Data Visualization

Lecture 7 – Monday April 3, 2017

Outline

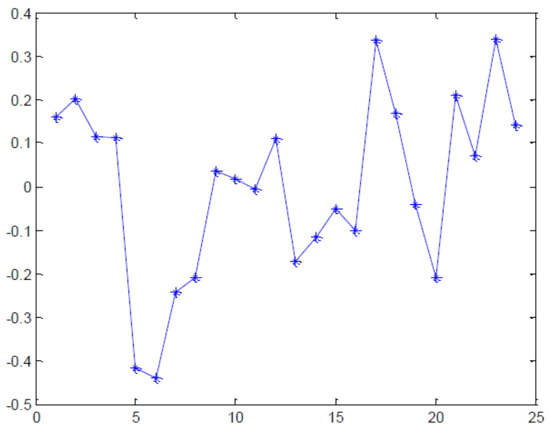
- How to visualize your data?
- 2D graphics
- 3D graphics
- Animation

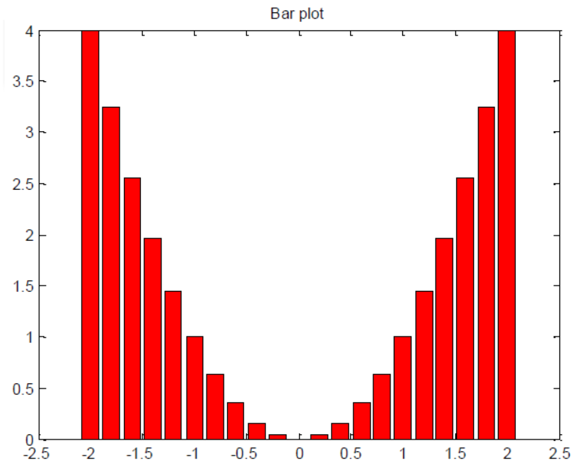
Outline

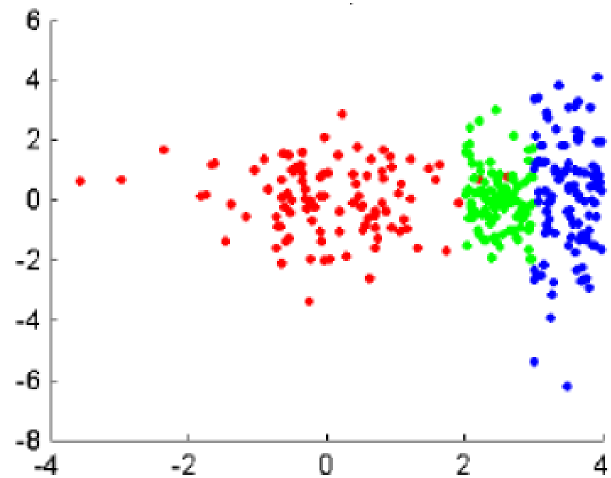
- **<u>How to visualize your data?</u>**
- 2D graphics
- 3D graphics
- Animation

How to Visualize your data

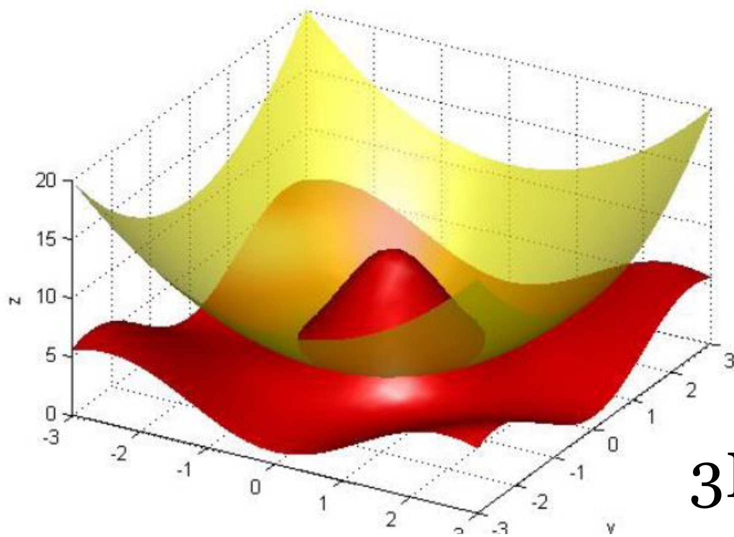
Matlab Graphics



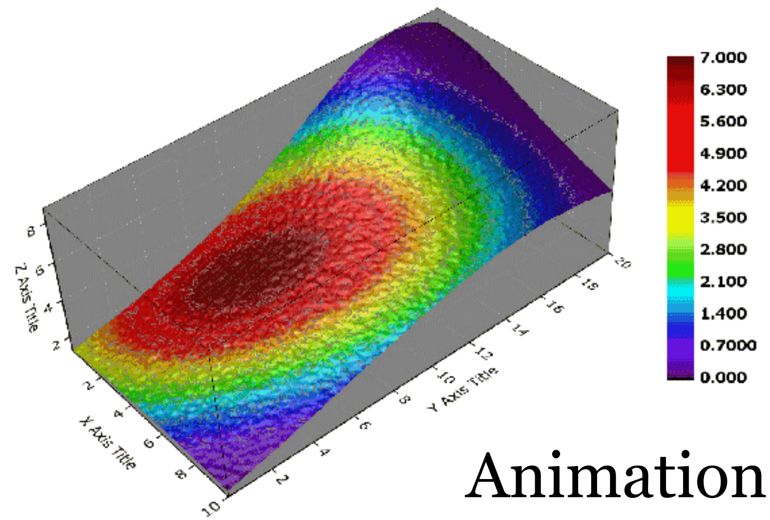




2D graphics



3D graphics

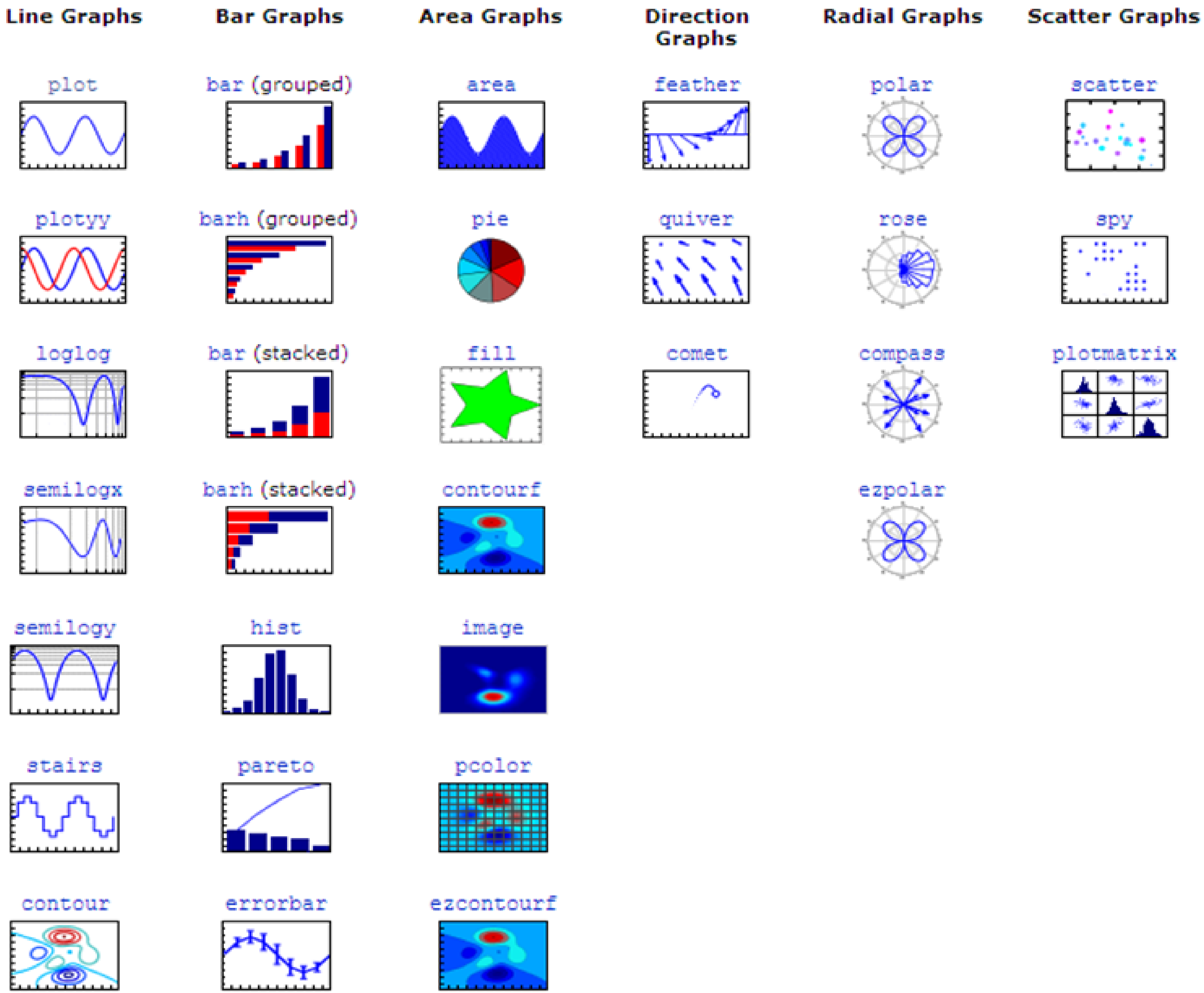


Animation

Outline

- How to visualize your data?
- **2D graphics**
- 3D graphics
- Animation

2D graphics



Line Graphs
Bar Graphs
Area Graphs
Direction Graphs
Radial Graphs
Scatter Graphs
plot
bar (grouped)
area
feather
polar
scatter
plotyy
barh (grouped)
pie
quiver
rose
spy
loglog
bar (stacked)
fill
comet
compass
plotmatrix
semilogx
barh (stacked)
contourf
ezpolar
semilogy
hist
image
stairs
pareto
pcolor
contour
errorbar
ezcontourf

2D graphics

- **Plot of dots**

  plot is the most basic function for creating 2D graphics.

```
plot(x1, y1, c1, x2, y2, c2, …)
```

x coordinate of first dot

y coordinate of first dot

Color & marker of first dot

Symbol	Color	Symbol	Marker	Symbol	Line style
b	blue	.	point	-	solid
g	green	o	circle	:	dotted
r	red	x	x-mark	-.	dashdot
c	cyan	+	plus	--	dashed
m	magenta	*	star	(none)	no line
y	yellow	s	square		
k	black	d	diamond		
		v	triangle (down)		
		^	triangle (up)		
		<	triangle (left)		
		>	triangle (right)		
		p	pentagram		
		h	hexagram		

