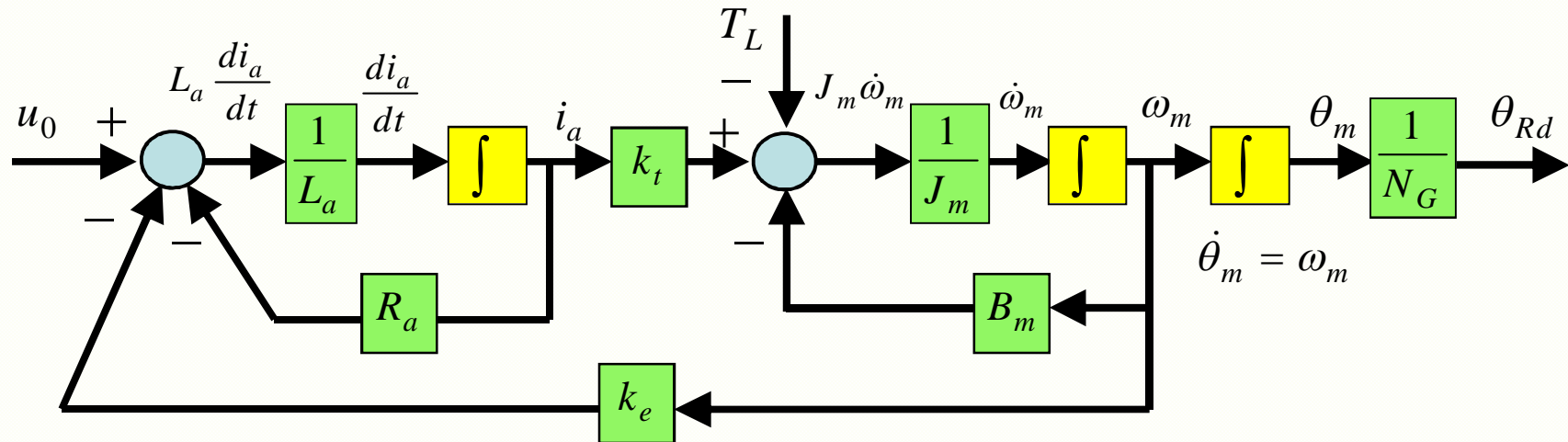


Engineering Softwares



Prof. Dr. İsmail H. Altaş

Karadeniz Teknik Üniversitesi
Elektrik-Elektronik Mühendisliği
61080 Trabzon

www.altas.org



Engineering Softwares

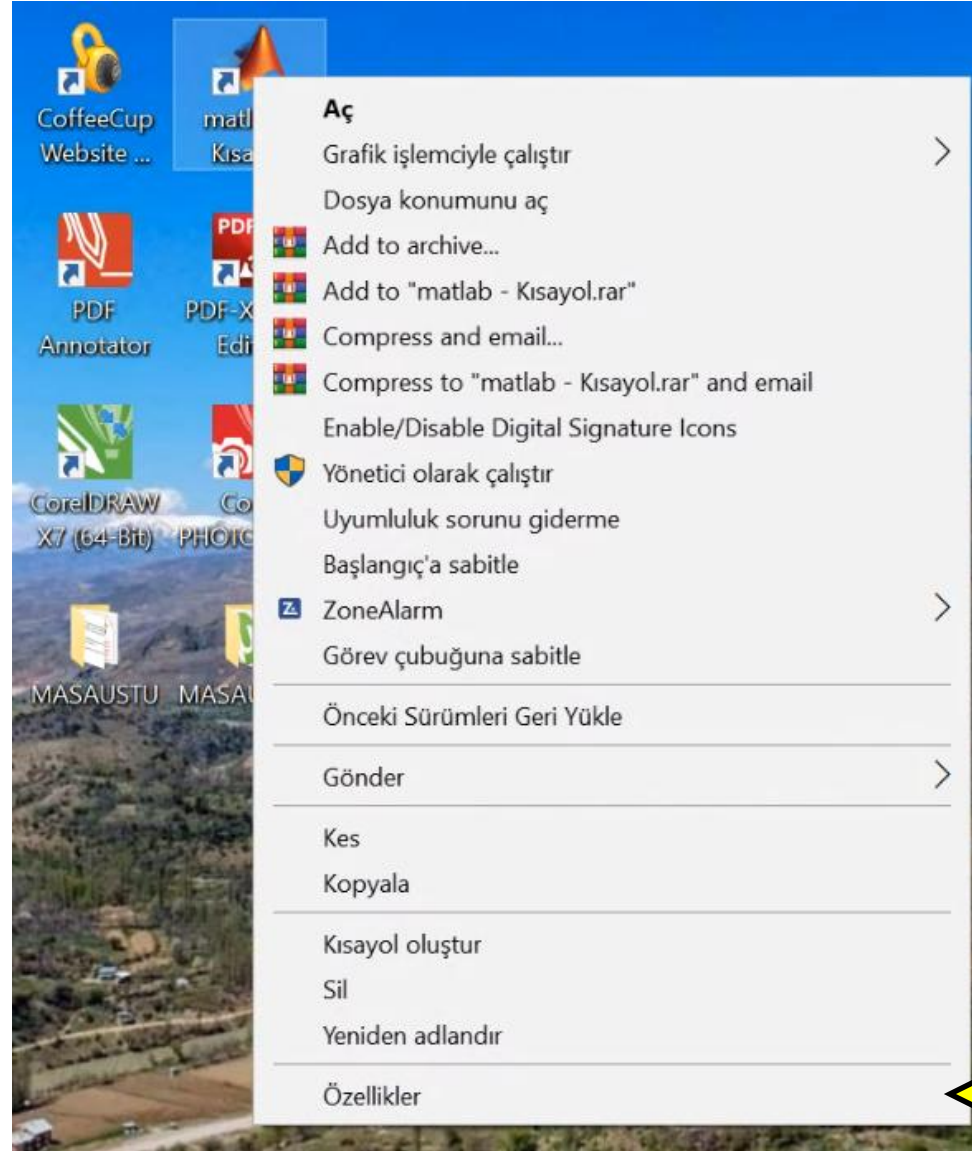
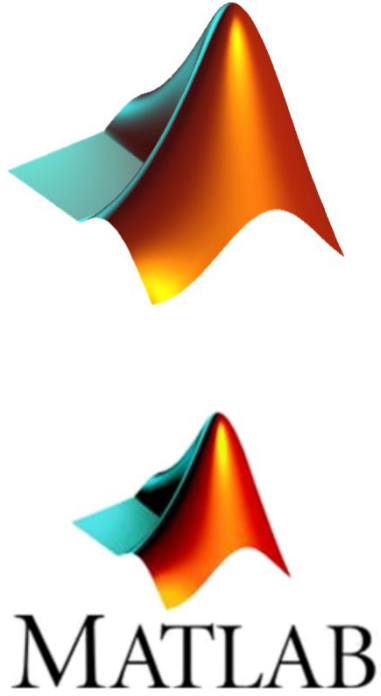
Chapter 1 – MATLAB Basics

