

**Sinyaller ve Sistemler**

**“Signal and Systems with MATLAB”**

**Dr. Cahit Karakuş, 2020**



**Latex**

<code>^{ }</code>	Superscript	<code>'text^{superscript}'</code>
<code>_{ }</code>	Subscript	<code>'text_{subscript}'</code>
<code>\bf</code>	Bold font	<code>'\bf text'</code>
<code>\it</code>	Italic font	<code>'\it text'</code>
<code>\sl</code>	Oblique font (usually the same as italic font)	<code>'\sl text'</code>
<code>\rm</code>	Normal font	<code>'\rm text'</code>

Character Sequence	Symbol	Character Sequence	Symbol	Character Sequence	Symbol
<code>\alpha</code>	$\alpha$	<code>\upsilon</code>	$\upsilon$	<code>\sim</code>	$\sim$
<code>\angle</code>	$\angle$	<code>\phi</code>	$\phi$	<code>\leq</code>	$\leq$
<code>\ast</code>	$*$	<code>\chi</code>	$\chi$	<code>\infty</code>	$\infty$
<code>\beta</code>	$\beta$	<code>\psi</code>	$\psi$	<code>\clubsuit</code>	$\clubsuit$
<code>\gamma</code>	$\gamma$	<code>\omega</code>	$\omega$	<code>\diamondsuit</code>	$\diamondsuit$
<code>\delta</code>	$\delta$	<code>\Gamma</code>	$\Gamma$	<code>\heartsuit</code>	$\heartsuit$
<code>\epsilon</code>	$\epsilon$	<code>\Delta</code>	$\Delta$	<code>\spadesuit</code>	$\spadesuit$
<code>\zeta</code>	$\zeta$	<code>\Theta</code>	$\Theta$	<code>\leftrightarrow</code>	$\leftrightarrow$
<code>\eta</code>	$\eta$	<code>\Lambda</code>	$\Lambda$	<code>\leftarrow</code>	$\leftarrow$
<code>\theta</code>	$\theta$	<code>\Xi</code>	$\Xi$	<code>\Leftarrow</code>	$\Leftarrow$
<code>\vartheta</code>	$\vartheta$	<code>\Pi</code>	$\Pi$	<code>\uparrow</code>	$\uparrow$
<code>\iota</code>	$\iota$	<code>\Sigma</code>	$\Sigma$	<code>\rightarrow</code>	$\rightarrow$
<code>\kappa</code>	$\kappa$	<code>\Upsilon</code>	$\Upsilon$	<code>\Rightarrow</code>	$\Rightarrow$
<code>\lambda</code>	$\lambda$	<code>\Phi</code>	$\Phi$	<code>\downarrow</code>	$\downarrow$
<code>\mu</code>	$\mu$	<code>\Psi</code>	$\Psi$	<code>\circ</code>	$\circ$
<code>\nu</code>	$\nu$	<code>\Omega</code>	$\Omega$	<code>\pm</code>	$\pm$
<code>\xi</code>	$\xi$	<code>\forall</code>	$\forall$	<code>\geq</code>	$\geq$
<code>\pi</code>	$\pi$	<code>\exists</code>	$\exists$	<code>\propto</code>	$\propto$
<code>\rho</code>	$\rho$	<code>\equiv</code>	$\equiv$	<code>\partial</code>	$\partial$
<code>\sigma</code>	$\sigma$	<code>\cong</code>	$\cong$	<code>\bullet</code>	$\bullet$
<code>\varsigma</code>	$\varsigma$	<code>\approx</code>	$\approx$	<code>\div</code>	$\div$
<code>\tau</code>	$\tau$	<code>\Re</code>	$\Re$	<code>\neq</code>	$\neq$
<code>\equiv</code>	$\equiv$	<code>\oplus</code>	$\oplus$	<code>\aleph</code>	$\aleph$
<code>\Im</code>	$\Im$	<code>\cup</code>	$\cup$	<code>\wp</code>	$\wp$

<code>\otimes</code>	$\otimes$	<code>\subseteq</code>	$\subseteq$	<code>\oslash</code>	$\oslash$
<code>\cap</code>	$\cap$	<code>\in</code>	$\in$	<code>\supseteq</code>	$\supseteq$
<code>\supset</code>	$\supset$	<code>\lceil</code>	$\lceil$	<code>\subset</code>	$\subset$
<code>\int</code>	$\int$	<code>\cdot</code>	$\cdot$	<code>\o</code>	$\o$
<code>\rfloor</code>	$\rfloor$	<code>\neg</code>	$\neg$	<code>\nabla</code>	$\nabla$
<code>\lfloor</code>	$\lfloor$	<code>\times</code>	$\times$	<code>\ldots</code>	$\dots$
<code>\perp</code>	$\perp$	<code>\surd</code>	$\surd$	<code>\prime</code>	$\prime$
<code>\wedge</code>	$\wedge$	<code>\varpi</code>	$\varpi$	<code>\emptyset</code>	$\emptyset$
<code>\rceil</code>	$\rceil$	<code>\rangle</code>	$\rangle$	<code>\mid</code>	$\mid$
<code>\vee</code>	$\vee$	<code>\langle</code>	$\langle$	<code>\copyright</code>	$\copyright$



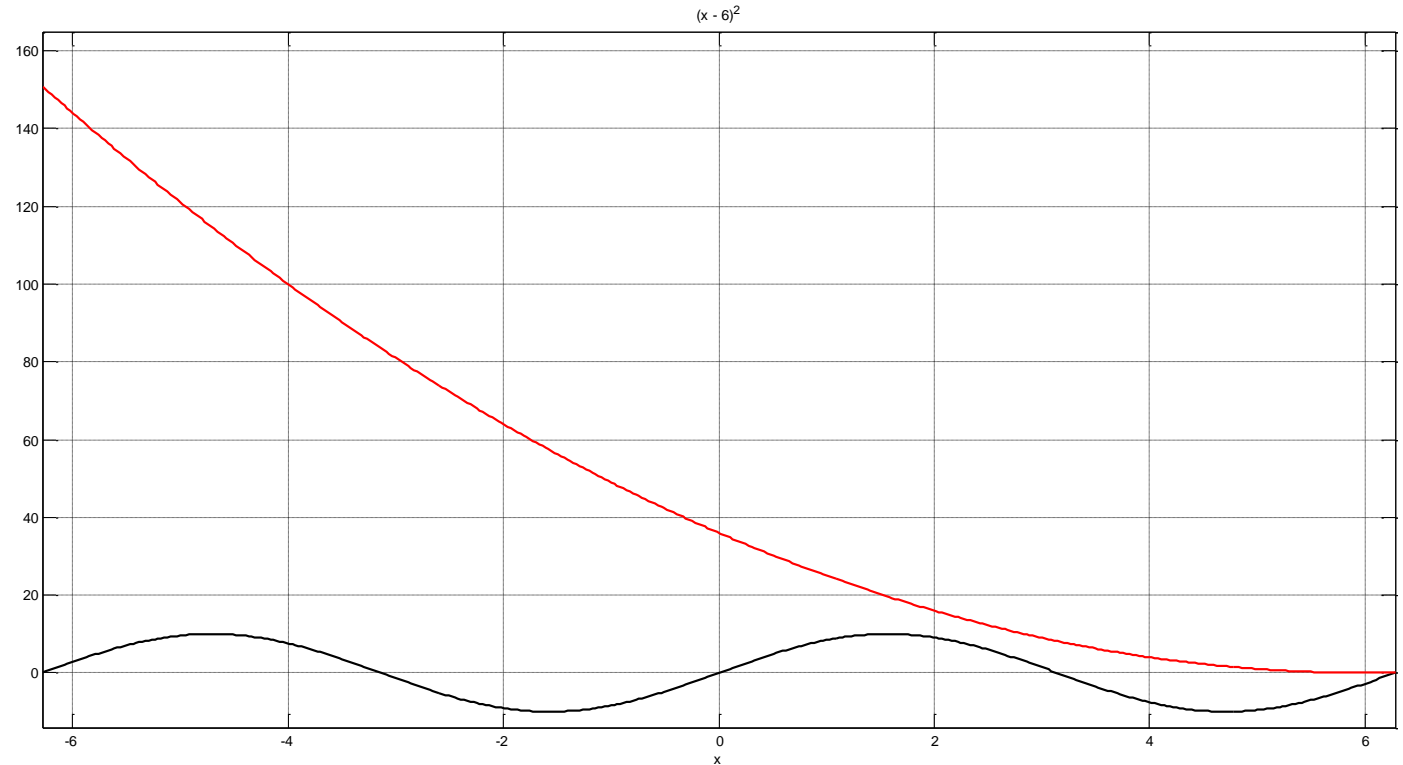
# Symbolic Expressions

# Manipulating symbolic expressions

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

# syms ezplot

```
clear
clc
syms x
%functia obiectiv
f=ezplot(10*sin(x));
set(f,'Color','black', 'LineWidth', 2)
hold on
%conditia la limita
j=ezplot((x-6)^2)
set(j,'Color','red', 'LineWidth', 2)
grid on
```



