

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

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №0
По дисциплине «Моделирование»

Выполнил:
Студент гр. ИВ-622
Ольман Р. Ю.

Проверила:
Ассистент Кафедры ВС
Петухова Я. В.

Новосибирск 2020

Оглавление

Постановка задачи.....	3
Теоретические сведения	3
Ход работы.....	5
Заключение.....	7
Листинг	8

Постановка задачи

Необходимо взять готовую реализацию трех генераторов псевдослучайных чисел на отрезке $[0, 1]$ и убедиться в их равномерном распределении, используя такие параметры, как распределение Пирсона и автокорреляции.

Теоретические сведения

Пусть X – исследуемая случайная величина. Требуется проверить гипотезу H_0 о том, что данная случайная величина подчиняется закону распределения $F(x)$. Для этого необходимо произвести выборку из n независимых наблюдений над случайной величиной $X: X^n = (x_1, \dots, x_n), x_i \in [a, b], \forall i = 1 \dots n$, и по ней построить эмпирический закон распределения $F'(x)$ случайной величины X .

Критерий согласия Пирсон – метод, который позволяет оценить значимость различий между фактическим (выявленным в результате исследования) количеством исходов или качественных характеристик выборки, попадающих в каждую категорию, и теоретическим количеством, которое можно ожидать в изучаемых группах при справедливости нулевой гипотезы.

Процедура проверки гипотез с использованием критериев типа χ^2 предусматривает группирование наблюдений. Область определения случайной величины разбивают на k непересекающихся интервалов граничными точками $(x_0, x_1, x_2, \dots, x_k)$, где x_0 – нижняя грань области определения случайной величины; x_k – верхняя грань. В соответствии с заданным разбиением подсчитывают число n_i выборочных значений, попавших в i -й интервал, и вероятности попадания в интервал, соответствующие теоретическому закону с функцией распределения. Рассчитывается критерий «хи-квадрат» по формуле:

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

- p_i – теоретическая вероятность попадания чисел в i -ый интервал

- k – количество интервалов
- N – общее количество сгенерированных чисел
- n_i – количество попавших чисел в интервал
- χ^2 – критерий, который позволяет определить, удовлетворяет ли генератор случайных чисел требованиям равномерного распределения или нет

Если $\chi_{\text{экс}}^2 \leq \chi_{\text{таб}}^2$, то гипотеза не противоречит опытным данным, иначе отвергается.

Условие применения критерия Пирсона является наличие в каждом интервале не менее пяти наблюдений. И следует помнить, что для равномерного распределения все значения вероятности одинаковы. Далее надо провести корреляционный анализ. Корреляционный анализ – популярный метод статистического исследования, который используется для выявления степени зависимости одного показателя от другого.

Автокорреляция – статистическая взаимосвязь между последовательностями величин одного ряда, взятыми со сдвигом, например, для случайного процесса – со сдвигом по времени.

$$a(\tau) = \frac{\sum_{i=1}^{N-\tau} (x_i - Ex) * (x_{i+\tau} - Ex)}{(N - \tau) * S^2(x)}$$

$E(x)$ – математическое ожидание – среднее значение случайной величины при стремлении количества выборок или количества ее измерений:

$$E(x) = \frac{1}{N} \sum_{i=1}^N X_i$$

$S^2(x)$ – выборочная дисперсия случайной величины – это оценка теоретической дисперсии распределения, рассчитанная на основе данных выборки:

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

$x_{i+\tau}$ – множество значений другой случайной величины (полученной из значений прошлой случайной величины, но с некоторым смещением) n – мощность множества случайных величин τ – смещение последовательности Генераторы случайных чисел:

Ход работы

Для выполнения данной лабораторной работы было выбрано три генератора случайных чисел, независимых друг от друга:

- В качестве первого генератора был взят генератор случайных чисел из стандартной библиотеки `random()` языка программирования Python.
- Вторым генератором случайных чисел был генератор `rand()` из библиотеки модуля NumPy для Python.
- А третьим генератором стал `SystemRandom()`.

