**ИДЗ**

**Тема**: Метод наименьших квадратов (МНК).

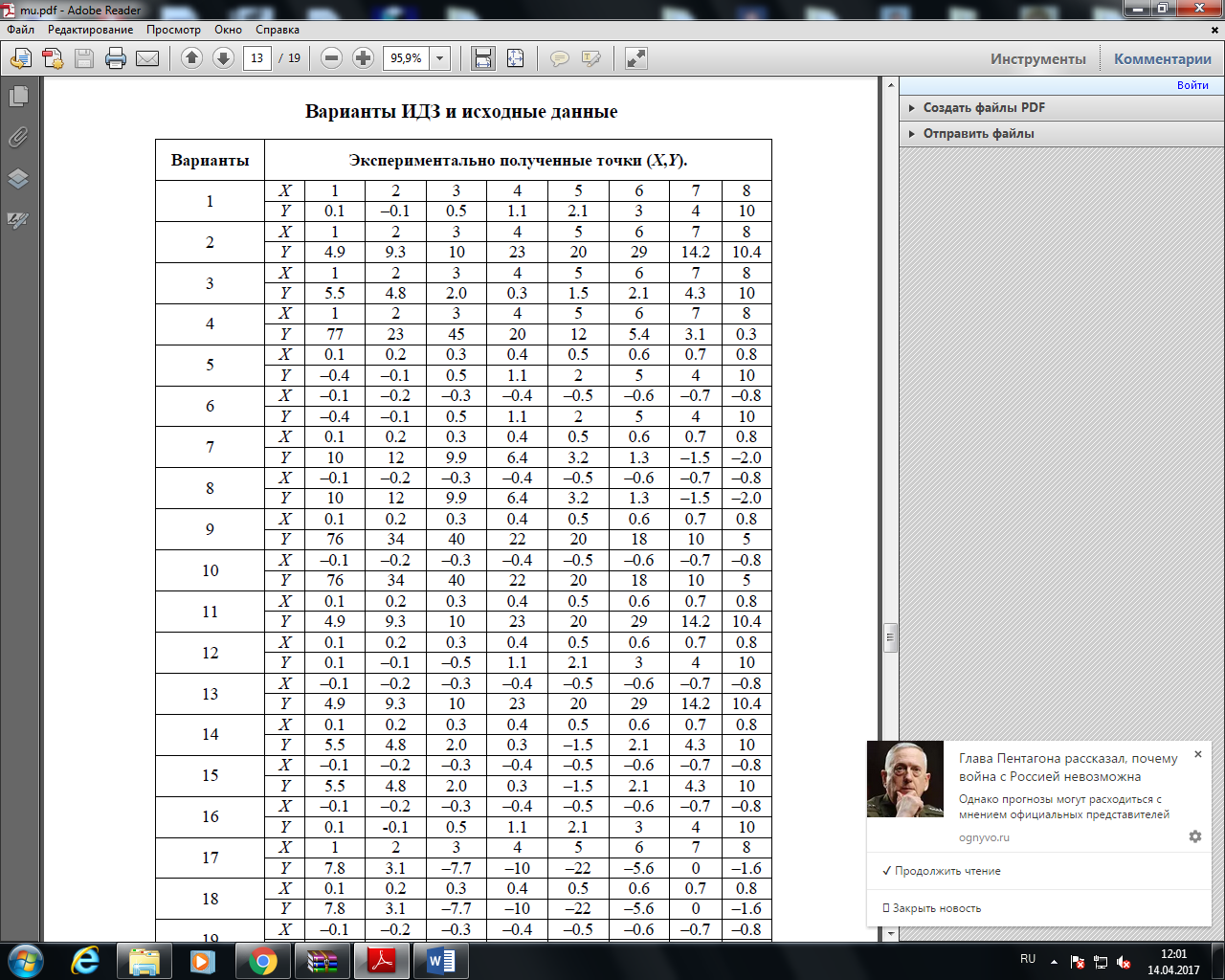
**Цель**: Изучение метода и практическое его применение для обработки данных.

**Задачи**: Изучить метод наименьших квадратов. Составить программу на языке Pascal для нахождения аппроксимирующей функции по исходным точкам, полученным в результате эксперимента. Исходные данные ввести из файла.

**Содержание отчета**

1. Задание.

Составить программу на языке Pascal для нахождения аппроксимирующей функции по исходным точкам, полученным в результате эксперимента. Исходные данные ввести из файла.



1. Краткое описание МНК.

Метод наименьших квадратов (МНК) — математический метод, применяемый для решения различных задач, основанный на минимизации суммы квадратов отклонений некоторых функций от искомых переменных. Он может использоваться для «решения» переопределенных систем уравнений (когда количество уравнений превышает количество неизвестных), для поиска решения в случае обычных (не переопределенных) нелинейных систем уравнений, для аппроксимации точечных значений некоторой функции. МНК является одним из базовых методов регрессионного анализа для оценки неизвестных параметров регрессионных моделей по выборочным данным.

