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 Тартыков Л.Е.

Группа ИУ7-44Б

Оценка (баллы) _____

Преподаватель Градов В.М.

2021 г.

Содержание

Цель работы: Получение навыков владения методами интерполяции таблично заданных функций с помощью кубических сплайнов.

Исходные данные

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

x	**y**	**P_i**
0	0	1
1	3	5
2	8	2
3	5	4
4	4	1
5	2	5

2. Степень аппроксимирующего полинома - n.

\

Код программы

Замечание: данная программа написана на языке Python 3.8

Листинг 1; main.py

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

MIN_ITEM_MENU = '0'; MAX_ITEM_MENU = '4'


def print_menu():
    print('Меню:\n' +
          '1. Загрузить данные из файла.\n' +
          '2. Вывести таблицу.\n' +
          '3. Найти решение.\n' +
          '4. Изменить вес точки.\n' +
          '0. Выйти из программы.\n')

def input_menu_item():
    menu_choice = ''
    print(f"Введите пункт меню: ", end='')
    menu_choice = input()
    while menu_choice < MIN_ITEM_MENU or menu_choice > MAX_ITEM_MENU:
        print(f"Некорректный пункт меню. Введите еще раз: ", end='')
        menu_choice = input()
    return menu_choice

def choose_menu_item(menu_choice, data, data_2, polynom_coeffs_a, a_equal_mass):
    if menu_choice == '1':
        file_name_1 = 'data.txt'
        file_name_2 = 'data_2.txt'
        data = read_data_from_file(file_name_1)
        data_2 = read_data_from_file(file_name_2)
        print(f"\nДанные загружены успешно.\n")
    elif menu_choice == '2':
        print_data(data)
    elif menu_choice == '3':
        degree_polynom = input_polynom_degree()
        polynom_coeffs_a = find_root(data, degree_polynom)
        a_equal_mass = find_root(data_2, degree_polynom)
        create_graph(data, polynom_coeffs_a, a_equal_mass, degree_polynom)
    elif menu_choice == '4':
        data = change_mass_point(data)
        print(f"Изменение прошло успешно.\n")
    return data, data_2, polynom_coeffs_a, a_equal_mass

def read_data_from_file(file_name):
    file = open(file_name, 'r')
```

```
    #считывание x, y, p из файла
    data = [line.replace("\n", "").split() for line in file]
    for i in range(0, len(data)):
        for j in range(0, len(data[i])):
            data[i][j] = int(data[i][j])

    return data

def print_data(data):
    if (data == []):
        print(f"Ошибка: данные еще не загружены.")
    else:
        print(f"x | y | p\n" +
              "----------")
        for i in range(len(data)):
            print(f"{data[i][0]} | {data[i][1]} | {data[i][2]}")

def create_graph(data, a, a_equal_mass, degree_polynom):
    dx = 10
    if (len(data) > 1):
        dx = data[1][0] - data[0][0]

    x = np.linspace(data[0][0] - dx, data[-1][0] + dx, 100)
    y = []
    for i in x:
        temp_value_y = 0
        for j in range(0, degree_polynom + 1):
            temp_value_y += f(i, j) * a[j]
        y.append(temp_value_y)
    plt.plot(x, y, label='разные веса точек')

    x = np.linspace(data[0][0] - dx, data[-1][0] + dx, 100)
    y = []
    for i in x:
        temp_value_y = 0
        for j in range(0, degree_polynom + 1):
            temp_value_y += f(i, j) * a_equal_mass[j]
        y.append(temp_value_y)
    plt.plot(x, y, label='одинаковые веса точек')

    x_table = [a[0] for a in data]
    y_table = [a[1] for a in data]

    plt.plot(x_table, y_table, 'ro', color='red', label='таблица')
    plt.xlabel('x')
    plt.ylabel('y')
    plt.grid(True)
    plt.legend(loc = 'best')

    plt.show()
```

```
def f(x, n):
    return x ** n

def change_mass_point(data):
    if (data == []):
        print(f"Ошибка: данные еще не загружены.")
    else:
        i_change = input_index_change()
        new_mass = input_new_mass(data)
        change_mass_value(data[i_change], new_mass)
    return data

