Basic Plotting

1. Overview
2. Axis labels, and annotations
3. Adding Titles
4. Specifying line styles and colours
5. creating simple plots
6. multiple data sets in one plot

Overview

plot(*xvalues,yvalues,'style-options'*)

examples

```
plot(x,y)          plots y vs x with a solid line

plot(x,y,'--')     plots y vs x with a dashed line

plot(x)            plots the elements of x afainst the rows
```

Style Options

Color Style options

```
y = yellow

m = magneta

c = cyan

r = red

g = green

b = blue

w = white

k = black
```

Line Style options

```
-  = solid

-- = dashed

: = dotted

-. = dashed-dot

none = no line
```

Marker Style options

```
+ = plus sign

o = circle

* = asterick

x = x-mark

. = point

^ = up triangle

s = square

d = diamond
```

Lable and Title

xlabel('Pipe Length')	labels x-axis with Pipe Length
ylabel('Fluid Pressure')	labels y-axis with Fluid Pressure
title('Pressure Variation')	titles the plot with Pressure Variation
legend(string1,string2,....)	produces legend using the text in string1 string2 etc as labels
legend(LineStyle1,sting1,....)	specifies the line style of each label writes the legend outside the plot frame
legend(....,pos)	if pos = -1 and inside if pos = 0,

there are other options for pos too, and

legned off	deletes the legend from the plot

Axis Control

axis([xmin xmax ymin ymax])

Examples

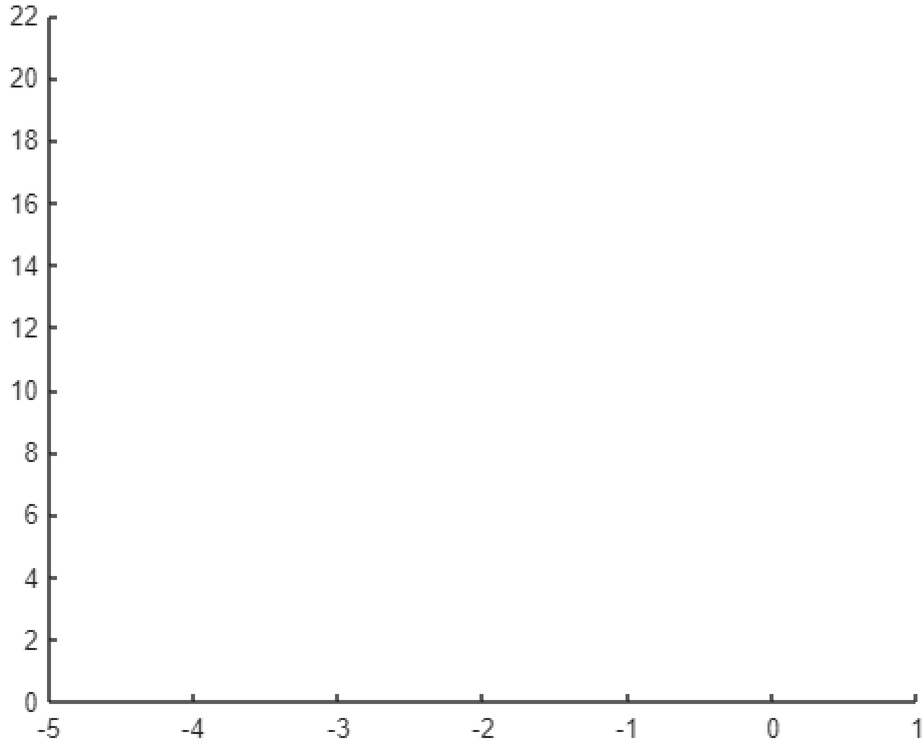
```
axis([-5 10 2 22]);                        %set the x-axis from 05 to 10 and y axis from 2 to
axy = [-5 10 2 22]; axis(axy)              %same as above,and
ax = [-5 10]; ay = [2 22]; axis([ax ay]);  %same as above
```

axis (' equal')	**sets equal scale on both axes,**
axis (' square')	**sets the default rectangular frame to a square,**
axis ('normal')	**resets the axis to default values,**
axis ('axis')	**freezes the current axes limits, and**
axis (' off')	**removes the surrounding frame and the tick marks.**

Semi Axis Control

```
axis ( [-5 10 -inf inf] )       %sets the x-axis limits at -5 and 10 and lets
```

```
                                %the y-axis limits be set automatically, and
axis ( [-5 inf -inf 22])        %sets the lower limit of the x-axis and the
```




```
                                % upper limit of the y-axis, and leaves the
                                % other two limits to be set automatically.
```

