# Introduction to Neural Computation

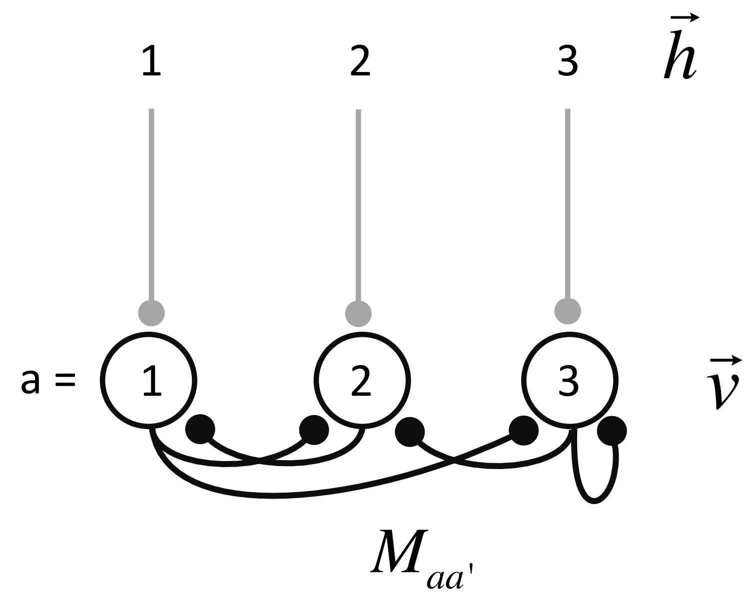
Prof. Michale Fee
MIT BCS 9.40 — 2018

Lecture 20
Hopfield Networks

# Recurrent networks

- We have been considering the case where there are connections between different neurons in the output layer

- Develop an intuition for how recurrent networks respond to their inputs




- Examine computations performed by recurrent networks

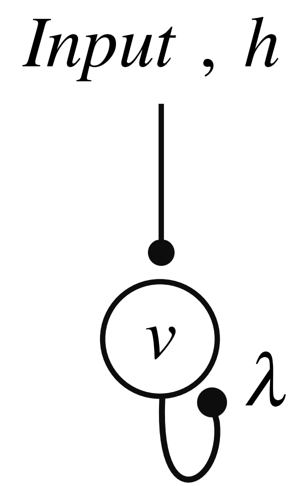
- Use all the powerful linear algebra tools we have developed

# Recurrent networks

- Simplest recurrent network – a neuron with an autapse

$$\tau_n \frac{dv}{dt} = -v + h + \lambda v$$

$$\tau_n \frac{dv}{dt} = -\underbrace{(1-\lambda)}_{}v + h$$




- We examined three cases:

$\lambda < 1$ $\qquad$ $\lambda = 1$ $\qquad$ $\lambda > 1$

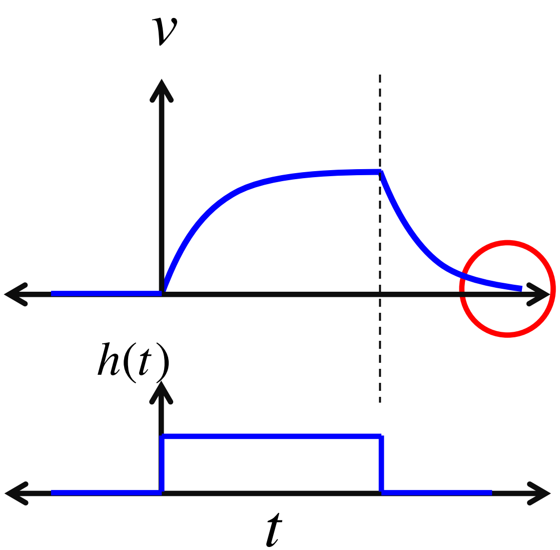
# Recurrent networks

$$\tau_n \frac{dv}{dt} = -(1-\lambda)v + h$$

- The behavior of the network depends critically on $\lambda$

$\lambda < 1$

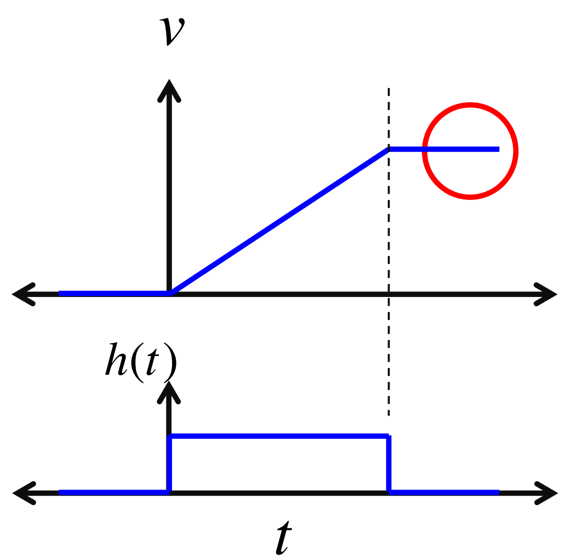
Exponential relaxation




With zero input…
relaxation back to zero

$\lambda = 1$

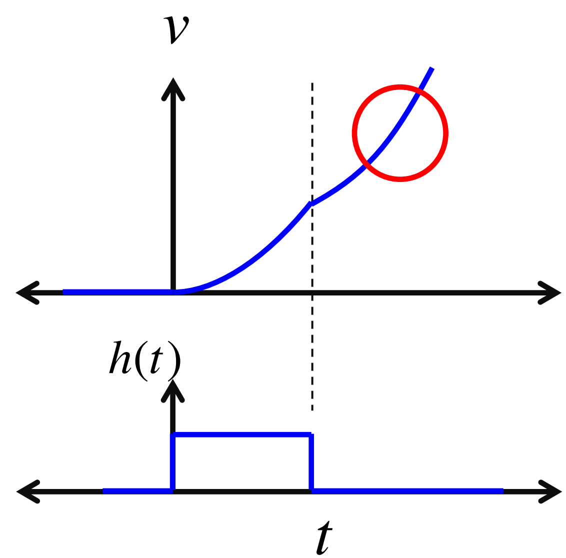
Integration




With zero input…
persistent activity!

$\lambda > 1$

Exponential growth




MEMORY!

# Learning Objectives for Lecture 20

- Recurrent networks with lambda greater than one
  - Attractors
- Winner-take-all networks
- Attractor networks for long-term memory (Hopfield model)
- Energy landscape
- Hopfield network capacity

# Learning Objectives for Lecture 20

- Recurrent networks with lambda greater than one
  - Attractors

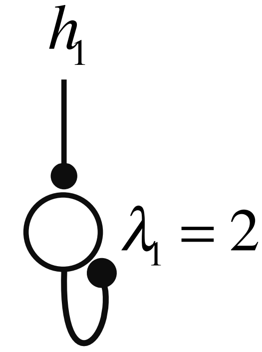
- Winner-take-all networks

- Attractor networks for long-term memory (Hopfield model)