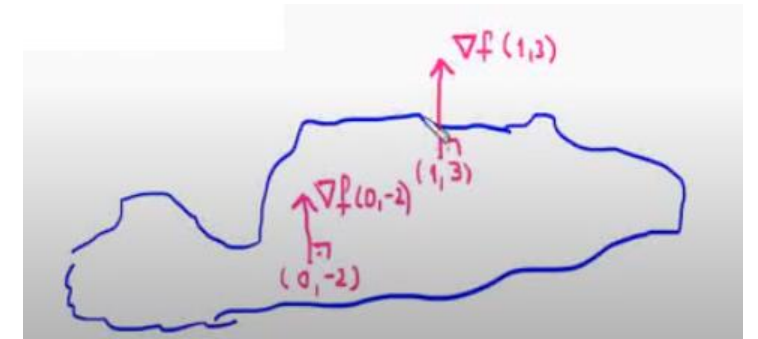


# Gradient of Function



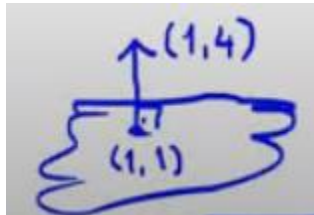
# Gradyan

- Bir skaler alanın yön türevi artımın en çok olduğu yere doğru yönelmiş bir vektör alanını verir ve büyüklüğü değişimin en büyük değerine eşittir. Örnekleme gerekirse bir odadaki zamandan bağımsız sıcaklık dağılımı düşünülebilir.
- Gradyan vektörü,  $f(x,y)$  veya  $f(x, y, z)$  gibi çok değişkenli fonksiyonlardan elde edilir.  $\nabla f$  ile gösterilir,  $f$  fonksiyonun gradyan vektörü denir.
- $\nabla f(x,y)=(f_x, f_y)$  dir. Bir vektördür, türev yardımı ile elde edilir.
- $\nabla f(x,y,z)=(f_x, f_y, f_z)$  dir. Bir vektördür, türev yardımı ile elde edilir.
- Gradyan vektörü fonksiyonun kısmi türevinden elde edilen bir vektördür.
- $f(x,y)$ , bir yüzeysel fonksiyondur. Yeri gelir bir maksimum, yeri gelir bir minimumdur. Bu fonksiyondan elde edilen gradyan vektörü, bu fonksiyonun herhangi bir noktasında çizilen dik vektörü verir.



# Örnek

- $f(x,y)=x^2y+y^3-x+1$  ise gradyan vektörü nedir?
- $\nabla f(x,y)=(2xy-1, x^2+3y^2)$
- $\nabla f(1,1)=(1, 4)$



clear all  
close all

```
syms x y z  
f = x^2*y+y^3-x+1;  
gradient(f, [x, y])
```

ans =

```
2*x*y - 1  
x^2 + 3*y^2
```

# Gradient vector of scalar function

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

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

```
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)
```

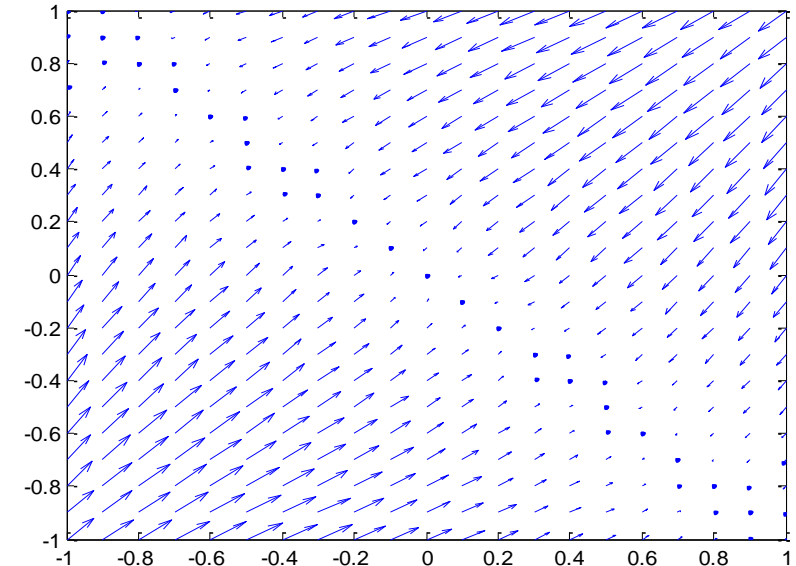
```
clear all  
close all
```

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

Answer

```
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
```

# Quiver or vector plot

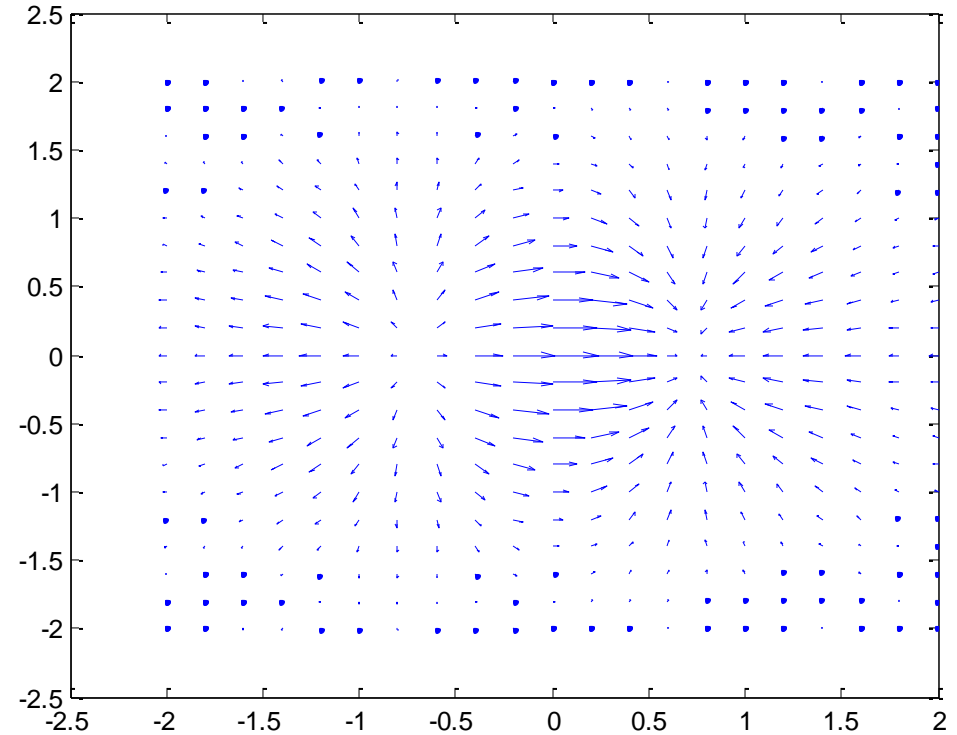
- $\text{quiver}(X,Y,U,V)$ ,  $X$  ve  $Y$  tarafından belirtilen Kartezyen koordinatlarda yön bileşenleri  $U$  ve  $V$  olan okları çizer. Örneğin, ilk ok  $X(1)$  ve  $Y(1)$  noktalarından kaynaklanır, yatay olarak  $U(1)$ 'e göre ve  $V(1)$ 'e göre dikey olarak uzanır. Varsayılan olarak, titreme işlevi ok uzunluklarını örtüşmeyecek şekilde ölçeklendirir.

# Gradient and Quiver

```
clear all
close all

spacing = 0.2;
[X,Y] = meshgrid(-2:spacing:2);
Z = X.*exp(-X.^2 - Y.^2);
[DX,DY] = gradient(Z,spacing);

quiver(X,Y,DX,DY)
```



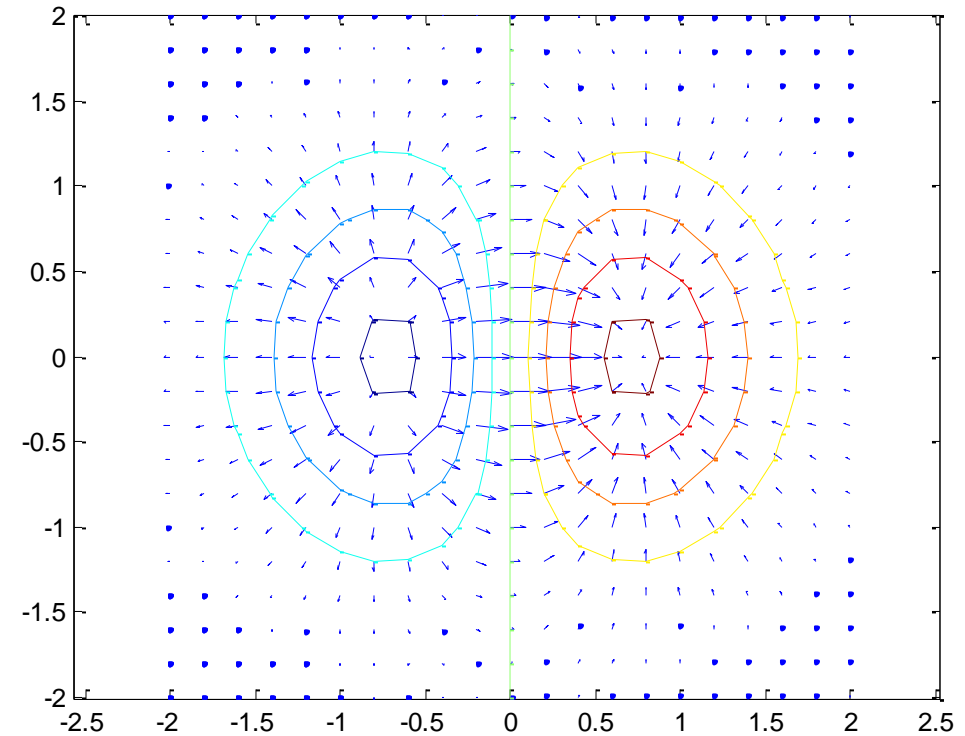
# Contour

- $\text{Contour}(Z)$ ,  $Z$  matrisinin izolinlerini içeren bir kontur grafiği oluşturur; burada  $Z$ ,  $x$ - $y$  düzlemindeki yükseklik değerlerini içerir.
- $\text{Contour}(X,Y,Z)$ ,  $Z$ 'deki değerler için  $x$  ve  $y$  koordinatlarını belirtir.

