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 Наилучшее среднеквадратичное приближение

Студент Сукочева А.

Группа ИУ7-43Б

Оценка (баллы) ___________________

Преподаватель Градов В. М.

Москва.
2020 г.

Задание:

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.

Задано:

1. Таблица функции с весами ρ_i с количеством узлов N.
2. Степень аппроксимирующего полинома - n.

Теория:

При интерполировании функции строят некоторую новую функцию, совпадающую с заданной в фиксированных узлах. В данном случае мы приближаем функцию не по точкам, а в среднем. Это целесообразно использовать, когда, к примеру, значения функции в узлах определены не точно.

Пусть имеется множество функций $\varphi(x)$, принадлежащих линейному пространству функций. Под близостью в среднем исходной y и аппроксимирующей φ функций будем понимать результат оценки суммы:

$I = \sum_{i=1}^{N} \rho_i [\, y(x_i) - \varphi(x_i)\,]^2$, Где ρ_i - вес точки (Под весом точки будем понимать величину, обратную относительной погрешности задания функции (т.е. чем более точное значение имеет табличная функция в некоторой точке, тем больше ее вес и тем ближе к ней пройдет график аппроксимирующей функции.)

Такой вид аппроксимации называют среднеквадратичным приближением.

Имеем задачу: Найти наилучшее приближение, т.е. такую функцию , чтобы было справедливым соотношение:

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

Наилучшее приближение, которое применительно к таблично заданным функциям называется методом наименьших квадратов.
Система линейных алгебраических уравнений:

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

Где:

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

Порядок действий:

1. Выбирается степень полинома n<<N.
2. Составляется система линейных алгебраических уравнений.
3. В результате решения СЛАУ находятся коэффициенты полинома a_k

Программа:

Главная функция main:

```
def main():
    x, y, ro = read_file()
    n = int(input("Введите степень: "))

    coefficients = root_mean_square(x, y, ro, n)
    x_list, y_list = create_list_points(x[0], x[-1], coefficients)
    show_graph(x, y, x_list, y_list)
```

Считывание из файла:

```python
def read_file():
    f = open(FILE_NAME)

    x, y, ro = list(), list(), list()

    for line in f:
        line = line.split(" ")
        x.append(float(line[0]))
        y.append(float(line[1]))
        ro.append(float(line[2]))

    f.close()
    return np.array(x), np.array(y), np.array(ro)
```

Функция, которая находит коэффициенты полинома a_k

```python
def root_mean_square(list_x, list_y, list_ro, n):
    # Матрица коэффициентов (Т.е. (x^k, x^m))
    matrix = zeros((n + 1, n + 1))
    # Массив значений (То что стоит после равно, т.е. (y, x^k))
    list_value = zeros(n + 1)

    for i in range(n + 1):
        for j in range(i, n + 1):
            matrix[i][j] = matrix[j][i] = sum(list_ro * list_x**(i + j))

    for i in range(n + 1):
        list_value[i] = sum(list_ro * list_y * list_x**i)

    # Возвращаем коэффициенты полинома (ak).
    return linalg.solve(matrix, list_value)
```

Функция, которая получает начальное и конечное значение x (они известны из файла) и коэффициенты полинома. После чего возвращает два списка (для построения кривой список с x (x_start, x_start + STEP, … , x_end) и список со значениями).

```python
def create_list_points(x_start, x_end, coefficients):
    x = arange(x_start, x_end + 0.1, STEP)
    y = zeros(len(x))

    for i in range(len(coefficients)):
        y += coefficients[i] * x**i

    return x, y
```

Функция, которая занимается отрисовкой. Получает точки (исходные из файла (x_points, y_points) и два списка (x_list, y_list). Эти два списка и содержат наш полином.