2D graphics

- **Plot of dots: Example**

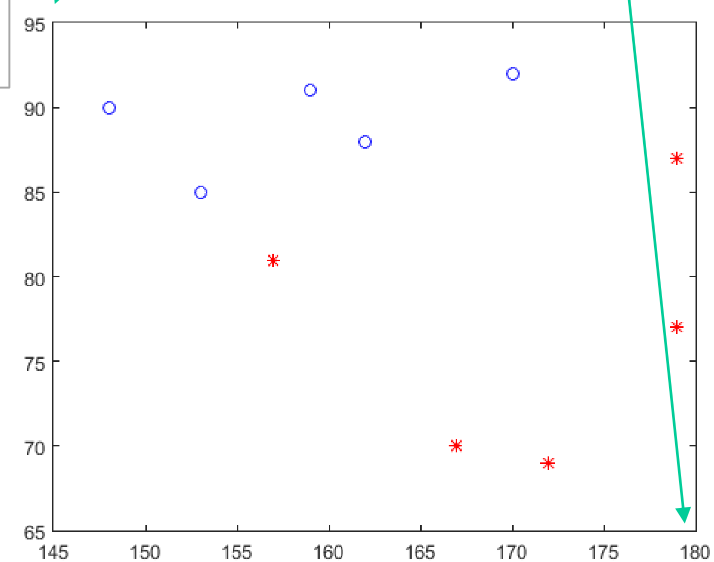
```
%Group #1
w_pre1 = [ 148 153 170 159 162]; %weight in previous month
w_cur1 = [ 90 85 92 91 88 ]; %weight in current month

%Group #2
w_pre2 = [157 172 179 167 179]; %weight in previous month
w_cur2 = [81 69 87 70 77 ]; %weight in current month

%Plotting the previous vs. current week weights of each contestant
plot(w_pre1(1), w_cur1 (1),'bo', w_pre1(2), w_cur1 (2), 'bo', ...
w_pre1(3), w_cur1 (3), 'bo', w_pre1(4), w_cur1 (4), 'bo', ...
w_pre1(5), w_cur1 (5), 'bo', ...
w_pre2(1), w_cur2 (1),'r*', w_pre2(2), w_cur2 (2), 'r*', ...
w_pre2(3), w_cur2 (3), 'r*', w_pre2(4), w_cur2 (4), 'r*', ...
w_pre2(5), w_cur2 (5), 'r*');

set(gcf,'color','w'); % set a white background for the plot
```

gcf – get handle of current figure

This is very labor intensive…

The same result can be achieved with much less work using vector notation

Notice that Matlab automatically chooses the axes borders that fit the plot…



2D graphics

- **Plot of dots using vectors**

```
plot(x, y, c)
```

A vector containing coordinates $x_1 \dots x_n$

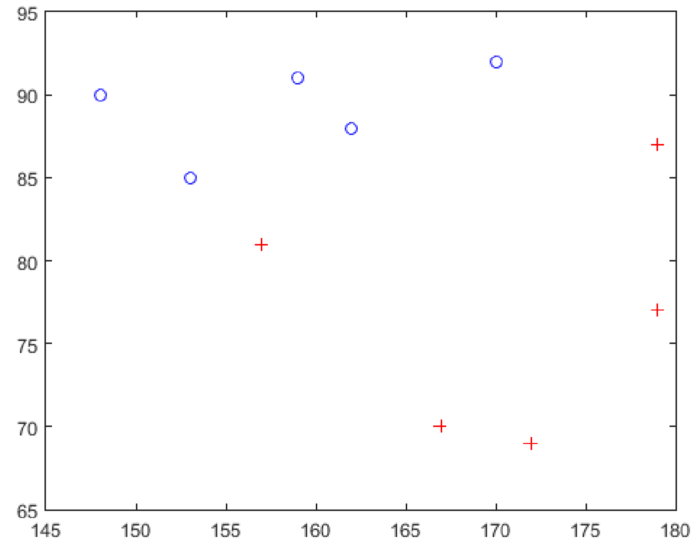
A vector containing coordinates $y_1 \dots y_n$

Color & marker of first dot

```
% using vector notation
plot(w_pre1, w_cur1, 'bo');
hold on

plot(w_pre2, w_cur2, 'r+');
hold off
```

Cancel *hold on*. The following plot will override current figure

From now on all other plots will be superimposed on the original figure



We get the exact same plot

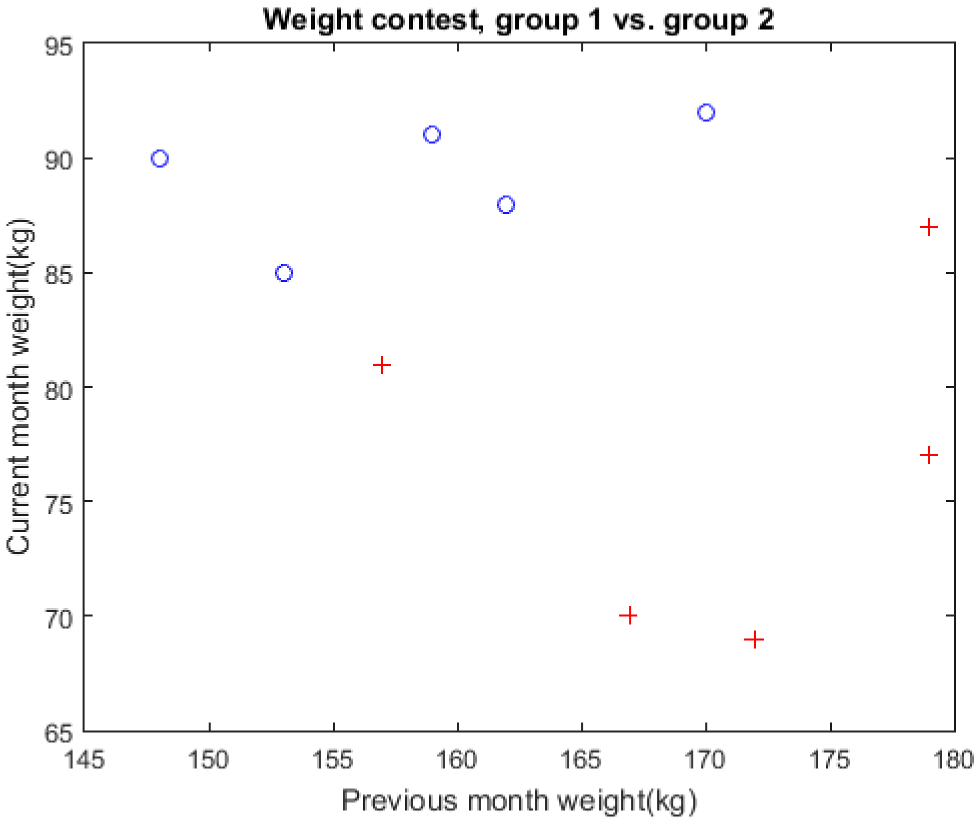
2D graphics

- **Plot (opening and closing)**
  - ◇ Notice that every time we plot a figure it **overrides** the previous figure (unless we use **hold on**)
  - ◇ If we want to open a new figure without erasing the previous one we use a command called **figure**
  - ◇ If we want to close all the figures we use the command **close all**

2D graphics

- **Adding labels and titles to the plot**

```
% Add labels and titles
xlabel('Previous month weight(kg)');
ylabel('Current month weight(kg)');
title('Weight contest, group 1 vs. group 2');
```

