Федеральное государственное автономное образовательное
учреждение высшего образования Национальный
исследовательский университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №5
Вычислительная математика

Вариант: №1

Группа	P3208
Студент	Абдуллин И.Э.
Преподаватель	Машина Е.А.

Цель работы:

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

Выполнение

1) Вычислительная реализация задачи

Таблица из варианта:

x	y	x1	x2
0,25	1,2557	0,251	0,402
0,3	2,1764	0,512	0,372
0,35	3,1218	0,255	0,405
0,4	4,0482	0,534	0,384
0,45	5,9875	0,272	0,445
0,5	6,9195	0,551	0,351
0,55	7,8359	0,294	0,437

Таблица конечных разностей:

x_i	y_i	Δy_i	$\Delta^2 y_i$	$\Delta^3 y_i$	$\Delta^4 y_i$	$\Delta^5 y_i$	$\Delta^6 y_i$
0,25	1,2557	0,9207	0,0247	-0,0437	1,0756	-4,1277	10,1917
0,3	2,1764	0,9454	-0,019	1,0319	-3,0521	6,064	
0,35	3,1218	0,9264	1,0129	-2,0202	3,0119		
0,4	4,0482	1,9393	-1,0073	0,9917			
0,45	5,9875	0,932	-0,0156				
0,5	6,9195	0,9164					
0,55	7,8359						

Вычисление формулой Ньютона:

Интерполирования вперед для:

x = {0,251; 0,255; 0,272; 0,294}

$$N_6(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_0 + \frac{t(t-1)(t-2)}{3!}\Delta^3 y_0 + \frac{t(t-1)(t-2)(t-3)}{4!}\Delta^4 y_0 + \frac{t(t-1)(t-2)(t-3)(t-4)}{5!}\Delta^5 y_0 + \frac{t(t-1)(t-2)(t-3)(t-4)(t-6)}{6!}\Delta^6 y_0$$

$$t = \frac{x - x_0}{h}$$

x	t	$N_6(x)$
0.251	0.02	1,22013
0.255	0.1	1,12252
0.272	0.44	1,26425
0.294	0.88	1.98

Интерполирования назад для:

x = {0,534; 0,551; 0,512}

$$N_6(x) = y_6 + t\Delta y_5 + \frac{t(t+1)}{2!}\Delta^2 y_4 + \frac{t(t+1)(t+2)}{3!}\Delta^3 y_3 + \frac{t(t+1)(t+2)(t+3)}{4!}\Delta^4 y_2 + \frac{t(t+1)(t+2)(t+3)(t+4)}{5!}\Delta^5 y_1 + \frac{t(t+1)(t+2)(t+3)(t+4)(t+6)}{6!}\Delta^6 y_0$$

$$t = \frac{x - x_i}{h}$$

x	t	$N_6(x)$
0.534	0.68	7,54265
0.551	0.02	7.94313
0.512	0.24	7.13944

Вычисление формулой Гаусса:

Интерполирование второй формулой для x = {0,372; 0,384; 0,351}

$$P_6(x) = y_0 + t\Delta y_{-1} + \frac{t(t+1)}{2!}\Delta^2 y_{-1} + \frac{(t+1)t(t-1)}{3!}\Delta^3 y_{-2} + \frac{(t+2)(t+1)t(t-1)}{4!}\Delta^4 y_{-2} + \frac{(t+2)(t+1)t(t-1)(t-2)}{5!}\Delta^5 y_{-3} + \frac{(t+3)(t+2)(t+1)t(t-1)(t-2)}{6!}\Delta^6 y_{-3}$$

$$t = \frac{x - x_0}{h}$$

x	t	$N_6(x)$
0,372	-0.056	3.15171
0,384	-0.32	3.46121
0,351	0.24	3.10282

Интерполирование первой формулой для x = {0,402; 0,405; 0,445; 0,437}

$$P_6(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_{-1} + \frac{(t+1)t(t-1)}{3!}\Delta^3 y_{-1} + \frac{(t+1)t(t-1)(t-2)}{4!}\Delta^4 y_{-2} + \frac{(t+2)(t+1)t(t-1)(t-2)}{5!}\Delta^5 y_{-2} + \frac{(t+2)(t+1)t(t-1)(t-2)(t-3)}{6!}\Delta^6 y_{-3}$$

x	t	$N_6(x)$
0,402	0.004	4.15755
0,405	0.01	4.31572
0,445	0.09	5.85526
0,437	0.074	5.62025

