

**LAB EXPERIMENT # 10: Study of numerical integration (i) Trapezoidal rule (ii) Simpson's rule**

<b>Name &amp; Roll No.:</b> Ashraf Al- Khalique, 1801171	<b>Grade (20)</b>
<b>Registration Number:</b> 351	
<b>Lab Section:</b> C-2	

**10.1 Objectives**

- To understand the numerical integration method and its MATLAB implementation
- To understand Trapezoidal rule and its MATLAB implementation
- To understand Simpson's rule and its MATLAB implementation
- To analyze of results using different values

**10.2 Theory**

Conventional integrations are difficult to solve, especially for large amounts of data such as data engineering. Usually, to find the definite integral, we use the fundamental theorem of calculus, where we have to apply the antiderivative techniques of integration. However, sometimes, it isn't easy to find the antiderivative of an integral, like in Scientific experiments, where the function has to be determined from the observed readings. Therefore, numerical methods are used to approximate the integral in such conditions.

The trapezoidal rule is applied to solve the definite integral of the form  $\int_a^b f(x) dx$ , by approximating the region under the graph of the function  $f(x)$  as a trapezoid and calculating its area. Under the trapezoidal rule, we evaluate the area under a curve is by dividing the total area into little trapezoids rather than rectangles.

$$\int_{x_0}^{x_n} y dx = s = h \left( \frac{f_1}{2} + f_2 + f_3 + f_4 + f_5 + \dots + \frac{f_{n+1}}{2} \right)$$

Simpson's rule is one of the numerical methods which is used to evaluate the definite integral. Simpson's rule methods are more accurate than the other numerical approximations.

$$\int_{x_0}^{x_n} y dx = \frac{h}{3} (y_1 + 4(y_2 + y_4 + y_6) + 2(y_3 + y_5 + y_7) + y_n)$$

**10.3 Apparatus**

- MATLAB

**10.4 Algorithm**

- **Trapezoidal rule**

**Step: 1** Start

**Step: 2** Define function

**Step: 3** Read the lower integration limit, upper integration limit, and number of sub intervals.

**Step: 4** Calculate the step size = (upper limit - lower limit)/number of sub interval

**Step: 5** Now, set the integration value =  $f(\text{lower limit}) + f(\text{upper limit})$  &  $i = 1$

**Step: 6** Calculate the integration value = integration Value +  $2 * f(k)$

Again, integration value = integration value \* step size/2

**Step: 7** Display integration value as required answer

**Step: 8** Stop

- **Simpson's 1/3 rule**

**Step.1** Start

**Step.2** Define function  $f(x)$

**Step.3** Read lower limit of integration, upper limit of integration and number of sub interval

**Step.4** Calculate the step size = (upper limit - lower limit)/number of sub interval

**Step.5** Calculate the integration value =  $f(\text{lower limit}) + f(\text{upper limit})$

**Step.6** Set:  $i = 1$

**Step.7** If  $i > \text{number of sub interval}$  then calculate  $k = \text{lower limit} + i * h$

**Step.8** Increase  $i$  by 1 and go to step 7

**Step.9** Calculate the integration value = integration value \* step size/3

**Step.10** Display Integration value as required answer

**Step.11** Stop

## 10.5 Pseudocode

- **Trapezoidal rule**

```

start
input x,y
n = length(x);
h = x(2)-x(1);
for i=2 to n do
    sum = sum +f(i);
end for
integral = h*sum;
disp(integral)
stop
  
```

- **Simpson's 1/3 rule**

```

start
input x,y
  
```

```

n=length(x)
Define
    sum1=0;
    sum2=0;
h=x(2)-x(1)
sum=(y(1)+y(n));

for i=2 to 2 with interval n-1 do
    sum1=sum1+4*y(i);
end for

for j=3 to 2 with interval n-1 do
    sum2=sum2+2*y(j);
end for

integral=((h/3)*(sum+sum1+sum2))
disp(integral)
stop

