تصميم شبكة عصبونية لحساب مساحة أسطوانة

```
>> network4
network4 =

  Neural Network

          name: 'Custom Neural Network'
      userdata: (your custom info)

  dimensions:

       numInputs: 1
       numLayers: 10
      numOutputs: 1
  numInputDelays: 0
  numLayerDelays: 0
numFeedbackDelays: 0
numWeightElements: 11025
      sampleTime: 1

  connections:

     biasConnect: [1; 1; 1; 1; 1; 1; 1; 1; 1; 1]
    inputConnect: [1; 0; 0; 0; 0; 0; 0; 0; 0; 0]
    layerConnect: [10x10 boolean]
   outputConnect: [0 0 0 0 0 0 0 0 0 1]

  subobjects:

         input: Equivalent to inputs{1}
        output: Equivalent to outputs{10}
```

```
       inputs: {1x1 cell array of 1 input}
       layers: {10x1 cell array of 10 layers}
      outputs: {1x10 cell array of 1 output}
       biases: {10x1 cell array of 10 biases}
 inputWeights: {10x1 cell array of 1 weight}
 layerWeights: {10x10 cell array of 9 weights}

functions:

     adaptFcn: 'adaptwb'
   adaptParam: (none)
     derivFcn: 'defaultderiv'
    divideFcn: 'dividerand'
  divideParam: .trainRatio, .valRatio, .testRatio
   divideMode: 'sample'
      initFcn: 'initlay'
   performFcn: 'mse'
 performParam: .regularization, .normalization
     plotFcns: {'plotperform', plottrainstate,
               plotregression}
   plotParams: {1x3 cell array of 3 params}
    trainFcn: 'trainlm'
   trainParam: .showWindow, .showCommandLine, .show, .epochs,
               .time, .goal, .min_grad, .max_fail, .mu, .mu_dec,
               .mu_inc, .mu_max

weight and bias values:

          IW: {10x1 cell} containing 1 input weight matrix
          LW: {10x10 cell} containing 9 layer weight matrices
```

b: {10x1 cell} containing 10 bias vectors

methods:

adapt: Learn while in continuous use

configure: Configure inputs & outputs

gensim: Generate Simulink model

init: Initialize weights & biases

perform: Calculate performance

sim: Evaluate network outputs given inputs

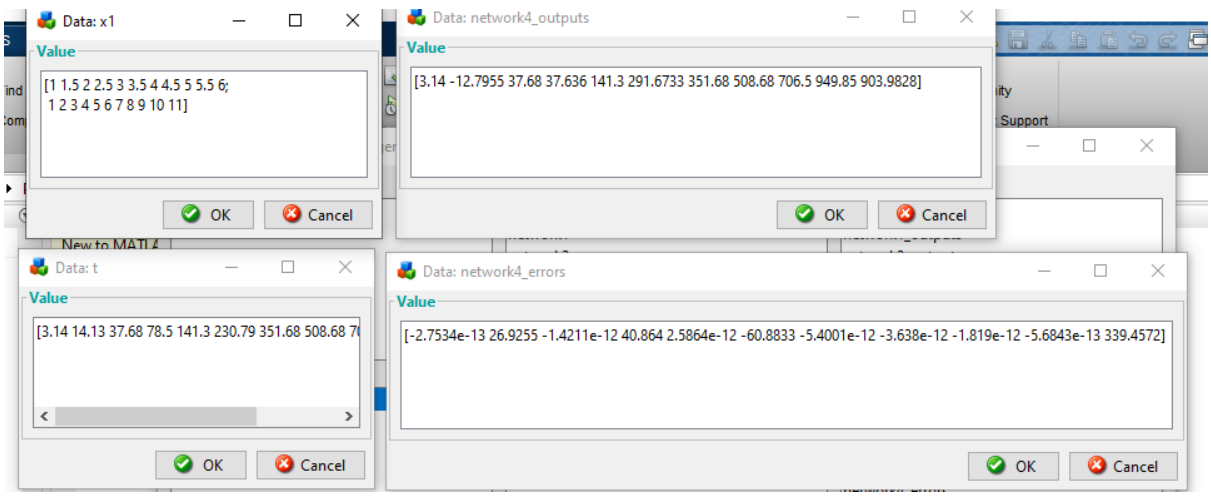
train: Train network with examples

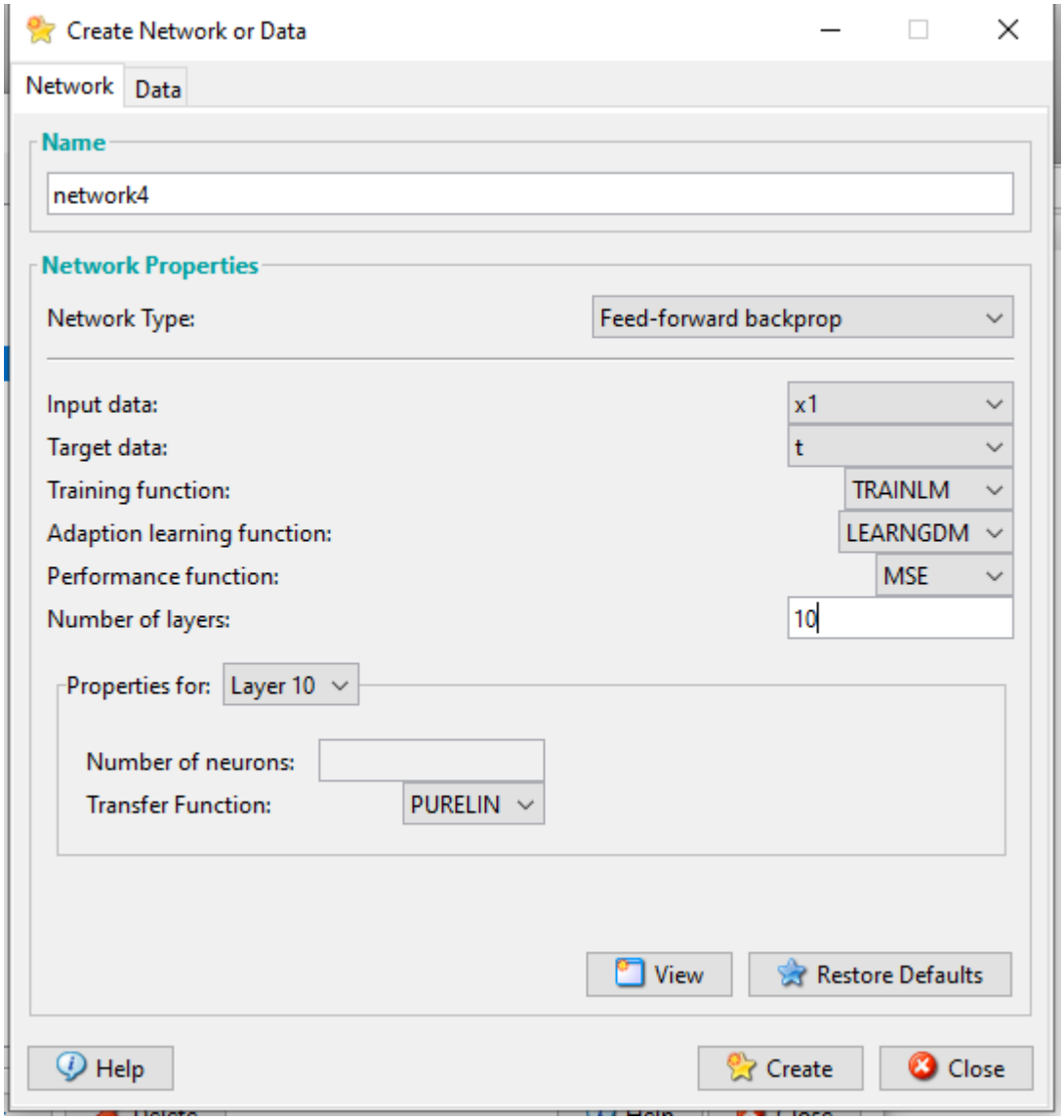
view: View diagram

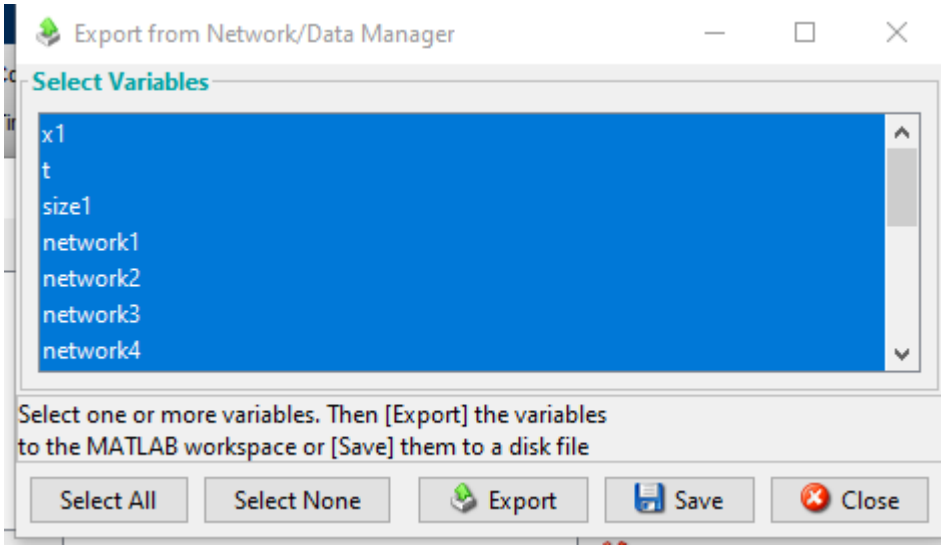
unconfigure: Unconfigure inputs & outputs

evaluate: outputs = network4(inputs)

>>




Create Network or Data
Network
Data
Name
network4
Network Properties
Network Type:
Feed-forward backprop
Input data:
x1
Target data:
t
Training function:
TRAINLM
Adaption learning function:
LEARNGDM
Performance function:
MSE
Number of layers:
10
Properties for:
Layer 10
Number of neurons:
Transfer Function:
PURELIN
View
Restore Defaults
Help
Create
Close


Export from Network/Data Manager
Select Variables
x1
t
size1
network1
network2
network3
network4
Select one or more variables. Then [Export] the variables
to the MATLAB workspace or [Save] them to a disk file
Select All
Select None
Export
Save
Close

Neural Network Training (nntraintool)

Neural Network

Algorithms

Data Division:	Random	(dividerand)
Training:	Levenberg-Marquardt	(trainlm)
Performance:	Mean Squared Error	(mse)
Calculations:	MEX	

Progress

Epoch:	0	6 iterations	1000
Time:		0:02:05	
Performance:	3.17e+06	7.83e-24	0.00
Gradient:	3.31e+07	2.44e-08	1.00e-07
Mu:	0.00100	1.00e-07	1.00e+10
Validation Checks:	0	0	6

Plots

Performance (plotperform)

