Name:	Suhas Kamath	**Assignment No:**	Assignment 1
SR No:	06-18-01-10-51-25-1-25945	**Course Code:**	DS288/UMC202
Email ID:	suhaskamath@iisc.ac.in	**Course Name:**	Numerical Methods
Date:	August 23, 2025	**Term:**	AUG 2025

Solution 1

Part (a)

First, we shall set up an equation f() such that f(h) = 0 is a solution to the problem. For this, we shall need to solve the equation. For this, we shall be using the following known formulae:

$$W_{obj} = \rho_{obj} \times V_{sphere} \times g \tag{1}$$

$$W_{disp} = \rho_{water} \times V_{disp} \times g \tag{2}$$

$$V_{disp} = \frac{\pi h^2}{3}(3R - h) \tag{3}$$

$$V_{sphere} = \frac{4\pi r^3}{3} \tag{4}$$

$$\begin{aligned}
W_{obj} &= W_{disp} && \text{Equilibrium Condition} \\
\rho_{obj} \times V_{sphere} \times g &= \rho_{water} \times V_{disp} \times g && \text{Using formulae (1) \& (2)} \\
\rho_{obj} \times V_{sphere} &= \rho_{water} \times V_{disp} && \text{Cancel g on both sides} \\
\rho_{obj} \times \frac{4\pi r^3}{3} &= \rho_{water} \times \frac{\pi h^2}{3}(3r - h) && \text{Substitute equations (3) \& (4)} \\
\rho_{obj} \times 4r^3 &= \rho_{water} \times h^2(3r - h) && \text{Cancel } \frac{\pi}{3} \text{ on both sides} \\
600 \times 4 \times (0.5)^3 &= 1025 \times h^2 \times (3 \cdot 0.5 - h) && \text{Substitute values} \\
\frac{300}{1025} &= h^2 \times (1.5 - h) && \text{Simplify} \\
h^2(1.5 - h) - \frac{12}{41} &= 0 && \text{Equation}
\end{aligned}$$

$$f(h) = h^2(1.5 - h) - \frac{12}{41} = 0 \tag{5}$$

Before we implement the bisection method, we need to find a range [a, b] in which the root should lie. This range should also fulfil the property defined by the Intermediate Value Theorem, i.e. $f(a) \cdot f(b) < 0$. This proves the existence of a root in range. The range [0,1] fulfils this condition.

$f(0) = -\frac{12}{41} \approx -0.2927$ and $f(1) = 0.5 - \frac{12}{41} \approx 0.2073$

$f(0) \cdot f(1) \approx -0.06068 < 0$. Hence, we can use this particular interval.

After the execution of the Python code (attached), the iteration-wise output is as follows:

Iteration	Value of h
0	0.50000
1	0.75000
2	0.62500
3	0.56250
4	0.53125
5	0.54688
6	0.55469
7	0.55860
8	0.55664
9	0.55762
10	0.55713
11	0.55737
12	0.55725
13	0.55719

A total of 13 iterations were required to approximate the root of f(h) to an absolute error of 10^{-4}.

Part (b)

Since there is the guarantee of a solution, as demonstrated by the Intermediate Value Theorem or Bolzano's theorem, the Bisection is guaranteed to converge to any one of the intermediate roots within the particular range.

This is because in every iteration, we halve the range of the solution. Hence, the length of the range in the n^{th} iteration will be $\frac{b-a}{2^n}$. As $n \Rightarrow \infty$, $\frac{b-a}{2^n} \Rightarrow 0$, and $h \Rightarrow \alpha$, where α is the "real" (analytical) solution to the problem.