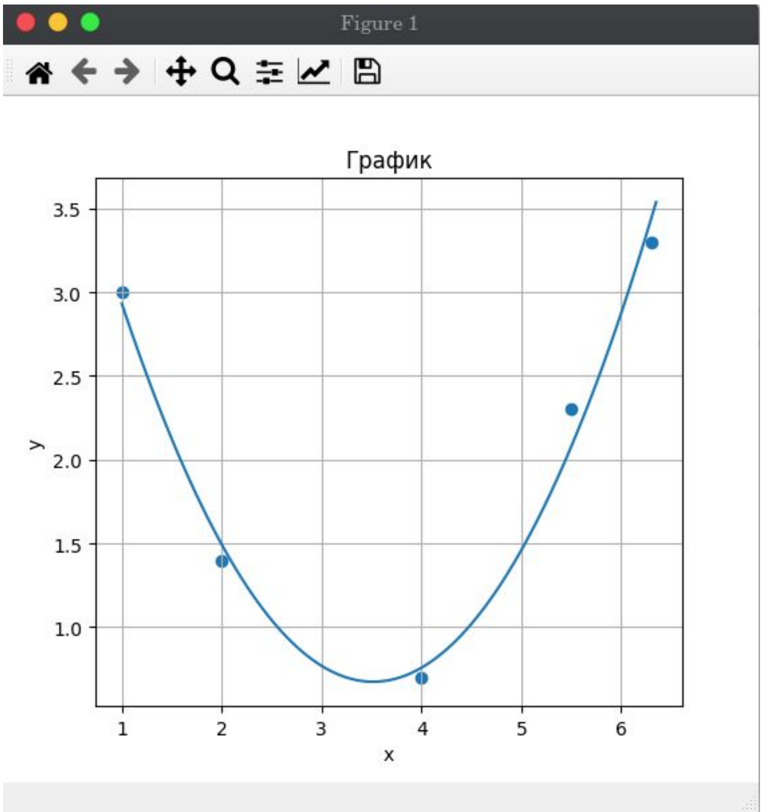
```python
def show_graph(x_points, y_points, x_list, y_list):
    # Построение графика.
    plt.title("График")  # Заголовок.
    plt.xlabel("x")      # Ось абсцисс.
    plt.ylabel("y")      # Ось ординат.
    plt.grid()           # Включение отображение сетки.

    # Отразим точки из файла.
    plt.scatter(x_points, y_points)
    plt.plot(x_list, y_list)

    plt.show()
```

Результат работы программы:

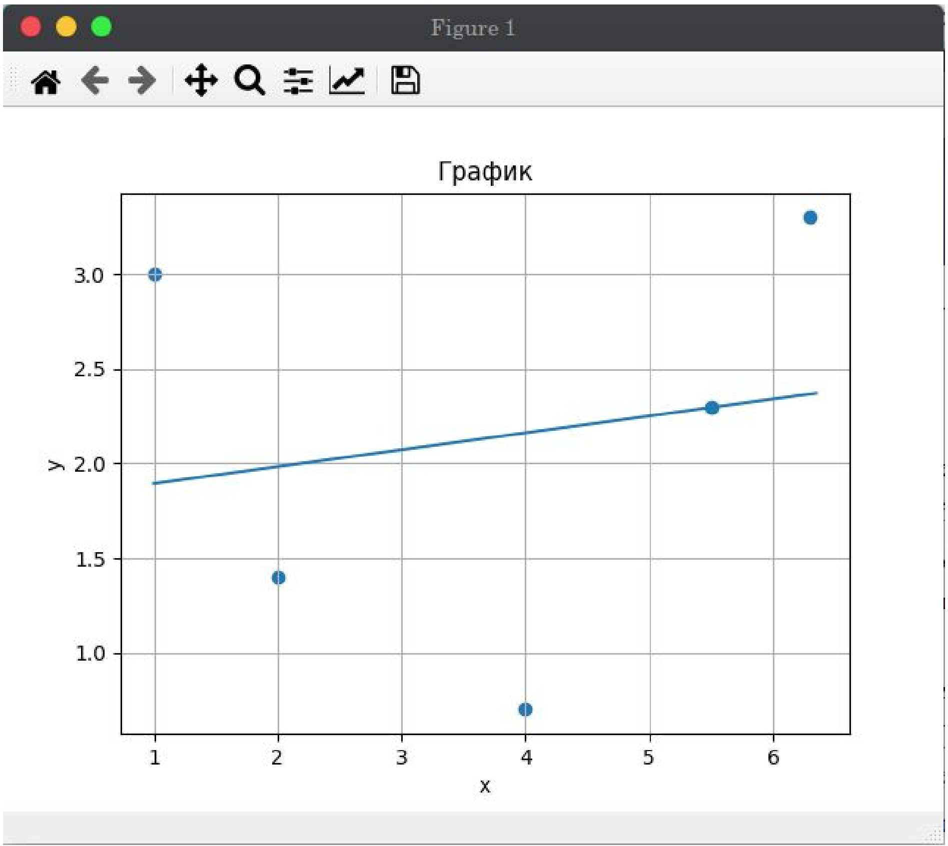
1. Таблица 1:

```
1 3 1
2 1.4 1
4 0.7 1
5.5 2.3 1
6.3 3.3 1
```

