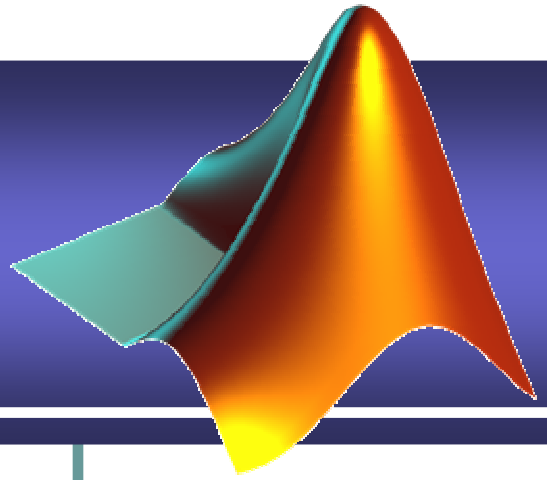


MATLAB

Y. Doç. Dr. Aybars UĞUR

Yapay Sinir Ağları

Ders Notları

MATLAB Nedir?

- MATLAB, Mathworks firmasının geliştirdiği teknik bir programlama dilidir. (www.mathworks.com)
- MATLAB, teknik hesaplamalar ve matematiksel problemlerin çözümü ve analizi için tasarlanmış bir yazılım geliştirme aracıdır.
- "MATrix LABoratory" kelimesinin kısaltması olan MATLAB, matris yani dizi tabanlıdır.