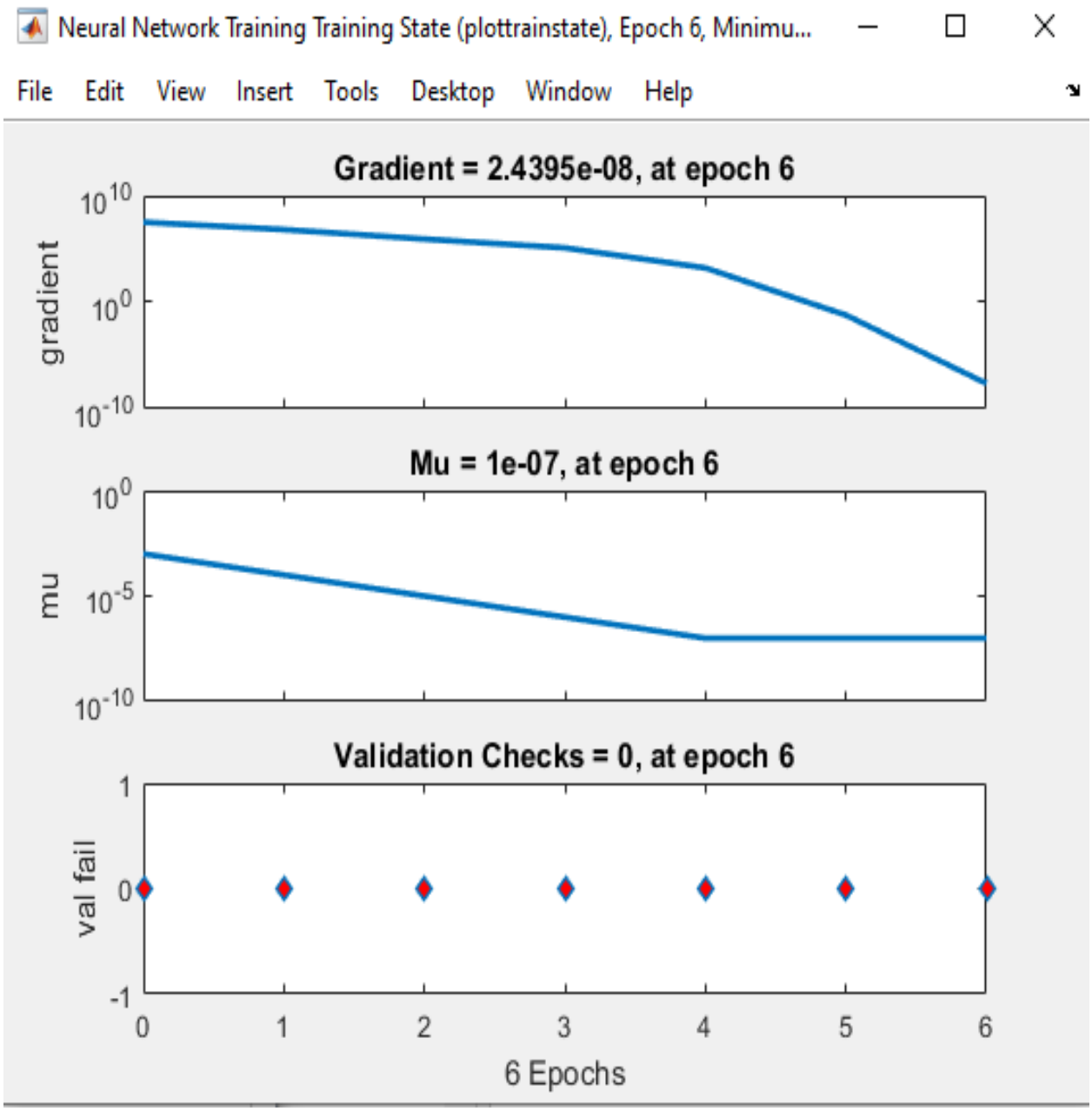
Training State (plottrainstate)

