Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Лабораторна робота № 6

з дисципліни «Спеціальні розділи математики-2.  
Чисельні методи»

Виконав:

студент гр. ІС-34

Колосов Ігор

Викладач:

доц. Рибачук Л.В.

Київ – 2024

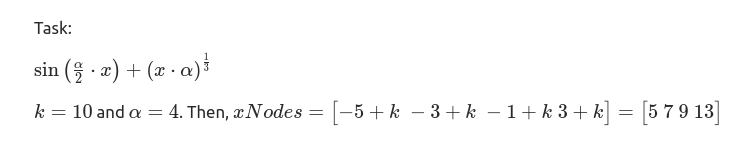
Тема: Інтерполяційні поліноми

Постановка задачі:

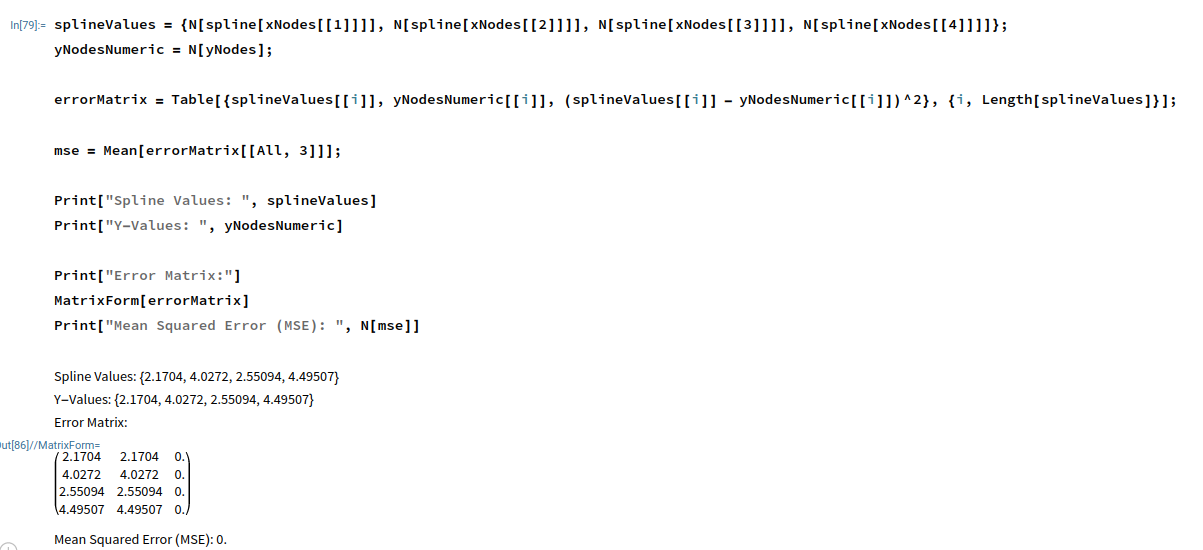
