| | |
|---|---|
| | **Министерство науки и высшего образования Российской Федерации<br>Федеральное государственное бюджетное образовательное учреждение<br>высшего образования<br>«Московский государственный технический университет<br>имени Н.Э. Баумана<br>(национальный исследовательский университет)»<br>(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:** Хамзина Р.Р.

**Группа:** ИУ7-43Б

**Оценка (баллы):** _______________

**Преподаватель:** Градов В.М.

Москва

*2021 г*

**Цель работы:** получение навыков построения алгоритма метода наименьших квадратов с использованием полинома заданной степени при аппроксимации табличных функций с весами.

# 1 Исходные данные

1. Таблица функции с количеством узлов N.

   Для отладки:

| x | y | ρ |
|---|---|---|
| $x_0$ | $y_1$ | $p_1$ |
| ... | ... | ... |

2. Степень аппроксимирующего полинома - n.

   Для отладки: n = 1, 2, 5

# 2 Код программы

```python
from dataclasses import dataclass
import tkinter as tk
import numpy as np
import matplotlib.pyplot as plt
from math import sqrt
from random import uniform

@dataclass
class Graphic:
    """
        Константы графического модуля.
    """

    window_length = 2000
    window_heigth = 1500
    window_size = "2000x1500"
    window_title = "Лабораторная работа № 4"

    font_bold = "FreeMono 14 bold"
    font = "FreeMono 14"

@dataclass
class Colour:

    back = "#ABCDEF"
    button = "#DDF1FF"


def draw(all_rms):
    """
        Вывод графиков.
    """

    plt.plot(all_rms[0][2], all_rms[0][3], 'ro', label = "Данные")

    for rms in all_rms:
        plt.plot(rms[2], rms[0], label = "p = " + str(rms[1]))

    plt.grid()
    plt.legend(fontsize = 20,
               ncol = 2)
    plt.tick_params(labelsize = 20)
    plt.title("Наилучшее среднеквадратичное приближение", fontsize = 20)
    plt.xlabel("x", fontsize = 20)
    plt.ylabel("y", fontsize = 20)

    plt.show()

def get_rms(arguments, factors_a):
    """
        Получить наилучшее среднеквадратичное приближение.
    """

    rms = []

    for x in arguments:
        result = 0

        for i in range(len(factors_a)):
            result += factors_a[i] * (x ** i)

        rms.append(result)

    return rms

def get_x_y(table_nodes):
    """
        Получить список аргументов и значений.
    """

    x, y = [], []

    for node in table_nodes:
        x.append(node[0])
