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 (ИУ-7)»

Лабораторная работа № 4

Тема: Построение и программная реализация алгоритма наилучшего среднеквадратичного приближения.

Студент Сироткина П.Ю.

Группа ИУ7-46Б

Оценка (баллы)

Преподаватель Градов В.М.

Москва, 2021 год

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. Сформировать таблицу самостоятельно со случайным разбросом точек.
2. Степень аппроксимирующего полинома - n.

Результат: аппроксимирующий полином. Графическая иллюстрация: полином и сами точки.

Описание алгоритма:

Алгоритм наилучшего среднеквадратичного приближения целесообразно применять в тех случаях, когда значения функции в узлах определены неточно, или, когда требуется построить аппроксимирующую функцию для всего диапазона задания таблицы.

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

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

P_i - вес точки.

Суммирование выполняется по всем N узлам заданной функции.

Необходимо найти наилучшее приближение, т.е.:

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

Разложим **φ** на линейно независимые функции:

$$\varphi(x) = \sum_{k=0}^{N} a_k \varphi_k(x) \tag{3}$$

Определим скалярное произведение 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:

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

Подставим (3) в (2):

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

Дифференцируя это выражение по k a и приравнивая производные нулю, найдем:

$$\sum_{m=0}^{n} (\varphi_k, \varphi_m) a_m = (y, \varphi_k), \quad 0 \le k \le n \ . \tag{4}$$

Матрица данной системы представляет собой матрицу Грама. Определитель этой матрицы в силу линейной независимости функций **φ_k(x)** не равен нулю.

Следовательно, из системы (4) можно найти коэффициенты a_k, определяющие функцию **φ_k** согласно (3) и минимизирующие (1). Таким образом, наилучшее среднеквадратичное приближение существует, и оно единственно.

В качестве **φ_k(x)** чаще всего используют полиномы Лежандра, Чебышева, Лагерра, Эрмита, ортогональные с заданным весом.

Наиболее употребительный вариант метода наименьших квадратов соответствует случаю степенного вида функций **φ_k(x)**. Обычно степень полинома не превышает пяти-шести. Система уравнений (4) при этом принимает вид:

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

где $(x^k, x^m) = \sum_{i=1}^{N} \rho_i x_i^{k+m}, \quad (y, x^k) = \sum_{i=1}^{N} \rho_i y_i x_i^k$. (5)

Итоговой алгоритм:

1. Выбирается степень полинома n<<N. Обычно степень полинома не превышает 5-6.

2. Составляется система линейных алгебраических уравнений типа (5).

3. В результате решения СЛАУ находятся коэффициенты полинома a_k.

Код программы:

Решение СЛАУ для нахождения коэффициентов полинома:

```python
def solve_slae(p, x, y, degree):

    slae = make_slae(p, x, y, degree)

    for i in range(degree + 1):
        for j in range(degree + 1):
            if i == j:
                continue
            mul = slae[j][i] / slae[i][i]
            for k in range(degree + 2):
                slae[j][k] -= mul * slae[i][k]

    for i in range(degree + 1):
        div = slae[i][i]
        for j in range(degree + 2):
            slae[i][j] /= div

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

Подфункция составления СЛАУ по имеющимся данным:

```python
def make_slae(p, x, y, degree):

    slae = [0] * (degree + 1)
    for i in range(degree + 1):
        slae[i] = [0] * (degree + 2)

    for i in range(degree + 1):
        for j in range(degree + 1):
            slae[i][j] = get_coeff(p, x, x, i, j)
        slae[i][degree + 1] = get_coeff(p, y, x, 1, i)

    return slae
```

Подфункция вычисления так называемого скалярного произведения:

