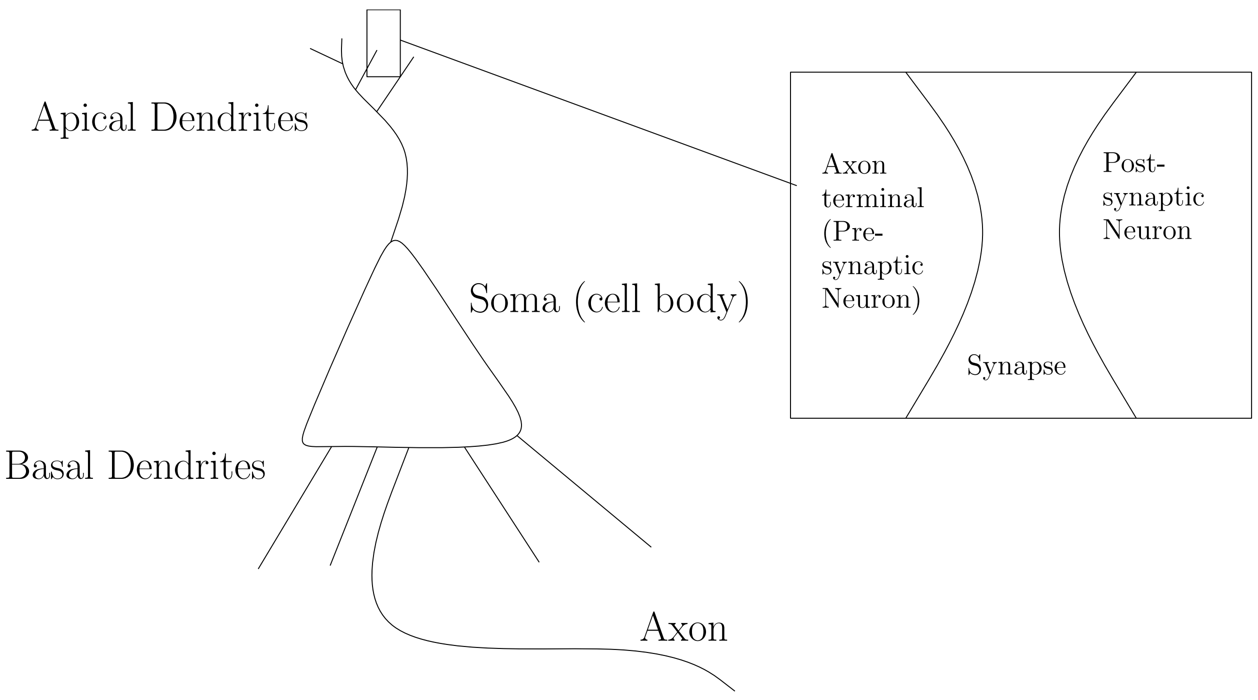
1. August 27, 2015

1.1. **Basic Neuro Facts.** Pyramidal neurons have structures like this cartoon:



Neurons fire action potentials when they receive enough “coincident” inputs from synapses onto their dendrites. The Hodgkin-Huxley model gives a threshold intensity of inputs in a time window which will trigger it to fire. Typically, many excitatory inputs are needed near-simultaneously in order for a neuron to fire. θ denotes the threshold value (usually measured as voltage, in units of mV).

Neurons may influence each other with different weights. For instance one neuron may be connected to another at multiple synapses. Size of the synapse/dendrite may also influence signal intensity - this is called synaptic strength. [Jeff Lichtmann’s laboratory images neurons and synapses to count connections.]

Synaptic plasticity is the ability of a synapse to change its strength. Hebb’s rule states that neurons that fire frequently together have the synapse strengthened.

An action potential is “all-or-none”; this is usually true but photoreceptor neurons for example have a *graded response*. The firing rate of a neuron describes the number of action potentials per unit time (unit: Hz). Many computational network models reflect the firing rate, NOT the specific action potentials. LIF “Leaky-integrate-and-fire”.

Neurons are either excitatory or inhibitory in their effect on other neurons. The inputs can be varied, but outputs are all positive or all negative. Some inhibitory signals go directly into the soma - these are called shunting inhibition.

1.2. **McCulloch-Pitts Model of the Neuron.** The first abstract mathematical model of a neuron was the *McCulloch-Pitts model*. It was introduced in 1943, when Boolean logic was “in the air”. The idea is that neurons can perform Boolean logic operations AND, NOT, OR.

The neuron is then pictured as a node with n inputs (inbound directed edges) all with edge weight 0 or 1, and one output of weight 0 or 1. Call the inputs $x_1, \dots, x_n$ and the output y.

Rule for transforming inputs to the output:

$$y = H\left(\sum_{i=1}^{n} x_i - \theta\right) \qquad H(z) = \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases}$$

Recall that θ denotes our threshold.

Definition 1.1. A *Boolean function* is a function $f : \{0,1\}^n \to \{0,1\}$.

The choice of θ selects a particular f_θ performed by the M-P neuron.

Example 1.2. Consider the case of two inputs, with $\theta = 2$. This executes the logical "AND". If $\theta = 1$, then the M-P neuron executes logical "OR". "XOR" is not possible with a single M-P neuron without inhibition (since $2 > 1$).

Question 1.3. Which Boolean functions can be implemented by a single M-P neuron?