Name:	Suhas Kamath	**Assignment No:**	Assignment 1
SR No:	06-18-01-10-51-25-1-25945	**Course Code:**	DS288/UMC202
Email ID:	suhaskamath@iisc.ac.in	**Course Name:**	Numerical Methods
Date:	August 23, 2025	**Term:**	AUG 2025

Solution 2

Part (a)

Using Newton's Method

The iterative formula for using Newton's formula is as follows:

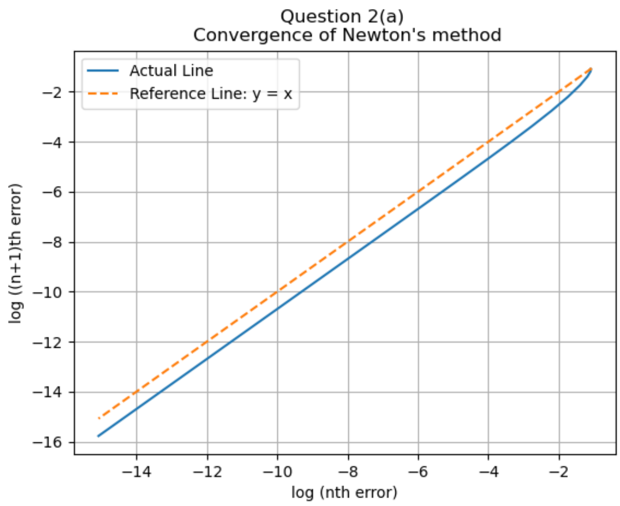
$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

For this, we shall require the first derivative of f(x), i.e. f'(x). The calculation for the derivative is as follows:

$$\begin{aligned}
f'(x) &= \frac{d}{dx}(f(x)) \\
f'(x) &= \frac{d}{dx}(x^2 - \frac{cosx}{e^x})^2 && \text{Substitute f(x)} \\
f'(x) &= 2(x^2 - \frac{\cos x}{e^x})(\frac{d}{dx}(x^2) - \frac{d}{dx}(\frac{cosx}{e^x}))) && \text{Differentiate using chain rule} \\
f'(x) &= 2(x^2 - \frac{\cos x}{e^x})(2x - \frac{d}{dx}(\frac{cosx}{e^x})) \\
f'(x) &= 2(x^2 - \frac{\cos x}{e^x})(2x - (\frac{e^x(-sinx) - cosx(e^x)}{(e^x)^2})) \\
f'(x) &= 2(x^2 - \frac{\cos x}{e^x})(2x + \frac{e^x(sinx) + e^x(cosx)}{e^{2x}}) \\
f'(x) &= 2(x^2 - \frac{\cos x}{e^x})(2x + \frac{e^x(sinx + cosx)}{e^{2x}}) && \text{Factorize } e^x \\
f'(x) &= 2(x^2 - \frac{\cos x}{e^x})(2x + \frac{sinx + cosx}{e^x}) && \text{Cancel } e^x \text{ on both numerator \& denominator}
\end{aligned}$$

After following the iterative formulae with an initial value $x_0 = 5$, the Python code took a total of 29 iterations to solve the equation. The root obtained was 0.6465958.

The convergence graph for this method is as follows:



As can be seen from the graph, the method has a convergence rate of $O(n^1)$.

Using Newton's Modified Method

The overhead of using Newton's modified method is that the double derivative of f(x), i.e f''(x), shall be required. We can just differentiate f'(x) to obtain f''(x). The calculation of f''(x) is as follows:

$$f'(x) = 2\left(x^2 - \frac{\cos x}{e^x}\right)\left(2x + \frac{\sin x + \cos x}{e^x}\right)$$

$$f''(x) = 2 \times \frac{d}{dx}\left[\left(x^2 - \frac{\cos x}{e^x}\right)\left(2x + \frac{\sin x + \cos x}{e^x}\right)\right]$$

$$f''(x) = 2\left\{\left(x^2 - \frac{\cos x}{e^x}\right)\left[\frac{d}{dx}\left(2x + \frac{\sin x + \cos x}{e^x}\right)\right] + \left(2x + \frac{\sin x + \cos x}{e^x}\right)\left[\frac{d}{dx}\left(x^2 - \frac{\cos x}{e^x}\right)\right]\right\}$$

