

ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ _____

ПРЕПОДАВАТЕЛЬ

Профессор				Татарникова Т. М.
должность, уч. степень, звание		подпись, дата		инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №2

МОДЕЛИРОВАНИЕ БАЗОВОЙ СЛУЧАЙНОЙ ВЕЛИЧИНЫ

Вариант 5

по курсу: Моделирование систем

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	4128			Воробьев В. А.
			подпись, дата	инициалы, фамилия

Санкт-Петербург 2024

СОДЕРЖАНИЕ

1	Постановка задачи	3
1.1	Цель работы	3
1.2	Содержание отчета	3
1.3	Вариант	3
2	Выполнение работы	4
2.1	Математическая модель	4
2.2	Реализация модели	4
2.3	Результат моделирования	7
2.4	Анализ	9
3	Вывод	11
	ПРИЛОЖЕНИЕ	12

1 Постановка задачи

1.1 Цель работы

Построить датчик базовой случайной величины по заданному алгоритму и выполнить тестирование датчика на соответствие основным свойствам базовой случайной величины.

1.2 Содержание отчета

1. Цель, задание и последовательность выполнения работы.
2. Результаты сравнений математического ожидания и дисперсии псевдослучайных значений z_i с теоретическими значениями M и D .
3. Гистограмма распределения относительных частот попаданий псевдослучайных величин в отрезки интервала $[0, 1]$.
4. Графики зависимости коэффициента корреляции для $s = 2, s = 5, s = 10$.
5. Выводы о результатах моделирования БСВ.

1.3 Вариант

5 Вариант, аддитивный генератор Fish:

$$\begin{aligned} A_i &= (A_{i-55} + A_{i-24}) \bmod (2^{32}) \\ B_i &= (B_{i-52} + B_{i-19}) \bmod (2^{32}) \\ z_i &= \frac{A_i}{2^{32}}, z_{i+1} = \frac{B_i}{2^{32}} \end{aligned} \quad (1)$$

Состоит из двух аддитивных генераторов A_i и B_i , начальные состояния которых создаются Random. Эти последовательности прореживаются попарно в зависимости от случая: если значение младшего значащего бита B_i равно 1, то пара используется, если 0 - игнорируется.

2 Выполнение работы

Исходный код доступен в Приложении и репозитории GitHub (URI - https://github.com/vladcto/suai-labs/tree/main/6_semester).

2.1 Математическая модель

$$M(z) = 0.5, D(z) = 1/12, \quad (2)$$

где:

- $M(z)$ - математическое ожидание;
- $D(z)$ - дисперсия.

$$f(z) = 1, (0 \leq z \leq 1), \quad (3)$$

где:

- $f(z)$ - плотность распределения;
- z - непрерывная случайная величина.

$$z_{i+1} = f(z_i), \quad (4)$$

где:

- $f(z_i)$ - функция программного датчика БСВ.

$$\hat{R} = 12 \frac{1}{T-s} \left(\sum_{i=1}^{T-s} z_i \cdot z_{i+s} \right) - 3, \quad (5)$$

где:

- \hat{R} - коэффициент корреляции для значений БСВ.

2.2 Реализация модели

Для расчета характеристик был написан скрипт на Python, который изображен ниже. Выбор пал на Python ввиду его простоты, удобства и хорошей поддержке математических операций.

Листинг lcg.py:

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from fish_generator import FishGenerator
4
5
6 def plot_histogram(generated_values, K):
```

```

7     bins = [i/K for i in range(K+1)]
8     values_A = [value[0] for value in generated_values]
9
10    counts, _ = np.histogram(values_A, bins=bins)
11
12    frequencies = counts / len(values_A)
13    plt.bar(bins[:-1], frequencies, width=1/K, align='edge')
14    plt.xlabel('Значение')
15    plt.ylabel('Частота')
16    plt.show()
17
18
19    def calculate_correlation(generated_values, s, T):
20        values_A = [value[0] for value in generated_values[:T]]
21
22        values_A1 = values_A[:-s]
23        values_A2 = values_A[s:]
24
25        sum_product = sum(a*b for a, b in zip(values_A1,
26                                              values_A2))
27        if T - s != 0:
28            R_hat = 12 * sum_product / (T - s) - 3
29        else:
30            R_hat = None
31        return R_hat
32
33    length = 10000
34    fish_gen = FishGenerator().generate()
35    generated_values = [next(fish_gen) for _ in range(length)]
36
37    # 2 task
38    values_A, values_B = zip(*generated_values)
39    mean_A = np.mean(values_A)
40    variance_A = np.var(values_A)
41
42    mean_B = np.mean(values_B)
43    variance_B = np.var(values_B)
44
45    print(f"Ошибка МО A: {abs(0.5 - mean_A)}")
46    print(f"Ошибка МО B: {abs(0.5 - mean_B)}")

```

