МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА №2

КУРСОВАЯ РАБОТА
ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

доц., канд. техн. наук		Галанина В.А.
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К КУРСОВОЙ РАБОТЕ

Аппроксимация функции методом наименьших квадратов

по дисциплине: ИНФОРМАТИКА

РАБОТУ ВЫПОЛНИЛА

СТУДЕНТКА ГР. № 2746		Келлер А.Г.
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2018

Оглавление

1.Цель работы

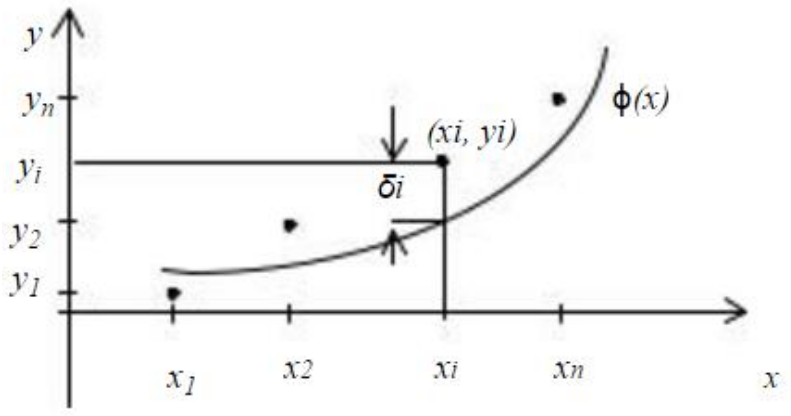
1. Закрепление навыков алгоритмизации и программирования.
2. Практическое освоение типовых вычислительных методов прикладной математики;
3. Совершенствование навыков разработки алгоритмов и построение программ на языке высокого уровня;
4. Освоение принципов модульного программирования и техники использования библиотек на DevC++.

2.Постановка задачи

Тематика курсовой работы связана с решением задачи аппроксимации зависимости между величинами X и Y, полученной экспериментальным путем, с помощью известных функций или их комбинаций, подобранных надлежащим образом. Эта зависимость может быть задана значениями в отдельных точках (xi, yi), либо некоторой функцией y=f(x), заданной на интервале [a, b].

Исходными данными являются функциональная зависимость между X и Y, заданная таблично в n точках, и набор базисных функций. Требуется, используя МНК, найти аппроксимирующую функцию ϕ(x) из заданного класса функций и оценить степень ее отклонения от исходной функции.

Пусть из теоретических или иных соображений (например, по графику) выбран вид аппроксимирующей функции $y = \varphi(x)$. Эта функция в качестве параметров помимо x содержит еще ряд числовых параметров (C1, …, Cm).



Один из распространённых подходов опирается на использование метода наименьших квадратов (МНК), в соответствии с которым наилучшей считается такая аппроксимирующая функция $f(x)$, для которой достигается наименьшее значение суммы квадратов отклонений от значений аппроксимирующей функции во всех точках *x*, принимаемых во внимание.

Запишем требование МНК аналитически:

$$J=\sum_{i=1}^{n}(\delta i)^2=\min\sum_{i=1}^{n}(yi-\varphi(xi,C1,C2,\dots,Cm))^2 \qquad (1)$$

В настоящей курсовой работе исходные данные заданы в виде табличной зависимости $y_i(x_i)$. Уточним условия МНК для этой задачи.

2.1. Методика выбора аппроксимирующей функции

Аппроксимирующую функцию $\varphi(x)$ выбирают из некоторого семейства функций, для которого задан вид функции, но остаются неопределенными (и подлежат определению) её параметры $C_1, C_2, \dots, C_m$, т.е.

$$\varphi(x)=\varphi(C_1,C_2,\dots,C_m) \qquad (2)$$

Определение аппроксимирующей функции $\varphi(x)$ подразделяется на два основных этапа:

- подбор подходящего вида функций $\varphi(x)$;
- нахождение параметров функции $\varphi(x)$ в соответствии с критерием МНК.

Подбор вида функции $\varphi(x)$ представляет собой сложную задачу, решаемую методом проб и последовательных приближений. Исходные данные, представленные в графической форме, сопоставляются с семействами графиков ряда типовых функций, используемых обычно для аппроксимации.

После того как выбран вид аппроксимирующей функции $\varphi(x)$ и, следовательно, определена функциональная зависимость, необходимо найти в соответствии с требованием МНК значения параметров C1, C2,…, Cm.

Параметры должны быть определены таким образом, чтобы значения критерия было наименьшим.

Для решения задачи подставим выражение (2) в выражение (1) и проведем необходимые операции суммирования. В результате величинаJ, критерий аппроксимации, представится функцией искомых параметров

$J(x) = J(C_1, C_2, \dots, C_m)$(3)

2.2.Описание метода выбора аппроксимирующей функции.

Аппроксимирующую функцию $\varphi(x)$ выбирают из некоторого семейства функций, для которого задан вид функции, но остаются неопределенными (и подлежат определению) её параметры $C_1, C_2, \dots, C_m$, т.е.

$\varphi(x) = \varphi(C_1, C_2, \dots, C_m)$ (2)

Для решения задачи подставим выражение (2) в выражение (1) и проведем необходимые операции суммирования. В результате величина J, критерий аппроксимации, представится функцией искомых параметров

$J(x) = J(C_1, C_2, \dots, C_m)$(3)

Последующие действия сводятся к отыскиванию минимума этой функции J переменных C_k. Определение значений, соответствующих этому минимуму J, и является целью решаемой задачи. Поскольку величина J неотрицательна (как сумма квадратов) и нижняя её граница есть 0 ($J = 0$), т.о., если существующее решение системы единственно, оно отвечает именно минимуму J.

Уравнения, используемые в МНК, называются нормальными, поэтому описываемый способ решения задачи условимся называть методом нормальных уравнений.

Структура этих уравнений получается более простой в том случае, когда аппроксимирующая функция $\varphi(x)$ выбирается линейной функцией искомых параметров C_k и выражение (2) имеет вид:

$$\varphi(x) = \sum_{k=1}^{m} C_k \varphi_k(x) \text{ (4)}$$

где $C_k -$ определяемые параметры; $\varphi_1(x), \varphi_2(x), \dots, \varphi_m(x)$ – система некоторых линейно-независимых функций, называемых в курсовой работе базисными функциями.

Замечание. Функции $\varphi_1(x), \varphi_2(x), \dots, \varphi_m(x)$называются линейно-независимыми, если при любых *x* равенство

$$\sum_{k=1}^{m} C_k f_k(x) = 0$$

справедливо только тогда, когда все C_k=0.