- Energy landscape

- Hopfield network capacity

# Recurrent networks

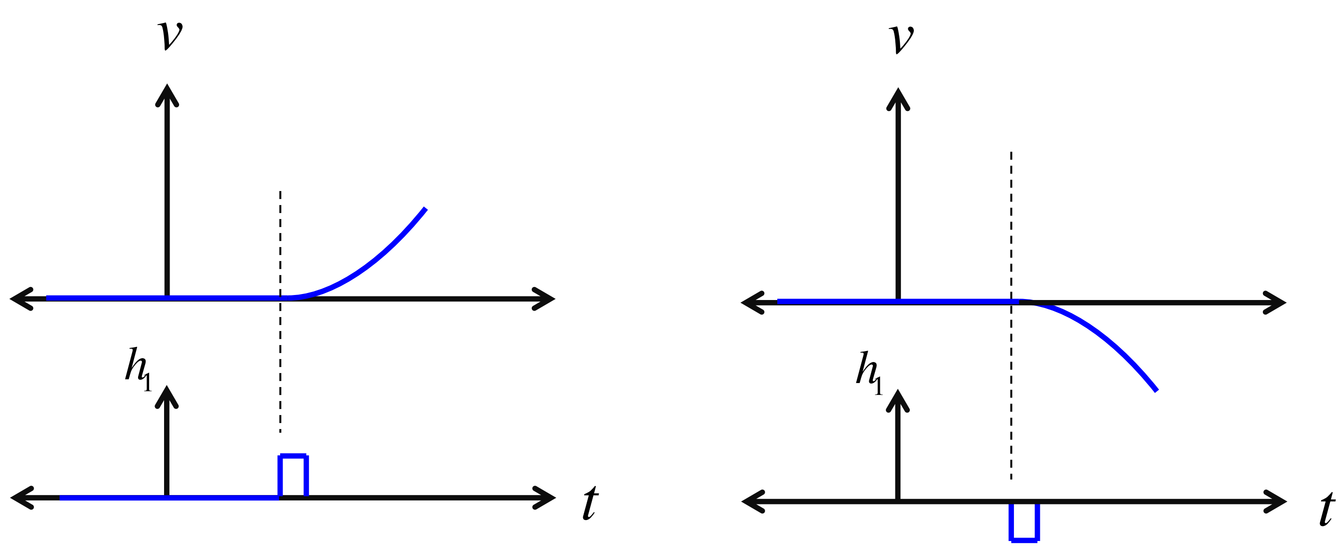



- Networks with $\lambda \geq 1$ have memory!

$$\tau_n \frac{dv}{dt} = (\lambda_1 - 1)v + h(t)$$

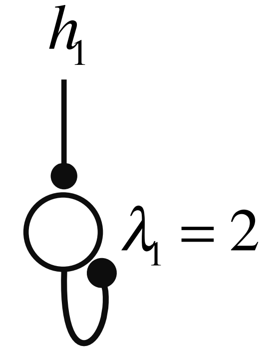
$$\tau_n \frac{dv}{dt} = v \qquad\qquad v(t) = 0$$

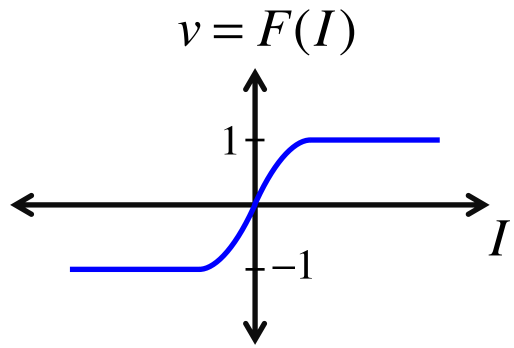
- With zero input, zero is an 'unstable fixed point' of the network

