Лосева Елизавета Юрьевна, группа 8-2

Лабораторная работа № 1

Вариант № 16

Моделирование случайных величин

Цель работы

Исследовать алгоритмы генерации случайных величин в среде Python. Научиться определять значения параметров случайной величины

Задание

Описание варианта: Биномиальное распределение,"n = 42, p = 0.1"

1. Постройте график зависимости значения выборочного математического ожидания от числа реализаций СВ. Так же отобразите на графике значение математического ожидания, вычисленное на основе соотношений из таблицы.

2. Постройте график зависимости значения выборочной дисперсии от числа реализаций СВ. Так же отобразите на графике значение дисперсии, вычисленное на основе соотношений из таблицы.

3. Выполните визуализацию эмпирической плотности распределения СВ (гистограммы с необходимой нормировкой). Так же отобразите на рисунке график плотности распределения, определённой на основе соотношений из таблицы. Постройте график зависимости значения средней абсолютной разности эмпирических и теоретических значений плотности распределения от числа реализаций СВ.

Код программы

```
import matplotlib.pyplot as plt
import numpy as np
import math

# Параметры распределения
n = 42   # число испытаний
p = 0.1  # вероятность успеха

# Теоретические характеристики
m_theory = n * p
D_theory = n * p * (1 - p)

print(f"Теоретическое мат. ожидание: {m_theory:.3f}")
print(f"Теоретическая дисперсия: {D_theory:.3f}")

# Генерация одной случайной величины Binomial(n, p)
# Используем метод Бернулли: X = сумма n независимых {0,1} с вероятностью успеха
p
def generate_binomial(n, p):
    count = 0
    for _ in range(n):
        u = np.random.rand()  # равномерное [0,1)
        if u < p:
            count += 1
    return count
```

```python
# Теоретическая pmf биномиального распределения
def binomial_pmf(k, n, p):
    # P(X=k) = C(n,k) * p^k * (1-p)^(n-k)
    return math.comb(n, k) * (p ** k) * ((1 - p) ** (n - k))

# Генерация выборки
np.random.seed(42)
n_max = 10000
n_values = np.arange(1, n_max + 1)
all_samples = np.array([generate_binomial(n, p) for _ in range(n_max)])

# Расчет выборочных характеристик
cumulative_mean = np.cumsum(all_samples) / n_values
cumulative_var = np.zeros(n_max)

for k in range(1, n_max):
    sample_k = all_samples[:k+1]
    mean_k = np.mean(sample_k)
    cumulative_var[k] = np.sum((sample_k - mean_k) ** 2) / k  # несмещённая
оценка с (k) в знаменателе

# Задание 1: График выборочного мат. ожидания
plt.figure(figsize=(10, 6))
plt.plot(n_values, cumulative_mean, label='Выборочное мат. ожидание')
plt.axhline(y=m_theory, color='r', linestyle='--', label='Теоретическое мат.
ожидание')
plt.xlabel('Число реализаций')
plt.ylabel('Мат. ожидание')
plt.title('Зависимость выборочного мат. ожидания от числа реализаций')
plt.legend()
plt.grid(True)
plt.show()

# Задание 2: График выборочной дисперсии
plt.figure(figsize=(10, 6))
plt.plot(n_values, cumulative_var, label='Выборочная дисперсия')
plt.axhline(y=D_theory, color='r', linestyle='--', label='Теоретическая
дисперсия')
plt.xlabel('Число реализаций')
plt.ylabel('Дисперсия')
plt.title('Зависимость выборочной дисперсии от числа реализаций')
plt.legend()
plt.grid(True)
plt.show()

# Задание 3: Эмпирическая и теоретическая плотности
plt.figure(figsize=(10, 6))
n_bins = max(all_samples) - min(all_samples) + 1
counts, bin_edges, _ = plt.hist(all_samples, bins=n_bins, density=True,
                                alpha=0.7, label='Эмпирическая плотность')
```

```
# Теоретическая pmf
x = np.arange(0, n + 1)
pmf_values = [binomial_pmf(k, n, p) for k in x]
plt.plot(x, pmf_values, 'r-', linewidth=2, label='Теоретическая вероятность')

# Отметим среднее и ±σ
mu = m_theory
sigma = math.sqrt(D_theory)
plt.axvline(mu, color='k', linestyle='--', linewidth=2, label='Среднее (μ)')
plt.axvline(mu - sigma, color='g', linestyle=':', linewidth=2, label='μ - σ')
plt.axvline(mu + sigma, color='g', linestyle=':', linewidth=2, label='μ + σ')

plt.xlabel('Значение СВ')
plt.ylabel('Плотность вероятности')
plt.title('Эмпирическая и теоретическая плотности биномиального распределения')
plt.legend()
plt.grid(True)
plt.show()

# Функция для расчета средней абсолютной разности
def calculate_mean_abs_diff(samples, n, p, bins=None):
    hist, bin_edges = np.histogram(samples, bins=bins, density=True)
    bin_centers = np.round((bin_edges[:-1] + bin_edges[1:]) / 2).astype(int)
    theoretical_pdf = np.array([binomial_pmf(k, n, p) for k in bin_centers])
    return np.mean(np.abs(hist - theoretical_pdf))

# График точности оценки
mad_values = []
for k in range(100, n_max + 1, 100):
    mad = calculate_mean_abs_diff(all_samples[:k], n, p, bins=n_bins)
    mad_values.append(mad)

plt.figure(figsize=(10, 6))
plt.plot(range(100, n_max + 1, 100), mad_values)
plt.xlabel('Число реализаций')
plt.ylabel('Средняя абсолютная разность')
plt.title('Зависимость точности оценки плотности от числа реализаций')
plt.grid(True)
plt.show()
```

