BIOSYSTEMS II: NEUROSCIENCES

2015 Spring Semester

Lecture 31

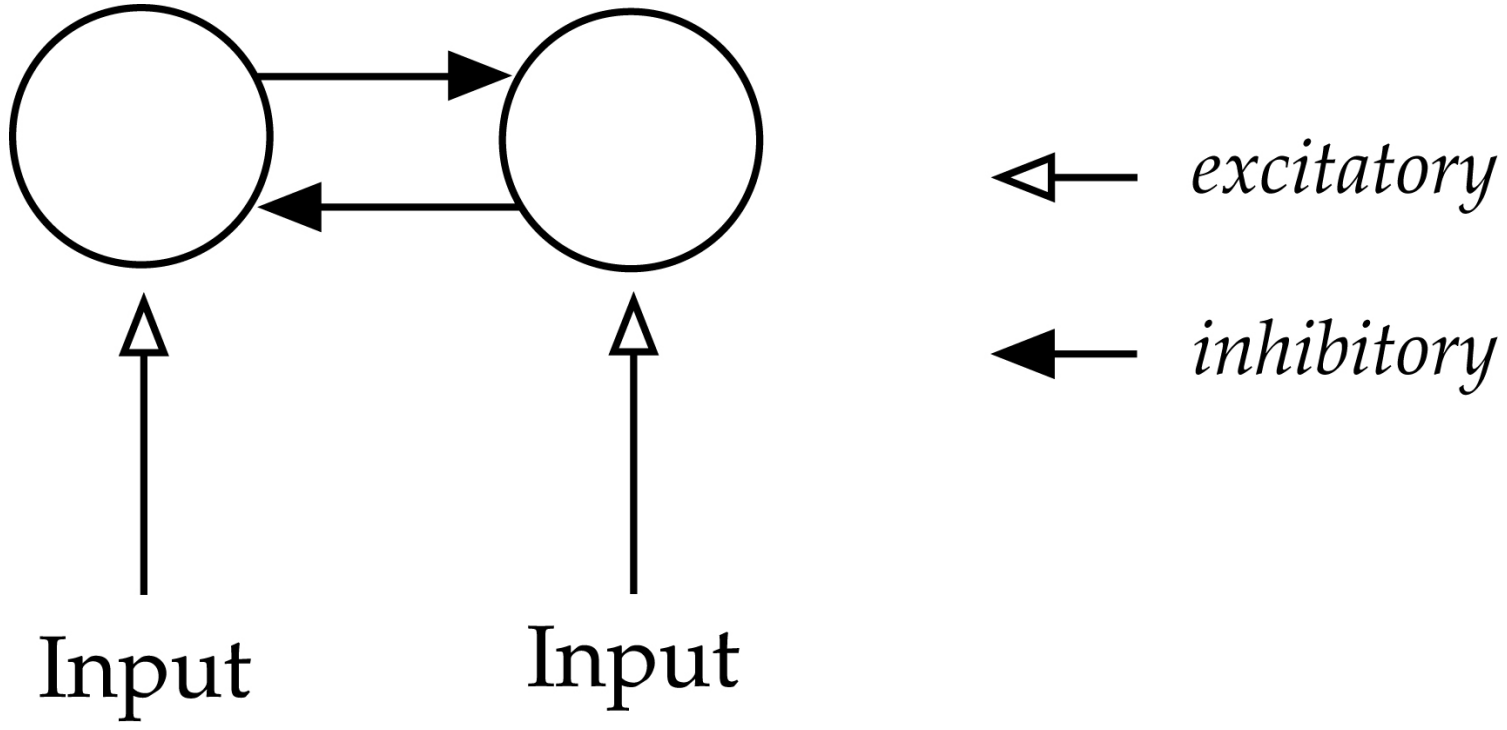
Kechen Zhang

4/13/2015

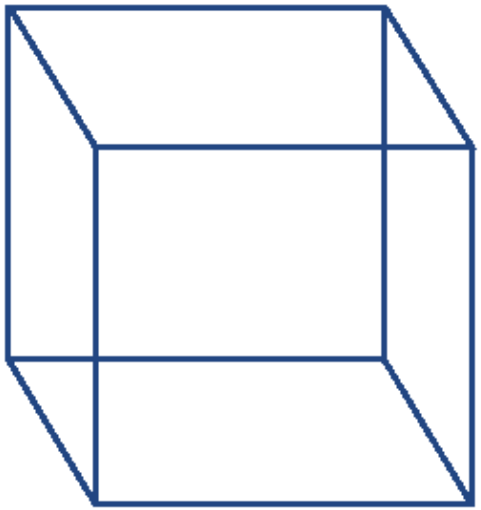
Examples of recurrent networks

- The same input may lead to different outputs (equilibrium states), depending on the initial state and noise.
- Hopfield network for memory storage and retrieval
- Oscillations

