Signal Analysis with MATLAB

Dr. Cahit Karakuş

Matlab Programming Structures

- **Loops**

```
for i=1:100
    sum = sum+i;
end
```

- Goes round the for loop 100 times, starting at i=1 and finishing at i=100

```
i=1;
while i<=100
    sum = sum+i;
    i = i+1;
end
```

- Similar, but uses a while loop instead of a for loop

- **Decisions**

```
if i==5
    a = i*2;
else
    a = i*4;
end
```

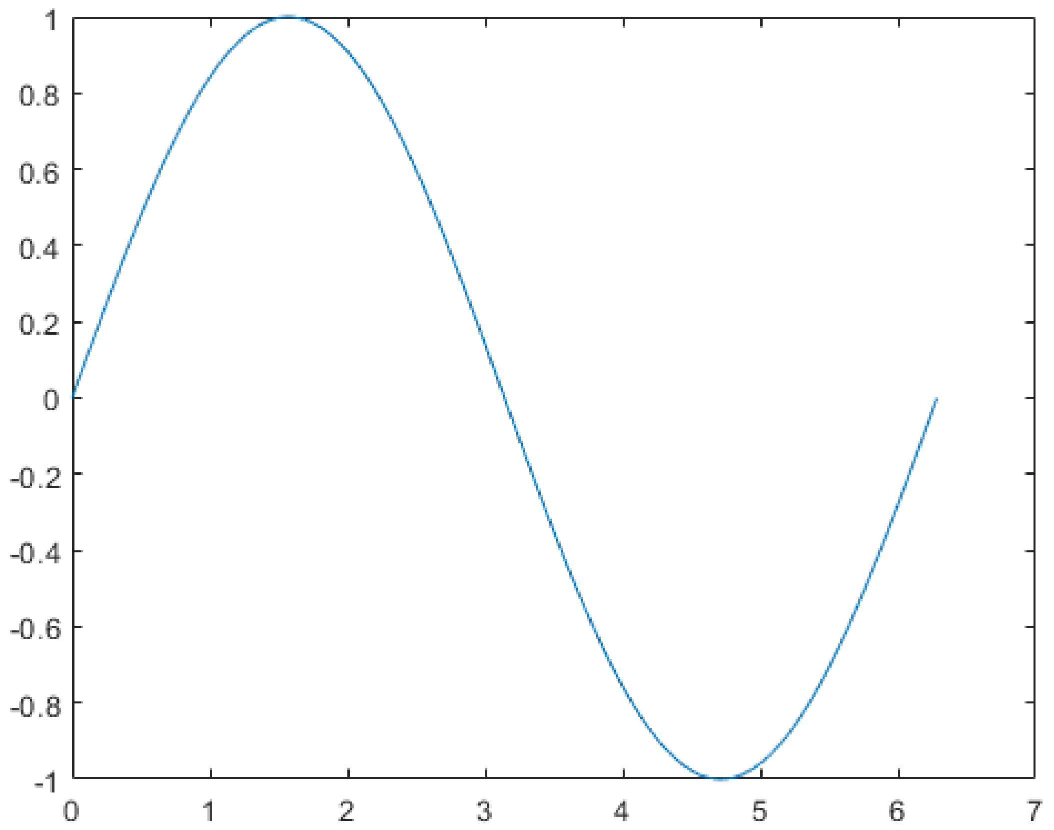
- Executes whichever branch is appropriate depending on test

```
switch i
case 5
    a = i*2;
otherwise
    a = i*4;
end
```

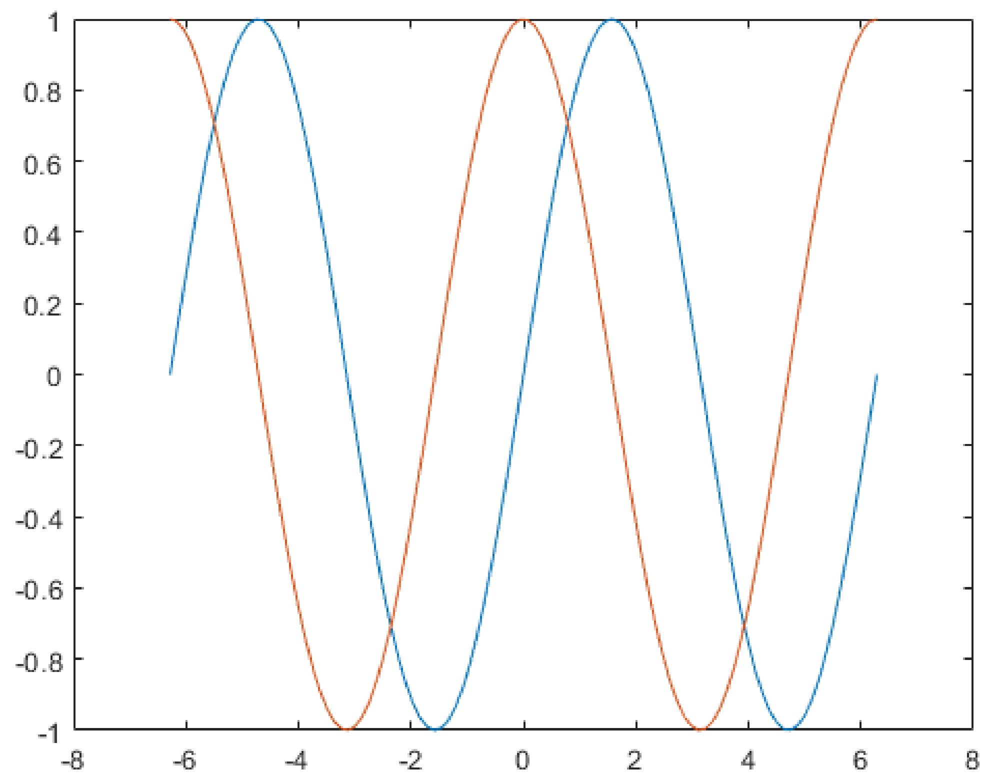
- Similar, but uses a switch

2-D Ploting

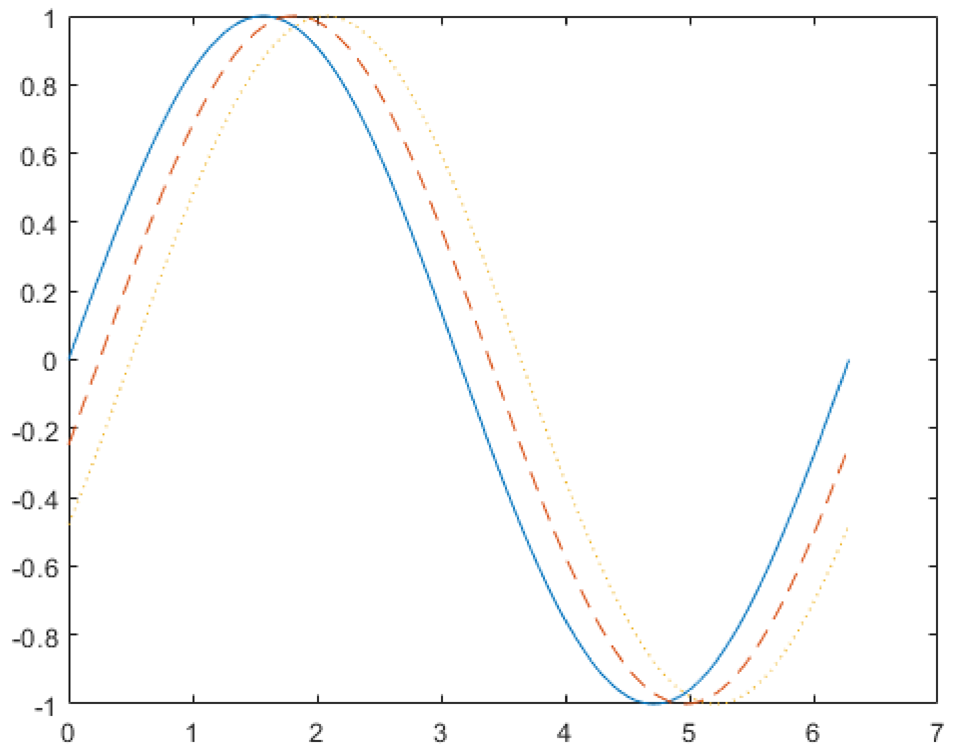
```
x = 0:pi/100:2*pi;
y = sin(x);
plot(x,y)
```



```
x = linspace(-2*pi,2*pi);
y1 = sin(x);
y2 = cos(x);

figure
plot(x,y1,x,y2)
```




```
x = 0:pi/100:2*pi;
y1 = sin(x);
y2 = sin(x-0.25);
y3 = sin(x-0.5);

figure
plot(x,y1,x,y2,'--',x,y3,':')
```

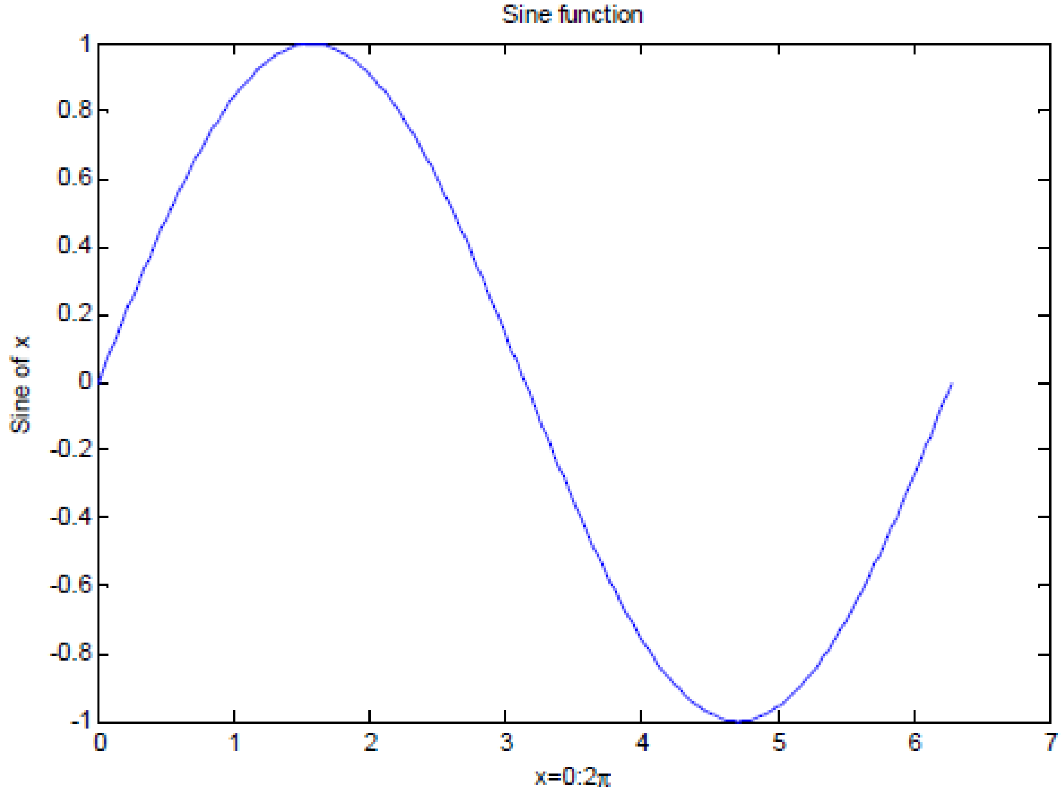


Basic plotting in MATLAB

MATLAB has an excellent set of graphic tools. Plotting a given data set or the results of computation is possible with very few commands

The MATLAB command to plot a graph is plot(x,y), e.g.

