

**DEMOCRITUS UNIVERSITY OF THRACE
DEPARTMENT OF FORESTRY AND MANAGEMENT OF THE ENVIRONMENT AND NATURAL RESOURCES
LAB OF FOREST INFORMATICS**

LAZAROS ILIADIS

ASSOCIATE PROFESSOR OF DUTH

**THE GRAPHICAL INTERFACE OF NEURAL NETWORK TOOLKIT IN MATLAB
AND APPLICATIONS**

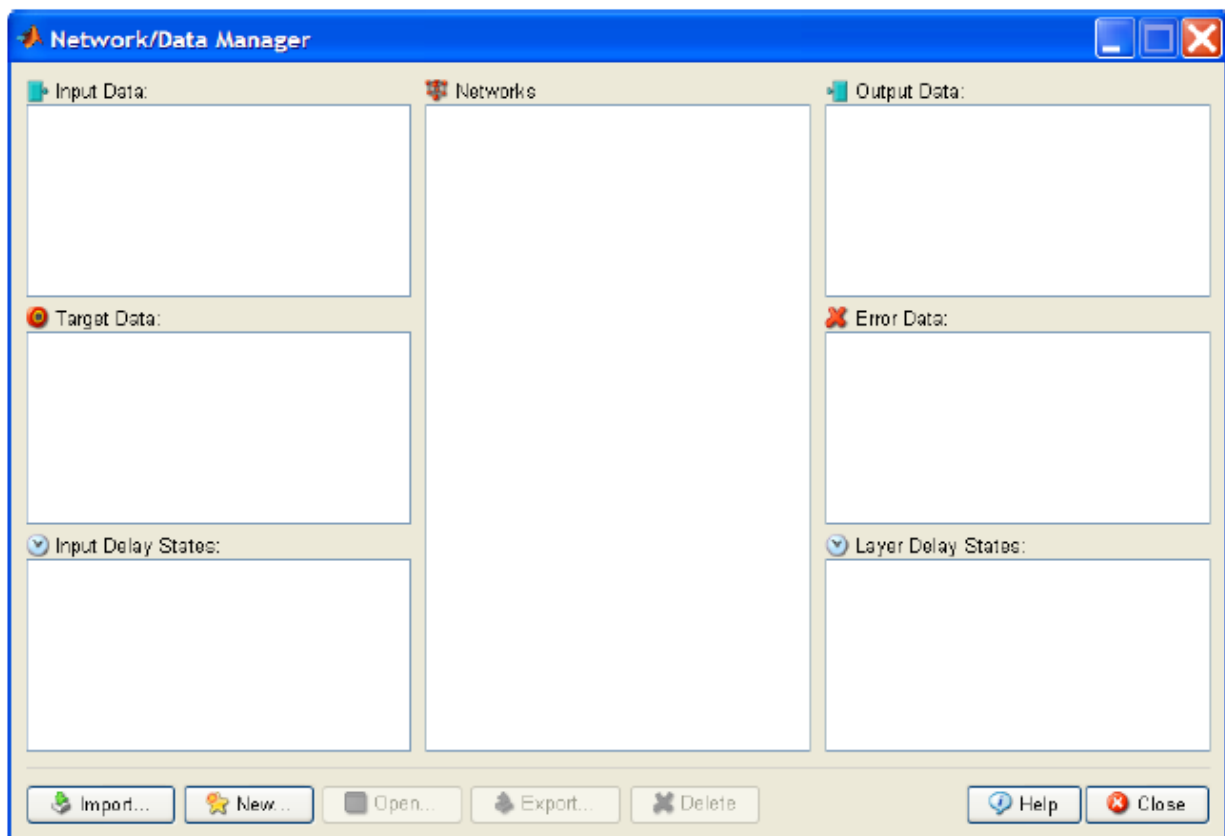
CONTENTS

Creating a Perceptron network (nntool)	- 3 -
Input and Target data	- 4 -
Creation of the network.....	-5-
Training the perceptron.....	- 6 -
Simulating the perceptron.....	- 7 -
Exporting results in the Workspace.....	- 8 -
Cleaning the window Network/Data Manager.....	- 9 -
Input data from the command line.....	-10-

Introduction to the Graphical Interface GUI of Neural Network Toolkit

The development and the use of Artificial Neural Networks is enabled through the use of NNToolKit. The graphical user interface has been designed to be user friendly and simple.