Overlay Plot

Method - 1 Using the plot command to generate overlay plots

plot(x1,y1,x2,y2,':',x3,y3,'o')

Method - 2 Using the hold command to generate overlay plots

```
x = linspace(0,2*pi,100)
```

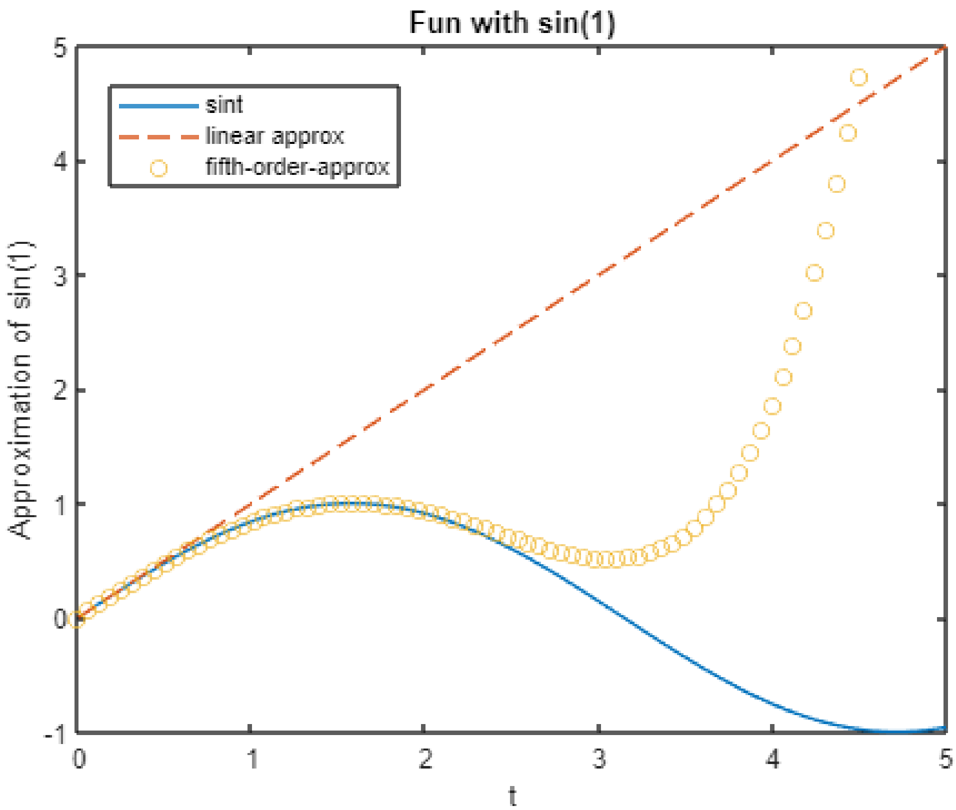
```
x = 1×100
         0    0.0635    0.1269    0.1904    0.2539    0.3173    0.3808    0.4443 ⋯
```

```
y1 = sin(x)
```

```
y1 = 1×100
         0    0.0634    0.1266    0.1893    0.2511    0.3120    0.3717    0.4298 ⋯
```

```
plot(x,y1)
hold on
y2 = x;
plot(x,y2,'--')
y3 = x - (x.^3)/6 + (x.^5)/120;
plot(x,y3,'o')
axis([0 5 -1 5])
xlabel('t')
ylabel('Approximation of sin(1)')
title('Fun with sin(1)')
legend('sint','linear approx','fifth-order-approx')
hold off

legend("Position",[0.16091,0.76058,0.25714,0.11905])
```



Method - 3 Using the line command to generate overlay plots

Specailized 2D plot

```
area - creates a filled area plot
bar - creates a bar graph
barh - creates horizontal bar graph
comet - makes animated 2D plot
compass - creates arrow graph for complex numbers
contour - makes contour plots
contourf - makes filled contour plots
errorbar - plots a graph and puts error bars
feather - makes feather plot
fill - draws a filled ploygon
fplot - plots a function of a single variable
hist - makes histograms,
loglog - creates plot with log scale on both the r-axis and the y-axis,
pareto - makes pareto plots,
pcolor - makes pseudocolor plot of a matrix,
pie - creates a pie chart,
plotyy - makes a double y-axis plot,
plotmatrix - makes a scatter plot of a matrix,
polar - plots curves in polar coordinates,
quiver - plots vector fields,
rose - makes angled histograms,
```

```
scatter - creates a scatter plot,
semilogx - makes semilog plot with log scale on the x-axis,
semilogy - makes semilog plot with log scale on the y-axis,
stairs - plots a stair graph, and
stem - plots a stem graph.
```

Let’s say that you want to plot these two equations in the same window:

$y_1 = \cos(x)$

$y_2 = x^2 - 1$

Steps for 2D Plots

1. Define your interval of interest, think of highest and lowest values, and a step.
2. Define your function y = f(x). Take into account that you’re working with arrays, not with scalars, use dot operators.
3. Use appropriate 2D built-in functions

1. Define your Interval

Think:

- What values for x do I want to take intoaccount?
- What steps in the array should I consider?

