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
Шаврин К.С.

Проверила:
ассистент кафедры ВС
Петухова Я.В.

Новосибирск, 2020

Содержание

Задание на проектирование

Необходимо убедиться в равномерности трех генераторов псевдослучайных чисел, с помощью критерия согласия Пирсона и автокорреляции.

Теория по проектной части

Проверка по критерию «хи-квадрат»

p_i — теоретическая вероятность попадания чисел в i-ый интервал (всего этих интервалов k) равна $p_i = 1/k$

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 имеет следующий вид.

1. Диапазон от 0 до 1 разбивается на k равных интервалов.
2. Запускается ГСЧ N раз (N должно быть велико, например, $N/k > 5$).
3. Определяется количество случайных чисел, попавших в каждый интервал
4. Вычисляется экспериментальное значение $\chi^2_{\text{набл}}$ по следующей формуле:

$$\chi^2_{\text{эксп.}} = \frac{(n_1 - p_1 \cdot N)^2}{p_1 \cdot N} + \frac{(n_2 - p_2 \cdot N)^2}{p_2 \cdot N} + \ldots + \frac{(n_k - p_k \cdot N)^2}{p_k \cdot N}$$

$$\chi^2_{\text{эксп.}} = \sum_{i=1}^{k} \frac{(n_i - p_i \cdot N)^2}{p_i \cdot N} = \frac{1}{N} \sum_{i=1}^{k} \left(\frac{n_i^2}{p_i} \right) - N$$

5. Выбирается уровень значимости α критерия, по таблице $\chi 2$-распределения находят квантиль $\chi_{\alpha,m} 2$;
6. Если $\chi 2_{\text{набл}} \leq \chi_{\alpha,m} 2$, то гипотеза не противоречит опытным данным, иначе отвергается;

Проверка по критерию автокорреляции

E_x – математическое ожидание
S^2 - выборочная дисперсия
$\hat{\alpha}(\tau)$ - автокорреляция
x_i - множество псевдослучайных чисел
τ - смещение

$E_x = \sum_{i=1}^{N} \frac{x_i}{N}$; $S^2 = \sum \frac{x_i^2}{N} - (E_x)^2$;

$\hat{\alpha}(\tau) = \frac{1}{(N-\tau)*S^2} \sum_{i=1}^{N-\tau} (x_i - E_x)(x_{i+\tau} - E_x)$;

В данной лабораторной работе были использованы три генератора псевдослучайных чисел:

1. PCG64 – это 64-битный конгруэнтный генератор перестановок который принимает на вход функцию перестановок.
2. SFC64 – это генератор псевдослучайных чисел основанный на кривой Пиано. В наиболее общем виде область действия такой функции может лежать в произвольном топологическом пространстве, но в наиболее часто изучаемых случаях область будет лежать в евклидовом пространстве.
3. Вихрь Мерсенна – это генератор псевдослучайных чисел основываемый на свойствах простых чисел Мерсенна.

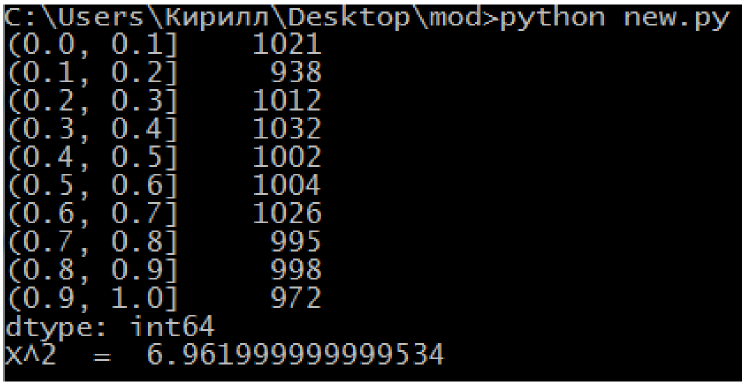
Пример работы программы:

Кол-во свободных степеней: $m = k - 1 = 9$.

Уровень значимости $\alpha = 0.1$.

Квантиль $\chi^2_{\text{кр}} = \chi_{\alpha=0.1,\, m=9} = 14{,}684$.

Оценка псевдослучайного генератора SFC64 с помощью критерия Пирсона, $k = 10$, $N = 10000$.