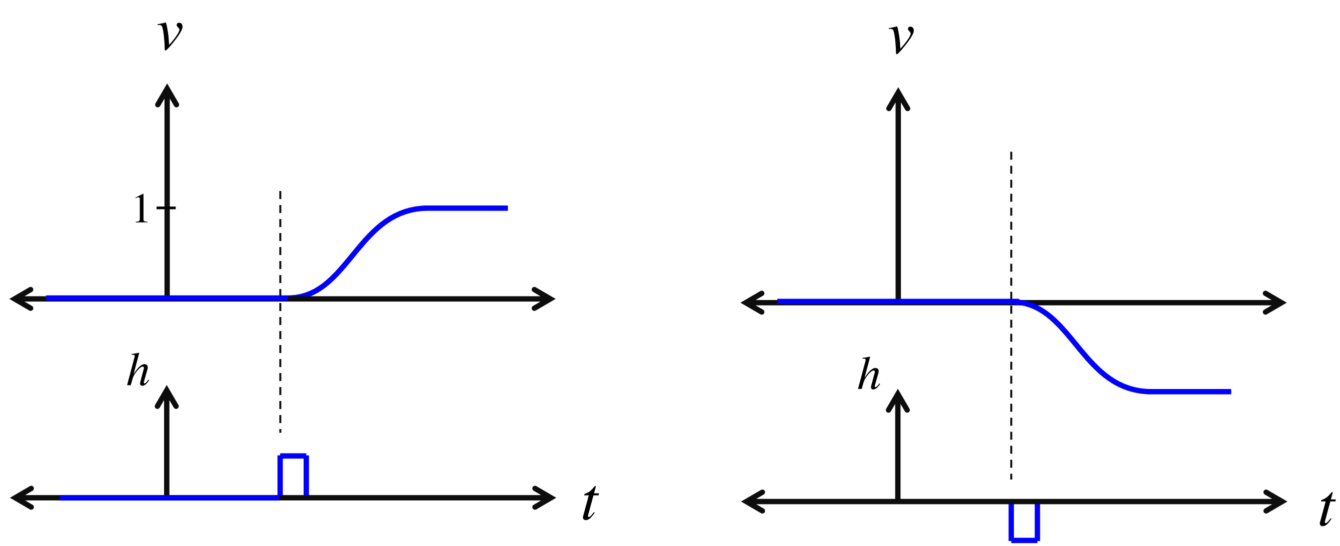


# Recurrent networks

- Add a saturating activation function $F(x)$

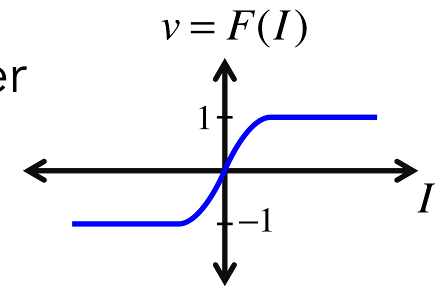


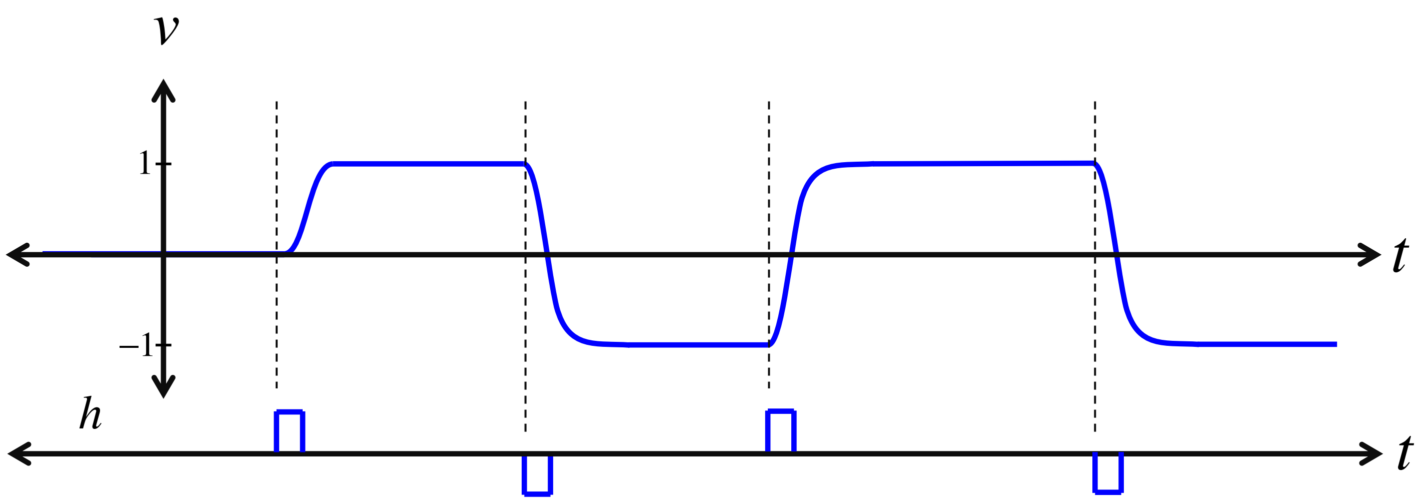



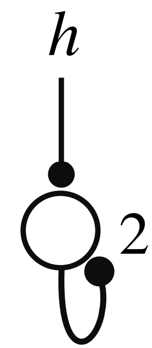



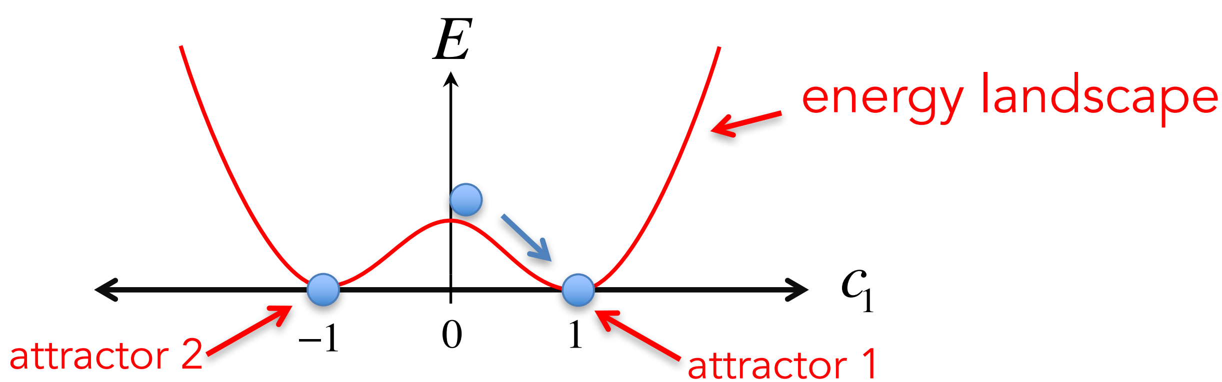
# Recurrent networks

- Saturating activation function plus eigenvalues greater than 1 lead to stable states other than zero!



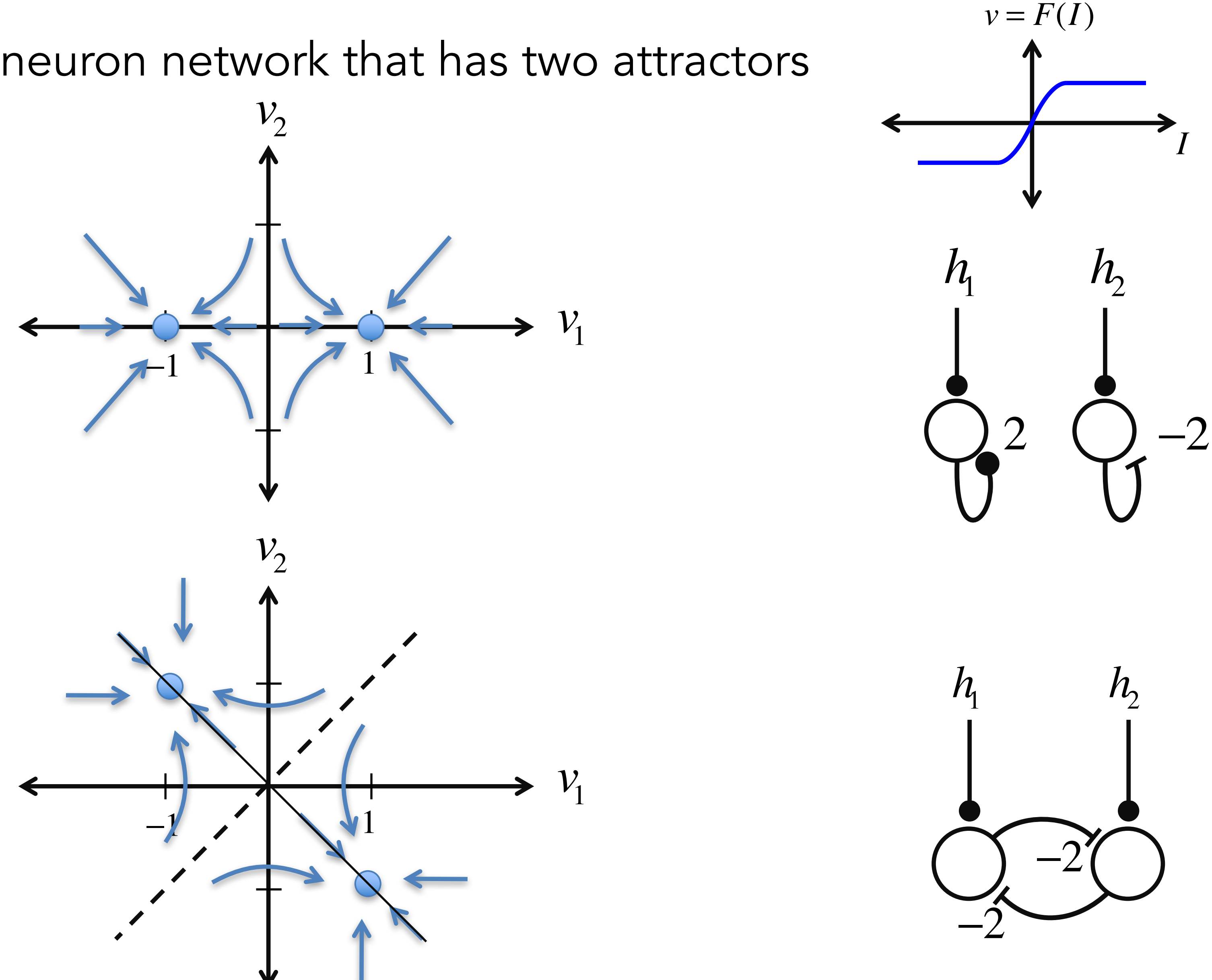



# Recurrent networks

- Two-neuron network that has two attractors



# Learning Objectives for Lecture 20

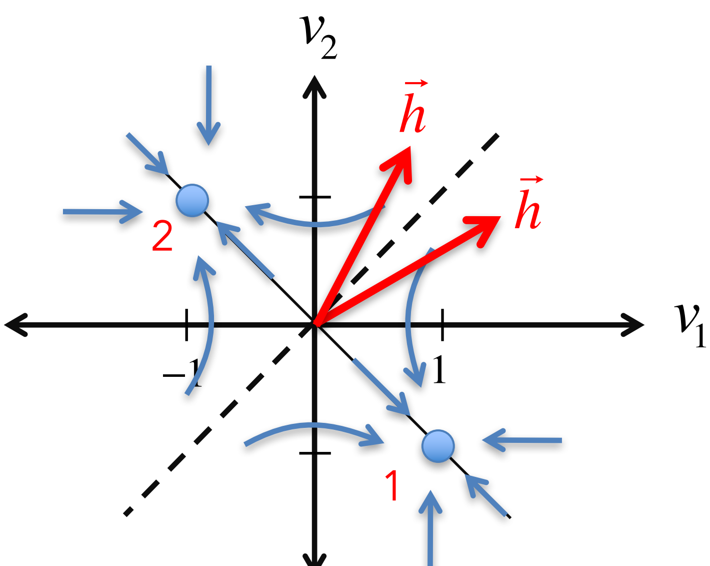
- Recurrent networks with lambda greater than one
  - Attractors
- Winner-take-all networks
- Attractor networks for long-term memory (Hopfield model)
- Energy landscape
- Hopfield network capacity

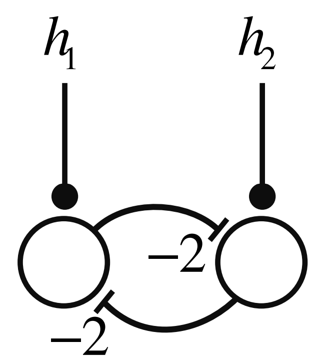
# Winner-take-all network

- Implements decision making

  Network state will move to attractor 1 if $h_1 > h_2$

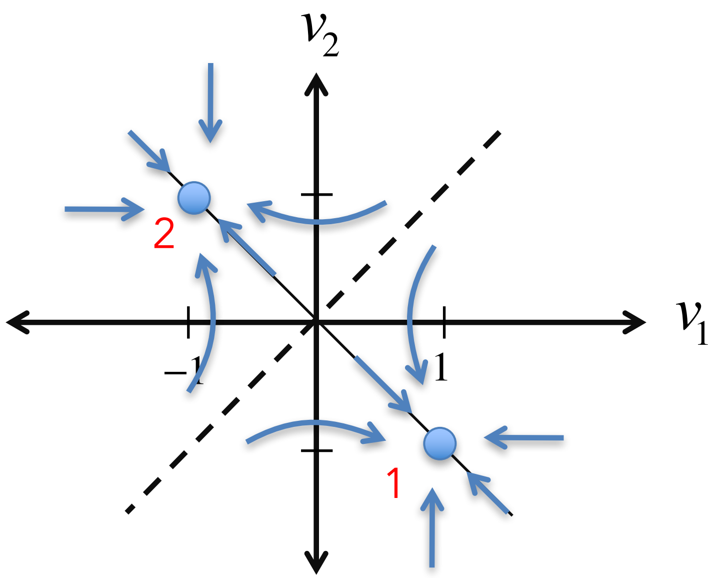
  Network state will move to attractor 2 if $h_2 > h_1$