Результат работы (n=2):



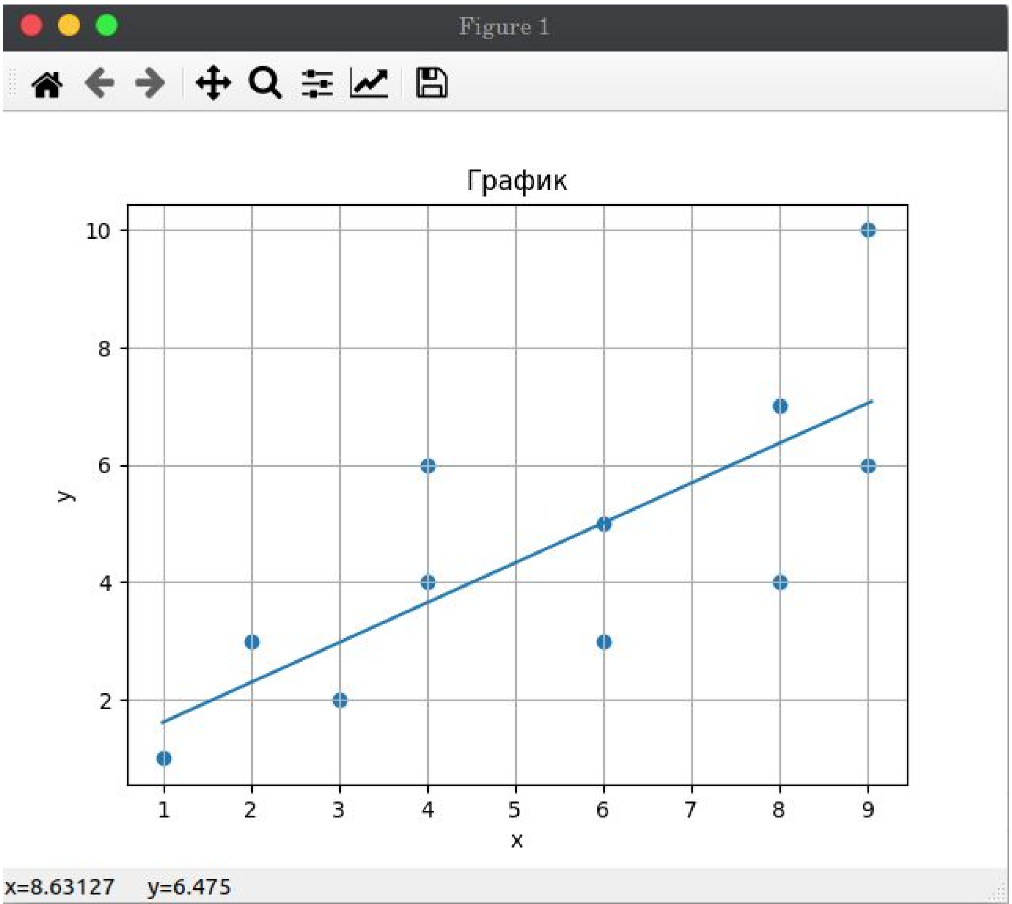
Результат работы (n=1):



