Міністерство освіти і науки України
Національний університет “Львівська політехніка”
Інститут комп’ютерних наук та інформаційних технологій
Кафедра програмного забезпечення

Звіт

Про виконання лабораторної роботи №8

На тему:

«Наближення дискретних (таблично заданих) функцій»
з дисципліни «Чисельні методи»

Лекторка:
доцент каф. ПЗ
Мельник Н. Б.

Виконав:
ст. гр. ПЗ-11
Ясногородський Н.В.

Прийняла:
доцент каф. ПЗ
Мельник Н. Б.

« __ » ________ 2022 р.

∑ = _____ _________ .

Львів – 2022

Тема: Наближення дискретних (таблично заданих) функцій.
Мета: Ознайомитися з методом інтерполяції таблично заданих функцій.

Теоретичні відомості

Інтерполяційний поліном Лагранжа – поліном, в основу якого покладено те, що в одному довільному вузлі інтерполяції поліном приймає значення одиниці, а у всіх інших – нуль. Наближена функція матиме вигляд $F(x) = \sum_{i=0}^{n} P_i(x)f(x_i)$, де

$$P_i(x) = \frac{(x-x_0)(x-x_1)\dots(x-x_{i-1})(x-x_{i+1})\dots(x-x_n)}{(x_i-x_0)(x_i-x_1)\dots(x_i-x_{i-1})(x_i-x_{i+1})\dots(x_i-x_n)}$$

Звідки отримаємо інтерполяційний многочлен Лагранжа

$$F(x) = \sum_{i=0}^{n} \frac{(x-x_0)(x-x_1)\dots(x-x_{i-1})(x-x_{i+1})\dots(x-x_n)}{(x_i-x_0)(x_i-x_1)\dots(x_i-x_{i-1})(x_i-x_{i+1})\dots(x_i-x_n)} f(x_i)$$

Інтерполяційний поліном Ньютона – використовує дещо інший принцип побудови:

$$P_n(x) = f(x_0) + f(x_0, x_1)(x-x_0) + f(x_0, x_1, x_2)(x-x_0)(x-x_1) + \dots + +f(x_0, x_1, \dots, x_n)(x-x_0)(x-x_1)\dots(x-x_{n-1})$$
,

Де $f(x_0, x_1, \dots, x_n) = \dfrac{f(x_1, \dots, x_n) - f(x_0, \dots, x_{n-1})}{x_n - x_0}$ – розділена різниця n-го порядку для нерівновіддалених вузлів.

У випадку рівновіддалених вузлів використовуємо скінчені різниці

$\Delta^n f(x_i) = \Delta^{n-1} f(x_{i+1}) - \Delta^{n-1} f(x_i)$, де $x_i = x_0 + ih$

Тоді $f(x_0, x_1, \dots, x_n) = \dfrac{\Delta^n f(x_0)}{n!h^n}$, звідки отримуємо інтерполяційний поліном Ньютона для рівновіддалених вузлів

$$P_n(x) = f(x_0) + \frac{\Delta f(x_0)}{1!h}(x-x_0) + \frac{\Delta^2 f(x_0)}{1!h^2}(x-x_0)(x-x_1) + \dots +$$

$$+\frac{\Delta^n f(x_0)}{n!h^n}(x-x_0)(x-x_1)\ldots(x-x_{n-1})$$

Індивідуальне завдання

Використовуючи інтерполяційні поліноми Лагранжа та Ньютона, обчислити значення табличної заданої функції у точці x0:

15-й варіант

x	0,115	0,120	0,125	0,130	0,135	0,140	0,145	0,150	0,165	0,170
y	4,48	4,95	5,47	5,99	6,05	6,68	6,909	7,38	8,166	9,025