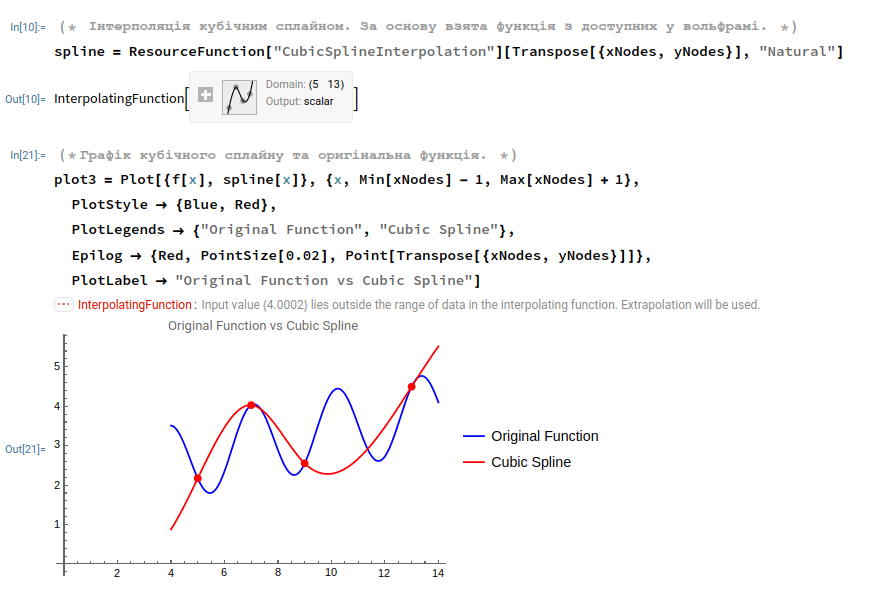
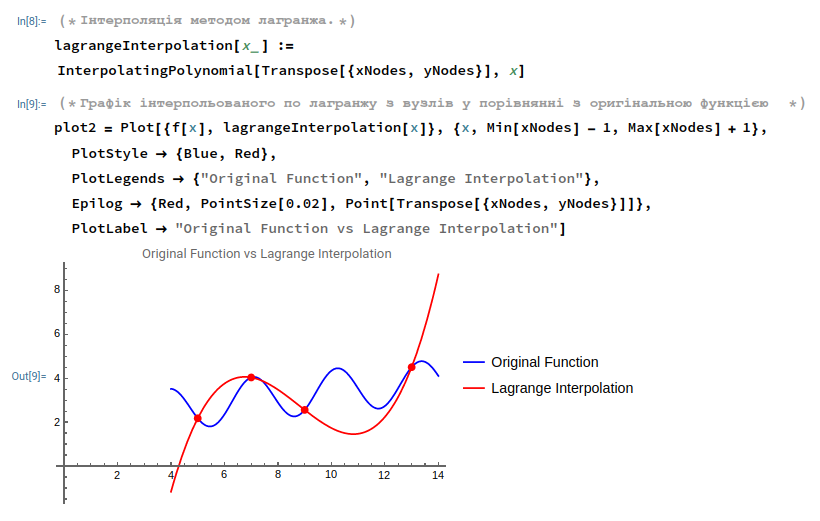
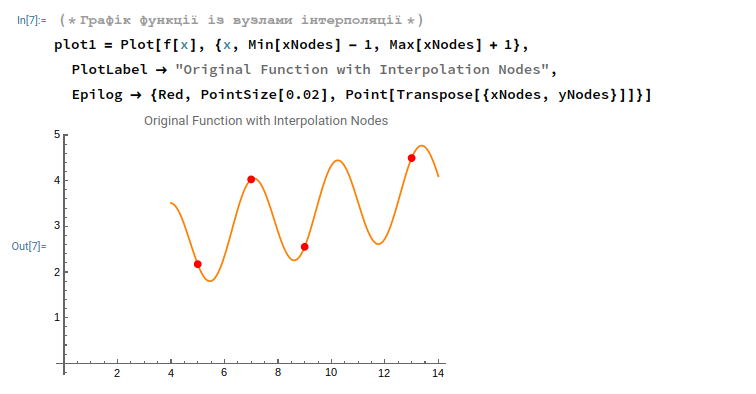
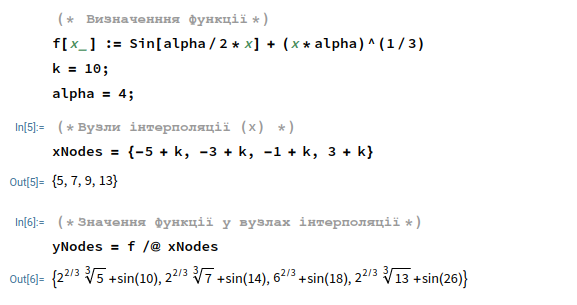
Створити програму, яка для заданої функції по заданим точкам будує інтерполяційний поліном Pn(x) у формі Лагранжа або Ньютона, а також здійснює інтерполяцію кубічними сплайнами.

