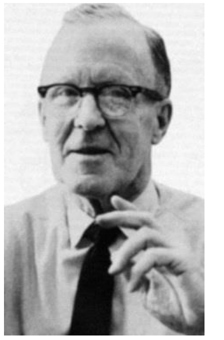
PCA by neurons

Hebb rule



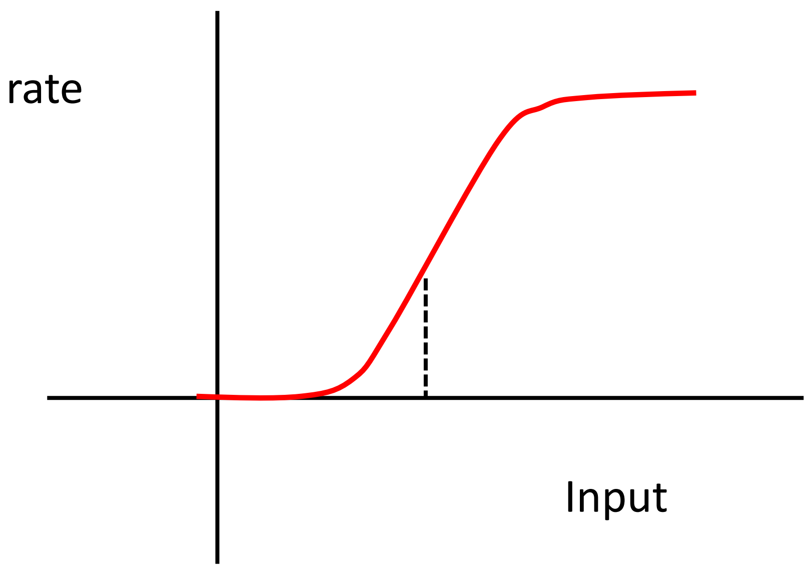
1949 book: 'The Organization of Behavior'
Theory about the neural bases of learning

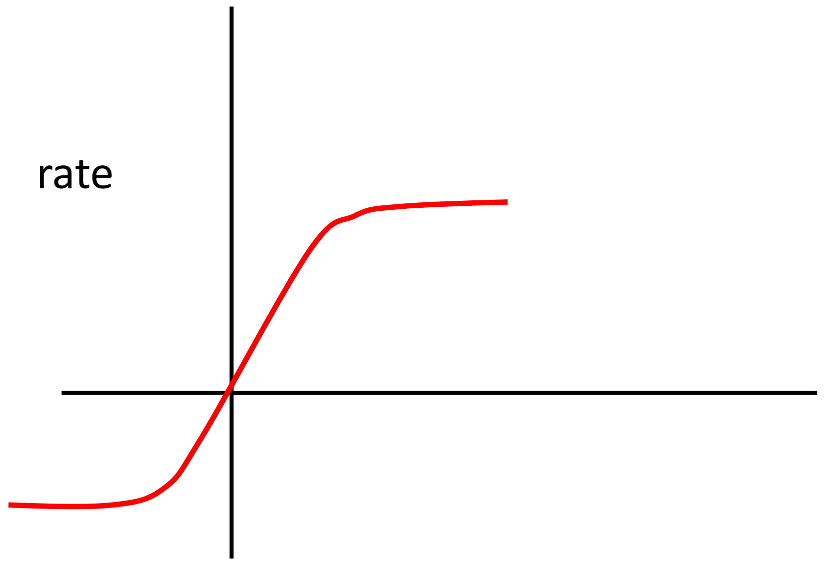
Learning takes place in synapses.
Synapses get modified, they get stronger when the pre- and post-synaptic cells fire together.

‘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’

"Cells that fire together, wire together"

Hebb Rule (simplified linear neuron)





The neuron performs $v = \underline{w}^T \underline{x}$

Hebb rule: $\Delta \underline{w} = \alpha \, \underline{x} \, v$ w, x can have negative values

Stability

The neuron performs $v = \underline{w}^T \underline{x}$

Hebb rule: $\Delta \underline{w} = \alpha \, \underline{x} \, v$

What will happen to the weights over a long time?