$x_0 = 0{,}142$

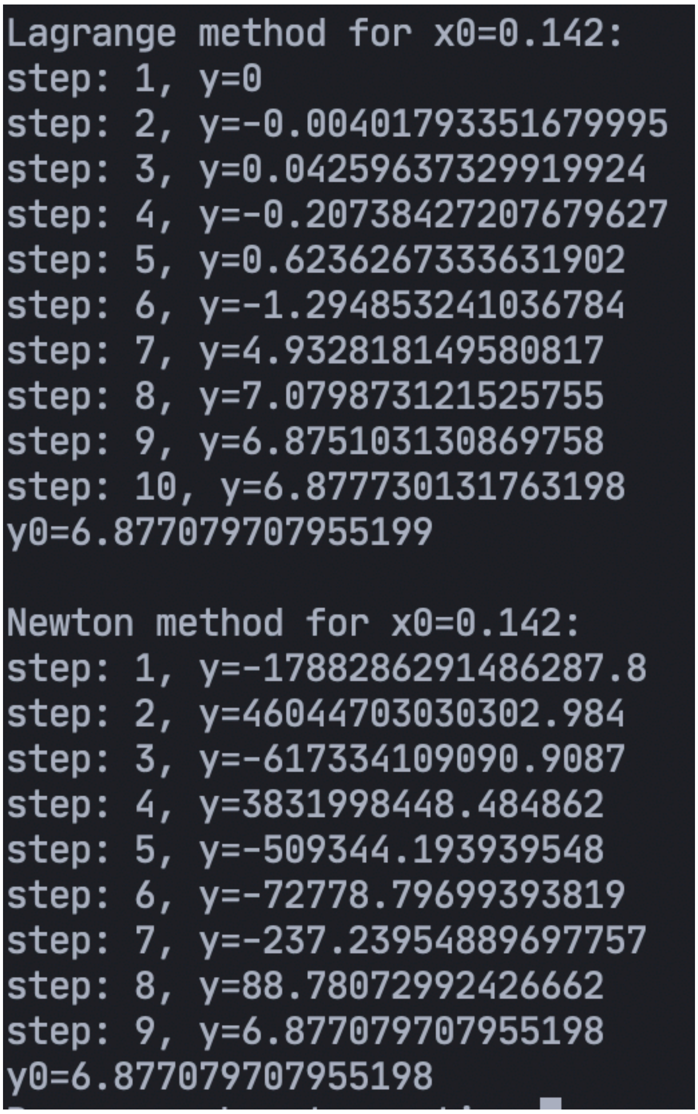
Код функцій

```
import numpy as np

points = (
    np.array(
        [
            0.115,
            0.12,
            0.125,
            0.13,
            0.135,
            0.14,
            0.145,
            0.15,
            0.165,
            0.17,
        ]
    ),
    np.array([4.48, 4.95, 5.47, 5.99, 6.05, 6.68, 6.909, 7.38, 8.166, 9.025]),
```

```python
)
x0 = 0.142


def lagrange_at_x(points, x0):
    print(f"\nLagrange method for x0={x0}:")
    x_points, y_points = points

    n = len(x_points)
    yp = 0  # interpolated value

    for i in range(n):
        p = 1
        for j in range(n):
            if i == j:
                continue
            p *= (x0 - x_points[j]) / (x_points[i] - x_points[j])
        print(f"step: {i+1}, y={yp}")
        yp += p * y_points[i]

    print(f"y0={yp}")
    return yp


def newton_at_x(points, x0):
    print(f"\nNewton method for x0={x0}:")
    x_points, y_points = points
    m = len(x_points)
    a_coff = np.copy(y_points)

    for k in range(1, m):
        a_coff[k:m] = (a_coff[k:m] - a_coff[k - 1]) / (x_points[k:m] - x_points[k - 
1])

    n = len(x_points) - 1  # Degree of polynomial
    yp = a_coff[n]

    for k in range(1, n + 1):
        yp = a_coff[n - k] + (x0 - x_points[n - k]) * yp
        print(f"step: {k}, y={yp}")

    print(f"y0={yp}")
    return yp
```

```python
if __name__ == "__main__":
    lagrange_at_x(points, x0)
    newton_at_x(points, x0)
```

Протокол роботи



```
Lagrange method for x0=0.142:
step: 1, y=0
step: 2, y=-0.004017933516799995
step: 3, y=0.042596373299l9924
step: 4, y=-0.20738427207679627
step: 5, y=0.6236267333631902
step: 6, y=-1.294853241036784
step: 7, y=4.932818149580817
step: 8, y=7.079873121525755
step: 9, y=6.875103130869758
step: 10, y=6.877730131763198
y0=6.877079707955199

Newton method for x0=0.142:
step: 1, y=-1788286291486287.8
step: 2, y=46044703030302.984
step: 3, y=-617334109090.9087
step: 4, y=3831998448.484862
step: 5, y=-509344.193939548
step: 6, y=-72778.79699393819
step: 7, y=-237.23954889697757
step: 8, y=88.78072992426662
step: 9, y=6.877079707955198
y0=6.877079707955198
```

Рис.1. Робота програми

Висновки

Виконуючи лабораторну роботу №8, я ознайомився з методом інтерполяції таблично заданих функцій, та склав програму для інтерполяції методом Лагранжа для нерівновіддалених вузлів та методом Ньютона для рівновіддалених.