

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ
СІКОРСЬКОГО»**

**Навчально-науковий інститут прикладного системного аналізу
Кафедра штучного інтелекту**

**Звіт
про виконання лабораторної роботи №3 з дисципліни
«Обчислювальна математика»**

Виконав:

студент II курсу, групи КІ-32

Присяжнюк Владислав

Прийняв:

доцент Квітка О. О.

Варіант 38

Рівняння:

$$0.81 \cdot e^{-0.6x} - 2x = 0$$

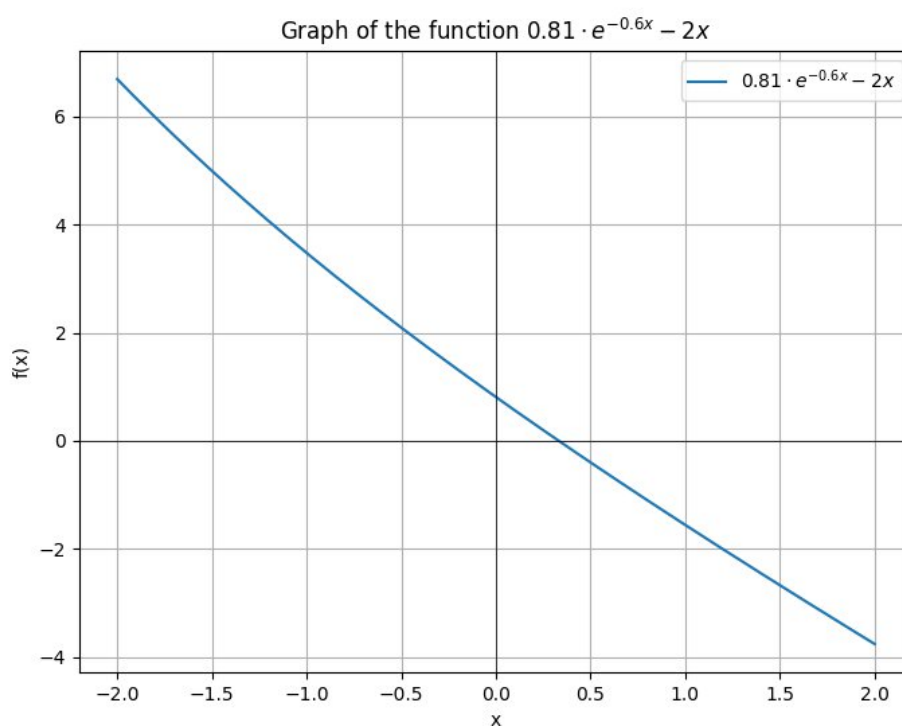


Рис 1.1 – Графік функції побудований за допомогою matplotlib

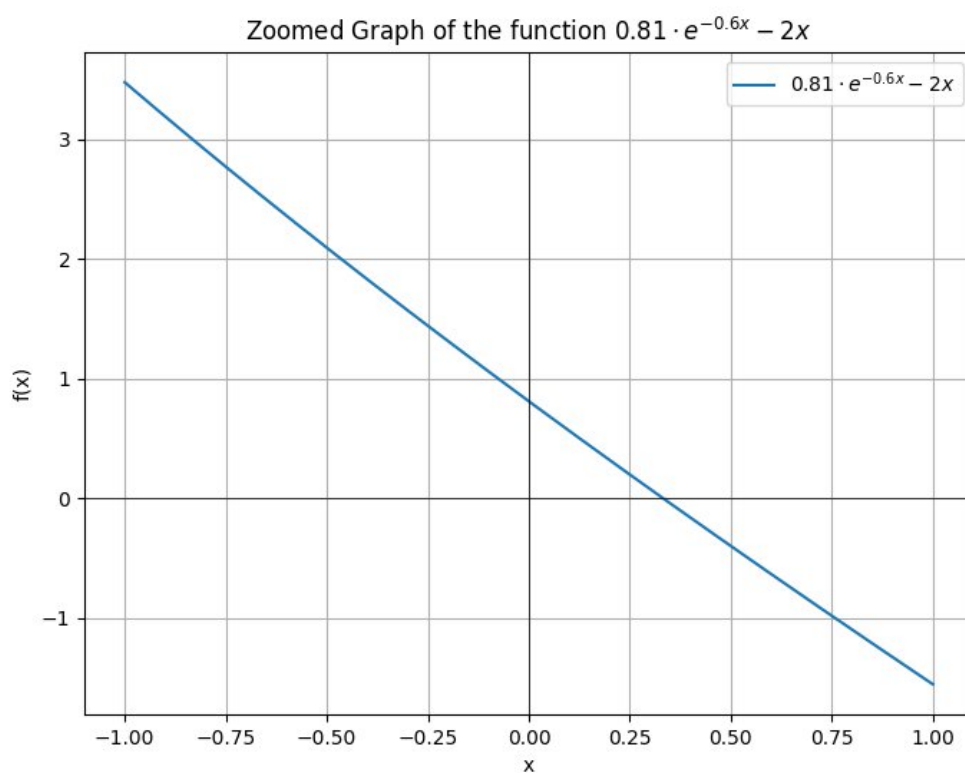


Рис 1.2 – Приближений графік функції

З побудованого графіку можна зробити висновок що дійсний корінь знаходиться на інтервалі $[0.25, 0.5]$.

1. Виконати початкове дослідження заданої функції аналітичним методом.

Щоб визначити екстремуми, необхідно знайти першу похідну функції.

$$f'(x) = -0.486 \cdot e^{-0.6x} - 2$$

Щоб знайти екстремуми, прирівнюємо першу похідну до нуля:

$$-0.486 \cdot e^{-0.6x} - 2 = 0$$

$$e^{-0.6x} = -\frac{2}{0.486} \approx -4.115$$

Експоненціальна функція e^x завжди додатна, тому це рівняння не має дійсних розв'язків, що означає, що екстремумів у цій функції немає. Через відсутність екстремумів, функція або монотонно зростає, або монотонно спадає. Для аналізу знаків функції ми можемо підставити деякі значення x в початкову функцію.

Для $x = 0$:

$$f(0) = 0.81 \cdot e^0 - 2 \cdot 0 = 0.81$$

Для $x = 1$:

$$f(1) = 0.81 \cdot e^{0.6 \cdot 1} - 2 \cdot 1 = -1.555$$

Отже функція стає від'ємною при $x = 1$, це означає, що корінь функції знаходиться на інтервалі $(0, 1)$

Для подальшого наближення варто використовувати чисельні методи.

Частина 2