        y.append(node[1])

    return x, y
```

```python
def insert_column(matrix, free_membrs, j):
    """
        Вставка столбца свободных членов в
        матрицу.
    """

    i = 0

    for row in matrix:
        row[j], free_membrs[i] = free_membrs[i], row[j]
        i += 1

    return matrix, free_membrs


def get_factors_a(matrix, free_membrs):
    """
        Получить коэффициенты а.
    """

    main_det = np.linalg.det(matrix)
    factors_a = []

    for j in range(len(matrix[0])):
        matrix, free_membrs = insert_column(matrix, free_membrs, j)
        now_det = np.linalg.det(matrix)
        factors_a.append(now_det / main_det)
        matrix, free_membrs = insert_column(matrix, free_membrs, j)

    return factors_a

def get_prod_x(table_nodes, k, m):
    """
        Получить произведение (x^k, x^m).
    """

    prod_x = 0

    for node in table_nodes:
        prod_x += node[2] * (node[0] ** (k + m))

    return prod_x

def get_prod_y(table_nodes, k):
    """
        Получить произведение (y, x^k).
    """

    prod_y = 0

    for node in table_nodes:
        prod_y += node[2] * node[1] * (node[0] ** k)

    return prod_y

def get_matrix(table_nodes, power):
    """
        Создание СЛАУ.
    """

    matrix = []
    free_membrs = []

    for k in range(power + 1):
        row = []

        for m in range(power + 1):
            row.append(get_prod_x(table_nodes, k, m))

        matrix.append(row)

        free_membrs.append(get_prod_y(table_nodes, k))

    return matrix, free_membrs
```

```python
def get_table_nodes(table_entry):
    """
        Ввод таблицы узлов.
    """

    table_nodes = []

    for entry in table_entry:
        x = float(entry[0].get())
        y = float(entry[1].get())
        p = float(entry[2].get())

        table_nodes.append([x, y, p])

    return table_nodes

def mediator(table_entry, power_entry, all_rms):
    """
        Получить список данных для вывода графиков.
    """

    table_nodes = get_table_nodes(table_entry)
    power = int(power_entry.get())

    matrix, free_membrs = get_matrix(table_nodes, power)
    factors_a = get_factors_a(matrix, free_membrs)
    x, y = get_x_y(table_nodes)
    all_rms.append([get_rms(x, factors_a), power, x, y])

def input_nodes(window, count_nodes_entry):
    """
        Ввод узлов.
    """

    x_lbl = tk.Label(window, text = "x:",