Definition 1.4. We say that a boolean function is *monotone increasing* in the i-th variable (input x_i) if for any set of values $x_1, \dots, \hat{x_i}, \dots, x_n \in \{0,1\}$ we have $f(x_1, \dots, 0, \dots, x_n) \leq f(x_1, \dots, 1, \dots, x_n)$.

Similarly f is *monotone decreasing* in the i-th variable (input x_i) if for any set of values $x_1, \dots, \hat{x_i}, \dots, x_n \in \{0,1\}$ we have $f(x_1, \dots, 0, \dots, x_n) \geq f(x_1, \dots, 1, \dots, x_n)$.

f is *unate* if it is monotone increasing or decreasing in that variable.

XOR is not unate in either variable; therefore, it cannot be realized by an M-P neuron.

Exercise 1.5.
(1) Show that any B.F. $f : \{0,1\}^n \to \{0,1\}$, represented by an M-P neuron is monotone increasing in each variable.
(2) Find an example of a B.F. that is monotone increasing in each variable but cannot be implemented by a single M-P neuron. (i.e. show that the converse is false.)

Linear-threshold neurons or "perceptrons" will be discussed next week.

2. September 1, 2015

2.1. Linear-threshold neurons.

Let $x_1, \dots, x_n$ denote the input layer which may take values in $\{0,1\}$. Let $w_1, \dots, w_n \in \mathbb{R}$ be weights on the edges from $x_i \to y$. These can be positive (excitatory) or negative (inhibitory). These functions then induce a value of 0 or 1 for y by the function

$$y = H\left(\sum_{i=1}^{n} w_i x_i - \theta\right).$$

where H is the Heaviside function, and θ is a threshold.

Like $M - P$ neurons, Linear-threshold neurons represent Boolean functions: $f : \{0,1\}^n \to \{0,1\}$. Recall that a Boolean function is unate if it is monotone increasing or decreasing on each variable x_i.

Lemma 2.1. *Every Boolean function that can be represented by a linear-threshold neuron is unate.*

Proof. Pick x_i and show it's either monotone increasing or decreasing.

Case 1. $w_i \geq 0$. Heaviside functions are monotone increasing, so increasing the input will increase the output.

Case 2. $w_1 < 0$. Decreasing the input decreases the output.

□

Question 2.2. Is every unate Boolean function representable by an LT neuron?

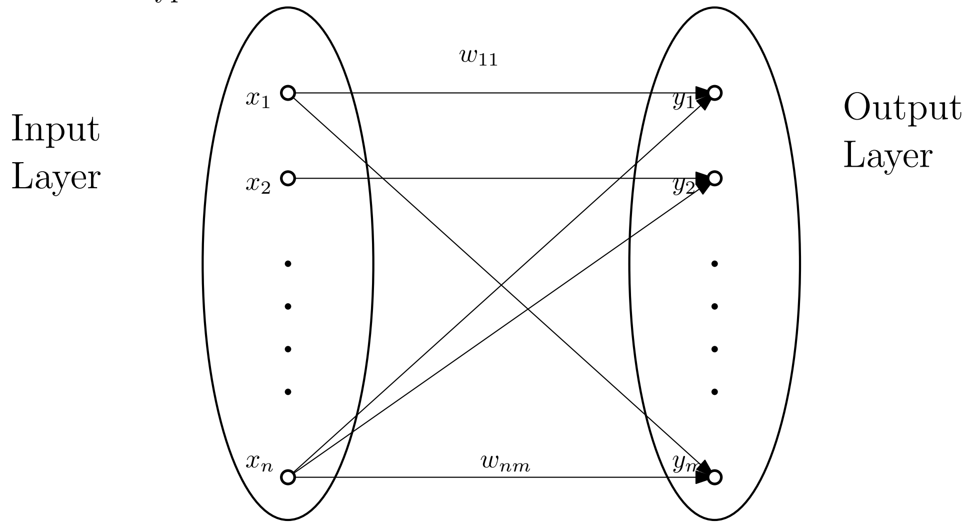
Answer: NO.

Example 2.3. Let $f(x_1,\dots,x_4) = (x_1 \wedge x_2) \vee (x_3 \wedge x_4)$. These are binary Boolean operators "OR" and "AND". This function is monotone increasing on all neurons.

However, we claim that it is not realizable as a linear threshold neuron.

Proof: Assume $f(x_1,\dots,x_4) = H\left(\sum_{i=1}^4 w_i x_i - \theta\right)$. We know $w_1+w_2-\theta \geq 0$ and $w_3+w_4-\theta \geq 0$. If we take $\max(w_1,w_2)+\max(w_3,w_4)$, this is guaranteed to exceed θ.

2.2. **Single-layer Feed-forward Network of Linear-threshold Neurons.** Below is the schematic for networks of this type:



Each output is a linear-threshold neuron.

$$y_i = H\left(\sum_{j=1}^n w_{ij}x_j - \theta_i\right)$$

We can summarize w as a row vector:

$$w^{(i)} = (w_{ii},\dots,w_{in}) \in \mathbb{R}^n. \qquad W = [w_{ij}].$$

$$\Rightarrow y_i = H\left(w^{(i)} \cdot \mathbf{x} - \theta_i\right)$$

Remark 2.4.
- w_{ij} is the weight from $j \to i$.
- θ_i is the threshold of the i-th LT neuron; in particular, $\theta_i \in \mathbb{R}$.
- H acting entry wise on column vectors, we can write $y = H(Wx - \theta)$. Note that W is an $m \times n$ matrix.