Програма має розрахувати значення похибки | P (x) y(x) | e = n − , для чого потрібно вивести на графік із кроком (графік можна будувати допоміжними засобами, наприклад, у Mathcad), меншим у 5-6 разів, ніж крок інтерполяції, відповідні значення поліному та точної функції. Якщо похибка дуже мала, застосувати масштабування.

Знайти кубічний інтерполяційний сплайн для заданої функції у Mathcad. Вивести графік результатів.

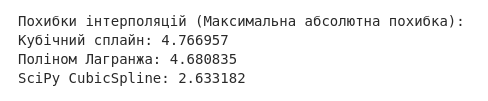
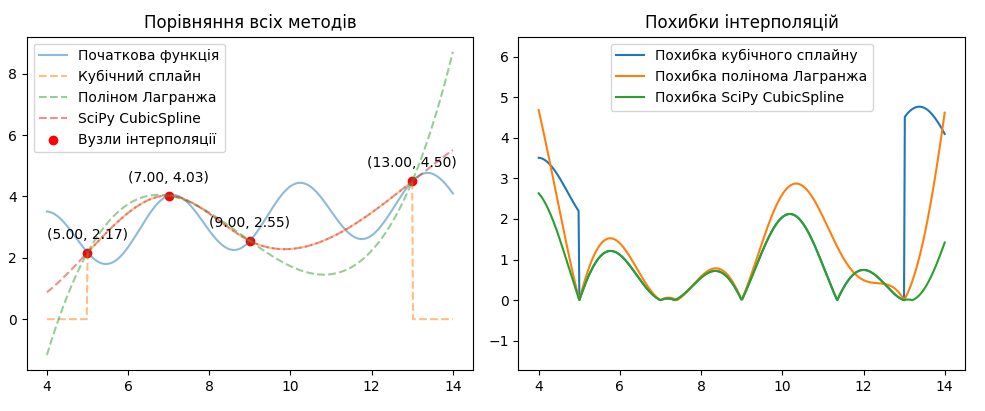
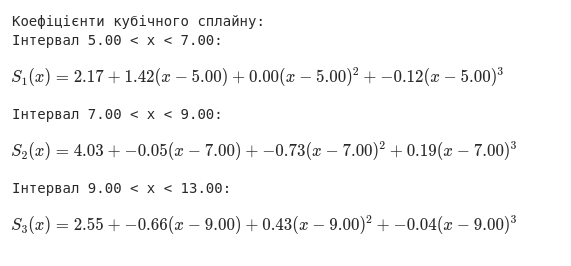
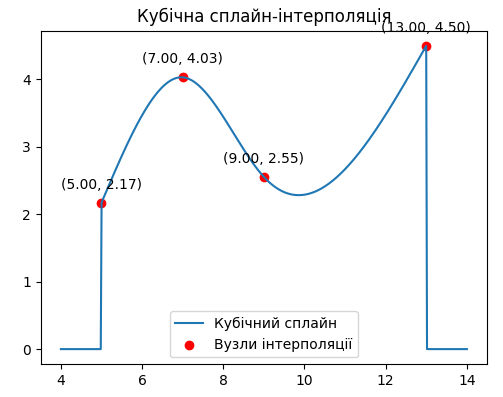
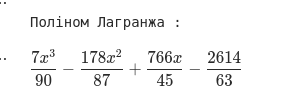
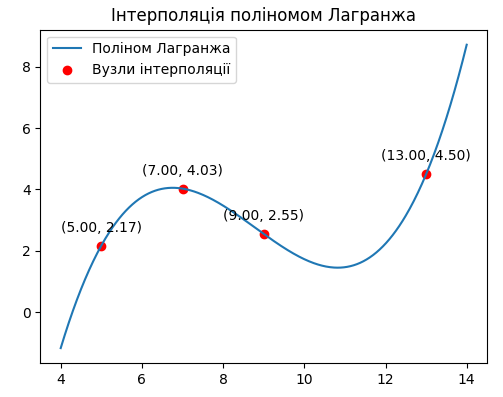
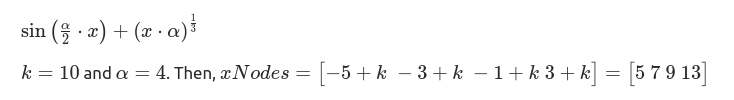
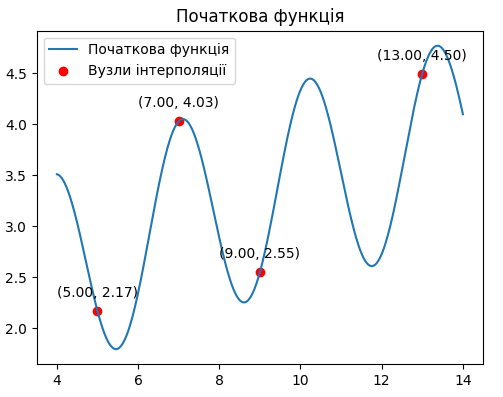


Розв’язок:



Обрахунок помилки.

Скріншоти програми:



Лістинг програми:

import numpy as np

import matplotlib.pyplot as plt

from scipy.interpolate import CubicSpline

from sympy import Symbol, Integer, lambdify, latex

from sympy import nsimplify

from sympy import nsimplify

from IPython.display import display, Math

def f(x, alpha=4):

return np.sin(alpha/2 \* x) + (x \* alpha)\*\*(1/3)

def lagrange\_pol(x, y, argx, print\_func=False):

"""

Інтерполяційний поліном Лагранжа.