2. Define your Function(s)

Think of lower and upper values, and steps

```
x = -1 : 0.1 : 1.5;
y1 = cos(x);
y2 = x.^2 - 1;
```

Now, x, y1 and y2 are vectors with appropriate values.

3. Use 2D built-in Functions

You can use functions such as:

plot

stem

polar, compass, rose

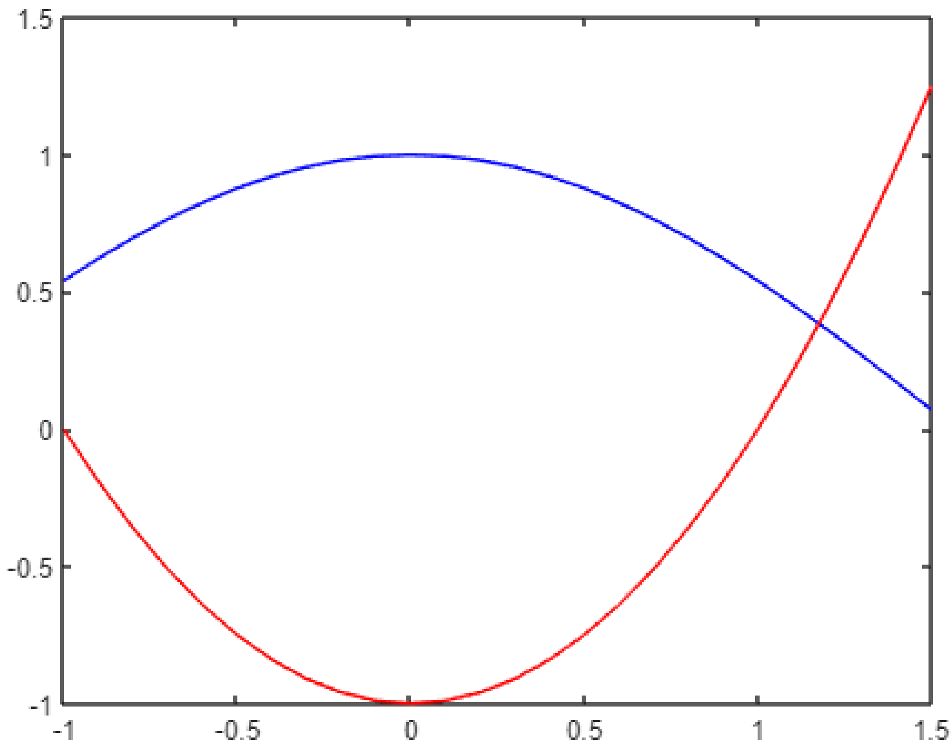
loglog, semilogx, semilogy

area,fill

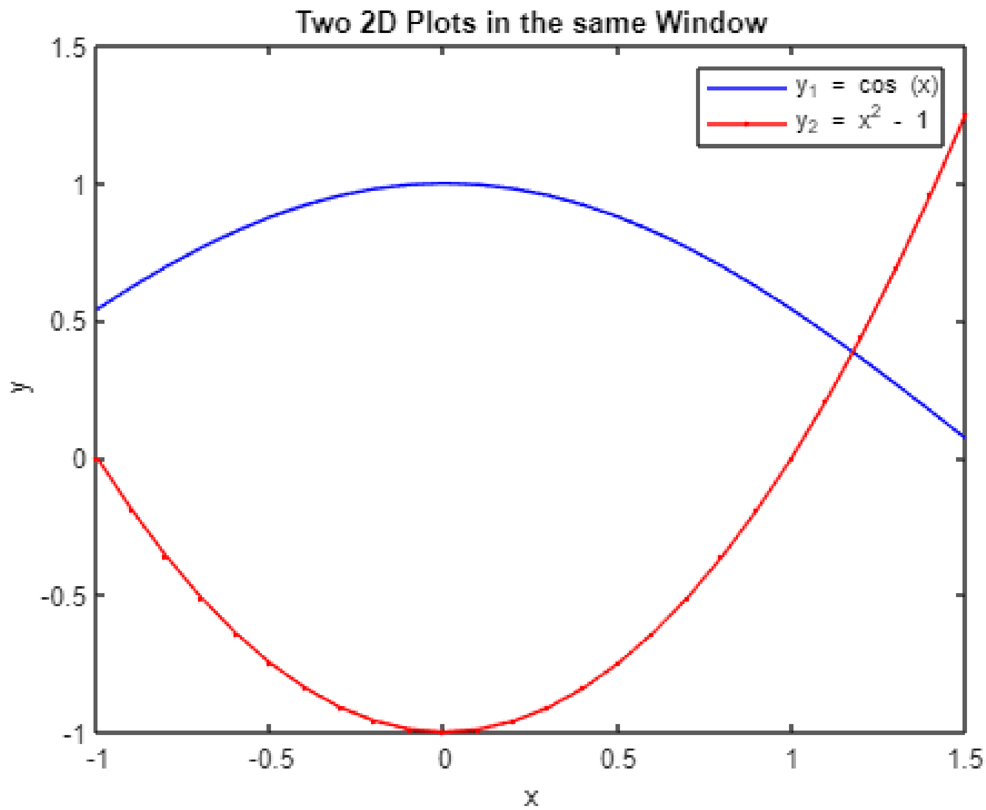
pie

hist, stairs

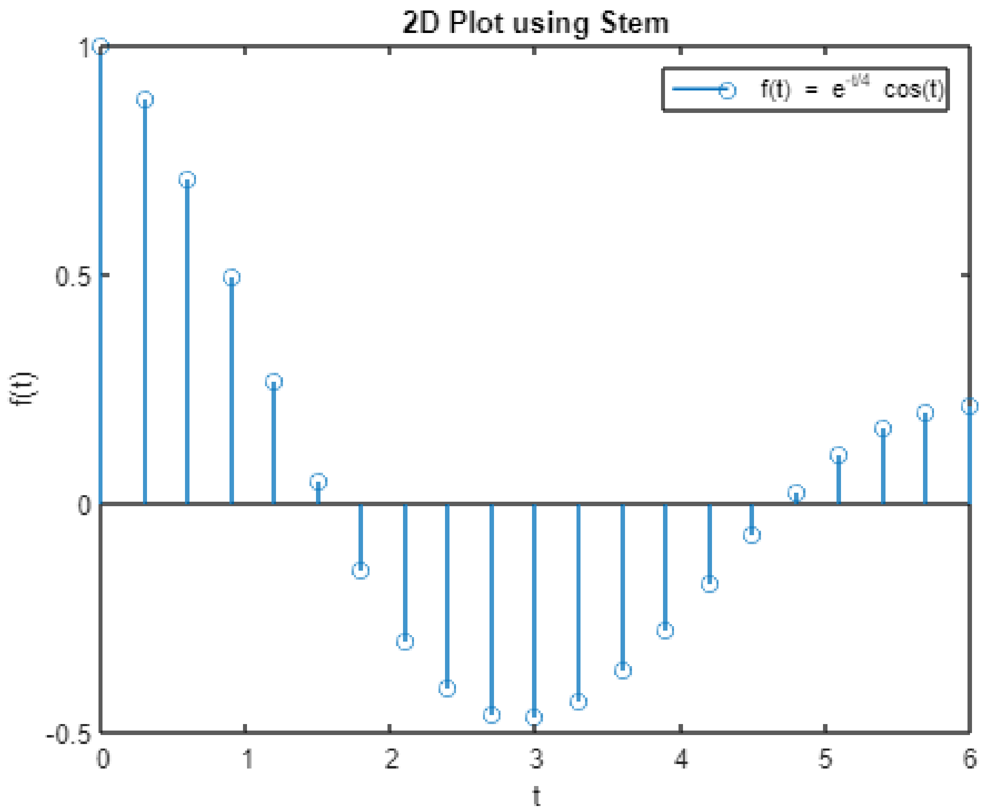
```
plot(x,y1,'b',x,y2,'r')
```




```
plot(x, y1, 'b' , x, y2, 'r.-' )
title('Two 2D Plots in the same Window' )
legend('y_1 = cos (x)','y_2 = x^2 - 1')
xlabel('x')
ylabel('y')
```




```
t = 0 : .3 : 2*pi;
f = exp(-t/4) .* cos(t);
stem(t,f)
title('2D Plot using Stem' )
legend( 'f(t) = e^{-t/4} cos(t) ')
xlabel('t' )
ylabel('f(t)')
```




```
rows = randi(1,5)
```

```
rows = 5×5
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
```

```
col = randi(1,5)
```

```
col = 5×5
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
```

```
x = length(rows)
```

```
x = 5
```

```
y = length(col)
```

```
y = 5
```

```
mat = zeros(x,y)
```

```
mat = 5×5
     0     0     0     0     0
```

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

fplot

- `fplot(@fun,lims)` - plots the function fun between the x-axis limits
- `lims = [xmin xmax ymin ymax]` - axis limits
- The function `fun(x)` must return a row vector for each element of vector x.

