Exercício Programa 1

Gabriel Tetsuo Haga

Abril 2021

1 Problema

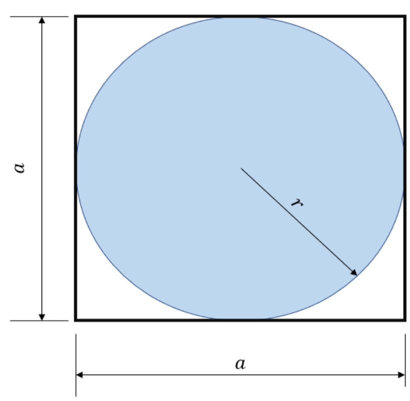
Suponha um círculo de raio r inscrito em um quadrado de lado a, como mostra a imagem ao lado:

Sabe-se que a área A do círculo é dada pela fórmula:

$$A = \pi r^2 \tag{1}$$

, onde r é o raio do círculo. Contudo sabe-se que para a utilização dessa fórmula é necessário saber o valor numérico da constante π. E o problema proposto está em estimar um valor para essa constante.

Figure 1: Círculo inscrito



Fonte: Autoria própria

Para tal irá se utilizar de um outro método para calcular a área de um círculo. Antes de explicar o método, vamos definir que $a = 2$ e $r = 1$, não definiremos unidade de medidas, pois não estamos interessados na parte física do problema. E definiremos um sistema de coordenadas Oyz na qual a origem O coincide com o centro da circunferência e as retas y e z são paralelas as arestas do quadrado. Nesse método estamos interessado, nos pontos que estão dentro do quadrado, logo os conjuntos de pontos $x = (y, z)$ que trabalharemos serão $[-1, 1]^2$. A última definição a ser apresentada é a de uma função $T(x)$, que tem o valor 1 se o ponto x está dentro da circunferência, caso o contrário $T(x)$ tem o valor 0. O método consite em gerar n pontos x_i ($i \in \{1, ..., n\}$) de forma aleatória tais que $x_i \in [-1, 1]^2$, para, assim calcularmos a proporção p_n entre os pontos que estão dentro da circunferência e o número total de pontos através da seguinte equação:

$$p_n = \frac{1}{n}\sum_{i=1}^{n} T(x_i) \tag{2}$$

Além disso, pode-se dizer que sabendo a área A_Q do quadrado é possível determinar a área A do círculo pela relação

$$A = pA_Q \tag{3}$$

, sendo p definido da seguinte maneira:

$$p = \lim_{n\to\infty} p_n \tag{4}$$

Logo, é possível fazer uma aproximação de p para p_n, para n suficientenmente grande. E assim calcular a área do círculo pela equação 3, e ao saber o valor da área do círculo é possível determinar o valor de π pela equação 1. Como estamos trabalhando um quadrado de lado $a = 2$ e círculo de raio $r = 1$, pode-se dizer que a área do quadrado é $A_Q = 4$ e a área do círculo é $A = \pi$, logo pode-se modificar a equação 3 para os valores de área nessa situação e adotar a hipotése de que $p_n = p$, para um n suficientemente grande, e assim é obtido:

$$\pi = 4p_n \tag{5}$$

Assim o problema consite em determinar um n, para poder estimar a constante π, tal que, o erro do π estimado seja menor que 0.05%.

