Федеральное агентство связи
Федеральное государственное бюджетное образовательное учреждение
высшего образования «Сибирский государственный университет
телекоммуникаций и информатики»

Кафедра вычислительных систем

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к нулевой лабораторной работе по дисциплине
«Моделирование»

Выполнил:
студент гр. ИВ-621
Черненко А.С.

Проверила:
ассистент кафедры ВС
Петухова Я.В.

Новосибирск, 2020

Содержание

ЗАДАНИЕ

В рамках лабораторной работы необходимо убедиться в равномерности генератора псевдослучайных чисел, используя параметры χ^2 и автокорреляция.

ТЕОРИЯ

- Проверка по критерию χ^2

  p_i - теоретическая вероятность попадания чисел в i-ый интервал (всего этих интервалов k); она равна $p_i = \frac{1}{k}$

  N - общее количество сгенерированных чисел

  n_i - попадание чисел в каждый интервал

  χ^2 - критерий, который позволяет определить, удовлетворяет ли ГСЧ требования равномерного распределения или нет.

  Процедура проверки:

  1. Диапазон от 0 до 1 разбивается на k равных интервалов

  2. С помощью генератора случайных чисел получаем N чисел (по условию $\frac{N}{k} > 5$)

  3. Определяется количество случайных чисел, попавших в каждый интервал 4. Вычисляется экспериментальное значение χ^2 по следующей формуле:

$$\chi^2 = \frac{(n_1 - p_2 * N)^2}{p_1 * N} + \frac{(n_2 - p_2 * N)^2}{p_2 * N} + \cdots + \frac{(n_k - p_k * N)^2}{p_k * N}$$

$$\chi^2 = \sum_{i=1}^{k} \frac{(n_k - p_k * N)^2}{p_k * N} = \frac{1}{N} \sum_{i=1}^{k} \left(\frac{n_i^2}{p_i}\right) - N$$

  p_i - теоретическая вероятность попадания чисел в i-ый интервал (всего этих интервалов k); она равна $p_i = \frac{1}{k}$

  N - общее количество сгенерированных чисел

n_i - попадание чисел в каждый интервал

χ^2 - критерий, который позволяет определить, удовлетворяет ли ГСЧ требования равномерного распределения или нет.

- Проверки по критерию автокорреляции

$Ex = \bar{x}$ - математическое ожидание

s^2 - выборочная дисперсия

$\hat{\alpha}(\tau)$ – автокорреляция

x_i – множество случайных чисел

τ – смещение

$$Ex = \frac{1}{n}\sum_{i=1}^{n} x_i = \bar{x}$$

$$Dx = Ex^2 - (Ex)^2$$

$$\hat{\alpha}(\tau) = \frac{1}{(N-\tau)s^2}\sum_{t=1}^{N-\tau}(x_t - \bar{x})(x_{t+\tau} - \bar{x})$$

РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

Для проектирования были взяты следующие генераторы случайных чисел:

1. PCG64 – 128-битная имплементация перестановочного конгруэнтного генератора.
2. Philox — это 64-битный генератор, который использует конструкцию на основе счетчика, основанную на более слабых (и более быстрых) версиях криптографических функций.
3. MT1997 - Вихрь Мерсе́нна (англ. Mersenne twister, MT) — генератор псевдослучайных чисел (ГПСЧ), основывающийся на свойствах простых чисел Мерсенна.