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



MATLAB enables you to add axis Labels and titles, e.g.

```
>> xlabel('x=0:2\pi');
>> ylabel('Sine of x');
>> tile('Sine function')
```

Nyquist Sampling Theorem

For lossless digitization, the sampling rate should be at least twice the maximum frequency response. Sayısal iletim ortamının performansını belirler.

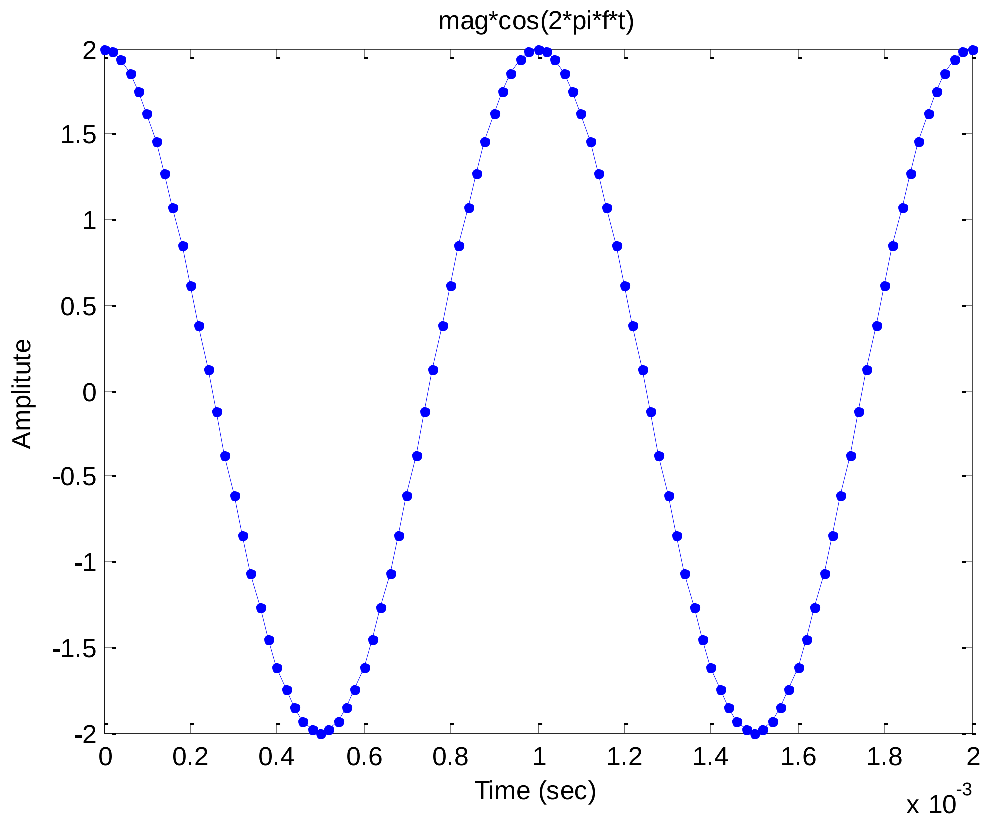
- In mathematical terms:

$$f_s \geq 2{*}f_m$$

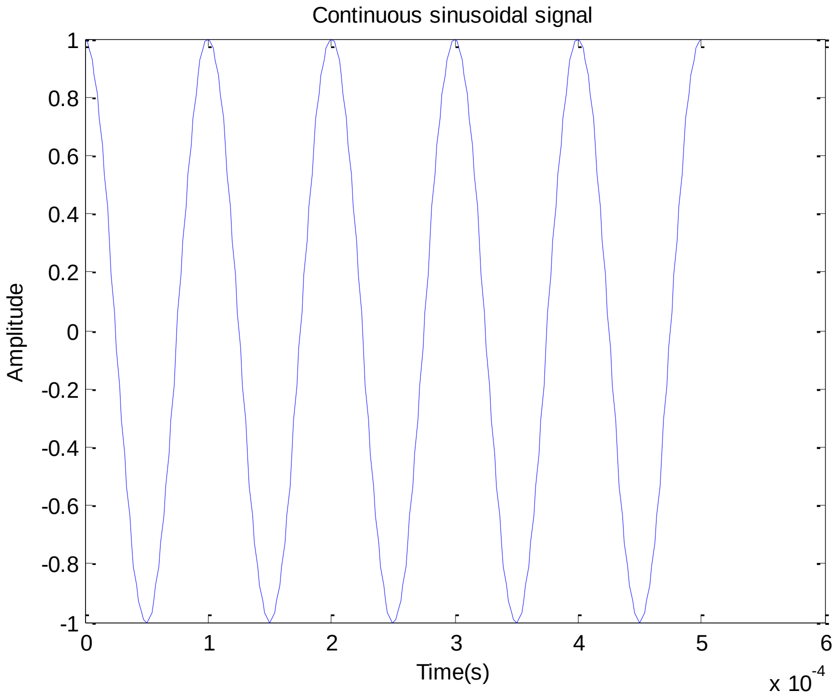
$$f_s \geq 2{*}B$$

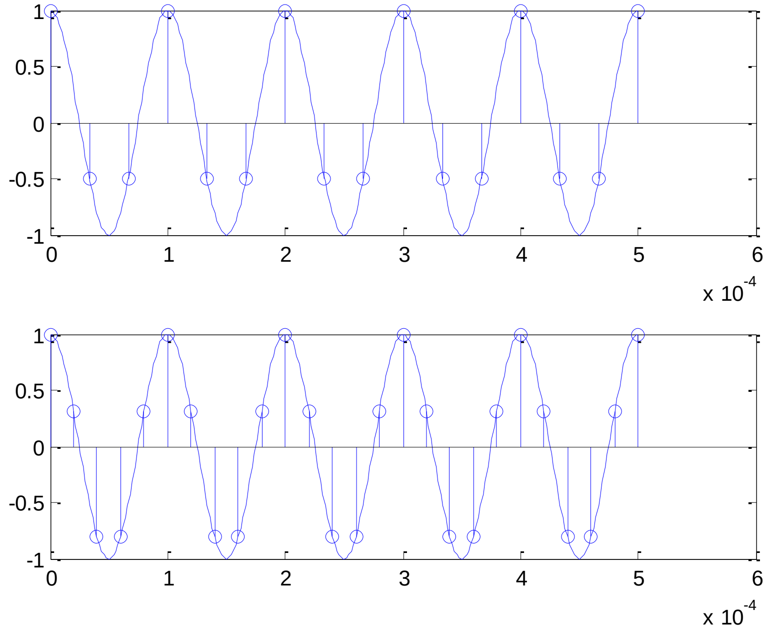
- where f_s is sampling frequency and f_m is the maximum frequency in the signal; B is the bandwidth. Birimleri Hz=1/sec. Frekans bir saniyedeki titreşim sayısıdır.

```
clear all
  close all
 mag  = 2;     % magnitude (arbitrary units)
 f = 1000;     % frequency in Hz
 Ps=50;        % number of sampling on a periot
 samp = f*Ps;   % sampling rate in Hz
 M=2;  %   Displaying number of period
 del=1/samp;


 t       = 0:del:M*Ps*del;  % time
 N       = length(t)
 x = mag*cos(2*pi*f*t);    % the signal equation
 figure
 plot(t,x,'.-');
 xlabel('Time (sec)');
 ylabel('Amplitute');
 title('mag*cos(2*pi*f*t)')
```

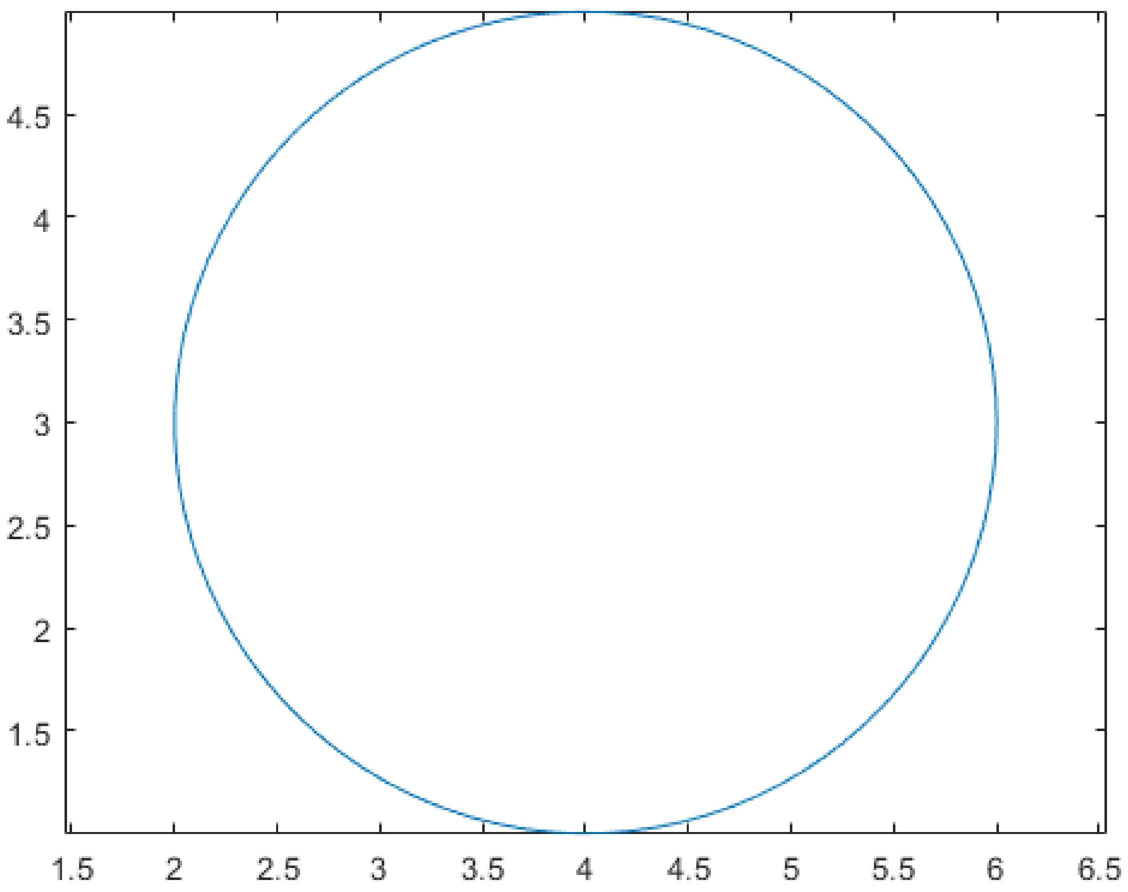


```
clear all
close all
 fs=500e3; %Very high sampling rate 500 kHz
f=10e3; %Frequency of sinusoid
nCyl=5; %generate five cycles of sinusoid
t=0:1/fs:nCyl*1/f; %time index
x=cos(2*pi*f*t);

figure, plot(t,x)
title('Continuous sinusoidal signal');
xlabel('Time(s)');
ylabel('Amplitude');


fs1=30e3; %30kHz sampling rate
t1=0:1/fs1:nCyl*1/f; %time index
x1=cos(2*pi*f*t1);

fs2=50e3; %50kHz sampling rate
t2=0:1/fs2:nCyl*1/f; %time index
x2=cos(2*pi*f*t2);

figure
subplot(2,1,1);
plot(t,x);
hold on;
stem(t1,x1);
subplot(2,1,2);
plot(t,x);
hold on;
stem(t2,x2);
```





```
r = 2;
xc = 4;
yc = 3;

theta = linspace(0,2*pi);
x = r*cos(theta) + xc;
y = r*sin(theta) + yc;
plot(x,y)
axis equal
```



