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 Среднеквадратичное приближение

Студент Пересторонин Павел

Группа ИУ7-43Б

Оценка (баллы) _____

Преподаватель Градов В. М.

Москва.
2020 г.

Техническое задание

Тема: Построение и программная реализация алгоритма наилучшего среднеквадратичного приближения.

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** ρ_i с количеством узлов N.

x	y	ρ_i

Предусмотреть в интерфейсе удобную возможность изменения пользователем весов в таблице.

2. Степень аппроксимирующего полинома — n.

Теоретическая часть.

Под близостью в среднем исходной y и аппроксимирующей φ функций будем понимать результат оценки суммы

$$I = \sum_{i=1}^{N} \rho_i [y(x_i) - \varphi(x_i)]^2 \qquad (4.1)$$

где ρ_i - вес точки. Суммирование выполняется по всем N узлам заданной функции.

Такой вид аппроксимации называют среднеквадратичным приближением.

Наша задача — найти наилучшее приближение (функцию $\varphi(x)$), то есть такое, которое сведет ошибку к минимуму:

$$\sum_{i=1}^{N} \rho_i [y(x_i) - \varphi(x_i)]^2 = min \qquad (4.2)$$

Представим искомую функцию $\varphi(x)$ как некоторую линейную комбинацию n линейно-независимых функций $\varphi_k(x)$:

$$\varphi(x) = \sum_{k=0}^{n} a_k \varphi_k(x) \ . \qquad (4.3)$$

В дальнейшем для сокращения записи будем пользоваться определением скалярного произведения в пространстве дискретно заданных функций:

$$(f, \varphi) = \sum_{i=1}^{N} \rho_i \, f(x_i) \, \varphi(x_i), \ \ \rho_i > 0 .$$

Несложно установить, что имеют место следующие равенства, справедливые для обычного скалярного произведения элементов линейного пространства:

1. $(f, \varphi) = (\varphi, f)$
2. $(f + \varphi, \gamma) = (f, \gamma) + (\varphi, \gamma)$ (4.4)

Подставляя (4.3) в условие (4.2), получим с учетом (4.4.)

$$((y - \varphi), (y - \varphi)) = (y, y) - 2\sum_{k=0}^{n} a_k (y, \varphi_k) + \sum_{k=0}^{n} \sum_{m=0}^{n} a_k a_m (\varphi_k, \varphi_m) = \min \ .$$

Чтобы найти минимум этого выражения, нужно приравнять производную к нулю, так как в точки экстремума она всегда равна нулю, а минимум является таковой.

Дифференцируя это выражение по a_k и приравнивая производные нулю, найдем

$$\sum_{m=0}^{n}(\varphi_k,\varphi_m)a_m=(y,\varphi_k),\quad 0\le k\le n\ . \tag{4.5}$$

Определитель этой системы в силу линейной независимости функций $\varphi_k(x)$ не равен нулю. Следовательно, из системы (4.5) можно найти коэффициенты a_k, определяющие функцию $\varphi(x)$ согласно (4.3) и минимизирующие (4.1). Таким образом, наилучшее среднеквадратичное приближение существует и оно единственно.

Наиболее употребительный вариант метода наименьших квадратов соответствует случаю степенного вида функций $\varphi_k(x)$, т.е. $\varphi_k(x)=x^k$, причем $0\le k\le n$. Обычно в сумме (4.3) берут не более пяти-шести членов.

Система уравнений (4.5) при этом принимает вид

$$\sum_{m=0}^{n}(x^k,x^m)\,a_m=(y,x^k)\ ,\ 0\le k\le n\ , \tag{4.6}$$

Результат работы программы.

Пример как в файле с лекцией:

