NNTOOLBOX

FERRAMENTA DO MATLAB PARA REDES NEURONAIS

Conhecimento e Raciocínio
DEIS/ISEC

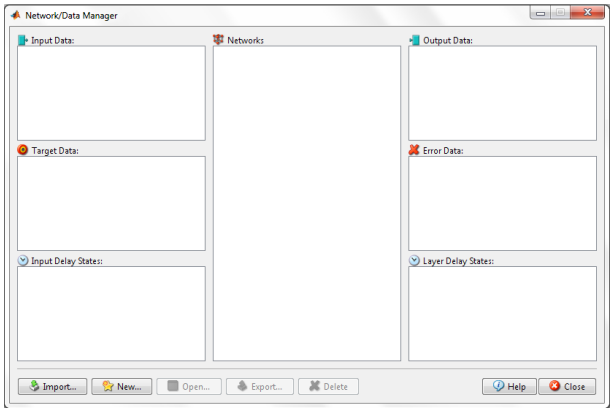
Neural Network ToolBox

- Ferramenta do Matlab com funções próprias para manipulação de redes neuronais
- Possui uma janela de edição que disponibiliza um interface gráfico para:
  - Criar, configurar, visualizar, treinar e testar redes neuronais
- As redes criadas e os resultados obtidos podem ser exportados para o *workspace* do Matlab.

NNtool

Primeiro exemplo: criar e treinar um perceptrão para a função lógica AND

- escrever na linha comandos do Matlab: **>>nntool**



A função lógica AND é definida por:

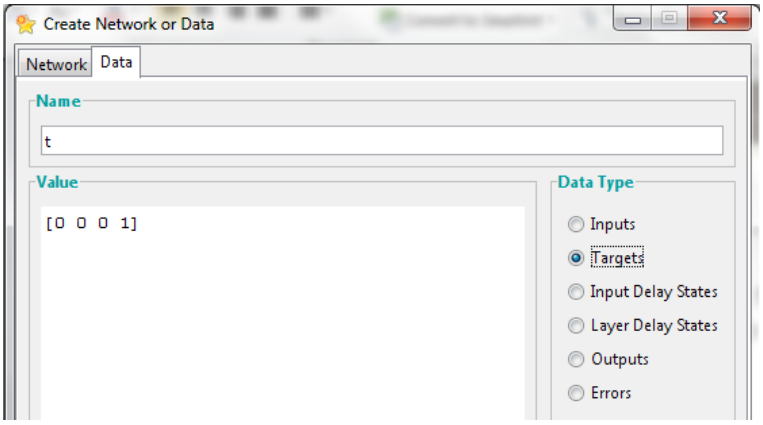
AND

I_1	I_2	t
0	0	0
0	1	0
1	0	0
1	1	1

- Clicar em **New**
- Selecionar o separador **tab Data**
  - Definir os parâmetros *name*, *value* e *data type* como se segue para os dados de treino

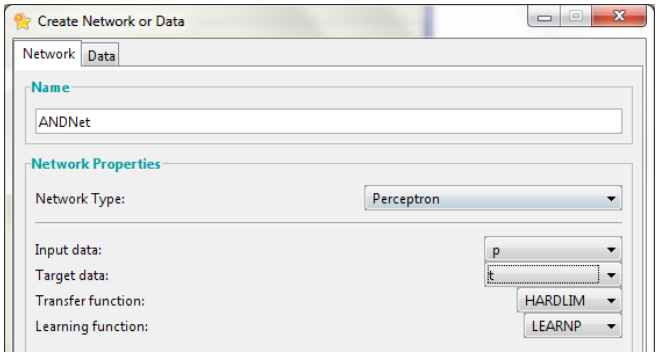
  - Clicar **Create** e depois clicar **OK**

- Ainda no **separador Data**
  - Definir os parâmetros *name*, *value* e *data type* como se segue para os alvos (*targets*) usados no treino

