

CNS 187 - Neural Computation Problem Sheet 4

Handed out: 2 Nov 2001
Due: 9 Nov 2001, 5:00pm

Each problem is weighted equally, and within each problem the subparts are weighted equally, totally 100 points. Please use separate pages for each problem, stapled separately, with your name on each page. You may collaborate on working out ideas, but if so, name your collaborators, write up your own solutions, and write your own code.

4.1 Analog match: spike timing and linear thresholds

Consider a recognition system which can measure as its input K positive analog numbers each of which has some noise in it. Based on these values the system wants to “recognize” one of several stored patterns. Perhaps the numbers are the outputs of various chemical receptors and the patterns are different odors; perhaps they are red, green, and blue intensities and the patterns are different colors.

If we order the numbers then we can view the input as a vector. This K -dimensional input vector is believed to be a noisy *and scaled* version of one of N *prototype* vectors each of which is also K -dimensional. In other words, we postulate that the *ratios* of the various components are preserved whenever a pattern from a particular class is encountered, but their absolute scale is unimportant. For example, a faint whiff of a rose and the strong scent of a rose are both instances of the odor rose; a teal car at twilight and at high noon looks the same color, more or less. Additionally, our idea of “noise” includes the possibility of completely obliterating the component in question, for example by setting it to a uniformly random value, or to zero.

The goal then of the recognition system is to discover from which class the input came; in other words to discover the prototype vector that “best” matches the input in some sense. This problem is generically known as *analog match*.

We will use the spike timing matching method described in Hopfield’s paper “Pattern-recognition computation using action-potential timing for stimulus representation”, *Nature* **376** (6535) pp. 33-36 (1995).

Each component of the input vector \vec{I} first passes through a spiking encoder before heading on to the recognition circuit. We will assume that the spikes encode the logarithm of the strength of the input component as the spike time. So our first step is to take the logarithm of each component – the result represents a spike time. Call the spike time of each component t_i , so that $t_i = \log(I_i)$.

Next the components are sent on to the various recognizers, which are coincidence detectors. Each recognizer has a set of delays in the lines leading to it that are set by the prototype

pattern which corresponds to that unit. (That is, if we have memorized n possible patterns, we have n coincidence detector units, each with a set of delay lines leading to it. If any one of coincidence detectors fires, it is saying “I think this input corresponded to my prototype vector”.) Consider one of the recognizers corresponding to prototype vector \vec{p} . The delay on the i th line should be $\log(p_i)$, so subtract this from the spike time, to get a new spike time for each component: $\log(I_i) - \log(p_i)$.¹

We will model the coincidence detection in the following way: as each spike arrives at the recognition unit, it raises the membrane potential in the recognition unit by a unit voltage, for a time w . After time w has elapsed, that spike’s contribution to the membrane voltage is set back to zero and is forgotten (note that the contribution from *other* components is not affected by this). That is, the excitatory postsynaptic potential (EPSP) is modeled as a square boxcar shape² of time width w . Different spikes simply contribute additively to the postsynaptic membrane voltage. If this voltage ever passes above a threshold θ , the cell fires, otherwise not. For example, if there are N components to each vector and the input vector was strictly proportional to the prototype vector, then all the spikes would arrive at exactly the same time and the membrane potential of the coincidence detector (the recognizer unit) would suddenly rise by N units, stay there for w seconds, and then go back to resting potential. We use this coarse model so that you can understand the essential qualities of the system without getting lost in technical details.

The basic questions we will explore are: What parts of the input space does this unit fire for (classify as being \vec{p})? How does this change as we change θ , w or the synaptic strengths? How does this compare to thresholded dot product?

1. Suppose we are dealing with vectors that have two components only. Set the threshold at 1.5 units. Thus, only if the two spikes are within time w of each other will the cell spike. What does the receptive field of the cell look like? That is, for what region of input space does the recognition cell produce a spike? Assume that p_1, p_2 and w are known, derive an analytic expression involving these, and sketch the receptive field in input space (i.e. the set of inputs that cause the cell to fire). One of the axes of your graph should be I_1 , the first input, and the other should be I_2 , the second input. Does this sketch look like the recognition is scale invariant, in the sense that if \vec{I} is classified as being \vec{p} then so is $\alpha\vec{I}$ (where α is a positive scalar?). How does the value of w affect the sketch?
2. Sketch the receptive field in three dimensions, for three inputs I_1, I_2 , and I_3 , for a threshold of 2.5.
3. In the section above we set the threshold so that the cell only fired if *all* of the D input components’ spikes were within time w of each other. What if we lower the threshold? Suppose we lowered it so that it is enough to cause the cell to fire if $D - 1$ components are within w of each other³

¹Don’t worry about spike times possibly going negative– just add some appropriate constant to make them all positive. This will not affect the recognition.

