Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 4

Тема Построение и программная реализация алгоритма наилучшего среднеквадратичного приближения

Студент Климов И.С.

Группа ИУ7-42Б

Оценка (баллы) _____

Преподаватель Градов В.М.

Москва.
2021 г

Цель работы: Получение навыков владения метода наименьших квадратов с использованием полинома заданной степени при аппроксимации табличных функций с весами.

1. Исходные данные

1. Таблица функции с **весами** p_i с количеством узлов N. Сформировать таблицу самостоятельно со случайным разбросом точек.
2. Степень аппроксимирующего полинома.

2. Код программы

Листинг 1. table.py

```
def print_table(table):
    def separator():
        print('|', '-' * 9, '|', '-' * 9, '|', '-' * 9, '|', sep='')

    print('Исходная таблица:')
    separator()
    print('|    x    |    y    |   Вес   |')
    separator()
    for line in table:
        print(f'|{str(line[0]):^9s}|{str(line[1]):^9s}|{str(line[2]):^9s}|')
        separator()
    print()


def create_table(filename):
    with open(f'data/{filename}') as f:
        table = [tuple(map(float, line.split())) for line in f if line]

    return table
```

Листинг 2. find.py

```
def get_system(table, degree):
    degree += 1
    n = len(table)
    zeros = [0 for _ in range(degree)]
    system = [zeros[:] for _ in range(degree)]
    free_column = zeros[:]

    for i in range(degree):
        for j in range(n):
            tmp = table[j][2] * (table[j][0] ** i)
            for z in range(degree):
                system[i][z] += tmp * (table[j][0] ** z)
            free_column[i] += tmp * table[j][1]

    return system, free_column


def get_inverse_matrix(matrix):
    n = len(matrix)
```

```
    zeros = [0 for _ in range(n)]
    inverse_matrix = [zeros[:] for _ in range(n)]
    for i in range(n):
        column = convert_column(matrix, i)
        for j in range(n):
            inverse_matrix[j][i] = column[j]

    return inverse_matrix


def convert_column(matrix, index):
    n = len(matrix)
    extended_matrix = [matrix[i][:] for i in range(n)]
    new_column = [0 for _ in range(n)]

    for i in range(n):
        extended_matrix[i] += [1.0] if i == index else [0.0]

    for i in range(n):
        if not extended_matrix[i][i]:
            for j in range(i + 1, n):
                if extended_matrix[j][j]:
                    extended_matrix[i], extended_matrix[j] = extended_matrix[j],
extended_matrix[i]

        for j in range(i + 1, n):
            d = -extended_matrix[j][i] / extended_matrix[i][i]
            for z in range(n + 1):
                extended_matrix[j][z] += d * extended_matrix[i][z]

    for i in range(n - 1, -1, -1):
        result = 0
        for j in range(n):
            result += extended_matrix[i][j] * new_column[j]

        new_column[i] = (extended_matrix[i][n] - result) / extended_matrix[i][i]

    return new_column


def multiply(inverse_matrix, free_column):
    n = len(free_column)
    result = [0 for _ in range(n)]
    for i in range(n):
        for j in range(n):
            result[i] += free_column[j] * inverse_matrix[i][j]

    return result


def find_coefficients(table, degree):
    system, free_column = get_system(table, degree)
    inverse_matrix = get_inverse_matrix(system)
    coefficients = multiply(inverse_matrix, free_column)

    return coefficients
```

Листинг 3. show.py

```
import numpy as np
import matplotlib
```

```python
import matplotlib.pyplot as plt


def show_result(table, result, degree):
    matplotlib.use('TkAgg')
    if len(table) > 1:
        dx = table[1][0] - table[0][0]
    else:
        dx = 10

    x = np.linspace(table[0][0] - dx, table[-1][0] + dx, 100)
    y = []
    for xi in x:
        tmp = 0
        for i in range(degree + 1):
            tmp += xi ** i * result[i]
        y.append(tmp)
    plt.plot(x, y)

    y_min, y_max = min(y), max(y)

    x = [point[0] for point in table]
    y = [point[1] for point in table]
    plt.plot(x, y, 'o', color='red', label='Исходные точки')
    plt.legend(loc='best')

    y_min, y_max = min(y_min, min(y)), max(y_max, max(y))
    dy = (y_max - y_min) * 0.03

    plt.axis([table[0][0] - dx, table[-1][0] + dx, y_min - dy, y_max + dy])
    plt.show()
```