                           font = Graphic.font_bold,
                           bg = Colour.back)
    x_lbl.place(x = 50, y = 250)

    y_lbl = tk.Label(window, text = "y:",
                           font = Graphic.font_bold,
                           bg = Colour.back)
    y_lbl.place(x = 300, y = 250)

    p_lbl = tk.Label(window, text = "p:",
                           font = Graphic.font_bold,
                           bg = Colour.back)
    p_lbl.place(x = 550, y = 250)

    count_nodes = int(count_nodes_entry.get())
    y_start = 310

    table_entry = []

    for i in range(count_nodes):
        x_entry = tk.Entry(window, font = Graphic.font,
                                   justify = tk.CENTER,
                                   width = 8)
        x_entry.insert(tk.END, str(i + 1))
        x_entry.place(x = 50, y = y_start + i * 60)

        y_entry = tk.Entry(window, font = Graphic.font,
                                   justify = tk.CENTER,
                                   width = 8)
        y_entry.insert(tk.END, str(uniform(0, 10)))
        y_entry.place(x = 300, y = y_start + i * 60)

        p_entry = tk.Entry(window, font = Graphic.font,
                                   justify = tk.CENTER,
                                   width = 8)
        p_entry.place(x = 550, y = y_start + i * 60)
        p_entry.insert(tk.END, "1")

        table_entry.append([x_entry, y_entry, p_entry])
```

```
    # Ввод степени полинома

    power_lbl = tk.Label(window, text = "Введите степень полинома:",
                             font = Graphic.font_bold,
                             bg = Colour.back)
    power_lbl.place(x = 50, y = y_start + 60 * count_nodes)

    power_entry = tk.Entry(window, font = Graphic.font,
                               justify = tk.CENTER)
    power_entry.place(x = 50, y = y_start + 60 * (count_nodes + 1))

    # Решение

    all_rms = []

    solve_button = tk.Button(window, text = "Решить",
                                 bg = Colour.button,
                                 font = Graphic.font,
                                 bd = 6,
                                 command = lambda : mediator(table_entry, power_en
try, all_rms))
    solve_button.place(x = 50, y = y_start + 60 * (count_nodes + 2))

    draw_button = tk.Button(window, text = "Нарисовать",