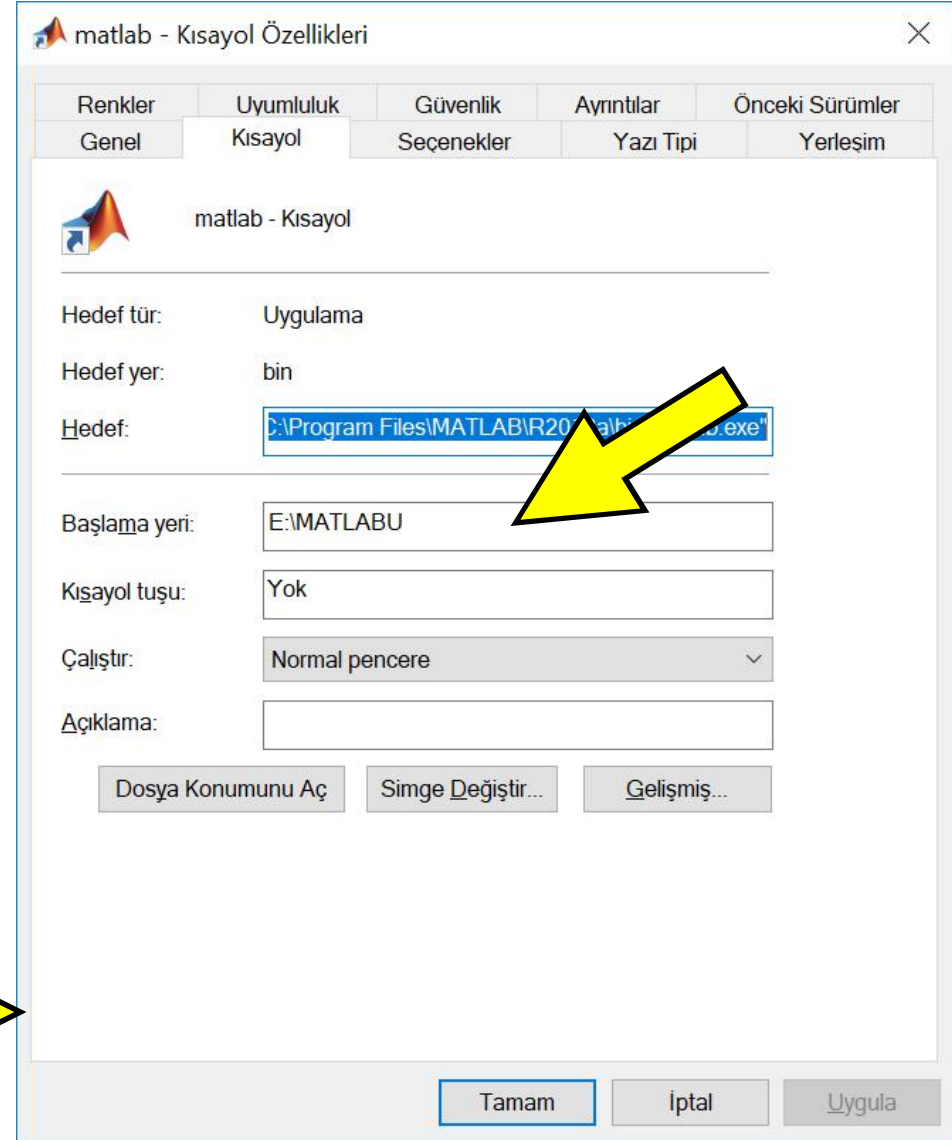
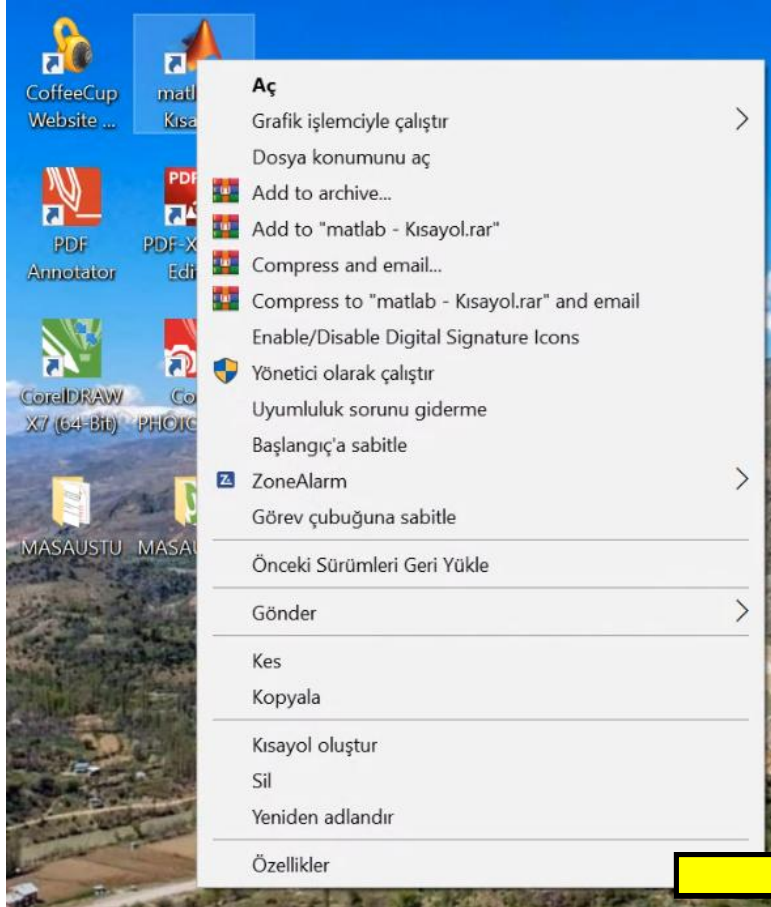
These lecture notes contain copyrighted materials. Therefore, this lecture notes are free to be used only by the students who take this course. It is subject to permission to be given to third parties and to be reproduced by any method and used elsewhere. Commercial use of this notes are prohibited. Otherwise, legal action will be taken.



Publication of these course presentation notes by others on any platform is subject to permission. **Remember, Stealing is not sharing.**









matlab - Kısayol Özellikleri

Renkler	Uyumluluk	Güvenlik	Ayrıntılar	Önceki Sürümler
Genel	Kısayol	Seçenekler	Yazı Tipi	Yerleşim

matlab - Kısayol

Hedef tür: Uygulama

Hedef yer: bin

Hedef: C:\Program Files\MATLAB\R2018a\bin\matlab.exe"

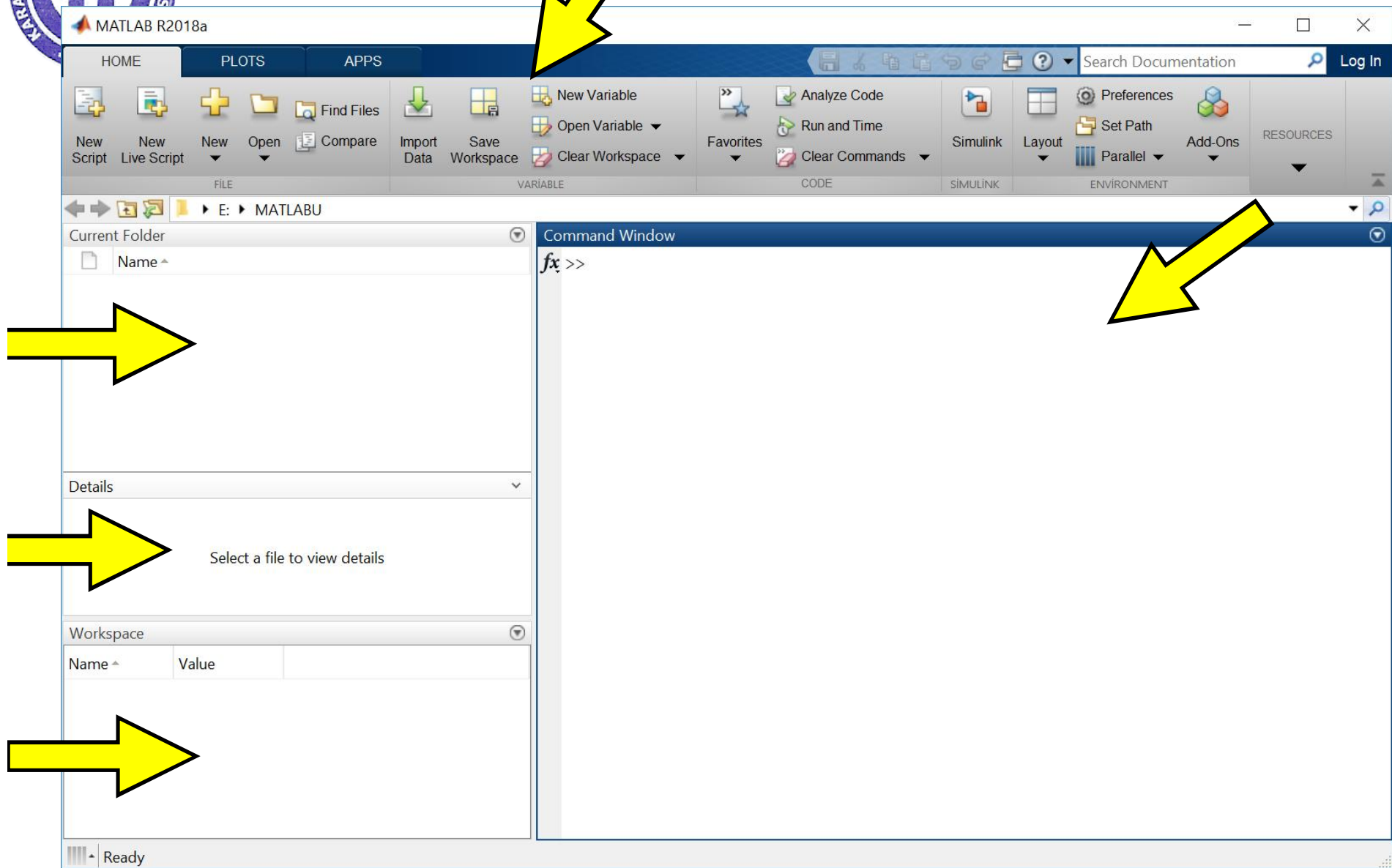
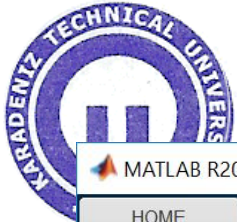
Başlama yeri: E:\MATLABU