AXIS Control

- axis scaling and appearance
- `axis([xmin xmax ymin ymax])`
- Sets scaling for the x- and y-axes on the current plot
- **`axis auto`** - returns the axis scaling to its default, automatic mode.
- **`axis off`** - turns off all axis labelling,tick marks and background.
- **`axis on`** - turms axis labeling, tick marks and background.
- **`axis equal`** - makes both axes equal.

3D Plot

the general syntax for plot3 command is

plot3(x,y,z,'*style-option*')

`plot3` - plots curves in space,

`stem3` - creates discrete data plot with stems in 3-D,

`bar3` - plots 3-D bar graph,

`bar3h` - plots 3-D horizontal bar graph,

`pie3` - makes 3-D pie chart,

`comet3` - makes animated 3-D line plot,

`fill3` - draws filled 3-D polygons,

`contour3` - makes 3-D contour plots,

`quivers` - draws vector fields in 3-D,

`scatter3` - makes scatter plots in 3-D,

`mesh` - draws 3-D mesh surfaces (wire-frame),

`meshc` - draws 3-D mesh surfaces along with contours,

`meshz` - draws 3-D mesh surfaces with reference plane curtains,

`surf` - creates 3-D surface plots,

`surfc` - creates 3-D surface plots along with contours, `surfl` - creates 3-D surface plots with specified light source,

`trimesh` - mesh plot with triangles,

`trisurf` - surface plot with triangles,

`slice` - draws a volumetric surface with slices,

`waterfall` - creates a waterfall plot of 3-D data,

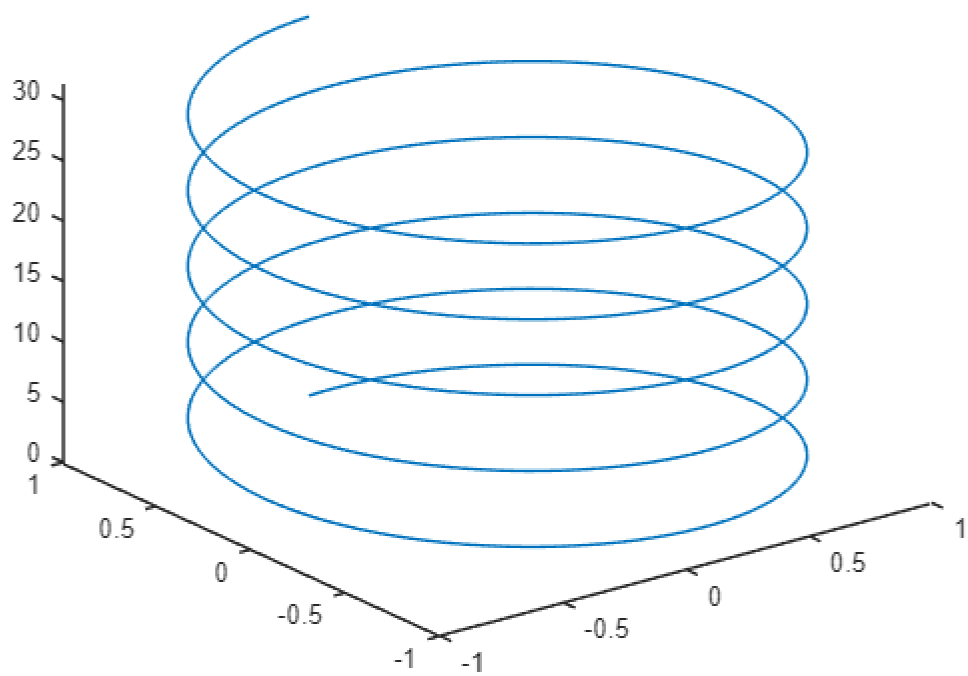
`cylinder` - generates a cylinder,

`ellipsoid` - generates an ellipsoid, and

`sphere` - generates a sphere.

