**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Уфимский университет наук и технологий»**

**Кафедра** Высокопроизводительных вычислений и дифференциальных уравнений

**Дисциплина:** «Численные методы»

**Отчет по лабораторной работе №1**

**Тема:** «Приближение функций»

Группа МКН-318

Выполнил \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ Баймуратова А. А.

(дата) (подпись) (ФИО)

Проверил \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_ Феоктистов Б. А.

(дата) (подпись) (ФИО)

Уфа 2024

**Цель:** получить навык проведения вычислительного эксперимента, направленного на решение задач интерполирования и аппроксимации функций.

**Теоретическая часть**

***Построение интерполяционных многочленов***

***Задача 1. Интерполяционный многочлен Лагранжа на равномерной сетке***

Построение интерполяционного многочлена Лагранжа Ln(x) для произвольной степени n по известным значениям функции yi=f(xi), заданным на сетке узлов

производится по формуле

Оценка погрешности приближения функции находится следующим образом:

* + - 1. Задается равномерная сетка узлов
      2. Для каждого n=1…15 строится интерполяционный многочлен Лагранжа Ln(x) по значениям функции на заданной равномерной сетке узлов.
      3. Оценка погрешности приближения функции находится как  
            
         Оценка проводится численно посредством вычисления модуля ошибки приближений в узлах мелкой равномерной сетки, состоящей из ~105 узлов, с выбором максимального значения в качестве искомой оценки.
      4. Оптимальная степень n0, при которой погрешность минимальна, определяется при помощи построения графика зависимости от n.

***Задача 2. Интерполяционный многочлен Лагранжа на неравномерной сетке***

Построение сетки узлов, составленных из нулей многочлена Чебышева степени n0, найденной при решении задачи 1 производится по формуле

По построенной сетке узлов можно построить многочлен Лагранжа Ln0(x) степени n0 по аналогии с предыдущей задачей по формуле

где xi теперь узлы сетки, составленные из нулей многочлена Чебышева.

Оценка погрешности приближения функции ∆n0 производится методом, описанным в задаче 1.

Теоретическая минимальная оценка погрешности интерполяции многочленом Лагранжа представляется в виде

где

***Задача 3. Интерполяционный многочлен Ньютона***

Интерполяционный многочлен в форме Лагранжа не удобен для вычисления тем, что при увеличении числа узлов интерполяции приходится перестраивать весь многочлен заново.

Перепишем многочлен Лагранжа в другом виде:

где Pi(x) – многочлены Лагранжа степени i ≤ n.

Пусть Qi(x) = Pi(x) - Pi-1(x) (\*). Этот многочлен имеет степень i и обращается в ноль при x=x0, x=x1,…, x=xi-1. Поэтому он представим в виде:

где Ai – коэффициент при xi. Так как xi входит в Pi-1(x), то Ai совпадает с коэффициентом при xi в многочлене Pi(x). Таким образом, из определения Pi(x) получаем:

где

Перепишем формулу (\*) в виде

Рекуррентно выражая Pi(x) получаем окончательную формулу для многочлена:

Такое представление многочлена удобно для вычисления, потому что увеличение числа узлов на единицу требует добавления только одного слагаемого.

Оценка погрешности приближения функции ∆n0 производится методом, описанным в задаче 1.

***Тригонометрическая интерполяция***

***Задача 4. Тригонометрический многочлен***

Интерполяция функции , заданной своими значениями в узлах равномерной сетки, тригонометрическим многочленом Fn(x) степени n:

Коэффициенты ak и bk находятся следующим образом:

Оценка погрешности приближения функции ∆n0 производится методом, описанным в задаче 1.

***Наилучшее равномерное приближение***

***Задача 5. Многочлен наилучшего равномерного приближения***

Рассмотрим задачу нахождения многочлена наилучшего приближения степени n в случае, когда

Тогда и оценки сверху и снизу для En(f) совпадают:

Таким образом, многочленом наилучшего приближения оказывается интерполяционный многочлен Qn(x) с узлами интерполяции

Можно получить другое представление этого многочлена наилучшего приближения, записав его в виде

Выражение в правой части является многочленом степени n, поскольку коэффициент при xn+1 равен нулю. Точки образуют чебышевский альтернанс.

Для определения наименьшей степени n многочлена Qn, обеспечивающего приближение исходной функции f(x) с точностью, указанной в задании, пользоваться соотношением

Многочлен Pm(x), представляющий собой отрезок ряда Тейлора, приближает функцию f(x) с указанным в задании предельным уровнем погрешности ∆:

***Интерполяция сплайнами***

***Задача 6. Построение интерполирующего кубического сплайна***

Сплайн-функцией называют кусочно-полиноминальную функцию, определенную на отрезке [a,b] и имеющую на этом отрезке некоторое число непрерывных производных.

Пусть на отрезке [a,b] задана непрерывная функция f(x). Введем сетку

и обозначим

Коэффициенты многочлена на каждом интервале определяют из условий в узлах. Очевидно, в узлах многочлен должен принимать табличные значения функции:

Кроме того, на границе при x=x0 иx=xn ставятся условия

Будем искать кубический многочлен в виде

Из условия имеем

Число этих уравнений вдвое меньше числа неизвестных коэффициентов, поэтому для определенности задания нужны дополнительные условия. Для их получения вычислим первую и вторую производные многочлена:

И потребуем непрерывности этих производных во всех точках, включая узлы. Приравнивая во внутреннем узле правые и левые пределы производных, получим:

Недостающие два условия обычно получают из естественного предположения о нулевой кривизне графика на концах:

что соответствует свободно опущенным концам линейки.