# Gradient – Quiver - Contour

```
clear all
close all

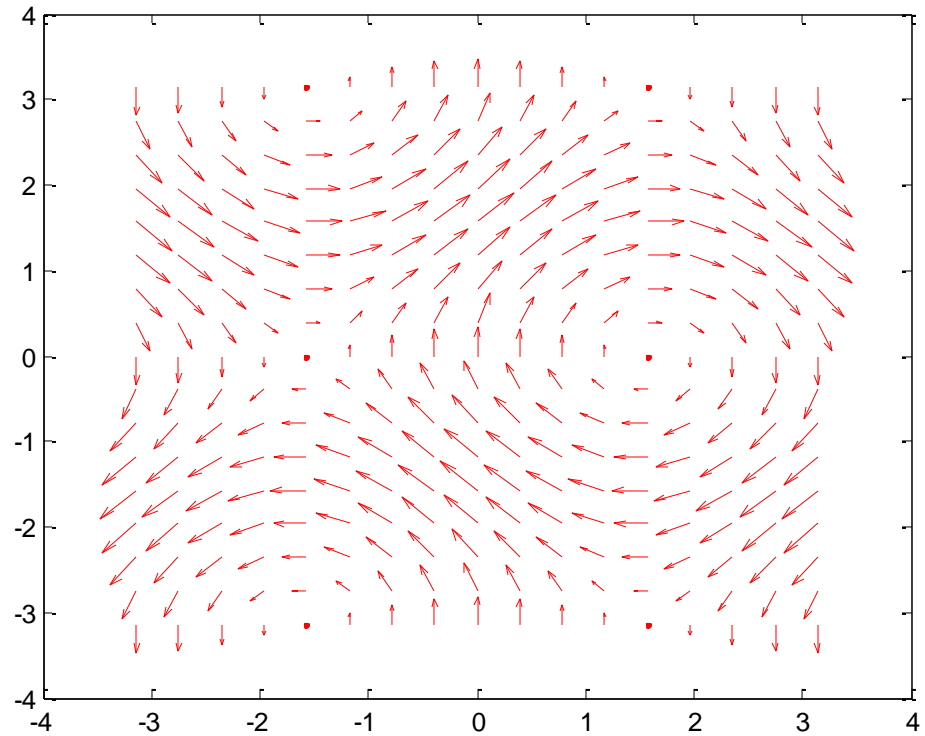
spacing = 0.2;
[X,Y] = meshgrid(-2:spacing:2);
Z = X.*exp(-X.^2 - Y.^2)
[DX,DY] = gradient(Z,spacing);
quiver(X,Y,DX,DY)
hold on
contour(X,Y,Z)
axis equal
hold off
```



# Gradient and Quiver

```
clear all  
close all
```

```
[X,Y] = meshgrid(-pi:pi/8:pi,-pi:pi/8:pi);  
U = sin(Y);  
V = cos(X);  
quiver(X,Y,U,V,'r')
```

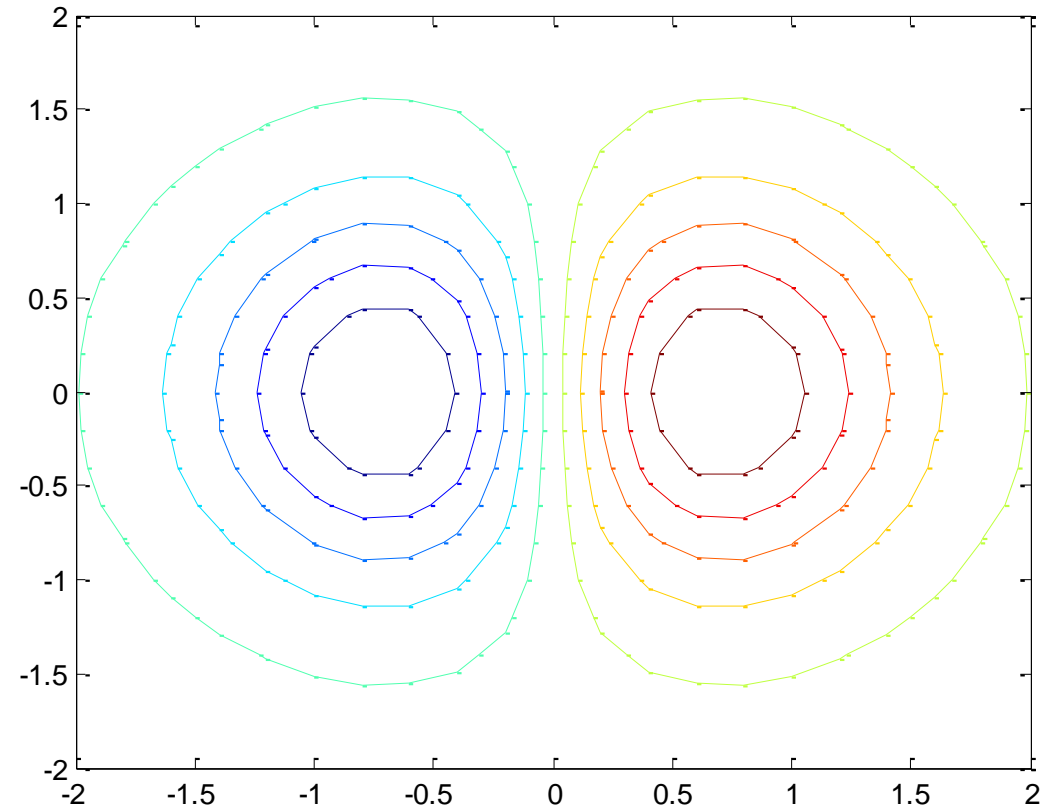




Örnek: Plot contours of  $xe^{-x^2-y^2}$  over a grid from -2 to 2 in the  $x$  and  $y$  directions.

```
clear all  
close all
```

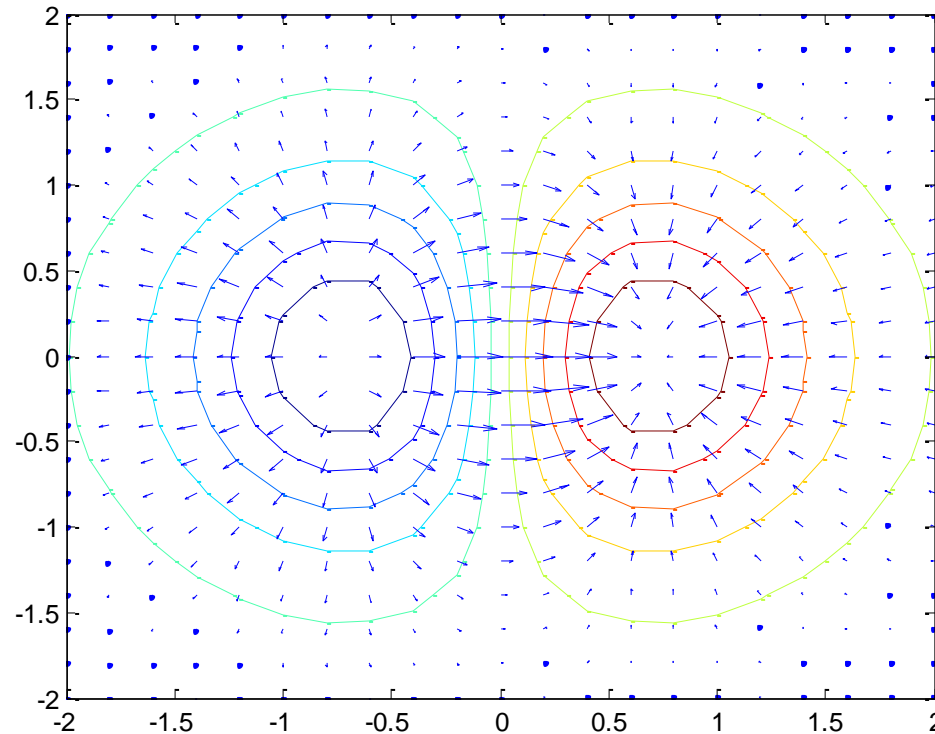
```
[X,Y] = meshgrid(-2:0.2:2);  
Z = X .* exp(-X.^2 - Y.^2);  
contour(X,Y,Z,10)
```



Örnek: Plot contours of  $xe^{-x^2-y^2}$  over a grid from -2 to 2 in the  $x$  and  $y$  directions.

```
clear all  
close all
```