Line Style	Description
-	Solid line (default)
--	Dashed line
:	Dotted line
-.	Dash-dot line

Color	Description
y	yellow
m	magenta
c	cyan
r	red
g	green
b	blue
w	white
k	black

Marker	Description
o	Circle
+	Plus sign
*	Asterisk
.	Point
x	Cross
s	Square
d	Diamond
^	Upward-pointing triangle
v	Downward-pointing triangle
>	Right-pointing triangle
<	Left-pointing triangle
p	Pentagram
h	Hexagram

Symbolic Expressions

Manipulating symbolic expressions

- numden()
- expand()
- factor()
- collect()
- simplify()
- simple()
- poly2sym()

Gradient of Function

Gradient vector of scalar function

The *gradient* of a function of two variables, $F(x, y)$, is defined as

$$\nabla F = \frac{\partial F}{\partial x}\hat{i} + \frac{\partial F}{\partial y}\hat{j}$$

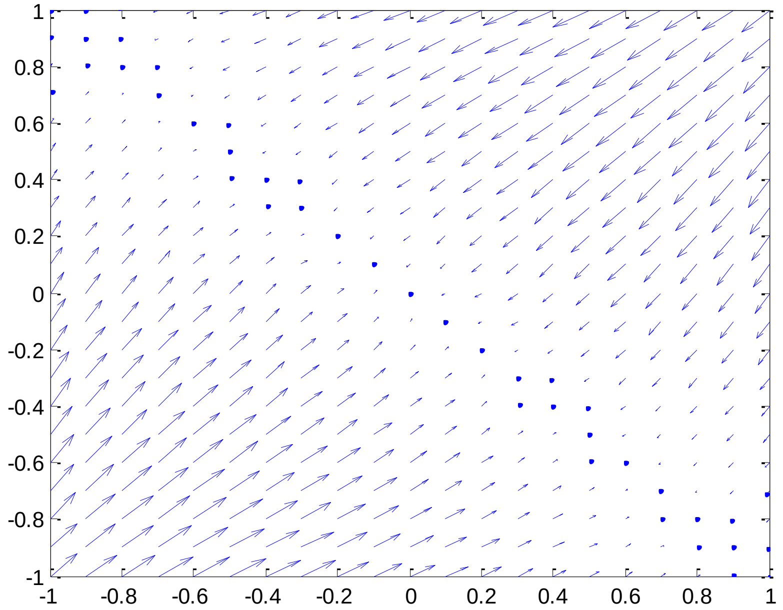
```
clear all
close all

syms x y z
f = 2*y*z*sin(x) + 3*x*sin(z)*cos(y);
gradient(f, [x, y, z])
```

Answer:
fx=2*y*z*cos(x) + 3*cos(y)*sin(z)
fy=2*z*sin(x) - 3*x*sin(y)*sin(z)
fz=2*y*sin(x) + 3*x*cos(y)*cos(z)

```
clear all
close all

syms x y
f = -(sin(x) + sin(y))^2;
g = gradient(f,[x,y])
```

Answaer
g =
 -2*cos(x)*(sin(x) + sin(y))
 -2*cos(y)*(sin(x) + sin(y))

```
[X, Y] = meshgrid(-1:.1:1,-1:.1:1);
G1 = subs(g(1),[x y],{X,Y});
G2 = subs(g(2),[x y],{X,Y});
quiver(X,Y,G1,G2)
```

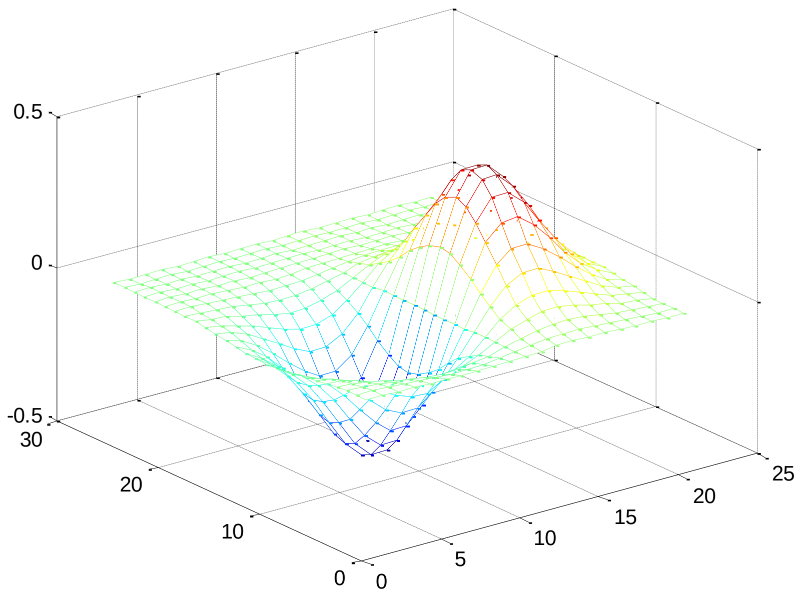


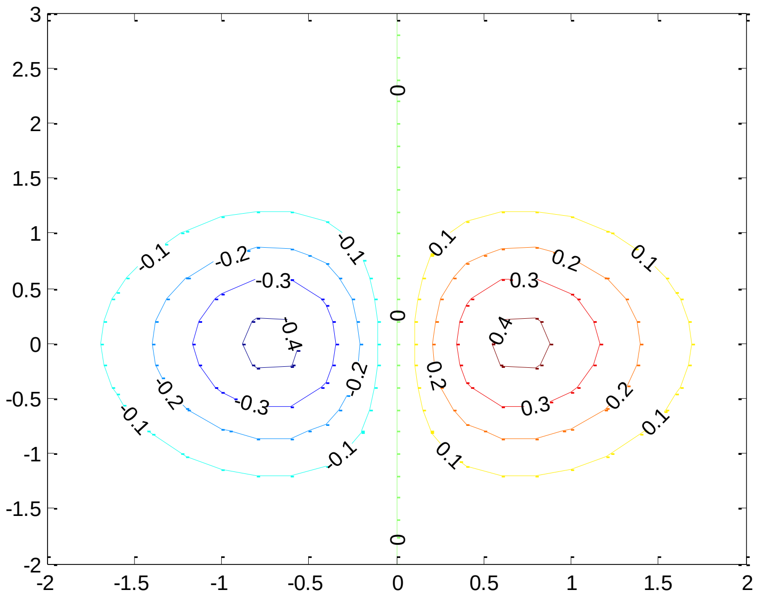
Not: subs(s,old,new) returns a copy of s, replacing all occurrences of old with new, and then evaluates s.

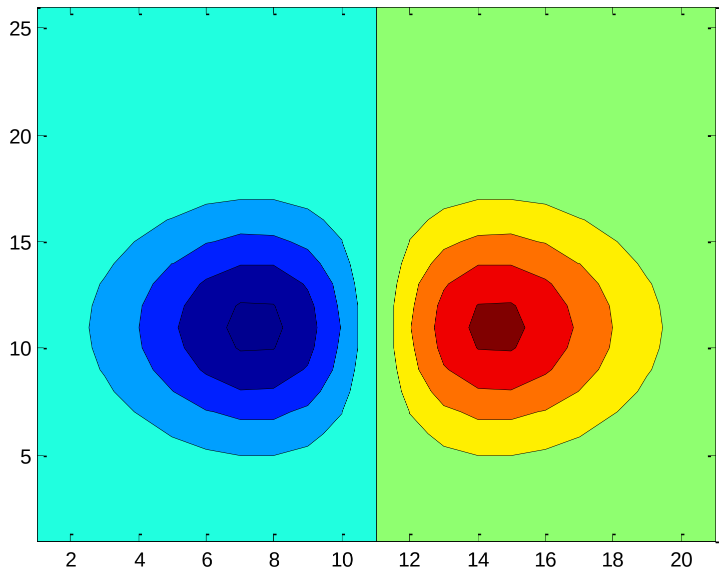
```
syms a b
subs(a + b, a, 4)
ans = b + 4
```

3-D Imaging

```
clear all
close all

x = -2:0.2:2;
y = -2:0.2:3;
[X,Y] = meshgrid(x,y);
Z = X.*exp(-X.^2-Y.^2);

figure, mesh(Z)
figure, contour(X,Y,Z)
figure, contourf(Z)
```

