Leren in netwerkmodellen 1a

Fleur Zeldenrust
Leren & Geheugen, 2017

Tentamenstof

Doel:

van Perceptie tot Bewustzijn: Hoe kan je iets (een wortel) herkennen met een netwerk van neuronen?

Leren & geheugen: Hoe kan een netwerk *leren* iets (een wortel) te herkennen?

Je kunt de eigenschappen van verschillende modellen van leren in neurale netwerken beschrijven en de modellen uitleggen.

Tentamenstof

Doel: Je kunt de eigenschappen van verschillende modellen van leren in neurale netwerken beschrijven en de modellen uitleggen.

1. Kandel et al.: Principles of Neural Science. McGraw-Hill, appendix E&F

2. Gerstner et al.: Neuronal Dynamics (online), hoofdstuk 19

3. O'Reilly et al.: Computational cognitive neuroscience (online), hoofdstuk 7

4. Slides (blackboard)

NB Op blackboard staat overzicht (welke onderwerpen in welke tekst)

Focus bij Kandel:

- Nadruk op leren, veranderen gewichten:
    - (Hebbian) plasticity
    - box E2 & box E3
- Rest bekend verondersteld uit Perceptie

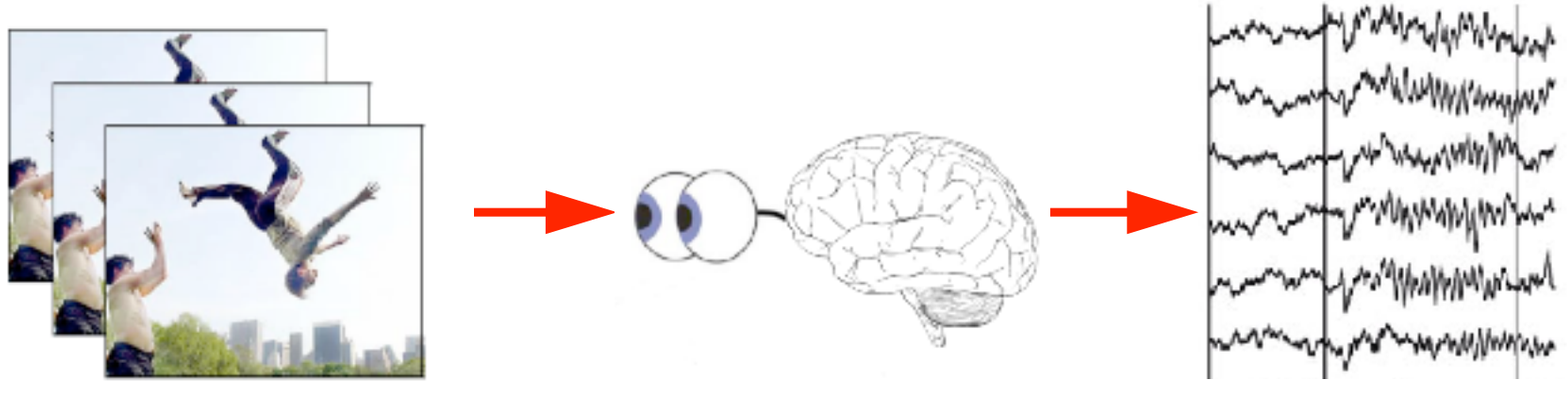
Herhaling Perceptie

Er is een relatie tussen de wereld, de activiteit van onze hersenen, ons gedrag en wat we waarneming noemen.

Wat is deze relatie? Neural coding, neurale netwerken

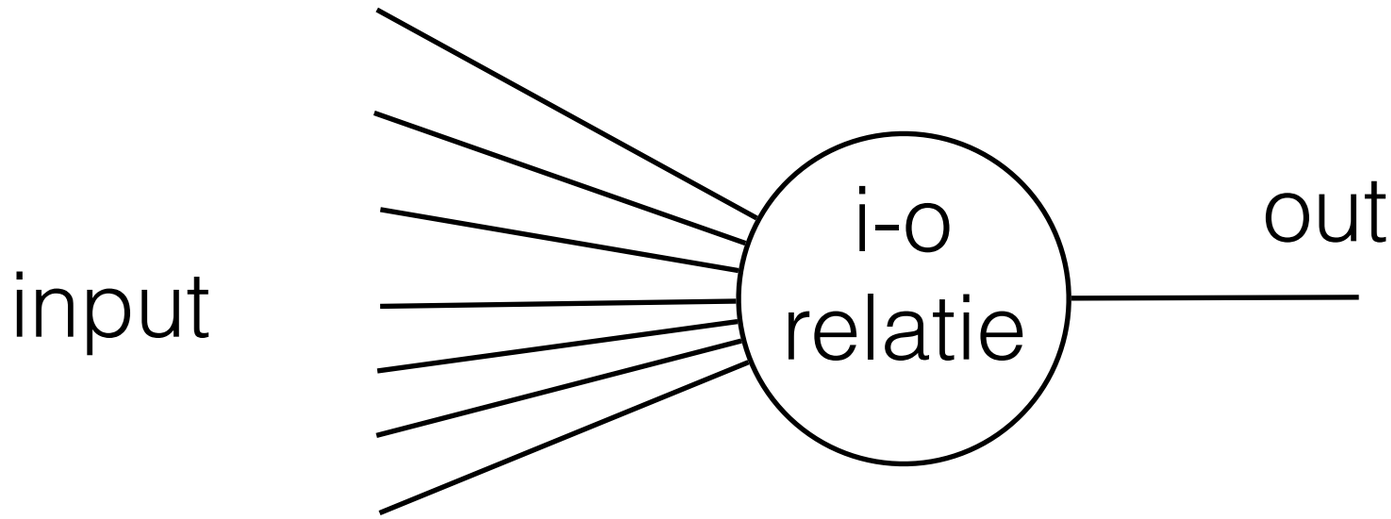
Representaties: hoe wordt wat we waarnemen weergegeven in hersenactiviteit?

Leren: hoe kan een netwerk dat leren?