Уравнения образуют систему линейных уравнений для определения 4N неизвестных коэффициентов. Эту систему можно решить методом Гаусса. Найдем коэффициенты:

Исключая величины и и по соответствующим уравнениям, увеличивая во втором случае индекс на единицу. Остается система линейных уравнений для коэффициентов , приводящаяся к виду:

Матрица этой системы трехдиагональная, т.е. ненулевыми в ней являются только элементы главной диагонали и двух соседних. Решается методом прогонки. После нахождения коэффициентов , находим остальные коэффициенты.

Оценка погрешности приближения функции ∆n производится методом, описанным в задаче 1.

**Построение интерполяционных многочленов**

***Задача 1.***

1. Написать вычислительную программу на языке программирования C++ для построения интерполяционного многочлена Лагранжа Ln(x) произвольной степени n по известным значениям функции yi=f(xi), заданным на сетке узлов

a=x0<x1<…<xn-1<xn=b.

1. Для каждого n=1,...15 построить интерполяционный многочлен Лагранжа Ln(x) по значениям функции на равномерной сетке узлов

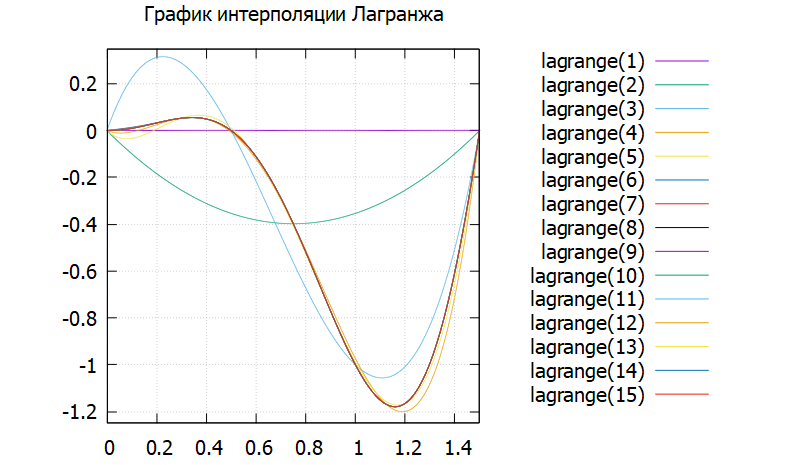
xi+1 =xi+h, x0=a, h=(b-a)/n

и найти оценку погрешности приближения функции

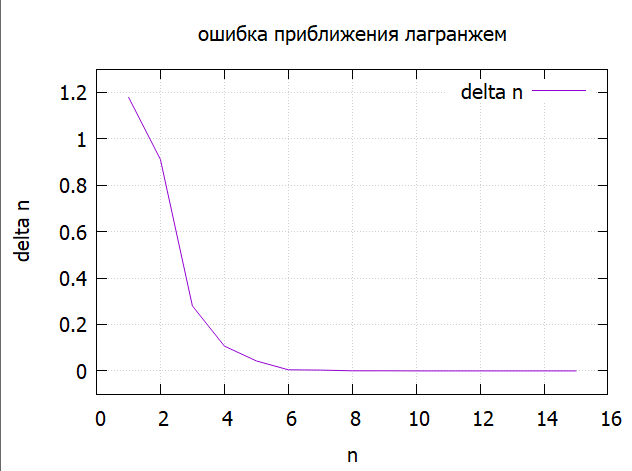
Δn=sup|f(x)-Ln(x)|, x∈[a,b].

Оценку Δn провести численно посредством вычисления модуля ошибки приближений |f(x)-Ln(x)| в узлах мелкой равномерной сетки, состоящей из ~105 узлов, с выбором максимального значения в качестве искомой оценки.

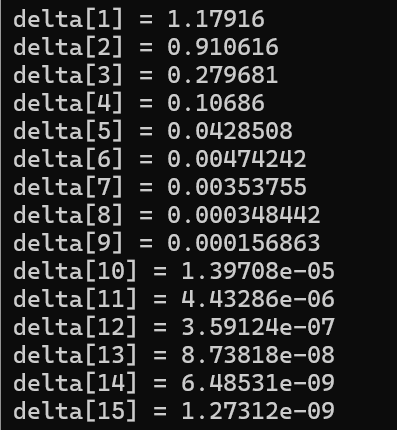
1. Построить график зависимости Δn от n определить оптимальную степень n0, при которой погрешность минимальна.
2. Построить график ошибки приближения f(x)-Ln0(x).



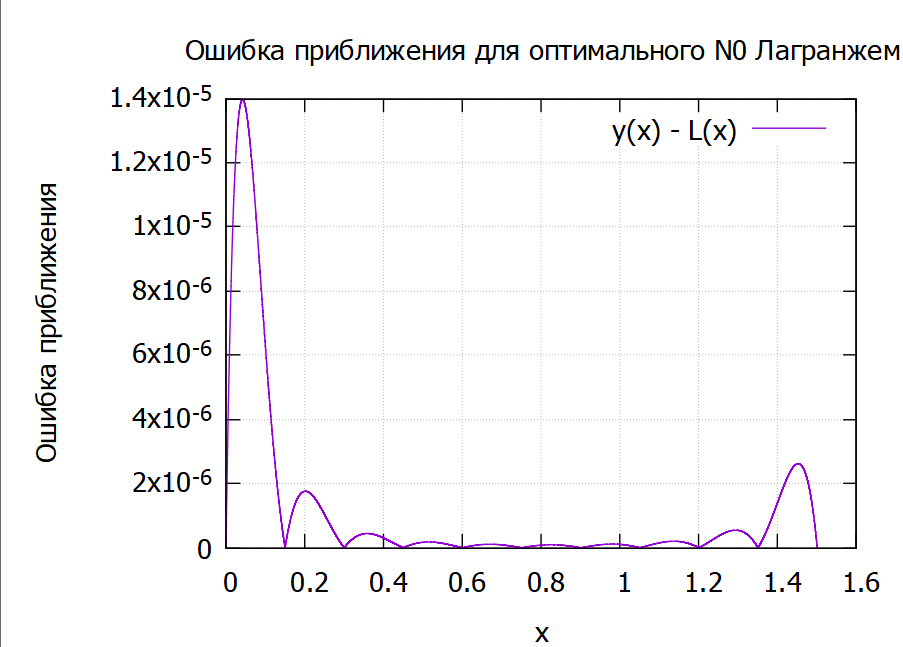
1. График интерполяции Лагранжа



1. График зависимости Δn от n



1. Ошибка приближения многочленом Лагранжа на сетке из 10^5 узлов с минимальными значениями