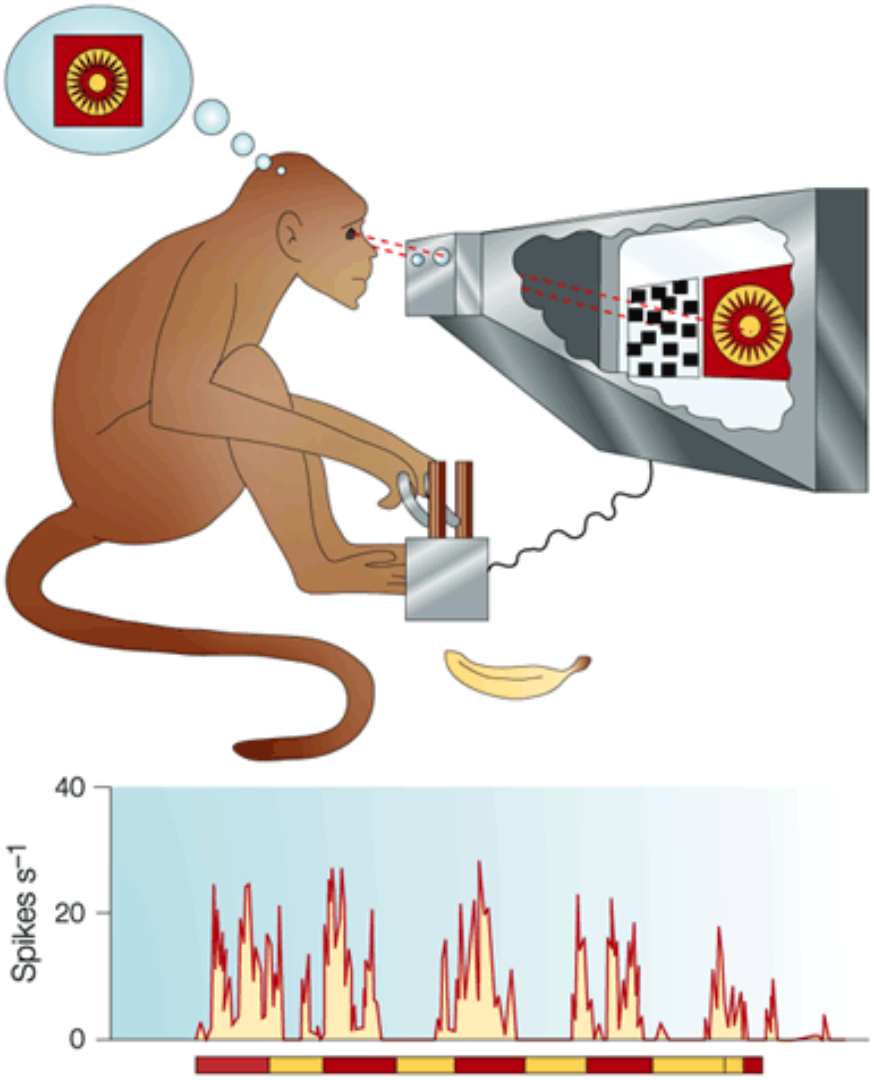
Rivalry of mutually inhibitory neuronal groups



Ambiguous figures: multiple percepts

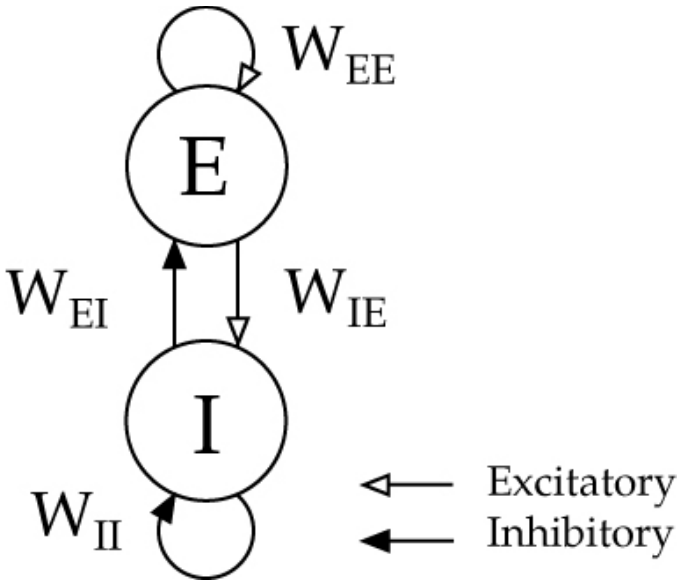


Binocular Rivalry Experiment



When two different pictures are shown to the two eyes, an animal typically sees only one of them. The monkey indicated which picture he saw by pressing a level. The perceived picture alternated randomly in time (colored bar at the bottom), but was well correlated with the firing rate of a neuron in the visual cortex.