The set of all LT neurons in a single-layer feed-forward network corresponds to a hyperplane arrangement in $\mathbb{R}^n$ (m hyperplanes).

Example 2.5. $n = 2$ and $m = 3$. So we need a 3×2 matrix W and a 3×1 matrix Θ.

$$W = \begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{pmatrix}. \qquad \Theta = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{pmatrix}.$$

Let us fix values for these:

$$W = \begin{pmatrix} 1 & 2 \\ 2 & -2 \\ -1 & 0 \end{pmatrix}. \qquad \Theta = \begin{pmatrix} 2 \\ 1 \\ -1/2 \end{pmatrix}.$$

The hyperplane arrangement will be in $\mathbb{R}^2$ with three half-spaces. The equations are as below:

$$\begin{aligned} y_1 &= H(x_1 + 2x_2 - 2) \\ y_2 &= H(2x_1 - 2x_2 - 1) \\ y_3 &= H(x_1 - 1/2) \end{aligned}$$

The values of these outputs depend on the inputs of x_1, x_2. In Figure 1, we draw the lines as well as each point corresponding to all possible $0 - 1$ values for x_1, x_2.

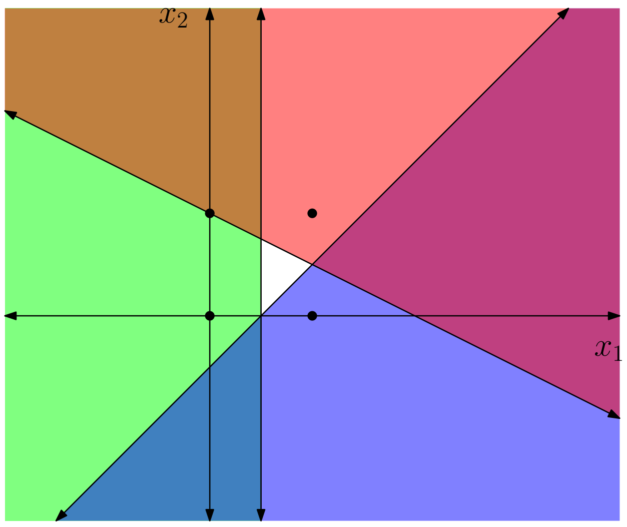


Figure 1. Regions corresponding to output layer responses.

If we take the outputs corresponding to these inputs, then we get four responses: $001, 010, 101, 100$ (Note that values on a line are considered to trigger the neuron). In some settings, we also allow nonnegative inputs for x; then every region in the positive quadrant counts as an output of the system. Note that the point $(1/2, 0)$ actually contributes a new response.

3. September 3, 2015

3.1. **Comment on Homework #1.** The assignment was to find an example of a monotone increasing Boolean function that cannot be realized by a single M-P neuron. Some people suggested the following function:

$$\begin{array}{ccc} f : \{0,1\}^2 & \longrightarrow & \{0,1\} \\ (0,0) & \mapsto & 0 \\ (1,0) & \mapsto & 0 \\ (0,1) & \mapsto & 1 \\ (1,1) & \mapsto & 1 \end{array}$$

In some sense, this is correct, since the Heaviside function $f(x) = H(x_1 + x_2 - \theta)$ does not fit this form. But, the edge weights in the feed-forward network are actually allowed to be 0 as well; therefore this function is realizable.

Another suggested example is:

$$f(x_1, x_2, x_3) = (x_1 \wedge x_2) \vee (x_2 \wedge x_3)$$

Indeed this is not realizable as an MP neuron, but it is realizable as an LT neuron. Not MP: since the AND forces all edge weights to 1 and the threshold to 2 which would imply $f(1,0,1) = 1$. However, it is LT since you can give edge 2 weight 2 and make the threshold 3.

3.2. **The code of a single-layer feed-forward network of LT neurons.** A code is a subset $\mathcal{C} \subset \{0,1\}^n$. n is called the *length* of the code. An element of $\mathcal{C}$ is called a *codeword*.

Example 3.1 ($n = 3$)**.**

$$\mathcal{C} = \{000, 100, 110, 111\}$$

Remark 3.2. A popular field of study is error-correcting codes. The idea is that $\mathcal{C} \subsetneq \{0,1\}^n$, not all $0-1$ strings are legal codewords. For example,

$$\begin{array}{ccc} A & \to & 0010 \\ B & \to & 0101 \\ C & \to & 1000 \end{array}$$

Sending these strings through a noisy channel may change a small number of bits, but since there is some minimal Hamming distance between words, we can correct the error. The Hamming distance between codewords $c_1, c_2 \in \{0,1\}^n$ is defined as

$$d_H(c_1, c_2) = |\{i \in [n] | (c_1)_i \neq (c_2)_i\}|.$$

The art/science of creating good codes is to pick a set of codewords with higher mutual Hamming distance. Claude Shannon working around 1948 is one of the founders of information theory and coding theory.

The neural code, i.e. the way in which patterns of neural activity represent information (stimuli). A stimulus motivates some collection of neurons to fire, giving us a codeword.

Question 3.3. What kinds of codes $\mathcal{C} \subseteq \{0,1\}^n$ can be realized by a single-layer network of LT neurons?

In the last class, in Figure 1, we saw a half-plane arrangement giving 7 codewords - all codewords of length 3 except 111.

But why should the possible codes be restricted at all?

Example 3.4. $\mathcal{C} = \{0,1\}^3 \setminus \{001, 111\}$. The code has length 6.