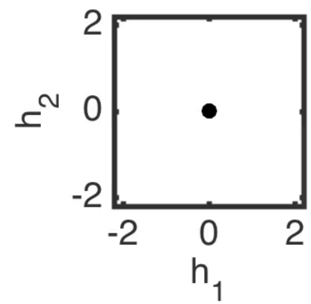


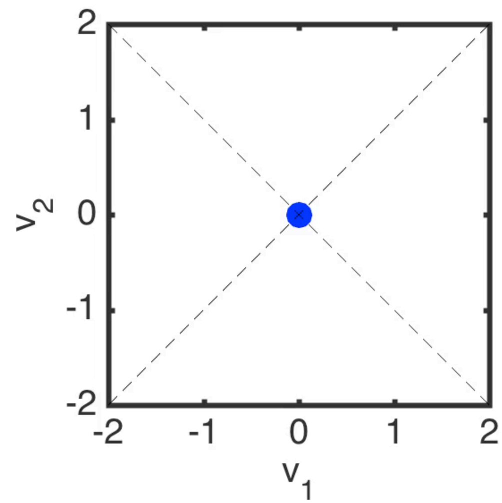



# Winner-take-all network

- Implements decision making



# Learning Objectives for Lecture 20

- Recurrent networks with lambda greater than one
  - Attractors
- Winner-take-all networks
- Attractor networks for long-term memory (Hopfield model)
- Energy landscape
- Hopfield network capacity

# Hippocampus

- The region CA3 of the hippocampus gets sensory inputs and forms a dense recurrently connected network

Figure removed due to copyright restrictions. Figure 5 in Mazzantini, M. "Functional Neural Differentiation of Adult Hippocampus Derived Stem Cells." Thesis submission, University College London, 2010.

See Lecture 20 video to view figures.

# Hippocampus

- Some neurons in CA3 represent ‘memories’ of locations in space (place cells).

Figures removed due to copyright restrictions. Sources unknown.

See Lecture 20 video to view figures.

# Hippocampus

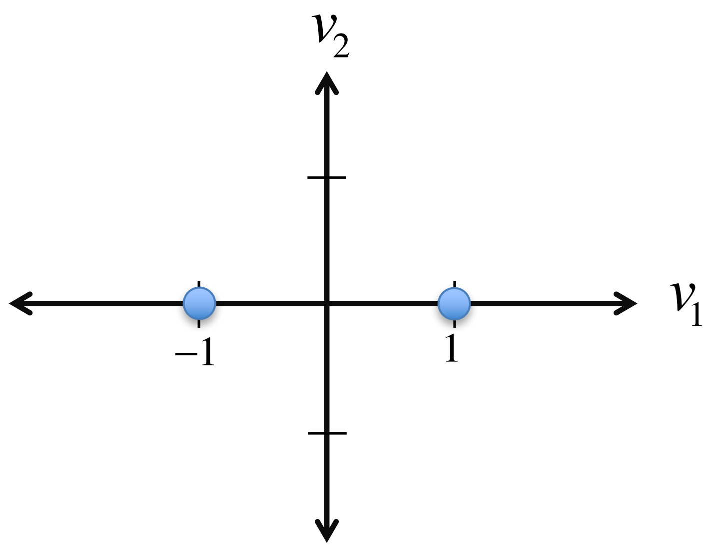
- Different neurons represent different remembered locations.

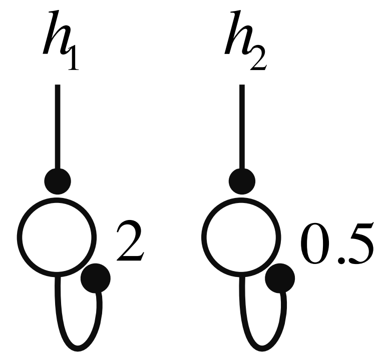
Figure removed due to copyright restrictions. Sources unknown.

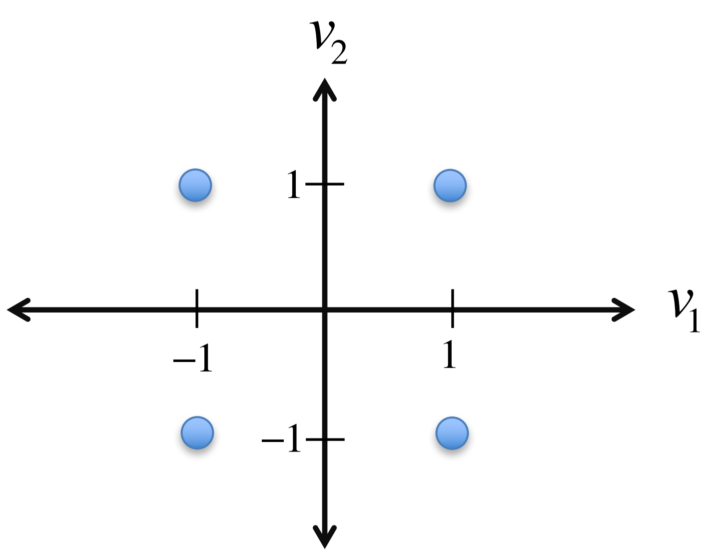
See Lecture 20 video to view figures.

