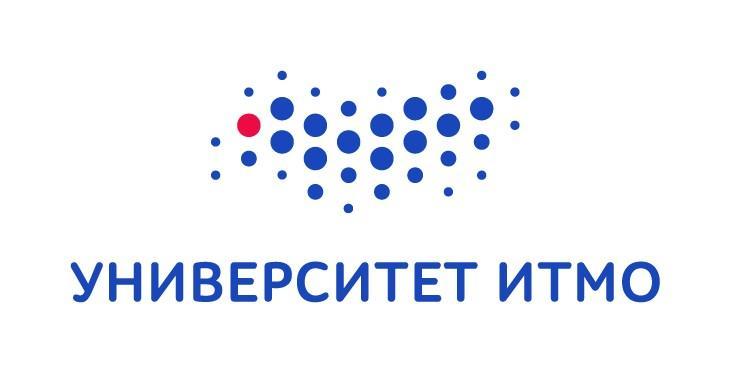
­­­Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет Программной Инженерии и Компьютерной Техники

****

**Вычислительная математика**

**Лабораторная работа № 4**

Аппроксимация функции методом наименьших квадратов

Вариант № 7

Работу выполнил: Иванов Евгений

Группа: Р3213

Преподаватель: Малышева Татьяна Алексеевна

г. Санкт-Петербург

2022

**Цель лабораторной работы**: найти функцию, являющуюся наилучшим приближением заданной табличной функции по методу наименьших квадратов.

***Вычислительная реализация:***

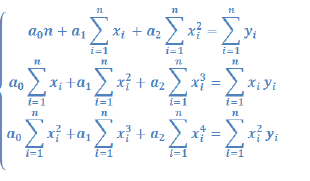
|  |  |
| --- | --- |
|  |  |

Таблица значений:

|  |  |
| --- | --- |
| X | Y |
| -2 | -0.3158 |
| -1.8 | -0.4001 |
| -1.6 | -0.5024 |
| -1.4 | -0.6139 |
| -1.2 | -0.7096 |
| -1.0 | -0.75 |
| -0.8 | -0.7039 |
| -0.6 | -0.5752 |
| -0.4 | -0.3966 |
| -0.2 | -0.1999 |
| 0 | 0 |

**Квадратичная аппроксимация**

Решим следующую систему:



Полученные результаты:

а2 = 0.580

a1 = 1.287

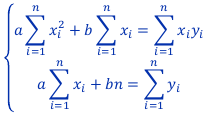
a0 = 0.005

**Полученная аппроксимация: 0.580x^2 + 1.287x + 0.005**

Мера отклонения **0,0186**

**Линейная аппроксимация**

Решим следующую систему:



Полученные результаты:

a = 0.1264

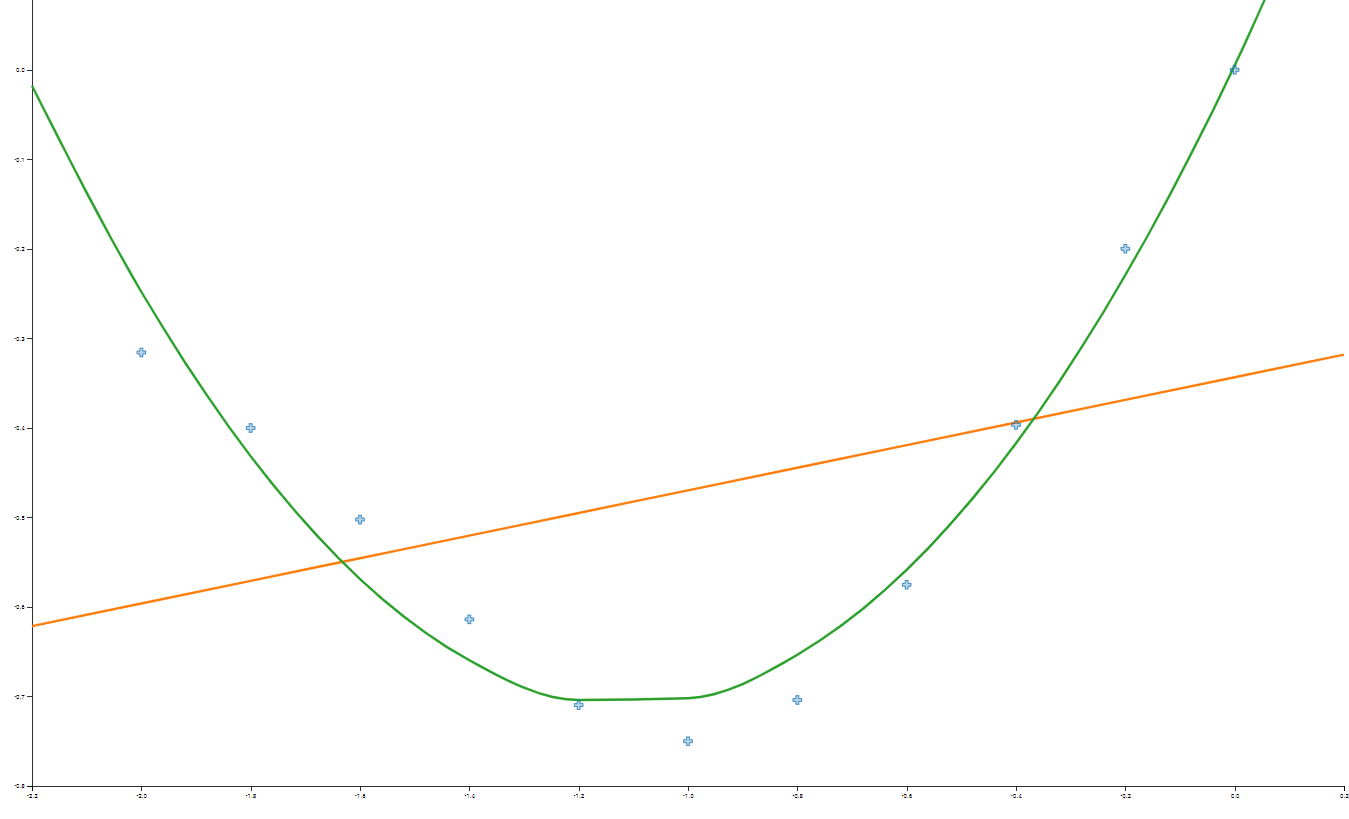
b = -0.3434

**Полученная аппроксимация: 0.1264x - 0.3434**

Мера отклонения: **0.4809**

**Коэффициент корреляции: 0.9883**

**Приведём график с точками и линейной и квадратичной аппроксимациями:**

****

Из графика видно, что квадратичная аппроксимация работает лучше на данном наборе точек, что и подтверждается мерой отклонения.

**Программная реализация:**

Метод Гаусса с расчетом матриц для определения коэффициентов при аппроксимациях:

function methodGauss(matrix) {  
 let n = matrix.size;  
 let m = matrix.matrix;//new Array(n); //Определение рабочего массива  
 let l = new ***Array***(n); //Массив ответов  
 let i, j, k; //Вспомогательные переменные  
  
 Iteration(n);  
 Answers();  
  
 function Iteration(iter\_item) { //Функция итеррация  
 for (iter\_item = 0; iter\_item < (n - 1); iter\_item++) { //Цикл выполнения итерраций  
 if (m[iter\_item][iter\_item] === 0) SwapRows(iter\_item); //Проверка на ноль  
 for (j = n; j >= iter\_item; j--) {  
 m[iter\_item][j] /= m[iter\_item][iter\_item]; //Делим строку i на а[i][i]  
 }  
 for (i = iter\_item + 1; i < n; i++) { //Выполнение операций со строками  
 for (j = n; j >= iter\_item; j--) {  
 m[i][j] -= m[iter\_item][j] \* m[i][iter\_item];  
 }  
 }  
 }  
 }  
  
 function SwapRows(iter\_item) { //Функция перемены строк  
 for (i = iter\_item + 1; i < n; i++) {  
 if (m[i][iter\_item] !== 0) {  
 for (j = 0; j <= n; j++) {  
 k = m[i - 1][j];  
 m[i - 1][j] = m[i][j];  
 m[i][j] = k;  
 }  
 }  
 }  
 }  
  
 function Answers() { //Функция поиска и вывода корней  
 l[n - 1] = m[n - 1][n] / m[n - 1][n - 1];  
 for (i = n - 2; i >= 0; i--) {  
 k = 0;  
 for (j = n - 1; j > i; j--) {  
 k = (m[i][j] \* l[j]) + k;  
 }  
 l[i] = m[i][n] - k;  
 }  
 }  
  
 return l;  
}

Линейная аппроксимация:

var ***linearAnswers*** = [];  
  
function linearAppr() {  
 let matrix = {  
 size: 2,  
 matrix: []  
 }  
  
 matrix.matrix = new ***Array***(2);  
 matrix.matrix[0] = new ***Array***(3);  
 matrix.matrix[1] = new ***Array***(3);  
  
 matrix.matrix[0][0] = ***CONSTANTS***.x2\_sum;  
 matrix.matrix[0][1] = ***CONSTANTS***.x1\_sum;  
 matrix.matrix[0][2] = ***CONSTANTS***.xy\_sum;  
 matrix.matrix[1][0] = ***CONSTANTS***.x1\_sum;  
 matrix.matrix[1][1] = ***inputPoints***.length;  
 matrix.matrix[1][2] = ***CONSTANTS***.y\_sum;  
  