1. График ошибки приближения многочленом Лагранжа

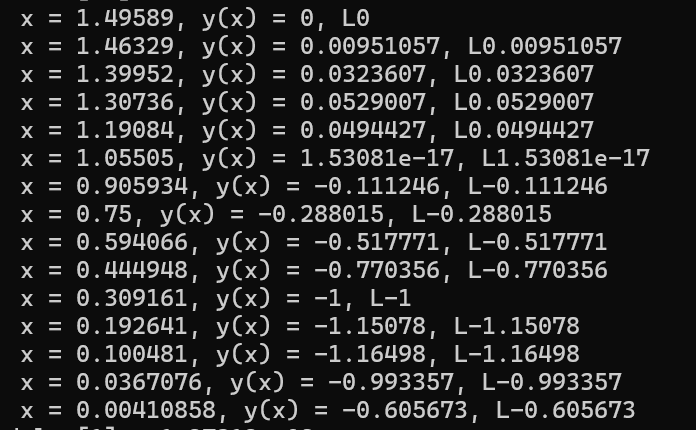
Таким образом, была проведена оценка погрешности приближения функции в узлах мелкой сетки из 10^5 узлов, откуда была найдена оптимальная степень N0 = 10, delta[10] = 1.39708-e05. Ошибка приближения функции многочленом Лагранжа при оптимальной степени N0 минимальна при значениях аргумента на отрезке от [0;1].

***Задача 2.***

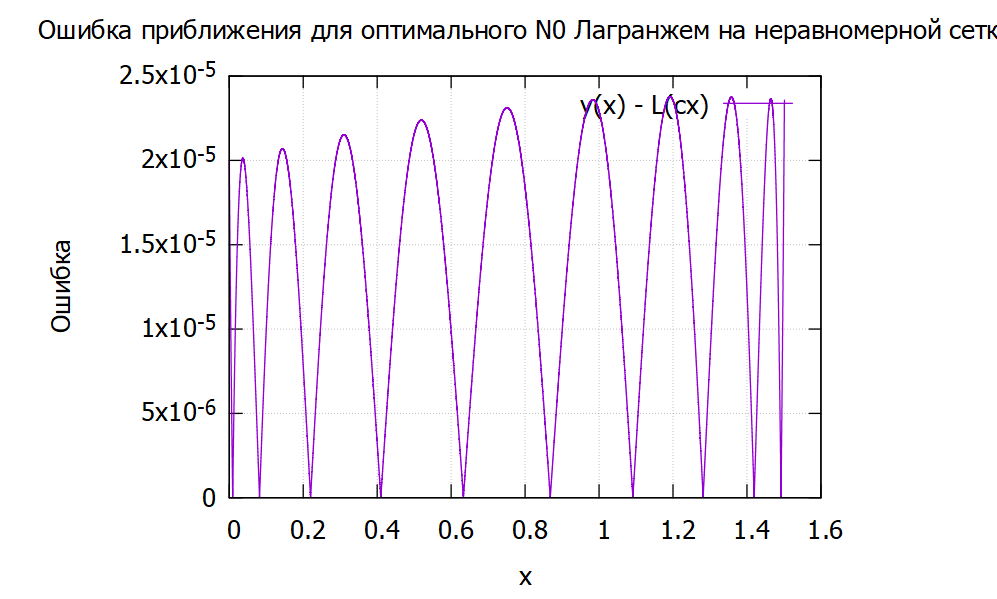
1. Построить сетку узлов, составленных из нулей многочлена Чебышева степени n0, найденной при решении предыдущей задачи:

Найти численные значения заданной функции f(x) в этих узлах: yi=f(xi).

1. С использованием написанной при решении Задачи 1 программы построить по этим данным многочлен Лагранжа Ln0(x) степени n0.
2. Найти оценку погрешности приближения функции Δn0 и сравнить ее с известной теоретической минимальной оценкой погрешности интерполяции многочленом Лагранжа.
3. Выполнить сравнение двух многочленов Лагранжа Ln0(x) на равномерной и неравномерной сетках, построенных в этой и предыдущей задачах.



1. Функция Лагранжа через узлы Чебышева

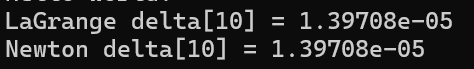


1. Разность значений исходной и функции через многочлены Лагранжа степени Н0 на неравномерной сетке