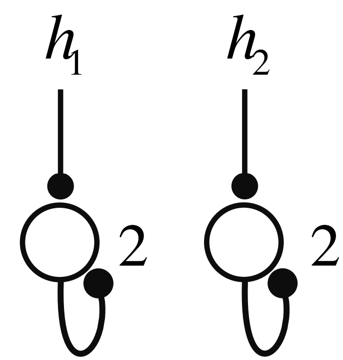
# Recurrent networks

- Networks with many attractors…



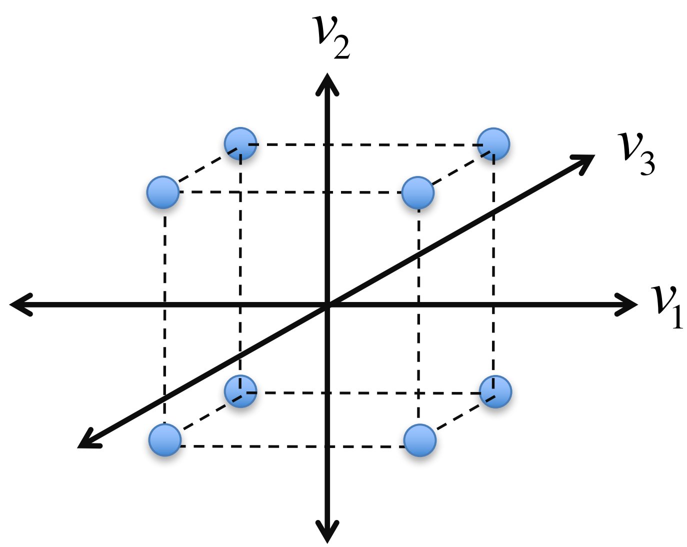



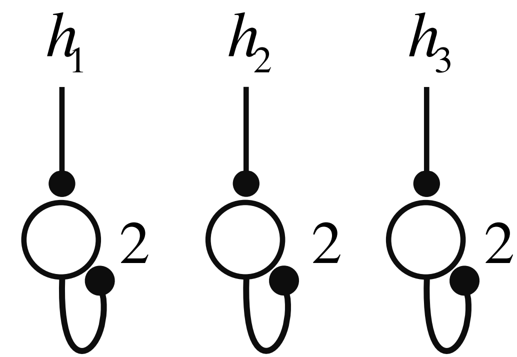



# Hopfield networks

- Networks with many attractors…

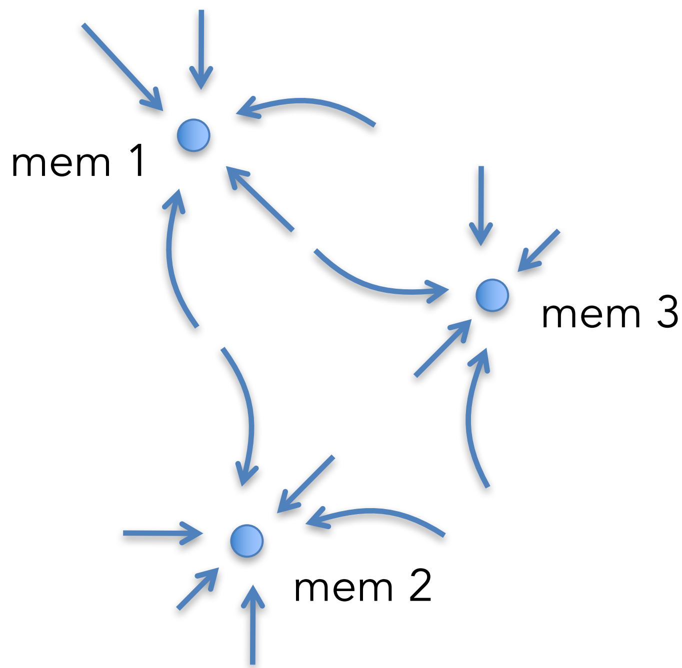


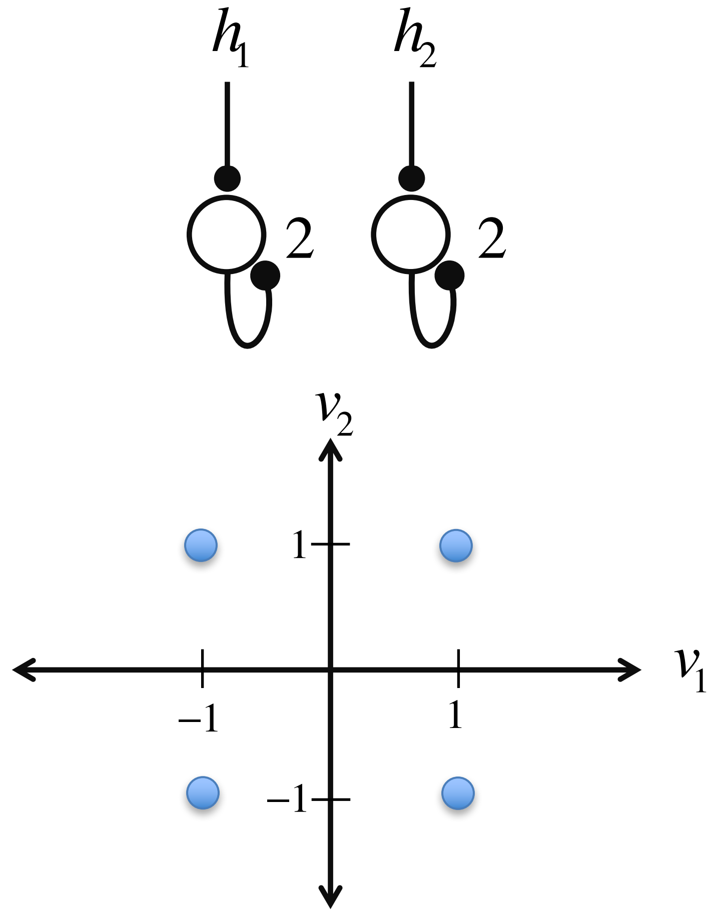




$2^n$ possible states !

# Hopfield networks

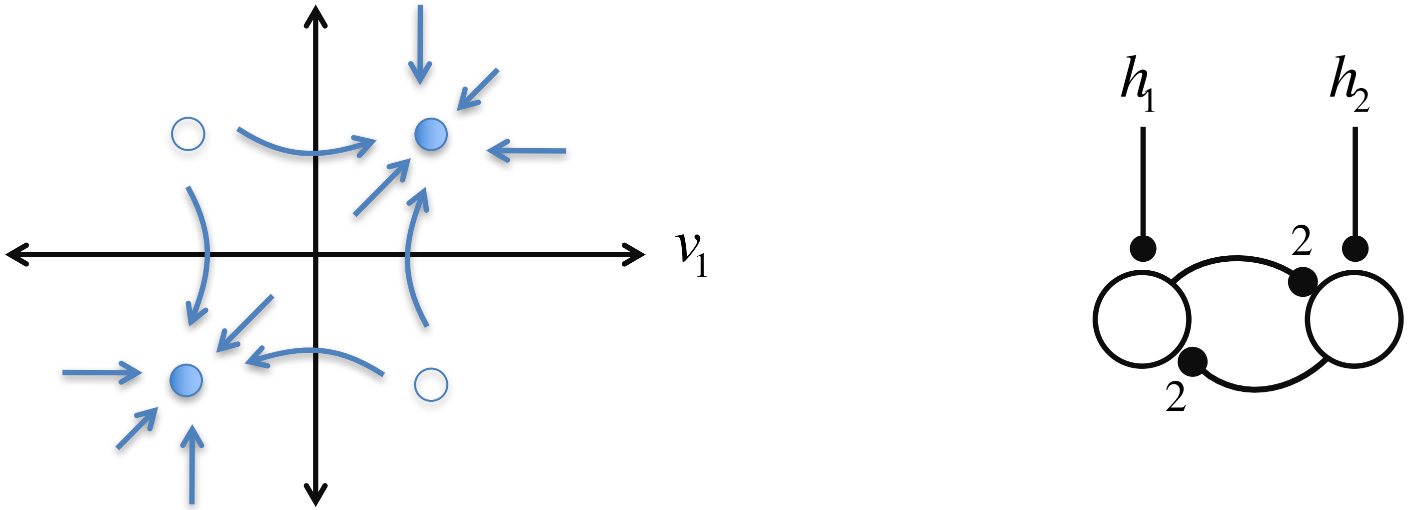
- We only want some of the possible states to be stable

