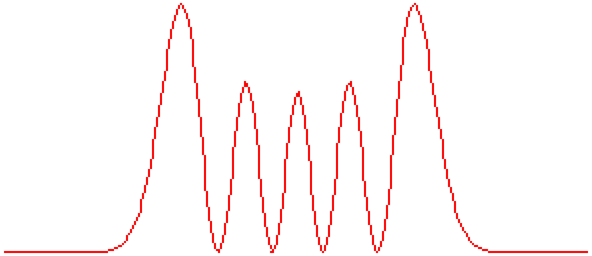
EP547 Computational Methods in QM

Topic 4

Plotting



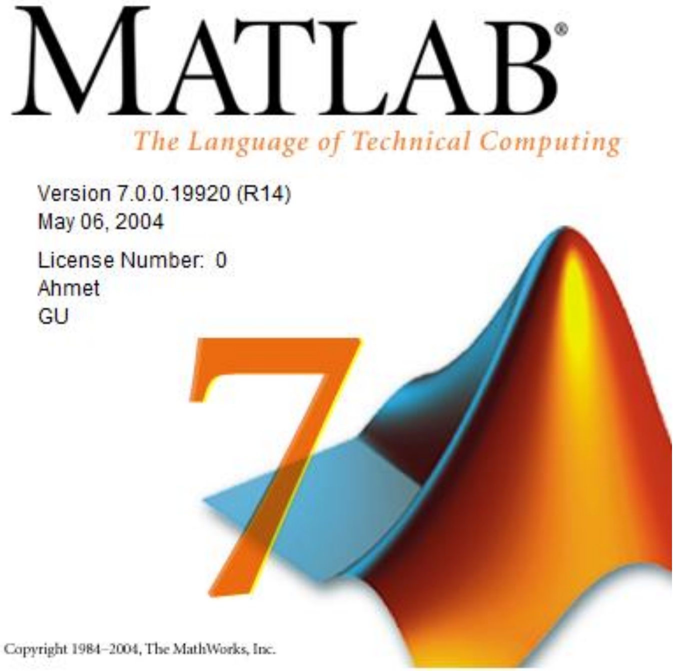
Department of
Engineering Physics

University of Gaziantep

Feb 2013

Content

1. **2D and 3D Plotting**



MATLAB®

The Language of Technical Computing

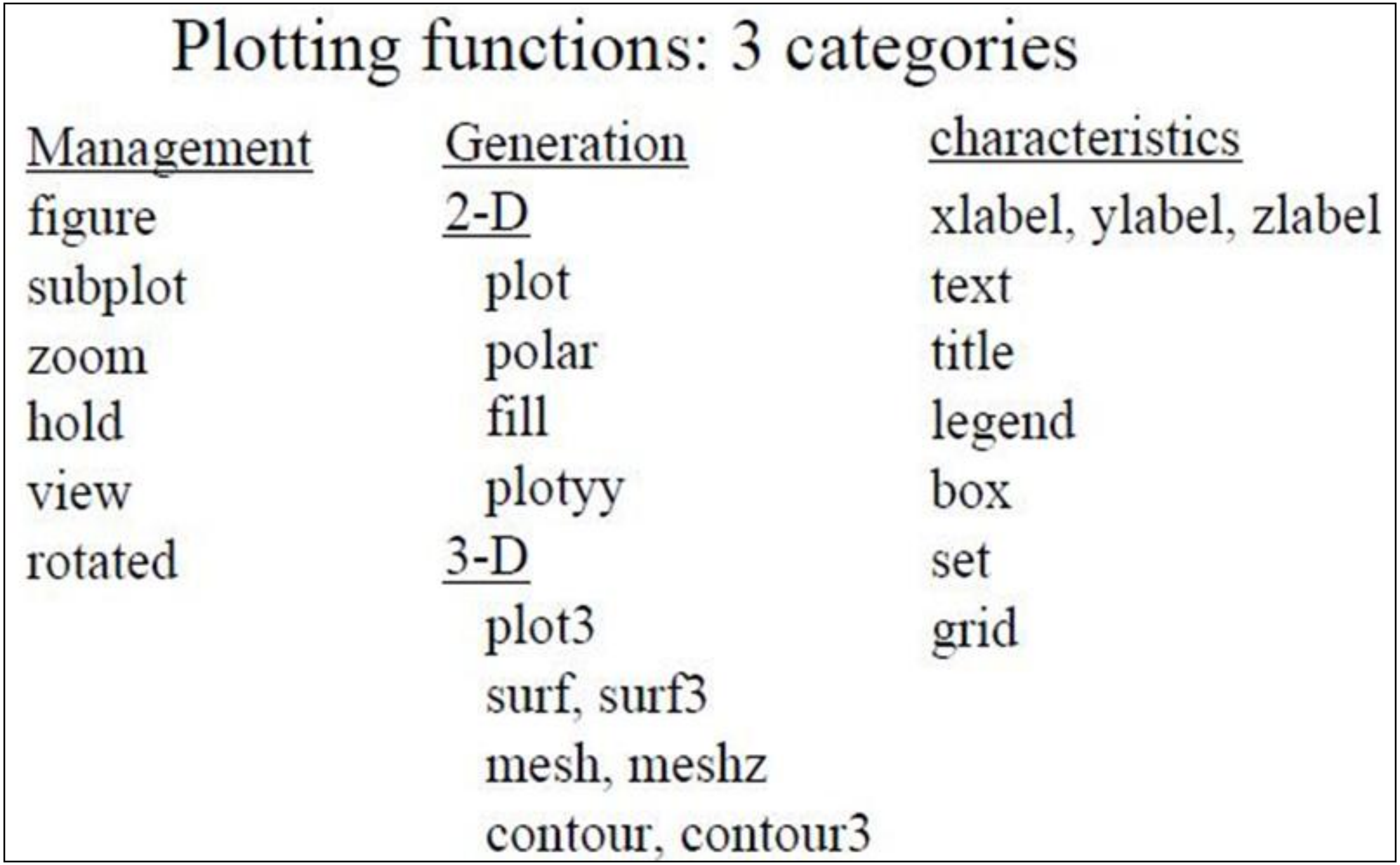
Version 7.0.0.19920 (R14)
May 06, 2004

License Number: 0
Ahmet
GU

7

Copyright 1984–2004, The MathWorks, Inc.