 ***linearAnswers*** = methodGauss(matrix);  
  
 ***console***.log(***linearAnswers***);  
}  
  
function outputLinear() {  
 ***document***.getElementById("linear-out").value = "";  
 ***document***.getElementById("linear-out").value = ***linearAnswers***[0] + "\*x + " + ***linearAnswers***[1];  
  
 for (let i in ***inputPoints***) {  
 drawPoint(0, ***inputPoints***[i].x, ***linearAnswers***[0]\****inputPoints***[i].x+***linearAnswers***[1], "white");  
 }  
}

Квадратичная аппроксимация:

var ***quadraticAnswers*** = [];  
  
function quadraticAppr() {  
 let matrix = {  
 size: 3,  
 matrix: []  
 }  
  
 matrix.matrix = new ***Array***(3);  
 matrix.matrix[0] = new ***Array***(4);  
 matrix.matrix[1] = new ***Array***(4);  
 matrix.matrix[2] = new ***Array***(4);  
  
 matrix.matrix[0][0] = ***inputPoints***.length;  
 matrix.matrix[0][1] = ***CONSTANTS***.x1\_sum;  
 matrix.matrix[0][2] = ***CONSTANTS***.x2\_sum;  
 matrix.matrix[0][3] = ***CONSTANTS***.y\_sum;  
 matrix.matrix[1][0] = ***CONSTANTS***.x1\_sum;  
 matrix.matrix[1][1] = ***CONSTANTS***.x2\_sum;  
 matrix.matrix[1][2] = ***CONSTANTS***.x3\_sum;  
 matrix.matrix[1][3] = ***CONSTANTS***.xy\_sum;  
 matrix.matrix[2][0] = ***CONSTANTS***.x2\_sum;  
 matrix.matrix[2][1] = ***CONSTANTS***.x3\_sum;  
 matrix.matrix[2][2] = ***CONSTANTS***.x4\_sum;  
 matrix.matrix[2][3] = ***CONSTANTS***.x2y\_sum;  
  
 ***quadraticAnswers*** = methodGauss(matrix);  
  
 ***console***.log(***quadraticAnswers***);  
}  
  
function outputQuadratic() {  
 ***document***.getElementById("quadratic-out").value = "";  
 ***document***.getElementById("quadratic-out").value =  
 ***quadraticAnswers***[2] + "\*x^2 + " + ***quadraticAnswers***[1] + "\*x + " + ***quadraticAnswers***[0];  
  
 for (let i in ***inputPoints***) {  
 drawPoint(1,  
 ***inputPoints***[i].x,  
 ***quadraticAnswers***[2]\****inputPoints***[i].x\****inputPoints***[i].x+***quadraticAnswers***[1]\****inputPoints***[i].x+***quadraticAnswers***[0],  
 "white");  
 }  
}

Аппроксимация полиномом третьей степени:

var ***thirdAnswers*** = [];  
  
function thirdAppr() {  
 let matrix = {  
 size: 4,  
 matrix: []  
 }  
  
 matrix.matrix = new ***Array***(4);  
 matrix.matrix[0] = new ***Array***(5);  
 matrix.matrix[1] = new ***Array***(5);  
 matrix.matrix[2] = new ***Array***(5);  
 matrix.matrix[3] = new ***Array***(5);  
  
 matrix.matrix[0][0] = ***inputPoints***.length;  
 matrix.matrix[0][1] = ***CONSTANTS***.x1\_sum;  
 matrix.matrix[0][2] = ***CONSTANTS***.x2\_sum;  
 matrix.matrix[0][3] = ***CONSTANTS***.x3\_sum;  
 matrix.matrix[0][4] = ***CONSTANTS***.y\_sum;  
  
 matrix.matrix[1][0] = ***CONSTANTS***.x1\_sum;  
 matrix.matrix[1][1] = ***CONSTANTS***.x2\_sum;  
 matrix.matrix[1][2] = ***CONSTANTS***.x3\_sum;  
 matrix.matrix[1][3] = ***CONSTANTS***.x4\_sum;  
 matrix.matrix[1][4] = ***CONSTANTS***.xy\_sum;  
  
 matrix.matrix[2][0] = ***CONSTANTS***.x2\_sum;  
 matrix.matrix[2][1] = ***CONSTANTS***.x3\_sum;  
 matrix.matrix[2][2] = ***CONSTANTS***.x4\_sum;  
 matrix.matrix[2][3] = ***CONSTANTS***.x5\_sum;  
 matrix.matrix[2][4] = ***CONSTANTS***.x2y\_sum;  
  
 matrix.matrix[3][0] = ***CONSTANTS***.x3\_sum;  
 matrix.matrix[3][1] = ***CONSTANTS***.x4\_sum;  
 matrix.matrix[3][2] = ***CONSTANTS***.x5\_sum;  
 matrix.matrix[3][3] = ***CONSTANTS***.x6\_sum;  
 matrix.matrix[3][4] = ***CONSTANTS***.x3y\_sum;  
  