Dynamics of excitatory and inhibitory neuronal populations



$$\frac{dE}{dt} = -E + W_{EE}g(E) + W_{EI}g(I) + b_E$$

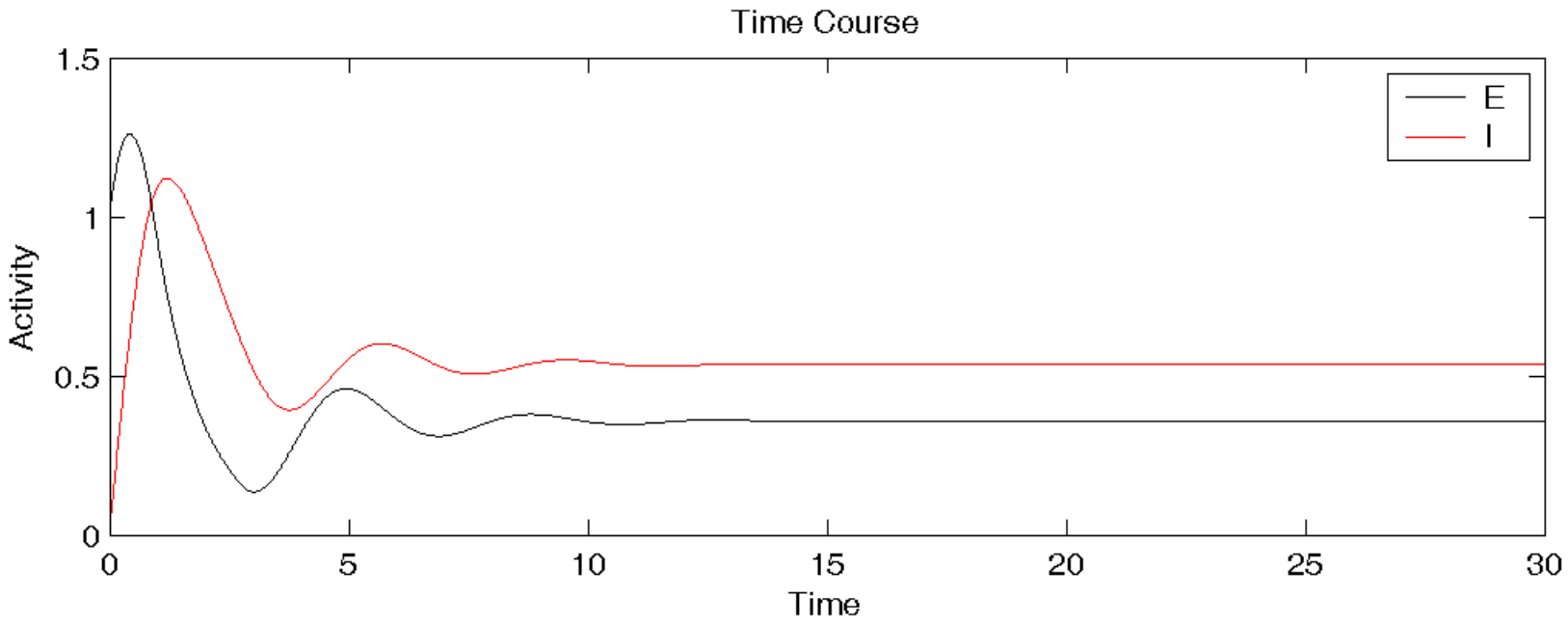
$$\frac{dI}{dt} = -I + W_{II}g(I) + W_{IE}g(E) + b_I$$

