Київський національний університет імені Тараса Шевченка, факультет радіофізики, електроніки та комп’ютерних систем

Звіт

## На тему

## «ІНТЕРПОЛЯЦІЯ ДАНИХ. ІНТЕРПОЛЯЦІЙНИЙ ПОЛІНОМ ЛАГРАНЖА. ЧИСЕЛЬНЕ ДИФЕРЕНЦІЮВАННЯ»

## Варіант №9

## Виконав:

## Студент 1-го курсу, Група ІКСМ-1 Кобрин Г.С

Мета роботи: застосування алгоритмів інтерполяції для побудови поліноміального наближення функції; вивчення методів числового диференціювання функції однієї змінної.

Завдання: апроксимувати функцію f(x) поліномом Лагранжа, використовуючи глобальну та кускову інтерполяції на заданих вузлах. За формулами числового диференціювання наближено знайти першу та другу похідні функції f(x).

# Вимоги до виконання роботи

1. Запрограмуйте обчислення інтерполяційного поліному L(x) довільного степеню.

2. Функція f (x) задана аналітично та у вигляді таблиці. Згідно з вашим варіантом побудуйте таблицю значень функції f (x) на відрізку [a,b] з кроком h:

а) отриманих безпосередньо за формулою;   
б) отриманих за допомогою кускової інтерполяції поліномами Лагранжа вказаного степеню;  
в) отриманих за допомогою глобальної інтерполяції поліномом Лагранжа.

3. Побудуйте таблицю значень функцій f ′(x) та f ′′(x) на відрізку [a,b] з кроком h :

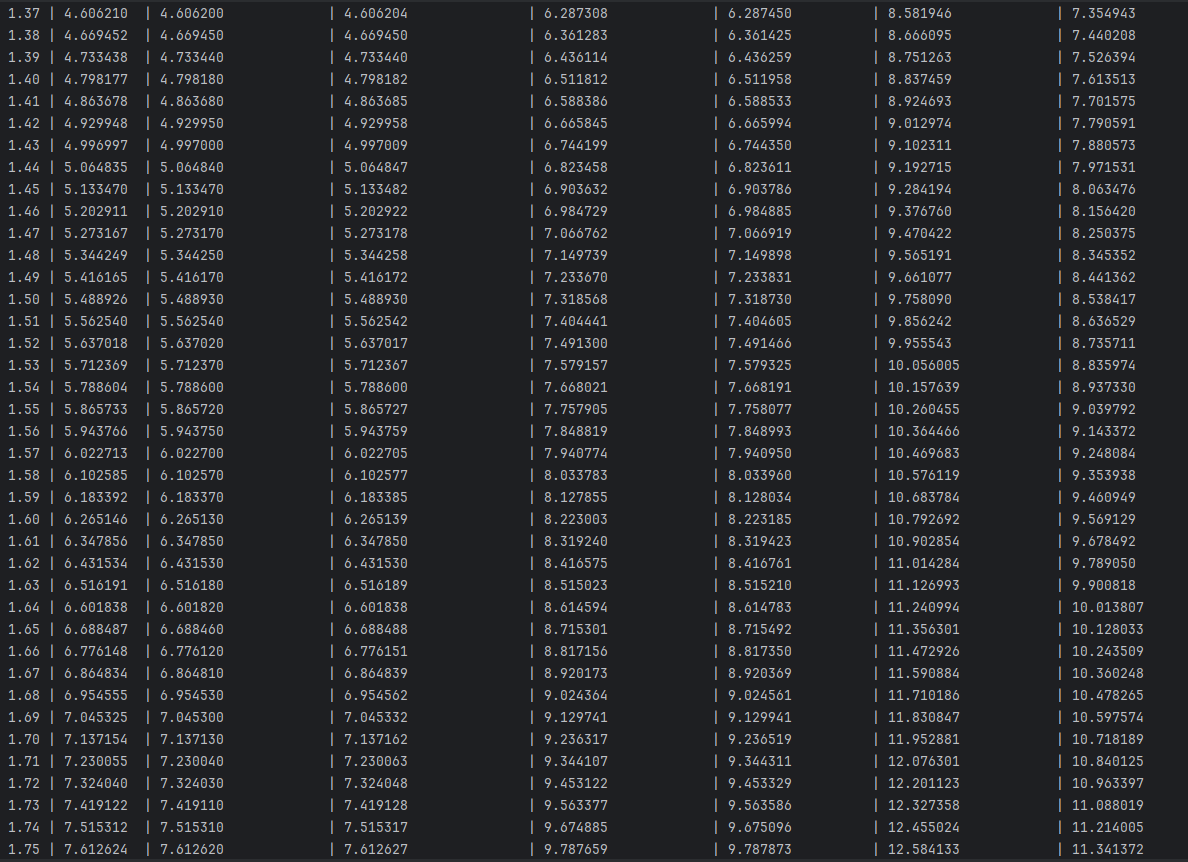
а) отриманих безпосереднім диференціюванням;   
б) отриманих за формулами числового диференціювання. Для обчислення значень функції, що використовуються у формулах диференціювання, скористайтесь кусково-поліноміальною інтерполяцію вказаного степеню.