                                bg = Colour.button,
                                font = Graphic.font,
                                bd = 6,
                                command = lambda : draw(all_rms))
    draw_button.place(x = 50, y = y_start + 60 * (count_nodes + 3) + 40)


def create_interface():
    """
        Создание интерфейса.
    """

    window = tk.Tk()
    window.title(Graphic.window_title)
    window.geometry(Graphic.window_size)
    window.config(bg = Colour.back)

    # Ввод количества узлов

    count_nodes_lbl = tk.Label(window, text = "Введите число узлов:",
                                   font = Graphic.font_bold,
                                   bg = Colour.back)
    count_nodes_lbl.place(x = 50, y = 50)

    count_nodes_entry = tk.Entry(window, font = Graphic.font,
                                     justify = tk.CENTER)
    count_nodes_entry.place(x = 50, y = 110)

    # Ввод узлов

    input_nodes_button = tk.Button(window, text = "Ввести узлы",
                                       bg = Colour.button,
                                       font = Graphic.font,
                                       bd = 6,
                                       command = lambda : input_nodes(window, coun
t_nodes_entry))
    input_nodes_button.place(x = 50, y = 170)

    window.mainloop()

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

# 3 Результаты работы

1. Необходимо найти наилучшее приближение, т. е. такую функцию $\varphi(x)$, которая удовлетворяет условию:

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

   N - число узлов, $p_i$ - вес узла $(x_i;y_i)$.

   Для этого используем метод приближения - метод наименьших квадратов. Представим $\varphi(x)$ следующим образом:

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

   Подставив равенство в условие, получим:

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

   n - степень полинома.

   Продифференцируем по $a_k$ и приравняем производные 0, тогда:

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

   $$\varphi_k(x) = x^k$$

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

   $$(x^k, x^m) = \sum_{i=1}^{N} \rho_i x_i^{k+m}, \quad (y, x^k) = \sum_{i=1}^{N} \rho_i y_i x_i^k$$

2. Найдем все $(x^k, x^m)$.
3. Найдем все $(y, x^k)$.
4. Решим полученную СЛАУ и найдем $a_i$, $1 \le i \le N$.
5. Найдем приближенные значения следующим образом:

   $y = a_1 + a_2 x + a_3 x^2 + \ldots + a_N x^{N-1}$

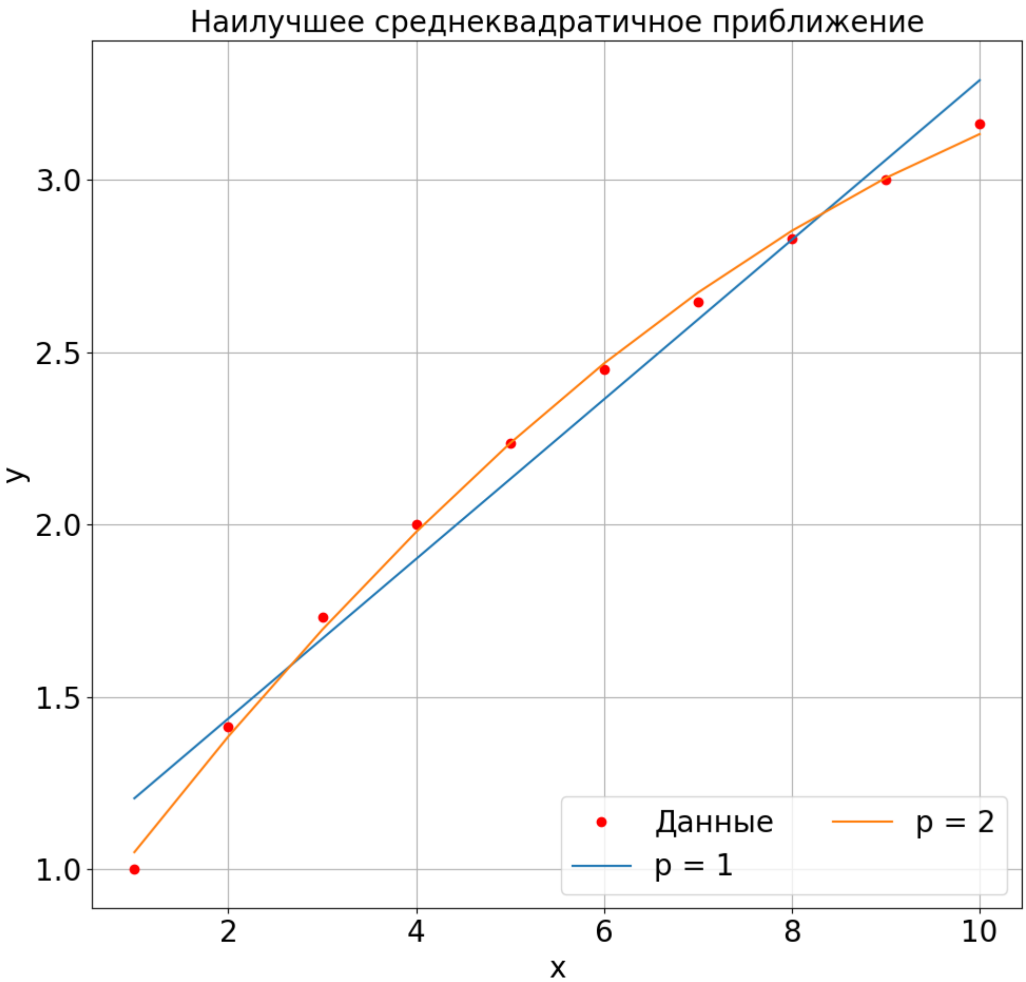
1. Веса всех  точек одинаковы:

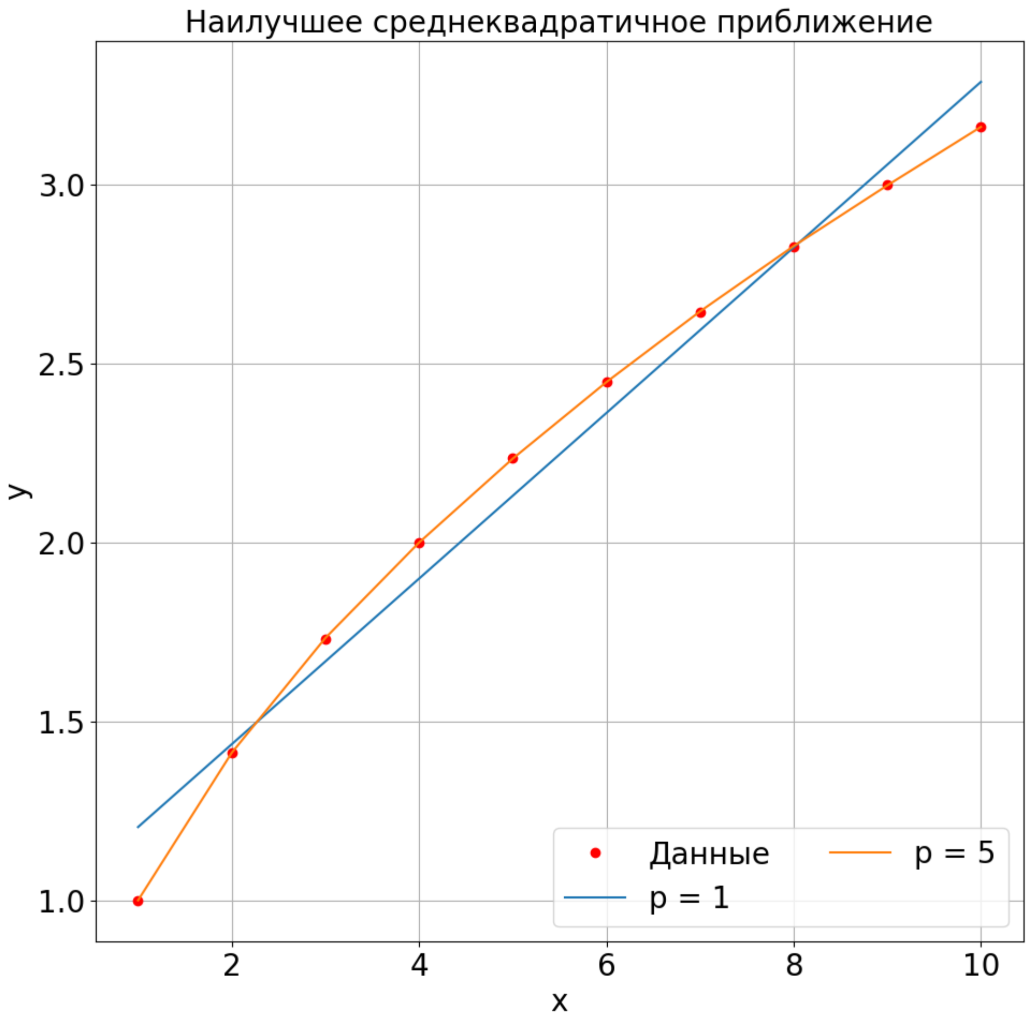
**Введите число узлов:**

10

Ввести узлы

| x: | y: | p: |
|---|---|---|
| 1 | 1.0 | 1 |
| 2 | 1.414213 | 1 |
| 3 | 1.732050 | 1 |
| 4 | 2.0 | 1 |
| 5 | 2.236067 | 1 |
| 6 | 2.449489 | 1 |
| 7 | 2.645751 | 1 |
| 8 | 2.828427 | 1 |
| 9 | 3.0 | 1 |
| 10 | 3.162277 | 1 |






2. Веса точек разные:

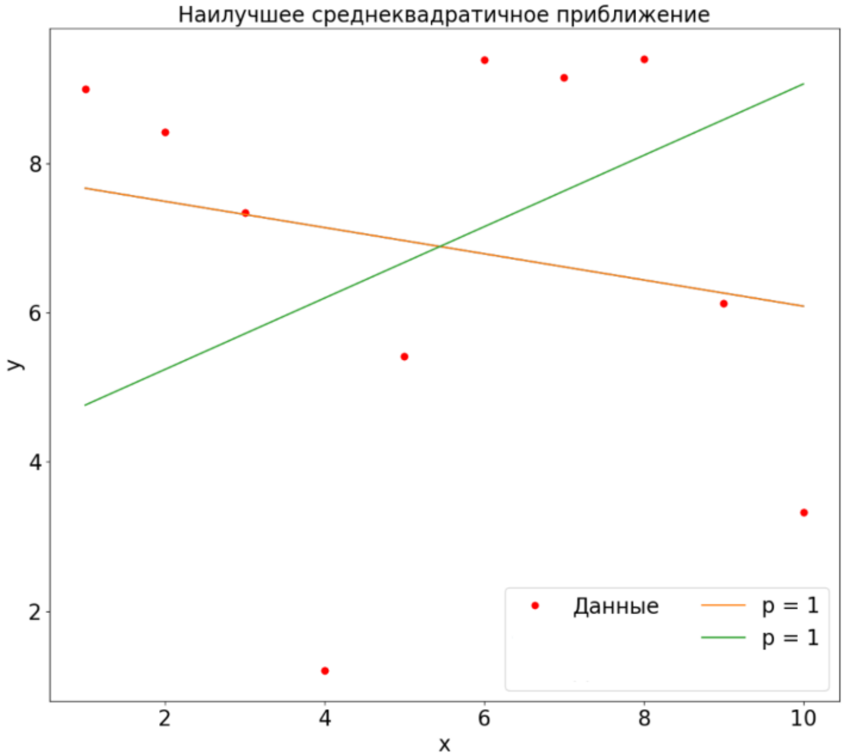
Изменение знака углового коэффициента:

**Введите число узлов:**

10

Ввести узлы

| x: | y: | p: |
|---|---|---|
| 1 | 9.002135 | 1 |
| 2 | 8.423846 | 1 |
| 3 | 7.337867 | 1 |
| 4 | 1.200247 | 1 |
| 5 | 5.412123 | 15 |
| 6 | 9.397060 | 1 |
| 7 | 9.158902 | 1 |
| 8 | 9.400456 | 15 |
| 9 | 6.126599 | 1 |
| 10 | 3.326074 | 1 |




Оранжевая прямая - все веса точек равны 1, зеленая прямая - веса точек разные.

## 4 Вопросы при защите лабораторной работы

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

График полинома будет проходить через все N точек, так как для определения полинома N - 1 степени необходимо N точек. Поэтому $\varphi(x)_i = 0$, $1 \leq i \leq N$. Тогда,

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

минимальна, независимо от весов узлов.

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

Программа будет работать. При n = N определитель СЛАУ равен 0, и полином с такими условиями построить нельзя.

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

Начальные данные:

| $x_i$ | $y_i$ | $\rho_i$ |
|---|---|---|
| $x_0$ | $y_0$ | $\rho_0$ |
| $x_1$ | $y_1$ | $\rho_1$ |
| ... | ... | ... |
| $x_N$ | $y_N$ | $\rho_N$ |

$(x^0, x^0)a_0 = (y, x^0)$

$(\rho_0 x_0^0+\rho_1 x_1^0 + \dots + \rho_N x_N^0)a_0 = \rho_0 y_0 x^0 + \rho_1 y_1 x^0 + \dots +\rho_N y_N x^0$

$a_0 = (y, x^0) / (x^0, x^0) = (\rho_0 y_0 x^0 + \rho_1 y_1 x^0 + \dots +\rho_N y_N x^0) / (\rho_0 x_0^0+\rho_1 x_1^0 + \dots + \rho_N x_N^0) = (\rho_0 y_0 + \rho_1 y_1 + \dots +\rho_N y_N) / (\rho_0+\rho_1 + \dots + \rho_N)$

Величина, которая представляет коэффициент $a_0$ - математическое ожидание.

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

Начальные данные:

| $x_i$ | $y_i$ | $\rho_i$ |
|---|---|---|
| $x_0$ | $y_0$ | 1 |
| $x_1$ | $y_1$ | 1 |

Построим СЛАУ:

$(x^0, x^0)a_0 + (x^0, x^1)a_1 + (x^0, x^2)a_2 = (y, x^0)$

