Week 9: Newton's Method

Graded

Student

Harper Chen

Total Points

70 / 100 pts

Question 1

NM illustration

0 / 10 pts

- **− 0 pts** Correct
- **− 9 pts** This is not Newton's Method - see the examples in Piazza.
- **− 10 pts** This doesn't look like #1?

Question 2

NM illustration 2

0 / 10 pts

- **− 0 pts** Correct
- **− 0 pts** Click here to replace this description.
- **− 10 pts** Please email me - I think you m ay have misunderstood the instructions.

Question 3

NM Fail

10 / 10 pts

- **− 0 pts** Correct
- ✔ **− 0 pts** Not a tangent line

Please email me if you would like to redo this Week's Activities -

Question 4

NW Explore

0 / 10 pts

- **− 0 pts** Correct
- ✔ **− 10 pts** not Newton's Method.

Question 5

Screenshot NM code

10 / 10 pts

- ✔ **− 0 pts** Correct

Question 6

Answer questions from 6 about how it works

10 / 10 pts

- ✔ **− 0 pts** Correct

Question 7

Answer questions from 7 about how it works

10 / 10 pts

✔ **− 0 pts** Correct

Question 8

Pretty Graphs

30 / 30 pts

✔ **− 0 pts** Correct

− 30 pts missing??

− 0 pts Click here to replace this description.

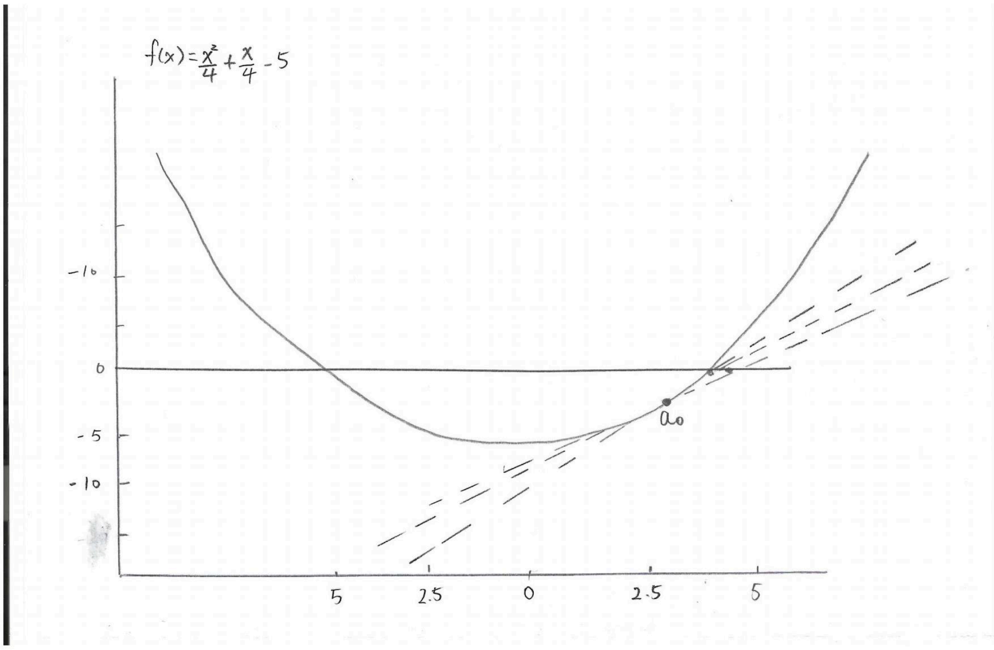
Question assigned to the following page: 1