Методом Ньютона (дотичних) обрахувати корінь рівняння у інтервалі $[0, 1]$ з точністю $\varepsilon = 10^{-3}$

$$f(x) = 0.81 \cdot e^{-0.6x} - 2x$$

Для цього використаємо похідну рівняння отриману в частині 1.

$$f'(x) = -0.486 \cdot e^{-0.6x} - 2$$

Метод Ньютона використовує формулу:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Оскільки ми знаємо приблизний інтервал $[0,1]$ то оберемо початкове наближення $x_0 = 0.5$:

$$x_1 = 0.5 - \frac{0.81 \cdot e^{-0.6(0.5)} - 2(0.5)}{-0.486 \cdot e^{-0.6(0.5)} - 2}$$

Тепер отриманий результат $x_1 = 0.3305$ підставимо у наступну ітерацію:

$$x_2 = 0.3305 - \frac{0.81 \cdot e^{-0.6(0.3305)} - 2(0.3305)}{-0.486 \cdot e^{-0.6(0.3305)} - 2}$$

Отримаємо що $x_2 = 0.3318$, перейдемо до наступної ітерації:

$$x_3 = 0.3318 - \frac{0.81 \cdot e^{-0.6(0.3318)} - 2(0.3318)}{-0.486 \cdot e^{-0.6(0.3318)} - 2}$$

Під час цього обрахунку ми отримаємо $x_3 = 0.3318$, це значить що наша відповідь збігається і починає відрізнятись тільки дуже малими числами. В такому випадку можна сказати що ми досягли заданої точності, оскільки різниця між останніми двома ітерація буде явно менша ніж $\varepsilon = 10^{-3}$.

Для зручності побудуємо таблицю:

Ітерації	Корінь
0	0.5
1	0.3305
2	0.3318
3	0.3318

У висновку можемо сказати що корінь буде $x \approx 0.3318$. *Примітка: результат округлений за до трьох значущих цифр за правилом парної цифри.*

Давайте підставимо наш отриманий корінь у рівняння для визначення правильності попередніх обрахунків:

$$f(0.3318) = 0.81 \cdot e^{-0.6(0.3318)} - 2(0.3318)$$

Отримана відповідь $f(x) = 0.00018$, що підтверджує дуже близьке наближення до кореня.