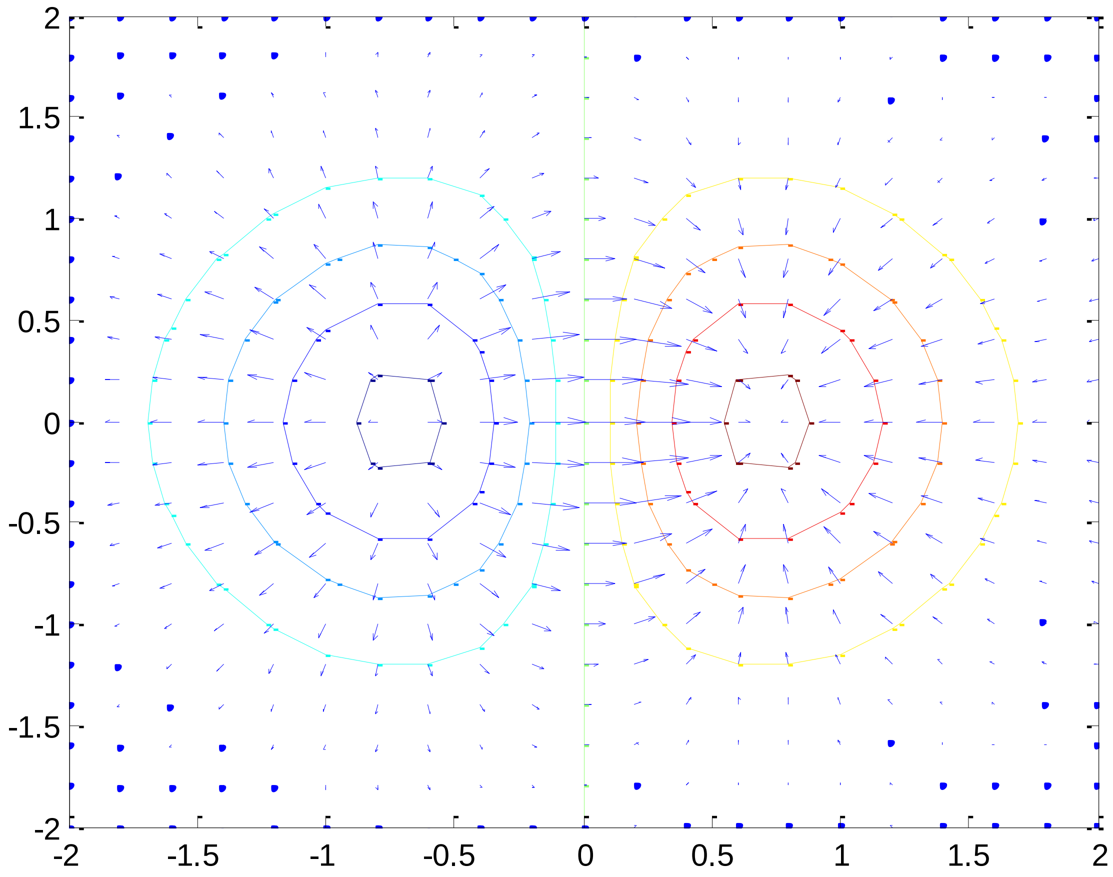






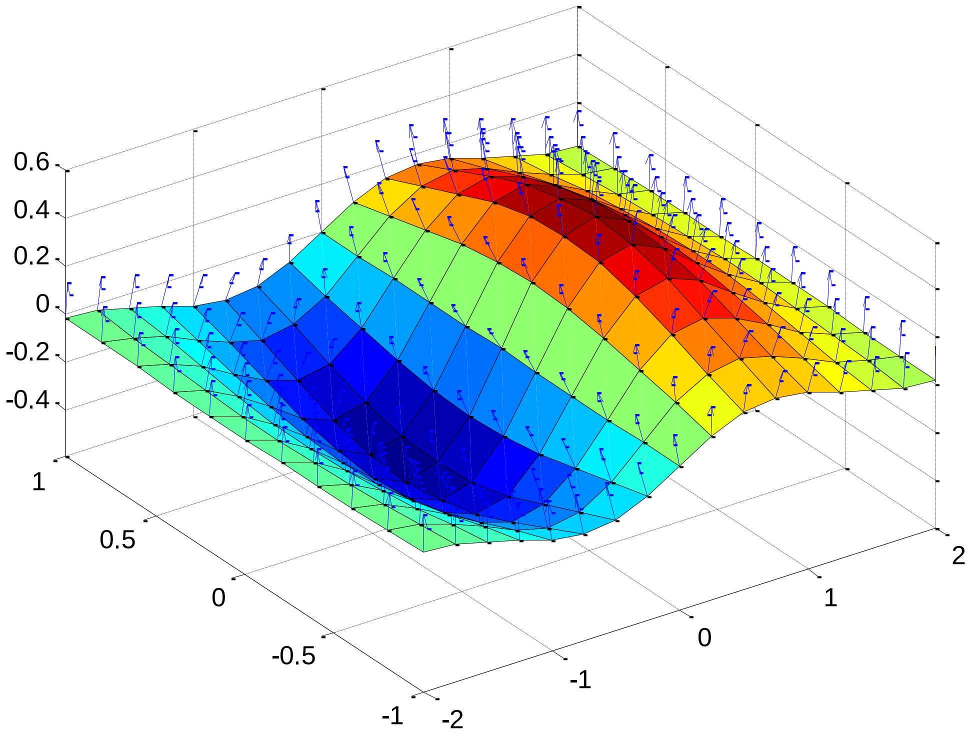
Plot the gradient of the function $z = xe^{-x2-y2}$

```
clear all
close all

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

figure
contour(X,Y,Z)
hold on
quiver(X,Y,DX,DY)
hold off
```

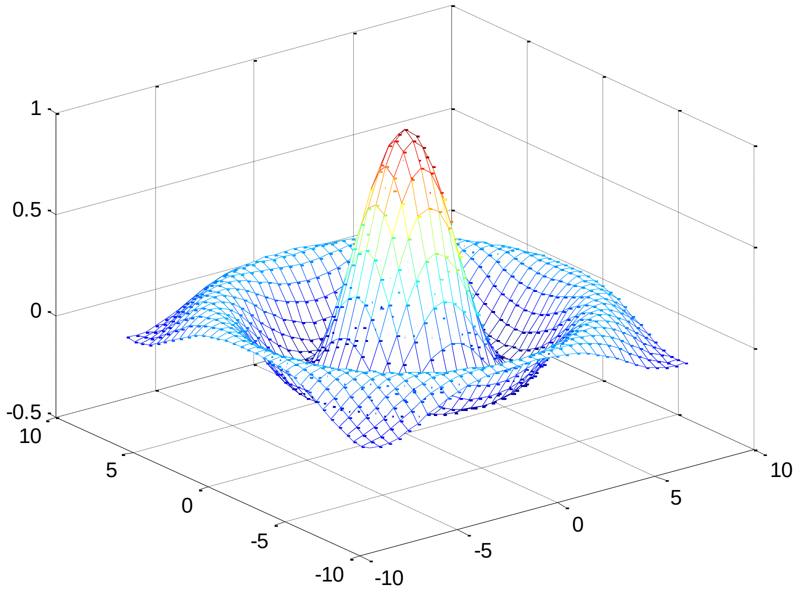


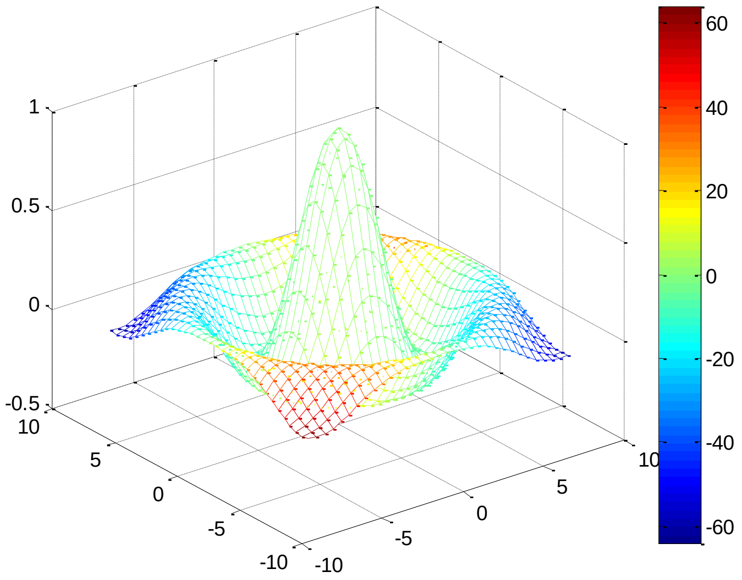
Plot the gradient of the function $z=xe^{-x^2-y^2}$

```
clear all
close all

[X,Y] = meshgrid(-2:0.25:2,-1:0.2:1);
Z = X.* exp(-X.^2 - Y.^2);
[U,V,W] = surfnorm(X,Y,Z);

figure
quiver3(X,Y,Z,U,V,W,0.5)

hold on
surf(X,Y,Z)
view(-35,45)
axis([-2 2 -1 1 -.6 .6])
hold off
```



```
clear all
close all

[X,Y] = meshgrid(-8:.5:8);
R = sqrt(X.^2 + Y.^2) + eps;
Z = sin(R)./R;
figure, mesh(X,Y,Z)


C = X.*Y;
figure, mesh(X,Y,Z,C), colorbar
```