Примеры функций, которые можно использовать в качестве базисных

Вид функции $\varphi k(x)\ (k = 1, \dots, m)$	Название функций
$\varphi 1(x) = x$	линейная
$\varphi 2(x) = x^2$	параболическая
$\varphi 3(x) = 1/x$	обратно-пропорциональная
$\varphi 4(x) = \log_a x$	логарифмическая
$\varphi 5(x) = \sin(x)$	тригонометрические

В этом случае, подставляя (4) в выражение (1) и выполняя дифференцирование, получим систему уравнений относительно искомых C_k.

Подставим выражение аппроксимирующей функции для m базисных функций

$$\varphi(x) = C1\varphi 1(x) + C2\varphi 2(x) + \cdots + Cm\varphi m(x)$$

в формулу критерия аппроксимации.

Получим

$J = \sum_{i=1}^{n} [yi - C1\varphi 1(xi) - C2\varphi 2(xi) - \cdots - Cm\varphi m(xi)]^2$

Применим операцию дифференцирования к параметру C1:

$$2\sum_{i=1}^{n} [yi - C1\varphi 1(xi) - C2\varphi 2(xi) - \cdots - Cm\varphi m(xi)]\left(-\varphi 1(xi)\right) = 0$$

и, выполняя необходимые алгебраические преобразования, получим уравнение

$$C1\sum_{i=1}^{n}\varphi 1(xi)\varphi 1(xi)+\cdots+Cm\sum_{i=1}^{n}\varphi 1(xi)\varphi m(xi)=\sum_{i=1}^{n}yi\varphi 1(xi)$$

Аналогичные уравнения можно получить, применяя описанные выше действия по отношению к переменным C2, …, Cm. Эти уравнения образую систему нормальных уравнений

$$\begin{aligned}
&a_{11}C_1+a_{12}C_2+\ldots+a_{1m}C_m=b_1\\
&a_{21}C_1+a_{22}C_2+\ldots+a_{2m}C_m=b_2,\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&a_{m1}C_1+a_{m2}C_2+\ldots+a_{mm}C_m=b_m
\end{aligned}$$

где коэффициенты a_{kl} и величины b_k(k, l=1, 2, …, m) определяются выражениями

$$a(kl)=\sum_{i=1}^{n}\varphi k(xi)\varphi l(xi),\qquad b(k)=\sum_{i=1}^{n}yi\varphi k(xi)$$

Систему m линейных уравнений можно записать посредством матричных обозначений в следующем виде:

$$A\times C=B$$

Квадратная матрица A называется матрицей системы, вектор C – вектором-столбцом неизвестных системы, а вектор B – вектором-столбцов свободных членов.

Решение системы линейных уравнений сводится к отысканию значений элементов вектора-столбца C, называемых корнями системы. Для получения единственного решения системы, входящие в нее m уравнений должны быть линейно независимыми. Необходимым и достаточным условием этого является неравенство нулю определителя данной системы, то есть det A≠0.

Алгоритмы решения систем линейных уравнений подразделяются на прямые и итерационные. Теоретически для получения точного решения итерационные методы требуют бесконечного числа в арифметических операций. Практически это число приходится брать конечным, поэтому решение имеет некоторую ошибку. Что же касается прямых методов, то они даже при конечном числе операций могут в принципе дать

точное решение, если пренебречь ошибками округлений при вычислениях на современных ПК.

2.3.Описание метода простой итерации.

Итерационные методы позволяют получить значения корней системы с заданной точностью в виде предела последовательности некоторых векторов C(0),C(1), …,C(k). Процесс получения элементов такой последовательности носит итерационный (повторяющийся) характер.

Эффективность применения таких методов зависит от удачного выбора начального вектора C(0) и быстроты сходимости процесса.

Полагая, что в системе уравнений все диагональные элементы отличны от нуля, то есть $aii \neq 0$ ($i = 1, 2, \ldots, m$), выразим $C1$ через первое уравнение системы, $C2$ через второе уравнение и т.д. В результате получим новую систему.

$$C_1 = \frac{b_1}{a_{11}} - \frac{a_{12}}{a_{11}} C_2 - \frac{a_{13}}{a_{11}} C_3 - \ldots - \frac{a_{1m}}{a_{11}} C_m$$

$$C_2 = \frac{b_2}{a_{22}} - \frac{a_{21}}{a_{22}} C_1 - \frac{a_{23}}{a_{22}} C_3 - \ldots - \frac{a_{2m}}{a_{22}} C_m$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$C_m = \frac{b_m}{a_{mm}} - \frac{a_{m1}}{a_{mm}} C_1 - \frac{a_{m2}}{a_{mm}} C_2 - \ldots - \frac{a_{m,m-1}}{a_{mm}} C_{m-1}.$$

Обозначим bi / aij =i- aij. где i,j = 1, 2,...,m. В этом случае новая система принимает вид

$$C_1 = \beta_1 + \alpha_{12} C_2 + \alpha_{13} C_3 + \ldots + \alpha_{1m} C_m$$
$$C_2 = \beta_2 + \alpha_{21} C_1 + \alpha_{23} C_3 + \ldots + \alpha_{2m} C_m$$
$$C_m = \beta_m + \alpha_{m1} C_1 + \alpha_{m2} C_2 + \ldots + \alpha_{m\,m-1} C_{m-1}.$$

Введём обозначения

$$\alpha = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \ldots & \alpha_{1m} \\ \alpha_{21} & \alpha_{22} & \ldots & \alpha_{m} \\ \ldots & \ldots & \ldots & \ldots \\ \alpha_{m1} & \alpha_{m2} & \ldots & \alpha_{mm} \end{bmatrix} \quad \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \ldots \\ \beta_m \end{bmatrix}.$$

Отметим, что в данном случае aij=0(i=1, 2, …, m). Тогда система может быть записана в матричной форме

$$C=\beta+ a\, c$$

или

$$\begin{bmatrix} C_1 \\ C_2 \\ \dots \\ C_m \end{bmatrix} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \dots \\ \beta_m \end{bmatrix} + \begin{bmatrix} \alpha_{11} & \alpha_{12} & \dots & \alpha_{1m} \\ \alpha_{21} & \alpha_{22} & \dots & \alpha_{m} \\ \dots & \dots & \dots & \dots \\ \alpha_{m1} & \alpha_{m2} & \dots & \alpha_{mm} \end{bmatrix} \times \begin{bmatrix} C_1 \\ C_2 \\ \dots \\ C_m \end{bmatrix}.$$