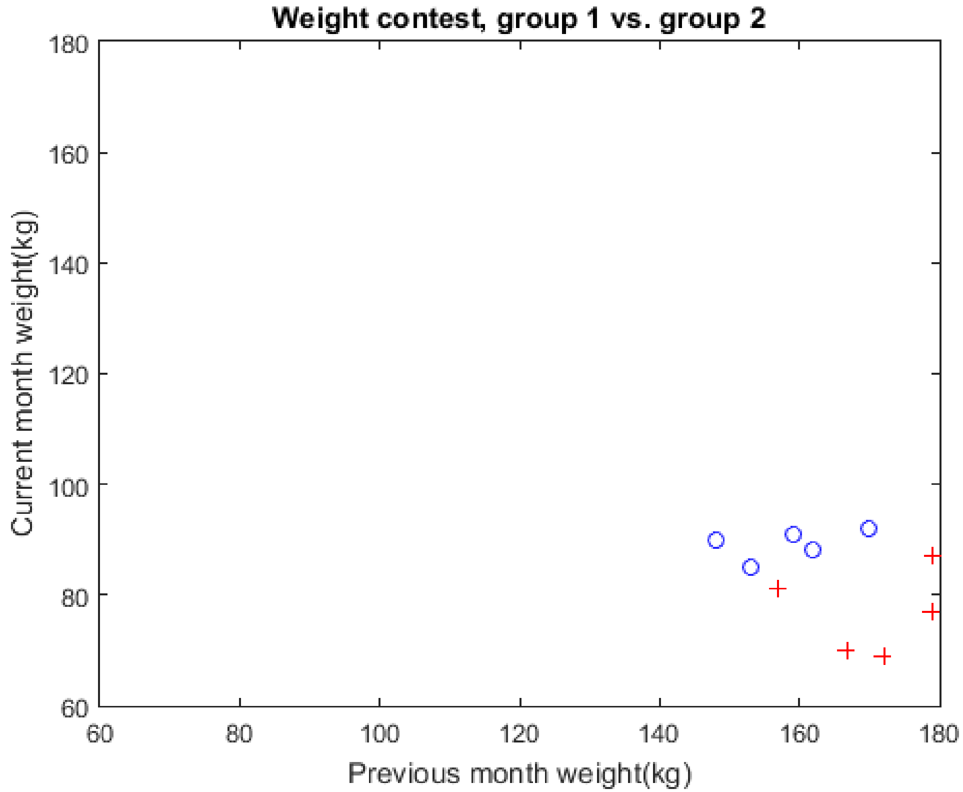


2D graphics

- **Plot (manipulating the axis)**

```
axis([60, 180, 60, 180])
```

x_{min} x_{max} y_{min} y_{max}



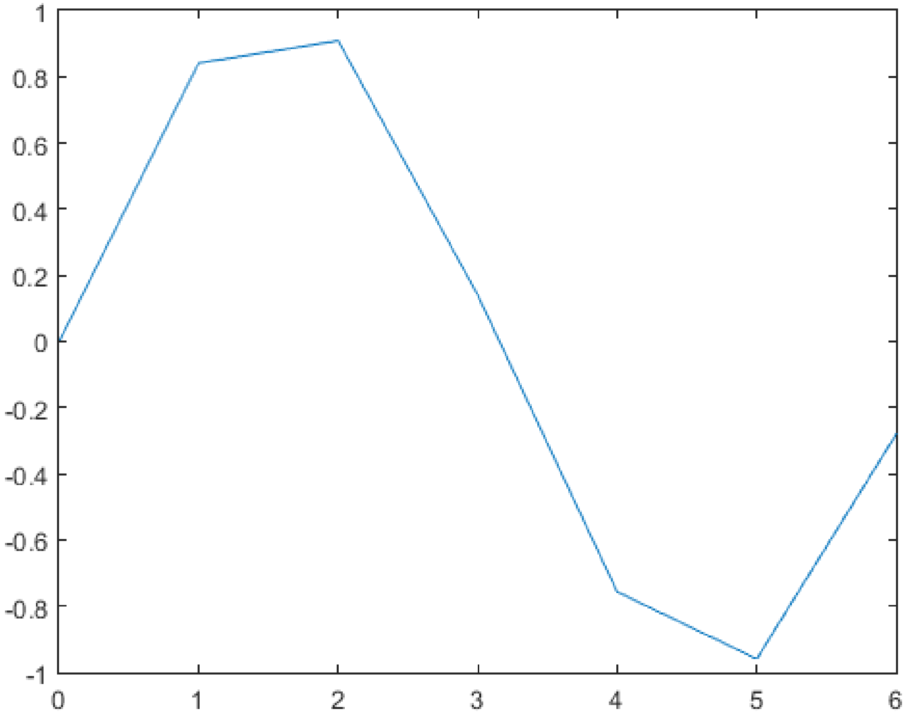
2D graphics

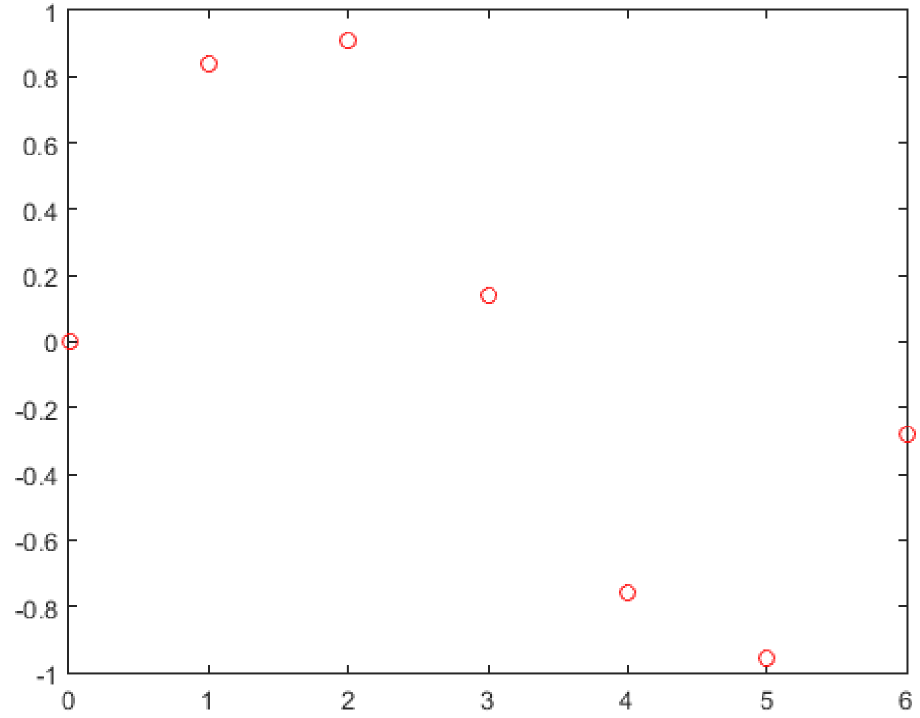
- **Plot: Example-2**

```
x = 0 : 2 * pi;
y = sin(x);
plot(x, y);

figure;
plot(x, y,'ro');

set(gcf,'color','w'); % set a white background for the plot
```

By default Matlab will connect the dots…

If we use: plot(x, y,'ro'), Matlab will display a dot plot



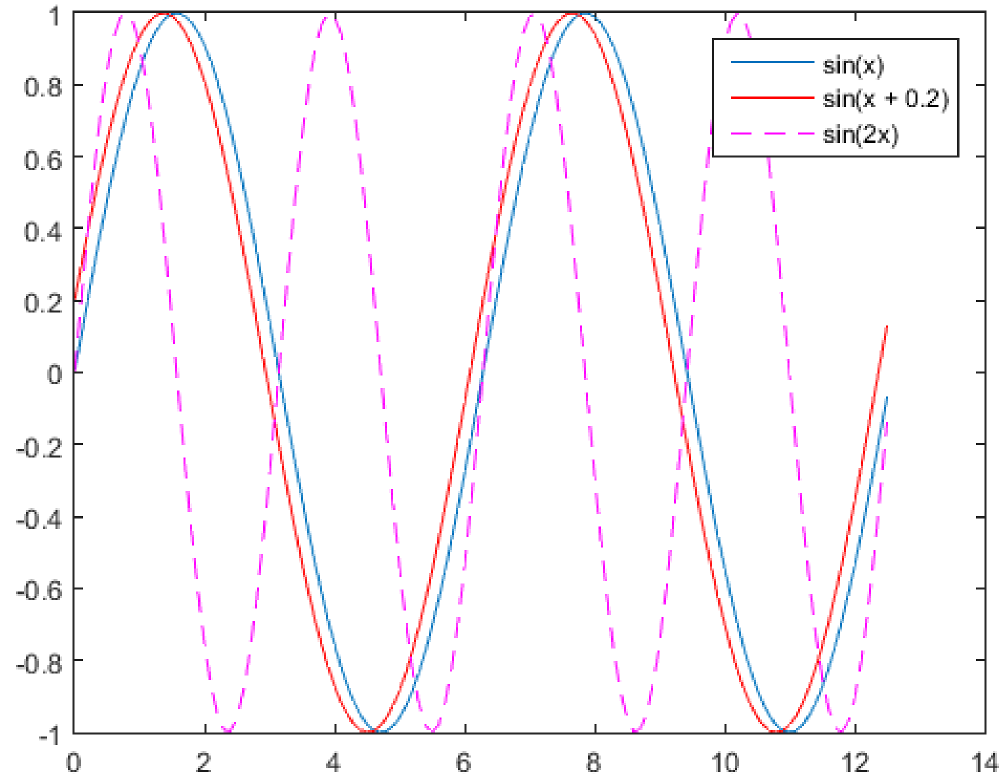


2D graphics

- **Adding legend**

```
x = 0 : 0.1 : 4*pi
y_sin1 = sin(x);
y_sin2 = sin(x + 0.2);
y_sin3 = sin(2 * x);
plot(x, y_sin1);
hold on
plot(x, y_sin2, 'r');
plot(x, y_sin3, 'm--');
legend('sin(x)', 'sin(x + 0.2)', 'sin(2x)');
hold off
```

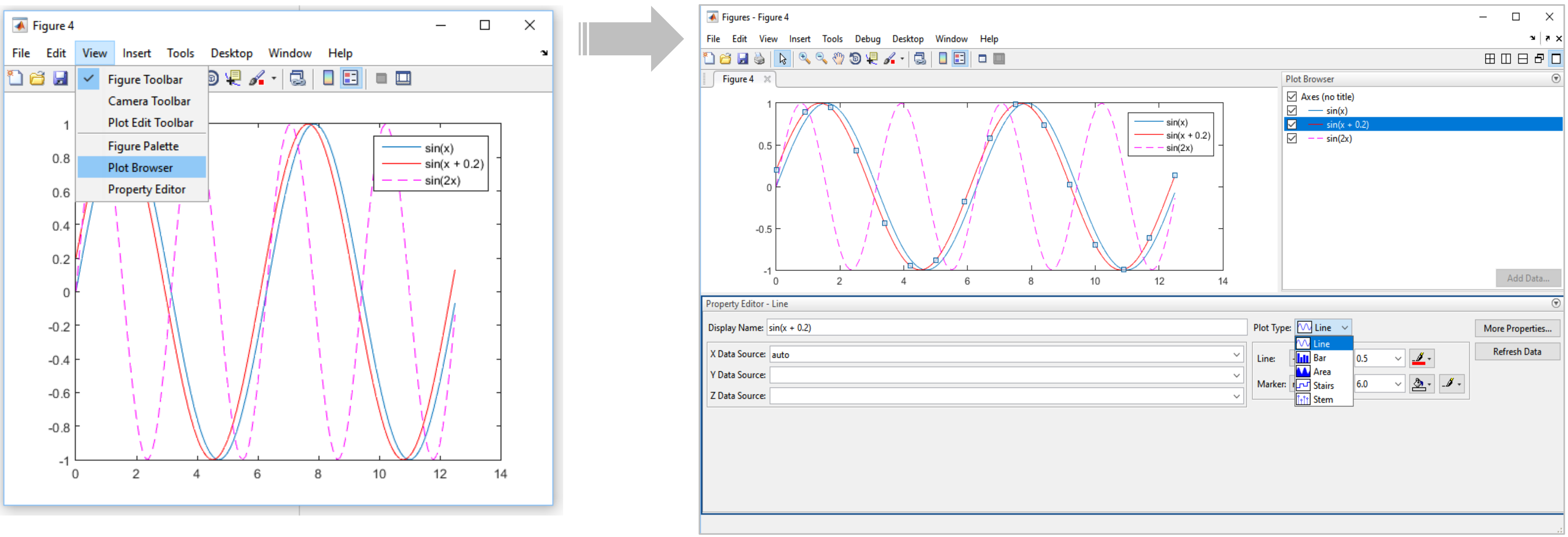
A figure legend can be added using the **legend** command



2D graphics

- **Plot browser**

You can make additional modifications to your plot using the plot browser.



2D graphics

- **Plotting Multiple Rows**

The variable **soil_prop** contains the a soil property values of 7 oil well locations in 6 different samples.