Note that codewords for an LT network correspond to regions defined by intersections of half-spaces. Regions corresponding must all be codewords are all convex.

Definition 3.5. A set $X \subset \mathbb{R}^d$ is convex if the line segment $\ell_{p_1p_2} = \{tp_1 + (1-t)p_2 \mid t \in [0,1]\}$ between any two points $p_1, p_2 \in X$ is also fully contained in X.

In an LT network, neuron i has a corresponding half-space H_i^+ given by the side of H_i where i fires.

The fact that codewords "come from" overlapping half-spaces (convex sets) creates constraints on the possible codes. With this architecture, not all codes are possible.

Claim: $\mathcal{C}$ is not realizable with a single-layer LT network.

Proof. Let $U_i = H_i^+ \cap \mathbb{R}^m_{\geq 0}$. U_1, U_2, U_3 are all convex sets, assuming the network is realizable.

$$\begin{array}{rcl} 110 \in \mathcal{C} & \Rightarrow & U_1 \cap U_2 \neq \varnothing \\ 101 \in \mathcal{C} & \Rightarrow & (U_1 \cap U_3) \setminus U_2 \neq \varnothing \\ 011 \in \mathcal{C} & \Rightarrow & (U_2 \cap U_3) \setminus U_1 \neq \varnothing \end{array}$$

Since $p_1, p_2 \in U_3$ and U_3 is convex, $\ell_{p_1p_2} \subset U_3$. Two possibilities (since both U_1 and U_2 are closed sets [the argument also holds if both open]): $\ell_{p_1p_2}$ either intersects $U_1 \cap U_2$ or goes outside $U_1 \cup U_2$.

But $\ell_{p_1p_2} \subset U_3$ which means that either 111 or 001 are in the code. Contradiction. ☐

Exercise 3.6 (Homework 2)**.**

(1) Let W be an $n \times m$ matrix for a single-layer feed-forward network of LT neurons. Recall

$$y_i = H\left(\sum_{j=1}^{m} w_{ij}x_j - \theta_i\right)$$

Assume that W satisfies Dale's Law: a fixed input neuron has all outgoing edge weights positive or all negative. In terms of W, all columns are of the same sign.

Show that $\mathcal{C} \subset \{0,1\}^n$ for this network has a unique maximal codeword. In particular $\exists y^{max} \in \mathcal{C}$ such that $\forall y \in \mathcal{C}$, all $y_i \leq y_i^{max} \forall i \in [n]$.

(2) Find a *new* example of a code $\mathcal{C} \subseteq [0,1]^3$ that is not realizable by single-layer LT network.

4. September 8, 2015

4.1. **New Theme: Associative Memory Models.** Last week, we discussed feed-forward (FF) networks. This involved an input layer of neurons and an output layer of neurons. In multilayer networks, there could be intervening layers between input and output. This is one type of model for the visual system. For more information, see Rosenblatt's work on perceptrons.

This week, we discuss recurrent networks, which are used in associative memory models. In this setup, the connectivity graph can contain loops. A directed loop between two nodes can be considered an undirected edge.

Remark 4.1. We distinguish between:

(1) **Feedback in the network.** Here the network is primarily feed-forward but there is some feedback from later layers to earlier layers. One example is the model in Figure 2.

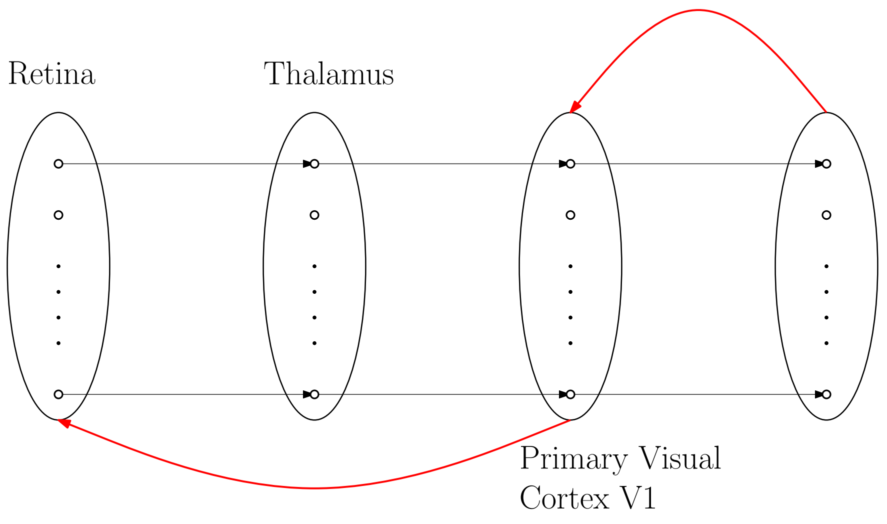


Figure 2. Visual network with feedback

(2) (Local) recurrent networks. V1 CA3 of hippocampus

Local Area Local connectivity

Local circuits: Associative memory. Storage and recall. Encoding of sequences of neural activity.

