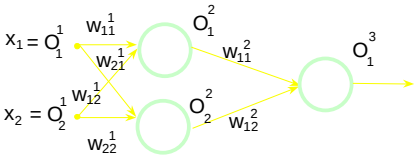
Soft Computing
Kai Goebel/Bill Cheetham
Homework 5: Backpropagation exercise
Due date: Oct. 16th at the beginning of class (no later than 6:10pm) (32 points)

Part 1:
In this exercise, you are asked to implement the backpropagation algorithm. The network structure is a 2x2x1 network (input layer with two nodes, one hidden layer with two nodes, and the output layer with one node) as shown below.



There are no thresholds in this network. The data (patterns) are the two-dimensional exclusive-or (XOR) problem as shown in the table below:

Data	x1	x2	target output d_1^p
Pattern 1	0	0	0
Pattern 2	0	1	1
Pattern 3	1	0	1
Pattern 4	1	1	0

Use the backpropagation process as discussed in class; to recap,
Use the following initial weights:

Weight	Value
w_{11}^1	-0.4
w_{12}^1	-0.4
w_{21}^1	-0.1
w_{22}^1	-0.1
w_{11}^2	-0.6
w_{12}^2	0.3

- Apply the first pattern to the input layer and propagate the signal forward through the network to get the output $O_i^m = f\left(h_i^m\right)$

where

m are the layers

$f = \frac{1}{1+e^{-2\beta h}}$ is the activation function

β is 0.5

$h_i^m = \sum_j w_{ij}^{m-1} O_j^{m-1} + t_i^m$ (note: the output of the input layer is the pattern (x1,x2) itself)

w_{ij}^{m-1} are the weights connecting node j from layer m-1 to node i at layer m

t_i^m are the thresholds connecting to node i at layer m

- Check the output manually to make sure you are on the right track
- Compute the delta for the output layer $\delta_i^m = f'\left(h_i^m\right).\left(d_i^p - O_i^m\right)$

where

f' is the first derivative of activation function *f*. It can be conveniently computed by $f' = 2\beta f \cdot (1-f)$

d_i^p is the desired (target) output for pattern p (note: in this network, i=1 for the output layer since there is only one node)

- Compute the deltas for the preceding layers by propagating the error backwards $\delta_i^{m-1} = f'\left(h_i^{m-1}\right)\sum_i w_{ij}^{m-1}\delta_i^m + t_i^m\delta_i^m$ until a delta has been calculated for every node. (note: in this network, the two nodes of the hidden layer only)
- Update the weights with $w_{ij}^{new} = w_{ij}^{old} + \Delta w_{ij}$

where $\Delta w_{ij}^{m-1} = \eta\delta_i^m O_j^{m-1}$ (where η=0.3 is the learning rate)

- Repeat for the other three patterns for 1 epoch. Record the results (output, weights, deltas).
- Train the network for 10,000 epochs and plot the output over the epochs.

What to hand in:

Calculated values for the output after the first forward stroke with unadjusted weights (5 points). Calculated values for the 3 deltas (10 points) and 6 weights (10 points) after each pattern for the first backpropagation stroke and document values in tabular form as suggested below:

	O_1^3	δ_1^2	δ_2^2	δ_1^3	w_{11}^1	w_{12}^1	w_{21}^1	w_{22}^1	w_{11}^2	w_{12}^2
Pattern 1										
Pattern 2										
Pattern 3										
Pattern 4										

- Plot of the 4 outputs over 10,000 epochs. (7 points)
- Discuss the results

Bonus question: implement performance boosting measures. Discuss and prove the performance increase via plots (up to 15 points)

Note: Code the homework yourself. Do not use any 3rd party tools.

General submission guidelines:
Please submit your homework solutions (excluding source code) including the results in tabular form and plots as a hard copy. Submit source code electronically to the email of the instructors (goebel@cs.rpi.edu, cheetham@cs.rpi.edu). If possible, zip up the source code files and use your last name as the archive name. Identify the homework in the subject line.

Part 2:
Start thinking about your project.