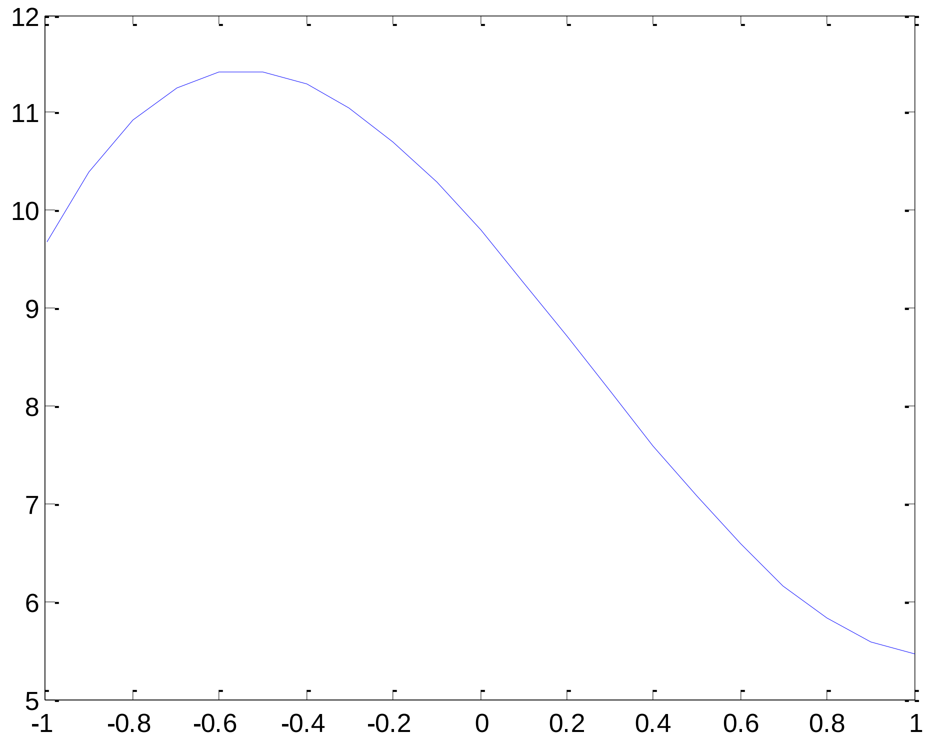


Polynomials and Roots

Polynomials and Roots

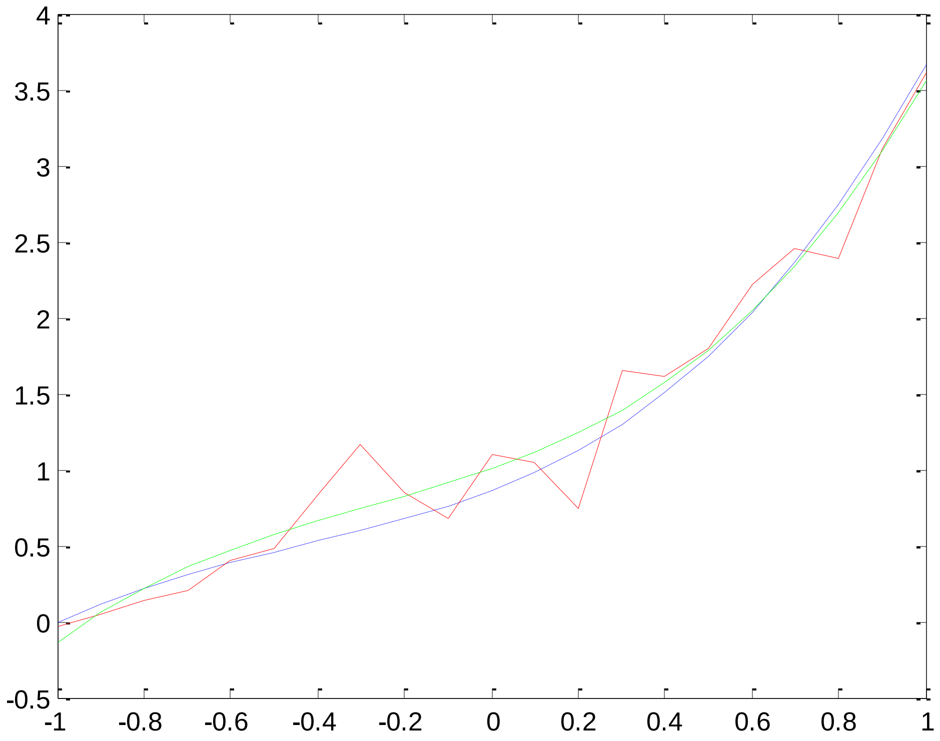
$$3x^3 - 2.23x^2 - 5.1x + 9.8$$

```
clear all
close all

c = [ 3 -2.23 -5.1 9.8 ]
 x = -1:0.1:1; % define range for plotting
 y = polyval(c,x); % compute samples
plot(x,y) % make the plot
r = roots(c)

cr = poly(r) % defines (x-r(1))*(x-r(2))*...
norm(c(1)*cr-c) % how far from original polynomial?

r =

  -1.5985 + 0.0000i
   1.1709 + 0.8200i
   1.1709 - 0.8200i

cr =    1.0000   -0.7433   -1.7000    3.2667
ans =   1.1374e-14
```



Polynomials and Roots

```
clear all
close all

 c3 = [0.74 0.97 1.1 0.86]; % data source
 x = -1:0.1:1; % sample range for x
 y = polyval(c3,x); % sample cubic at x
 noise = randn(1,size(y,2)); % random noise of same size as y
 noise = noise/norm(noise); % normalize the noise
ey = y + noise; % make noisy data
ec = polyfit(x,ey,3); % fit noisy data with cubic

 plot(x,y,'b--') % exact polynomial
 hold on
 plot(x,ey,'r') % noisy data
 hold on
 fy = polyval(ec,x) % sample fitted polynomial
 plot(x,fy,'g') % reconstructed data source
```



Integral – Derivative - Limit

Finding the integral of a function

Mathematical Operation	**MATLAB Command**
$\int x^n \, dx = \frac{x^{n+1}}{n+1}$	`int(x^n)` or `int(x^n,x)`
$\int_0^{\pi/2} \sin(2x) \, dx = 1$	`int(sin(2*x),0,pi/2)` or `int(sin(2*x),x,0,pi/2)`

$g = \cos(at+b)$
$\int g(t)dt = \frac{\sin(at+b)}{a}$
	`g = cos(a*t + b)`
	`int(g)` or
	`int(g,t)`

$\int J_1(z) \, dz = -J_0(z)$	`int(besselj(1,z))` or `int(besselj(1,z),z)`

$q = integral(fun,xmin,xmax),$ $$f(x)=e^{-x^2}(\ln x)^2$$

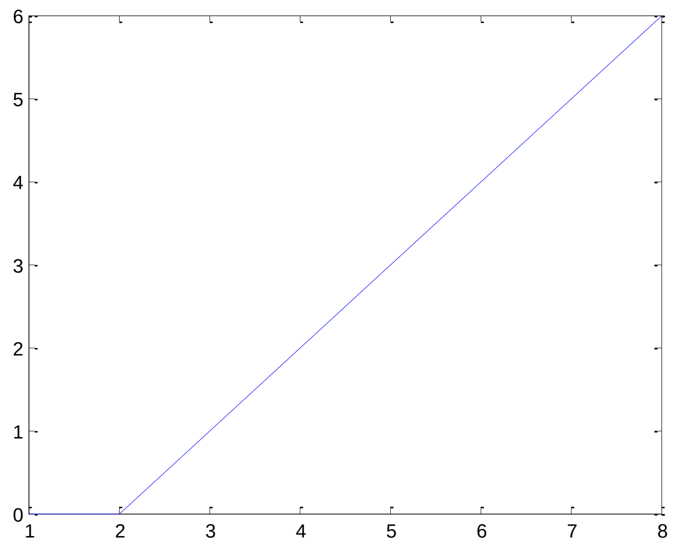
```
clear all
close all

fun = @(x) exp(-x.^2).*log(x).^2;

q = integral(fun,0,Inf)
```

q = 1.9475

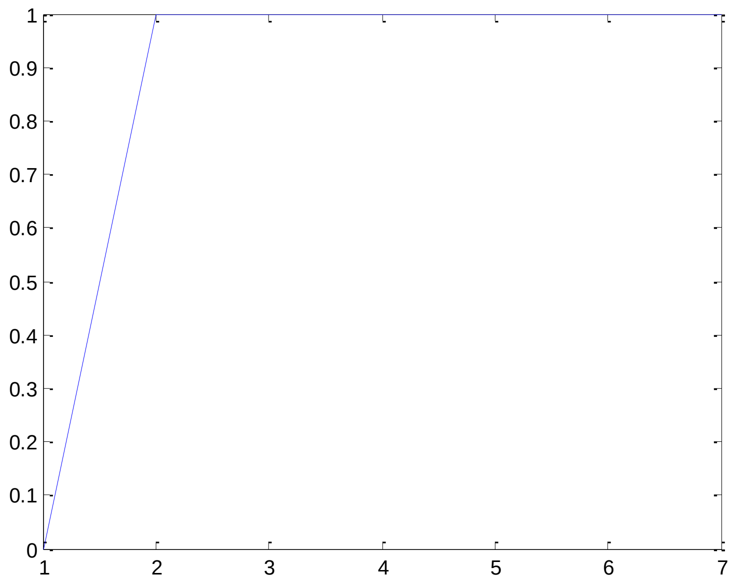
İntegral, derivative

```
clear all
close all

syms x
y1=int(4*x^2+3)
y2=int(4*x^2+3, x, -1, 3)

y1 = (4*x^3)/3 + 3*x
y2 = 26/3
```

```
clear all
close all

X = [0 0 1 2 3 4 5 6];
figure, plot(X)
Y = diff(X)
figure, plot(Y)
```




```
clear all
close all
syms x real
f = 1/x
y= diff(f)

y=-1/x^2
```



```
clear all
close all

syms x
f = cos(8*x)
g = sin(5*x)*exp(x)
h =(2*x^2+1)/(3*x)
a1=diff(f)
a2=diff(g)
a3=diff(h)
```

```
f =cos(8*x)
g = sin(5*x)*exp(x)
h =(2*x^2 + 1)/(3*x)

a1 = -8*sin(8*x)
a2 = 5*cos(5*x)*exp(x) + sin(5*x)*exp(x)
a3 = 4/3 - (2*x^2 + 1)/(3*x^2)
```

$$\frac{\partial f}{\partial x} = ?$$

```
clear all
close all

syms x y
f = sin(x*y)
diff(f,x)
```

```
Ans=y*cos(x*y)
```

```
clear all
close all

syms x a b c
S = simplify(sin(x)^2 + cos(x)^2)
S = simplify(exp(c*log(sqrt(a+b))))

S = 1
S =(a + b)^(c/2)
```

Finding the Limits of a function- f(x)

Mathematical Operation	MATLAB Command
$\lim_{x \to 0} f(x)$	`limit(f)`
$\lim_{x \to a} f(x)$	`limit(f,x,a)` or `limit(f,a)`
$\lim_{x \to a+} f(x)$	`limit(f,x,a,'right')`
$\lim_{x \to a-} f(x)$	`limit(f,x,a,'left')`

Evaluate: (a) $\lim_{x \to 0} \left(\frac{1}{x}\right)$ (b) $\lim_{x \to 0+} \left(\frac{1}{x}\right)$ (c) $\lim_{x \to 0-} \left(\frac{1}{x}\right)$

Solution of Differential Equations with MATLAB

$$d^2y/dx^2=\cos(2x)-y,\ y(0)=1, y'(0)=0.$$

```
clear all
close all

syms y(x)
Dy = diff(y);

ode = diff(y,x,2) == cos(2*x)-y;
cond1 = y(0) == 1;
cond2 = Dy(0) == 0;

conds = [cond1 cond2];
ySol(x) = dsolve(ode,conds);
ySol = simplify(ySol)


ySol(x) = 1 - (8*sin(x/2)^4)/3
```

Differential Equation	MATLAB® Commands

$\frac{dy}{dt} + 4y(t) = e^{-t},$
$y(0) = 1.$
	`syms y(t)`
	`ode = diff(y)+4*y == exp(-t);`
	`cond = y(0) == 1;`
	`ySol(t) = dsolve(ode,cond)`
	
	`ySol(t) =`
	`exp(-t)/3 + (2*exp(-4*t))/3`

$2x^2\frac{d^2y}{dx^2} + 3x\frac{dy}{dx} - y = 0.$	`syms y(x)`
	`ode = 2*x^2*diff(y,x,2)+3*x*diff(y,x)-y == 0;`
	`ySol(x) = dsolve(ode)`
	
	`ySol(x) =`
	`C2/(3*x) + C3*x^(1/2)`

The Airy equation.
$\frac{d^2y}{dx^2} = xy(x).$
	`syms y(x)`
	`ode = diff(y,x,2) == x*y;`
	`ySol(x) = dsolve(ode)`
	
	`ySol(x) =`
	`C1*airy(0,x) + C2*airy(2,x)`

Solve Differential Equations in Matrix Form

Solve differential equations in matrix form by using `dsolve`.