def input_index_change():
    print(f"Введите индекс элемента, который хоите изменить: ", end='')
    index_change = input()
    while index_change < '0' or index_change > '6':
        print(f"Некорректный пункт меню. Введите еще раз: ", end='')
        index_change = input()
    return int(index_change)

def input_new_mass(data):
    flag_is_digit = 0
    while flag_is_digit == 0:
        print(f"Введите новый вес: ", end='')
        new_mass = input()
        flag_is_digit = check_is_digit(new_mass)
        if flag_is_digit == 0:
            print(f"Некорректное значение.", end=' ')
    return new_mass

def check_is_digit(new_mass):
    flag_is_digit = False
    if new_mass.isdigit():
        flag_is_digit = True
    return flag_is_digit

def change_mass_value(point_change, new_mass):
    point_change[2] = int(new_mass)

def find_root(data, degree_polynom):
    if data == []:
        print(f"Ошибка: данные еще не загружены.")
    else:
        matrix, column = create_slau_matrix(data, degree_polynom)
        append_column_of_free_members(matrix, column)
        solve_matrix_by_gauss(matrix)
        a = find_polynomial_coeffs(matrix)
    return a

def create_slau_matrix(data, degree_polynom):
    len_data = len(data)
```

```
    matrix = [[0 for i in range(0, degree_polynom + 1)] for j in range (0, degree_polynom + 1)]
    column = [0 for i in range(0, degree_polynom + 1)]
    for m in range(0, degree_polynom + 1):
        for i in range(0, len_data):
            temp_value = data[i][2] * f(data[i][0], m)
            for k in range(0, degree_polynom + 1):
                matrix[m][k] += temp_value * f(data[i][0], k)
            column[m] += temp_value * data[i][1]
    return matrix, column

def append_column_of_free_members(matrix, column):
    for i in range(len(column)):
        matrix[i].append(column[i])

def solve_matrix_by_gauss(matrix):
    len_matrix = len(matrix)
    for i in range(len_matrix):
        for j in range(i + 1, len_matrix):
            coeff = -(matrix[j][i] / matrix[i][i])
            for k in range(i, len_matrix + 1):
                matrix[j][k] += coeff * matrix[i][k]

def find_polynomial_coeffs(matrix_x_degree):
    len_matrix = len(matrix_x_degree)
    a = [0 for i in range(len_matrix)]
    for i in range(len_matrix - 1, -1, -1):
        for j in range(len_matrix - 1, i, -1):
            matrix_x_degree[i][len_matrix] -= a[j] * matrix_x_degree[i][j]
        a[i] = matrix_x_degree[i][len_matrix] / matrix_x_degree[i][i]
    return a

def input_polynom_degree():
    print(f"Введите степень полинома (от 0 до 6): ", end='')
    degree_polynom = input()
    while degree_polynom < '0':
        print(f"Некорректный ввод. Введите еще раз: ", end='')
        degree_polynom = input()
    return int(degree_polynom)

def main():
    menu_choice = -1
    data = []; data_2 = []
    polynom_coeffs_a = []; a_equal_mass = []
    while menu_choice != '0':
        print_menu()
        menu_choice = input_menu_item()
        data, data_2, polynom_coeffs_a, a_equal_mass = choose_menu_item(menu_choice, data, data_2,
polynom_coeffs_a, a_equal_mass)

