**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

**Дисциплина:** Вычислительная математика

**Лабораторная работа №1**

**«Решение системы линейных алгебраических уравнений СЛАУ»**

**Вариант №9**

Выполнил:

Мокров Семён Андреевич

Группа:

P3215

Проверила:

Малышева Татьяна Алексеевна

Санкт-Петербург

2022

# Цель работы

Определить следующие статистические характеристики: вариационный̆ ряд, экстремальные значения и размах, оценки математического ожидания и среднеквадратического отклонения, эмпирическую функцию распределения и построить её график, а также построить гистограмму и полигон приведенных частот группированной̆ выборки.

# Листинг программы

Весь исходный код программы располагается на GitHub (адрес: https://github.com/semwett0301/theory\_of\_possibility\_5)

import numpy as np  
import pandas as pd  
import seaborn as sns  
import matplotlib.pyplot as plt  
from termcolor import colored  
  
selection = np.array([-0.34, -0.73, 0.69, 0.55, 1.11, 0.62, 0.93, 0.42, 1, -0.48, -1.1, -0.66, -0.93, 0.42, -0.22, -0.11, -1.06, -0.72, -0.1, 1.05])  
sort\_selection = np.sort(selection)  
  
min = sort\_selection[0]  
max = sort\_selection[sort\_selection.size - 1]  
  
print('Вариационный ряд:\n' + colored(str(sort\_selection), 'green'))  
print('\nНаименьшее значение (первая порядковая статистика)\n' + colored(str(min), 'green'))  
print('Наибольшее значение (n-ая порядковая статистика)\n' + colored(str(max), 'green'))  
print('Размах\n' + colored(str(max - min), 'green'))  
  
mathematical\_expectation = np.sum(sort\_selection) / 20  
average\_deviation = (np.sum(sort\_selection \*\* 2) / 20 - mathematical\_expectation \*\* 2) \*\* 0.5  
  
print('\nМатематическое ожидание:')  
print(colored('%.3f' % mathematical\_expectation, 'green'))  
print('Cреднеквадратичное отклонение:')  
print(colored('%.3f' % average\_deviation, 'green'))  
  
f\_empire = []  
for i in range(sort\_selection.size):  
 f\_empire.append(i / sort\_selection.size)  
segments = ['(' + str(sort\_selection[i]) + ';' + str(sort\_selection[i + 1]) + ']' for i in range(sort\_selection.size - 1)]  
segments.insert(0, str('(-oo;' + str(sort\_selection[0]) + '0]'))  
frame = pd.DataFrame({'f(x)': f\_empire, 'segment': segments})  
  
print('\nЗначения эмпирической функции:')  
print(colored(frame.set\_index('f(x)'), 'green'))  
  
plt.step(sort\_selection, f\_empire)  
plt.show()  
  
data\_for\_gist = int(1 + np.log2(sort\_selection.size))  
sns.histplot(sort\_selection, kde=True, stat='density', bins=data\_for\_gist)  
plt.show()  
  
data\_for\_polygon = (max - min) / data\_for\_gist  
y, edges = np.histogram(sort\_selection, np.arange(min, max + 0.1, data\_for\_polygon))  
centers = 0.5 \* (edges[1:] + edges[:-1])  
plt.plot(centers, y)  
plt.xticks(edges)  
plt.yticks(y)  
plt.show()

# Результат работы программы

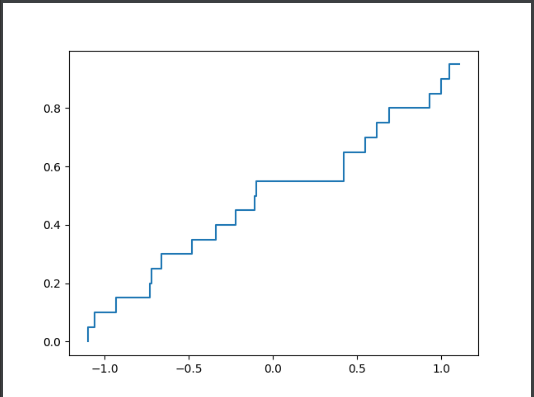
Изображение выглядит как текст

Автоматически созданное описание

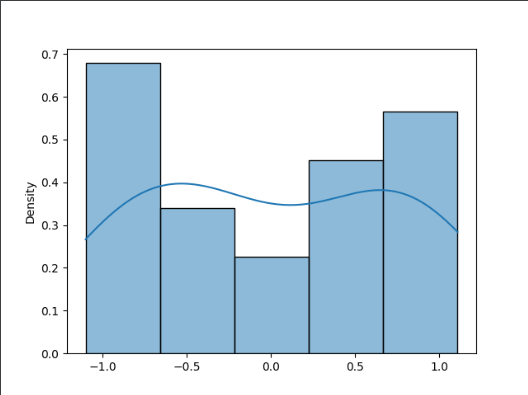
Изображение выглядит как текст

Автоматически созданное описание

**График эмпирической функции:**



**Гистограмма приведенных частот группированной выборки:**



**Полигон приведенных частот группированной выборки:**

