Pode-se ver claramente que a seqüência $x_k$ tende a se aglomerar nos limites do intervalo.

comentários do programa, há sutilezas.

Este EP contem duas partes, que serão expostas logo a seguir. Para atacá-las, vocês deverão usar geradores pseudo-aleatórios. Vocês deverão implementar os seus, não vale usar a função `RAND()`, mesmo sabendo como ela funciona. Podem usar qualquer gerador, até mesmo inventar um, desde que ele passe nos testes da distribuição homogênea plana e espacial. Em caso de dúvida e/ou algum impasse, use a congruência linear mínima. Um ótimo exercício de aquecimento seria implementar o padrão do Numerical Recipes (Ver tabela 1). Note, porem, que o valor de $c$ é maior do que o máximo que pode ser armazenado num `unsigned int`. Isto pode ser resolvido sem

[5]Para mais detalhes, aqui.

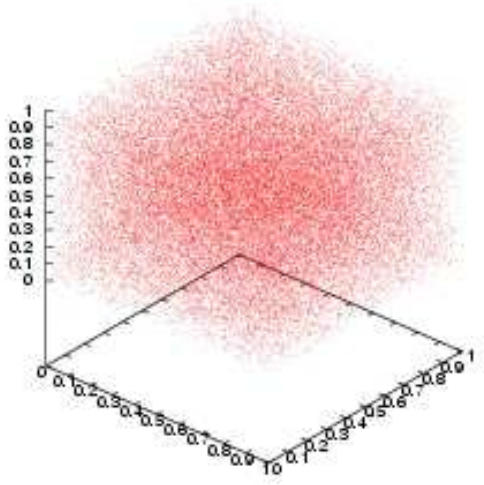

Figura 5: Imagem tri-dimensional correspondente a 50.000 pontos gerados a partir da iteração do mapa logístico (6). A distribuição é homogênea, como esperado.

a introdução de esquemas de MPA. Tentem. As seqüências não periódicas como a gerada a partir do mapa logístico devem ser usadas sempre que o número de pontos for muito alto, quando correríamos o risco de ultrapassar o período dado por $c$. Tipicamente, é menos custoso computacionalmente implementar um gerador pseudo-aleatório baseado no mapa logístico do que uma congruência linear usando MPA.

# 2 Integração de Monte Carlo

Os algoritmos de integração de Monte Carlo se baseiam em uma idéia simples e intuitiva. A implementação de uma versão simples destes algoritmos é a primeira parte deste EP. Suponha que você tem uma função suave $f(x)$ e que deseja calcular a integral definida $S = \int_a^b f(x)dx$, a qual corresponde a área sob a curva $f(x)$, veja Figura 6. Sorteie N pontos na região $[a, b] \times [0, c]$ do plano $(x, y)$, distribuídos de maneira homogênea, como na Figura 1. É

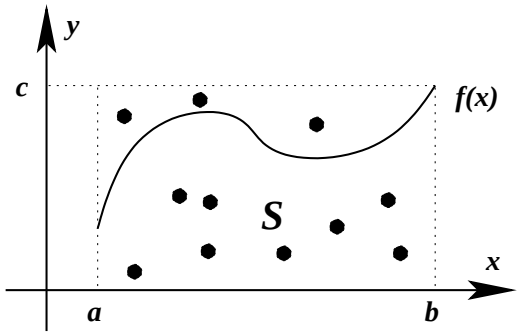


Figura 6: Integração de Monte Carlo. Mais referências, aqui.

natural esperar que

$$\lim_{N\to\infty} \frac{m(N)}{N} = \frac{S}{c(b-a)}, \tag{7}$$

sendo $m(N)$ o número de pontos que caem abaixo da curva $f(x)$. Este resultado pode ser provado, vejam as referências pertinentes aqui. Escolhendo-se um $N$ grande, tipicamente tem-se um erro pequeno nestes algoritmos. Uma versão mais moderna do algoritmo de Monte Carlos, mais robusta com relação a erros, é chamada de VEGAS[6]. Obviamente, estes método requerem gerados de números aleatórios confiáveis e de ciclo longo, já que boas precisões certamente irão requerer muitos pontos. São três as atividades desta primeira parte do EP:

## 2.1 Cálculo de $\pi$

Seja o quarto de circunferência unitária $f(x) = \sqrt{(1-x^2)}$, $x \in [0,1]$. Sabe-se que $S = \int_0^1 \sqrt{(1-x^2)}\, dx = \pi/4$. Use o método de Monte Carlo para estimar o valor de $\pi$. Como sempre, as melhores aproximações serão premiadas...[7]

[6] São todas referências a cassinos, obviamente. Físicos (e matemáticos) mostram com freqüência interesse por jogos e cassinos, vejam o processo Urca, por exemplo

[7] Para comparações, ai vão 20 casas decimais: $\pi = 3.14159265358979323846\ldots$

## 2.2 Cálculo de $e$

O número $e$ é tal que

$$S(e) = \int_1^e \frac{dx}{x} = 1.$$

Das propriedades elementares da hipérbole, sabe-se que $2 < e < 3$. Use o método de Monte Carlo para implementar a função $s(y) = \int_1^y \frac{dx}{x}$ (trata-se, obviamente, da função $\ln y$). Note que, para $y = e$, $s(y) = 1$.[8]

## 2.3 A tragédia anunciada: calculando-se volumes com o RANDU

Como vimos, o problema mais sério da implementação do RANDU da IBM eram as correlações da Fig. 3. Implemente um algoritmo do tipo Monte Carlo para se calcular volumes. Encontre um sólido cujo volume calculado via Monte Carlo com o gerador RANDU difira em mais de 20% do seu valor real.

