Федеральное государственное автономное
образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский университет
ИТМО»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ
ТЕХНИКИ

ЛАБОРАТОРНАЯ РАБОТА № 5

по дисциплине

‘ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА’

Вариант №11

Выполнила
Студентка группы Р32151
Ярусова Анна Александровна
Преподаватель:
Машина Екатерина Алексеевна

Санкт-Петербург,
2023

Цель

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

Задание

1. № варианта определяется как номер в списке группы согласно ИСУ.
2. **Программная реализация задачи:**
   - Исходные данные задаются тремя способами:
     1. в виде набора данных (таблицы x,y), пользователь вводит значения с клавиатуры;
     2. в виде сформированных в файле данных;
     3. 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).
   - Сформировать и вывести таблицу конечных разностей;
   - Вычислить приближенное значение функции для заданного значения аргумента, введенного с клавиатуры, указанными методами (Многочлен Лагранжа, Ньютона с конечными разностями). Сравнить полученные значения;
   - Построить графики заданной функции с отмеченными узлами интерполяции и интерполяционного многочлена Ньютона;
   - Программа должна быть протестирована на различных наборах данных, в том числе и некорректных.
3. **Вычислительная реализация задачи:**
   - Выбрать из табл. 1 заданную по варианту таблицу y = f(x);
   - Построить таблицу конечных разностей для заданной таблицы. Таблицу отразить в отчете;
   - Вычислить значения функции для аргумента X1 (см. табл.1), используя первую или вторую интерполяционную формулу Ньютона. Обратить внимание какой конкретно формулой необходимо воспользоваться;

- Вычислить значения функции для аргумента X2(см. табл. 1), используя первую или вторую интерполяционную формулу Гаусса. Обратить внимание какой конкретно формулой необходимо воспользоваться;
- Подробные вычисления привести в отчете.

Описание метода, расчетные формулы

Лагранж

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

Ньютон

Интерполяция вперёд:

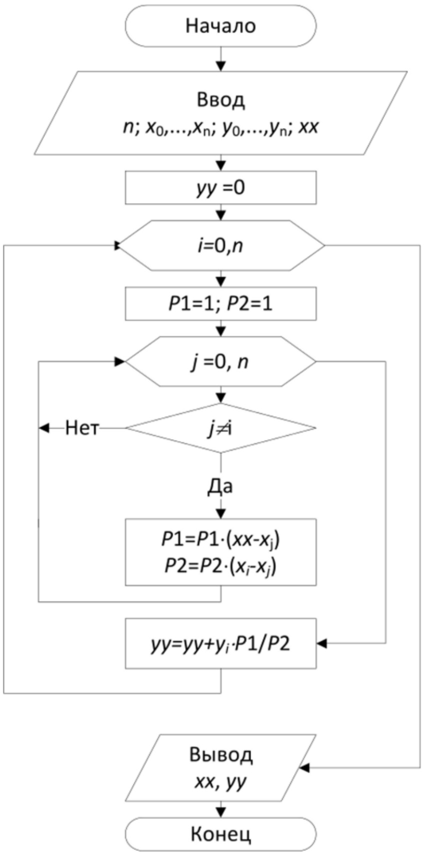
$$N_n(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_0 + \dots + \frac{t(t-1)\dots(t-n+1)}{n!}\Delta^n y_0$$

Интрополяция назад:

$$N_n(x) = y_n + t\Delta y_{n-1} + \frac{t(t+1)}{2!}\Delta^2 y_{n-2} + \dots + \frac{t(t+1)\dots(t+n-1)}{n!}\Delta^n y_0$$

Блок-схема

Лагранж



<u>Ньютон</u>

Вычислительная часть

X1 = 0,255
X2 = 0,405

<u>x</u>	<u>y</u>
0,25	1,2557
0,3	2,1764

0,35	3,1218
0,40	4,0482
0,45	5,9875
0,5	6,9195
0,55	7,8359

Таблица конечных разностей

i	xi	yi	Δyi	Δ2yi	Δ3yi	Δ4yi	Δ5yi	Δ6yi
0(-3)	0,25	1,2557	0,9207	0,0247	-0,0437	1,0756	-4,1277	10,1917
1(-2)	0,3	2,1764	0,9454	-0,019	1,0319	-3,0521	6,064	
2(-1)	0,35	3,1218	0,9264	1,0129	-2,0202	3,0119		
3(0)	0,40	4,0482	1,9393	-1,0073	0,9917			
4(1)	0,45	5,9875	0,932	-0,0156				
5(2)	0,5	6,9195	0,9164					
6(3)	0,55	7,8359						

Исходя из таблицы строим многочлен Ньютона для первой точки X1 = 0,255

t = (0,255 - 0,25) / 0,05 = 0,1

Воспользуемся формулой вперёд:

$$N_n(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_0 + \dots + \frac{t(t-1)\dots(t-n+1)}{n!}\Delta^n y_0$$

y(0,255) = 1,2557 + 0,1 * 0,9207 + 0,1 * (0,1 - 1) *0,0247 / 2 + 0,1 * (0,1 - 1) * (0,1 - 2) * (-0,0437) / 6 + 0,1 * (0,1 - 1) * (0,1 - 2) * (0,1 - 3) * 1,0756 / 24 + 0,1

* (0,1 - 1) * (0,1 - 2) * (0,1 - 3) * (0,1 - 4) * (-4,1277) / 120 + 0,1 * (0,1 - 1) * (0,1 - 2) * (0,1 - 3) * (0,1 - 4) * (0,1 - 5)*(10,1917) / 720 = 1,1225