Use differential equation for Hebb:

$(1/\tau) \; dw / dt = \alpha \, \underline{x} \, v$

$d/dt \; |\underline{w}|^2 = 2\underline{w}^T d\underline{w} / dt$ (τ is taken as 1)

$= 2\underline{w}^T \alpha \underline{x} \, v$

$\underline{w}^T \underline{x} = v$ therefore:

$= 2\alpha v^2$

Therefore: $d/dt \; |\underline{w}|^2 = 2\alpha v^2$

The derivative is always positive, therefore *w will grow in size over time*

Oja's rule and normalization

length normalization:

$\underline{w} \leftarrow (\underline{w} + \alpha v\, \underline{x}) / ||\underline{w}||$

With Taylor expansion to first term:

$\underline{w}(t+1) = \underline{w}(t) + \alpha\, v(\underline{x} - v\underline{w})$ (Oja's rule)

Oja $\sim$ 'normalized Hebb'

Similarity to Hebb:

$\underline{w}(t+1) = w(t) + \alpha v \underline{x}'$ with $\underline{x}' = (\underline{x} - v\underline{w})$

Feedback, or forgetting term: $-\alpha v^2\, \underline{w}$

Erkki Oja

Oja E. (1982) A simplified neuron model as a principal component analyzer. Journal of Mathematical Biology, 15:267-2735

Oja rule: effect on stability

we used above:

$d/dt \ |\underline{w}|^2 = 2\underline{w}^T d\underline{w} / dt$

Put the new dw/dt from Oja rule: $\alpha \, v(\underline{x} - \underline{v}w)$

$= 2\alpha\underline{w}^T v(\underline{x} - v\underline{w})$ = (as before, $\underline{w}^T\underline{x} = v$)

$= 2\alpha v^2(1 - |w|^2)$

Instead of $2\alpha v^2$ we had before

Steady state is when $|w|^2 = 1$

Comment: Neuronal Normalization

Normalization as a canonical neural computation   Carandini & Heeger 2012

Uses a general form:

$$R_j = \gamma \frac{(\Sigma_k w_{jk} I_k)^n + \beta}{\sigma^n + (\Sigma_k \alpha_{jk} I_k^m)^p}$$

Different systems have somewhat different specific forms.
For contrast normalization:

$$R_j = \gamma \frac{\Sigma_i w_i C_i}{\sigma + \sqrt{\Sigma_k \alpha_k C_k^2}}$$

C_i are the input neurons, ‘local contrast elements’

Summary

Hebb rule:

$\underline{w}(t+1) = w(t) + \alpha v \underline{x}$

Normalization:

$\underline{w} \leftarrow (\underline{w} + \alpha v\, \underline{x}) / \|\underline{w}\|$

Oja rule:

$\underline{w} \leftarrow \underline{w} + \alpha v\, (\underline{x} - v\underline{w})$

Summary

For Hebb rule

$d/dt \ |w|^2 \sim 2\alpha v^2$ (growing)

For Oja rule:

$d/dt \ |w|^2 \sim 2\alpha v^2(1 - |w|^2)$ (stable for $|w| = 1$)

Convergence

- The exact dynamics of the Oja rule have been solved by Wyatt and Elfaldel 1995
- It shows that the $\underline{w} \rightarrow \underline{u}_1$ which is the first eigen-vector of $X^T X$
- Qualitative argument, not the full solution

Final value of $\underline{w}$

$\underline{\Delta w} = \alpha(\underline{x}v - v^2\,\underline{w})$ Oja rule

$v = \underline{x}^T\underline{w} = \underline{w}^T\underline{x}$

$\underline{\Delta w} = \alpha(\underline{x\ x}^T\underline{w} - \underline{w}^T\underline{x\ x}^T\underline{w}\ \underline{w})$

Averaging over inputs $\underline{x}$:

$\underline{\Delta w} = \alpha(C\underline{w} - \underline{w}^TC\underline{w}\ \underline{w}) = 0$ (0 for steady-state)

$\underline{w}^TC\underline{w}$ is a scalar, λ

$C\underline{w} - \lambda\underline{w} = 0$

At convergence (assuming convergence) $\underline{w}$ *is an eigenvector of C*

Weight will be normalized:

Also at convergence:

We defined $\underline{w}^T C \underline{w}$ as a scalar, λ

$\lambda = \underline{w}^T C \underline{w} \quad = \quad \underline{w}^T \lambda \underline{w} = \lambda ||\underline{w}||^2$

$\rightarrow \quad ||\underline{w}||^2 = 1$

Oja rule results in final length normalized to 1

It will in fact be the largest eigenvector.