"""

s = 0 # Змінна для зберігання числового значення полінома Лагранжа

m = len(x) # Кількість вузлів інтерполяції

x\_arg = Symbol("x") # Символьна змінна для полінома Лагранжа

f = Integer(0) # Символьний вираз полінома Лагранжа

# Цикл по всіх вузлах інтерполяції

for i in range(m):

c = 1 # Змінна для зберігання числового значення базисного полінома Лагранжа

temp = 1 # Змінна для зберігання символьного виразу базисного полінома Лагранжа

# Цикл по всіх вузлах інтерполяції, крім поточного

for j in range(m):

if i != j:

# Обчислення числового значення базисного полінома Лагранжа

c \*= (argx - x[j]) / (x[i] - x[j])

# Обчислення символьного виразу базисного полінома Лагранжа

temp \*= (x\_arg - x[j]) / (x[i] - x[j])

# Обчислення числового значення полінома Лагранжа

s += c \* y[i]

# Обчислення символьного виразу полінома Лагранжа

f += temp \* y[i]

# Повернення числового значення та символьного виразу полінома Лагранжа

return s, f

def cubic\_spline\_interp(x, y):

"""

Кубічна сплайн-інтерполяція.

"""

n = len(x) # Кількість вузлів інтерполяції

h = np.diff(x) # Обчислення довжин відрізків між вузлами

# Крок 1: Підготовка тридіагональної системи

A = np.zeros((n, n)) # Матриця системи

f = np.zeros((n, 1)) # Права частина системи

# Заповнення матриці системи A

A[0, 0] = 1 # Крайова умова на лівому кінці

A[-1, -1] = 1 # Крайова умова на правому кінці

for i in range(1, n-1):

A[i, i-1] = h[i-1] # Коефіцієнт a\_i

A[i, i] = 2 \* (h[i-1] + h[i]) # Коефіцієнт b\_i

A[i, i+1] = h[i] # Коефіцієнт c\_i

f[i] = 3 \* ((y[i+1] - y[i]) / h[i] - (y[i] - y[i-1]) / h[i-1]) # Права частина рівняння

# Крок 2: Розв'язування тридіагональної системи методом прогонки

# Прямий хід прогонки:

# - Виражаємо кожну невідому x\_i через x\_{i+1} за допомогою прогоночних коефіцієнтів A\_i та B\_i.

# - Прогоночні коефіцієнти обчислюються рекурентно, починаючи з першого рівняння.

# Обернений хід прогонки:

# - Послідовно обчислюємо невідомі x\_i, починаючи з останнього рівняння.

# - Використовуємо прогоночні коефіцієнти, обчислені на прямому ході.

c = np.linalg.solve(A, f) # Розв'язування системи методом прогонки

# Крок 3: Обчислення коефіцієнтів сплайну

d = np.zeros(n-1)

b = np.zeros(n-1)

for i in range(n-1):

d[i] = (c[i+1] - c[i]) / (3 \* h[i]) # Обчислення коефіцієнтів d\_i

b[i] = (y[i+1] - y[i]) / h[i] - h[i] \* (c[i+1] + 2 \* c[i]) / 3 # Обчислення коефіцієнтів b\_i

a = y[:-1] # Коефіцієнти a\_i

return a, b, c[:-1], d

# Задані параметри

k = 10

alpha = 4

xNodes = np.array([-5 + k, -3 + k, -1 + k, 3 + k])

yNodes = f(xNodes, alpha)

x\_interp = np.linspace(min(xNodes) - 1, max(xNodes) + 1, 500)

y\_true = f(x\_interp, alpha)

# Обчислення інтерполяцій

lagrange\_numeric, lagrange\_symbolic = lagrange\_pol(xNodes, yNodes, Symbol('x'), print\_func=True)

lagrange\_func = lambdify(Symbol('x'), lagrange\_symbolic, 'numpy')

y\_lagrange = lagrange\_func(x\_interp)

a, b, c, d = cubic\_spline\_interp(xNodes, yNodes)

y\_cubic = np.piecewise(x\_interp,

[np.logical\_and(x\_interp >= xNodes[i], x\_interp <= xNodes[i+1]) for i in range(len(xNodes)-1)],

[lambda x, i=i: a[i] + b[i]\*(x-xNodes[i]) + c[i]\*(x-xNodes[i])\*\*2 + d[i]\*(x-xNodes[i])\*\*3 for i in range(len(xNodes)-1)])

cs\_scipy = CubicSpline(xNodes, yNodes, bc\_type='natural')

y\_scipy = cs\_scipy(x\_interp)

# Побудова графіків

fig, axs = plt.subplots(3, 2, figsize=(10, 12))

axs = axs.flatten()