Код программы

```
@InterpolationMethod(id = 1, name = "Лагранжа")
public static double lagrange(double[] x, double[] y, double value) {
 double result = 0;
 for (int i = 0; i < x.length; i++) {
   double c1 = 1, c2 = 1;
   for (int j = 0; j < x.length; j++) {
     if (i != j) {
       c1 *= value - x[j];
       c2 *= x[i] - x[j];
     }
   }
   result += y[i] * c1 / c2;
 }
 return result;
}

private static double[][] subNewtonCreateTable(double[] y) {
 int n = y.length;
```

```java
 double[][] table = new double[n][n];
 for (int i = 0; i < n; i++) {
   table[i][0] = y[i];
 }
 for (int j = 1; j < n; j++) {
   for (int i = 0; i < n - j; i++) {
     table[i][j] = table[i + 1][j - 1] - table[i][j - 1];
   }
 }
 return table;
}


@InterpolationMethod(id = 3, name = "Ньютона с конечными разностями")
public static double newtonFiniteDifferences(double[] x, double[] y, double
value) {
 double[][] table = subNewtonCreateTable(y);
 double t, result;
 if (value <= x[x.length - 1]) {
   int x0 = 0;
   for (int i = x.length - 1; i >= 0; i--) {
     if (value >= x[i]) {
       x0 = i;
       break;
     }
   }
   t = (value - x[x0]) / (x[1] - x[0]);
   result = table[x0][0];
   for (int i = 1; i < table[x0].length; i++) {
     double temp = t;
     for (int yi = 1; yi < i; yi++) temp *= (t - yi);
     result += (temp * table[x0][i]) / factorial(i);
   }
 } else {
   t = (value - x[x.length - 1]) / (x[1] - x[0]);
   result = table[x.length - 1][0];
   for (int i = 1; i < x.length; i++) {
```

```java
      double temp = t;
      for (int yi = 1; yi < i; yi++) temp *= (temp + yi);
      result += (temp * table[x.length - i - 1][i]) / factorial(i);
    }
  }
  return Math.round(result);
}
```

```java
@InterpolationMethod(id = 4, name = "Стирлинга")
public static double stirling(double[] x, double[] y, double value) {
  if (y.length % 2 == 0) {
    System.out.println("Четное число узлов. Формула Стирлинга не применяется");
    return NaN;
  }
  double[][] table = createTableGauss(y);
  int mid = y.length / 2;
  double h = x[1] - x[0];
  double t = (value - x[mid]) / h;
  if (Math.abs(t) > 0.25) {
    System.out.println("Результат по формуле Стирлинга содержит большую погрешность");
  }
  double result = y[mid];
  for (int i = 1; i <= mid; i++) {
    double mul = 1;
    for (int j = 1; j < i; j++) {
      mul *= (t * t - j * j);
    }
    result +=
        t
            * mul
            * (table[2 * i - 1][mid - i] + table[2 * i - 1][mid - i + 1])
            / (2 * factorial(2 * i - 1));
    result += t * t * mul * (table[2 * i][mid - i]) / factorial(2 * i);
  }
  return result;
}

@InterpolationMethod(id = 5, name = "Бесселя")
public static double bessel(double[] x, double[] y, double value) {
  if (y.length % 2 != 0) {
```

```
  System.out.println("Нечетное число узлов. Формула Бесселя не применяется");
  return NaN;
}
double[][] table = createTableGauss(y);
int mid = y.length / 2;
double h = x[1] - x[0];
double t = (value - x[mid]) / h;
if (Math.abs(t) < 0.25 || Math.abs(t) > 0.75) {
  System.out.println("Результат по формуле Бесселя содержит большую погрешность");
}
double result = (y[mid] + y[mid + 1]) / 2 + (t - 0.5) * table[1][mid];
for (int i = 2; i < mid; i++) {
  double mul = 1;
  for (int j = 0; j < i; j++) {
    mul *= (t + Math.pow(-1, j) * j);
  }
  int n = i - 1;
  result +=
      mul * (table[2 * n][mid - n] + table[2 * i - 2][mid - n + 1]) / (2 * factorial(2 * n));
  result += (t - 0.5) * mul * (table[2 * n + 1][mid - n]) / factorial(2 * n + 1);
}
return result;
}
```

Вывод

В ходе лабораторной работы я научился интерполировать функции с помощью метода Лагранжа, метода Ньютона с конечными разностями, метода Бесселя и Стирлинга.