

+++++

Lab 007

+++++

+++++

2154638

+++++

Mathematical Skills for Data Scientists

Lab Exercises 7

CSCM70

Lab 7 Solutions

Mathematical Skills for Data Scientists

Lab Exercises 7 – 5 Marks (Due: 12/11/22)

Gibin Powathil

g.g.powathil@swansea.ac.uk

Submitted by: Pallav Shukla

Student ID: 2154638

Question 1

Exercise 1. Write a routine that takes as input a function f and a natural number N , then generates N random numbers x_i between 0 and 1 and computes the average of all function values $f(x_i)$. Compare the result to $\int_0^1 f(x)dx$. (marks 2)

Answer 1:

We make a function named as routine(func,N).



routine.m

Where func will be the passed function with N as the number of intermediate numbers between 0 and 1 . We calculate the average of N function values.

We chose the function as $f(x) = 2x$.

The indefinite integral of which is x^2 . Definite integral of $2x$ would be $1^2 - 0^2$.

We then compare the result with the built-in function : `int` which calculate the definite integral of a function between certain range. Reference: <https://in.mathworks.com/help/symbolic/sym.int.html> . The output of which is equal to our manual routine output.

Ref: From Sir's Lecture -

OUTPUT MATLAB: COMMAND WINDOW

Output of answer 1

TOR PUBLISH FILE VERSIONS VIEW Search (Ctrl+Shift+Space)

monty_hall_v1.m routine.m

```
1 function res = routine(func,N)
2
3     sum = 0;
4     N_random_numbers = 1 .* rand(N,1);
5     for i = 1:N
6         sum = sum + func(N_random_numbers(i));
7     end
8
9     res = sum/N;
```

Function to calculate the average of all passed function value $f(x_i)$

Command Window

```
>> fun2x = @(x) 2*x;
>> routine(fun2x,1000)

ans =

    0.9869

>> routine(fun2x,10000000)

ans =

    1.0000
```

→ Output from manual routine

```
>> syms x;
>> expr = 2*x

expr =

2*x

>> int(expr,[0 1])

ans =

1
```

→ Output by built-in function

>> |

2 usages of "N_random_numbers" found

The screenshot shows the MATLAB Editor with a file named `routine.m` open. The function is defined as follows:

```
function res = routine(func,N)
1
2
3 sum = 0;
4 N_random_numbers = 1 .* rand(N,1);
5 for i = 1:N
6     sum = sum + func(N_random_numbers(i));
7 end
8
9 res = sum/N;
```

A red callout box points to the function definition with the text: "Function to calculate the average of all passed function value f(x)".

The Command Window shows the following execution steps and results:

```
>> fun2x = @(x) 2^x;
>> routine(fun2x,1000)

ans =
    0.9869

>> routine(fun2x,10000000)

ans =
    1.0000
```

Annotations in the Command Window:

- A green arrow points from the text "Output from manual routine" to the value `1.0000`.
- A green arrow points from the text "Output by built-in function" to the value `0.9869`.

The Workspace window shows the following variables:

Name	Value	Size	Class
ans	1	1x1	sym
expr	2^x	1x1	sym
f	@(x)2^x	1x1	function_handle
fun2x	@(x)2^x	1x1	function_handle
i	5	1x1	double
r	[0.0154 0.0430 ...]	5x1	double
x	x	1x1	sym

Question 2

Exercise 2. Write a function that generates points in the unit square (points (x, y) with $0 \leq x \leq 1, 0 \leq y \leq 1$) uniformly at random, and keeps track of which fraction of them are within a distance of 1 from the origin. What does this Monte-Carlo method compute? (marks 2)

Answer 2.1

Developed a routine named `monte_carlo(N)`, where N denotes the number of randomly generated points between 0 and 1 on the x and y axis.



