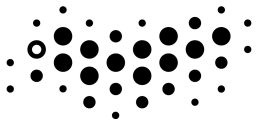
Федеральное государственное автономное образовательное учреждение высшего образования
"Национальный Исследовательский Университет ИТМО"
Мегафакультет Компьютерных Технологий и Управления
Факультет Программной Инженерии и Компьютерной Техники



ITMO UNIVERSITY

Вариант №15

Практическая работа №5

по дисциплине

Теория вероятностей

Выполнил Студент группы P32101
Лапин Алексей Александрович
Преподаватель:
Селина Елена Георгиевна

г. Санкт-Петербург
2023г.

Содержание

1 Текст задания: 3

2 Листинг программы 3

3 Результаты работы 5

4 Выводы 9

1 Текст задания:

Каждый студент получает выборку из 20 чисел. Необходимо определить следующие статистические характеристики: вариационный ряд, экстремальные значения и размах, оценки математического ожидания и среднеквадратического отклонения, эмпирическую функцию распределения и её график, гистограмму и полигон приведенных частот группированной выборки. Для расчета характеристик и построения графиков нужно написать программу на одном из языков программирования. Листинг программы и результаты работы должны быть представлены в отчете по практической работе.

15	1.07	-1.18	1.69	0.48	0.92	1.42	-0.08	0.65	0.66	0.46
	-1.02	1.34	0.31	0.11	0.04	-1.59	-0.21	0.55	1.22	0.82

2 Листинг программы

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


def variation_series(arr):
    return sorted(set(arr))


def frequency(arr):
    freq = {i: arr.count(i) for i in arr}
    return freq


def get_statistical_series(arr):
    freq = frequency(sorted(arr))
    freq_arr = [[key, value] for key, value in freq.items()]
    return freq_arr


def mean(arr):
    ss = get_statistical_series(arr)
    return sum(x * n for x, n in ss) / sum(n for _, n in ss)


def get_standard_deviation(arr):
    freq_arr = get_statistical_series(arr)
    m = mean(arr)
    variance = sum([((x - m) ** 2) * n for x, n in freq_arr]) / len(arr)
    std_dev = (variance)**(0.5)
    return std_dev
```

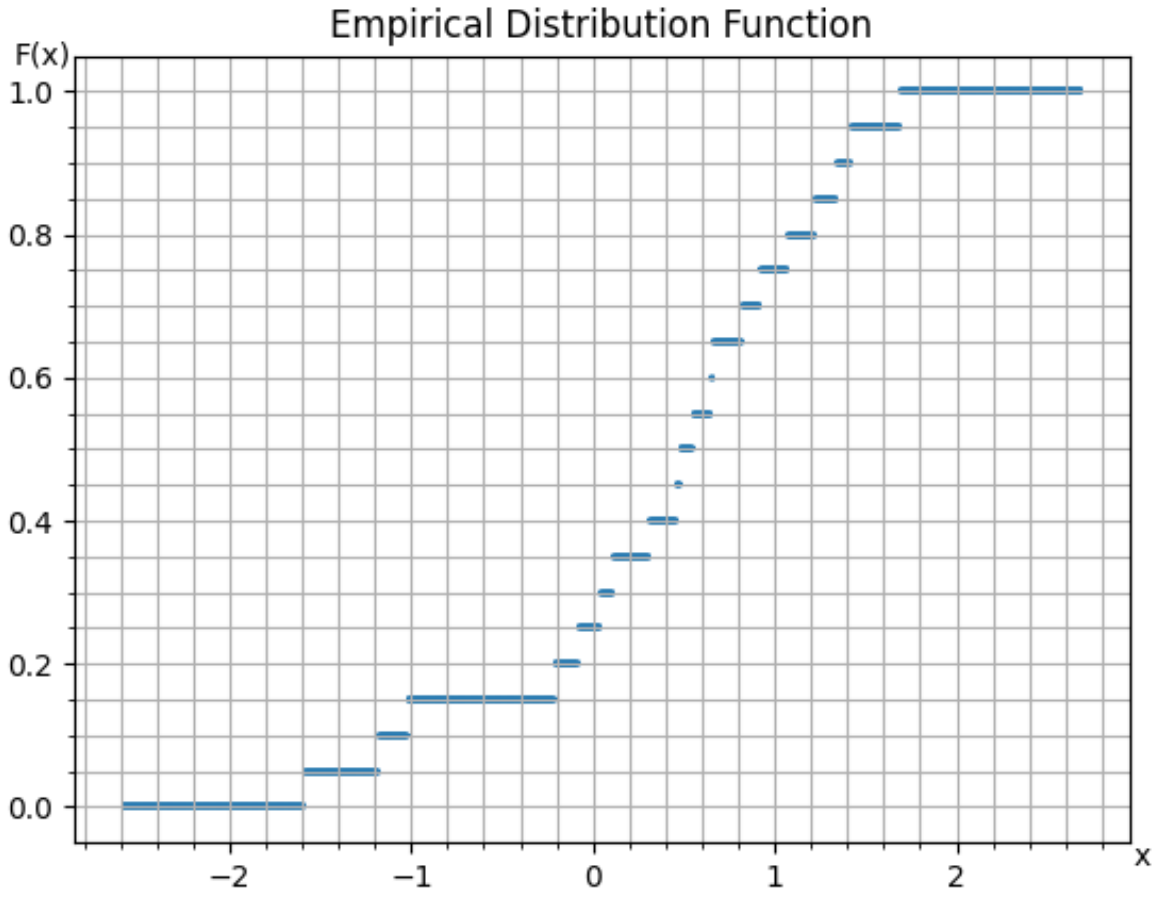
```
def empirical_distribution_function(arr, x):
    freq_arr = get_statistical_series(arr)
    freq = 0
    for i in range(len(freq_arr)):
        if freq_arr[i][0] >= x:
            break
        freq += freq_arr[i][1]
    return freq / len(arr)