Для анализа критерия χ^2 по таблице было выбрано значение нормали равное 14.1 по параметрам $k = 10 - 2 - 1 = 7$ (число степеней свободы,

которое считается по формуле $k = m - r - 1$, где m – это количество интервалов, а r – это количество параметров в конкретной функции распределения) и $a = 0.05$ (уровень значимости, выбранное случайным образом).

```
N = 1k, k = 5

PCG64 generator N=1000 k=5 chi_sqr=5.9300
0.00 to 0.20 = 220
0.20 to 0.40 = 174
0.40 to 0.60 = 210
0.60 to 0.80 = 199
0.80 to 1.00 = 197


Philox generator N=1000 k=5 chi_sqr=6.9700
0.00 to 0.20 = 211
0.20 to 0.40 = 209
0.40 to 0.60 = 194
0.60 to 0.80 = 170
0.80 to 1.00 = 216


MT1997 generator N=1000 k=5 chi_sqr=1.0700
0.00 to 0.20 = 195
0.20 to 0.40 = 200
0.40 to 0.60 = 211
0.60 to 0.80 = 192
0.80 to 1.00 = 202
```

Рисунок 1. Три ГСЧ по критерию χ^2 с параметрами N = 1000 и k = 5

```
N = 1k, k = 10

PCG64 generator N=1000 k=10 chi_sqr=13.3000
0.00 to 0.10 = 125
0.10 to 0.20 = 95
0.20 to 0.30 = 89
0.30 to 0.40 = 85
0.40 to 0.50 = 116
0.50 to 0.60 = 94
0.60 to 0.70 = 102
0.70 to 0.80 = 97
0.80 to 0.90 = 95
0.90 to 1.00 = 102

Philox generator N=1000 k=10 chi_sqr=10.4400
0.00 to 0.10 = 112
0.10 to 0.20 = 99
0.20 to 0.30 = 98
0.30 to 0.40 = 111
0.40 to 0.50 = 89
0.50 to 0.60 = 105
0.60 to 0.70 = 85
0.70 to 0.80 = 85
0.80 to 0.90 = 103
0.90 to 1.00 = 113

MT1997 generator N=1000 k=10 chi_sqr=7.6400
0.00 to 0.10 = 88
0.10 to 0.20 = 107
0.20 to 0.30 = 111
0.30 to 0.40 = 89
0.40 to 0.50 = 98
0.50 to 0.60 = 113
0.60 to 0.70 = 91
0.70 to 0.80 = 101
0.80 to 0.90 = 107
0.90 to 1.00 = 95
```

Рисунок 2. Три ГСЧ по критерию χ^2 с параметрами N = 1000 и k = 10

```
N = 10k, k = 10

PCG64 generator N=10000 k=10 chi_sqr=3.4740
0.00 to 0.10 = 993
0.10 to 0.20 = 1024
0.20 to 0.30 = 1024
0.30 to 0.40 = 967
0.40 to 0.50 = 1017
0.50 to 0.60 = 1001
0.60 to 0.70 = 1007
0.70 to 0.80 = 974
0.80 to 0.90 = 988
0.90 to 1.00 = 1005

Philox generator N=10000 k=10 chi_sqr=3.7680
0.00 to 0.10 = 993
0.10 to 0.20 = 996
0.20 to 0.30 = 997
0.30 to 0.40 = 970
0.40 to 0.50 = 1019
0.50 to 0.60 = 1002
0.60 to 0.70 = 1022
0.70 to 0.80 = 965
0.80 to 0.90 = 1012
0.90 to 1.00 = 1024

MT1997 generator N=10000 k=10 chi_sqr=3.8960
0.00 to 0.10 = 978
0.10 to 0.20 = 1000
0.20 to 0.30 = 982
0.30 to 0.40 = 972
0.40 to 0.50 = 1005
0.50 to 0.60 = 993
0.60 to 0.70 = 1013
0.70 to 0.80 = 1011
0.80 to 0.90 = 1044
0.90 to 1.00 = 1002
```