Пусть x — набор n неизвестных переменных (параметров), fi(x), i=1,.. ,m и m>n — совокупность функций от этого набора переменных. Задача заключается в подборе таких значений x, чтобы значения этих функций были максимально близки к некоторым значениям yi. По существу речь идет о «решении» переопределенной системы уравнений fi(x)=yi i=1,.. ,m в указанном смысле максимальной близости левой и правой частей системы. Сущность МНК заключается в выборе в качестве «меры близости» суммы квадратов отклонений левых и правых частей |f i (x)-y i|. Таким образом, сущность МНК может быть выражена следующим образом:

В случае, если система уравнений имеет решение, то минимум суммы квадратов будет равен нулю и могут быть найдены точные решения системы уравнений аналитически или, например, различными численными методами оптимизации. Если система переопределена, то есть, говоря нестрого, количество независимых уравнений больше количества искомых переменных, то система не имеет точного решения и метод наименьших квадратов позволяет найти некоторый «оптимальный» вектор x в смысле максимальной близости векторов y и f(x) или максимальной близости вектора отклонений e к нулю (близость понимается в смысле евклидова расстояния).

1. Блок-схема алгоритма.

4. Текст программы. Текст файла исходных данных.

Программа на языке Pascal, содержит:

* ввод исходных точек из файла данных;
* формирование матрицы Грамма;
* решение системы линейных уравнений методом Гаусса;
* вывод результатов и проверку.

//Метод наименьших квадратов

**program**Mnk;

**type**matrix=**array**[0..100,0..100] **of** real;

vector=**array** [0..100] **of** real;//Нумеруем точки с нуля

**var**n,m,k,i:integer;

x,f,c:vector;

a:matrix;

x0,x9,h,x1:real;

**procedure**InputData (n:integer; **var**x,f:vector); //Вводисходныхданныхизфайлавмассивы

**var**i:integer;

f1:text;

**begin**

assign(f1,'data.txt');

reset(f1);

**for**i:=0 **to** n **do begin**

readln (f1,x[i],f[i]);

**end**;

close(f1);

**end**;

**procedure**Vivod(n:integer; x,f:vector); //Выводисходныхданныхнаэкран

**var**i:integer;

**begin**

write('Номерузла':10);

**for**i:=0 **to** n **do**

write(i:6);

writeln;

write('X(i)':10);

**for**i:=0 **to** n **do**

write(x[i]:6:2);

writeln;

write('Y(i)':10);

**for**i:=0 **to** n **do**

write(f[i]:6:2);

writeln;

**end**;

**function**ex (a:real; n:integer):real;

//Показательная функция для формирования матрицы Грама

**var**i:integer;

e:real;

**begin**

e:=1;

**for**i:=1 **to** n **do** e:=e\*a;

ex:=e;

**end**;

**procedure**Gram (n,m:integer; **var**x,f:vector; **var**a:matrix);

//Формирование матрицы Грама A по векторам данных X,F

**var**i,j:integer;

p,q,r,s:real;

**begin**

**for**j:=0 **to** m **do begin**

s:=0; r:=0; q:=0;

**for**i:=0 **to** n **do begin**

p:=ex(x[i],j);

s:=s+p;

r:=r+p\*f[i];

q:=q+p\*ex(x[i],m);

**end**;

a[0,j]:=s;

a[j,m]:=q;

a[j,m+1]:=r;

**end**;

//Надо формировать только 1-ю строку и 2 последних столбца матрицы Грама,

//остальные элементы легко получить циклическим копированием:

**for**i:=1 **to** m **do**

**for**j:=0 **to** m-1 **do** a[i,j]:=a[i-1,j+1];

**end**;

**procedure**Gauss(n:integer; **var**a:matrix; **var**x:vector);

//Решение СЛАУ методом Гаусса

//a - расширенная матрица системы, x - вектор результата

**var**i,j,k,l,k1,n1:integer;

r,s:real;

**begin**

//Прямойход:

n1:=n+1;

**for**k:=0 **to** n **do begin**

k1:=k+1;

s:=a[k,k];

**for**j:=k1 **to** n1 **do** a[k,j]:=a[k,j]/s;

**for**i:=k1 **to** n **do begin**

r:=a[i,k];

**for**j:=k1 **to** n1 **do** a[i,j]:=a[i,j]-a[k,j]\*r;

**end**;

**end**;

//Обратныйход:

**for**i:=n **downto**0 **do begin**

s:=a[i,n1];

**for**j:=i+1 **to** n**do** s:=s-a[i,j]\*x[j];

x[i]:=s;

**end**;

**end**;

**function**fi (m:integer; **var**c:vector; x1:real):real;

{Аппроксимирующая функция по найденным коэффициентам МНК

m - степень полинома, c - вектор коэффициентов,

x1 - точка, в которой ищем значение}

**var**i:integer; p:real;

**begin**

p:=c[m];

**for**i:=m-1 **downto**0 **do** p:=c[i]+x1\*p;

fi:=p;

