LAPORAN PRAKTIKUM PENGOLAHAN CITRA DIGITAL

6. IMAGE HISTOGRAMS

Disusun oleh :

Nama : Valencio Arjun

NPM : 2327250049

Kelas : IF41

PROGRAM STUDI INFORMATIKA

FAKULTAS ILMU KOMPUTER DAN REKAYASA

UNIVERSITAS MULTI DATA PALEMBANG

2025

TUTORIAL : IMAGE HISTOGRAMS

Goal

The goal of this tutorial is to use MATLAB and IPT to calculate and display image histograms.

Objectives

- Learn how to use the IPT function `imhist`.
- Learn how other MATLAB plotting techniques can be used to view and analyze histogram data.

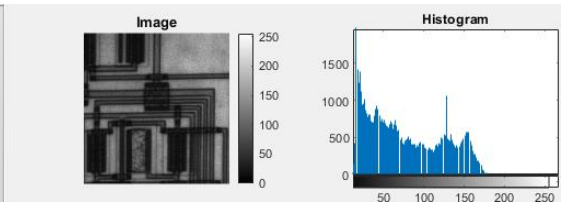
Procedure

Let us begin exploring the `imhist` function that is responsible for computing and displaying the histogram of an image.

1. Display an image and its histogram.

```
I = imread('circuit.tif');
figure, subplot(2,2,1), imshow(I), title('Image')
subplot(2,2,2), imhist(I,256), axis tight, title('Histogram')
```

```
>> I = imread('circuit.tif');
>> subplot(2,2,1), imshow(I), title('Image')
subplot(2,2,2), imhist(I,256), axis tight, title('Histogram')
>> %valencio arjun
```



2. The previous step displayed the default histogram for the image—a histogram with 256 bins. Let us see what happens if we change this value to 64 and 32.

```
subplot(2,2,3), imhist(I,64), axis tight, ...
    title('Histogram with 64 bins')
subplot(2,2,4), imhist(I,32), axis tight, ...
    title('Histogram with 32 bins')
```

You may have noticed that we set the axis to tight when displaying histograms. This adjusts the axis limits to the range of the data.

Question 1 Explain the drastic change of the Y-axis values when the histogram is displayed with fewer bins.

With less bins, the Y-axis adds frequently

There may be a need to postprocess the histogram data or display it using other plotting techniques. To do this, we need the values for each bin of the histogram. The following step illustrates this procedure.

3. Get the values of each bin in the histogram for later use.

```
c = imhist(I,32);
```

We can now use the values in c to display histogram using other plotting techniques. Naturally, the plot of a histogram displays the count of each bin,

but it may be more relevant to plot each bin's percentage. This can be done by normalizing the data, as shown in the next step.

4. Normalize the values in c.

```
c_norm = c / numel(I);
```

Question 2 What does the function numel do?

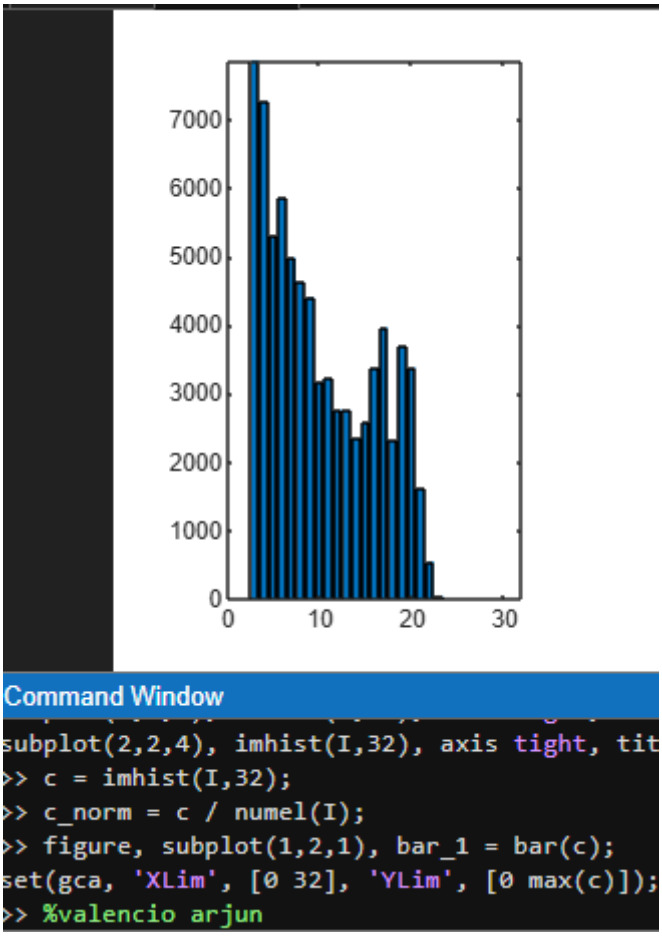
Normalizes the histogram so that the values represent probabilities (i.e., the fraction of the image that falls into each bin).