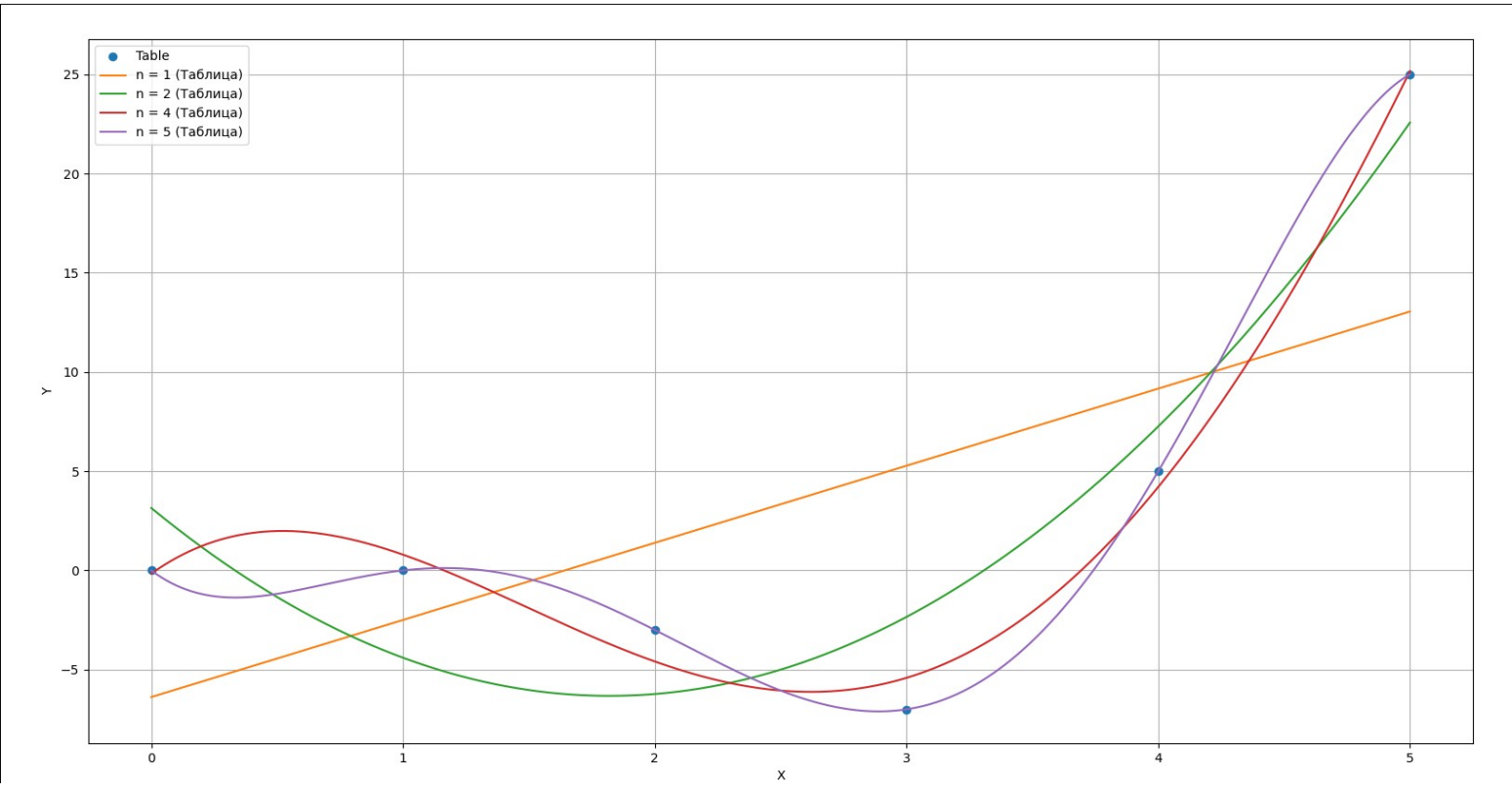
Таблица:

Таблица

X	Y	weight
0.00	0.00	1.00
1.00	0.00	1.00
2.00	-3.00	1.00
3.00	-7.00	1.00
4.00	5.00	1.00
5.00	25.00	1.00

Графики:

Примечание: В скобках указано, по какой таблице строился полином, n — степень полинома.



См. верхний левый угол:

желтая кривая — полином 1 степени.
зеленая кривая — полином 2 степени.
красная кривая — полином 4 степени.
фиолетовая кривая — полином 5 степени.