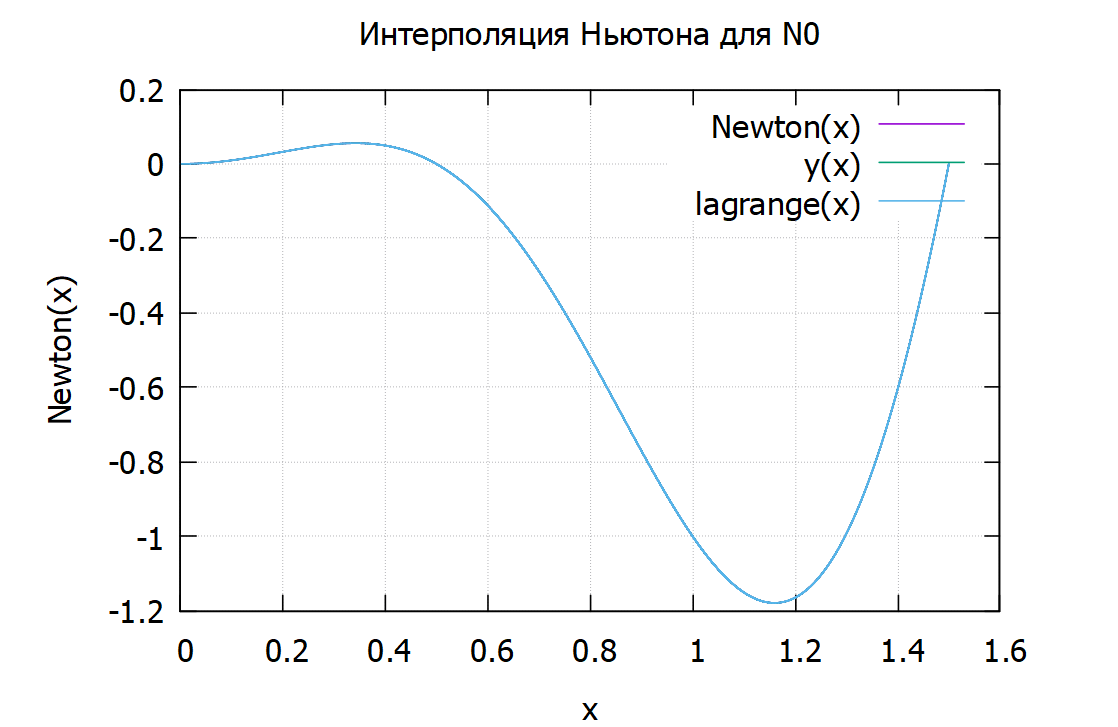
Таким образом, погрешностью приближения многочленом Чебышева при оптимальной степени N0=10 равна delta[10] = 2.37402e-05.

**Задача 3.**

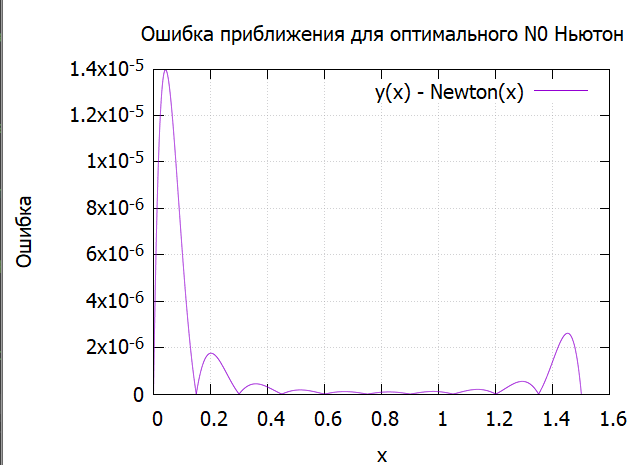
1. Написать вычислительную программу на языке программирования C++ для построения интерполяционного многочлена Ньютона порядка n0 (найдено при решении Задачи 1) на равномерной сетке через вычисление разделенных разностей.
2. Выполнить сравнение построенного многочлена Ньютона с аналогичным многочленом Лагранжа, построенного при решении задачи 1.



1. Ошибка приближения функции многочленом Лагранжа и Ньютона на оптимальной степени N0



1. График интерполяции Ньютона при степени N0=10



1. График ошибки приближения многочлена Ньютона для N0

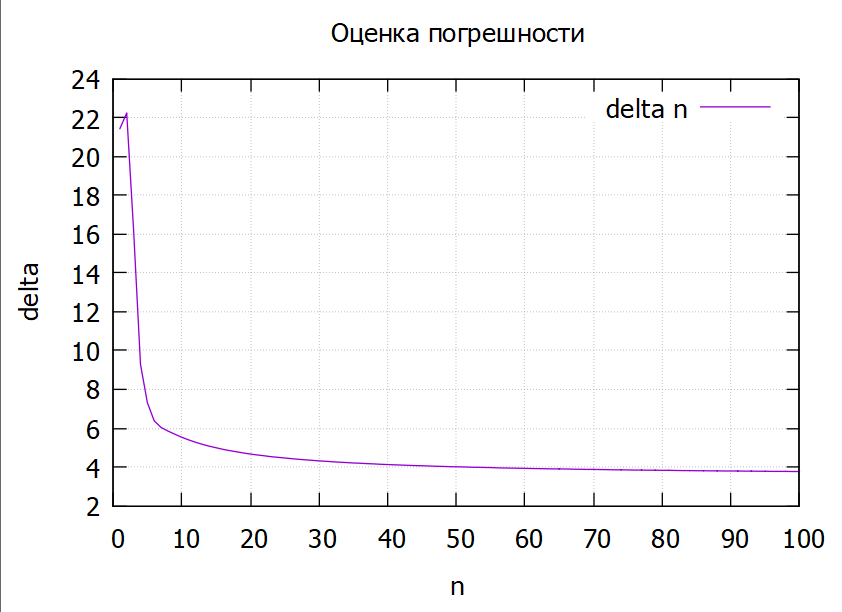
Таким образом, при интерполяции многочленом Ньютона ошибка приближения будет совпадать с ошибкой при приближении многочленом Лагранжа и delta[10] = 1.39708e-05.