Consider this system of differential equations.

$$\frac{dx}{dt} = x + 2y + 1,$$
$$\frac{dy}{dt} = -x + y + t.$$

The matrix form of the system is

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 1 \\ t \end{bmatrix}.$$

Let

$$Y = \begin{bmatrix} x \\ y \end{bmatrix}, A = \begin{bmatrix} 1 & 2 \\ -1 & 1 \end{bmatrix}, B = \begin{bmatrix} 1 \\ t \end{bmatrix}.$$

The system is now $Y' = AY + B$.

```
syms x(t) y(t)
A = [1 2; -1 1];
B = [1; t];
Y = [x; y];
odes = diff(Y) == A*Y + B
```

```
odes(t) =
  diff(x(t), t) == x(t) + 2*y(t) + 1
   diff(y(t), t) == t - x(t) + y(t)
```

Solve the matrix equation using `dsolve`. Simplify the solution by using the `simplify` function.

```
[xSol(t), ySol(t)] = dsolve(odes);
xSol(t) = simplify(xSol(t))
ySol(t) = simplify(ySol(t))
```

Suppose we want to solve and plot the solution to the second order equation

$$y''(x) + 8y'(x) + 2y(x) = \cos(x); \qquad y(0) = 0,\ y'(0) = 1.$$

```
clear all
close all

eqn2 = 'D2y + 8*Dy + 2*y = cos(x)';
inits2 = 'y(0)=0, Dy(0)=12';
y=dsolve(eqn2,inits2,'x')
y1 = simplify(y)
```

```
y1 = cos(x)/65 + (8*sin(x))/65 - exp(x*(14^(1/2) - 4))/130 - exp(-
x*(14^(1/2) + 4))/130 +
(192*14^(1/2)*exp(x*(14^(1/2) - 4)))/455 - (192*14^(1/2)*exp(-
x*(14^(1/2) + 4)))/455
```

Suppose we want to solve and plot solutions to the system of three ordinary differential equations

$$
\begin{aligned}
x'(t) &= x(t) + 2y(t) - z(t) \\
y'(t) &= x(t) + z(t) \\
z'(t) &= 4x(t) - 4y(t) + 5z(t).
\end{aligned}
$$

```
clear all
close all

[x,y,z]=dsolve('Dx=x+2*y-z','Dy=x+z','Dz=4*x-4*y+5*z')


x = - (C3*exp(t))/2 - (C1*exp(2*t))/2 - (C2*exp(3*t))/4
y = (C3*exp(t))/2 + (C1*exp(2*t))/4 + (C2*exp(3*t))/4
 z = C3*exp(t) + C1*exp(2*t) + C2*exp(3*
```

Suppose we want to solve and plot solutions to the system of three ordinary differential equations

$$
\begin{aligned}
x'(t) &= x(t) + 2y(t) - z(t)\\
y'(t) &= x(t) + z(t)\\
z'(t) &= 4x(t) - 4y(t) + 5z(t).
\end{aligned}
$$

```
clear all
close all
inits='x(0)=1,y(0)=2,z(0)=3';
[x,y,z]=dsolve('Dx=x+2*y-z','Dy=x+z','Dz=4*x-4*y+5*z',inits)

t=linspace(0,.5,25);
xx=eval(vectorize(x));
yy=eval(vectorize(y));
zz=eval(vectorize(z));
plot(t, xx, t, yy, t, zz)
```

Symbolic Differential Equation Terms

y	•y
$\dfrac{dy}{dt}$	•Dy
$\dfrac{d^2 y}{dt^2}$	•D2y
$\dfrac{d^n y}{dt^n}$	•Dny

$$b_2 \frac{d^2 y}{dt^2} + b_1 \frac{dy}{dt} + b_0 y = A \sin at$$

$$y(0) = C_1 \qquad \text{and } y'(0) = C_2$$

```
y = dsolve(‘b2*D2y+b1*D1y+b0*y=A*sin(a*t)’,
 ‘y(0)=C1’, ‘Dy(0)=C2’)

ezplot(y, [t1 t2])
```

$$\frac{dy}{dt} + 2y = 12 \qquad\qquad y(0) = 10$$

```
y = dsolve('Dy + 2*y = 12', 'y(0)=10')


>> y =6+4*exp(-2*t)
>> ezplot(y, [0 3])
>> axis([0 3 0 10])
```

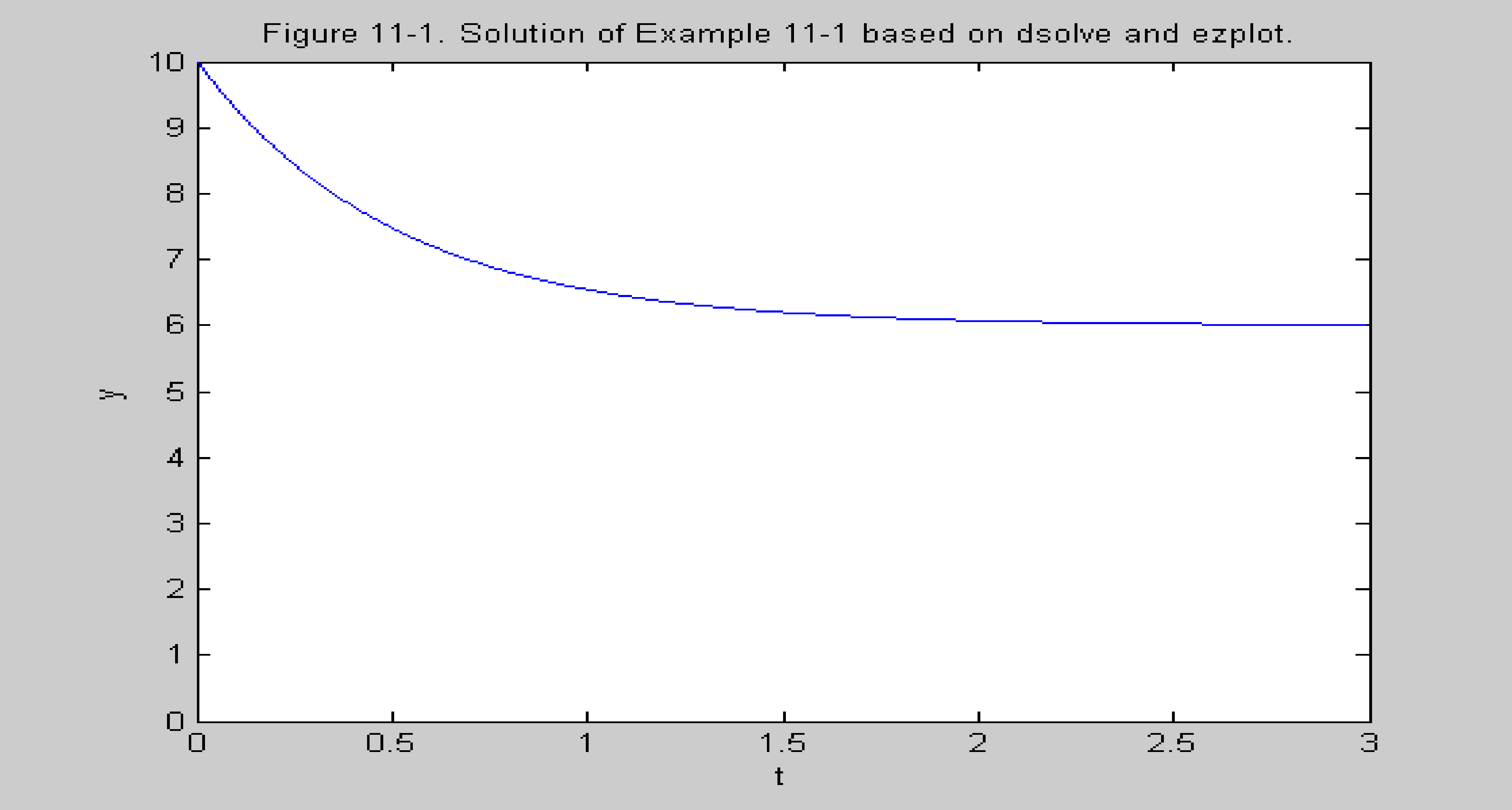


Figure 11-1. Solution of Example 11-1 based on dsolve and ezplot.
10
9
8
7
6
5
4
3
2
1
0
y
0
0.5
1
1.5
2
2.5
3
t

$$\frac{d^2y}{dt^2}+3\frac{dy}{dt}+2y=24$$

$$y(0)=10 \quad y'(0)=0$$

- >> y = dsolve('D2y + 3*Dy + 2*y = 24', 'y(0)=10', 'Dy(0)=0')

- y =
- 12+2*exp(-2*t)-4*exp(-t)

- >> ezplot(y, [0 6])