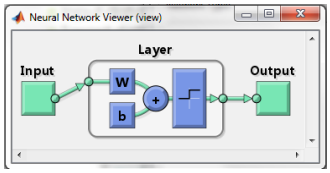


  - Clicar **Create** e depois clicar **OK**

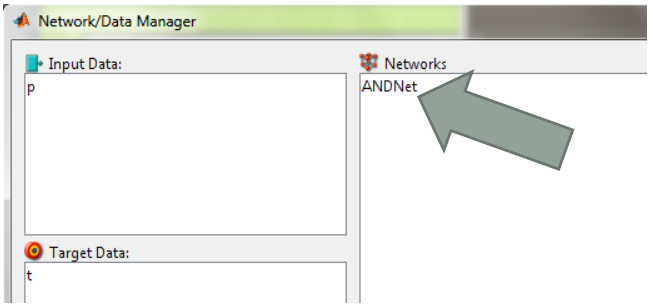
- No **separador Network** definir
  - Name: ANDNet
  - Network type: perceptron
  - Input data: **p**
  - Target data: **t**
  - Transfer function: hardlim
  - Learning function: learnp



- Para ver a rede clicar em **View**

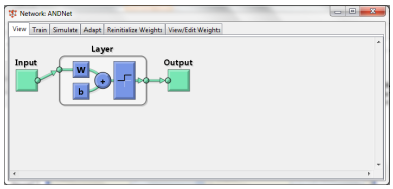


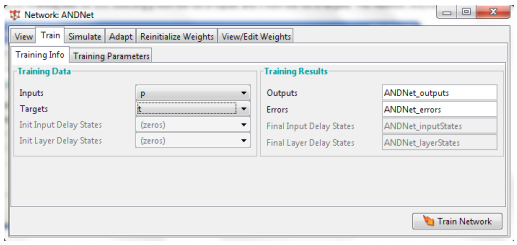
- Para gerar a rede neuronal clicar em **Create**, depois **OK**. No interface principal está agora a rede criada



Para treinar a rede

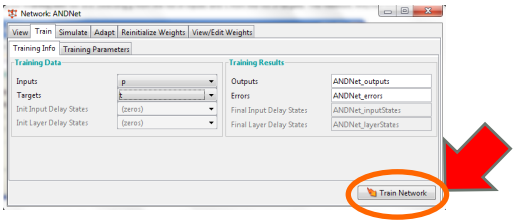
- Seleccionar a rede (clique simples sobre o nome)
- Clicar em **Open**
- No **separador *Train***
  - **Definir Inputs: p**
  - **Definir Targets: t**





NOTA: Na *tab Training Parameters* pode especificar outros parâmetros do treino como sejam #épocas, erro final aceitável para treino (goal), etc

- Para inicializar o treino clicar em **Train Network**

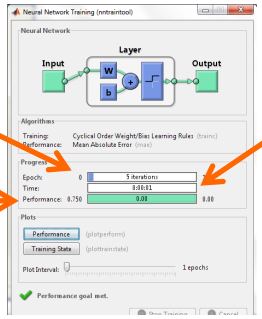


- No final do treino deve obter algo semelhante à imagem que se segue

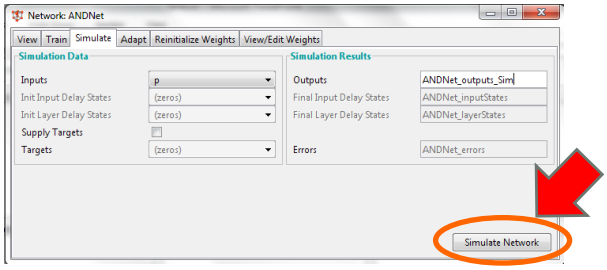
Épocas dispendidas no treino (= 5)

Tempo gasto no treino

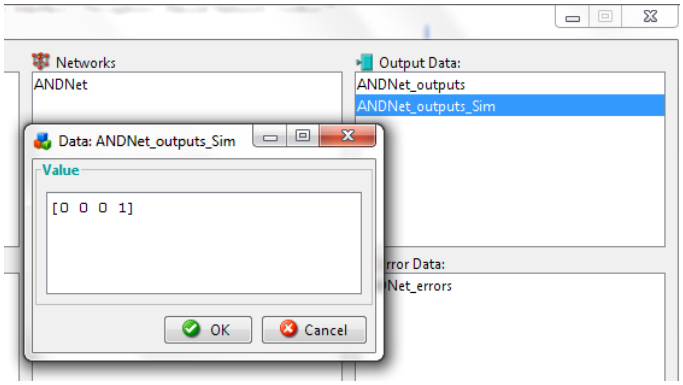
Erro de predição da rede no final do treino (= 0)