# Хід роботи:

## Отримані результати:

Зображення, що містить текст, знімок екрана, меню

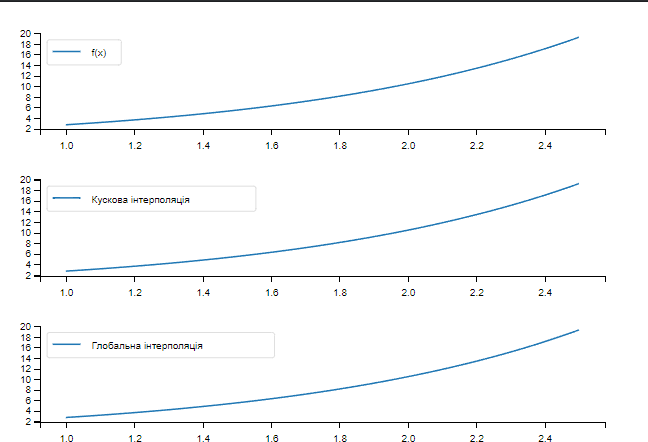
Автоматично згенерований опис 

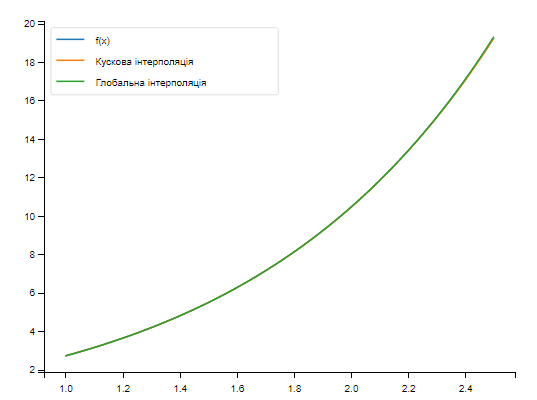
Зображення, що містить текст, знімок екрана, меню, книга

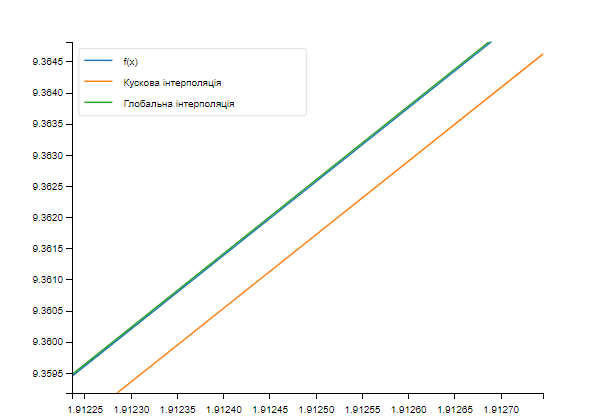
Автоматично згенерований опис Зображення, що містить текст, знімок екрана, меню, книга