Была рассчитана автокорреляция для каждого ГСЧ при $N = 1.000.000$ и $k = 1.000$, где $\alpha(\tau)$ – автокорреляция, τ – смещение интервалов.

```
Генератор Random()
N: 1000000
K: 1000
Хи квадрат: 991.3479999999992
a(tao) 0.000102
tao: 1
a(tao) -0.000047
tao: 2
a(tao) -0.000049
tao: 3
a(tao) 0.000031
tao: 4
a(tao) 0.000050
tao: 5
a(tao) 0.000084
tao: 6
a(tao) 0.000051
tao: 7
a(tao) -0.000133
tao: 8
a(tao) 0.000001
tao: 9
a(tao) 0.000091
tao: 10
a(tao) 0.000074
tao: 11
a(tao) 0.000010
tao: 12
a(tao) 0.000069
tao: 13
a(tao) -0.000060
tao: 14
a(tao) 0.000012
tao: 15
```

```
Генератор модуля NumPy
N: 1000000
K: 1000
Хи квадрат: 1015.1500000000002
a(tao) 0.000007
tao: 1
a(tao) 0.000012
tao: 2
a(tao) -0.000009
tao: 3
a(tao) 0.000003
tao: 4
a(tao) 0.000002
tao: 5
a(tao) -0.000006
tao: 6
a(tao) 0.000000
tao: 7
a(tao) -0.000002
tao: 8
a(tao) 0.000001
tao: 9
a(tao) -0.000009
tao: 10
a(tao) 0.000003
tao: 11
a(tao) -0.000001
tao: 12
a(tao) -0.000012
tao: 13
a(tao) -0.000003
tao: 14
a(tao) -0.000001
tao: 15
```

```
Генератор SystemRandom()
N: 1000000
K: 1000
Хи квадрат: 985.2379999999991
a(tao) 0.000014
tao: 1
a(tao) 0.000005
tao: 2
a(tao) 0.000012
tao: 3
a(tao) -0.000001
tao: 4
a(tao) -0.000005
tao: 5
a(tao) -0.000003
tao: 6
a(tao) -0.000001
tao: 7
a(tao) -0.000001
tao: 8
a(tao) -0.000002
tao: 9
a(tao) -0.000005
tao: 10
a(tao) 0.000006
tao: 11
a(tao) -0.000005
tao: 12
a(tao) -0.000013
tao: 13
a(tao) 0.000010
tao: 14
a(tao) -0.000002
tao: 15
```

Рисунок 1, 2, 3 –Random(), Rand(), SystemRand()

Исходя из этих данных можно сделать вывод, что числа действительно являются случайными, так как значения $\alpha(\tau)$ стремятся к нулю.

Далее было несколько попыток генерации случайных чисел каждым ГСЧ с изменением количества чисел и количества интервалов. Все результаты показаны в таблице 1.

Таблица 1 – показатели критерия «хи квадрат» при разных N и k

ГСЧ	N = 1.000.000 k = 1.000	N = 1.000.000 k = 100	N = 10.000 k = 1.000
Random()	$\chi^2 = 978.865$	$\chi^2 = 89.26$	$\chi^2 = 979.199$
NumPy Rand()	$\chi^2 = 967.755$	$\chi^2 = 87.353$	$\chi^2 = 923$
SystemRamdom()	$\chi^2 = 952.463$	$\chi^2 = 93.321$	$\chi^2 = 956.4$
Табличное значение χ^2	$\chi^2 = 1142.848$	$\chi^2 = 134,642$	$\chi^2 = 1142.848$

Заключение

В данной лабораторной работе были реализованы три независимых друг от друга генератора случайных чисел языка программирования Python и реализованы три программы для генерации псевдослучайных чисел, расчетов автокорреляции и критерия согласия Пирсона.

Понимание автокорреляция дает нам знать, то насколько ряды чисел генерируются случайным образом, если она стремится к нулю, то числа являются случайными. Из наших опытов видно, что все три генератора являются действительно псевдослучайными. Так же стоит заметить, что коэффициент автокорреляции напрямую зависит от количества точек на интервалах. К примеру, если уменьшить количество интервалов, то, следовательно, точек на меньшем количестве интервалов будет больше и коэффициент автокорреляции будет стремиться к нулю. А если увеличить количество интервалов, а точек уменьшить, то на каждый интервал будет попадать неравное количество точек и как следствие коэффициент автокорреляции будет возрастать.

Во время проведения экспериментов было выявлено какую информацию дает Критерий согласия Пирсона, а именно:

1. Чем больше отклонение, тем менее равномерным становится распределение чисел на интервалах.
2. Так как значение, используемое в критерии согласия Пирсона, не превышало критического значения, следует, что гипотезу о равномерном распределении нельзя отвергнуть.

И по результатам опытов следует сказать, что критерии «хи квадрат» которые равны $\chi^2 = 991.347$, $\chi^2 = 1015.15$ и $\chi^2 = 985.237$ соответственно для каждого из них принимается гипотеза о равновероятном распределении в ГСЧ исходя их табличных значений для каждого случая.

