	Министерство образования и науки 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 (ИУ)

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии (ИУ7)

ЛАБОРАТОРНАЯ РАБОТА №4
« Построение и программная реализация алгоритма наилучшего среднеквадратичного приближения»

Студент группы ИУ7-41Б
Костев Дмитрий

Москва 2021

Цель работы

Получение навыков построения алгоритма метода наименьших квадратов с использованием полинома заданной степени при аппроксимации табличных функций с весами.

Исходные данные

1. Таблица функции с весами p_i с количеством узлов N.

2. Степень аппроксимирующего полинома - n.

Код программы

```
# Лабораторная работа №4

from math import *
import matplotlib.pyplot as plt

def phi(x, k):
     return pow(x, k)

def loadTableXY(nameFile):
     tableXY = [[], [], []]
     with open(nameFile, 'r') as f:
          tmpArr = f.readlines()
     for string in tmpArr:
          tmp = []
          tmp = string.split(" ")
          tableXY[0].append(float(tmp[0]))
          tableXY[1].append(float(tmp[1]))
          tableXY[2].append(float(tmp[2].split("\n")[0]))

     return tableXY

def getCoeffSLAU(tableXY, n):
     slau = []
     column = []

     for k in range(n + 1):
          tmp = 0
          slau.append([0 for m in range(n + 1)])
          for i in range(len(tableXY[0])):
               tmp += tableXY[2][i] * tableXY[1][i] *
phi(tableXY[0][i], k)
               for j in range(n + 1):
                    tmpCoeff = tableXY[2][i] * phi(tableXY[0][i],
k + j)
                    slau[k][j] += tmpCoeff
          column.append(tmp)

     return slau, column

def getReverseSLAU(slau):
     sz = len(slau)
     res = []

     for i in range(sz):
          res.append([])
          for j in range(sz):
               if (i == j):
                    res[i].append(1.0)
               else:
```

```
                res[i].append(0.0)

    for i in range(sz):
        for j in range(sz):
            if (i != j):
                tmp = slau[j][i] / slau[i][i]

                for k in range(sz):
                    slau[j][k] -= slau[i][k] * tmp
                    res[j][k] -= res[i][k] * tmp

    for i in range(sz):
        for j in range(sz):
            res[i][j] /= slau[i][i]

    return res

def multiplication(slau, column):
    sz = len(slau)
    res = []

    for i in range(sz):
        res.append(0)
        for j in range(sz):
            res[i] += slau[i][j] * column[j]

    return res

def graphic(tableXY, x, y):
    fig = plt.figure()
    plt.scatter(tableXY[0], tableXY[1], label = "исходная табли-
ца", color = "blue")
    plt.plot(x, y, color = "green")
    plt.grid(True)
    plt.legend()
    plt.show()

if (__name__ == "__main__"):
    tableXY = loadTableXY("data.txt")
    # n = 2

    for i in range(len(tableXY)):
        print(tableXY[0][i], tableXY[1][i], tableXY[2][i])

    n = int(input("Введите степень полинома: "))
    while (n <= 0):
        print("Степень полинома должна быть больше 0\n")
        n = int(input("Введите степень полинома: "))

    slau, column = getCoeffSLAU(tableXY, n)
    print(slau, column)
```

