Aproximando π com o Método Estocástico de Monte Carlo

Laboratório de Computação e Simulação

(MAP2212)

Aluno:	**Milton Leal Neto**
Curso:	Bacharelado de Matemática Aplicada Computacional
NUSP:	8973974
Professor:	Julio Stern

São Paulo, 26 de abril de 2021

Conteúdo

1 Apresentação

Este relatório apresenta uma solução para o primeiro Exercício Programa (EP) proposto pelo professor Julio Stern no âmbito da disciplina de Laboratório de Computação e Simulação (MAP 2212) do Bacharelado de Matemática Aplicada Computacional (BMAC) do Instituto de Matemática e Estatística (IME) da Universidade de São Paulo (USP).

O objetivo do EP é utilizar o método estocástico de Monte Carlo para estimar o valor do número irracional π com acurácia de 0.05%.

O enunciado do EP exige a utilização da biblioteca *random* da linguagem *Python* para gerar pontos uniformemente distribuídos $x_i \in [-1, 1]^2, i \in \{1, ..., n\}$, ou seja, pontos gerados em um quadrado centrado na origem do plano cartesiano cujo lado mede duas unidades. A partir destes pontos, estima-se π pela proporção $p = (1/n) \sum_{i=1}^{n} T(x_i)$, sendo que $T(x) = \mathbb{1}\|x\|^2$ testa se o ponto x_i caiu dentro do círculo unitário inscrito no quadrado.

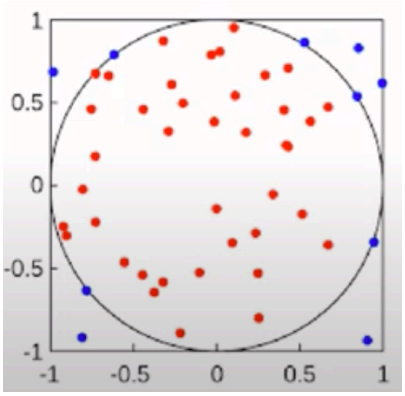

Figura 1: Quadrado com círculo unitário inscrito e pontos aleatórios

Desta forma, o EP constitui-se na criação de um programa que estime π com a acurácia desejada e também na apresentação do racional por trás da escolha do número n de pontos que serão aleatoriamente gerados para a simulação de Monte Carlo.

2 Definindo n

Como o objetivo é encontrar o número n de pontos aleatórios que precisam ser gerados pelo programa para aproximar o valor real de π, com probabilidade δ, tal que $\hat{\pi}$ esteja dentro de um percentual ϵ do valor real, adotaremos o critério de parada do programa atrelado ao erro relativo. Ou seja, queremos que a seguinte desigualdade seja verdadeira:

$$P(\frac{|\pi - \hat{\pi}|}{\pi} \geq \epsilon) \leq \delta.$$

Para n suficientemente grande, podemos argumentar que um bom indicador para a veracidade da desigualdade acima é tal que:

$$\text{(I) erro relativo} = \frac{\text{erro padrão}}{\hat{\pi}} \leq \frac{\epsilon}{\Phi^{-1}(1 - \frac{\delta}{2})}.$$

Vale dizer que Φ^{-1} é a inversa da função densidade de probabilidade Φ de uma v.a. Normal Padrão Z. Além disso, o erro padrão é dado por

$$\text{erro padrão} = \frac{\hat{\sigma}_n}{\sqrt{n}},$$

sendo $\hat{\sigma}_n$ o desvio padrão da amostra de tamanho n.

Antes de mostrarmos a validade de (I), denotemos de agora em diante o erro relativo pela sigla *e.r.* e o erro padrão pela sigla *e.p.* Vejamos agora (I).

Observe que

$$P(|\pi - \hat{\pi}| \geq \epsilon\pi) = P(\frac{|\pi - \hat{\pi}|}{\text{e.p}} \geq \frac{\epsilon\pi}{\text{e.p.}}).$$

Mas para valores grandes de n, a função de distribuição acumulada (fda) de

$$Z = \frac{\hat{\pi} - \pi}{\sqrt{\hat{\sigma}^2/n}}$$

converge para a fda de $\Phi(Z)$ de uma Normal Padrão e isso implica que

$$Z = \frac{\hat{\pi} - \pi}{e.p.}$$

é aproximadamente Normal Padrão. Portanto, uma condição suficiente para

$$P(\frac{|\pi - \hat{\pi}|}{\pi} \geq \epsilon) \leq \delta$$

ser verdade é tal que

$$\Phi^{-1}(1-\frac{\delta}{2}) \leq \frac{\epsilon\pi}{e.p.},$$

que é verdade se e somente se

$$\frac{e.p.}{\pi} \leq \frac{\epsilon}{\Phi^{-1}(1-\frac{\delta}{2})}.$$

Finalmente, o lado esquerdo pode ser estimado pelo erro relativo $e.r./\hat{\pi}$.

