Islamic University of Technology (IUT)

Organization of Islamic Cooperation (OIC)

Department of Electrical and Electronic Engineering (EEE)

COURSE NO : EEE 4416
LAB NO : 09 (Part – A)
TOPIC : Introduction to Plotting in MATLAB

Plotting is a powerful tool for understanding trends, visualizing data, and communicating results effectively. In the last lab, you worked with large datasets, which can be difficult to interpret just by examining raw CSV files. By applying appropriate visualization techniques, you can represent the underlying patterns and insights in the data more clearly and accurately.

MATLAB is great for visualizing data. It supports a wide range of plotting options, making it easy to explore, analyze, and present data effectively. In this lab, we will discuss –

i. Basic plots
ii. Subplots
iii. Customization
iv. 3D plots
v. Geographical plots
vi. Visualization from dataset
vii. Gif/video creation

Basic Plots

Some common types of plots include –

Type	MATLAB Command	Example Use
Line plot	plot	x-y graphs
Bar chart	bar	Compare values across categories
Scatter plot	scatter	Show patterns in points
Histogram	histogram	Distribution of data
Pie chart	pie	Percent distribution
3D plot	plot3, surf	3D data visualization

1. Line Plot

- **Use Case:** Visualize continuous trends or functions.
- **Basic Syntax:** plot(x, y)
- **Example:** visualizing a sine wave, plotting the V=IR curve

Asif Newaz, AKM Rakib
Lecturer, EEE, IUT

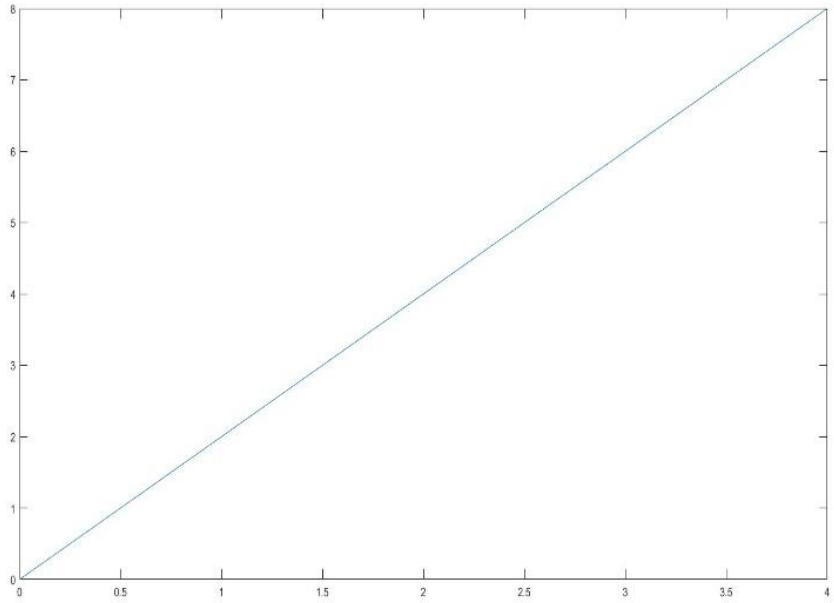
⇨ To obtain this kind of graph, you need to have a set of points for x (x-axis values) and their corresponding values of y.

IR characteristics curve

For instance, if you want to produce an ideal IR characteristics curve, it can be achieved this way –

```
x= [0, 1, 2, 3, 4]      % you have total 5 data points on the x-axis; indicating Voltage values
y= [0, 2, 4, 6, 8]      % corresponding Current values
plot (x, y)             % it will plot y values against x
```

The resultant plot from the above code is shown below. As you can see, this is a straight line.



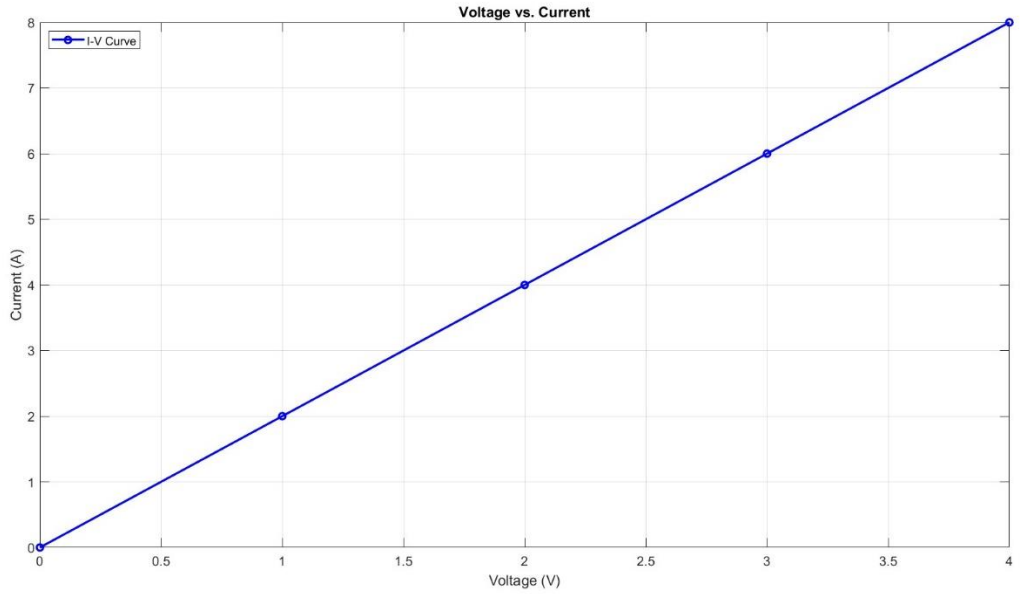
The figure will appear in a separate window. You can save the figure from there. Go to **File => Save as**. Give the file a name and save it in the .jpg or .png format so that Windows can read it. MATLAB has its own **.fig format,** which allows you to edit the figure later. However, you can only open a .fig file through MATLAB; Windows cannot read/open it.