# Графік 1: Кубічний сплайн з точками

axs[0].plot(x\_interp, y\_cubic, label='Кубічний сплайн')

axs[0].scatter(xNodes, yNodes, color='red', label='Вузли інтерполяції')

axs[0].set\_title('Кубічна сплайн-інтерполяція')

axs[0].legend()

# Графік 2: Поліном Лагранжа з точками

axs[1].plot(x\_interp, y\_lagrange, label='Поліном Лагранжа')

axs[1].scatter(xNodes, yNodes, color='red', label='Вузли інтерполяції')

axs[1].set\_title('Інтерполяція поліномом Лагранжа')

axs[1].legend()

# Графік 3: Початкова функція з точками

axs[2].plot(x\_interp, y\_true, label='Початкова функція')

axs[2].scatter(xNodes, yNodes, color='red', label='Вузли інтерполяції')

axs[2].set\_title('Початкова функція')

axs[2].legend()

# Графік 4: SciPy CubicSpline

axs[3].plot(x\_interp, y\_scipy, label='SciPy CubicSpline')

axs[3].scatter(xNodes, yNodes, color='red', label='Вузли інтерполяції')

axs[3].set\_title('SciPy CubicSpline Інтерполяція')

axs[3].legend()

# Графік 5: Всі методи разом

axs[4].plot(x\_interp, y\_true, label='Початкова функція',alpha=0.5)

axs[4].plot(x\_interp, y\_cubic, label='Кубічний сплайн',alpha=0.5, ls='--')

axs[4].plot(x\_interp, y\_lagrange, label='Поліном Лагранжа',alpha=0.5, ls='--')

axs[4].plot(x\_interp, y\_scipy, label='SciPy CubicSpline', alpha=0.5, ls='--')

axs[4].scatter(xNodes, yNodes, color='red', label='Вузли інтерполяції')

axs[4].set\_title('Порівняння всіх методів')

axs[4].legend()

# Графік 6: Похибки

axs[5].plot(x\_interp, np.abs(y\_true - y\_cubic), label='Похибка кубічного сплайну')

axs[5].plot(x\_interp, np.abs(y\_true - y\_lagrange), label='Похибка полінома Лагранжа')

axs[5].plot(x\_interp, np.abs(y\_true - y\_scipy), label='Похибка SciPy CubicSpline')

axs[5].set\_title('Похибки інтерполяцій')

axs[5].legend()

# Підписи значень вузлів на графіках

for ax in axs[:-1]:

for x, y in zip(xNodes, yNodes):

ax.annotate(f'({x:.2f}, {y:.2f})', (x, y), textcoords="offset points", xytext=(0,10), ha='center')

plt.axis('equal')

plt.tight\_layout()

plt.show()

# Виведення полінома Лагранжа

print("\nПоліном Лагранжа :")

display(Math(latex(nsimplify(lagrange\_symbolic.simplify(), rational=True, tolerance=0.01))))

# Виведення коефіцієнтів кубічного сплайну

print("\nКоефіцієнти кубічного сплайну:")

for i in range(len(a)):

interval = f"{xNodes[i]:.2f} < x < {xNodes[i+1]:.2f}"

equation = f"S\_{{{i+1}}}(x) = {float(a[i]):.2f} + {float(b[i]):.2f}(x-{float(xNodes[i]):.2f}) + {float(c[i]):.2f}(x-{float(xNodes[i]):.2f})^2 + {float(d[i]):.2f}(x-{float(xNodes[i]):.2f})^3"

print(f"Інтервал {interval}:")

display(Math(equation))

# Виведення похибок інтерполяцій

print("\nПохибки інтерполяцій (Максимальна абсолютна похибка):")

print(f"Кубічний сплайн: {np.max(np.abs(y\_true - y\_cubic)):.6f}")

print(f"Поліном Лагранжа: {np.max(np.abs(y\_true - y\_lagrange)):.6f}")

print(f"SciPy CubicSpline: {np.max(np.abs(y\_true - y\_scipy)):.6f}")

Висновок

У ході виконання лабораторної роботи я дізнався про інтерполювання функцій за допомогою інтерполяційних многочленів Лагранжа та Ньютона та метод інтерполювання сплайнами. Я навився програмно реалізовувати знаходження інтерполяційного многочлена Лагранжа і використовувати його та метод інтерполювання сплайнами у середовищі Wolfram для знаходження значень функції у вузлах інтерполяції.

У результаті виконання завдання було практично підтверджено, що знаходження кубічних сплайнів при великих степенях полінома спричиняє коливання полінома на проміжках між вузлами інтерполювання, проте на самих вузлах інтерполяції є точним.