- Pode testar a rede para verificar que o erro é 0
  - Voltar ao interface principal e fazer duplo clique sobre o nome da rede
  - No separador **Simulate** definir:
    - Inputs: **p**
    - Outputs: ANDNet_outputs_Sim (para não confundir com as saídas produzidas durante o treino anteriormente executado)
    - Clicar **Simulate Network** no canto inferior direito e depois **OK**



- No interface principal fazer duplo clique na variável ANDNet_outputs_Sim
  - As saídas previstas são todas zero excepto para o último caso, tendo assim o modelo previsto os grupos de pertença com 100% de acerto



Segundo exemplo: XOR

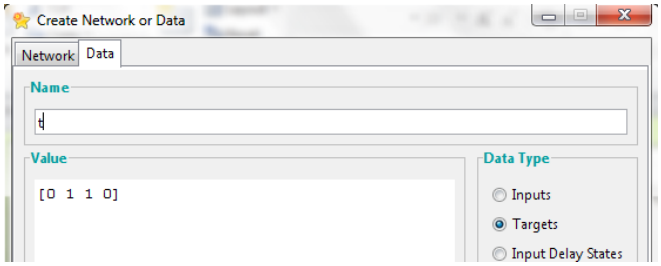
XOR

I_1	I_2	t
0	0	0
0	1	1
1	0	1
1	1	0

- Clicar em **New**
- Seleccionar o separador **Data**
  - Definir os parâmetros name, value e data type como se segue para os dados de treino

  - Clicar **Create** e depois clicar **OK**

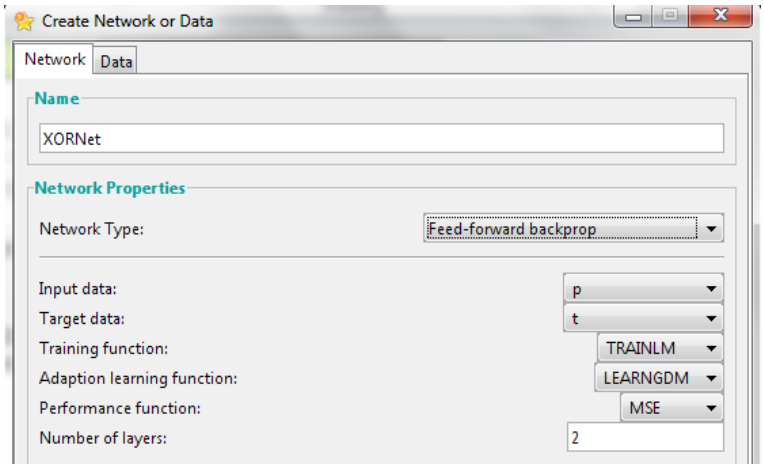
- Ainda no separador **Data**
  - Definir os parâmetros name, value e data type como se segue para os alvos (*targets*) usados no treino



  - Clicar **Create** e depois clicar **OK**

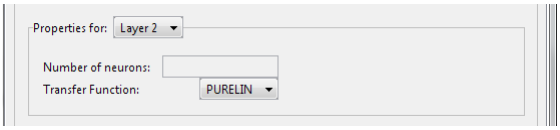
Para o XOR vamos criar uma rede do tipo *feed-forward backpropagation* com duas camadas

