BISECTION METHOD AND NEWTON-RAPHSON

Muthoni Ivy SCT211-0011/2021

March 2023

1 BISECTION METHOD

This method involves dividing the interval containing the root into subintervals and determining which subinterval the root is in.In my case, the function was:

$$x^4 - x^3 - 10$$

The tolerance was given as $1e-6$ while the maximum iteration was 100.The code was to calculate how many times it took to get to the root.And in this case, this was found to be 11. The code was also to be timed and the execution time determined,hence the use of the use of the timeit function.The total execution time was found to be 0.00043 seconds. The root was found at:

$$x = 2.092090$$

The python code

```
import timeit
import math

def f(x):
return x4 - x3 - 10

a = 2
b = 3
tolerance = 1e-6
max_iterations = 100
count = 0

def bisection():
global a, b, count
for i in range(max_iterations):
```

```
c = (a + b) / 2
count += 1
if abs(f(c)) < tolerance:
print(f"Root found at x = {c:.6f}")
break
elif f(c) * f(a) < 0:
b = c
else:
a = c

Measure the execution time of bisection
execution_time = timeit.timeit(bisection, number=1)

Print the execution time
print(f"Execution time: {execution_time:.5f} seconds")

Print count
print(count)
```

2 NEWTON-RAPHSON METHOD

This method is used to find the roots of differentiable functions.Uses idea of tangent line.The function in question is:

$$x^4 - x^3 - 10$$

On differentiating,the function will be $4x^3 - 3x^2$.The tolerance in the question is $1e-6$, maximum iteration is 100 while the initial count is set to zero.We are supposed to determine how many times it will take to get to the root and that's why we use count.The count eventually sums up to:11 The time for execution is 0.0043, which is got by the timeit function. The root is found at:

$$x = 2.092090$$

3 The python code

```
import timeit
import math
def f(x):
    return x**4 - x**3 -10
def df(x):
    return 4*x**3 - 3*x**2

x0=1
```

```
tolerance=1e-6
max_iteration = 100
count =0

def newton_raphson():
    global x0, count,tolerance,max_iteration
    for i in range (max_iteration):
        fx=f(x0)
        dfx =df(x0)
        x1 =x0 -f(x0)/df(x0)
        count+=1
        if abs (f(x1)) < tolerance:
            print(f"Root found at x ={x1:.6f}")
            break
        else:
            x0 = x1
# Measure the execution time of newton_raphson
execution_time = timeit.timeit(newton_raphson, number=1)

# Print the execution time
print(f"Execution time: {execution_time:.5f} seconds")


print(f"count is: {count} in total")
```