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

Результаты работы

Первый случай: веса точек одинаковые

Таблица 1: Исходная таблица с одинаковыми весами

x	y	P_i
0	0	1
1	3	1
2	8	1
3	5	1
4	4	1
5	2	1

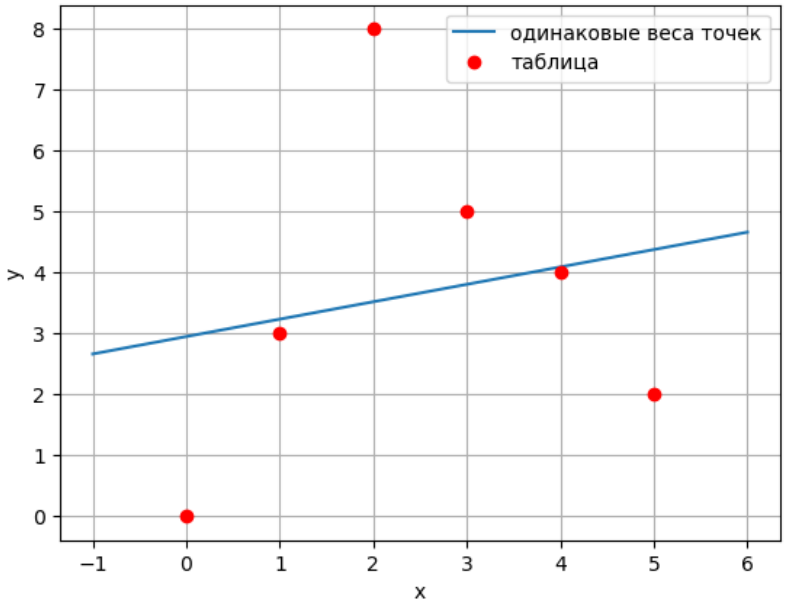

Рисунок 1: график при n = 1

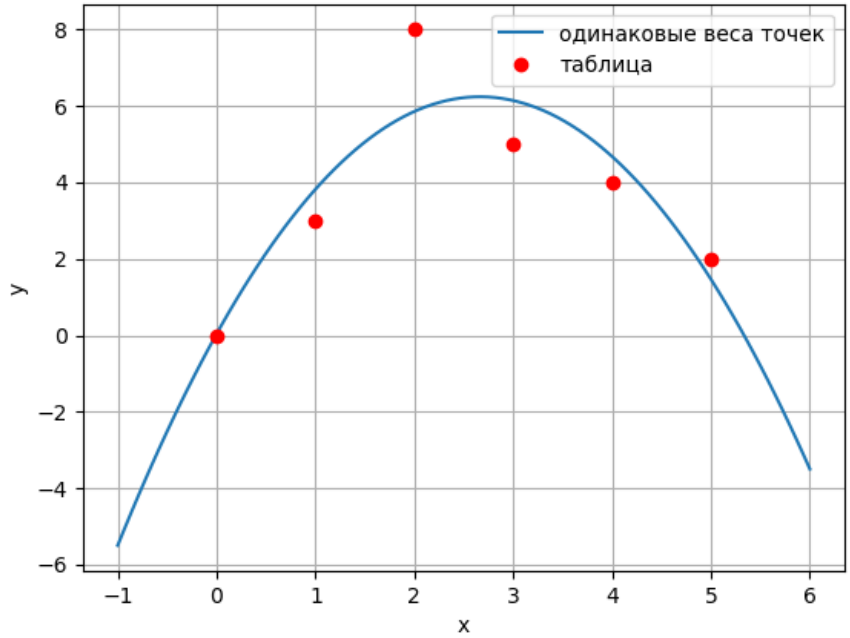

Рисунок 2: график при n = 2

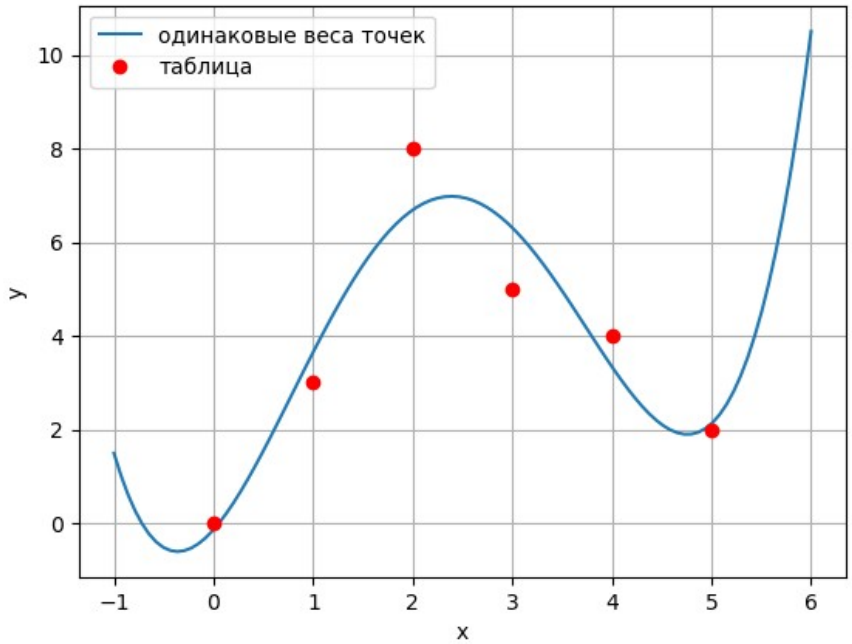

Рисунок 3: график при n = 4

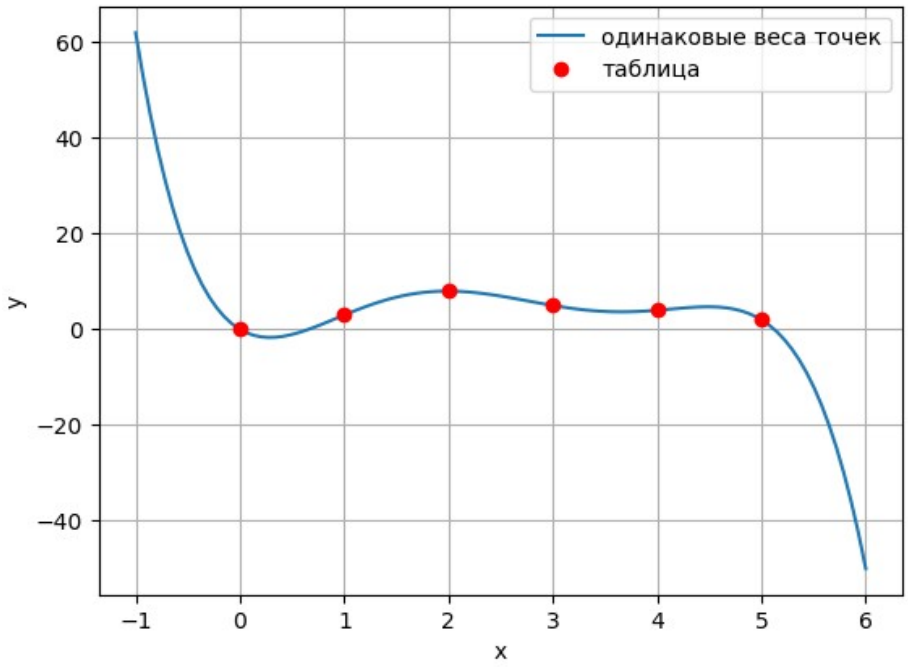

Рисунок 4: график при n = 5

Второй случай: веса точек разные

x	**y**	**P_i**
0	0	1
1	3	5
2	8	2
3	5	4
4	4	1
5	2	5

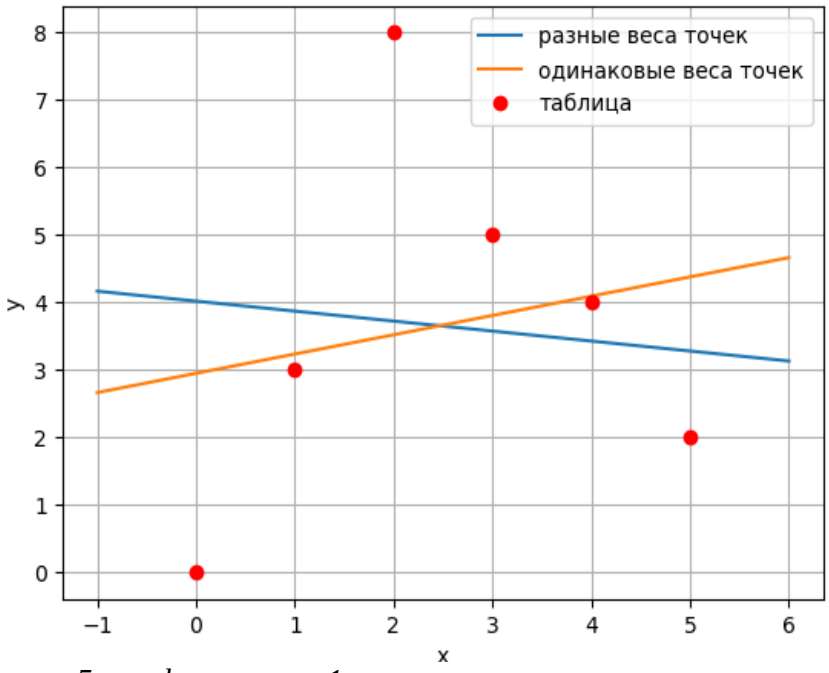

Рисунок 5: график при n = 1

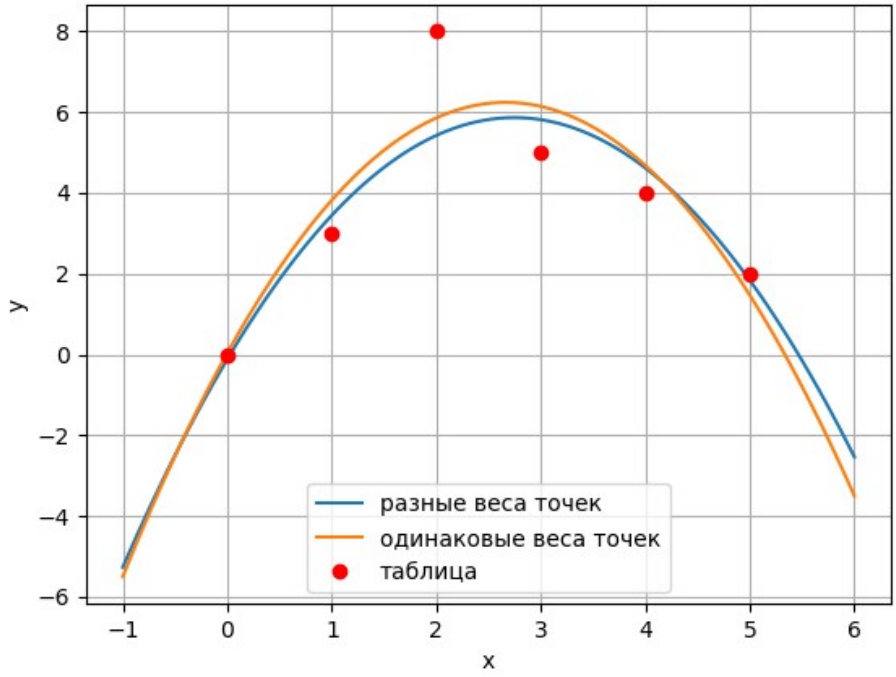