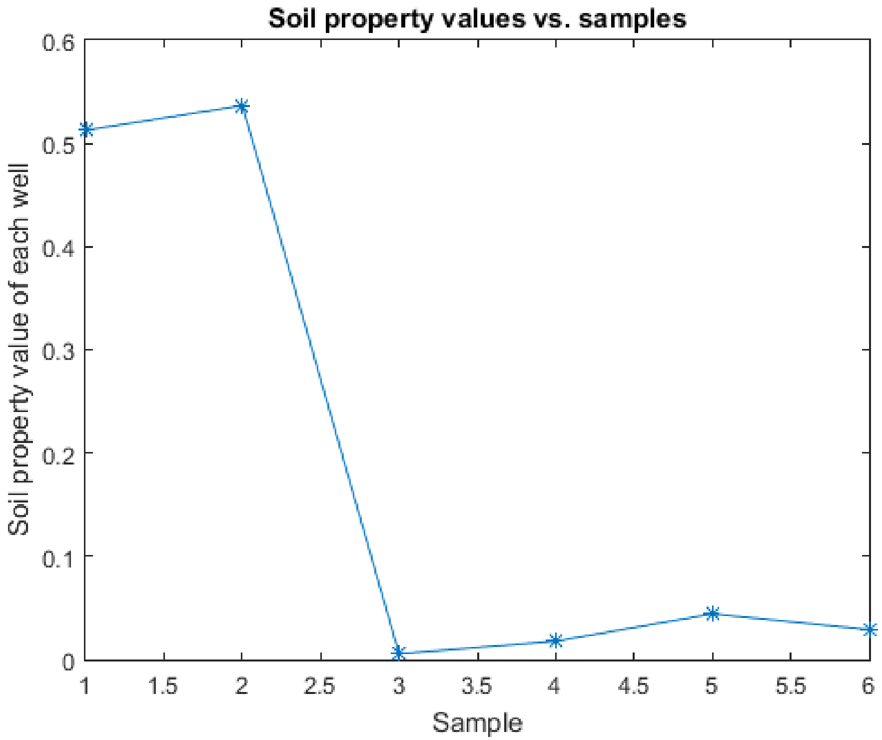
	S1	S2	S3	S4	S5	S6
Well1	0.3767	0.4701	0.0175	-0.0712	0.03	0.022
Well2	0.5128	0.5367	0.0056	0.0179	0.0443	0.0291
Well3	0.4303	0.4447	0.0326	0.0498	0.1646	0.049
Well4	0.4745	0.5575	0.1232	0.1444	0.0259	0.0187
Well5	0.2148	0.238	0.1591	0.1438	0.1826	0.1717
Well6	0.4852	0.4029	0.0542	0.1435	0.1424	0.0546
Well7	0.4258	0.3948	0.023	0.1261	0.0398	0.0199

2D graphics

- **Plotting Multiple Rows**

  Plot the expression of the first well

```
% % File: wells.m
% % Author: Alaa Khamis
% % Last modified on November 10, 2015, 11:47AM
% %
% % This script plots analyzes the soil property values of
% % different oil wells
% %
clc;
close all;

data_file='wellData.txt';

% % Reading data
disp(['-->Reading data from file: ',data_file]);
soil_vals=dataset('file',data_file);

% Plot the expression of the oil well
plot(soil_vals(1, :), '-*');
set(gcf,'color','w');
xlabel('Sample');
ylabel('Soil property value of each well');
title('Soil property values vs. samples');
```

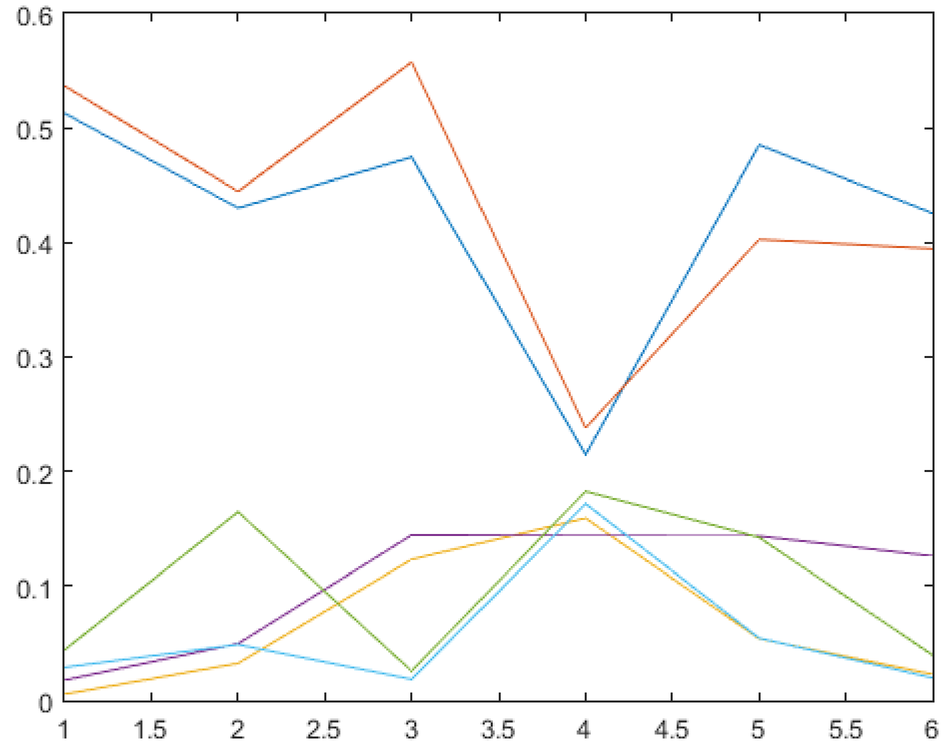


2D graphics

- **Plotting Multiple Rows**

  Plot the expression of all the oil wells

```
% Plot the expression of all the oil wells
figure;
plot(soil_vals);
set(gcf,'color','w');
```

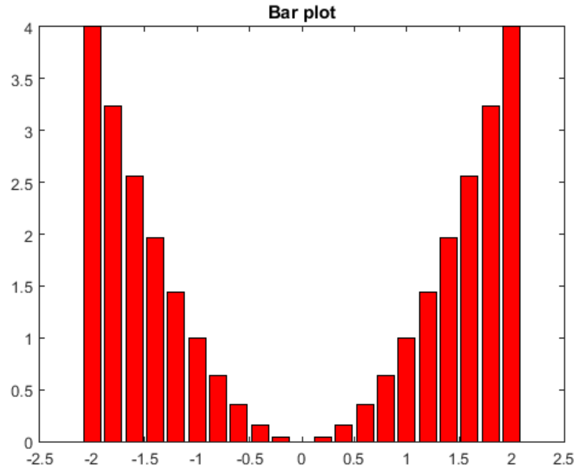


2D graphics

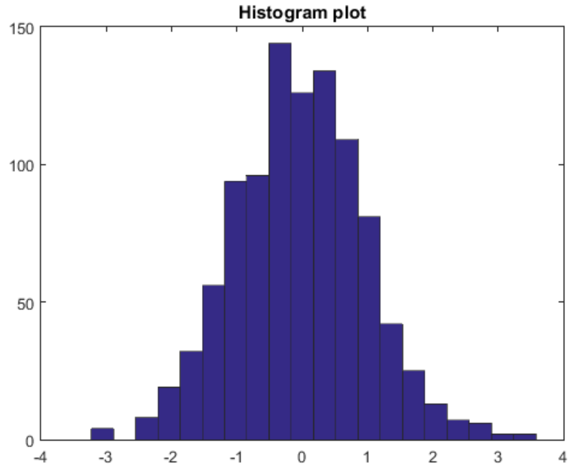
- **Plotting other types of graphs**

  Matlab has many other types of plotting capabilities

```
% Bar plot
x = -2 : 0.2 : 2;
y = x .* x;
bar(x, y, 'r');
title('Bar plot');
set(gcf,'color','w');
```



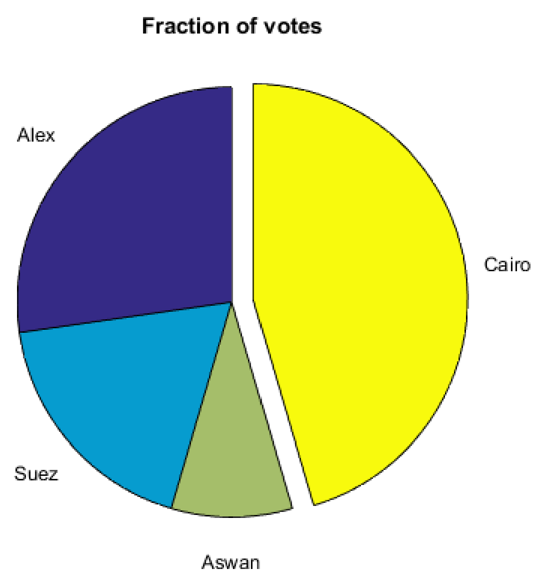

```
% Histogram
norm_rand_values = randn(1, 1000);
figure;
hist(norm_rand_values, 20);
title('Histogram plot');
set(gcf,'color','w');
```



2D graphics

- **Plotting other types of graphs**

```
 % pie chart
figure;
set(gcf,'color','w');
pie([3000 2000 1000 5000],[0 0 0 1], ...
{'Alex', 'Suez','Aswan','Cairo'});
title('Fraction of votes');
```