```

47 print(f"Ошибка дисперсии A: {abs(1/12 - variance_A)}")
48 print(f"Ошибка дисперсии B: {abs(1/12 - variance_B)}")
49
50 # 3 task
51 plot_histogram(generated_values, K=10)
52
53 # 4 task
54 s_values = [2, 5, 10]
55 T_values = range(10, len(generated_values), 10)
56
57 correlations = [
58     [calculate_correlation(generated_values, s, T) for T in
59      T_values]
60     for s in s_values
61 ]
62 for s, correlation in zip(s_values, correlations):
63     plt.plot(T_values, correlation, label=f's={s}')
64
65 plt.xlabel('T')
66 plt.ylabel('R_hat')
67 plt.legend()
68 plt.show()

```

Листинг fish_generator.py:

```

1 import random
2
3
4 class FishGenerator:
5     def __init__(self, seed=None):
6         self.A = [random.randint(0, 2**32 - 1) for _ in range
7                    (55)]
8         self.B = [random.randint(0, 2**32 - 1) for _ in range
9                    (52)]
10        self.index_A = 0
11        self.index_B = 0
12        random.seed(seed)
13
14    def generate(self):
15        while True:
16            self.A[self.index_A] = (
17                self.A[(self.index_A - 55) % 55] + self.A[(

```

```

        self.index_A - 24) % 55]) % (2**32)
16         self.B[self.index_B] = (
17             self.B[(self.index_B - 52) % 52] + self.B[(
                self.index_B - 19) % 52]) % (2**32)
18         if self.B[self.index_B] & 1:
19             yield self.A[self.index_A] / (2**32), self.B[
                self.index_B] / (2**32)
20         self.index_A = (self.index_A + 1) % 55
21         self.index_B = (self.index_B + 1) % 52

```

2.3 Результат моделирования

Для начала была подсчитана ошибка фактического и теоретического МО и дисперсии. Результат изображен на рисунке 2.1.

