МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра математического моделирования и анализа данных

АВСЯННИК
Елизавета Дмитриевна

Лабораторная работа №3

Имитационное и статистическое моделирование

Проверил:
В.П. Кирлица

Минск, 2021

0.0.1 Лабораторная работа 3

Используя метод Монте-Карло вычислить интеграл:

$I = \int_0^{\pi/2} cosxdx.$

Сравнить полученную оценку с оценкой, полученной по методу выделения главной части ($h(x) = 1 - \frac{x^2}{2}$). Сравнить дисперсии этих оценок.

[87]:
```
import numpy as np
import pandas as pd
from math import floor, pi, cos
import time
from scipy import integrate

import typing as tp

from tabulate import tabulate
```

Мультипликативный конгруэнтный метод моделирования БСВ

[88]:
```
def generate_brv_congruential_sample(M: int=2**31, alpha_star0: int=65539, beta :
 ↪int=65539) -> float:
    alpha_t = alpha_star0
    while True:
        alpha_t = (alpha_t * beta) % M
        yield alpha_t / M

def generate_brv_congruential(n: int=100, M: int=2**31, alpha_star0: int=65539,␣
 ↪beta :int=65539) -> np.ndarray:
    alpha = np.array([])
    generator = generate_brv_congruential_sample(M=M, alpha_star0=alpha_star0,␣
 ↪beta=beta)
    for i in range(n):
        alpha = np.append(alpha, next(generator))
    return alpha
```

Метод Макларена-Марсальи моделирования БСВ для моделирования равномерного распределения на отрезке

[89]:
```
def generate_brv_mm_sample(k: int=128, a:float=0, b:float=1) -> float:
    v = np.array([])
    brv_cong_b = generate_brv_congruential_sample()
    t = time.perf_counter()
    alpha_star0 = beta =  int(10**9*float((t-int(t))))
    brv_cong_c = generate_brv_congruential_sample(alpha_star0=alpha_star0,␣
 ↪beta=beta)
    for i in range(k):
            v = np.append(v, next(brv_cong_b))
```

```
    while True:
        index = floor(next(brv_cong_c) * k)
        alpha_t = v[index]
        v[index] = next(brv_cong_b)
        yield alpha_t * (b - a) + a

def generate_brv_mm(n: int=100, k: int=128,  a:float=0, b:float=1) -> np.ndarray:
    alpha = np.array([])
    generator = generate_brv_mm_sample(k=k, a=a, b=b)
    for i in range(n):
        alpha = np.append(alpha, next(generator))
    return alpha
```

0.0.2 Основной цикл программы

Метод Монте-Карло:

В качестве ξ была выбрана СВ равномерно распределенная на отрезке $[0, pi/2]$.

Таким образом: $p_{\xi}(x) = 2/\pi > 0, x \in [x_0, x_1] \int_{x_0}^{x_1} p_{\xi}(x)dx = 1$, где $x_0 = 0, x_1 = \pi/2$

$\eta = g(\xi) = \frac{cos\xi}{p_{\xi}(x)} = \frac{\pi cos\xi}{2}$

Можно показать, что $M\{\eta\} = I, D\{\eta\} < \infty$. Поэтому в качестве приближенного значения интеграла можно использовать статистическую оценку I_n, построенную в выборке из n независимых случайных величин $\eta_1, \eta_2 ... \eta_n$:

$I_n = \frac{1}{n}\sum_{i=0}^{n}\eta_i = \frac{1}{n}\sum_{i=0}^{n}\frac{g(\xi_i)}{p_{\xi}(\xi_i)}$

Метод выделения главной части:

$I = \int_0^{\pi/2} cosxdx = \int_0^{\pi/2} g(x)p_{\xi}(x)dx \int_0^{\pi/2} \frac{\pi}{2}cosx\frac{2}{\pi}dx$

$h(x) = \frac{\pi}{2}(1 - \frac{x^2}{2})$

$a = \int_0^{\pi/2} h(x)\frac{2}{\pi}dx = \int_0^{\pi/2}(1 - \frac{x^2}{2})dx = -\frac{1}{48}\pi(\pi^2 - 24)$

$\eta_1 = g_1(\xi) = a + \frac{cos(\xi)}{p_{\xi}(x)} - \frac{h(\xi)}{p_{\xi}(x)} = a + cos(\xi)\frac{\pi}{2} - (1 - \frac{\xi^2}{2})\frac{\pi}{2}$