The Network/Data Manager window is a basic part of the GUI. This window has its own work space distinct from the command line of MATLAB workspace. During its use we can import data from the command prompt of MATLAB and also we can export results from the Network Data Manager to the command prompt and store them in Tables.



The following example shows the creation of a perceptron network. It follows all the steps of an ANN creation and it is quite descriptive step by step.

Creating a Perceptron network (nntool)

In this example we will create a perceptron which will perform the logical operation AND... This means that it will perform a deterministic task. The input vector will be Input = [0 0 1 1; 0 1 0 1], and the output vector will be Target = [0 0 0 1]. The Input table has two lines because the network will have two inputs. The columns of the network correspond to the four input cases that are produced by combining the two potential inputs which are 0 and 1

In the command prompt workspace we type the command:

```
>>nntool
```

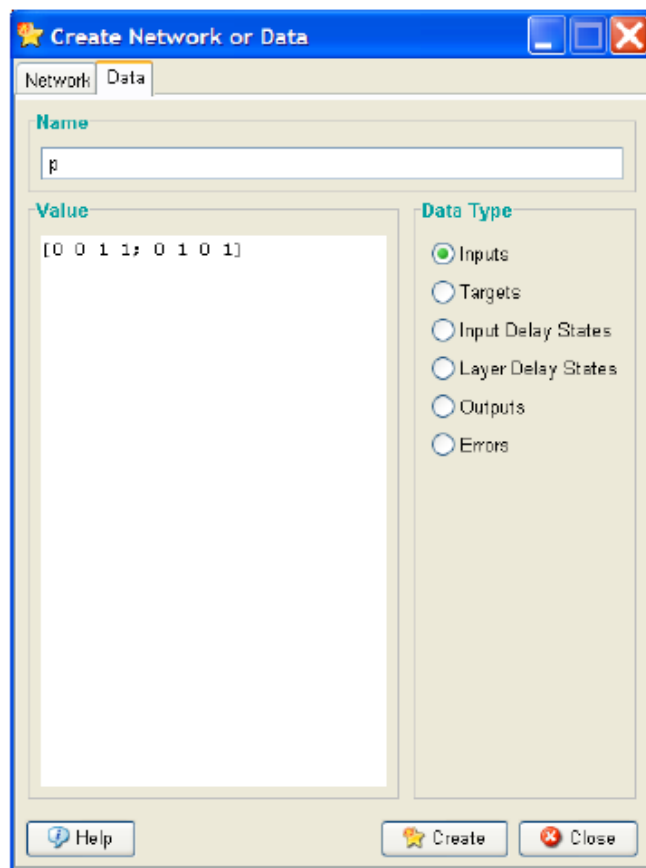
The Network/Data manager window appears. In this window we can chose the help option to go around and see the description of the operations performed by various command buttons.

Input and Target data

First we create the data that will be used as input to the network. The [New] button is pressed and we see that the “**Create Network or Data**” window appears. We chose the Data cart.

We chose from the option “Data Type” the option button “Inputs” and then we define the data table with the name p in the “Name” field. In the field “Value” we give values in the table as follows:

[0 0 1 1; 0 1 0 1]

