Лихацький Денис Сергійович

КН-42/2

Варіант 14

**Тема**: Чисельне розв’язання нелінійних рівнянь

**Мета**: навчитися розв’язувати задачі одновимірної оптимізації за

допомогою методу простої ітерації та методу Ньютона.

**Завдання**. Розв’язати нелінійне рівняння f(x) = 0 з точністю ϵ=0,001.

**I. Теоретична частина**

**1. Метод простої ітерації**

Ідея методу полягає у перетворенні рівняння f(x)=0 до еквівалентного вигляду x=ϕ(x). Пошук кореня зводиться до знаходження нерухомої точки функції ϕ(x). Формула має вигляд: xn+1​=ϕ(xn​)

Математичне обґрунтування (збіжність): Метод гарантовано збігається, якщо на деякому ізольованому проміжку [a,b], що містить корінь, виконується достатня умова збіжності:

1. Функція ϕ(x) не виводить за межі відрізка [a,b].
2. Похідна функції ϕ(x) обмежена числом q, меншим за одиницю:∣ϕ′(x)∣≤q<1

Ця умова гарантує, що ітераційна послідовність є стискаючим відображенням, а отже, похибка на кожному кроці зменшується, забезпечуючи збіжність до кореня. Швидкість збіжності методу — лінійна.

**2. Метод Ньютона**

Ідея методу полягає у послідовній заміні функції f(x) на її лінійну апроксимацію — дотичну до графіка. Наступне наближення до кореня — це точка перетину дотичної з віссю абсцис.Ітераційна формула, що випливає з рівняння дотичної, має вигляд: xn+1​=xn​−f′(xn​)/f(xn​)​

Математичне обґрунтування (збіжність): Метод збігається, якщо початкове наближення x0​ обрано "достатньо близько" до кореня. Достатньою умовою збіжності є вибір початкової точки x0​ на інтервалі [a,b], де перша та друга похідні (f′(x),f′′(x)) не змінюють знак, і де виконується умова Фур'є: f(x0​)⋅f′′(x0​)>0

Головною перевагою методу є його квадратична швидкість збіжності для простих коренів. Це означає, що кількість вірних значущих цифр у наближенні приблизно подвоюється на кожній ітерації, що забезпечує дуже швидке знаходження розв'язку.

**II. Чисельне розв’язання за допомогою програмного пакета MathCad 14**

**Изображение выглядит как текст, диаграмма, снимок экрана, линия

Содержимое, созданное искусственным интеллектом, может быть неверным.**

**A math equations and formulas on a white background

AI-generated content may be incorrect.**

**Изображение выглядит как текст, снимок экрана, диаграмма, число

Содержимое, созданное искусственным интеллектом, может быть неверным.**

**III. Програмна реалізація алгоритму методу дотичних**

**Програмне розв’язання методом простих ітерацій**

import math

def f(x: float) -> float:

    return 5 \* math.sin(x) - x + 0.5

def f\_prime(x: float) -> float:

    return 5 \* math.cos(x) - 1

def phi\_convergent(x: float, lambda\_val: float) -> float:

    return x + lambda\_val \* f(x)

def get\_user\_input():

    while True:

        try:

            a\_str = input("Введіть початок інтервалу a: ")

            b\_str = input("Введіть кінець інтервалу b: ")

            eps\_str = input("Введіть бажану точність ε: ")

            a = float(a\_str)

            b = float(b\_str)

            epsilon = float(eps\_str)

            if a >= b:

                print("ПОМИЛКА: Початок інтервалу 'a' має бути меншим за кінець 'b'. Спробуйте ще раз.\n")

                continue

            if epsilon <= 0:

                print("ПОМИЛКА: Точність ε має бути додатним числом. Спробуйте ще раз.\n")

                continue

            if f(a) \* f(b) >= 0:

                print(f"ПОМИЛКА: На інтервалі [{a}, {b}] немає гарантованого кореня. Оберіть інший інтервал.\n")

                continue

            return a, b, epsilon

        except ValueError:

            print("ПОМИЛКА: Введено не число. Будь ласка, вводьте тільки числа. Спробуйте ще раз.\n")

def simple\_iteration\_method(x0: float, lambda\_val: float, epsilon: float, max\_iterations: int = 50):

    x\_prev = x0

    print("\nМетод Простої Ітерації")

    print(f"Початкове наближення x0 = {x\_prev}")

    print(f"Розрахований параметр λ = {lambda\_val:.4f}")

    for i in range(1, max\_iterations + 1):

        x\_next = phi\_convergent(x\_prev, lambda\_val)

        error = abs(x\_next - x\_prev)

        print(f"Ітерація {i}: x = {x\_next:.6f}, Похибка = {error:.6f}")

        if error < epsilon:

            print(f"Результат знайдено за {i} ітерацій.")

            return x\_next

        x\_prev = x\_next

    print("Метод не збігся за максимальну кількість ітерацій.")