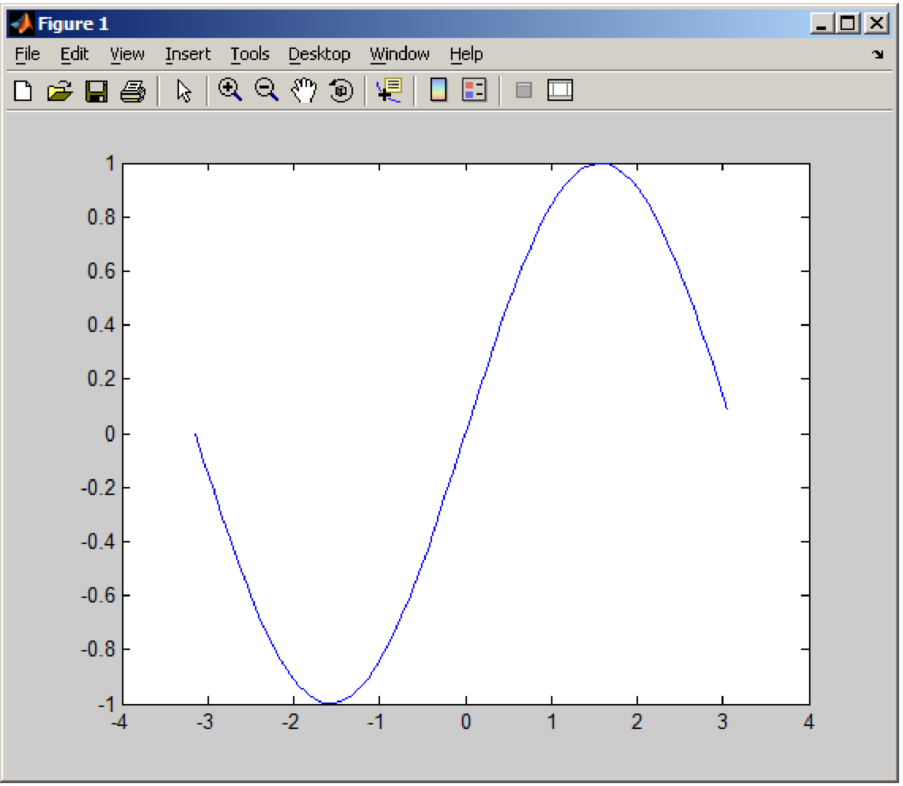
- MATLAB has extensive plotting capabilities.
- Here we illustrate some basic commands for 2D and 3D plots.



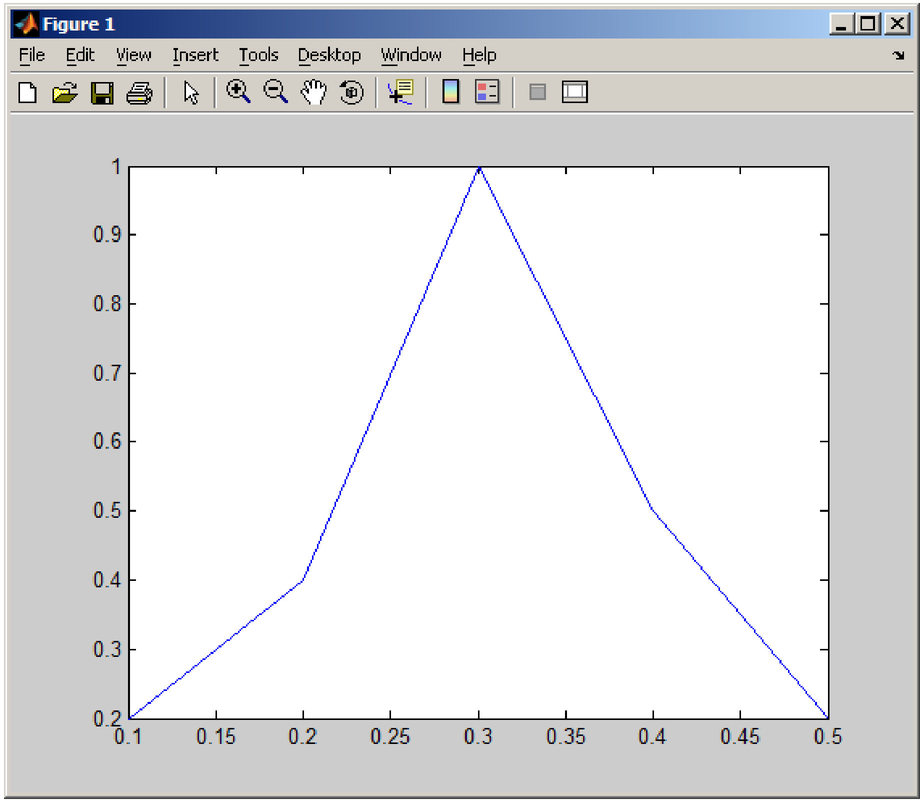
Plotting functions: 3 categories

Management	Generation	characteristics
figure	2-D	xlabel, ylabel, zlabel
subplot	plot	text
zoom	polar	title
hold	fill	legend
view	plotyy	box
rotated	3-D	set
	plot3	grid
	surf, surf3	
	mesh, meshz	
	contour, contour3	

```
>> x = -pi:0.1:pi;
>> y = sin(x);
>> plot(x,y)
>>
```