week 9

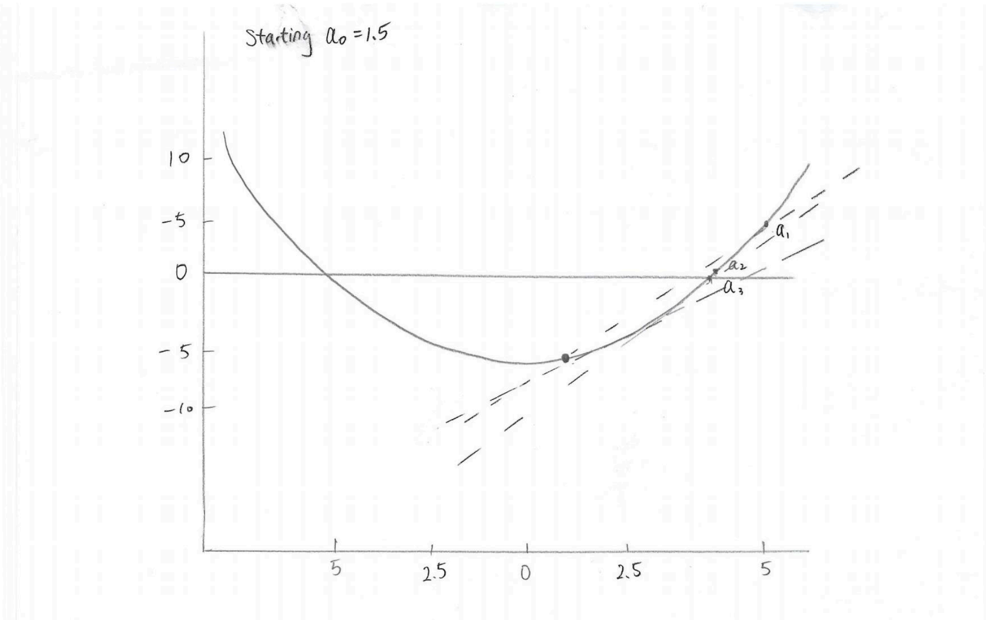
October 30, 2024

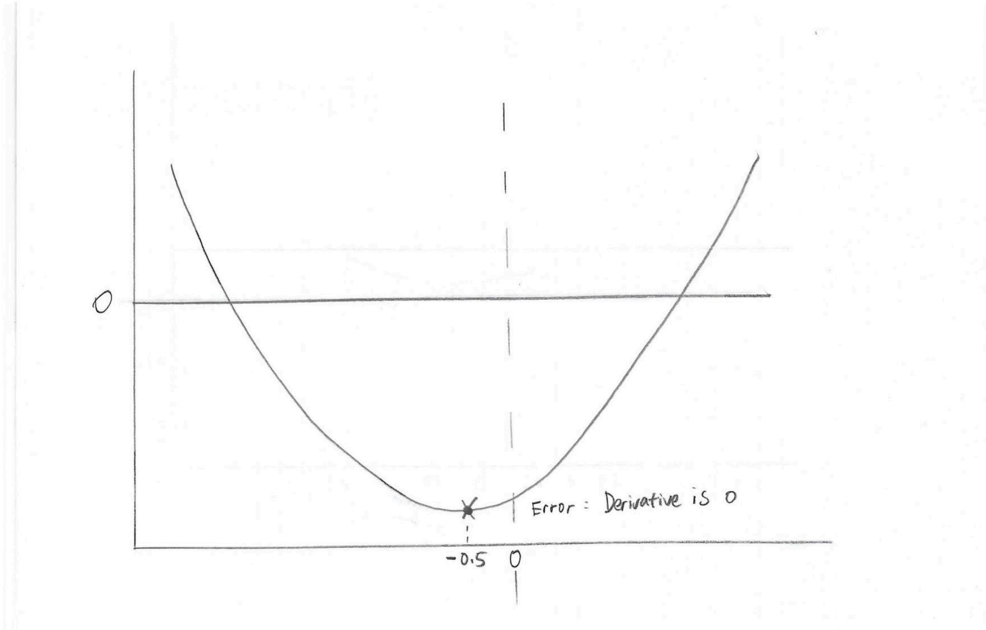
[26]:
```
from IPython.display import Image, display

display(Image(filename="1.jpg"))
display(Image(filename="2.jpg"))
display(Image(filename="3.jpg"))
```



Questions assigned to the following page: 3, 4, and 2


Starting $a_0 = 1.5$
10
-5
0
-5
-10
5
2.5
0
2.5
5
a_1
a_2
a_3


0
Error : Derivative is 0
-0.5
0

Question assigned to the following page: 5

0.0.1 Question 2:

To test Newton's Method with a different starting point between 0 and the approximate root (r = 4), I selected (x = 1.5) as the initial value. Starting from this point, I applied Newton's Method iteratively to observe if it converges to the same root.

With each iteration, the method produced successive approximations, each closer to (r = 4). The tangent lines at each point led to new x-intercepts, refining the guess for the root. After several steps, the method converged to the root at (r = 4), consistent with the earlier result using different starting points.

for this function, Newton's Method consistently converges to the root at (r = 4) even with various starting points within the range (0, 4).