2D graphics

- **Plotting other types of graphs**

Making scatter plots

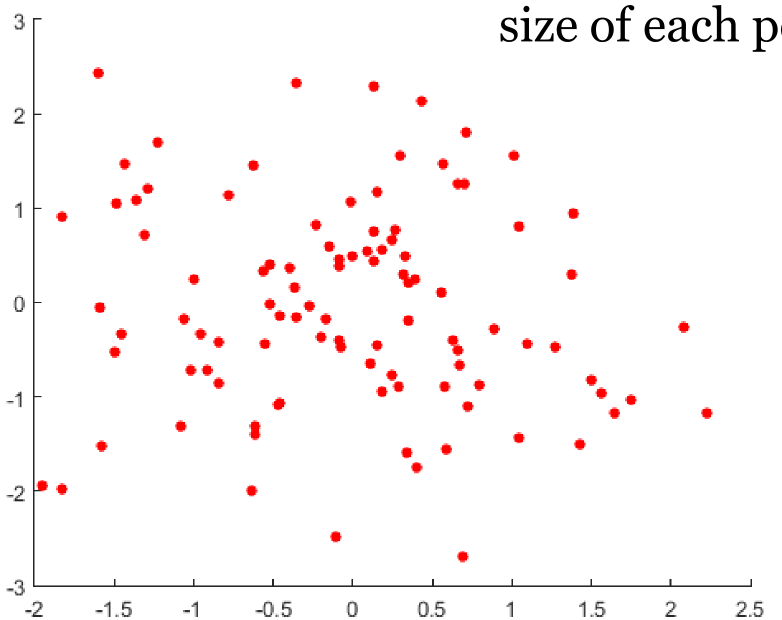
returns a 1-by-100 vector of uniformly distributed random numbers

```
x1 = randn(1, 100);
y1 = randn(1, 100);
scatter(x1, y1, 25, [1 0 0], 'filled');
```

size of each point

Color of each point

Fill the interior of each point



2D graphics

- **Plotting other types of graphs**

  Colors can be represented as a combination of Red Green Blue

```
scatter(x1, y1, 25, [1 0 0], 'filled');
```

↑ Color of each point

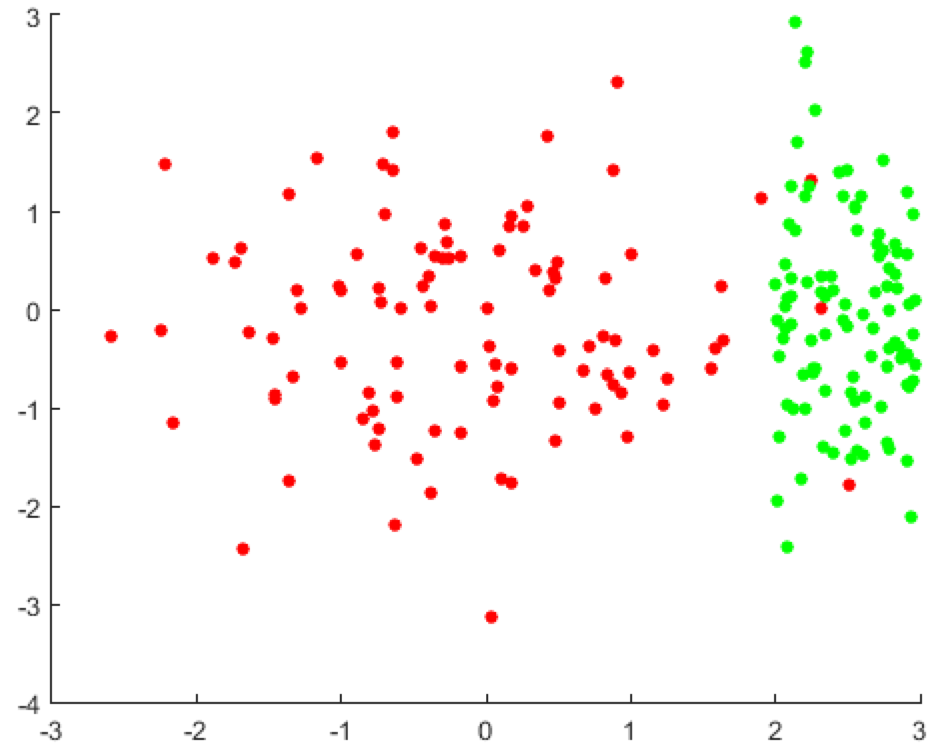
R	G	B	**Color**
1	0	0	Red
0	1	0	Green
0	0	1	Blue
0	0	0	Black
1	1	1	White
1	1	0	Yellow
1	0.6	0.4	Copper
...	...	...	

2D graphics

- **Plotting other types of graphs**

  Making scatter plots

```
% scatter plots
figure;
set(gcf,'color','w');
x1 = randn(1, 100);
y1 = randn(1, 100);
scatter(x1, y1, 25, [1 0 0], 'filled');

hold on
x2 = rand(1, 100) + 2;
y2 = randn(1, 100);
scatter(x2, y2, 25, [0 1 0] , 'filled');
```

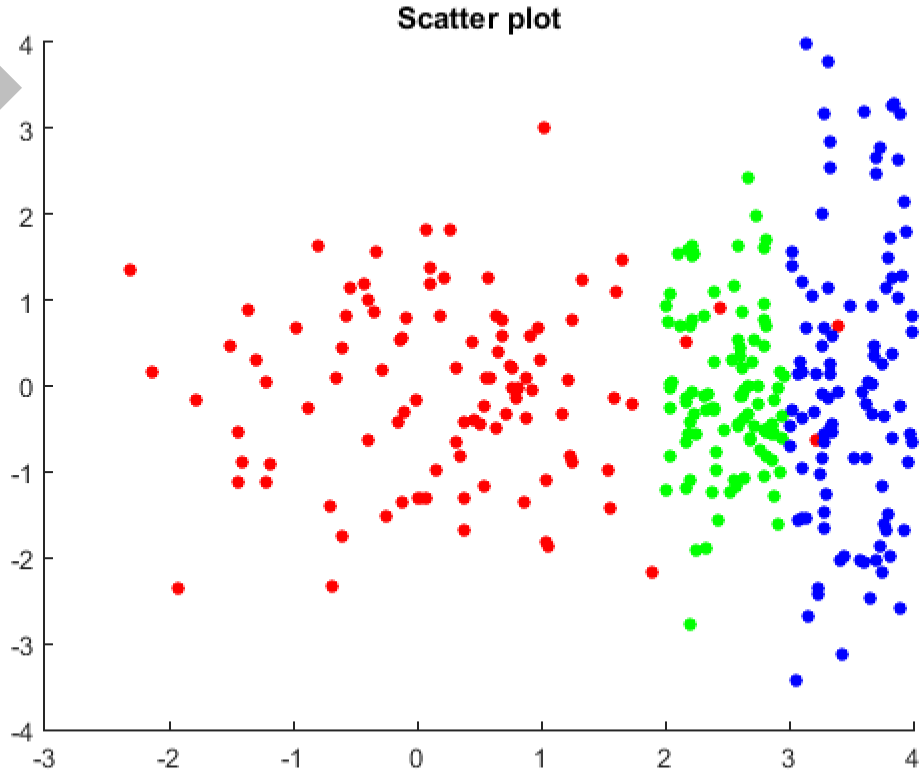


2D graphics

- **Plotting other types of graphs**

  Making scatter plots

```
% scatter plots
figure;
set(gcf,'color','w');
x1 = randn(1, 100);
y1 = randn(1, 100);
scatter(x1, y1, 25, [1 0 0], 'filled');

hold on
x2 = rand(1, 100) + 2;
y2 = randn(1, 100);
scatter(x2, y2, 25, [0 1 0] , 'filled');

x3 = rand(1, 100) + 3;
y3 = randn(1, 100) * 2;
scatter(x3, y3, 25, [0 0 1], 'filled');
title('Scatter plot');
hold off
```

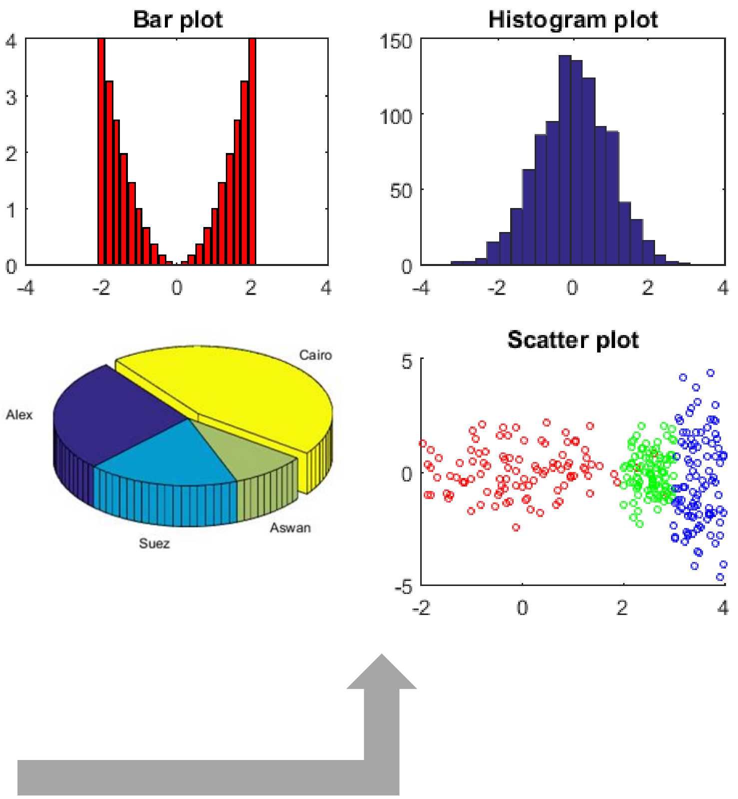


2D graphics

- **Putting multiple plots in the same figure**

```
subplot(# rows, # columns, current plot position)
```

```
figure;
subplot(2, 2, 1)
x = -2 : 0.2 : 2;
y = x .*x;
bar(x,y, 'r');
title('Bar plot')
subplot(2, 2, 2);
norm_rand_values = randn(1, 1000);
hist(norm_rand_values, 20);
title('Histogram plot');
subplot(2, 2, 3);
pie3([3 2 1 5],[0 0 0 1],{'Bibi','Barak','Liberman','Donald Duck'})
title('Pie plot');
subplot(2, 2, 4);
x1 = randn(1, 100)
x2 = rand(1, 100) + 2
x3 = rand(1, 100) + 3
y1 = randn(1, 100);
y2 = randn(1, 100);
y3 = randn(1, 100) * 2;
z = [repmat([1 0 0], 100, 1); ...
repmat([0 1 0], 100, 1); repmat([0 0 1], 100, 1)];
scatter([x1 x2 x3], [y1 y2 y3], 10, z);
title('Scatter plot');
set(gcf,'color','w');
```