`monte_carlo.m`

The points which are at within a distance of 1 from origin means that, their unit distance from the origin is 1 unit. Distance of point (x, y) from $(0, 0)$ is calculated by : $((x-0)^2 + (y-0)^2)^{1/2} = (x^2 + y^2)^{1/2}$

The screenshot shows the MATLAB Editor with a file named `monte_carlo.m` open. The function is defined as follows:

```
function res = monte_carlo(N)% generate a set of random numbers x
1
2 % (0, pi) and y(0, 1)
3 x = rand(1, N);
4 y = rand(1, N);
5 counter = 1;
6 num_samples = N;
7 % number of samples which will be at unit distance form origin
8 favourable_samples = 0;
9 Y = [];
10
11 % find all points which are at unit distance from origin
12 while counter < num_samples
13     if ((x(counter) ^ 2 + y(counter) ^ 2) ^ 0.5) <= 1
14         favourable_samples = favourable_samples + 1;
15         X(end+1) = x(counter);
16         Y(end+1) = y(counter);
17     end
18     counter = counter + 1;
19 end
20 % calculating fraction of favaourable outcomes from sample space
21 res = favourable_samples / num_samples;
22 %displaying the results
23 disp(['Fraction of points which are unit distance from origin from sample space ' res])
```

The Workspace window shows the following variables:

Name	Value	Size	Class
ans	0.7855	1x1	double
r	[0.1576 0.9706 ...]	1x10	double

We use a while loop to check the distance of each point and count the number of favourable outcomes by the `favourable_samples` variable.

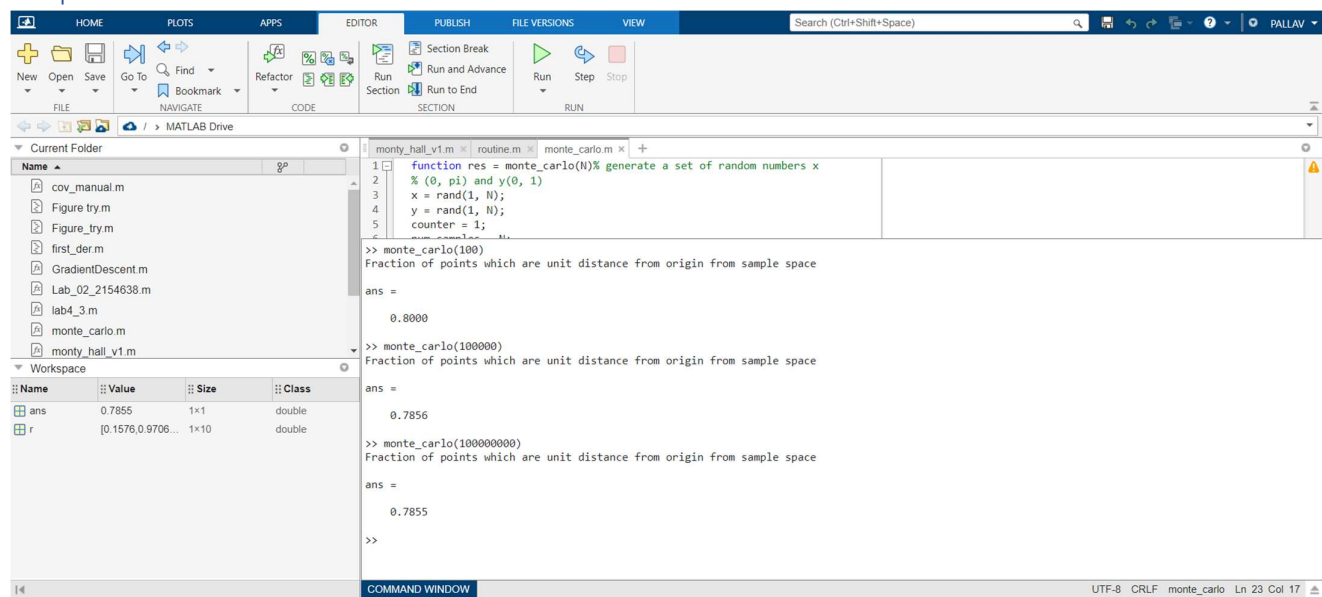
The result, that is the fraction of `favourable_samples` is stored in `res` and calculated as:

$$\frac{\text{No. of favourable outcomes}}{\text{Total number of samples}}$$

Ref: From Sir's Lecture -

OUTPUT MATLAB: COMMAND WINDOW

Output of answer 2



The screenshot shows the MATLAB Command Window with the following content:

```
function res = monte_carlo(N)% generate a set of random numbers x
1
2 % (0, pi) and y(0, 1)
3 x = rand(1, N);
4 y = rand(1, N);
5 counter = 1;
6
>> monte_carlo(100)
Fraction of points which are unit distance from origin from sample space
ans =
    0.8000
>> monte_carlo(100000)
Fraction of points which are unit distance from origin from sample space
ans =
    0.7856
>> monte_carlo(100000000)
Fraction of points which are unit distance from origin from sample space
ans =
    0.7855
>>
```

```
monte_carlo(100)
Fraction of points which are unit distance from origin from sample space
```

```
ans =

    0.8000
```

```
monte_carlo(100000)
Fraction of points which are unit distance from origin from sample space
```

```
ans =

    0.7856
```

