Matlab's "net" structure

When we type in the command:

```
net=feedforwardnet(10);
```

Matlab initializes a structure to contain all of the parameters that a neural net needs in order to function. In the second command,

```
net=train(net,x,t);
```

Matlab will also initialize numerical values for the weights and biases. To see all of the things that Matlab uses, you can type the name of the network (in this example, `net`) to see what's inside.

While we are only building feed forward nets, Matlab's structure allows it to build more elaborate connections- We'll focus only on the basics for now.

- To see the information about layer i, type `net.layers{i}`, where i is either 1 (for the hidden layer), or 2 (for the output layer). Matlab doesn't have an input layer.

  The transfer function is given there (the default for the hidden layer is `tansig` and for the output layer is `purelin`. To change the transfer function in the hidden layer, we would type:

  ```
  net.layers{1}.transferFcn='logsig';
  ```

Full Example

Here, we load in a sample dataset from Matlab, then we build a neural network to find the functional relationship. We will then show you how to actually compute the output of the neural network by extracting the necessary information from the `net` structure.

```
[x,t]=simplefit_dataset;   %Load the data.  x is the domain, t is the target.
net=feedforwardnet(10);    %Initialize a feed forward neural net with 10 nodes in t
net=train(net,x,t);        %Train the network (using the default settings)
```

Now that we have the trained network, we can "simulate" the net using other domain points. Here, I've created 100 equally spaced points between the minimum and maximum of the data in the vector `x`.

```
xx=linspace(min(x),max(x),100);
tt=sim(net,xx);
```

We can plot that data, but I wanted to show you how to compute `tt` "manually". First, Matlab needs to process the input data (it automatically, or by default, scales the incoming data).

```
xa=mapminmax('apply',xx,net.input.processSettings{1});
W1=net.IW{1,1}; b1=net.b{1};
W2=net.LW{2,1}; b2=net.b{2};
P1=W1*xa+repmat(b1,1,100);
S1=tansig(P1);
P2=W2*S1+b2;
Y=mapminmax('reverse',P2,net.output.processSettings{1});
plot(x,t,'k-',xx,Y,'r-')
```