Question 3 Write a one line MATLAB statement that will verify that the sum of the normalized values add to 1.

isequal(round(sum(c_norm), 10), 1)

5. Close any open figures.
6. Display the histogram data using a bar chart.

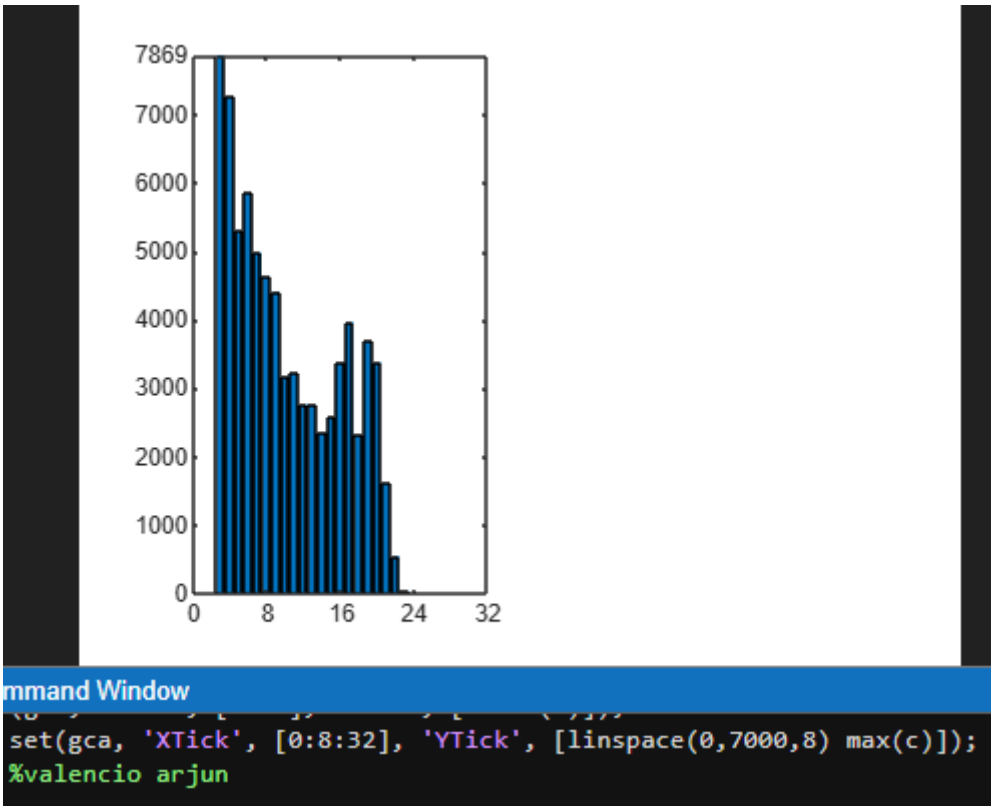
```
figure, subplot(1,2,1), bar_1 = bar(c);
set(gca, 'XLim', [0 32], 'YLim', [0 max(c)]);
```



In the previous step, we saw how the bar chart can be customized. In MATLAB, almost every object you create can be customized. When we create the bar chart, there is an axes object and a bar chart object displayed on the axes object. Here, the variable `bar_1` is set to the *bar chart* object so that we can reference it later for further customization. The `set` function allows us to change settings of a particular object. The first parameter of the `set` function is the object you wish to customize. In this case, the first object we customize is `gca`, which stands for get current axes. Here, we set the limits of the X and Y axes. Even though the limits have been set, the graph is still ambiguous because the tick marks on the X and Y axes do not reflect the limits.

