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
по курсу «Вычислительные алгоритмы»

Тема Построение и программная реализация алгоритма наилучшего среднеквадратичного приближения

Студент Колосов Д.В.

Группа ИУ7-42Б

Оценка (баллы) _______________

Преподаватель Градов В.М.

Москва.
2020 г.

Содержание

1 Цель работы

Получение навыков построения алгоритма метода наименьших квадратов с использованием полинома заданной степени при аппроксимации табличных функций с весами.

2 Исходные данные данные

1. Таблица функции с весами с количеством узлов N.

X	Y	ρ_i

Предусмотреть в интерфейсе удобную возможность изменения пользователем весов в таблице.

2. Степень аппроксимирующего полинома - n.

3 Результат работы программы

Графики, построенные по аналогии с рис.1 в тексте Лекции №4: точки - заданная табличная функция, кривые- найденные полиномы.

Обязательно приводить таблицы, по которым работала программа.

При каких исходных условиях надо представить результаты в отчете?

1. Веса всех точек одинаковы и равны, например, единице. Обязательно построить полиномы при значениях его степени n=1, 2. Можно привести результаты и при других степенях полинома, однако не загромождая сильно при этом рисунок.

2. Веса точек разные. Продемонстрировать, как за счет назначения весов точкам можно изменить положение на плоскости прямой линии (полином первой степени), аппроксимирующей один и тот же набор точек (одну таблицу y(x)). Например, назначая веса узлам в таблице изменить знак углового коэффициента прямой. На графике в итоге должны быть представлены точки исходной функции и две аппроксимирующие их прямые линии. Одна отвечает значениям =1 для всех узлов, а другая- назначенным разным весам точек. Информацию о том, какие именно веса были использованы в расчете обязательно указать, чтобы можно было проконтролировать работу программы (лучше это сделать в виде таблицы).

4 Описание алгоритма

Цель алгоритма - найти наилучшее приближение, т.е. такую функцию phi, чтобы было справедливым соотношение

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

Разложим функцию $\phi(x)$ по системе линейно независимых функций $\phi_k(x)$

$$\phi(x) = \sum_{k=0}^{n} a_k \phi_k(x)$$

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

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

Примем во внимание следующие свойства, присущие скалярному произведению элементов линейного пространства

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

С учётом всех вышенаписанных формул, получим:

$$((y - \phi), (y - \phi)) = (y, y) - 2 \sum_{k=0}^{n} a_k (y, \phi_k) + \sum_{k=0}^{n} \sum_{m=0}^{n} a_k a_m (\phi_k, \phi_m) = min$$

Дифференцируя это выражение по a_k и приравнивая производные к нулю, найдём

$$\sum_{m=0}^{n} (\phi_k, \phi_m) a_m = (y, \phi_k), 0 <= k <= n$$

Определитель этой системы в силу линейной независимости функций не ра-вен нулю. Следовательно, из системы можно найти коэффициенты, определяющие функцию. Таким образом, наилучшее среднеквадратичное приближение существует и оно единственно.

Наиболее употребительный вариант метода наименьших квадратов соответствует случаю степенного вида функций $\phi_k(x)$, т.е. $\phi_k(x) = x^k$, причем $0 <= k <= n$ Обычно в сумме берут не более пяти-шести членов. Система уравнений при этом принимает вид

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

Где

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

Итого:

Для применения метода наименьших квадратов в случае аппроксимации полиномом следует действовать следующим образом.

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

2. Составляется система линейных алгебраических уравнений.

3. В результате решения СЛАУ находятся коэффициенты полинома a_k

В качестве исходных данных используется произвольная табличная функция, для каждого узла i которой пользователь задает вес ρ_i по своему усмотрению.

5 Код программы

5.1 Основной модуль - получение коэффициентов a

```
std::vector <double> x;
std::vector <double> y;
std::vector <double> rho;
int n = ui->nDegree->text().toInt() + 1;  // Because amount of a = degree + 1
int N = ui->tableWidget->rowCount();

for (int i = 0; i < N; i++) {
    x.push_back(dynamic_cast<QLineEdit *>
               (ui->tableWidget->cellWidget(i,0))->text().toDouble());

    y.push_back(dynamic_cast<QLineEdit *>
               (ui->tableWidget->cellWidget(i,1))->text().toDouble());

    rho.push_back(dynamic_cast<QLineEdit *>
               (ui->tableWidget->cellWidget(i,2))->text().toDouble());
}

std::vector <std::vector <double>> matrix(n);
for (int i = 0; i < n; i++) {
    matrix[i] = std::vector <double> (n + 1); // matrix[i][n] == (y, x ^ k)
    for (int j = 0 ; j < n; j++) {
        double sigma = 0;
        for (int k = 0; k < N; k++) {
            sigma += pow(x[k], i + j) * rho[k];
        }
        matrix[i][j] = sigma;
    }
    double sigmaY = 0;
    for (int k = 0; k < N; k++) {
        sigmaY += pow(x[k], i) * y[k] * rho[k];
    }
    matrix[i][n] = sigmaY;
}

for (int iter = 0; iter < n; iter++) {
    matrix_max_first(matrix, n, iter);
    matrix_normalize_rows(matrix, n, iter);
}
```

```
std::vector <double> a(n);
matrix_get_solutions(matrix, n, a);
```