```

## 10.6 MATLAB Code

- Trapezoidal rule

```

clc;
clear all;
x=[0;0.125;0.250;0.375;0.5;0.625;0.750;0.875;1.0];
y=[1.0;0.8889;0.8;0.7273;0.667;0.6154;0.5714;0.5333;0.5];
table(x,y)
n=length(x)
h=x(2)-x(1)
sum=(y(1)+y(n))/2;

for i=2:n-1;
    sum=sum+y(i);
end
integral = (h*sum)
disp(integral)

```

- Simpson's 1/3 rule

```

clc;
clear all;
x=[0;0.125;0.250;0.375;0.5;0.625;0.750;0.875;1.0];
y=[1.0;0.8889;0.8;0.7273;0.667;0.6154;0.5714;0.5333;0.5];

```

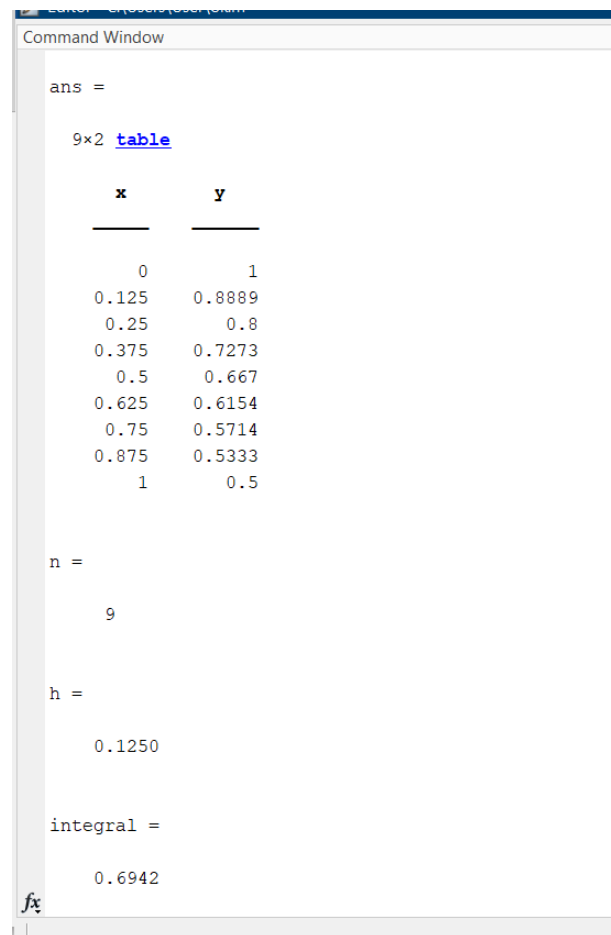
```

table(x,y)
n=length(x)
sum1=0;
sum2=0;
h=x(2)-x(1)
sum=(y(1)+y(n));
for i=2:2:n-1;
    sum1=sum1+4*y(i);
end
for j=3:2:n-1;
    sum2=sum2+2*y(j);
end
integral=((h/3)*(sum+sum1+sum2))
disp(integral)

```

## 10.7 MATLAB Output

- Trapezoidal rule



Command Window

ans =

9x2 [table](#)

x	y
0	1
0.125	0.8889
0.25	0.8
0.375	0.7273
0.5	0.667
0.625	0.6154
0.75	0.5714
0.875	0.5333
1	0.5

n =

9

h =

0.1250

integral =

0.6942

- **Simpson's 1/3 rule**

```

ans =

9x2 table

    x    y
    ---  ---
    0    1
  0.125  0.8889
   0.25  0.8
  0.375  0.7273
   0.5   0.667
  0.625  0.6154
   0.75  0.5714
  0.875  0.5333
    1    0.5

n =

9

h =

0.1250

integral =

0.6932

```

## 10.8 Discussion & Analysis

In this experiment, we used MATLAB to code the trapezoidal rule, Simpson's one-third rule of numerical integration. The algorithm was structured based on the equation developed using theory and pseudocode for mentioned methods. Later, it was coded on MATLAB according to the algorithm and pseudocode and desired output was obtained.

As Simpson's 1/3rd rule is an extension of the trapezoidal rule in which the integrand is approximated by a second-order polynomial, the algorithm for Simpson's 1/3 rule was originally derived from trapezoidal rule. Thus, both methods provided quite close output to each other.

Thus, the experiment was done successfully and MATLAB implementation of the method was carried out accordingly.