EECS/EEAP 484 Computational Intelligence I, Fall 2011
Problem Set 4: Feedforward Neural Networks with Error Backpropagation
Assigned: 10/3/11
Due: 10/10/11

Neural networks with nonlinear activation functions and hidden layer(s) can behave as universal function approximators. In this problem set, you will set up a neural network with two inputs (plus a bias), a number of "hidden" nodes in an intermediate layer (plus bias) and a single output node. The objective is to fit data points with a smooth function by adjusting the weights of the connections between layers.

The training data for this assignment corresponds to the XOR classification problem.

Your neural-net code should fit the training data at the four corners and should have a smooth surface in the interior.

Starter code is provided. The main file "ps4_fdfwd_net.m" uses 6 functions, also included. (Three of these are optional diagnostic and visualization functions). The key function is "compute_W_derivs.m", which is far from complete. You will need to insert the necessary code for computing back-propagation derivatives.

Calls to the functions numer_est_Wji.m and numer_est_Wkj.m perform an alternative estimation of dEsqd/dw terms by numerical approximation. You should prove that your analytic computation matches the values of these estimates. Include proof in your submission.

Once dE/dW is debugged, you can comment out the debug tests and let your code fit the training data. Evaluate the influence of number of interneurons and choice of epsilon for gradient-descent computations. Include a plot of your function fit. Comment on convergence rates. (note: "for" loops in Matlab can run very slowly—so be patient!).

Repeat the functional fit using Matlab's neural-net toolbox. Edit the M-file nntbox_example.m. Read about how to use newff(), init(net), train() and sim(). Experiment with 'purelin' vs 'tansig'activation functions for the output layer. Comment on the influence of number of interneurons and speed of convergence.