Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Фізико-технічний інститут

Комп'ютерний практикум №5
Інтерполяція

предмет «Методи обчислень»

Роботу виконав:
Студент 3 курсу ФТІ, групи ФІ-91
Цибульник Антон Владиславович

Приймала:
Стьопочкіна Ірина Валеріївна

2022

Завдання

За таблично заданою функцією слід побудувати:

- інтерполяційний поліном $P_n(x)$ у формі Ньютона або Лагранжа;
- здійснити інтерполяцію сплайнами (другого чи третього порядку);
- побудувати графік похибки інтерполяції.

Примітка щодо останнього пункту: функція, яка інтерполюється, задана аналітично, отже, похибку інтерполяції можна визначити безпосередньо як максимум різниць між значеннями точної функції та інтерполюючої функції у ряді точок, при цьому точки не повинні співпадати із вузлами інтерполяції.

Відрізок інтерполяції розбити не менш ніж на 10 вузлів. Крім того, використовуючи аналітичне задання функції, визначене варіантом, побудувати таблицю значень функції у вузлах на відповідному відрізку інтерполяції.

Варіант завдання

На відрізку інтерполяції $\left[-\frac{\pi}{3}, \frac{\pi}{3}\right]$ аналітично задана функція $F(x) = x \cdot \operatorname{tg} x$.

Теоретичні відомості

Інтерполяційний поліном Лагранжа

Нехай задано таблицю значень $y_k = F(x_i)$ у відповідних вузлах x_i. Необхідно побудувати інтерполяційний поліном за цією таблицею значень. Поліном Лагранжа $L_n(x)$ використовується як для рівномірних (вузли на інтерполяційному відрізку розподілені на однакових відстанях), так і для нерівномірних вузлів. Тож сам поліном у загальному виді шукається так:

$$L_n(x) = c_i(x - x_0)(x - x_1)\dots(x - x_{i-1})(x - x_{i+1})\dots(x - x_{n-1})(x - x_n)$$

При цьому необхідно задовільнити рівності $L_n(x_i) = y_i$, тому модифікована побудова відбуватиметься так:

$$L_n(x) = \sum_{i=1}^{n} y_i \cdot \frac{(x - x_0)(x - x_1)\dots(x - x_{i-1})(x - x_{i+1})\dots(x - x_{n-1})(x - x_n)}{(x_i - x_0)(x_i - x_1)\dots(x_i - x_{i-1})(x_i - x_{i+1})\dots(x_i - x_{n-1})(x_i - x_n)}$$

Сплайн-інтерполяція

Ідея сплайн-інтерполяції полягає у побудові поліномів між парами сусідніх вузлів інтерполяції, причому для кожної пари вузлів будується свій поліном.

Найпоширеніший у практиці є кубічний сплайн, для побудови якого необхідно побудувати n многочленів третьої степені такого виду:

$$S(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \text{ де } x_i \leqslant x \leqslant x_{i+1}$$

Тож першочерговою задачею побудови кубічного сплайну є знаходження коефіцієнтів a_i, b_i, c_i, d_i. Пошук відбувається із накладених обмежень на вираз $S(x)$:

- у сусідніх вузлах інтерполяції x_i та x_{i+1} побудований поліном $S(x)$ має задовільняти значенням y_i та y_{i+1};
- має виконуватися умова неперервності перших і других похідних у вузлах інтерполяції, тобто умова гладкості кривої в усіх точках. Інакше кажучи, зліва і справа від вузла інтерполяції перші й другі похідні сусідніх сплайнів мають досягати рівності: $S_i'(x_i) = S_{i+1}'(x_i),\ S_i''(x_i) = S_{i+1}''(x_i)$.
- другі похідні в першому та в останньому вузлах інтерполяції мають дорівнювати нулю: $S(x_0)'' = 0,\ S(x_n)'' = 0$.

Отже, із заданих трьох умов складається система обмежень на шукані коефіцієнти a_i, b_i, c_i, d_i. Фінальні формули обчислення коефіцієнтів наведені у цьому джерелі.

