

Método de Monte Carlo para caminhos aleatórios

Emmanuel Podestá Jr., Gustavo Zambonin

Modelagem e Simulação (UFSC-INE5425)

1 Documentação

Toda a documentação pode ser consultada junto ao código-fonte, acessível pelo repositório referenciado neste documento (em inglês). Abaixo, uma breve explanação das técnicas utilizadas para criar o programa será feita, dividida pelos arquivos que compõem o projeto.

1.1 `__main__.py`

Responsável por executar código referente à inicialização do programa, visto que este é distribuído com classes separadas. A janela para interface com o usuário é criada, e após o término de sua execução, bibliotecas internas lidam com a finalização do programa.

1.2 `custom_canvas.py`

Apresenta uma classe pai representando as figuras criadas pela biblioteca de gráficos, compatíveis com a interface escolhida, bem como três especializações desta (uma para o gráfico do caminho percorrido, outra para o gráfico das distâncias, e a final para a criação do histograma). A interatividade desses gráficos também é adicionada nessa classe. Os dados necessários são obtidos a partir da *thread* que faz todas as computações.

1.3 `monte_carlo.py`

Implementa uma *thread* específica para a resolução dos cálculos apresentados no problema, ou seja, geração de números pseudo-aleatórios, emissão de sinais referentes ao progresso do cômputo para popular a barra de progresso e transferir dados para as classes referentes à produção dos gráficos, e avanço nas coordenadas (x, y) de acordo com certo ângulo sorteado anteriormente. Caso não existisse barra de progresso, o isolamento destes cálculos em uma especialização de *thread* seria desnecessária.

1.4 `qt_window.py`

Consiste da construção de janela personalizada para a finalidade do simulador, concisa e discreta, com apenas um botão de controle, cuja função é inicializar a *thread* responsável pela simulação numérica, e dois parâmetros modificáveis. Também lida com eventos de teclado e *mouse* para controle do programa e posicionamento dos elementos internos por meio de *layouts*, e da janela em relação à resolução disponível no computador utilizado.

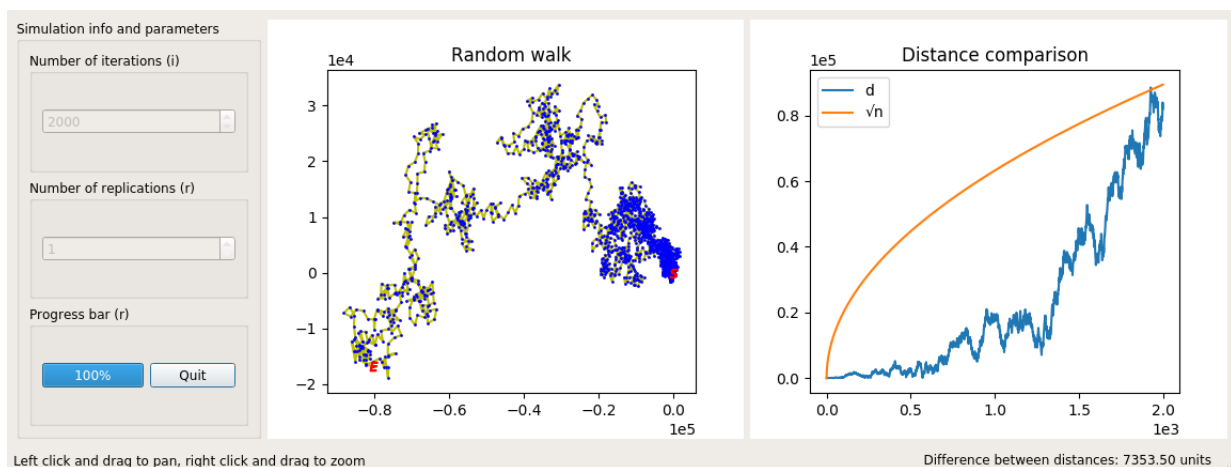
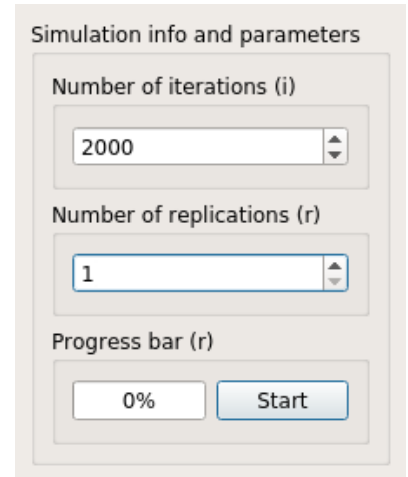


Figura 1: Tela de resultados do programa se $r = 1$. Note que sua aparência pode mudar entre sistemas operacionais.

2 Utilização

O simulador pode ser obtido a partir do repositório referenciado neste documento, [na seção de downloads](#). Após descompactá-lo, o usuário deve acessar a pasta criada e executar o arquivo `drwalk.exe`. Os parâmetros podem ser configurados com números no intervalo $[1, 2^{31} - 1]$, porém aconselha-se a utilização de valores razoáveis, em virtude do desempenho da linguagem escolhida.

Ao acionar o botão para iniciar a simulação (ou apertar **ENTER**), o usuário poderá acompanhar a computação a partir de uma barra de progresso. Ao fim desta, se $r = 1$, dois gráficos serão mostrados: o caminho aleatório feito a partir das i iterações, bem como a distância percorrida e esperada. Caso contrário, um histograma das diferenças entre as distâncias será apresentado. Todos os gráficos são interativos: arrastar o clique esquerdo do *mouse* moverá o ponto de referência do gráfico, e arrastar o clique direito proverá *zoom*. O programa termina quando **Esc** é acionado em qualquer momento de sua execução.



The image shows a window titled "Simulation info and parameters". It contains three sections:

- Number of iterations (i)**: A text box containing the value "2000" with up and down arrow buttons on the right.
- Number of replications (r)**: A text box containing the value "1" with up and down arrow buttons on the right.
- Progress bar (r)**: A section containing a text box with "0%" and a blue "Start" button.

Figura 2: Tela inicial do programa.