Сходимость процесса зависит от величины элементов матрицы a следующим образом: если сумма модулей элементов строк или сумма модулей элементов столбцов меньше единицы, то итерационный процесс отыскания приближений к истинным значениям корней системы сходится к единственному решению независимо от выбора начальных (в частности, нулевых) приближений,то есть условия сходимости можно записать следующим образом:

$$\sum_{j=1}^{m} |\alpha_{ij}| < 1,\ i=1,\dots,m) \quad \text{или} \quad \sum_{i=1}^{m} |\alpha_{ij}| < 1,\ j=1,\dots,m).$$

Сходимость итерационного процесса можно оценить и посредством норм матрицы a. А именно, процесс сходится, если выполняется одно из следующих условий:

$$\|\alpha\|_1 = \max \sum_{i=1}^{m} \alpha_{ij} < 1,$$

или

$$\|\alpha\|_2 = \max \sum_{j=1}^{m} \alpha_{ij} < 1,$$

или

$$\|\alpha\|_3 = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{m} \left|\alpha_{ij}\right|^2} < 1.$$

Эти условия как достаточные, предъявляют завышенные требования к матрице, поэтому в некоторых случаях сходимость обеспечивается, если даже |a|≥1.

Условия сходимости также выполняются, если в матрице A диагональные элементы преобладают, то есть

$$|\alpha_{ii}| \geq \sum_{j \neq i}^{m} |\alpha_{ij}|.$$

Другими словами, модули диагональных коэффициентов в каждом уравнении системы больше суммы модулей недиагональных коэффициентов (свободные члены не рассматриваются).

Пусть C- вектор точных значенийй неизвестных (корней) системы уравнений, а C^k – вектор k-х приближений к этимточным значениям. Тогда для оценки погрешности метода последовательных приближений, где eps – относительная точность вычислений.

$$||C\text{-}C^{(k)}|| \leq \text{eps}$$

На практике при заданном значении eps часто в качестве критерия окончания итерационного процесса вычисления значений$(C_1, C_2, \ldots, C_m)$вектора C^kиспользуется неравенство

$$\sum_{i=1}^{m} \left| C_i^{(k+1)} - C_i^{(k)} \right| < \text{eps}$$

2.4.Схема алгоритма решения системы линейных уравнений методом последовательных приближений

Любое (k + 1) - е приближение вычисляется по формуле

$C_{k+1} = \beta + \alpha \times C_k$.

Алгоритм начинается с определения условия сходимости итерационного процесса. В случае, если условия сходимости не выполняется, происходит аварийное завершение программы и вычисления прекращаются.

2.5. Оценка погрешности аппроксимации

Результатом этапа решения системы нормальных уравнений является получение значений параметров аппроксимирующей функции

$$\varphi(x) = C1\varphi1(x) + C2\varphi2(x) + \cdots + Cm\varphi m(x)$$

для заданного набора базисных аппроксимирующих функций

$\varphi1(x), \varphi2(x), \ldots, \varphi m(x)$

Решение системы нормальных уравнений определяет значение параметров, при которых критерий качества аппроксимации J принимает минимально возможное значение $J_{min}=J(C1^*, C2^*, \ldots, Cm^*)$. При всех других допустимых значениях параметров величина критерия будет больше. Тем самым полученное значение J_{min} может быть принято за характеристику эффективности аппроксимации заданной функциональной зависимости функциями $\varphi(x)$ выбранного класса. При изменении класса аппроксимирующей функции, а также при изменении набора базисных функций значение J_{min} может меняться. Сравнение различных классов функций по их эффективности аппроксимации может осуществляться на основе сравнений соответствующих значений J_{min}.

Для количественной оценки погрешности аппроксимации может использоваться также величина Δ максимального отклонения исходной функциональной зависимости от найденной аппроксимирующей. Для этого определяется отклонение $\delta i = yi - \varphi(xi)$ во всех заданных точках и определяется максимальное из этих отклонений:

$$\Delta = \max|\delta i| = \max\ |yi - \varphi(xi)|$$

3.Ручной счёт.

Представление исходных данных (табличное)

Номер варианта	n	Значения x_i и y_i	Базисные функции			Метод решения СЛАУ
			$\varphi_1(x)$	$\varphi_2(x)$	$\varphi_3(x)$	

7	5	X	-1	-0.6	-0.1	0.2	0.7	x	1	$3x^2 - 1$	Простой итерации
		У	0.4	0.6	1	1.3	1.8				

Выражение для аппроксимации функции будет иметь следующий вид:
F(x)=$C_1\varphi_1(x) + C_2\varphi_2(x) + C_3\varphi(x)$
F(x)=$C_1x + C_21 + C_3(3x^2 - 1)$
Выражение для критерия аппроксимации:

$$J = \sum_{i=1}^{5}[y_i - C_1x - C_21 - C_3(3x^2 - 1)]^2$$

В соответствии с условиями локального минимума функции J(C_1, C_2, C_3) найдём частные производные $\frac{\partial J(C_1,C_2,C_3)}{\partial C_1}$, $\frac{\partial J(C_1,C_2,C_3)}{\partial C_2}$, $\frac{\partial J(C_1,C_2,C_3)}{\partial C_3}$ и приравняем их к нулю:

$$\frac{\partial J}{\partial C_1} = \sum_{i=1}^{5} 2 \cdot \left(y_i - C_1x - C_21 - C_3(3x^2 - 1) \cdot (-x)\right) =$$

$$= -2 \cdot (\sum_{i=1}^{5} y_i \cdot x - C_1\sum_{i=1}^{5} x^2 - C_2\sum_{i=1}^{5} x - C_3\sum_{i=1}^{5}(3x^3 - x)) = 0$$

$$\frac{\partial J}{\partial C_2} = \sum_{i=1}^{5} 2 \cdot \left(y_i - C_1x - C_21 - C_3(3x^2 - 1)\right) \cdot (-1) =$$

$$= -2 \cdot (\sum_{i=1}^{5} y_i - C_1\sum_{i=1}^{5} x - C_2\sum_{i=1}^{5} 1 - C_3\sum_{i=1}^{5}(3x^2 - 1)) = 0$$

$$\frac{\partial J}{\partial C_3} = \sum_{i=1}^{5} 2 \cdot \left(y_i - C_1x - C_21 - C_3(3x^2 - 1)\right) \cdot \left(-(3x^2 - 1)\right) =$$

$$= -2 \cdot (\sum_{i=1}^{5} y_i \cdot (3x^2 - 1) - -C_1\sum_{i=1}^{5}(3x^3 - x) - C_2\sum_{i=1}^{5}(3x^2 - 1) -$$

$$- C_3\sum_{i=1}^{5}(3x^2 - 1)^2) = 0$$

Получаем систему уравнений:

$$C_1\sum_{i=1}^{5} x^2 + C_2\sum_{i=1}^{5} x + C_3\sum_{i=1}^{5}(3x^3 - x) = \sum_{i=1}^{5} y_i \cdot x$$

$$C_1\sum_{i=1}^{5} x + C_2\sum_{i=1}^{5} 1 + C_3\sum_{i=1}^{5}(3x^2 - 1) = \sum_{i=1}^{5} y_i$$

$$C_1\sum_{i=1}^{5}(3x^3-x)+C_2\sum_{i=1}^{5}(3x^2-1)+C_3\sum_{i=1}^{5}(3x^2-1)^{2=\sum_{i=1}^{5}y_i\cdot(3x^2-1)}$$

i	x	y	x^2	$3x^3-x$	$3x^2-1$	$(3x^2-1)^2$	$y_i\cdot x_i$	$y_i\cdot(3x^2-1)$
1	-1	0,4	1	-2	2	4	-0,4	0,8
2	-0,6	0,6	0,36	-0,048	0,08	0,0064	-0,36	0,048
3	-0,1	1	0,01	0,097	-0,97	0,9409	-0,1	-0,97
4	0,2	1,3	0,04	-0,176	-0,88	0,7744	0,26	-1,144
5	0,7	1,8	0,49	0,329	0,47	0,2209	1,26	0,846
Σ	-0,8	5,1	1,9	-1,798	0,7	5,9426	0,66	-0,42

Используя значение из таблицы, запишем систему уравнений в окончательном виде:

$1{,}9C_1-0{,}8C_2-1{,}798C_3=0{,}66$

$-0{,}8C_1+5C_2+0{,}7C_3=5{,}1$

$-1{,}798C_1+0{,}7C_2+5{,}9426C_3=-0{,}42$

$$\begin{pmatrix}1{,}9 & -0{,}8 & -1{,}798\\ -0{,}8 & 5 & 0{,}7\\ -1{,}798 & 0{,}7 & 5{,}9426\end{pmatrix}\times\begin{pmatrix}C_1\\C_2\\C_3\end{pmatrix}=\begin{pmatrix}0{,}66\\5{,}1\\-0{,}42\end{pmatrix}$$

A·C=B

Решаем систему уравнений методом простой итерации

Оценка сходимости процесса:

$$max\left[\sum|a_{ij}|\right]=0{,}421+0{,}946=1{,}367>1$$

$$C_1=\frac{0{,}66}{1{,}9}+\frac{0{,}8}{1{,}9}C_2+\frac{1{,}798}{1{,}9}C_3$$

$$C_1=0{,}347+0{,}421C_2+0{,}946C_3$$

$$C_2=\frac{5{,}1}{5}+\frac{0{,}8}{5}C_1-\frac{0{,}7}{5}C_3$$

$$C_2=1{,}02+0{,}16C_1-0{,}14C_3$$

$$C_3=-\frac{0{,}42}{5{,}9426}+\frac{0{,}7}{5{,}9426}C_2-\frac{1{,}798}{5{,}9426}C_1$$

$$C_3=-0{,}0707+0{,}303C_1-0{,}118C_2$$

C_1	C_2	C_3
2,097284211	1,1844	-0,471732912-
0,399654855	1,421608081	0,424365936

1,347524	1,024534	-0,11721
0,667831051	1,252013641	0,216348979
1,079266514	1,096564111	-0,016095534
0,793848168	1,194936017	0,126699814
0,97039741	1,129277733	0,028755729
0,8500658	1,1712378	0,0899068
0,925601393	1,143423574	0,048556495
0,87476	1,161298	0,074687
0,907014	1,149505	0,057199
0,8855	1,1571	0,0683

Вычисляем погрешность:

$$\frac{1{,}367}{1-1{,}367}0{,}0003 = -0{,}001117$$

$$C_1 = 0{,}8855$$

$$C_2 = 1{,}1571$$

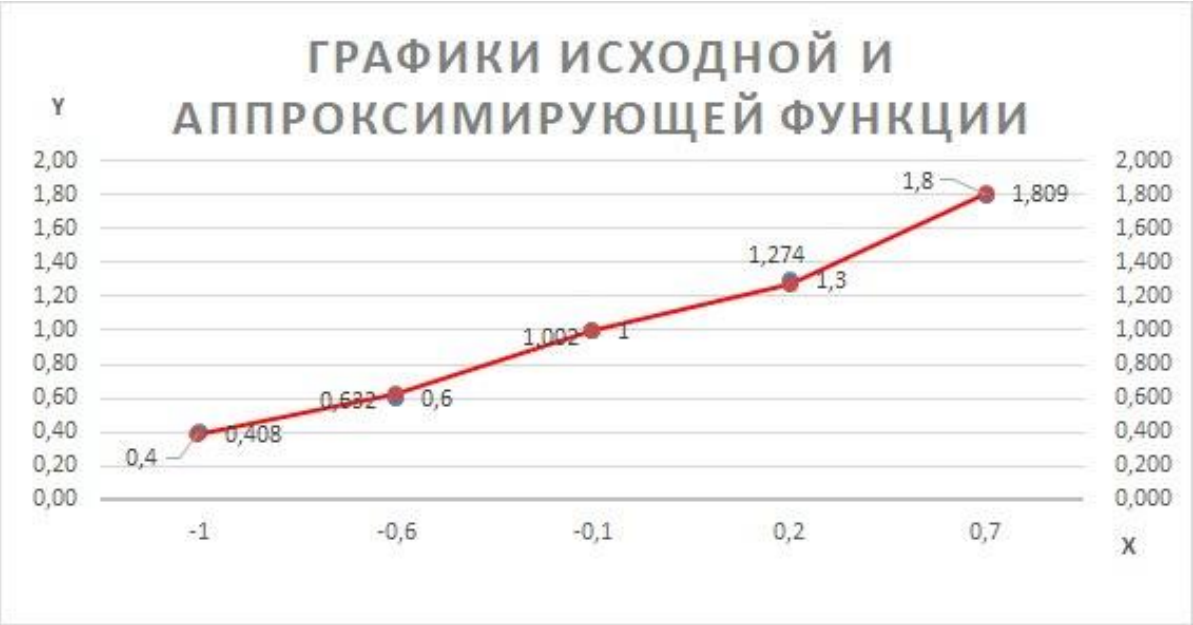
$$C_3 = 0{,}0683$$

$$\varphi\,(\text{x}) = 0{,}8855\text{X} + 1{,}1571 + 0{,}0683\,(3x^2 - 1)$$

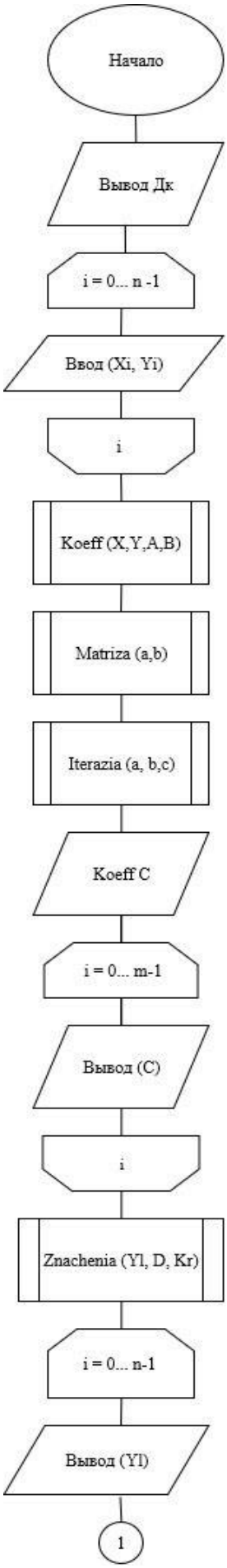
x_i	-1	-0,6	-0,1	0,2	0,7
y_i	0,4	0,6	1	1,3	1,8
$\varphi\,(x_i)$	0,408	0,631	1	1,274	1,809
δ_i	0,0083	0,031	0,002	0,026	0,009

$J_{min} = \text{J}(C_1^*, C_2^*, C_3^*) = 0{,}0083^2 + (0{,}031)^2 + 0{,}002^2 + 0{,}026^2 + (0{,}009)^2 = 0{,}001$

$|\delta_{max}| = 0{,}031$ при x=-0,6


ГРАФИКИ ИСХОДНОЙ И АППРОКСИМИРУЮЩЕЙ ФУНКЦИИ
Y
2,00
1,80
1,60
1,40
1,20
1,00
0,80
0,60
0,40
0,20
0,00
2,000
1,800
1,600
1,400
1,200
1,000
0,800
0,600
0,400
0,200
0,000
0,4
0,408
0,632
0,6
1,002
1
1,274
1,3
1,8
1,809
-1
-0,6
-0,1
0,2
0,7
X

4.Схемы алгоритмов и их описание.



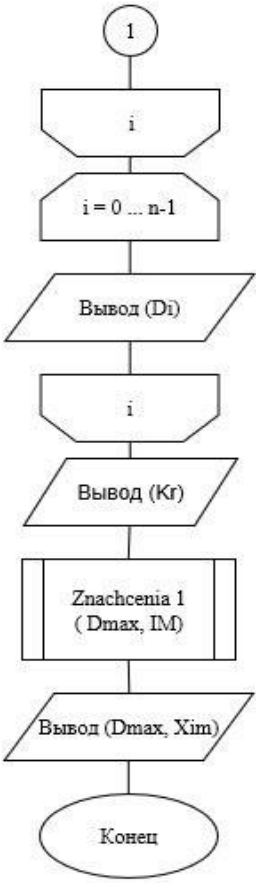
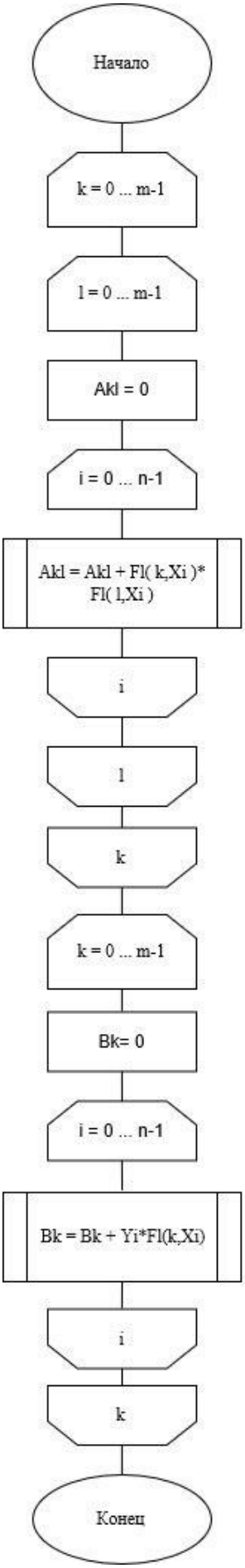
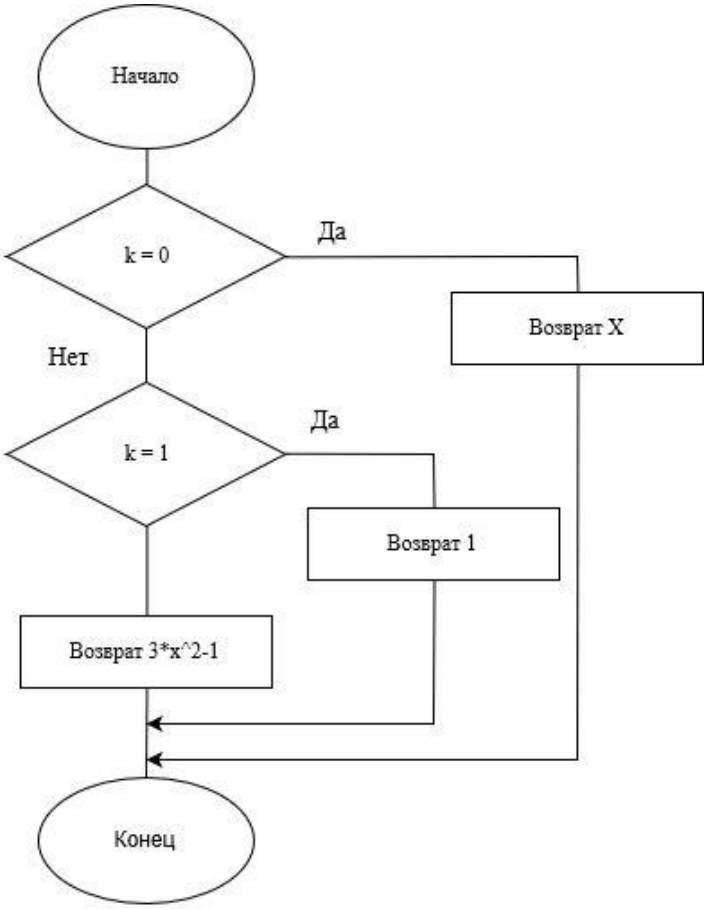


Схема алгоритма основной программы int main ()