7. Set the tick marks to reflect the limits of the graph.

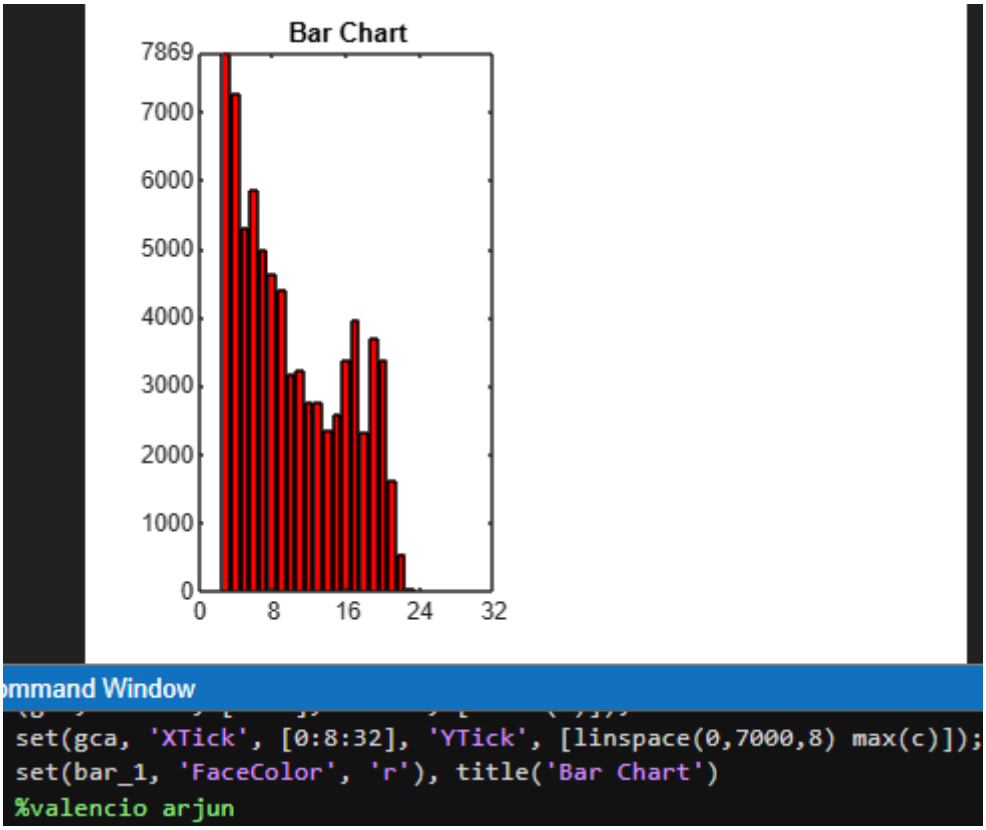
```
set(gca, 'XTick', [0:8:32], 'YTick', ...
    [linspace(0,7000,8) max(c)]);
```



Now the tick marks reflect the limits of the data. We used the set function to change settings of the current axes, but we can just as easily use it to customize the bar chart.

8. Use the set function to change the color of the bar chart. Also, give the chart a title.

```
set(bar_1, 'FaceColor', 'r'), title('Bar Chart')
```