MATLAB Ortamı

Komut Penceresi

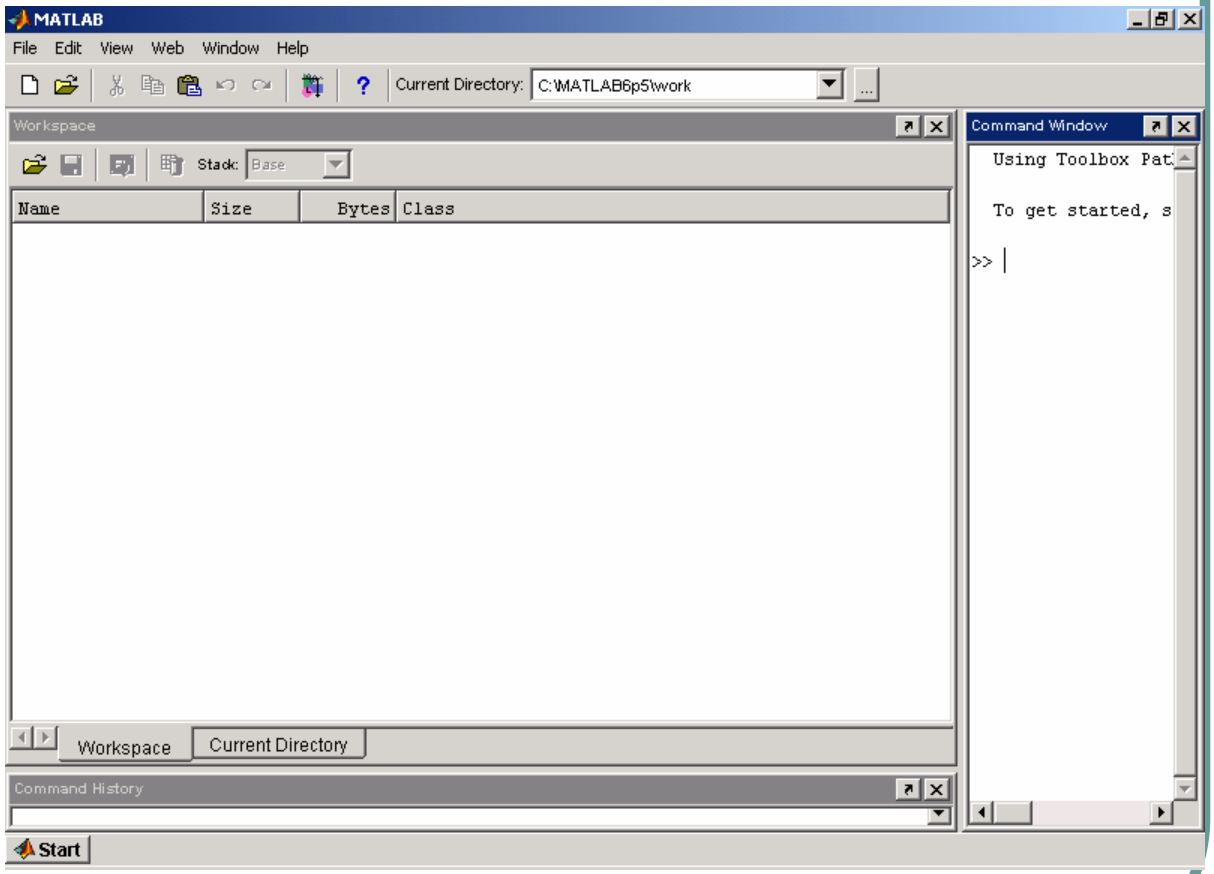
Çalışma Alanı

Etkin Klasör

M-File

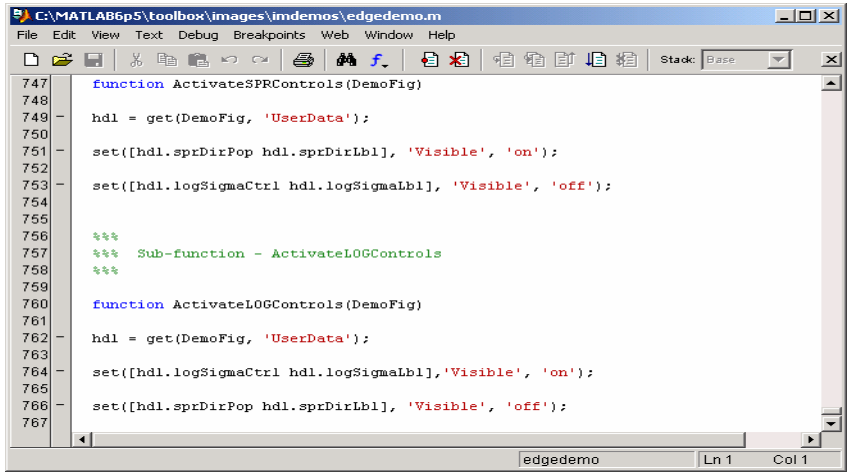
Help

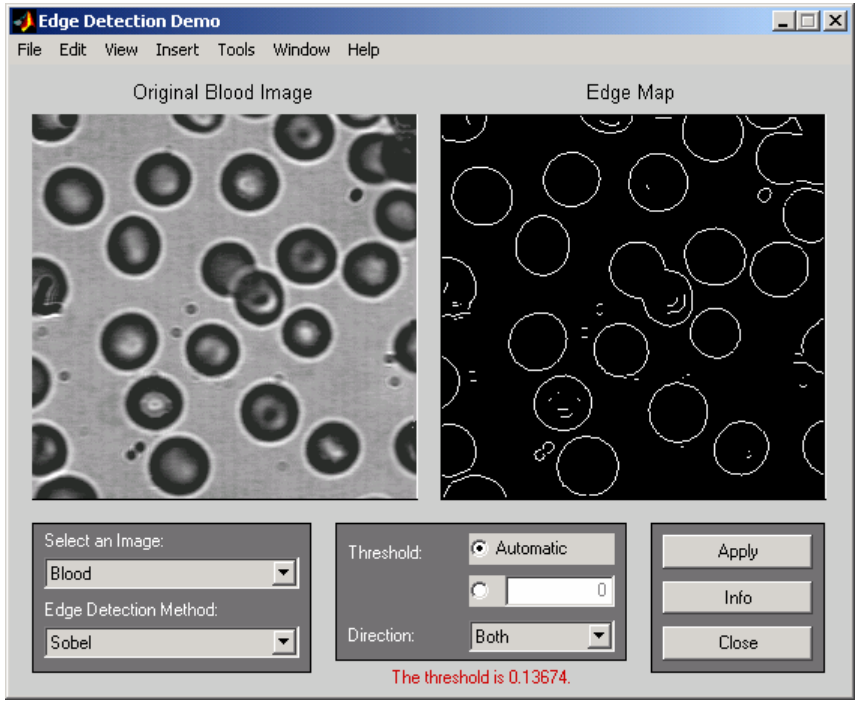
Demolar



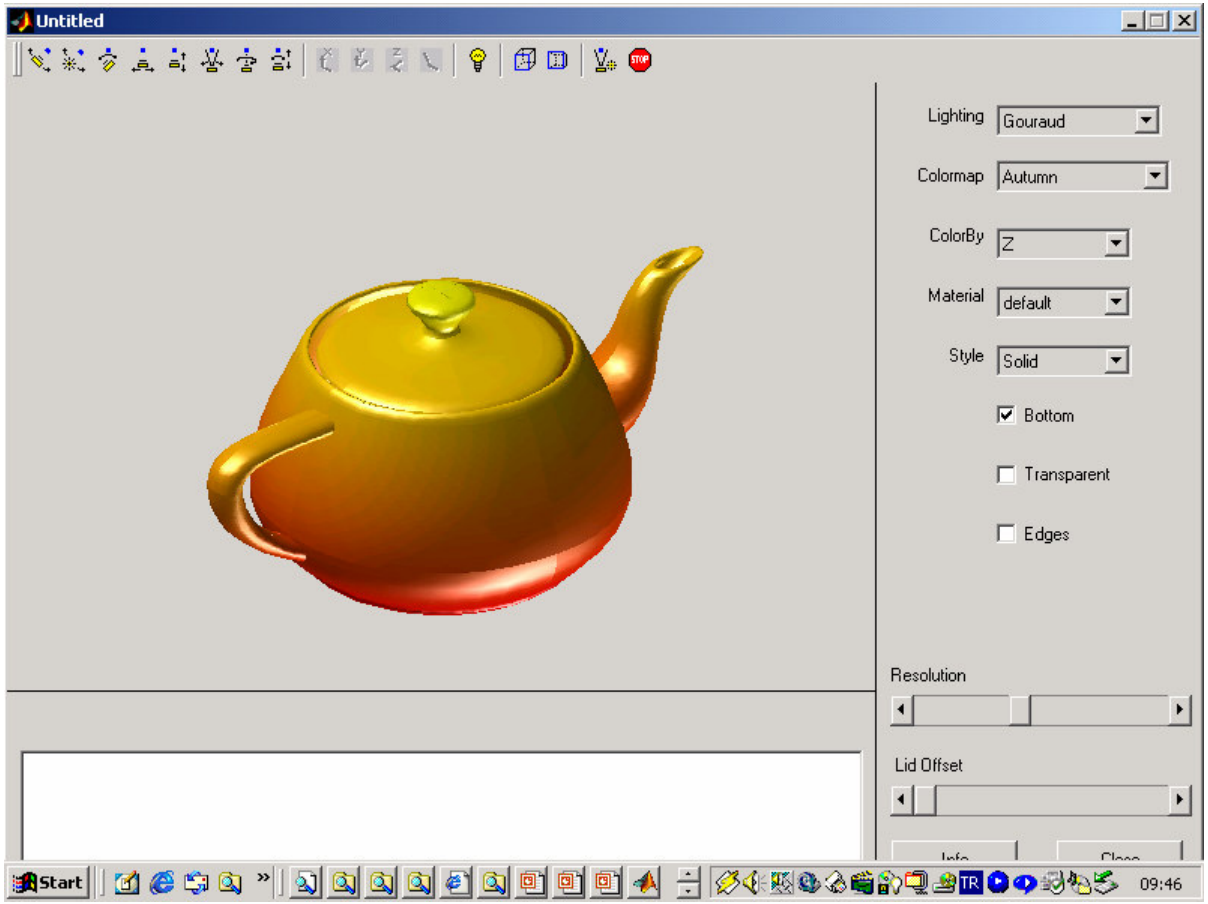
Görüntü İşleme

Edge Detection Demo





Grafik


Untitled
Lighting
Gouraud
Colormap
Autumn
ColorBy
Z
Material
default
Style
Solid
Bottom
Transparent
Edges
Resolution
Lid Offset
Start
09:46

MATLAB Araç Kutuları

- Kontrol
- İstatistik
- Optimizasyon
- Finans
- ...
- Yapay Sinir Ağları
- Genetik Algoritmalar
- Görüntü İşleme
- Bulanık Mantık
- Grafik
- Veritabanı
- Web Sunucusu
- ...

Programlama Dillerindeki Yeri

- Matlab, denklem çözümü, grafik çizimi, simülasyon, veri analizi, görüntü işleme karakter tanıma (OCR) gibi işlevleri içeren yazılımların geliştirilmesini kolaylaştırıp hızlandırır. Geleneksel programlama dillerinde yeterli düzeyde hazır bütünleşik araçlar yoktur.
- Örnek : Sadece, görüntü işleme araç kutusu 200 kadar fonksiyondan oluşur.
- Genel bir programlama dili değildir.

MATLAB Programlarının İşletimi

MATLAB'da yazılan programlar,

- MATLAB içinden çalıştırılabilir
- EXE veya DLL oluşturulabilir
- C/C++ kodlarına dönüştürülebilir
- ...
- Matlab'ın Windows, Linux sürümleri ...
- GUI aracı ile formlar da oluşturulabilir.

Değişkenler ve Veri Tipleri

```
>> sayi=55

sayi =

    55

>> size(sayi)

ans =

     1     1
Matris->Dizi
```

```
deger=[10,20,sayi]
deger =
    10    20    55

>> whos
     Name          Size          Bytes    Class
     ans           1x2              16    double array
     deger         1x3              24    double array