The plot **lacks clarity** regarding the axis labels—specifically, it does not indicate what the x-axis and y-axis represent. Clearly labeling axes and providing appropriate titles or legends is essential when creating plots, as it ensures the figure is **informative, self-explanatory**, and **easy to interpret.**

You can incorporate this information using code or directly through the figure window. To do this using code, there are some simple functions such as xlabel, ylabel, title, legend, etc. Always increase the font size so that the labels are clearly visible. Use the **'insert' tab** in the figure window to add that information directly.

Run the following code. The resultant plot looks much better and clearer.

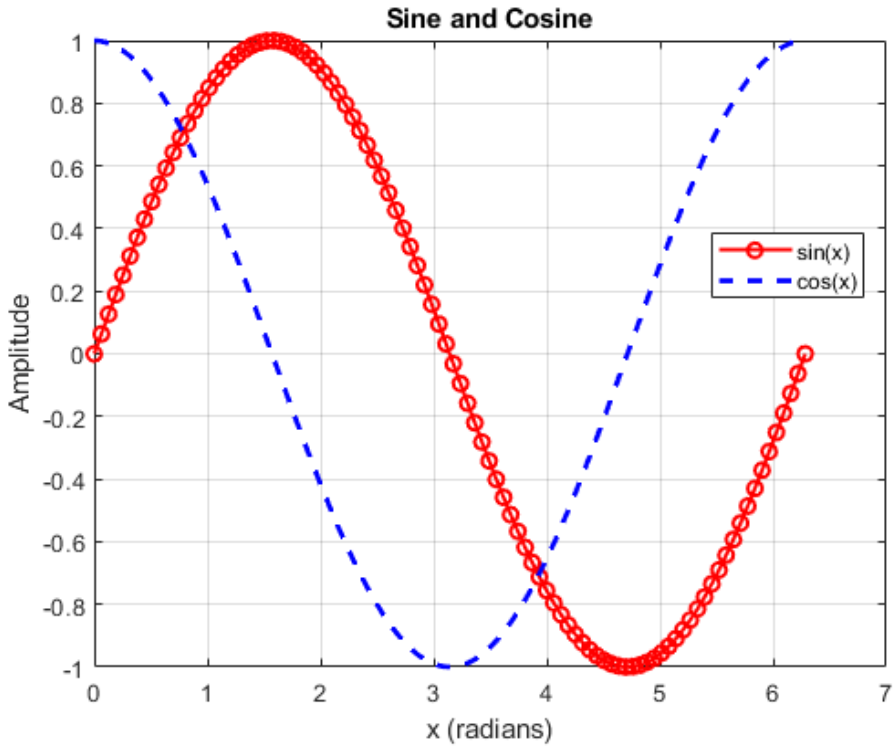
```
x = [0, 1, 2, 3, 4];       % Voltage values (x-axis)
y = [0, 2, 4, 6, 8];       % Current values (y-axis)

plot(x, y, 'bo-', 'LineWidth', 2);  % Plot with blue circles and solid line
grid on;                            % Add grid for better readability

xlabel('Voltage (V)', 'FontSize', 12);      % Label for x-axis
ylabel('Current (A)', 'FontSize', 12);      % Label for y-axis
title('Voltage vs. Current', 'FontSize', 14); % Title of the plot
legend('I-V Curve', 'Location', 'northwest'); % Optional legend

set(gca, 'FontSize', 12);  % Set font size for tick labels
```



Sine/Cosine Curve

Similarly, if you want to observe the sine/cosine function curve, you can write the code this way –

```
x = linspace(0,2*pi,100);
y1 = sin(x);
y2 = cos(x);


figure;
p1 = plot(x,y1,'-or','LineWidth',1.5,'MarkerSize',6);
hold on;
p2 = plot(x,y2,'--b','LineWidth',2);
hold off;


title('Sine and Cosine');
xlabel('x (radians)');
ylabel('Amplitude');
legend([p1,p2],{'sin(x)','cos(x)'},'Location','best');
grid on;
```



There are a variety of customization techniques available. Some of them are mentioned below. You can find more details in the MATLAB documentation.

Symbol	Line Style	Symbol	Line Style
-	solid line (default)	:	dotted line
--	dashed line	-.	dash-dot line

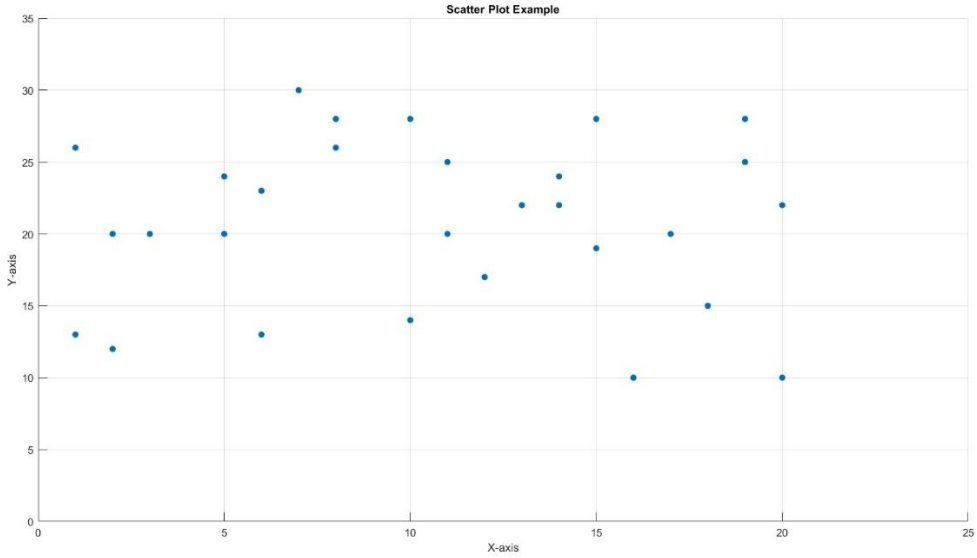
Marker Specifier	Description
+	plus sign
o	circle
*	asterisk
.	point
x	cross
s	square
d	diamond
^	upward pointing triangle
v	downward pointing triangle
>	right-pointing triangle
<	left pointing triangle
p	five-pointed star (pentagram)
h	six-pointed star (hexagram)