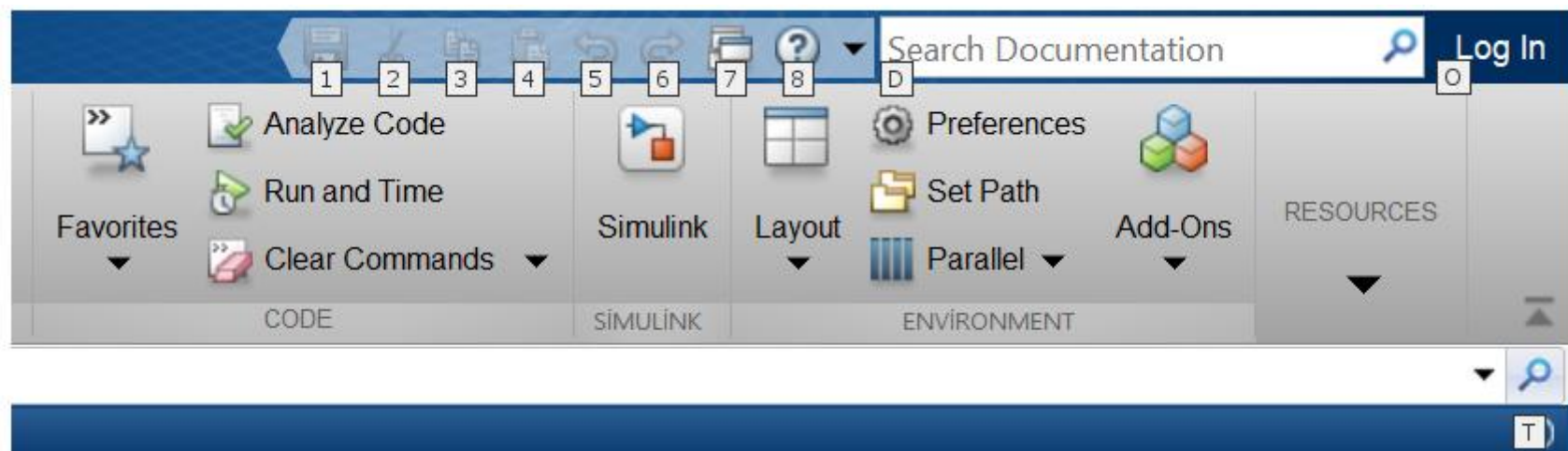
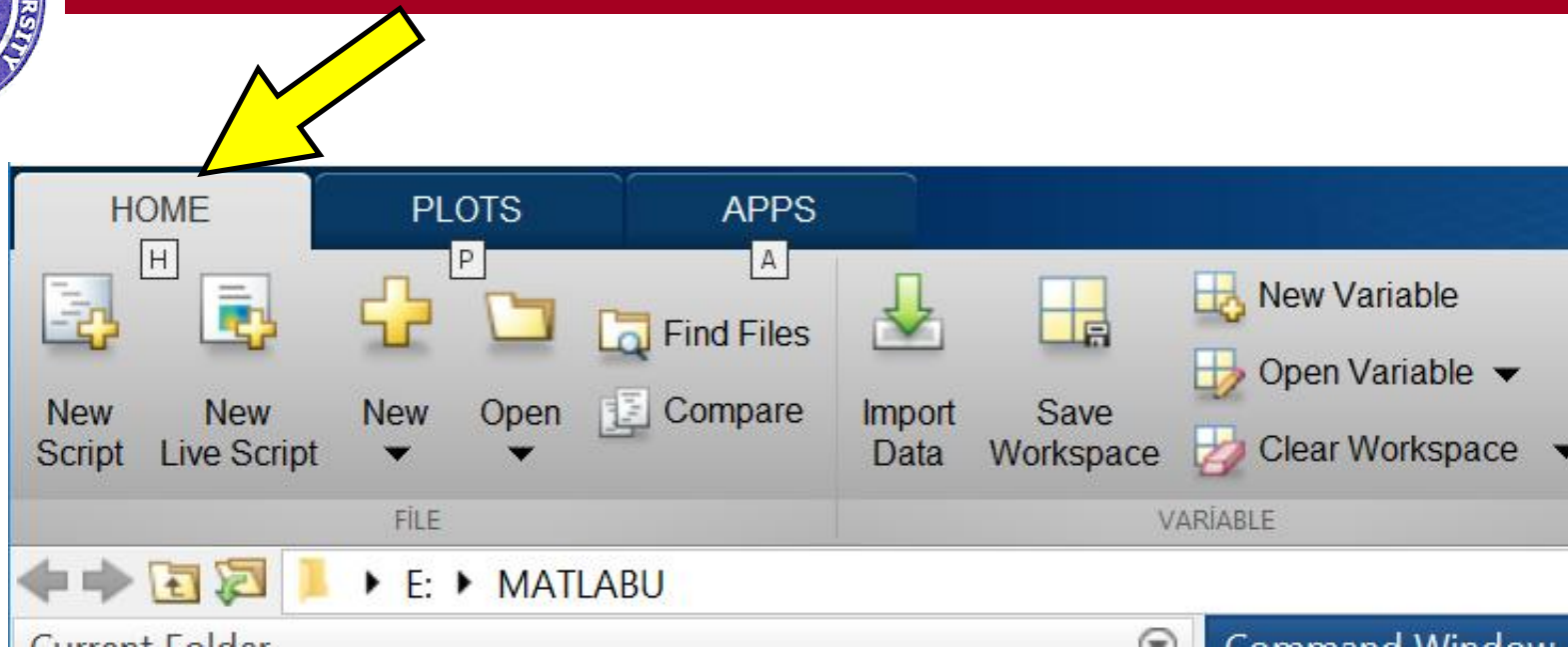
Kısayol tuşu: Yok

Çalıştır: Normal pencere

Açıklama:

Dosya Konumunu Aç Simge Değiştir... Gelişmiş...