Програмний код та проміжні результати

№	**Інтерполяційні вузли** x_i	**Значення функції** $F(x_i)$
x_0	-1.0472	1.8138
x_1	-0.8378	0.9304
x_2	-0.6283	0.4565
x_3	-0.4189	0.1865
x_4	-0.2094	0.0445
x_5	0	0
x_6	0.2094	0.0445
x_7	0.4189	0.1865
x_8	0.6283	0.4565
x_9	0.8378	0.9304
x_{10}	1.0472	1.8138

Табл. 1: Таблиця значень на відрізку

Інтерполяційний поліном Лагранжа

```
import numpy as np
import matplotlib.pyplot as plt

plt.figure(figsize=(9, 6))

def f(x):
    return x*np.tan(x)

x = np.arange(-np.pi/3, np.pi/3, np.pi/15)
x = np.append(x, np.pi/3)
y = [f(x_i) for x_i in x]

plt.plot(x, y, "o", color = "red", markersize = 6)

def L(X):
    S0 = y[0]*((X-x[1])*(X-x[2])*(X-x[3])*(X-x[4])*(X-x[5])*(X-x[6])*(X-x[7])*
        (X-x[8])*(X-x[9])*(X-x[10])) /
        ((x[0]-x[1])*(x[0]-x[2])*(x[0]-x[3])*(x[0]-x[4])*(x[0]-x[5])*(x[0]-x[6])*
        (x[0]-x[7])*(x[0]-x[8])*(x[0]-x[9])*(x[0]-x[10]))
    S1 = y[1]*((X-x[0])*(X-x[2])*(X-x[3])*(X-x[4])*(X-x[5])*(X-x[6])*(X-x[7])*
        (X-x[8])*(X-x[9])*(X-x[10])) /
        ((x[1]-x[0])*(x[1]-x[2])*(x[1]-x[3])*(x[1]-x[4])*(x[1]-x[5])*(x[1]-x[6])*
        (x[1]-x[7])*(x[1]-x[8])*(x[1]-x[9])*(x[1]-x[10]))
    S2 = y[2]*((X-x[0])*(X-x[1])*(X-x[3])*(X-x[4])*(X-x[5])*(X-x[6])*(X-x[7])*
        (X-x[8])*(X-x[9])*(X-x[10])) /
        ((x[2]-x[0])*(x[2]-x[1])*(x[2]-x[3])*(x[2]-x[4])*(x[2]-x[5])*(x[2]-x[6])*
        (x[2]-x[7])*(x[2]-x[8])*(x[2]-x[9])*(x[2]-x[10]))
    S3 = y[3]*((X-x[0])*(X-x[1])*(X-x[2])*(X-x[4])*(X-x[5])*(X-x[6])*(X-x[7])*
        (X-x[8])*(X-x[9])*(X-x[10])) /
        ((x[3]-x[0])*(x[3]-x[1])*(x[3]-x[2])*(x[3]-x[4])*(x[3]-x[5])*(x[3]-x[6])*
        (x[3]-x[7])*(x[3]-x[8])*(x[3]-x[9])*(x[3]-x[10]))
    S4 = y[4]*((X-x[0])*(X-x[1])*(X-x[2])*(X-x[3])*(X-x[5])*(X-x[6])*(X-x[7])*
        (X-x[8])*(X-x[9])*(X-x[10])) /
        ((x[4]-x[0])*(x[4]-x[1])*(x[4]-x[2])*(x[4]-x[3])*(x[4]-x[5])*(x[4]-x[6])*