Neural computation: Partition completion Error Correction

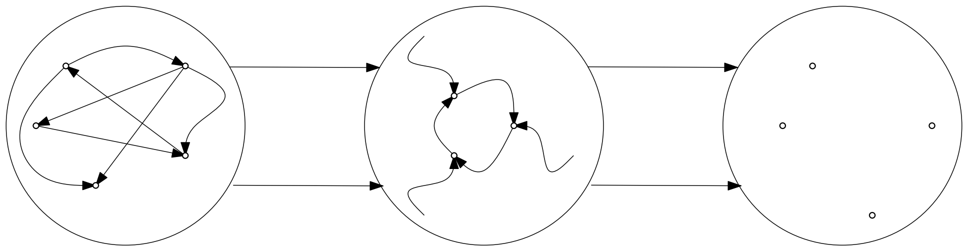


Figure 3. Recurrent Networks

The *Hopfield Model* was introduced by J.J. Hopfield, physicist by training, in a PNAS paper in 1982. This paper catalyzed a migration of physicists into theoretical neuroscience. It modeled "deep" questions about learning and memory in a simple mathematically tractable form.

In the 1980's, experiments in neuroscience still focused largely on one neuron at a time, applying stimuli to actual neurons in the brain. The ability to tackle questions at a network level was limited. The mathematical model allowed researchers to gain some insights, prove theorems, and perform numerical experiments.

4.2. **Hopfield Model.** We begin with **Ingredients**:

1. A symmetric weight matrix J which is $n \times n$, real valued.
2. A vector of thresholds $\theta = (\theta_1, \ldots, \theta_n)$.

Remark 4.2. The symmetry is known to be biologically unrealistic. Reciprocal connections occur more than expected by chance though not all the time. Values of J_{ij}, J_{ji} can never be exactly equal! This does not model inhibition explicitly. Instead the effects of inhibition are implicit when the value of J_{ij} is negative.

The Hopfield model is a "toy model" that helps us imagine the space of possibilities for recurrent network dynamics. It gives a qualitative picture of what might be going on.

Appendix E describes the prevailing ideas about memory networks. In the 90's, the influential ideas were Hebbian learning and cell assemblies. Even if it's not an accurate description, it is still informative.

Example 4.3. Consider a network on a graph G as in Figure 4.

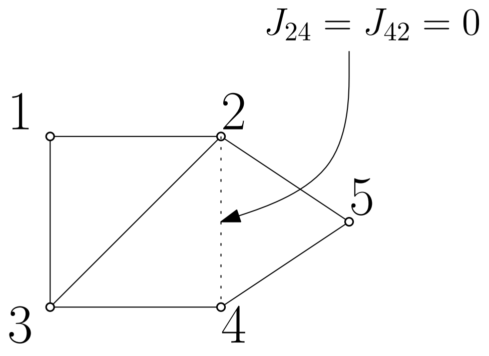


Figure 4. Example of a Hopfield network

Set the value for all θ to be 1, and

$$J_{ij} = \begin{cases} 1 & (ij) \in G \\ 0 & (ij) \notin G \end{cases}.$$

We consider the dynamics that update the state of the network at each time step.

Definition 4.4. A *State* of the network is a length n binary string $x \in \{0,1\}^n$; each index is labeled $x_i \in \{0,1\}$. (E.g. (11001 indicates that neurons 1,2, and 5 are on).

Remark 4.5. Note on notation: In the 1982 paper, x_i was taken to be in $\{-1.1\}$.

We also must consider *update rate*: starting from time t with state $x(t)$ and pass to state $x(t+1)$ the new state.

For $i = 1, \dots, n$ update by

$$x_i = H(J^{(i)}x - \theta_i$$

where H is the Heaviside function, and $J^{(i)}$ is the i-th row of J.

There are several possibilities for the update:

(1) Asynchronous:
  (a) In order: $1, 2, \dots, n$.
  (b) Randomly: Select $i \in [n]$ randomly at each step.
(2) Synchronous. This can yield behavior that is not robust - disappears with asynchronous updates.

Question 4.6. What does the network from our example do?

$$J = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Let us fix asynchronous ordered updates. Consider various initial states:

(1) $x_{init} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \end{pmatrix}^T = x(0)$. Any update returns the zero vector.
(2) $x(0) = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \end{pmatrix}^T$. The following series of updates happens:

$$\begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \overset{x_1}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \overset{x_2}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \overset{x_3}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \overset{x_4}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \overset{x_5}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \overset{x_1}{\to} \cdots$$

Thus the state $\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \end{pmatrix}^T$ is a fixed point. Note that (123) is a clique of G. Recall: a *clique* of a graph is an all-to-all connected subgraph.

(3) $x(0) = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \end{pmatrix}^T$. The following series of updates happens:

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} \overset{x_1}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} \overset{x_2}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} \overset{x_3}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} \overset{x_4}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \overset{x_5}{\to} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \overset{x_1}{\to} \cdots$$

Indeed, the state $\begin{pmatrix}1 & 1 & 1 & 1 & 1\end{pmatrix}^T$ is another fixed point of the network, even though it is not a clique.

5. September 10, 2015

5.1. **Hopfield Model.** Let J (or W) be the synaptic weight matrix. Two options:

(1) J fixed: no changes to snypases.
(2) J dynamic: J_{ij} is strengthened when neurons i, j are coactivated and J_{ij} decays otherwise.

Synaptic Plasticity refers to changes in synaptic strength.
Mathematical Model:

(1) Learning phase – training J, θ. Memories get "stored" as attractors of the network via modification of the weights W_{ij}.
(2) Retrieval phase –
    (a) Fix W.
    (b) Provide an input to the network (initial condition $x(0)$).
    (c) The network evolves to a steady state (attractor) yielding its output.