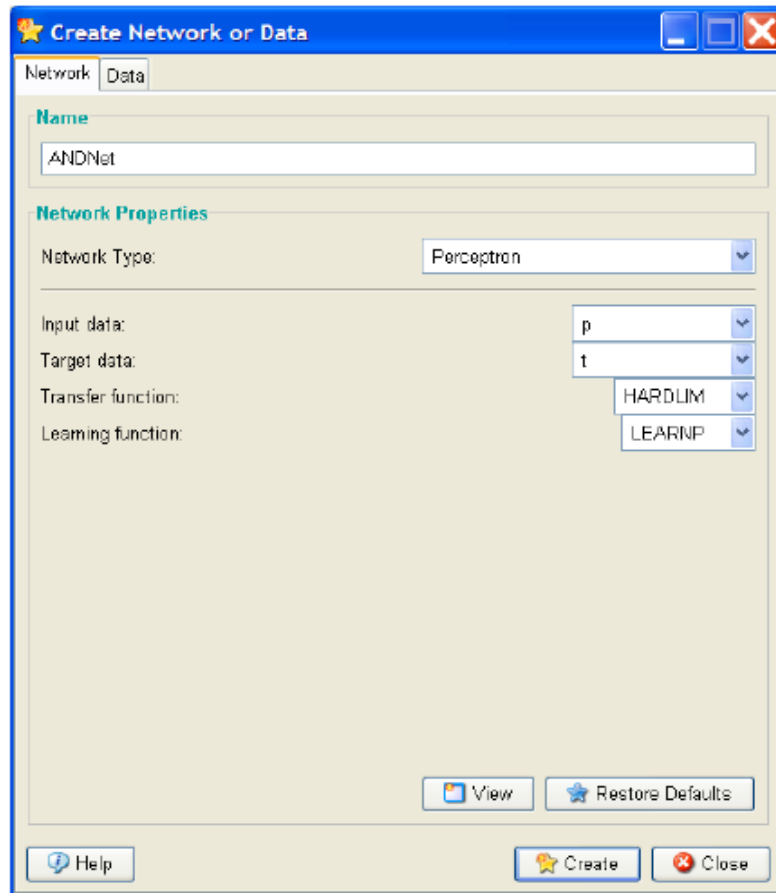


Click on the button “Create” and then [OK] in order to input table p in the folder “Input Data” of the Network Data manager.

Then based on the logic of the AND operator we create the target values “network target”. The name of the table to contain the target values is t and the values are [0 0 0 1]. Finally the data type is chosen to be “Target”.

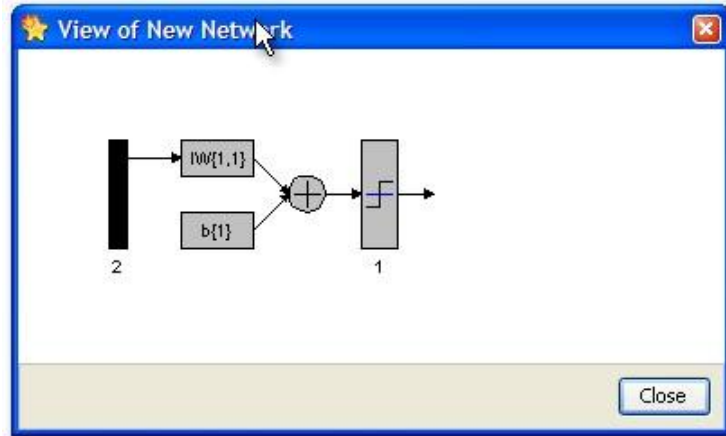
Creation of the network

To create the network we chose the card “Network” in the “Create Network or Data” window.



In the “Name” field we name the network as “ANDNet”. From the “Network Type” catalogue we chose the option “Perceptron”.

We chose the table p from the catalog “Input data”, and the table t from the catalogue “Target data” respectively. We can chose a transfer function (e.g. hardlim) with output vector [0, 1], and also a learning (optimization) algorithm. We chose HARDLIM as a transfer function and the LEARNP as a Learning function.



We press the “Create” button and then we press OK. The new perceptron network has been added in the Networks catalogue of the Network/Data Manager window under the name ANDNet.