Asif Newaz, AKM Rakib
Lecturer, EEE, IUT

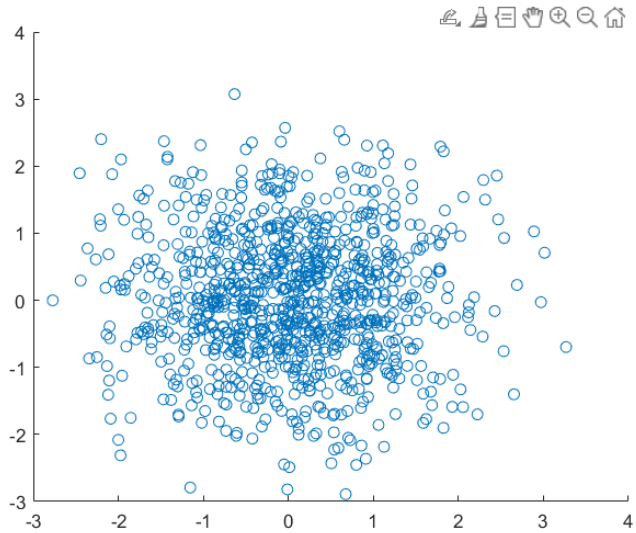
2. Scatter plot

A scatter plot is used to display **continuous** values for two variables for a set of data. It can show relationships and the density of data points. It can also help you find clusters in the data.

```
x= randi(20,1,30);
y= randi([10,30],1,30);


% Scatter plot
scatter(x, y, 'filled') % 'filled' fills the markers
title('Scatter Plot Example')
xlabel('X-axis')
ylabel('Y-axis')
grid on
axis([0,25,0,35])              % set the axis range
```




```
x = randn(1000,1);
y = randn(1000,1);
s = scatter(x,y);
```



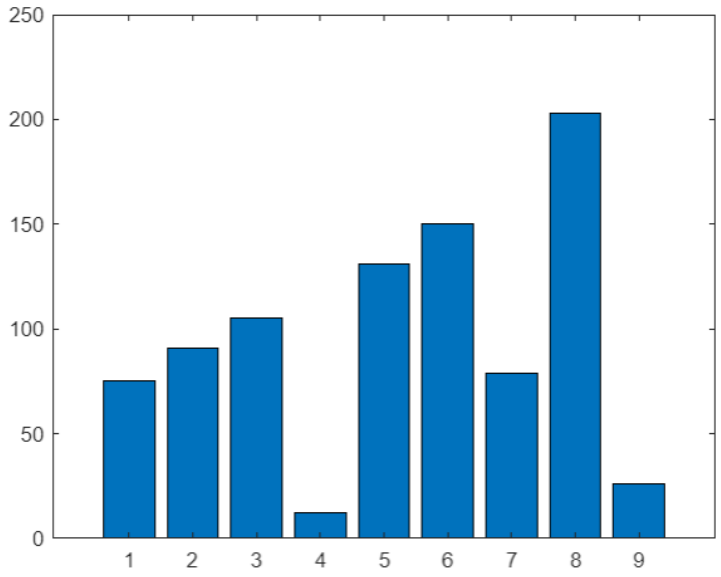
Asif Newaz, AKM Rakib
Lecturer, EEE, IUT

3. Bar Chart (`bar`, `barh`)

- **Use Case:** Compare **discrete categories** or grouped data.
- **Vertical Bar:** `bar(x,y)` or `bar(y)` where x can be indices.
- **Horizontal Bar:** `barh(y)`.
- **Grouped Bars:** `bar(X)` if X is a matrix (columns become groups).
- **Stacked Bars:** `bar(X,'stacked')`

```
y = [75 91 105 12 131 150 79 203 26]
bar(y)
```

⇨ The index values are automatically assigned to the x-axis.

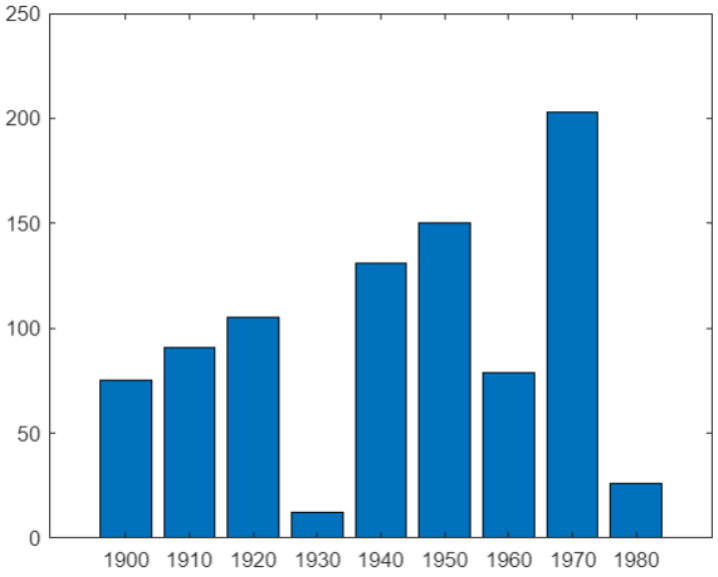



```
bar(y, 'r')
```

⇨ Use this to change the color to red.

```
x = 1900: 10: 1980
y = [75 91 105 12 131 150 79 203 26]
bar(x,y)
```