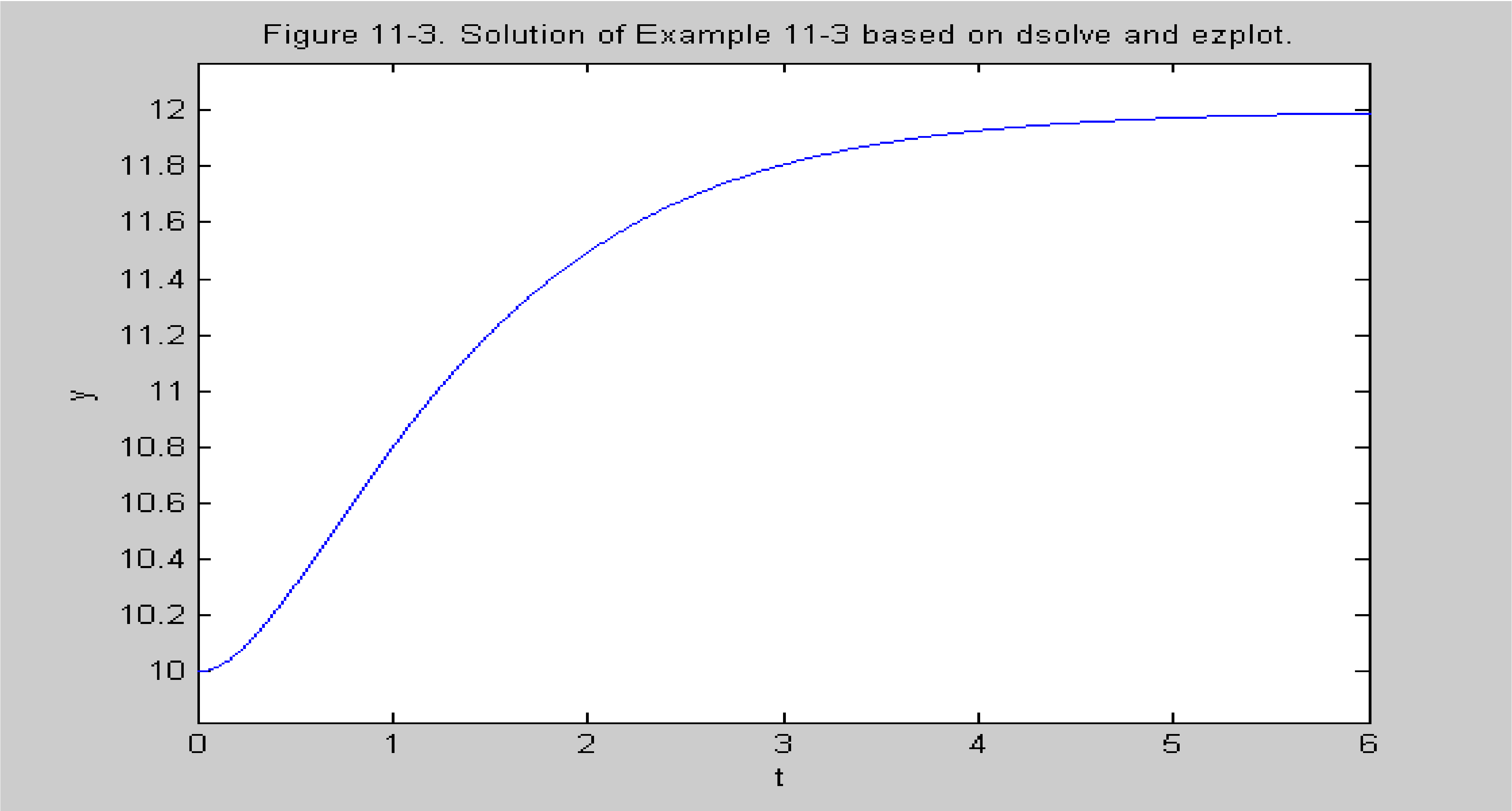
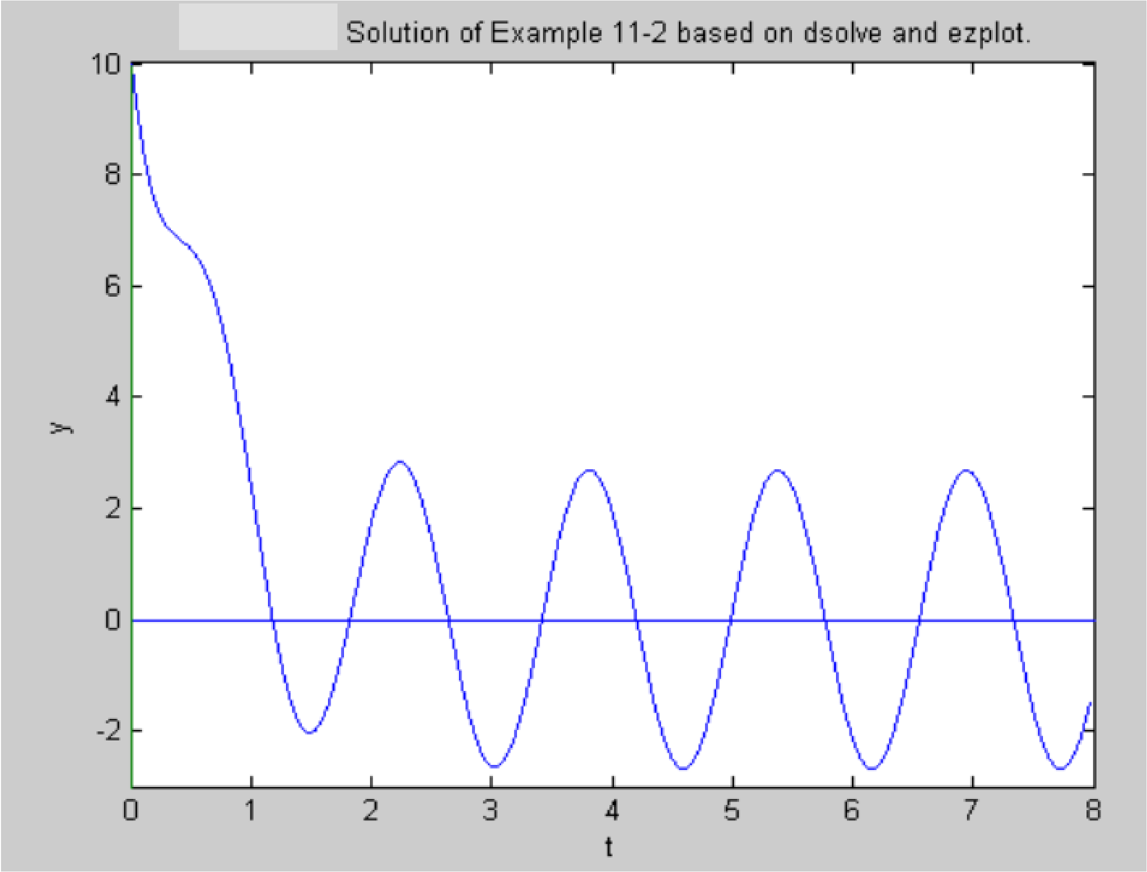


Figure 11-3. Solution of Example 11-3 based on dsolve and ezplot.
12
11.8
11.6
11.4
11.2
11
10.8
10.6
10.4
10.2
10
y
0
1
2
3
4
5
6
t

$$\frac{dy}{dt} + 2y = 12\sin 4t \qquad y(0) = 10$$

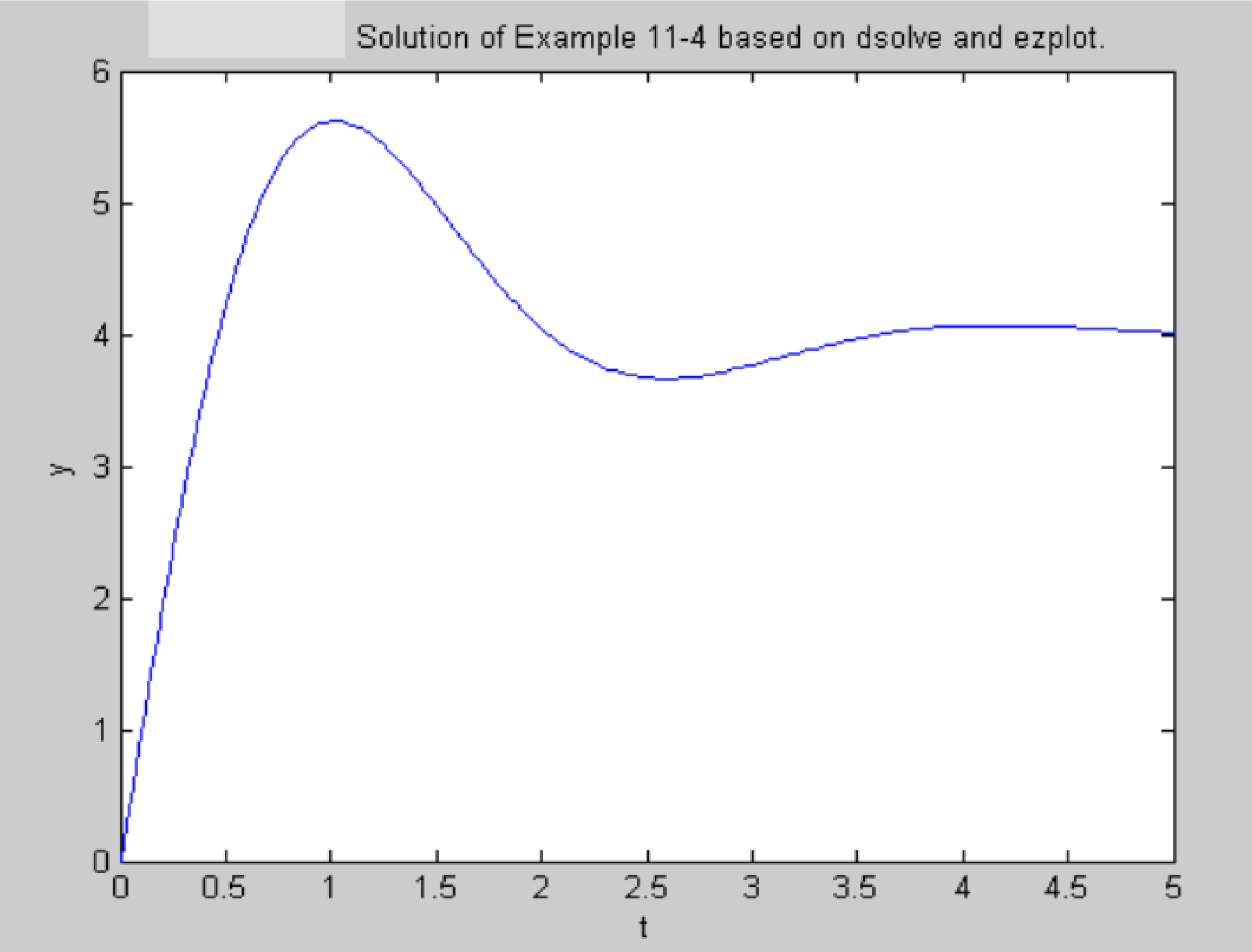
```
y = dsolve('Dy + 2*y = 12*sin(4*t)', 'y(0)=10')

ezplot(y, [0 8])
axis([0 8 -3 10])

y =  -12/5*cos(4*t)+6/5*sin(4*t)+62/5*exp(-2*t)
```



$$\frac{d^2y}{dt^2} + 2\frac{dy}{dt} + 5y = 20$$

$$y(0) = 0 \quad y'(0) = 10$$

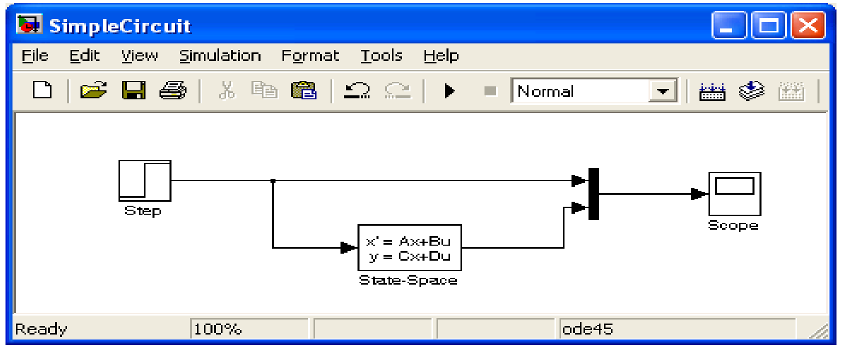
```
y = dsolve('D2y + 2*Dy + 5*y = 20',
    'y(0) = 0',  'Dy(0) = 10')
ezplot(y, [0 5])
y =   4+3*exp(-t)*sin(2*t)-4*exp(-t)*cos(2*t)
```