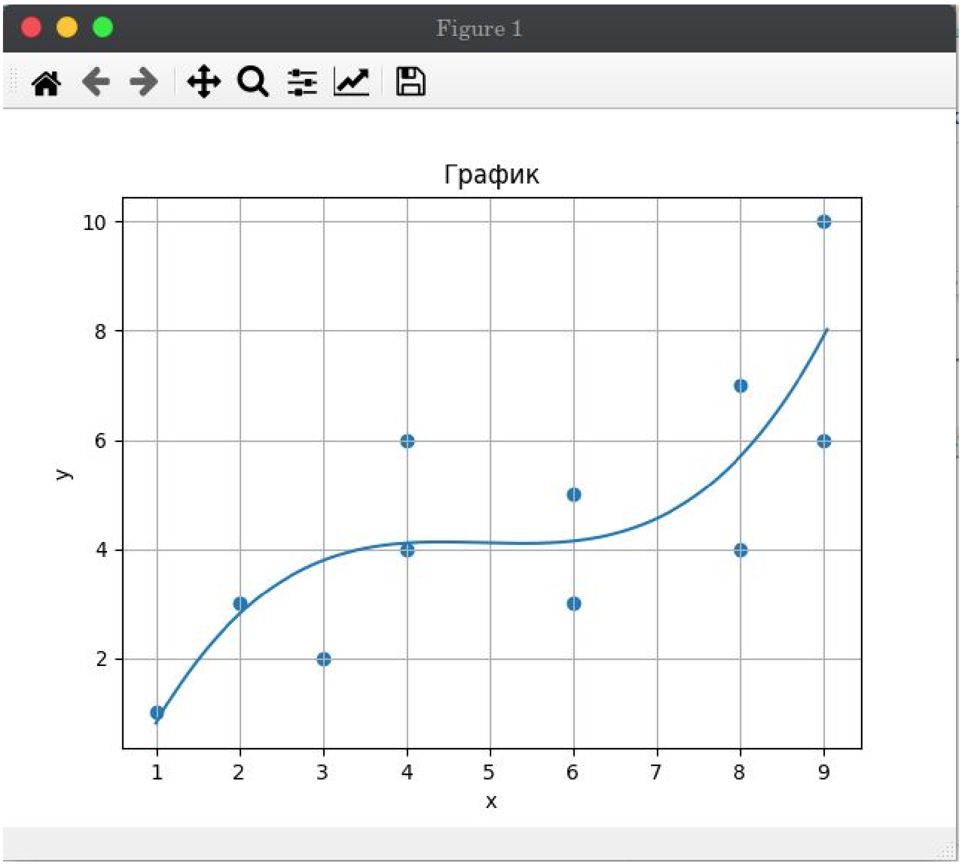
2. Таблица 2:

```
1 1 1
2 3 1
3 2 1
4 4 1
4 6 1
6 3 1
6 5 1
8 4 1
8 7 1
9 6 1
9 10 1
```

Результат работы (n=1):



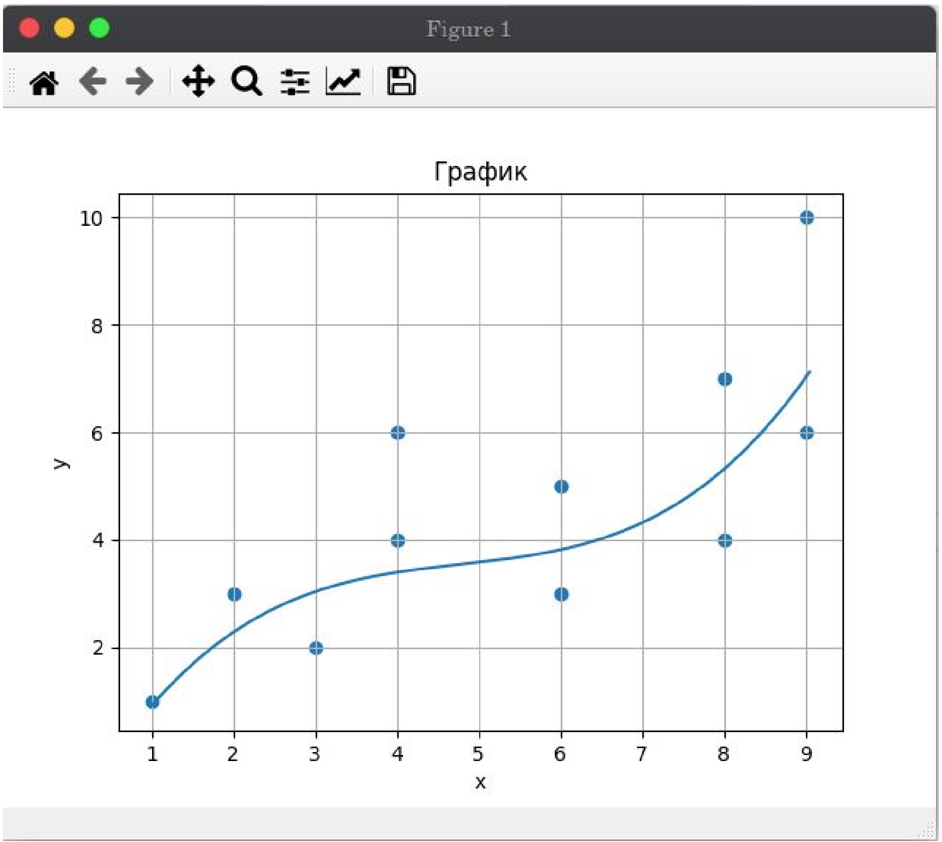
Результат работы (n=3):