²more generally, each input channel could have a different EPSP height also, but we won’t consider that here.

³For example, the various components might represent various features of Grandma’s. But maybe Grandma

Sketch the receptive field in three dimensions, for three inputs I_1, I_2 , and I_3 , for a threshold of 1.5, i.e. we allow any one of the three components to be wrong? Think of it first without worrying about scale invariance, and then add that later. We do not ask for an analytic expression, just a clear statement or picture. The problem is not hard but it is not completely trivial either.

4. The proportion of space classified as \vec{p} gives us an idea of how *selective* a unit is. Consider the case where \vec{p} lies in the center of a unit cube, and I_i vary between 0 and 1, and $w = 0.2$. We would like to recognize inputs where at least half of the D input channels are approximately equal; set the threshold appropriately as a function of the number of dimensions, D .

How selective is this cell in high dimensions, specifically, for $D=2,3,4,5,10$? If you wish, you may use numerical monte carlo simulations to estimate this probability that a uniformly randomly chosen input value results in a spike; estimate error bars.

5. Now consider an alternative classification scheme based on normalization of the inputs (to obtain scale invariance) followed by a linear threshold unit classification, as follows. Normalize each input vector and the prototype vector to have unit length. Then compute their dot product with the prototype vector, and “fire” if the result is above a threshold θ . As before, we would like to guarantee that the cell will recognize correct inputs, $\vec{I} = \vec{p} = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \dots)$, as well as those in which fewer than half of the components have been set to zero. In D dimensions, what must the threshold be for all of the vectors generated in this way to be classified as \vec{p} ? What proportion of the space is classified as \vec{p} under this scheme and with this threshold, as a measure of how “selective” this scheme is, compared to the spike timing method? Compute this fraction for $D = 2, 3, 4, 5, 10$. You may do it by monte carlo simulation if you are so inclined, but include error bars in that case. What do the receptive fields of this scheme look like? Sketch them for $D = 2$ and $D = 3$.

What we hope you will take away from this exercise is a feeling for how the receptive fields of the two schemes differ, and what some of the implications might be.

4.2 Lyapunov Energy Functions

We will consider a physical system which is a modification of the damped mass spring oscillator (see figure below).

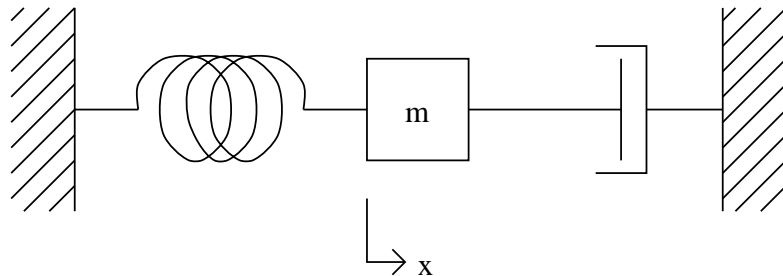
In the original linear system, we examine a mass m attached to a linear spring (recall Hooke’s law from basic physics) which experiences frictional damping proportional to velocity. The damping coefficient is denoted $\gamma > 0$, and the spring constant $\kappa > 0$. The dynamics of such a system, derived from Newton’s laws, give rise to a second order linear differential equation

is wearing an eyepatch on her right eye today– yet we’d still like to recognize her. Or maybe the eyepatch is over her left eye. Or maybe the parrot on her shoulder has leaned over to whisper “pieces of eight” in her ear and is occluding her very idiosyncratic golden hoop earring. In any case, we’d like to say “it doesn’t matter if a few pieces of Grandma don’t look right, it’s still Grandma.” This is what lowering the threshold in the manner described above represents.

whose behavior is easy to understand and should be quite familiar. In this problem, we introduce a nonlinear variation of the damped oscillator. To analyze it, we will use a key concept from nonlinear stability analysis: the Lyapunov, or energy function.