3D Plot: Question 1

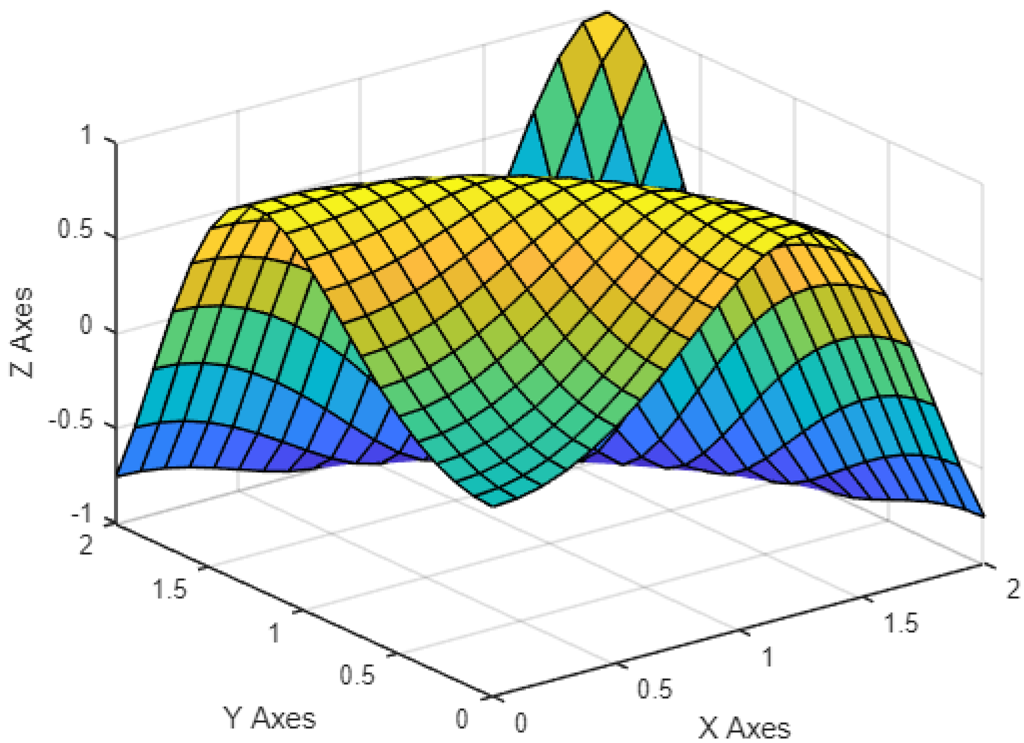
```
t = 0:pi/50:10*pi;
plot3(sin(t),cos(t),t)
```



Surface Plot: Question 2

```
x = 0:0.1:2;
y = 0:0.1:2;
[xx,yy] = meshgrid(x,y);
zz = sin(xx.^2+yy.^2);
surf(xx,yy,zz)
```

```
xlabel('X Axes')
ylabel('Y Axes')
zlabel('Z Axes')
```