where $g(\ )$ is gain function,

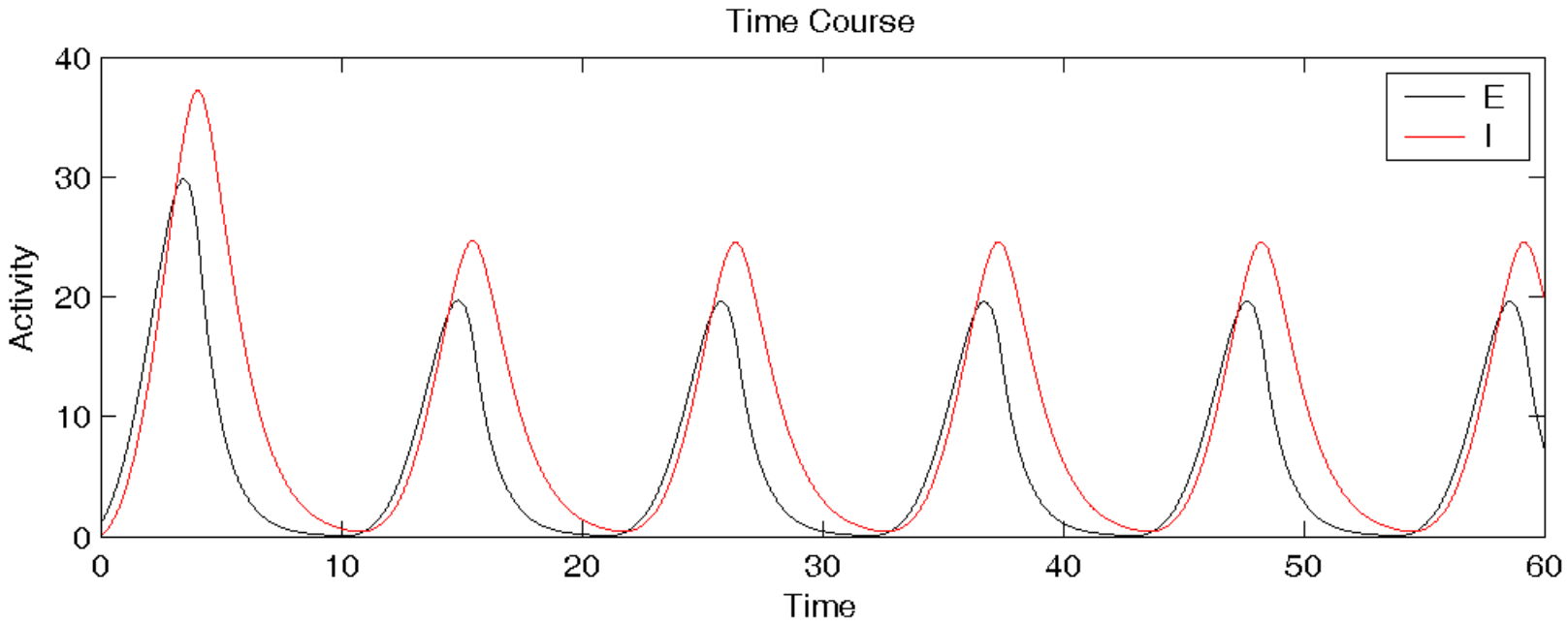
and b_E and b_I are the

external inputs to the E and I

populations, respectively.

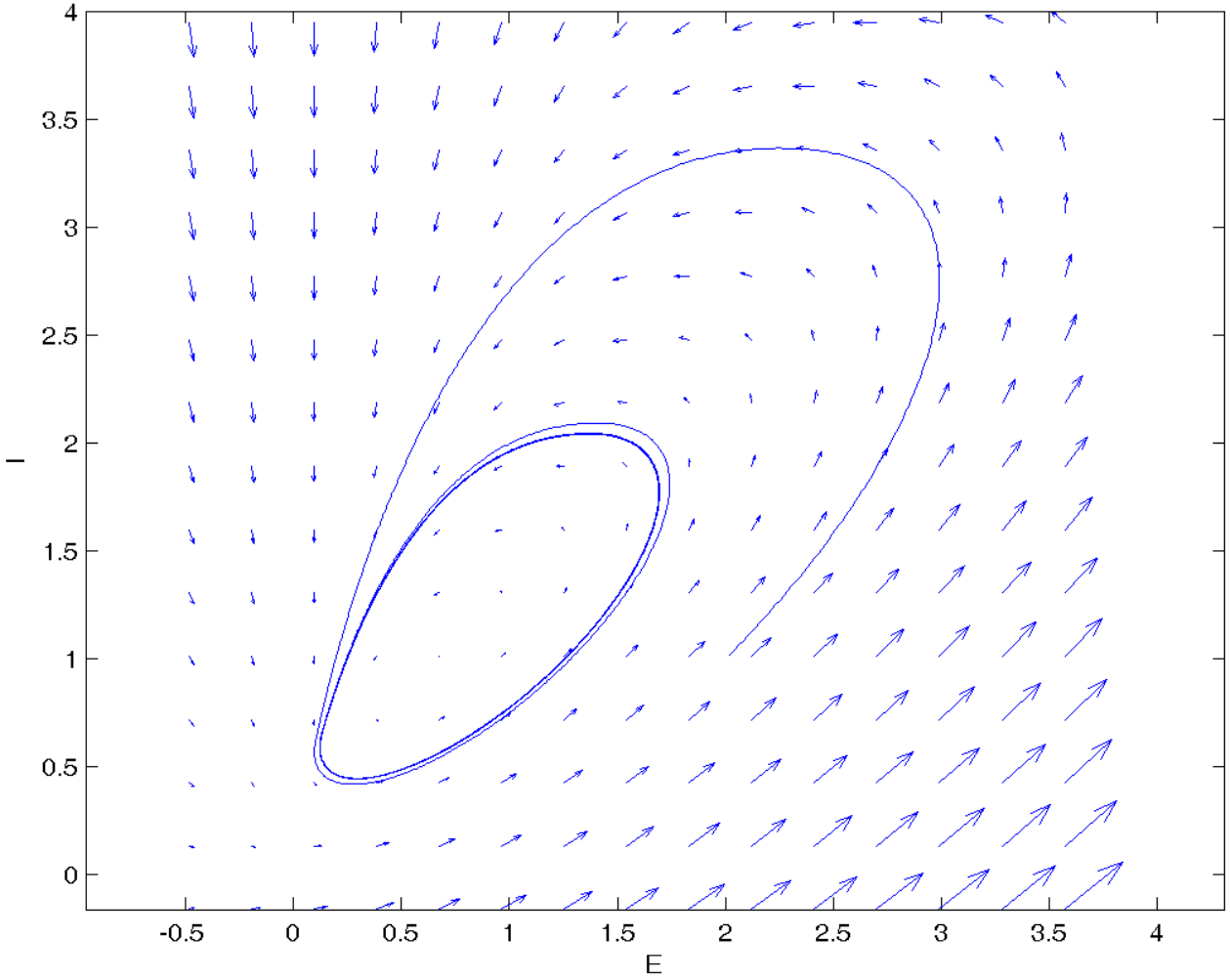


Spontaneous time evolution of the excitatory-inhibitory network that approaches a stationary state. The external inputs are always kept constant here.



