

**NMiNE**

**HA06**

**By**

**Faisal Ahmed Moshiur**

```

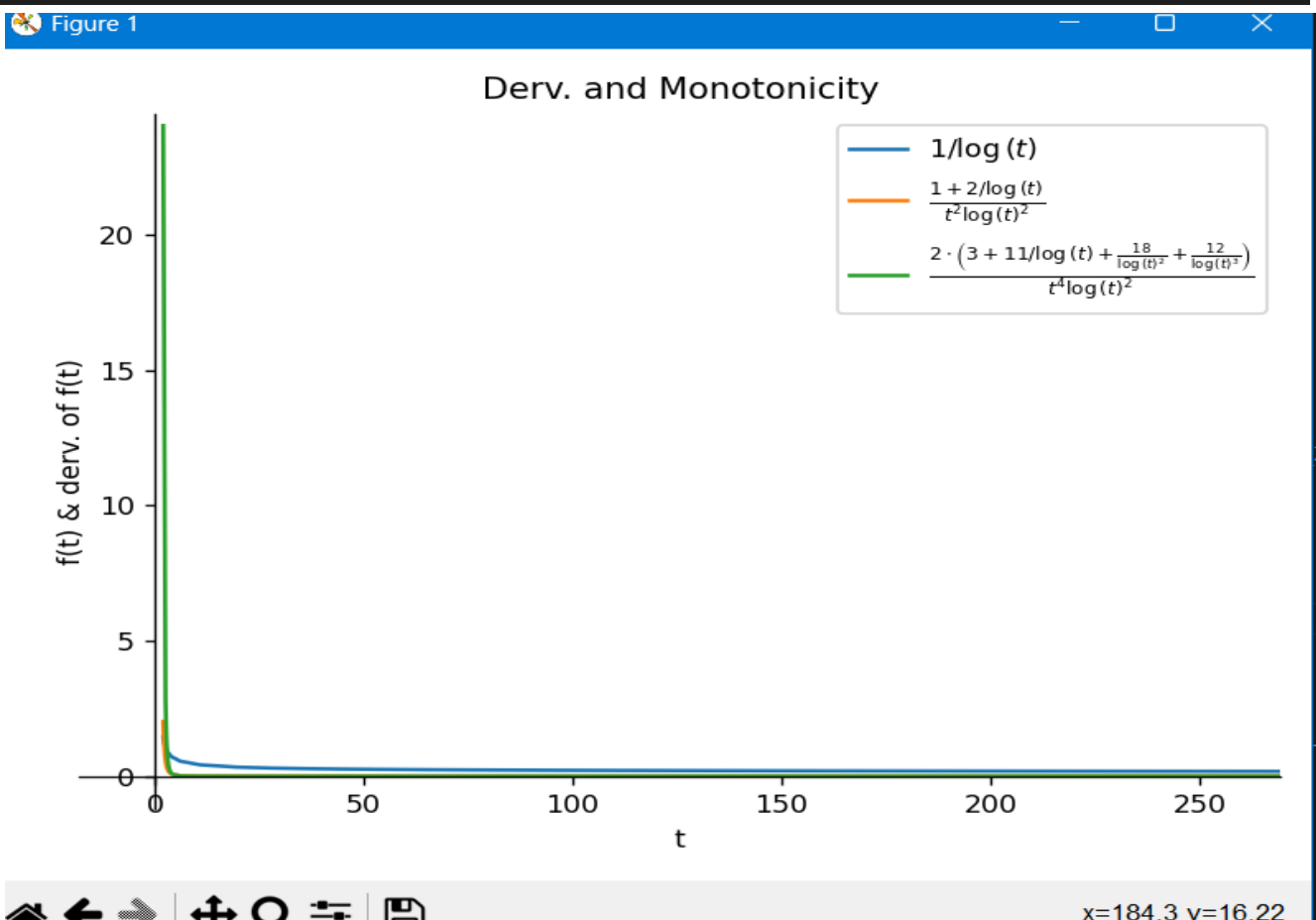
1 from sympy import *
2 from sympy import log as ln
3 from sympy import symbols
4 from sympy import diff
5 from sympy import Function
6 from sympy import plot
7
8 t = Symbol('t')
9 f = 1 / ln(t)
10 f_2nd_derv = diff(f, t, 2)
11 f_4th_derv = diff(f, t, 4)
12
13 print(f"Function, f(t): {f}\nSecond Derv, f''(t): {f_2nd_derv}\nFourth Derv, f''''(t): {f_4th_derv}")
14
15 f_2 = lambdify(t, f_2nd_derv)
16 f_4 = lambdify(t, f_4th_derv)
17
18 print(f"Second Derv, f''(2): {f_2(2)}\nFourth Derv, f''''(2): {f_4(2)}")
19
20 Function('f_2nd_derv')
21 Function('f_4th_derv')
22 plt = plot(f, f_2nd_derv, f_4th_derv, (t, 2, 500), title="Derv. and Monotonicity",
23           legend= True, xlabel='t', ylabel='f(t) & derv. of f(t)')
24 plt.show