Формула вероятности для дискретной случайной величины:

$$P(X = k) = \binom{n}{k} \cdot p^k \cdot (1-p)^{n-k}$$

, где:

P(X=k) — вероятность того, что случайная величина X примет значение k

n — общее число испытаний;

k — количество успехов, для которого мы хотим посчитать вероятность;

p — вероятность успеха в одном испытании;

1−p — вероятность неуспеха;

$\binom{n}{k}$ — число способов выбрать k успехов из n испытаний (комбинации).

Результаты выполнения задания

1. График зависимости значения выборочного математического ожидания от числа реализаций СВ.

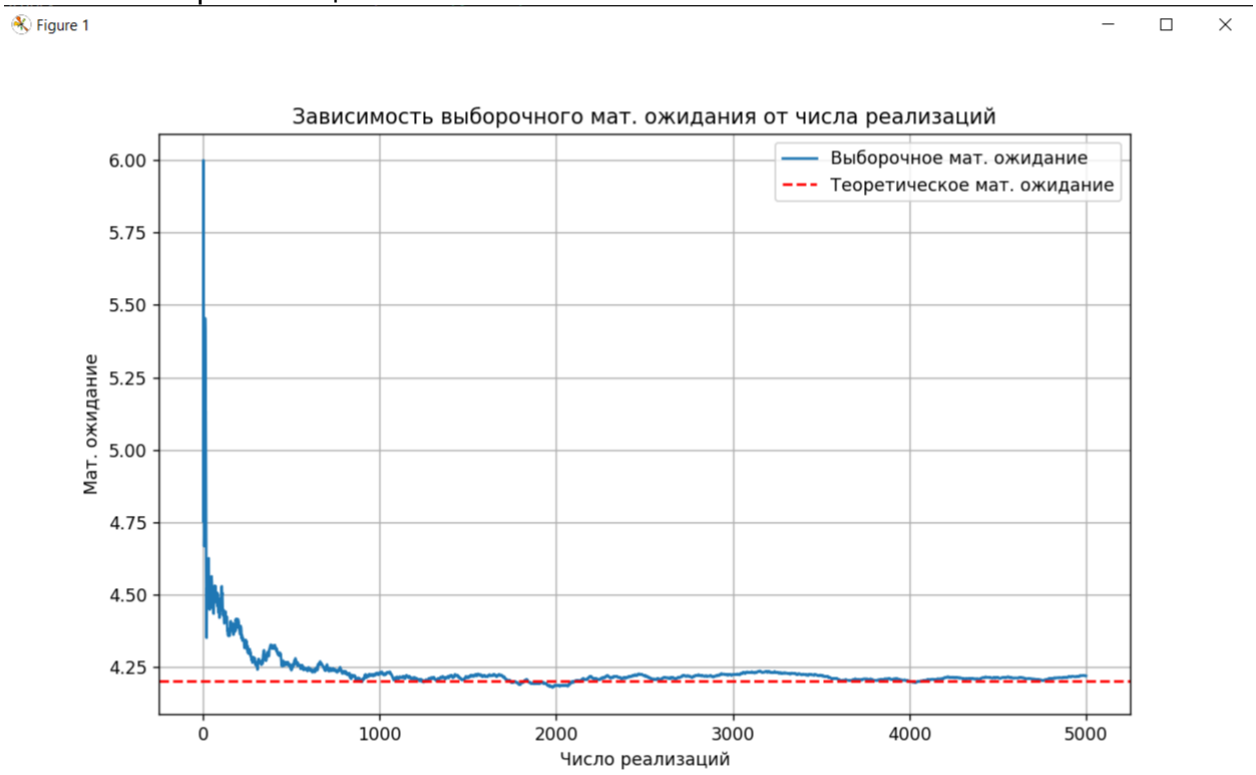


Рисунок 1.

2. График зависимости значения выборочной дисперсии от числа реализаций СВ.

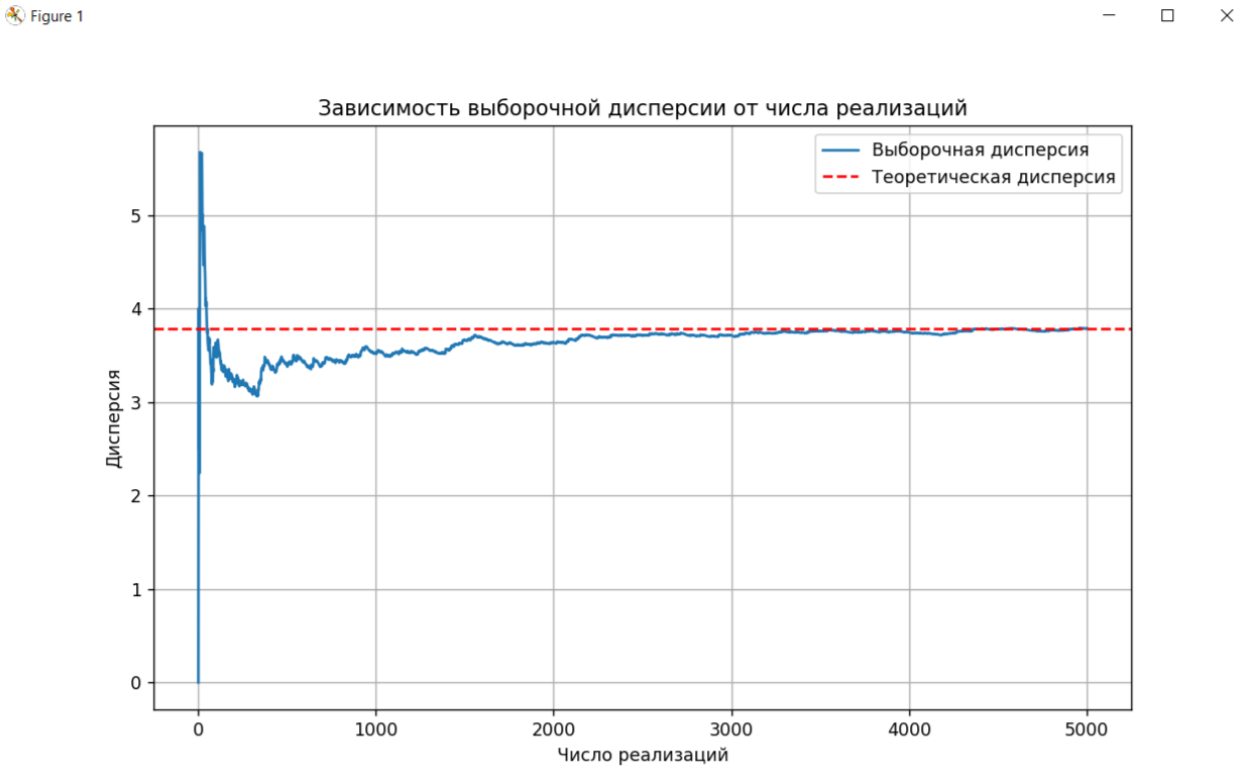


Рисунок 2.