Without normalization each dimension grows exponentially with λ_i

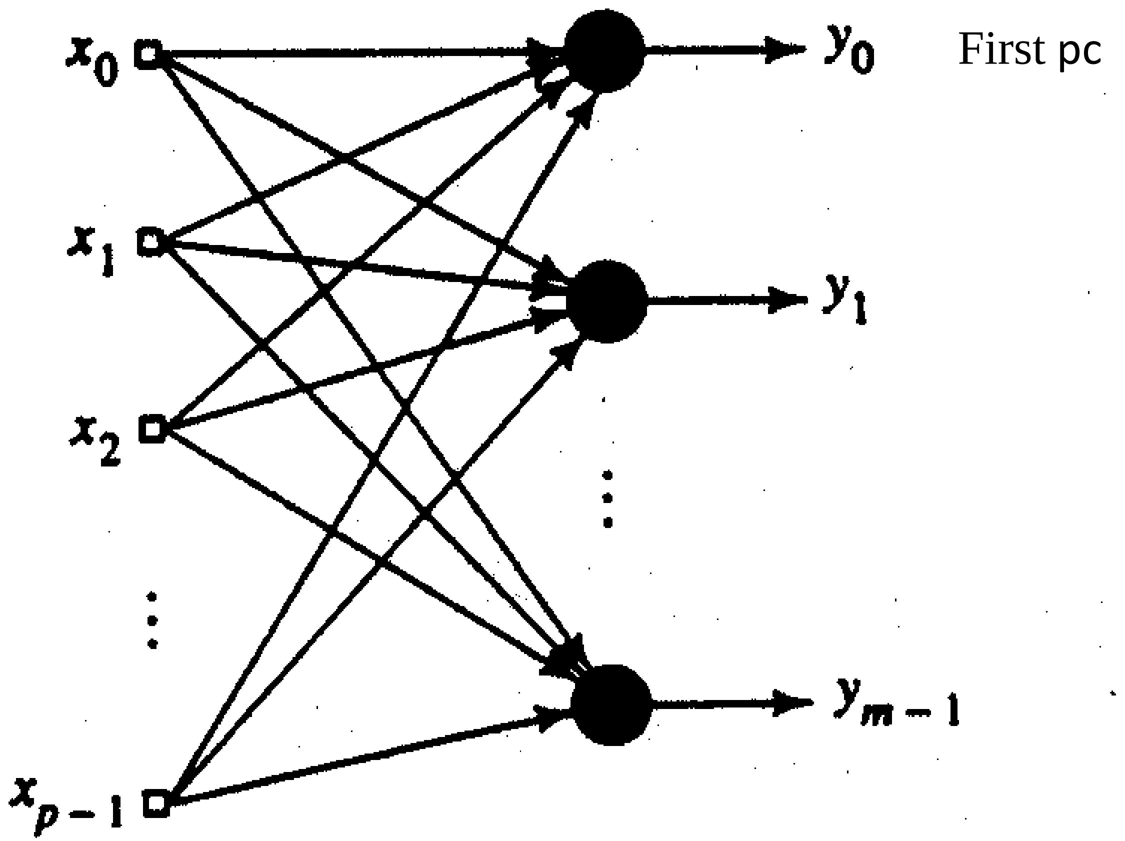
With normalization only the largest λ_i survives

If there is more than one eignevector with the largest eigenvalue it will converge to a combination, that depends on the starting conditions

Following Oja's rule, $\underline{w}$ will converge to the largest eigenvectors of the data matrix XX^T

For full convergence, the learning rate α has to decrease over time. A typical decreasing sequence is $\alpha(t) = 1/t$

Full PCA by Neural Net



- Procedure
  - Use Oja's rule to find the principal component
  - Project the data orthogonal to the first principal component
  - Use Oja's rule on the projected data to find the next major component
  - Repeat the above for $m \leq p$ (m = desired components; p = input space dimensionality)
- How to find the projection onto orthogonal direction?
  - Deflation method: subtract the principal component from the input

Oja rule:

$\Delta \underline{w} = \alpha v(\underline{x} - v\underline{w})$

Sanger rule:

$\Delta \underline{w}_i = \alpha v_i (\underline{x} - \Sigma_{k=1}^{i} v_k \underline{w}_k)$

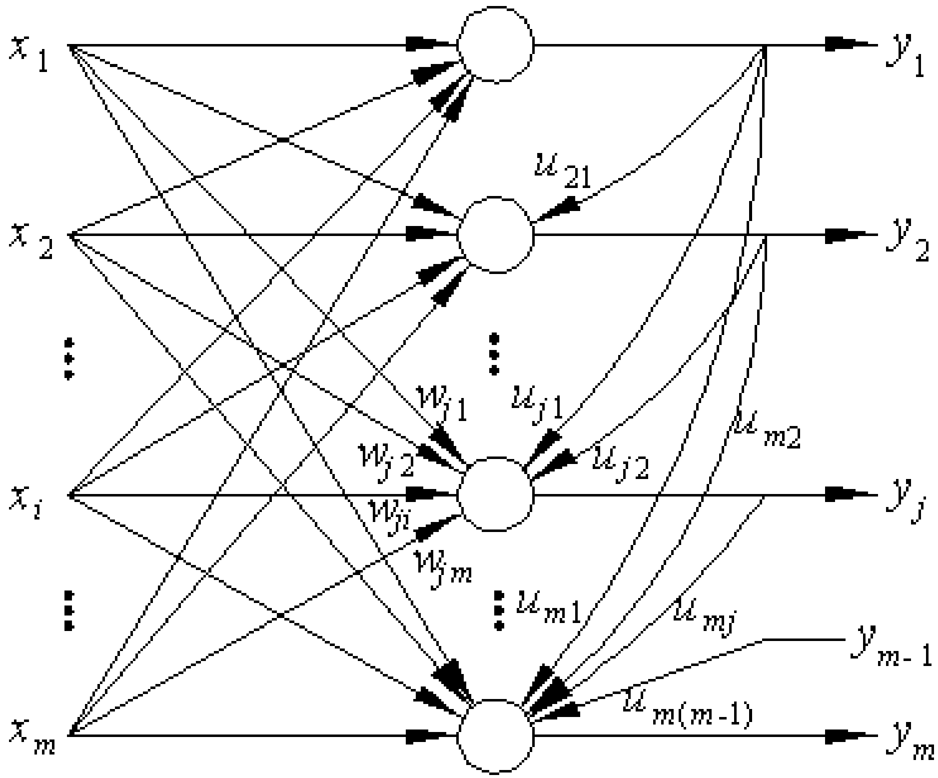
Oja multi-unit rule:

$\Delta \underline{w}_i = \alpha v_i (\underline{x} - \Sigma_1^N v_k \underline{w}_k)$

In Sanger the sum is for k up to j, all previous units, rather than all units. Was shown to converge

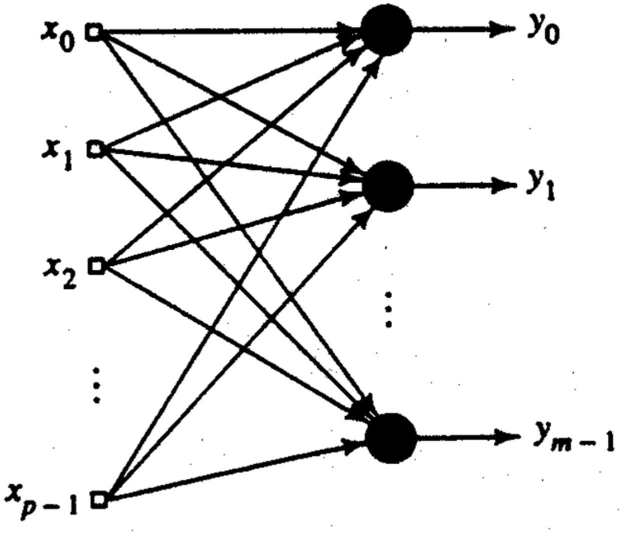
Oja network converges in simulations

Connections in Sanger Network



$$\Delta \underline{w}_j \;=\; \alpha\, v_j \,(x - \Sigma^j\, v_k\, w_k)$$

PCA by Neural Network Models:



- The Oja rule extracts ‘on line’ the first principal component of the data
- Extensions of the network can extract the first m principal components of the data