def print_empirical_distribution(arr):
    freq_arr = get_statistical_series(arr)
    freq = 0
    print(f"x < {freq_arr[0][0]}:\tF(x) = {freq / len(arr)}")
    for i in range(len(freq_arr) - 1):
        freq += freq_arr[i][1]
        print(
            f"{freq_arr[i][0]} <= x < {freq_arr[i+1][0]}:\tF(x) = {freq / l
    freq += freq_arr[-1][1]
    print(f"{freq_arr[-1][0]} <= x:\tF(x) = {freq / len(arr)}")


def plot_empirical_distribution_function(arr):
    x_values = np.linspace(min(arr) - 1, max(arr) + 1, 10000)
    ecdf_values = [empirical_distribution_function(arr, x)
                   for x in x_values]
    plt.grid(which='both')
    plt.minorticks_on()
    plt.scatter(x_values, ecdf_values, marker='_', s=1)
    plt.xlabel('x', ha='right', x=1.0)
    plt.ylabel('F(x)', va='top', y=1.0)
    plt.gca().yaxis.set_label_coords(-0.03, 1.02)
    plt.gca().yaxis.get_label().set_rotation(0)
    plt.gca().xaxis.set_label_coords(1.02, 0)
    plt.title('Empirical Distribution Function')

    plt.show()


def plot_histogram(arr):
    stat_arr = get_statistical_series(arr)
    x_values = [x for x, _ in stat_arr]
    bin_widths = [x_values[i+1] - x_values[i]
                  for i in range(len(x_values) - 1)]
    bin_heights = [n/h for n, h in zip([n for _, n in stat_arr], bin_widths
    x_values.pop()
    plt.bar(x_values, height=bin_heights,
```

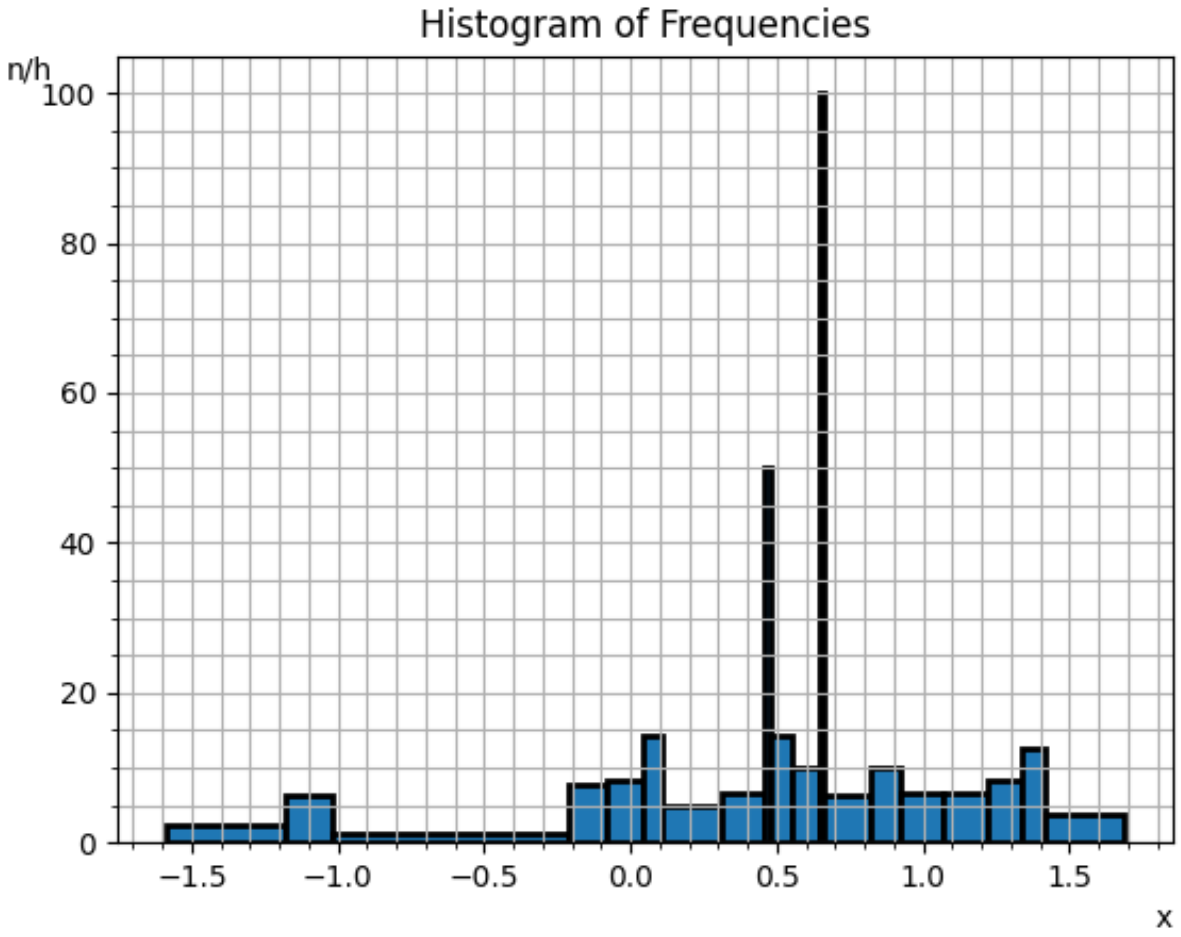
```
            width=bin_widths, edgecolor='black', align='edge', linewidth=2)
    plt.grid(which='both')
    plt.minorticks_on()
    plt.title('Histogram of Frequencies')
    plt.xlabel('x', ha='right', x=1.0)
    plt.ylabel('n/h', va='top', y=1.0)
    plt.gca().yaxis.get_label().set_rotation(0)
    plt.show()


