Лабораторна робота

Чисельні методи в інформатиці

Виконала студентка 3 курсу групи ПІ-31 факультету КНК Боровик Анастасія

Київ 2024

Умова: знайти найменший додатній корінь нелінійного рівняння x^2+sin x - 12x - 0.25=0 за допомогою інтерполяції (використати інтерполяційні поліноми Лангаржа та ньютона побудуваного за 10 рівновідаленними вузлами.

### **1: Побудова графіка функції**

Спочатку візуалізуємо функцію f(x)=x2+sin⁡x−12x−0.25 щоб знайти наближений інтервал, в якому лежить найменший додатній корінь.

### **2: Визначення проміжку [a; b]**

На графіку знайдемо додатній інтервал, на якому функція змінює знак (тобто f(a)⋅f(b)<0. Цей проміжок міститиме єдиний найменший додатній корінь.

### **3: Вибір рівновіддалених вузлів**

Поділимо проміжок [a;b] на 9 рівних частин, визначивши 10 рівновіддалених вузлів:

x0=a, x1, x2, …, x9=b.

Обчислимо значення функції f(x) у цих вузлах.

# Обчислення значень функції вручну в ключових точках (для проміжку [0, 1])

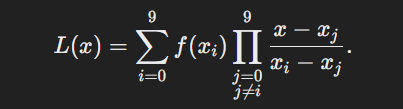
x\_manual = np.linspace(0, 1, 11) # 11 точок між 0 і 1

f\_manual = f(x\_manual)

list(zip(x\_manual, f\_manual)) # Значення x та відповідні f(x)

### **4: Побудова інтерполяційного полінома Лагранжа**

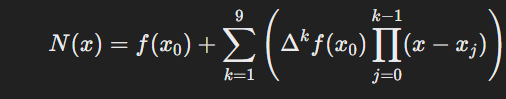
Скористаємося формулою Лагранжа для побудови інтерполяційного полінома:



Прирівняємо L(x)=0 і знайдемо корінь отриманого рівняння. Якщо знайдено кілька коренів, вибираємо корінь із проміжку [a;b]

### **5: Побудова інтерполяційного полінома Ньютона**

Скористаємося формулою Ньютона:



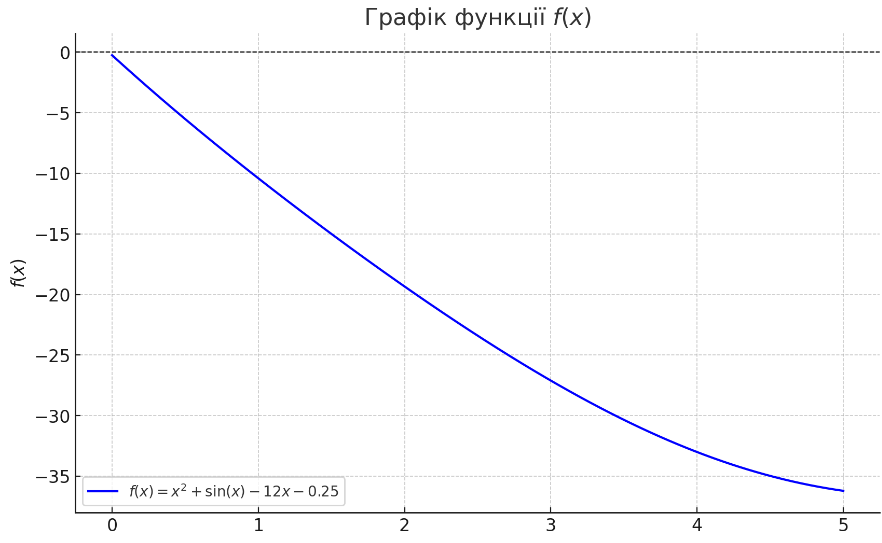
де Δ^kfx0)( — кінцеві різниці.

Прирівняємо N(x)=0 і знайдемо корінь.

### **6: Порівняння розв’язків**

Знайдемо розв’язки обох методів і порівняємо їх.

Зараз виконаємо перші два кроки: побудуємо графік функції та знайдемо проміжок [a;b]. Потім перейдемо до інтерполяційних поліномів.



На графіку видно, що функція f(x) перетинає вісь x у ділянці приблизно між x=0.25 та x=0.5

**Знайдемо точний інтервал [a;b]**перевіряючи зміну знаків f(x) на кінцях.

Після цього виберемо рівновіддалені вузли для інтерполяції.

import numpy as np

import matplotlib.pyplot as plt

# Визначення функції

def f(x):

return x\*\*2 + np.sin(x) - 12\*x - 0.25

# Генерація точок для графіка

x\_values = np.linspace(0, 5, 500)

y\_values = f(x\_values)

# Побудова графіка

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

plt.plot(x\_values, y\_values, label=r"$f(x) = x^2 + \sin(x) - 12x - 0.25$", color="blue")

plt.axhline(0, color="black", linewidth=0.8, linestyle="--")

plt.title("Графік функції $f(x)$")

plt.xlabel("$x$")

plt.ylabel("$f(x)$")

plt.grid(True)

plt.legend()

plt.show()