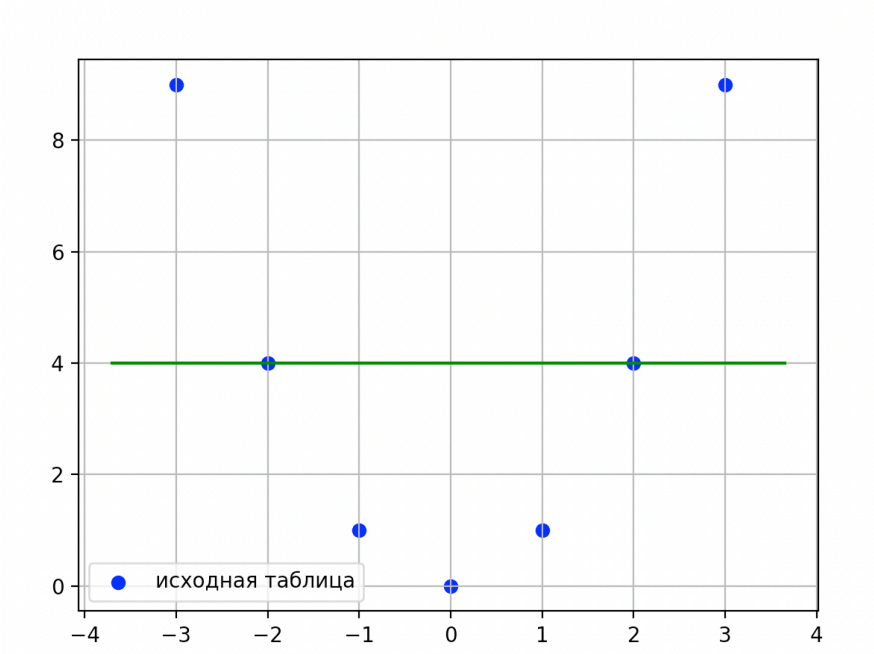
```
slau = getReverseSLAU(slau)
a = multiplication(slau, column)

print(a)

distanceX = tableXY[0][-1] - tableXY[0][0] + 1

stepX = distanceX / 100
x = []
y = []

tmpX = tableXY[0][0] - stepX * 10

while (tmpX < tableXY[0][-1] + stepX * 10):
     x.append(tmpX)
     tmpY = 0
     for k in range(n + 1):
          tmpY += phi(tmpX, k) * a[k]
     y.append(tmpY)
     tmpX += stepX

graphic(tableXY, x, y)
```

Результат работы программы

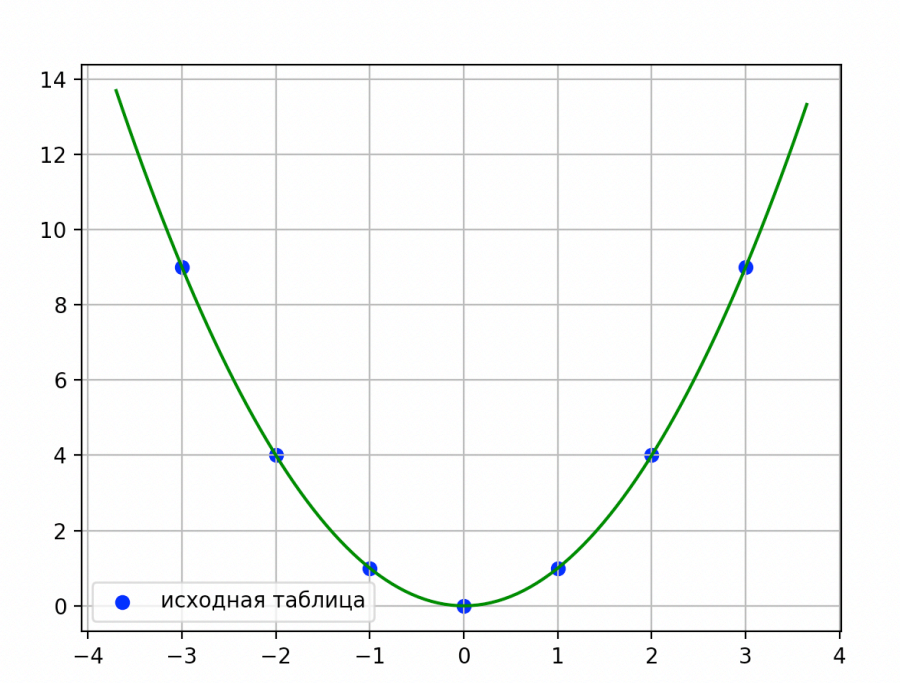
1. Одинаковые веса

X	Y	W
-3	9	1
-2	4	1
-1	1	1
0	0	1
1	1	1
2	4	1
3	9	1

Полином 1 степени:



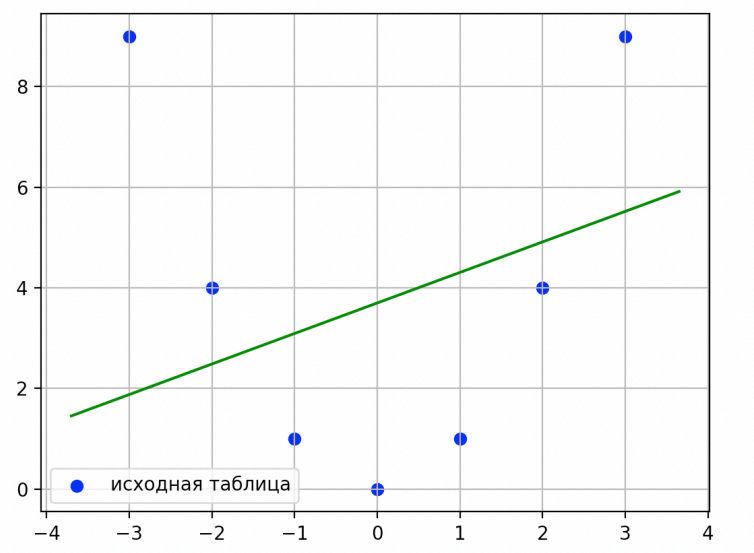
Полином 2 степени:



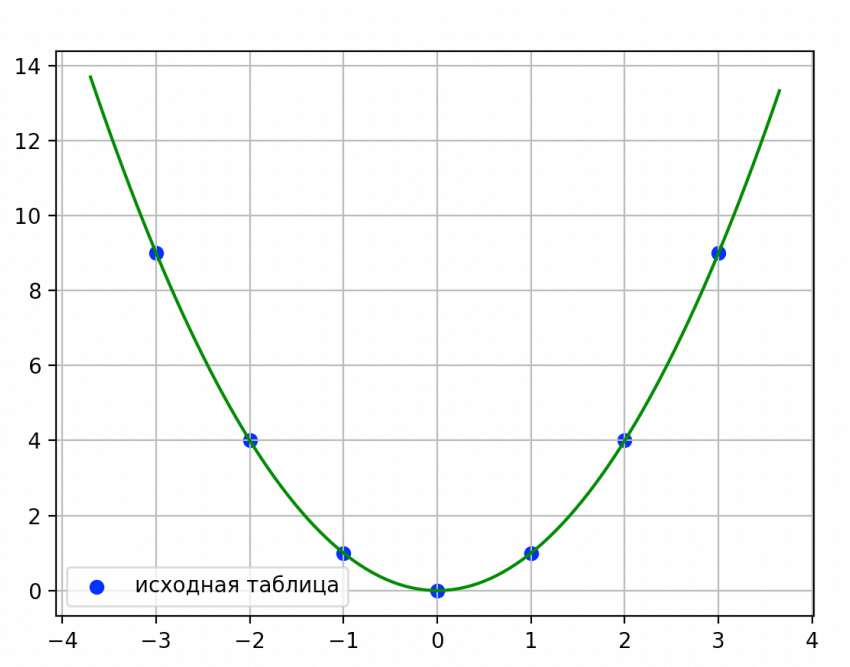
2. Разные веса

X	Y	W
-3	9	2
-2	4	3
-1	1	3
0	0	4
1	1	5
2	4	6
3	9	7

Полином 1 степени:



Полином 2 степени:



Ответы на вопросы

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

N точками можно определить однозначно полином N - 1 степени. Таким образом, мы построим полином, который пройдет через все табличные точки, причем в выражении выражение в скобках будет тождественно равно нулю, что позволяет сделать вывод о том, что в данном случае у нас еще и нет зависимости от весов (то есть при любых весах полином будет иметь минимально возможное значение в случае прохода через заданные в таблице точки – то есть иметь одни и те же коэффициенты)

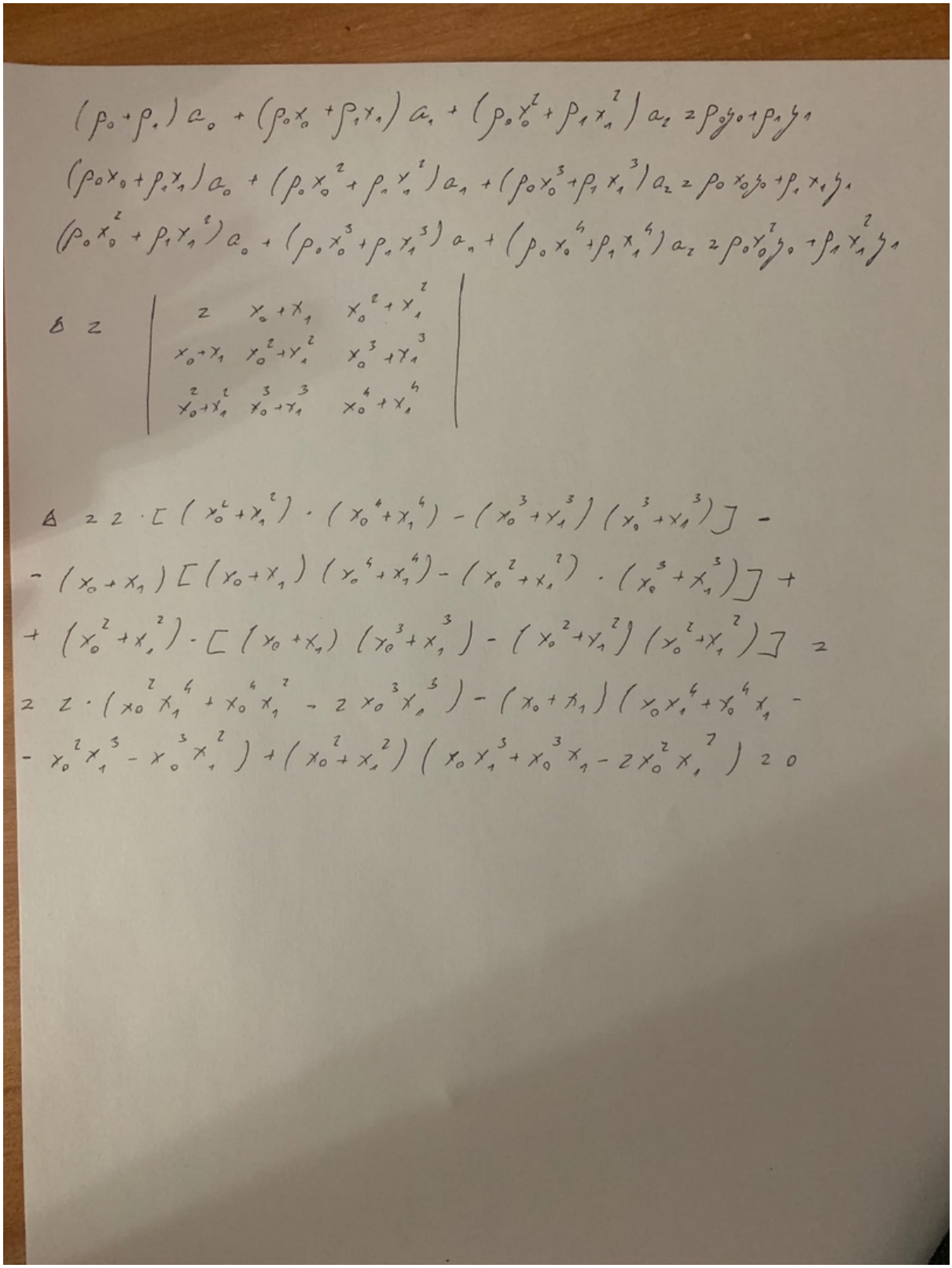
2. Будет ли работать Ваша программа при n >= N ? Что именно в алгоритме требует отдельного анализа данного случая и может привести к аварийной остановке?