# 3 O modelo DLA

O modelo DLA (*Diffusion Limited Aggregation*) é usado para descrever certos processos de crescimento por agregação. É o que acontece, por exemplo, com certos cristais de sais que crescem em torno de impurezas. Considerem uma certa quantidade de algum sal dissolvida na água. Se a concentração for baixa, pode-se supor que as pequenas partículas dissolvidas vagam pela água num movimento aleatório, o nosso já conhecido movimento Browniano. Considerem agora que há alguma impureza, muito mais pesada que as partículas, nessa solução. Por ser pesada, a impureza está em repouso. Se, por acaso, algumas das partículas de sal que vagam pela solução encostar na impureza, acabará se ligando a ela de maneira definitiva. Desta maneira, novas partículas vão se agregando, formando uma estrutura com certas propriedades que dependem da impureza e das partículas, como a mostrada na Fig. 7. Este processo não ocorre só para cristais, mas também em diversas estruturas biológicas.

Nosso modelo será um modelo plano, em $\mathbb{R}^2$. O plano será discretizado com, por exemplo, $500 \times 500$ pontos. Um par de inteiros $(i, j)$ localizará

[8] Para comparações, ai vão 20 casas decimais: $e = 2.71828182845904523536\ldots$

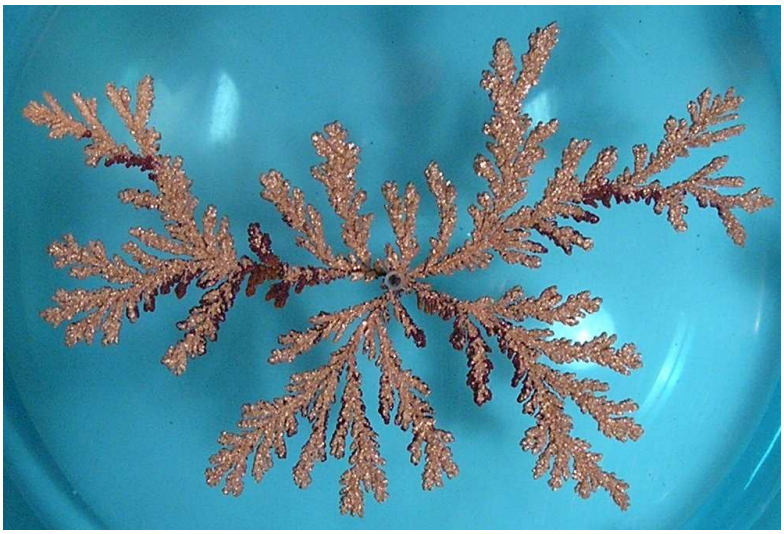

Figura 7: Estrutura de cobre formada a partir de uma solução de sulfato de cobre numa célula de eletrodeposição. Mais informações, aqui.

um ponto no plano. No centro, haverá um ponto contendo a impureza em torno da qual o cristal irá crescer. Uma hipótese bastante razoável se a concentração for baixa é que as partículas se agregam uma a uma à estrutura. Essencialmente, o movimento descrito pela partícula antes de uma eventual agregação é um *random walk*, ou passeio aleatório. Isto significa que, se a partícula está na posição $(i, j)$, no próximo passo pode estar em qualquer uma das 8 posições vizinhas $((i-1, j-1), (i-1, j)$ , $(i-1, j+1)$ , $(i, j-1)$, $(i, j+1)$, $(i+1, j-1)$, $(i+1, j)$ ou $(i+1, j+1))$, com igual probabilidade, e assim por diante.

A dinâmica do modelo pode ser descrita da seguinte forma. Uma partícula surge aleatóriamente em alguma posição do plano e caminha num random walk até encontrar a estrutura (ou a impureza, caso seja a primeira) e se ligar a ela, ou até que eventualmente saia do plano. Então, uma nova partícula surge num outro ponto aleatório e a dinâmica segue. Uma imagem de uma estrutura plana típica gerada por este processo está na Fig. 8. Num dado ponto do plano, só pode haver uma única partícula. Portanto, as partículas se agregam à estrutura sempre que chegam a alguma posição vizinha de algum ponto ocupado. Em geral, o processo para após um certo número de partículas terem sido agregadas, ou até que alguma parte da estrutura atinja

Figura 8: Estrutura plana (fractal!) gerada por DLA. Notem a semelhança com a estrutura real de cobre da Fig. 7.

os limites de nosso segmento de plano.