3. Визуализация эмпирической плотности распределения СВ (гистограммы с необходимой нормировкой). График зависимости значения средней абсолютной разности эмпирических и теоретических значений плотности распределения от числа реализаций СВ.

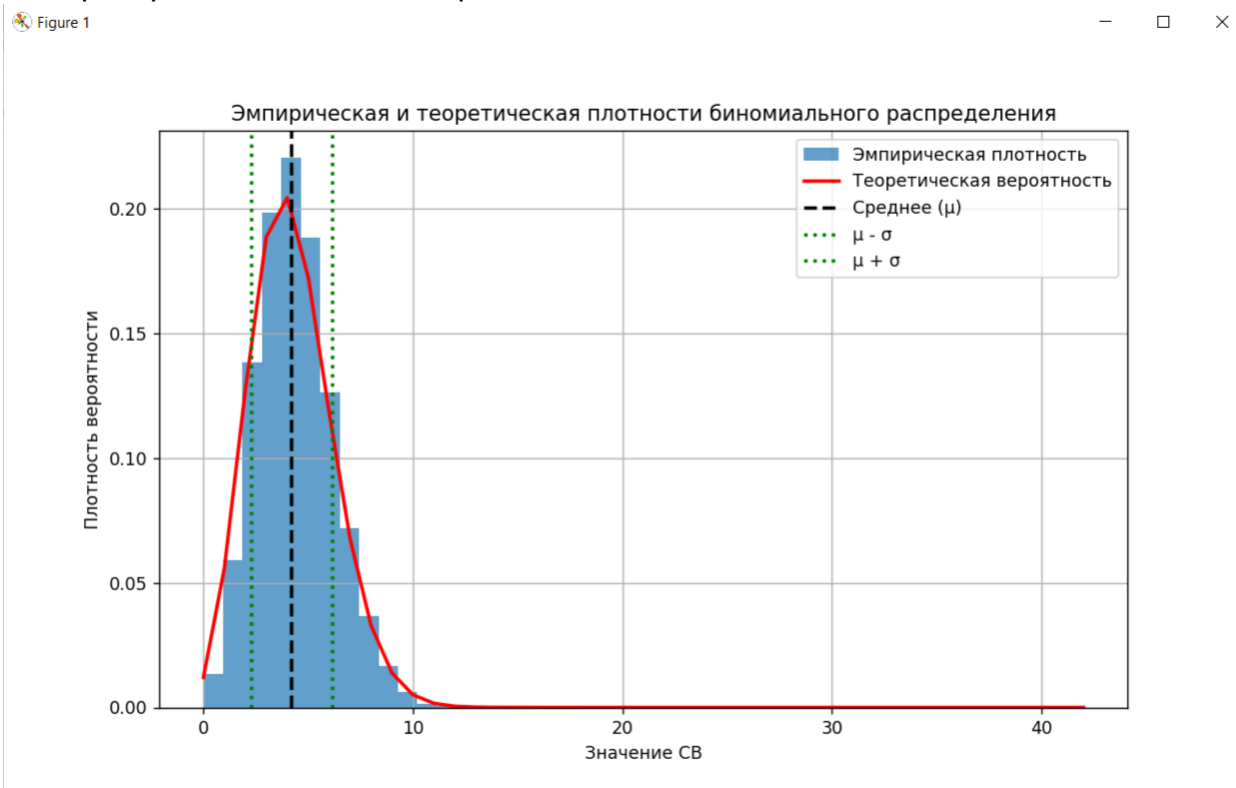


Рисунок 3.