При заданном условии (n >= N) невозможно построить полином n степени (из-за недостатка точек), так как при этом определитель матрицы будет равен нулю. Программа работать будет, то в некоторые моменты может возникнуть непредвиденная ситуация

3. Получить формулу для коэффициента полинома a0 при степени полинома n=0. Какой смысл имеет величина, которую представляет данный коэффициент?

$a_0 = (\sum p_i y_i) / \sum p_i$; где p_i – вес i-ой точки.
Если разделить числитель и знаменатель на сумму весов, то в знаменателе будет единица, а в числителе – значения точек умноженные на их вес в приведенном состоянии (все веса в пределах от 0 до 1, соотношения остаются). Данная величина – математическое ожидание.

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



$$(\rho_0+\rho_1)a_0+(\rho_0x_0+\rho_1x_1)a_1+(\rho_0x_0^2+\rho_1x_1^2)a_2=\rho_0y_0+\rho_1y_1$$

$$(\rho_0x_0+\rho_1x_1)a_0+(\rho_0x_0^2+\rho_1x_1^2)a_1+(\rho_0x_0^3+\rho_1x_1^3)a_2=\rho_0x_0y_0+\rho_1x_1y_1$$

$$(\rho_0x_0^2+\rho_1x_1^2)a_0+(\rho_0x_0^3+\rho_1x_1^3)a_1+(\rho_0x_0^4+\rho_1x_1^4)a_2=\rho_0x_0^2y_0+\rho_1x_1^2y_1$$

$$\Delta=\begin{vmatrix} 2 & x_0+x_1 & x_0^2+x_1^2 \\ x_0+x_1 & x_0^2+x_1^2 & x_0^3+x_1^3 \\ x_0^2+x_1^2 & x_0^3+x_1^3 & x_0^4+x_1^4 \end{vmatrix}$$

$$\Delta=2\cdot[(x_0^2+x_1^2)\cdot(x_0^4+x_1^4)-(x_0^3+x_1^3)(x_0^3+x_1^3)]-$$

$$-(x_0+x_1)[(x_0+x_1)(x_0^4+x_1^4)-(x_0^2+x_1^2)\cdot(x_0^3+x_1^3)]+$$

$$+(x_0^2+x_1^2)\cdot[(x_0+x_1)(x_0^3+x_1^3)-(x_0^2+x_1^2)(x_0^2+x_1^2)]=$$

$$=2\cdot(x_0^2x_1^4+x_0^4x_1^2-2x_0^3x_1^3)-(x_0+x_1)(x_0x_1^4+x_0^4x_1-$$

$$-x_0^2x_1^3-x_0^3x_1^2)+(x_0^2+x_1^2)(x_0x_1^3+x_0^3x_1-2x_0^2x_1^2)=0$$

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

$(x^0, x^0)a_0 + (x^0, x^m)a_1 + (x^0, x^n)a_2 = (y, x^0)$
$(x^m, x^0)a_0 + (x^m, x^n)a_1 + (x^m, x^n)a_2 = (y, x^m)$
$(x^n, x^0)a_0 + (x^n, x^m)a_1 + (x^n, x^m)a_2 = (y, x^n)$

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.

Устроить полный перебор по степеням n и m для вычисления коэффициенты ***a*** и вычислить значение матрицы, а потом выбрать самое близкое значение