Another example that shows periodic oscillation of the excitatory-inhibitory network. All parameters are identical to those in the preceding example except that the value of the excitatory-to-excitatory synaptic connection (W_{EE}) is stronger.

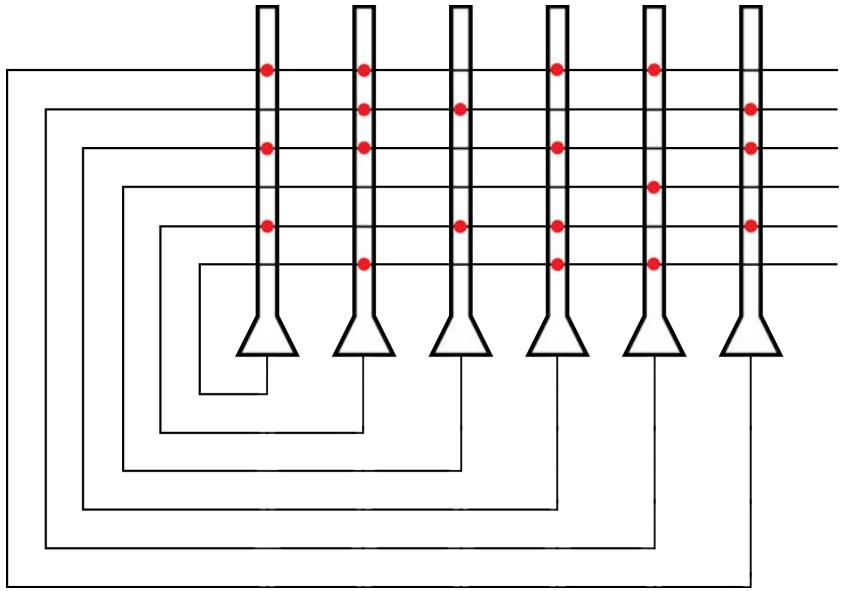
Phase plane analysis



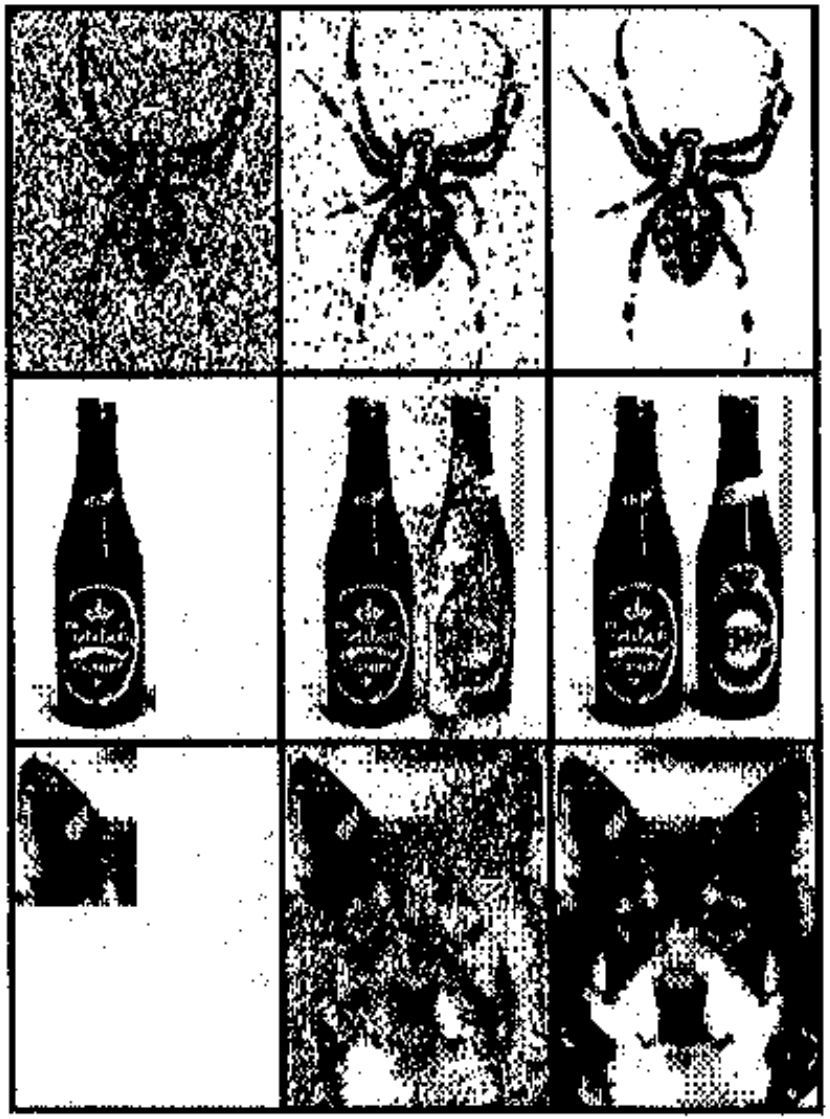
The arrows indicate the velocity vector (dE/dt, dI/dt). Starting from any initial state, the time evolution of the state (E, I) traces a trajectory that follows the local direction of the velocity vector. Robust oscillation of the excitatory-inhibitory network implies the existence of a limit cycle in the phase plane.