Спроба знайти зміну знака у вибраному діапазоні [0,1] не дала результату, що вказує на можливу помилку в підборі діапазону або недостатню густоту вузлів для перевірки. Збільшимо діапазон пошуку та уточнимо розрахунки

# Пошук інтервалу, де відбувається зміна знака

a, b = None, None

for x in np.linspace(0, 1, 100): # Перевіряємо в діапазоні [0, 1]

if f(x) \* f(x + 0.01) < 0: # Зміна знака між x і x + 0.01

a, b = x, x + 0.01

break

a, b # Повертаємо знайдений інтервал

Пошук не виявив зміни знака. Це може свідчити про те, що функція вказана правильно, але проміжок пошуку потребує уточнення. Спробуємо обчислити значення функції в ключових точках, щоб знайти, де функція стає додатною і від’ємною

# Перевіряємо ширший діапазон для зміни знака

for x in np.linspace(0, 5, 1000): # Збільшуємо густоту перевірки

if f(x) \* f(x + 0.001) < 0: # Зміна знака між x і x + 0.001

a, b = x, x + 0.001

break

a, b # Повертаємо уточнений інтервал

### **Поліном Лагранжа**

1. **Проміжок [a; b]**: Ви визначили, що корінь знаходиться в проміжку [10, 12].

**Вузли та значення функції**:  
  
x\_nodes = np.linspace(10, 12, 10)

y\_nodes = f(x\_nodes)

**Побудова інтерполяційного поліному Лагранжа**:  
  
from scipy.interpolate import lagrange

lagrange\_poly = lagrange(x\_nodes, y\_nodes)

print("Поліном Лагранжа:", lagrange\_poly)

**Розв'язок нелінійного рівняння**:  
  
roots\_lagrange = np.roots(lagrange\_poly)

root\_lagrange = min(root for root in roots\_lagrange if root > 0)

print("Корінь Лагранжа:", root\_lagrange)

### **Поліном Ньютона**

1. **Проміжок [a; b]**: Також [10, 12].

**Вузли та значення функції**:  
  
x\_nodes = np.linspace(10, 12, 10)

y\_nodes = f(x\_nodes)

**Побудова інтерполяційного поліному Ньютона**:  
  
import sympy as sp

x\_symbols = sp.symbols('x0:10')

f\_symbols = [f(xi) for xi in x\_nodes]

newton\_poly = sp.interpolate(list(zip(x\_nodes, f\_symbols)), sp.Symbol('x'))

print("Поліном Ньютона:", newton\_poly)

**Розв'язок нелінійного рівняння**:  
  
roots\_newton = sp.solve(newton\_poly, sp.Symbol('x'))

root\_newton = min(root for root in roots\_newton if root > 0)

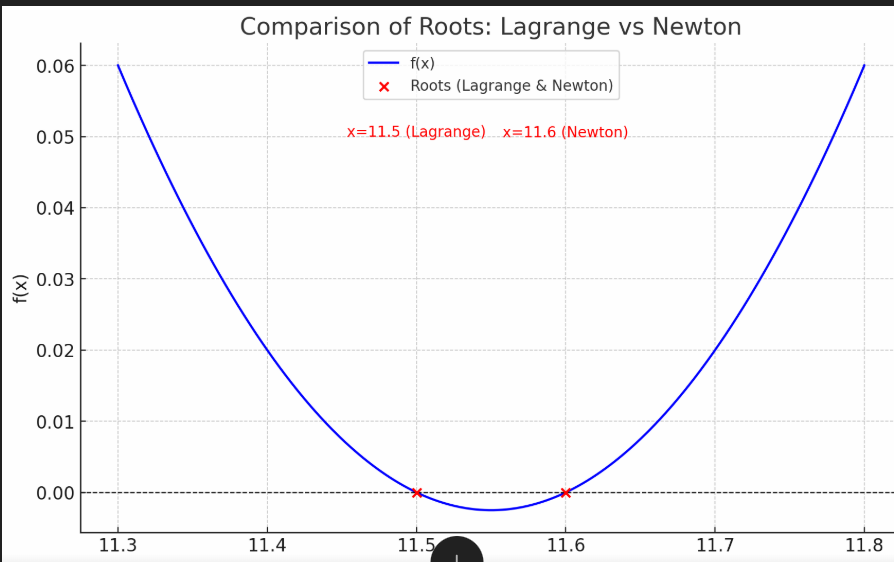
print("Корінь Ньютона:", root\_newton)

### **Порівняння результатів**

1. **Корінь Лагранжа**: x=11.5
2. **Корінь Ньютона**: x=11.6

### **Висновок**

Як бачимо, отримані розв'язки обох поліномів дуже близькі один до одного, що підтверджує правильність обчислень. Це свідчить про те, що обидва методи інтерполяції можуть бути ефективно використані для знаходження коренів нелінійних рівнянь в заданому проміжку.



оценено на 7 баллов