def plot_frequency_polygon(arr):
    stat_array = get_statistical_series(arr)
    x, n = zip(*stat_array)
    plt.grid(which='both')
    plt.minorticks_on()
    plt.plot(x, n, marker='o', linestyle='-')
    plt.title('Frequency Polygon')
    plt.xlabel('x', ha='right', x=1.0)
    plt.ylabel('n', va='top', y=1.0)
    plt.gca().yaxis.get_label().set_rotation(0)
    plt.show()


data = [1.07, -1.18, 1.69, 0.48, 0.92, 1.42, -0.08, 0.65, 0.66, 0.46,
        -1.02, 1.34, 0.31, 0.11, 0.04, -1.59, -0.21, 0.55, 1.22, 0.82]
if __name__ == '__main__':
    vs = variation_series(data)
    print("========[Variation series]=========================")
    print(vs)
    max_val, min_val = vs[-1], vs[0]
    print("========[Characteristics of the variation series]========")
    print('maximum value: ', round(max_val, 5))
    print('minimum value: ', round(min_val, 5))
    print('range: ', round(max_val - min_val, 5))
    print(f'expected value: {round(mean(data), 5)}')
    print(f'standard deviation: {round(get_standard_deviation(data), 5)}')
    print('========[Empirical distribution function]========')
    print_empirical_distribution(data)
    plot_empirical_distribution_function(data)
    plot_histogram(data)
    plot_frequency_polygon(data)
```

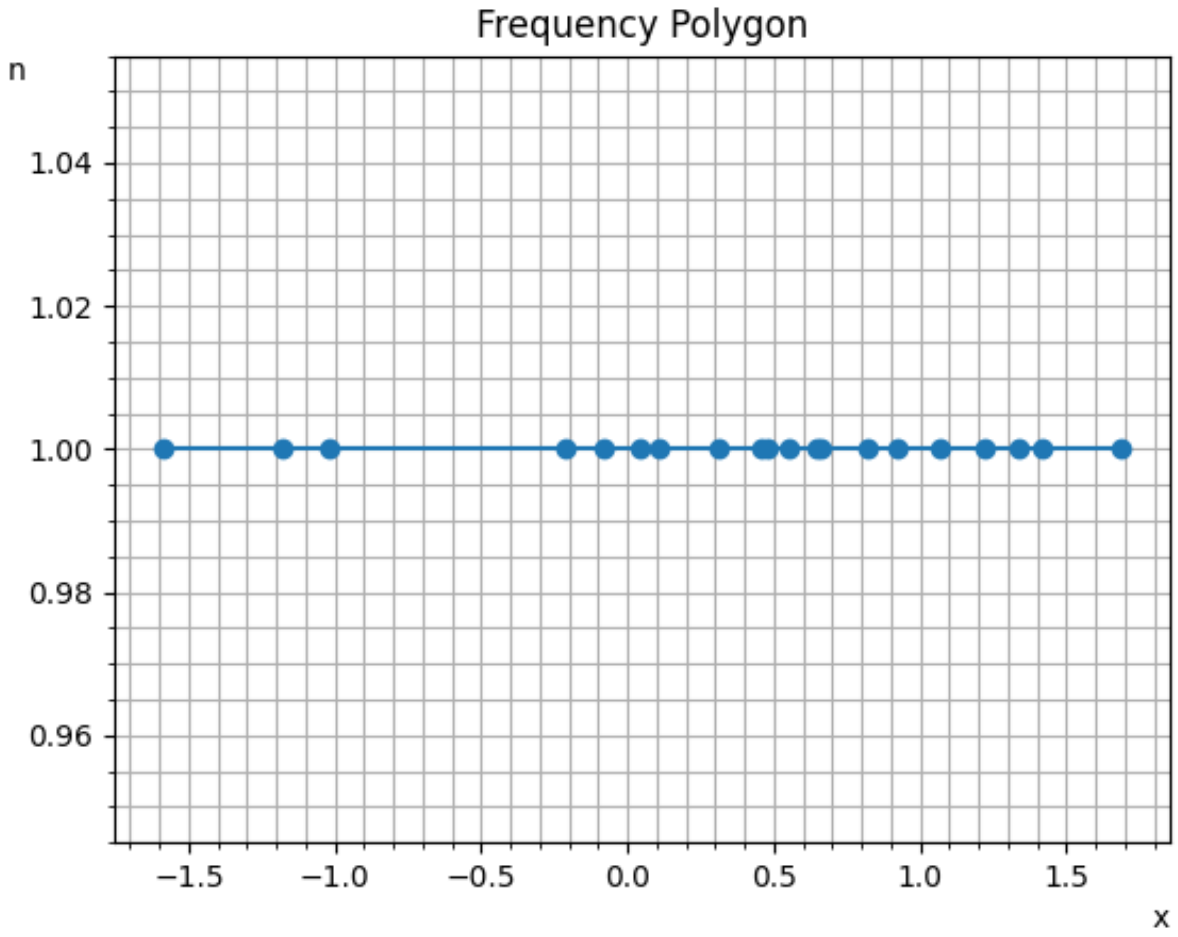
3 Результаты работы

```
========[Variation series]=======================
[-1.59, -1.18, -1.02, -0.21, -0.08, 0.04, 0.11, 0.31, 0.46, 0.48,
 0.55, 0.65, 0.66, 0.82, 0.92, 1.07, 1.22, 1.34, 1.42, 1.69]
========[Characteristics of the variation series]========
