НАЦИОНАЛЬНЫЙ

ИССЛЕДОВАТЕЛЬСКИЙ

УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 - Программная инженерия

Дисциплина – Вычислительная математика

Лабораторная работа №5

Вариант №11

Выполнил: Мухсинов С.П

Группа: P3217

Преподаватель: Малышева Т.А

Санкт-Петербург

2024

**Цель работы**

Решить задачу интерполяции, найти значения функции при заданных значениях аргумента, отличных от узловых точек.

**Задание**

**Обязательное задание (до 80 баллов)**

**Вычислительная реализация задачи:**

1. Выбрать из табл. 1 заданную по варианту таблицу 𝑦 = 𝑓(𝑥) (таблица 1.1 –таблица 1.5).
2. Построить таблицу конечных разностей для заданной таблицы. Таблицу отразить в отчете.
3. Вычислить значения функции для аргумента 𝑋1 (см. табл.1), используя первую или вторую интерполяционную формулу Ньютона. Обратить внимание какой конкретно формулой необходимо воспользоваться.
4. Вычислить значения функции для аргумента 𝑋2 (см. табл. 1), используя первую или вторую интерполяционную формулу Гаусса. Обратить внимание какой конкретно формулой необходимо воспользоваться.

**Программная реализация задачи:**

1. Исходные данные задаются тремя способами:
2. в виде набора данных (таблицы x,y), пользователь вводит значения с клавиатуры.
3. в виде сформированных в файле данных (подготовить не менее трех тестовых вариантов).
4. на основе выбранной функции, из тех, которые предлагает программа, например, sin 𝑥. Пользователь выбирает уравнение, исследуемый интервал и количество точек на интервале (не менее двух функций).
5. Сформировать и вывести таблицу конечных разностей.
6. Вычислить приближенное значение функции для заданного значения аргумента, введенного с клавиатуры, указанными методами (см. табл. Сравнить полученные значения.
7. Построить графики заданной функции с отмеченными узлами интерполяции и интерполяционного многочлена Ньютона/Гаусса (разными цветами).
8. Программа должна быть протестирована на различных наборах данных, в том числе и некорректных.

**Используемые формулы и методы**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

**Вычислительная реализация**

**Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание**

Построим таблицу конечных разностей

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № |  |  |  |  |  |  |  |  |
| 0 | 0,25 | 1,2557 | 0,9207 | 0,0247 | -0,0437 | 1,0756 | -4,1277 | 10,1917 |
| 1 | 0,30 | 2,1764 | 0,9454 | -0,0190 | 1,0319 | -3,0521 | 6,0640 |  |
| 2 | 0,35 | 3,1218 | 0,9264 | 1,0129 | -2,0202 | 3,0119 |  |  |
| 3 | 0,40 | 4,0482 | 1,9393 | -1,0073 | 0,9917 |  |  |  |
| 4 | 0,45 | 5,9875 | 0,9320 | -0,0156 |  |  |  |  |
| 5 | 0,50 | 6,9195 | 0,9164 |  |  |  |  |  |
| 6 | 0,55 | 7,8359 |  |  |  |  |  |  |

находится в левой половине отрезка, поэтому воспользуемся формулой Ньютона для интерполирования вперед:

Воспользуемся многочленом Гаусса для расчета

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № |  |  |  |  |  |  |  |  |
| -3 | 0,25 | 1,2557 | 0,9207 | 0,0247 | -0,0437 | 1,0756 | -4,1277 | 10,1917 |
| -2 | 0,30 | 2,1764 | 0,9454 | -0,0190 | 1,0319 | -3,0521 | 6,0640 |  |
| -1 | 0,35 | 3,1218 | 0,9264 | 1,0129 | -2,0202 | 3,0119 |  |  |
| 0 | 0,40 | 4,0482 | 1,9393 | -1,0073 | 0,9917 |  |  |  |
| 1 | 0,45 | 5,9875 | 0,9320 | -0,0156 |  |  |  |  |
| 2 | 0,50 | 6,9195 | 0,9164 |  |  |  |  |  |
| 3 | 0,55 | 7,8359 |  |  |  |  |  |  |