Recurrent networks

In a fully recurrent network, each neuron may potentially have synaptic connections with every other neuron. An interesting example of recurrent network is a Hopfield network, where the dynamics of each individual neuron or unit is extremely simple, but because these neurons are connected together in a clever way, the network as a whole has useful emergent or collective properties.



Initial state → intermediate → final state



Hopfield network

- Fully connected recurrent network
- Each memory pattern is a stable state of the network
- Multiple memory patterns can be stored in the same network
- Memory retrieval from partial cue (content-addressable associative memory)
- Robust against noise and damages such as deleting units and connections
- One-shot Hebbian learning for adding a new memory pattern

Hopfield Network: Discrete Dynamics

The activity or state s_i of unit i at each time step depends on the states of all other units at the preceding time step:

$$s_i(t+1) = \text{sign}\left(\sum_j w_{ij} s_j(t) \right)$$

where the sign function is the nonlinear gain function,
with $\text{sign}(x) = 1$ for positive x and $= -1$ for negative x,
and w_{ij} is the weight of the connection from unit j to unit i.

Hopfield Network: Continuous Dynamics

The state u_i of unit i at time t is governed by the differential equation

$$\frac{du_i(t)}{dt} = -u_i(t) + \sum_j w_{ij} s_j(t)$$

where $s_j(t) = g\left(u_j(t)\right)$, with g a monotonically increasing gain function. In an equilibrium state, we have $u_i(t) = \sum_j w_{ij} s_j(t)$ and $s_i(t) = g\left(\sum_j w_{ij} s_j(t)\right)$.

Weight Rule in a Hopfield Network and Hebbian Learning

To store a memory pattern $(S_1, S_2, S_3, ...)$ where the state S_i of each unit i is either -1 or $+1$ with equal probability, the weight should be

$$w_{ij} = S_i S_j$$

This rule resembles Hebb's idea of synaptic learning because the weight increases when both pre - and post - synaptic neurons are active, but not when one is active and the other is inactive.

The analogy is not completely correct because the weight increases also when both pre - and post - synaptic neurons are inactive.

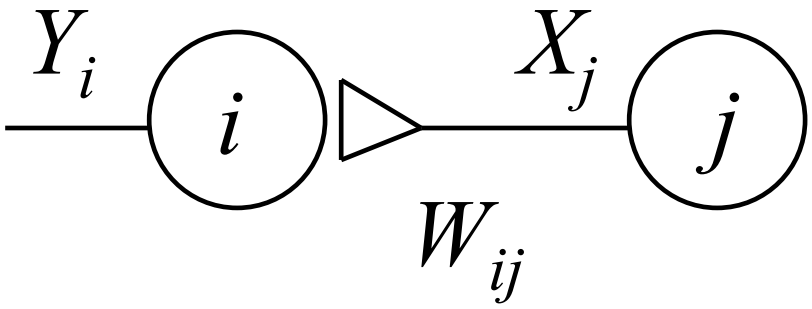
Hebb's learning rule

A neurophysiological postulate:

When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.

D. O. Hebb: The Organization of Behavior, 1949.

Simple Hebb rule for synaptic learning



$$\Delta W_{ij} = cX_jY_i$$

where X_j and Y_i are presynaptic and postsynaptic activities, respectively, and $c > 0$ is learning rate.
Another modified learning rule is:

$$\Delta W_{ij} = c\left(X_j - \bar{X}_j\right)\left(Y_i - \bar{Y}_i\right)$$

where a bar indicates the value of past average.

Storage of multiple memory patterns

We start from the weight for storing a single memory pattern $(S_1, S_2, S_3, ...)$:

$$w_{ij} = S_i S_j$$

If another memory pattern $(S_1', S_2', S_3', ...)$ with $S_i' = \pm 1$ with equal probability also needs to be stored, then the weights should be

$$w_{ij} = S_i S_j + S_i' S_j'$$

The two memory patterns have no interference if they are statistically independent of each other. To store more memory patterns, use $w_{ij} = S_i S_j + S_i' S_j' + S_i'' S_j'' + \cdots$ and so on. Storage of too many noisy memory patterns can generate interference. The maximum number of memory patterns stored in a Hopfield network is typically less than 14% of the total number of neurons in the network.

