CАНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

**Лабораторная работа по выч.математике №5**

*«Приближение функции»*

Выполнил: Припадчев Артём

группа 2125

Проверил: Шипилов П.А.

2013 г.

**Задание:** составить программу, выполняющую приближение сложной функции с помощью интерполяционного полинома Лагранжа. В этой постановке узлы интерполяции определяются как корни полинома Чебышева.

**Описание**

**Интерполяцио́нный многочле́н Лагра́нжа** — многочлен минимальной степени, принимающий данные значения в данном наборе точек. Для n+1 пар чисел (x_0, y_0), (x_1, y_1)\dots ,(x_n, y_n), где все x_j различны, существует единственный многочлен L(x) степени не более n, для которого L(x_j) = y_j.

В простейшем случае (n=1) — это линейный многочлен, график которого — прямая, проходящая через две заданные точки.

Лагранж предложил способ вычисления таких многочленов:

L(x) = \sum_{i=0}^n y_i l_i(x)

где базисные полиномы определяются по формуле:

l_i(x)=\prod_{j=0, j\neq i}^{n} \frac{x-x_j}{x_i-x_j} = \frac{x-x_0}{x_i-x_0} \cdots \frac{x-x_{i-1}}{x_i-x_{i-1}} \frac{x-x_{i+1}}{x_i-x_{i+1}} \cdots \frac{x-x_{n}}{x_i-x_{n}}\,\!

l_i(x) обладают следующими свойствами:

* являются многочленами степени n
* l_i(x_i)=1
* l_i(x_j)=0 при j\ne i

Отсюда следует, что L(x), как линейная комбинация l_i(x), может иметь степень не больше n, и L(x_i)=y_i, что и требовалось доказать.

Заметим также, что для некоторых функций f(x), определенных на отрезке xО [a,b], интерполяционный многочлен Pn(x) построенный по значениям f(xi) в равноотстоящих узлахxi=a+ih, x0=a, xn=b, h=(b-a)/n с ростом числа узлов n, вовсе не будет иметь убывающую погрешность интерполирования. Это обусловлено тем, что равноотстоящие узлы не являются лучшими с точки зрения уменьшения погрешности интерполирования. Если имеется возможность выбора узлов интерполирования, то их следует выбирать так, чтобы обеспечить минимум погрешности интерполяции min|Rn(x)|. Такому требованию на отрезке [-1,1] удовлетворяют узлы полиномов Чебышева http://vm.psati.ru/online-vmath/pics/Image173.gif

вычисляемые по формуле http://vm.psati.ru/online-vmath/pics/Image174.gif k=0,1,2,...,n.

В случае произвольного отрезка [*a,b*] узлы интерполяции вычисляют по формулам

http://vm.psati.ru/online-vmath/pics/Image175.gif*k*=0,1,2,...*,n.*

Интерполяционный полином построенный по таким узлам является алгебраическим полиномом наилучшего приближения.

**Код программы**

public enum InterpolationType

{

Equidistant,

Optimal

}

public class PolynomialInterpolation

{

private void NodesInitialisation(int countofnodes, InterpolationType \_type)

{

\_nodes = new double[countofnodes];

switch (\_type)

{

case InterpolationType.Equidistant:

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

{

\_nodes[i] = a + i \* (b - a) / (countofnodes - 1);

}

break;

case InterpolationType.Optimal:

for (int k = 0; k < countofnodes; k++)

{

\_nodes[k] = Math.Cos(Math.PI \* (2 \* k + 1) / (2 \* countofnodes)) \* (b - a) / 2 + (b + a) / 2;

}

break;

}

}

private PolynomialInterpolation(double a, double b, int n, InterpolationType type)

{

this.a = a;

this.b = b;

NodesInitialisation(n, type);

}

public PolynomialInterpolation(double a, double b, int n, InterpolationType type, MainWindow.Calc f)

: this(a, b, n, type)

{

\_sourcefunction = f;

\_values = new double[\_nodes.Length];

for (int i = 0; i < \_nodes.Length; i++)

{

\_values[i] = \_sourcefunction(\_nodes[i]);

}

}

public PolynomialInterpolation(double a, double b, int n, InterpolationType type, double[] values)

: this(a, b, n, type)

{

\_values = values;

}

public double[] Nodes

{

get

{

return \_nodes;

}

}

public double LagrangePolynomial(double x)

{

double sum = 0;

if (\_sourcefunction != null)

{

for (int i = 0; i < \_nodes.Length; i++)

{

sum += \_sourcefunction(\_nodes[i]) \* BasisPolynomial(i, x);

}

}

else

{

for (int i = 0; i < \_values.Length; i++)

{

sum += \_values[i] \* BasisPolynomial(i, x);

}

}

return sum;

}

public double BasisPolynomial(int k, double x)

{

double result = 1;

for (int i = 0; i < \_nodes.Length; i++)

{

if (i != k)

{

result \*= (x - \_nodes[i]) / (\_nodes[k] - \_nodes[i]);

}

}

return result;

}

public double[] \_values;

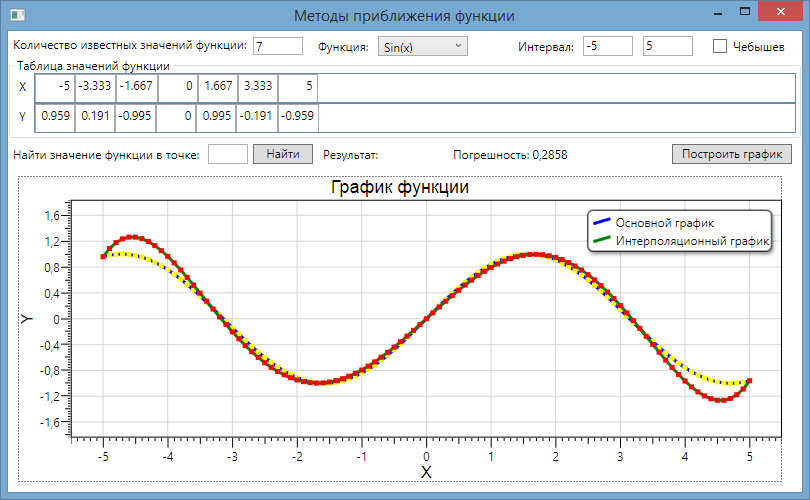
private MainWindow.Calc \_sourcefunction;

public double[] \_nodes;

private double a, b;

}

**Пример работы**





**Вывод:** в процессе выполнения лабораторной работы была рассмотрена интерполяция функции полиномом Лагранжа и сделаны следующие выводы:

* в узлах интерполяции погрешность вычислений равна нулю
* при интерполяции выгодно использовать четное число n узлов, симметрично расположенных относительно точки x\* (x1, xn)
* если при увеличении кол-ва известных точек функции не удается обеспечить требуемую точность интерполяции, то целесообразнее не увеличивать n, а уменьшать шаг между соседними узлами, т.е. использовать (если это возможно) таблицу значений yi=fi(x) с меньшим шагом по x. (*замеч.* данное правило работает и в обратную сторону, т.е. если не удается достичь требуемой точности интерполяции уменьшая шаг, то целесообразнее увеличивать (если это возможно) кол-во известных точек функции)
* Лучше выбирать не равномерно распределенные узлы интерполяции, а узлы, являющиеся решениями уравнения Чебышева. В результате бОльшее кол-во узлов будет сконцентрировано на концах отрезка интерполирования, на которых обычно и находится самая высокая погрешность.