        (x[4]-x[7])*(x[4]-x[8])*(x[4]-x[9])*(x[4]-x[10]))
    S5 = y[5]*((X-x[0])*(X-x[1])*(X-x[2])*(X-x[3])*(X-x[4])*(X-x[6])*(X-x[7])*
        (X-x[8])*(X-x[9])*(X-x[10])) /
        ((x[5]-x[0])*(x[5]-x[1])*(x[5]-x[2])*(x[5]-x[3])*(x[5]-x[4])*(x[5]-x[6])*
        (x[5]-x[7])*(x[5]-x[8])*(x[5]-x[9])*(x[5]-x[10]))
    S6 = y[6]*((X-x[0])*(X-x[1])*(X-x[2])*(X-x[3])*(X-x[4])*(X-x[5])*(X-x[7])*
        (X-x[8])*(X-x[9])*(X-x[10])) /
        ((x[6]-x[0])*(x[6]-x[1])*(x[6]-x[2])*(x[6]-x[3])*(x[6]-x[4])*(x[6]-x[5])*
        (x[6]-x[7])*(x[6]-x[8])*(x[6]-x[9])*(x[6]-x[10]))
    S7 = y[7]*((X-x[0])*(X-x[1])*(X-x[2])*(X-x[3])*(X-x[4])*(X-x[5])*(X-x[6])*
        (X-x[8])*(X-x[9])*(X-x[10])) /
        ((x[7]-x[0])*(x[7]-x[1])*(x[7]-x[2])*(x[7]-x[3])*(x[7]-x[4])*(x[7]-x[5])*
        (x[7]-x[6])*(x[7]-x[8])*(x[7]-x[9])*(x[7]-x[10]))
    S8 = y[8]*((X-x[0])*(X-x[1])*(X-x[2])*(X-x[3])*(X-x[4])*(X-x[5])*(X-x[6])*
        (X-x[7])*(X-x[9])*(X-x[10])) /
        ((x[8]-x[0])*(x[8]-x[1])*(x[8]-x[2])*(x[8]-x[3])*(x[8]-x[4])*(x[8]-x[5])*
        (x[8]-x[6])*(x[8]-x[7])*(x[8]-x[9])*(x[8]-x[10]))
```

```
    S9 = y[9]*((X-x[0])*(X-x[1])*(X-x[2])*(X-x[3])*(X-x[4])*(X-x[5])*(X-x[6])*
        (X-x[7])*(X-x[8])*(X-x[10])) /
        ((x[9]-x[0])*(x[9]-x[1])*(x[9]-x[2])*(x[9]-x[3])*(x[9]-x[4])*(x[9]-x[5])*
        (x[9]-x[6])*(x[9]-x[7])*(x[9]-x[8])*(x[9]-x[10]))
    S10 = y[10]*((X-x[0])*(X-x[1])*(X-x[2])*(X-x[3])*(X-x[4])*(X-x[5])*(X-x[6])*
        (X-x[7])*(X-x[8])*(X-x[9])) /
        ((x[10]-x[0])*(x[10]-x[1])*(x[10]-x[2])*(x[10]-x[3])*(x[10]-x[4])*
        (x[10]-x[5])*(x[10]-x[6])*(x[10]-x[7])*(x[10]-x[8])*(x[10]-x[9]))

    return S0 + S1 + S2 + S3 + S4 + S5 + S6 + S7 + S8 + S9 + S10

X_rez = np.arange(-np.pi/3, np.pi/3, 0.01)
Y_rez = [L(x_i) for x_i in X_rez]

plt.grid()
plt.axis("equal")
plt.plot(X_rez, Y_rez, color = "blue")
```