Листинг 4. main.py

```python
from find import find_coefficients
from show import show_result
from table import create_table, print_table


def main():
    filename = '1.txt'
    print(f'Результат на основе данных файла data/{filename}\n')
    table = create_table(filename)
    print_table(table)

    try:
        degree = int(input('Введите степень полинома: '))
    except ValueError:
        return print('Ошибка! Вы должны были ввести число')

    result = find_coefficients(table, degree)
    print('\nРезультат можно увидеть на появившемся графике')
    show_result(table, result, degree)


if __name__ == '__main__':
    main()
```

3. Результаты работы

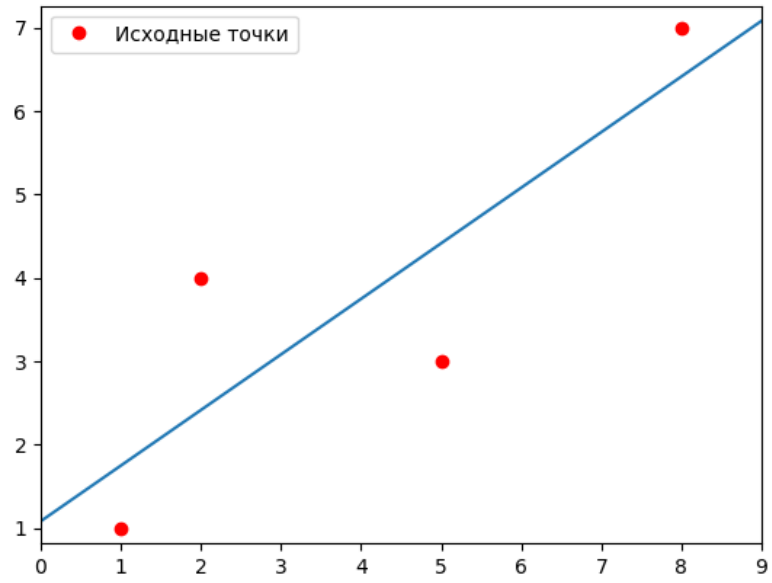
1. Веса всех точек одинаковые

```
Результат на основе данных файла data/1.txt

Исходная таблица:
|---------|---------|---------|
|    x    |    y    |   Вес   |
|---------|---------|---------|
|   1.0   |   1.0   |   1.0   |
|---------|---------|---------|
|   2.0   |   4.0   |   1.0   |
|---------|---------|---------|
|   5.0   |   3.0   |   1.0   |
|---------|---------|---------|
|   8.0   |   7.0   |   1.0   |
|---------|---------|---------|

Введите степень полинома: 1

Результат можно увидеть на появившемся графике
```

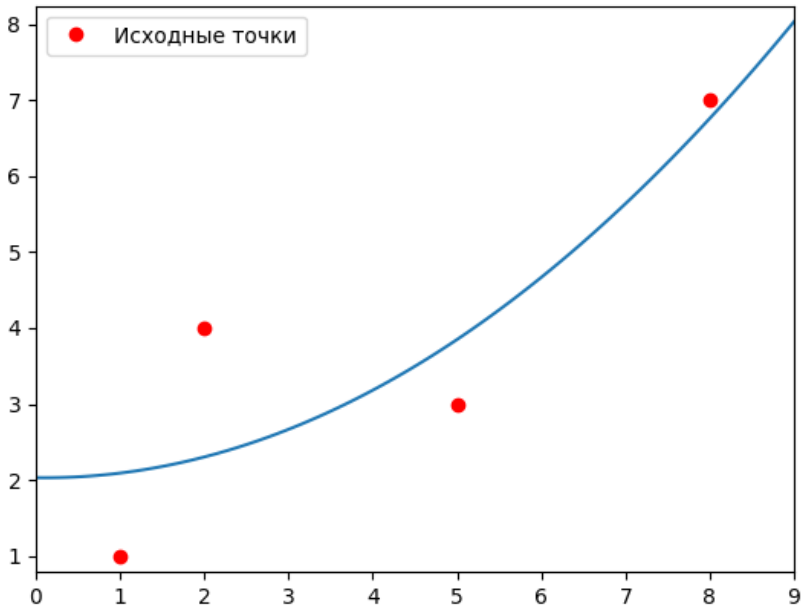



```
Результат на основе данных файла data/1.txt

Исходная таблица:
|---------|---------|---------|
|    x    |    y    |   Вес   |
|---------|---------|---------|
|   1.0   |   1.0   |   1.0   |
|---------|---------|---------|
|   2.0   |   4.0   |   1.0   |
|---------|---------|---------|
|   5.0   |   3.0   |   1.0   |
|---------|---------|---------|
|   8.0   |   7.0   |   1.0   |
|---------|---------|---------|

Введите степень полинома: 2

Результат можно увидеть на появившемся графике
```