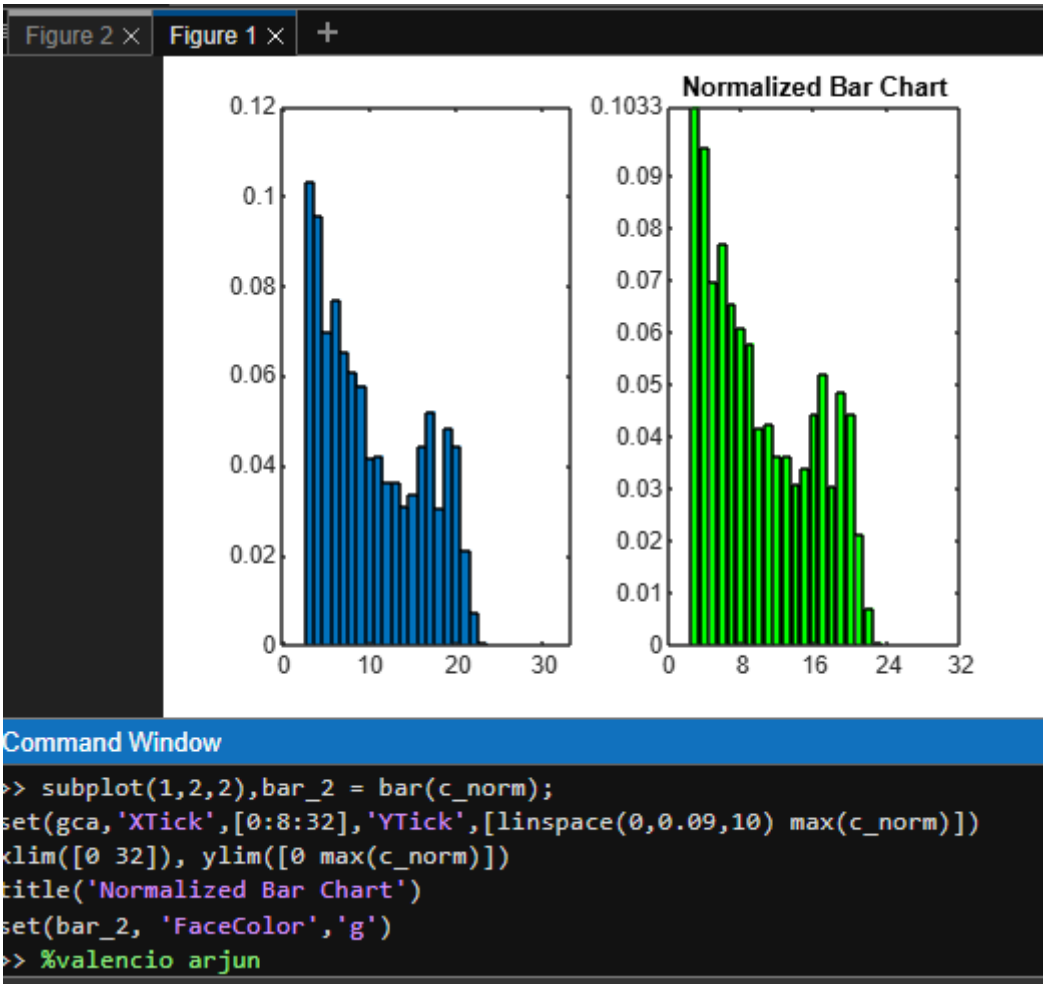
Question 4 How would we change the width of the bars in a bar chart?

bar_2 = bar(c_norm, 'BarWidth', 1);

Notice in the previous step how we used the bar chart object `bar_1` when changing settings. Similarly, we can display the normalized bar chart on the same figure using `subplot`.