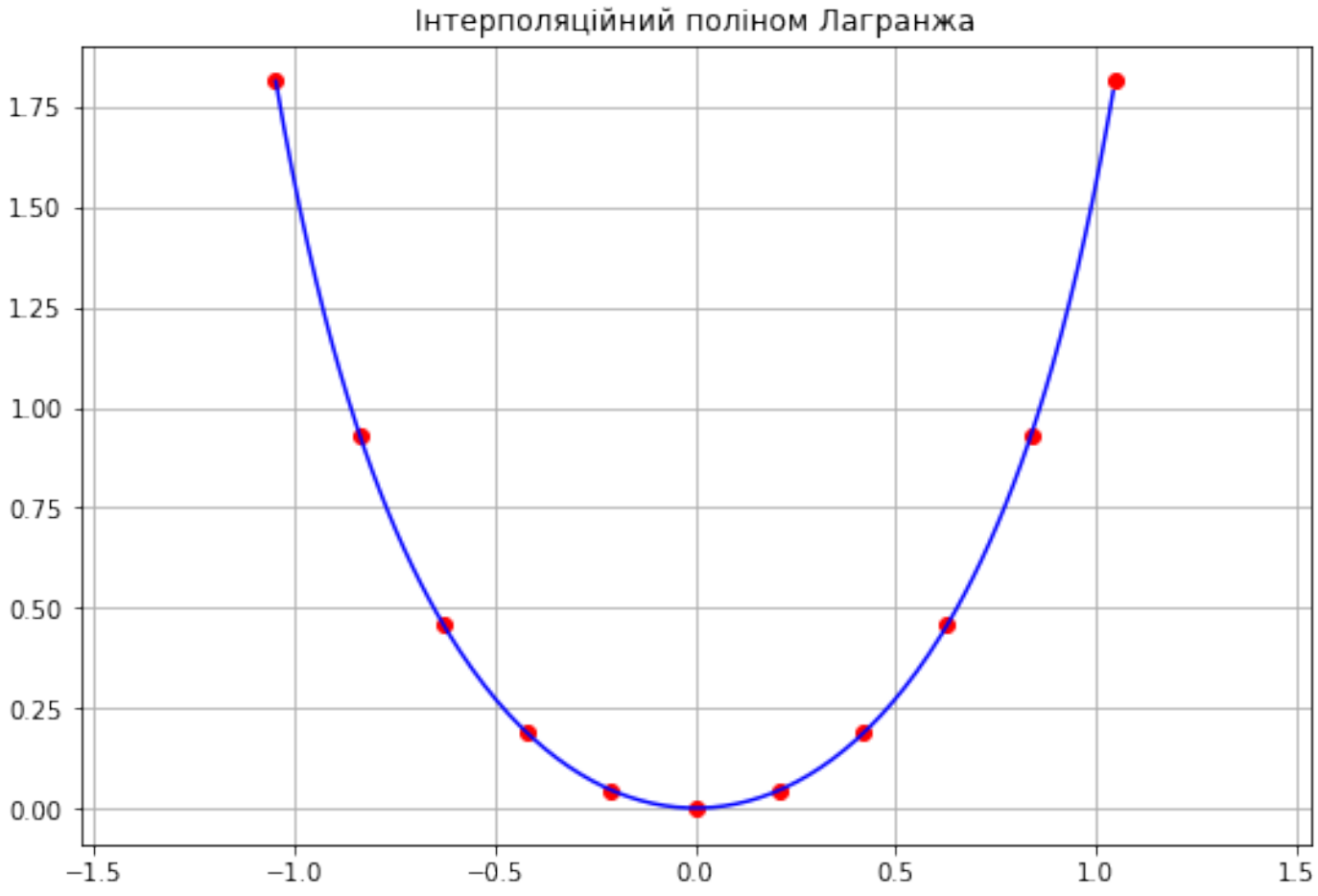

Рис. 1: Результати інтерполяції поліномом Лагранжа

Сплайн-інтерполяція

```
import numpy as np
import matplotlib.pyplot as plt

plt.figure(figsize=(9, 6))

def f(x):
    return x*np.tan(x)

x = np.arange(-np.pi/3, np.pi/3, np.pi/15)
x = np.append(x, np.pi/3)
y = [f(x_i) for x_i in x]

plt.plot(x, y, "o", color = "red", markersize = 6)

a,b,c,d,h = [], [], [], [], []

for i in range(0,10):
    a.append(y[i])

for i in range(0,10):
    h.append(x[i+1]-x[i])

C_a = np.array([[1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                [h[1], 2*(h[1]+h[2]), h[2], 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                [0.0, h[2], 2*(h[2]+h[3]), h[3], 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                [0.0, 0.0, h[3], 2*(h[3]+h[4]), h[4], 0.0, 0.0, 0.0, 0.0, 0.0],
                [0.0, 0.0, 0.0, h[4], 2*(h[4]+h[5]), h[5], 0.0, 0.0, 0.0, 0.0],
                [0.0, 0.0, 0.0, 0.0, h[5], 2*(h[5]+h[6]), h[6], 0.0, 0.0, 0.0],