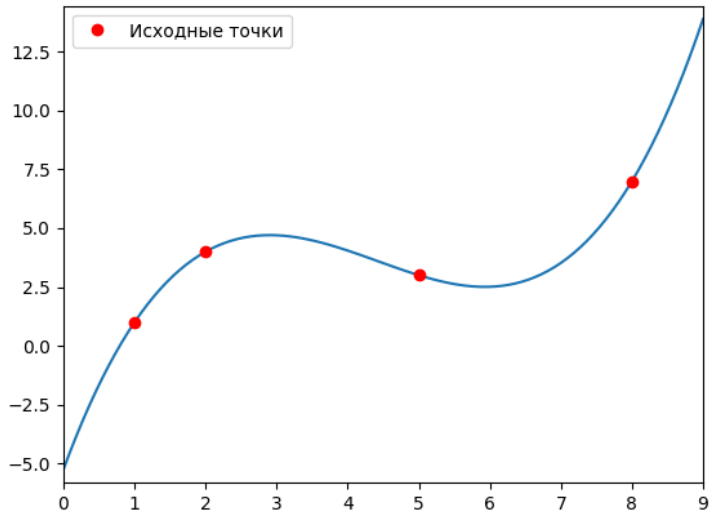



```
Результат на основе данных файла data/1.txt

Исходная таблица:
|---------|---------|---------|
|    x    |    y    |   Вес   |
|---------|---------|---------|
|   1.0   |   1.0   |   1.0   |
|---------|---------|---------|
|   2.0   |   4.0   |   1.0   |
|---------|---------|---------|
|   5.0   |   3.0   |   1.0   |
|---------|---------|---------|
|   8.0   |   7.0   |   1.0   |
|---------|---------|---------|

Введите степень полинома: 3

Результат можно увидеть на появившемся графике
```

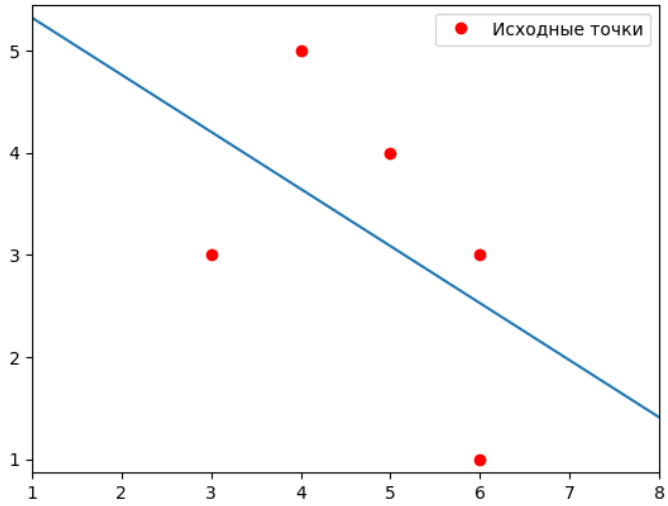


2. Веса одних и тех же точек одинаковые и разные (сравнение)

```
Результат на основе данных файла data/2.txt

Исходная таблица:
|---------|---------|---------|
|    x    |    y    |   Вес   |
|---------|---------|---------|
|   3.0   |   3.0   |   1.0   |
|---------|---------|---------|
|   5.0   |   4.0   |   1.0   |
|---------|---------|---------|
|   4.0   |   5.0   |   1.0   |
|---------|---------|---------|
|   6.0   |   3.0   |   1.0   |
|---------|---------|---------|
|   6.0   |   1.0   |   1.0   |
|---------|---------|---------|

Введите степень полинома: 1
```

