Национальный исследовательский университет ИТМО
Факультет программной инженерии и компьютерной
Дисциплина «Вычислительная математика»

Отчет
по лабораторной работе №5
Вариант 25

Выполнил:

Туляков Е.В

P32101

Преподаватель:

Рыбаков С. Д.

Санкт-Петербург
2023 г.

Цель лабораторной работы: решить задачу интерполяции, найти значения функции при заданных значениях аргумента, отличных от узловых точек.

Многочлен Лагранжа:

$$L_n(x) = \sum_{i=0}^{n} y_i\, l_i(x)$$

$$Ln(x) = \sum_{i=0}^{n} y_i \frac{(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}$$

$$L_n(x) = \sum_{i=0}^{n} y_i \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x-x_j}{x_i-x_j}$$

Многочлен Ньютона:

Разделенные разности *k*-го порядка определяются через разделенные разности порядка *k*-1:

$$f(x_i, x_{i+1}, \ldots, x_{i+k}) = \frac{f(x_{i+1}, \ldots, x_{i+k}) - f(x_i, x_{i+k-1})}{x_{i+k} - x_i}$$

Используя понятие разделенной разности интерполяционный многочлен Ньютона можно записать в следующем виде:

$$N_n(x) = f(x_0) + f(x_0,\ x_1)\cdot(x-x_0) + f(x_0,\ x_1, x_2)\cdot(x-x_0)\cdot(x-x_1) + \cdots + f(x_0,\ x_1, \ldots, x_n)\cdot(x-x_0)\cdot(x-x_1)\ldots(x-x_{n-1})$$

$$N_n(x) = f(x_0) + \sum_{k=1}^{n} f(x_0, x_1, \ldots, x_k) \prod_{j=0}^{k-1}(x-x_j)$$

Вычислительная реализация:

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
0.30	2.1764	0.9454	-0.0190	1.0319	-3.-521	6.0640	
0.35	3.1218	0.9264	1.0129	-2.0202	3.0119		
0.40	4.0482	1.9393	-1.0073	0.9917			
0.45	5.9875	0.9320	-0.0156				
0.50	6.9195	0.9164					
0.55	7.8359						

Ньютон:

$$t = \frac{x - x_0}{h} = \frac{0.255 - 0.25}{0.05} = 0.1$$

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

$$N_n(0.255) = 1.1225$$

Гаус:

$$t = \frac{x - a}{h} = \frac{0.405 - 0.4}{0.5} = 0.1$$

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

$$P_n(x) = 4.2097$$

Программная реализация:

```
public class Lagrange extends Polynomial{
    @Override
    public double execute() {
        double x = getX();
        ArrayList<Double[]> values = getValues();
        double result = 0;
        for(int i = 0; i < values.size(); i++){
            double intermediateResult = 1;
            for(int j = 0; j < values.size(); j++){
                if (i == j) continue;
                intermediateResult *= (
                        (x - values.get(j)[0]) /
                        (values.get(i)[0] - values.get(j)[0])
                );
            }
            result += intermediateResult * values.get(i)[1];
        }
        return result;
    }
}
```

```java
public class Newton extends Polynomial{
    @Override
    public double execute() {
        double x = getX();
        ArrayList<Double[]> values = getValues();
        double result = values.get(0)[1];
        for(int i = 2; i < values.size() + 1; i++){
            List<Double[]> mas = values.subList(0, i);
            double production =
                        calculateDividedDifference(mas);
            for (int j = 0; j < i - 1; j++){
                production *= (x - values.get(j)[0]);
            }
            result += production;
        }
        return result;
    }
```

Результат:

```
Choose type of input:
1. Table
2. File
3. Function
1
Enter number of values:
n > 2
7
Enter table:
x1 y1
x2 y2
... ...
0.25 1.2557
0.30 2.1764
0.35 3.1218
0.40 4.0482
0.45 5.9875
0.50 6.9195
0.55 7.8359
Enter X coordinate to find an approximate value:
0.6

———————RESULT———————
Lagrange:  28,99600
Newton:    28,99600

————————————————————Finite Difference Table————————————————————
0,250   1,256   0,921   0,025   -0,044  1,076   -4,128  10,192
0,300   2,176   0,945   -0,019  1,032   -3,052  6,064   null
0,350   3,122   0,926   1,013   -2,020  3,012   null    null
0,400   4,048   1,939   -1,007  0,992   null    null    null
0,450   5,988   0,932   -0,016  null    null    null    null
0,500   6,920   0,916   null    null    null    null    null
0,550   7,836   null    null    null    null    null    null
```

Вывод:

Во время выполнения лабораторной работы я изучил работу с многочленами Лагранжа и многочленами Ньютона. Эффективность алгоритмов примерно одинаковая. Но при добавлении новых точек, функцию Лагранжа приходится пересчитывать полностью, тогда как при использовании формулы Ньютона достаточно добавить к уже существующему многочлену только одно слагаемое.
В сравнении с рассмотренными методами большую точность интерполяции можно получить применением методов сплайн – интерполяции.