**end**;

**begin**

writeln ('Подбор зависимости методом наименьших квадратов, число узлов 8');

writeln ('Исходные данные');

n:=8;

n:=n-1; //нумерация будет с нуля!

write ('Введите степень полинома (1<=m<=',n,'):');

read (m);

writeln;

InputData (n,x,f); //вводимданные

writeln;

Vivod(n,x,f);// вывод исходных данных из файла на экран

Gram (n,m,x,f,a); //считаем матрицу Грама

Gauss (m,a,c); //решаем систему линейных уравнений

writeln ('Коэффициенты полинома МНК ',m,' степени:');

**for**i:=0 **to** m **do** write (c[i] : 15 : 4);

writeln;

writeln ('Введите границы по оси X для построения полинома:');

read (x0,x9);

writeln ('Введите шаг по X для построения значений полинома:');

read (h);

k:=round((x9-x0)/h+1);

x1:=x0;

**for**i:=1 **to** k **do begin**

//строим и выводим полином по найденным коэффициентам

writeln (x1:10:4,fi(m,c,x1) : 15 : 4);

x1:=x1+h;

**end**;

**end**.

5. Результаты вычислений.

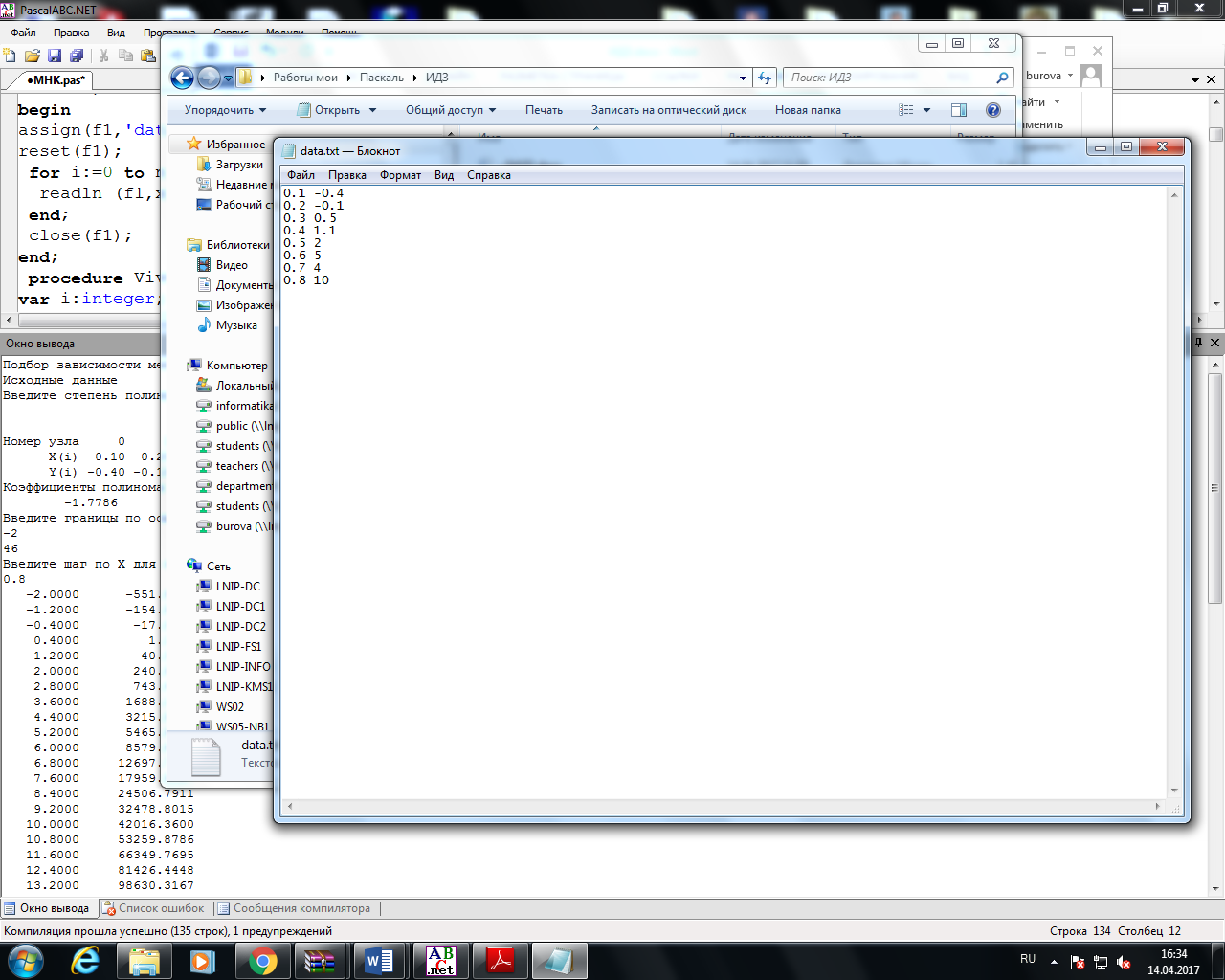


Рисунок 1-Текстовый файл с данными

Сами данные

Х У

0.1 -0.4

0.2 -0.1

0.3 0.5

0.4 1.1

0.5 2

0.6 5

0.7 4

0.8 10

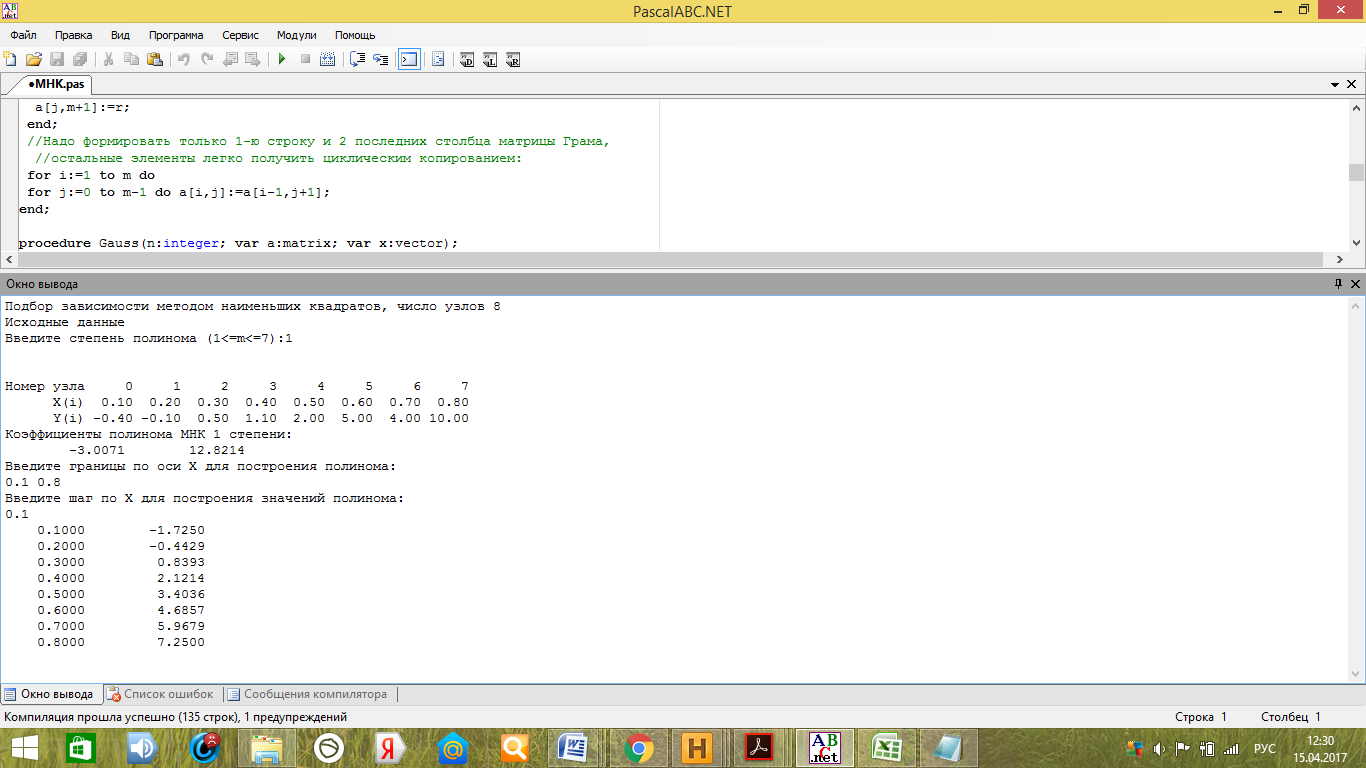