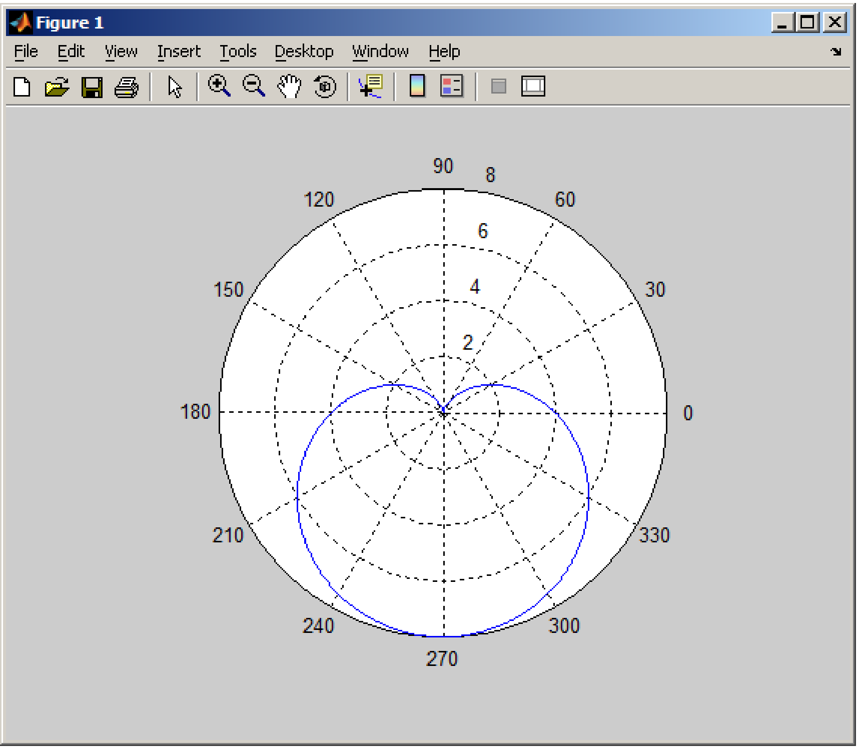


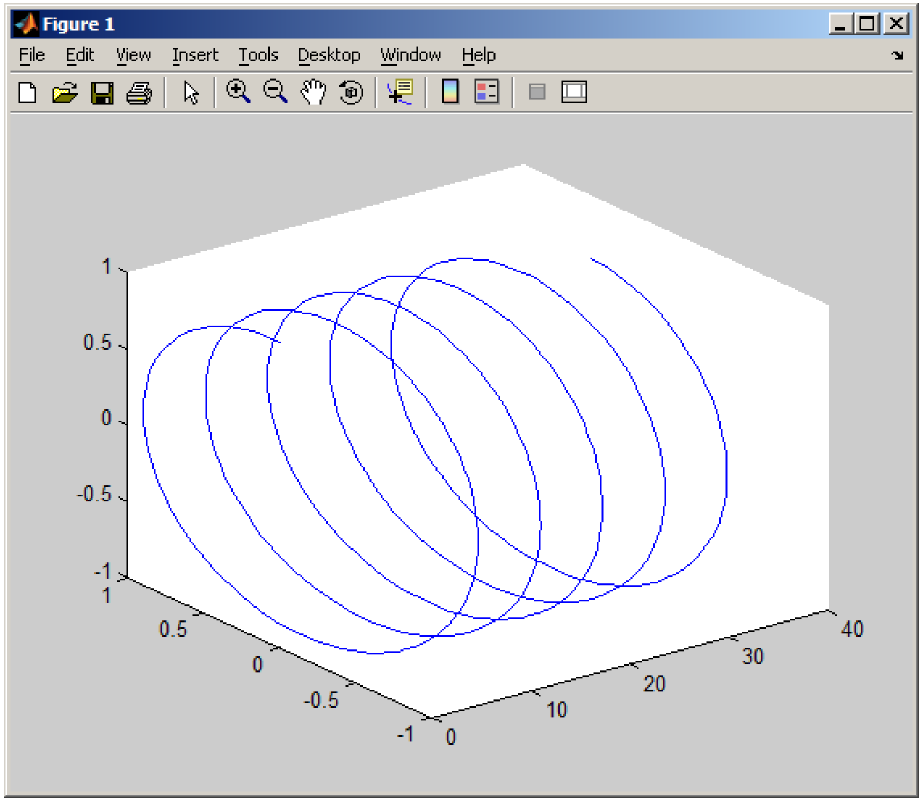
```
>> x = [0.1 0.2 0.3 0.4 0.5];
>> y = [0.2 0.4 1.0 0.5 0.2];
>> plot(x,y);
>>
```



```
>> t = 0:0.01:2*pi;
>> polar(t, 4-4*sin(t));
>>
```