5.2 Вспомогательный модуль для решения СЛАУ методом Гаусса

```
void matrix_max_first(std::vector <std::vector <double>> &matrix,
                      int x_vars, int iter) {
    int mx = iter;
    for (int i = iter; i < x_vars; i++) {
        if (matrix[i][iter] > matrix[mx][iter])
            mx = i;
    }
    auto tmp = matrix[iter];
    matrix[iter] = matrix[mx];
    matrix[mx] = tmp;
}


void matrix_normalize_rows(std::vector <std::vector <double>> &matrix,
                           int x_vars, int iter) {
    for (int i = iter; i < x_vars; i++) {
        double normalize = matrix[i][iter];
        for (int j = iter; j < x_vars + 1; j++) {
            matrix[i][j] /= normalize;
        }
    }

    for (int i = iter + 1; i < x_vars; i++) {
        for (int j = iter; j < x_vars + 1; j++) {
            matrix[i][j] -= matrix[iter][j];
        }
    }
}


void matrix_get_solutions(std::vector <std::vector <double>> &matrix,
                          int x_vars, std::vector <double> &x) {
    for (int i = x_vars - 1; i >= 0; i--) {
        double sigma = 0;
        for (int j = x_vars - 1; j > i; j--) {
            sigma += matrix[i][j] * x[j];
        }
        x[i] = (matrix[i][x_vars] - sigma) / matrix[i][i];
    }
}
```

6 Результат работы и анализ

6.1 Веса всех точек одинаковы и равны единице

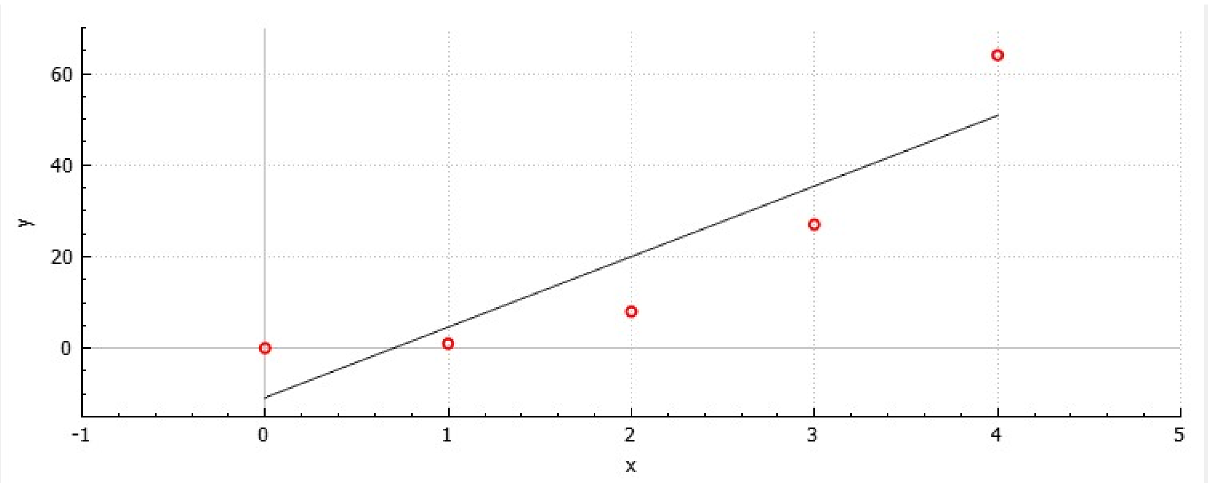
Зададим функцию $y = (x)^3$ таблицей:

X	Y	ρ_i
0	0	1
1	1	1
2	8	1
3	27	1
4	64	1

	X	Y	Rho
1	0	0	1
2	1	1	1
3	2	8	1
4	3	27	1
5	4	64	1

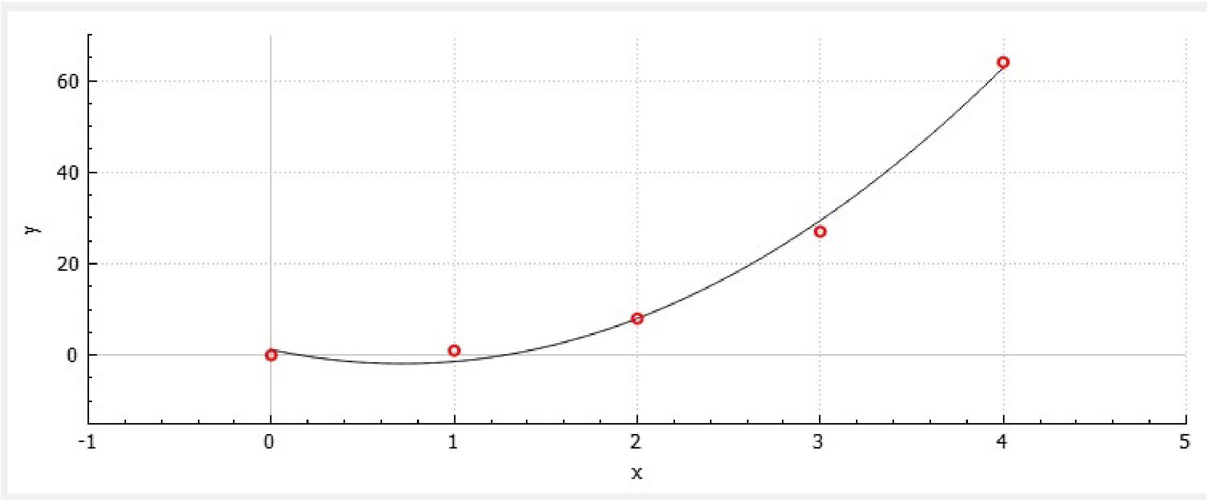
Степень полинома $n = 1$

Наблюдаем следующий результат:

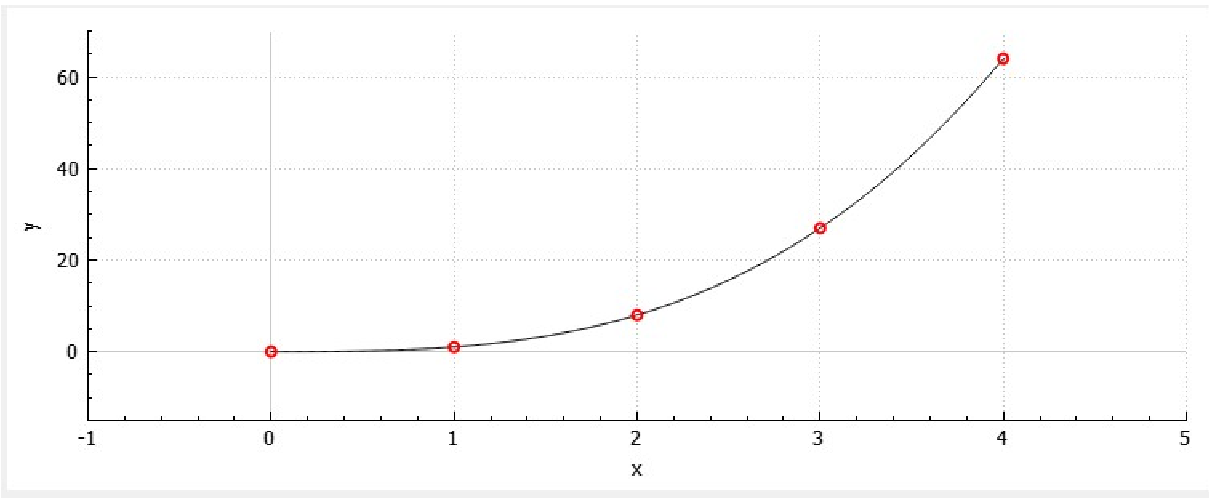


Теперь построим полином со степенью $n = 2$.