0.0.2 Question 3:

To explore what happens when I start Newton's Method at a point where the derivative of (f(x)) is zero, I will first analyze the function:

$$f(x) = \frac{x^2}{4} + \frac{x}{4} - 5$$

The derivative of (f(x)) is:

$$f'(x) = \frac{x}{2} + 0.25$$

Setting (f'(x) = 0), And solve for (x):

$$\frac{x}{2} + 0.25 = 0$$

This gives (x = -0.5). Therefore, if start Newton's Method at (x = -0.5), the method will encounter a zero in the denominator of the update formula:

$$a_{n+1} = a_n - \frac{f(a_n)}{f'(a_n)}$$

When (f'(x) = 0), Newton's Method cannot proceed due to a division by zero. This causes the method to fail.

If were coding this method, an appropriate error message would be like the code below.

[15]:
```
# Question 5

def f(x):
    return (x**2) / 4 + (x / 4) - 5

def f_prime(x):
    return (x / 2) + 0.25
```

Questions assigned to the following page: 5, 6, and 7

```
def newtons_method_root(starting_point, tolerance=1e-5, max_iterations=100):
    x = starting_point
    for iteration in range(max_iterations):
        if f_prime(x) == 0:
            return "Error: Derivative is zero at the starting point. Cannot␣
↪proceed with Newton's Method."

        x_next = x - f(x) / f_prime(x)

        if abs(x_next - x) < tolerance:
            return round(x_next, 5)  # return the root to 5 decimal places

        x = x_next  # update x to the next value

    return "Maximum iterations reached. Method did not converge."

try:
    user_starting_point = float(input("Enter a starting point for Newton's␣
↪Method: "))
    root_result = newtons_method_root(user_starting_point)
    print("Root found:", root_result)
except ValueError:
    print("Invalid input. Please enter a numeric starting point.")
```

```
Enter a starting point for Newton's Method:  3
```

```
Root found: 4.0
```

0.0.3 Question 6:

- **Can your code find both roots?**
  Yes, the code can find both roots of the function

$$f(x) = \frac{x^2}{4} + \frac{x}{4} - 5$$

  which are located at $ 4 $ and $ -5 $. The root identified depends on the chosen starting point, demonstrating the effectiveness of Newton's Method in converging to different roots based on initial inputs.

- **How many random inputs does it take for you to find both roots?**
  Through testing with six different starting points (3, 6, -6, 0, 4.5, -4.5), the code successfully located both roots. Specifically, it found the root $ 4 $ from the starting points 3, 6, 0, and 4.5, and the root $ -5 $ from the starting points -6 and -4.5. This suggests that with a few well-chosen starting values, Newton's Method can reliably identify multiple roots of a function.

0.0.4 Question 7:

- **Starting Value for Error**
  Newton's Method will encounter an error if I start at a point where the derivative is zero.

Question assigned to the following page: 7

The derivative of this function is:

$$f'(x) = \frac{x}{2} + 0.25$$

Setting this equal to zero, I find:

$$f'(-0.5) = 0$$

Therefore, starting with (x = -0.5) will cause a division by zero in Newton's Method, leading to an error.

When I input (x = -0.5) as the starting point: - The function evaluates (f(-0.5)) and (f'(-0.5)). - Upon finding that (f'(-0.5) = 0), the method halts and returns the error message, indicating that Newton's Method cannot proceed with this starting point.

[22]:
```
## Qustion 8

import matplotlib.pyplot as plt
import numpy as np

def f(x):
    return (x**2) / 4 + (x / 4) - 5

def f_prime(x):
    return (x / 2) + 0.25

def Newton_Method_Single_Plot(start):
    x_vals = np.linspace(start - 10, start + 10, 100)
    y_vals = f(x_vals)
    x = start

    plt.figure(figsize=(10, 8))
    plt.plot(x_vals, y_vals, color='blue', label="f(x)")

    for i in range(8):
        f_x = f(x)
        tangent_y = f_x + f_prime(x) * (x_vals - x)

        plt.plot(x_vals, tangent_y, linestyle='--', label=f"Tangent at step␣
↪{i+1}", alpha=0.7)
        plt.scatter(x, f_x, color='red', s=60, zorder=5)

        x = x - f_x / f_prime(x)

    plt.axhline(0, color='black', lw=1)
    plt.axvline(0, color='black', lw=1)
    plt.xlabel("x")
    plt.ylabel("f(x)")
    plt.title("Newton's Method for Finding a Root")
    plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
    plt.grid(True)
```

Question assigned to the following page: 8

```
    plt.show()

Newton_Method_Single_Plot(8)
```