Рисунок 3. Три ГСЧ по критерию χ^2 с параметрами N = 10000 и k = 10

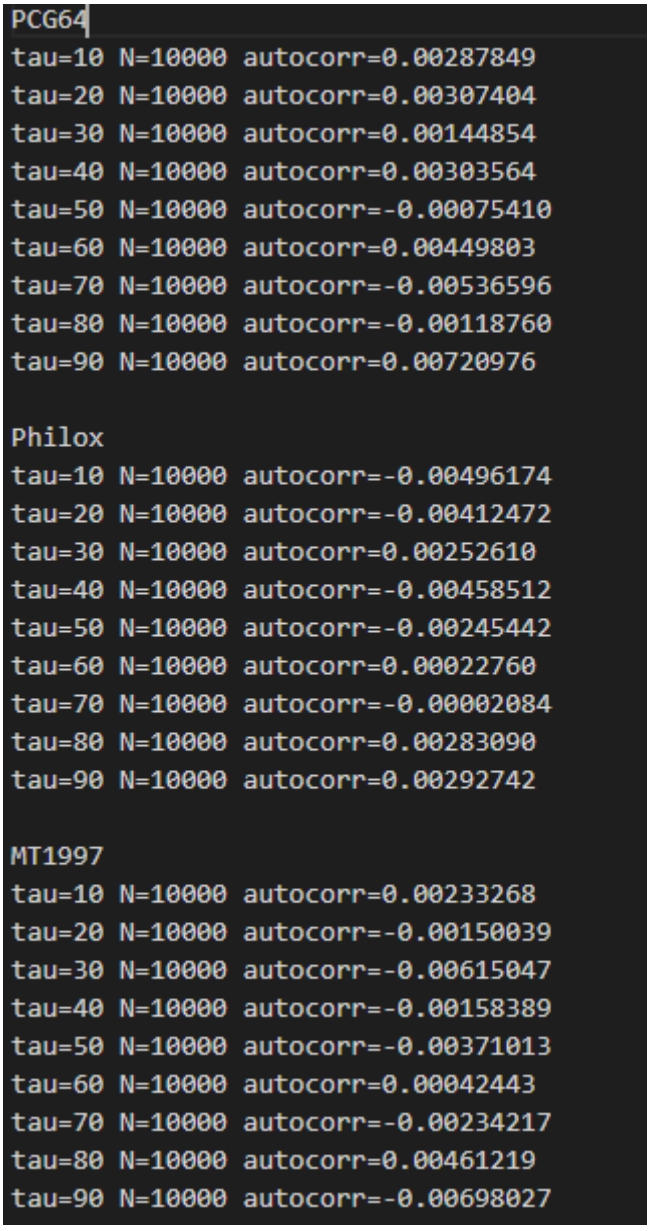

Рисунок 4. Три ГСЧ по критерию «автокорреляции» с N = 10000.

```
PCG64
tau=10 N=1000000 autocorr=0.00053938
tau=20 N=1000000 autocorr=-0.00041643
tau=30 N=1000000 autocorr=-0.00038661
tau=40 N=1000000 autocorr=0.00063233
tau=50 N=1000000 autocorr=-0.00003584
tau=60 N=1000000 autocorr=0.00018875
tau=70 N=1000000 autocorr=-0.00036635
tau=80 N=1000000 autocorr=0.00067801
tau=90 N=1000000 autocorr=0.00030831

Philox
tau=10 N=1000000 autocorr=-0.00013928
tau=20 N=1000000 autocorr=-0.00013370
tau=30 N=1000000 autocorr=0.00000410
tau=40 N=1000000 autocorr=-0.00019723
tau=50 N=1000000 autocorr=-0.00001218
tau=60 N=1000000 autocorr=-0.00015091
tau=70 N=1000000 autocorr=-0.00017069
tau=80 N=1000000 autocorr=-0.00010528
tau=90 N=1000000 autocorr=-0.00015212

MT1997
tau=10 N=1000000 autocorr=-0.00014089
tau=20 N=1000000 autocorr=-0.00008862
tau=30 N=1000000 autocorr=-0.00005420
tau=40 N=1000000 autocorr=-0.00026220
tau=50 N=1000000 autocorr=0.00042774
tau=60 N=1000000 autocorr=-0.00063585
tau=70 N=1000000 autocorr=0.00007771
tau=80 N=1000000 autocorr=0.00012340
tau=90 N=1000000 autocorr=-0.00053645
```

Рисунок 5. Три ГСЧ по критерию «автокорреляции» с N = 1000000.