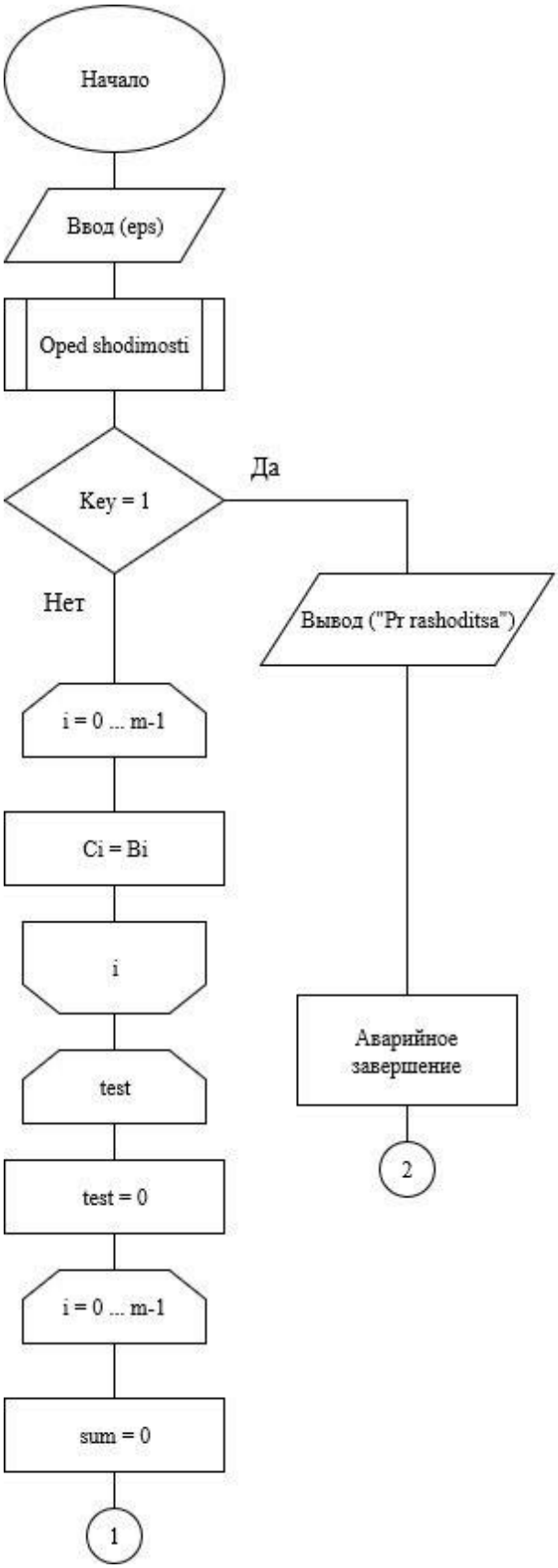
Схема алгоритма koeff

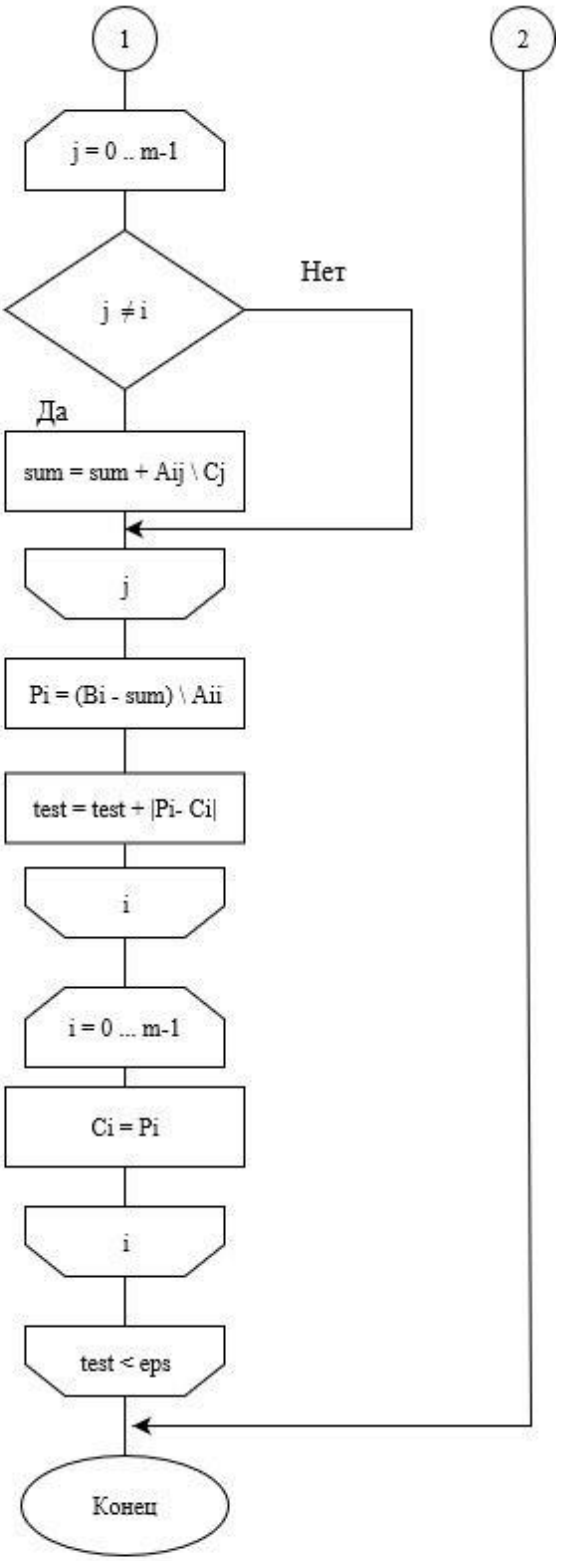


Fl (k,x[i])

