ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к первой лабораторной работе
«Генерация случайных чисел с заданным распределением»
по дисциплине «Моделирование»

Выполнил студент Федосеев Вячеслав Эдуардович
Ф.И.О.

Группы ИВ – 621

Работу принял ассистент кафедры ВС
Я. В. Петухова
подпись

Новосибирск – 2020

СОДЕРЖАНИЕ

ПОСТАНОВКА ЗАДАЧИ..3
ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ 3
РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ 5
ВЫВОДЫ 7

ПОСТАНОВКА ЗАДАЧИ

В рамках лабораторной работы необходимо смоделировать генерацию независимых случайных величин:

1. Непрерывное распределение с.в. методом отбраковки;
2. Дискретное распределение с.в. с возвратом;
3. Дискретное распределение с.в. без возврата.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

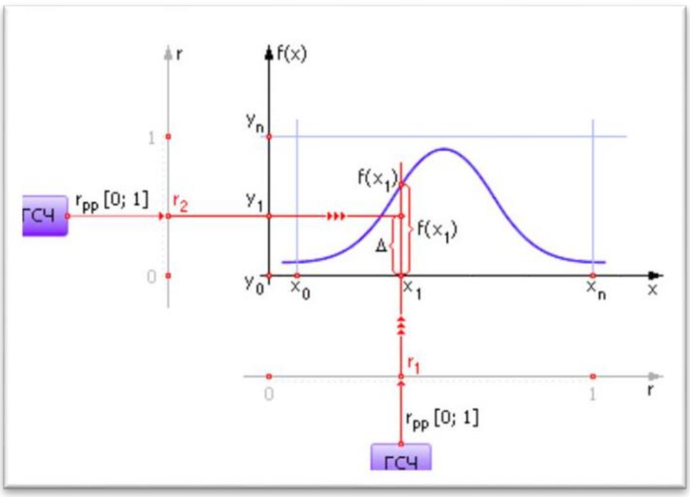
Метод отбраковки

Законы распределения вероятности событий могут быть различной формы, поэтому необходимо уметь превращать равномерный ГСЧ в генератор случайных чисел с заданным произвольным законом распределения.

Один из методов для моделирования непрерывной случайной величины является метод отбраковки. Метод используется в случае, когда функция задана аналитически. График функции вписывают в прямоугольник. На ось Y подают случайное равномерно распределенное число из ГСЧ. На ось X подают случайное равномерно распределенное число из ГСЧ. Если точка в пересечении этих двух координат лежит ниже кривой плотности вероятности, то событие X произошло, иначе нет.

Недостатком метода является то, что те точки, которые оказались выше кривой распределения плотности вероятности, отбрасываются как ненужные, и время, затраченное на их вычисление, оказывается напрасным. Метод применим только для аналитических функций плотности вероятности.

Рисунок 1. Иллюстрация метода отбраковки.



Алгоритм: в цикле генерируется два случайных числа из диапазона от 0 до 1. Числа масштабируются в шкалу X и Y, и проверяется попадание точки со сгенерированными координатами под график заданной функции *Y* = *f(X)*. Если точка находится под графиком функции, то событие *X* произошло с вероятностью *Y*, иначе точка отбрасывается.

Моделирование выборок дискретных случайных величин

Случайная величина ξ называется дискретной, если она может принимать дискретное множество значений $(x_1, x_2, x_3 \dots, x_n)$.

Дискретная случайная величина ξ определяется таблицей (распределением случайной величины ξ) или аналитически в виде формулы:

$$\xi = \begin{pmatrix} x_1 & x_2 & x_3 \dots & x_n \\ p_1 & p_2 & p_3 \dots & p_n \end{pmatrix}$$

где $(x_1 \quad x_2 \quad x_3 \dots \quad x_n)$ – возможные значения величины ξ; $(p_1 \quad p_2 \quad p_3 \dots \quad p_n)$ – соответствующие им вероятности:

$$P(\xi = x_i) = p_i$$

Числа $(x_1 \quad x_2 \quad x_3 \dots \quad x_n)$могут быть любыми, а вероятности $(p_1 \quad p_2 \quad p_3 \dots \quad p_n)$ удовлетворяют условиям:

1. $p_i > 0$;
2. $\sum_{i=1}^{n} p_i = 1$.

Функция распределения выглядит так:

$$F(x) = \sum_{x_k < x} P(X = x_k)$$

Наиболее общий случай построения генератора дискретных случайных величин основывается на следующем алгоритме. Пусть имеется таблица пар (x_i, p_i). Тогда кумулятивную сумму можно представить полуинтервалом $[0, 1)$, разбитым на полуинтервалы $[v_{i-1}, v_i)$, v_0 = 0, v_n = 1 длины p_i. Случайная величина ξ с неизбежностью принадлежит одному из этих полуинтервалов, тем самым определяя индекс дискретного значения. Номер полуинтервала, очевидно, определяется как:

$$\min\{i \mid \xi < v_i\} = \min\{i \mid \xi < \sum_{j=1}^{i} p_j\}$$

Дискретное распределение без возврата

Есть n случайных величин с одинаковой вероятностью (при следующих выборках вероятность распределяется поровну между величинами), мы выбираем 3 ·

n / 4 следующих величин без повторений, проделываем это большое количество раз и считаем частоты этих значений.

РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

Метод отбраковки

Пусть случайная величина X задана плотностью вероятности:

$$f(x) = \begin{cases} 0 & x \leq -4 \\ a \cdot \sqrt{x+4} & -4 < x \leq 5 \\ 0 & x > 5 \end{cases}$$

Известно, что несобственный интеграл от плотности вероятности есть вероятность достоверного события (условие нормировки):

$$\int_{-\infty}^{\infty} f(x)\,dx = \mathrm{P}\{-\infty < X < \infty\} = \mathrm{P}\{\Omega\} = 1$$

Исходя из этого свойства, можем найти параметр a:

$$\int_{-\infty}^{\infty} f(x)\,dx = \int_{-4}^{5} a \cdot \sqrt{x+4}\,dx = \frac{2}{3} \cdot a \cdot \sqrt{(x+4)^{3}}\,\Big|_{-4}^{5} = \frac{2}{3} \cdot a \cdot \sqrt{9^{3}} = 1$$

$$a = \frac{1}{18}$$

С помощью прикладной библиотеки matplotlib получим график плотности вероятности:

Рисунок 2. График функции плотности.

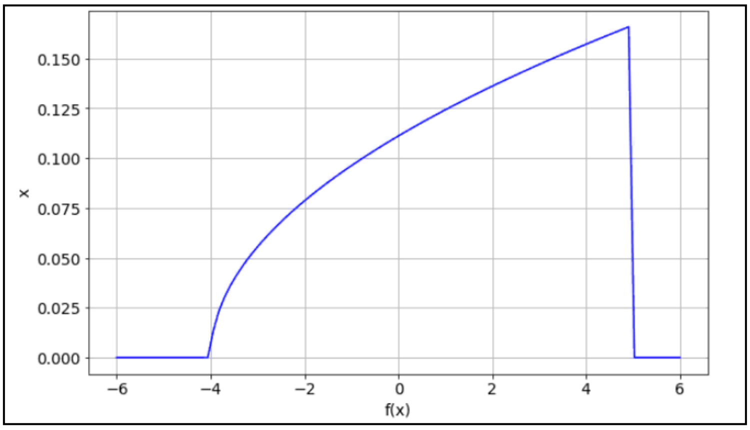


Рисунок 3. Результаты работы метода отбраковки.

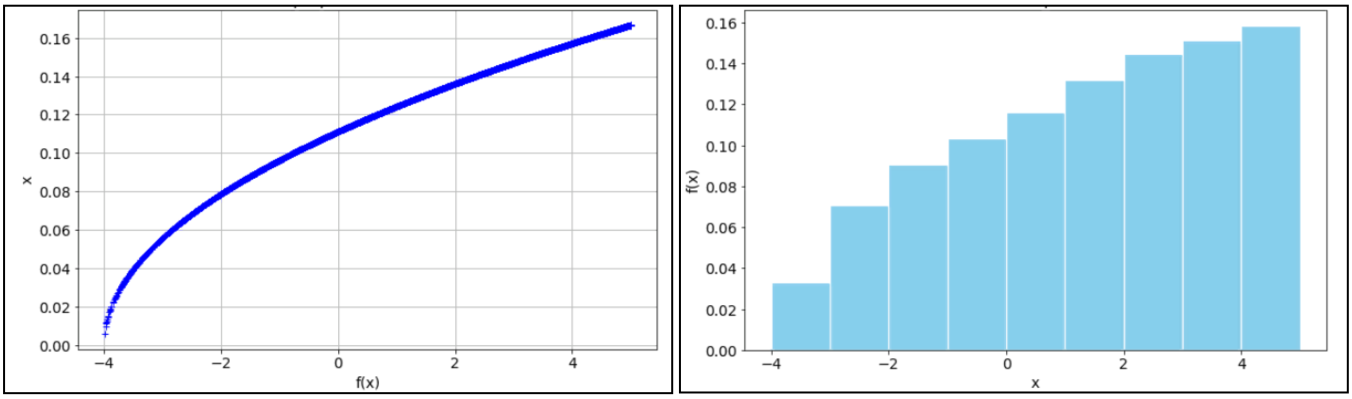
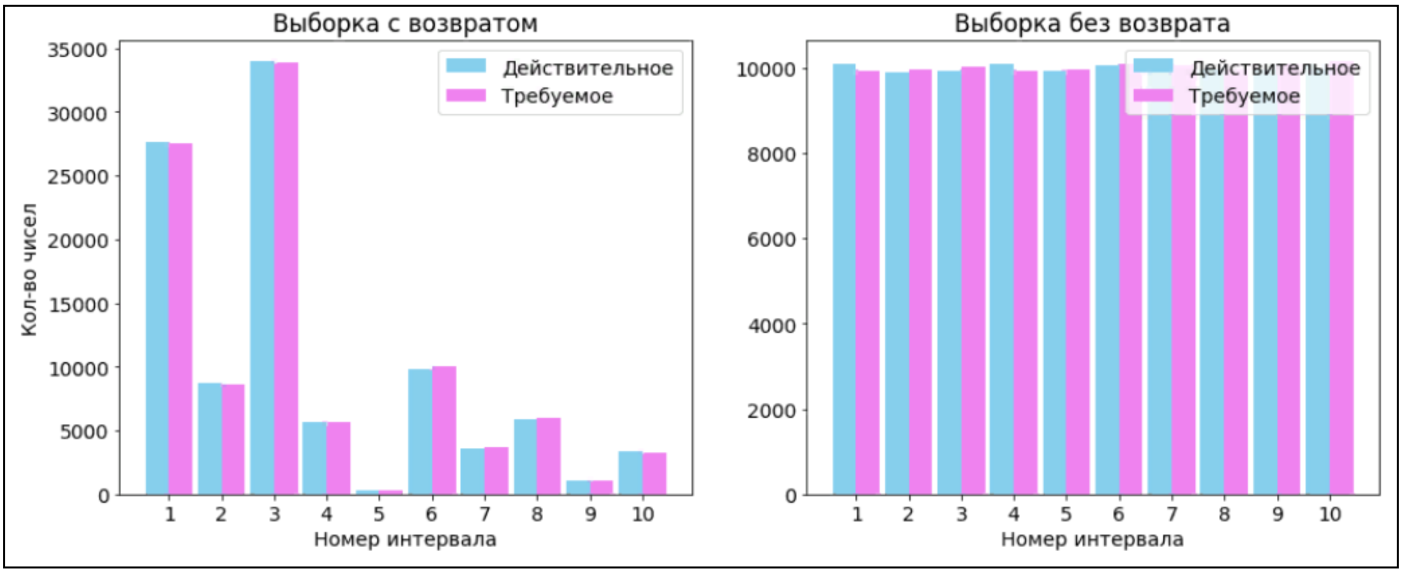