Agora suponha que, usando n pontos, nós cheguemos a

$$r.e. = \frac{e.p.}{\hat{\pi}} > \frac{\epsilon}{\Phi^{-1}(1-\frac{\delta}{2})}.$$

Como um próximo passo, computamos quantos pontos adicionais m são necessários para obtermos

$$r.e. = \frac{e.p.}{\hat{\pi}} \leq \frac{\epsilon}{\Phi^{-1}(1-\frac{\delta}{2})}.$$

Primeiramente, assumimos que, depois de amostrar m pontos adicionais, ao recalcular $\hat{\pi}$ e $\hat{\sigma}$ estes valores não mudarão significativamente.

Se este é o caso, então o erro relativo irá mudar por um fator de $\sqrt{\frac{n}{m+n}}$.

Queremos então

$$\sqrt{\frac{n}{m+n}}\frac{\hat{\sigma}_n}{\hat{\pi}_n} \leq \frac{\epsilon}{\Phi^{-1}(1-\frac{\delta}{2})}.$$

Isolando m, temos

$$m \geq [\frac{\Phi^{-1}(1-\frac{\delta}{2})\hat{\sigma}_n}{\epsilon\hat{\pi}_n}]^2 - n.$$

Dessa maneira, partindo de um n inicial escolhido como 10.000, obtemos, por meio do programa descrito na próxima seção, que o valor aproximadamente ótimo para a simulação proposta é de n na casa dos $4,1$ milhões, considerando um intervalo de confiança com 95% de confiabilidade, ou seja, utilizando $\Phi^{-1}(1-\frac{\delta}{2}) = 1.96$.

3 Estrutura do programa

O programa escrito em *Python* está estruturado em três funções:

1) calcula_proporcao(n):

Esta função gera valores aleatórios para x_i e y_i entre $[-1, 1]$ e verifica se a soma ao quadrado destes valores é menor ou igual a 1, caracterizando o ponto dentro ou fora do círculo unitário. A função devolve a proporção de pontos que caiu dentro do círculo em relação ao total de pontos amostrados.

2) calcula_n(n_inicial):

A partir de um valor inicial de n, que no programa foi escolhido como 10.000, a função calcula i) a proporção estimada dos pontos dentro do círculo em relação aos pontos totais através da função calcula_proporcao(n), ii) a variância amostral, iii) o desvio padrão amostral e iv) realiza o cálculo do n "ótimo".

3) main:

É a função de chamada principal, na qual é estipulada a seed, o valor inicial de n, é realizada a chamada da função calcula_n e os comandos de impressão dos resultados.

4 Conclusão

Os resultados da simulação via método de Monte Carlos mostram que em 10 rodadas do programa a precisão do valor aproximado de π ficou oscilando na terceira casa decimal entre 0,1,2 e 3, como era de se esperar considerando a acurácia de 0.05%. Com isso, concluímos que a estratégia para determinar o valor de n é adequada para atingir o objetivo pedido.

```
Número de pontos aleatórios = 4163849
1 ) Pi é aproximadamente =  3.1417431323758 38
2 ) Pi é aproximadamente =  3.1421936770521697
3 ) Pi é aproximadamente =  3.141527946858784
4 ) Pi é aproximadamente =  3.1416307363691622
5 ) Pi é aproximadamente =  3.1411014184231947
6 ) Pi é aproximadamente =  3.1408372397750255
7 ) Pi é aproximadamente =  3.1434204266293038
8 ) Pi é aproximadamente =  3.141784440309915
9 ) Pi é aproximadamente =  3.1419391049002976
10 ) Pi é aproximadamente =  3.1416105627269384
Tempo de execução = 65.85567998886108
```

Figura 2: Resultados de 10 rodadas do programa

5 Referências

[1] Ebert, T – The Monte Carlo Method [https://bit.ly/3nnaXfq];