Автоматично згенерований опис

## Графіки отримані на базі вимірювань







Зображення, що містить текст, знімок екрана, Графік, схема

Автоматично згенерований опис

Зображення, що містить текст, знімок екрана, ряд, Графік

Автоматично згенерований опис­­­

## Програма написана для обробки та виведення даних

from math import \*

import matplotlib.pyplot as plt

import numpy as np

xl = [1, 1.21, 1.38, 1.39, 1.5, 1.54, 1.62, 1.76, 1.77, 1.88, 2.16, 2.77, 2.88]

fl = [2.71828, 3.68883, 4.66945, 4.73344, 5.48893, 5.78860, 6.43153, 7.71107,

      7.81066, 8.98571, 12.74392, 26.56046, 30.23182]

a, b, h = 1, 2.5, 0.01

def polynomial(x, lx, ly):

    ln1 = 1

    ln = 0

    for k in range(len(ly)):

        for j in range(len(lx)):

            if j != k:

                ln1 = ln1 \* (x - lx[j]) / (lx[k] - lx[j])

        ln = ln + ly[k] \* ln1

        ln1 = 1

    return ln

def polynomial\_lagrange(x, lx, ly, n):

    y = 0

    for i1 in range(0, len(lx) - 1, n):

        if (x >= lx[i1]) and (x <= lx[i1 + n]):

            y += polynomial(x, lx, ly)

    return round(y, 8)

def func(x):

    y = sqrt(x) \* exp(x)

    return y

def interpolation\_piecewise(x):

    i = next(i for i, xi in enumerate(xl) if xi > x) - 1

    i = max(min(i, len(xl) - 3), 1)

    y = round(fl[i - 1] \* ((x - xl[i]) \* (x - xl[i + 1]) \* (x - xl[i + 2]))

              / ((xl[i - 1] - xl[i]) \* (xl[i - 1] - xl[i + 1]) \* (xl[i - 1] - xl[i + 2]))

              + fl[i] \* ((x - xl[i - 1]) \* (x - xl[i + 1]) \* (x - xl[i + 2]))

              / ((xl[i] - xl[i - 1]) \* (xl[i] - xl[i + 1]) \* (xl[i] - xl[i + 2]))

              + fl[i + 1] \* ((x - xl[i - 1]) \* (x - xl[i]) \* (x - xl[i + 2]))

              / ((xl[i + 1] - xl[i - 1]) \* (xl[i + 1] - xl[i]) \* (xl[i + 1] - xl[i + 2]))

              + fl[i + 2] \* ((x - xl[i - 1]) \* (x - xl[i]) \* (x - xl[i + 1]))

              / ((xl[i + 2] - xl[i - 1]) \* (xl[i + 2] - xl[i]) \* (xl[i + 2] - xl[i + 1])), 5)

    return round(y, 8)

def diff1(x):

    y = round(sqrt(x) \* exp(x) + exp(x) / (2 \* sqrt(x)), 8)

    return y

def diff1int(x):

    if x == a:

        y = round((func(a + h) - func(a)) / h, 8)

        return y

    if x == b:

        y = round((func(b) - func(b - h)) / h, 8)

        return y

    for i in np.arange(a, b + h, h):

        if x == i:

            y = round((func(i + h) - func(i - h)) / (2 \* h), 8)

            return y

def diff2(x):

    y = round(sqrt(x) \* exp(x) + exp(x) / sqrt(x) + exp(x) / (4 \* x \*\* (3 / 2)), 8)

    return y

def diff2int(x):

    y = round((func(x + h) + func(x - h) - 2 \* func(x)) / (h \*\* 2), 8)

    return y

x\_values = np.arange(a, b + h, h)

results = np.array([[func(x), interpolation\_piecewise(x), polynomial\_lagrange(x, xl, fl, 6), diff1(x), diff1int(x),

                     diff2(x), diff2int(x)] for x in x\_values])