     sayi          1x1               8    double array

Grand total is 6 elements using 48 bytes
clear all
veya
clear ans sayi deger
```

Workspace,

Command History

Dizi ve Matris İşlemleri

- d1 = -3:.5:3
- d1
- d1(3) ... Çıktı : -2
- d1 = [-2:.5:2]
- size(d1)
- i=1
- d1(i)=-7
- d1(17)=1 (genişletme)
- d1=[1 4;2 5;3 6]

m1 = rand(3,3)

m1 =

0.6721	0.6813	0.5028
0.8381	0.3795	0.7095
0.0196	0.8318	0.4289

m1(5), m1(2,2)'ye eşittir.

```
m1(2:3,2:3) =?, m1(1,:) =?
m1([2 3],[3 1 2]) ile
m1([2:3], [3;1;2])
```

```
d=[1:3;4:6;7:9]
     1     2     3
     4     5     6
     7     8     9
```

Dizi ve Matris İşlemleri

Dizi oluşturan temel fonksiyonlar :

- zeros(2,3)
- ones(2,3)
- eye(2,3)
- rand(5,5)

- Diğer : any, all, find, isreal, help isreal
- ....

Operatörler :

- + -
- .* ./ .^
- * / ^(matrisler için)
- .' (devrik alma)
- \ .\ (sol bölme)
- <,>,<=,>=,==,~= ilişkisel
- & | xor bitor bitand ...
- && || mantık kontrol

Hücre (Cell)

- Elemanları farklı türlerden dizi, hücre veya yapı olabilir.

```
>> m2={[1:2];['Matlab',[2:3,4:5]]}

m2 =

    [1x2 double]
    'Matlab□□□□'
```

Yapı (Structure)

```
>> ogrenci.ad='Ali'

ogrenci =

    ad: 'Ali'

>> ogrenci.yas=25

ogrenci =

     ad: 'Ali'
    yas: 25
```

```
>> ogrenci.adres.il ='Izmir'

ogrenci =

       ad: 'Ali'
      yas: 25
    adres: [1x1 struct]

>> ogrenci.adres.ilce ='Bornova'
>> ogrenci.adres

ans =

      il: 'Izmir'
    ilce: 'Bornova'
```

Dizi Kaydetme ve Yükleme

Kaydetme

- Workspace'den değişkenler seçilerek veya
- Save Workspace As ile veya
- Komut satırından save dosyaad d1 d2 şeklinde yapılabilir.