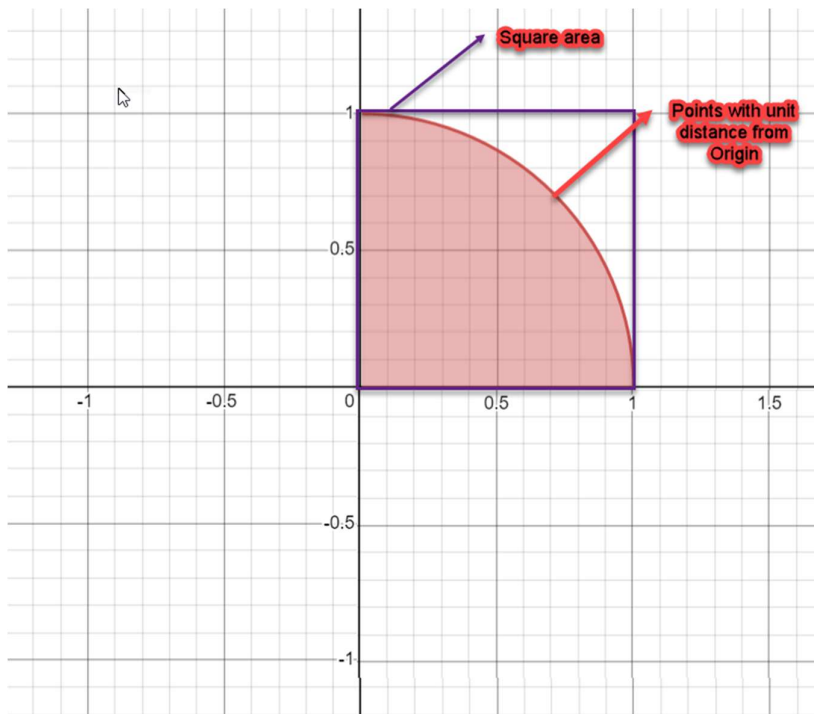
```
monte_carlo(100000000)
Fraction of points which are unit distance from origin from sample space
```

```
ans =

    0.7855
```

Answer 2.2

This Monte-Carlo indicates gives us a probability distribution of points which are at a distance of 1 from the origin in the space $0 \leq x \leq 1$ and $0 \leq y \leq 1$. That is this gives us $\frac{1}{4}$ of a circle of radius = 1 unit .



Reference: <https://www.desmos.com/calculator/ivud1auwvw>

Question 3

Exercise 3. Download the data1 file from the assignment portal. Calculate the mean and standard deviation of the data set.

Assuming the data is normally generated, how many values would exceed 300? How many values would be negative?

Compare this to how many values actually exceed 300 and how many actually are negative and comment on any discrepancies. (marks 1)

Answer 3

Reference: https://www.varsitytutors.com/hotmath/hotmath_help/topics/normal-distribution-of-data

```
%read data from table
t = readtable("data_1.csv", "Delimiter", ',')
array_data1 = table2array(t)

mean(array_data1) = 66.3294

std(array_data1) = 209.8658

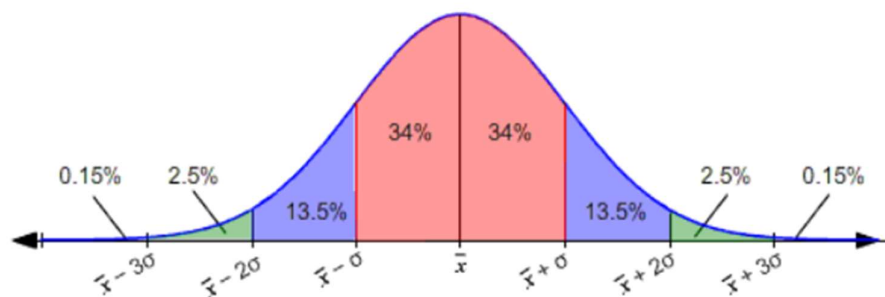
Mean of data: 66.3294

Standard Deviation of data: 209.8658
```

Mean + Std deviation = 276.1952

Mean + 2*std deviation = 486.0610

That is, if \bar{x} is the mean and σ is the standard deviation of the distribution, then 68% of the values fall in the range between $(\bar{x} - \sigma)$ and $(\bar{x} + \sigma)$. In the figure below, this corresponds to the region shaded pink.