Исходя из таблицы строим многочлен Гаусса для второй точки X2 = 0,405

a = 0,4

t = (0,405 - 0,4) / 0,05 = 0,1

Воспользуемся первой формулой:

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

y(0,405) = 4,0482 + 0,1 * 1,9393 + 0,1 * (0,1 - 1) * 1,0129 / 2 + 0,1 * (0,1 + 1) * (0,1 - 1) * (-2,0202) / 6 + 0,1 * (0,1 + 1) * (0,1 - 1)*(0,1 - 2) * (-3,0521) / 24 + 0,1 * (0,1 + 1) * (0,1 + 2) * (0,1 - 1)* (0,1 - 2) * (6,064) / 120 + 0,1 * (0,1 + 1) * (0,1 + 2) * (0,1 - 1)* (0,1 - 2) * (0,1 - 3) * (10,1917) / 720 = 4,21

Листинг численного метода

Лагранж:

```
def lagrange(data, interpolation_dot):
    n = len(data['x'])
    result = 0
    for i in range(n):
        l = 1
        for j in range(n):
            if i != j:
                l *= (interpolation_dot - data['x'][j]) /
(data['x'][i] - data['x'][j])
        result += l * data['y'][i]
    return result
```

Ньютон с конечными разностями:

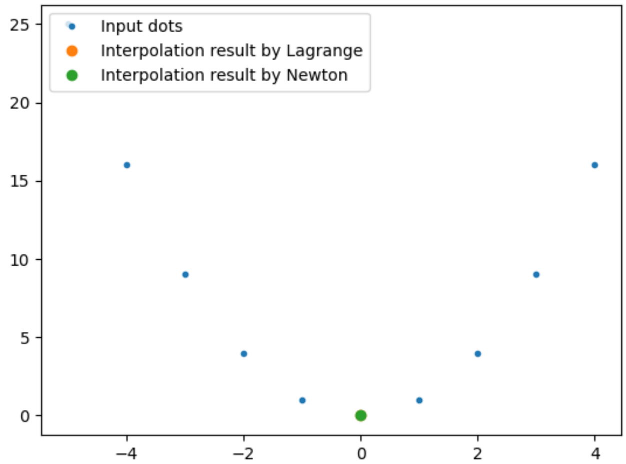
```
def newton(data, interpolation_dot):
    n = len(data['x'])
    x = data['x']
```

```
    y = data['y']
    h = check_h_is_constant(x)
    difference = count_difference(x, y)
    print_difference(difference)
    if interpolation_dot <= x[n // 2]:
        print("Newton interpolation forward")
        x0 = find_x0(x, interpolation_dot)
        t = (interpolation_dot - x[x0]) / h
        result = difference[x0][0]
        for i in range(1, n):
            result += taylor_forward(t, i) * difference[x0][i]
/ factorial(i)
    else:
        print("Newton interpolation backward")
        t = (interpolation_dot - x[n - 1]) / h
        result = difference[n - 1][0]
        for i in range(1, n):
            result += taylor_backward(t, i) * difference[n - 1
- i][i] / factorial(i)
    return result
```

Пример работы

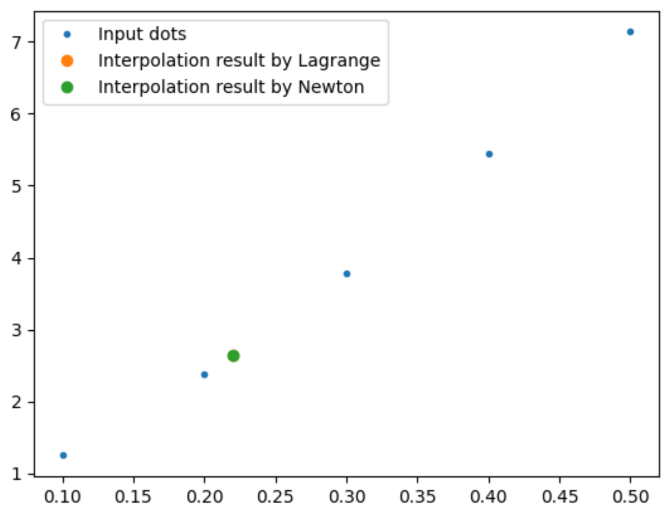
<u>Пример 1:</u>

```
^y0i 25.000 16.000 9.000 4.000 1.000 0.000 1.000 4.000 9.000 16.000
^y1i -9.000 -7.000 -5.000 -3.000 -1.000 1.000 3.000 5.000 7.000
^y2i 2.000 2.000 2.000 2.000 2.000 2.000 2.000 2.000
^y3i 0.000 0.000 0.000 0.000 0.000 0.000 0.000
^y4i 0.000 0.000 0.000 0.000 0.000 0.000
^y5i 0.000 0.000 0.000 0.000 0.000
^y6i 0.000 0.000 0.000 0.000
^y7i 0.000 0.000 0.000
^y8i 0.000 0.000
^y9i 0.000
Newton interpolation forward
Result by Lagrange: 0.0
Result by Newton: 0.0
```



Пример 2:

```
^y0i 1.250 2.380 3.790 5.440 7.140
^y1i 1.130 1.410 1.650 1.700
^y2i 0.280 0.240 0.050
^y3i -0.040 -0.190
^y4i -0.150
Newton interpolation forward
Result by Lagrange: 2.63872
Result by Newton: 2.6336799999999996
```



Вывод

В ходе выполнения лабораторной работы я узнала о интерполяции функции, написала для этого код на языке Python.