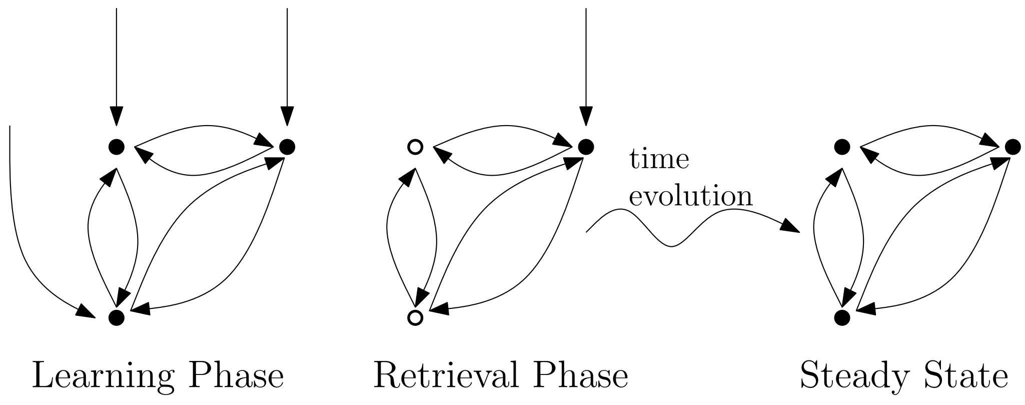


Figure 5. Hopfield Model with Dynamic Weights.

Question 5.1. What types of dynamics can occur in _____ networks?
For example: Hopfield, variations, threshold-linear, etc.

Fact 5.2. *Hopfield networks with symmetric J (or W) and $J_{ii} = 0$ always evolves to a steady state (a fixed point attractor) for asynchronous updates.*

Remark 5.3. There are two versions of the Hopfield model:
Version 1: $x_i(t+1) = H\left(\sum_{j=1}^{n} J_{ij}x_j(t) - \theta_i\right)$. Set $x_i \in \{0, 1\}$, with asynchronous update.
Version 2: States $S_i \in \{-1, 1\}$.

$$S_i(t+1) = \operatorname{sgn}\left(\sum_{j=1}^{n} W_{ij}S_j\right), \qquad \text{where } \operatorname{sgn}(x) = \begin{cases} +1 & x > 0 \\ -1 & x \leq 0 \end{cases}$$

Exercise 5.4 (Homework 3). (1) Consider Version 2 of the Hopfield model. Design a symmetric 2×2 synaptic matrix W with nonnegative diagonal so that there is a sequence of states – under *synchronous* update – that does not converge to any fixed point attractor.
(2) Design an asymmetric 2×2 matrix W with nonnegative diagonal s.t. there are no fixed points (again, for synchronous update rule.)

We return our attention to Version 1 of the Hopfield model.

Key Idea: An "energy" function for the network

$$E(x) = -\frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n} J_{ij}x_ix_j + \sum_{i=1}^{n}\theta_i x_i \tag{*}$$

E is a function dependent on J, θ; x is the state of the network at a given time.

[See also the Ising Model, with a similar energy function.] Think of this function as a (discretized) *Lyapunov function.*

Definition 5.5. For a dynamical system, a *Lyapunov function* is a function that evaluated at states $f(x)$ is always decreasing (non-increasing) along trajectories.

If you have such an "energy function" you know that your trajectories will converge to states corresponding to minima of $E(x)$. Idea: The network is optimizing something! It's going to minima of $E(x)$.

Theorem 5.6. *Consider Hopfield model (version 1), with update function*

$$x_i(t+1) = H\left(\sum_{j=1}^{n} W_{ij}x_j(t) - \theta_i\right)$$

where $x_i \in \{0,1\}$ and asynchronous update. Then the energy function $E(x)$ (*) is non-increasing along trajectories of the dynamics.

Specifically, if $x \to x'$ in one time step, then $E(x) - E(x') \geq 0$, with equality holding if and only if $x = x'$.

Proof. Because update is asynchronous, x and x' can only differ in a single neuron. There exists k such that $x_i = x'_i$ for all $i \neq k$. Suppose $x_k \neq x'_k$. (If $x_k = x'_k$, the energies are equal.)

Compute:

$$\begin{aligned} E(x) - E(x') &= \left(-\sum_{j=1}^{n} W_{kj}x_kx_j + \sum_{i=1}^{n}\theta_i x_i\right) - \left(-\sum_{j=1}^{n} W_{kj}x'_kx_j + \sum_{i=1}^{n}\theta_i x_i\right) \\ &= -(x_k - x'_k)\sum_{j=1}^{n} W_{kj}x_j + (x_k - x'_k)\theta \\ &= -(x_k - x'_k)\left(\sum_{j=1}^{n} W_{kj}x_j - \theta_k\right) = -(x_k - x'_k)e_k, \end{aligned}$$

where e_k is the argument of the Heaviside function.

<u>Cases</u>:

(1) If $x'_k > x_k$ then the sign of the first factor is negative, and the sign of e_k is positive so the total is positive.

(2) If $x'_k < x_k$ then the sign of the first factor is positive, and the sign of e_k is negative so the total is positive.

□

Remark 5.7. Dynamical Systems Concepts – *Multistability*: The existence of multiple steady states (fixed points), or stored memories. This only occurs in nonlinear dynamics (because of the Heaviside function). Without the Heaviside function, all we would have is a system of linear ODEs, which would have at most one fixed point, which would be stable or unstable.

