

Additional Material (Lab3_1)

Create your network

Time Delay Neural Network

IDNN in the lectures

```
net = timedelaynet(inputDelays,hiddenSizes,trainFcn)
```

inputDelays	Row vector of increasing 0 or positive delays (default = 1:2)
hiddenSizes	Row vector of one or more hidden layer sizes (default = 10)
trainFcn	Training function (default = 'trainlm')

Note: for input delays standard dependencies from present input are represented by a 0 delay. Thereby, use inputDelays = 0:4 for an input window of size 5 that spans from the present input to the input 5 steps before.

Layer Recurrent Neural Network (RNNs)

SRN with BPTT in the lectures

```
net = layrecnet(layerDelays,hiddenSizes,trainFcn)
```

layerDelays	Row vector of increasing positive delays (default = 1:2) – <u>use layerDelays = 1</u>
hiddenSizes	Row vector of one or more hidden layer sizes (default = 10)
trainFcn	Training function (default = 'trainlm')

Note: delays corresponding to feedback connections should be >0.

Nonlinear autoregressive with exogenous inputs (NARX)

```
net = narxnet(inputDelays,feedbackDelays,hiddenSizes,trainFcn)
```

inputDelays	Row vector of increasing 0 or positive delays (default = 1:2)
feedbackDelays	Row vector of increasing 0 or positive delays (default = 1:2)
hiddenSizes	Row vector of one or more hidden layer sizes (default = 10)
trainFcn	Training function (default = 'trainlm')

Note: the network should be trained and used in closeloop form.

In older versions of Matlab you should set it using the command, i.e.

```
net = closeloop(net)
```

From 2016a version of Matlab you need to specify the openloop ore closeloop option as a feedback mode when you create the NARX network, according to the following:

```
net = narxnet(inputDelays,feedbackDelays,hiddenSizes, feedbackMode,trainFcn)
```

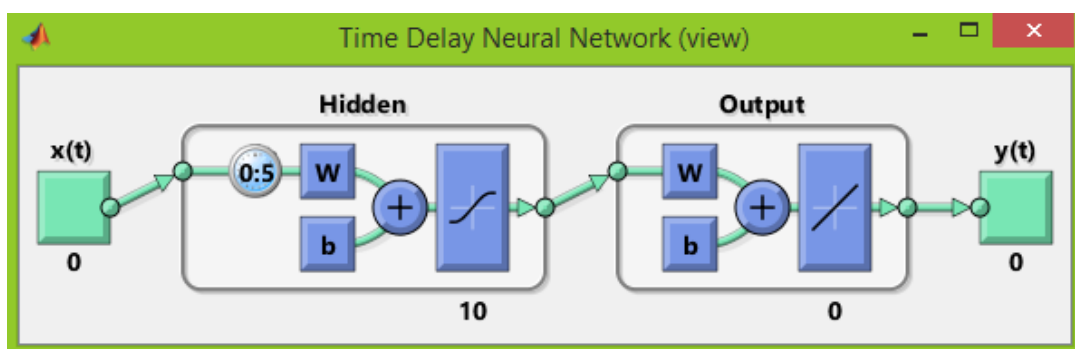
inputDelays	Row vector of increasing 0 or positive delays (default = 1:2)
feedbackDelays	Row vector of increasing 0 or positive delays (default = 1:2)
hiddenSizes	Row vector of one or more hidden layer sizes (default = 10)
feedbackMode	An 'open' or 'closed' feedback mode
trainFcn	Training function (default = 'trainlm')

Customize your network

- Sizes of hidden layers, e.g.
`net.layers{1}.size = 30;`
- Training function (weight update rule), e.g.
`net.trainFcn = 'traingd'; %gradient descent`
`net.trainFcn = 'traingdm'; %gradient descent with momentum`
`net.trainFcn = 'traingdx'; %gradient descent with momentum and adaptive %learning rate`
`net.trainFcn = 'trainrp'; %rprop`
`net.trainFcn = 'trainlm'; %Levenberg-Marquardt optimization`
- Training parameters, e.g.
`net.trainParam.lr = 0.001; %learning rate for gradient descent alg.`
`net.trainParam.mc = 0.5; %momentum constant`
`net.trainParam.epochs = 1000; %maximum number of epochs`
- Regularization, e.g.
`net.performParam.regularization = 0.1; %for weight decay regularization`
- Data splitting (divide function), e.g.
`net.divideFcn = 'dividerand'; %automatic split in tr, vl, ts sets %not recommended!`
`net.divideFcn = 'dividetrain'; %all samples provided with train command %are used for training`

View your network's architecture using the command:

`view(net)`



Prepare the (time series) input for your network

preparets

Prepare input and target time series data for network simulation or training

Syntax

```
[Xs,Xi,Ai,Ts,EWs,shift] = preparets(net,Xnf,Tnf,Tf,EW)
```

Input:

net	Neural network
Xnf	Non-feedback inputs
Tnf	Non-feedback targets
Tf	Feedback targets
EW	Error weights (default = {1})

Output:

Xs	Shifted inputs
Xi	Initial input delay states
Ai	Initial layer delay states
Ts	Shifted targets
EWs	Shifted error weights
shift	The number of timesteps truncated from the front of X and T in order to properly fill Xi and Ai.

e.g.

```
[delayedInput, initialInput, initialStates, delayedTarget] = preparets(net,  
input, target)
```

Train your network

train

Train neural network

[collapse all in page](#)

Syntax

```
[net,tr] = train(net,X,T,Xi,Ai,EW)
```

Input:

net	Network
X	Network inputs
T	Network targets (default = zeros)
Xi	Initial input delay conditions (default = zeros)
Ai	Initial layer delay conditions (default = zeros)
EW	Error weights

Output:

net	Newly trained network
tr	Training record (epoch and perf)

e.g.

```
[net,tr] = train(net, delayedInput, delayedTarget, initialInput);  
%for time delay neural networks!
```

Note: net and tr are two of the structures that you have to include in your assignment output.

Note: Try passing also the “Initial layer delay states” (initialStates) as parameter for the train function.