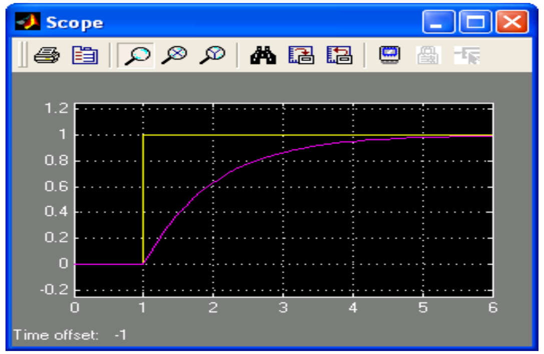


Simulink

Introduction to Simulink

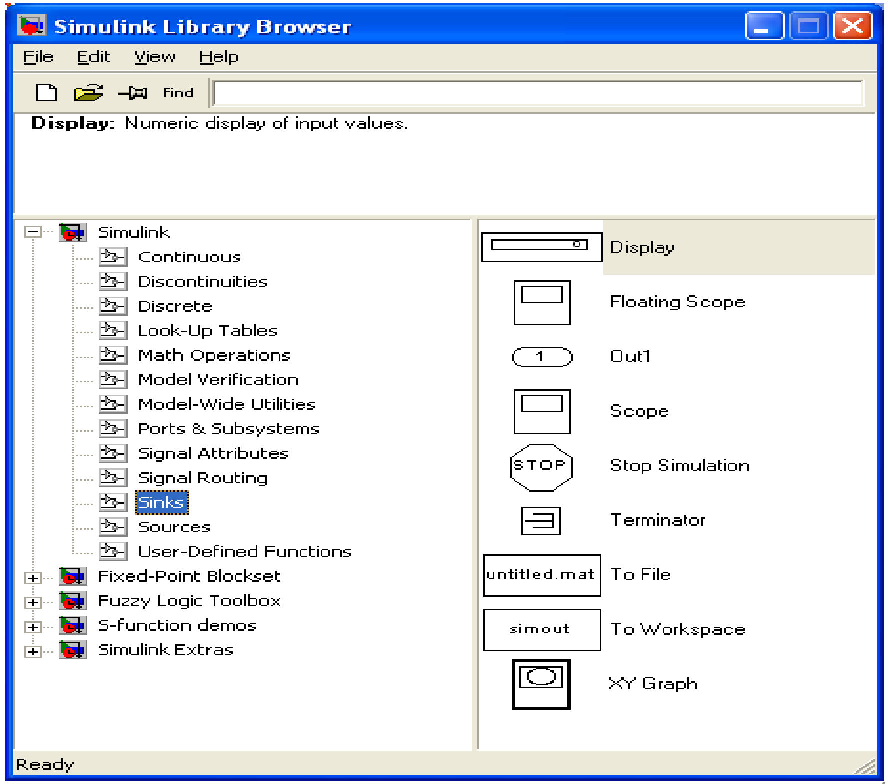
- Simulink is a graphical, “drag and drop” environment for building simple and complex signal and system dynamic simulations.
- It allows users to concentrate on the structure of the problem, rather than having to worry (too much) about a programming language.
- The parameters of each signal and system block is configured by the user (right click on block)
- Signals and systems are simulated over a particular time.

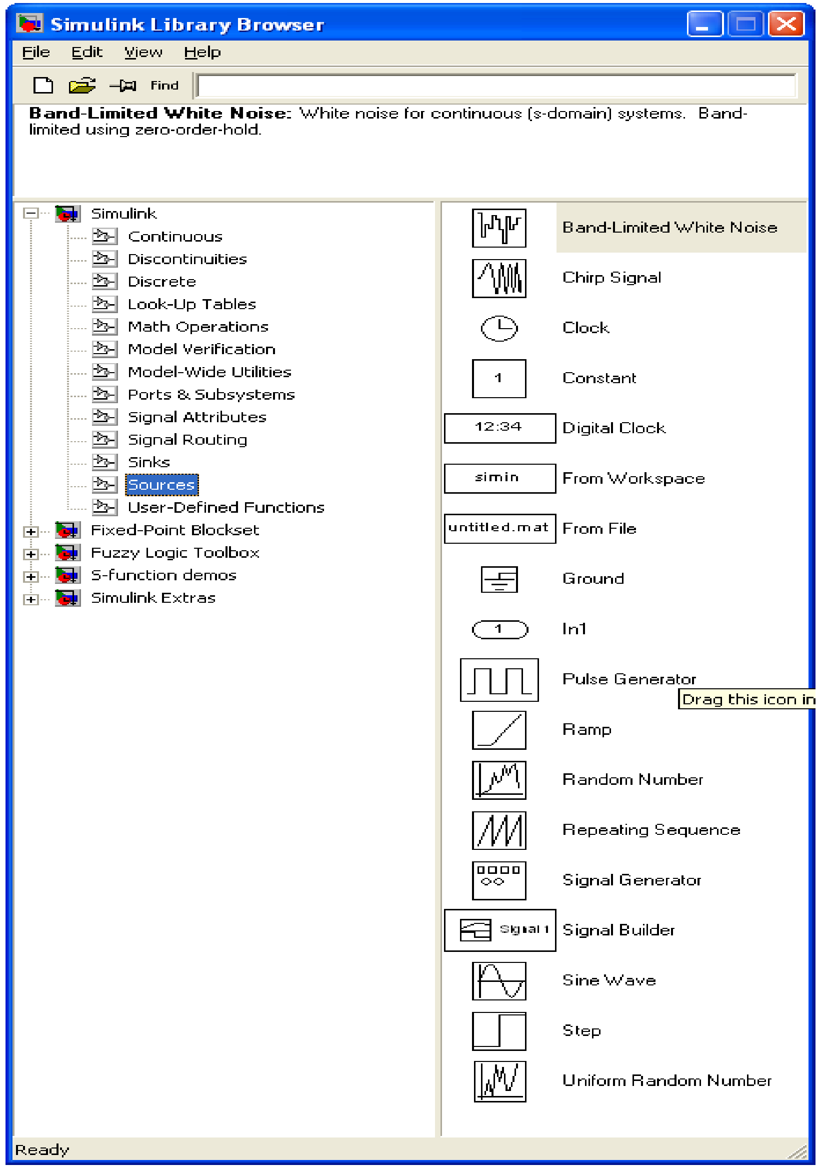




Signals in Simulink

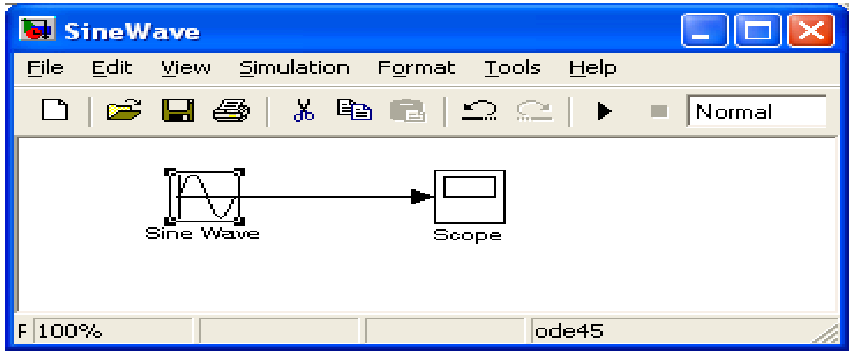
- Two main libraries for manipulating signals in Simulink:
- **Sources**: generate a signal
- **Sink**: display, read or store a signal

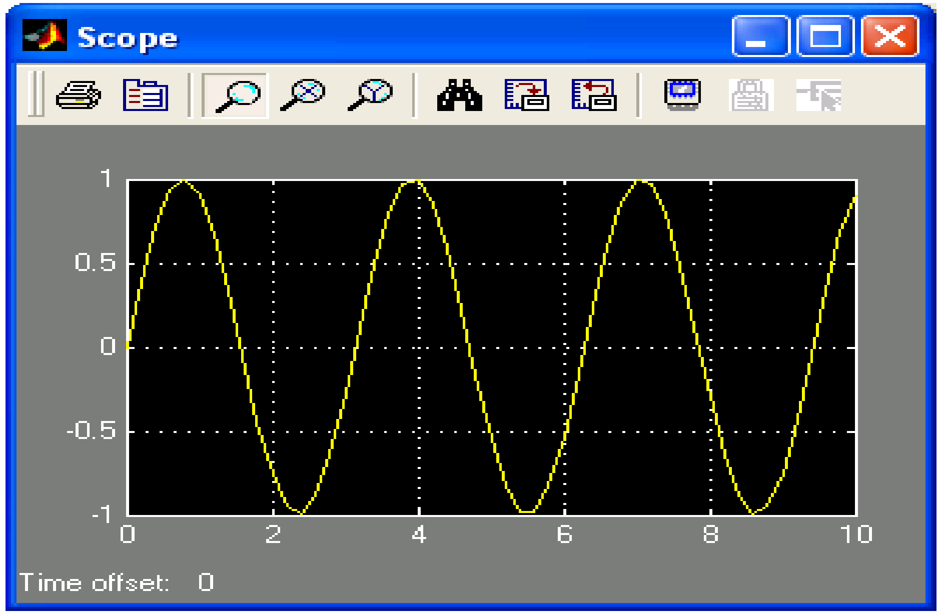




Example: Generate and View a Signal

- Copy “sine wave” source and “scope” sink onto a new Simulink work space and connect.
- Set sine wave parameters modify to 2 rad/sec
- Run the simulation:
- Simulation - Start
- Open the scope and leave open while you change parameters (sin or simulation parameters) and re-run





Summary

- This lecture has looked at signals:
- Power and energy
- Signal transformations
  - Time shift
  - Periodic
  - Even and odd signals
- Exponential and sinusoidal signals
- Unit impulse and step functions

- Matlab and Simulink are complementary environments for producing and analysing continuous and discrete signals.
- This will require some effort to learn the programming syntax and style!

Usage Notes

- These slides were gathered from the presentations published on the internet. I would like to thank who prepared slides and documents.
- Also, these slides are made publicly available on the web for anyone to use
- If you choose to use them, I ask that you alert me of any mistakes which were made and allow me the option of incorporating such changes (with an acknowledgment) in my set of slides.

Sincerely,

Dr. Cahit Karakuş

cahitkarakus@esenyurt.edu.tr