MATLAB R2018a

HOME PLOTS APPS

No Variable Selected

SELECTION

Details

Select a file

Workspace

Name	Value
------	-------

Search Documentat

★ FAVORITES

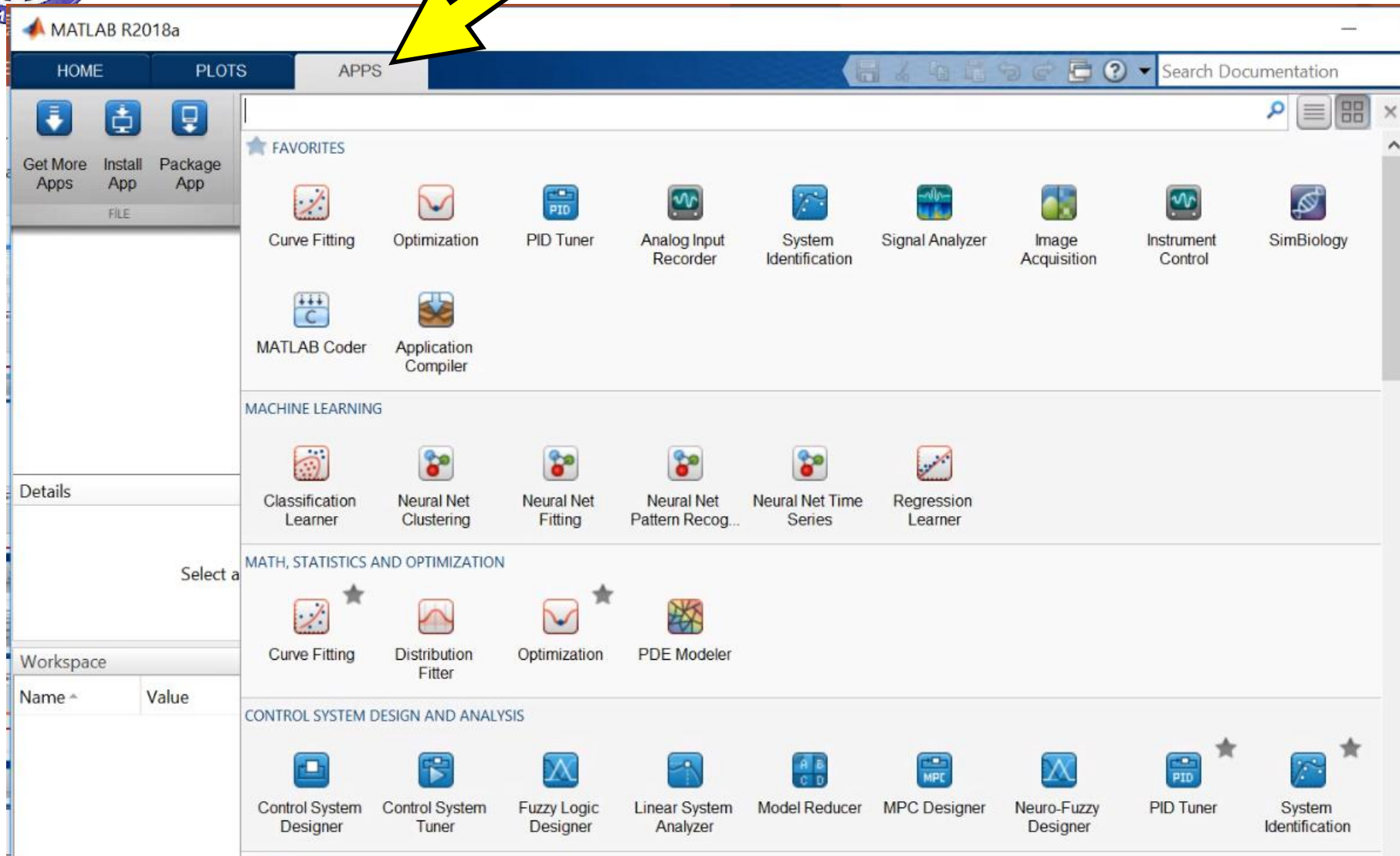
- plot
- Plot as multi...
- Plot as multi...
- area
- bar
- scatter
- pie
- histogram
- contour
- surf
- mesh


MATLAB LINE PLOTS

- plot
- Plot as multi...
- Plot as multi...
- plotyy
- semilogx
- semilogy
- loglog
- area
- errorbar
- errorbar (hor...
- plot3
- comet


MATLAB STEM AND STAIR PLOTS

- stem
- stairs
- stem3




Current Folder 

Name ^

Details 

Select a file to view details

Command Window 

```

>> A=5

A =

    5

>> B=20

B =


    20






>> C=30;
>> D=A+B+C

D =

    55

>> E=A+B+C;
fx >>
    
```

Workspace 

Name ^	Value
 A	5
 B	20
 C	30
 D	55
 E	55



MATLAB R2018a

HOME PLOTS APPS Search Documentation Log In

E: \MATLABU

Current Folder

Name ^

Details

Select a file to view details

Workspace

Name ^	Value
A	[0 2 4 6 8
B	[10 8 6 4
C	[0 4 16 36

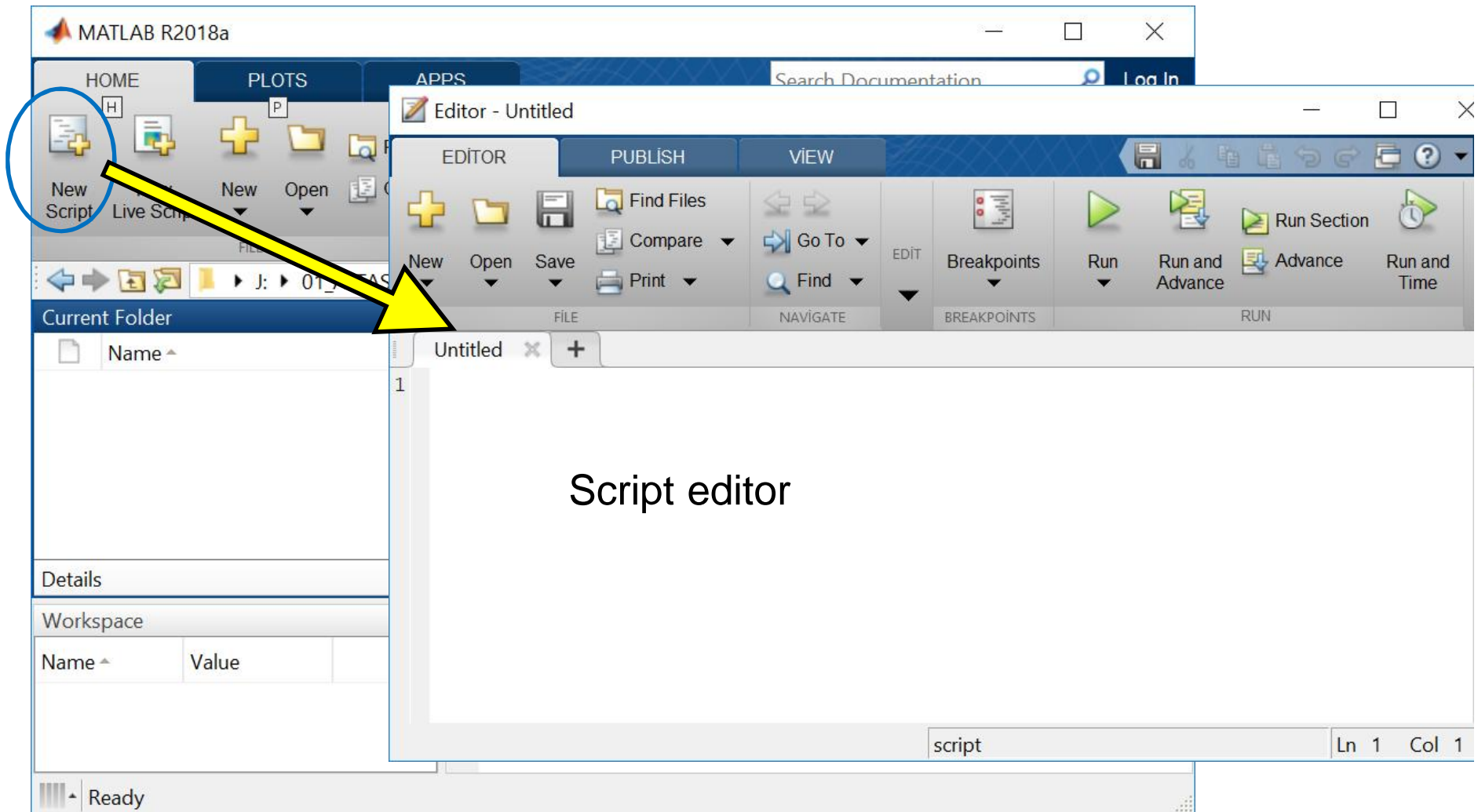
Command Window

```
>> A=0:2:10  
  
A =  
  
    0    2    4    6    8   10  
  
>> B=10:-2:0  
  
B =  
  
   10    8    6    4    2    0  
  
>> C=A.^2  
  
C =  
  
    0    4   16   36   64  100
```

fx



Writing codes with script editor





Code: for loop

1

2

3

4

```
1 clear
2 A=1:1:10;
3
4 for k=1:10
5     B(k)=11-A(k);
6 end
```

for loop

www.altas.org



Code: for loop and data generation

The screenshot displays the MATLAB R2018a environment. The script editor shows the following code:

```
clear
A=1:1:10;
for k=1:10
    B(k)=11-A(k);
end
```

The Command Window shows the execution results:

```
>> example_01
>> A
A =
     1     2     3     4     5     6     7     8     9    10

>> B
B =
    10     9     8     7     6     5     4     3     2     1

fx >>
```

The Workspace window shows the following variables:

Name	Value
A	[1 2 3 4 5 6 7 ...]
B	[10 9 8 7 6 5 ...]
k	10

Blue arrows indicate the flow of data from the script editor to the Command Window and then to the Workspace window. The number 5 is placed next to the 'A' output, and the number 6 is placed next to the 'B' output.