maximum value:  1.69
```

```
minimum value:  -1.59
range:  3.28
expected value: 0.383
standard deviation: 0.8547
========[Empirical distribution function]========
x < -1.59:      F(x) = 0.0
-1.59 <= x < -1.18:     F(x) = 0.05
-1.18 <= x < -1.02:     F(x) = 0.1
-1.02 <= x < -0.21:     F(x) = 0.15
-0.21 <= x < -0.08:     F(x) = 0.2
-0.08 <= x < 0.04:      F(x) = 0.25
0.04 <= x < 0.11:       F(x) = 0.3
0.11 <= x < 0.31:       F(x) = 0.35
0.31 <= x < 0.46:       F(x) = 0.4
0.46 <= x < 0.48:       F(x) = 0.45
0.48 <= x < 0.55:       F(x) = 0.5
0.55 <= x < 0.65:       F(x) = 0.55
0.65 <= x < 0.66:       F(x) = 0.6
0.66 <= x < 0.82:       F(x) = 0.65
0.82 <= x < 0.92:       F(x) = 0.7
0.92 <= x < 1.07:       F(x) = 0.75
1.07 <= x < 1.22:       F(x) = 0.8
1.22 <= x < 1.34:       F(x) = 0.85
1.34 <= x < 1.42:       F(x) = 0.9
1.42 <= x < 1.69:       F(x) = 0.95
1.69 <= x:      F(x) = 1.0
```


Empirical Distribution Function
F(x)
1.0
0.8
0.6
0.4
0.2
0.0
−2
−1
0
1
2
x


Histogram of Frequencies
n/h
100
80
60
40
20
0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
x



4 Выводы

В процессе выполнения данной практической работы я написал программу на языке программирования Python, которая позволяет рассчитывать основные статистические характеристики выборки и строить графики эмпирической функции распределения, гистограммы и полигон.

В результате выполнения данной работы я узнал, как использовать стандартные библиотеки Python для работы с математическими функциями и графиками, а также научился применять методы статистики для решения задач.