```

Ошибка МО А: 0.0038210243285633494
Ошибка МО В: 0.0010430984205566096
Ошибка дисперсии А: 0.00043322179779030356
Ошибка дисперсии В: 0.00016016455947871766

```

Рисунок 2.1 - Ошибка МО и дисперсии

Была построена гистограмма распределения изображенная на рисунке 2.2.

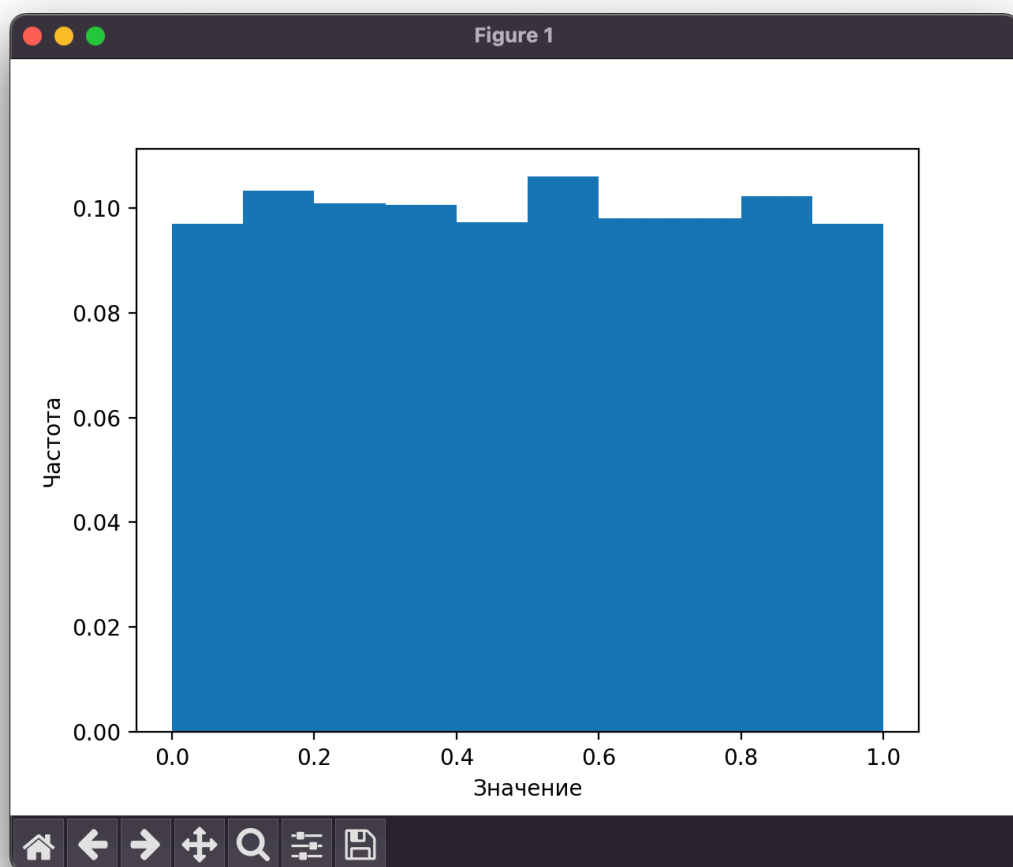


Рисунок 2.2 - Гистограмма распределения БСВ

Были получены графики зависимости коэффициента корреляции R , полученные по формуле (5) и представленные на рисунке 2.3.

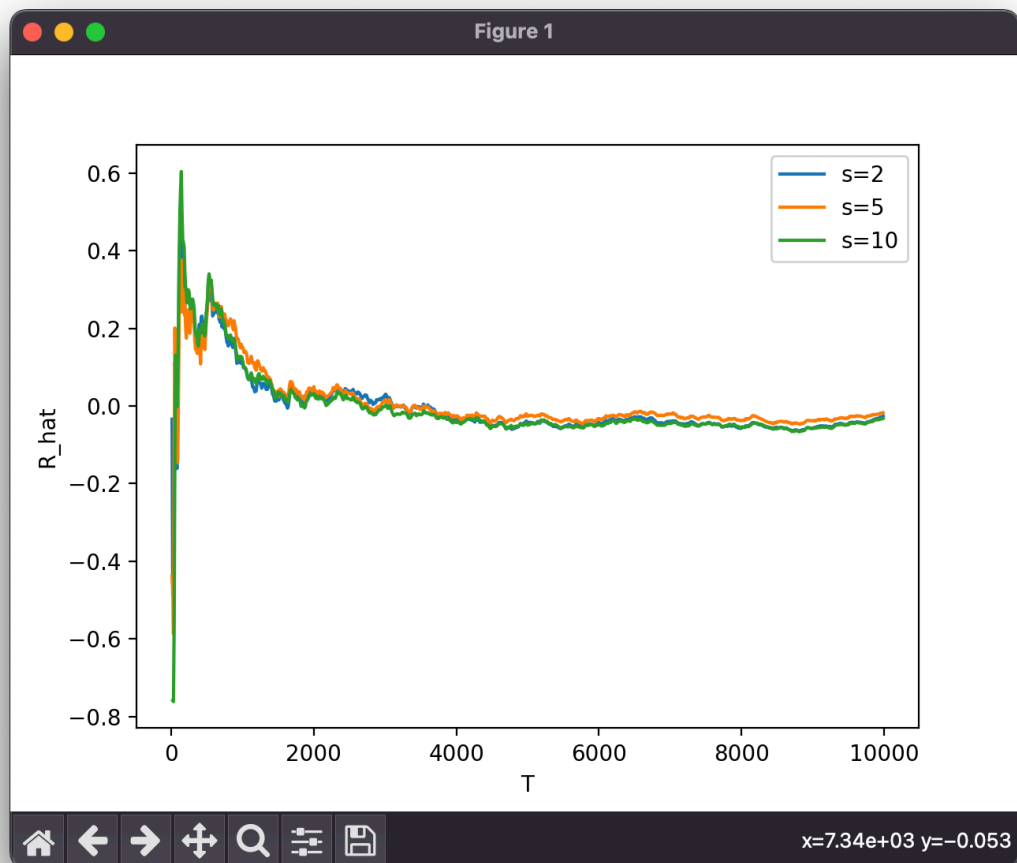


Рисунок 2.3 - Графики корреляции

2.4 Анализ

В результате проведенного анализа работы следует отметить, что разработанный датчик базовой случайной величины, основанный на аддитивном генераторе Fish, успешно соответствует теоретическим ожиданиям. Согласованность математического ожидания M и дисперсии D полученных псевдослучайных значений с соответствующими теоретическими значениями подтверждает, а также гистограмма распределения на интервале $[0, 1]$ подтверждает верную реализацию аддитивного генератора Fish.

На основе графика зависимости коэффициента корреляции для различных сдвигов s , можно выделить, что для всех $s(2, 5, 10)$ графики стремятся к нулю, и имеют одинаковую кривизну. Это может говорить о том, что значения генератора являются статистически независимыми.

В целом, полученные результаты и анализ указывают на успешную разработку и тестирование датчика базовой случайной величины на основе ад-

дитивного генератора Fish. Этот датчик представляет собой эффективный инструмент для генерации псевдослучайных последовательностей с удовлетворительными статистическими характеристиками.

3 Вывод

В результате выполнения данной работы был разработан и исследован датчик базовой случайной величины (БСВ) на основе аддитивного генератора Fish. В ходе построения датчика был реализован алгоритм, объединяющий два аддитивных генератора A_i и B_i , их начальные состояния создаются с использованием функции Random. Последовательности затем прореживаются попарно в зависимости от значения младшего значащего бита B_i . Результаты тестирования данного датчика на соответствие основным свойствам БСВ позволяют сделать вывод о его эффективности в создании псевдослучайных последовательностей.

Для оценки качества полученных псевдослучайных значений проведено сравнение математического ожидания и дисперсии с соответствующими теоретическими значениями. Предложенный датчик демонстрирует согласованность с теоретическими характеристиками, что подтверждает его пригодность для моделирования случайных величин. Гистограмма распределения относительных частот попаданий в отрезки интервала $[0, 1]$ дополнительно подтверждает равномерность распределения, что является важным свойством для многих приложений в области статистики и моделирования.

Графики зависимости коэффициента корреляции для различных сдвигов s позволяют оценить корреляционные свойства последовательности псевдослучайных величин. Обнаруживается устойчивость к корреляции при различных значениях s , что является положительным результатом. В целом, разработанный датчик базовой случайной величины на основе аддитивного генератора Fish успешно соответствует поставленной цели, предоставляя надежный инструмент для генерации случайных последовательностей в различных областях применения.

ПРИЛОЖЕНИЕ