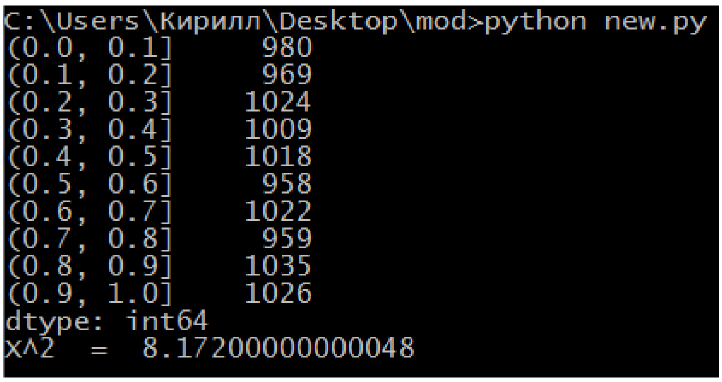


```
C:\Users\Кирилл\Desktop\mod>python new.py
(0.0, 0.1]    1021
(0.1, 0.2]     938
(0.2, 0.3]    1012
(0.3, 0.4]    1032
(0.4, 0.5]    1002
(0.5, 0.6]    1004
(0.6, 0.7]    1026
(0.7, 0.8]     995
(0.8, 0.9]     998
(0.9, 1.0]     972
dtype: int64
X^2  =  6.961999999999534
```

Рис.1. Запуск программы с использованием SFC64 при K=10, N=10000

$\chi^2_{\text{набл}} = 6{,}962 \leq \chi^2_{\text{кр}}$

Оценка псевдослучайного генератора PCG64 с помощью критерия Пирсона, $k = 10$, $N = 10000$.



```
C:\Users\Кирилл\Desktop\mod>python new.py
(0.0, 0.1]     980
(0.1, 0.2]     969
(0.2, 0.3]    1024
(0.3, 0.4]    1009
(0.4, 0.5]    1018
(0.5, 0.6]     958
(0.6, 0.7]    1022
(0.7, 0.8]     959
(0.8, 0.9]    1035
(0.9, 1.0]    1026
dtype: int64
X^2  =  8.17200000000048
```

Рис.2. Запуск программы с использованием PCG64 при K=10, N=10000

$\chi^2_{\text{набл}} = 8.172 \leq \chi^2_{\text{кр}}$

Оценка псевдослучайного генератора Вихрь Мерсенна с помощью критерия Пирсона, k = 10, N = 10000.

```
C:\Users\Кирилл\Desktop\mod>python new.py
(0.0, 0.1]    1012
(0.1, 0.2]     976
(0.2, 0.3]    1048
(0.3, 0.4]     993
(0.4, 0.5]     980
(0.5, 0.6]    1012
(0.6, 0.7]     986
(0.7, 0.8]    1014
(0.8, 0.9]     994
(0.9, 1.0]     985
dtype: int64
X^2  =  4.270000000000437
```

Рис.3. Запуск программы с использованием вихрь Мерсенна при K=10, N=10000

$$\chi^2_{\text{набл}} = 4.27 \leq \chi^2_{\text{кр}}$$

Оценка псевдослучайного генератора SFC64 с помощью автокорреляционной функции, k = 5, N = 10000.

```
Автокорреляция
tau ( 1 ) = 0.00985662454007967
tau ( 11 ) = 0.002959515103085928
tau ( 21 ) = -0.009030433172056564
tau ( 31 ) = -0.0036907502244416307
tau ( 41 ) = 0.001977699249354216
tau ( 51 ) = 0.00533830849442333
tau ( 61 ) = -0.004672352303370108
tau ( 71 ) = 0.005215987700816697
tau ( 81 ) = -0.0013249219272683703
tau ( 91 ) = -0.012043429740339968
tau ( 101 ) = -0.016566957358362378
tau ( 111 ) = 0.0002134764571851003
tau ( 121 ) = -0.011857595128698675
tau ( 131 ) = 0.005176597196464216
tau ( 141 ) = 0.010266211507929017
tau ( 151 ) = 0.005330120338821622
tau ( 161 ) = -0.010208043309930743
tau ( 171 ) = 0.0017164635774476495
tau ( 181 ) = 0.0057398803104735325
tau ( 191 ) = -0.004500572064464113
```

Рис.4. Запуск программы с использованием SFC64 при K=5, N=10000

Оценка псевдослучайного генератора вихрь Мерсенна с помощью автокорреляционной функции, k = 5, N = 10000.

```
Автокорреляция
tau ( 1 ) = -0.0005058237855334239
tau ( 11 ) = -0.012680269469829777
tau ( 21 ) = 0.0081379114394118
tau ( 31 ) = 0.006045559414479578
tau ( 41 ) = -0.022215141392329843
tau ( 51 ) = 0.0026050870171652946
tau ( 61 ) = -0.021032763431551277
tau ( 71 ) = -0.00220151390249791
tau ( 81 ) = -0.006941117629491806
tau ( 91 ) = -0.006768980796972737
tau ( 101 ) = 0.011427307386155836
tau ( 111 ) = -0.014671624431704685
tau ( 121 ) = 0.003983728331891491
tau ( 131 ) = -0.0047856004262451365
tau ( 141 ) = 0.010734797043341718
tau ( 151 ) = -0.004815283074738728
tau ( 161 ) = -0.0025233213897930080
tau ( 171 ) = -0.015653859173522394
tau ( 181 ) = 0.007336543974184079
tau ( 191 ) = 0.008102026052817585
```