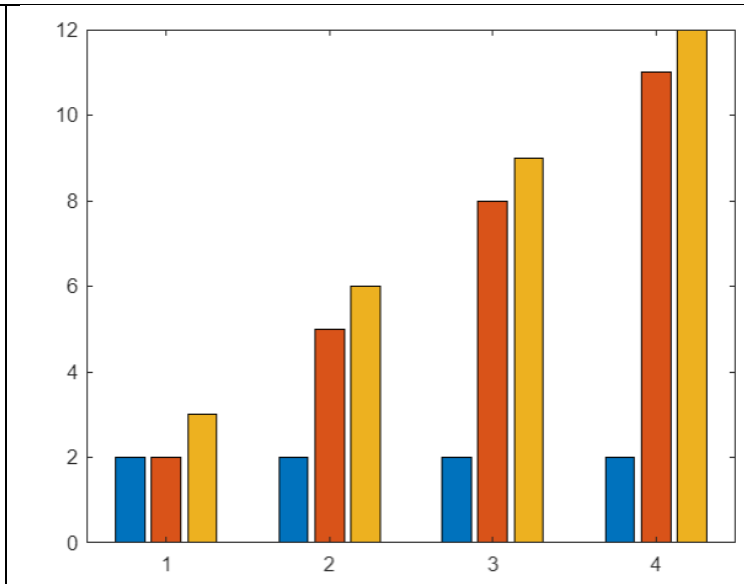
⇨ You can also change the size of the labels or xtick values; give the figure a title.



```
y = 4×3
     2     2     3
     2     5     6
     2     8     9
     2    11    12
```

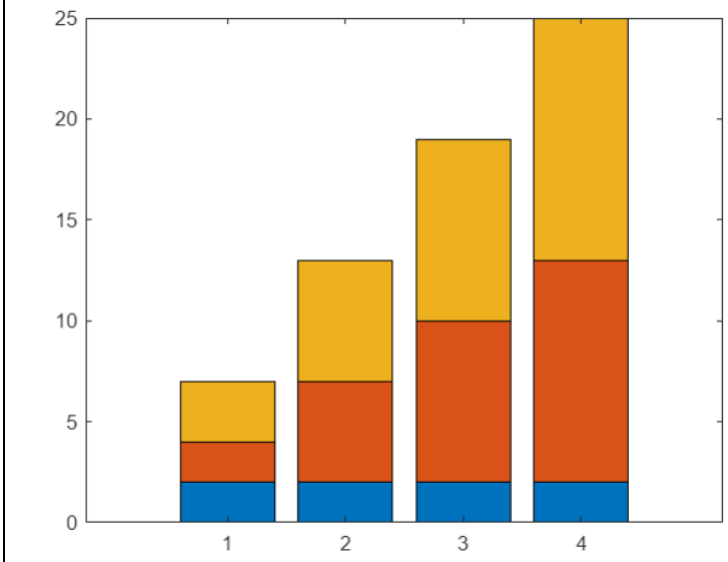
bar (y)

- ⇨ If y is a matrix, each row is considered one set, and they are grouped together.

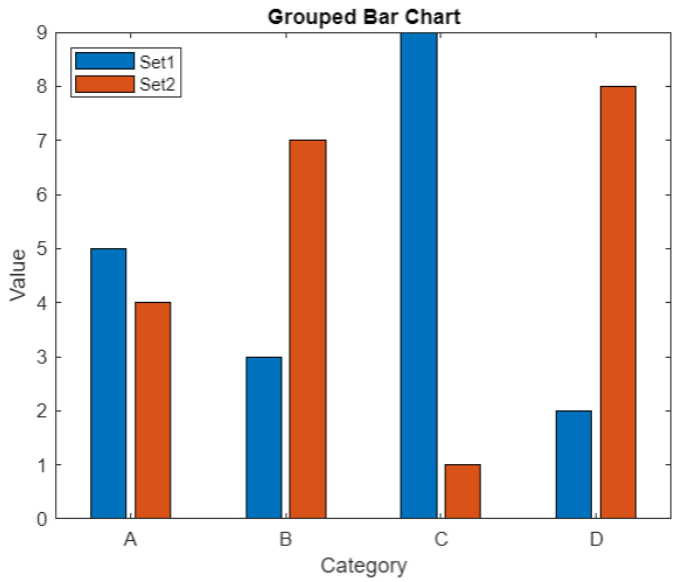


bar (y, 'stacked')

- ⇨ Using this, you can make the plot stacked.




```
categories = {'A','B','C','D'};

values1 = [5,3,9,2];
values2 = [4,7,1,8];

figure;
bar([values1; values2]','grouped');

set(gca,'XTickLabel',categories);

xlabel('Category'); ylabel('Value');
title('Grouped Bar Chart');

legend({'Set1','Set2'}, 'Location',
'northwest');
```



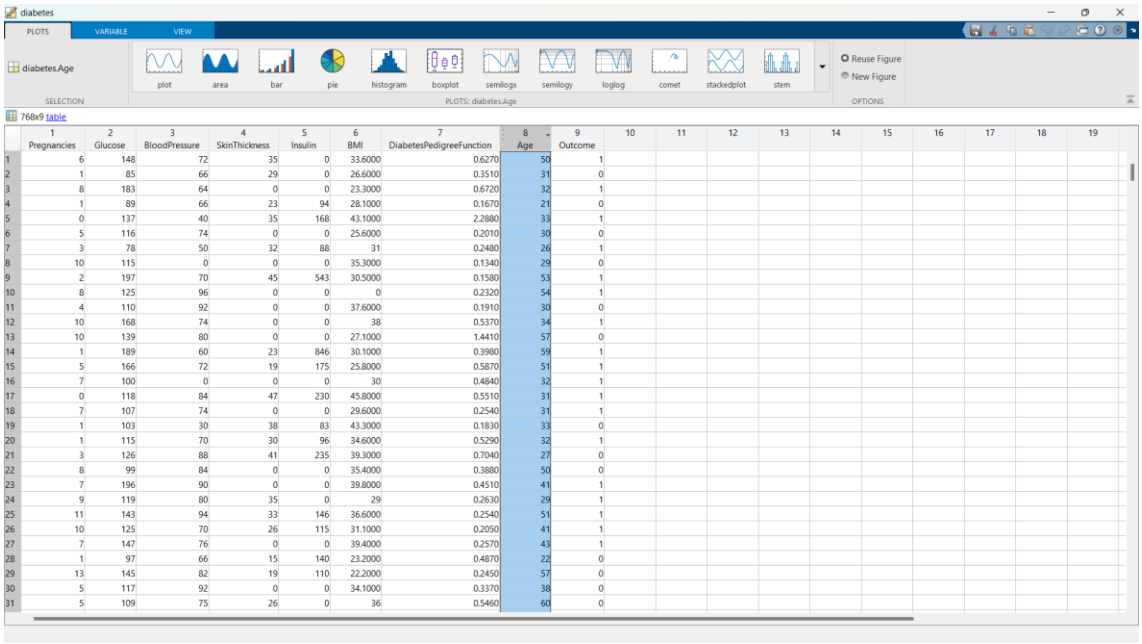
Asif Newaz, AKM Rakib
Lecturer, EEE, IUT