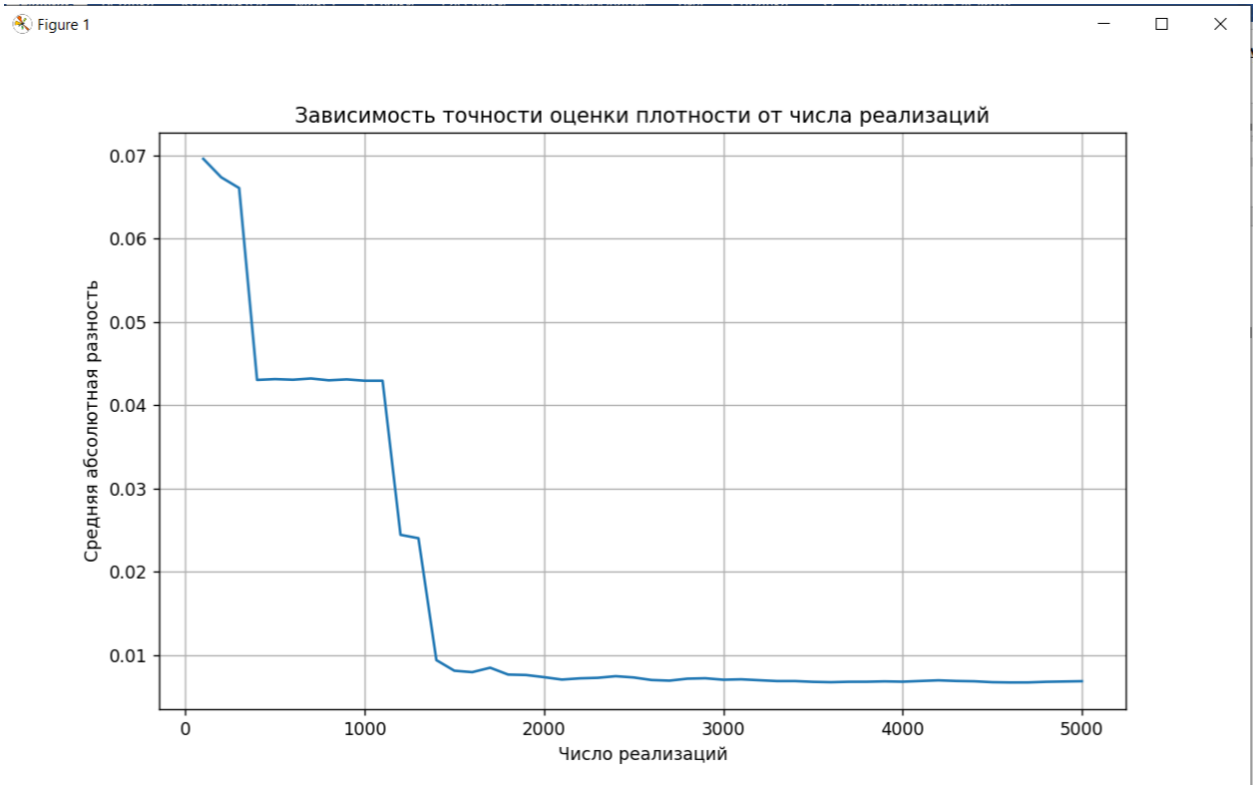
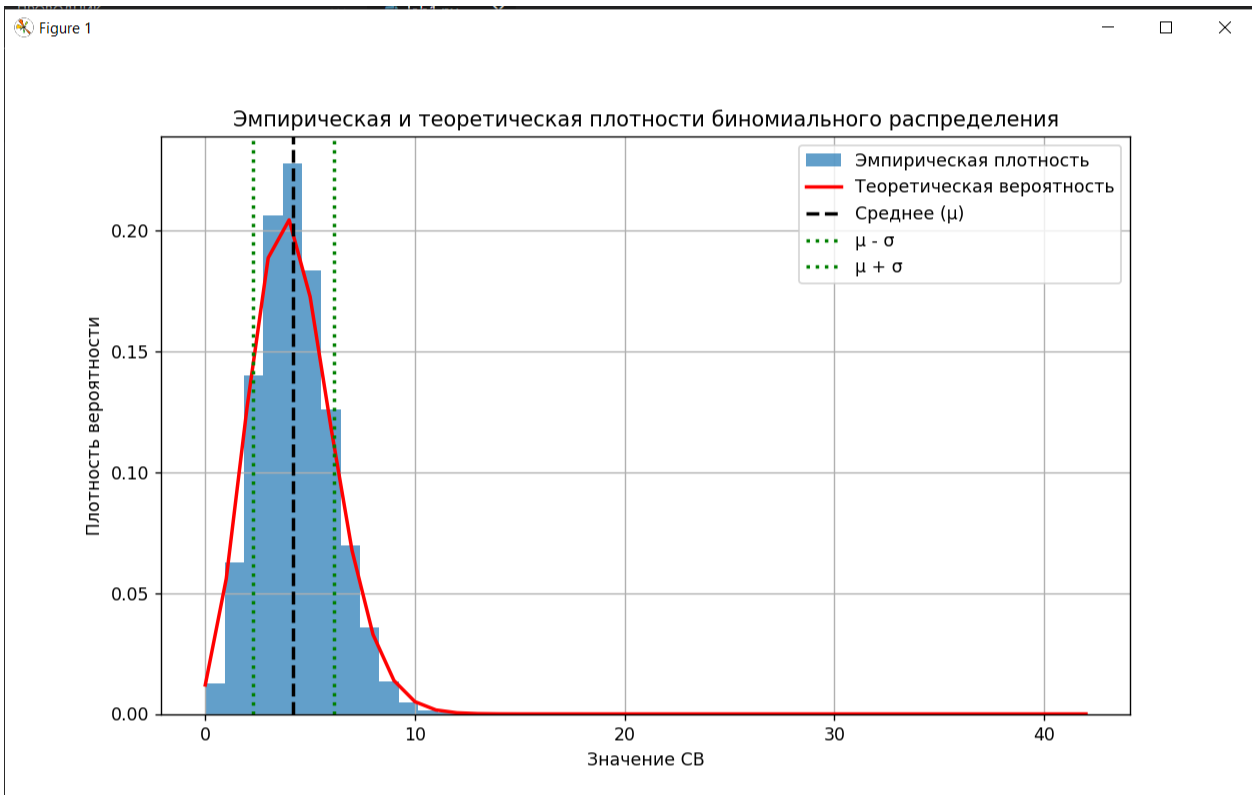


