Digital Signal Processing

Lab2: MatLab Graphs and scripts

Instructor: Eng\ Samar Shaaban
E-mail: ssa10@fayoum.edu.eg

Github Repo: https://github.com/SamarShabanCS/DSP

Slack workspace: https://fayoum-university-fci.slack.com

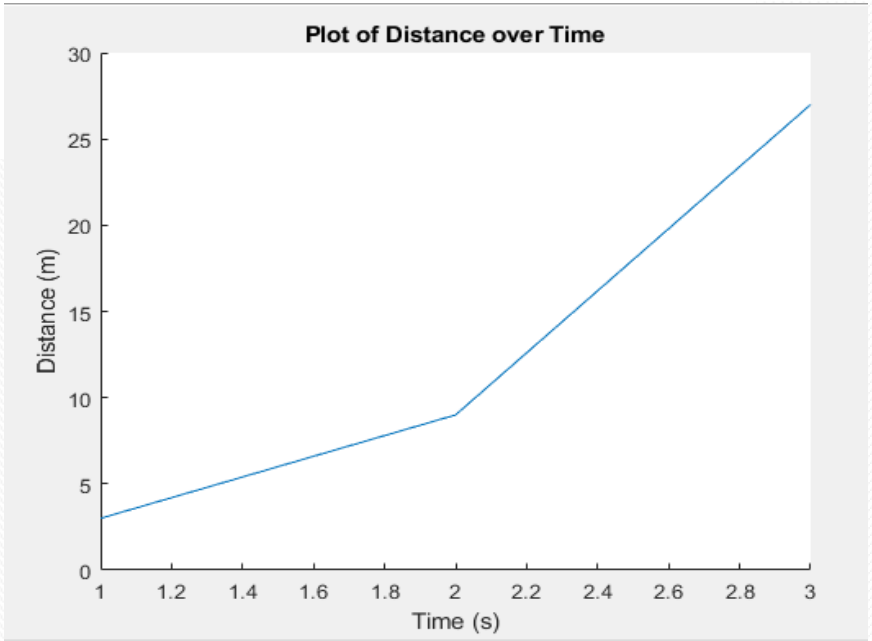
MATLAB Graphics

- Main MATLAB functions for plotting graphs
- For more information help graph2d

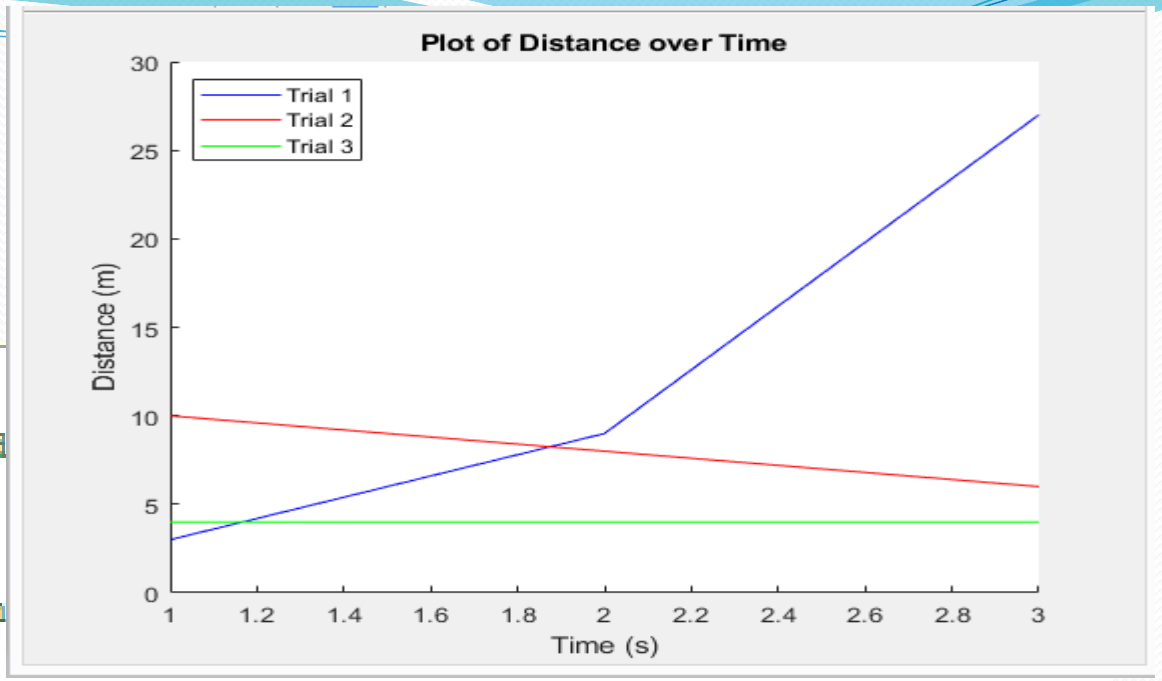
Function	Meaning
`plot (x1, y1, x2, y2,…)`	Linear graphics
`stem`	Sequence graphs
`stairs`	Stairs graphs
`loglog`	Both Logarithmic axis Im and Re
`semilogx` `semiloxy`	Logarithmic Re axis Logarithmic Im axis