```
1 import random
2
3
4 class FishGenerator:
5     def __init__(self, seed=None):
6         self.A = [random.randint(0, 2**32 - 1) for _ in range
7                     (55)]
8         self.B = [random.randint(0, 2**32 - 1) for _ in range
9                     (52)]
10        self.index_A = 0
11        self.index_B = 0
12        random.seed(seed)
13
14    def generate(self):
15        while True:
16            self.A[self.index_A] = (
17                self.A[(self.index_A - 55) % 55] + self.A[(
18                    self.index_A - 24) % 55]) % (2**32)
19            self.B[self.index_B] = (
20                self.B[(self.index_B - 52) % 52] + self.B[(
21                    self.index_B - 19) % 52]) % (2**32)
22            if self.B[self.index_B] & 1:
23                yield self.A[self.index_A] / (2**32), self.B[
24                    self.index_B] / (2**32)
25            self.index_A = (self.index_A + 1) % 55
26            self.index_B = (self.index_B + 1) % 52
```

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from fish_generator import FishGenerator
4
5
6 def plot_histogram(generated_values, K):
7     bins = [i/K for i in range(K+1)]
8     values_A = [value[0] for value in generated_values]
9
10    counts, _ = np.histogram(values_A, bins=bins)
11
12    frequencies = counts / len(values_A)
13    plt.bar(bins[:-1], frequencies, width=1/K, align='edge')
```

```

14     plt.xlabel('Значение')
15     plt.ylabel('Частота')
16     plt.show()
17
18
19 def calculate_correlation(generated_values, s, T):
20     values_A = [value[0] for value in generated_values[:T]]
21
22     values_A1 = values_A[: -s]
23     values_A2 = values_A[s:]
24
25     sum_product = sum(a*b for a, b in zip(values_A1,
26                                           values_A2))
27     if T - s != 0:
28         R_hat = 12 * sum_product / (T - s) - 3
29     else:
30         R_hat = None
31     return R_hat
32
33 length = 10000
34 fish_gen = FishGenerator().generate()
35 generated_values = [next(fish_gen) for _ in range(length)]
36
37 # 2 task
38 values_A, values_B = zip(*generated_values)
39 mean_A = np.mean(values_A)
40 variance_A = np.var(values_A)
41
42 mean_B = np.mean(values_B)
43 variance_B = np.var(values_B)
44
45 print(f"Ошибка МО A: {abs(0.5 - mean_A)}")
46 print(f"Ошибка МО B: {abs(0.5 - mean_B)}")
47 print(f"Ошибка дисперсии A: {abs(1/12 - variance_A)}")
48 print(f"Ошибка дисперсии B: {abs(1/12 - variance_B)}")
49
50 # 3 task
51 plot_histogram(generated_values, K=10)
52
53 # 4 task

```

```

54 s_values = [2, 5, 10]
55 T_values = range(10, len(generated_values), 10)
56
57 correlations = [
58     [calculate_correlation(generated_values, s, T) for T in
59      T_values]
60     for s in s_values
61 ]
62 for s, correlation in zip(s_values, correlations):
63     plt.plot(T_values, correlation, label=f's={s}')
64
65 plt.xlabel('T')
66 plt.ylabel('R_hat')
67 plt.legend()
68 plt.show()

```