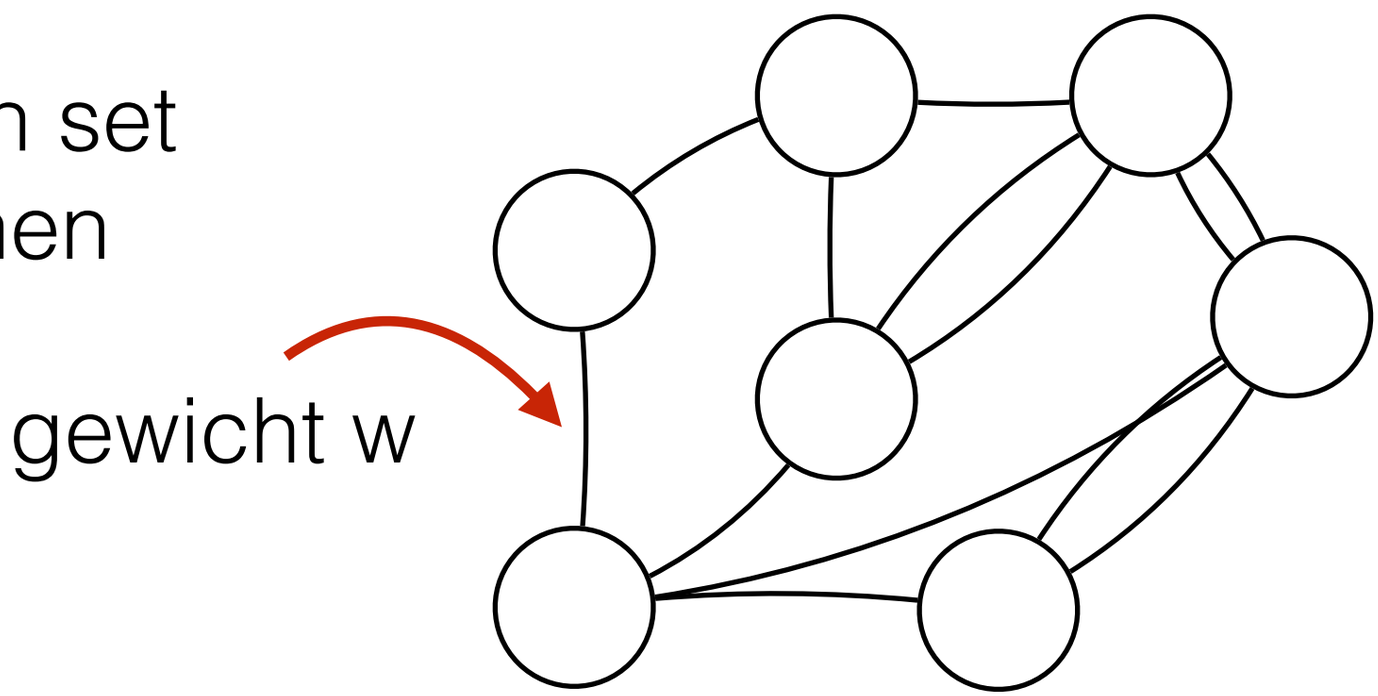


Herhaling Perceptie

- Neuron integreert input, zet om naar output

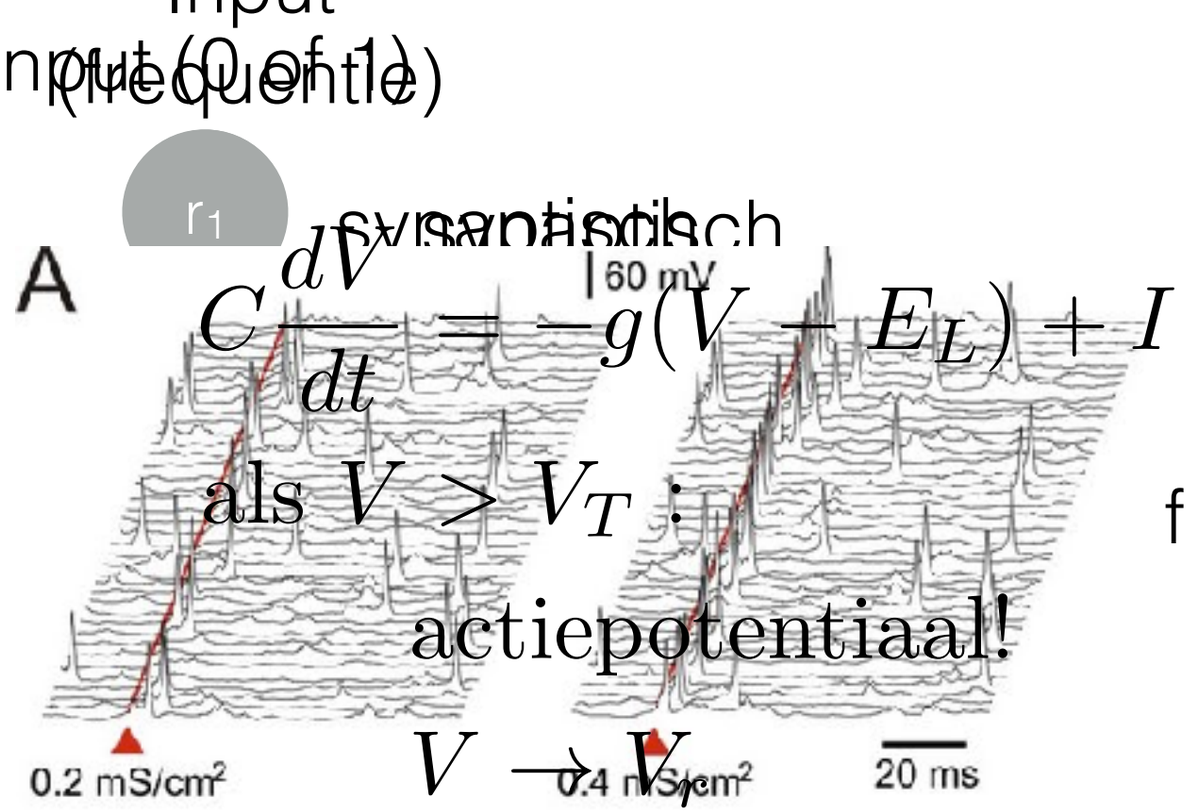


- Een netwerk is een set verbonden neuronen



Neuron-modellen

- Binair neuron
- Rate neuron
- Leaky Integrate-and-Fire (LIF)
  - high-conductance state

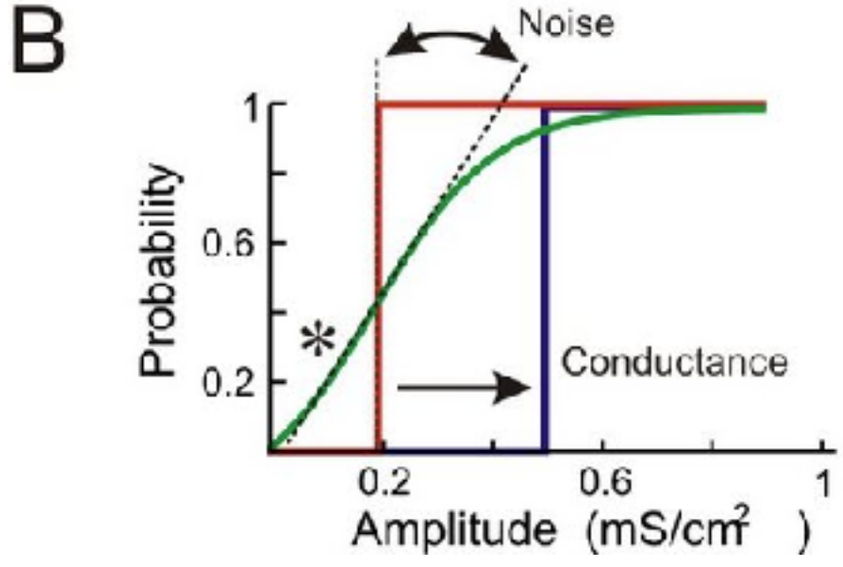


$$C\frac{dV}{dt} = -g(V - E_L) + I$$

als $V > V_T$:

actiepotentiaal!

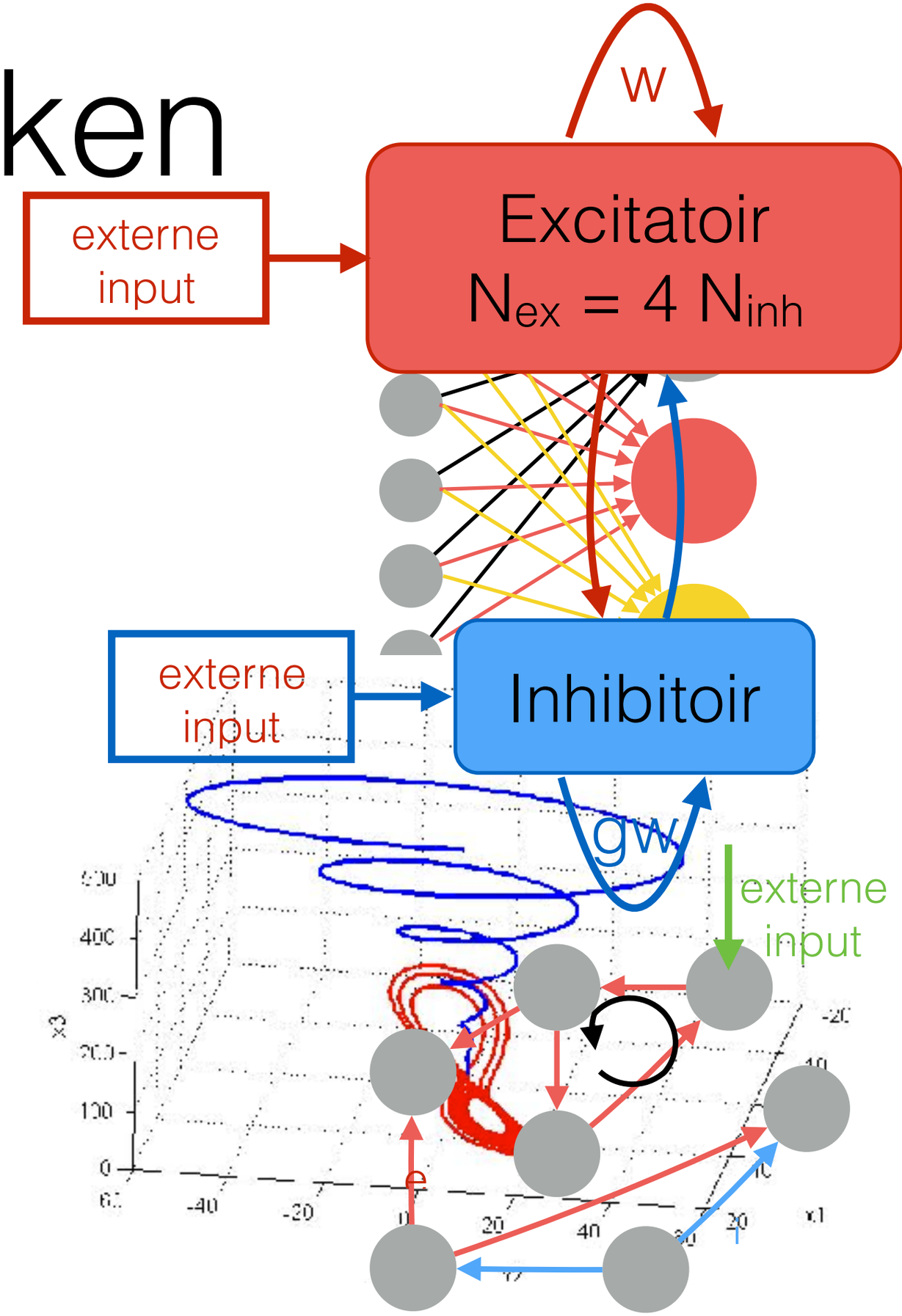
$V \rightarrow V_r$



Netwerken

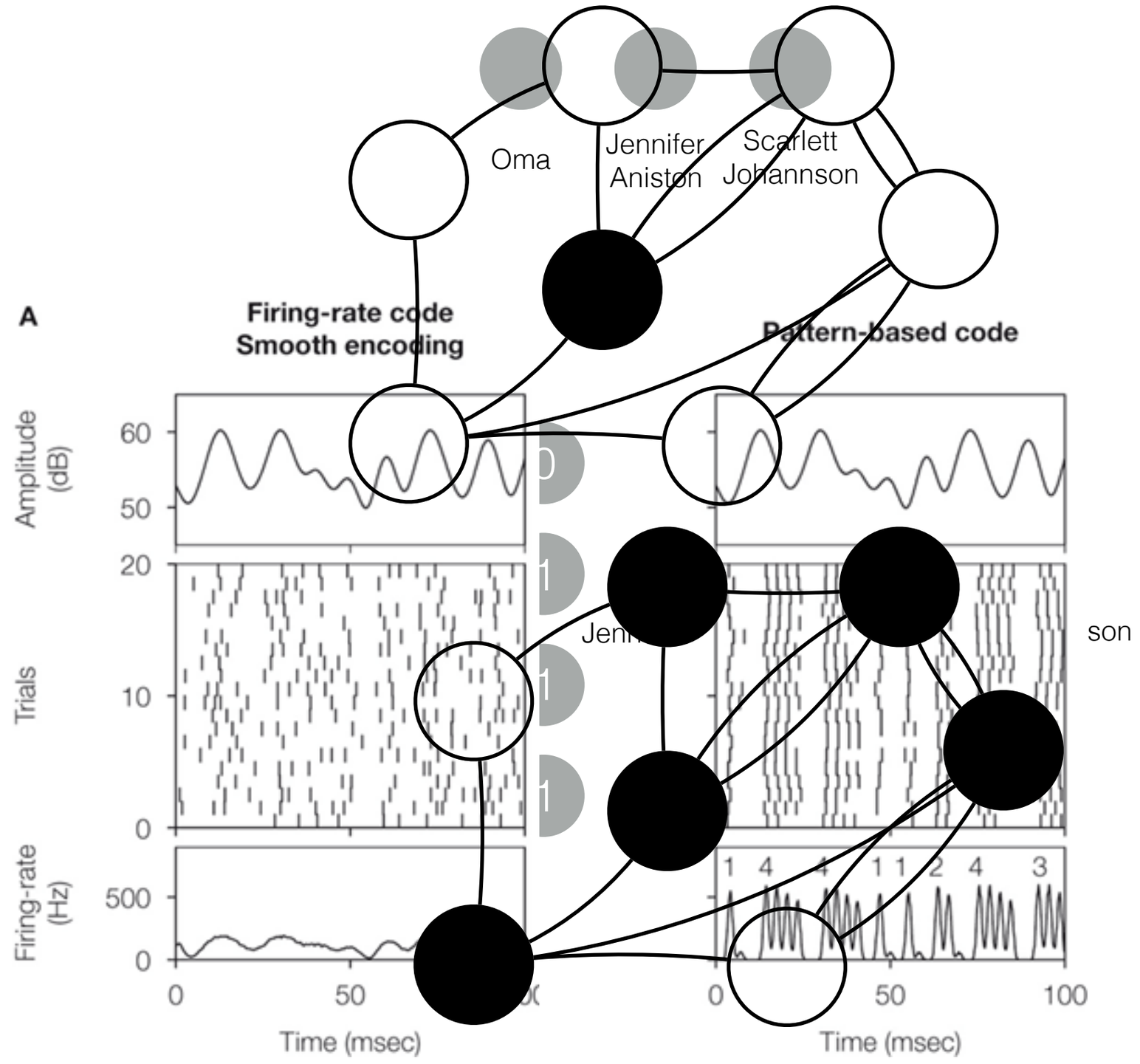
- Feedforward netwerken
  - Perceptron

- Recurrente netwerken
  - Hopfield netwerk
  - Attractor netwerk
    - point attractor
    - line attractor
    - cyclic attractor
    - chaotic attractor
  - Balanced networks
    - selective amplification



Coding

- local
- distributed
  - sparse
  - dense
- rate
- temporal



Deze twee colleges

- We weten nu dat we patronen kunnen opslaan in de neurale activiteit van kunstmatige netwerken

- Maar hoe kunnen deze netwerken leren de juiste output bij de juiste input te geven?

Onderwerp van dit college:

- Hoe kunnen kunstmatige netwerken ‘leren’?

- Dit geeft modellen van leren: verklaring van hoe het zou kunnen gaan in de hersenen

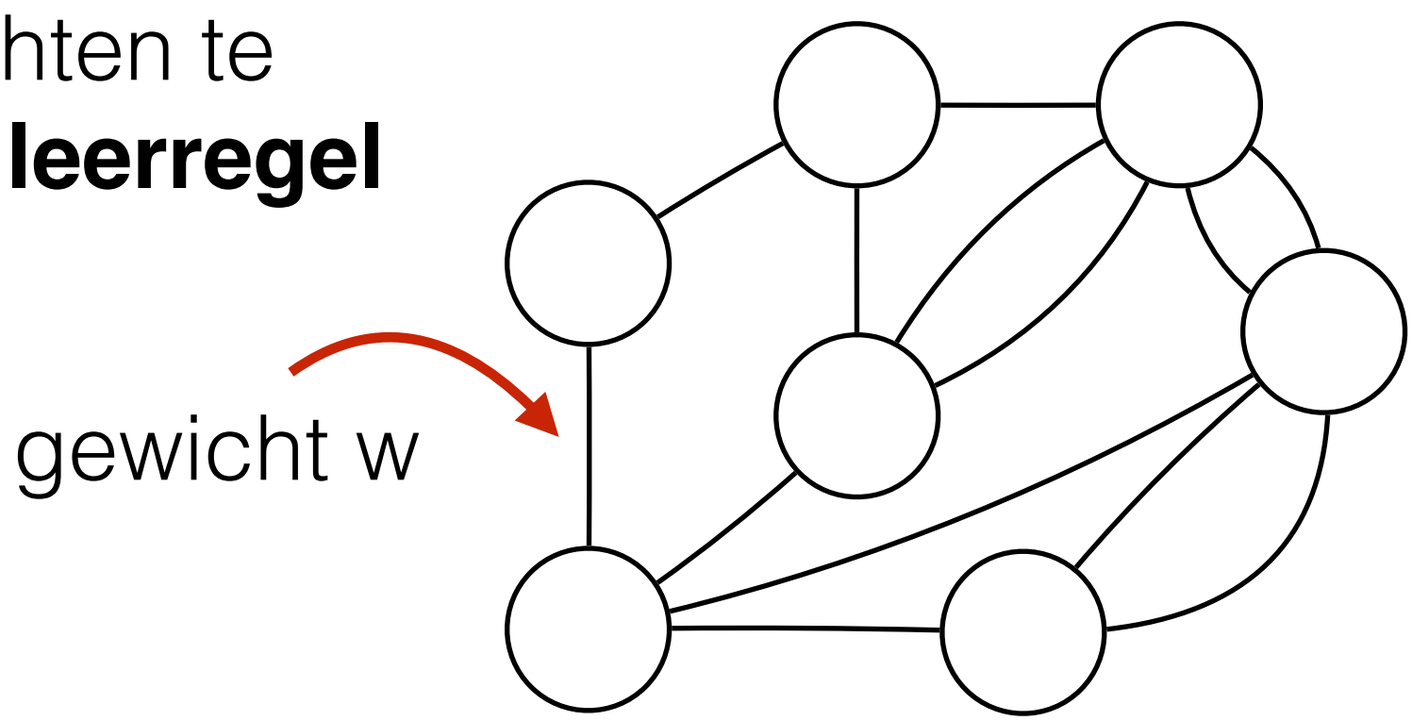
- Maar ook: zo kunnen we computers laten leren (Machine Learning), bv computer vision, spraakherkenning, woorden voorspellen (swiftkey)

Leren in neurale netwerken

- Aanname: netwerken leren door veranderingen in gewichten (synapsen)

- Dus vraag is: hoe verander ik gewichten zo dat mijn netwerk de correcte output bij de input geeft?

- Een recept om gewichten te veranderen heet een **leerregel**



Soorten leren

We kunnen 3 soorten leren onderscheiden:

1. **Unsupervised learning**: netwerk leert op basis van de input alleen (bijvoorbeeld receptive field)

2. **Supervised learning**: er is een leraar die vertelt hoe het netwerk het fout had (veel gebruikt in AI, misschien in cerebellum?)

3. **Reinforcement learning**: er is een (eventueel vertraagde) beloning of straf (bijvoorbeeld leren fietsen)

Programma

Vandaag

1. **Unsupervised learning**
2. **Supervised learning**

College 2

3. **Reinforcement learning**

Programma

Vandaag

1. **Unsupervised learning**
    - LTP/LTD modellen: rate-based Hebbian learning
        - voorbeeld: Leren in Hopfield netwerk
        - Andere ‘Hebb-achtige’ regels
        - STDP: spike-based Hebbian learning
2. **Supervised learning**

College 2

3. **Reinforcement learning**

Programma

Vandaag

1. **Unsupervised learning**
    - LTP/LTD modellen: rate-based Hebbian learning
        - voorbeeld: Leren in Hopfield netwerk
        - Andere ‘Hebb-achtige’ regels
        - STDP: spike-based Hebbian learning
2. **Supervised learning**

College 2

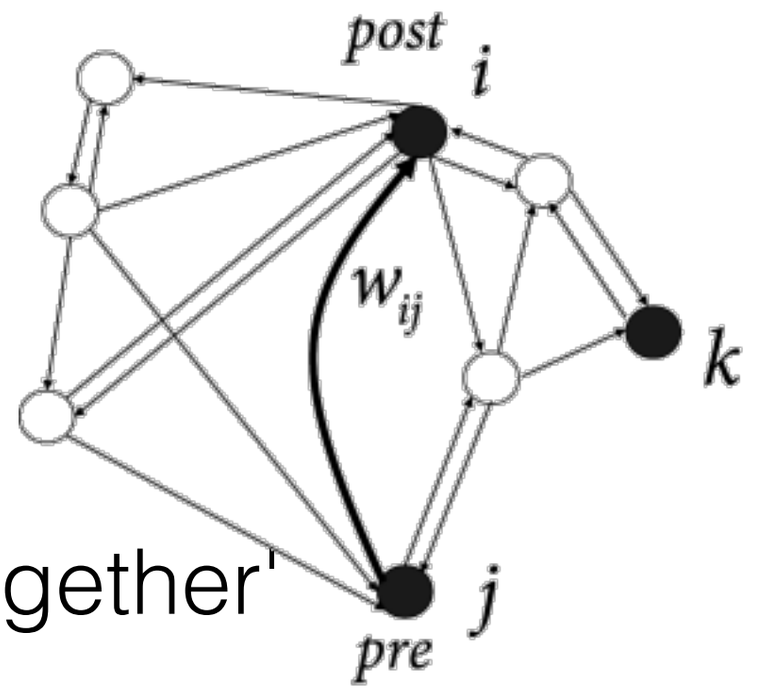
3. **Reinforcement learning**

Hebb rule

"When an axon of cell A is near enough to excite 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.”

–Donald Hebb, 1949

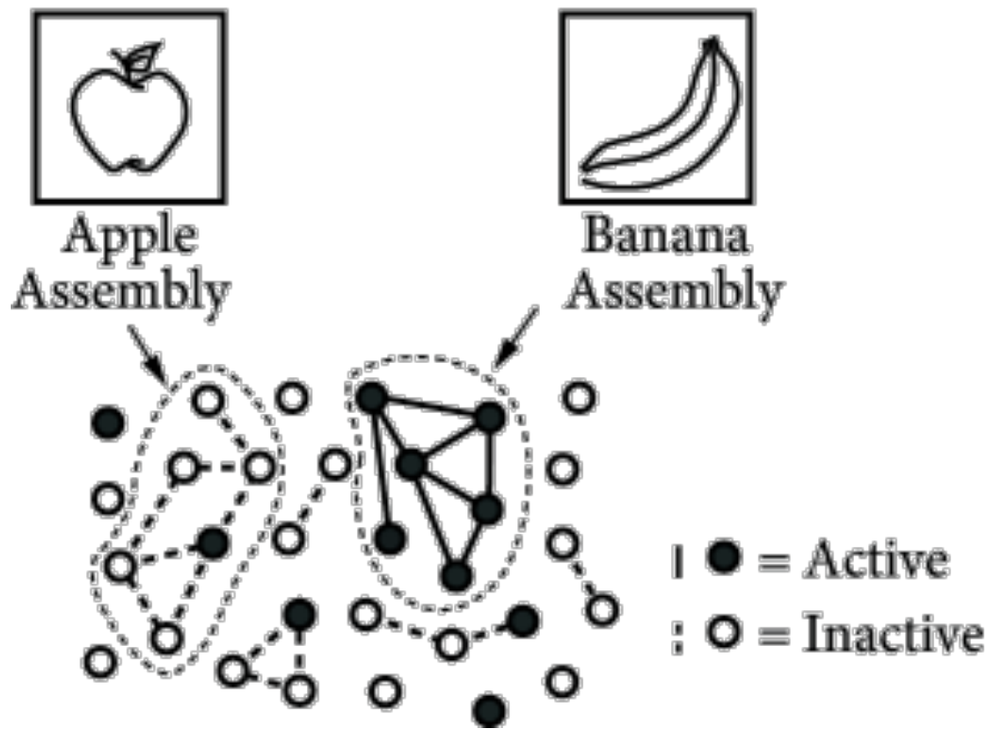
Hebb rule



‘Neurons that fire together wire together'

Cell Assemblies:
Relaties die je veel ziet worden versterkt en vormen ‘groepjes'

Pattern completion:
als een gedeelte van de ‘banaan' actief wordt, kan het netwerk zelf de andere neuronen activeren



Hebbian learning

Er is niet één 'Hebb regel', het zijn er heel veel!

Maar altijd: verandering synapssterkte hangt alleen af van activiteit pre- en postsynaptische neuron, dus

- lokaal
- correlatie, covariantie

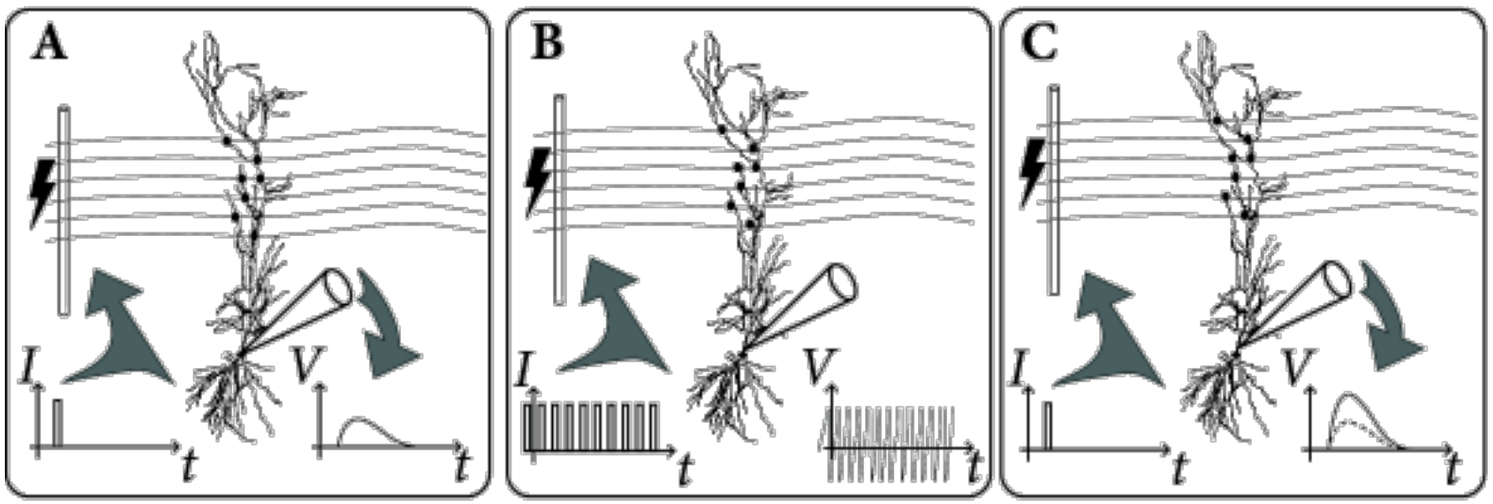
Anti-Hebbian learning: als een neuron een cel niet activeert - verzwak synaps (LTD)

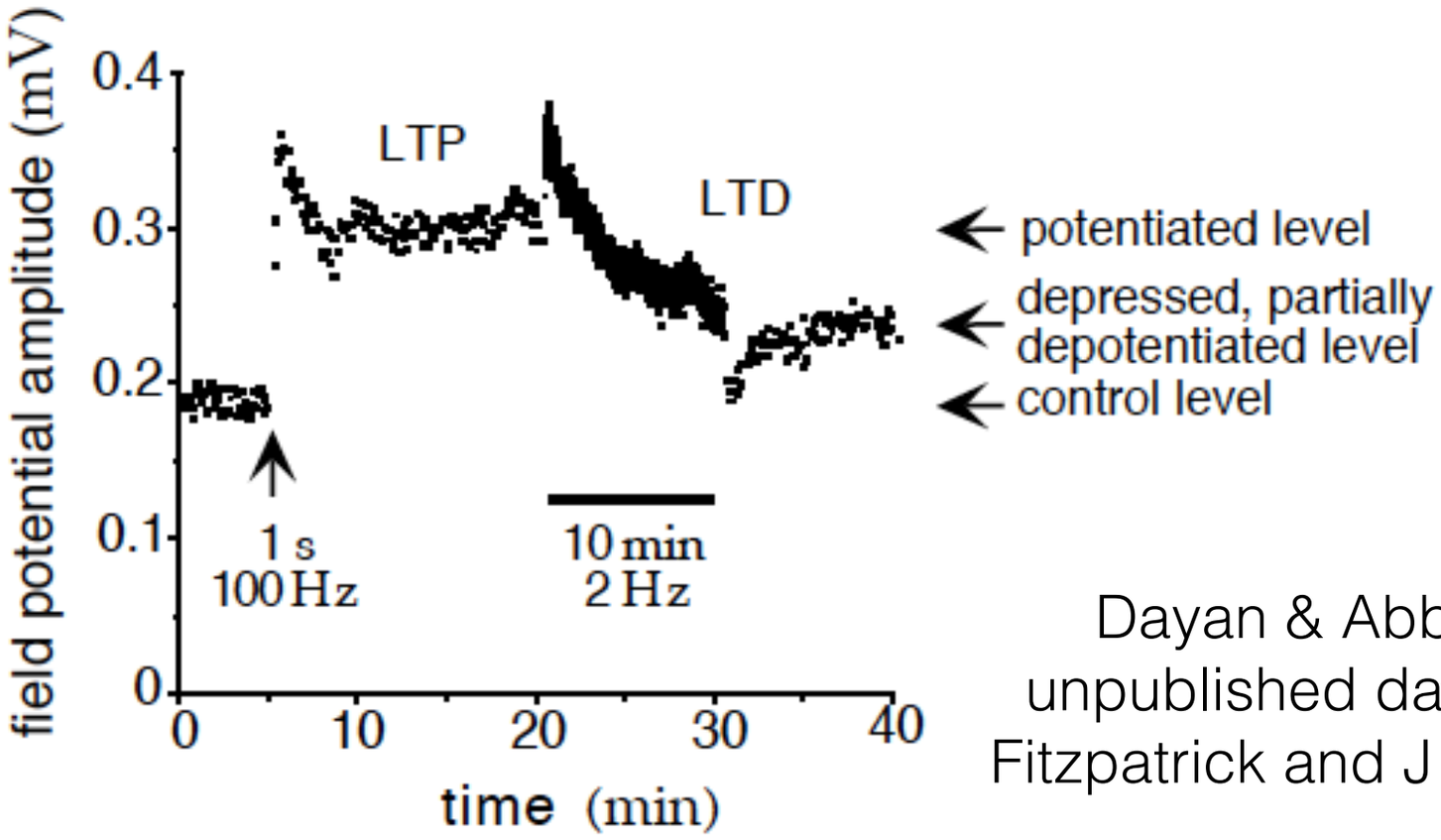
Non-Hebbian learning: verandering van intrinsieke exciteerbaarheid (homeostatic scaling)

Is er ook **leren bij interneuronen**?

Long Term Potentiation en Depression

Zie colleges
Harm Krugers



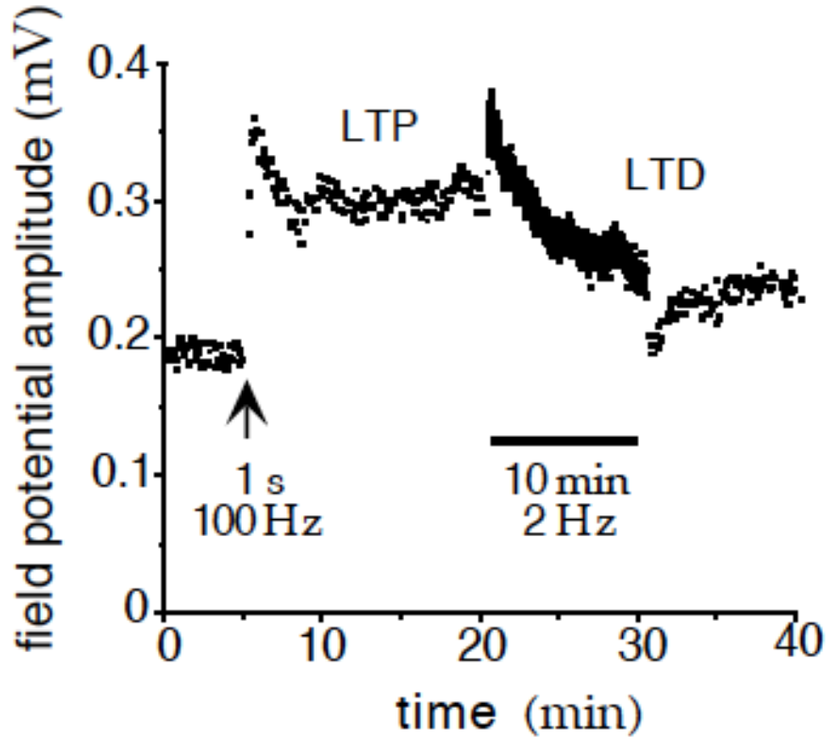


Dayan & Abbott, unpublished data of J Fitzpatrick and J Lisman

Long Term Potentiation en Depression

LTP / LTD

- Laat zien *dat* synapsen van sterkte kunnen veranderen
- Laat zien *dat* dit inderdaad van de activiteit (correlatie) pre- en postsynaptische cel afhangt
- Maar verklaart niet *hoe* je hiermee dan ook daadwerkelijk herinneringen (patronen) kunt opslaan in een netwerk
- Hier heb je modellen voor nodig!



Dayan & Abbott, unpublished data of J Fitzpatrick and J Lisman

Programma

Vandaag

1. **Unsupervised learning**
    - LTP/LTD modellen: rate-based Hebbian learning
        - voorbeeld: Leren in Hopfield netwerk
        - Andere ‘Hebb-achtige’ regels
        - STDP: spike-based Hebbian learning
2. **Supervised learning**

College 2

3. **Reinforcement learning**

Hoe modelleren we dit?

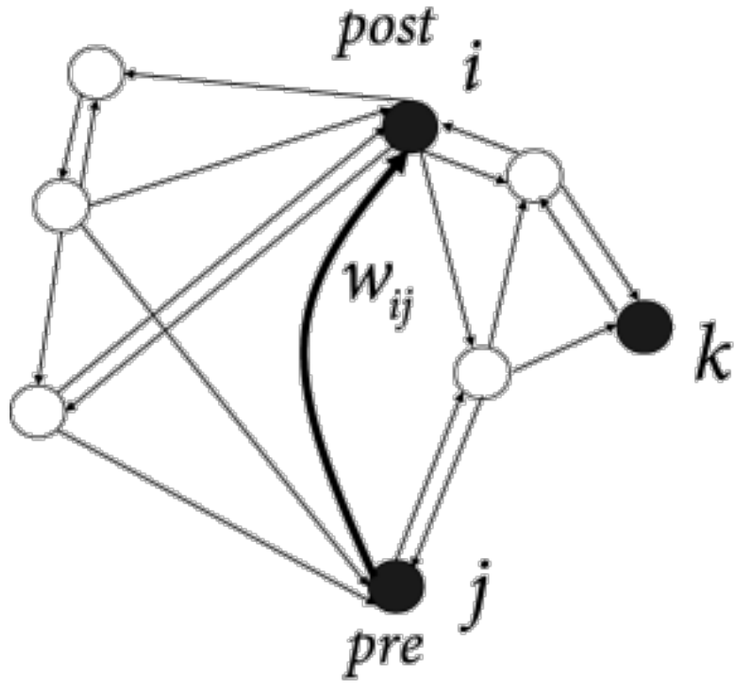
$$w_{ij,\ \text{nieuw}} = w_{ij,\ \text{oud}} + \Delta w_{ij}$$

$$\Delta w_{ij} = f(a_i, a_j, w_{ij})$$

- Wiskundige vorm **leerregel (learning rule)**
- hoe hangt de *verandering* van synaps-sterkte af van activiteit

$$\frac{\Delta w_{ij}}{\Delta t} = f(a_i, a_j, w_{ij})$$

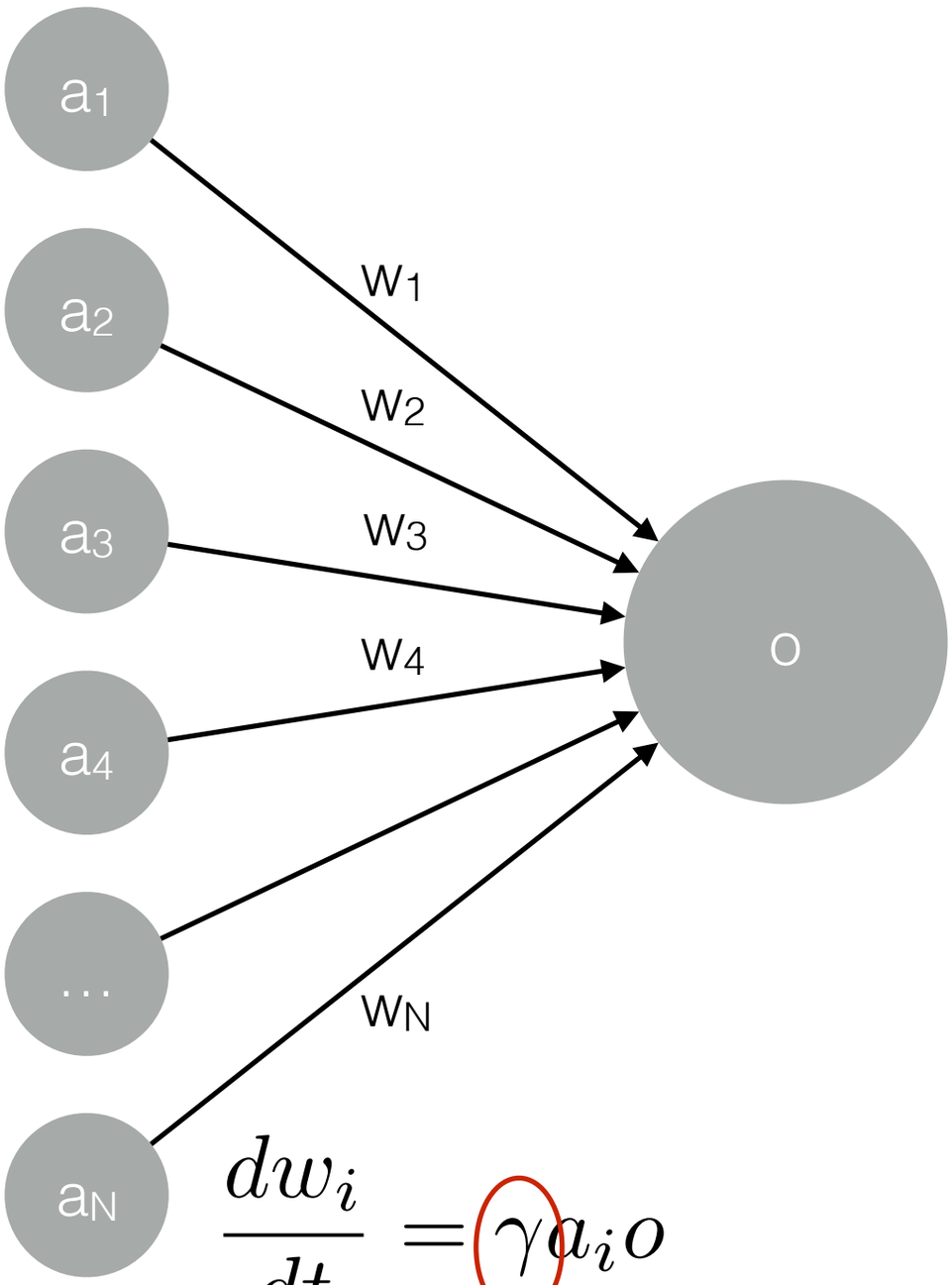
$$\frac{dw_{ij}}{dt} = f(a_i, a_j, w_{ij})$$



Differentiaalvergelijking

$$\frac{dw_{ij}}{dt} = f(a_i, a_j, w_{ij})$$

- Beschrijft verandering in w
- voor tijd continu (niet in stapjes)
- verandering w stopt als $f(a_i, a_j, w^*) = 0$
- dus als $w = w^*$, geen verandering w: **evenwichtspunt**
- Evenwichtspunt kan **stabiel** (aantrekkend) of **instabiel** (afstotend) zijn
- colleges André Heck



Eenvoudig voorbeeld

- Hebb: verandering van synaps-sterkte neemt alleen toe als zowel presynaptische (a) als postsynaptische (o) neuron actief zijn

- γ is ‘learning rate’: hoe snel neemt w af/toe met activiteit

- aanname: leren is langzaam → gemiddelde over veel patronen

- aanname: postsynaptische activiteit o hangt lineair af van input

$$\frac{dw_i}{dt} = \gamma a_i o$$

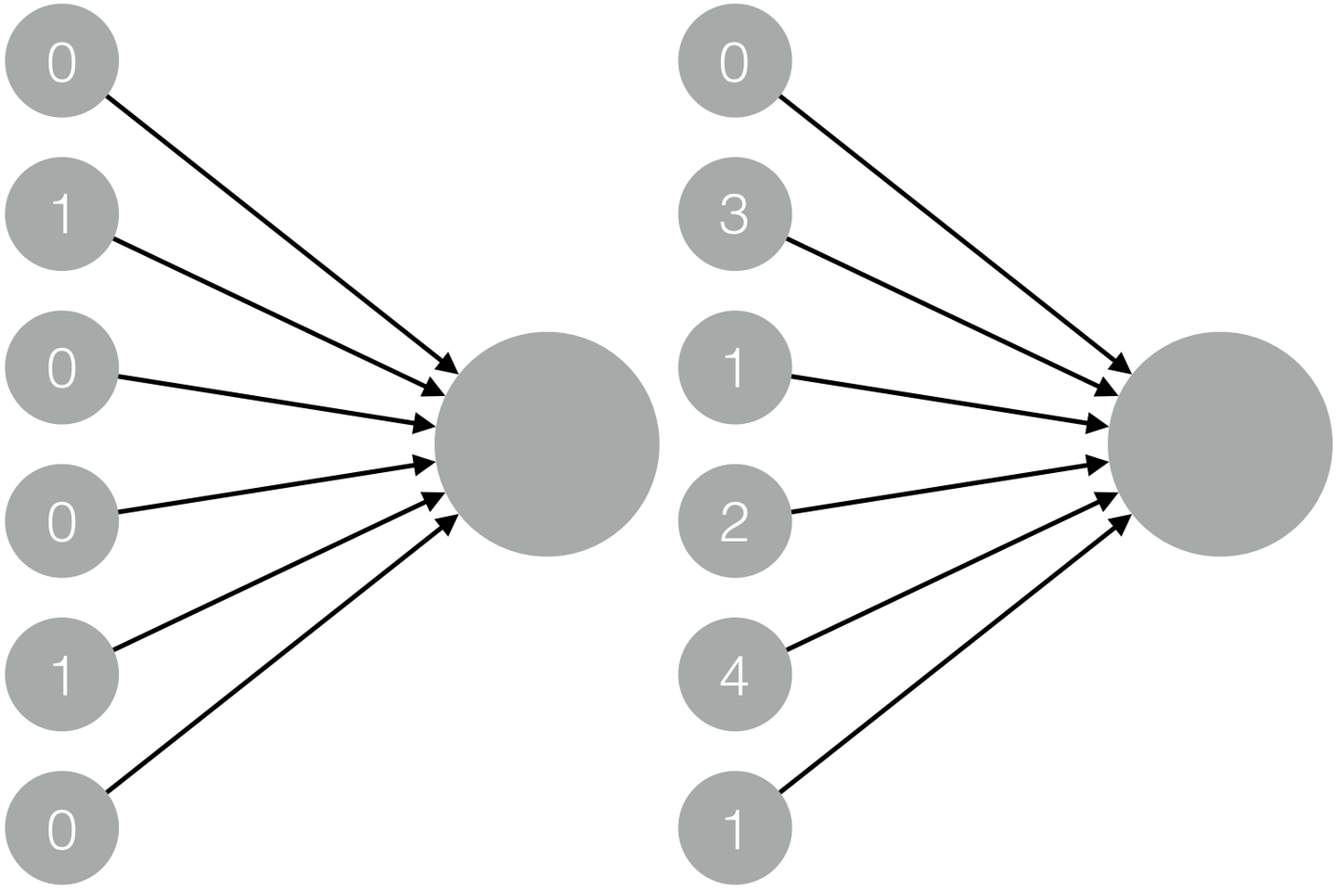
$$\frac{dw_i}{dt} = \gamma \langle a_i o \rangle_{\text{patronen}}$$

$$o = \sum_j w_j a_j$$

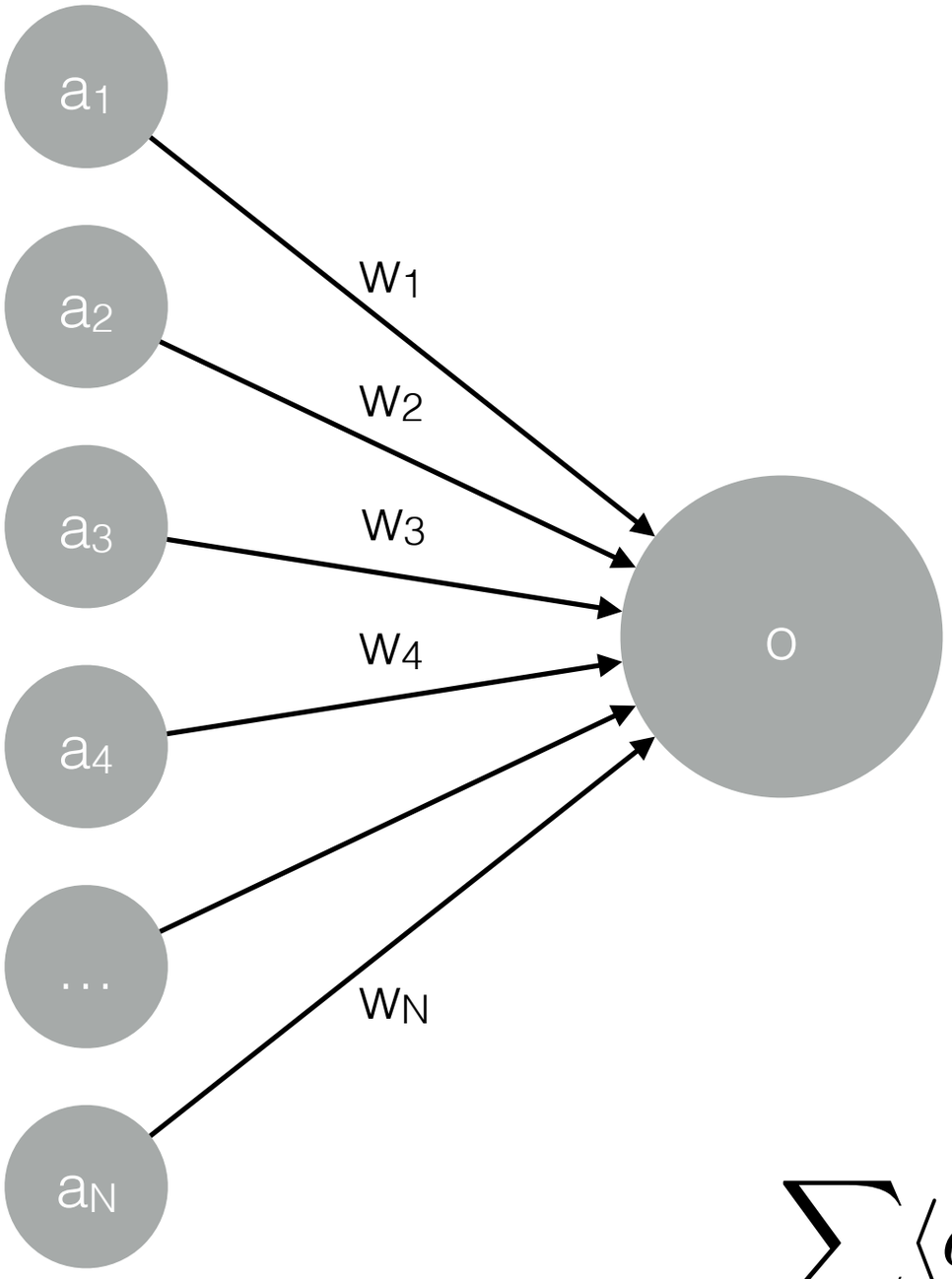
Intermezzo - Patronen

- Netwerk leert meerdere **patronen**

- **patroon**: activiteit van het netwerk, dus activiteit alle neuronen



Eenvoudig voorbeeld



$$\frac{dw_i}{dt} = \gamma \langle a_i o \rangle_{\text{patronen}} \qquad o = \sum_j w_j a_j$$

$$\frac{dw_i}{dt} = \gamma \sum_j \langle a_i a_j \rangle_{\text{patronen}} w_j$$

$$\sum_j \langle a_i a_j \rangle_{\text{patronen}} = Q = \text{correlatiematrix}$$

Correlatiematrix: hoeveel hangen gemiddeld de activiteit van neuron i en j samen?

Dus: gewicht van neuron dat veel tegelijk met andere neuronen vuurt wordt sterker (want dan is het outputneuron actief): 1 dominant inputpatroon

Eenvoudig voorbeeld

$$\frac{dw_i}{dt} = \gamma \sum_j Q_{ij} w_j$$

- Dus: 'Naïef' Hebbiaans leren geeft gewichten die afhangen van correlaties in de inputpatronen (en van andere gewichten)
- Wanneer veranderen gewichten niet meer?
- Evenwicht als Qw = 0. Maar Q positief!
- Probleem: **Instabiliteit** (gewichten groeien zonder grens)
- Probleem: alleen LTP, geen LTD (want correlaties positieve activiteit altijd positief)

Programma

Vandaag

1. **Unsupervised learning**
    - LTP/LTD modellen: rate-based Hebbian learning
        - voorbeeld: Leren in Hopfield netwerk
        - Andere ‘Hebb-achtige’ regels
        - STDP: spike-based Hebbian learning

2. **Supervised learning**

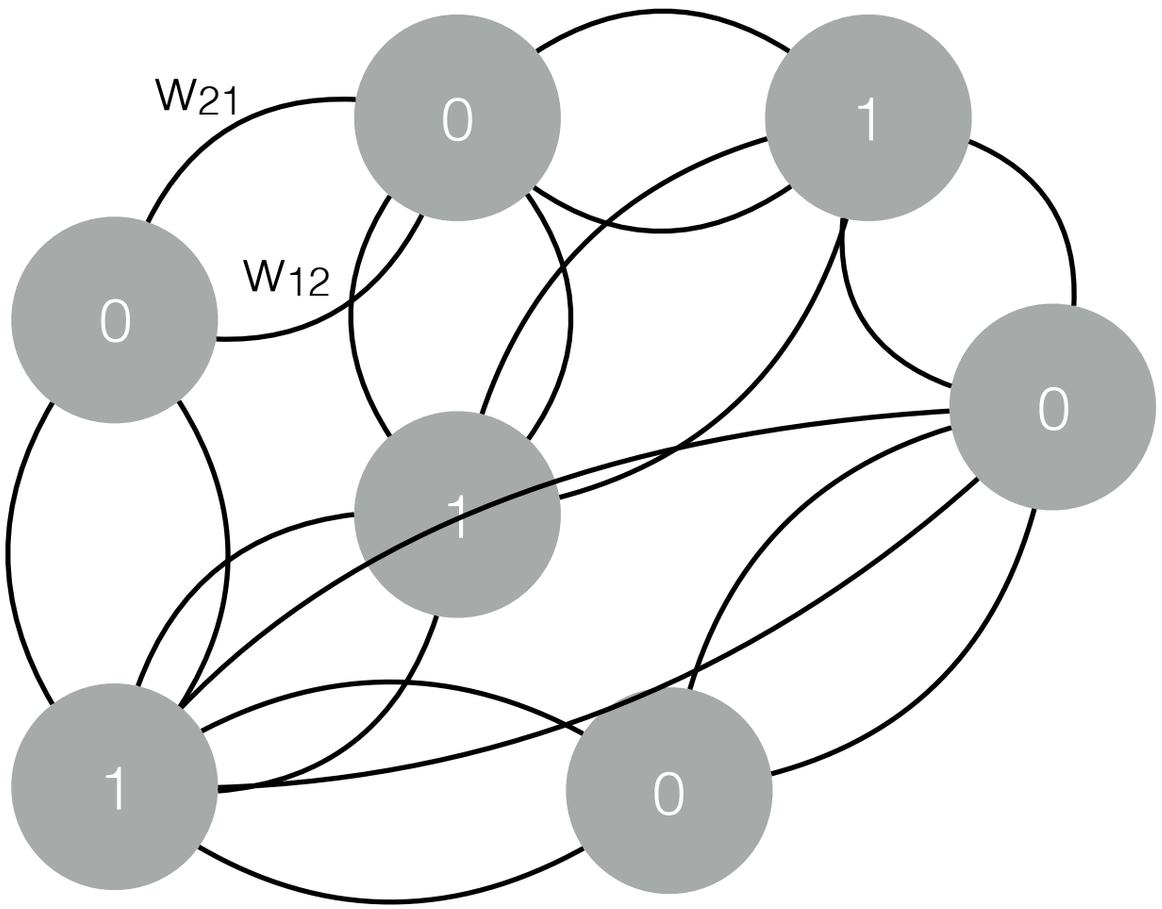
College 2

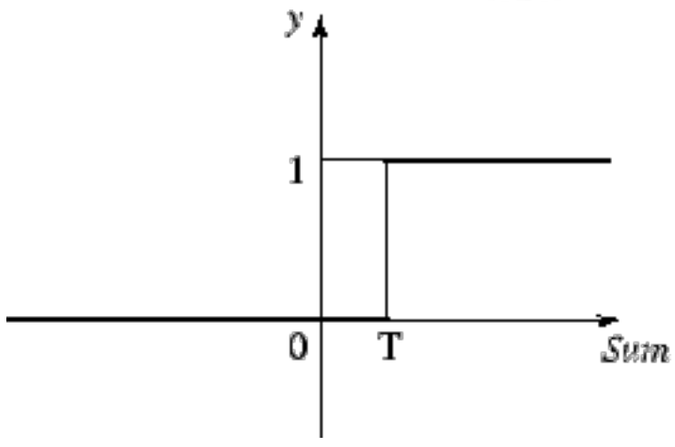
3. **Reinforcement learning**

Voorbeeld: Hebbian learning in Hopfield netwerk

- Wat was ook alweer een Hopfield netwerk?

Hopfield network (1982)



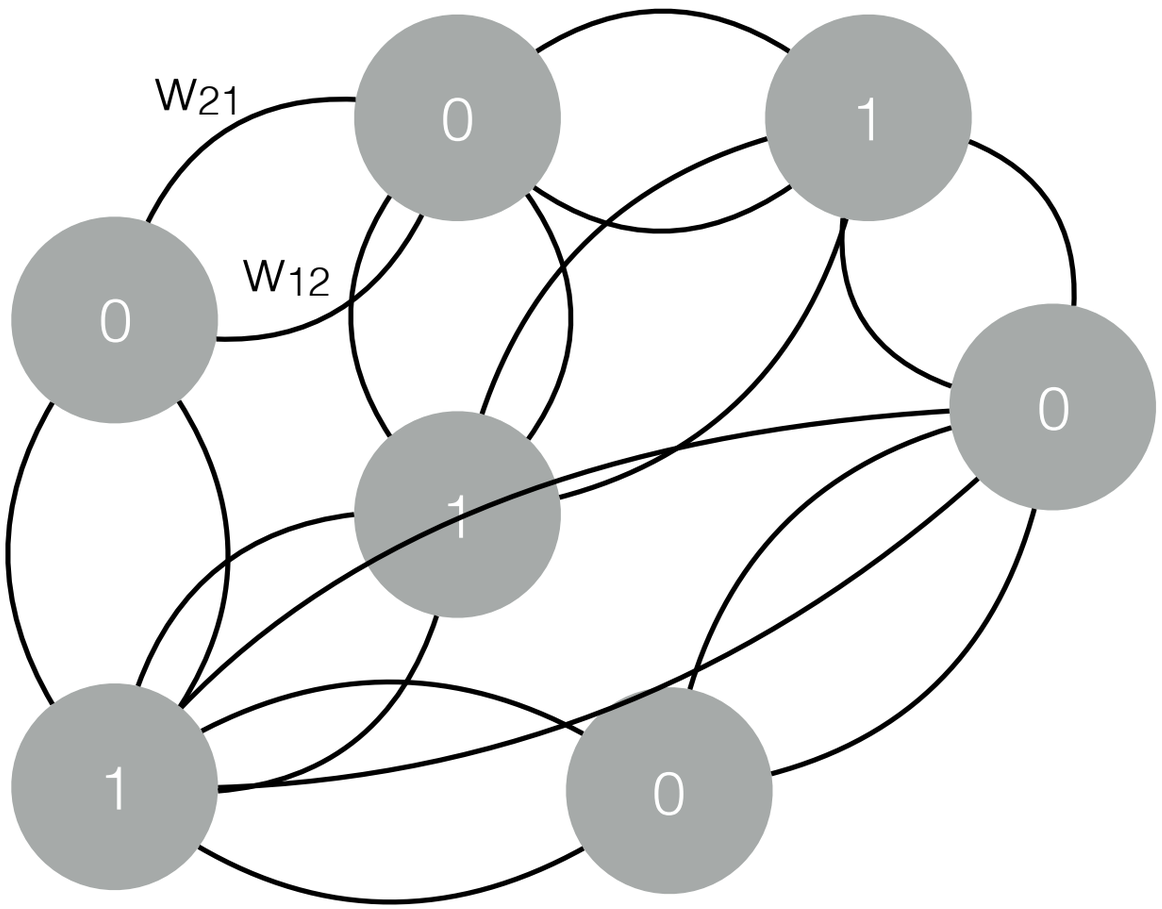


- Neuronen: MPN, activiteit neuron n is x_n (0 of 1)
- Alle verbindingen symmetrisch: $w_{21} = w_{21}$
- Geen zelf-verbindingen : $w_{kk} = 0$

$$\text{output} = H(\text{input}_1 * w_1 + \text{input}_2 * w_2 + ... - T)$$

$$= H(\sum_{n=1}^{N} \text{input}_n * w_n - T)$$

Hopfield network (1982)



- Je kiest een input-patroon:
  - geef elk neuron x=0 of x=1
  - zet de T's (externe input & drempel)

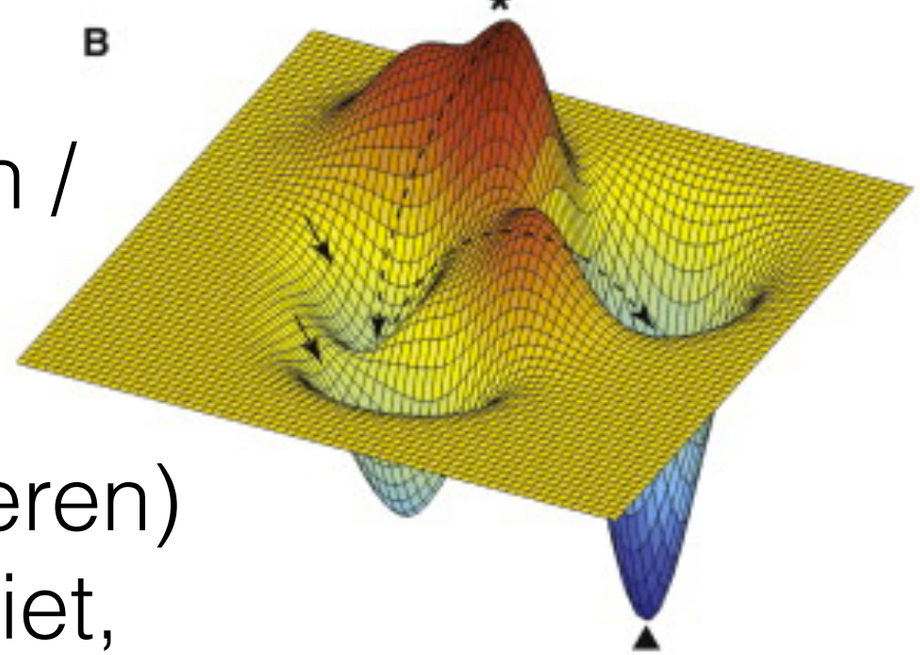
- Updating: één voor één of tegelijk

$$x_j(t_{n+1}) = H(\sum_i w_{ij} x_i(t_n) - T_j)$$

Hopfield network (1982)

- Een Hopfield netwerk convergeert altijd naar een stabiel patroon (x verandert niet meer)

- Dit is een minimum van de **energie-functie**



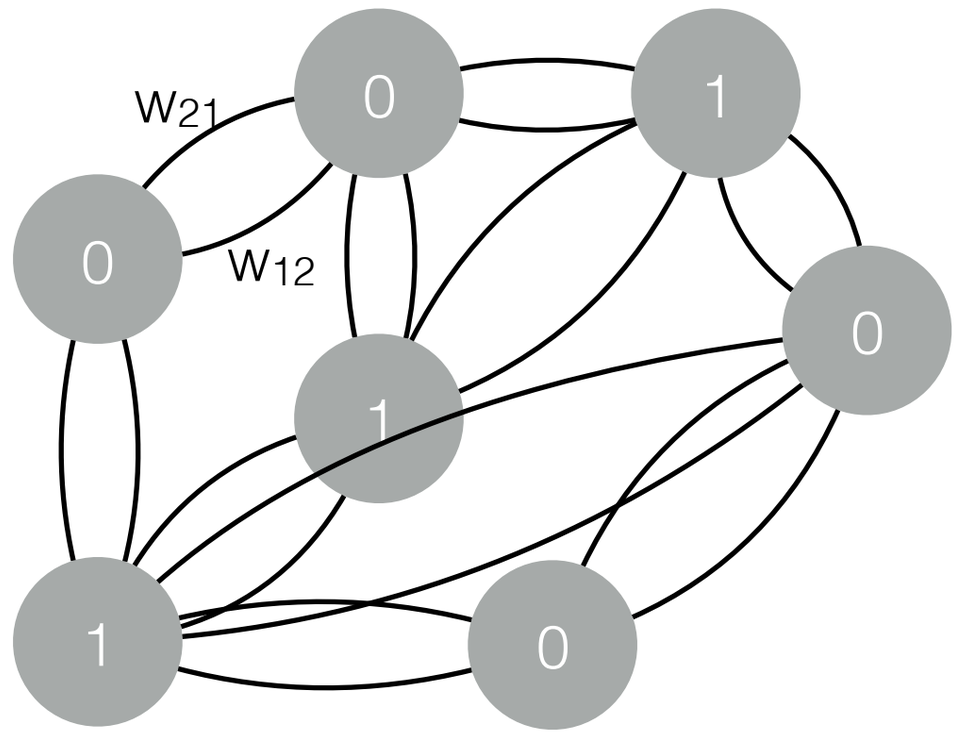
- Hoe kan een netwerk de verbindingen / energiefunctie leren?

- Onthou: 'leerfase' (gewichten veranderen) en ‘testfase' (gewichten veranderen niet, runnen netwerk; besproken bij perceptie)

Voorbeeld: Hebbian learning in Hopfield netwerk

Leerfase

- ‘Eenvoudige' Hebbiaanse leerregel in Hopfield netwerk



- Stel steeds de beginwaarden van de activiteit van de neuronen in: dit zijn de ‘patronen' die je wilt dat het netwerk leert

- noem zo’n patroon **σ**

- Je wilt dat dit de ‘dalen' in de energiefunctie worden

Voorbeeld: Hebbian learning in Hopfield netwerk

Leerfase

- Na enige tijd leren zijn je gewichten proportioneel met correlatiematrix input-patronen (maar blijven groeien!):

$$w_{ij} = \frac{1}{N} \sum_{\text{patronen } \mu} \sigma_i^\mu \sigma_j^\mu$$

- Klopt het dat ze symmetrisch zijn?

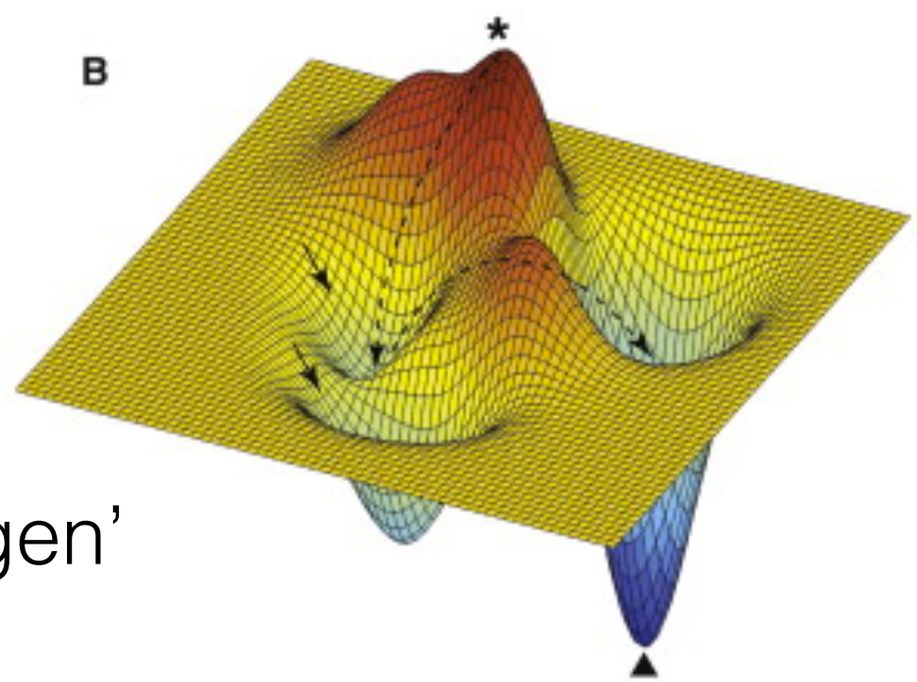
- Je kunt wiskundig aantonen dat geleerde patronen nu minima van de energie-functie zijn

Voorbeeld: Hebbian learning in Hopfield netwerk

Probleem 1:

- Dit werkt goed als het aantal patronen p << N (aantal neuronen)
- Maar als p > ~N/10 : er komen veel locale minima in de energie-functie →

  - Netwerk zal in testfase niet meer naar ‘geleerde' minimum gaan, maar blijft ‘hangen' in ander minimum
  - Dus: er komen ‘valse herinneringen’



Voorbeeld: Hebbian learning in Hopfield netwerk

- Probleem 1: hoe groter het aantal patronen p ten opzichte van aantal neuronen N, hoe meer lokale minima

- Oplossing 1: vergeten!

- Laat gewichten in een ‘slaap’-sessie langzaam minder worden:
    1. begin netwerk random
    2. laat lopen tot minimum: vind zo verschillende minima s_i^∞
    3. Laat netwerk ‘vergeten’: $w_{ij} \to w_{ij} - \frac{\lambda}{N} s_i^\infty s_j^\infty$
    4. Zo worden alle dalen minder diep (lokale minima verdwijnen)

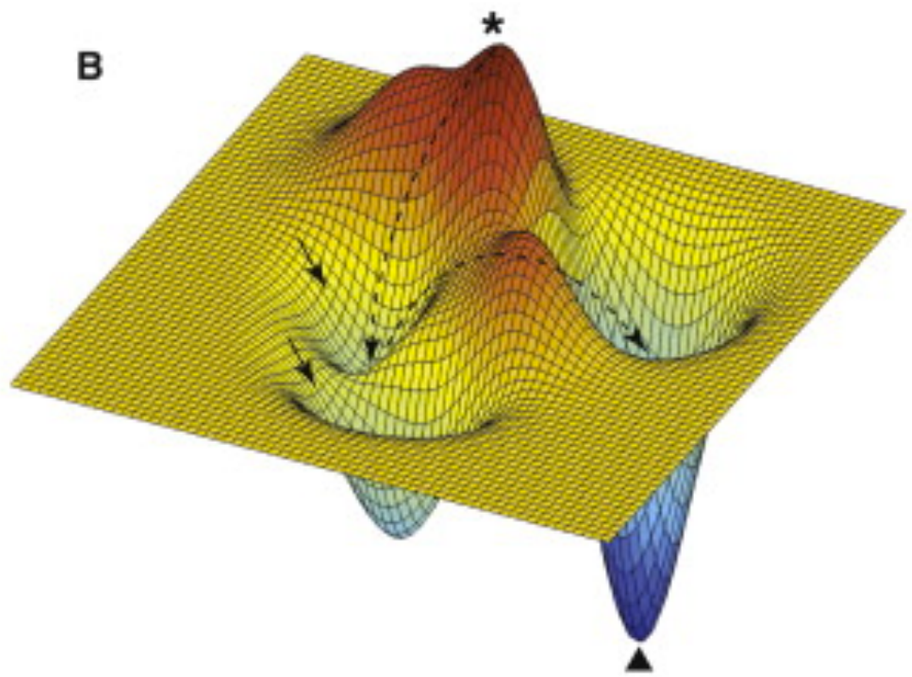
Voorbeeld: Hebbian learning in Hopfield netwerk

- Probleem 1: hoe groter het aantal patronen p ten opzichte van aantal neuronen N, hoe meer lokale minima

- Oplossing 1: vergeten! (in **leerfase**)

- Oplossing 2: voeg ruis toe bij **testfase**

    - zo blijf je minder snel ‘hangen' in lokale minima

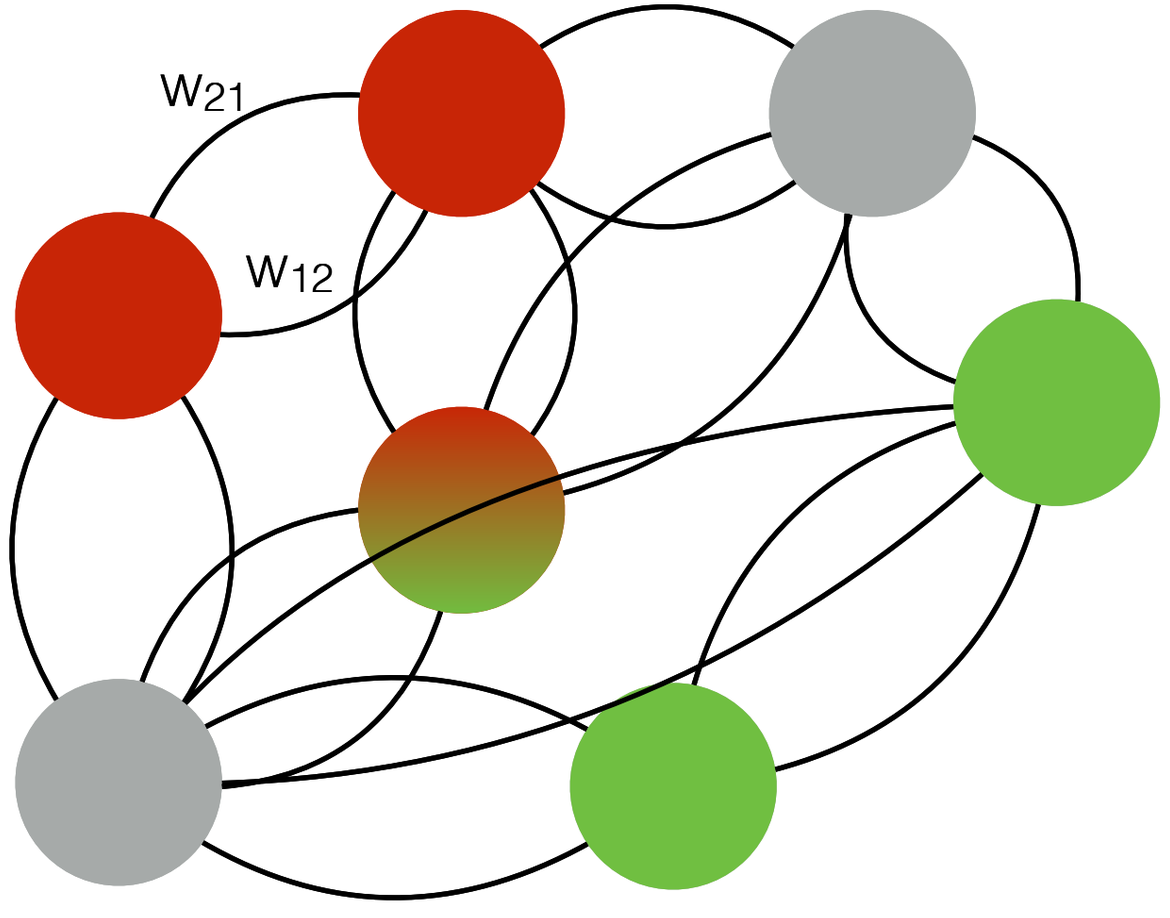


Voorbeeld: Hebbian learning in Hopfield netwerk

Probleem 2: als er patronen zijn die neuronen ‘delen'

- Bijvoorbeeld:
  - patroon 1
  - patroon 2

- Deze patronen kunnen makkelijk instabiel worden
- Dus: je ‘vergeet' patronen!



‘Eenvoudige' Hebb regel

Zorgt voor:

- cell assemblies / pattern completion
- leert correlaties in input

Problemen:

- Instabiel (gewichten blijven groeien)
- Alleen LTP, geen LTD (correlaties positief)
- lokale minima: ‘vergeten' nodig, of noisy updates
- instabiliteit patronen die neuronen 'delen'

Pauze