$$f''(x) = 2\left\{\left(x^2 - \frac{\cos x}{e^x}\right)\left(2 + \frac{(e^x)(\cos x - \sin x) - (\sin x + \cos x)(e^x)}{(e^x)^2}\right)\right.$$
$$\left. + \left(2x + \frac{\sin x + \cos x}{e^x}\right)\left(2x - \frac{(e^x)(-\sin x) - (\cos x)(e^x)}{(e^x)^2}\right)\right\}$$

$$f''(x) = 2\left\{\left(x^2 - \frac{\cos x}{e^x}\right)\left(2 + \frac{(e^x)(\cos x - \sin x - \sin x - \cos x)}{(e^x)^2}\right)\right.$$
$$\left. + \left(2x + \frac{\sin x + \cos x}{e^x}\right)\left(2x - \frac{(e^x)(-\sin x - \cos x)}{(e^x)^2}\right)\right\}$$

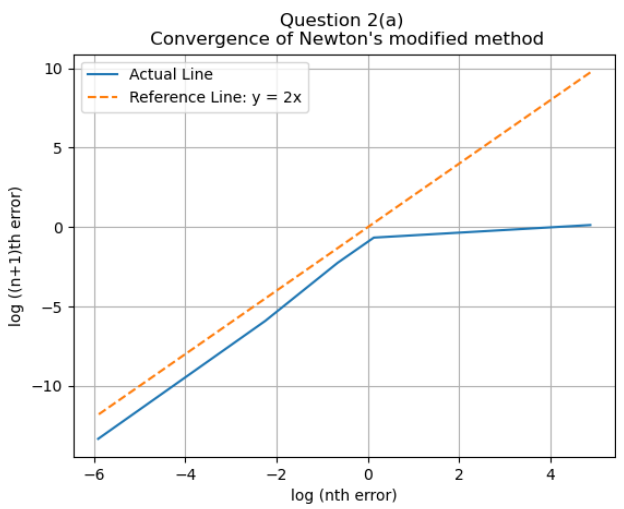
$$f''(x) = 2\left\{\left(x^2 - \frac{\cos x}{e^x}\right)\left(2 - \frac{2\sin x}{e^x}\right) + \left(2x + \frac{\sin x + \cos x}{e^x}\right)\left(2x + \frac{\sin x + \cos x}{e^x}\right)\right\}$$

$$f''(x) = 2\left\{\left(x^2 - \frac{\cos x}{e^x}\right)\left(2 - \frac{2\sin x}{e^x}\right) + \left(2x + \frac{\sin x + \cos x}{e^x}\right)^2\right\}$$

The iterative formula for Newton's modified method is as follows:

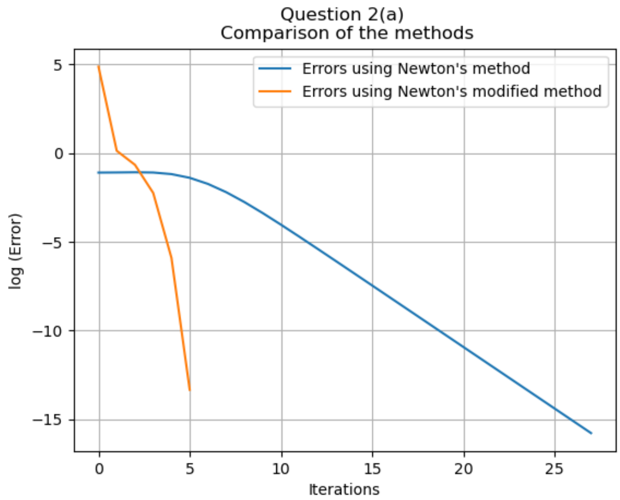
$$x_{n+1} = x_n - \frac{f(x_n)\, f'(x_n)}{\left[f'(x_n)\right]^2 - f(x_n)\, f''(x_n)}$$

After following the iterative formulae with the initial value $x_0 = 5$, the Python code took a total of 7 iterations to solve the equation. The root obtained was 0.6465958.
The convergence plot for this method is as follows:



As can be seen from the graph, the method has a convergence rate of $O(n^2)$.