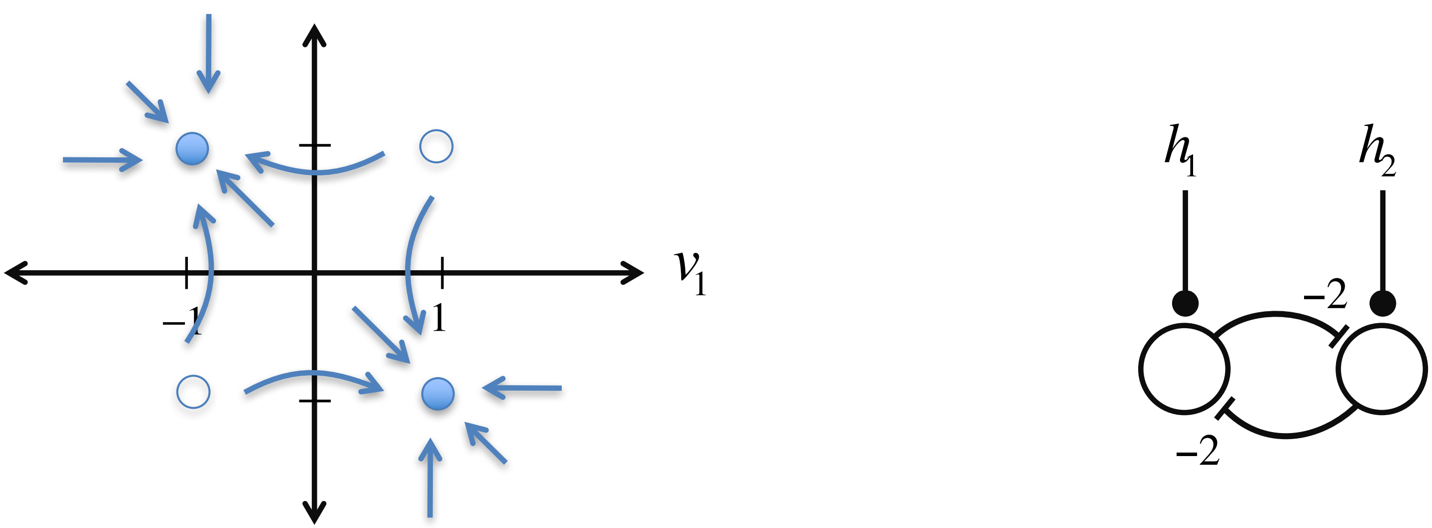


# Hopfield networks

- Networks with many attractors…



# Hopfield networks

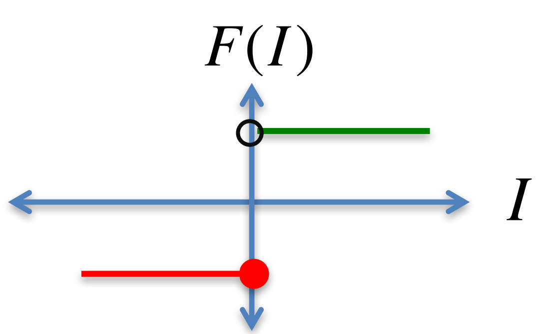
- We started with this dynamical equation

$$\tau_n \frac{d\vec{v}}{dt} = -\vec{v} + F\left[\vec{h} + M\,\vec{v}\right]$$

- We are going to simplify this as follows:

$$\vec{v}(t+1) = F\left[M\,\vec{v}(t)\right] \qquad v_i(t+1) = F\left[\sum_{j=1}^{N} M_{ij} v_j(t)\right]$$

where the neuronal activation function is



binary threshold neuron

$$F(x) = \mathrm{sgn}(x) = \begin{cases} 1 \; if \;\; x > 0 \\ -1 \; if \;\; x \le 0 \end{cases}$$

# Hopfield networks

- Our goal is to make a network that evolves so that it approaches any desired pattern $\vec{\xi}$

  Where $\xi_i$ is the activity (1 or -1) of the *i*th neuron.

- The condition for $\vec{\xi}$ to be a stable pattern is

$$v_i(t+1) = \operatorname{sgn}\left[\sum_{j=1}^{N} M_{ij}\xi_j\right] = \xi_i$$

# Hopfield networks

- Let's try the weight matrix $M_{ij} = \alpha \xi_i \xi_j$ *where* $\alpha > 0$

$$v_i(t+1) = \mathrm{sgn}\left[\sum_{j=1}^{N} M_{ij}\xi_j\right]$$

$$= \mathrm{sgn}\left[\sum_{j=1}^{N} (\alpha \xi_i \xi_j)\xi_j\right]$$

$$= \mathrm{sgn}\left[\alpha \xi_i \sum_{j=1}^{N} \xi_j \xi_j\right]$$

$$= \mathrm{sgn}\left[\alpha N \xi_i\right]$$

$$v_i(t+1) = \xi_i$$

NOTE: This is a symmetric matrix!

$$\boxed{M_{ij} = \frac{1}{N}\xi_i \xi_j} \; !$$

$$M = \frac{1}{N}\vec{\xi}\,\vec{\xi}^{T}$$

# Hopfield networks

- Let's take an example: Design a network of 3 neurons that remembers a pattern (1,1,-1).

$$\vec{\xi} = \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}$$

$$M_{ij} = \frac{1}{N}\vec{\xi}\,\vec{\xi}^{T} = \frac{1}{3}\begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}\begin{pmatrix} 1 & 1 & -1 \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 1 & 1 & -1 \\ 1 & 1 & -1 \\ -1 & -1 & 1 \end{pmatrix}$$

- Is the pattern (1,1,-1) a stable state?

$$v(t+1) = \operatorname{sgn}\left[\frac{1}{3}\begin{pmatrix} 1 & 1 & -1 \\ 1 & 1 & -1 \\ -1 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}\right] = \operatorname{sgn}\left[\frac{1}{3}\begin{pmatrix} 3 \\ 3 \\ -3 \end{pmatrix}\right] = \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}$$

Yes!

# Hopfield networks

- Does our network have an 'attractor' at the pattern (1,1,-1)?
  - Let's start the network at a different state and see what happens…

$$\vec{v}_0 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$v(t+1) = \operatorname{sgn}\left[\frac{1}{3}\begin{pmatrix} 1 & 1 & -1 \\ 1 & 1 & -1 \\ -1 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}\right] = \operatorname{sgn}\left[\frac{1}{3}\begin{pmatrix} 3 \\ 3 \\ -3 \end{pmatrix}\right] = \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}$$

- The network evolves toward the 'attractor' state (1,1,-1) !

# Hopfield networks

- Let's prove that $\vec{\xi}$ is an attractor of the network $M_{ij} = \frac{1}{N}\vec{\xi}\,\vec{\xi}^{T}$

Calculate total input onto the i$^{\text{th}}$ neuron starting in arbitrary state $\vec{v}$

$$k_i = \sum_{j=1}^{N} M_{ij} v_j$$

where $v_j$ is the firing rate of the j$^{\text{th}}$ neuron $\vec{k} = M\,\vec{v}$

$$k_i = \sum_{j=1}^{N}\left(\frac{1}{N}\xi_i\,\xi_j\right)v_j = \frac{1}{N}\xi_i \underbrace{\sum_{j=1}^{N}\xi_j v_j}$$

$$\sum_{j=correct}\xi_j v_j + \sum_{j=incorrect}\xi_j v_j$$

$$k_i = \frac{1}{N}\xi_i\left(N_{correct} - N_{incorrect}\right)$$

$$v_i(t+1) = \mathrm{sgn}\left[k_i\right] = \xi_i$$

The total input has the correct sign **if** the majority of the neurons have the correct state!

# Learning Objectives for Lecture 20

- Recurrent networks with lambda greater than one
  - Attractors
- Winner-take-all networks
- Attractor networks for long-term memory (Hopfield model)
- Energy landscape
- Hopfield network capacity

# The Energy Function

$$k_i = \sum_{j=1}^{N} M_{ij} v_j \qquad \vec{k} = M \vec{v}$$

- Each possible state of the network has an energy given by:

$$H = -\frac{1}{2} \vec{v} \cdot \vec{k} \qquad H = -\frac{1}{2} \vec{v}^T M \vec{v}$$

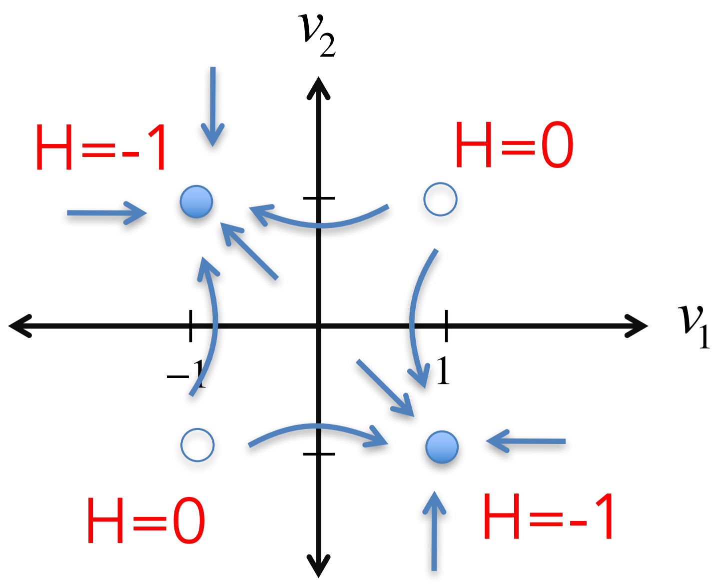
This is just the overlap of the current state of the network with the pattern of inputs to all the neurons!