Получим следующий график:



Видно, что в интервале (0, 1) график лежит ниже оси OX, что не соответствует идеальному графику.
Построим график со степенью полинома $n = 3$



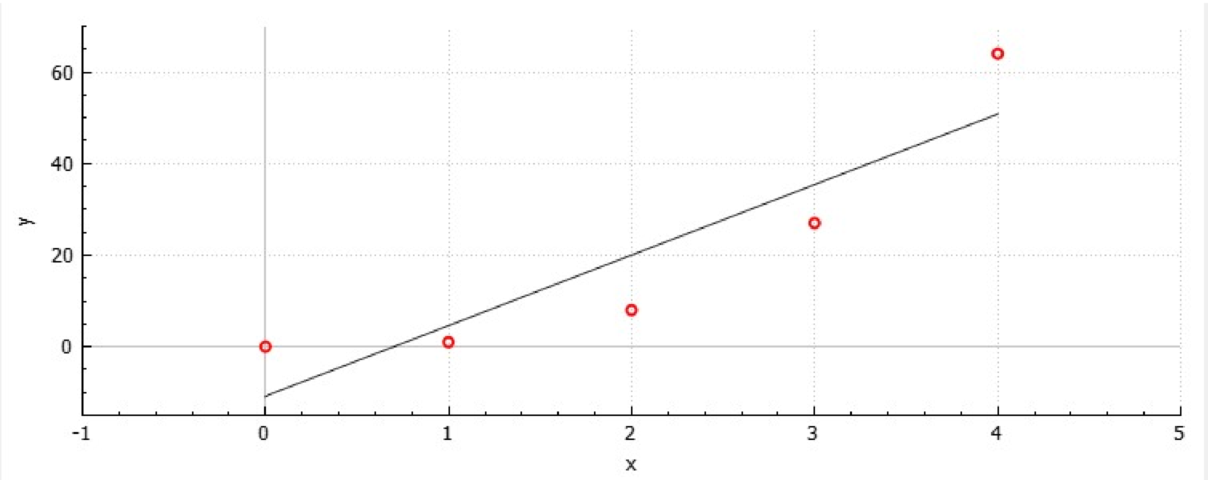
Практически идеальный график

6.2 Веса точек разные

	X	Y	Rho
1	0	0	1
2	1	1	1
3	2	8	1
4	3	27	1
5	4	64	1

Степень полинома $n = 1$

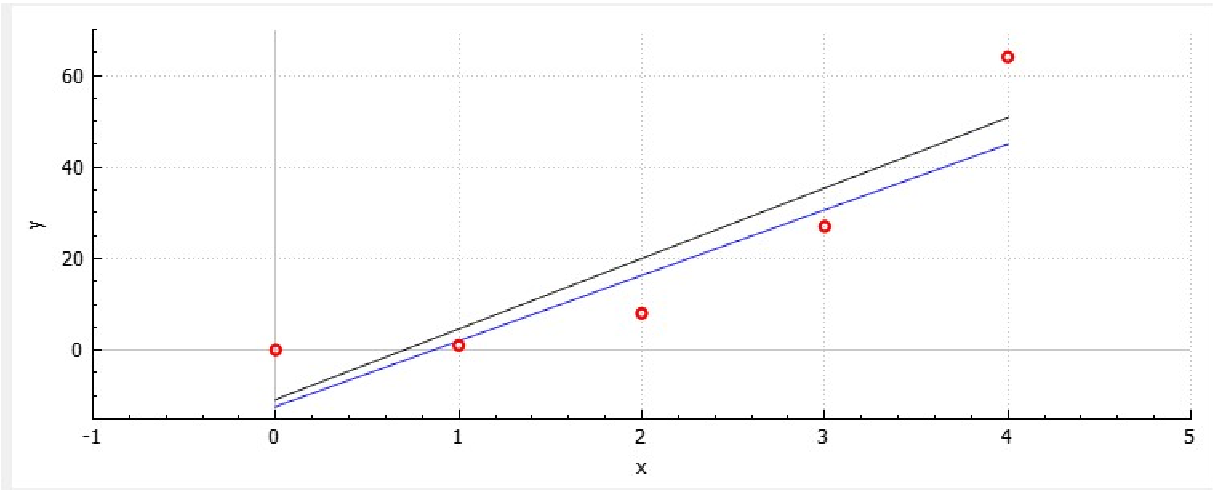
Наблюдаем следующий результат:



Теперь изменим значения весовых коэффициентов, благо в моей программе это можно легко сделать прямо в таблице:

	X	Y	Rho
1	0	0	1
2	1	1	5
3	2	8	1
4	3	27	5
5	4	64	1

Получаем два разных графика. Можно заметить, что график действительно сместился к тем точкам, у которых весовой коэффициент больше:



7 Контрольные вопросы

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

Количество коэффициентов а в аппроксимирующем полиноме будет равно количеству узлов в таблице. Таким образом график аппрокисмирующего полинома будет проходить через все узлы таблицы вне зависимости от их весов ρ_i

7.2 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 и выводить корректный результат несмотря на то, что по N точкам невозможно построить полином степени n ввиду равенста определителя нулю. Связано это с тем, что вещественная арифметика компьютера имеет погрешность, вследствие чего данные расчёты не являются идеальными. Также при слишком больших степенях может произойти переполнение при расчётах x^m, что в свою очередь может привести к аварийной ситуации.