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

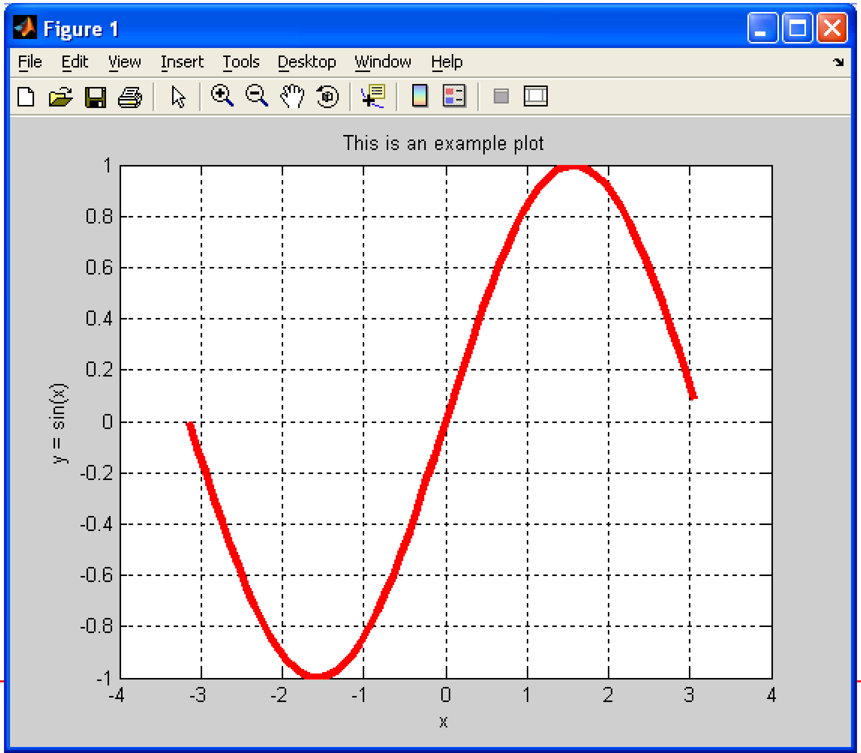


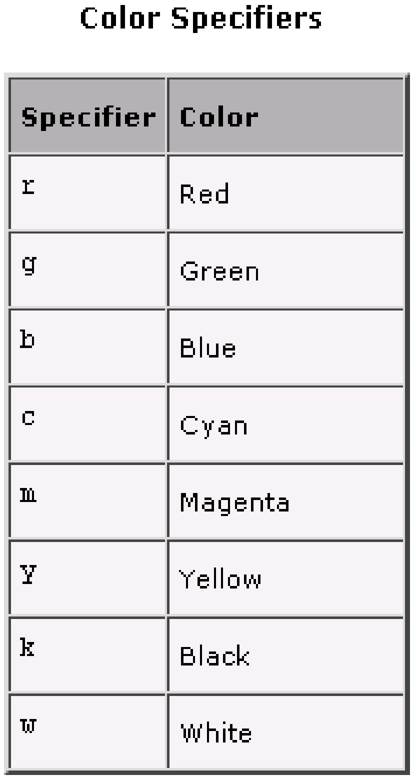


ciz1.m

```
x = -pi:0.1:pi;
y = sin(x);
p = plot(x,y);
set(p,'Color','red','LineWidth',4);
title('This is an example plot');
xlabel('x');
ylabel('y = sin(x)');
grid on;
```

```
>> ciz1
>>
```





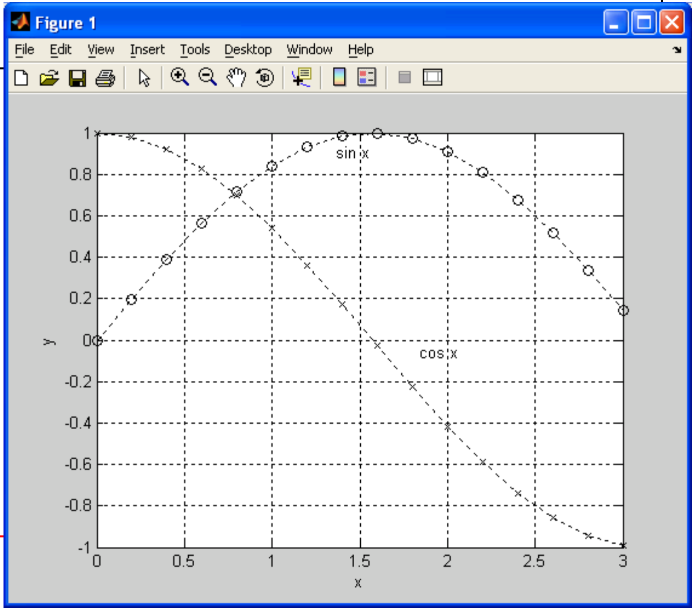
Color Specifiers

Specifier	Color
r	Red
g	Green
b	Blue
c	Cyan
m	Magenta
y	Yellow
k	Black
w	White

ciz2.m

```
x = 0:0.2:pi;         % Create x-array
y = sin(x);           % Create y-array
plot(x,y,'k:o');      % Plot x-y points with specified color
                      % and symbol (’k’ = black, ’o’ = circles)
hold on;              % Allow overwriting of current plot
z = cos(x);           % Create z-array
plot(x,z,'b:x');      % Plot x-z points (’x’ = crosses)
grid on;              % Display coordinate grid
xlabel('x');          % Display label for x-axis
ylabel('y');          % Display label for y-axis
gtext('sin x');       % Create mouse-movable text
gtext('cos x');       %
```