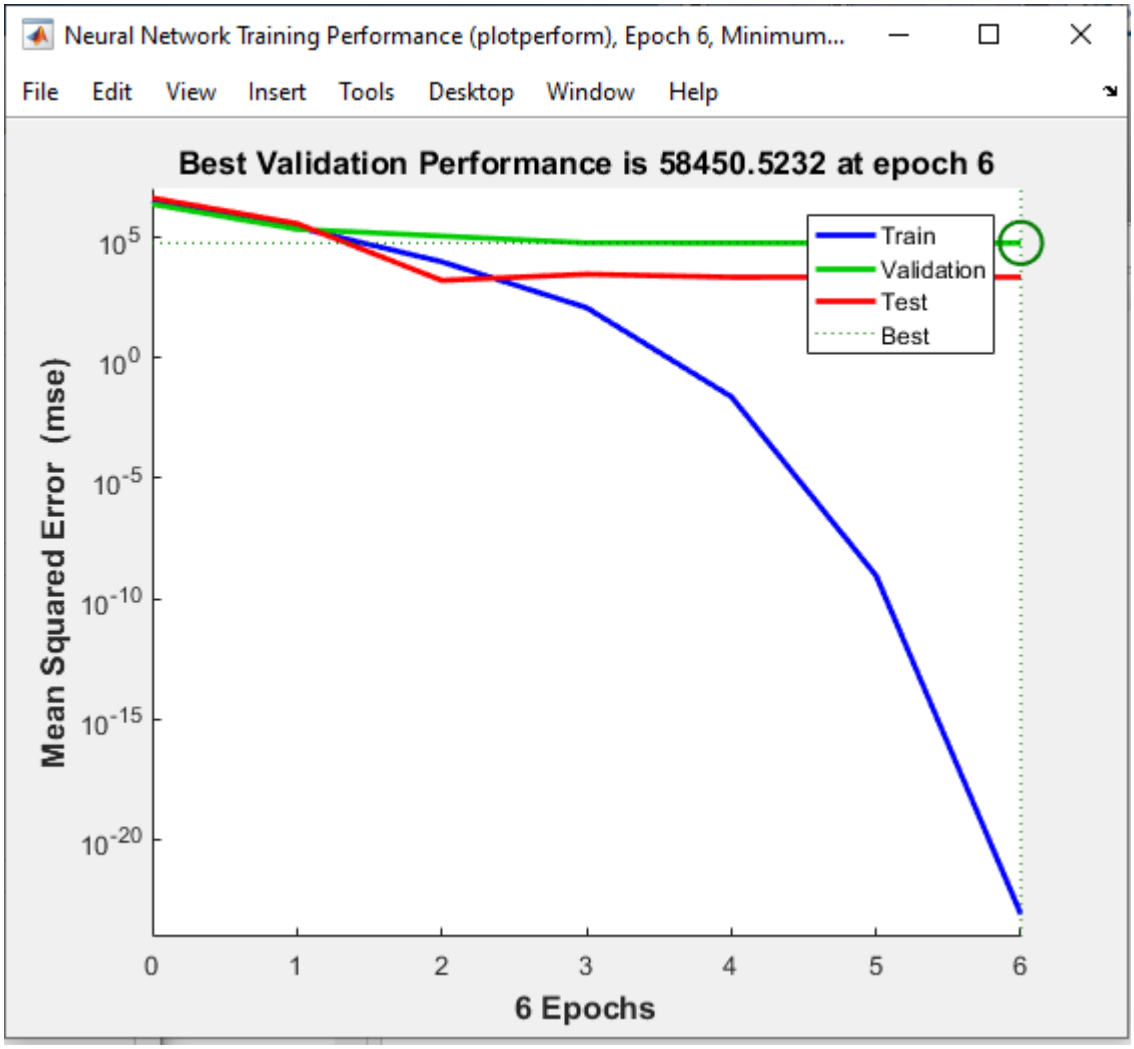
Regression (plotregression)

Plot Interval: 1 epochs

Minimum gradient reached.

Stop Training Cancel


Neural Network Training Training State (plottrainstate), Epoch 6, Minimu...
File Edit View Insert Tools Desktop Window Help
Gradient = 2.4395e-08, at epoch 6
gradient
10^10
10^0
10^-10
Mu = 1e-07, at epoch 6
mu
10^0
10^-5
10^-10
Validation Checks = 0, at epoch 6
val fail
1
0
-1
0 1 2 3 4 5 6
6 Epochs


Neural Network Training Performance (plotperform), Epoch 6, Minimum...
File Edit View Insert Tools Desktop Window Help
Best Validation Performance is 58450.5232 at epoch 6
Train
Validation
Test
Best
Mean Squared Error (mse)
10^5
10^0
10^-5
10^-10
10^-15
10^-20
0 1 2 3 4 5 6
6 Epochs

File Edit View Insert Tools Desktop Window Help