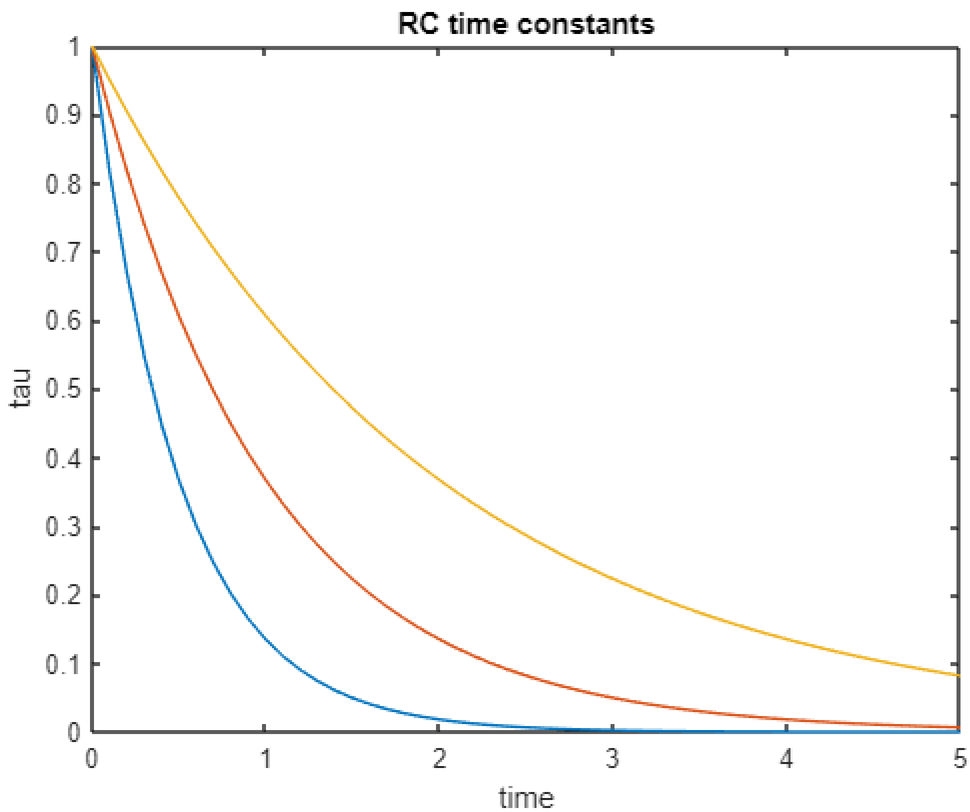


Question 3

Plot voltage vs time for various RC time constants

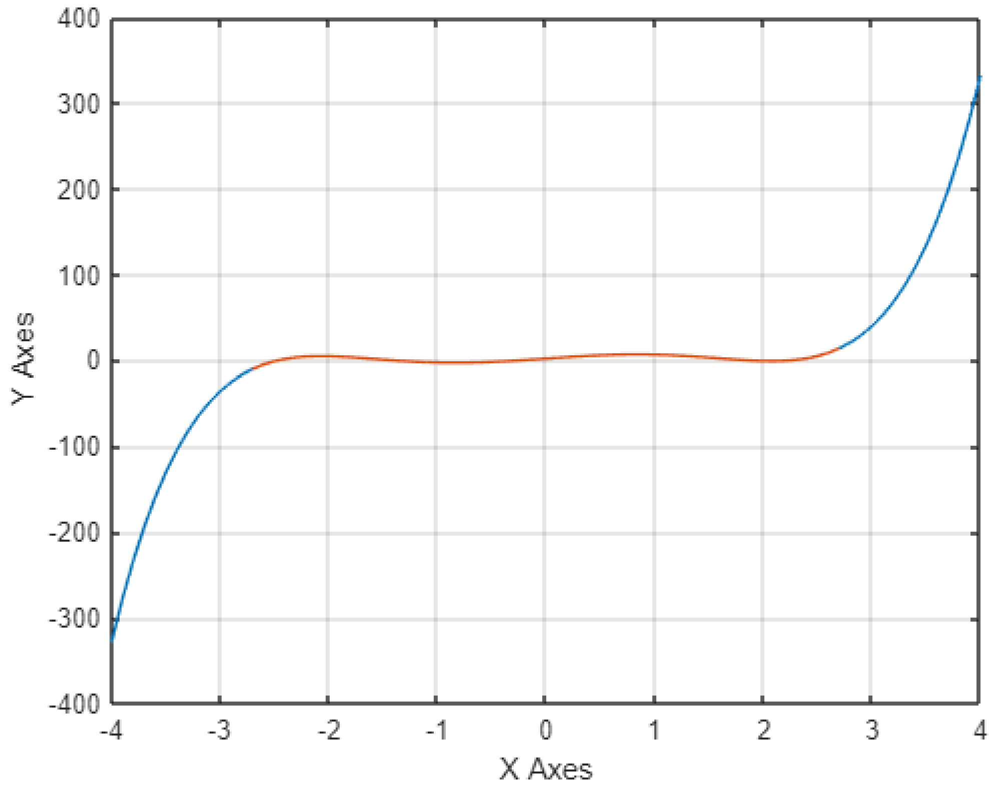
$$\frac{v}{V} = e^{-\frac{t}{\tau}}$$

```
time = 0:0.1:5;
tau = [0.5 1.0 2.0];
[TIME TAU] = meshgrid(time,tau);
V = exp(-TIME./TAU);
plot(time,V)
xlabel('time')
ylabel('tau')
title('RC time constants')
```



Question 4

```
x1 = -4:0.1:4;
x2 = -2.7:0.1:2.7;
f1 = 0.6*x1.^5 - 5*x1.^3+9*x1+2;
f2 = 0.6*x2.^5 - 5*x2.^3+9*x2+2;
plot(x1,f1,x2,f2)
grid on
xlabel('X Axes')
ylabel('Y Axes')
```