9. Display the normalized bar chart and customize its display.

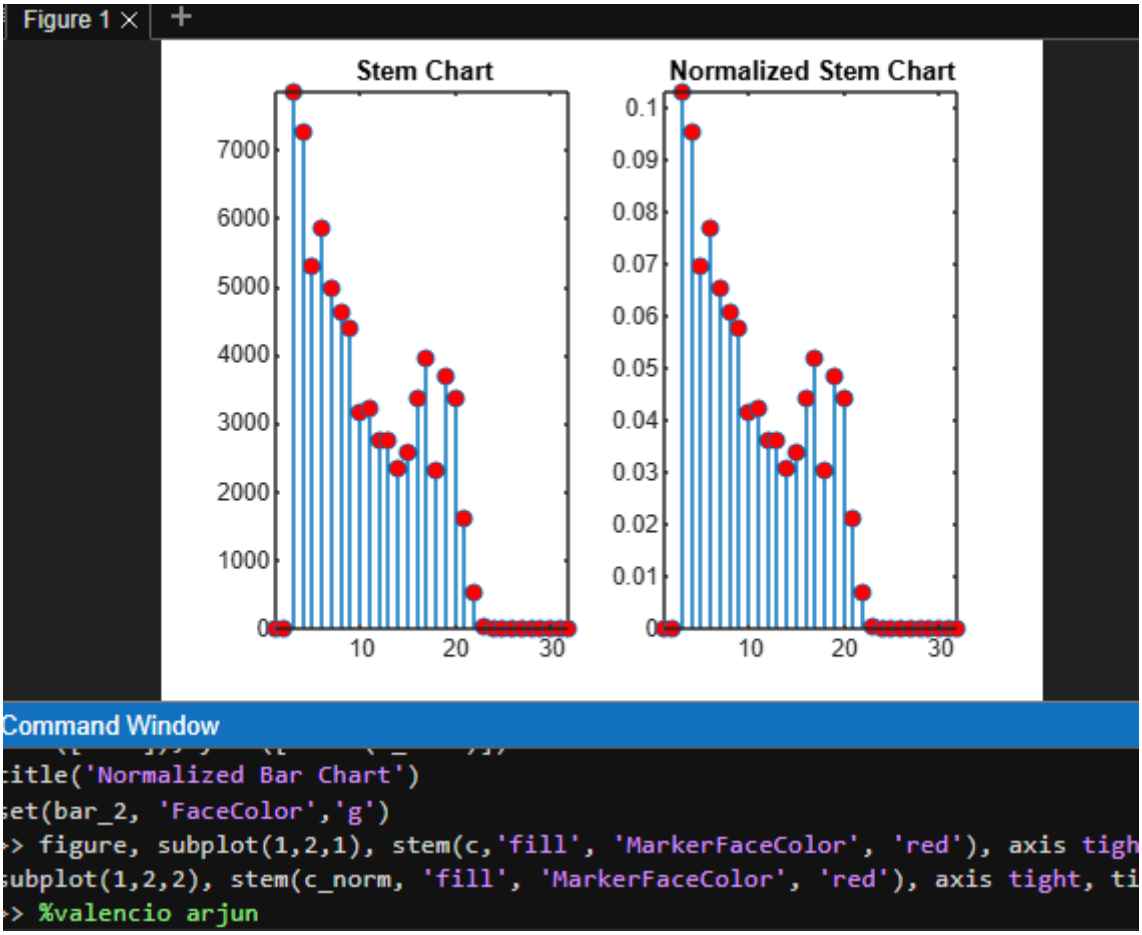
```
subplot(1,2,2), bar_2 = bar(c_norm);
set(gca, 'XTick', [0:8:32], 'YTick', ...
    [linspace(0,0.09,10) max(c_norm)])
xlim([0 32]), ylim([0 max(c_norm)])
title('Normalized Bar Chart')
set(bar_2, 'FaceColor', 'g')
```



Here, we made similar modifications as before. You may have noticed that we used `xlim` and `ylim` functions to set the limits of the axes. Sometimes there is more than one way to accomplish the same task, and this is an example of just that. Stem charts are similar to bar charts.

10. Close any open figures.
11. Display stem charts for both standard and normalized histogram data.

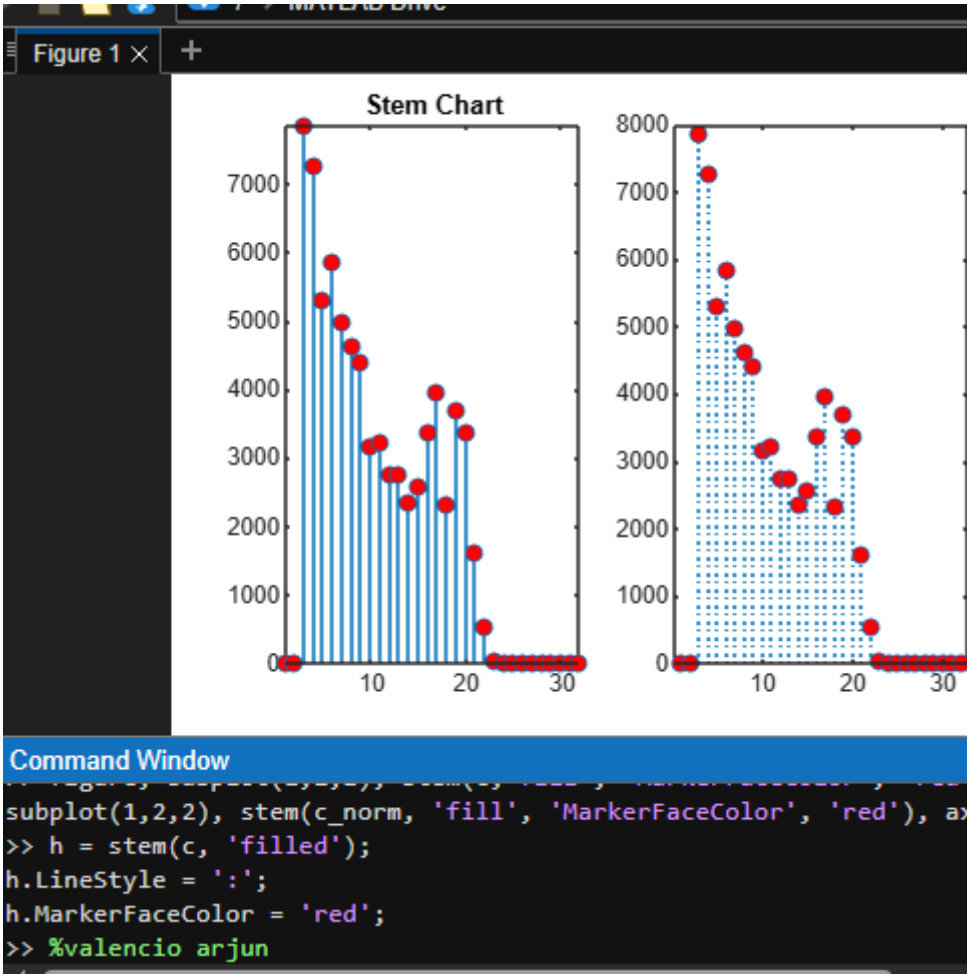
```
figure,
subplot(1,2,1), stem(c,'fill','MarkerFaceColor','red'), ...
    axis tight, title('Stem Chart')
subplot(1,2,2), stem(c_norm,'fill','MarkerFaceColor','red'), ...
    axis tight, title('Normalized Stem Chart')
```