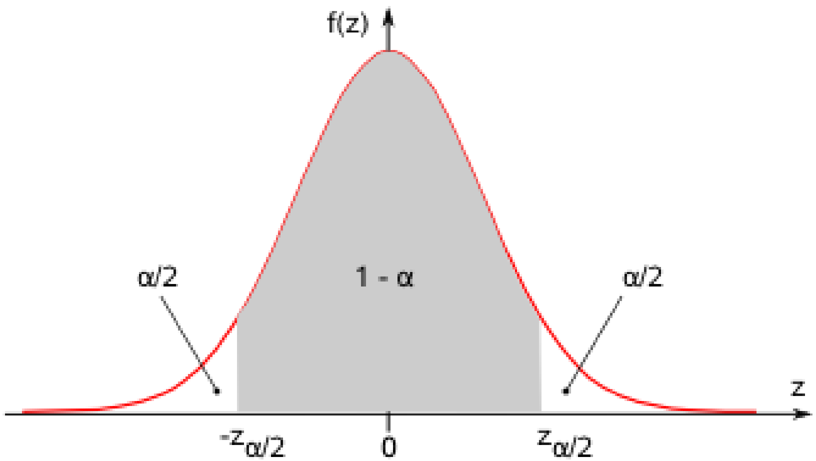
2 Solução proposta

Primeiramente, temos que entender com o que estamos trabalhando. No caso, é possível perceber que se trata de um problema de estatística, pois trabalhamos com uma amostra (n pontos) para representar uma população (os infinitos pontos contidos em $[-1, 1]^2$). Assim pode-se utilizar ideias dessa área para resolver o problema.

A ideia que foi escolhida para resolver esse problema é a de utilizar um intervalo de confiança para a proporção p. Para tal é necessário saber a distribuição de probabilidade das proporções p_n, a média $\overline{p_n}$ e a variância s^2 dela. Como estamos trabalhando com proporções e com n's grandes (tal que $np_n \geq 5en(1 - p_n) \geq 5$), pode-se aproximar a distribução de probabilidade da proporção para uma distribuição normal, com média igual a $\overline{p_n} = p$ e com variância (s^2) igual a $s^2 = \dfrac{p_n(1-p_n)}{n}$. Essas hipóteses serão úteís para delinear a solução. Vale observar que se a hipótese de ser uma curva normal não fosse válida deveria se usar uma distribuição binomial.

Pode-se utilizar a variável Z que é a variável p_n padronizada, ou seja, $Z = \dfrac{p_n - p}{s}$, sendo $s = \sqrt{s^2}$. Isso é feito, pois os valores da distribuição de probabilidade para Z são tabelados. Abaixo é mostrado uma imagem da curva normal padronizada:

Figure 2: Distribuição normal



Fonte: https://www.mspc.eng.br/dir20/prob$_e st362.shtml$

A figura 2 ajuda a visualizar o intervalo de confiança que é $[-Z_{\alpha/2}; Z_{\alpha/2}]$. Queremos que a nossa variável Z esteja dentro desse intervalo de confiança, ou seja, $-Z_{\alpha/2} \leq Z \leq Z_{\alpha/2}$, sendo que α é o nível de significância do intervalo de confiança, que pode ser definido como a probabilidade de Z não estar dentro do intervalo de confiança. Logo $P(-Z_{\alpha/2} \leq Z \leq Z_{\alpha/2}) = 1 - \alpha$, para nosso problema definiu-se que tenha um $\alpha = 0.1\%$, para esse valor $Z_\alpha = 3.30$, obtido através de uma tabela normal. Podemos, então, fazer a seguite manipulação:

$$-Z_\alpha \leq Z \leq Z_\alpha \Rightarrow -Z_\alpha \leq \frac{p_n - p}{s} \leq Z_\alpha \Rightarrow -Z_\alpha s \leq p_n - p \leq Z_\alpha s \Rightarrow$$

$$p_n + Z_\alpha s \leq p \leq p_n - Z_\alpha s$$

Logo chegamos que o erro e_0 máximo da nossa aproximação é dado por:

$$e_0 = Z_\alpha s = Z_\alpha \sqrt{\frac{p_n(1-p_n)}{n}} \tag{6}$$

. Mas como nosso erro máximo já está definido como $e_0 = 0.05\% \times p = 0.0005 \times p$. Lembrando que estamos usando que $p \approx p_n$, assim $e_0 \approx 0.0005 \times p_n$, substituindo essa aproximação na equação (6), obtém-se:

$$0.0005 \times p_n = Z_\alpha \sqrt{\frac{p_n(1-p_n)}{n}} \Rightarrow$$

$$n = \left(\frac{Z_\alpha}{0.0005 \times p_n} \right)^2 p_n(1 - p_n) \tag{7}$$

Com a equação (7) é possível estimar um n, que respeite as condições do problema. E com n é possível estimar o valor de π, como foi solicitado.