ВЫВОДЫ

В ходе работы были изучены критерии проверки качества работы генераторов псевдослучайных чисел: χ^2 и автокорреляция. С помощью этих критериев были протестированы три генератора псевдослучайных чисел: PCG64, Philox, MT1997.

Из полученных результатов по критерию χ^2 можно сделать следующие выводы: у генератора MT1997 среднее отклонение эмпирических частот от теоретических меньше, чем у остальных генераторов, следовательно, данный генератор в проведенном эксперименте выдал более равномерное распределение, чем другие генераторы.

Тем не менее и остальные генераторы показывали значения χ^2 не превышающие критического. Это говорит о том, что для всех генераторов, участвующих в эксперименте, гипотезу о равномерном распределении нельзя опровергнуть. С увеличением количества точек результаты по критерию χ^2 перестают сильно отличаться. В последнем испытании все генераторы показали практически идентичный результат.

Коэффициент автокорреляции колеблется около 0, т. е. принимает как положительные значения, так и отрицательные, поэтому статистическая взаимосвязь между исходной и сдвинутой последовательностью пренебрежимо мала.

ПРИЛОЖЕНИЕ

Листинг

```
from numpy.random import Philox, PCG64, Generator
from numpy import random as nprandom
import random
import math
import sys

def math_expect(list):
    acc = 0.0
    n = len(list)
    for i in range(n - 1):
        acc += list[i]
    return acc / n


def dispersion(list, x):
    return math_expect(list) * math.pow(x, 2) -
math.pow(math_expect(list) - x, 2)

def compute_chi_square(N=100, k=10):
    step = 1 / k

    randlist = [(random.randrange(0, sys.maxsize) /
sys.maxsize) for i in range(N)]
    intervals = {i: 0 for i in range(k)}

    for i in range(N):
        for j in range(k):
            if randlist[i] < ((j + 1) * step):
                intervals[j] += 1
                break

    acc = 0.0
    for i in range(k):
        acc += math.pow(intervals[i], 2) * k

    return (acc / N) - N


def compute_chi_square(randlist, k=10):
    N = len(randlist)
    step = 1 / k

    intervals = {i: 0 for i in range(k)}

    for i in range(N):
        for j in range(k):
            if randlist[i] < ((j + 1) * step):
                intervals[j] += 1
                break

    result_chi_square.write("Intervals:\n")
    for i in range(k):
        result_chi_square.write("from {0:.2f} to {1:.2f}
= {2:d}\n".format(step * i, step * (i + 1),
intervals[i]))

        acc = 0.0
        for i in range(k):
            acc += math.pow(intervals[i], 2) * k

    return (acc / N) - N


def autocorrelation(N=100, offset=0):
    list = [random.randrange(0, sys.maxsize) for i in
range(N)]
    expect = math_expect(list)

    upValue = 0.0
    for i in range(N - offset):
        upValue += (list[i] - expect) * (list[i +
offset] - expect)
```

```
    bottomValue = 0.0
    for i in range(N):
        bottomValue += math.pow(list[i] - expect, 2)

    return upValue / bottomValue


def autocorrelation(x_i, tau=500):
    N = len(x_i)
    sum_x_i = 0.0

    for i in range(N):
        sum_x_i += x_i[i]

    sum_x_i_sqr = sum_x_i / N
    ksi_sqr = sum_x_i_sqr / N - math.pow(sum_x_i_sqr,
2.0)

    need_print = True


    for i in range(tau):
        acc = 0.0
        for j in range(1, N - tau):
            acc += ((x_i[j] - sum_x_i_sqr) * (x_i[j +
tau] - sum_x_i_sqr)) / ksi_sqr
        if need_print:
            result_autocorr.write("tau={0:d} N={1:d}
autocorr={2:.8f}\n".format(tau, N, acc / (N - tau)))
            need_print = False
```