Рисунок 6: график при n = 2

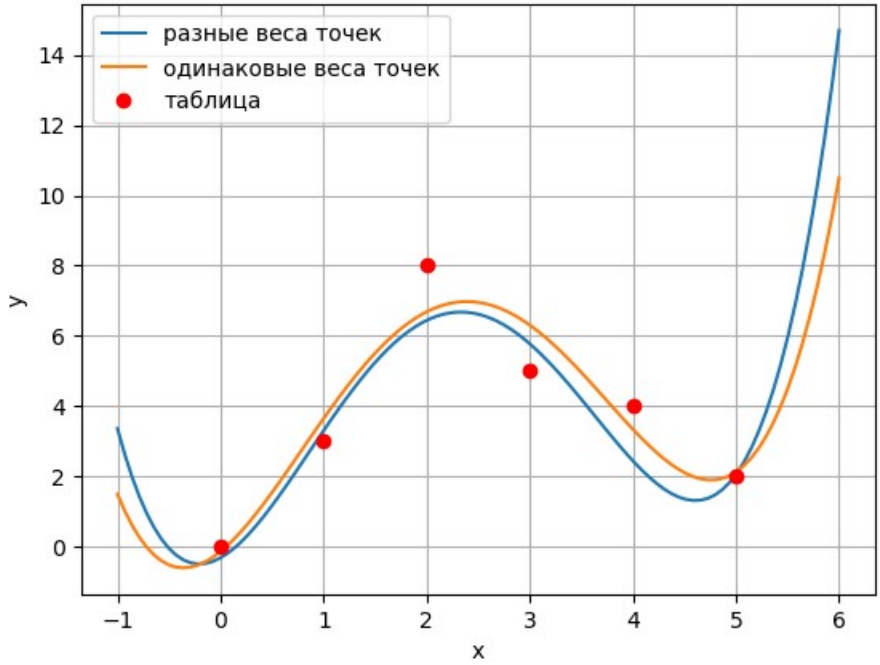

Рисунок 7: график при n = 4

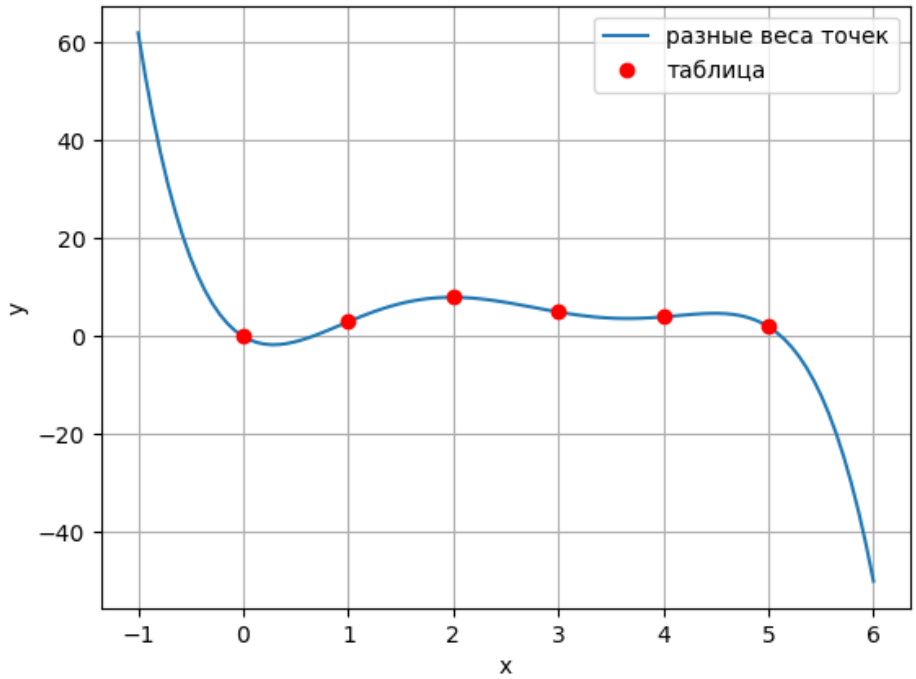

Рисунок 8: график при n = 5

Ответы на вопросы при защите лабораторной работы

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

Построенный полином будет проходить через все точки, причем веса точек не будут сказываться на результате построенного полинома.

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 и выведет результат. Это связано с тем, что в результате вычислений с действительными числами будут возникать погрешности. Но на самом деле построить полином степени $n \geq N$ нельзя, так как определитель матрицы, составленной как СЛАУ, будет равен нулю