 ***thirdAnswers*** = methodGauss(matrix);  
  
 ***console***.log(***thirdAnswers***);  
}  
  
function outputThird() {  
 ***document***.getElementById("third-out").value = "";  
 ***document***.getElementById("third-out").value =  
 ***thirdAnswers***[3] + "\*x^3 + " + ***thirdAnswers***[2] + "\*x^2 + " + ***thirdAnswers***[1] + "\*x^1 + " + ***thirdAnswers***[0];  
  
 for (let i in ***inputPoints***) {  
 drawPoint(2,  
 ***inputPoints***[i].x,  
 ***thirdAnswers***[3]\****Math***.pow(***inputPoints***[i].x, 3) + ***thirdAnswers***[2]\****Math***.pow(***inputPoints***[i].x, 2)  
 +***thirdAnswers***[1]\****Math***.pow(***inputPoints***[i].x, 1) + ***thirdAnswers***[0],  
 "white");  
 }  
}

Экспоненциальная аппроксимация:

var ***exponentAnswers*** = [];  
  
function exponentAppr() {  
 ***exponentAnswers*** = new ***Array***(2);  
  
 ***exponentAnswers***[1] =  
 (***inputPoints***.length \* ***CONSTANTS***.x\_ln\_y\_sum - ***CONSTANTS***.x1\_sum \* ***CONSTANTS***.ln\_y\_sum)  
 /  
 (***inputPoints***.length \* ***CONSTANTS***.x2\_sum - ***Math***.pow(***CONSTANTS***.x1\_sum, 2));  
  
  
 ***exponentAnswers***[0] =  
 ***CONSTANTS***.ln\_y\_sum / ***inputPoints***.length  
 -  
 ***exponentAnswers***[1] \* ***CONSTANTS***.x1\_sum / ***inputPoints***.length;  
  
 ***console***.log(***exponentAnswers***);  
}  
  
function outputExponent() {  
 ***document***.getElementById("exponent-out").value = "";  
 ***document***.getElementById("exponent-out").value = "e^(" + ***exponentAnswers***[0] + " + " + ***exponentAnswers***[1] + "\*x)";  
  
 for (let i in ***inputPoints***) {  
 drawPoint(3, ***inputPoints***[i].x, ***Math***.exp(***exponentAnswers***[0] + ***exponentAnswers***[1]\****inputPoints***[i].x), "white");  
 }  
}

Логарифмическая аппроксимация:

var ***logAnswers*** = [];  
  
function logAppr() {  
 ***logAnswers*** = new ***Array***(2);  
  
 ***logAnswers***[1] =  
 (***inputPoints***.length \* ***CONSTANTS***.y\_ln\_x\_sum - ***CONSTANTS***.ln\_x\_sum \* ***CONSTANTS***.y\_sum)  
 /  
 (***inputPoints***.length \* ***CONSTANTS***.ln\_x\_ln\_x\_sum - ***Math***.pow(***CONSTANTS***.ln\_x\_sum, 2));  
  
  
 ***logAnswers***[0] =  
 ***CONSTANTS***.y\_sum / ***inputPoints***.length  
 -  
 ***logAnswers***[1] \* ***CONSTANTS***.ln\_x\_sum / ***inputPoints***.length;  
  
 ***console***.log(***logAnswers***);  
}  
  
function outputLog() {  
 ***document***.getElementById("log-out").value = "";  
 ***document***.getElementById("log-out").value = ***logAnswers***[0] + " + " + ***logAnswers***[1] + "\*ln(x)";  
  
 for (let i in ***inputPoints***) {  
 drawPoint(4, ***inputPoints***[i].x, ***logAnswers***[0] + ***logAnswers***[1]\****Math***.log(***inputPoints***[i].x), "white");  
 }  
}