Training the perceptron

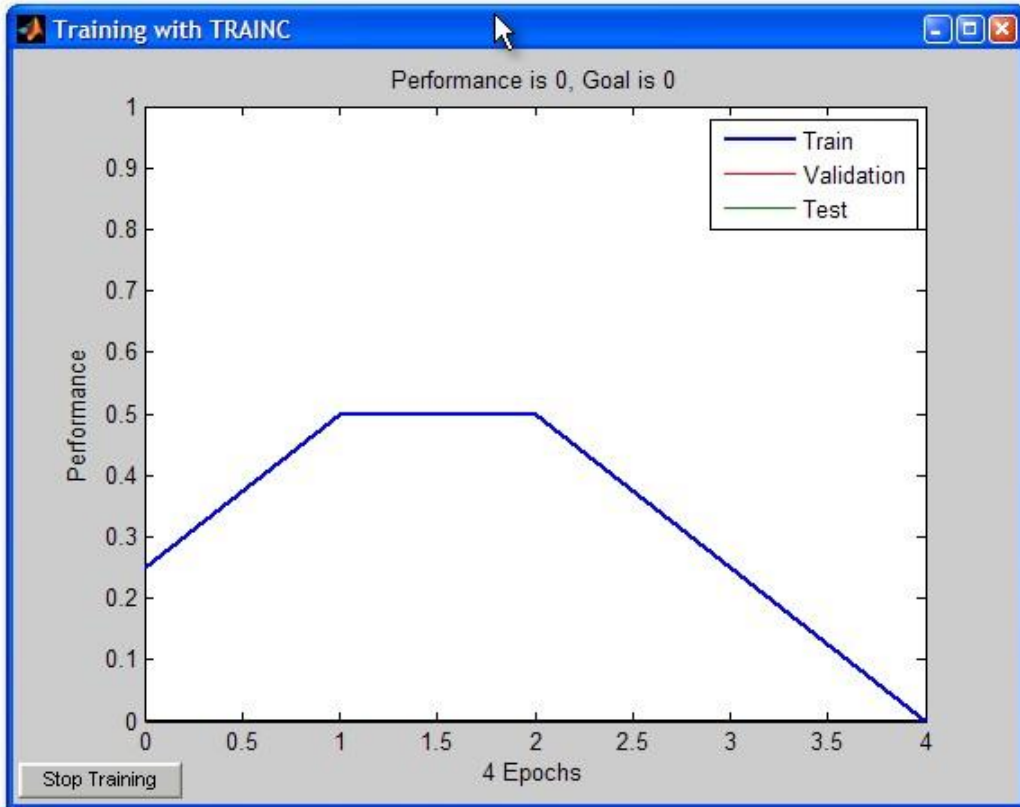
To train the network we chose to click the ANDNet from the catalogue Networks of the Network/Data Manager and we click [Open]. A new window the “Network:ANDNet” opens.

The window titled "Network: ANDnet" has a menu bar with "View", "Train", "Simulate", "Adapt", "Reinitialize Weights", and "View/Edit Weights". Below the menu bar are two tabs: "Training Info" and "Training Parameters". The "Training Parameters" tab is active and contains two main sections: "Training Data" and "Training Results".

Training Data		Training Results	
Inputs	p	Outputs	ANDnet_outputs
Targets	t	Errors	ANDnet_errors
Init Input Delay States	(zeros)	Final Input Delay States	ANDnet_inputStates
Init Layer Delay States	(zeros)	Final Layer Delay States	ANDnet_layerStates

A "Train Network" button is located at the bottom right of the window.

The Training Results are stored in the variable (vector) **Outputs** and the deviations from the target values are given in the variable (vector) **Errors**. We can export both vectors to the MATLAB workspace. In the next graphical display we can see the Mean Square Error per Training Epoch.