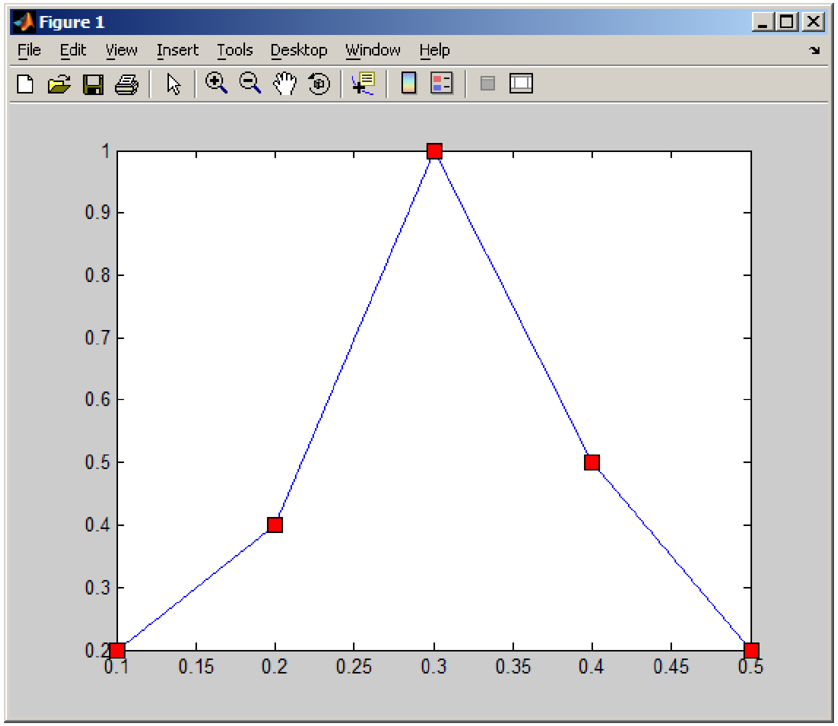
```
>> ciz2
>>
```

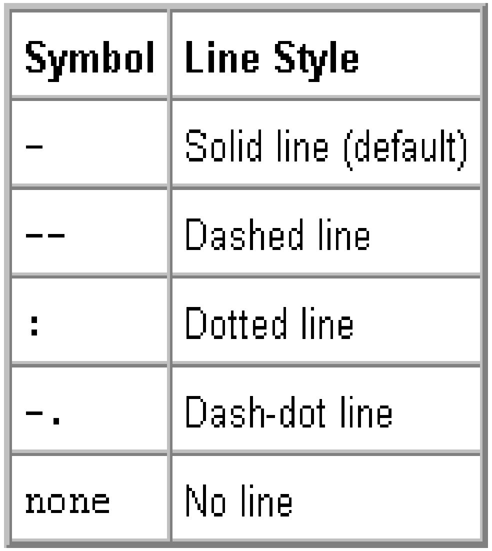


ciz3.m

```
x = [0.1 0.2 0.3 0.4 0.5];
y = [0.2 0.4 1.0 0.5 0.2];
plot(x,y,'-s', 'MarkerFaceColor','r',...
               'MarkerEdgeColor','k',...
               'MarkerSize',10);
```

```
>> ciz3
>>
```



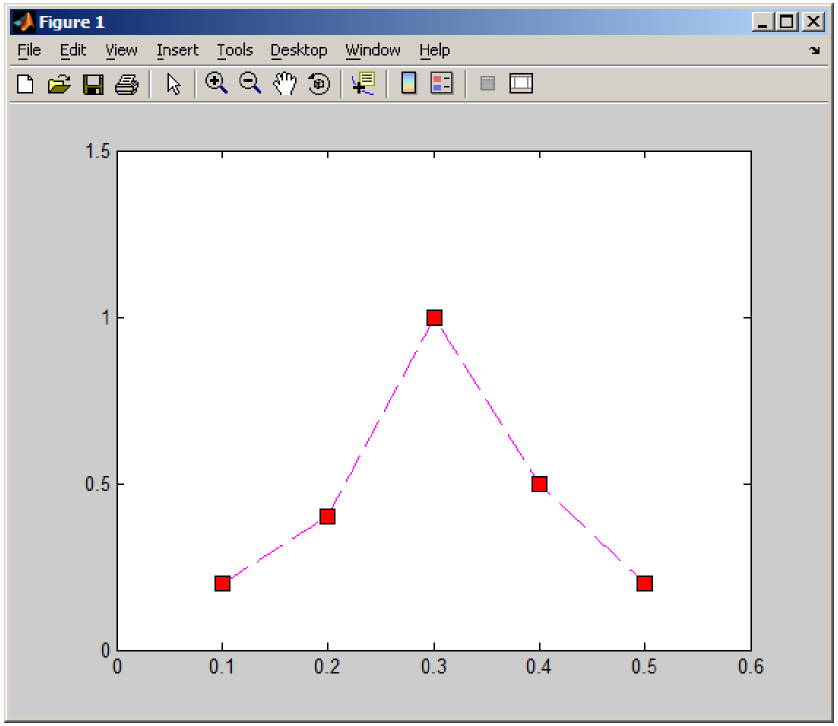


Symbol	Line Style
-	Solid line (default)
--	Dashed line
:	Dotted line
-.	Dash-dot line
none	No line

ciz4.m

```
x = [0.1 0.2 0.3 0.4 0.5];
y = [0.2 0.4 1.0 0.5 0.2];
plot(x,y,'--ms', 'MarkerFaceColor','r',...
                 'MarkerEdgeColor','k',...
                 'MarkerSize',10);
axis([0 0.6 0 1.5]);
```