Рисунок 2-МНК1 степени

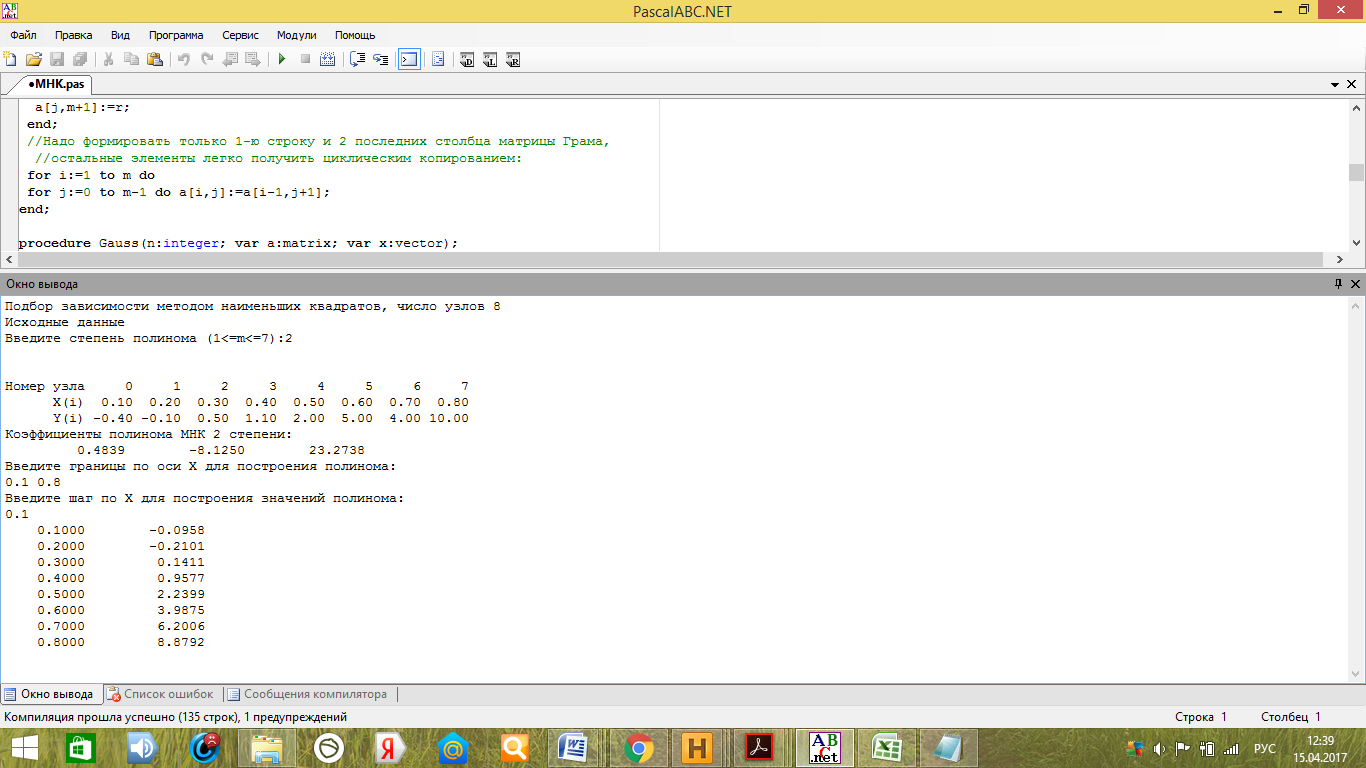


Рисунок 3-МНК 2 степени

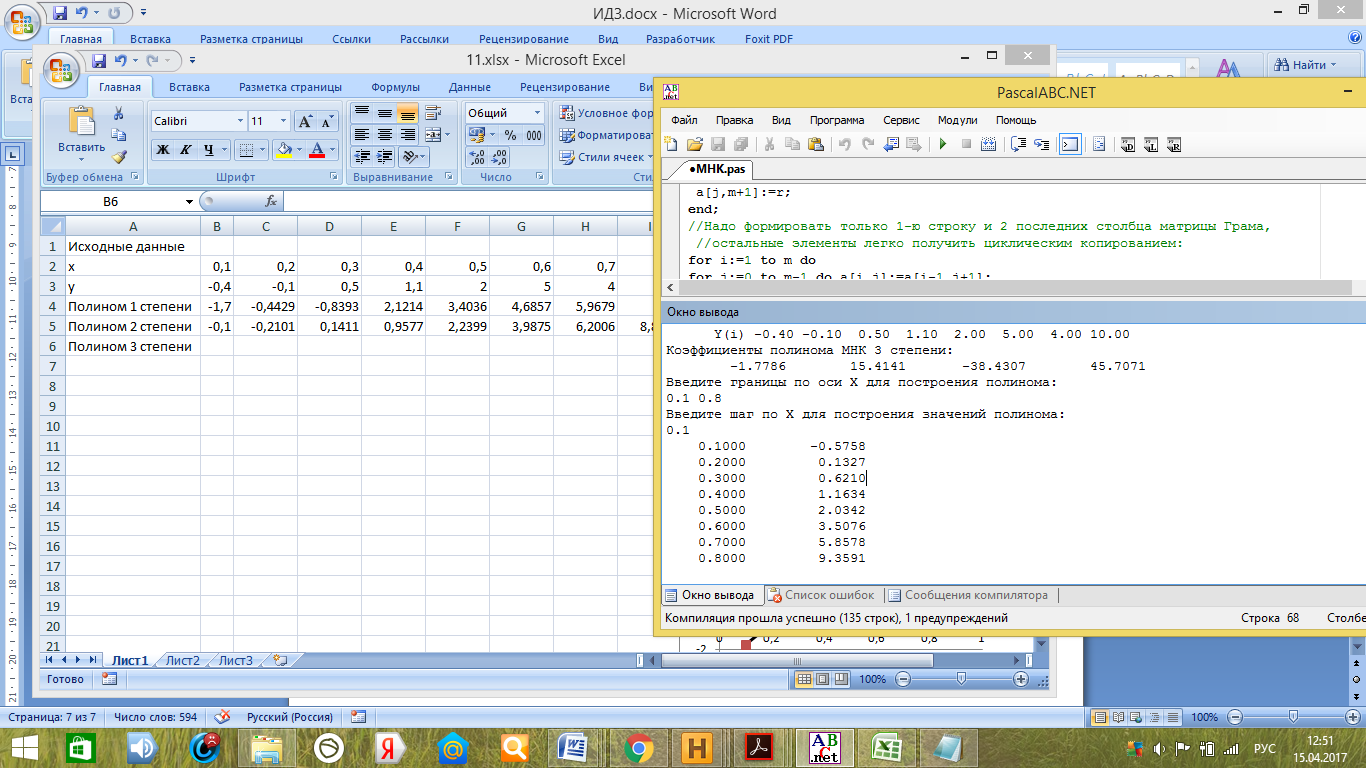


Рисунок 4-МНК 3 степени

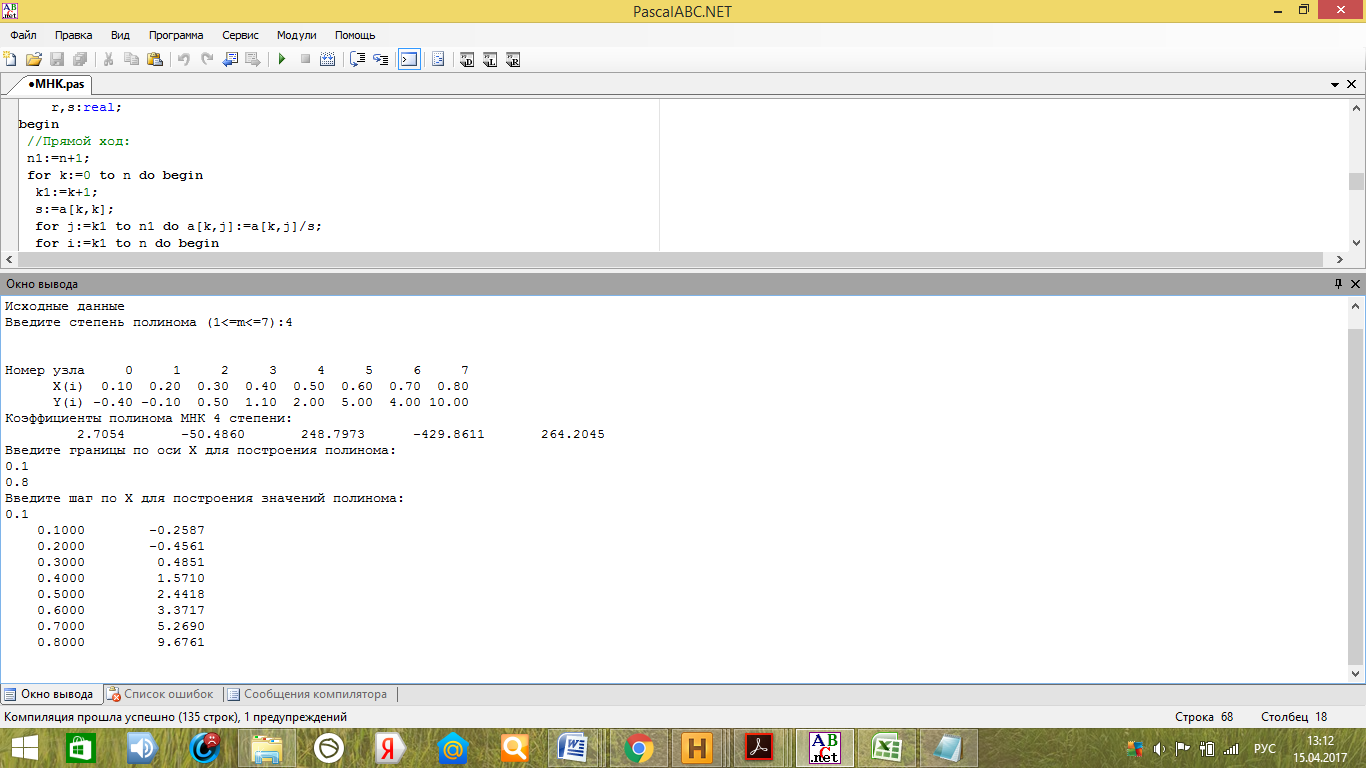


Рисунок 5-МНК 4 степени

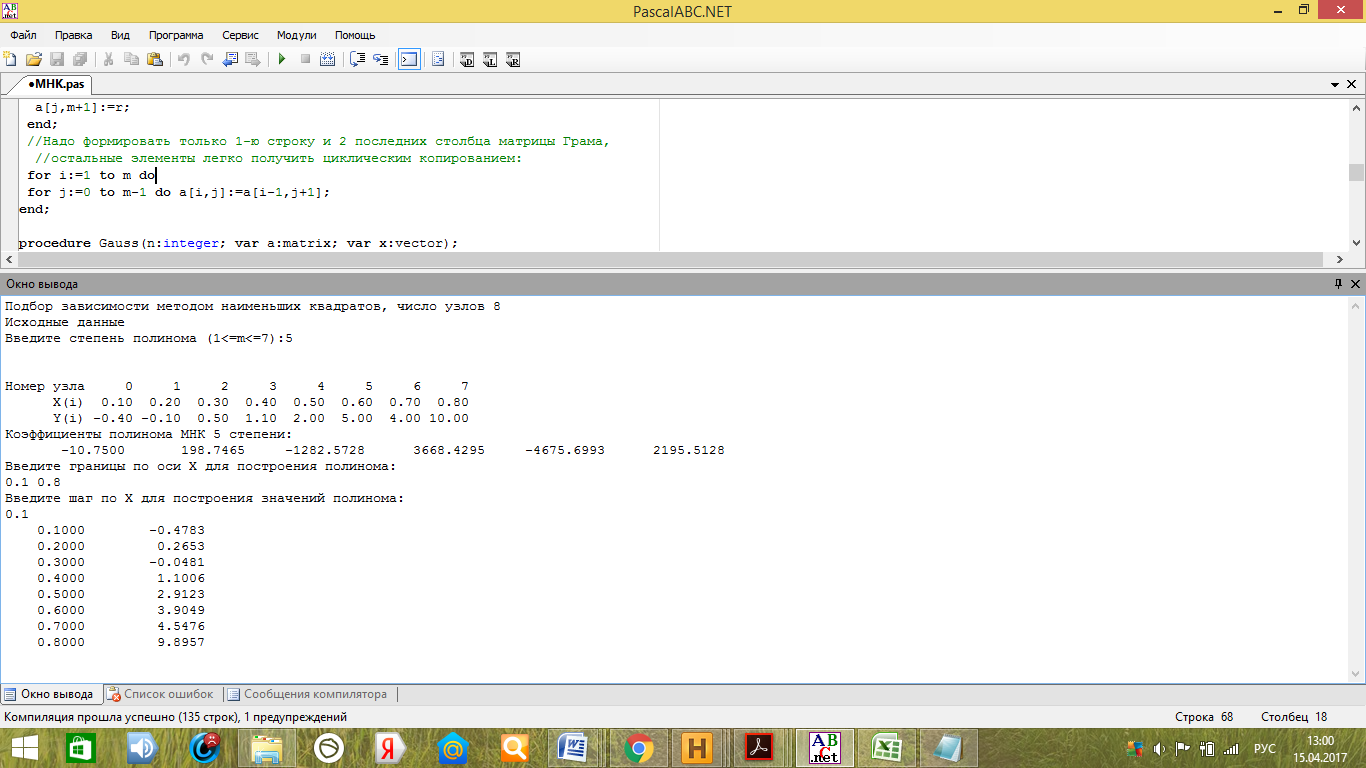