For this question, the comparison between Newton's method and Newton's Modified method is as follows:



From this graph, we can see that Newton's modified method converged much faster than Newton's method. This is due to the multiplicity of the root, which is discussed later in the analysis.

Part(b)

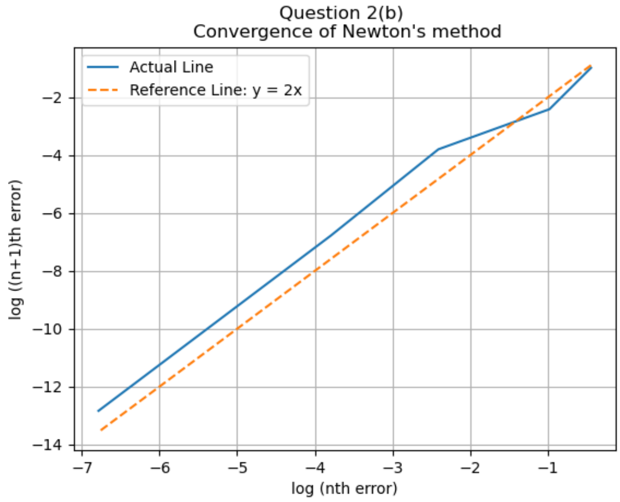
Using Newton's Method

As before, we shall take the first derivative of f(x), i.e. f'(x). The calculation for the derivative is as follows:

$$f'(x) = \frac{d}{dx}(f(x))$$
$$f'(x) = \frac{d}{dx}(\arctan(2x+3)+1)$$
$$f'(x) = \frac{d}{dx}(\arctan(2x+3)) + \frac{d}{dx}(1)$$
$$f'(x) = \frac{d}{dx}(\arctan(2x+3))$$
$$f'(x) = \frac{1}{(2x+3)^2+1} \cdot \frac{d}{dx}(2x+3)$$
$$f'(x) = \frac{2}{(2x+3)^2+1}$$

After following the iterative formulae with the initial value $x_0 = -1$, the Python code took a total of 7 iterations to solve the equation. The root obtained was -2.2787039.

The convergence graph for this method is as follows:



As can be seen from the graph, the method has a convergence rate of $O(n^2)$.

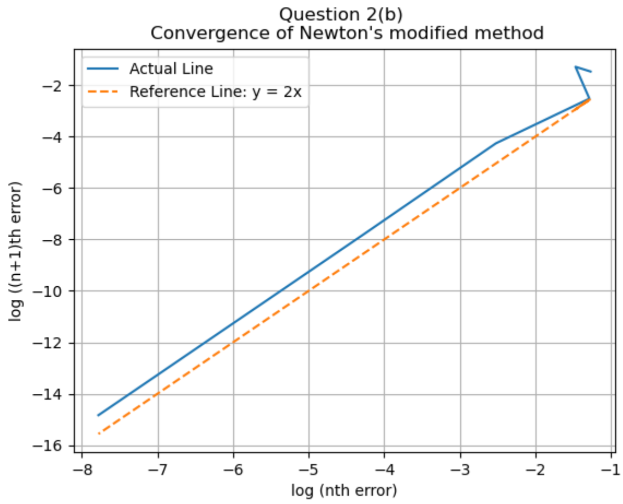
Using Newton's Modified Method

Just like for part (a), we shall require the double derivative of f(x), i.e f"(x) shall be required. The calculation of f"(x) is as follows:

$$f'(x) = \frac{2}{(2x+3)^2+1} = 2\left[(2x+3)^2+1\right]^{-1}$$
$$f''(x) = 2(-1)\left(\frac{d}{dx}\left((2x+3)^2+1\right)\right)\left[(2x+3)^2+1\right]^{-2}$$
$$f''(x) = -\frac{2}{\left[(2x+3)^2+1\right]^2} \times \left[2\left(\frac{d}{dx}(2x+3)\right)(2x+3)\right]$$
$$f''(x) = -\frac{2}{\left[(2x+3)^2+1\right]^2} \times \left[2(2)(2x+3)\right]$$
$$f''(x) = -\frac{8(2x+3)}{\left[(2x+3)^2+1\right]^2}$$