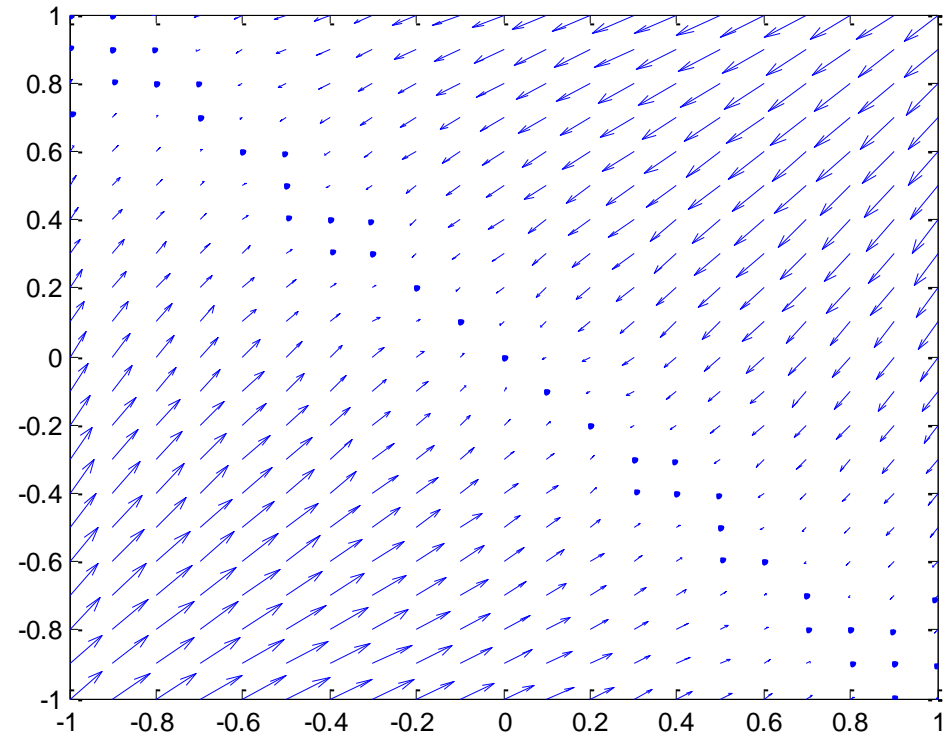
```
[X,Y] = meshgrid(-2:0.2:2);  
Z = X .* exp(-X.^2 - Y.^2);  
contour(X,Y,Z,10)  
[U,V] = gradient(Z,0.2,0.2);  
hold on  
quiver(X,Y,U,V)  
hold off
```



```
clear all  
close all
```

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

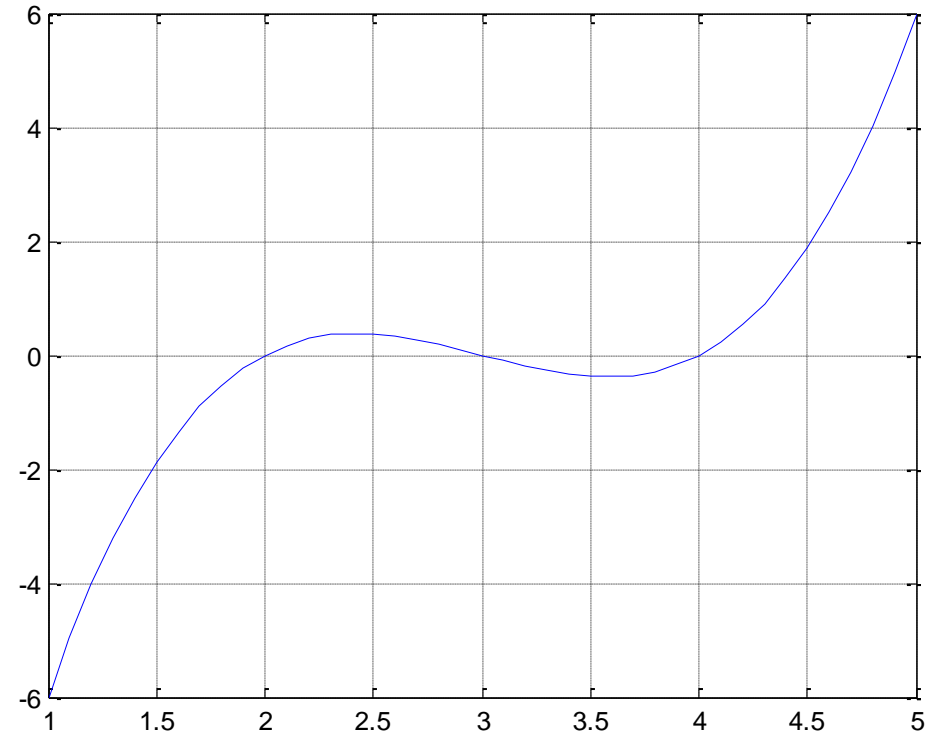




# Polynomials and Roots

```
clear all  
close all
```

```
c = [ 1 -9 26 -24 ]  
x = 1:0.1:5; % define range for plotting  
y = polyval(c,x); % compute samples  
plot(x,y) % make the plot  
grid on  
r = roots(c)
```



# Polynomials and Roots

```
clear all; close all
```

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

```
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 =
```

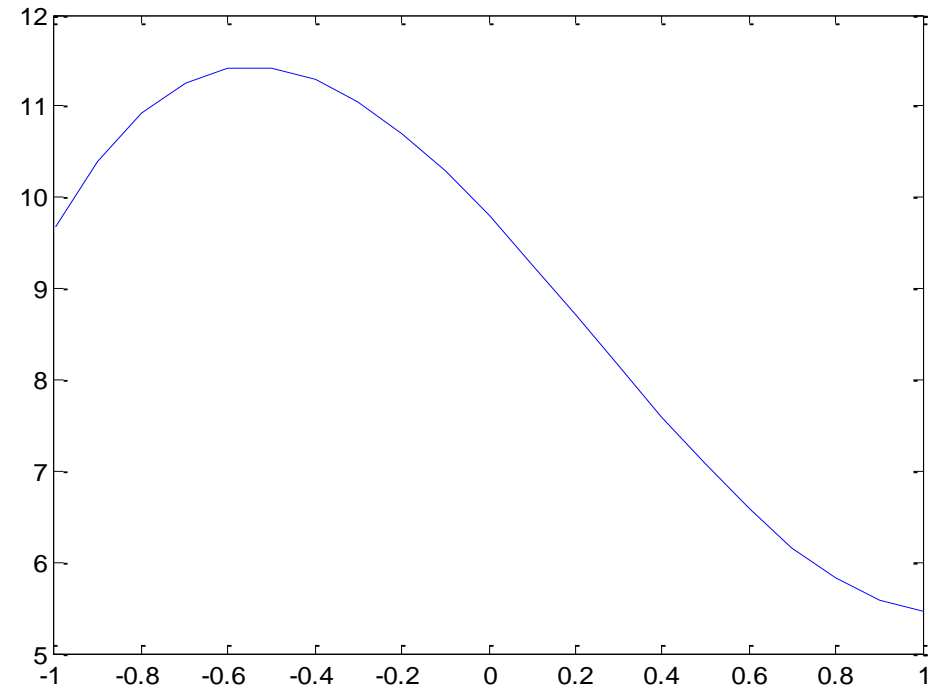
```
-1.5985 + 0.0000i
```

```
1.1709 + 0.8200i
```

```
1.1709 - 0.8200i
```

```
cr = 1.0000 -0.7433 -1.7000 3.2667
```

```
ans = 1.1374e-14
```

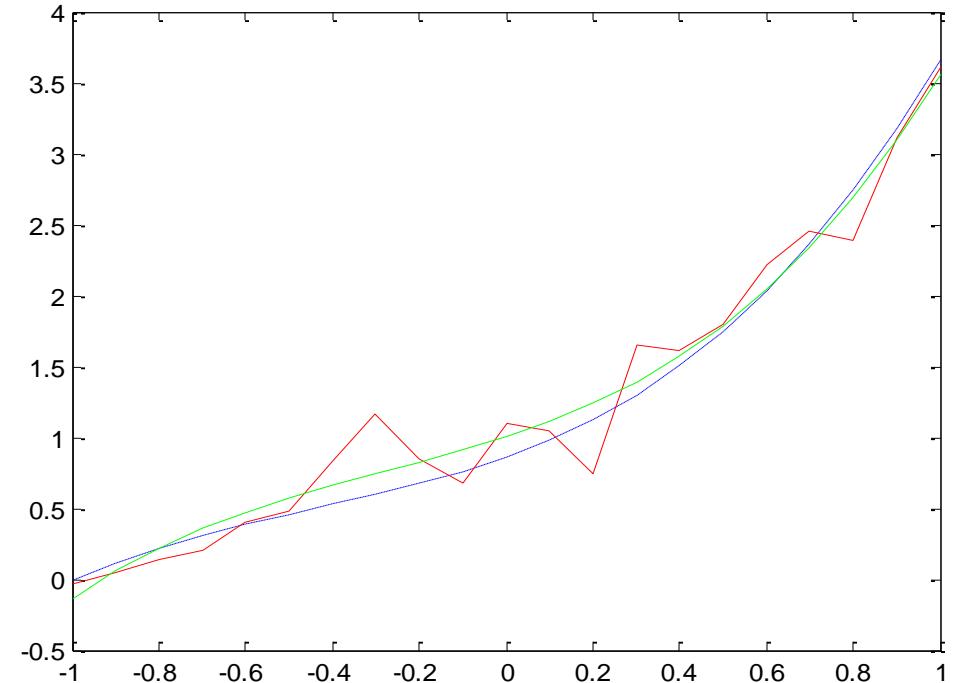


# Polynomials and Roots

```
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



# Finding the integral of a function

Mathematical Operation	MATLAB Command
$\int x^n dx = \frac{x^{n+1}}{n+1}$	<code>int(x^n) or int(x^n, x)</code>
$\int_0^{\pi/2} \sin(2x) dx = 1$	<code>int(sin(2*x), 0, pi/2) or int(sin(2*x), x, 0, pi/2)</code>
$g = \cos(at + b)$ $\int g(t) dt = \frac{\sin(at + b)}{a}$	<code>g = cos(a*t + b) int(g) or int(g, t)</code>
$\int J_1(z) dz = -J_0(z)$	<code>int(besselj(1, z)) or int(besselj(1, z), z)</code>

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

