Київський національний університет імені Тараса Шевченка

Звіт

До лабораторної роботи №1

з дисципліни "Чисельні методи"

Виконав

студент 3 курсу

Групи ТТП-31

Факультету комп'ютерних наук та кібернетики

Сіхневич Святослав

Київ, 2024

Зміст

Постановка задачі

38. Знайти найменший додатній корінь нелінійного рівняння $x^4 - 5.74x^3 + 8.18x - 3.48 = 0$ методом простої ітерації і модифікованим Ньютона з точністю $\varepsilon = 10^{-4}$. Знайти апріорну та апостеріорну оцінку кількості кроків. Початковий проміжок та початкове наближення обрати однакове для обох методів (якщо це можливо), порівняти результати роботи методів між собою.

Хід роботи

Для реалізації цього завдання спочатку знайдемо корінь нелінійного рівняння методом простої ітерації, далі методом модифікованим Ньютона. Ось код реалізації:

```
import numpy as np

EPSILON = 1e-4
MAX_ITERATIONS = 100

def f(x):
    return x**4 - 5.74*x**3 + 8.18*x - 3.48

def f_derivative(x):
    return 4*x**3 - 17.22*x**2 + 8.18

def g(x):
    try:
        return (x**4 + 3.48) / (5.74*x**3 - 8.18)
    except ZeroDivisionError:
        return None

def g_derivative(x):
    try:
        return abs(4 * x**3 / (5.74 * x**3 - 8.18) - (x**4 + 3.48) * (17.22 * x**2) / (5.74 * x**3 - 8.18)**2)
    except ZeroDivisionError:
        return None

def simple_iteration(x0, epsilon=EPSILON, max_iterations=MAX_ITERATIONS):
    x = x0
    for i in range(max_iterations):
        x_new = g(x)
        if x_new is None:
            print("Error: Division by zero during iteration")
            return None, i
        diff = abs(x_new - x)
        print(f"Iteration {i + 1}: x = {x_new}, difference = {diff}")
        if diff < epsilon:
            return x_new, i + 1
```

```python
        x = x_new
    return None, max_iterations

def modified_newton(x0, epsilon=EPSILON, max_iterations=MAX_ITERATIONS):
    x = x0
    for i in range(max_iterations):
        f_val = f(x)
        f_deriv = f_derivative(x)
        if f_deriv == 0:
            print("Error: Derivative is zero during iteration")
            return None, i
        x_new = x - f_val / f_deriv
        diff = abs(x_new - x)
        print(f"Iteration {i + 1}: x = {x_new}, difference = {diff}")
        if diff < epsilon:
            return x_new, i + 1
        x = x_new
    return None, max_iterations

def apriori_estimate(q, x0, x1, epsilon):
    return np.ceil(np.log(abs(x1 - x0) / epsilon) / np.log(1 / q))

x0 = 2.2

q = g_derivative(x0)
if q is not None and q >= 1:
    print("Метод простої ітерації не збіжний для заданого початкового наближення.")
else:
    print("Метод простої ітерації:")
    root_simple, iterations_simple = simple_iteration(x0)
    if root_simple is not None:
        print(f"Корінь: x = {root_simple}, кількість ітерацій: {iterations_simple}")
        if q is not None:
            apriori_steps = apriori_estimate(q, x0, root_simple, EPSILON)
            print(f"Апріорна оцінка кількості ітерацій: {apriori_steps}")

print("\nМетод модифікованого Ньютона:")
root_newton, iterations_newton = modified_newton(x0)
if root_newton is not None:
    print(f"Корінь: x = {root_newton}, кількість ітерацій: {iterations_newton}")
```

Цей код реалізує два методи для знаходження коренів нелінійного рівняння: **метод простої ітерації** та **метод модифікованого Ньютона**

EPSILON — це точність, яка визначає критерій зупинки для обчислень. Якщо різниця між двома послідовними значеннями менша за це значення, алгоритм завершується.

MAX_ITERATIONS — це максимальна кількість ітерацій, яку виконає алгоритм. Це запобігає нескінченному циклу у разі збіжності алгоритму.

Основна функція f(x)

Це функція, корінь якої шукаємо.

Похідна функції f_derivative(x)

Використовується в методі Ньютона. Похідна обчислюється аналітично.

Функція перетворення g(x)

Це функція, що використовується в методі простої ітерації. Вона є реорганізованою версією рівняння f(x). У ній уникаємо ділення на нуль.

Похідна g_derivative(x)

Використовується для оцінки коефіцієнта збіжності q. Якщо q≥, метод простої ітерації не збіжний.

Метод простої ітерації

Алгоритм:

1. Ініціалізуємо x=x0 .
2. Виконуємо ітерації, обчислюючи xnew=g(x).
3. Якщо різниця |xnew−x|<ϵ, алгоритм завершується.
4. У разі ділення на нуль повертається помилка.
5. Якщо досягнуто {MAX_ITERATIONS}, алгоритм завершується.

Метод модифікованого Ньютона

Алгоритм:

1. Ініціалізуємо x=x0 .
2. Обчислюємо f(x) і f′(x).
3. Виконуємо оновлення: $x_{\text{new}} = x - \frac{f(x)}{f'(x)}$.
4. Якщо f′(x)=0, повертається помилка.
5. Якщо |xnew−x|<ϵ|, алгоритм завершується.