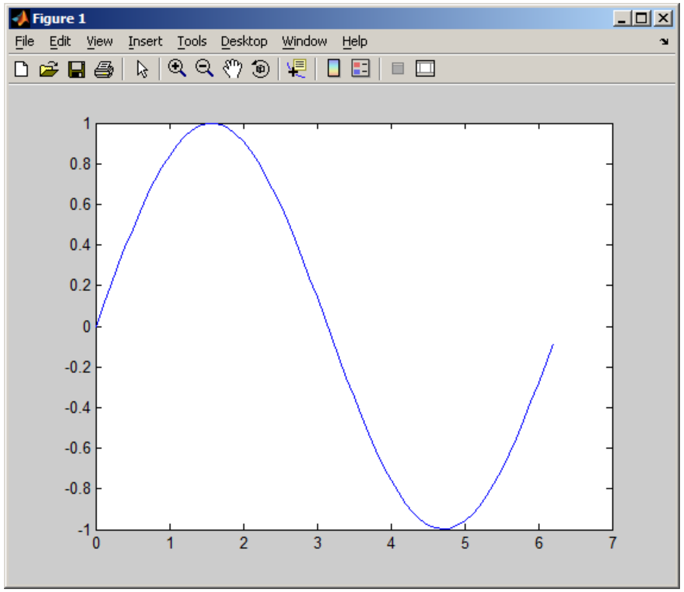
```
>> ciz4
>>
```

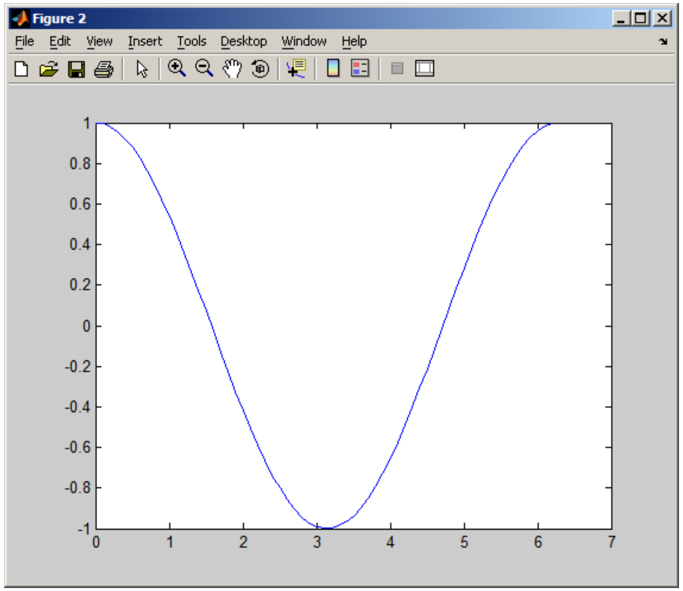


ciz5.m

```
x = 0.0:0.1:2*pi;
y = sin(x);
z = cos(x);

figure(1);
plot(x,y);

figure(2);
plot(x,z);
```

```
>> ciz5
>>
```

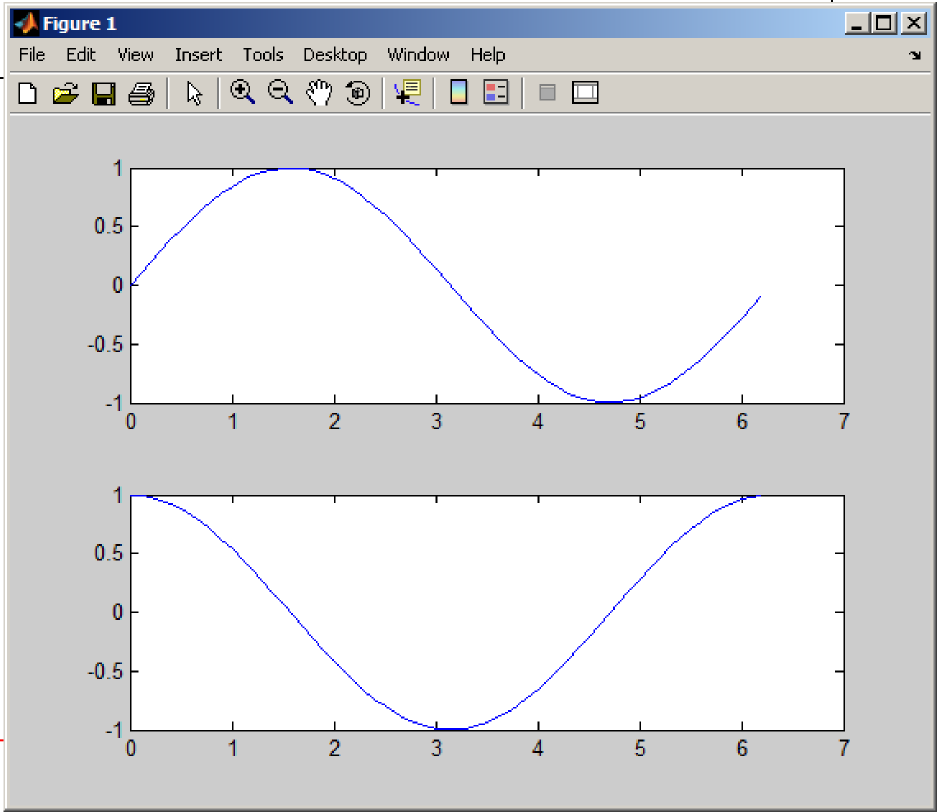




ciz6.m

```
x = 0.0:0.1:2*pi;
y = sin(x);
z = cos(x);

subplot(2,1,1);
plot(x,y);

subplot(2,1,2);
plot(x,z);
```