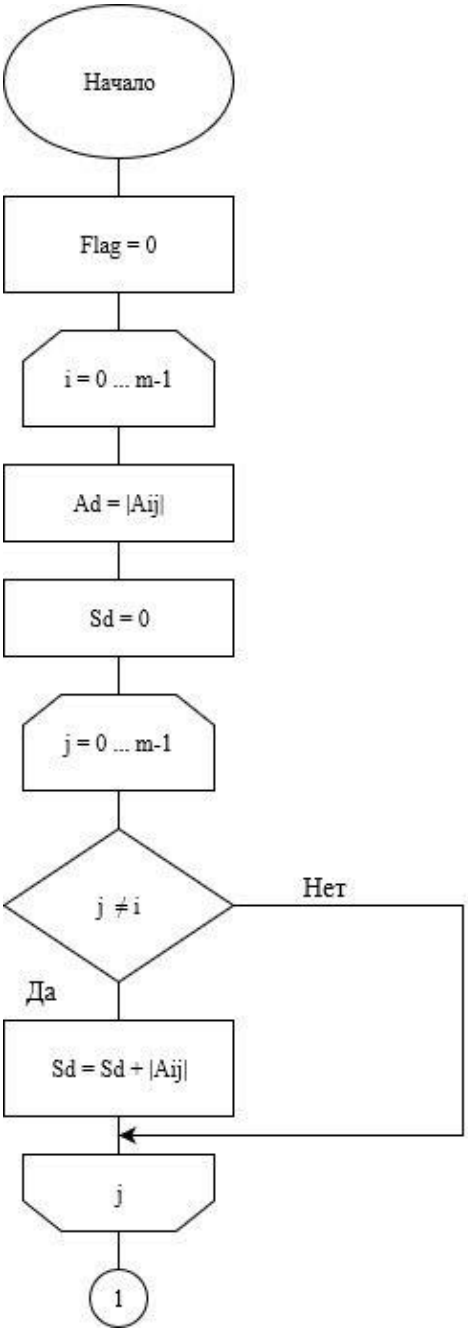


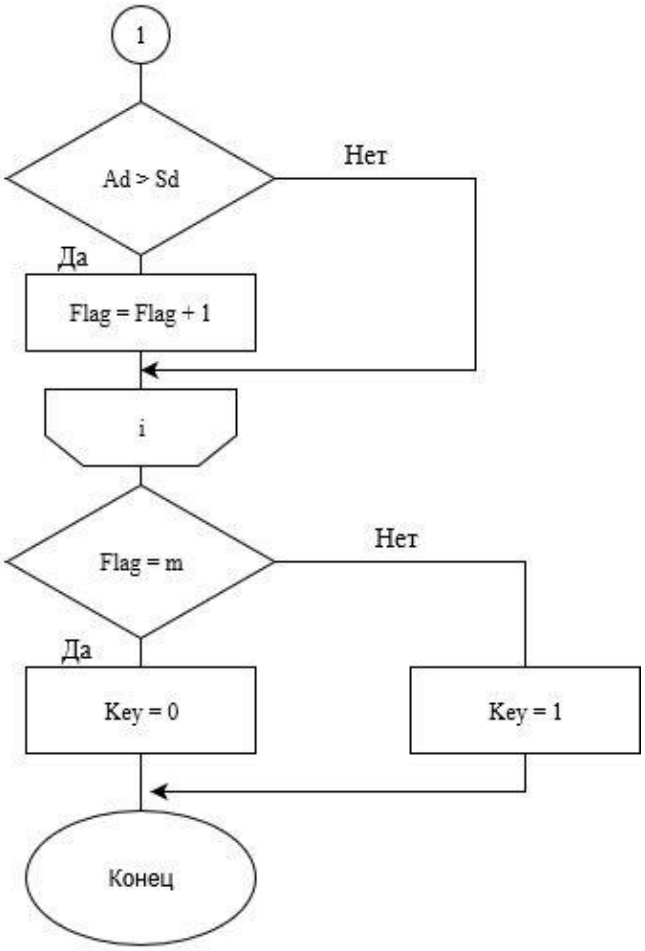
Iterazia


Начало
Ввод (eps)
Oped shodimosti
Key = 1
Да
Нет
Вывод ("Pr rashoditsa")
i = 0 ... m-1
Ci = Bi
i
test
test = 0
i = 0 ... m-1
sum = 0
1
Аварийное завершение
2

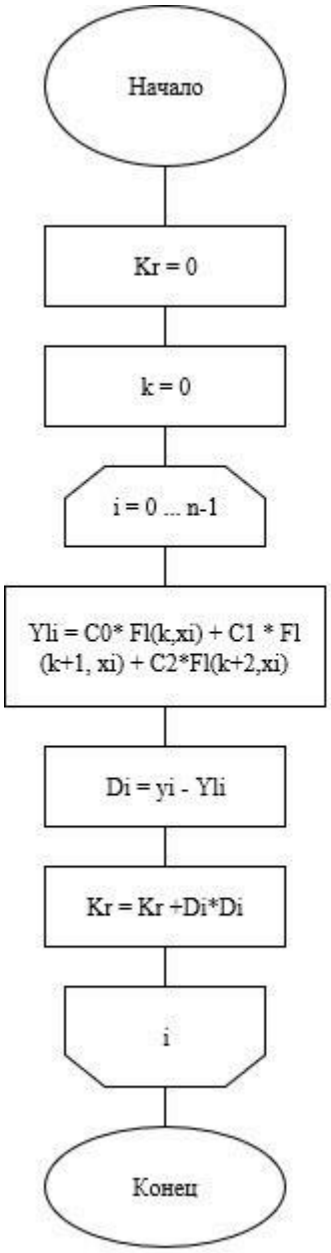

1
2
j = 0 .. m-1
j ≠ i
Нет
Да
sum = sum + Aij \ Cj
j
Pi = (Bi - sum) \ Aii
test = test + |Pi- Ci|
i
i = 0 ... m-1
Ci = Pi
i
test < eps
Конец

Opred shodimosti

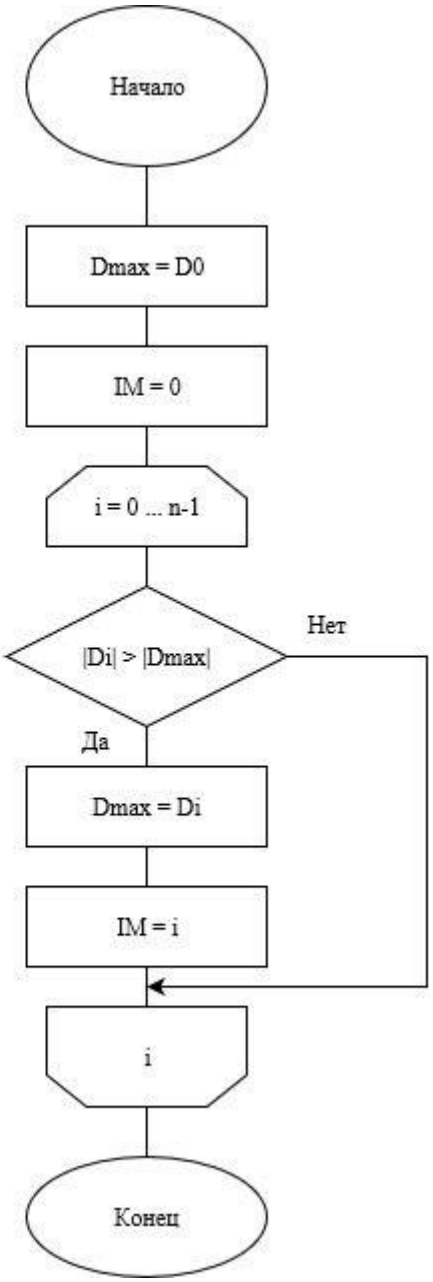




Znachenia (Yl, D, Kr)



