Neural Network Toolbox

newff

Create a feed-forward backpropagation network

Syntax

```
net = newff
net = newff(PR,[S1 S2...SNl],{TF1 TF2...TFNl},BTF,BLF,PF)
```

Description

`net = newff` creates a new network with a dialog box.

`newff(PR,[S1 S2...SNl],{TF1 TF2...TFNl},BTF,BLF,PF)` takes,

- `PR` - `R` x `2` 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` - Backpropagation network training function, default = '`traingdx`'.
- `BLF` - Backpropagation weight/bias learning function, default = '`learngdm`'.
- `PF` - Performance function, default = '`mse`'.

and returns an `N` layer feed-forward backprop network.

The transfer functions `TFi` can be any differentiable transfer function such as `tansig`, `logsig`, or purelin.

The training function `BTF` can be any of the backprop training functions such as trainlm, `trainbfg`, `trainrp`, `traingd`, etc.

Caution: trainlm is the default training function because it is very fast, but it requires a lot of memory to run. If you get an "out-of-memory" error when training try doing one of these:

1. Slow trainlm training, but reduce memory requirements by setting `net.trainParam.mem_reduc` to 2 or more. (See `help trainlm`.)
2. Use `trainbfg`, which is slower but more memory-efficient than trainlm.
3. Use `trainrp`, which is slower but more memory-efficient than `trainbfg`.

The learning function `BLF` can be either of the backpropagation learning functions such as `learngd` or `learngdm`.

The performance function can be any of the differentiable performance functions such as `mse` or `msereg`.

Examples

Here is a problem consisting of inputs `P` and targets `T` that we would like to solve with a network.

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

Here a two-layer feed-forward network is created. The network's input ranges from [0 to 10]. The first layer has five tansig neurons, the second layer has one purelin neuron. The trainlm network training function is to be used.

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

Here the network is simulated and its output plotted against the targets.

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

Here the network is trained for 50 epochs. Again the network's output is plotted.

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

Algorithm

Feed-forward networks consist of `Nl` layers using the `dotprod` weight function, `netsum` net input function, and the specified transfer functions.

The first layer has weights coming from the input. Each subsequent layer has a weight coming from the previous layer. All layers have biases. The last layer is the network output.

Each layer's weights and biases are initialized with `initnw`.

Adaption is done with `trains`, which updates weights with the specified learning function. Training is done with the specified training function. Performance is measured according to the specified performance function.

See Also

`newcf`, `newelm`, `sim`, `init`, `adapt`, `train`, `trains`

newelm newfftd