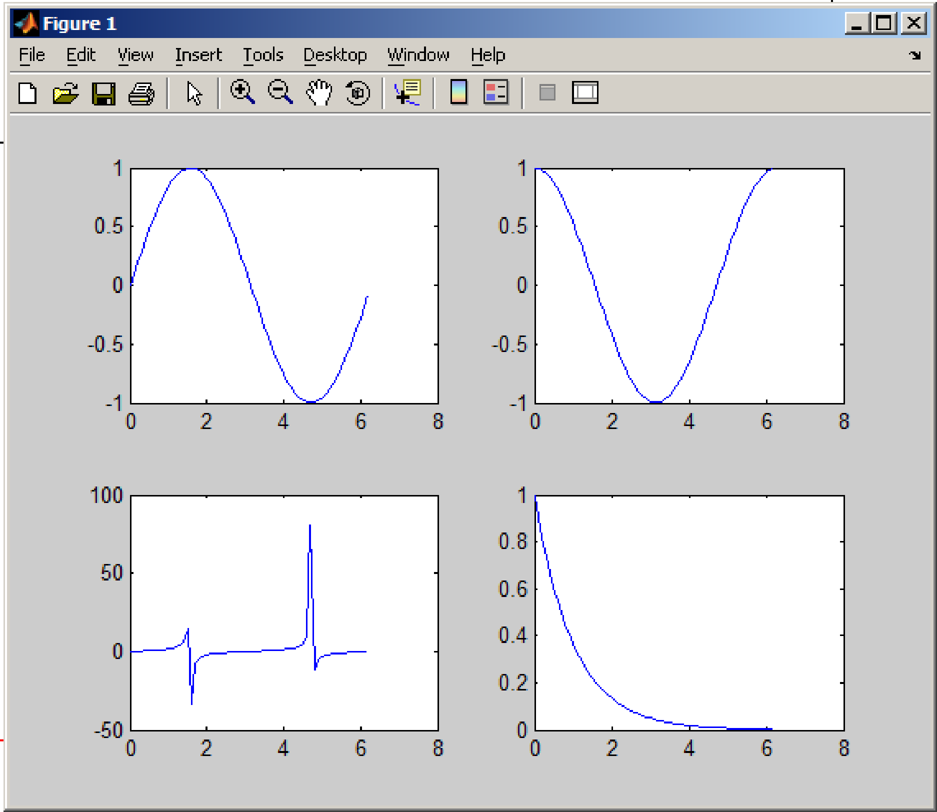
```
>> ciz6
>>
```



ciz7.m

```
x = 0.0:0.1:2*pi;
y = sin(x);
z = cos(x);
w = tan(x);
q = exp(-x);

subplot(2,2,1); plot(x,y);
subplot(2,2,2); plot(x,z);
subplot(2,2,3); plot(x,w);
subplot(2,2,4); plot(x,q);
```

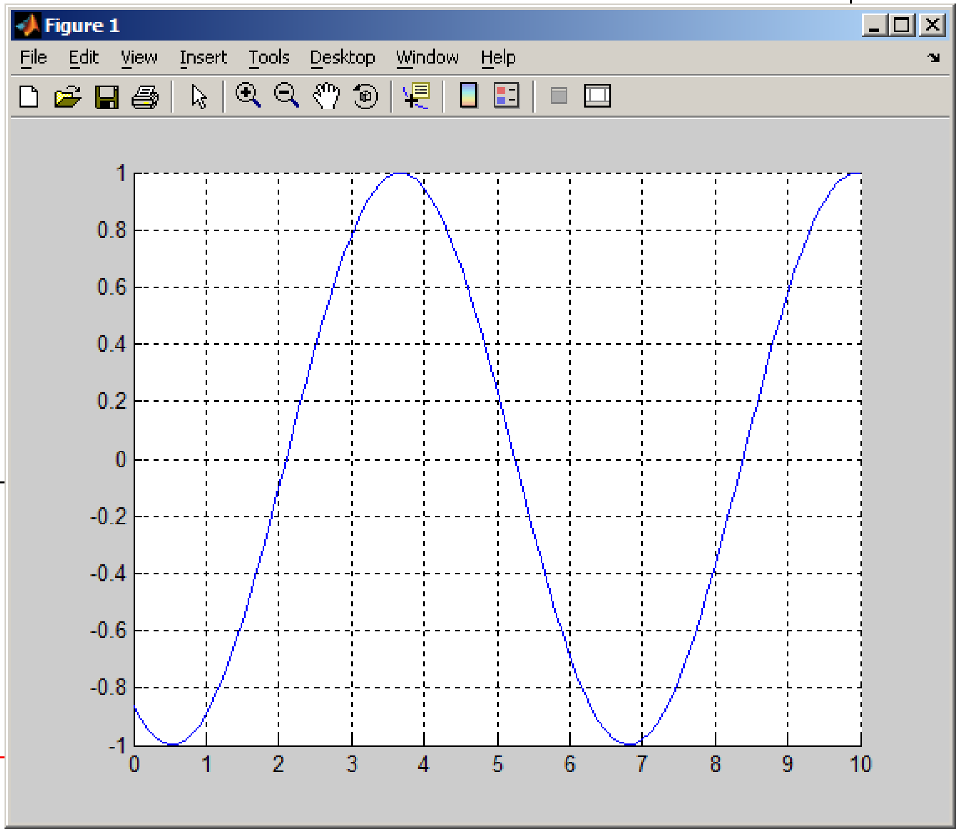
```
>> ciz7
>>
```



ciz8.m

```
% Dynamic Plot
figure;
hold on;
grid on;
v = 10.0;
x = 0.0:0.1:10;
t = 0.0;

while 1
   y = sin(x-v*t);
   h = plot(x,y);
   pause(0.1);
   delete(h);
   t = t + 0.01;
end
```