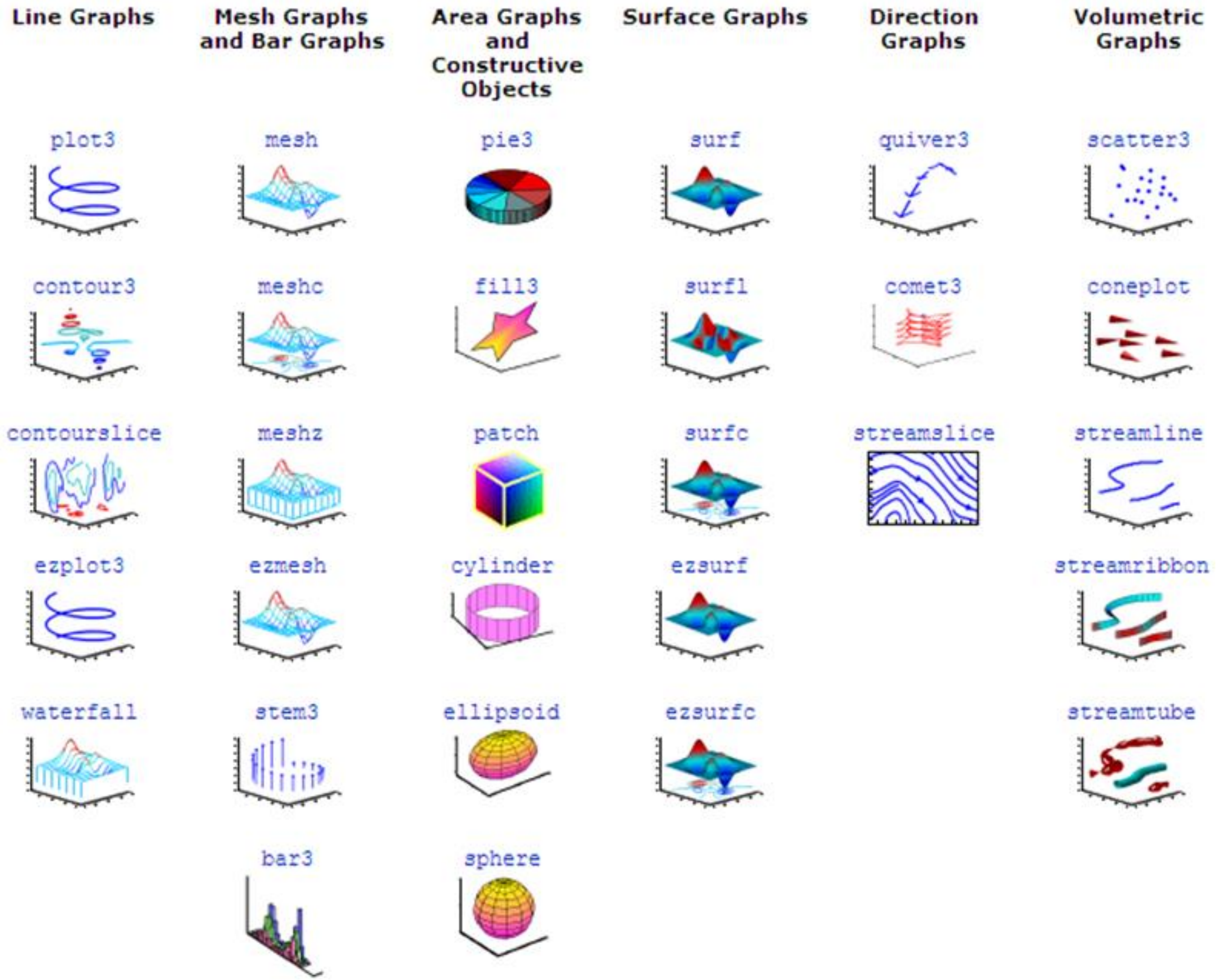
repmat =replicate matrix

z is a [300x3] matrix for indicating color.

Outline

- How to visualize your data?
- 2D graphics
- **3D graphics**
- Animation

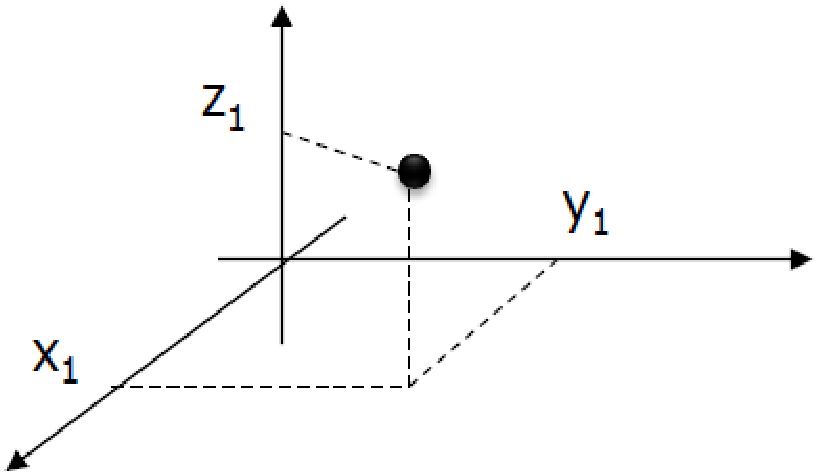
3D graphics


Line Graphs
Mesh Graphs and Bar Graphs
Area Graphs and Constructive Objects
Surface Graphs
Direction Graphs
Volumetric Graphs
plot3
mesh
pie3
surf
quiver3
scatter3
contour3
meshc
fill3
surfl
comet3
coneplot
contourslice
meshz
patch
surfc
streamslice
streamline
ezplot3
ezmesh
cylinder
ezsurf
streamribbon
waterfall
stem3
ellipsoid
ezsurfc
streamtube
bar3
sphere

3D graphics

A 3D surface is defined as:

$$z = f(x, y)$$



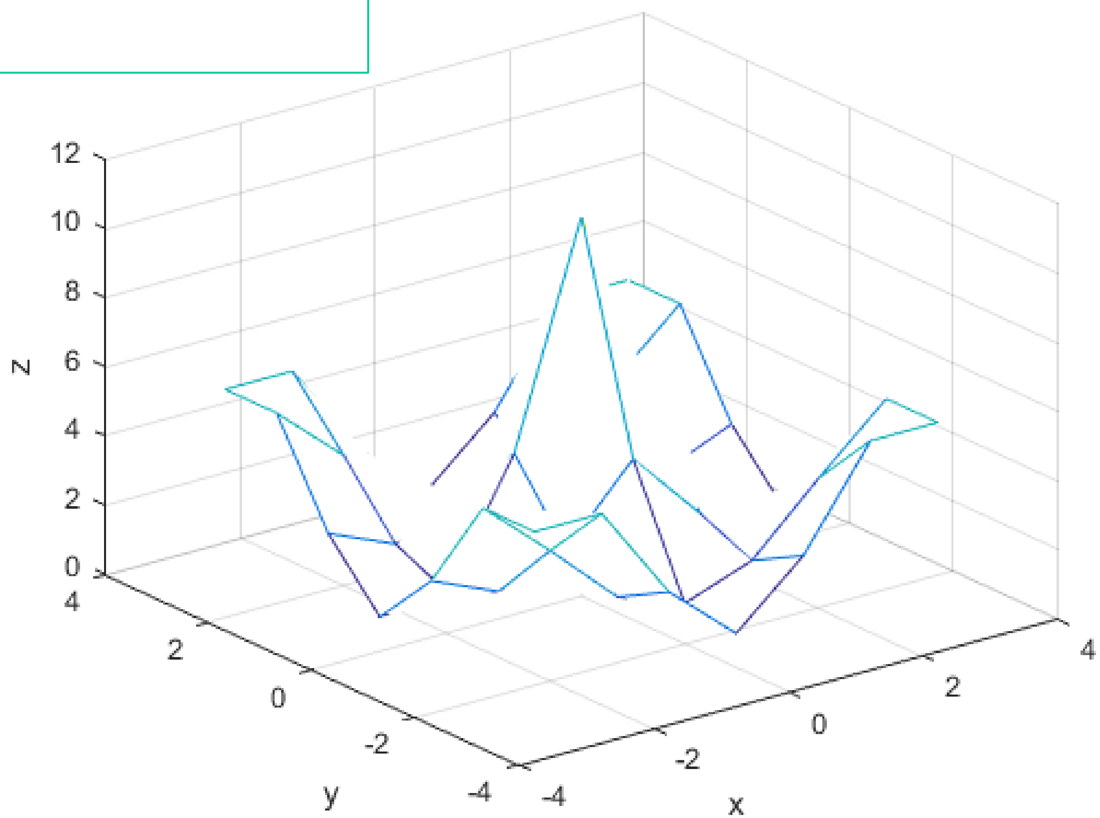
We can create 3D surfaces using 2 functions:

- `mesh(x, y, z);`
- `surf(x, y, z);`

3D graphics

- **Mesh Plot**

```
% mesh plot
xx = -3 : 1 : 3
yy = -3 : 1 : 3
[x, y] = meshgrid(xx, yy)
z = 5 * sin(pi / 15 * x .* y).^2 + 10 * exp( -(x.^2 + y.^2)) + 1
figure;
mesh(x, y, z);
xlabel('x'); ylabel('y'); zlabel('z');
set(gcf,'color','w');
```