example

```
>> X = [3 9 27]; % my dependent vector of interest
>> t = [1 2 3]; % my independent vector
>> figure % create new figure
>> hold on
>> title('Plot of Distance over Time') % title
>> xlabel('Time (s)') % label for x axis
>> ylabel('Distance (m)') %
>> plot(t, X)
```



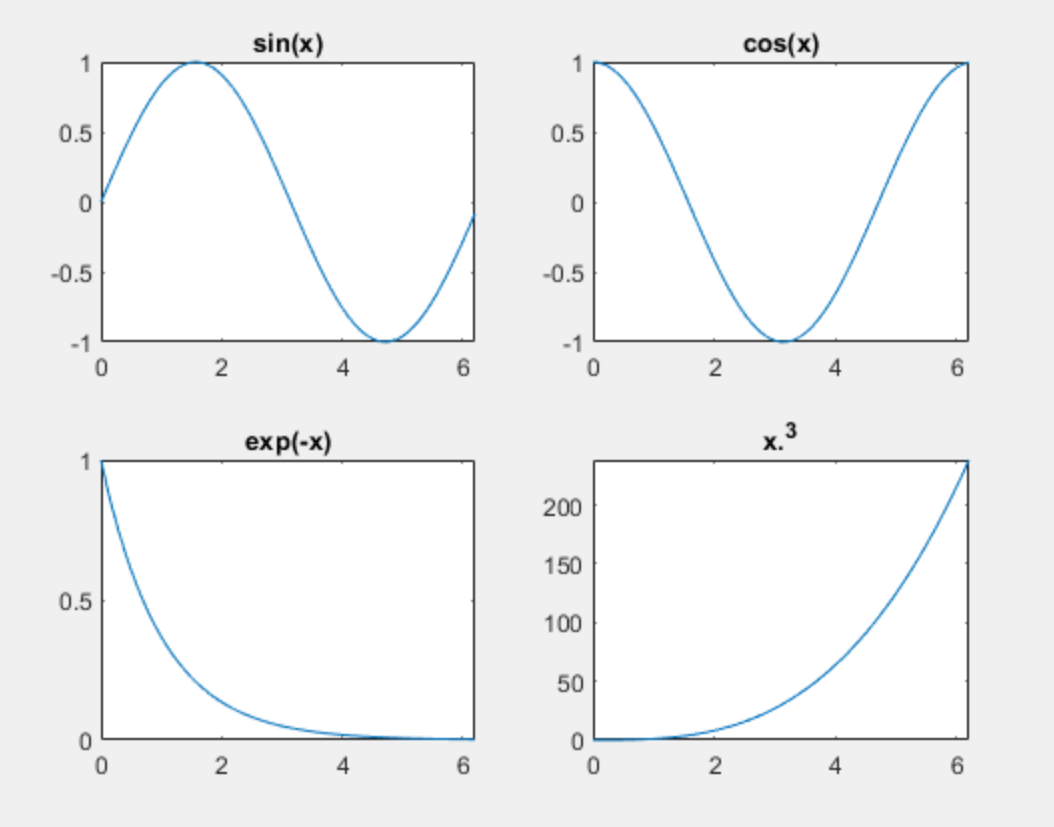
Legends




```
>>
X = [3 9 27]; % depend
Y = [10 8 6];
Z = [4 4 4];
t = [1 2 3]; % indepen
figure
hold on % allow all vectors to be plotted in same figure
plot(t, X, 'blue', t, Y,'red', t, Z,'green')
title('Plot of Distance over Time') % title
ylabel('Distance (m)') % label for y axis
xlabel('Time (s)') % label for x axis
legend('Trial 1','Trial 2','Trial 3')
legend('Location','NorthWest') % move legend to upper left
>>
```