Yükleme

- Etkin klasörden
- load dosyaad
- File -> Import Data xls, txt, mat ...
- .....

Prosedürler ve Fonksiyonlar

- New M-File

```
% Cevre ve Alan hesaplar
% Kucuk buyuk harf duyarli
r=5.5
cevre=2*pi*r
alan=pi*r*r
```

Yazılır ve çalıştırılır.

Ekran Çıktısı

```
r =
    5.5000
cevre =
   34.5575
alan =
   95.0332
```

- New M-File

```
function cevre=chesapla(r)
cevre=2*pi*r
```

.m dosyasına chesapla ismi verilerek kaydedilir komut satırından çalıştırılabilir

Ekran Çıktısı

```
>>chesapla(5)

cevre =
   31.4159
ans =
   31.4159
```

Denetim Deyimleri

Seçim

- if..end, switch..end

```
a=3; b=4
if a>b
    disp('a b den büyüktür');
    x=1
elseif a<b
    disp('a b den kücüktür');
    x=-1;
else
    disp('a b esittir');
    x=0;
end
```

Ekran Çıktısı

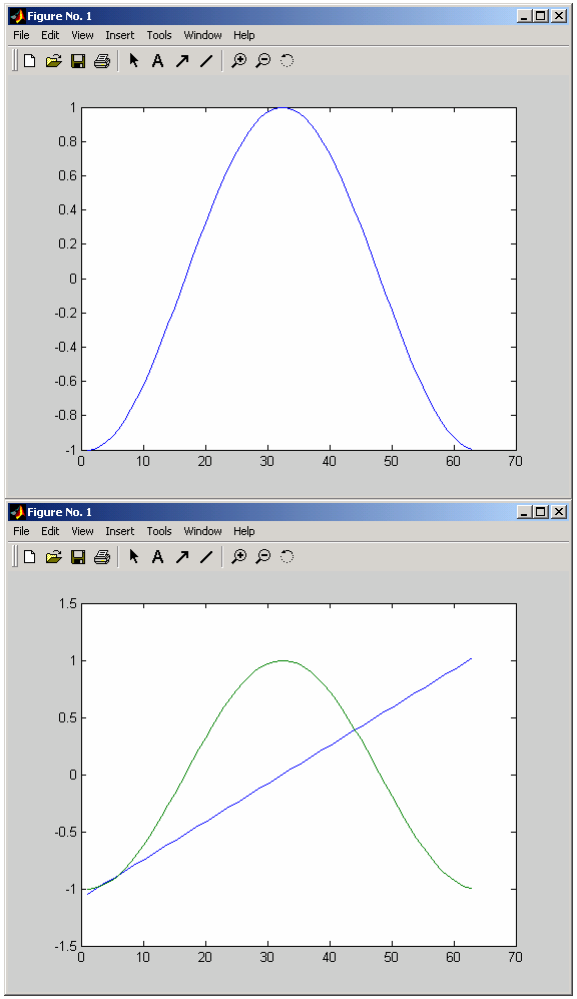
```
a =
     3
b =
     4
a b den kücüktür
```

- Döngü
- for, while, break, continue

```
n=input('Sayiyi Verin')
top=0;
for i=1:2:n
    top=top+i;
end
disp(top)
top=0; i=0;
while i<100
        top=top+i;
        i=i+1;
end
```

Grafikler - Plot

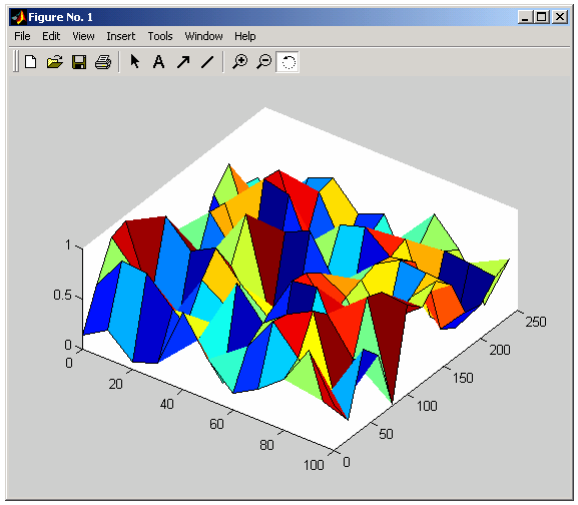
```
t=-pi:0.1:pi;
y=cos(t);
plot(y)
%Grafik özellikleri değiştirilebilir
%Birden çok grafik
% aynı pencerede
plot([t‘/3 y'])
bar(t,y) ?(t’ye göre y)
```



Grafikler- Mesh

```
x=0:10:100
y=0:20:250
z=rand(13,11)
%[x,y] = meshgrid(x,y)
yuzey1 = surface(x,y,z)

% Hazır fonksiyon
z=peaks(30)
yuzey1 = surface(z)
```



Çizim - Genel

- 2-D vectors: `plot(x, y)`
  - `plot(0:0.01:2*pi, sin(0:0.01:2*pi))`
- 3-D: `plot3(x, y, z)` (space curve)
- Surfaces
  - `meshgrid` makes surface from axes, `mesh` plots it
    - ```
      [X,Y] = meshgrid(-2:.2:2, -2:.2:2);
      Z = X .* exp(-X.^2 - Y.^2);
      mesh(Z)
      ```
  - `surf`: Solid version of `mesh`
      ```

Bazı önemli deyimler ve fonk.