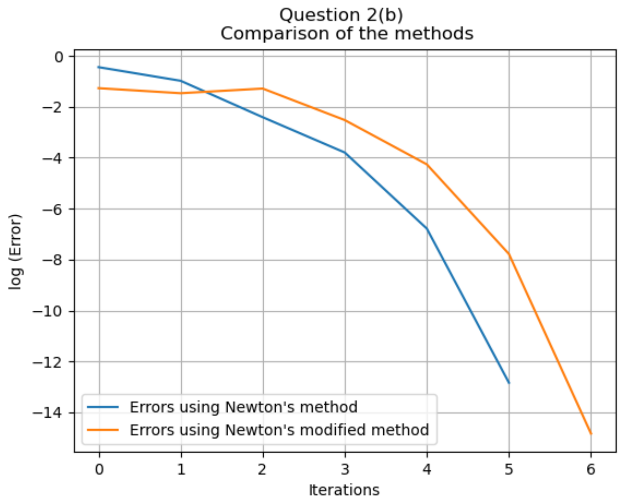
After following the iterative formulae with initial value $x_0 = -1$, the Python code took a total of 8 iterations to solve the equation. The root obtained was -2.2787039.

The convergence graph for this method is as follows:



As can be seen from the graph, the method has a convergence rate of $O(n^2)$.

For this question, the comparison between Newton's method and Newton's Modified method is as follows:



In this graph, it can be seen that both the methods converge at a similar rate. This can also be attributed to the multiplicity of the root, as discussed in the below analysis.

Analysis

The overall analysis for using the two methods is as follows:

Question Part	Number of Iterations using Newton's method	Number of Iterations using Newton's modified method
(a)	29	7
(b)	7	8

Hence, we can see that the modified Newton's method need not always be faster than Newton's method. It is faster only in case of repeated roots, i.e. roots which have a multiplicity of ≥ 2.

To verify this particular theory, let us check the multiplicity of both functions concerning their solutions. The theorem that will be used to check this is as follows:

The function $f \in C^m[a, b]$ has a root of multiplicity m at α if and only if:

$$0 = f(\alpha) = f'(\alpha) = f''(\alpha) = \cdots = f^{(m-1)}(\alpha), \text{ but } f^{(m)}(\alpha) \neq 0$$

Part (a)

The approximate solution obtained (to a relative error of 10^{-7}) for this equation was 0.6465958. Let us check using the aforementioned theorem:

$$f(0.6465958) = 3.54 \times 10^{-16} \approx 0$$
$$f'(0.6465958) = 7.63 \times 10^{-8} \approx 0$$
$$f''(0.6465958) = 8.22 \neq 0$$

From this, we can say that the multiplicity of the root of this equation is 2. This is the reason because of why Newton's modified method provides such an improvement over Newton's method.

Part (b)

The approximate solution obtained (to a relative error of 10^{-7}) for this equation was -2.2787039. Let us check using the aforementioned theorem:

$$f(-2.2787039) = -2.20 \times 10^{-8} \approx 0$$
$$f'(-2.2787039) = 0.583 \neq\approx 0$$

From this, we can say that the multiplicity of the root of this equation is 1. This is the reason because of why Newton's modified method converges more slowly than Newton's method.

Name:	Suhas Kamath	**Assignment No:**	Assignment 1
SR No:	06-18-01-10-51-25-1-25945	**Course Code:**	DS288/UMC202
Email ID:	suhaskamath@iisc.ac.in	**Course Name:**	Numerical Methods
Date:	August 23, 2025	**Term:**	AUG 2025

Solution 3

To obtain the equation in the form p = g(p), we need to solve the equation first:

$$
\begin{gathered}
Q_d = Q_s \\
10000 - 20p = 150\sqrt{p-25} \\
20p = 10000 - 150\sqrt{p-25} \\
p = \frac{10000 - 150\sqrt{p-25}}{20} \\
p = 500 - \frac{15\sqrt{p-25}}{2}
\end{gathered}
$$

Hence, we can say that $g(p) = 500 - \frac{15\sqrt{p-25}}{2}$.
Before we can continue, we need to verify that the necessary conditions are fulfilled to apply this method. The necessary conditions are:

1. If $g \in C[a,b]$, and $a \leq g(p) \leq b$
2. $g'(p)$ exists for all $p \in [a,b]$
3. There exists a positive number k such that $|g'(p)| \leq k < 1$ for all $p \in [a,b]$.