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

$$\text{Исходная система } \sum_{m=0}^{n}(x^k, x^m)a_m = (y, x^k), 0 <= k <= n$$
$$(x^0, x^0)a_0 = (y, x^0)$$
$$\sum_{i=1}^{N}\rho_i x_i^0 a_0 = \sum_{i=1}^{N}\rho_i y_i x_i^0$$
$$\sum_{i=1}^{N}\rho_i a_0 = \sum_{i=1}^{N} y_i \rho_i$$
$$(\rho_1 + \rho_2 + ... + \rho_N)a_0 = (\rho_1 y_1 + \rho_2 y_2 + ... + \rho_N y_N)$$
$$a_0 = \frac{(\rho_1 y_1 + \rho_2 y_2 + ... + \rho_N y_N)}{(\rho_1 + \rho_2 + ... + \rho_N)}$$

Смысл - Математическое ожидание.

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

Задана таблица

X_i	Y_i	ρ_i
x_1	y_1	1
x_2	y_2	1

Составим СЛАУ:

$$\begin{cases} \sum\limits_{i=1}^{N}(\rho_i)a_0 + \sum\limits_{i=1}^{N}(x_i\rho_i)a_1 + \sum\limits_{i=1}^{N}(x_i^2\rho_i)a_2 = \sum\limits_{i=1}^{N}(y_i\rho_i) \\ \sum\limits_{i=1}^{N}(x_i\rho_i)a_0 + \sum\limits_{i=1}^{N}(x_i^2\rho_i)a_1 + \sum\limits_{i=1}^{N}(x_i^3\rho_i)a_2 = \sum\limits_{i=1}^{N}(y_ix_i\rho_i) \\ \sum\limits_{i=1}^{N}(x_i^2\rho_i)a_0 + \sum\limits_{i=1}^{N}(x_i^3\rho_i)a_1 + \sum\limits_{i=1}^{N}(x_i^4\rho_i)a_2 = \sum\limits_{i=1}^{N}(y_ix_i^2\rho_i) \end{cases}$$

Вычислим определитель:

$$\Delta = \sum_{i=1}^{N}(\rho_i) * \sum_{i=1}^{N}(x_i^2\rho_i) * \sum_{i=1}^{N}(x_i^4\rho_i) - \sum_{i=1}^{N}(\rho_i) * \sum_{i=1}^{N}(x_i^3\rho_i) * \sum_{i=1}^{N}(x_i^3\rho_i) -$$
$$\sum_{i=1}^{N}(x_i\rho_i) * \sum_{i=1}^{N}(x_i\rho_i) * \sum_{i=1}^{N}(x_i^4\rho_i) + \sum_{i=1}^{N}(x_i\rho_i) * \sum_{i=1}^{N}(x_i^3\rho_i) * \sum_{i=1}^{N}(x_i^2\rho_i) +$$
$$\sum_{i=1}^{N}(x_i^2\rho_i) * \sum_{i=1}^{N}(x_i\rho_i) * \sum_{i=1}^{N}(x_i^3\rho_i) - \sum_{i=1}^{N}(x_i^2\rho_i) * \sum_{i=1}^{N}(x_i^2\rho_i) * \sum_{i=1}^{N}(x_i^2\rho_i) = 0$$

Т.к. $\Delta = 0$, то система является несовместной и решений нет.

7.5 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.

$$\begin{cases} \sum\limits_{i=1}^{N}(\rho_i)a_0 + \sum\limits_{i=1}^{N}(x_i^m\rho_i)a_1 + \sum\limits_{i=1}^{N}(x_i^n\rho_i)a_2 = \sum\limits_{i=1}^{N}(y_i\rho_i) \\ \sum\limits_{i=1}^{N}(x_i^m\rho_i)a_0 + \sum\limits_{i=1}^{N}(x_i^{2m}\rho_i)a_1 + \sum\limits_{i=1}^{N}(x_i^{m+n}\rho_i)a_2 = \sum\limits_{i=1}^{N}(y_ix_i^m\rho_i) \\ \sum\limits_{i=1}^{N}(x_i^n\rho_i)a_0 + \sum\limits_{i=1}^{N}(x_i^{m+n}\rho_i)a_1 + \sum\limits_{i=1}^{N}(x_i^{2n}\rho_i)a_2 = \sum\limits_{i=1}^{N}(y_ix_i^n\rho_i) \end{cases}$$