4. Histogram Plot

A histogram is a type of bar chart that shows the **distribution of numerical data** by grouping values into bins (intervals). It gives a quick visual impression of **how the data is spread**, including patterns like **skewness, peaks,** gaps, or **outliers.**

- X-axis: Represents the bins (ranges of data values).
- Y-axis: Represents the frequency (number of data points in each bin).
- The height of each bar indicates how many values fall within that range.

Import the **'diabetes.csv'** dataset into your workspace. Double-click on the data variable in your workspace. It will open the data in a data viewer window. From there, you can select a column separately and directly plot the values without writing any code using the **PLOTS** tab.

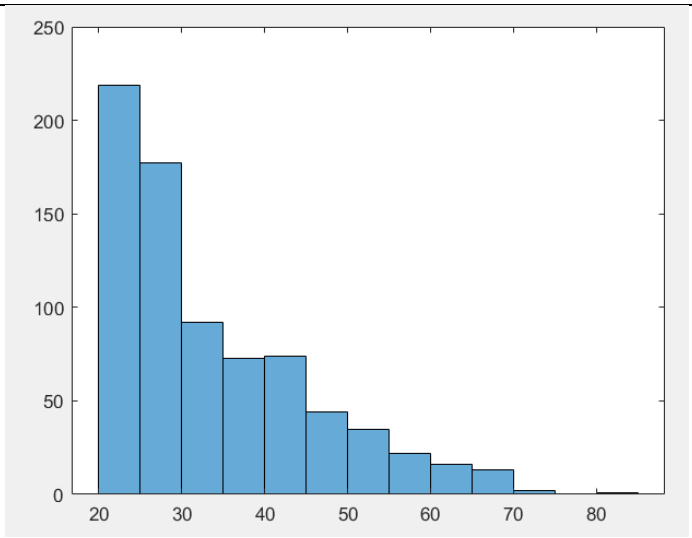


Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6000	0.6270	50	1
1	85	66	29	0	26.6000	0.3510	31	0
8	183	64	0	0	23.3000	0.6720	32	1
1	89	66	23	94	28.1000	0.1670	21	0
0	137	40	35	168	43.1000	2.2880	33	1
5	116	74	0	0	25.6000	0.2010	30	0
3	78	50	32	88	31	0.2480	26	1
10	115	0	0	0	35.3000	0.1340	29	0
2	197	70	45	543	30.5000	0.1580	53	1
8	125	96	0	0	0	0.2320	54	1
4	110	92	0	0	37.6000	0.1910	30	0
10	168	74	0	0	38	0.5370	34	1
10	139	80	0	0	27.1000	1.4410	57	0
1	189	60	23	846	30.1000	0.3980	59	1
5	166	72	19	175	25.8000	0.5870	51	1
7	100	0	0	0	30	0.4840	32	1
0	118	84	47	230	45.8000	0.5510	31	1
7	107	74	0	0	29.6000	0.2540	31	1
1	103	30	38	83	43.3000	0.1830	33	0
1	115	70	30	96	34.6000	0.5290	32	1
3	126	88	41	235	39.3000	0.7040	27	0
8	99	84	0	0	35.4000	0.3880	50	0
7	196	90	0	0	39.8000	0.4510	41	1
9	119	80	35	0	29	0.2630	29	1
11	143	94	33	146	36.6000	0.2540	51	1
10	125	70	26	115	31.1000	0.2050	41	1
7	147	76	0	0	39.4000	0.2570	43	1
1	97	66	15	140	23.2000	0.4870	22	0
13	145	82	19	110	22.2000	0.2450	57	0
5	117	92	0	0	34.1000	0.3370	38	0
5	109	75	26	0	36	0.5460	60	0

Age variable

Insights:

- There are a few patients over age 80.
- Most patients (> 200) are aged 20-30.
- The data is rightly skewed.

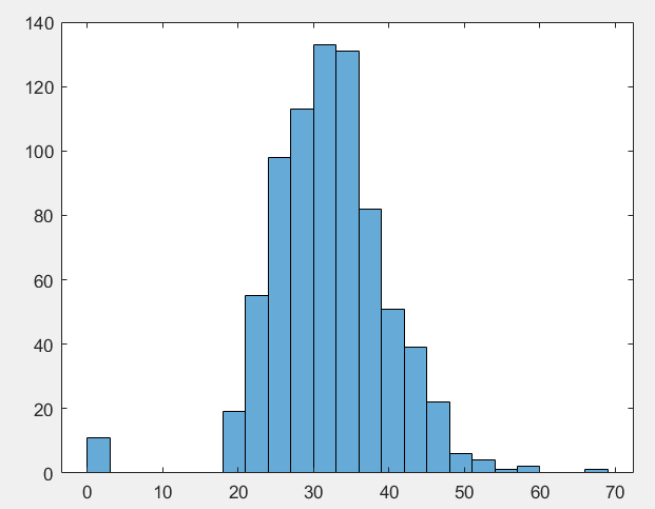


Asif Newaz, AKM Rakib
Lecturer, EEE, IUT

BMI

Insights:

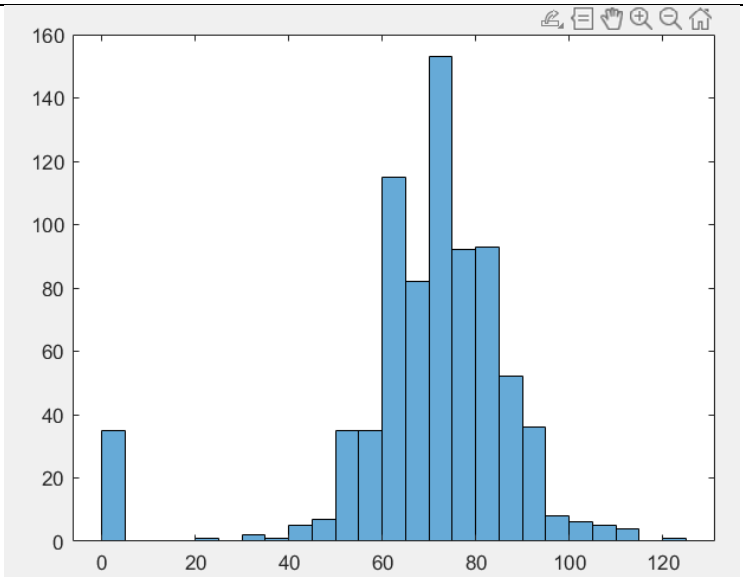
- There are some outliers (BMI > 55).
- Some data are faulty (BMI = 0).
- The average BMI is around 30.



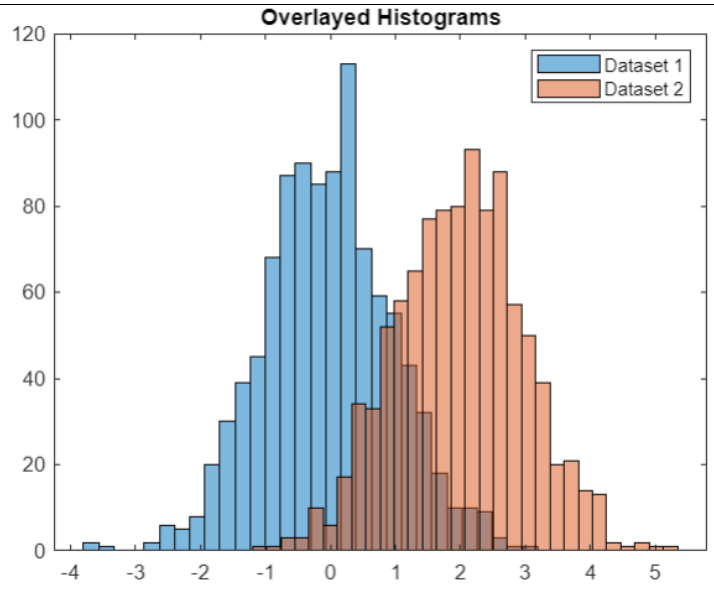
Blood Pressure

Insights:

- Some patients have BP = 0. This indicates that some data may be missing.




```
data1 = randn(1,1000);
data2 = randn(1,1000) + 2;
figure;
h1 =
histogram(data1,30,'FaceAlpha',0.5);
hold on;
h2 =
histogram(data2,30,'FaceAlpha',0.5);
hold off;
legend({'Dataset 1','Dataset 2'});
title('Overlayed Histograms');
```



Asif Newaz, AKM Rakib
Lecturer, EEE, IUT

5. Pie Charts

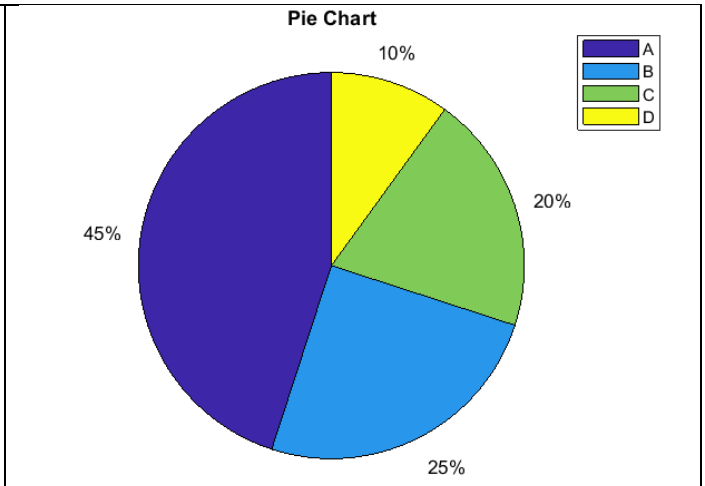
A pie chart is a type of circular chart used to show how a whole is divided into parts. It can be used to represent categorical data. Here,

- The circle represents 100% of the data.
- Each slice (sector) shows the proportion of one category relative to the total.
- The size (angle) of each slice is proportional to its value.

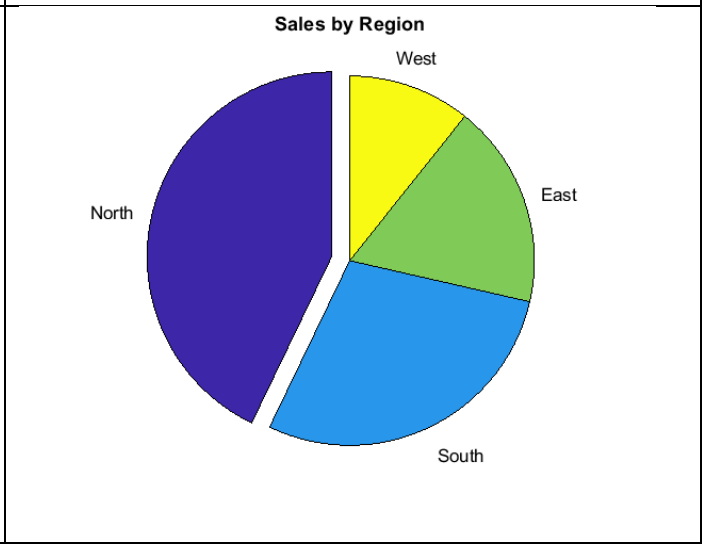
```
labels = {'A','B','C','D'};
values = [45 25 20 10];


figure
pie(values)
title('Pie Chart')
legend(labels, 'Location', 'northeastoutside')
```