$(x^1, x^0)a_0 + (x^1, x^1)a_1 + (x^1, x^2)a_2 = (y, x^1)$

$(x^2, x^0)a_0 + (x^2, x^1)a_1 + (x^2, x^2)a_2 = (y, x^2)$

$(x_0{}^0+x_1{}^0)a_0 + (x_0{}^1+x_1{}^1)a_1 + (x_0{}^2+x_1{}^2)a_2 = y_0x^0 + y_1x^0$

$(x_0{}^1+x_1{}^1)a_0 + (x_0{}^2+x_1{}^2)a_1 + (x_0{}^3+x_1{}^3)a_2 = y_0x^1 + y_1x^1$

$(x_0{}^2+x_1{}^2)a_0 + (x_0{}^3+x_1{}^3)a_1 + (x_0{}^4+x_1{}^4)a_2 =y_0x^2 + y_1x^2$

Матрица СЛАУ:

| | | |
|---|---|---|
| $(x_0{}^0+x_1{}^0)$ | $(x_0{}^1+x_1{}^1)$ | $(x_0{}^2+x_1{}^2)$ |
| $(x_0{}^1+x_1{}^1)$ | $(x_0{}^2+x_1{}^2)$ | $(x_0{}^3+x_1{}^3)$ |
| $(x_0{}^2+x_1{}^2)$ | $(x_0{}^3+x_1{}^3)$ | $(x_0{}^4+x_1{}^4)$ |

Её определитель:

$(x_0{}^0+x_1{}^0)(x_0{}^2+x_1{}^2)(x_0{}^4+x_1{}^4) + 2 (x_0{}^1+x_1{}^1)(x_0{}^2+x_1{}^2)(x_0{}^3+x_1{}^3) - (x_0{}^2+x_1{}^2)(x_0{}^2+x_1{}^2)(x_0{}^2+x_1{}^2) - (x_0{}^1+x_1{}^1)(x_0{}^1+x_1{}^1)(x_0{}^4+x_1{}^4) - (x_0{}^0+x_1{}^0)(x_0{}^3+x_1{}^3)(x_0{}^3+x_1{}^3) = 0$ ⇒система не имеет решения.

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

Начальные данные:

| $x_i$ | $y_i$ | $\rho_i$ |
|---|---|---|

| | | |
|---|---|---|
| $x_0$ | $y_0$ | $\rho_0$ |
| $x_1$ | $y_1$ | $\rho_1$ |
| ... | ... | ... |
| $x_N$ | $y_N$ | $\rho_N$ |

Построим СЛАУ:

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

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

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

$(\rho_0 x_0^0+\rho_1 x_1^0 + \ldots + \rho_N x_N^0)a_0 + (\rho_0 x_0^m+\rho_1 x_1^m + \ldots + \rho_N x_N^m)a_1 + (\rho_0 x_0^n+\rho_1 x_1^n+ \ldots + \rho_N x_N^n)a_2 = \rho_0 y_0 x^0 + \rho_1 y_1 x^0 + \ldots +\rho_N y_N x^0$

$(\rho_0 x_0^m+\rho_1 x_1^m + \ldots + \rho_N x_N^m)a_0 + (\rho_0 x_0^{2m}+\rho_1 x_1^{2m} + \ldots + \rho_N x_N^{2m})a_1 + (\rho_0 x_0^{m+n}+\rho_1 x^{m+n} + \ldots + \rho_N x_N^{m+n})a_2 = \rho_0 y_0 x^m + \rho_1 y_1 x^m + \ldots +\rho_N y_N x^m$

$(\rho_0 x_0^n+\rho_1 x_1^n + \ldots + \rho_N x_N^n)a_0 + (\rho_0 x_0^{n+m}+\rho_1 x_1^{n+m} + \ldots + \rho_N x_N^{n+m})a_1 + (\rho_0 x_0^{2n}+\rho_1 x_1^{2n} + \ldots + \rho_N x_N^{2n})a_2 = \rho_0 y_0 x^n + \rho_1 y_1 x^n + \ldots +\rho_N y_N x^n$

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.

1. Перебор всех степеней n и m.
2. Поиск для каждой пары коэффициента $a_i$.
3. Сравнение полученных наборов с данными. Искомый набор - тот, у которого разница минимальна.