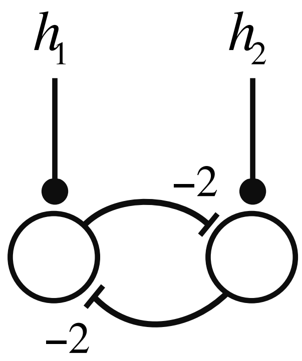
The energy is lowest when current state has high overlap with the synaptic drive to the next state ---> in an attractor

# The Energy Function

- Each possible state of the network has an energy given by:

$$H = -\frac{1}{2}\,\vec{v}^T M\,\vec{v} \qquad M = \begin{pmatrix} 0 & -2 \\ -2 & 0 \end{pmatrix}$$

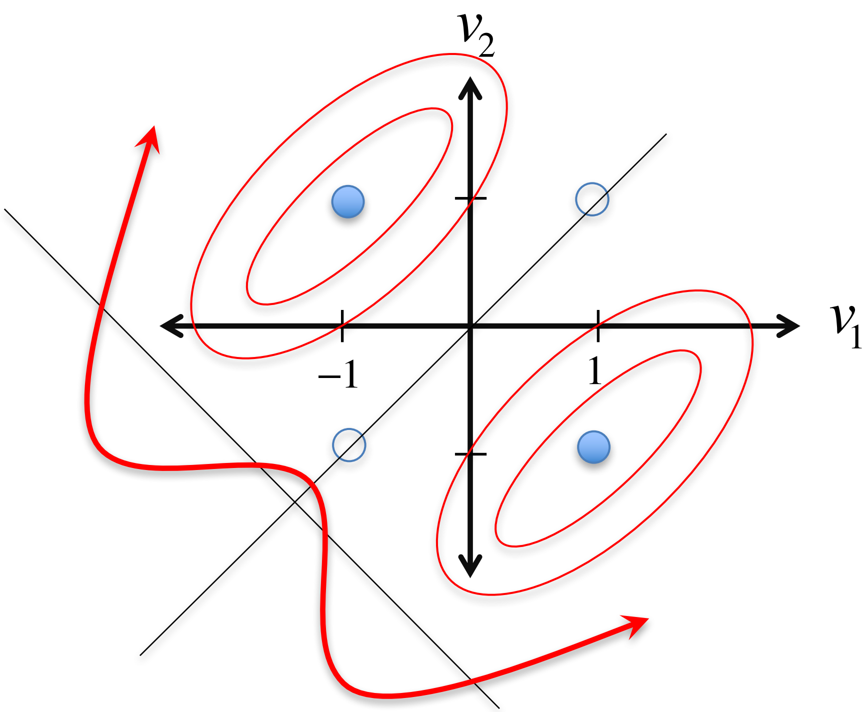


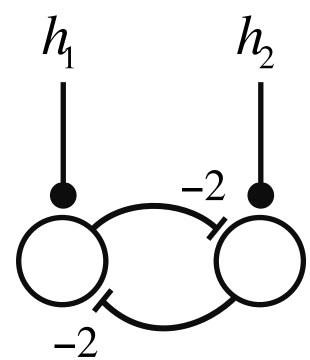



# The Energy Function

- Each possible state of the network has an energy given by:

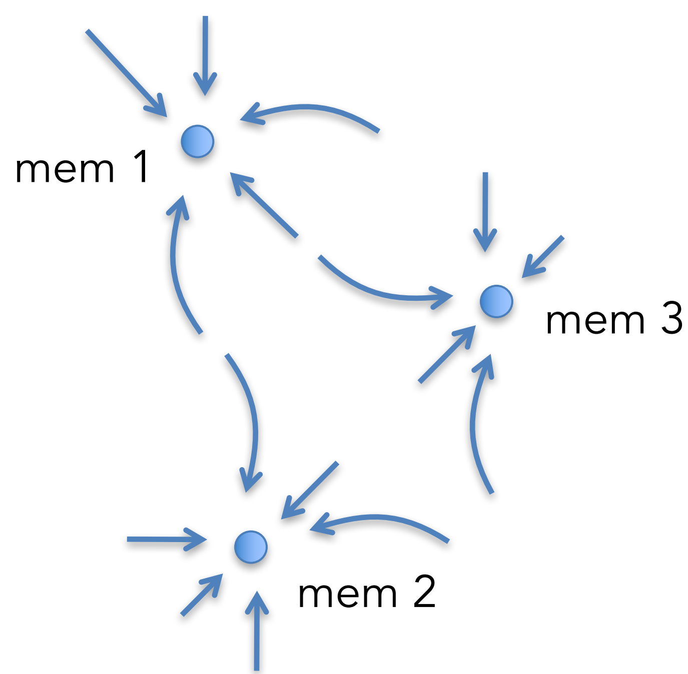
$$H = -\frac{1}{2}\,\vec{v}^T M\,\vec{v} \qquad M = \begin{pmatrix} 0 & -2 \\ -2 & 0 \end{pmatrix}$$



# Hopfield networks

- A Hopfield network can 'reconstruct' a memory or pattern from a partial pattern
- It will evolve 'downhill' toward whichever memory is closest to the input pattern (content addressable memory)



# Hopfield networks can be used to store a memory of an image

- 'Content addressable memory'

Video: splitfoot99. "Image Recognition with Hopfield Net."
May 10, 2009. YouTube.

# Learning Objectives for Lecture 20

- Recurrent networks with lambda greater than one
  - Attractors
- Winner-take-all networks
- Attractor networks for long-term memory (Hopfield model)
- Energy landscape
- Hopfield network capacity

# Multiple memories

- We want our network to remember P different patterns $\vec{\xi}^{\mu}$

$$\mu = 0, 1, 2 \ldots, P - 1$$

- We compute the contribution to the weight matrix from each pattern…

$$M_{ij}^{\mu} = \frac{1}{N} \xi_i^{\mu} \xi_j^{\mu}$$

and add them up!

$$M_{ij} = \frac{1}{N} \sum_{\mu=0}^{P-1} \xi_i^{\mu} \xi_j^{\mu}$$

- The network state will evolve to the attractor that is 'closest' to the initial state (that with the biggest overlap).

# Multiple memories

Video: macheads202. "Hopfield Networks." Feb. 28, 2016. YouTube.

Video: Aaron Vose. "An Interactive Hopfield Neural Network Restoring Corrupted Bitmaps." Mar. 18, 2013. YouTube.

# What is the capacity?

- Do the same stability analysis we did before...

For a pattern $\vec{\xi}^{\,0}$ to be an attractor, we want

$$v_i(t+1) = \operatorname{sgn}\left(\sum_j M_{ij}\xi_j^0\right) = \xi_i^0 \qquad M_{ij} = \frac{1}{N}\sum_\mu \xi_i^\mu \xi_j^\mu$$

- Let's plug in our weight matrix and see what we get…

$$v_i(t+1) = \operatorname{sgn}\left(\sum_j \left[\frac{1}{N}\sum_\mu \xi_i^\mu \xi_j^\mu\right]\xi_j^0\right) = ??$$

# What is the capacity?

$$v_i(t+1) = \mathrm{sgn}\left(\sum_j \left[\frac{1}{N}\sum_\mu \xi_i^\mu \xi_j^\mu\right]\xi_j^0\right) = ??$$

- Rearrange…

$$= \mathrm{sgn}\left(\frac{1}{N}\sum_\mu \xi_i^\mu \sum_j \xi_j^\mu \xi_j^0\right)$$

- Separate terms where $\mu = 0$ and $\mu \neq 0$

$$= \mathrm{sgn}\left(\frac{1}{N}\xi_i^0 \underbrace{\sum_j \xi_j^0 \xi_j^0}_{N} + \frac{1}{N}\sum_{\mu\neq 0} \xi_i^\mu \sum_j \xi_j^\mu \xi_j^0\right)$$

# What is the capacity?

$$\mathrm{sgn}\left(\sum_j M_{ij}\xi_j^0\right) = \xi_i^0 \quad ???$$

$$= \mathrm{sgn}\left(\xi_i^0 + \underbrace{\frac{1}{N}\sum_{\mu\neq 0}\xi_i^\mu \underbrace{\sum_j \xi_j^\mu \xi_j^0}_{\vec{\xi}^{\,\mu}\cdot\vec{\xi}^{\,0}}}\right)$$

Cross-talk between our pattern $\vec{\xi}^{\,0}$ and all the other memories depends on how much overlap there is !