6. September 17, 2015

Missed one class on Sep 15.

6.1. **Threshold-linear Networks.** Today we are discussing the paper "Permitted and Forbidden Sets in Symmetric Threshold-Linear Networks" by Hahnloser and Seung.

The system discussed has this form:

$$(*) \qquad \frac{dx_i}{dt} = -x_i + \left[\sum_{j=1}^{n} W_{ij}x_j + b_i\right]_+$$

Conditions are $x_i \geq 0$ i.e. $x \in \mathbb{R}^n_{\geq 0}$ the nonnegative orthant and $b_i \in \mathbb{R}$. τ is set to 1.

Differential equation, continuous time (vs. discrete time update rule) We will consider attractors of this network for arbitrary b. *Think:* $b_i = e_i - \theta_i$. External input minus threshold.

Notation 6.1. For $\nu \in \mathbb{R}^n$, we write $\nu \geq 0$ to indicate that $\nu_i \geq 0$ for all $i \in [n]$. I.e. $\nu \in \mathbb{R}^n_{\geq 0}$ nonnegative orthant.

Question 6.2. Under what conditions can we guarantee that all trajectories (initialized with $x(0) \in \mathbb{R}^n_{\geq 0}$ converge to a steady state)?

Theorem 6.3. *Assuming W symmetric, the following are equivalent:*

(1) All positive eigenvectors of all principal submatrices of $I - W$ have positive eigenvalues. The principal submatrices are defined by fixing the same set $\sigma \subseteq [n]$ as the set of rows and columns.

(2) $I - W$ is co-positive:

$$x^T(I-W)x > 0 \qquad \forall x \geq 0, \text{ except } x = 0.$$

(3) For all $b \in \mathbb{R}^n$ and all initial conditions, the dynamics of (*) converge to an equilibrium point.

Proof of (2) ⇒ (3). Suppose $I - W$ is co-positive. **Strategy**: Find a Lyapunov function that is strictly decreasing under the dynamics (*) and is only 0 at steady states. How about this:

$$\begin{aligned} L &= \tfrac{1}{2}x^T(I-W)x - b^Tx \\ &= \tfrac{1}{2}\sum x_i^2 - \sum_{i,j=1}^{n} W_{ij}x_ix_j - \sum_{i=1}^{n} b_ix_i. \end{aligned}$$

Compute the derivative:

$$\begin{aligned} \dot{L} = \frac{dL}{dt} &= \dot{x}^T(x - Wx - b) \\ &= (-x + [wx+b]_+)^T(x - (wx+b)) \end{aligned}$$

This computation depends on symmetry of W.

Let $y = Wx + b$, and let $y^+ = [Wx+b]_+ = [y]_+$. Then, substituting:

$$\dot{L} = -(x - y^+)^T(x-y) = -\sum_{i=1}^{n}(x-y^+)_i(x-y)_i.$$

To prove this derivative is negative we can show that each summand $(x_i - y_i^+)(x_i - y_i)$ is positive. We go case by case:

Case 1 $y_i^+ = y_i \geq 0$. Then $(x_i - y_i)^2 \geq 0$.

Case 2 $y_i \leq 0 \Rightarrow y_i^+ = 0$. Then,

$$(x_i - y_i^+)(x_i - y_i) = x_i \cdot (x_i - y_i) \geq 0,$$

since x_i and $-y_i$ are both nonnegative.

What is left? We still need to show that the equilibria of L (i.e. where $\dot{L} = 0$) correspond to steady states. Specifically, we must show that $\dot{L} = 0 \Rightarrow \dot{x} = 0$.

$$\begin{aligned} \dot{L} &= -(x - y^+)^T(x - y) &= 0 \\ &\Rightarrow \quad (x_i - y_i^+)(x_i - y_i) &= 0 \quad \forall i \end{aligned}$$

since the summands all have the same sign. This leaves two options:

(1) $x_i - y_i^+ = 0 \Rightarrow x_i = y_i^+$. Since $\dot{x}_i = -(x_i - y_i^+)$, this implies that $\dot{x}_i = 0$.
(2) $x_i - y_i = 0 \Rightarrow x_i = y_i \geq 0$. This means that $y_i = y_i^+$ which gives us the same result as case 1: $\dot{x}_i = 0$.

Therefore, $\dot{L} = 0 \Rightarrow \dot{x} = 0$ so we have an equilibrium point. □

Remark 6.4. Since we are allowed to vary b, we can always find network (W, b) such that v is a fixed point of (*) for any $v \geq 0$.

Lemma 6.5. *For any $v \geq 0$, there exists b such that v is a steady state of (*) with input b.*

Warning: v won't necessarily be stable.

Proof. Choose $b = (I - W)v$. Let $x = v$ and compute $\dot{x}$.

$$\begin{aligned} \dot{x} &= -x + [Wx + b]_+ \\ &= -v + [Wv + (I - W)v]_+ \\ &= -v + [Iv]_+ \\ &= -v + [v]_+ = 0, \end{aligned}$$

since v is presumed to be positive. □

The idea is that there are some collection of stable steady states. The computation of the network takes some input and sends it to the closest stable fixed point, as imagined in Figure 6.