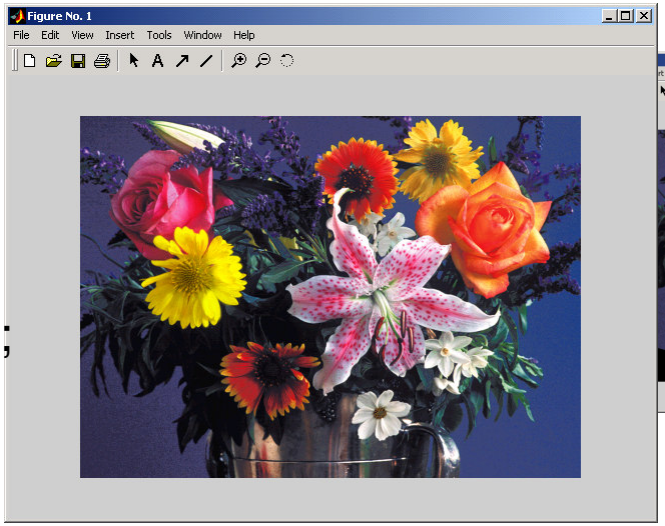
- Return
- Global
- Try..catch..end
- Sub Functions
- Set
- Figure
- Axes
- Max
- Min
- Mean
- Std
- Sort
- Factorial
- Sin, cos, tan, ...
- Exp, log, sqrt, ...

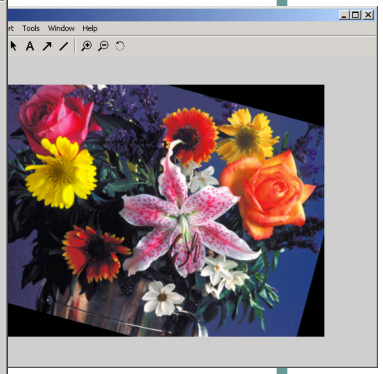
Görüntü İşleme

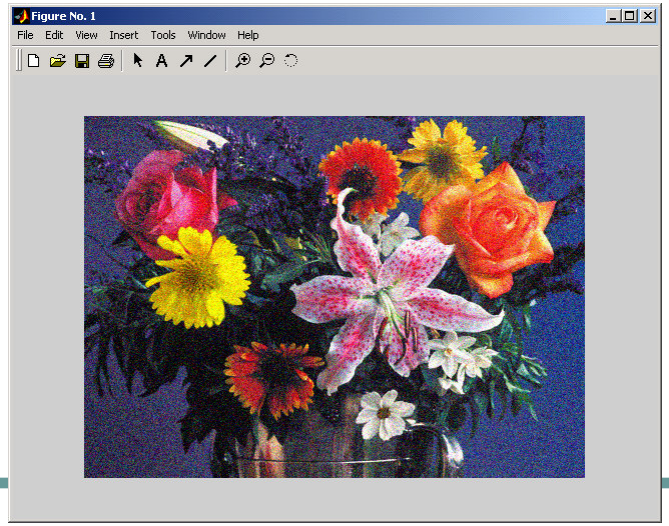
```
im1 = imread('flowers.tif')
im2 = imread('forest.tif')
imshow(im1)
J=imrotate(im1,-15,'bilinear','crop');
imshow(J);
imshow(imnoise(im1,'gaussian'))
```

İmsubtract ?

İmwrite ?







MATLAB‘da YSA (NN) Uygulamaları

Örnek 1.1

newff: a feed-forward backpropagation network

Example 1 :

- P = [0 1 2 3 4 5 6 7 8 9 10];
- T = [0 1 2 3 4 3 2 1 2 3 4];

- net = newff([0 10],[5 1],{'tansig' 'purelin'});

- Y = sim(net,P);
- plot(P,T,P,Y,'o')

Örnek 1.2

net = newff([0 10],[5 1],{'tansig' 'purelin'});

Net adlı bir ağ nesnesi oluşturduk.

[0 10] girdi elemanlarının min ve max değerleri 0 ile 10 arasında Rx2'lik bir matriste (R tane girdi değeri için).

İki katmanlı, İlk katmanda 5 giriş elemanı ve 1 çıkış elemanı var.

Üçüncü parametre, her bir katmanda kullanılan transfer fonksiyonları:

İlk katman, TANSIG(N) calculates its output according to:

n = 2/(1+exp(-2*n))-1 (tansig nöronları)

İkinci katman, PURELIN (Linear transfer function) nöronları.

Örnek 1.3

- NEWFF(PR,[S1 S2...SNl],{TF1 TF2...TFNl},BTF,BLF,PF) takes,
  PR - Rx2 matrix of min and max values for R input elements.
  Si - Size of ith layer, for Nl layers.
  TFi - Transfer function of ith layer, default = 'tansig'.
  BTF - Backprop network training function, default = 'trainlm'.
  BLF - Backprop weight/bias learning function, default = 'learngdm'.
  PF - Performance function, default = 'mse'.
  and returns an N layer feed-forward backprop network.