Therefore, about $13.5 + 2.5 + 0.15 = 16.15\%$ values will be greater than 300 .

That is $0.1615 \times 2000 = 323$ values would be approximately greater than 300.

Negative values: almost $13.5 + 2.5 + 0.15 = 16.15\%$ values must be negative , i.e., 323 values will surely be negative.

Also as from 34% , approximately $68\% (209.8658 - 66.3294 / 209.8658)$ values , i.e , 23% of 2000 = 465 , therefore $323 + 465 = 788$ values must be negative.

ACTUAL DATA :

Only 73 values are greater than 300

```
>> sum(a>300)
ans =
    73
>> |
```

Negative values:

```
>> sum(a<0)
ans =
    0
>> |
```

😞 , there are no values which are negative.

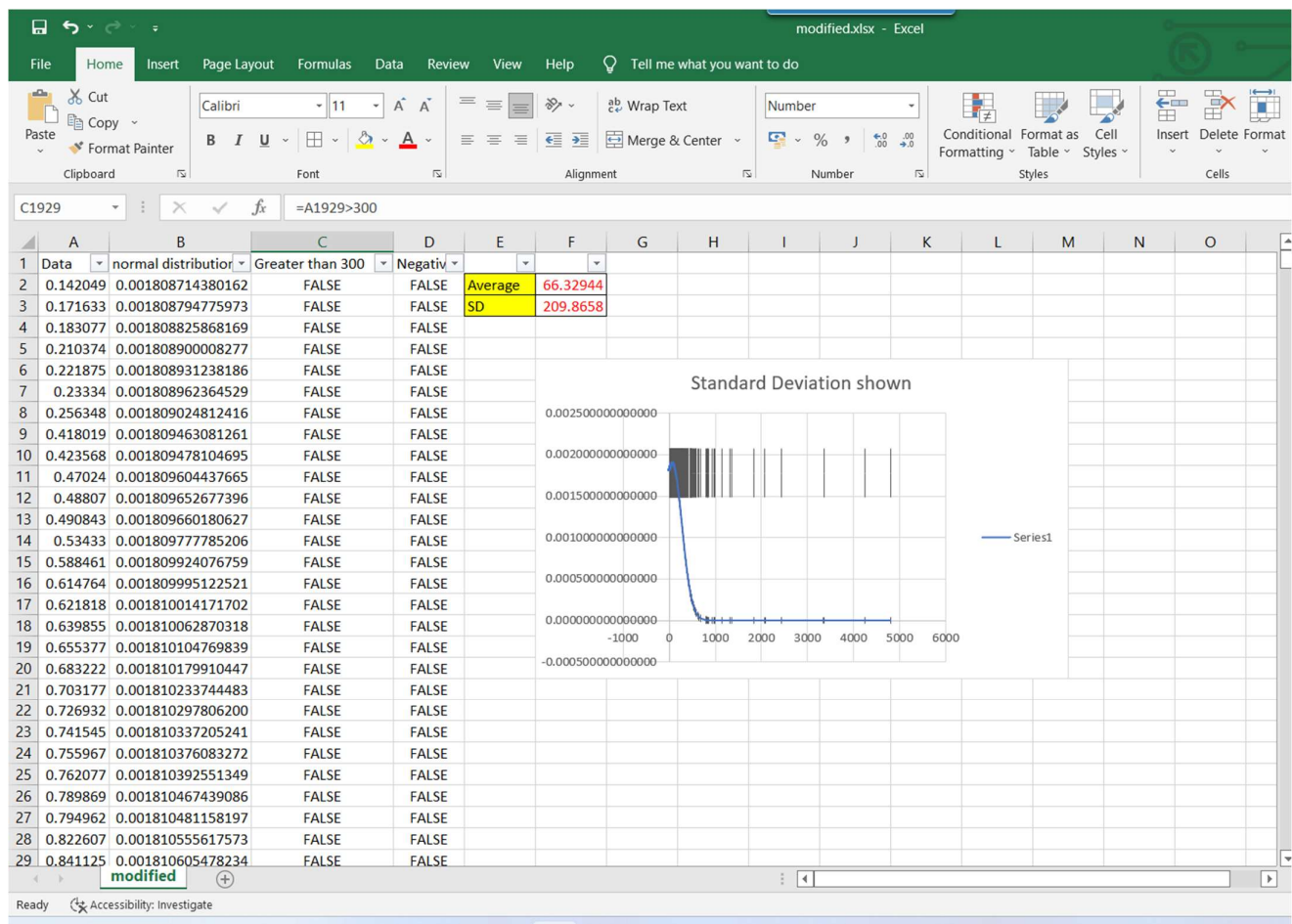
Reason of such discrepancies:

We assumed the data as normally distributed and did our analysis on the same , rather than that we when we normalized the data



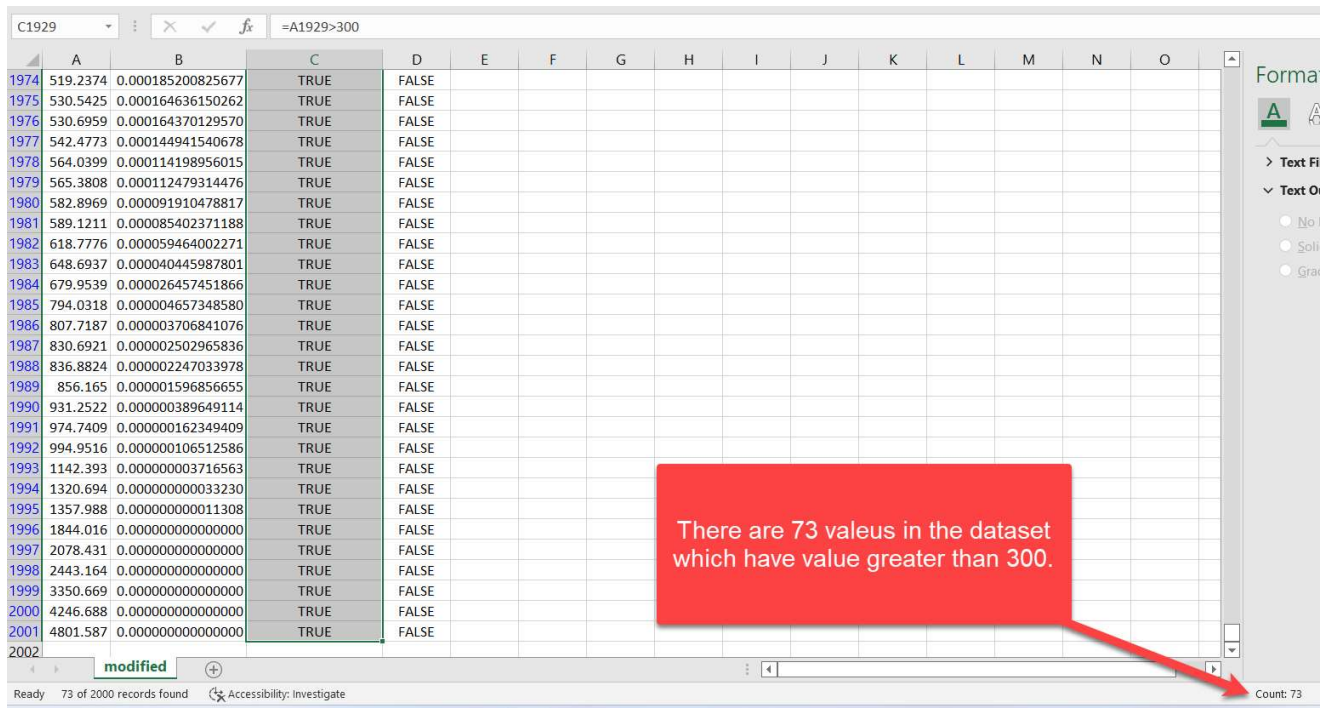
modified.xlsx

OUTPUT EXCEL:

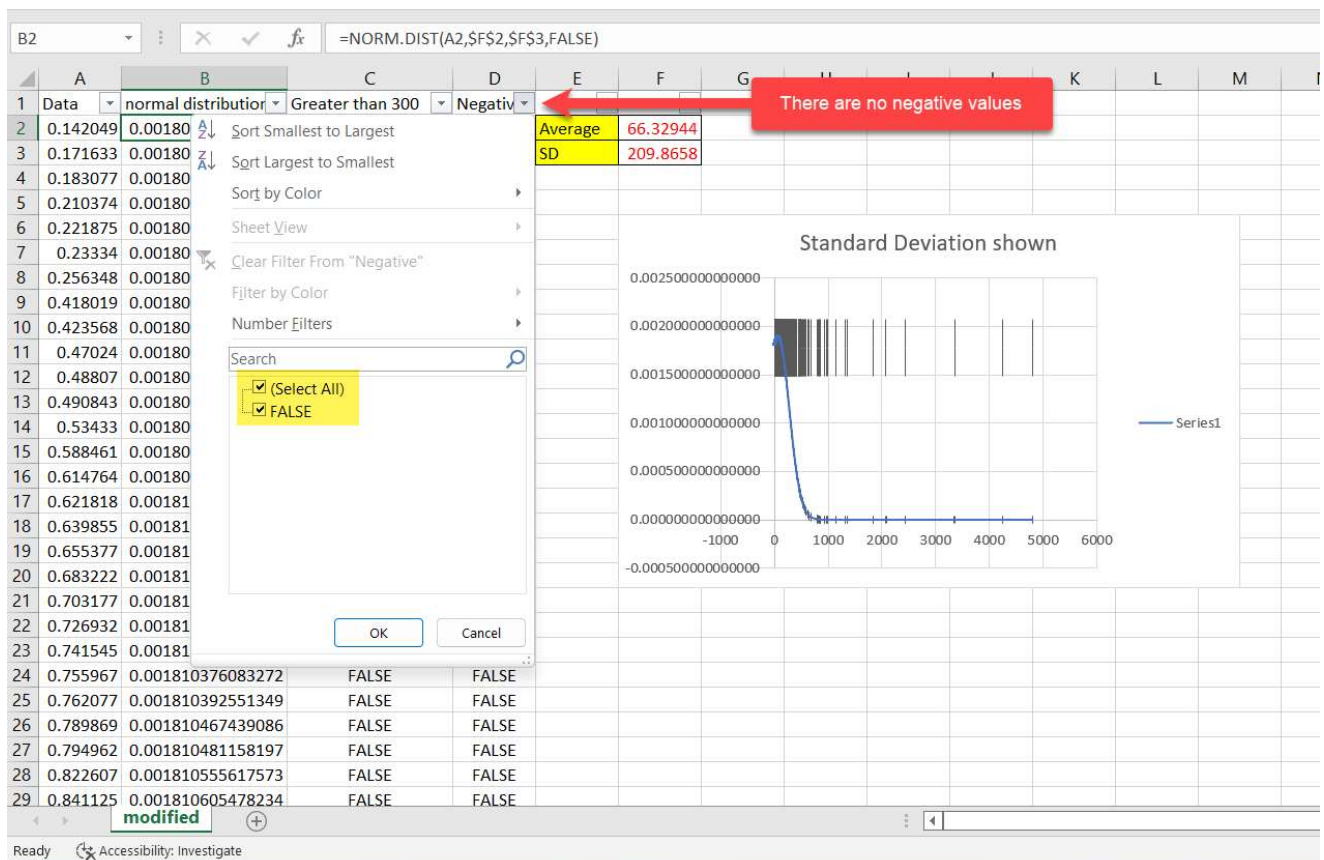


Actual data:

Column C calculates if a value is greater than 300. That is only 73 values are >300 .



Column D shows if the value in A is less than 0 . We find that after applying filter, there are no negative values.



The reason of discrepancies is because the dataset contains extreme values or outliers. Therefore our distribution is skewed. And hence our initial estimate did not give proper results.

References:

1. *Definite and indefinite integrals - MATLAB int - MathWorks India*. In.mathworks.com. Retrieved December 16, 2022, from <https://in.mathworks.com/help/symbolic/sym.int.html>
2. Conic Sections: Circle. Desmos. Retrieved December 16, 2022, from <https://www.desmos.com/calculator/ivud1auwvw>
3. Varsity Tutors. (2016). Normal Distribution of Data. Varsitytutors.com. https://www.varsitytutors.com/hotmath/hotmath_help/topics/normal-distribution-of-data

End

+++++

Lab 007

+++++

+++++

2154638

+++++