Листинг

gen_1.py

```
1  import random
2
3  N = int(input("N: "))
4  K = int(input("K: "))
5  y = range(N)
6  x_1 = []
7  numbers_1 = [0] * K
8  ksi = 0
9
10
11 def gen_1():
12     for k in y:
13         rand_1 = random.uniform(0, 1)
14         x_1.append(rand_1)
15         inter_1 = rand_1 / (1 / K)
16         numbers_1[int(inter_1)] += 1
17     print(f'Генератор Random()')
18     f'Всего точек: {len(x_1)}'
19
20     p_i = 1 / K
21     i = 0
22     ksi = 0
23     while i != K:
24         ksi += ((numbers_1[i] - p_i * N) ** 2) / (p_i * N)
25         i += 1
26     print(f'Хи квадрат: {ksi}')
27
28     i = 0
29     Ex = 0
30     while i != N:
31         Ex += x_1[i]
32         i += 1
33     Ex = (1 / N) * Ex
34
35     i = 0
36     S2x = 0
37     while i != N:
38         S2x += ((x_1[i]) ** 2) - (Ex ** 2)
39         i += 1
40     S2x = S2x / N
41
42     offset = 1
43     while offset <= K / 2:
44         a = 0
45         i = 0
46         while i < N - offset:
47             a += (x_1[i] - Ex) * (x_1[i + offset] - Ex)
48             i += 1
49         a = a / (N - offset) * S2x
50         print(f'a(tao) %f' % a)
51         print(f'tao: {offset}')
52         offset += 1
53
5
```


gen_2.py

```
1  from numpy.random import rand
2
3  N = int(input("N: "))
4  K = int(input("K: "))
5  y = range(N)
6  x_2 = []
7  numbers_2 = [0] * K
8  ksi = 0
9  offset = 1
10
11
12 def gen_2():
13     for k in y:
14         rand_2 = rand()
15         x_2.append(rand_2)
16         inter_2 = rand_2 / (1 / K)
17         numbers_2[int(inter_2)] += 1
18     print(f'Генератор модуля NumPy'
19           f'Всего точек: {(len(x_2))}')
20
21     p_i = 1 / K
22     i = 0
23     ksi = 0
24     while i != K:
25         ksi += ((numbers_2[i] - p_i * N) ** 2) / (p_i * N)
26         i += 1
27     print(f'Хи квадрат: {ksi}')
28
29     i = 0
30     Ex = 0
31     while i != N:
32         Ex += x_2[i]
33         i += 1
34     Ex = (1 / N) * Ex
35
36     i = 0
37     S2x = 0
38     while i != N:
39         S2x += ((x_2[i]) ** 2) - (Ex ** 2)
40         i += 1
41     S2x = S2x / N
42
43     offset = 1
44     while offset <= K / 2:
45         a = 0
46         i = 0
47         while i < N - offset:
48             a += (x_2[i] - Ex) * (x_2[i + offset] - Ex)
49             i += 1
50         a = a / (N - offset) * S2x
51         print(f'a(tao) %f' % a)
52         print(f'tao: {offset}')
53         offset += 1
54
```

gen_3.py

```
1 import random
2
3 print(f'Генератор SystemRandom()')
4 N = int(input("N: "))
5 K = int(input("K: "))
6 y = range(N)
7 x_3 = []
8 numbers_3 = [0] * K
9 ksi = 0
10
11
12 def gen_3():
13     for k in y:
14         rand_3 = random.SystemRandom().random()
15         x_3.append(rand_3)
16         inter_3 = rand_3 / (1 / K)
17         numbers_3[int(inter_3)] += 1
18
19     p_i = 1 / K
20     i = 0
21     ksi = 0
22     while i != K:
23         ksi += (((numbers_3[i] - p_i * N) ** 2) / (p_i * N))
24         i += 1
25     print(f'Хи квадрат: {ksi}')
26
27     i = 0
28     Ex = 0
29     while i != N:
30         Ex += x_3[i]
31         i += 1
32     Ex = (1 / N) * Ex
33
34     i = 0
35     S2x = 0
36     while i != N:
37         S2x += ((x_3[i]) ** 2) - (Ex ** 2)
38         i += 1
39     S2x = S2x / N
40
41     offset = 1
42     while offset <= K / 2:
43         a = 0
44         i = 0
45         while i < N - offset:
46             a += (x_3[i] - Ex) * (x_3[i + offset] - Ex)
47             i += 1
48         a = a / (N - offset) * S2x
49         print(f'a(tao) %f' % a)
50         print(f'tao: {offset}')
51         offset += 1
52
53
```