Рисунок 4.

Ответы на контрольные вопросы

1. При увеличении n распределение будет шире, если увеличить p, распределение станет более симметричным, при малых p распределение скошено влево
2. Графики выборочного мат. ожидания и дисперсии будут прижиматься к теоретическим значениям, гистограмма будет лучше совпадать с теоретической вероятностью, ошибка аппроксимации уменьшается
3. Среднее значение отмечается красной линией в центре распределения, +-СКО отмечены зелеными линиями.



4. Гистограмма – графическое представление распределения данных, показывающее частоту попадания значений в определенные интервалы.
5. np.random.uniform(a, b, n), где a – нижняя граница диапазона, b – верхняя, а n – размер массива
6. Submatrix = matrix[row_start:row_end, col_start:col_end]
7. Можно использовать:

- Среднеквадратичную ошибку (среднее значение квадратов разностей между прогнозируемыми и фактическими значениями):
  MSE=$\frac{1}{n}\sum(x_i - y_i)^2$
- Среднюю абсолютную ошибку (среднюю абсолютную ошибку между фактическими и прогнозируемыми значениями):
  MAE=$\frac{1}{n}\sum|x_i - y_i|$

В работе мы использовали MAE для оценки различий между эмпирической и теоретической плотностью.

8. np.dot(A,B) или A @ B
9. A * B
10. for value in vector:
        print(value)
11. plt.plot(x_values, y_values)
    plt.show()
12. plt.hist(data, bins=50)
    plt.show()

Вывод

В ходе работы успешно смоделировано биномиальное распределение с параметрами n=42, p=0.1. Подтверждена сходимость выборочных характеристик к теоретическим значениям (μ=4.2, σ^2=3.78). Визуализирована плотность распределения и показано, что при увеличении числа реализаций ошибка аппроксимации уменьшается. Результаты соответствуют теоретическим ожиданиям.