# What is the capacity?

$$v_i(t+1) = \operatorname{sgn}\left(\xi_i^0 \;+\; \frac{1}{N}\sum_{\mu\neq 0}\xi_i^\mu\left(\vec{\xi}^{\,\mu}\cdot\vec{\xi}^{\,0}\right)\right)$$

- You can see that if all the memories are orthogonal, then all are stable attractors.
- But if one of the memories (e.g. $\vec{\xi}^{\,1}$ ) is close to $\vec{\xi}^{\,0}$ then

$$\vec{\xi}^{\,0}\cdot\vec{\xi}^{\,1} \approx N$$

- And…

$$v_i(t+1) = \operatorname{sgn}\left(\xi_i^0 \;+\; \xi_i^1\right) \neq \xi_i^0$$

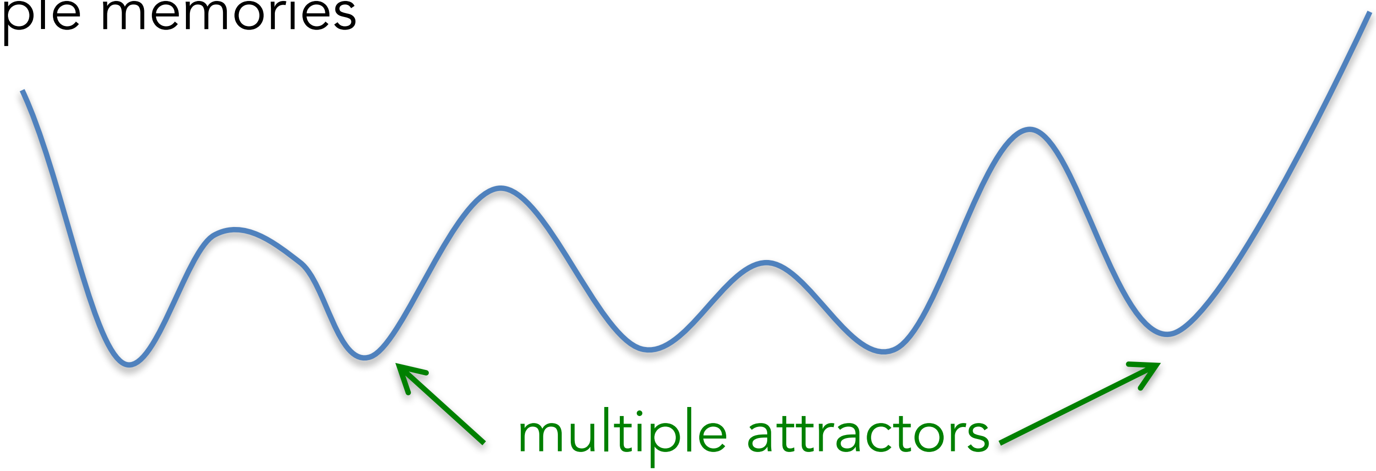
# What is the capacity?

- Memories don't need to be orthogonal, as long as the cross-talk term is not large enough to change the sign of the inputs, the memories will not have any errors.

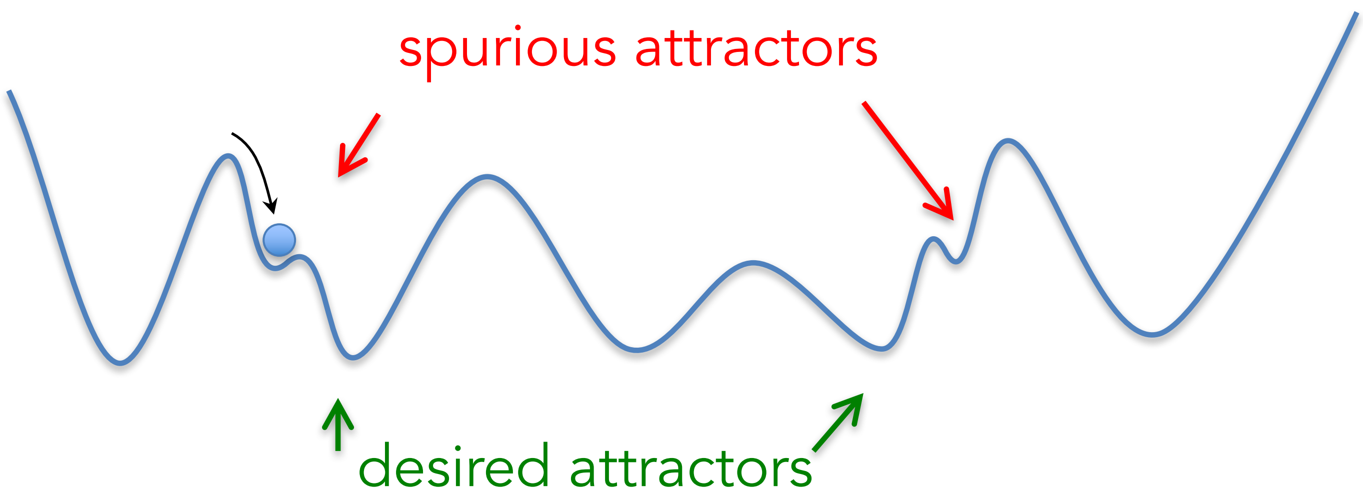
- For random values of $\xi_i$, a Hopfield network can store up to 0.15N memories and still have a very small probability ($p<0.01$) of having an error in one neuron.

# What is the capacity?

- Multiple memories

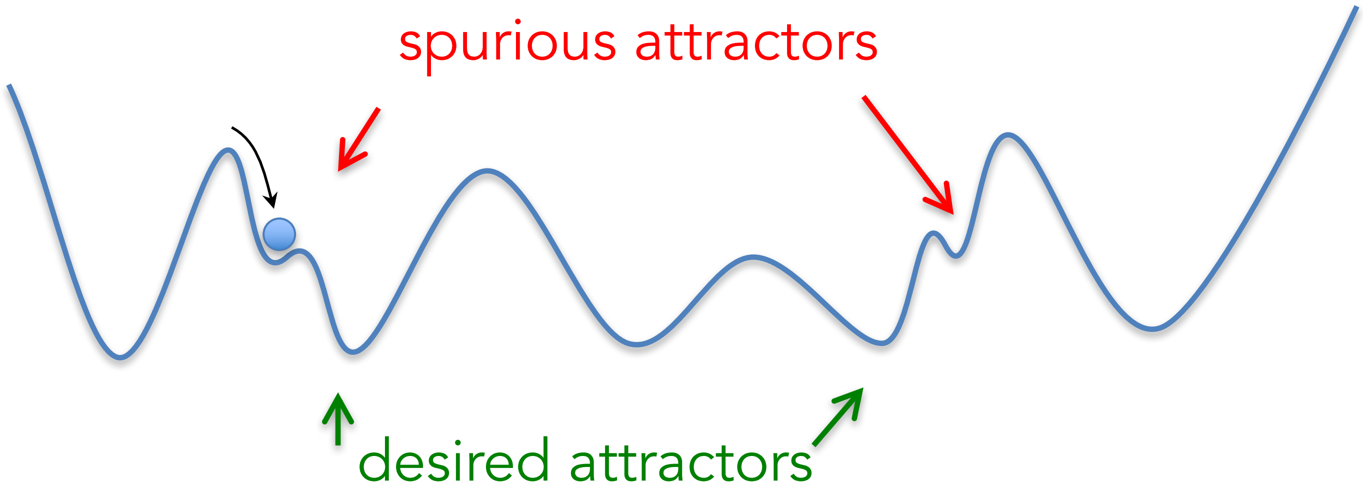



- Too many memories…



# What is the capacity?

- Too many memories…



# Learning Objectives for Lecture 20

- Recurrent networks with lambda greater than one
  - Attractors

- Winner-take-all networks

- Attractor networks for long-term memory (Hopfield model)

- Energy landscape

- Hopfield network capacity

MIT OpenCourseWare
https://ocw.mit.edu/

9.40 Introduction to Neural Computation
Spring 2018

For information about citing these materials or our Terms of Use, visit: https://ocw.mit.edu/terms.