Рисунок 4. Результаты работы моделирования дискретной с.в.



ВЫВОДЫ

В ходе работы были изучены методы моделирования случайных величин, реализованы соответствующие алгоритмы. Результаты экспериментов представлены на графиках и гистограммах. Для реализации алгоритмов был использован язык Python 3.6 с библиотеками для линейной алгебры NumPy (стандартным генератор случайных чисел PCG64 из этой же библиотеки), для визуализации matplotlib, для структурирования данных в виде таблиц pandas.

По результатам работы метода отбраковки можно сделать следующие выводы. На взятой функции плотности метод отбраковки показал неплохие результаты, но метод имеет ряд недостатков: точки, которые оказались выше кривой распределения плотности вероятности, отбрасываются как ненужные, и время, затраченное на их вычисление, оказывается напрасным; метод применим только для аналитических функций, неэффективность алгоритма для распределений с длинными «хвостами» очевидна, поскольку в этом случае часты повторные испытания.

По результатам моделирования дискретных с.в. с реализацией выборки с возвратом и без возврата можно следующие выводы: среднее отклонение требуемого распределения от действительного достаточно мало, чтобы утверждать ГСЧ PCG64 выдает распределение близкое к равномерному.

ПРИЛОЖЕНИЕ

Листинг

```
import numpy as np
import pandas as pd
import scipy.stats as st
import matplotlib.pyplot as plt
def functor(x):
    if -4 <= x <= 5:
        return np.sqrt(x + 4) / 18
    return 0
def method_reject(a, b, c, f):
    generator = np.random.default_rng()

    condition = True
    while condition:
        x1 = generator.random()
        x2 = generator.random()
        condition = f(a + (b - a) * x1) <= c * x2

    return a + (b - a) * x1

def start_distribution(n):
    d = [0] * n
    residue_chance = 1.0
    for i in range(n-1):
        d[i] = np.absolute(np.remainder(np.random.rand(), residue_chance))
        residue_chance -= d[i]
    d[n-1] = residue_chance
    return d

def repeat(probablity, MAX, n):
    hits = [0] * n
    for i in range(MAX):
        rv = np.random.rand()
        acc_sum = 0.0
        for j in range(n):
            acc_sum += probablity[j]
            if rv < acc_sum:
                hits[j] += 1
                break
    return hits
def no_repeat(probablity, MAX, n):
    hits = [0] * n
    k = int(3 * n / 4)
    part = int(MAX / k + 1)
    for j in range(part):
        nums = np.arange(0, n)
        if j == part - 1:
            k = MAX % k
        for i in range(k):
            idx = int(np.random.rand() * (n - i))
            hits[nums[idx]] += 1
            nums = np.delete(nums, idx)
    return hit
```