                [0.0, 0.0, 0.0, 0.0, 0.0, h[6], 2*(h[6]+h[7]), h[7], 0.0, 0.0],
                [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, h[7], 2*(h[7]+h[8]), h[8], 0.0],
                [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, h[8], 2*(h[8]+h[9]), h[9]],
                [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0]])

C_b = []
C_b.append(0.0)
for i in range(1,9):
    C_b.append(3*(((y[i+1]-y[i])/h[i+1]) - ((y[i]-y[i-1])/h[i])))
C_b.append(0.0)

c = np.linalg.solve(C_a, C_b)

for i in range(0,9):
    b.append((y[i+1]-y[i])/h[i] - h[i]*(c[i+1]+2*c[i])/3)
b.append((y[10]-y[9])/h[9] - 2*h[9]*c[9]/3)

for i in range(0,9):
    d.append((c[i+1]-c[i])/(3*h[i]))
d.append(-c[9]/(3*h[9]))

def S(a,b,c,d,h,i,X):
    return a[i] + b[i]*(X-x[i]) + c[i]*pow((X-x[i]),2) + d[i]*pow((X-x[i]),3)
```

```
plt.grid()
plt.axis("equal")

for i in range(10):
    X_rez = np.arange(x[i], x[i+1]+0.01, 0.01)
    Y_rez = [S(a,b,c,d,h,i,x_i) for x_i in X_rez]

    plt.plot(X_rez, Y_rez, color = "blue")
```