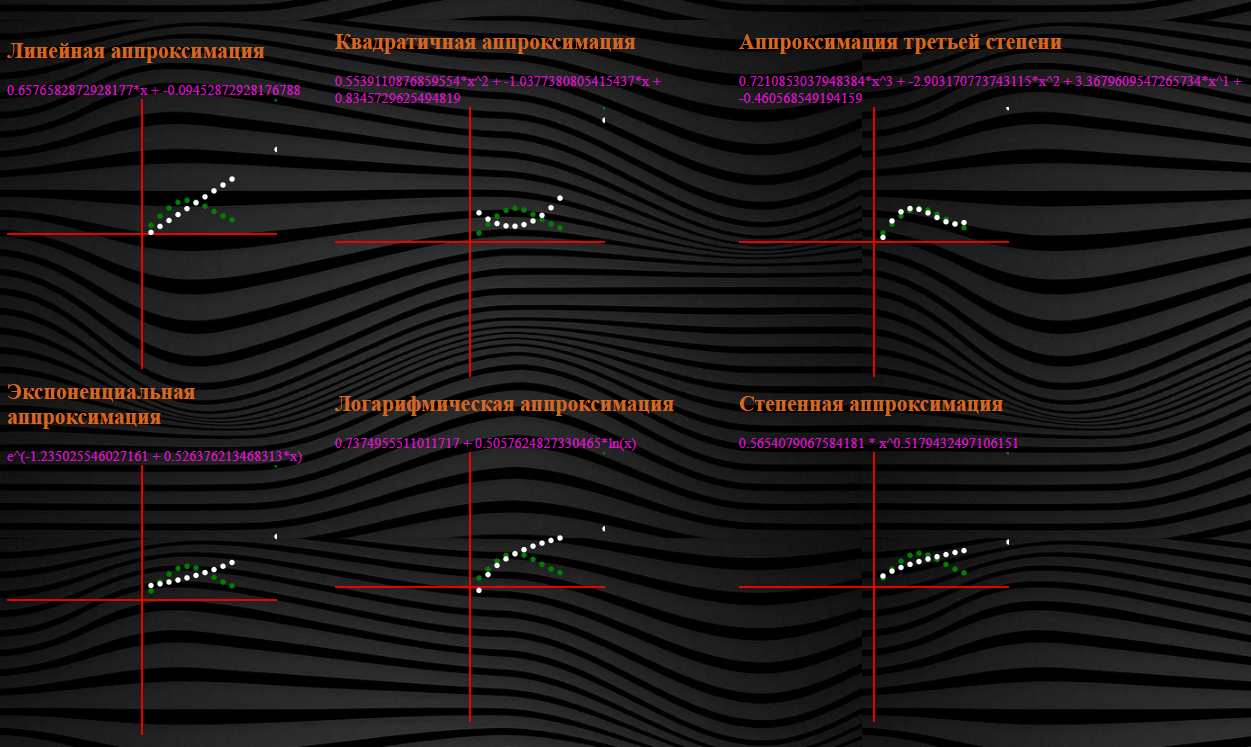
Степенная аппроксимация:

var ***powAnswers*** = [];  
  
function powAppr() {  
 ***powAnswers*** = new ***Array***(2);  
  
 ***powAnswers***[1] =   
 (***inputPoints***.length \* ***CONSTANTS***.ln\_x\_ln\_y\_sum - ***CONSTANTS***.ln\_x\_sum \* ***CONSTANTS***.ln\_y\_sum)  
 /  
 (***inputPoints***.length \* ***CONSTANTS***.ln\_x\_ln\_x\_sum - ***Math***.pow(***CONSTANTS***.ln\_x\_sum, 2))  
  
  
 ***powAnswers***[0] = ***Math***.exp(  
 ***CONSTANTS***.ln\_y\_sum / ***inputPoints***.length  
 -  
 ***powAnswers***[1] \* ***CONSTANTS***.ln\_x\_sum / ***inputPoints***.length  
 );  
  
 ***console***.log(***powAnswers***);  
}  
  
function outputPow() {  
 ***document***.getElementById("pow-out").value = "";  
 ***document***.getElementById("pow-out").value = ***powAnswers***[0] + " \* x^" + ***powAnswers***[1];  
  
 for (let i in ***inputPoints***) {  
 drawPoint(5, ***inputPoints***[i].x, ***Math***.pow(***inputPoints***[i].x, ***powAnswers***[1])\****powAnswers***[0], "white");  
 }  
}

**data.txt:**

|  |
| --- |
|  |

**Вывод программы:**



**Вывод**: в ходе данной лабораторной работы я научился выводить различные аппроксимации по методу наименьших квадратов. Аппроксимации были запрограммированы, осуществлён вывод точек – как исходных, так и тех, что соответствуют точкам функции(аппроксимирующих) при тех же входных аргументах.