- Try –

```
pie(values, labels)
```




```
sales = [120,80,50,30];
regions = {'North', 'South', 'East', 'West'};
explode = [1,0,0,0];  % emphasize first slice
figure;
pie(sales,explode,regions);
title('Sales by Region')
```

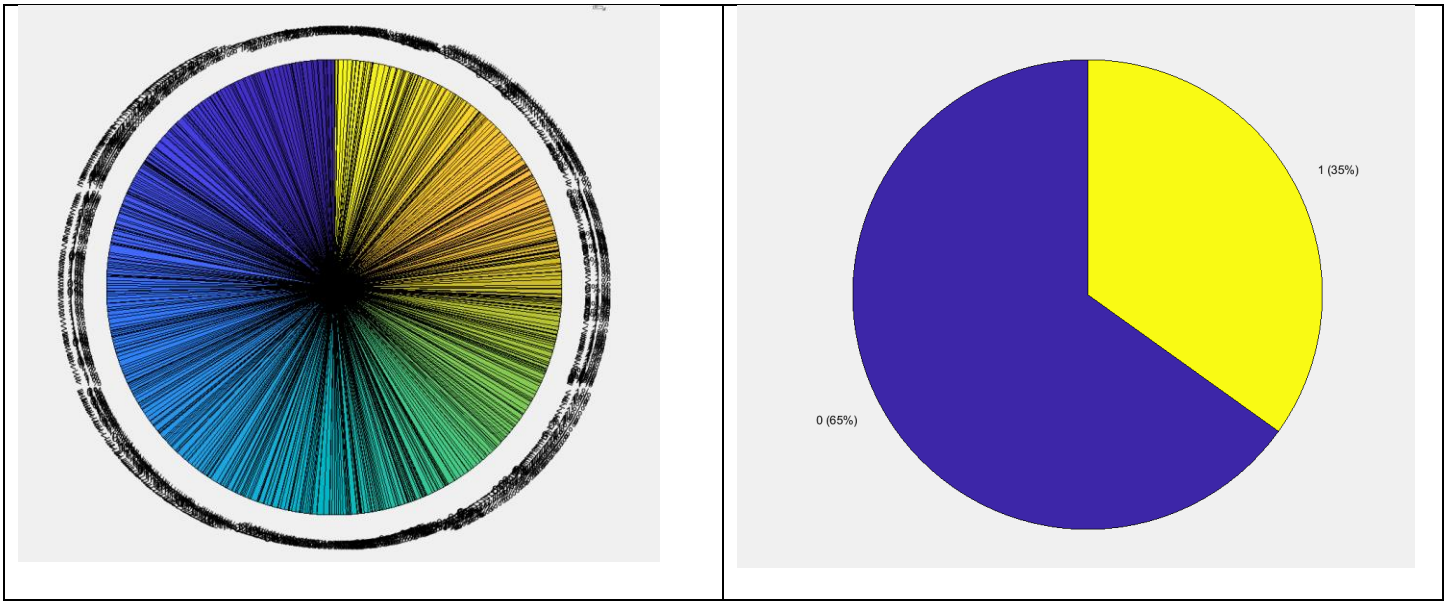


Pie chart from 'diabetes' dataset

- For the diabetes dataset, which features/columns can be plotted using a pie chart?
- If you try to plot columns like Outcome or Pregnancies, you will get quite an odd-looking figure (see below). Why is that?
- Check the datatype of each column. (You can use the 'summary' function)
- Which features are of categorical data types?

- The reason behind getting such an odd figure is that these two features are not of categorical data types (they are imported as a double data type). So, you need to convert them to obtain the desired pie chart.

```
diabetes.Outcome=categorical(diabetes.Outcome)
```

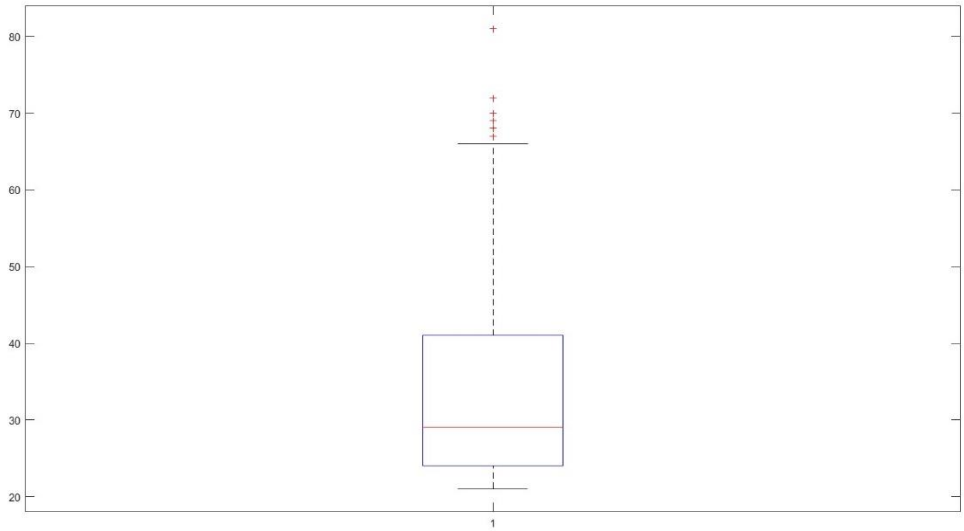


6. Box Plot

A box plot (also called a whisker plot) is used to show the distribution of a dataset and highlights its **spread, median, and outliers.**