Each stored memory state is a stationary state :

In other words, if $s_i(t) = S_i$, then $s_i(t+1) = S_i$ $(i = 1,2,...,N,$ where N is the total number of neurons in the network).

$$\sum_{j=1}^{N} w_{ij} S_j = \sum_{j=1}^{N} (S_i S_j + S_i' S_j' + \cdots) S_j = S_i \sum_{j=1}^{N} S_j S_j + S_i' \sum_{j=1}^{N} S_j' S_j + \cdots$$

$$= S_i N + S_i' O\left(\sqrt{N}\right) + \cdots$$

The second sum (cross-talk term) is of the same order of magnitude as $\sqrt{N}$ because different memory patterns are assumed to be independent so that $S_j' S_j = \pm 1$ with equal chance.

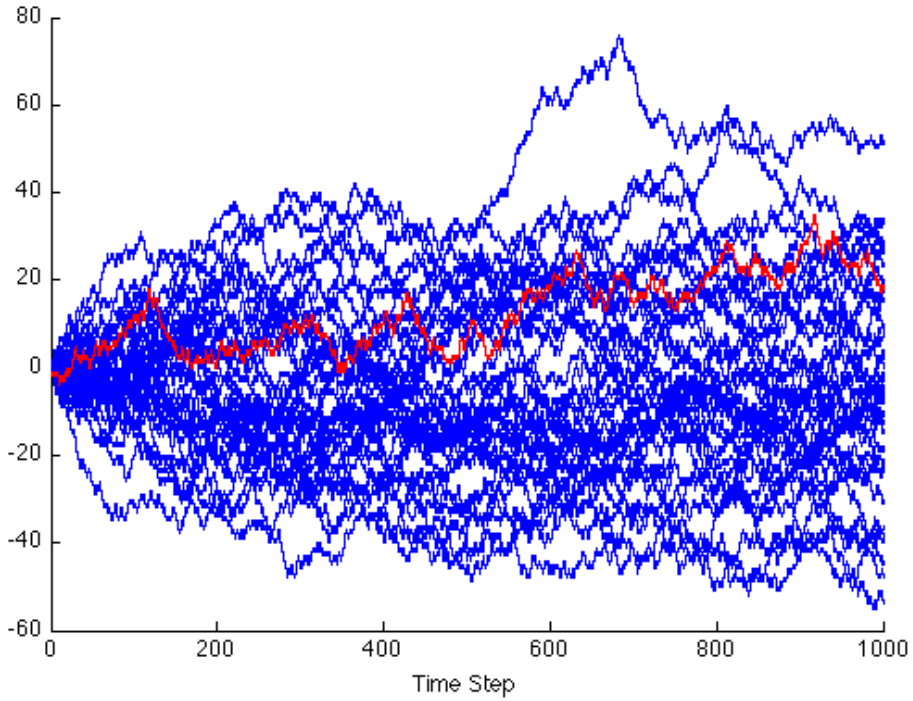
For large N, the first term determines the sign.

So if $s_i(t) = S_i$, then $s_i(t+1) = \text{sign}\left(\sum_{j=1}^{N} w_{ij} S_j \right) = \text{sign}(S_i N) = S_i$.

Random walk math

Suppose each step $x_i = -1$ or $+1$ with equal probability. The total distance covered by N steps is $L = \sum_{i=1}^{N} x_i$. Consider $L^2 = \sum_{i=1}^{N} x_i^2 + \sum_{i \neq j}^{N} x_i x_j = N + \sum_{i \neq j}^{N} x_i x_j$.

Average over an ensemble of paths yields $\left\langle L^2 \right\rangle = N$.



Inner product and outer product between vectors

Let $\mathbf{a} = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}$ and $\mathbf{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$ be two column vectors.

Their inner product $\mathbf{a}^{\mathrm{T}}\mathbf{b} = a_1 b_1 + \cdots + a_n b_n$ is a scalar.

Their outer product is a matrix:

$$\mathbf{a}\mathbf{b}^{\mathrm{T}} = \begin{pmatrix} a_1 b_1 & a_1 b_2 & \cdots & a_1 b_n \\ a_2 b_1 & a_2 b_2 & \cdots & a_2 b_n \\ \vdots & \vdots & \ddots & \vdots \\ a_n b_1 & a_n b_2 & \cdots & a_n b_n \end{pmatrix}$$

Hebbian learning a Hopfield network

The weight matrix is a correlation matrix of the inputs:

$$\mathbf{W} = \mathbf{s}_1\mathbf{s}_1^{\mathrm{T}} + \mathbf{s}_2\mathbf{s}_2^{\mathrm{T}} + \cdots + \mathbf{s}_m\mathbf{s}_m^{\mathrm{T}}$$

where the memory patterns are orthogonal to one another:

$$\mathbf{s}_i^{\mathrm{T}}\mathbf{s}_j = \begin{cases} N & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

where N is the number of units in the network. Then

$$\mathbf{W}\mathbf{s}_i = N\mathbf{s}_i$$

Each memory pattern is an eigenvector of the weight matrix.