Метод Хорд

Ітераційний числовий метод знаходження наближених коренів нелінійного алгебраїчного рівняння. Його мета — наблизити корінь функції шляхом побудови послідовності відрізків (хорд), які перетинають графік функції $f(x)$ та сходяться до кореня

Тепер перейдемо до побудови алгоритму на мові програмування Python.

Код програми:

```
import math

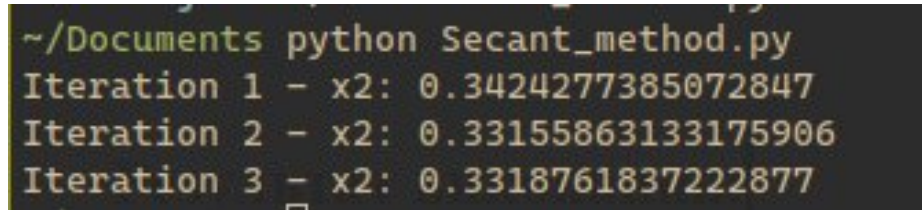
function = lambda x: 0.81 * math.exp(-0.6 * x) - (2 * x)
x0 = 0
x1 = 1
tol = 10**-3
max_iter = 100

def secant_method(f, x0, x1, tol, max_iter):
    for i in range(max_iter):
        x2 = x1 - f(x1) * (x1 - x0) / (f(x1) - f(x0))
        if abs(x2 - x1) < tol:
            print(f'Iteration {i + 1} - x2: {x2}')
```

```
    return x2
    x0, x1 = x1, x2
    print(f'Iteration {i + 1} - x2: {x2}')
    return None
```

```
secant_method(function, x0, x1, tol, max_iter)
```

Результат виконання:



```
~/Documents python Secant_method.py
Iteration 1 - x2: 0.3424277385072847
Iteration 2 - x2: 0.33155863133175906
Iteration 3 - x2: 0.3318761837222877
```

Рис 1.3 – Результат виконання методу Хорд

Видно що результат обрахунку коріння збігся з результатами попередніх методів на 3-ій ітерації.

Метод простої ітерації

Чисельний метод для знаходження коренів нелінійних рівнянь. Цей метод полягає в перетворенні вихідного рівняння у форму, яка дозволяє побудувати послідовність наближень до кореня за допомогою ітераційного процесу.

Код програми:

```
import math

def function(x):
    return 0.81 * math.exp(-0.6 * x) - (2 * x)

def simple_iteration(x0, eps=10**-3, max_iter=100):
    x_prev = x0
    iteration = 0
```

```

while iteration < max_iter:
    x_next = x_prev + 0.05 * function(x_prev)

    if abs(x_next - x_prev) < eps and abs(function(x_next)) < eps:
        return x_next, iteration

    x_prev = x_next
    iteration += 1

    print(f'Iteration {iteration}: x = {x_next:.6f}, f(x) = {function(x_next):.6f}')

return x_next, iteration

if __name__ == "__main__":
    x0 = 0.5
    root, iterations = simple_iteration(x0)
    print(f'\nFinal Results:')
    print(f'Root found: {root:.6f}')
    print(f'Iterations: {iterations}')
    print(f'f(root) = {function(root):.6f}')

```

Результат роботи програми:

```

Iteration 46: x = 0.332352, f(x) = -0.001142
Iteration 47: x = 0.332295, f(x) = -0.001005

Final Results:
Root found: 0.332245
Iterations: 47
f(root) = -0.000885

```

Рис 1.4 – Результат роботи метода простої ітерації

Точність: 10^{-3}	Метод Ньютона	Метод Хорд	Метод простої ітерації
---------------------	---------------	------------	------------------------

Кількість ітерацій	3	3	47
Корінь	0.3318	0.3318	0.3322

Висновки:

У процесі виконання лабораторної роботи було досліджено різні чисельні методи знаходження коренів нелінійних рівнянь: метод Ньютона, метод хорд та метод простої ітерації. Найбільш ефективним методом для даного рівняння є метод Ньютона, але його застосування доцільне тільки за умови, що похідну функції можна обчислити. Метод хорд є гарною альтернативою для ситуацій, коли похідну отримати складно. Метод простої ітерації є найпростішим у реалізації, але менш ефективним у випадках, коли потрібна висока точність результату.