Code: for loop and plotting

The image shows the MATLAB environment with the following code in the editor:

```
1 clear  
2 A=1:1:10;  
3  
4 for k=1:10  
5     B(k)=11-A(k);  
6 end  
7  
8 L=size(A);  
9 N=L(:,1);  
10 M=L(:,2);  
11 C0=0;  
12 for i=1:M  
13     C=C0+A(i);  
14     C0=C;  
15 end  
16 plot(A,B)  
17 title('Example 2')  
18 xlabel('Axis X')  
19 ylabel('Axis y')  
20
```

The plot, titled "Example 2", shows a linear relationship between Axis X and Axis y. The x-axis ranges from 1 to 10, and the y-axis ranges from 0 to 10. The data points are (1, 10), (2, 9), (3, 8), (4, 7), (5, 6), (6, 5), (7, 4), (8, 3), (9, 2), and (10, 1). A blue arrow points from the "Run" button in the toolbar to the plot area.



Code: for loop and plotting

The image displays the MATLAB environment with a script editor and a figure window. The script editor shows the following code:

```
1 clear
2 A=1:1:10;
3
4 for k=1:10
5     B(k)=11-A(k);
6 end
7
8 plot(A,B)
9 title('Example 03')
10 xlabel('Variable A')
11 ylabel('Variable B')
12
```

The figure window, titled "Figure 1", displays a plot of Variable B versus Variable A. The x-axis is labeled "Variable A" and ranges from 0 to 10. The y-axis is labeled "Variable B" and ranges from 0 to 10. The plot shows a straight line with a negative slope, starting at (1, 10) and ending at (10, 1). The plot is titled "Example 03".



Code: while loop and plotting

The image displays a MATLAB script editor window titled 'example_04.m'. The script contains the following code:

```
1 clear
2 % t=0:0.001:1;
3 t=0;
4 k=1;
5 while t<=1
6     y(k)=exp(-2*t);
7     tt(k)=t;
8     t=tt(k)+0.1;
9     k=k+1;
10 end
11
12 plot(tt,y)
13 title('Example 04')
14 xlabel('Time (s)')
15 ylabel('Magnitude y')
16
```

Blue arrows point from the code to the plot. The plot, titled 'Example 04', shows a decaying exponential curve. The x-axis is labeled 'Time (s)' and ranges from 0 to 1. The y-axis is labeled 'Magnitude y' and ranges from 0.1 to 1. The curve starts at (0, 1) and decays towards 0 as time increases.

Time (s)	Magnitude y
0.0	1.0000
0.1	0.8187
0.2	0.6703
0.3	0.5488
0.4	0.4493
0.5	0.3679
0.6	0.3012
0.7	0.2466
0.8	0.2019
0.9	0.1630
1.0	0.1353



Generating data without loops and plotting

```
1 clear
2 t=0:0.001:1;
3 y=exp(-2*t);
4
5 plot(t,y)
6 title('Example 04')
7 xlabel('Time (s)')
8 ylabel('Magnititude y')
9
```

The figure shows a MATLAB workspace with a 'Current Folder' containing several example files. The file 'example_05.m' is selected. The code in this file defines a time vector t from 0 to 1 with a step of 0.001, and a magnitude vector y calculated as $y = \exp(-2 \cdot t)$. The code then uses the `plot` function to generate a plot titled 'Example 04'. The plot shows a decaying exponential curve of 'Magnititude y' (y-axis, 0 to 1) versus 'Time (s)' (x-axis, 0 to 1). The curve starts at (0, 1) and decays towards 0 as time increases.



Processing equations

The image shows a MATLAB workspace with a 'Current Folder' window on the left listing files from 'example_01.m' to 'roots.m'. The 'example_06.m' file is selected. The main window displays the script content:

```
1 clear
2 %ax^2+bx+c=0
3 a=1;
4 b=5;
5 c=6;
6
7 x1=(-b+sqrt(b^2-4*a*c))/(2*a)
8 x2=(-b-sqrt(b^2-4*a*c))/(2*a)
9
10
```

The Command Window on the right shows the execution of the script:

```
>> example_06
x1 =
    -2
x2 =
    -3
fx >>
```

Blue arrows indicate the flow of execution from the script to the Command Window output.



Working with functions

```
roots.m x +
1 function X=roots(a,b,c)
2     x1=(-b+sqrt(b^2-4*a*c))/(2*a);
3     x2=(-b-sqrt(b^2-4*a*c))/(2*a);
4     X=[x1;x2]
5
```

```
Command Window
>> % 5X^2 +3X-7=0
>> X1X2=roots(5,3,-7)

X1X2 =

    0.9207
   -1.5207

>> % X^2 +10X+25=0
>> X1X2=roots(1,10,25)

X1X2 =

    -5
    -5

fx >> |
```



Working with functions

Examples

```
Command Window

>> % X^2 +9X+20=0
>> X1X2=roots(1,9,20)

X1X2 =

    -4
    -5

fx >>
```

```
Command Window

>> % X^2 +6X+9=0
>> X1X2=roots(1,6,9)

X1X2 =

    -3
    -3
```

```
>> % X^2 +3X+9=0
>> X1X2=roots(1,3,9)

X1X2 =

   -1.5000 + 2.5981i
   -1.5000 - 2.5981i
```



EXAMPLE

Write a Matlab function to calculate equivalent value of **parallel** connected resistances.

PS: Resistance values will be entered as a vector as below

$R=[R_1 \ R_2 \ R_3 \ \dots \ R_n]$

```
function RT = paralelnr(R)
% R: paralel bađlý direnç deđerlerinden oluđan vektör
% Bu vektörün sayısal olarak önceden oluđturulması
% gerekir
n=length(R);
RTOP1=0;
for i=1:n
    RTOP1=RTOP1+(1/R(i));
end
RT=1/RTOP1;
```



EXAMPLE

Write a Matlab function to calculate equivalent value of **series** connected resistances.

PS: Resistance values will be entered as a vector as below

$R=[R_1 \ R_2 \ R_3 \ \dots \ R_n]$

```
function RT = serinr(R)
% R: Seri bađlý direnç deđerlerinden oluđan vektör
% Bu vektörün sayýsal olarak önceden oluđturulması
% gerekir
n=length(R);
RT1=0;
for i=1:n
    RT1=RT1+R(i);
end
RT=RT1;
```



Conditional Statments

```
x=input('Enter x value ==>')
```

```
x1=4; x2=8; x3=12;
```

```
if x <= x1
```

```
    A=5
```

```
elseif x<=x2
```

```
    A=10
```

```
elseif x <= x3
```

```
    A=20
```

```
else
```

```
    A =100
```