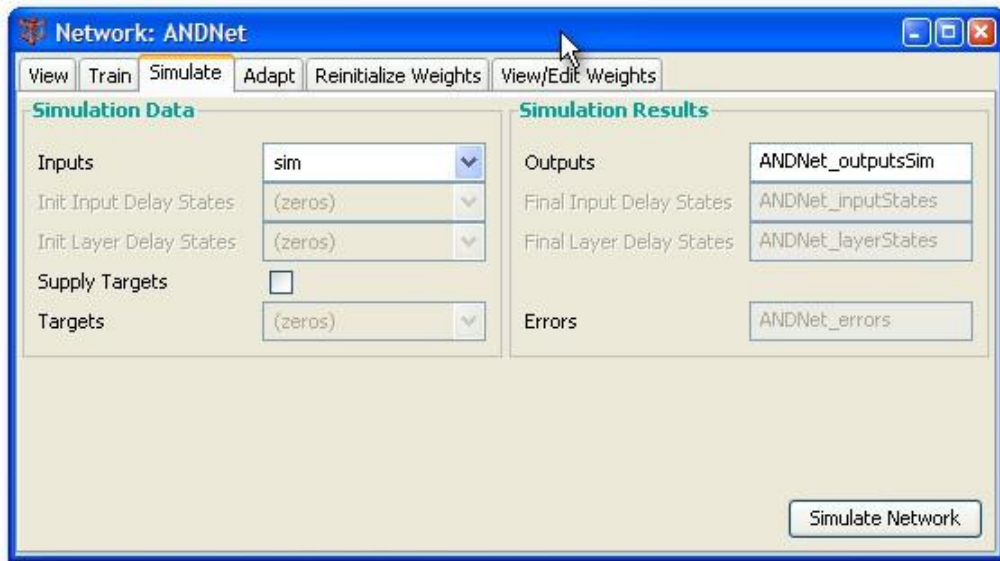
The network in this example has been trained to accept and tolerate only zero error in four Epochs. Other networks usually are not developed and trained by using zero error. However in this case we are working in a deterministic problem so we do not wish to have any kind of error. Usually the deviations of the produced values from the actual ones can have a large magnitude.

In the ANDNet_outputs vector we store the training output results which will be compared to the desired vector. In the ANDNet_errors variable (Vector) we store the errors and the deviations of the produced from the desired values

Simulating the perceptron network

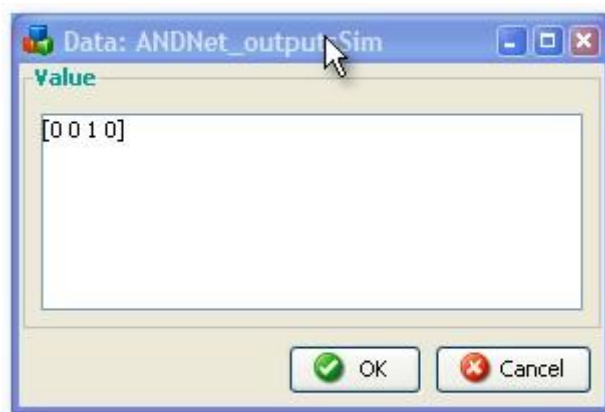
To recall and use the developed network we have to create the input vector first. We name the input vector as Sim and we assign it the following values [0 0 1 1; 1 1 1 0].

The simulation is done by choosing the card Simulate of the window "Network:ANDNet" which appears when we chose the network option form the "Networks" catalogue and we click in the option Open:



Here we define as Inputs the variable `sim` and as target the variable `Outputs`. Also we define a new variable named “ANDNet_outputsSim”, if we wish the results to be stored in another variable than the `ANDNet_outputs` that is used by default. Finally we click the option “Simulate Network” to perform the simulation.

The results of this simulation are stored in the “ANDNet_outputsSim” option that has been added in the Output Data catalogue of the central window “Network/Data manager”. We click the name of the variable and then we chose “Open” to see the results.



Exporting the results in the MATLAB Workspace.

In order to export the variable (vector) “ANDNet_outputs” and the variable (vector) “ANDNet_errors” that contain the results and the errors of the ANN simulation process respectively in the workspace of MATLAB, we return in the “Network/Data Manager” window. The variable `ANDNet_outputs` and `ANDNet_errors` which correspond to the ANDNet ANN are recorded to the fields `Outputs` and `Errors` respectively. We click the option “Export” of the window “Network/Data Manager”.

In the window "Export» from Network/Data Manager we see a list of the existing variables. We click in the "ANDNet_outputs" and in the "ANDNet_errors" to select them and then we click in the Export option. Copies of these two variables are exported in the MATLAB workspace. We go to the MATLAB workspace and we type the command who to see all of the variables that are included in the workspace. This is a way to confirm that everything went right in the process.