First, we need to select a range $[a,b]$ to test this. This range shall bracket the fixed point. Let us choose the range as $[150, 420]$.

Since we know that g(x) is a decreasing function, we can only test the boundaries of the range.

$$
\begin{gathered}
g(150) = 416.1474 \\
g(420) = 350.9404
\end{gathered}
$$

Because both boundary values satisfy the first condition, and the function is decreasing, we can state that the first condition is fulfilled.

To test the second condition, we need to find the first derivative of $g(p)$, i.e. $g'(p)$.

$$
\begin{gathered}
g'(p) = \frac{d}{dp}\left(500 - \frac{15\sqrt{p-25}}{2}\right) \\
g'(p) = \frac{d}{dp}(500) - \frac{15}{2}\left(\frac{d}{dp}(\sqrt{p-25})\right) \\
g'(p) = 0 - \frac{15}{2}(p-25)^{-\frac{1}{2}}(1)\left(\frac{1}{2}\right) \\
g'(p) = -\frac{15}{4\sqrt{p-25}}
\end{gathered}
$$

Looking at the function, we can say that $g'(p)$ exists for all $x > 25$. Because we have selected $[150, 420]$ as our range, we can be sure that the second condition is also fulfilled.

Lastly, let us check the condition for convergence:

$$
\begin{aligned}
|g'(p)| &< 1 \\
\left| -\frac{15}{4\sqrt{p-25}} \right| &< 1 \\
\frac{15}{4\sqrt{p-25}} &< 1 \\
\sqrt{p-25} &> \frac{15}{4} \\
p - 25 &> \left(\frac{15}{4}\right)^2 \\
p - 25 &> 14.0625 \\
p &> 39.0625
\end{aligned}
$$

Hence, because all values in $[150, 420]$ fulfill the above inequality, we can state that the third condition is also fulfilled.

After iterating using the fixed point method, we obtain the result to an absolute accuracy of 10^{-4}. The final result is \$362.2643. A total of 11 iterations were required to obtain this result. The individual iterations are as follows:

Iteration	Value of Iteration
0	150.0000
1	416.1474
2	351.6691
3	364.4451
4	361.8197
5	362.3551
6	362.2458
7	362.2681
8	362.2635
9	362.2645
10	362.2643
11	362.2643

Solution 4

We need to find the location in the beam where the deflection is exactly 0.06 meters. The equation should be as follows:

$$y(x) = \frac{300x - 4x^3}{10^4} = 0.06$$
$$300x - 4x^3 = 600$$
$$4x^3 - 300x + 600 = 0$$

Hence, let us create a new function, i.e. $g(x) = 4x^3 - 300x + 600$. Now we need to find the roots of this function. First, let us ensure that there is a root in the range [1, 5] using the Intermediate Value Theorem.

$$g(a) \cdot g(b) < 0$$
$$g(1) \cdot g(5) < 0$$
$$304 \cdot -400 < 0$$
$$-121600 < 0$$

Since this inequality holds, we know for sure that there is at least one root in the range [1,5]. The Python required a total of 8 iterations to approximate the answer to a tolerance of 10^{-6}. The results of the iteration are as follows:

Iteration	Value of Iteration
1	2.727272
2	2.190574
3	2.134145
4	2.128092
5	2.128635
6	2.128596
7	2.128593
8	2.128593

Hence, the final answer is 2.128593.

References

[1] Faires, J. Douglas, and Richard L. Burden. Numerical methods, 4th. Cengage Learning, 2012.

[2] Lecture Notes