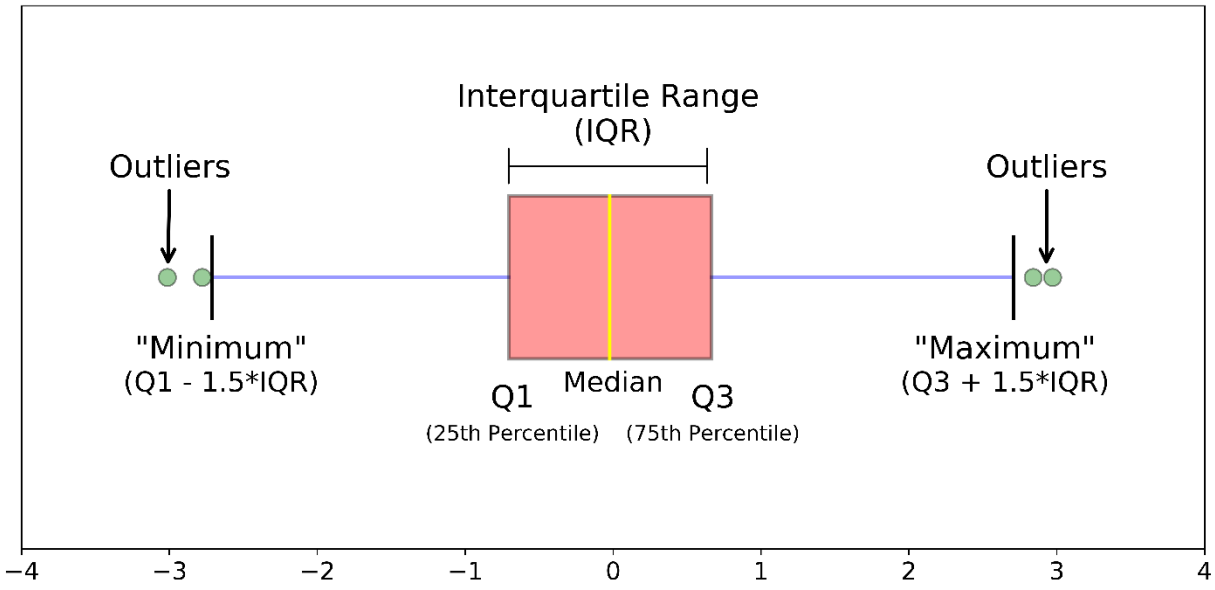
- An outlier is an extreme data point that deviates significantly from other observations in a dataset, often indicating a value that is much higher or lower than the majority of the data.

Let's try to obtain a box plot for the 'Age' column in the diabetes dataset using the PLOTS app.



A box plot contains several information –

i. **Box:** Interquartile range of the data (IQR = 25th percentile Q1 to 75th percentile Q3).
ii. **Line inside box:** Median (50th percentile)
iii. **Whiskers:** Extend to the smallest/largest values **within 1.5×IQR** of Q1 and Q3. Represented by a small line.
iv. **Points beyond whiskers:** Outliers



In different applications, box plots are quite useful in identifying the general distribution of the data and the presence of outliers.

⇨ Try to obtain box plots of other columns, such as insulin.

- These are some of the most commonly used plots, each with its own characteristics and specific purposes. Using them appropriately helps in extracting meaningful insights from data and effectively representing information.
- You will find several other types of plots in the 'PLOTS' tab in MATLAB. Explore yourself.

Lab Task

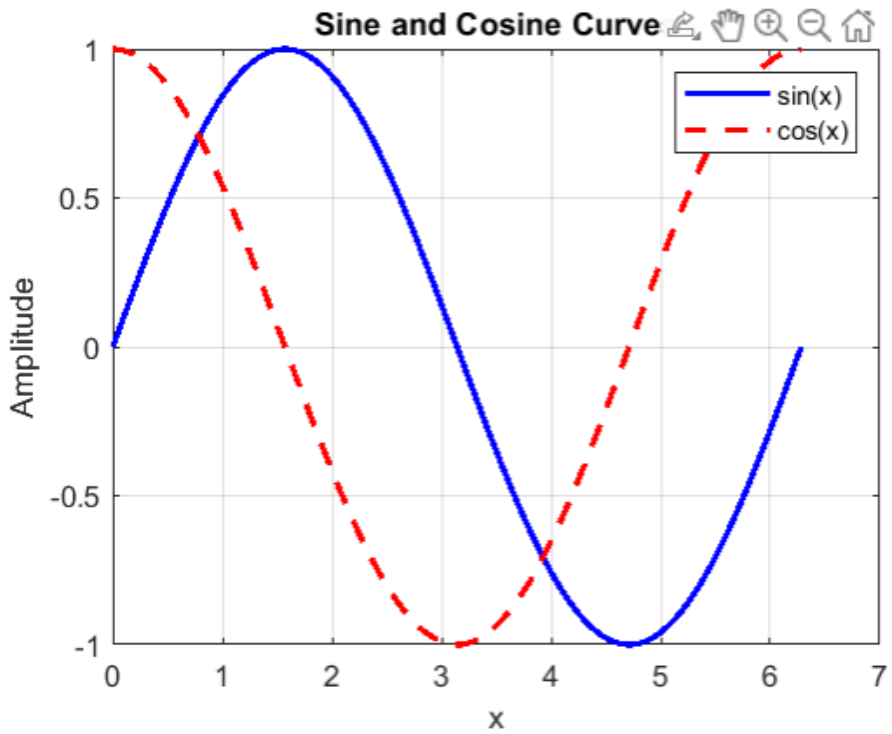
Car Dataset Exploration

- Load: load carbig
- Create:
  - Line plot of Acceleration vs. Model_Year.
  - Scatter of MPG vs. Weight with bubble sizes by Horsepower.
  - Histogram of Horsepower (30 bins, pdf normalization).
  - Pie chart of cylinder distribution.

Overlay multiple plots

The **hold on** command in MATLAB is used to overlay multiple plots on the same axes without erasing the existing content.

```
figure  % create a new figure

% Define x range
x = linspace(0, 2*pi, 100);


% Plot sine curve
plot(x, sin(x), 'b', 'LineWidth', 2);
hold on  % <-- keep current plot


% Plot cosine curve on same axes
plot(x, cos(x), 'r--', 'LineWidth', 2);
hold off % <-- release the hold on command


% Add labels and legend
xlabel('x');
ylabel('Amplitude');
title('Sine and Cosine Curves');
legend('sin(x)', 'cos(x)');
```



Asif Newaz, AKM Rakib
Lecturer, EEE, IUT