**Изображение выглядит как текст, Шрифт, рукописный текст, линия

Автоматически созданное описание**

**Программная реализация**

**Многочлен Лагранжа**

public class Lagrange extends Method{  
  
 protected Lagrange(double point, double[] x, double[] y) {  
 super(point, x, y);  
 }  
  
 @Override  
 public Result solve() {  
 double ln = 0;  
 for(int i = 0; i < n; i++){  
 ln += y[i] \* (numerator(i)/denominator(i));  
 }  
 return new Result(point, ln, MethodType.*LAGRANGE*);  
 }  
  
 private double numerator(int i){  
 double s = 1;  
 for(int j = 0; j < n; j++){  
 if(j != i){  
 s \*= (point - x[j]);  
 }  
 }  
 return s;  
 }  
  
 private double denominator(int i){  
 double s = 1;  
 for(int j = 0; j < n; j++){  
 if(j != i){  
 s \*= (x[i] - x[j]);  
 }  
 }  
 return s;  
 }  
}

**Многочлен Ньютона с разделенными разностями**

public class NewtonDivided extends Method{  
 protected NewtonDivided(double point, double[] x, double[] y) {  
 super(point, x, y);  
 }  
  
 @Override  
 public Result solve() {  
 double N = 0;  
 double p = 1;  
 for(int i = 0; i < n; i++){  
 N += f(i, 0)\*p;  
 p \*= (point - x[i]);  
 }  
 return new Result(point, N, MethodType.*NEWTON\_DIVIDED*);  
 }  
  
 private double f(int degree, int i){  
 if(degree == 0){  
 return y[i];  
 }  
 return (f(degree - 1, i + 1) - f(degree - 1, i))/(x[i + degree] - x[i]);  
 }  
}

**Многочлен Ньютона с конечными разностями**

public class NewtonFinite extends Method{  
 protected NewtonFinite(double point, double[] x, double[] y) {  
 super(point, x, y);  
 }  
  
 @Override  
 public Result solve() {  
 printTable();  
 final double h = x[1] - x[0];  
 if((x[0] <= point)&&(point <= (x[n-1] + x[0])/2)){  
 System.*out*.println("Используется первая формула");  
 double t = (point - x[0])/h;  
 return solveLeft(t);  
 }else{  
 System.*out*.println("Используется вторая формула");  
 double t = (point - x[n-1])/h;  
 return solveRight(t);  
 }  
 }  
  
 private double f(int degree, int i){  
 if(degree == 0){  
 return y[i];  
 }  
 return f(degree - 1, i + 1) - f(degree - 1, i);  
 }  
  
 public void printTable(){  
 for(int i = 0; i < n; i++){  
 for(int j = 0; j < n - i;j++){  
 System.*out*.print(f(j, i) + " ");  
 }  
 System.*out*.println();  
 }  
 }  
  
 public Result solveLeft(double t){  
 double N = y[0];  
 double p = 1;  
 for(int i = 0; i < (n-1); i++){  
 p \*= (t - i)/((i+1));  
 N += f(i + 1, 0)\*p;  
 }  
 return new Result(point, N, MethodType.*NEWTON\_FINITE*);  
 }  
  
 public Result solveRight(double t){  
 double N = y[n-1];  
 double p = 1;  
 for(int i = 0; i < n - 1; i++){  
 p \*= (t + i)/((i+1));  
 N += f(i + 1, (n - 2 - i))\*p;  
 }  
 return new Result(point, N, MethodType.*NEWTON\_FINITE*);  
 }  
}

**Результаты работы программы**

**Входные данные:**

**Изображение выглядит как Шрифт, снимок экрана, текст, число

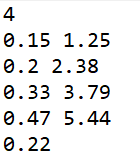
Автоматически созданное описание**

**Вывод:**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

**Входные данные:**

****

**Вывод:**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

**Вывод**

В ходе выполнения лабораторной работы были изучены различные методы интерполяции функций, а также была написана программа, вычисляющая приближенное значение функции в заданной точке, используя формулы интерполяции Лагранжа и Ньютона.