In the previous step, we set visual properties of the stem charts by specifying the settings directly in the stem function call—we filled the marker and colored it red. We could have just as easily set a variable equal to the stem plot object and used the `set` function to make the changes.

Question 5 Explore the properties of stem charts. How can we make the lines dotted instead of solid?

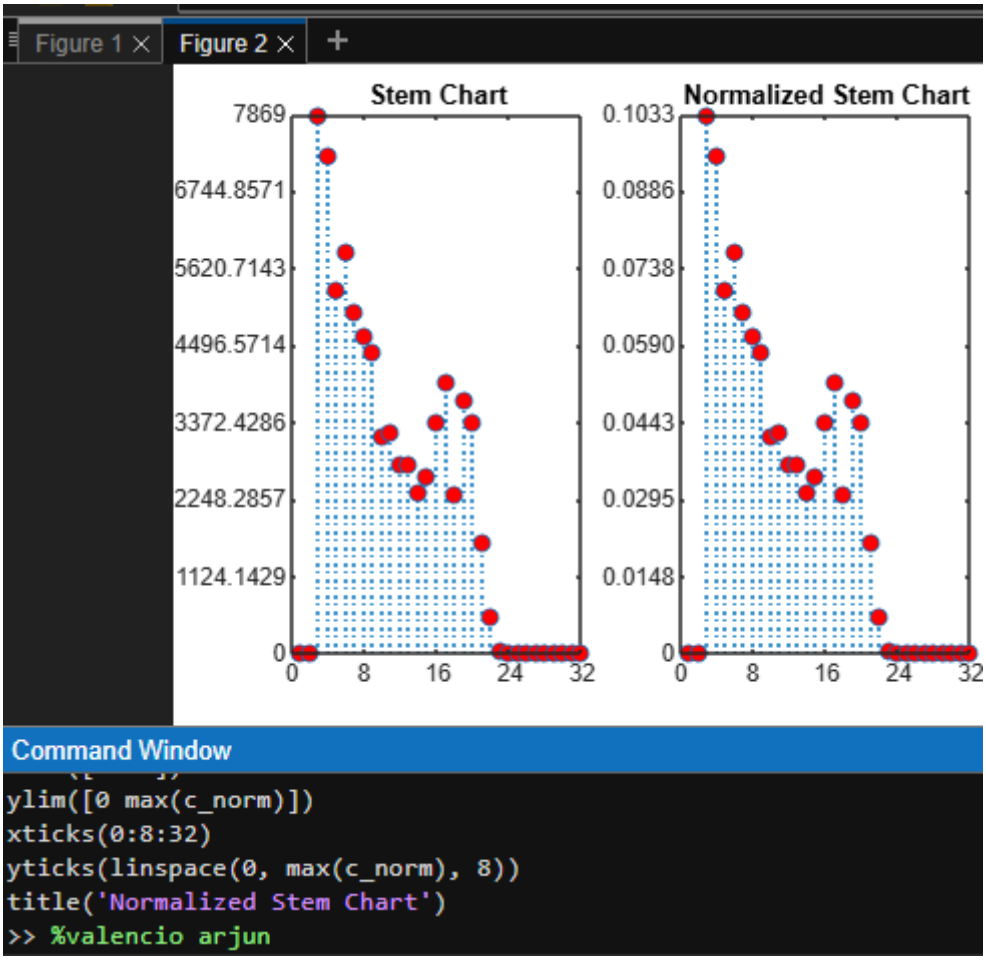
```
h = stem(c, 'filled');
h.LineStyle = ':';
h.MarkerFaceColor = 'red';
```