The result of the "who" command is the following:

```
ANDNet_errors
```

```
ANDNet_outputs
```

Then we type ANDNet_outputs and we receive::

```
ANDNet_outputs =
```

```
0 0 0 1
```

If we type ANDNet_errors we receive:

```
ANDNet_errors =
```

```
0 0 0 0
```

We can export p, t, and ANDNet in a similar manner. Then we can check if they are included in the workspace again by using "who". After exporting the ANDNet network in the workspace we can see the description of the network and check all the details. For example we can check even the weight matrix of the network by using the following command.

```
ANDNet.iw{1,1}  
that gives:
```

```
ans =
```

```
2 1
```

Also we can see the bias bias by using the command:

```
ANDNet.b (1)
```

That gives the output ans = -3

Cleaning the Network/Data Manager

We can clean the Network/Data Manager by choosing a variable and chose delete. We can delete all variables. In this way we can start from a clean workspace. Alternatively if we close MATLAB and we start it again, the nntool command offers a clear Network/Data Manager window. However the variables p,t, and all others that we have defined and exported

in the workspace are still there even after the clearance of the Network/Data Manager window.

Input of data from the command line

In the command line of the workspace of MATLAB we can create a new vector in the following way:

```
r = [0; 1; 2; 3]
```

```
r =
```

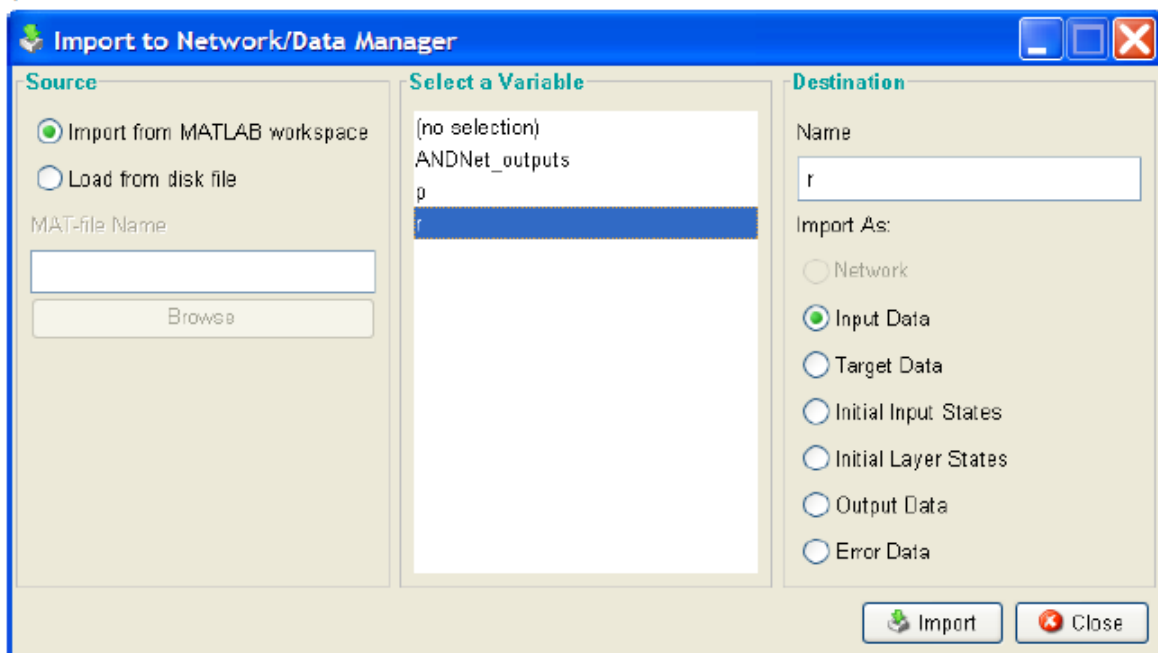
```
0
```

```
1
```

```
2
```

```
3
```

In the window “Network/Data Manager” we can click in the Input button. Then the window “Import to Network/Data Manager” opens.



In this window we can select by clicking the variable (vector) named `r` which now appears in the catalogue “Select a Variable” and we give destination name again the same `r` (the same as the name of the variable in the workspace so that it is possible to relate them).

From the Import As options we chose “Input Data” and then we click Import. In the catalogue Input Data of the window Network/Data Manager the variable `r` has been defined.