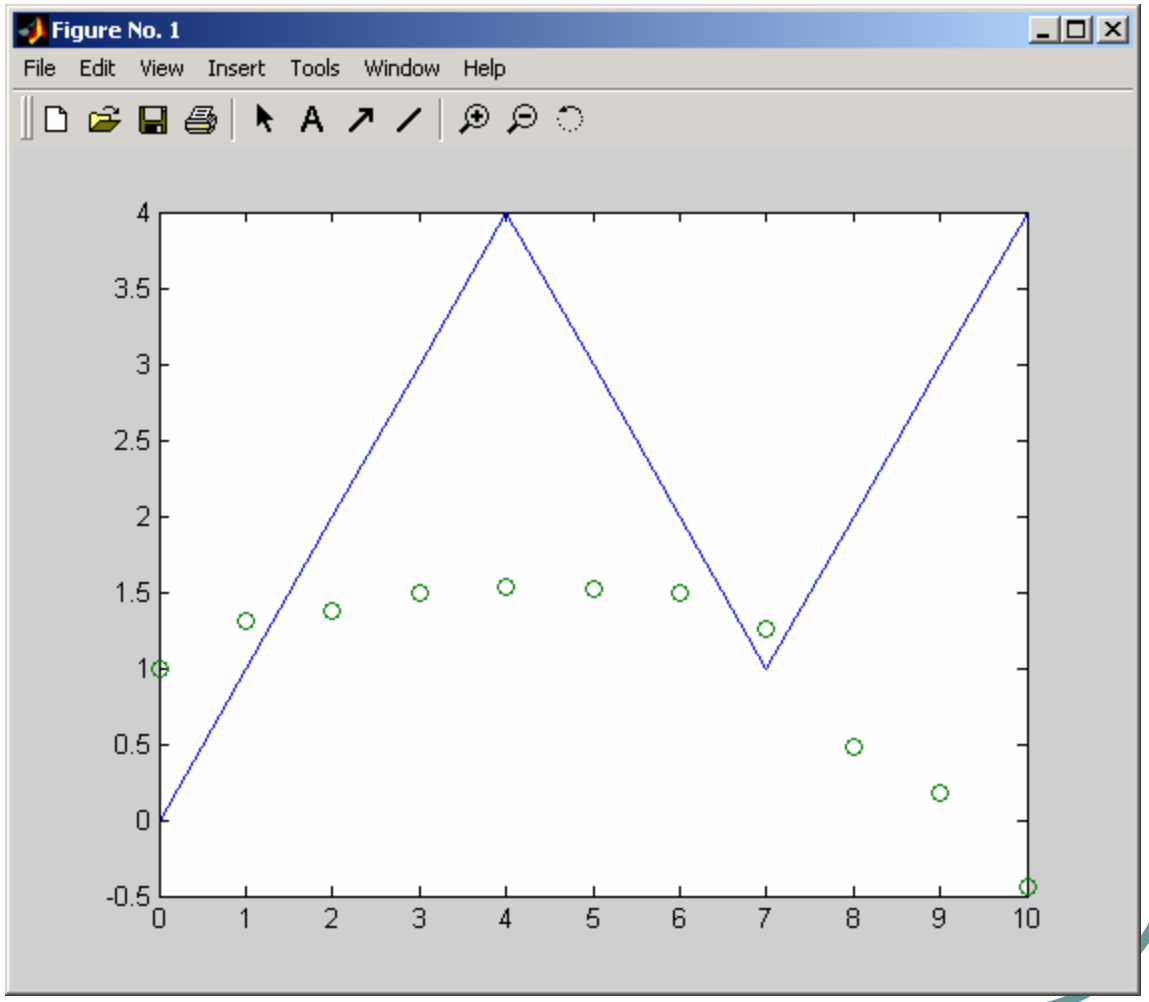
Örnek 1.4

- P girdileri
- T (target) hedefleri

- YSA simülasyonunun çıktısı ve hedeflenen çıktı çizdiriliyor.

Örnek 1.5

Her işletimde değişik değerler çıkar.



Geri Yayılımlı Öğrenme Modları

BP algoritmalarında kullanılabilecek iki mod

- Artırımlı Öğrenme modu : adapt komutu
  - Gradyanın hesaplanarak ağırlıkların güncellenmesi işlemi, her bir girdinin ağa uygulanması ile gerçekleştirilir.
- Grup Öğrenme modu : train komutu
  - Ağırlıklar, tüm eğitim seti ağa uygulandığında gerçekleştirilir.