```
clear all  
close all
```

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

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

```
q = 1.9475
```

# *Integral, derivative*

```
clear all  
close all
```

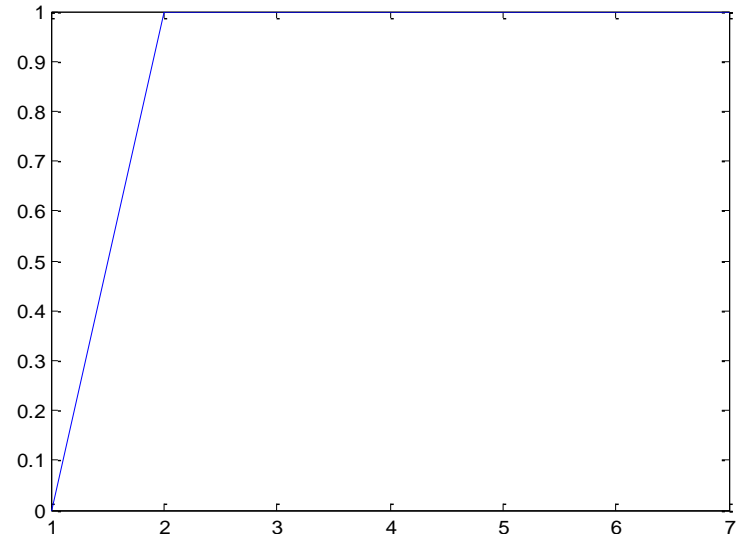
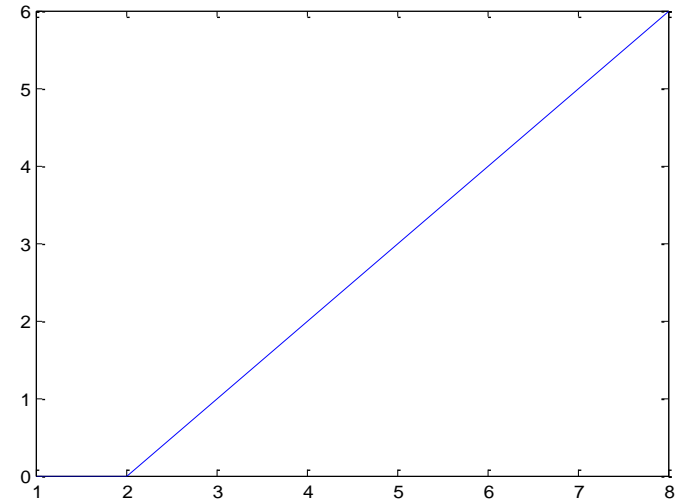
```
syms x  
y1=int(4*x^2+3)  
y2=int(4*x^2+3, x, -1, 3)
```

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

```
clear all  
close all  
syms x real  
f = 1/x  
y= diff(f)  
  
y=-1/x^2
```

```
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
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)
```



# Limit

# Finding the Limits of a function- $f(x)$

Mathematical Operation	MATLAB Command
$\lim_{x \rightarrow 0} f(x)$	<code>limit(f)</code>
$\lim_{x \rightarrow a} f(x)$	<code>limit(f,x,a)</code> or <code>limit(f,a)</code>
$\lim_{x \rightarrow a^+} f(x)$	<code>limit(f,x,a,'right')</code>
$\lim_{x \rightarrow a^-} f(x)$	<code>limit(f,x,a,'left')</code>

Evaluate: (a)  $\lim_{x \rightarrow 0} \left(\frac{1}{x}\right)$

(b)  $\lim_{x \rightarrow 0^+} \left(\frac{1}{x}\right)$

(c)  $\lim_{x \rightarrow 0^-} \left(\frac{1}{x}\right)$

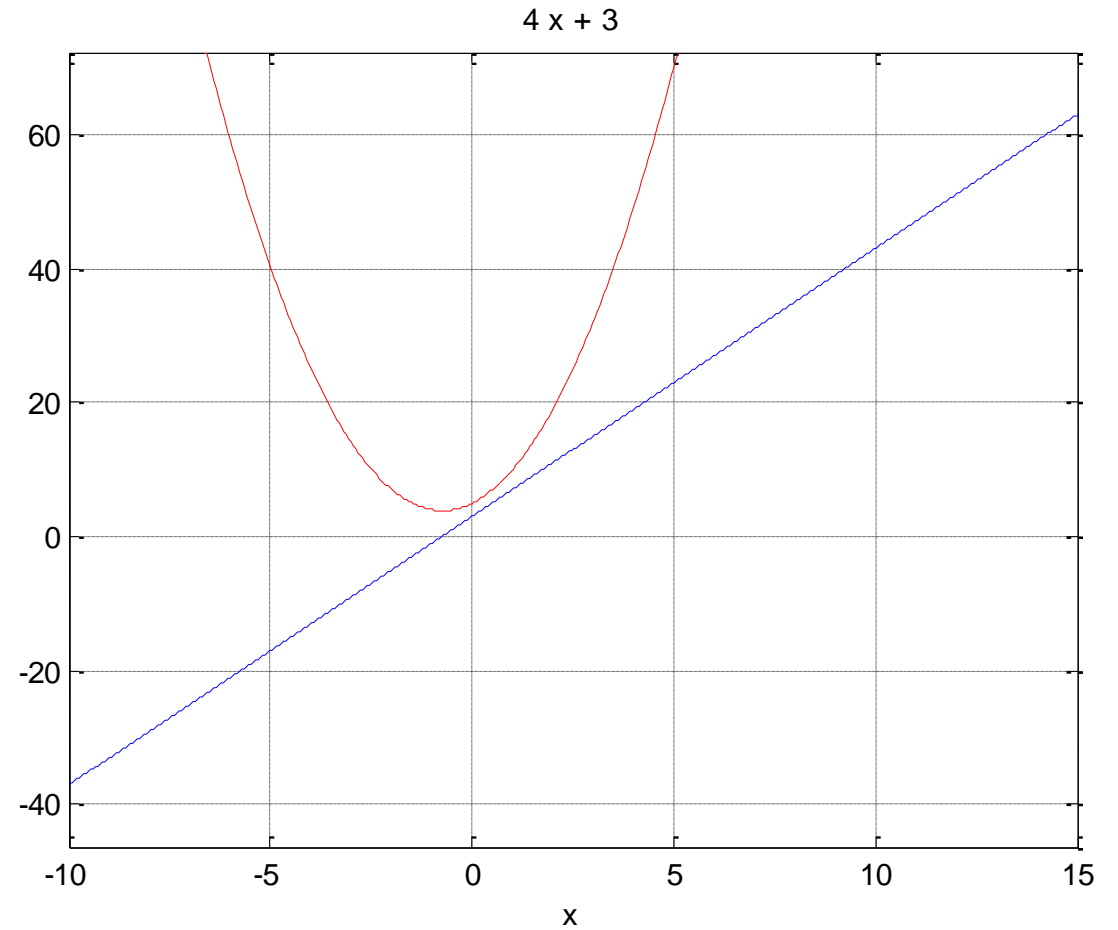
```
clear all  
close all
```

```
syms x  
f = 2*x^2+3*x+5
```

```
ez1=ezplot(f, [-10 15])  
hold on  
set(ez1,'color',[1 0 0])  
grid on
```

```
g=diff(f)  
ezplot(g, [-10 15])
```

```
hold off
```



# Limit of symbolic expression

- `limit(f,var,a)` returns the Bidirectional Limit of the symbolic expression  $f$  when  $var$  approaches  $a$ .
- `limit(f,a)` uses the default variable found by `symvar`.
- `limit(f)` returns the limit at 0.
- `limit(f,var,a,'left')` returns the Left Side Limit of  $f$  as  $var$  approaches  $a$ .
- `limit(f,var,a,'right')` returns the Right Side Limit of  $f$  as  $var$  approaches  $a$ .



# Örnek: Limit of symbolic expression

```
syms x  
f = 1/x;  
limit(f,x,0,'right')  
ans =  $\infty$ 
```

```
limit(f,x,0,'left')  
ans =  $-\infty$ 
```

Since the limit from the left does not equal the limit from the right, the two-sided limit does not exist. In this case, limit returns NaN (not a number).

