4. Learning Mechanisms

Overview of Learning

- Biology: synaptic plasticity
- Computation:
  - Self organized:
    - statistical regularities ⇒ internal models
    - long time scale
  - Error-driven: getting the right answers
    - outcomes ⇒ expectations
    - short time scale
  - Integration of two forms of learning

2/5/17 COSC 494/594 CCN 2

A. Biology of Synaptic Plasticity

2/5/17 COSC 494/594 CCN 3

Mechanisms of Synaptic Weight Change

- Amount of neurotransmitter released
- Nonspecific (extra-synaptic) neurotransmitter
- Change in number of post-synaptic receptors
- Rapid change in shape of dendritic spines
- Formation of new synapses

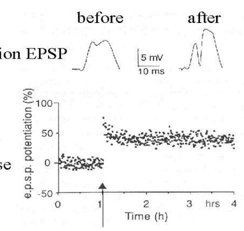
2/5/17 COSC 494/594 CCN 4

Synapses Change Strength
(in response to patterns of activity)

before after

population EPSP

slope of rising phase

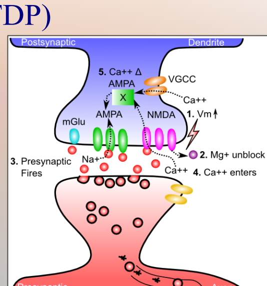


tetanic stimulation (250 Hz, 200 ms)

2/5/17 COSC 494/594 CCN 5
(slide < O'Reilly)

Spike Timing Dependent Plasticity (STDP)

1. V_m elevated by backpropagating action potential
2. Repels Mg^+ opening NMDA channels
3. Presynaptic neuron fires, releasing glutamate
4. Glutamate binds unblocked NMDA channels, allowing Ca^{++} influx
5. Ca^{++} increases number & efficacy of AMPA receptors



2/5/17 COSC 494/594 CCN 6
(fig. < O'Reilly, *Comp. Cog. Neurosci.*)

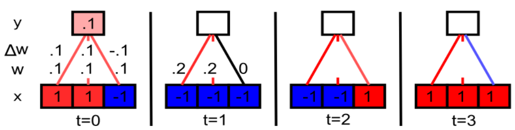
Hebb's Rule

- "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." — Donald Hebb
- I.e., neurons that fire together, wire together
- $\Delta W = xy$

  where x is sending activity and y is receiving activity

2/5/17 COSC 494/594 CCN 7

Hebb's Rule Learns Correlations



2/5/17 COSC 494/594 CCN 8

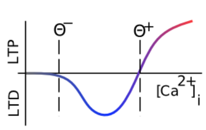
(slide < O'Reilly)

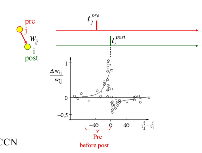
Long-term Potentiation (LTP) vs. Long-term Depression (LTD)

LTP vs. LTD depends on Ca^{++} concentration over several 100 msec

Records possible causal connection

Actual situation is more complicated with multiple APs

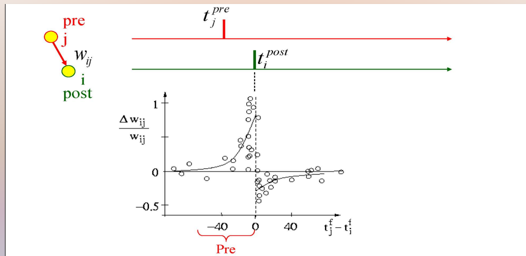




2/5/17 COSC 494/594 CCN 9

(figs. < O'Reilly, *Comp. Cog. Neurosci.*)

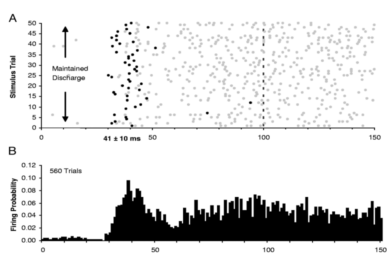
Causal Learning?



2/5/17 COSC 494/594 CCN 10

(slide < O'Reilly)

Let's Get Real…

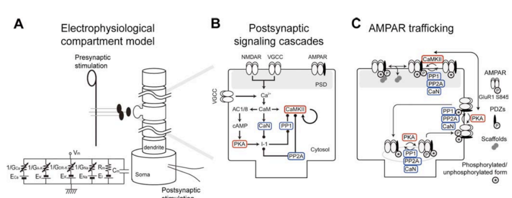


2/5/17 COSC 494/594 CCN 11

(slide < O'Reilly)

Urakubo et al. (2008) Model

- Highly detailed combination of 3 existing strongly-validated models:



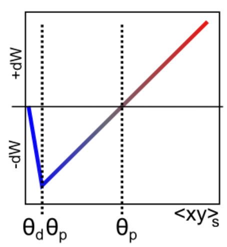
2/5/17 COSC 494/594 CCN 12

(slide < O'Reilly)

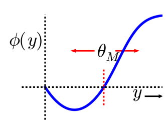
B. XCAL & BCM Models

2/5/17 COSC 494/594 CCN 13

XCAL = Linearized BCM

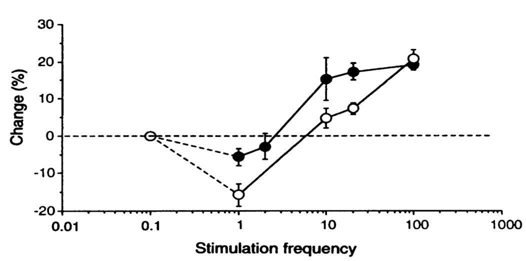


- Bienenstock, Cooper & Munro (1982) – BCM
- Adaptive threshold θ_M
  - ✓ Lower when less active
  - ✓ Higher when more
  - ✓ Homeostatic
  - ✓ Weights can decrease



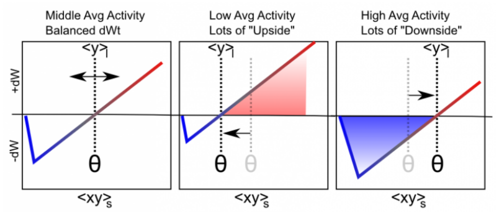
2/5/17 COSC 494/594 CCN 14
(slide based on O'Reilly)

Evidence for Floating Threshold



2/5/17 COSC 494/594 CCN 15
(slide based on O'Reilly)

Homeostatic Behavior



Floating threshold adapts to long-term postsynaptic activity

Tends to equalize activity among neurons

2/5/17 COSC 494/594 CCN 16

(fig. < O'Reilly, *Comp. Cog. Neurosci.*)

LTP/LTD Approximation

Piecewise linear approximation to LTP/LTD

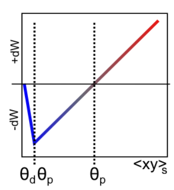
Typical $\theta_d = 0.1$

Floating threshold

$\Delta W = \epsilon f_{\mathrm{XCAL}}(\langle xy \rangle_s, \langle y \rangle_l)$

$$f_{\mathrm{XCAL}}(c, \theta_p) = \begin{cases} c - \theta_p & \text{if } c > \theta_p \theta_d \\ -c(1 - \theta_p)/\theta_d & \text{otherwise} \end{cases}$$

Note: $\Delta W \approx \epsilon x_s (y_s - y_l)$



2/5/17 COSC 494/594 CCN 17

(fig. < O'Reilly, *Comp. Cog. Neurosci.*)

Computation of Floating Threshold

$$\text{if } y > .2 \text{ then } y_l = y_l + \frac{1}{\tau_l}(\max - y_l)$$

$$\text{else } y_l = y_l + \frac{1}{\tau_l}(\min - y_l)$$

2/5/17 COSC 494/594 CCN 18

Integrating Over Experience

- Units that are activated (initially) by a wide range of stimuli end up representing an *average* of all of these stimuli
  - The unit will learn what they have in common
- Units that are activated (initially) by a narrow range of stimuli end up representing very *specific* features
  - If a unit is only activated by a single stimulus, it will only represent that stimulus
- Thus the learning rule allows you to represent concepts at varying levels of abstraction, depending on the excitability of the unit

2/5/17 COSC 494/594 CCN 19

(slide based on Frank)

Multiple Units

- One detector can only represent one “thing” (pattern of correlated features)
- Goal: We want to have different units in the network learn to “specialize” for different things, such that each thing is represented by at least one unit
- Random initial weights and inhibitory competition are important for achieving this goal

2/5/17 COSC 494/594 CCN 20

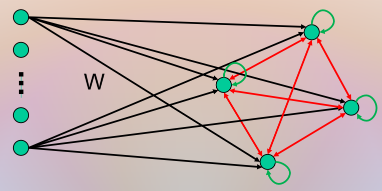
(slide based on Frank)

Simple Competitive Learning

- Competitive learning network
  - two layers, randomly initialized weights
  - second is self-reinforcing, mutually inhibitory
  - “winner takes all” dynamics
- Learning
  - winner moves toward last
  - weight vectors move to centers of clusters

2/5/17 COSC 494/594 CCN 21

Competitive Network



2/5/17 COSC 494/594 CCN 22

Diagram of Competitive Learning

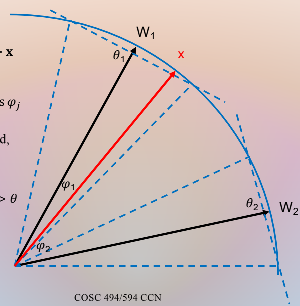
$$y_j = \sum_{k=1}^{n} W_{jk} x_k = \mathbf{W}_j \cdot \mathbf{x}$$

$$\mathbf{W}_j \cdot \mathbf{x} = \|\mathbf{W}_j\| \, \|\mathbf{x}\| \cos \varphi_j$$

If the $\mathbf{W}_j$ are normalized,

$$\mathbf{W}_j \cdot \mathbf{x} = \|\mathbf{x}\| \cos \varphi_j$$

$$y_j > \theta \text{ if } \|x\| \cos \varphi_j > \theta$$



2/5/17 COSC 494/594 CCN 23

Self-Organized Learning

- Inhibitory competition
  - ensures sparse representation
- Hebbian "rich get richer"
  - adjustment toward last pattern matched
- Slow threshold adaptation
  - adjusts receptive fields
  - equalizes cluster probabilities
- Homeostasis
  - distributes activity among neurons
  - more common patterns are more precisely represented
- Gradually develops statistical model of environment

2/5/17 COSC 494/594 CCN 24

emergent demonstration: Self_Organizing

2/5/17 COSC 494/594 CCN 25

C. Error-driven Learning

2/5/17 COSC 494/594 CCN 26

Self-organized vs. Error-driven Learning

- Last time we discussed self organized Hebbian learning
  - Leverage correlations to grow detectors that correspond to things in the world (cats, professors...)
- Today we will discuss error-driven (task) learning
  - Task = producing a specific output pattern in response to an input pattern
  - e.g., reading; giving the correct answer to 3 + 3

2/5/17 COSC 494/594 CCN 27
(slide based on Frank)

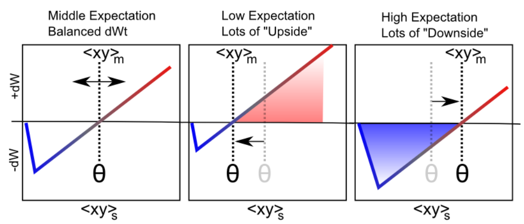
Task Learning

- Task learning encompasses:
  - — Giving an appropriate response to a stimulus
  - — Arriving at an accurate interpretation of a situation
  - — Generating a correct expectation of what will happen next
- In all of the above cases, there is a correct answer...

2/5/17 COSC 494/594 CCN 28

(slide < Frank)

Floating Threshold = Medium Term Synaptic Activity (Error-Driven)



2/5/17 COSC 494/594 CCN 29

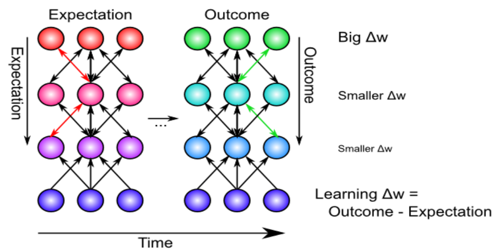
(slide < O'Reilly)

Error-Driven Learning

- For achieving intended outcomes
- Fast threshold adaptation
- Short-term outcome – medium-term expectation
  - ✓ "plus phase" – "minus phase"
- Depends on bidirectional connections
  - ✓ communicates error signals back to earlier layers
- Contrastive Attractor Learning (CAL)
  - ✓ approximately equivalent to backpropagation algorithm when combined with bidirectional connections

2/5/17 COSC 494/594 CCN 30

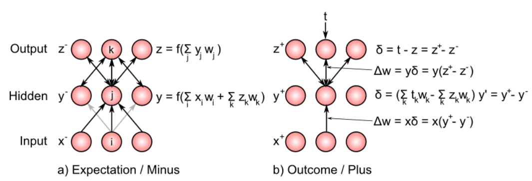
Backpropagation: Mathematics of Error-driven Learning



2/5/17 COSC 494/594 CCN 31

(slide < O'Reilly)

Credit/Blame Assignment via GeneRec



a) Expectation / Minus b) Outcome / Plus

The Plus Phase helps identify bridging units that are well connected to both the input and the target output, and GeneRec adjusts weights to maximize the activity of these units. — M. Frank

2/5/17 COSC 494/594 CCN 32

(fig. < O'Reilly, *Comp. Cog. Neurosci.*)

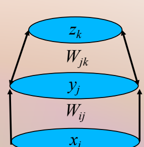
More precisely…

Minus Phase:

Clamp x_i^- (inputs)

$y_j^- = f\left(\sum_i x_i^- W_{ij} + \sum_k W_{jk} z_k^-\right)$

$z_k^- = f\left(\sum_j y_j^- W_{jk}\right)$



Plus Phase:

Clamp z_k^+ (target outputs)

$\Delta W_{jk} = y_j^-(z_k^+ - z_k^-) = y_j^- \delta_k$ where $\delta_k = z_k^+ - z_k^-$

$y_j^+ = f\left(\sum_i x_i^- W_{ij} + \sum_k W_{jk} z_k^+\right)$

$\Delta W_{ij} = x_i^-(y_j^+ - y_j^-) = x_i^- \delta_j$ where $\delta_j = y_j^+ - y_j^-$

2/5/17 COSC 494/594 CCN 33

Two Issues

- Two issues:
  - — Need weights to be symmetric
  - — Why should we use minus phase sending activity instead of plus phase?
- Solution: Average together plus and minus phase sending activation, and average together feedforward and feedback weight changes

$$\Delta W_{ij} = \epsilon\left[\frac{x_i^+ + x_i^-}{2}(y_j^+ - y_j^-) + \frac{y_j^+ + y_j^-}{2}(x_i^+ - x_i^-)\right]$$

$$= \frac{\epsilon}{2}\left[(x_i^+ + x_i^-)(y_j^+ - y_j^-) + (x_i^+ - x_i^-)(y_j^+ + y_j^-)\right]$$

$$= \epsilon(x_i^+ y_j^+ - x_i^- y_j^-)$$

2/5/17 COSC 494/594 CCN 34

(slide based on Frank)

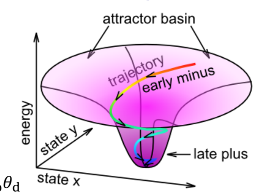
Contrastive Attractor Learning

Network learns contrast between:

early phase/expectation (minus)

late phase/outcome (plus)

Gets more quickly to late phase, which has integrated more constraints



$$f_{\text{XCAL}}(c, \theta_p) = \begin{cases} c - \theta_{\text{p}} & \text{if } c > \theta_{\text{p}}\theta_{\text{d}} \\ -c(1 - \theta_{\text{p}}) & \text{otherwise} \end{cases}$$

$$\Delta W = \epsilon f_{\text{XCAL}}(\langle xy\rangle_{\text{s}}, \langle xy\rangle_{\text{m}})$$

$$\approx \epsilon f_{\text{XCAL}}(x_{\text{s}} y_{\text{s}}, x_{\text{m}} y_{\text{m}})$$

$$\approx \epsilon(x_{\text{s}} y_{\text{s}} - x_{\text{m}} y_{\text{m}})$$

2/5/17 COSC 494/594 CCN 35

Backpropagation Algorithm

- Adjusts weights in multilayer artificial neural net by “steepest descent” in order to minimize error
  - — does this by estimating partial derivatives of error with respect to weights
- (Re)invention in 1986 rejuvenated artificial neural net research
  - — had been neglected for 15 years
- Basis for current “deep learning” methods
- Uses biologically implausible backwards signal propagation through synapses
- Nevertheless, the biologically plausible XCAL algorithm is a good approximation to BP

2/5/17 COSC 494/594 CCN 36

Relation to BP Algorithm (backward propagation of errors)

$$
\begin{aligned}
\delta_j &= y_j^+ - y_j^- \\
&= f\left(\textstyle\sum_i x_i^- W_{ij} + \sum_k W_{jk} z_k^+\right) - f\left(\sum_i x_i^- W_{ij} + \sum_k W_{jk} z_k^-\right) \\
&\approx f'\left(\textstyle\sum_i x_i^- W_{ij} + \sum_k W_{jk} z_k^-\right) \left(\sum_k W_{jk} z_k^+ - \sum_k W_{jk} z_k^-\right) \\
&= y_j{}' \left[\textstyle\sum_k W_{jk} (z_k^+ - z_k^-)\right] \\
&= y_j{}' \textstyle\sum_k W_{jk} \delta_k
\end{aligned}
$$

2/5/17 COSC 494/594 CCN 37

Summary: Error-driven Learning

- Adjusts expectations to better match outcomes (better prediction)
- Uses later, better information to train earlier expectations (later training earlier)
- Allows network to more quickly settle into attractor (quicker convergence)
- But at this time there is limited empirical support for medium timescale floating threshold

2/5/17 COSC 494/594 CCN 38

emergent demonstration: Pattern_Associator

2/5/17 COSC 494/594 CCN 39

emergent demonstration: Error_Driven_Hidden

2/5/17 COSC 494/594 CCN 40

Weight Bounding

- Biologically, synaptic weights are bounded
- Approximate computationally by exponential approach to limits 0 and 1:

```
if dwt > 0 then wt = wt + (1–wt)×dwt;
else wt = wt + wt×dwt;
```

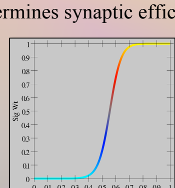
- Gradual approach preserves signal better than clipping

2/5/17 COSC 494/594 CCN 41

Contrast Enhancement

- To enhance contrast in activities for midrange weights, distinguish:
  - — internal weight w determined by learning
  - — contrast-enhanced weight $\hat{w}$ determines synaptic efficacy
- Contrast enhancement function:

$$\hat{w} = \frac{1}{1+\left(\frac{w}{\theta(1-w)}\right)^{-\gamma}}$$



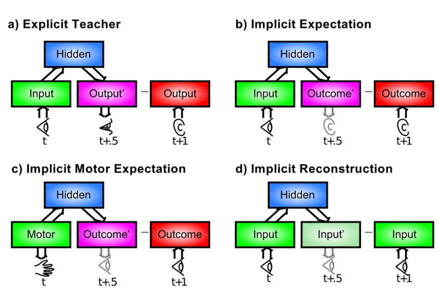
2/5/17 COSC 494/594 CCN 42

D. What drives learning?

Learning Signals?

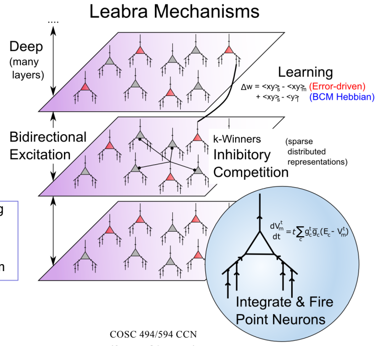
- What constitutes an "outcome"?
- Dopamine bursts arise from unexpected rewards or punishments (reinforcers)
  - — violation of expectation
  - — needs correction
- Dopamine modulates synaptic plasticity
  - — controls λ: $\Delta W = \eta f_{\mathrm{XCAL}}\left(x_s y_s, x_m\left(\lambda y_l + (1-\lambda) y_m\right)\right)$
- Probably not the whole story

Learning Situations



(fig. < O'Reilly, *Comp. Cog. Neurosci.*)

Leabra



Leabra = Learning in an Error-driven and Associative, Biologically Realistic Algorithm

2/5/17 COSC 494/594 CCN 46

(fig. < O'Reilly)

emergent demonstration: Family_Trees

2/5/17 COSC 494/594 CCN 47

Supplementary

Oja's Rule (1)

- To avoid instability due to unlimited growth, suppose we keep the weight vectors normalized:
$$\mathbf{w}' = \frac{\mathbf{w} + \epsilon y \mathbf{x}}{\|\mathbf{w} + \epsilon y \mathbf{x}\|}$$
- Consider one component: $w'_j = \frac{w_j + \epsilon y x_j}{\left[\sum_k (w_k + \epsilon y x_k)^2\right]^{1/2}}$
- Call RHS $f(\epsilon)$ and expand in Taylor series:
$$f(\epsilon) = f(0) + \epsilon f'(0) + \epsilon^2 f''(0) + \cdots$$

Oja's Rule (2)

- You can show (algebra!):
$$w'_j \approx \frac{w_j}{\|\mathbf{w}\|} + \epsilon \left[\frac{y x_j}{\|\mathbf{w}\|} - \frac{w_j \sum_k y x_k w_k}{\|\mathbf{w}\|} \right] + O(\epsilon^2)$$
- If $\mathbf{w}$ is normalized,
$$\begin{aligned} w'_j &= w_j + \epsilon \left(y x_j - w_j y \sum\nolimits_k x_k w_k \right) \\ &= w_j + \epsilon \left(y x_j - w_j y^2 \right) \end{aligned}$$
- In vector form, $\Delta \mathbf{w} = \epsilon y \mathbf{x} - \epsilon y^2 \mathbf{w}$
- This is Hebbian learning with decay

Stability and Convergence

- Under broad conditions, will stabilize: $0 = \Delta \mathbf{w} = \epsilon(\mathbf{x} y - y^2 \mathbf{w})$
- Note $y = \sum_k x_k w_k = \mathbf{x}^{\mathrm{T}} \mathbf{w} = \mathbf{w}^{\mathrm{T}} \mathbf{x}$
- Therefore, $0 = \epsilon\left[\mathbf{x}(\mathbf{x}^{\mathrm{T}}\mathbf{w}) - (\mathbf{w}^{\mathrm{T}}\mathbf{x})(\mathbf{x}^{\mathrm{T}}\mathbf{w})\mathbf{w}\right]$
  $0 = \epsilon\left[(\mathbf{x}\mathbf{x}^{\mathrm{T}})\mathbf{w} - \mathbf{w}^{\mathrm{T}}(\mathbf{x}\mathbf{x}^{\mathrm{T}})\mathbf{w}\mathbf{w}\right]$
  $0 = (\mathbf{x}\mathbf{x}^{\mathrm{T}})\mathbf{w} - \mathbf{w}^{\mathrm{T}}(\mathbf{x}\mathbf{x}^{\mathrm{T}})\mathbf{w}\mathbf{w}$
- The parenthesis is the covariance matrix $\mathbf{R} = \langle \mathbf{x}\mathbf{x}^{\mathrm{T}} \rangle$
- Hence $\mathbf{R}\mathbf{w} = (\mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w})\mathbf{w}$
- Therefore $\mathbf{w}$ is an eigenvector of $\mathbf{R}$ with eigenvalue $\lambda = \mathbf{w}^{\mathrm{T}}\mathbf{R}\mathbf{w}$
- In fact, you can show λ is the largest eigenvalue, and therefore $\mathbf{w}$ is the *first principal component*, which accounts for most variance in $\mathbf{x}$

Summary: Hebb's and Oja's Rules

- A single Hebbian neuron with decay (Oja's Rule) extracts the first principal component of a stationary input
- The variance of this component is given by the corresponding (maximum) eigenvalue
- Oja's Rule can be extended into a *generalized Hebbian algorithm* that extracts all the principal components and their variances

2/5/17 COSC 494/594 CCN 52