6pts: show all of your work to receive full credit

Homework #4 & 5 (due 10/20) by 1:35pm

1. (6) Build a SOM NN consisting of a 2D lattice drive by a 2D stimulus. This will demonstrate the ordering phase and the convergence phase that are characteristic of the SOM algorithm. The parameters for the NN are:
    a. First define a random distribution of input data of 1,200 points where x varies (randomly) between [-1,1] and y varies (randomly) between [-1,1]. That is, create random (x,y) pairs where x varies between -1:1 and y varies between -1:1.
    b. Initialize the weights in the NN to be (randomly) between [-.1, .1].
    c. The network has a 2D input vector as defined in a. above.
    d. The output is a 24 x 24 2D lattice of output neurons.
    e. Plot the initial random distribution of input points
    f. Plot the initial distribution of random weight values.
    g. Plot the 'Ordering Phase' of the Kohonen SOM (after approximately 150K – 170K iterations). Your results may vary. That is, if you find that the ordering phase is complete at 1K iterations, plot that with an indication of the number of iterations.
    h. Plot the 'Convergence Phase' of the Kohonen SOM (after approximately 750K – 850K iterations). Again, your convergence phase may vary. Typically it is 5x the ordering phase, but results may vary.
    i. You will need to pick a learning rate, annealing rate, a starting neighborhood and an ending neighborhood.
        i. Start with a learning rate around 0.1 and gradually decrease it but keep it >= 0.01. I would suggest an exponential decay factor rather than a linear factor.
        ii. Start with a neighborhood that encompasses all or almost all of the neurons in the network. The neighborhood (of course) is centered on the winning neuron and is shrunk over time.
    j. For the plots of f.), g.), and h.) plot the resulting (x,y) location of each weight pair or neuron and draw a connecting line between each.

**You must submit your results and working code to receive credit. If the code has errors or can't be run to verify the functionality, you will lose a significant majority of credit for the problem.