Na prática, uma simulação deste tipo levaria muito tempo. No início, quando a estrutura ainda é pequena, a probabilidade de que uma partícula que surja ao acaso num ponto aleatório do plano seja levada, por movimento Browniano, ao encontro da estrutura é baixíssima. Gastaria-se muito tempo simulando-se partículas que escapariam da estrutura. Isto pode ser melhorado adotando-se a seguinte estratégia. As novas partículas sempre surgem sobre (aproximadamente) um circulo que engloba a estrutura, cujo o diâmetro é cerca de duas vezes o tamanho da estrutura. Além disso, pode-se admitir que a partícula esta perdida se ela se ultrapassar um circulo com o dobro do tamanho deste último. Os applets disponíveis aqui e aqui podem ajudar a visualização deste processo.

A segunda parte do EP consiste em fazer um programa com o modelo DLA no plano. As figuras obtidas devem ser gravadas num arquivo PPM (ou equivalente). Sugere-se que as figuras sejam também enviadas a nossa galeria. Como sempre, há diversas maneiras de se "colorir"as figuras. A mais comum é estabelecer um código de cor que leve em conta o instante em que a partícula foi agregada. Tons mais escuros, por exemplo, poderiam indicar partículas mais antigas na estrutura, enquanto os mais claros corresponderiam as mais novas. Imagens semelhantes existem em estruturas de corais

(Fig. 9). Sugiro que vocês experimentem alterando a forma da impureza

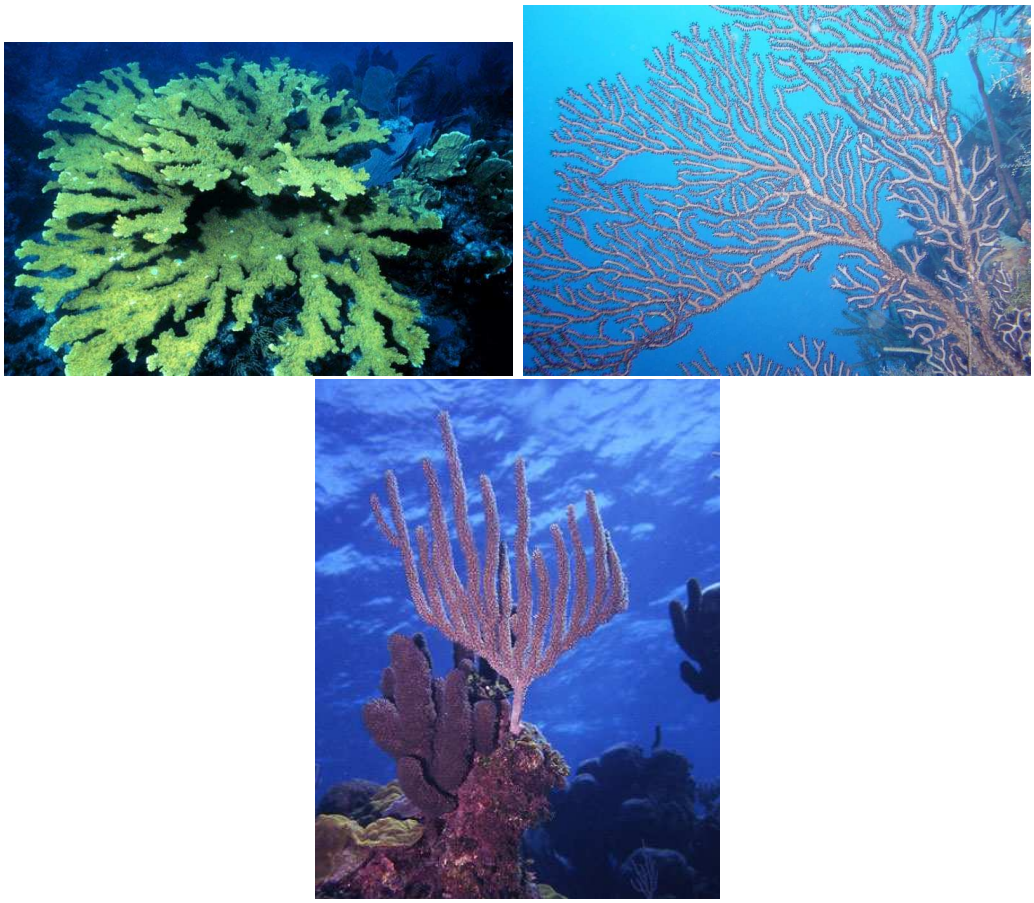

Figura 9: Estruturas de corais com forma semelhantes às obtidas por processos do tipo DLA.

inicial. Considerem linhas, quadrados, triângulos, etc. Poderemos ter várias surpresas agradáveis.

# Agradecimentos

Certos Punks regenerados, que nos dizem *It's something unpredictable, but in the end it's right.* Também a Olivia, por ter me apresentado ao Seu Jorge,

responsável por esta surpreendente “re-leitura” do velho DB, desperdiçada como trilha sonora no péssimo filme Aquatic Life, ambientado em magníficas aguas repletas de corais como as da Fig. 9.