Traingd, Learngd'nin eşleniği

Traingdm, Learngdm'nin eşleniği

Epoch : İterasyon sayısı

Örnek 2.1 (Grup Modu)

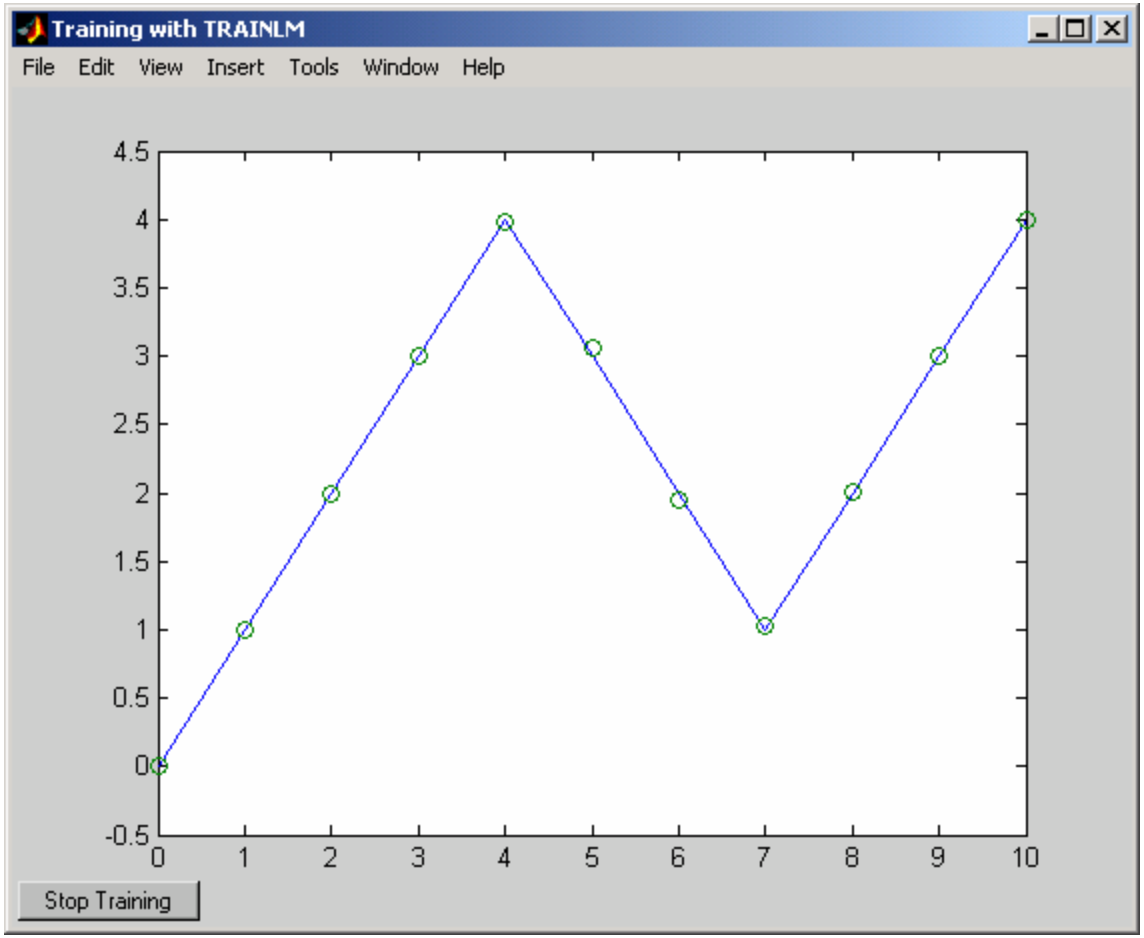
```
P = [0 1 2 3 4 5 6 7 8 9 10];
T = [0 1 2 3 4 3 2 1 2 3 4];

net = newff([0 10],[5 1],{'tansig' 'purelin'});

net.trainParam.epochs = 50;
net = train(net,P,T);
Y = sim(net,P);
plot(P,T,P,Y,'o')
```

Örnek 2.2

Değerlere Yakın!



Bazı Öğrenme Kuralları

GRADYAN Azaltımı Öğrenme Kuralı :

LEARNGD Gradient descent weight/bias learning function.

Learngd fonksiyonunun lr (öğrenme katsayısı) adlı parametresinin default değeri 0.20'dir. Öğrenme katsayısı büyük seçilirse, öğrenme algoritması kararsız olur; küçük seçilirse ağın sonuca yakınsama süresi artar.

Momentum Kullanan GRADYAN Azaltımı Öğrenme Kuralı :

LEARNGDM Gradient descent w/momentum weight/bias learning function.

Momentum kullanılmadığında ağ, yerel bir minimuma takılarak salınımlar yapabilir. Momentum kullanıldığında ise, sıçrama imkanı kazanabilir. Momentum 0 ile 1 arasındadır. Momentum değeri 0 ise, ağırlık değişimi tamamen gradyana bağımlıdır.

Tahminleme Örneği

f(x,y) = 1/(x+y) fonksiyonunun tahminlemesi

x	y	f(x,y)
1	1	0.5
1	3	0.25
4	1	0.2
3	2	0.2
5	5	0.1

```
input=[1 1 4 3 5; 1 3 1 2 5];
target=[0.5 0.25 0.20 0.20 0.10];
net=newff(minmax(input),[2,1],{'logsig','purelin'},'trainl
    m');
net.trainParam.goal=1e-5;
[net,tr]=train(net,input,target);
sim(net,[3;2])
sim(net,[1;2]) % sim : 0.3370      hedef : 0.3333
```

Test Verisi

sim(net,[3;2])