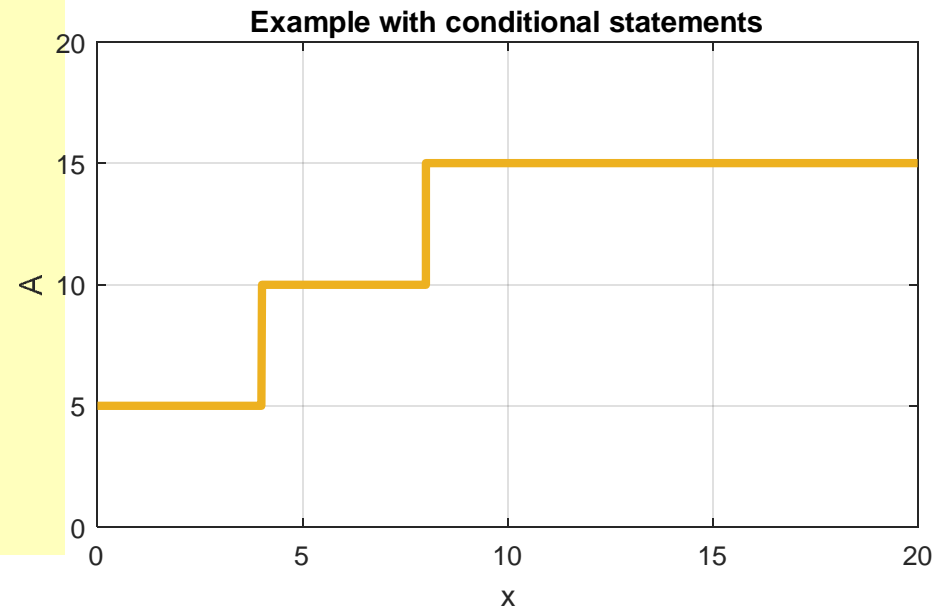
```
end
```




Conditional Statements

```
x1=4; x2=8; x3=12;  
x=0; k=1;  
while x <=20  
    if x <= x1  
        A(k)=5;  
    elseif x <= x2  
        A(k)=10;  
    else  
        A(k) =20;  
    end  
    xx(k)=x;  
    x=x+0.01;  
    k=k+1;  
end
```

```
plot(0,25,xx,A)  
title('Example with conditional  
statements')  
xlabel('x'); ylabel('A'); grid
```





Conditional Statements

```
% Inputs X, n and x1, x2 and x
clear; clc; clf
n=3; x1=-4; x2=-x1; X=x2-x1;
a=X/(n-1); b=(X/2)-a;
for m=1:n
    XP(m)=x1+a*(m-1);
end
XPA=XP(1); XPB=XP(2); XPC=XP(3);
A1=(b*pi)/(2*a); A2=A1+(pi/2);
k=1;
```

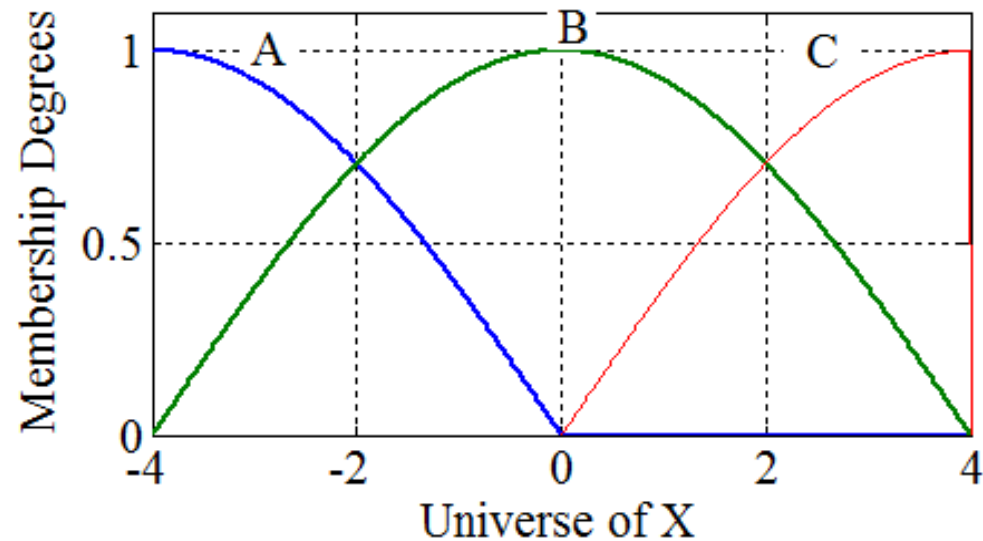
```
function mu=sinus01m(a,A,t)
    T=4*a;
    w1=(2*pi/T);
    mu=abs(sin(w1*t+A));
```

```
for x=x1:0.01:x2;
    if x<XPA
        SA(k)=0; SB(k)=0; SC(k)=0;
    elseif x<XPB
        SA(k)=sinus01m(a,A1,x);
        SB(k)=sinus01m(a,A2,x);
        SC(k)=0;
    elseif x<XPC
        SA(k)=0;
        SB(k)=sinus01m(a,A2,x);
        SC(k)=sinus01m(a,A1,x);
    else
        SA(k)=0; SB(k)=0; SC(k)=0;
    end
    X(k)=x; k=k+1;
end
```



Conditional Statments

```
plot(X,SA,X,SB,X,SC);  
xlabel('Universe of X');  
ylabel('Membership Degrees');  
grid
```





Matrices and Arrays

Array Creation

To create an array with four elements in a single row, separate the elements with either a comma (,) or a space.

```
a = [1 2 3 4]
```

```
a =  
1 2 3 4
```

This type of array is a *row vector*.



Matrices and Arrays

Array Creation

To create a matrix that has multiple rows, separate the rows with semicolons.

```
a = [1 2 3; 4 5 6; 7 8 10]
```

```
a =  
1   2   3  
4   5   6  
7   8  10
```

Another way to create a matrix is to use a function, such as ones, zeros, or rand. For example, create a 5-by-1 column vector of zeros.

```
b = 10+a
```

```
b =  
11  12  13  
14  15  16  
17  18  20
```

```
z = zeros(4,1)
```

```
z =  
0  
0  
0  
0
```



Matrices and Arrays

To **transpose** a matrix, use a single quote ('):

```
a =  
1 2 3  
4 5 6  
7 8 10  
c = a'  
>> c  
ans =  
1 4 7  
2 5 8  
3 6 10
```

Inverse of a matrix

```
d = inv(a)
```

```
d =  
  
-0.6667 -1.3333 1.0000  
-0.6667 3.6667 -2.0000  
1.0000 -2.0000 1.0000
```

```
E = a*inv(a)
```

```
E =  
  
1.0000 -0.0000 -0.0000  
0.0000 1.0000 -0.0000  
0.0000 -0.0000 1.0000
```



Matrices and Arrays

The image displays the MATLAB environment. On the left, the 'Current Folder' pane shows a directory structure with 'm_files' containing several example scripts, with 'example_07.m' selected. The main editor window shows the script 'example_07.m' with the following code:

```
1 clear
2 A=[ 2 5 6
3     1 2 3
4     0 4 2 ];
5
6 A2=[ 2 5 6; 1 2 3; 0 4 2];
7
8 B=[ 1 2 3
9     3 0 1
10    7 5 3 ];
11 C=A*B;
12 D=inv(A);
13 E=inv(B);
14 F=inv(C);
15
```

On the right, the 'Command Window' shows the execution results:

```
>> example_07
>> C
C = 59    34    29
    28    17    14
    26    10    10

>> D
D = 4.0000   -7.0000   -1.5000
    1.0000   -2.0000     0
   -2.0000    4.0000    0.5000

>> E
E = -0.1389    0.2500    0.0556
   -0.0556   -0.5000    0.2222
    0.4167    0.2500   -0.1667

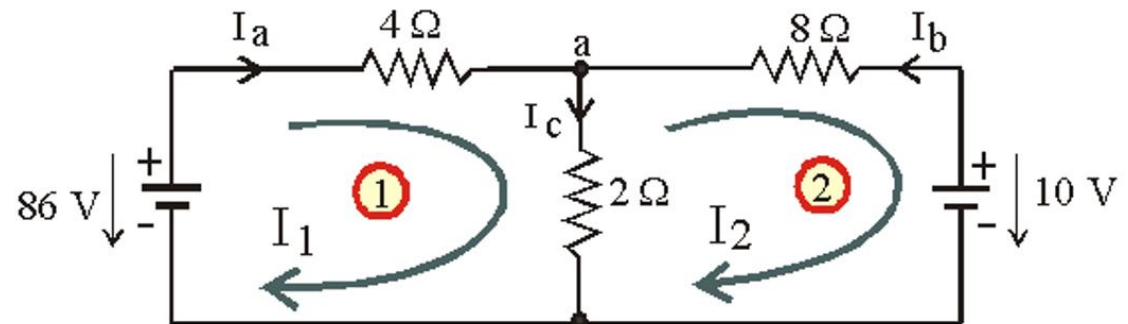
>> F
F = -0.4167    0.6944    0.2361
   -1.1667    2.2778    0.1944
    2.2500   -4.0833   -0.7083
```