Question 6 Alter the axes limits and tick marks to reflect the data being displayed in the stem plot.

```
figure


% Subplot 1: Original histogram stem plot

subplot(1,2,1)

h1 = stem(c, 'filled');

h1.MarkerFaceColor = 'red';

h1.LineStyle = ':';

axis tight  % Optional: fits axis tightly
```

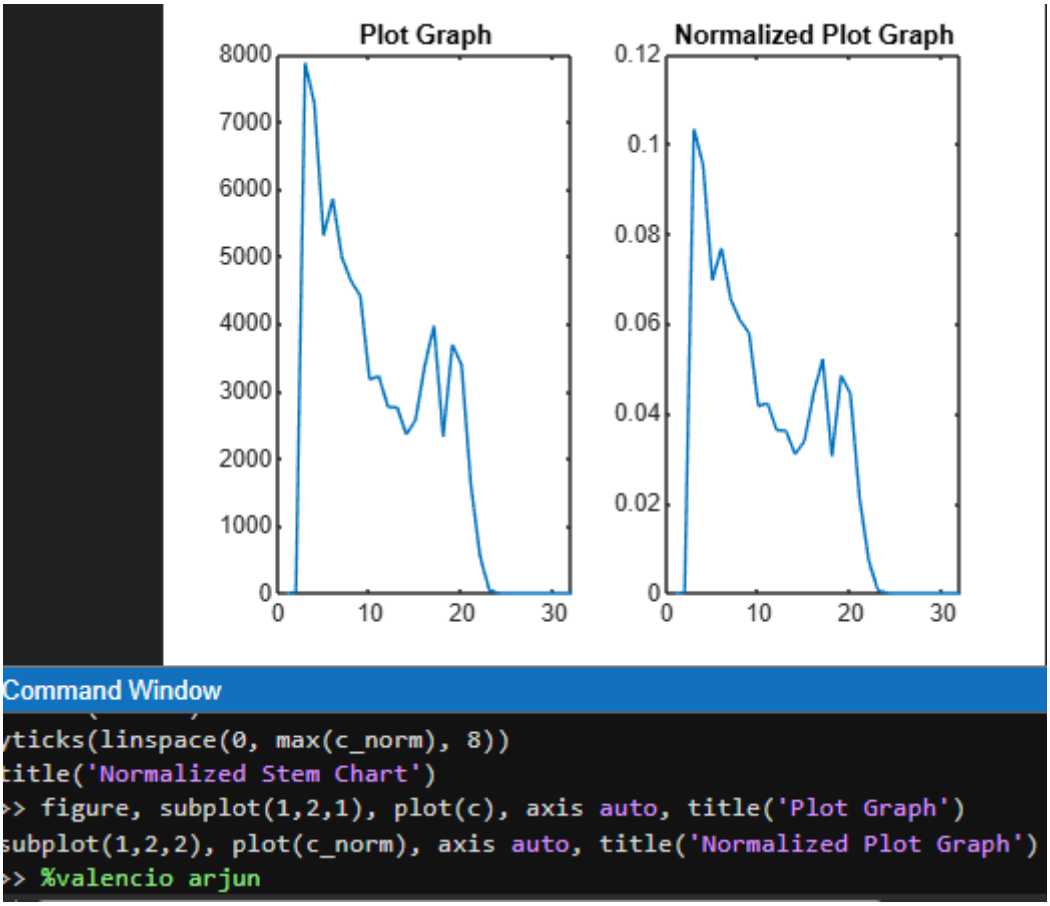
```
xlim([0 32])  % Set x-axis limits

ylim([0 max(c)])  % Set y-axis limits

xticks(0:8:32)

yticks(linspace(0, max(c), 8))

title('Stem Chart')



% Subplot 2: Normalized histogram stem plot

subplot(1,2,2)

h2 = stem(c_norm, 'filled');

h2.MarkerFaceColor = 'red';

h2.LineStyle = ':';

axis tight

xlim([0 32])

ylim([0 max(c_norm)])

xticks(0:8:32)

yticks(linspace(0, max(c_norm), 8))

title('Normalized Stem Chart')
```



The `plot` function will display the data by connecting each point with a straight line.