**Задача 4.**

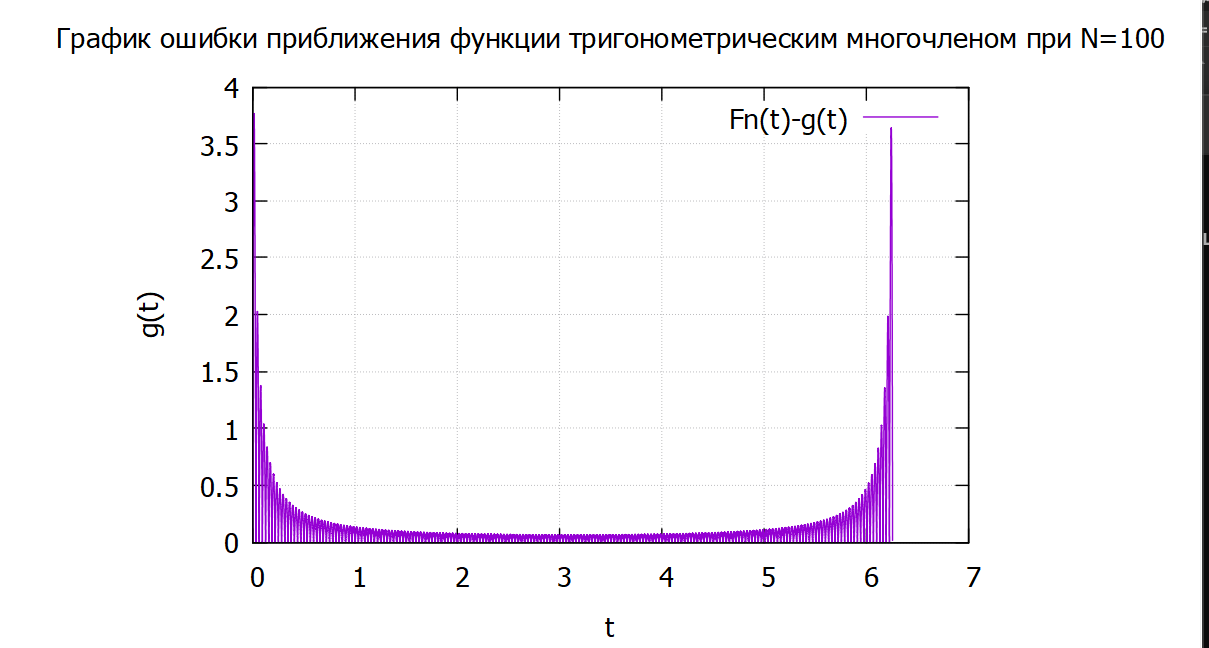
1. Написать вычислительную программу на языке программирования С++, осуществляющую интерполяцию функции g(t), t∈[0,2π], заданной своими значениями g(ti) (i=1,…,2n+1) в узлах равномерной сетки, тригонометрическим многочленом Fn(x) степени n:
2. Построить линейную замену переменных x=αt+β, переводящую заданный отрезок [a,b] в отрезок [0,2π]. Выполнить эту замену переменных в аргументе функции f(x): f(αt+β) = g(t).
3. С использованием написанной программы провести вычислительный эксперимент по нахождению минимальной степени m тригонометрического многочлена, обеспечивающего приближение функции с указанным в задании предельным уровнем погрешности Δ:

Оценку погрешности производить по способу, описанному в Задаче 1.

1. Построить график ошибки приближения функции многочленом.



1. График зависимости Δn от n при интерполяции тригонометрическим многочленом



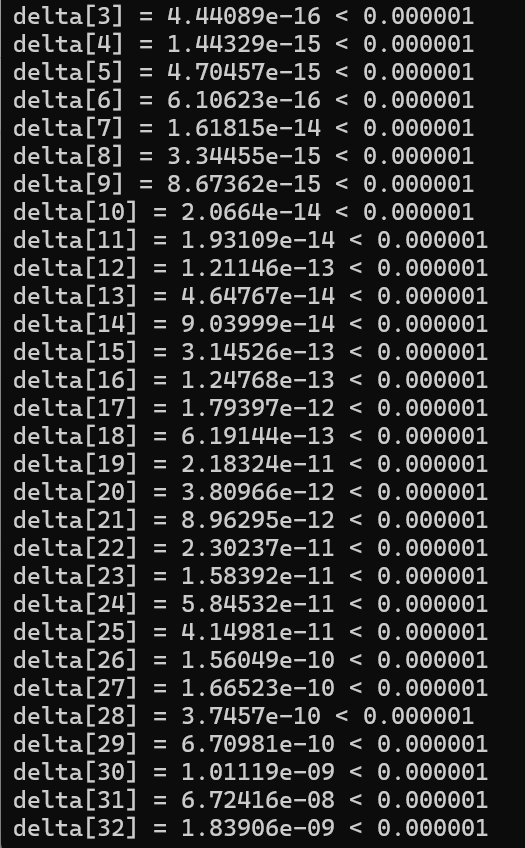
1. График ошибки приближения функции тригонометрическим многочленом при степени N=100

Таким образом, при приближении функции тригонометрическим многочленом при увеличении узлов интерполяции погрешность измерений падает и стремится к значению 3. Оптимальная погрешность не достигается засчёт эффект Гиббса в окрестностях точек разрыва функции.