Matrices and Arrays

EXAMPLE

Use matrix operations in Matlab and obtain the loop currents in circuit given below



Prepare the formulas from figure

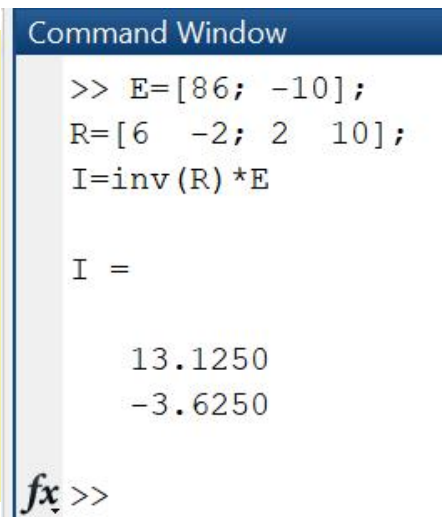
$$\begin{bmatrix} -86 \\ 10 \end{bmatrix} + \begin{bmatrix} 4+2 & -2 \\ -2 & 8+2 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = 0$$

$$\begin{bmatrix} 86 \\ -10 \end{bmatrix} = \begin{bmatrix} 4+2 & -2 \\ -2 & 8+2 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \end{bmatrix}$$

$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} 4+2 & -2 \\ -2 & 8+2 \end{bmatrix}^{-1} \begin{bmatrix} 86 \\ -10 \end{bmatrix}$$

```
E=[86; -10];
R=[6 -2; 2 10];
I=inv(R)*E
```

```
I=
15.0000
2.0000
```





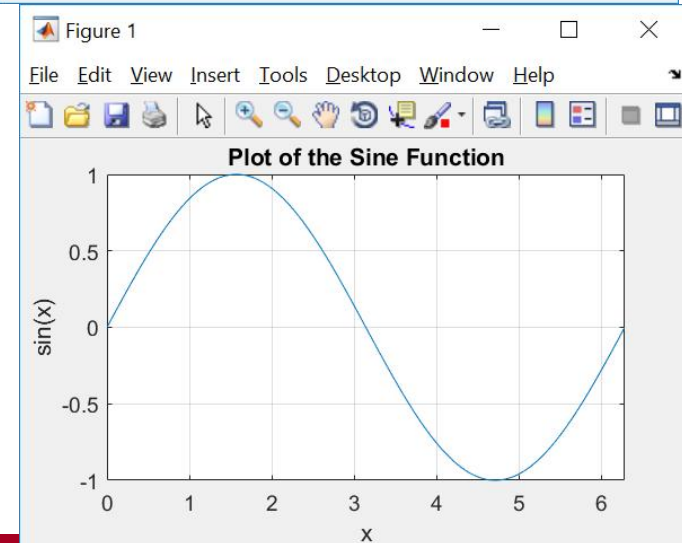
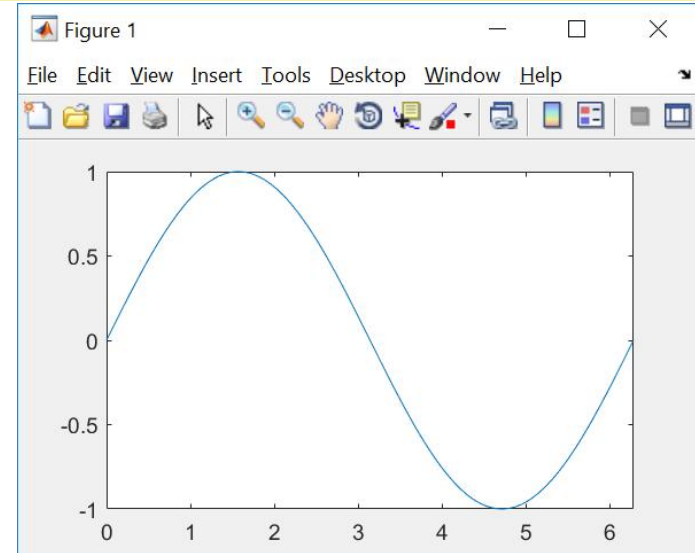
Line Plots

To create two-dimensional line plots, use the plot function. For example, plot the value of the sine function from 0 to 2π :

```
x = 0:pi/100:2*pi;  
y = sin(x);  
plot(x,y)
```

You can label the axes and add a title.

```
xlabel('x')  
ylabel('sin(x)')  
title('Plot of the Sine Function')  
grid
```

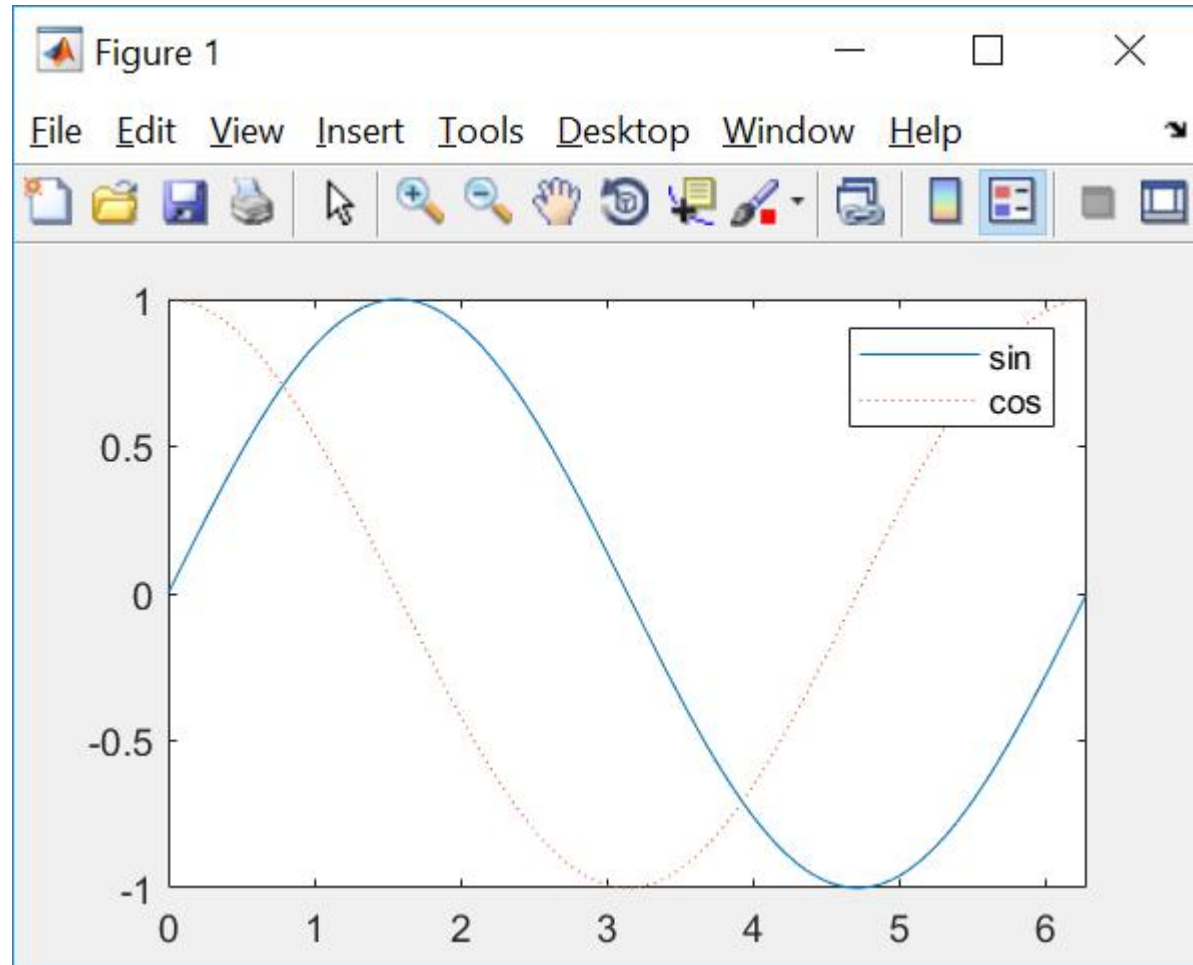




Line Plots - Examples

```
x = 0:pi/100:2*pi;  
y = sin(x); plot(x,y)
```

```
hold on  
y2 = cos(x);  
plot(x,y2, ':')  
legend('sin', 'cos')  
hold off
```