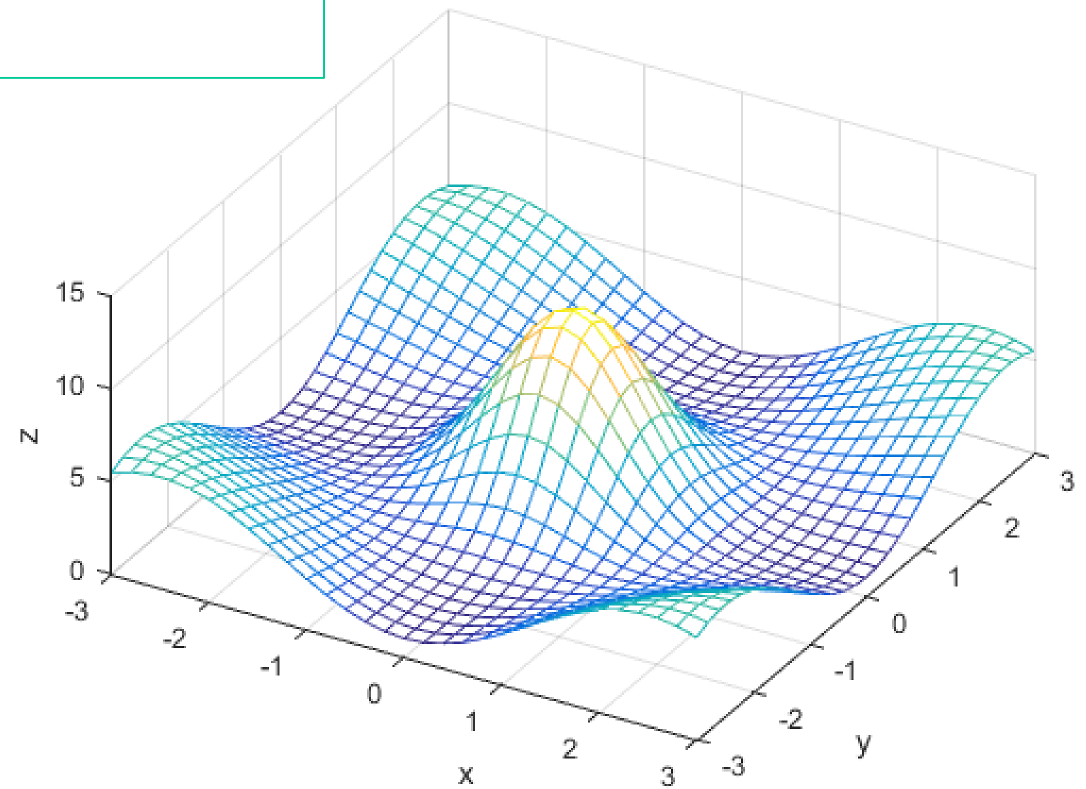
3D graphics

- **Mesh plot with finer grid**

```
% mesh with finer grid
xx = -3 : 0.2 : 3;
yy = -3 : 0.2 : 3;
[x, y] = meshgrid(xx, yy);
z = 5 * sin(pi / 15 * x .* y).^2 + 10 * exp( -(x.^2 + y.^2)) + 1;
figure;
mesh(x, y, z);
xlabel('x'); ylabel('y'); zlabel('z');
set(gcf,'color','w');
view(30, 50);
```

view([az, el]) sets the angle of the view from which an observer sees the current 3-D plot:

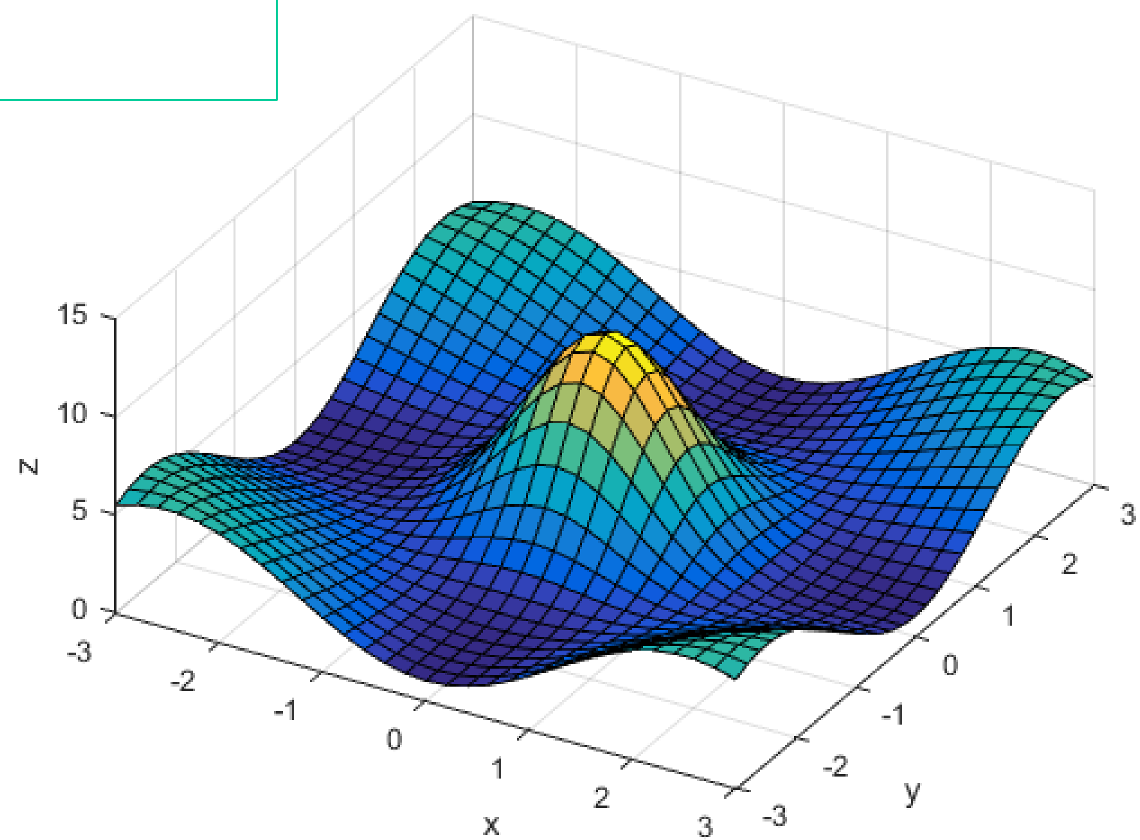
- az is the azimuth or horizontal rotation (degrees)
- el is the vertical elevation (degrees).



3D graphics

- **Surf plot**

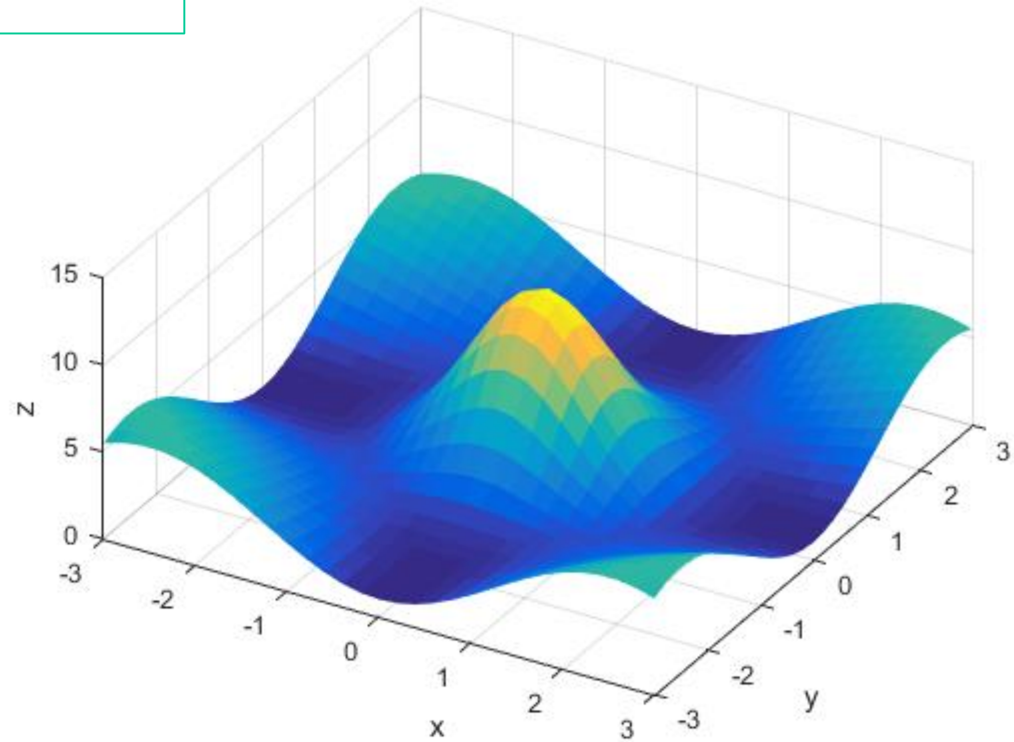
```
% surf plot
figure;
surf(x, y, z);
xlabel('x'); ylabel('y'); zlabel('z');
set(gcf,'color','w');
view(30, 50);
```



3D graphics

- **Surf plot: omitting the edges of the surface**

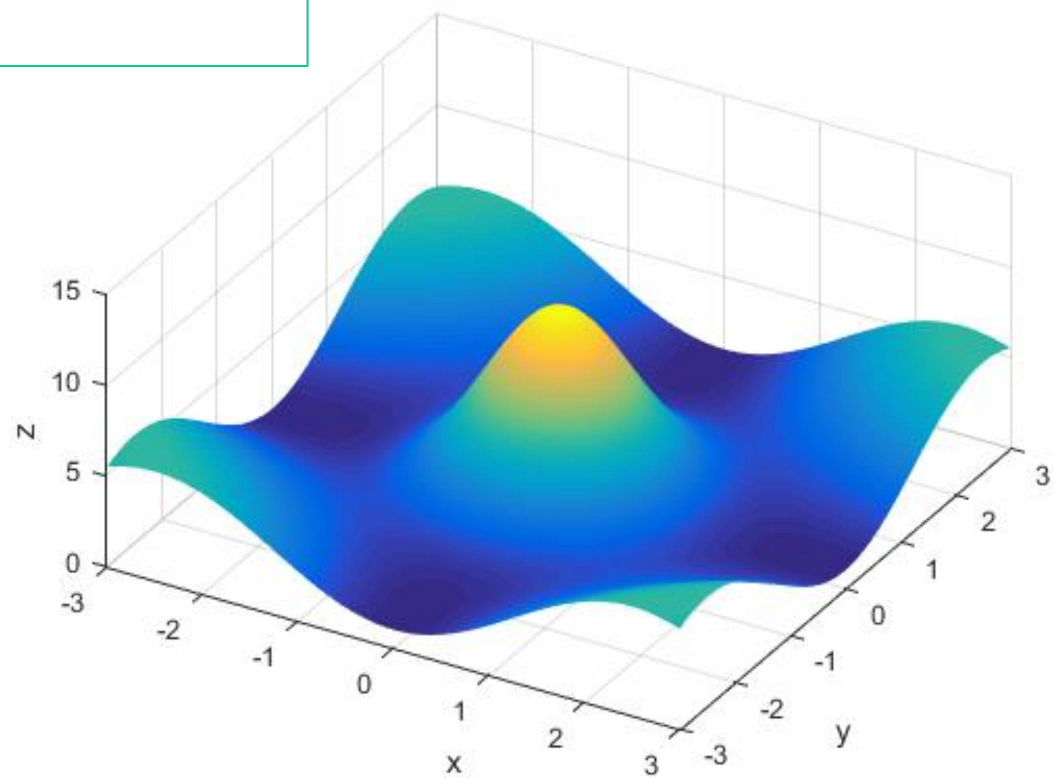
```
% Omitting the edges of the surface
figure;
surf(x, y, z, 'EdgeColor', 'none');
xlabel('x'); ylabel('y'); zlabel('z');
set(gcf,'color','w');
view(30, 50);
```