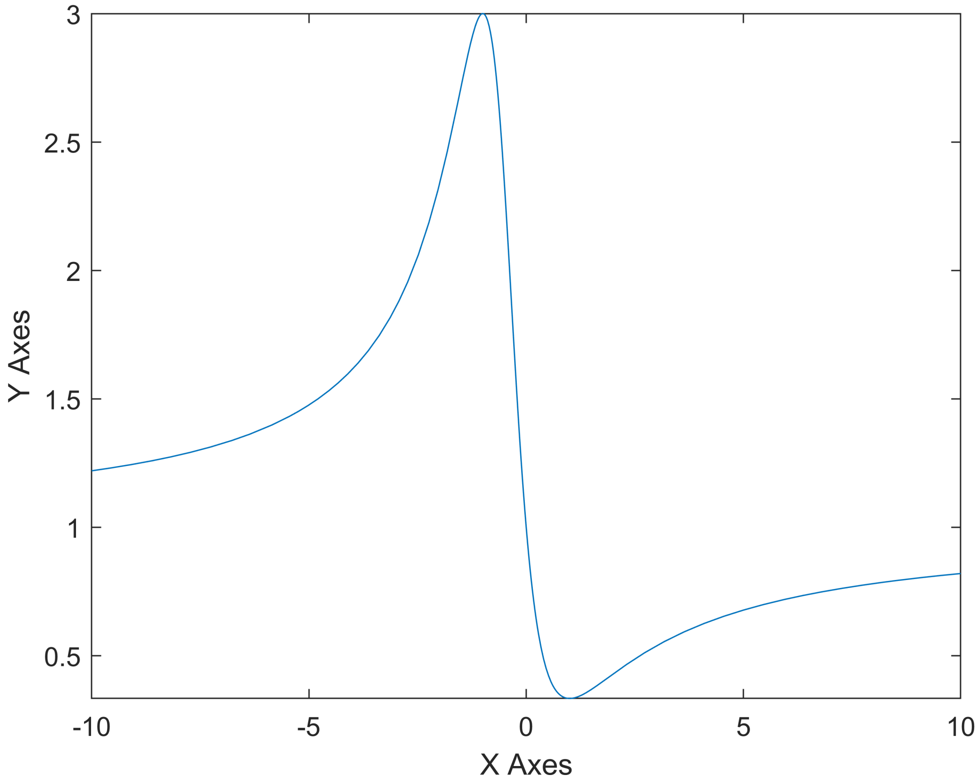


Question 5

```
f = @(x)(x^2 - x + 1)/(x^2 + x + 1);
l = [-10,10];
fplot(f,l)
```

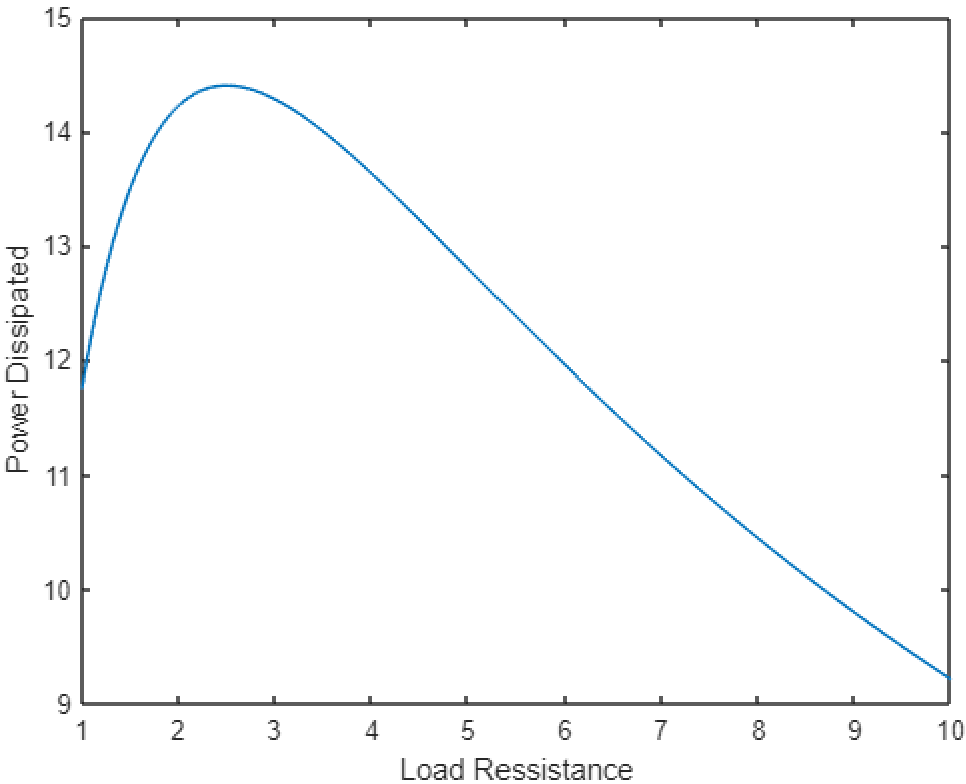
```
Warning: Function behaves unexpectedly on array inputs. To improve performance, properly vectorize your
function to return an output with the same size and shape as the input arguments.
```

```
xlabel('X Axes')
ylabel('Y Axes')
```



Question 6

```
RL = 1:0.01:10;
Vs = 12;
Rs = 2.5;
P = (Vs^2*RL)./(RL+Rs).^2;
plot(RL,P)
xlabel('Load Ressistance')
ylabel('Power Dissipated')
```


Power Dissipated
Load Ressistance
15
14
13
12
11
10
9
1
2
3
4
5
6
7
8
9
10