[90]:
```
x_0 = 0
x_1 = pi / 2
p = 2 / pi

h = lambda x: (1 - x**2 / 2)
a = integrate.quad(h, x_0, x_1)[0]

def cos_main_component(x:float=0) -> float:
    return a + cos(x) / p  - (1 - x**2 / 2) / p
```

```python
print("Истинная величина: 1")

for n in [1e3, 5e3, 1e4, 5e4, 1e5, 5e5, 1e6]:
    generator = generate_brv_mm_sample(a=x_0, b=x_1)
    result_monte_carlo = []
    result_main_component = []
    for i in range(int(n)):
        rand = next(generator)
        result_monte_carlo.append(cos(rand) / p)
        result_main_component.append(cos_main_component(rand))
    result_monte_carlo = np.array(result_monte_carlo)
    result_main_component = np.array(result_main_component)
    print(f"n = {int(n)}")
    print(f"Величина, используя метод Монте-Карло: {result_monte_carlo.mean()},␣
 ↪Дисперсия = {result_monte_carlo.var()}")
    print(f"Величина, используя метод выделения главной части:␣
 ↪{result_main_component.mean()}, Дисперсия = {result_main_component.var()}")
```

```
Истинная величина: 1
n = 1000
Величина, используя метод Монте-Карло: 0.9843287423637573, Дисперсия =
0.22941040163348658
Величина, используя метод выделения главной части: 1.0016535540498026, Дисперсия
= 0.009534640826746092
n = 5000
Величина, используя метод Монте-Карло: 1.0022757389426142, Дисперсия =
0.22936311887120583
Величина, используя метод выделения главной части: 0.9990209205504119, Дисперсия
= 0.009715987255176904
n = 10000
Величина, используя метод Монте-Карло: 1.0002487891845766, Дисперсия =
0.23243167728820102
Величина, используя метод выделения главной части: 0.999742549046459, Дисперсия
= 0.009692428618740302
n = 50000
Величина, используя метод Монте-Карло: 0.9965093450680339, Дисперсия =
0.23479704425321804
Величина, используя метод выделения главной части: 1.0007342830684043, Дисперсия
= 0.009901452269712327
n = 100000
Величина, используя метод Монте-Карло: 0.9967199297335406, Дисперсия =
0.234092183843278
Величина, используя метод выделения главной части: 1.0005721121178617, Дисперсия
= 0.009833764119653709
n = 500000
```

```
Величина, используя метод Монте-Карло: 1.0008996755188673, Дисперсия =
0.233713978701259
Величина, используя метод выделения главной части: 0.9998633312232682, Дисперсия
= 0.00977267225043554
n = 1000000
Величина, используя метод Монте-Карло: 1.000464464193042, Дисперсия =
0.23381803175895477
Величина, используя метод выделения главной части: 0.9999493008340916, Дисперсия
= 0.009786102825318416
```

0.0.3 Аналитическая оценка дисперсий

Метод Монте-Карло

$I = \int_0^{\pi/2} cosxdx = \int_0^{\pi/2} g(x)p_{\xi}(x)dx = 1$

$M\{\eta\} = I$

$D\{\eta\} = M\{\eta^2\} - M\{\eta\}^2 = \int_0^{\pi/2} \frac{\pi^2 cos^2 x}{4} \frac{2}{\pi} dx - 1 = \frac{\pi^2}{8} - 1 = 0.2337$

[91]:
```
pi**2 / 8 - 1
```

[91]:
```
0.23370055013616975
```

Метод выделения главной части

$M\{\eta_1\} = \int_0^{\pi/2} g_1(x)p_{\xi}(x)dx = \int_0^{\pi/2} (a + cos(x)\frac{\pi}{2} - (1 - \frac{x^2}{2})\frac{\pi}{2})\frac{2}{\pi}dx = 1$

$D\{\eta_1\} = M\{\eta_1^2\} - M\{\eta_1\}^2 = \int_0^{\pi/2} (a + cos(x)\frac{\pi}{2} - (1 - \frac{x^2}{2})\frac{\pi}{2})^2\frac{2}{\pi}dx - M\{\eta_1\}^2 = 0.0098$

[95]:
```
# Мат ожидание
from scipy import integrate
m = integrate.quad(cos_main_component, 0, pi/2)[0] * p
m
```

[95]:
```
1.0
```

[96]:
```
#Мат ожидание квадрата
m_2 = integrate.quad(lambda x: cos_main_component(x) ** 2, 0, pi/2)[0] * p
m_2
```

[96]:
```
1.009779978785009
```

[97]:
```
#Дисперсия
d = m_2 - m ** 2
d
```

[97]:
```
0.009779978785009025
```

[ ]:
```

```