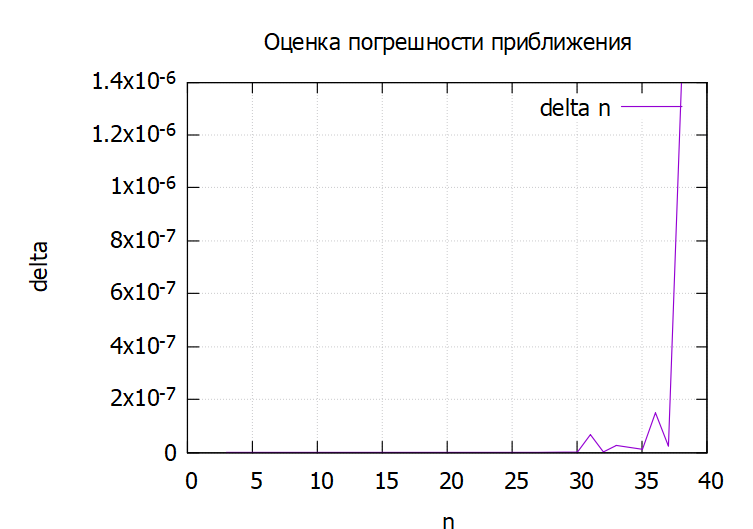
***Задача 6.***

1. Написать вычислительную программу на языке программирования C++ для построения интерполирующего кубического сплайна по значениям функции, известным в узлах равномерной сетки.
2. С использованием написанной программы провести вычислительный эксперимент по определению минимального количества узлов равномерной сетки, обеспечивающих построение интерполирующего сплайна для заданной функции с указанным в задании предельным уровнем погрешности. Погрешность интерполяции оценивать способом, описанным в Задаче 1.

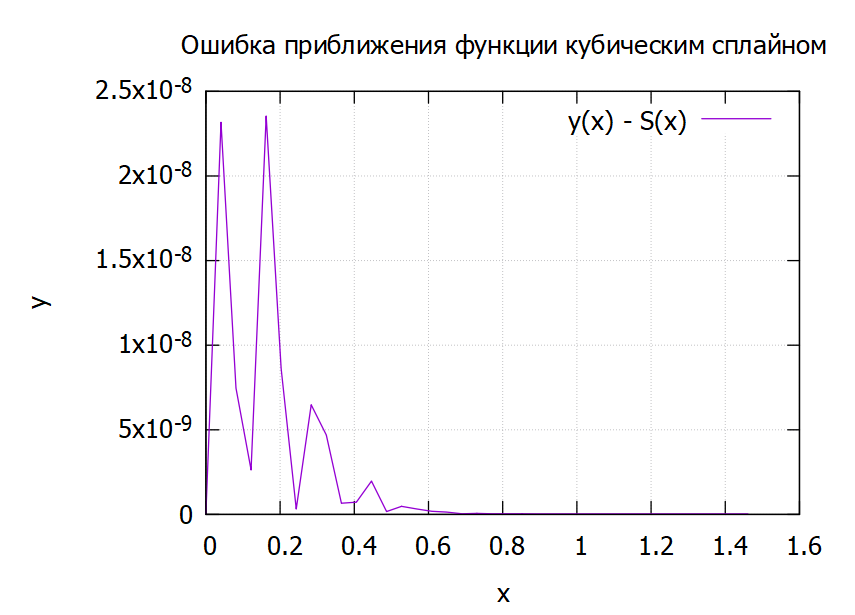
Построить график ошибки приближения заданной функции интерполирующим сплайном.



1. Оценка погрешности при интерполировании кубическими сплайнами



1. График оценки погрешности при интерполировании кубическими сплайнами



1. График ошибки приближения функции кубическим сплайном при N=38

Таким образом, при приближении функции кубическим сплайном при увеличении узлов интерполяции погрешность измерений растет. Оптимальная погрешность = 1e-6 перестает достигаться при степени N=38.

## Вывод

В ходе выполнения данной лабораторной работы были изучены методы решения задач для решения поставленных задач интерполирования и аппроксимации функций, а также были получены навыки проведения вычислительного эксперимента.

Для каждой поставленной задачи написана вычислительная программа на языке программирования С++.

## Список использованной литературы

1. Бахвалов Н.С., Жидков Н.П., Кобельков Г.М. Численные методы.
2. Калиткин Н.Н. Численные методы
3. Самарский А.А., Гулин А. В. Численные методы

## Приложение. Код программы

Lagrange.cpp

#include "Header.h"

double func1(double x) {

return x \* x \* cos(M\_PI \* x);

}

double\* grid\_step(double a, double b, int n) {

double h = (b - a) / n;

double\* x = new double[n + 1];

x[0] = a;

for (int i = 1; i <= n; ++i) {

x[i] = x[i - 1] + h;

}

return x;

}

double laGrange(double\* x, double X, int n)

{

double L = 0;

for (int i = 0; i <= n; i++)

{

double Pr = 1;

for (int j = 0; j <= n; j++)

{

if (i != j)

{

Pr = Pr \* (X - x[j]) / (x[i] - x[j]);

}

}

L = L + func1(x[i]) \* Pr;

}

return L;

}

double Newton(double\* x, double X, int n)

{

double y[16];

for (int i = 0; i <= n; i++) {

y[i] = func1(x[i]);

}

double Pr = 1;

double newton = y[0];

for (int j = 1; j <= n; j++)

{

for (int i = 0; i <= n - j; i++)

{

y[i] = (y[i + 1] - y[i]) / (x[i + j] - x[i]); //чем меньше степень тем больше игреков

}

Pr \*= (X - x[j - 1]);

newton = Pr \* y[0] + newton;

}

return newton;

}

int main()

{

std::cout << "Hello World!\n";

std::ofstream dataFile;

double a = 0.0;

double b = 1.5;

int N = 15;

//1.1

for (int n = 1; n <= N; n++) {

dataFile.open("lagrange\_data\_" + std::to\_string(n) + ".txt"); //интерполяиця лагранжа

double\* arg = grid\_step(a, b, n);

for (double X = a; X <= b; X += (b - a) / 1e5) {

double Y = laGrange(arg, X, n);

dataFile << X << " " << Y << " " << n << "\n";

}

dataFile << "\n";

delete[] arg;

dataFile.close();