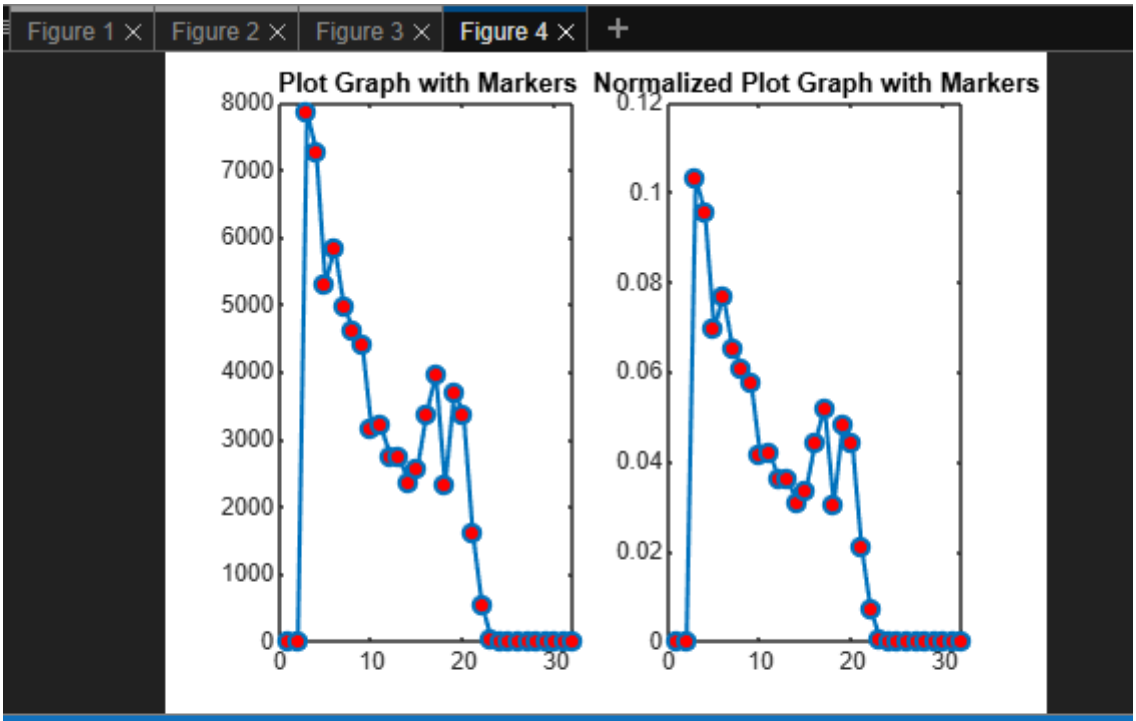
12. Display a plot graph for both standard and normalized histogram data.

```
figure, subplot(1,2,1), plot(c), axis auto, title('Plot Graph')
subplot(1,2,2), plot(c_norm), axis auto, ...
    title('Normalized Plot Graph')
```



Question 7 Explore the properties of plot graphs. In the above code, the points for each bin are visually lost within the graph line. How can we make the points bolder so that they are more visible?

```
figure % Subplot 1: Original histogram plot
subplot(1,2,1)
plot(c, '-o', 'MarkerSize', 6, 'LineWidth', 1.5, 'MarkerFaceColor', 'r')
axis auto
title('Plot Graph with Markers')
% Subplot 2: Normalized histogram plot
subplot(1,2,2)
plot(c_norm, '-o', 'MarkerSize', 6, 'LineWidth', 1.5, 'MarkerFaceColor', 'r')
axis auto
title('Normalized Plot Graph with Markers')
```



Command Window

```
subplot(1,2,2)
plot(c_norm, '-o', 'MarkerSize', 6, 'LineWidth', 1.5, 'MarkerFaceColor', 'r')
axis auto
title('Normalized Plot Graph with Markers')
>> %valencio arjun
```