Summary of Analysis

The analysis in the preceding slide shows the following :

First, each memory state is stationary in the sense that once the network state reaches a stored memory state, it will stay there indefinitely.

Second, different memory patterns that are statistically independent of one another can be stored in the same network without interferences.

Third, similar analysis can show that each memory state is not only stationary but also stable. That is, if the network starts from a state sufficiently close to a stored memory state, then it will reach the stored memory state in the next time step. In other words, if $s_i(t)$ is similar to S_i, then at the next time step, we should still have $s_i(t+1) = S_i \quad (i = 1,2,..., N)$.

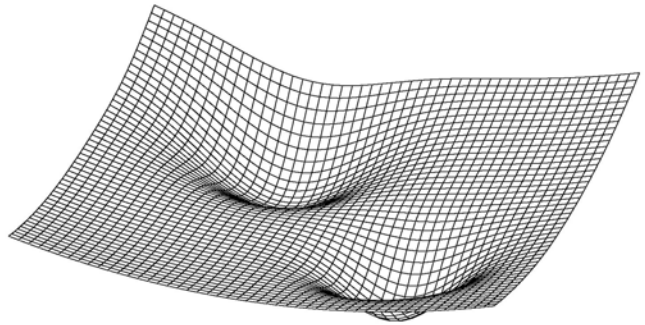
Robustness against noise and damage

A Hopfield is robust again noise or unreliability of individual neurons. It is also robust against damages to the connections. We can see from the analysis above that each neuron in a Hopfield network changes its state according to the weighted sum of the inputs from all other neurons in the entire network. The contribution from any small subsets of neurons is small and can be completely ignored without affecting the state of the network in the next time step.

Hopfield Network: Energy (Liapunov) Function

The energy function

$$H(t) = -\frac{1}{2}\sum_{i,j} w_{ij} s_i(t) s_j(t)$$

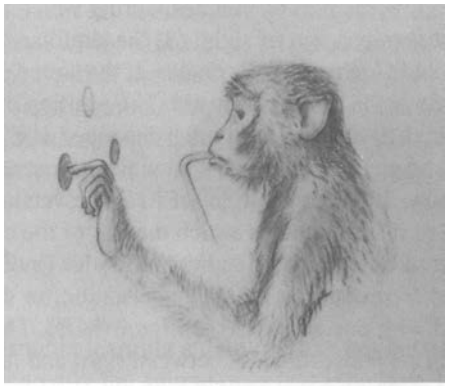


always decreases as the network evolves in time:

$$H(t+1) \leq H(t).$$

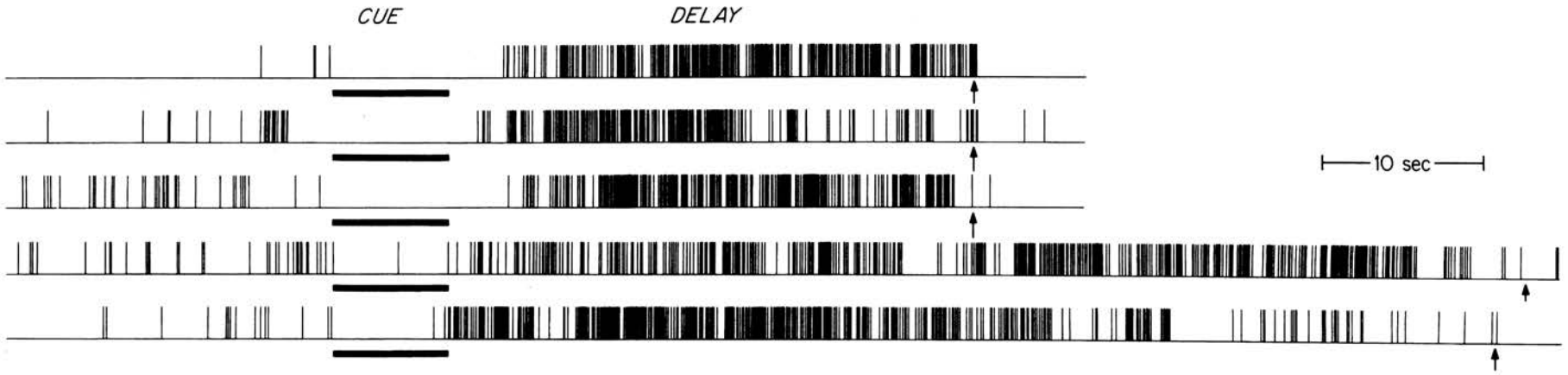
In other words, the system spontaneously seeks low energy states. Each stored memory pattern in the network correspond to a minimum of the energy function. The memory states are called point attractors because a network state close enough to a stored memory pattern will be attracted towards it and the full memory pattern can be recovered.

Memory-related persistent activity in prefrontal cortex



Delayed-Match-to-Sample task

A recurrent network is capable of sustaining the persistent activity during delay period as an attractor state



(Fuster)

Spontaneous cortical activity patterns that resemble point attractors