}

//1.2 1.3

dataFile.open("lagrange\_deltaN.txt"); //ошибка приближения для каждого н

double delta = 0;

double maxdelta = 0;

for (int n = 1; n <= N; n++)

{

double\* x = grid\_step(a, b, n);

delta = 0;

maxdelta = 0;

for (double i = a; i <= b; i += (b-a)/(1e5))

{

delta = abs(func1(i) - laGrange(x, i, n));

if (delta > maxdelta)

{

maxdelta = delta;

}

}

//std::cout << "delta[" << n << "] = " << maxdelta << std::endl;

//if (n==10) std::cout << "LaGrange delta[" << n << "] = " << maxdelta << std::endl;

dataFile << n << " "<< maxdelta << std::endl;

delete[]x;

}

dataFile.close();

int N0 = 10;

//1.4

double \*x = grid\_step(a, b, N0);

dataFile.open("laGrangeErrN0.txt");//ошибка приближения для оптимального N0 Лагранжем и исходной функции

for (double i = a; i <= b; i += (b - a) / (1e5)) //тут стоит количество точек для интерполирования

{

dataFile << i << " " << abs(func1(i) - laGrange(x, i, N0)) << std::endl; //тут стоит оптимальная степень

}

dataFile.close();

//2.1

double cx[15];

double cy[15];

dataFile.open("chebyshev.txt");

for (int i = 0; i <= N0-1; i++)//i = 0..n0-1

{

cx[i] = ((b + a) / 2) + ((b - a) / 2) \* (cos(M\_PI \* (2 \* i + 1) / (2 \* N0)));

cy[i] = func1(cx[i]);

dataFile << cx[i] << " " << cy[i] << std::endl;

}

dataFile.close();

//2.2

dataFile.open("chebyshev\_lagrange.txt");//многочлен лагранжа степени н0 по узлам чебышева

for (int i = 0; i < N; i++)

{

//std::cout << " x = " << cx[i] << " L = " << laGrange(cx, cx[i], 15) << std::endl;

dataFile << cx[i] << " " << laGrange(cx, cx[i], N0) << std::endl;

}

dataFile.close();

//2.3

dataFile.open("chebyshev\_deltaN.txt");//оценка погрешности приближения дельта н на неравномерной сетке

maxdelta = 0;

delta = 0;

for (double i = a; i <= b; i += (b - a) / (1e5))

{

delta = abs(func1(i) - laGrange(cx, i, N0));

if (delta > maxdelta)

{

maxdelta = delta;

}

}

//std::cout << "delta[" << N0 << "] = " << maxdelta << std::endl;

dataFile << maxdelta << std::endl; //одно значение

dataFile.close();

//2.4

dataFile.open("cheb\_lagrange\_grid.txt");//сравнение двух многочленов Лагранжа Ln0(x) на равномерной и неравномерной сетках

for (double i = 0.0; i <= 1.5; i += (b - a) / (1e5))

{

//dataFile << i << " " << laGrange(x, i, 10) << " " << i << " " << laGrange(cx, i, 10) << std::endl;

dataFile << i << " " << abs(laGrange(x, i, 10) - laGrange(cx, i, 10)) << std::endl;

}

dataFile.close();

//2.4.2

dataFile.open("cheb242.txt");

for (double i = a; i <= b; i += (b - a) / (1e5-15))

{

dataFile << i << " " << abs(func1(i) - laGrange(cx, i, N0)) << std::endl;

//std::cout << "x = " << i << ", y(x) = " << func1(i) << ", L(x) = " << laGrange(cx, i, N0) << std::endl;

}

dataFile.close();

//3.1

dataFile.open("newtonN0.txt");//ньютон для оптимального N0

double newt[11];

for (double i = a; i <= b; i+=(b-a)/1e5)

{

double nt = Newton(x, i, N0);

double lt = laGrange(x, i, N0);

dataFile << i << " " << nt << " " << func1(i) << " " << lt << std::endl;

}

dataFile.close();

//3.2

delta = 0;

maxdelta = 0;

dataFile.open("newtonErrN0.txt");//Ошибка приближения Ньютона для оптимального н

for (double i = a; i <= b; i += (b - a) / (1e5))

{

delta = abs(func1(i) - Newton(x, i, N0));

if (delta > maxdelta)

{

maxdelta = delta;

}

//dataFile << i << " " <<abs(func1(i) - Newton(x, i, N0)) << std::endl;

//dataFile << i << abs(laGrange(x, i, N0) - Newton(x, i, N0)) << std::endl;

//std::cout << "y[" << i << "] - N[" << i << "] = " << abs(func1(i) - Newton(x, i, 15)) << std::endl;

}

//std::cout << "Newton delta[10] = " << maxdelta << std::endl;

dataFile << maxdelta << std::endl;

dataFile.close();

std::ofstream gnuplot("sd.gp");

gnuplot << "set grid\n";

//1.1

/\*gnuplot << "set title 'График интерполяции Лагранжа'\n";

gnuplot << "set xlabel 'x'\n";

gnuplot << "set ylabel 'y'\n";

gnuplot << "set xrange [0:1.5]\n";

gnuplot << "set yrange [-1.25:0.35]\n";

gnuplot << "set key outside\n";

gnuplot << "set lmargin 1\n";

gnuplot << "set rmargin 23\n";

gnuplot << "set bmargin 2\n";

gnuplot << "set tmargin 2\n";

gnuplot << "plot ";

for (int i = 1; i <= 15; ++i) {

gnuplot << "'lagrange\_data\_" << i << ".txt' using 1:2 with lines title 'lagrange(" << i << ")',";

}\*/

//1.2 1.3

/\*gnuplot << "set title 'ошибка приближения лагранжем'\n";

gnuplot << "set xlabel 'n'\n";

gnuplot << "set ylabel 'delta n'\n";

