Math Modeling, Week 12

Start with the code for categorization models at
http://matt.colorado.edu/teaching/mathmodeling/week12code/catlearn.m

1. Run the models on the linearly separable structure in the current code.

a) Make sure you understand the code for creating the stimuli and simulating all three models, and the graph of predictions for each model.

b) Notice that I set the choice temperature differently for the three models to get comparable results. Why did I need to do this; was it for an interesting reason?

c) Imagine we changed the stimulus scaling, e.g. by replacing X with $\tilde{X} = 2X$. How would the predictions for each model change (if at all)? How could we adjust each model's parameters to counteract this change (i.e., so that the model's predictions are identical to those of the original)?

d) Imagine we used an unbalanced training set, e.g. setting `ntrainA = 200`. What should happen for each model? Try it and see. How does the exemplar model's sensitivity to unbalanced training sets depend on its similarity parameter (`alpha`), and why?

e) Notice I set the network's learning rate to .01, even though fits to human data usually yield results of .1 to .3. Try a larger learning rate and notice how much the model predictions change from one simulation to the next (even with the same stimuli). Why is this, and what are the psychological implications?

2. Change the categories to be anisotropic, i.e. with more variance along one dimension than another. For example: `Xtrain(:,1)=Xtrain(:,1)/2;Xtrain(:,2)=Xtrain(:,2)*2;`

a) Compare the behavior of the network and prototype models, and explain the difference in terms of the analysis we did in class.

b) The exemplar model should exhibit an S-shaped category boundary (seen most clearly from the test stimuli). Explain.

c) Try to think of a way to change the exemplar model so that its behavior is similar to the network's, i.e. with response depending more on the low-variance dimension even for peripheral stimuli.

3. Create a category structure in which the exemplar model outperforms the other two. Then augment the stimulus representation with a third dimension that's some nonlinear function of the other two (i.e., $X_3 = f(X_1, X_2)$) that will enable the network and prototype models to solve the task.

For example, you could use the XOR structure, with `XtrainA=[1 1; -1 -1]` and `XtrainB=[-1 1;1 -1]`, and then define `X(:,3) = X(:,1).*X(:,2)`. We've already discussed how Rescorla-Wagner can solve the XOR task with the addition of this third stimulus dimension. Try to think of some other nonlinear category structure for which you can do something similar. If you need a suggestion, try this:

```
ntrainA = 200; %number of A training stimuli
ntrainB = 200; %number of B training stimuli

%category A: circle of radius 5
radA = sqrt(rand(ntrainA,1)*5); %random radii (scaled to be uniform wrt area)
angleA = rand(ntrainA,1)*2*pi; %random angle
XtrainA = radA.*[sin(angleA) cos(angleA)]; %random points in circle

%category B: annulus with inner radius 5 and outer radius 10
radB = sqrt(rand(ntrainB,1)*5+5); %random radii (scaled to be uniform wrt area)
angleB = rand(ntrainB,1)*2*pi; %random angle
XtrainB = radB.*[sin(angleB) cos(angleB)]; %random points in annulus

Xtrain = [XtrainA;XtrainB]; %combine training stimuli
Xall = Xtrain; %no need for test stimuli
```