```
limit(f,x,0)  
ans = NaN
```

# Örnek: Limit of symbolic expression

```
syms x
```

```
f=(x^3 + 5)/(x^4 + 7)
```

```
limit(f)
```

```
ans = 5/7
```

```
syms x
```

```
f = (x-3)/(x-1)
```

```
limit(f,1)
```

```
ans = NaN
```

# Örnek: Limit of symbolic expression

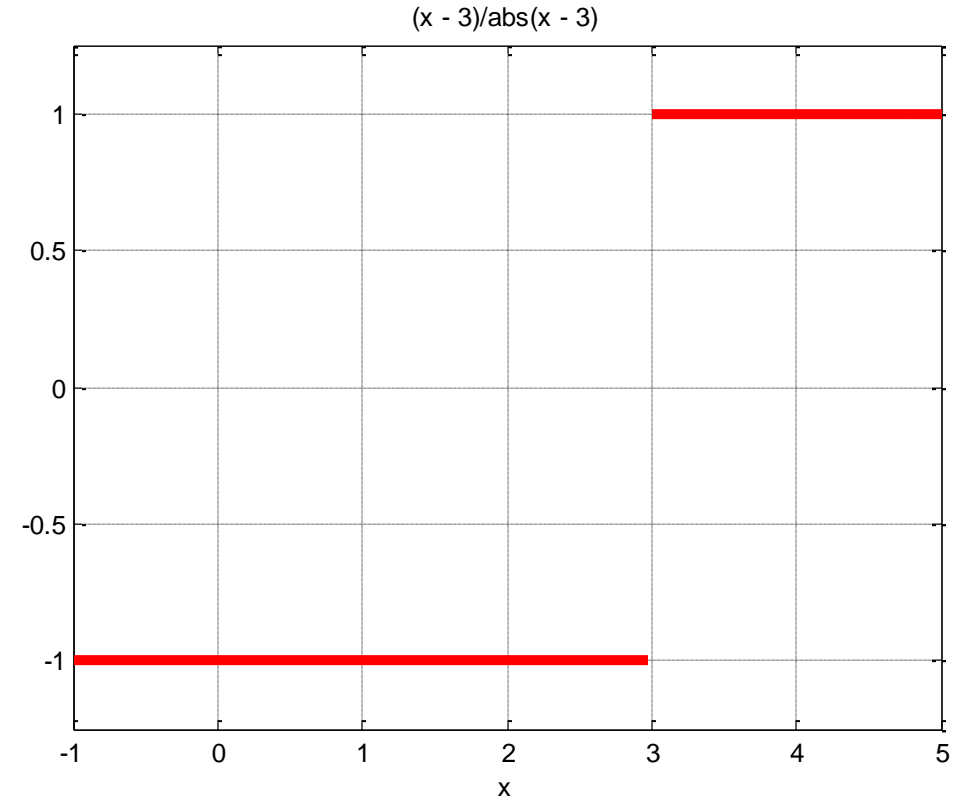
```
clear all  
close all
```

```
syms x  
f = (x - 3)/abs(x-3);  
h= ezplot(f,[-1,5])  
set(h,'Color','red', 'LineWidth', 4)  
grid on
```

```
l = limit(f,x,3,'left')  
r = limit(f,x,3,'right')
```

```
l = -1
```

```
r = 1
```

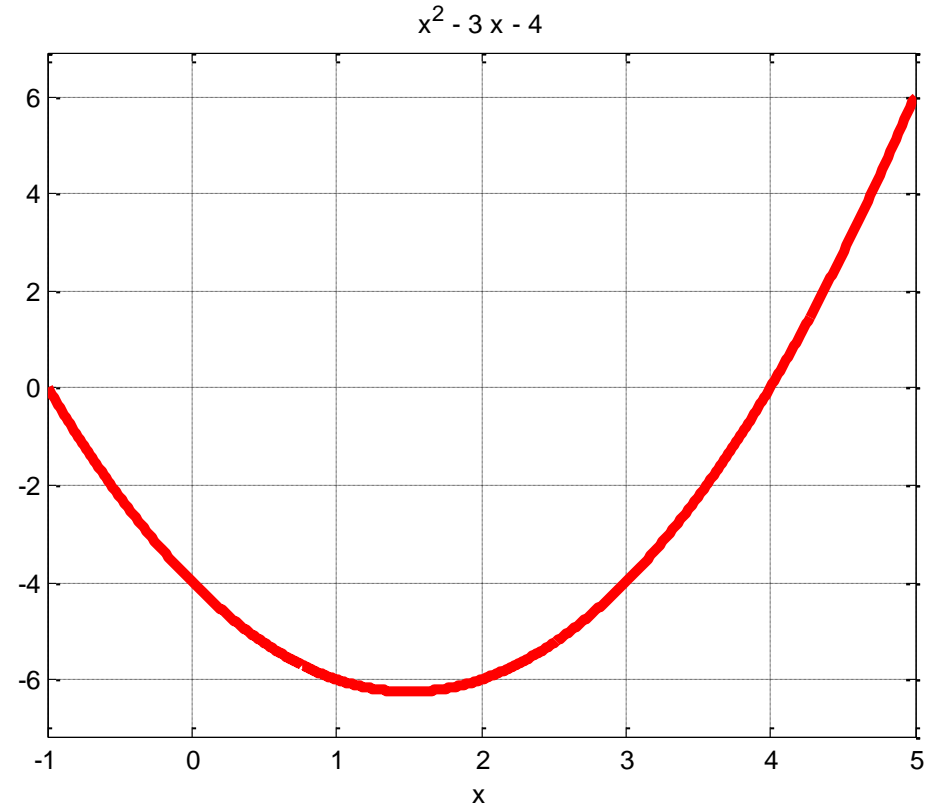


# Örnek: Limit of symbolic expression

```
clear all  
close all
```

```
syms x  
f = x^2-3*x-4;  
h= ezplot(f,[-1,5])  
set(h,'Color','red', 'LineWidth', 4)  
grid on
```

```
l = limit(f,x,1,'left')  
r = limit(f,x,1,'right')  
l = -6  
r = -6
```



# Örnek: Limit of symbolic expression

```
clear all
```

```
close all
```

```
syms x
```

```
f = (x^2-3*x-4)/(x-3);
```

```
h= ezplot(f,[-1,5])
```

```
set(h,'Color','red', 'LineWidth', 4)
```

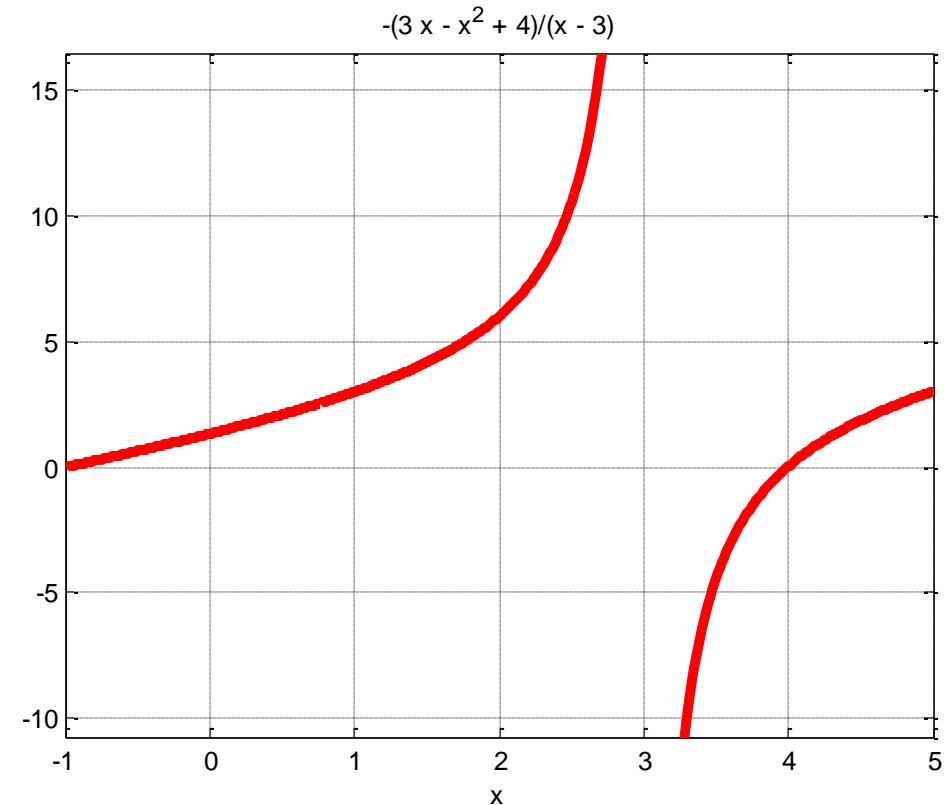
```
grid on
```

```
l = limit(f,x,3,'left')
```

```
r = limit(f,x,3,'right')
```

```
l =Inf
```

```
r =-Inf
```





# Solution of Differential Equations with MATLAB

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

```
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
$\frac{dy}{dt} + 4y(t) = e^{-t},$ $y(0) = 1.$	<pre>syms y(t) ode = diff(y)+4*y == exp(-t); cond = y(0) == 1; ySol(t) = dsolve(ode,cond)</pre> $ySol(t) = \exp(-t)/3 + (2*\exp(-4*t))/3$
$2x^2 \frac{d^2y}{dx^2} + 3x \frac{dy}{dx} - y = 0.$	<pre>syms y(x) ode = 2*x^2*diff(y,x,2)+3*x*diff(y,x)-y == 0; ySol(x) = dsolve(ode)</pre> $ySol(x) = C2/(3*x) + C3*x^{(1/2)}$
<p>The Airy equation.</p> $\frac{d^2y}{dx^2} = xy(x).$	<pre>syms y(x) ode = diff(y,x,2) == x*y; ySol(x) = dsolve(ode)</pre> $ySol(x) = C1*airy(0,x) + C2*airy(2,x)$



## 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

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

```
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
```

```
clear all  
close all
```

```
eqn2 = 'D2y + 8*Dy + 2*y = cos(t)';  
inits2 = 'y(0)=0, Dy(0)=12';  
y=dsolve(eqn2,inits2,'t')  
y1 = simplify(y)
```

```
ezplot(y, [-4.5 -3])
```