Рис.5. Запуск программы с использованием вихрь Мерсенна при K=5, N=10000

Оценка псевдослучайного генератора PCG64 с помощью автокорреляционной функции, $k = 5$, $N = 10000$.

```
Автокорреляция
tau ( 1 ) = 0.017473531825791193
tau ( 11 ) = 0.010202236044964117
tau ( 21 ) = 0.005108312590073678
tau ( 31 ) = -0.0005725585338264351
tau ( 41 ) = -0.007346184918140326
tau ( 51 ) = -0.010680083550211463
tau ( 61 ) = -0.015619375990680811
tau ( 71 ) = 0.006296716188762564
tau ( 81 ) = 0.0043926405487448575
tau ( 91 ) = -0.017567819943105772
tau ( 101 ) = 0.000586084150373967
tau ( 111 ) = -0.007103383828647346
tau ( 121 ) = 0.002248942915434774
tau ( 131 ) = -0.011134761035685604
tau ( 141 ) = -0.024904679522478906
tau ( 151 ) = -0.010099762077338158
tau ( 161 ) = 0.029947189853294257
tau ( 171 ) = -0.004483743344189943
tau ( 181 ) = -0.0002886671493085219
tau ( 191 ) = 0.018834135068505214
```

Рис.6. Запуск программы с использованием PCG64 при K=5, N=10000

Выводы по результатам проектирования

В ходе лабораторной работы были проверены три генератора псевдослучайных чисел: PCG64, вихрь Мерсенна и SFC64 на языке программирования python по критерию Пирсона и автокорреляции. По этим критериям все три генератора показали равномерное распределение.

При подтверждении нулевой гипотезы используем неравенство $\chi 2_{набл} \leq \chi_{кр}$, то есть Значение $\chi 2_{набл}$, используемое в критерии Пирсона, не должно превышать критического значения $\chi 2_{кр}$. Из первых трех рисунков: $\chi 2_{sfc64} = 6{,}962$, $\chi 2_{pcg64} = 8{,}172$, $\chi 2_{twist} = 4{,}27$, $\chi_{кр} = 14{,}684$. Из этого можно сделать вывод, что гипотеза о равенстве (согласии) частот не отклоняется. Следовательно, нулевая гипотеза подтверждена.

Генератор случайных чисел считается хорошим, если при величине смещения не равной нулю, модуль автокорреляции меньше 0.03, то есть |R| < 0.03. Значения R в трех экспериментах для трех генераторов при разных τ удовлетворяет условию. Следовательно, можно сказать, что числа полученные такими генераторами, близки к случайным. При увеличении общего числа точек или точек в каждом интервале отклонение также уменьшается. Автокорреляция при увеличении общего кол-ва чисел приближается к нулю, а при увеличении кол-во интервалов приближения к нулю не наблюдается.

Ближе всего к равномерному это результаты полученные с помощью генератора вихря Мерсенна, так как отклонение экспериментальных значений от теоритических больше чем при других запусках.

Листинг программы

```
import random
import _random
import time
import pandas as pn
import numpy as np
import matplotlib.pyplot as plt
from numpy.random import Generator, SFC64, PCG64
def x2(N, k):
	sum = 0
	V = 0
	pi = 1/k
	np.random.seed(int(time.time()))
	#series = pn.Series([Generator(SFC64()).random() for s in range(N)])
	massiv = []
	for i in range(N):
		massiv.append(random.random())
	#series = pn.Series([Generator(PCG64()).random() for s in range(N)])
	np.random.seed(int(time.time()))
	series = pn.Series(np.random.rand(N))
	bins = np.linspace(0, 1, k+1)
	res = series.groupby(pn.cut(series, bins=bins)).count()
	print(res)
	ymass = []
	for i in res:
		ymass.append(i)
		sum += (i * i) / pi
	V = (sum / N) - N
	print("X^2  = ", V)

def acr(N, k):
	np.random.seed(int(time.time()))
	#series = pn.Series([Generator(SFC64()).random() for s in range(N)])
	#massiv = []
	#for i in range(N):
	#	massiv.append(random.random())
	#series = pn.Series(massiv)
	massiv = []
	for i in range(N):
		massiv.append(random.SystemRandom().random())
	series = pn.Series([Generator(PCG64()).random() for s in range(N)])

	print("Автокорреляция")
	math = 0
	for i in range(1, N):
		math += series[i]
	math = math / N
	disp = 0
	for i in range(1, N):
		disp += series[i] * series[i]
	disp = (disp / N) - math ** 2
	tau = 0
	tau_mass = []
	for r in range(1, 200, 10):
		for i in range(1, N - r):
			tau += (series[i] - math) * (series[i + r] - math)
		tau_mass.append(tau / (disp * N))

		print("tau (", r, ") =", tau / (disp * N))
		tau = 0
	return np.mean(tau_mass)
acr(10000, 5)
```