Subplots

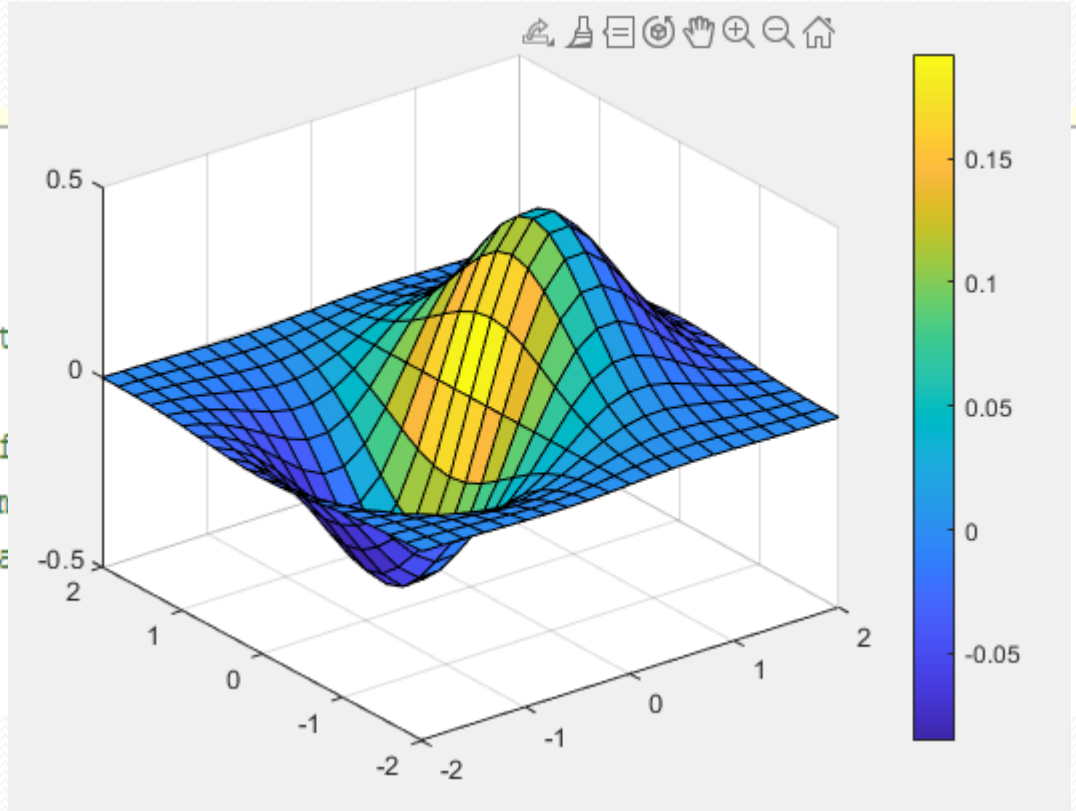
```
>> clear all
close all
% subplot (nrows,ncols,plot number)
x=0:.1:2*pi; % x vector
subplot(2,2,1); % plot
plot(x,sin(x));
title('sin(x)')
subplot(2,2,2); % plot
plot(x,cos(x));
title('cos(x)')
subplot(2,2,3) % plot ne
plot(x,exp(-x));
title('exp(-x)')
subplot(2,2,4); % plot
plot(x, x.^3);
title('x.^3')
>>
```



Plotting in 3-D

- There are also ways to plot in multiple dimensions in Matlab*. One type of 3D plot that may be useful is a surface plot, which requires you to generate some kind of x-y plane and then apply a 3rd function as the z dimension.