- No separador **Network** definir os parâmetros de acordo com a imagem abaixo

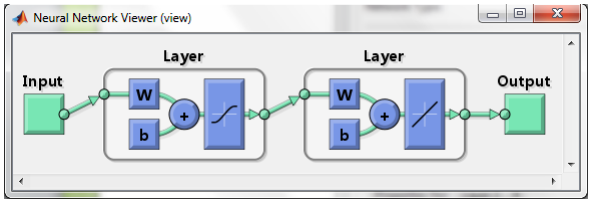


- Para a camada 1

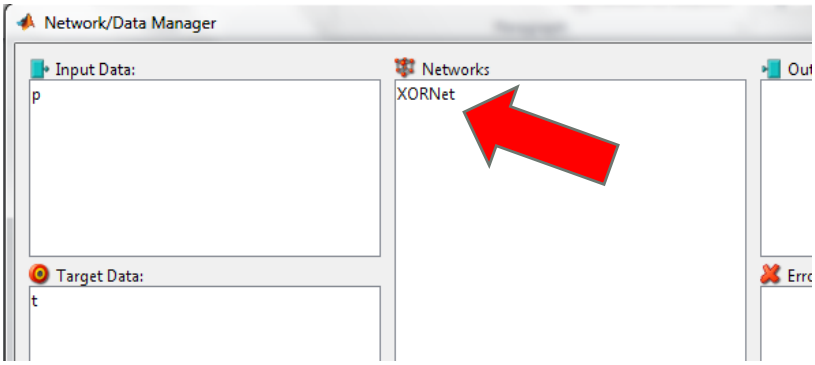
- Para a camada 2



- Clicando em **View** pode-se observar a rede criada

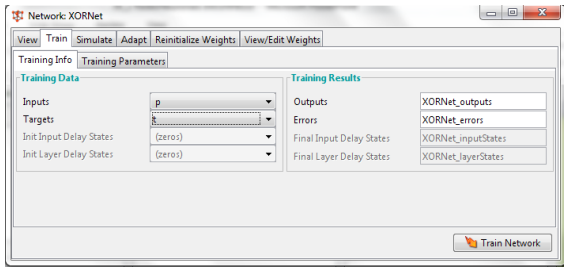


- Para gerar a rede neuronal clicar em **Create**, depois **OK**. No interface principal está agora a rede criada.



Para treinar a rede

- Selecionar a rede (clique simples sobre o nome)
- Clicar em **Open**
- No separador **train**
  - **Definir Inputs: p**
  - **Definir Targets: t**

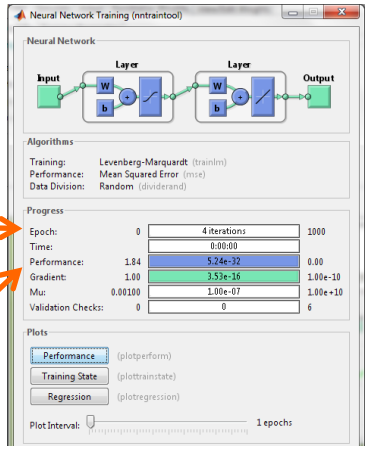


- Na *tab Training Parameters* deixar os valores definidos por defeito

- Inicializar o treino clicando em **Train Network**
- No final do treino deve obter algo semelhante à imagem que se segue

Épocas dispendidas no treino (= 4)

Erro de predição da rede no final do treino (~ 0)



- Pode testar a rede para verificar se o erro é 0
  - Voltar ao interface principal e fazer duplo clique sobre o nome da rede
  - Na **tab Simulate** definir:
    - Inputs: p
    - Outputs: XORNet_outputs_Sim (para não confundir com as saídas produzidas durante o treino anteriormente executado)
    - Clicar **Simulate Network** no canto inferior direito e depois **OK**

- No interface principal fazer duplo clique na variável XORNet_outputs_sim
  - As saídas previstas pela rede apresentam um formato diferente do perceptrão, já que se usou uma função de transferência diferente na camada de saída.
    - Isto exige o uso de uma regra simples para obter saídas binárias como desejado.
  - Por defeito o matlab considera um valor limite de **0.5**. Assim, aplicando a condição **"se saída>0.5 → saída = 1,e caso contrário saída = 0"**
    - Exemplo: Se o vector for [0.98 0.077 0.53 0.01]
    - Então as saídas são [1 0 1 0]
  - Qual o vector de saídas que obteve?
    - Compare-o com o vector target [0 1 1 0] para avaliar o processo de treino
  - A aprendizagem decorreu da forma pretendida?