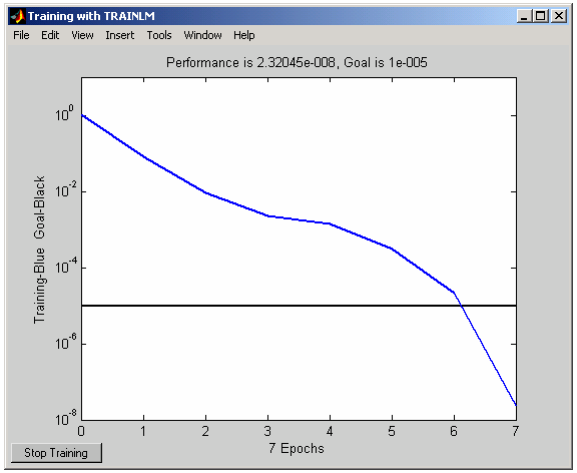
TRAINLM, Epoch 0/100, MSE 1.03675/1e-005, Gradient 6.62807/1e-010

TRAINLM, Epoch 7/100, MSE 2.32045e-008/1e-005, Gradient 5.43983e-005/1e-010

TRAINLM, Performance goal met.

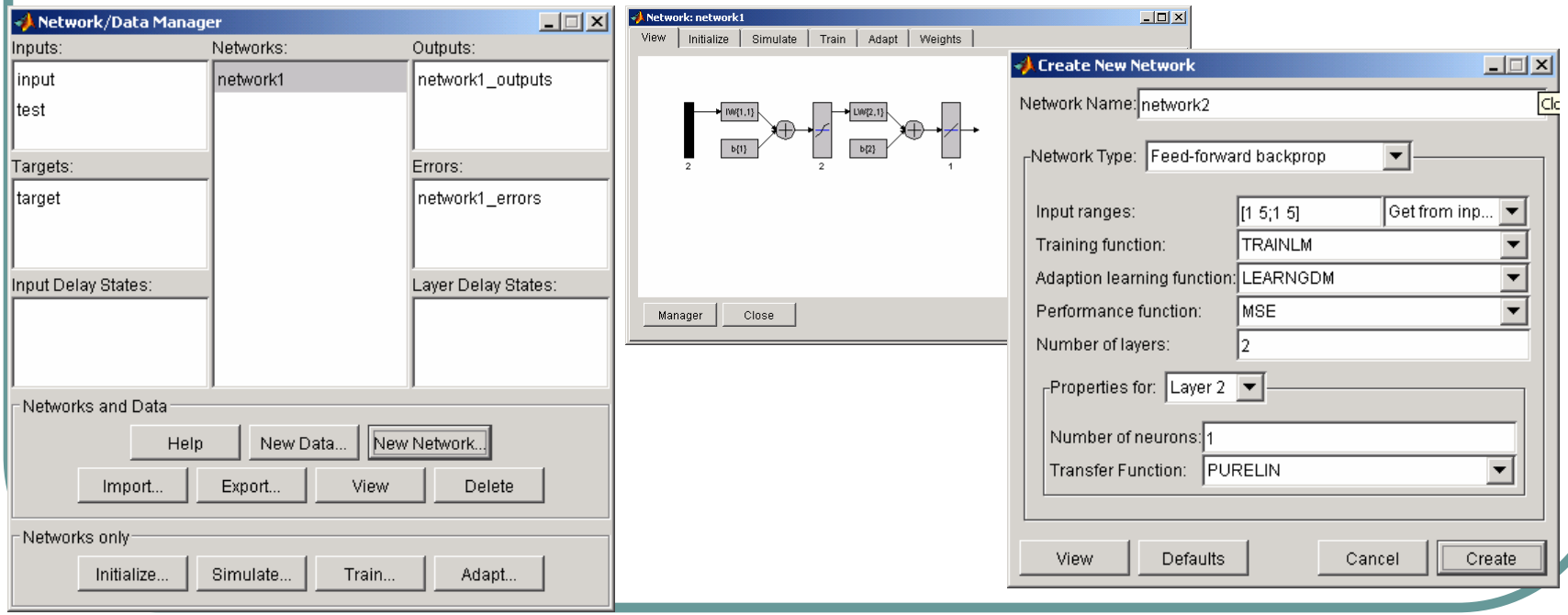
ans =

0.2002



NN Araç Kutusu ile Yapılması

- input ve target ile test verileri girilir, ağ (network) oluşturulur. Parametreler ayarlanır. Ağ Eğitilir (train) ve Test verisi simüle edilir.



Artırımlı Mod Örneği

```
p=[1 1 4 3 5; 1 3 1 2 5];
t=[0.5 0.25 0.20 0.20 0.10];
net=newff(minmax(p),[2,1],{'logsig','purelin'},'learngdm');
p=num2cell(p,1);
t=num2cell(t,1);
net.biases{1,1}.learnFcn='learngdm'
net.inputWeights{1,1}.learnFcn='learngdm‘ % Ağırlıkları ayarlama
net.layerWeights{2,1}.learnFcn='learngdm‘
net.biases{2,1}.learnFcn='learngdm‘ % Eşikleri ayarlama
% Öğrenme boyunca eğitim setinin kaç defa işleneceği
net.adaptparam.passes = 1000 % Ayarlayiniz.
net.layerWeights{2,1}.learnParam.lr=0.25 %Ayarlayiniz
[net,a,e] = adapt(net,p,t); a=sim(net,p)
```

Basit Örnekler

- $f(x) = \sin(x)$
- $f(x,y,z) = x+y^2+z^3$
- $f(x,y) = \sin(x) + 1/y$

- Değişik katman, nöron sayıları ve parametreler ile test ediniz.
- Demolar ve Help