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

$$x'(t) = x(t) + 2y(t) - z(t)$$

$$y'(t) = x(t) + z(t)$$

$$z'(t) = 4x(t) - 4y(t) + 5z(t).$$

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

$$x'(t) = x(t) + 2y(t) - z(t)$$

$$y'(t) = x(t) + z(t)$$

$$z'(t) = 4x(t) - 4y(t) + 5z(t).$$

```
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

 $y$  $\bullet y$  $\frac{dy}{dt}$  $\bullet Dy$  $\frac{d^2 y}{dt^2}$  $\bullet D^2y$  $\frac{d^n y}{dt^n}$  $\bullet D^n y$

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

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

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

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

$$\frac{dy}{dt} + 2y = 12$$

$$y(0) = 10$$

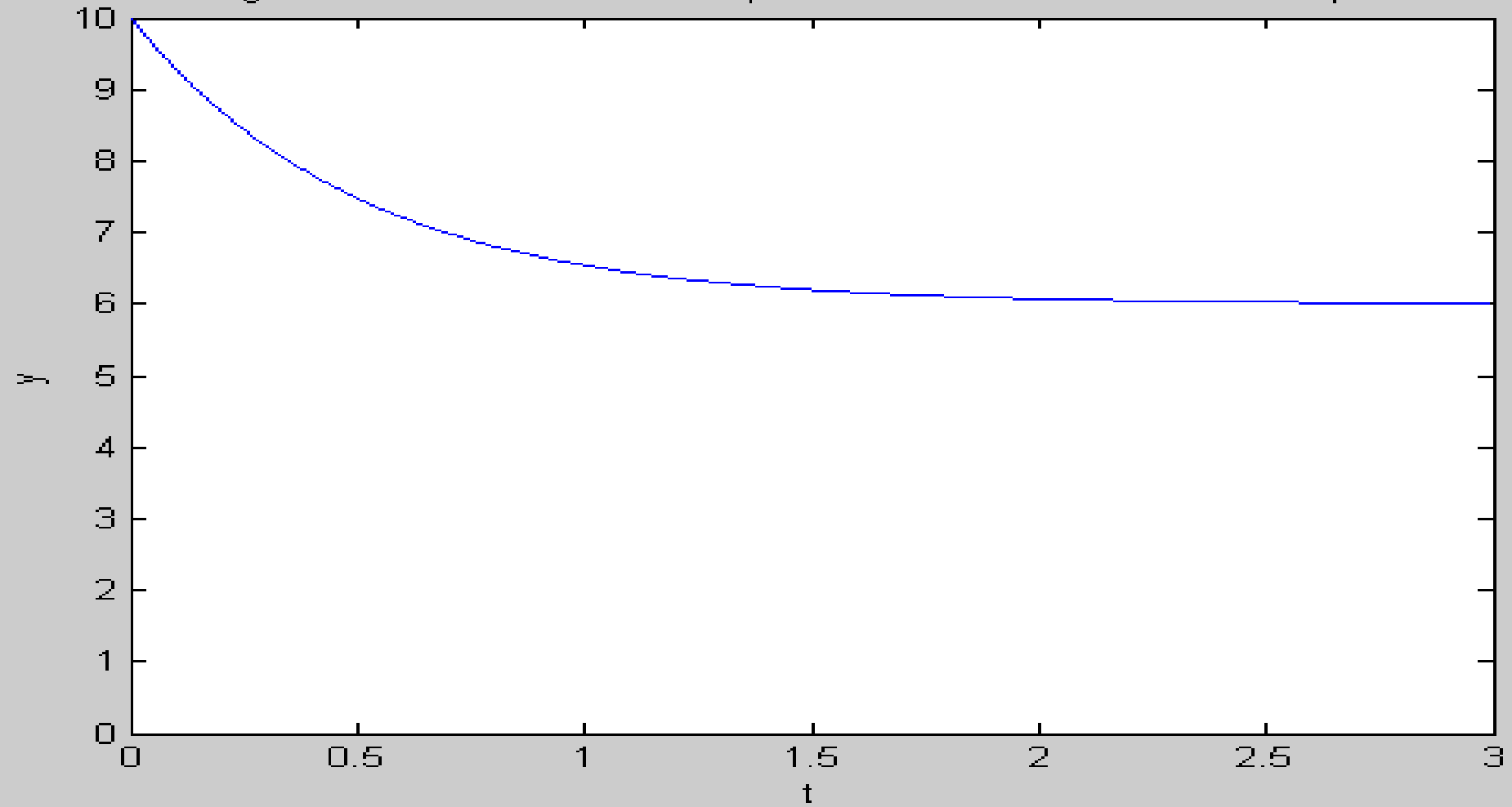
```
y = dsolve('Dy + 2*y = 12', 'y(0)=10')
```

```
>> 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.





$$\frac{d^2 y}{dt^2} + 3 \frac{dy}{dt} + 2y = 24$$
$$y(0) = 10 \quad y'(0) = 0$$

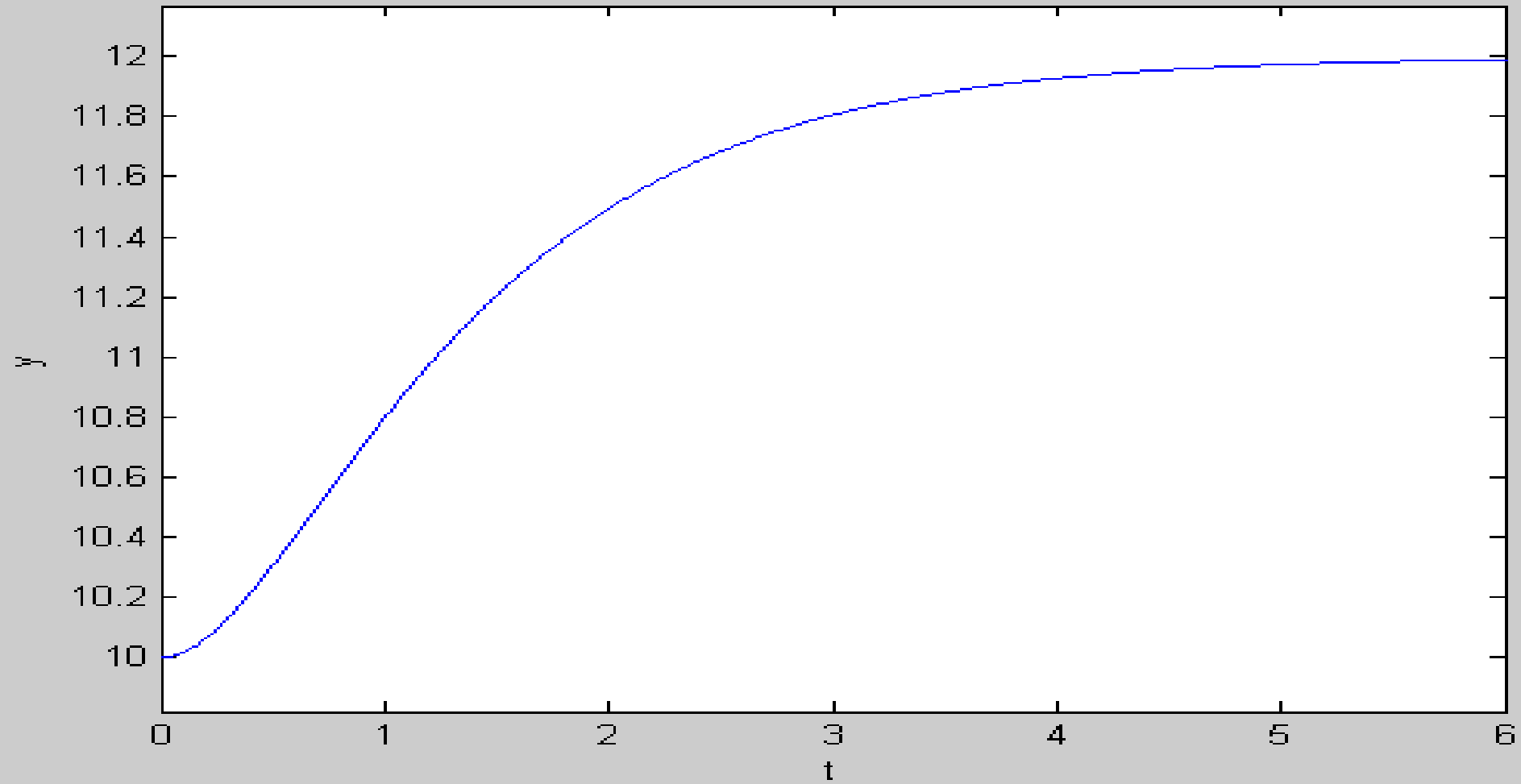
•>> `y = dsolve('D2y + 3*Dy + 2*y = 24', 'y(0)=10', 'Dy(0)=0')`

•`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.