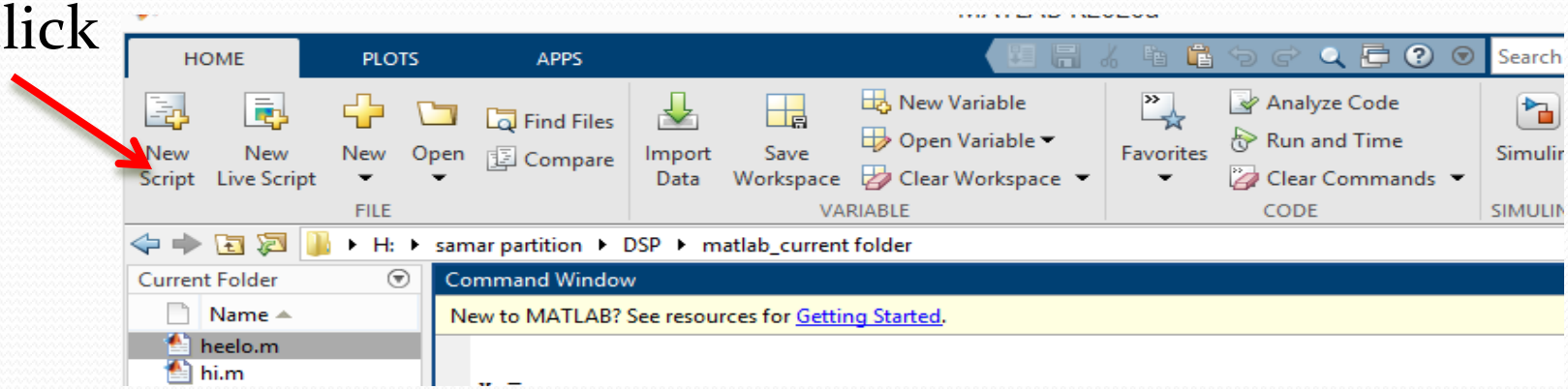
```
>> clear all
close all
[x,y] = meshgrid([-2:.2:2]); % set
Z = x.*exp(-x.^2-y.^2);      % plot
figure
surf(x,y,Z,gradient(Z))      % surf
                           % determ
colorbar                   % displa
>>
```



Programming Considerations

Scripts

- Scripts are
  - o collection of commands executed in sequence
  - o written in the MATLAB editor
  - o saved as MATLAB files (.m extension)
- To create an MATLAB file from command-line
  - »edit helloWorld.m.
  - or click



- All variables created and modified in a script exist in the workspace even after it has stopped running
- Just type the script name on the comand window without extension .m to run

Task

- Generate random vector to represent the salaries of 10 employees that in range of 700-900 L.E. Then present some statistic about these employees salaries :
  - Max. Salary
  - Empl. Max_ID
  - Min. Salary
  - Empl. Min_ID

Task: Sol

```
clear;
clc;
close all;
Salaries =randi([700,900],1,10);
MaxSalary = max(Salaries);                    % Max. Salary
EmplMax_ID = find(Salaries==MaxSalary);       %Empl. Max_ID
MinSalary = min(Salaries);                    %Min. Salary
EmplMin_ID = find(Salaries==MinSalary);       %Empl. Min_ID
```

String

- Any variable defined as string is considered a vector of characters, dealing with it as same as dealing with vectors.

```
>> str = 'hello matlab';
>> disp(str)
>> msgbox(str)
>> Num = input('Enter your number:')
>> str = input('Enter your name:','s')
---------------------------------------------------------------------
>> str = '7234'
>> Num = str2num(str)
>> number = 55
>> str = num2str(number)
```

Matlab programming

Relational Operators:

- *relational operators*
  - equal ==
  - notequal ~=
  - greater than >
  - less than <
  - greater or equal >=
  - less or equal <=

- *Logical operators*
  - And &&
  - Or ||
  - Not~
  - Xor xor

- *Boolean values*: zero is false, nonzero is true

Matlab programming

If / else / elseif :

- Basic flow-control, common to all languages
- No need for parentheses : command blocks are between reserved words

IF

```
if cond
    commands
end
```

Conditional statement: evaluates to true or false

ELSE

```
if cond
    commands1
else
    commands2
end
```

ELSEIF

```
if cond1
    commands1
elseif cond2
    commands2
else
    commands3
end
```

Matlab programming

Examples

cond.m*

```
1
2 -     num=input('enter your number');
3 -     if rem(num,2)==0
4 -         msgbox('your number is even')
5 -     else
6 -         msgbox('your number is odd')
7 -     end
```

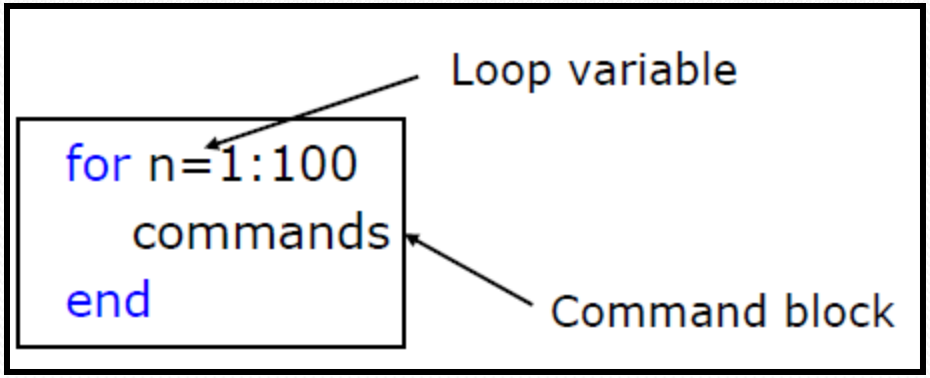
cond.m

```
1 -     if y < 0
2 -      M = y + 3;
3 -     elseif y > 5
4 -      M = y - 3;
5 -     else
6 -      M = 0;
7 -     end
8 -     M
9
```