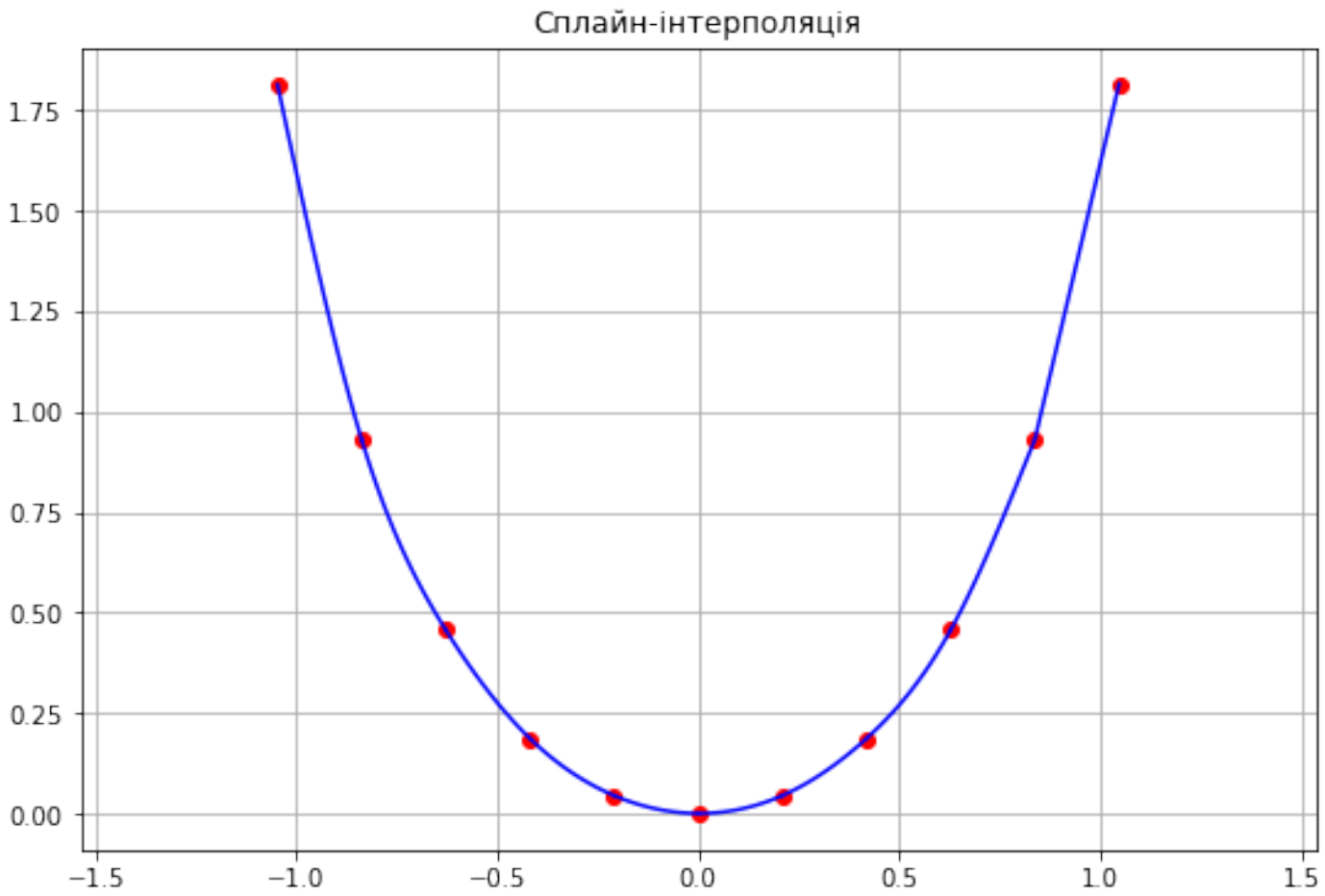

Рис. 2: Результати сплайн-інтерполяції

Похибки інтеполяції

```
plt.figure(figsize=(14, 4.5))

def oversight_L(X):
    return abs(f(X)-L(X))

plt.subplot(1, 2, 1)
plt.grid()

X_rez  = np.arange(-np.pi/3, np.pi/3+0.01, 0.01)
Y_rez = [oversight_L(x_i) for x_i in X_rez]

plt.plot(X_rez, Y_rez, color = "blue")
```

```
plt.subplot(1, 2, 2)
plt.grid()

for i in range(10):
    X_rez = np.arange(x[i], x[i+1]+0.01, 0.01)
    Y_rez = [abs(f(x_i) - S(a,b,c,d,h,i,x_i)) for x_i in X_rez]
    plt.plot(X_rez, Y_rez, color = "red")
```