    return None

if \_\_name\_\_ == "\_\_main\_\_":

    a, b, epsilon = get\_user\_input()

    f\_prime\_a = f\_prime(a)

    f\_prime\_b = f\_prime(b)

    lambda\_val = -1 / f\_prime\_a if abs(f\_prime\_a) > abs(f\_prime\_b) else -1 / f\_prime\_b

    x0\_si = a

    root\_si = simple\_iteration\_method(x0\_si, lambda\_val, epsilon)

    if root\_si is not None:

        print(f"\nЗнайдений корінь: {root\_si:.6f}")

        print(f"Перевірка f(корінь): {f(root\_si):.6f}")

**Результат роботи програми:**

**A screenshot of a computer

AI-generated content may be incorrect.**

Спочатку программа попросить ввести вхідні дані та епсілон. При не вірному вводу даних буде помилка та програма попросить ввести дані ще раз. Наприклад така помилка:

**A black screen with white text

AI-generated content may be incorrect.**

Після введення даних буде виведено результат обчислень в Термінал:

**A screenshot of a computer

AI-generated content may be incorrect.**

**Програмне розв’язання методом Ньютона:**

import math

def f(x: float) -> float:

    return 5 \* math.sin(x) - x + 0.5

def f\_prime(x: float) -> float:

    return 5 \* math.cos(x) - 1

def f\_double\_prime(x: float) -> float:

    return -5 \* math.sin(x)

def get\_user\_input():

    while True:

        try:

            a = float(input("Введіть початок інтервалу a: "))

            b = float(input("Введіть кінець інтервалу b: "))

            epsilon = float(input("Введіть бажану точність ε: "))

            if a >= b or epsilon <= 0 or f(a) \* f(b) >= 0:

                print("ПОМИЛКА: Вхідні дані невірні або на інтервалі немає гарантованого кореня.\n")

                continue

            return a, b, epsilon

        except ValueError:

            print("ПОМИЛКА: Введено не число. Спробуйте ще раз.\n")

def newton\_method(x0: float, epsilon: float):

    x\_prev = x0

    print(f"\nПочаткове наближення x0 = {x\_prev}")

    for i in range(1, 51):

        f\_val = f(x\_prev)

        f\_prime\_val = f\_prime(x\_prev)

        if abs(f\_prime\_val) < 1e-12:

            print("Помилка: похідна близька до нуля.")

            return None

        x\_next = x\_prev - f\_val / f\_prime\_val

        error = abs(x\_next - x\_prev)

        print(f"Ітерація {i}: x = {x\_next:.6f}, Похибка = {error:.6f}")

        if error < epsilon:

            print(f"Результат знайдено за {i} ітерацій.")

            return x\_next

        x\_prev = x\_next

    print("Метод не збігся.")

    return None

a, b, epsilon = get\_user\_input()

x0\_newton = a

root\_n = newton\_method(x0\_newton, epsilon)

if root\_n is not None:

    print(f"\nЗнайдений корінь: {root\_n:.6f}")

    print(f"Перевірка f(корінь): {f(root\_n):.6f}")

**Результат роботи программи:**

**A screen shot of a computer program

AI-generated content may be incorrect.**

Так само программа попросить ввести вхідні дані(Початок та кінець інтервалу та епсілон). При не вірному введені буде помилка(В обох методах різні помилки вводу даних).Після введення вхідних даних програма дасть виконання в Треміналі.

**Висновки**

У ході виконання роботи було розв'язано нелінійне рівняння 5·sin(x) - x + 0.5 = 0 на проміжку [2.5, 2.7] за допомогою двох чисельних методів: методу простої ітерації та методу Ньютона (методу дотичних).

Задана точність обчислень (похибка) становила ε = 0.001.

1. Метод простої ітерації:

Для забезпечення збіжності було використано універсальну ітераційну формулу x = x + λ·f(x) з параметром λ ≈ 0.181 та початковим наближенням x₀ = 2.5.

* Обчислене наближене значення кореня: x ≈ 2.689
* Необхідну точність було досягнуто за 3 ітерації.

2. Метод Ньютона:

Було обрано початкове наближення x₀ = 2.7, що задовольняє умову збіжності f(x₀)·f''(x₀) > 0.

* Обчислене наближене значення кореня: x ≈ 2.689
* Необхідну точність було досягнуто за 2 ітерації.

Порівняння та аналіз результатів:

Обидва методи успішно знайшли корінь рівняння із заданою точністю. Обчислені значення (≈2.689) практично збігаються, що підтверджує коректність розв'язання задачі.

Ключовою відмінністю є швидкість збіжності. Метод Ньютона виявився ефективнішим, оскільки для досягнення точності ε = 0.001 йому знадобилося лише 2 ітерації, тоді як методу простої ітерації — 3 ітерації. Це пояснюється квадратичною швидкістю збіжності методу Ньютона.

Аналіз значення функції в знайдених точках (f(x\*)) також показує, що метод Ньютона дав результат, значно ближчий до істинного нуля (f(x\*) ≈ -0.000002) порівняно з методом простої ітерації (f(x\*) ≈ -0.005), що свідчить про його вищу фактичну точність.Таким чином, для даної задачі метод Ньютона показав себе як більш швидкий та точний інструмент для знаходження наближеного кореня нелінійного рівняння.