3D graphics

- **Surf plot: making the grid even finer**

```
% Making the grid even finer
xx = -3 : 0.01 : 3;
yy = -3 : 0.01 : 3;
[x, y] = meshgrid(xx, yy);
z = 5 * sin(pi / 15 * x .* y).^2 + 10 * exp( -(x.^2 + y.^2)) + 1;
surf(x, y, z, 'EdgeColor', 'none');
xlabel('x'); ylabel('y'); zlabel('z');
set(gcf,'color','w');
view(30, 50);
```

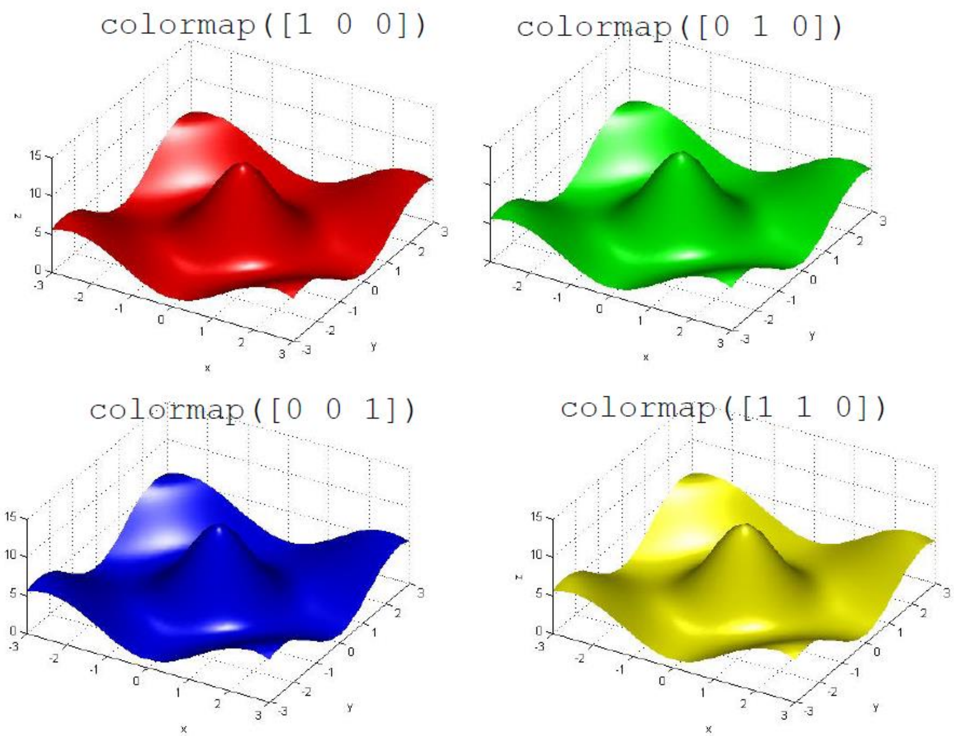


3D graphics

- **Surf plot: playing with the colors**

Colors can be represented as a combination of Red Green Blue

R	G	B	Color
1	0	0	Red
0	1	0	Green
0	0	1	Blue
0	0	0	Black
1	1	1	White
1	1	0	Yellow
1	0.6	0.4	Copper
…	…	…	



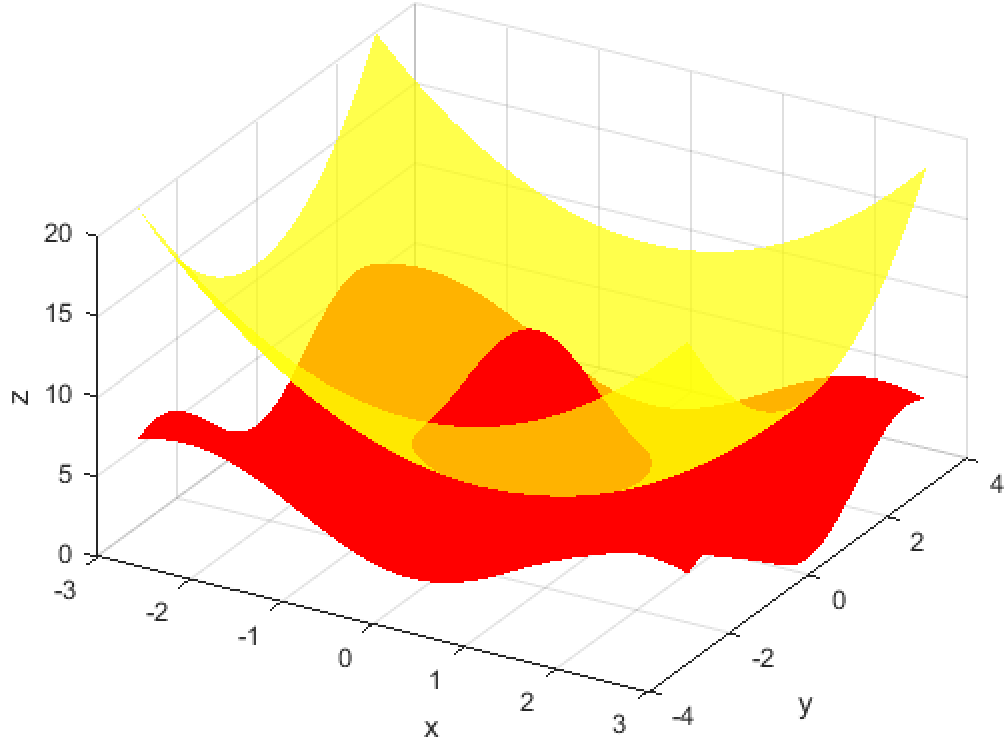
3D graphics

- **Surf plot: show several surfaces on the same plot**

```
% show several surfaces on the same plot
surf(x, y, z, 'EdgeColor', 'none', 'FaceColor', 'red');
xlabel('x'); ylabel('y'); zlabel('z');
hold on;
z2 = x.^2 + y.^2 + 2;
surf(x, y, z2, 'EdgeColor', 'none','FaceColor', 'yellow', 'FaceAlpha', 0.7);
set(gcf,'color','w');
view(30, 40);
```

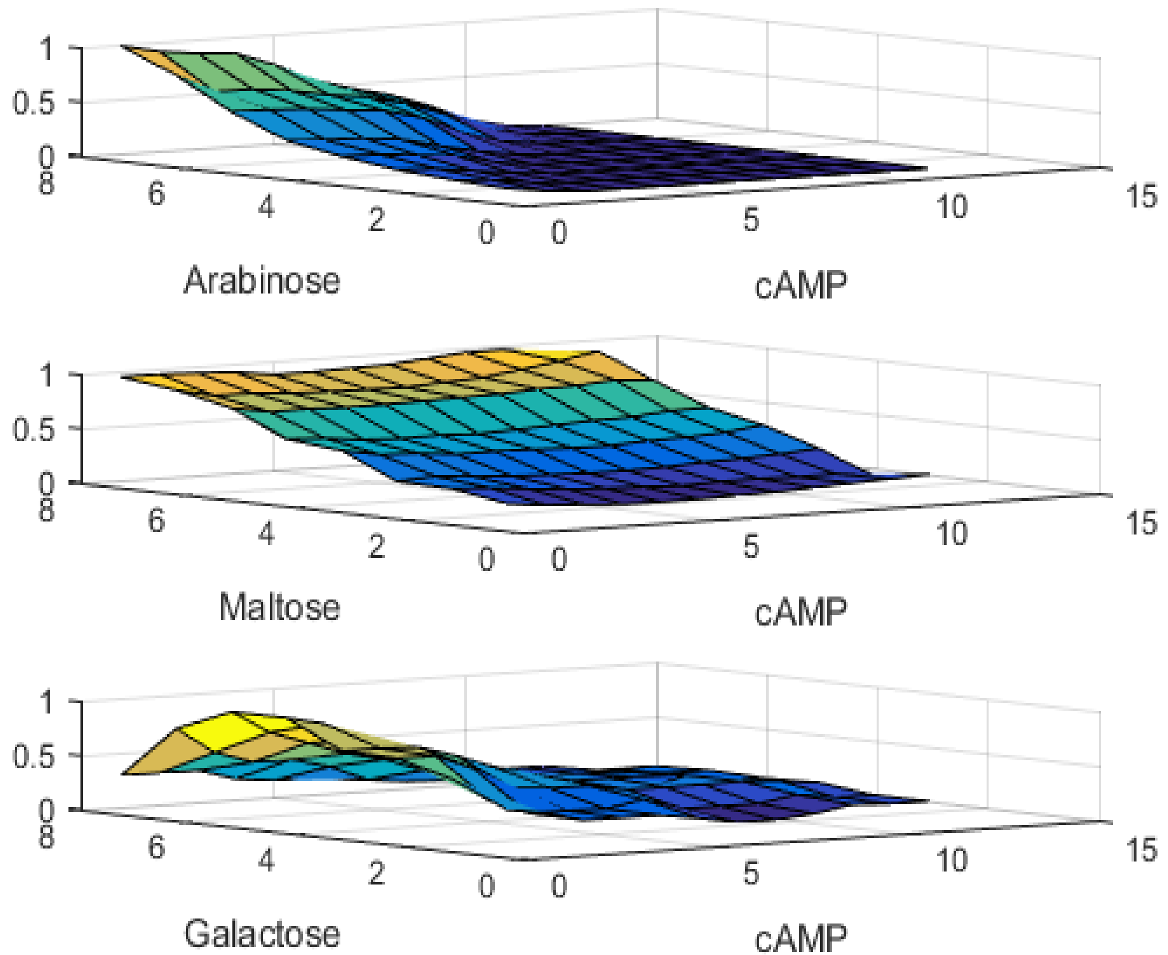
1-opaque

0-transparent



3D graphics

Using 3D graphics to visualize your experimental data



See example2_3d

Outline

- How to visualize your data?
- 2D graphics
- 3D graphics
- **Animation**

Animation

getframe	Capture axes or figure as movie frame
movie	Play recorded movie frames
frame2im	Return image data associated with movie frame
im2frame	Convert image to movie frame

Animation

```
figure
Z = peaks; % returns a 49-by-49 matrix of Gaussian distribution.
set(gcf,'color','w');

for j = 1:40
    X = sin(j*pi/10)*Z;
    surf(X,Z)
    F(j) = getframe;
end
```