//gnuplot << "set xrange [1:15]\n";

gnuplot << "set yrange [-0.1:1.3]\n";

gnuplot << "plot 'lagrange\_deltaN.txt' with lines title 'delta n'\n";\*/

//1.4

/\*gnuplot << "set title 'Ошибка приближения для оптимального N0 Лагранжем'\n";//разница на узлах с исх функцией = 0

gnuplot << "set xlabel 'x'\n";

gnuplot << "set ylabel 'Ошибка приближения'\n";

gnuplot << "plot 'laGrangeErrN0.txt' with lines title 'y(x) - L(x)'\n";\*/

//2.4

/\*gnuplot << "set title 'РАзница значений Лагранжа на равномерной и неравномерной сетке'\n";

gnuplot << "set xlabel 'x'\n";

//gnuplot << "set xrange [0:1.5]\n";

gnuplot << "set ylabel 'y'\n";

//gnuplot << "plot 'cheb\_lagrange\_grid.txt' using 1:2 with lines title 'L n0(x)', 'cheb\_lagrange\_grid.txt' using 3:4 with lines title 'Ln0(cx)\n";

gnuplot << "plot 'cheb\_lagrange\_grid.txt' with lines title 'L n0(x) - L n0(cx)\n";\*/

//2.4.2

/\*gnuplot << "set title 'Ошибка приближения для оптимального N0 Лагранжем на неравномерной сетке'\n";//разница на узлах с исх функцией = 0

gnuplot << "set xlabel 'x'\n";

gnuplot << "set ylabel 'Ошибка'\n";

gnuplot << "plot 'cheb242.txt' with lines title 'y(x) - L(cx)'\n";\*/

//3.1

/\*gnuplot << "set title 'Интерполяция Ньютона для N0'\n";

gnuplot << "set xlabel 'x'\n";

gnuplot << "set ylabel 'Newton(x)'\n";

//gnuplot << "plot 'newtonN0.txt' with lines title 'Newton(x)'\n";

gnuplot << "plot 'newtonN0.txt' using 1:2 with lines title 'Newton(x)', 'newtonN0.txt' using 1:3 with lines title 'y(x)', \

'newtonN0.txt' using 1:4 with lines title 'lagrange(x)'\n";\*/

//3.2

/\*gnuplot << "set title 'Ошибка приближения для оптимального N0 Ньютон'\n";

gnuplot << "set xlabel 'x'\n";

gnuplot << "set ylabel 'Ошибка'\n";

//gnuplot << "set yrange [-0.5e-15:1.7e-15]\n";

gnuplot << "plot 'newtonErrN0.txt' with lines title 'Ln0(x) - Newton(x)'\n";\*/

gnuplot.close();

system("gnuplot -p sd.gp");

}

Trig.cpp

#include "Header.h"

using namespace std;

int main() {

double at = 0.0;

double bt = 2 \* M\_PI;

ofstream dataFile, dataFile2;

dataFile.open("func\_fnt.txt");

double delta = 0;

double maxdelta = 0;

dataFile2.open("trig\_delta.txt");

for (double n = 1; n <= 100; ++n) {

vector <double> t(2 \* n + 2);

t[0] = 0;

for (int i = 1; i <= 2 \* n + 1; i++) {

t[i] = (2 \* M\_PI \* (i - 1)) / (2 \* n + 1);

}

double A0 = 0;

double sum = 0;

for (int i = 1; i <= 2 \* n + 1; i++) {

sum += func1(t[i]);

}

A0 = sum / (2 \* n + 1);

vector <double> ak(n + 1);

vector <double> bk(n + 1);

for (double j = 1; j <= n; j++) {

double suma = 0;

double sumb = 0;

for (int i = 1; i <= 2 \* n + 1; i++) {

suma += func1(t[i]) \* cos(j \* t[i]);

sumb += func1(t[i]) \* sin(j \* t[i]);

}

ak[j] = 2 \* suma / (2 \* n + 1);

bk[j] = 2 \* sumb / (2 \* n + 1);

}

int index = 0;

delta = 0;

maxdelta = 0;

for (double j =0; j <= t[2 \* n + 1]; j += 2\*M\_PI/ 1e5) { //до t[2\*n+1]

double fnt = A0;

for (double i = 1; i <= n; i++) {

fnt += ak[i] \* cos(i \* j) + bk[i] \* sin(i \* j);

}

//double gt = func1(j);

delta = abs(fnt - func1(j));

if (delta > maxdelta) maxdelta = delta;

//dataFile << j << " " << fnt << " " << func1(j) << "\n";

dataFile << j << " " << abs(fnt - func1(j)) << "\n";

}

if (maxdelta <= 1e-6) {

index = n;

break;

}

dataFile2 << n << " " << maxdelta << std::endl;

cout << "delta[" << n << "] = " << maxdelta << "\n";

}

dataFile.close();

dataFile2.close();

std::ofstream gnuplot("sd.gp");

gnuplot << "set grid\n";

//4.3

gnuplot << "set xlabel 'n'\n";

gnuplot << "set ylabel 'delta'\n";

//gnuplot << "set xrange [3:4]\n";

//gnuplot << "set yrange [0:10]\n";

gnuplot << "set title 'Оценка погрешности'\n";

gnuplot << "plot 'trig\_delta.txt' with lines title 'delta n' \n";

//4.4

//gnuplot << "set title 'График ошибки приближения функции тригонометрическим многочленом при N=100'\n";

//gnuplot << "set xlabel 't'\n";

//gnuplot << "set ylabel 'g(t)'\n";

//gnuplot << "plot 'func\_fnt.txt' with lines title 'Fn(t)-g(t)'\n";//ошибка приближения функции многочленом

gnuplot.close();

system("gnuplot -p sd.gp");

}

Spline.cpp