```python
def get_coeff(p, a, b, degree_a, degree_b):
    coeff = 0

    for i in range(len(p)):
        coeff += p[i] * (a[i] ** degree_a) * (b[i] ** degree_b)

    return coeff
```

Примечание:

В отчете приведены только ключевые функции. Организация меню, ввода и редактирования данных, рисовка графиков опущены, т.к. мешают описанию и понимаю главной идеи алгоритма в отчете и нагромождают его.

Полный код программы можно посмотреть в приложении к отчету.

Результат работы программы:

Таблица с одинаковыми весами:

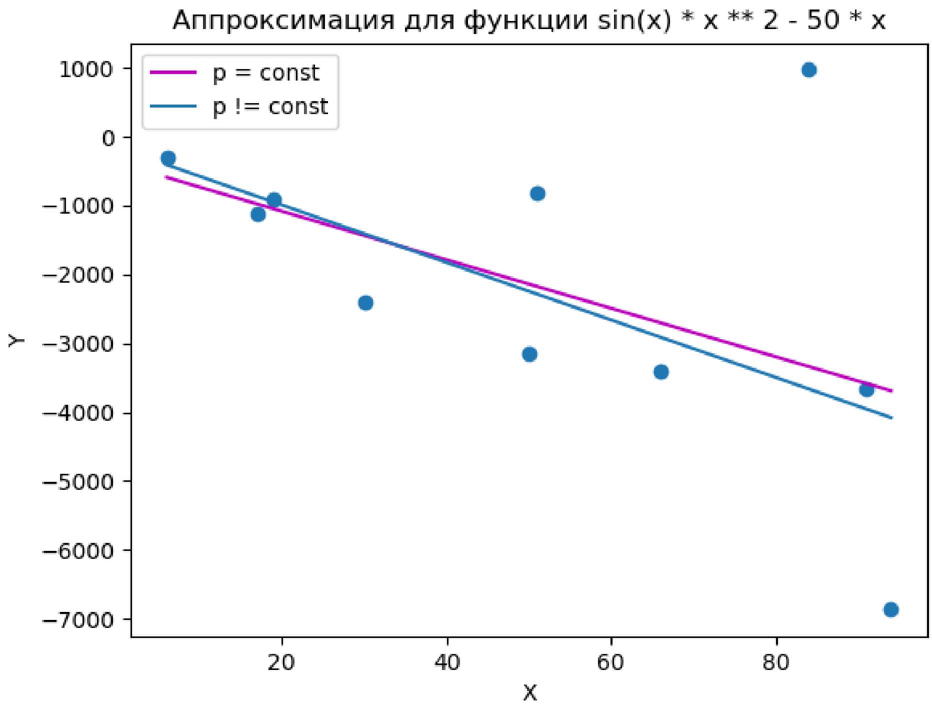
№ узла	X	Y	Вес
1	6.00	-310.06	1.00
2	17.00	-1127.84	1.00
3	19.00	-895.89	1.00
4	30.00	-2389.23	1.00
5	50.00	-3155.94	1.00
6	51.00	-806.73	1.00
7	66.00	-3415.66	1.00
8	84.00	973.39	1.00
9	91.00	-3672.32	1.00
10	94.00	-6867.05	1.00

Аналогичная таблица, в которой веса выбраны случайным образом:

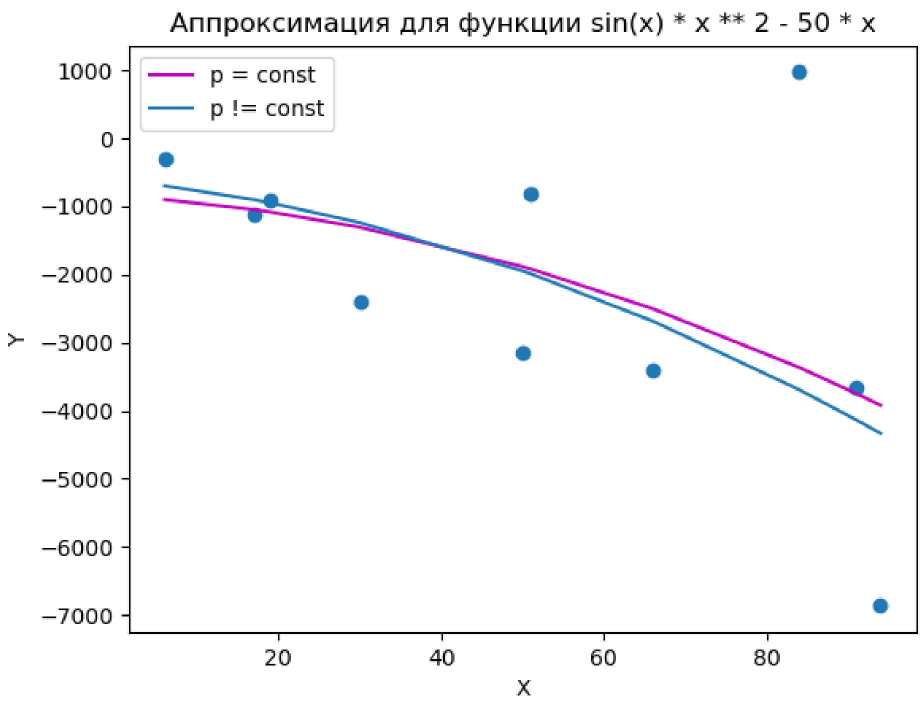
№ узла	X	Y	Вес
1	6.00	-310.06	0.85
2	17.00	-1127.84	0.67
3	19.00	-895.89	0.43
4	30.00	-2389.23	0.11
5	50.00	-3155.94	0.89
6	51.00	-806.73	0.46
7	66.00	-3415.66	0.83
8	84.00	973.39	0.62
9	91.00	-3672.32	0.58
10	94.00	-6867.05	0.84

Проведем сравнение в соответствии с приведенными выше данными:

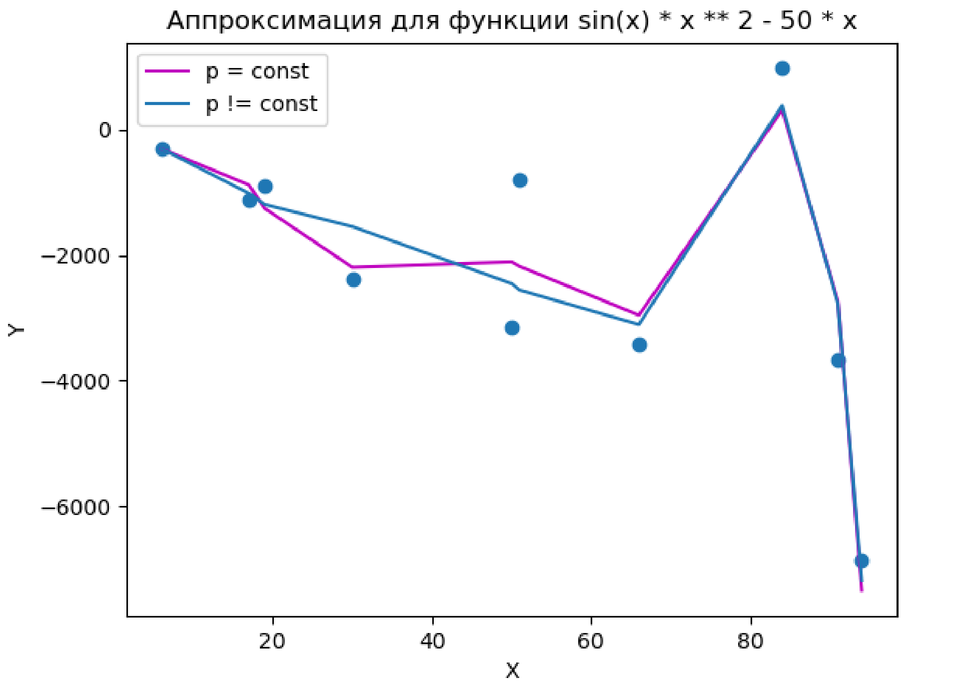
N = 1:


Аппроксимация для функции sin(x) * x ** 2 - 50 * x
p = const
p != const
1000
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
Y
20
40
60
80
X

N = 2:


Аппроксимация для функции sin(x) * x ** 2 - 50 * x
p = const
p != const
1000
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
Y
20
40
60
80
X

N = 6:



Ответы на контрольные вопросы:

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

N точками можно однозначно определить полином N – 1 степени. Это означает, что следующее выражение будет тождественно равно 0:

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

Это значит, что никак не используется информация о весах точек. Следовательно, независимо от веса, полином будет 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?

Программа будет работать, но некорректно. Это связано с тем, что при n>= N уравнения СЛАУ перестают быть линейно зависимыми и определитель матрицы становится тождественно равен нулю. К аварийной остановке может привести деление на ноль (в своей программе я решала

СЛАУ методом Гаусса-Жордана, ошибка может произойти при приведении диагональной матрицы к единичной). Лучше всего проводить анализ во время ввода степени полинома и количества узлов в таблице, хотя так же можно обрабатывать эту ситуацию и во время решения СЛАУ.

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

$$a_0 = \frac{\sum_{i=1}^{N} \rho_i y_i}{\sum_{i=1}^{N} \rho_i}$$

Разделив числитель и знаменатель на сумму весов по всем узлам, то получим математическое ожидание:

$$M[X] = \sum_{i=1}^{\infty} x_i \rho_i$$

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

Система примет следующий вид:

$$\begin{aligned} a_0 + (x_0 + x_1) \cdot a1 + (x_0^2 + x_1^2) \cdot a_2 &= y_0 + y_1 \\ (x_0 + x_1) \cdot a_0 + (x_0^2 + x_1^2) \cdot a1 + (x_0^3 + x_1^3) \cdot a_2 &= y_0 \cdot x_0 + y_1 \cdot x_1 \\ (x_0^2 + x_1^2) \cdot a_0 + (x_0^3 + x_1^3) \cdot a1 + (x_0^4 + x_1^4) \cdot a_2 &= y_0 \cdot x_0^2 + y_1 \cdot x_1^2 \end{aligned}$$

Определитель этой системы равен 0:

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

Система не будет иметь решений, как это было описано ранее при ответе на вопрос 2.