Теперь в данной таблице изменим вес точек (Увеличим вес точек (3 2), (6 3), (9 6)) Таблица:

```
1 1 1
2 3 1
3 2 3
4 4 1
4 6 1
6 3 3
6 5 1
8 4 1
8 7 1
9 6 3
9 10 1
```

Результат работы (n=3):



Мы наблюдаем, что кривая стала ближе к точкам, значение весов которых мы увеличили.

Ответы на вопросы:

1. Кривая пройдет по всем точкам независимо от весов.
2. По N точкам нельзя построить полином n-ой степени (Т.к. определитель равен нулю (см. ответ 4). Программа будет работать из-за погрешности вычислений чисел с плавающей точкой.
3. n = 0 - прямая, параллельная OX. Смысл коэффициента - мат. ожидание. Формула: $\frac{\sum_{i=1}^{N} y_i \rho_i}{\sum_{i=1}^{N} \rho_i}$
4.

Допустим дана таблица:

x0 y0 1

x1 y1 1

Тогда при n=2

$\varphi(x) = a_0 + a_1 x + a_2 x^2$

И система уравнений имеет вид:

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

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

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

Скалярные произведения в полученной системе записываются следующим образом (С учетом, что $\rho = 1$):

$(x^t,\ x^t)\ =\ \sum_{i=1}^{N} \rho_i x_i^t = \sum_{i=1}^{N} x_i^t$ (При t=0 $(x^0,\ x^0) = \sum_{i=1}^{N} \rho_i$)

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

$(x^0,\ x^2)\ =\ (x^2,\ x^0) = \sum_{i=1}^{N} \rho_i\, x_i^2 = \sum_{i=1}^{N} x_i^2$

$\mathrm{E}(x^1,\ x^2)\ =\ (x^2,\ x^1) = \sum_{i=1}^{N} \rho_i\, x_i^3 = \sum_{i=1}^{N} x_i^3$

Матрица:

2	$x_0 + x_1$	$x_0^2 + x_1^2$
$x_0 + x_1$	$x_0^2 + x_1^2$	$x_0^3 + x_1^3$
$x_0^2 + x_1^2$	$x_0^3 + x_1^3$	$x_0^4 + x_1^4$

Имеет определитель:

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

$$\equiv$$

$$2 \times \left(x_0^2 + x_1^2\right) \times \left(x_0^4 + x_1^4\right) + \left(x_0 + x_1\right) \times \left(x_0^3 + x_1^3\right) \times \left(x_0^2 + x_1^2\right) +$$
$$\left(x_0^2 + x_1^2\right) \times \left(x_0 + x_1\right) \times \left(x_0^3 + x_1^3\right) - \left(x_0^2 + x_1^2\right) \times \left(x_0^2 + x_1^2\right) \times \left(x_0^2 + x_1^2\right) -$$
$$\left(x_0^3 + x_1^3\right) \times \left(x_0^3 + x_1^3\right) \times 2 - \left(x_0^4 + x_1^4\right) \times \left(x_0 + x_1\right) \times \left(x_0 + x_1\right)$$
$$= 0$$

Определитель равен нулю и система не имеет решений.