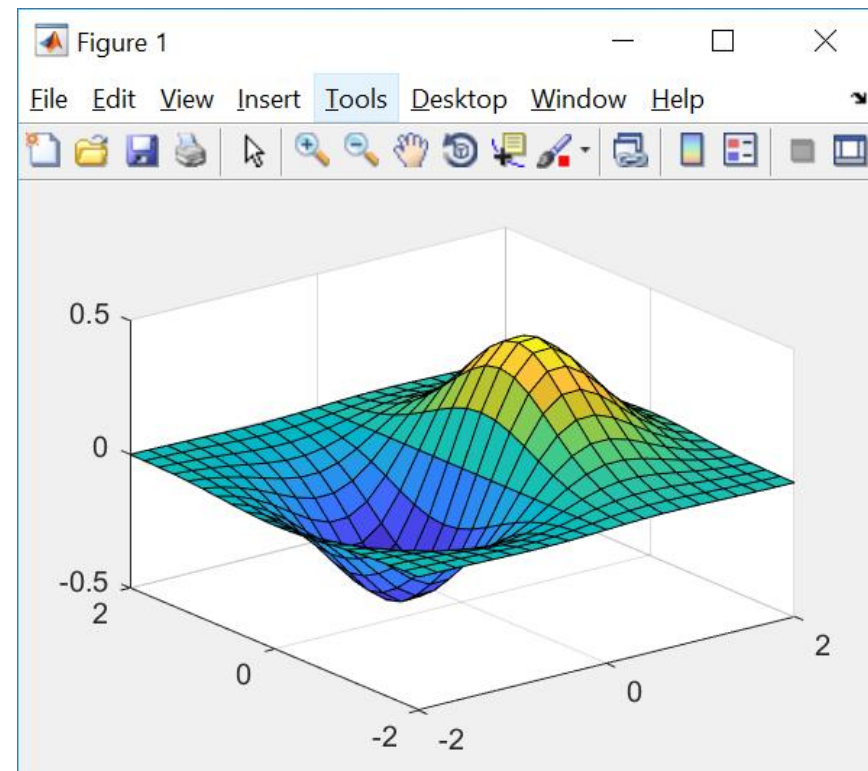
3-D Plots

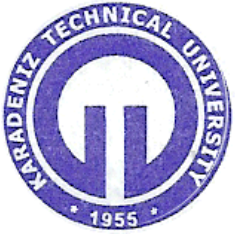
Three-dimensional plots typically display a surface defined by a function in two variables, $z = f(x,y)$.

To evaluate z , first create a set of (x,y) points over the domain of the function using **meshgrid**.

```
[X,Y] = meshgrid(-2:.2:2);  
Z = X .* exp(-X.^2 - Y.^2);
```

```
surf(X,Y,Z)
```





Subplots

You can display multiple plots in different subregions of the same window using the subplot function.

The first two inputs to subplot indicate the number of plots in each row and column. The third input specifies which plot is active. For example, create four plots in a 2-by-2 grid within a figure window.

```
t = 0:pi/10:2*pi;  
[X,Y,Z] = cylinder(4*cos(t));  
  
subplot(2,2,1); mesh(X); title('X');  
subplot(2,2,2); mesh(Y); title('Y');  
subplot(2,2,3); mesh(Z); title('Z');  
subplot(2,2,4); mesh(X,Y,Z); title('X,Y,Z');
```

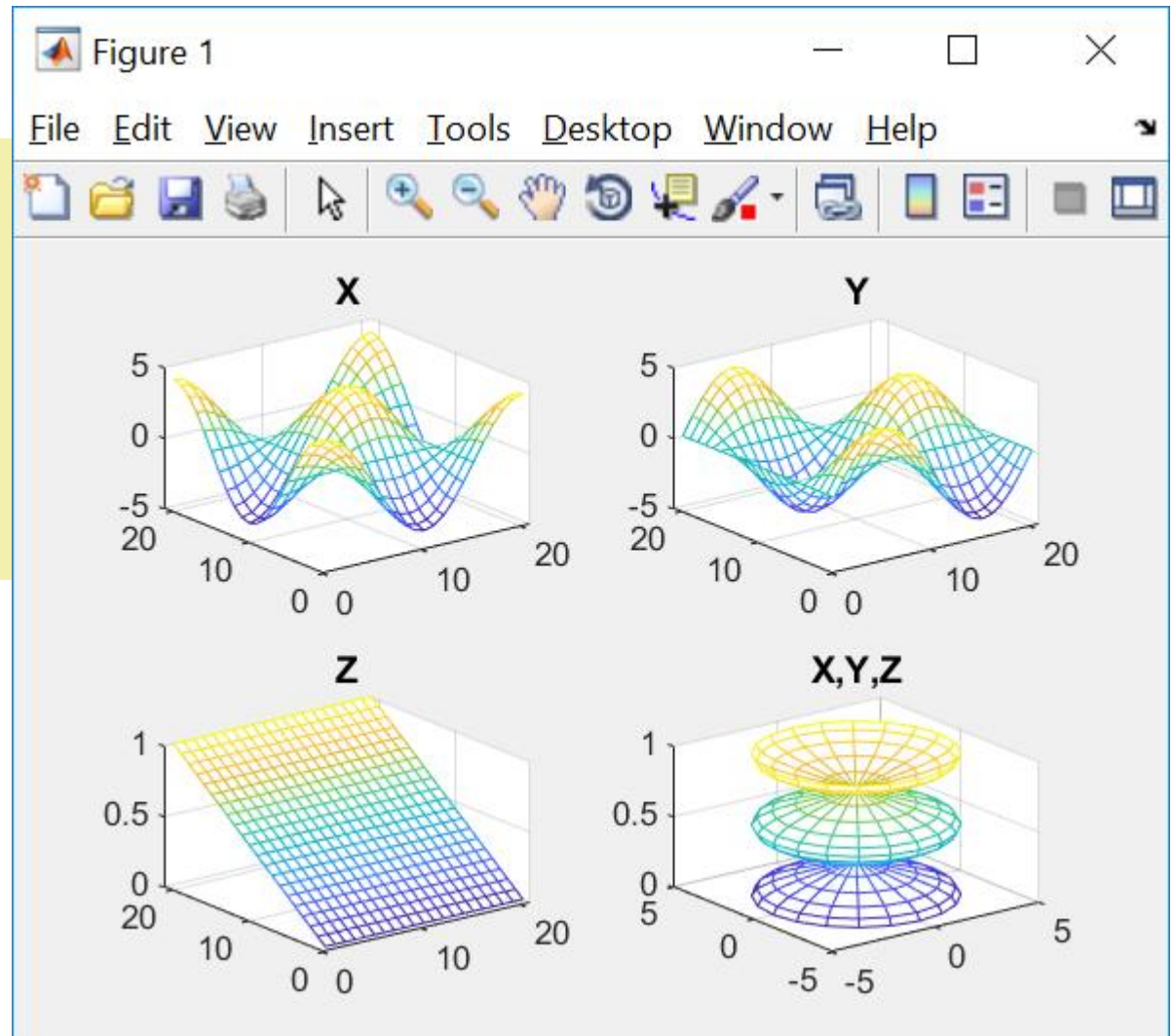


Subplots

```
t = 0:pi/10:2*pi;  
[X,Y,Z] = cylinder(4*cos(t));
```

```
subplot(2,2,1); mesh(X);  
subplot(2,2,2); mesh(Y);  
subplot(2,2,3); mesh(Z);  
subplot(2,2,4); mesh(X,Y,Z);
```

2,2,1	2,2,2
2,2,3	2,2,4





NEXT

Simulink Basics