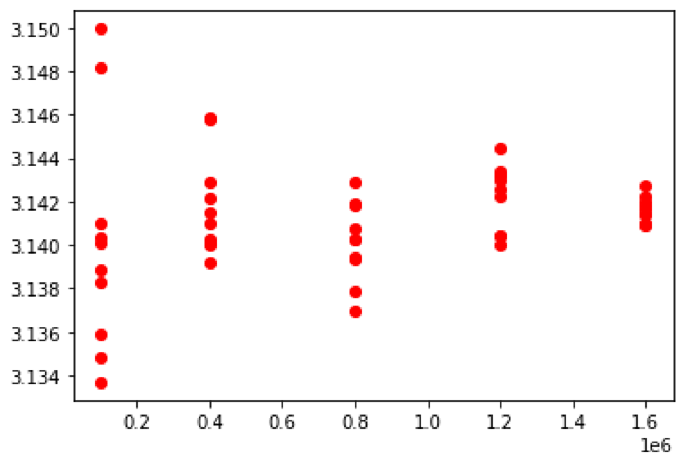
Observação: nós temos uma certeza de 99.9% de que o valor de p_n tem um erro menor que e_0 em relação à proporção "verdadeira" p.

3 Implementação no código

A ideia de implementação dentro do código para estimar o valor de π é a seguinte, primerio atribuímos um valor inicial qualquer para n, depois entramos em um laço, que condiciona um erro percentual menor que 0.05% para sair dele. Dentro do laço utilizamos a equação (2) para calcular a proporção p_n, após isso calcula-se o erro e_0 através da equação (6), e por fim calcula-se um novo n utilizando a equação (7) que substituirá o antigo n se não saírem do laço, ou seja, é utilizado um método iterativo até o valor convergir para um resultado que satisfaça a condição imposta.

Ao rodar o algoritmo, foi percebido que que o número de iterações será baixo, pois a aproximação para π através desse método estocástico é boa mesmo para n da ordem de 10^3, assim converge para um valor de n, rapidamente. Perceba que o valor de n é o mais importante, nesse caso, pois ao rodar várias vezes com um valor relativamente baixo para n, os valores de p_n terão uma grande variação entre eles então o valor final não será confiável, como mostra o gráfico abaixo:

Figure 3: Gráfico de dispersão dos valores estimados para π em função de n



Fonte: Autoria própria

Então como não sabemos o valor de π, o n que irá garantir que a estimativa é precisa com um nível de confiança de $1-\alpha = 99.9\%$. Os resultados obtidos para n e para π são mostrados abaixo:

n	π
11898082	3.141955484998028

Os códigos feitos para esse EP são mostrados a seguir:

1. O primeiro é o código que estima o valor de n e de π:

```
#### Importa funções utilizadas no EP
from random import uniform as random
# from random import random
from math import sqrt, pi
from statistics import stdev, mean
import matplotlib.pyplot as plt
from scipy.stats import t

#### Parâmetros iniciais
n_1 = 10**5 ## Chute inicial para n
err_p = 1 ## Valor do erro percentual, colocou-se 1 para entrar no laço
z_alpha = 3.3 ## Valor da variável z para alpha = 0.1%, esse valor muda para alpha diferentes
cont = 0 ## Contador que mostra quantas iterações foram feitas

#### Listas das variáveis
list_n = [] ## Lista dos n
list_p = [] ## Lista dos valores obtidos para p_n para cada n
list_err = [] ## Lista dos erros percentuais obtidos para cada n
list_pi = [] ## Lista dos pi's estimados
#### Algoritmo
while err_p > 0.05: ## Condiciona a saída do laço somente se o erro percentual for menor que 0.05%
    n = n_1 ## Substitui o antigo valor de n para um novo valor de n
    p = 0 ## Inicializa a proporção p_n com zero a cada rodada desse laço
    for k in range(n): ## Laço para para gerar os pontos aleatoriamente
        y = random(-1,1) ## Coordenada y do ponto x_i
        z = random(-1,1) ## Coordenada z do ponto x_i
        if sqrt((y**2 + z**2)) <= 1 : ## Faz o papel da função T(x) se o ponto estiver dentro da circunferência ele atribui 1 e se não
            T = 1
        else:
            T = 0
        p = p + T/n ## Faz o papel do termo: 1/n vezes somatoria de T de 1 até n

    pi_estimado = 4*p ## Calcula o pi_estimado nessa rodada
    err_p = 100*(z_alpha*sqrt(p*(1-p)/n))/p ## Calcula o erro percentual
```

```
    n_1 = round((z_alpha/(0.0005*p))**2*p*(1-p)) ## Recalcula o n
    cont+=1

    list_p.append(p) ## Indexa o valor de p_n dentro da lista
    list_err.append(err_p) ## Indexa o valor do erro percentual nessa rodada
    list_pi.append(pi_estimado) ## Indexa o valor do pi estimado nessa rodada
    list_n.append(n) ## Indexa o valor de n utilizado nessa rodada

#### Imprime as variáveis
print('Número de iterações: ', cont, '\n')
print('Lista das proporções p_n: ', list_p, '\n')
print('Lista dos n: ', list_n, '\n')
print('Lista dos erros percentuais: ', list_err, 'n')
print('Lista dos pi estimados: ', list_pi, '\n')
print('n estimado final: ',list_n[-1], '\n')
print('pi estimado final: ', list_pi[-1], '\n')
```

2. O segundo é o código que produz o gráfico 3:

```
## Importa funções utilizadas no EP
from random import uniform as random
# from random import random
from math import sqrt, pi
from statistics import stdev, mean
import matplotlib.pyplot as plt

# #### Funções criadas
## Primeira função: Utilizada para repetir n vezes cada elemento da lista
## Exemplo: lista = [1, 2, 3] e n = 2
##          lista2 = repmat(lista, n) = [[1, 1], [2, 2], [3, 3]]
def repmat(list_1, n):
    list_2 = []
    l= len(list_1)
    for j in range(l):
        list_aux = []
        for i in range(n):
            list_aux.append(list_1[j])

        list_2.append(list_aux)
    return list_2


### Parâmetros de entrada

## Lista dos n's utilizados para criar um gráfico
list_n = [1*10**5,4*10**5, 8*10**5, 12*10**5, 16*10**5]

## Número de vezes que será repetido o "experimento" com um certo n
n_r = 10

```

```
#### Variáveis calculadas a partir dos parâmetros de entrada

## Tamanho da lista dos n's
l_n = len(list_n)

## Lista utilizada para plotar gráfico
list_plot = repmat(list_n, n_r)

#### Lista de variáveis calculadas nos "experimentos"

list_pi =[] ## Lista das áreas obtidas através do método estocástico apresentado em aula
list_err = [] ## Lista dos erros das áreas em relação ao resultado analítico
list_p = []
for i in range(l_n):
    n = list_n[i]
    list_sample = []
    for j in range(n_r):
        pi_estimado = 0 ## Valor de pi_estimado inicializa com zero a cada rodada
        p = 0 ## Valor da proporção p_n inicializa com zero a cada rodada
        for k in range(n):
            # x = 2*random()-1
            # y = 2*random()-1
            x = random(-1,1) ## Chuta valor da coordenada x do ponto
            y = random(-1,1)
            if sqrt((x**2 + y**2)) <= 1 :
                p = p+1/n
        pi_estimado = 4*p

        list_p.append(p)
        list_sample.append(pi_estimado)
    list_pi.append(list_sample)


plt.figure(1)
plt.plot(list_plot, list_pi, 'ro')
plt.show()
```