# Print the table header

print(f"{'x':<4} | {'f(x)':<9} | {'Кускова інтерполяція':<20} | {'Глобальна інтерполяція':<22} | "

      f""f"{'Аналітична похідна 1':<20} | {'Числова похідна 1':<17} | {'Аналітична похідна 2':<20} |"

      f" {'Числова похідна 2':<15}")

# Print the table rows

for x, result in zip(x\_values, results):

    print(f"{x:<4.2f} | {result[0]:<9.6f} | {result[1]:<20.6f} | {result[2]:<22.6f} | {result[3]:<20.6f} |"

          f" {result[4]:<17.6f} | {result[5]:<20.6f} | {result[6]:<15.6f}")

plt.figure(figsize=(10, 6))

plt.plot(x\_values, results[:, 0], label='f(x)')

plt.plot(x\_values, results[:, 1], label='Кускова інтерполяція')

plt.plot(x\_values, results[:, 2], label='Глобальна інтерполяція')

plt.legend()

plt.show()

fig, axs = plt.subplots(3, figsize=(7, 4))

axs[0].plot(x\_values, results[:, 0], label='f(x)')

axs[0].legend()

axs[1].plot(x\_values, results[:, 1], label='Кускова інтерполяція')

axs[1].legend()

axs[2].plot(x\_values, results[:, 2], label='Глобальна інтерполяція')

axs[2].legend()

plt.tight\_layout()

plt.show()

fig, axs = plt.subplots(4, figsize=(7, 4))

axs[0].plot(x\_values, results[:, 3], label='Аналітична похідна 1')

axs[0].legend()

axs[1].plot(x\_values, results[:, 4], label='Числова похідна 1')

axs[1].legend()

axs[2].plot(x\_values, results[:, 5], label='Аналітична похідна 2')

axs[2].legend()

axs[3].plot(x\_values, results[:, 6], label='Числова похідна 2')

axs[3].legend()

plt.tight\_layout()

plt.show()

plt.figure(figsize=(10, 6))

plt.plot(x\_values, results[:, 3], label='Аналітична похідна 1')

plt.plot(x\_values, results[:, 4], label='Числова похідна 1')

plt.plot(x\_values, results[:, 5], label='Аналітична похідна 2')

plt.plot(x\_values, results[:, 6], label='Числова похідна 2')

plt.legend()

plt.show()

# Контрольні питання

1. У задачі поліноміальної інтерполяції знаходиться поліном степеня не більше n, який проходить через n+1 заданих точок на площині (або вище). Процес знаходження функції, що повністю відтворює задані дані точок. Ця задача має єдиний розв'язок для кожного набору точок, якщо всі точки є унікальними. Це відомо як теорема про існування та єдиність поліноміальної інтерполяції.

2. Похибка інтерполяції - це різниця між функцією, яку ми намагаємося наблизити поліномом, і самим поліномом, який проходить через задані точки. Іншими словами, похибка інтерполяції відображає різницю між реальними значеннями функції та значеннями, які дає поліноміальний апроксимант.

3. Глобальна інтерполяція передбачає побудову одного полінома, який проходить через всі задані точки на площині (або вище). Глобальна інтерполяція може бути корисною, у випадку, де потрібно знайти один поліном, який описує всю функцію на певному інтервалі.

Кускова інтерполяція, передбачає розбиття інтервалу на декілька менших інтервалів та побудову окремого полінома на кожному з цих інтервалів. Даний підхід дозволяє створити поліноми меншого степеню на кожному з інтервалів, що зазвичай забезпечує більшу точність апроксимації функції.

4. Апроксимація функції може бути використана для чисельного знаходження похідної функції у випадку, коли аналітична формула для похідної не є відомою або складно застосовуваною.

5. Порядок похибки формули чисельного диференціювання відносно кроку диференціювання визначає те, наскільки швидко зменшується похибка апроксимації збільшенням кроку диференціювання.