Consider the one-dimensional oscillator with mass m , damping coefficient γ , and a *nonlinear* spring with a characteristic $F = \kappa x(1 + \delta x^2)$; the extra spring constant is denoted $\delta > 0$. We have essentially added a cubic nonlinearity to the linear Hooke's Law.

1. Write down the dynamic equations in terms of the position x .
2. Recall your basic classical physics and write down a function describing the total energy E of the system as a function of x and its derivatives.
3. Given the dynamic equation you wrote down, find the steady-state (no change in time) of the system. Finally, noticing that the energy function you have constructed corresponds to the idea of a Lyapunov function introduced in the lectures, show that the steady state is *stable*. Specifically, show that E is bounded below, with a unique minimum at the steady-state point, and that $\frac{dE}{dt} \leq 0$. Is $\frac{dE}{dt} = 0$ iff the steady-state is achieved, i.e., is E always decreasing prior to the system reaching the steady-state, and if not, why does the system not stop there?



4.3 Lyapunov analysis of networks

Lyapunov analysis shows that recurrent networks with symmetric weights are guaranteed to be stable, no matter what the weights are. We also know that strictly feed-forward networks are always stable, and that there are many networks with asymmetric weights that oscillate or behave chaotically. Let's push the Lyapunov analysis to explore the middle ground.

First, let's consider a network where each neuron has its own time constant and its own activation function, with its own gain. I.e.,

$$\tau_i \dot{V}_i = -V_i + \sum_{j=1}^N w_{ij} S_j + I_i \quad (1)$$

$$S_i = g_i(V_i) = \alpha_i g(\beta_i V_i) \quad (2)$$

where $\tau_i > 0$ and the activation function $g(\cdot)$ is any⁴ bounded monotonic increasing differentiable function (so $g'(\cdot) > 0$).

Following the original analysis, we define a function

$$\mathcal{L} = -\frac{1}{2} \sum_{i,j} w_{ij} S_i S_j + \sum_i G_i(S_i) - \sum_i S_i I_i \quad (3)$$

where $G_k(S_k) = \int_0^{S_k} g_k^{-1}(z) dz$ and $g_k^{-1}(S_k) = V_k$ is the inverse activation function for unit k .

1. Consider a symmetric network, with $w_{ij} = w_{ji}$. Give sufficient (but not necessarily necessary) conditions on α_i and β_i under which \mathcal{L} is a Lyapunov function for this network. Show that \mathcal{L} satisfies the requirements for a Lyapunov function that we used in class: \mathcal{L} is bounded below and $\dot{\mathcal{L}} \leq 0$ with equality exclusively at steady-state solutions.
2. Symmetric networks with individually set activation functions (as described above) are equivalent to a subclass of networks with asymmetric weights, including feedforward networks as a limit. You will show this. Consider feedforward networks defined by

$$\tau \dot{u}_i = -u_i + \sum_j C_{ij} g(u_j) + i_i \quad (4)$$

where $g(\cdot)$ is the same function as above, and neurons only receive synaptic input from earlier neurons, i.e., $C_{ij} = 0$ for $i \leq j$.

For every $r > 0$, define an asymmetric recurrent network with

$$\tau \dot{u}_i = -u_i + \sum_j C_{ij}^r g(u_j) + i_i \quad (5)$$

where $C_{ij}^r = C_{ij}$ for $i > j$ and $C_{ij}^r = C_{ji} r^{i-j}$ for $i \leq j$. Describe in words how this asymmetric network's behavior is related to the feedforward network, as r goes from 1 to ∞ .

3. As a function of r , find $\tau_i, \alpha_i, \beta_i, I_i$, and $w_{ij} = w_{ji}$ such that Equation 1 is exactly equivalent to Equation 5 for $u_i = \beta_i V_i$.
4. We don't want to leave you with the impression that all networks are stable. Consider a three-node network, "ring oscillator," described by

$$\tau \dot{V}_i = -V_i + \sum_j w_{ij} g_\beta(V_j) \quad (6)$$

with $w_{13} = w_{21} = w_{32} = -1$ and other $w_{ij} = 0$, and $g_\beta(x) = g(\beta x) = \tanh(\beta x)$. First show that the network has a unique steady state ($\dot{V}_1 = \dot{V}_2 = \dot{V}_3 = 0$) at $V_1 = V_2 = V_3 = 0$. Then show that the network oscillates (i.e. that the steady state is unstable) for $\beta > \beta_0$. What is β_0 ?

⁴For concreteness you may take $g(x) = \tanh(x)$ if you wish.