```

```

Function, f(t): 1/log(t)
Second Derv, f''(t): (1 + 2/log(t))/(t**2*log(t)**2)
Fourth Derv, f''''(t): 2*(3 + 11/log(t) + 18/log(t)**2 + 12/log(t)**3)/(t**4*log(t)**2)
Second Derv, f''(2): 2.0217325988298547
Fourth Derv, f''''(2): 24.031396650133274

```



Q2\_Ans:

2(a):

We know that in Riemann sum technique, we select the numbers  $\xi_i$  that minimize  $f(x)$  on every sub-interval  $[x_{i-1}, x_i]$

$$\sum_{i=1}^n \min_{[x_{i-1}, x_i]} f(\xi_i) (x_i - x_{i-1}) \leq \int_a^b f(x) dx \leq \sum_{i=1}^n \max_{[x_{i-1}, x_i]} f(\xi_i) (x_i - x_{i-1})$$

Also, we define Reimann Left (L) and Right (R) as:

$$L(f, x_i) = \sum_{i=1}^n f(x_{i-1}) (x_i - x_{i-1})$$

$$R(f, x_i) = \sum_{i=1}^n f(x_i) (x_i - x_{i-1})$$

Hence, for a monotonically decreasing function, we minimalize the  $R(f, x_i)$  and take the maximum of  $L(f, x_i)$  to minimize the  $f(x)$  on every sub-interval  $[x_{i-1}, x_i]$ .

Therefore,

$$\min\{L(f, x_i), R(f, x_i)\} \leq \int_a^b f(x) dx \leq \max\{L(f, x_i), R(f, x_i)\}$$

Reduces to

$$R(f, x_i) \leq \int_a^b f(x) dx \leq L(f, x_i)$$

2(b):

$$L(f, x_i) = \sum_{i=1}^n f(x_{i-1}) (x_i - x_{i-1})$$

$$R(f, x_i) = \sum_{i=1}^n f(x_i) (x_i - x_{i-1})$$

So,

$$\begin{aligned} L - R &= \sum_{i=1}^n [f(x_{i-1}) (x_i - x_{i-1}) - f(x_i) (x_i - x_{i-1})] \\ &= \sum_{i=1}^n (x_i - x_{i-1}) [f(x_{i-1}) - f(x_i)] \end{aligned}$$

We know,

$$h = \max_{1 \leq i \leq n} (x_i - x_{i-1}) = \frac{b - a}{n}$$

and

$$x_i = b \text{ \& \; } x_{i-1} = a$$

Therefore,

$$L - R = \frac{b - a}{n} [f(a) - f(b)]$$

```

1 import numpy as np
2 from sympy import *
3 from sympy import log as ln
4 import sympy
5
6
7
8 def Reimann_Int(a, b, N):
9     h = (b - a) / (N - 1)
10    x = np.linspace(a, b, N)
11    f = 1 / np.log(x)
12
13    I_riemannL = h * sum(f[:N-1])
14
15    #I_riemannR = h * sum(f[1::])
16
17    return print(f"Reimann left Int:{I_riemannL}")
18
19 t = Symbol('t')
20 f = 1 / ln(t)
21
22 from numpy import integrate
23
24 Li_200_ = integrate(f, (t, 2, 200))
25
26 print(f"Li(200) using standard function:{Li_200_}")
27
28 for i in range (0, 2000):
29     i += 1
30     if (Li_200_ - Reimann_Int(2, 200, i)) == 1e4:
31         print(f"No. of steps(value of N) needed to produce 3 decimal place accuracy:
32 {i}")
33     else:
34         continue
35

