San Francisco Bay University

PRESENTED BY AAYUSH MAHARJAN  
ID\_- 19801

PRESENTED TO PROFESSOR ALEX YANG

# MATH201 Calculus-I PLO Signature Assignment

**Due day: 8/19/2023**

**Part I. Individual Homework Assignment**

## Question:

1.Use Newton’s method to approximate the negative root of the equation 𝑒𝑥 = 4 − 𝑥2 correct to six decimal places. And verify your answer by the plot in Excel.

Answer:

Explanation on how to do the following question. Here, we must use Newton’s method to find the negative root of the equation 𝑒𝑥 = 4 − 𝑥2

Newton’s method that is mentioned here is the Newton Raphson’s method which states that after each iteration the value will be closer to the root of the equation. The general formula is given below.

Here is the next guess of the root for the equation and is the current guess. And means the value of (f (current guess)) and is the value of first derivative of

As this is a second order function, we will have two roots, one positive and one negative. When we take an initial guess that is closer to the negative root then we will get the negative root but if we take an initial guess that is closer to the positive root then we will get the positive root.

Now, let’s get started.

We are given the equation 𝑒𝑥 = 4 − 𝑥2

We can also write this equation as

= 0

Again, this is our function, so we need to find the first derivative of this function with respect to x.

We have the following variables:

So, we need to put in an initial guess first where x ≠ 0. Then when we iterate, we will go closer and closer to the root. As the question asks us to get the negative root, we have to initialize the value of the x closer to the negative root. We can initialize it at -2 then proceed from there.

Here when x = -2 and n = 0.

Zeroth iteration:

Value of when x = -2:

Value of when x = -2:

Putting the values in the Newton Raphson’s equation.

So, after the first iteration we get the root value to be -1.964981365 but we are told to match the root value correct to six decimal places. The actual root value is - 1.964635 up to six decimal places. So, we need to iterate again.

Here when x = and n = 1.

First iteration:

Value of when x = :

Value of when x = :

Putting the values in the Newton Raphson’s equation.

After the second iteration we get root value to be but we still are off by some amount. So, we need to iterate again.

Here when x = and n = 2.

Second iteration:

Value of when x = :

Value of when x = :

Putting the values in the Newton Raphson’s equation.

After the third iteration we finally get the negative root of the equation that is correct up to six decimal places.

Now I am choosing python to write a code for the data points for x and y values so I can plot the graph on excel.

import math

def newton\_raphson\_method(x\_value\_guess):

points\_for\_graph = [] # list which will store the coordinates

tolerance\_for\_right\_answer = 1e-6

while (True):

function\_x = x\_value\_guess\*\*2 + math.exp(x\_value\_guess) - 4 # f(x) is x^2 + e^x - 4

first\_derivate\_function\_x = 2 \* x\_value\_guess + math.exp(x\_value\_guess) # f'(x) is 2x + e^x

# here x(n+1) is next guess value which is stored in variable next\_value\_of\_x

# x(n) i current guess value which is stored in variable x\_value\_guess

# f(xn) is the function value of x(n) stored in function\_x variable

# f'(xn) is the function value of first derivative of f(xn) which is stored in first\_derivate\_function\_x

next\_value\_of\_x = x\_value\_guess - function\_x / first\_derivate\_function\_x # this is because x(n + 1) = xn - f(xn) / f'(xn)

points\_for\_graph.append({'x': x\_value\_guess, 'y': function\_x})

if (abs(next\_value\_of\_x - x\_value\_guess) < tolerance\_for\_right\_answer):

return x\_value\_guess, points\_for\_graph

x\_value\_guess = next\_value\_of\_x

def main():

root\_approximate\_1, points\_1 = newton\_raphson\_method(1)

root\_approximate\_2, points\_2 = newton\_raphson\_method(-2) # Initial guess for the second root

print("x", "y")

for point in points\_1:

x = point['x']

y = point['y']

print(f"{x} {y}")

print(f"Root approximate 1 of function x^2 + e^x - 4 is {root\_approximate\_1}.")

print("\nx", "y")

for point in points\_2:

x = point['x']

y = point['y']

print(f"{x} {y}")

print(f"Root approximate 2 of function x^2 + e^x - 4 is {root\_approximate\_2}.")

main()

Output that this code gives is the following code:

I have used the python code to print both the roots positive and negative but we are interested in the negative only.

A computer screen with numbers and equations

Description automatically generated

The x, y are [{x: -2, y: 0.135335283236612}, {x: -1.96498136496819, y: 0.00131026312184445}, {x: -1.9646356312464, y: 1.2790947145902E-07}]

Plotting these in graph in excel it will gives us the following graph:

A graph on a graph

Description automatically generated

But putting the values from negative to positive we will get the following curve.

A graph on a graph

Description automatically generated

If we draw a trend line we can see that it will be a parabola. And this graphing calculator verifies it.

A graph of a function

Description automatically generated

In conclusion, through Newton’s Raphson method we were able to find the roots of the equation = 0. The negative root is close to six decimals. We found the root after 3 iterations. First, we found the first derivative of the function then we used the Newton Raphson’s method to find the root of the equation. We first guessed the root to be -2 then after iteration we found the root to be . After all these we also made a python program to get the data points. We then plotted the data points in excel. There is a graph posted which reflects the data points. And we also verified the graph by plotting the function in a graphing calculator.