Рисунок 6-МНК 5 степени

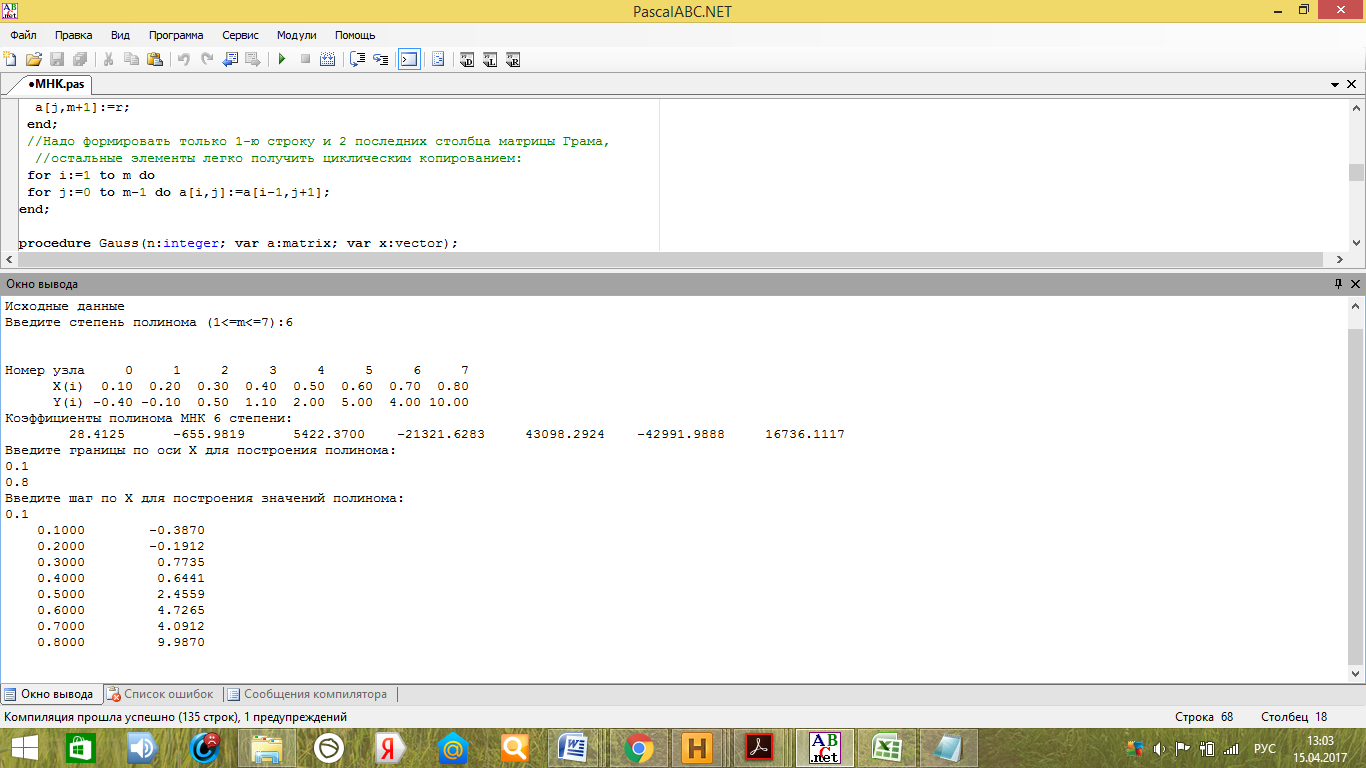


Рисунок 7-МНК 6 степени

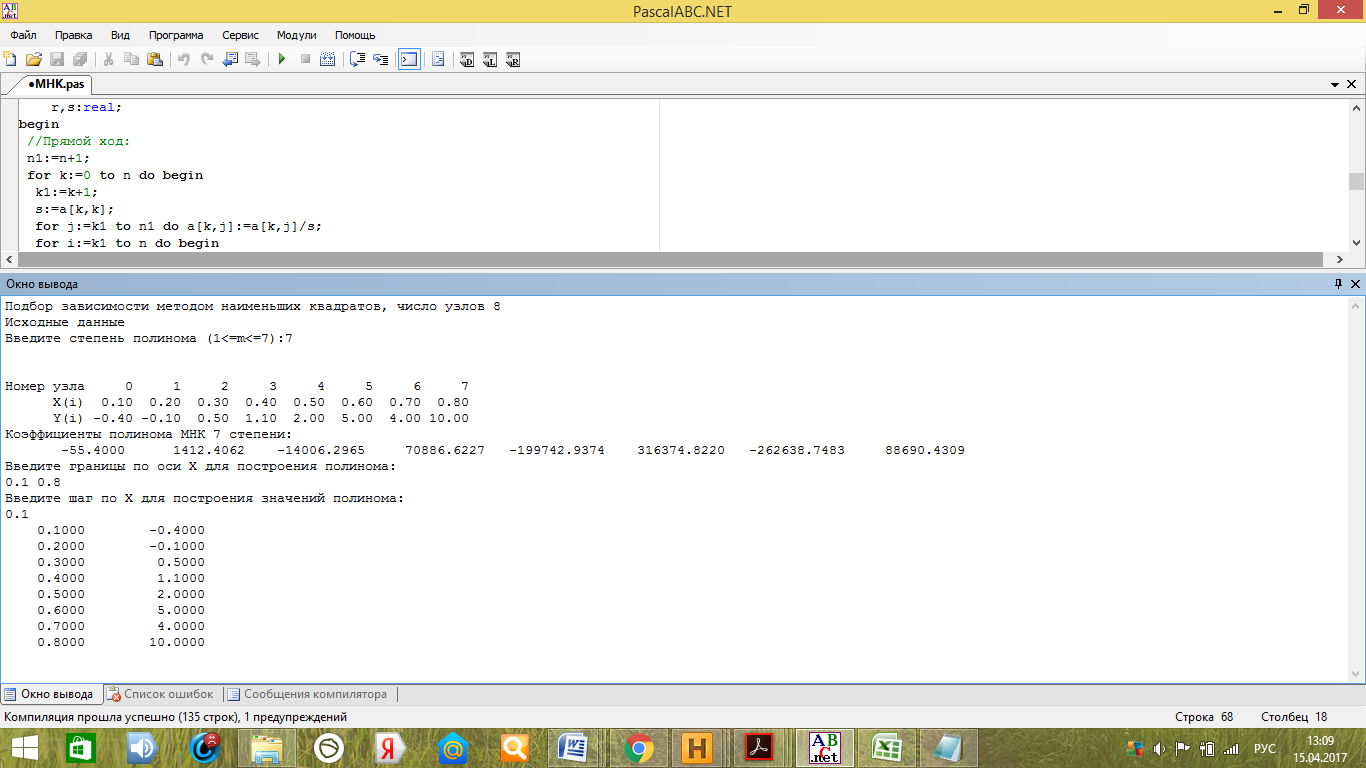


Рисунок 8-МНК 7 степени

6. Графики аппроксимирующих функций.

7. Выводы по работе.

Чем больше степень МНК, тем точнее апроксимация исходных данных.