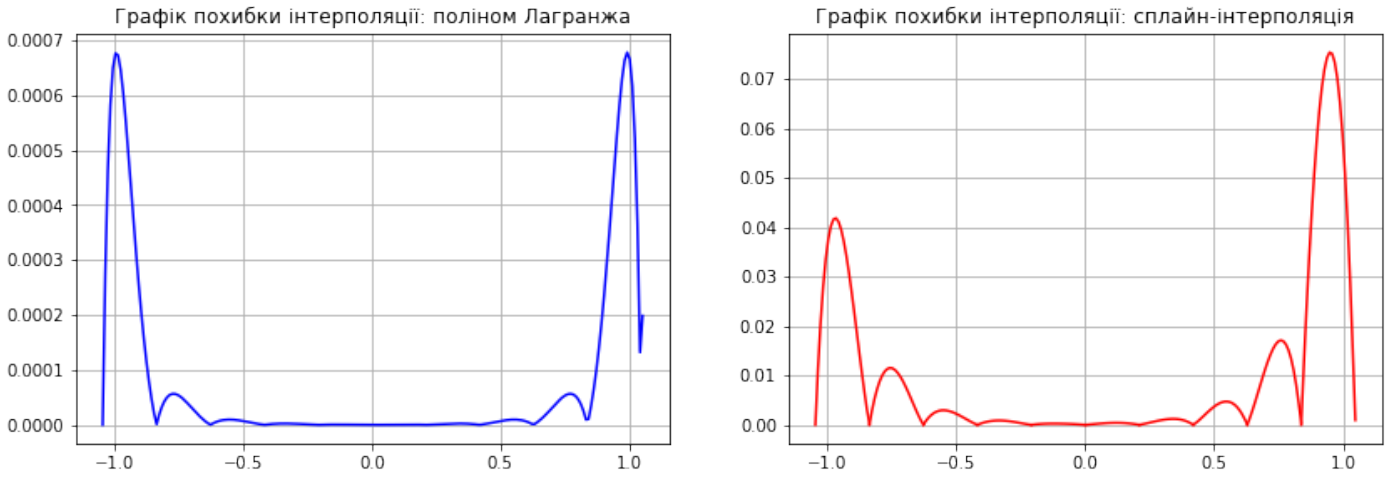

Рис. 3: Графіки похибок інтерполяції

№	**Спосіб інтерполяції**	**Значення похибки**
ε_1	Поліном Лагранжа	0.0006779999272341
ε_2	Сплайн-інтерполяція	0.0753149362380392

Табл. 2: Значення похибок інтерполяції

Контрольні запитання

1. *Чи будуть відрізнятись поліноми Ньютона та Лагранжа одного степеня, побудовані для однієї системи вузлів?*

   Згідно теореми про єдиність інтерполяційного поліному, поліноми Ньютона та Лагранжа одного степеня, побудовані для однієї системи вузлів, відрізнятись не будуть.

2. *У чому перевага побудови поліному Ньютона порівняно із побудовою поліному Лагранжа?*

   У випадку побудови поліному Лагранжа додавання до системи іще одного вузла інтерполяції змусить заново перераховувати й видозмінювати кожен з формуючих доданків поліному, а відтак – і весь існуючий поліном.

Натомість в аналогічній ситуації у випадку побудови поліному Ньютона достатньо просто додати до вже існуючого поліному лише один новий доданок.

3. *Яка кількість вузлів необхідна для побудови інтерполяційного поліному порядку n?*

   Для побудови інтерполяційного поліному порядку n необхідно визначити $n+1$ вузлів інтерполяції.

4. *Запишіть систему рівнянь для знаходження коефіцієнтів сплайнів другого порядку на двох сусідніх відрізках.*

   Як уже зазначалося, ідея сплайн-інтерполяції полягає у побудові поліномів між парами сусідніх вузлів інтерполяції, причому для кожної пари вузлів будується свій поліном.

   Найпоширеніший у практиці є кубічний сплайн, який і був розглянутий у цьому практикумі. Стосовно ж квадратичного сплайна, то для його побудови необхідно побудувати n многочленів другої степені такого виду:

$$S(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2, \text{ де } x_i \leqslant x \leqslant x_{i+1}$$

   Або ще зустрічається така аналогічна форма запису:

$$S(x) = a_i + b_i x + c_i x^2$$

   В обидвох формах запису першочерговою задачею побудови сплайну є знаходження коефіцієнтів a_i, b_i, c_i. Як і раніше, пошук цих величин відбувається із накладених обмежень на вираз $S(x)$:

   - у сусідніх вузлах інтерполяції x_i та x_{i+1} побудований поліном $S(x)$ має задовільняти значенням y_i та y_{i+1};
   - має виконуватися умова неперервності першої похідної у вузлах інтерполяції, тобто умова гладкості кривої в усіх точках. Інакше кажучи, зліва і справа від вузла інтерполяції перша похідна сусідніх сплайнів має бути рівною: $S'_i(x_i) = S'_{i+1}(x_i)$.
   - перші похідні сплайну $S(x)$ та початкової функції $F(x)$ в початковій точці x_0 мають збігатися: $S'(x_0) = F'(x_0)$.

   Отже, із заданих трьох умов складається система обмежень на шукані коефіцієнти a_i, b_i, c_i. Приклад побудови квадратичного спланй розглянутий у цьому джерелі.