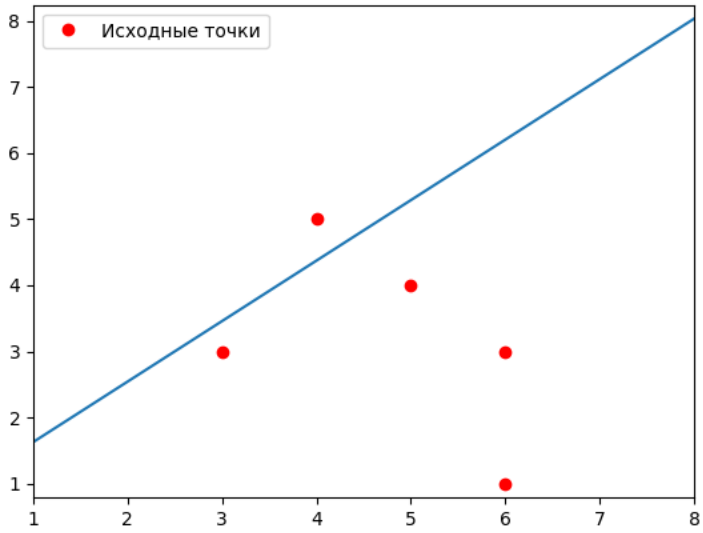



```
Результат на основе данных файла data/3.txt

Исходная таблица:
|---------|---------|---------|
|    x    |    y    |   Вес   |
|---------|---------|---------|
|   3.0   |   3.0   |   9.0   |
|---------|---------|---------|
|   5.0   |   4.0   |   0.1   |
|---------|---------|---------|
|   4.0   |   5.0   |  10.0   |
|---------|---------|---------|
|   6.0   |   3.0   |   0.3   |
|---------|---------|---------|
|   6.0   |   1.0   |   0.2   |
|---------|---------|---------|

Введите степень полинома: 1

Результат можно увидеть на появившемся графике
```



Благодаря изменению весов точек был изменен знак углового коэффициента прямой

4. Вопросы при защите лабораторной работы

1) Что произойдет при задании степени полинома n = N – 1 (числу узлов таблицы минус 1)?

При n = N – 1 получим полином, прошедший через все узлы таблицы, то есть график будет проходить через все табличные точки.

2) Будет ли работать Ваша программа при n ≥ N? Что именно в алгоритме требует отдельного анализа данного случая и может привести к аварийной остановке?

В моем случае программа работать будет, но не так, как ожидается. При таких входных данных СЛАУ не будет иметь решения, так как определитель матрицы будет равен 0. Поэтому в программу следует добавить дополнительную проверку.

3) Получить формулу для коэффициента полинома a_0 при степени полинома n = 0. Какой смысл имеет величина, которую представляет данный коэффициент?

$$a_0 = \frac{\sum_{i=1}^{N} p_i y_i}{\sum_{i=1}^{N} p_i}$$

Данная величина будет иметь смысл математического ожидания, где p_i – вероятность «встретить» y_i

4) Записать и вычислить определитель матрицы СЛАУ для нахождения коэффициентов полинома для случая, когда n = N = 2. Принять все p_i = 1

$$\begin{cases} a_0 + (x_0 + x_1)^2 + (x_0^2 + x_1^2)a_2 = y_0 + y_1 \\ (x_0 + x_1)a_0 + (x_0^2 + x_1^2)a_1 + (x_0^3 + x_1^3)a_2 = y_0 x_0 + y_1 x_1 \\ (x_0^2 + x_1^2)a_0 + (x_0^3 + x_1^3)a_1 + (x_0^4 + x_1^4)a_2 = y_0 x_0^2 + y_1 x_0^2 \end{cases}$$

$$(x_0^2 + x_1^2)(x_0^4 + x_1^4) + (x_0 + X_1)(x_0^3 + x_1^3)(x_0^2 + x_1^2) + (x_0^2 + x_1^2)(x_0 + x_1)(x_0^3 + x_1^3)$$
$$- (x_0^2 + x_1^2)(x_0^2 + x_1^2)(x_0^2 + x_1^2) - (x_0^3 + x_1^3)(x_0^3 + x_1^3) - (x_0 + x_1)(x_0 + x_1)(x_0^4 + x_1^4)$$
$$= 0$$

Данные входные данные подходят под случай, описанный в пункте 2, то есть определитель будет равен 0.

5) **Построить СЛАУ при выборочном задании степеней аргумента полинома $\boldsymbol{\varphi(x) = a_0 + a_1 x^m + a_2 x^n}$, причем степени n и m в этой формуле известны.**

Получим следующую СЛАУ (обнулив остальные коэффициенты):

$$\begin{cases} (x^0, x^0)a_0 + (x^0, x^m)a_1 + (x^0, x^n)a_2 = (y, x^0) \\ (x^m, x^0)a_0 + (x^m, x^m)a_1 + (x^m, x^n)a_2 = (y, x^m) \\ (x^n, x^0)a_0 + (x^n, x^m)a_1 + (x^n, x^n)a_2 = (y, x^n) \end{cases}$$

6) **Предложить схему алгоритма решения задачи из вопроса 5, если степени n и m подлежат определению наравне с коэффициентами a_k, т.е. количество неизвестных равно 5.**

Предварительно можно найти степени n и m, а затем использовать предыдущую формулу для составления СЛАУ.