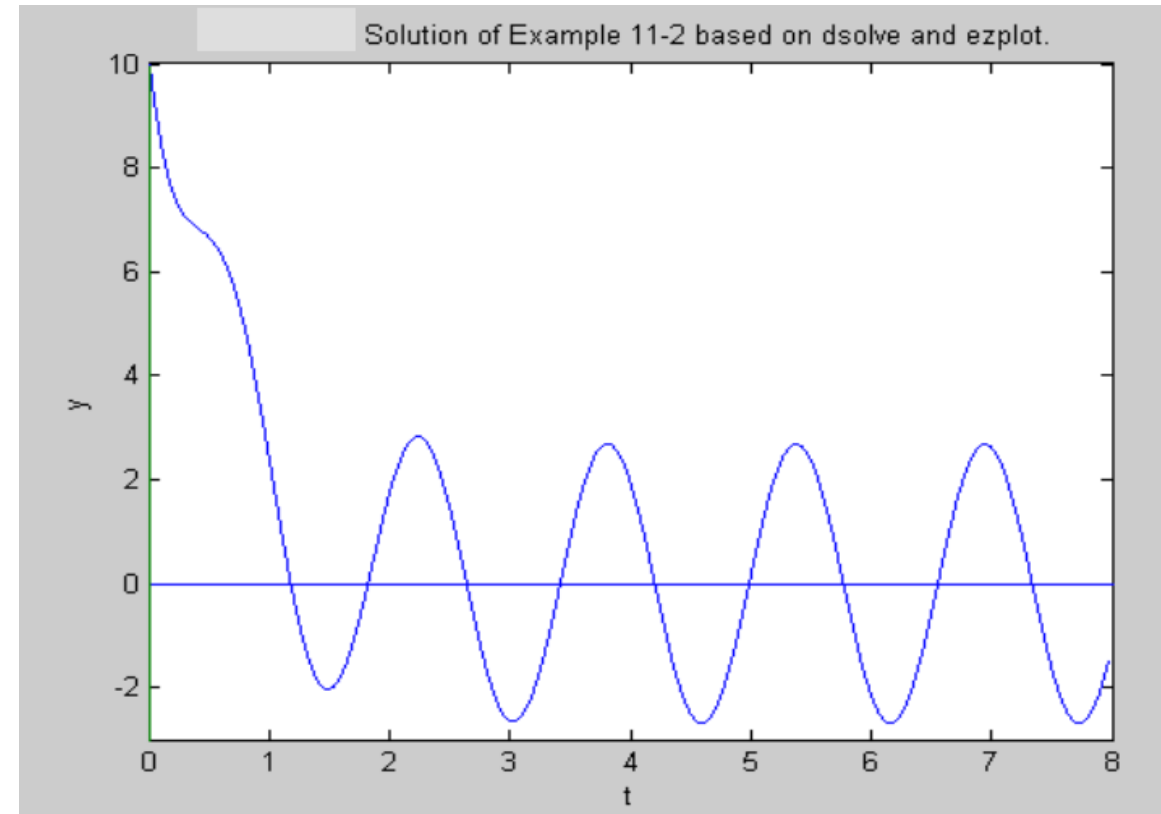
$$\frac{dy}{dt} + 2y = 12 \sin 4t \quad y(0) = 10$$

```
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)
```



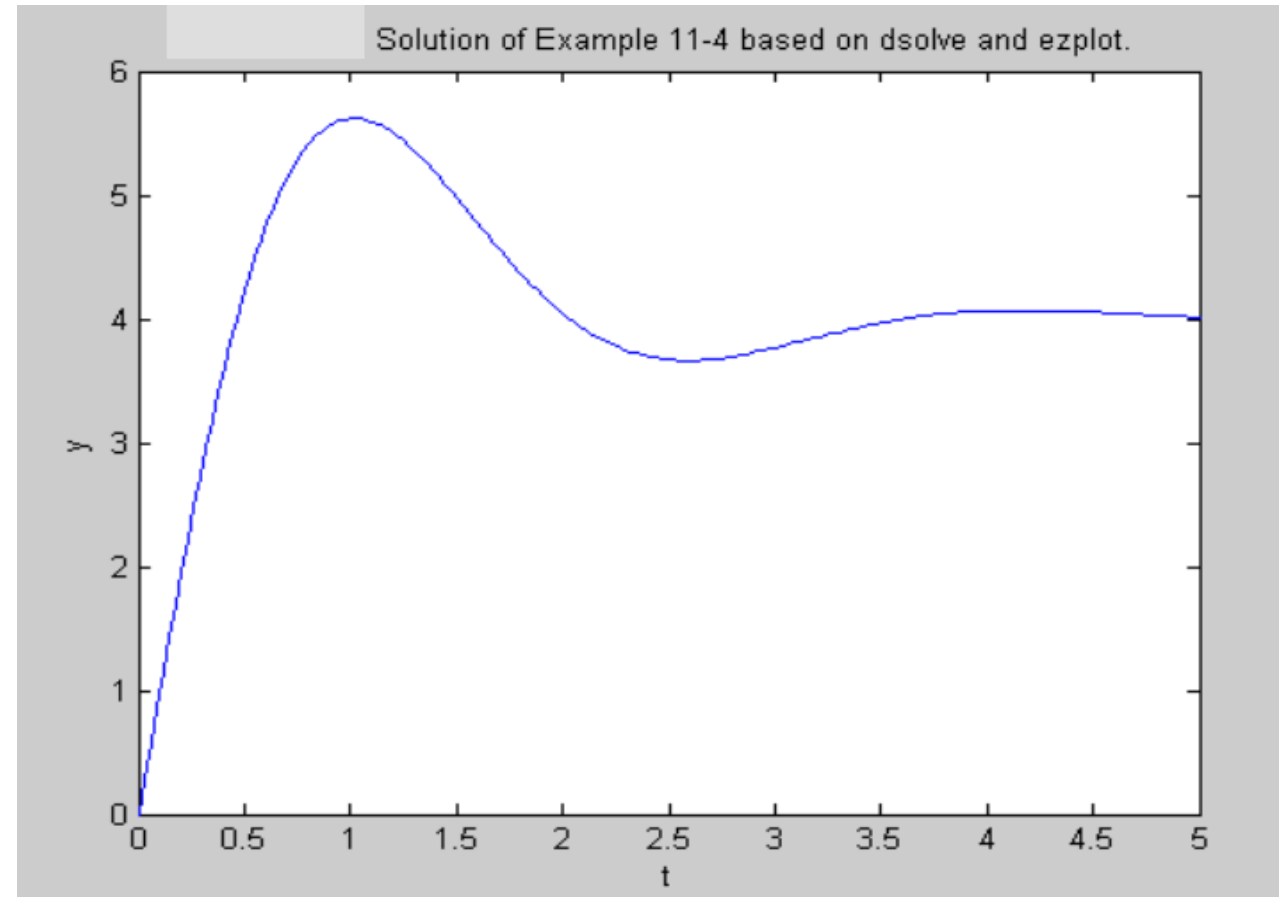
$$\frac{d^2 y}{dt^2} + 2 \frac{dy}{dt} + 5y = 20$$

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

```
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)
```



## Nonlinear Differential Equation with Initial Condition

Solve this nonlinear differential equation with an initial condition. The equation has multiple solutions.

$$\left(\frac{dy}{dt} + y\right)^2 = 1,$$

$$y(0) = 0.$$

```
syms y(t)
```

```
ode = (diff(y,t)+y)^2 == 1;
```

```
cond = y(0) == 0;
```

```
ySol(t) = dsolve(ode,cond)
```

```
ezplot(ySol, [0 6])
```

## Second-Order ODE with Initial Conditions

Solve this second-order differential equation with two initial conditions.

$$\frac{d^2y}{dx^2} = \cos(2x) - y,$$

$$y(0) = 1,$$

$$y'(0) = 0.$$

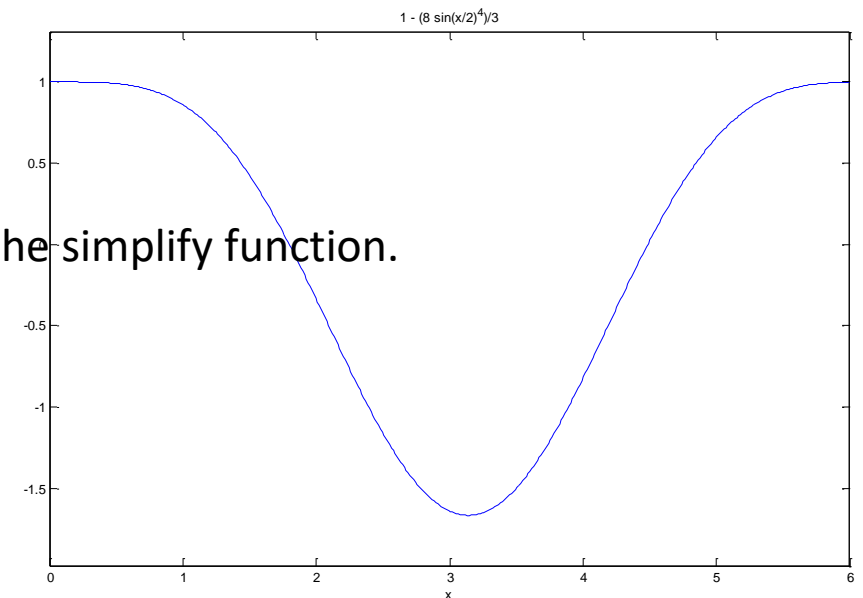
Define the equation and conditions. The second initial condition involves the first derivative of  $y$ . Represent the derivative by creating the symbolic function  $Dy = \text{diff}(y)$  and then define the condition using  $Dy(0)=0$ .

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

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

Solve ode for  $y$ . Simplify the solution using the `simplify` function.

```
conds = [cond1 cond2];
ySol(x) = dsolve(ode,conds);
ySol = simplify(ySol)
ezplot(ySol, [0 6])
ySol(x) =
1 - (8*sin(x/2)^4)/3
```



# 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**