```

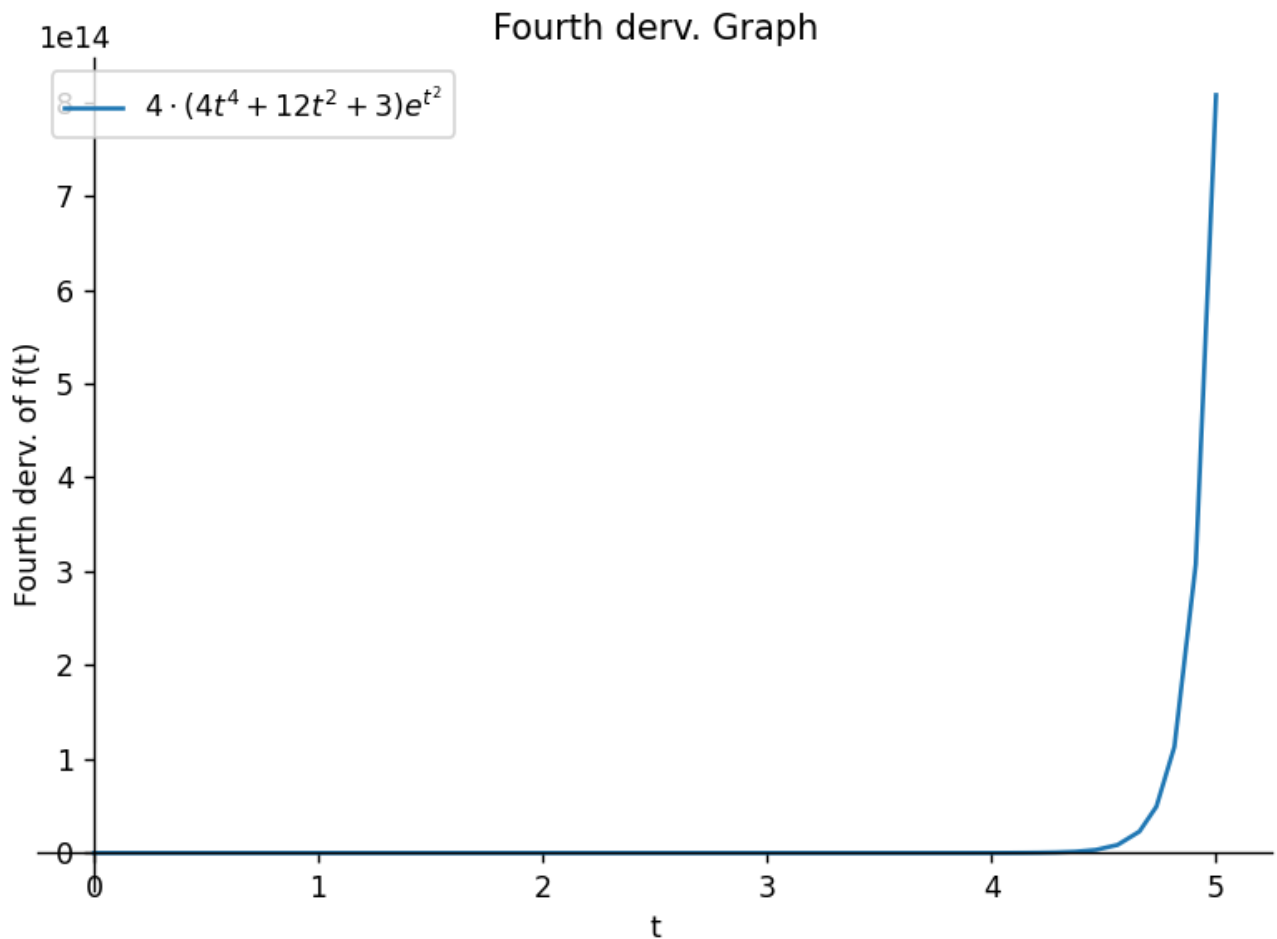
```
1 import numpy as np
2
3 def Reimann_Int_mid(a, b, N):
4     h = (b - a) / (N - 1)
5     x = np.linspace(a, b, N)
6     f = 1 / np.log(x)
7
8     I_mid = h * sum(1 / np.log((x[:N-1] \
9         + x[1:])/2))
10
11     return print(f"Reimann left Int:{I_mid}")
12
13
14
```

```
1 import numpy as np
2
3
4 def simp_intg(a, b, N):
5     h = (b - a) / (N - 1)
6     x = np.linspace(a, b, N)
7     f = np.exp^(-x)
8     I_simp = (h/3) * (f[0] + 2*sum(f[:N-2:2]) \
9         + 4*sum(f[1:N - 1:2]) + f[N-1])
10
11     return print(f"Simpson integration:{I_simp}")
12
13
14
15
```

```

1 from sympy import *
2 from sympy import exp
3 from sympy import symbols
4 from sympy import diff
5 from sympy import Function
6 from sympy import plot
7
8 t = Symbol('t')
9 f = 1 / exp(-1 * t**2)
10 f_4th_derv = diff(f, t, 4)
11
12 print(f"Fourth Derv, f''''(t): {f_4th_derv}")
13
14 Function('f_4th_derv')
15 plt = plot(f_4th_derv, (t, 0, 5), title="Fourth derv. Graph", legend= True,
16           xlabel='t', ylabel='Fourth derv. of f(t)')
17 plt.show
18

```





Q8\_Ans:

$$w_1 f(x_1) + w_2 f(x_2) = \int_{-1}^1 f(x) dx$$

To get the nodes and weight according to the conditions imposed:

$$w_1 + w_2 = \int_{-1}^1 f(1) dx = 2$$

$$w_1 x_1 + w_2 x_2 = \int_{-1}^1 x dx = 0$$

$$w_1 x_1^2 + w_2 x_2^2 = \int_{-1}^1 x^2 dx = \frac{2}{3}$$

$$w_1 x_1^3 + w_2 x_2^3 = \int_{-1}^1 x^3 dx = 0$$

Therefore, solving the above equation we can easily get the following values:

$$w_1 = w_2 = 1 \text{ \& } x_1 = \frac{-1}{\sqrt{3}}, x_2 = \frac{1}{\sqrt{3}}$$

We can say that this yield:

$$\int_{-1}^1 f(x) dx = f\left(\frac{-1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right)$$

Therefore we can infer that it has degree of precision equal to 3 since it integrates exactly all polynomials of degree  $\leq 3$ .