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

При степени полинома n = 0 получим уравнение
$(x^0, x^0)a_0 = (y, x^0)$, где

$$(x^0, x^0) = \sum_{i=1}^{N} (p_i)$$

$$(y, x^0) = \sum_{i=1}^{N} (p_i y_i)$$

Коэффициент a_0:

$$a_o = \frac{\sum_{i=1}^{N} (p_i y_i)}{\sum_{i=1}^{N} (p_i)}$$

Полученный коэффициент a^0- среднее значение случайной величины (взвешенное по вероятностям возможных значений)

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

Из исходных данных зададим таблицу

x_i	y_i	p_i
x_0	y_0	1
x_1	y_1	1

Составим систему линейных уравнений

$$\begin{cases} a_o+(x_0+x_1)a_1+(x_0^2+x_1^2)a_2=y_0+y_1 \\ (x_0+x_1)a_0+(x_0^2+x_1^2)a_1+(x_0^3+x_1^3)a_2=y_0x_0+y_1x_1 \\ (x_0^2+x_1^2)a_0+(x_0^3+x_1^3)a_1+(x_0^4+x_1^4)a_2=y_0x_0^2+y_1x_0^2 \end{cases}$$

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

5. Построить СЛАУ при выборочном задании степеней аргумента полинома $\phi(x)=a_0+a_1x^m+a_2x^n$, причем степени n и m в этой формуле известны.

Пусть степень полинома будет равна p (начальное значение равно 0). По формуле $\sum_{m=0}^{n}(x^k,x^m)a_m=(y,x^k)$, где $(x^k,x^m)=\sum_{i=1}^{N}\rho_i x_i^{k+m}$, $(y,x^k)=\sum_{i=1}^{N}\rho_i y_i x_i^k$ составим уравнение

$(x_0,x_0)a_0+(x_0,x^i)a_i+(x_0,x^j)a_2=(y,x^0)$, где $0\le i<n, 0\le j<m$

Таких уравнений будет p + 1 штук, в каждом последующем i и j будут увеличиваться на единицу