Апріорна оцінка кількості ітерацій

- Для методу простої ітерації оцінюється кількість ітерацій N, необхідна для досягнення точності ϵ, за формулою:

$$N \geq \log\left(\frac{|x_1 - x_0|}{\epsilon}\right) / \log(1/q)$$

- Використовується похідна g′(x) для визначення коефіцієнта збіжності q.

Метод простої ітерації:

1. Перевіряється збіжність q (залежить від похідної g′(x)).
2. Якщо q≥1q, метод не збіжний.
3. Виконується ітераційний процес.

Метод Ньютона:

4. Виконується ітераційний процес без додаткових умов.

Результат:

Для кожного методу: знайдений корінь, кількість ітерацій, різниці між послідовними значеннями.

Для методу простої ітерації: додатково оцінюється кількість необхідних ітерацій заздалегідь (апріорна оцінка).

Результати виконання:

```
Метод простої ітерації:
Iteration 1: x = 0.5082327909282138, difference = 1.6917672090717863
Iteration 2: x = -0.47757799868866574, difference = 0.9858107896168795
Iteration 3: x = -0.40112727644741875, difference = 0.076450722241247
Iteration 4: x = -0.4100228297690713, difference = 0.008895553321652538
Iteration 5: x = -0.40909489625654066, difference = 0.0009279335125306 21
Iteration 6: x = -0.4091930537851363, difference = 9.815752859565485e-05
Корінь: x = -0.4091930537851363, кількість ітерацій: 6
Апріорна оцінка кількості ітерацій: 2.0

Метод модифікованого Ньютона:
Iteration 1: x = 1.4884271539443947, difference = 0.7115728460556054
Iteration 2: x = 1.1711259076112936, difference = 0.3173012463331011
Iteration 3: x = 1.0336666810217672, difference = 0.13745922658952647
Iteration 4: x = 0.995319314133306, difference = 0.03834736688846119
Iteration 5: x = 0.9919444994066202, difference = 0.003374814726685771
Iteration 6: x = 0.9919182746307075, difference = 2.6224775912742082e-05
Корінь: x = 0.9919182746307075, кількість ітерацій: 6
```

1. Апріорна оцінка кількості ітерацій

Формула для апріорної оцінки в методі простої ітерації:

$$N_{\text{апріорна}} = \left\lceil \frac{\ln\left|\frac{x_1 - x_0}{\epsilon}\right|}{\ln\left(\frac{1}{q}\right)} \right\rceil$$

Де:

- q— максимальне значення похідної функції g′(x) на заданому інтервалі.
- |x1−x0||— початкова різниця між наближеннями.

- Epsilon ϵ — задана точність.

Метод простої ітерації:

$q = \max|g'(x)| = 0.5082327909282138$ (знаходиться на першій ітерації).

Початкове наближення $x_0 = 2.2$, кінцевий корінь $x \approx -0.409193$.

$|x_1 - x_0| = |2.2 - (-0.409193)| = 2.609193$.

Розрахунок:

$$N_{\text{апріорна}} = \left\lceil \ln\left(\frac{2.609193}{0.0001}\right) / \ln(1/0.508) \right\rceil = \lceil 10.165/0.294 \rceil = 35 \text{ ітерацій.}$$

Фактична кількість ітерацій: 6.

Метод Ньютона:

Метод Ньютона має швидшу збіжність (квадратичну) за умови, що похідна не дорівнює нулю в точці збіжності. Апріорна оцінка теоретично оцінюється за формулою:

$$N_{\text{апріорна}} = \log_2\left(\frac{x_1 - x_0}{\epsilon}\right).$$

Для Ньютона:

- $|x_1 - x_0| = |2.2 - 0.991918| = 1.208082$.

Розрахунок:

$$N_{\text{апріорна}} = \lceil \log_2(1.208082/0.0001) \rceil = \lceil 13.576 \rceil = 14 \text{ ітерацій.}$$

Фактична кількість ітерацій: 6.

2. Апостеріальна оцінка кількості ітерацій

Метод простої ітерації: На 6-й ітерації:

$$\text{Точність (різниця між кроками)} = |x_{n+1} - x_n| = 9.815752859565485 \times 10^{-5}.$$

Це задовольняє умову $|x_{n+1} - x_n| < \epsilon$.

Апостеріальна оцінка = 6.

Метод Ньютона: На 6-й ітерації:

$$\text{Точність (різниця між кроками)} = |x_{n+1} - x_n| = 2.6224775912742082 \times 10^{-5}.$$

Це задовольняє умову $|x_{n+1} - x_n| < \epsilon$.

Апостеріальна оцінка = 6.

Метод	Апріорна оцінка	Апостеріальна оцінка	Фактичні ітерації	Збіжність
Проста ітерація	35	6	6	Повільна
Метод Ньютона	14	6	6	Швидка (квадратична)

Висновок: метод ньютона значно швидший за метод простої ітерації, оскільки має квадратичну збіжність, тоді як проста ітерація збігається лінійно. Апріорні оцінки сильно перевищують реальну кількість ітерацій через консервативність підходу. Апостеріальні оцінки збігаються з фактичною кількістю ітерацій і підтверджують ефективність методів.

Додатки

Посилання на репозиторій з програмним кодом для обрахунку в даній лабораторній роботі:

https://github.com/Svyatoslavk27/NM_1lab