2 таблицы (с весами и без):

Таблицы:

Таблица без весов

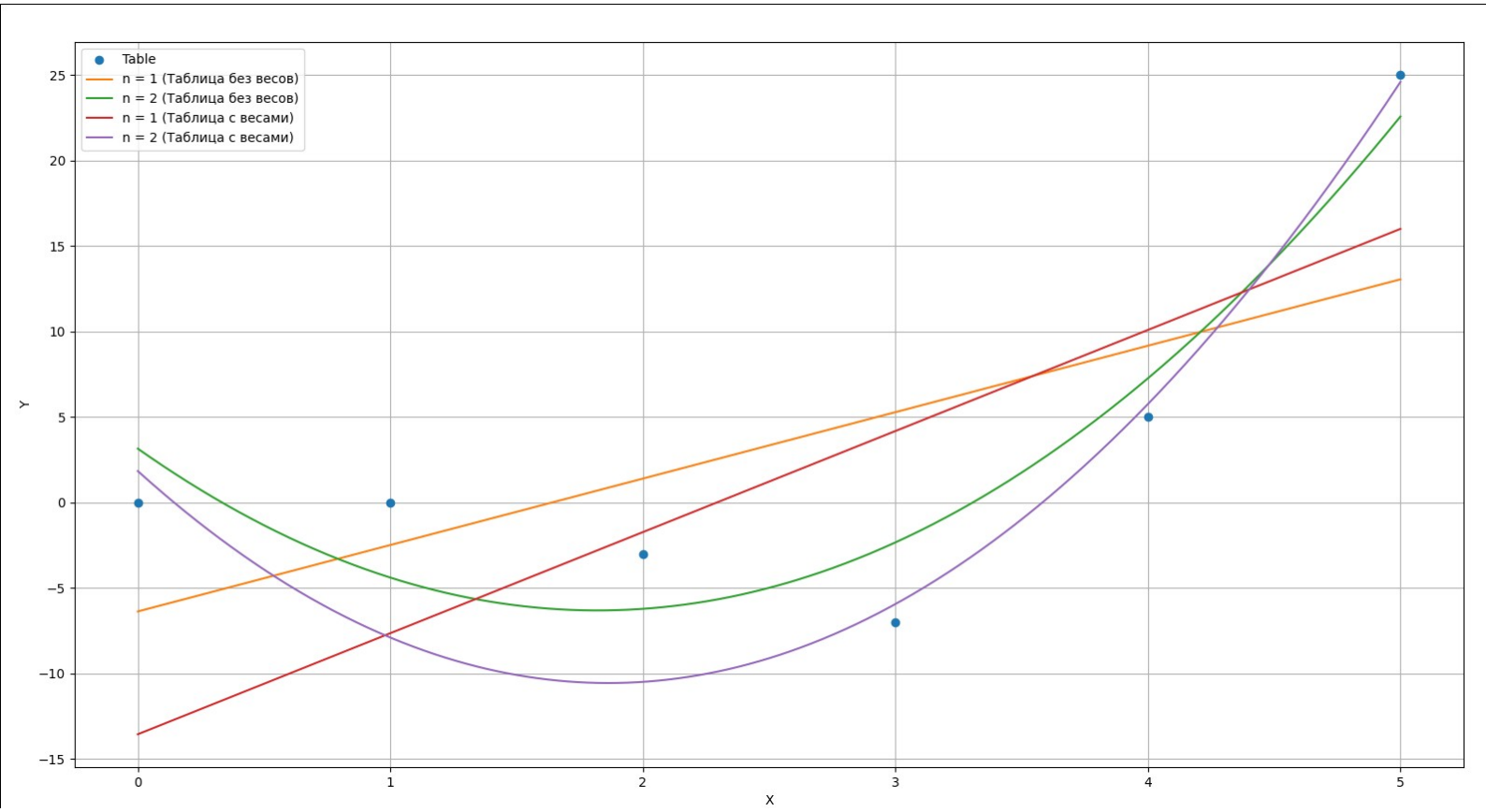
X	Y	weight
0.00	0.00	1.00
1.00	0.00	1.00
2.00	-3.00	1.00
3.00	-7.00	1.00
4.00	5.00	1.00
5.00	25.00	1.00

Таблица с Весами

X	Y	weight
0.00	0.00	3.00
1.00	0.00	1.00
2.00	-3.00	0.80
3.00	-7.00	10.00
4.00	5.00	1.50
5.00	25.00	8.00

Графики:

Примечание: В скобках указано, по какой таблице строился полином, n — степень полинома.



См. верхний левый угол:
желтая кривая — полином 1 степени для таблицы без весов.
зеленая кривая — полином 2 степени для таблицы без весов.
красная кривая — полином 1 степени для таблицы С весами.

фиолетовая кривая — полином 2 степени для таблицы С весами.