5. Построить СЛАУ при выборочном задании степеней аргумента полинома* φ = a0 + a1 * x ^ m + a2 * x ^ n, *причем степени n и m в этой формуле известны.

$$\phi = a_0 + a_1 \cdot x^m + a_2 \cdot x^n$$

Получаем : $\phi_0 = 1, \phi_1 = x^m, \phi_2 = x^n$

По определению, СЛАУ запишется следующим образом :

$$a_0 \cdot (\phi_0, \phi_0) + a_1 \cdot (\phi_0, \phi_1) + a2 \cdot (\phi_0, \phi_2) = (y, \phi_0)$$
$$a_0 \cdot (\phi_1, \phi_0) + a_1 \cdot (\phi_1, \phi_1) + a2 \cdot (\phi_1, \phi_2) = (y, \phi_1)$$
$$a_0 \cdot (\phi_2, \phi_0) + a_1 \cdot (\phi_2, \phi_1) + a2 \cdot (\phi_2, \phi_2) = (y, \phi_2)$$

Подставим значения :

$$a_0 \cdot (1, 1) + a_1 \cdot (1, x^m) + a_2 \cdot (1, x^n) = (y, 1)$$
$$a_0 \cdot (x^m, 1) + a_1 \cdot (x^m, x^m) + a_2 \cdot (x^m, x^n) = (y, x^m)$$
$$a_0 \cdot (x^n, 1) + a_1 \cdot (x^n, x^m) + a_2 \cdot (x^n, x^n) = (y, x^n)$$

Раскроем по формуле введенного понятия скалярного произведения:

$$a_0 \cdot \sum_{i=1}^{N} p_i + a_1 \cdot \sum_{i=1}^{N} p_i \cdot x^m + a_1 \cdot \sum_{i=1}^{N} p_i \cdot x^n = \sum_{i=1}^{N} p_i \cdot y_i$$
$$a_0 \cdot \sum_{i=1}^{N} p_i \cdot x^m + a_1 \cdot \sum_{i=1}^{N} p_i \cdot x^{2 \cdot m} + a_1 \cdot \sum_{i=1}^{N} p_i \cdot x^{n+m} = \sum_{i=1}^{N} p_i \cdot y_i \cdot x^m$$
$$a_0 \cdot \sum_{i=1}^{N} p_i \cdot x^n + a_1 \cdot \sum_{i=1}^{N} p_i \cdot x^{m+n} + a_1 \cdot \sum_{i=1}^{N} p_i \cdot x^{2 \cdot n} = \sum_{i=1}^{N} p_i \cdot y_i \cdot x^n$$

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.

Можно предположить, что значения полинома на концах исследуемого отрезка совпадают с табличной функцией или равны какому-то фиксированному числу, таким образом получится 3 уравнения из СЛАУ, как это описано выше, и эти 2 условия. Итого имеем 5 условий для определения 5 неизвестных.