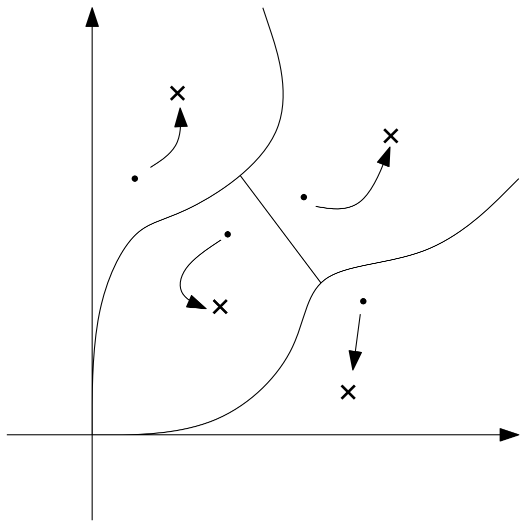


Figure 6. Computation of Memory Network

Exercise 6.6 (Homework 4). (1) What are permitted sets?

(2) Find all permitted sets for

$$W = \begin{pmatrix} 0 & a & a & b & b & b \\ a & 0 & a & a & b & b \\ a & a & 0 & a & b & b \\ b & a & a & 0 & a & b \\ b & b & b & a & 0 & b \\ b & b & b & b & b & 0 \end{pmatrix}$$

where $a = -1/2$ and $b = -3/2$.

7. September 22, 2015

7.1. **Permitted and Forbidden Sets.**

Definition 7.1. A steady state $x^* \in \mathbb{R}^n_{\geq 0}$ is *stable* if for all open balls B with $x^* \in B$ there exists an open ball A with $x^* \in A$ such that for any initial condition $x(0) \in A$, we have $x(t) \in B$ for all $t \geq 0$.

A steady state is *asymptotically stable* if it is stable and there exists a ball of initial conditions $x(0)$ that converge to it. [This is a stricter condition]

Definition 7.2. A neuron is activated if its activity is nonzero ($x_i > 0$).

A set of neurons $\sigma \subseteq [n] = \{1, \ldots, n\}$ is permitted with respect to $\dot{x} = -x + [Wx + b]_+$ if the neurons can be co-activated at an asymptotically stable steady state for some input $b \in \mathbb{R}^n$.

Otherwise the set of neurons is forbidden.

Fact 7.3. *For fixed (W, b), there can be a most one steady state x^* for each $\sigma \subseteq [n]$.*

Note: The definition of permitted sets does not depend on a choice of b.

Proposition 7.4. *Fix W and b. If there exists a fixed point x^* of $\dot{x} = -x + [Wx + b]_+$ such that $(-I + W)_{\sigma\sigma}$ is invertible and supp$(x^*) = \sigma$, then there is no other fixed point with support σ.*

For $x \in \mathbb{R}^n_{\geq 0}$ we define $\text{supp}(x) = \{i \in [n] \mid x_i > 0\}$.

Proof of Proposition 7.4. Let x^* be a fixed point of (*) with $\text{supp}(x^*) = \sigma \subseteq [n]$. Then because x^* is a fixed point, $\dot{x}|_{x=x^*} = 0$,

$$x^* = [Wx^* + b]_+,$$

or entrywise,

$$x_i^* = [\sum_{j=1}^{n} W_{ij} x_j^* + b_i]_+.$$

If $x_i^* > 0$ i.e. $i \in \sigma$:

$$\begin{array}{rrcl} & x_i^* & = & \sum_{j=1}^n W_{ij} x_j^* + b_i. \\ \Rightarrow & x_\sigma^* & = & W_{\sigma\sigma} x_\sigma^* + b_\sigma \\ \Rightarrow & (I - W)_{\sigma\sigma} x_\sigma^* & = & b_\sigma \\ \Rightarrow & x_\sigma^* & = & (I - W)_{\sigma\sigma}^{-1} b_\sigma \end{array}$$

This is the unique fixed point, though it is only valid if it is positive. □

This is motivation for focusing attention on permitted sets, rather than fixed points.

Theorem 7.5 (Useful)**.** *Given $n \times n$ matrix W, with $\sigma \subseteq [n]$ is a permitted set of (*) $\iff$ $(-I + W)_{\sigma\sigma}$ is a stable (Hurwitz) matrix; i.e. all eigenvalues have strictly negative real part.*

For W symmetric, we can use the "Useful" theorem together with Cauchy's interlacing theorem to obtain hierarchical structure of permitted sets.

Theorem 7.6. *Let A be a symmetric $N \times N$ matrix with real eignevalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$. Let B be an $(N-1) \times (N-1)$ principal submatrix of A with eigenvalues $\eta_1 \leq \cdots \eta_{N-1}$. The eigenvalues of B interlace the eigenvalues of A:*

$$\lambda_1 \leq \eta_1 \leq \lambda_2 \leq \eta_2 \leq \cdots \leq \eta_{N-1} \leq \lambda_N.$$

Corollary 7.7. *If A is stable then so is B.*
If σ is a permitted set, then so is τ for all $\tau \subset \sigma$. (Only for W symmetric)

Proof. σ permitted implies $(-I+W)_{\sigma\sigma}$ stable, then by induction using the Interlacing theorem, every principal submatrix is stable too. □

For next class, we will finish the "Permitted and Forbidden Sets" paper.

7.2. **Abstract simplicial complex.**

Definition 7.8. An abstract simplicial complex on $[n]$ is a collection of subsets $\Delta \subseteq 2^{[n]}$ such that $\sigma \in \Delta$ and $\tau \subseteq \sigma$ implies $\tau \in \Delta$.

Example 7.9. $\Delta = \{123, 34\}$ and all their subsets.