Код программы.

```python
from numpy import arange
import matplotlib.pyplot as plt


class Point:
    def __init__(self, x=0, y=0, weight=1):
        self.x = x
        self.y = y
        self.weight = weight

    def __str__(self):
        return f"|{self.x:10.2f} | {self.y:10.2f} | {self.weight:10.2f} |"


def print_table(table):
    print(" ------------------------------------ ")
    print("|     X     |     Y      |   weight   |")
    print(" ------------------------------------ ")
    for i in range(len(table)):
        print(table[i])
    print(" ------------------------------------ ")


def read_from_file(filename):
    points = list()
    with open(filename, "r") as f:
        line = f.readline()
        while line:
            x, y, weight = map(float, line.split())
            points.append(Point(x, y, weight))
            line = f.readline()
    return points


def append_right_side(matrix, points):
    for i in range(len(matrix)):
        res = 0
        for j in range(len(points)):
            res += points[j].weight * points[j].y * points[j].x ** i
        matrix[i].append(res)


def get_coeff(points, degree):
    coeff = 0
    for i in range(len(points)):
        coeff += points[i].weight * points[i].x ** degree
    return coeff


def find_slae_matrix(points, degree):
    matrix = [[get_coeff(points, j + i) for i in range(degree + 1)]
              for j in range(degree + 1)]
    append_right_side(matrix, points)
    return matrix


def get_polynomial_coeffs(matrix):
    for i in range(len(matrix)):
        for j in range(len(matrix)):
            if i == j:
                continue
            mult = matrix[j][i] / matrix[i][i]
            for k in range(0, len(matrix) + 1):
                matrix[j][k] -= mult * matrix[i][k]

    for i in range(len(matrix)):
        mult = matrix[i][i]
        for j in range(len(matrix[i])):
            matrix[i][j] /= mult

    return [matrix[i][-1] for i in range(len(matrix))]


def add_plot(coeffs, label, start, end):
    my_x = list()
    my_y = list()
    step = (end - start) / 1000
    for x in arange(start, end + step, step):
        my_x.append(x)
        y = 0
        for i in range(len(coeffs)):
            y += coeffs[i] * x ** i
        my_y.append(y)

    plt.plot(my_x, my_y, label=label)



def add_table(table, label):
    table_x = [table[i].x for i in range(len(table))]
    table_y = [table[i].y for i in range(len(table))]

    plt.plot(table_x, table_y, 'o', label=label)


def draw_result():
    plt.legend()

    plt.xlabel('X')
    plt.ylabel('Y')

    plt.grid()
    plt.show()



if __name__ == "__main__":
    filenames = input("Enter filenames: ").split()
    labels = input("Enter labels: ").split(',')
    degree = list(map(int, input("Enter polynomial degree: ").split()))

    points = read_from_file(filenames[0])
    add_table(points, "Table")

    for i in range(len(filenames)):

        points = read_from_file(filenames[i])
        print('\n' + labels[i] + '\n')
        print_table(points)

        for j in range(len(degree)):
            slae_matrix = find_slae_matrix(points, degree[j])
            coeffs = get_polynomial_coeffs(slae_matrix)
            add_plot(coeffs, f"n = {degree[j]} ({labels[i]})",
                     points[0].x, points[-1].x)
```

Ответы на вопросы для защиты ЛР.

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 $\sum_{i=1}^{N} \rho_i [y(x_i) - \varphi(x_i)]^2 = min$ выражение в скобках будет тожденственно 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 $n \geq N$? Что именно в алгоритме требует отдельного анализа данного случая и может привести к аварийной остановке?

Программа будет работать некорректно, в некоторых случаях выдавая исключение «ZeroDivisionError» (ошибка деления на ноль), в связи с тем, что в таком случае уравнения СЛАУ не будут линейно-независимыми (правые части), что приводит к тому, определитель матрицы тождественно равен 0. В случае решения методом, которым решал я (приведение левой части расширенной матрицы к единичному виду) это проявится в том, что мне не удастся привести к такому виду, потому что на диагонали найдется нулевой элемент. К аварийной остановке будет приводить процесс перехода от диагональной матрице, к единичной (потому что в одной из строк произойдет деление на ноль). Анализ можно проводить здесь, но лучше делать это при вводе степени полинома или размеров таблицы (в зависимости от очередности).

3. Получить формулу для коэффициента полинома a_0 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.

$$M(X) = \sum_{i=1}^{\infty} x_i p_i$$

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