Math Modeling, Week 9

1. Explore the Hopfield network code.

(a) Train the network on the training patterns. Start it in a noisy version of a training pattern (not 3 or 9) and run it until convergence. Start it again, this time using `clean<0`, and run until convergence. What happens, and what does this tell you about attractors in this kind of network?

Starting with a majority of nodes mismatching a given pattern generally results in convergence to the opposite of that pattern ($1 \leftrightarrow -1$). This reflects a general symmetry or sign-invariance of the network dynamics. Formally, if we define $\tilde{a} = -a$ and derive the update rule for $\tilde{a}$, it will be the same as that for a.

(b) Start the network in pattern 3 or 9 and run to convergence. What's happening? Do you think something like this could happen with the patterns in `setuplines.m`?

Interference between patterns 3 and 9 causes them to be unstable, and instead there is a stable pattern that is in a sense between these two trained patterns. Notice though that this effect depends on contributions from other trained patterns; if the network is trained only on patterns 3 and 9 then both are stable.

This interference happens because of the large similarity between patterns 3 and 9. As seen in question 3, interference between two patterns depends on $a_i^1 a_i^2 a_j^1 a_j^2$, which can be seen as a measure of pattern overlap. For the patterns in `setuplines.m`, pairwise overlap is always 50%. Therefore the interference terms balance out to zero (or very close; the lack of self-connections breaks the symmetry slightly), and the patterns don't interfere with each other.

2. Consider a Hopfield network with n units, trained by Hebbian learning on a single pattern a^1. That is, $a_i^1 \in \{-1, 1\}$ for all i, and the weight between any two distinct nodes is $w_{ij} = \frac{1}{n} a_i^1 a_j^1$, with $w_{ii} = 0$. Prove that a^1 is a stable state (i.e., an attractor) of the network.

More specifically: Imagine we put the network in state a^1, by setting the activation $a_j = a_j^1$ for all j, and we pick any node i and update it according to $a_i \leftarrow \text{sign}\left(\sum_j a_j w_{ij}\right)$. Prove that a_i doesn't change, i.e. that $\text{sign}\left(\sum_j a_j w_{ij}\right) = a_i^1$. Hint: substitute the definition of w into the update equation, and use the fact that $a_j a_j = 1$ for all j.

The input to node i can be written as $\sum_j w_{ij} a_j = \sum_{j \neq i} \frac{1}{n} a_i^1 a_j^1 a_j^1 = \frac{1}{n} \sum_{j \neq i} a_i^1 = \frac{n-1}{n} a_i^1$. These steps use the facts that (a) $w_{ij} = a_i^1 a_j^1$ for $i \neq j$ and $w_{ii} = 0$, (b) the network is in state a^1 meaning $a_j = a_j^1$, and (c) $a_j^1 a_j^1 = 1$ regardless of whether $a_j^1 = 1$ or $a_j^1 = -1$. Because $\text{sign}\left(\frac{n-1}{n} a_i^1\right) = a_i^1$, the updating doesn't change the state of the network, and therefore a^1 is stable.

3. Now imagine training the network on two patterns, a^1 and a^2, so that $w_{ij} = \frac{1}{n}a_i^1 a_j^1 + \frac{1}{n}a_i^2 a_j^2$ for all $i \neq j$ and $w_{ii} = 0$.

(a) Assume the network is in state a^1. Write an expression for the total input to any node i, in terms of a^1 and a^2 (i.e., eliminating w). Simplify the expression as much as possible, to separate the interference between a^1 and a^2 from the contribution of a^1 alone (the latter should match what you derived in question 2).

When the network is in state a^1, the total input to any node i is equal to

$$\begin{aligned}\sum_j w_{ij} a_j &= \sum_{j \neq i} \left(\frac{1}{n}a_i^1 a_j^1 + \frac{1}{n}a_i^2 a_j^2\right) a_j^1 \\ &= \frac{1}{n}\sum_{j \neq i} \left(a_i^1 + a_i^2 a_j^2 a_j^1\right) \\ &= \frac{n-1}{n}a_i^1 + \frac{1}{n}\sum_{j \neq i} a_i^2 a_j^2 a_j^1.\end{aligned}$$

The first term here is the contribution from a^1 alone (same as in the previous question), and the second term represents interference between the two patterns.

(b) Comparing the two terms in the previous answer (interference and contribution from a^1 alone), try to figure out what would need to happen for the training patterns not to be stable. That is, how would a^1 and a^2 need to be related in order for the interference terms to cause a problem?

Understanding the interference might be easier if we rewrite the total input as

$$\left(\frac{n-1}{n} + \frac{1}{n}\sum_{j \neq i} a_i^1 a_i^2 a_j^1 a_j^2\right) a_i^1.$$

Pattern a^1 will be stable iff the expression in the parentheses is positive for all i. The $\frac{n-1}{n}$ term is clearly positive, so the question depends on the $a_i^1 a_i^2 a_j^1 a_j^2$ terms.

You can think about $a_i^1 a_i^2 a_j^1 a_j^2$ as a second-order interaction between the patterns, a "sameness of sameness" relation. First, notice that $a_i^k a_j^k$ equals 1 if nodes i and j match in pattern k, and -1 if they mismatch. Therefore, $a_i^1 a_i^2 a_j^1 a_j^2$ will equal 1 if nodes i and j match in both patterns, or if they mismatch in both patterns. Likewise, $a_i^1 a_i^2 a_j^1 a_j^2$ will equal -1 if nodes i and j match in one pattern and mismatch in the other. Thus $a_i^1 a_i^2 a_j^1 a_j^2$ encodes whether the two patterns agree on whether nodes i and j should match.

The worst case is that all of the interference terms are negative. This happens when, for every j, nodes i and j match in one pattern and mismatch in the other. Because there are $n - 1$ interference terms, the total input to node i will be $\left(\frac{n-1}{n} + \frac{1}{n}\sum_{j \neq i}(-1)\right)a_i^1 = 0$. The update behavior of a_i can then be taken to be random.

There are two ways for the interference terms all to be negative, depending on whether the two patterns match or mismatch on node i. First assume they match, $a_i^1 = a_i^2$. Then we must have $a_j^1 \neq a_j^2$ for all j (other than i)—that is, the two patterns disagree everywhere except at node i. Second, assume that $a_i^1 \neq a_i^2$. Then we must have $a_j^1 = a_j^2$ for all j (other than i)—that is, the two patterns agree everywhere except at node i. In either case, notice that all weights involving i will equal zero: $\forall j, w_{ij} = \frac{1}{n}a_i^1 a_j^1 + \frac{1}{n}a_i^2 a_j^2 = \frac{1}{n} - \frac{1}{n}$ or $-\frac{1}{n} + \frac{1}{n}$.

In summary, instability can occur only if the two patterns agree on exactly one node or disagree on exactly one node, and in that case the instability is specific to the unique node. In fact, we have only semi-stability, because all inputs to that node equal zero. Therefore the network will randomly oscillate between the two training patterns (or their antipatterns).