Znachenia 1 (Dmax, IM)



5.Программа и результаты расчётов параметров на компьютере.

```
#include <conio.h>
#include <math.h>
#include <stdio.h>
#include <string.h>
#include <iostream>
#define n 5
#define m 3

using namespace std;
void vvod(float E[n])
{
          for(int i=0;i<n;i++)
          {
          cin>>E[i];
          }
          cout<<("\n");
}
 float  Fl(int k, float x)
{
     if (k==0)
        return x;
     if (k==1)
        return 1;
   if (k==2)
        return  3*pow(x,2)-1;
         else return 0;
}

void koeffA(float A[(m)][(m)],float X[n])
{
    int i,l,k;
        for (k=0; k<m; k++)
          {
                    for( l=0; l<m; l++)
                    {
                              A[k][l]=0;
                              for( i=0;i<n ;i++)
                              {
                                        A[k][l]=A[k][l]+Fl(k,X[i])*Fl(l,X[i]);
                              }
                    }
   }
}

void koeffB(float X[n],float Y[n],float B[(m)])
{
    int k,i;
          for (k=0; k<m; k++)
          {
          B[k]=0;
                for( i=0; i<n; i++)
```

```
        {
            B[k]=B[k]+Y[i]*Fl(k,X[i]);
        }
    }
}
int schod(float A[m][m])
{
        int Flag,i,j,Key;
        float Ad,Sd;
        Flag=0;
        for (i = 0;i <=(m-1);i++)
        {
            Ad=fabs(A[i][i]);
            Sd=0;
            for (j = 0;j <=(m-1);j++)
                if(j!=i)
                Sd+=fabs(A[i][j]);
            if(Ad>Sd)
            Flag=1;
        }
        if (Flag==1)
        Key=0;
        else
        Key=1;
        return Key;
}


void iterazia(float A[m][m],float B[m],float C[m])
{       int Key;
        float eps,test,sum,p[m];
cout<<"Vvedide eps \n";
        cin>>eps;
        Key=schod(A);
        if(Key==1)
        {
            cout<<("Pr rashoditsa\n");
            abort();
        }
        else
        {
            for(int i=0;i<m;i++)
            {
                C[i]=B[i];
            }
            do
            {
                test=0;
                for(int i=0;i<m;i++)
                {
                    sum=0;
                    for(int j=0;j<m;j++)
```

```
                                        if(j!=i)
                                                sum+=A[i][j]*C[j];
                                p[i]=(B[i]-sum)/A[i][i];
                                test+=fabs(p[i]-C[i]);
                                for(int i=0;i<m;i++)
                                        C[i]=p[i];

                        }
                }
                while (test>= eps);
        }
}

float approcs (float C[m],float Yl[n],float D[n],float X[n],float Y[n])
{
        float Kr;
        int k, i;
        Kr=0;
    k=0;
        for( i=0; i<n; i++)
        {
                Yl[i]=C[0]*Fl(k,X[i])+C[1]*Fl((k+1),X[i])+C[2]*Fl((k+2),X[i]);
                D[i]=fabs(Y[i]-Yl[i]);
                Kr=Kr+pow(D[i],2);
        }
        return Kr;
}

void krappr(float Y[m],float Yl[n],float D[n],float *Dmax,int *IM)
{
        int i;
        *Dmax=D[0];
        *IM=0;
        for(i=1; i<(n-1); i++)
        {
                if(fabs(D[i])>fabs(*Dmax))
                {
                        *Dmax=(D[i]);
                        *IM=i;
                }
        }
}
void vyvod(float E[n])
{
        for(int i=0;i<n;i++)
        {
         printf("%.3f ",E[i]);
        }
        printf("\n");
}
int main()
```

```
{
float X[n],Y[n], A[m][m],B[m],C[m],Yl[n],D[n],Dmax,Kr;
          int IM, i, j, k;
          cout<<("vvedite X\n");
          vvod(X);
          cout<<("vvedite Y\n");
          vvod(Y);
          koeffA(A,X);
          koeffB(X,Y,B);
            cout<<" matrica   A\n";

    for (i=0;i<m;i++)
  {
      for ( j=0;j<m;j++)
     cout<<" "<<A[i][j];
   cout<<"\n";
  }

    for (i=0; i<m; i++)
    {
     B[i]=0;
     for (k=0;k<n;k++)
     B[i]= B[i]+Fl(i,X[k])*Y[k];
     }
   cout<<"massiv  B\n";
    for (i=0;i<m;i++)

       cout<<" "<<B[i];
   cout<<"\n";
          iterazia(A,B,C);

          cout<<"Znachenie C: \n";
                    for (i=0;i<m;i++)
      {
          cout<<"   "<<C[i];
          cout<<"\n";
    }

          Kr=approcs(C,Yl,D,X,Y);
          krappr(Y,Yl,D,&Dmax,&IM);
   cout<<"Znachenie appr f(x):\n";
          vyvod(Yl);
          cout<<"Znacheniya otkloneniy: \n";
   vyvod(D);
          cout<<"Max pri = "<<Dmax<<"\n";
   cout<<"pri X= "<<X[IM]<<"\n";
          cout<<"Znachenie kriteriya= "<<Kr<<"\n";
          getch ();
          return 0;
}
```

```
vvedite X
-1.0 -0.6 -0.1 0.2 0.7

vvedite Y
0.4 0.6 1.0 1.3 1.8

 matricaa  A
 1.9 -0.8 -1.798
 -0.8 5 0.7
 -1.798 0.7 5.9426
massiva B
 0.66 5.1 -0.42
Vvedide eps
0.001
Znachenie C:
   0.893879
   1.15406
   0.0638363
Znachenie appr f(x):
0.388 0.623 1.003 1.277 1.810
Znacheniya otkloneniy:
0.012 0.023 0.003 0.023 0.010
Max pri = 0.0233431
pri X= 0.2
Znachenie kriteriya= 0.00131712
```

6.Заключение.

В результате проделанной работы был описан критерий аппроксимации, способ его минимизации, составлена система нормальных уравнений, параметры которой вычислили при помощи метода простой итерации, рассчитаны отклонения аппроксимирующей функции, а также максимальное по модулю отклонение. Расчёты составлены двумя способами:

- Подсчитанные вручную;
- Подсчитанные на ЭВМ, при помощи составленной программы.