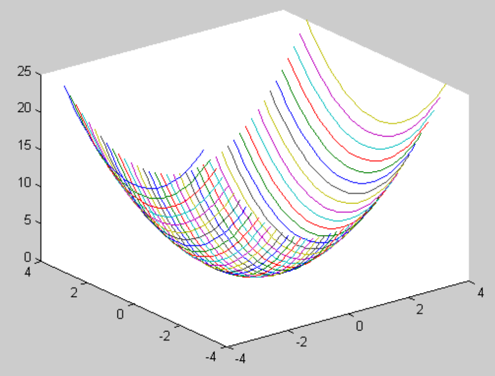
```
>> ciz8
>>
```

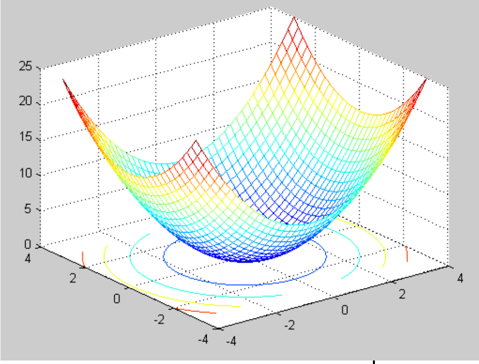


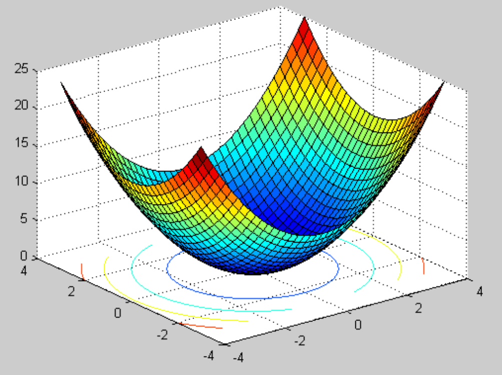
ciz9.m

```
% 3D Plots
clc; clear; close all; grid on;
% defines vectors x and y
vx = -4 :0.2: 4;
vy = -3 :0.2: 3;
% calculates the necessary grid
[x,y] = meshgrid(vx, vy);
% calculates z and avoids a null denominator adding 'eps`
z = x.*x + y.*y;

% generates the first figure using 'plot3'
figure
plot3(x,y,z)

% generates the second figure using 'meshc' to include the
% contour in the figure, and rotates the figure with 'view'
figure
meshc(x,y,z)

% generates the third 3D figure using 'surfc' to include the
% contour in the image, and also rotates the figure with 'view'
figure
surfc(x,y,z)
```

```
>> ciz9
>>
```







Line and Rectangle

* line function

```
>> axis([0 10 0 10])
>> x = [1 2];
>> y = [5 8];
>> line(x,y);
>> line(x,y,'Color','r')
```

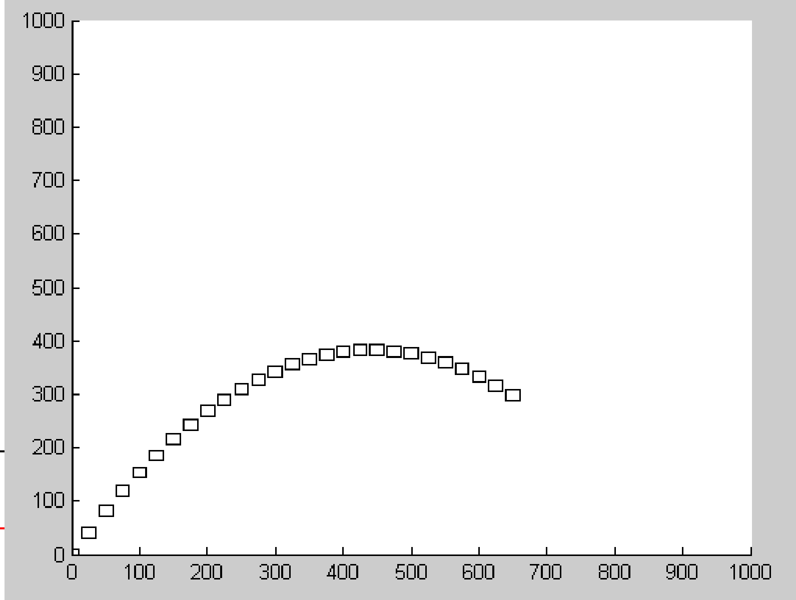
* rectangle function

```
>> axis([-3 3 -3 3])
>> rectangle('Position',[-1,-1,1,2])
>> rectangle('Position',[0,0,1,2],  ...
             'Curvature',[0.8 0.5], ...
             'FaceColor','r')
```

Basic Animation

ciz10.m

```
% Projectile Motion
hold on;
v0 = 100.0;            % initial velocity (m/s)
theta = 60 * pi/180;   % initial angle
g = 9.8;               % gravitational acceleration
t = 0.0;               % time
w = 20;                % width of the object
axis([0 1000 0 1000]); %

while t<=18
   x = v0*cos(theta)*t;
   y = v0*sin(theta)*t-0.5*g*t*t;
   r = rectangle('Position', ...
                [x-w/2,y-w/2,w,w]);
   t = t + 0.5;
   pause(0.5);
   delete(r);
end
```



References

[1]. http://www.mathworks.com/products/matlab

[2]. Numerical Methods in Engineering with MATLAB,
J. Kiusalaas, Cambridge University Press (2005)

[3]. Numerical Methods for Engineers, 6th Ed.
S.C. Chapra, Mc Graw Hill (2010)