Matlab programming

For loop

- The command block is anything between the for line and the end




```
for n = 1:32
        r(n) = n;
end
r
```

- **Nested For Loop**

```
for m = 1: 5
        for n = 1: 7
                A(m,n) = 1/(m+n-1);
        end
end
```

Matlab programming

While loop:

- The while is like a more general for loop
- Don't need to know number of iterations
- The command block will execute while the conditional expression is true
  - Beware of infinite loops!

WHILE

```
while cond
   commands
end
```

Editor - whileloop.m

cond.m | forloop.m | whileloop.m

```
1 -    x = 1;
2 -    while (x^2<50)
3 -      y=x^2;
4 -      plot(x,y,'or');
5 -      hold on
6 -      x = x+1;
7 -    end
```

Matlab programming

Continue:

- The continue statement passes control to the next iteration of the loop
- Skipping any remaining statements in the body of the loop.
- In nested loops, continue passes control to the next iteration of the loop enclosing it.
- **Example**

```
x=1;
for m=1:5

   if (m==3)
      continue;
   end
   x=m+x;
end
x
```

Matlab programming

Break:

- The break statement lets you exit early from a for loop or while loop.
- In nested loops, break exits from the innermost loop only.
- **Example:**

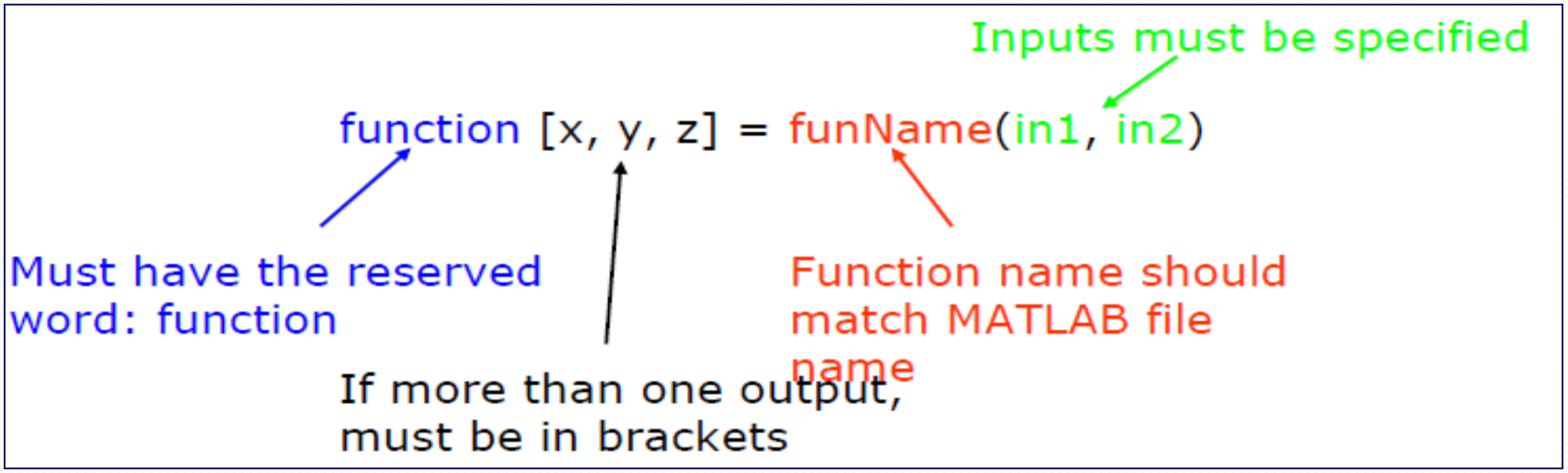
```
x=1;
for m=1:5

    if (m==3)
        break;
    end
    x=m+x;
end
x
```

Matlab programming

User-defined Functions:

- Functions look exactly like scripts, but for ONE difference Functions must have a function declaration.



- No need for return : MATLAB 'returns' the variables whose names match those in the function declaration.

Matlab programming

User-defined Functions: