Artificial Neural Networks
Unsupervised learning: SOM

Jan Drchal

drchajan@fel.cvut.cz

Computational Intelligence Group
Department of Computer Science and Engineering
Faculty of Electrical Engineering
Czech Technical University in Prague

Outline

- Competitive learning.
- Self-organization, Vector Quantization, Cluster Analysis.
- SOM architecture and learning.
- SOM visualizations.
- SOM evaluation.

Competitive Learning

- Nature inspired.
- No arbiter needed – unsupervised learning.
- Individuals (units, neurons) learn from examples.
- System **self-organizes**.
- Now we are going to apply this to **cluster analysis**.

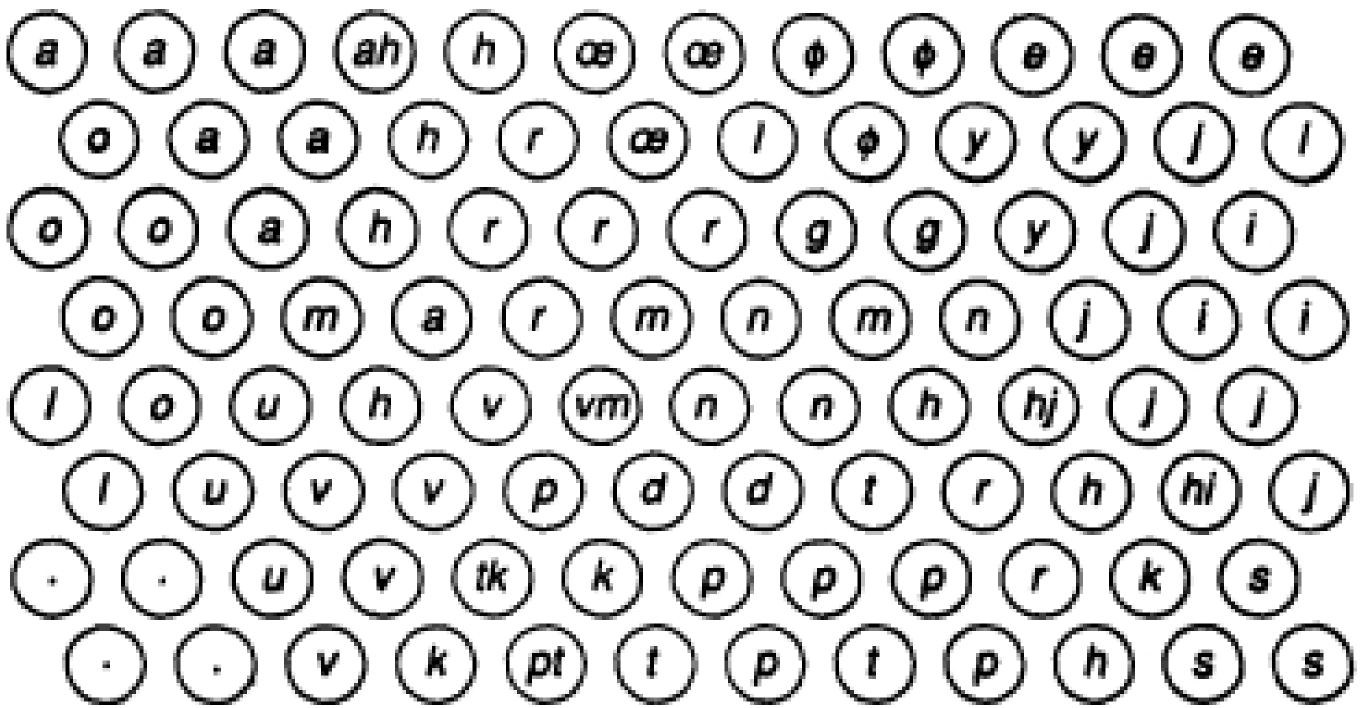
SOM

- SOM = Self Organizing Maps.
- Prof. Teuvo Kohonen, Finsko,

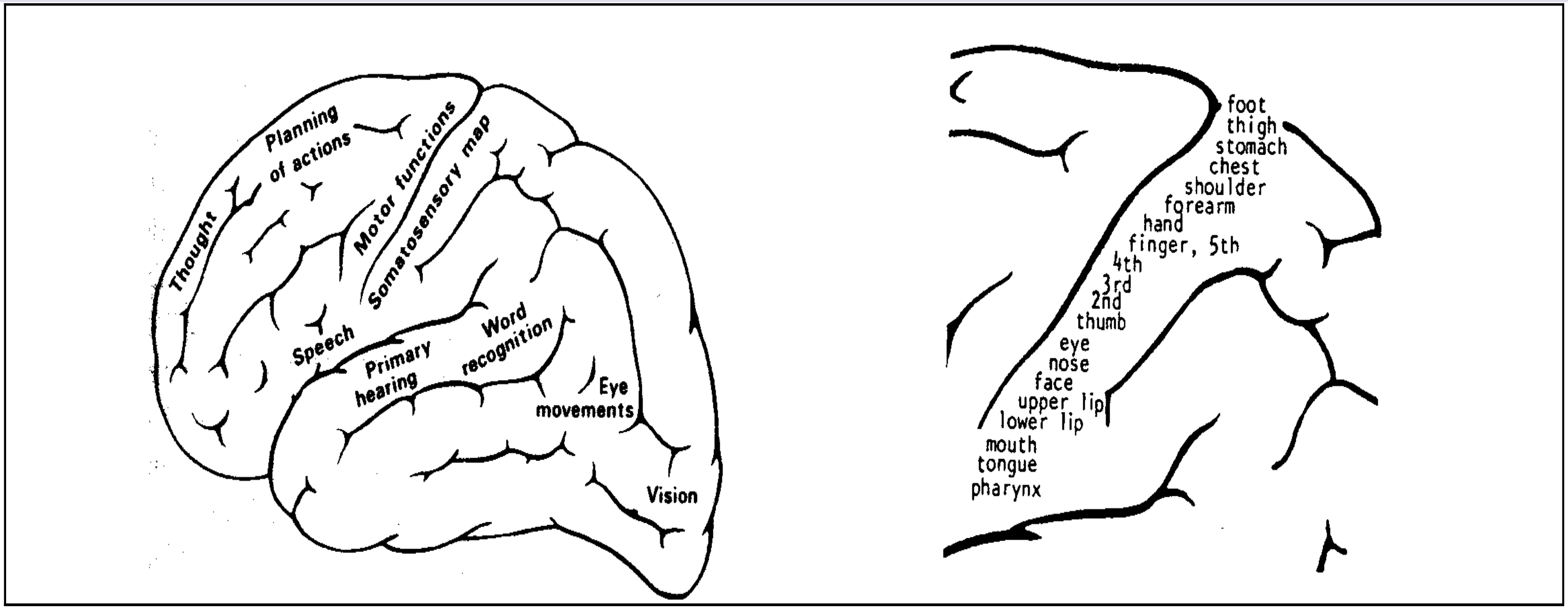
  TU Helsinki, 1981, several thousands scientific publications since...

SOM – Kohonen's Application

- Original application: phonetic “typewriter”:
  - Finish language.

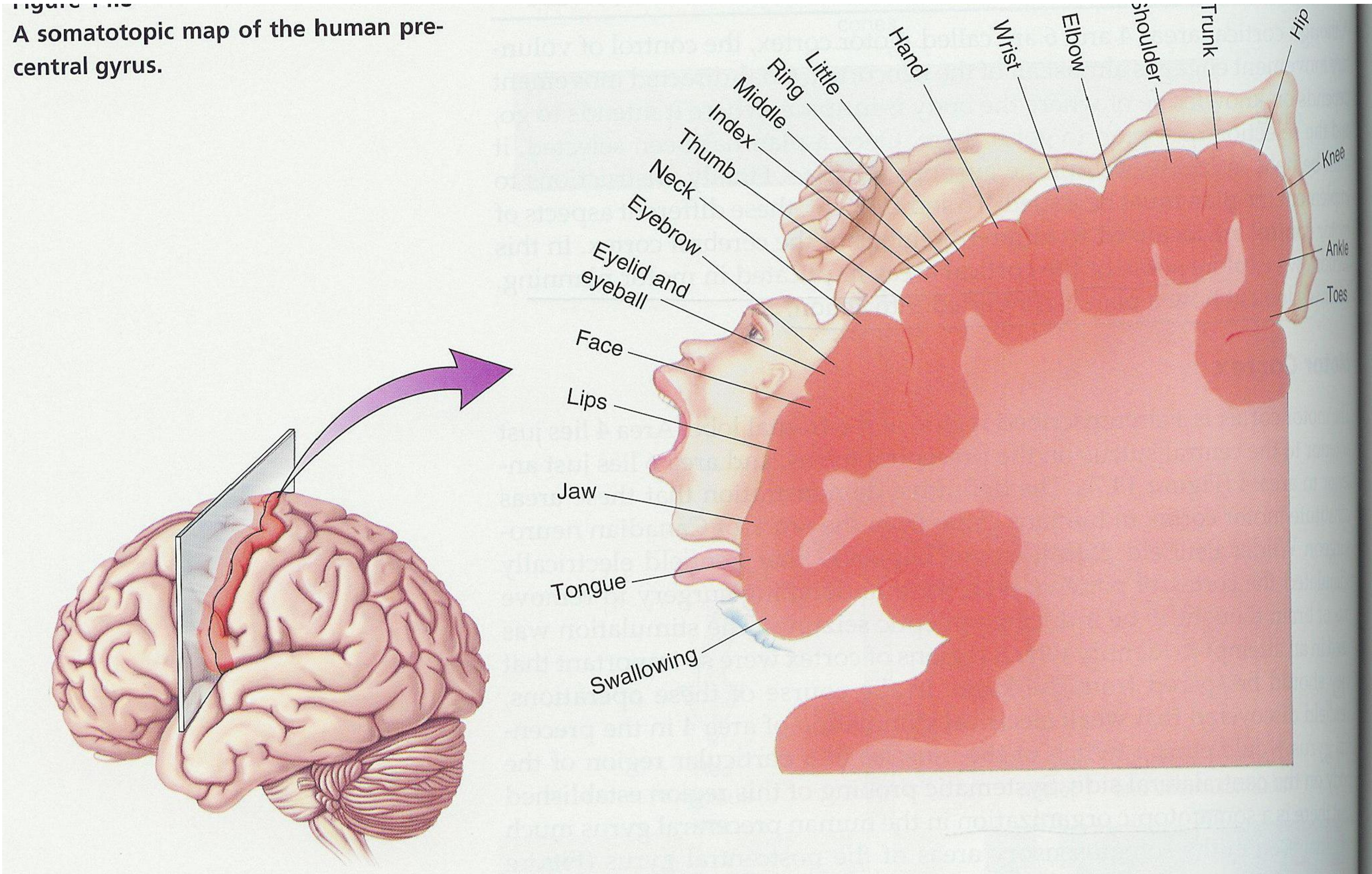


SOM Inspiration



- Brain represents the world in a **topological way**.
- Exterior spatial relations are mapped to similar spatial relations in the brain:
  - i.e. signals from hand and arm are processed nearby.

SOM Inspiration II



A somatotopic map of the human pre-central gyrus.

Bear, Connors & Paradiso (2001). *Neuroscience: Exploring The Brain*. Pg. 474.

SOM Overview

- Single layer, feed-forward.
- Unsupervised, **self-organization**.
- No output, instead **Winner-takes-all**.
- Used for **cluster analysis**.
- Performs **vector quantization**.
- Not a classifier!
  - But can be simply transformed into one by adding another layer.

What is Self-Organization?

- Self-organization of a system is a process which leads to a rise of a quality of its inner configuration while not using any information from outside.
- Self-organization clears up relationships between parts of a system.

What is Cluster Analysis

- Assignment of a set of observations into subsets (clusters).
- A measure of similarity is defined:
  - observations in the same cluster are similar,
  - observations between two clusters are dissimilar.
- Classic cluster analysis works with $\boldsymbol{R}^n$ input space observations.

See: http://en.wikipedia.org/wiki/Cluster_analysis

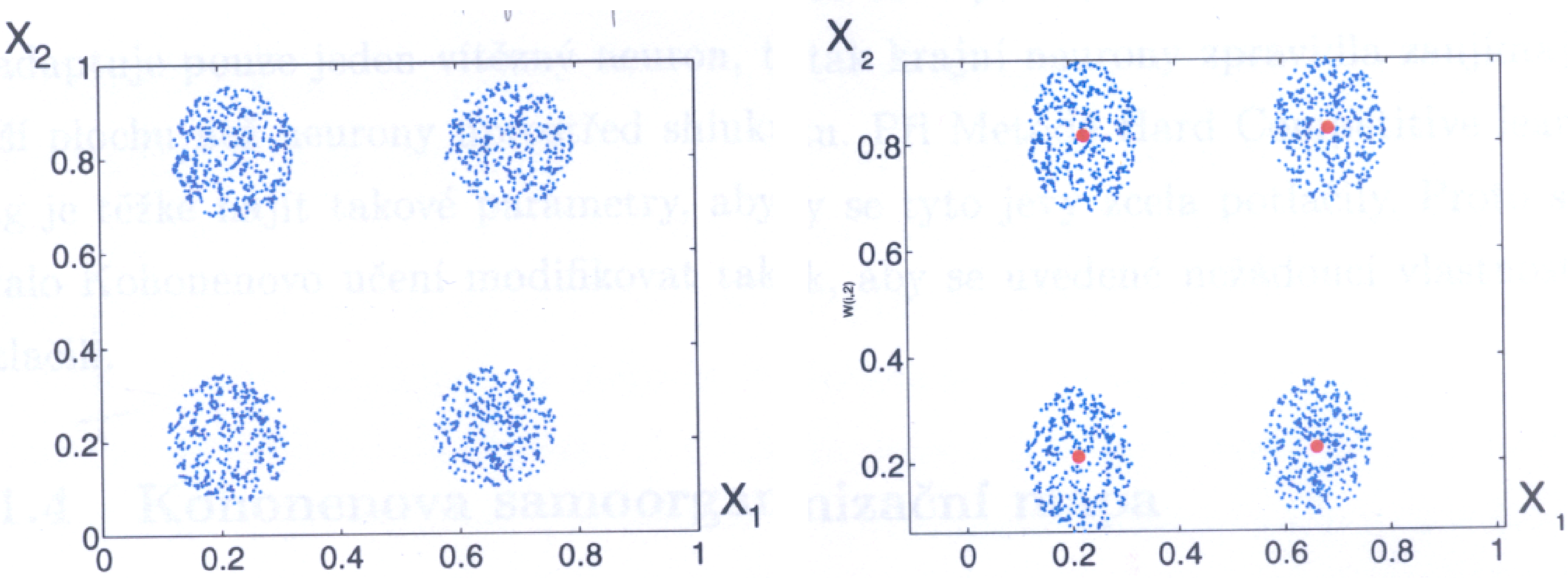
What is Vector Quantization

The goal of Vector Quantization is to approximate the probability density $p(x)$ of real input vectors $\boldsymbol{x} \in \boldsymbol{R}^n$ distribution using finite number of representatives $\boldsymbol{w}_i \in \boldsymbol{R}^n$.

The representative vectors tend to drift there where the data is dense, while there tends to be only a few of them where data is sparsely located. In this manner, the net tends to approximate the probability density of the input data.
Hollmen '96

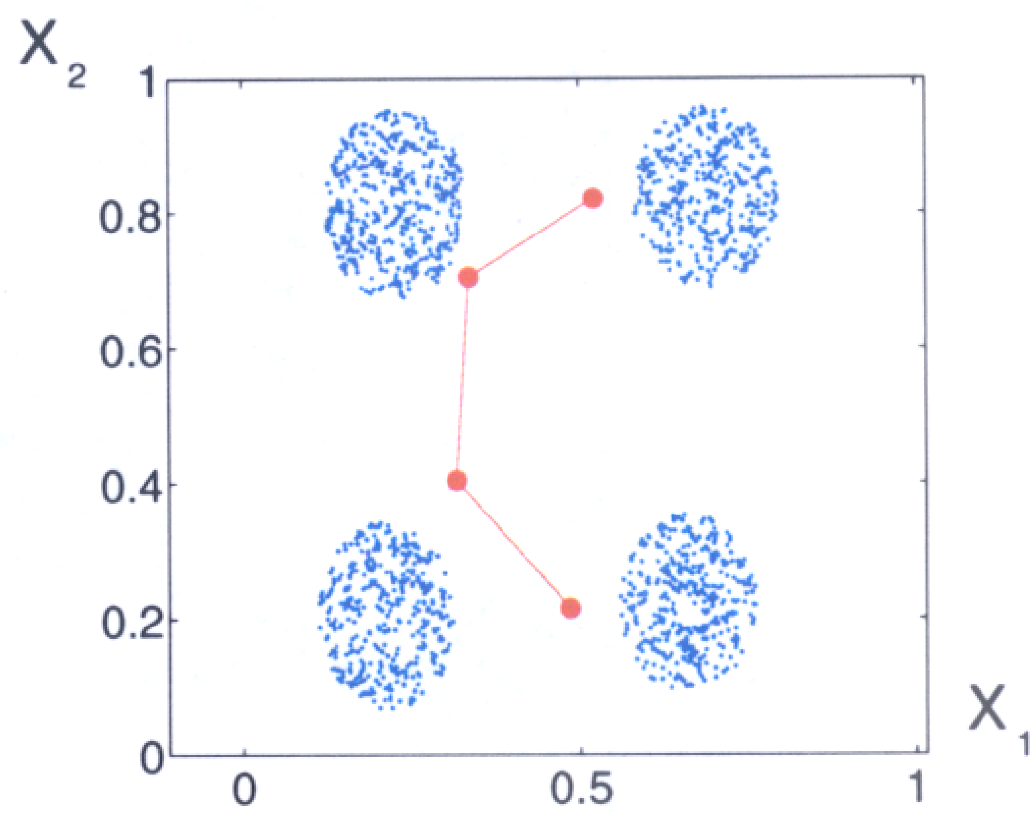
Vector Quantization Example

White points are the input vectors.



Red points are the representatives.

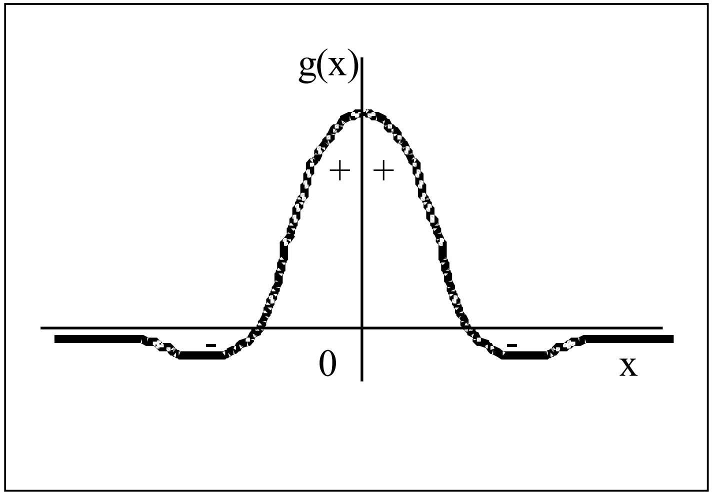
VQ by SOM



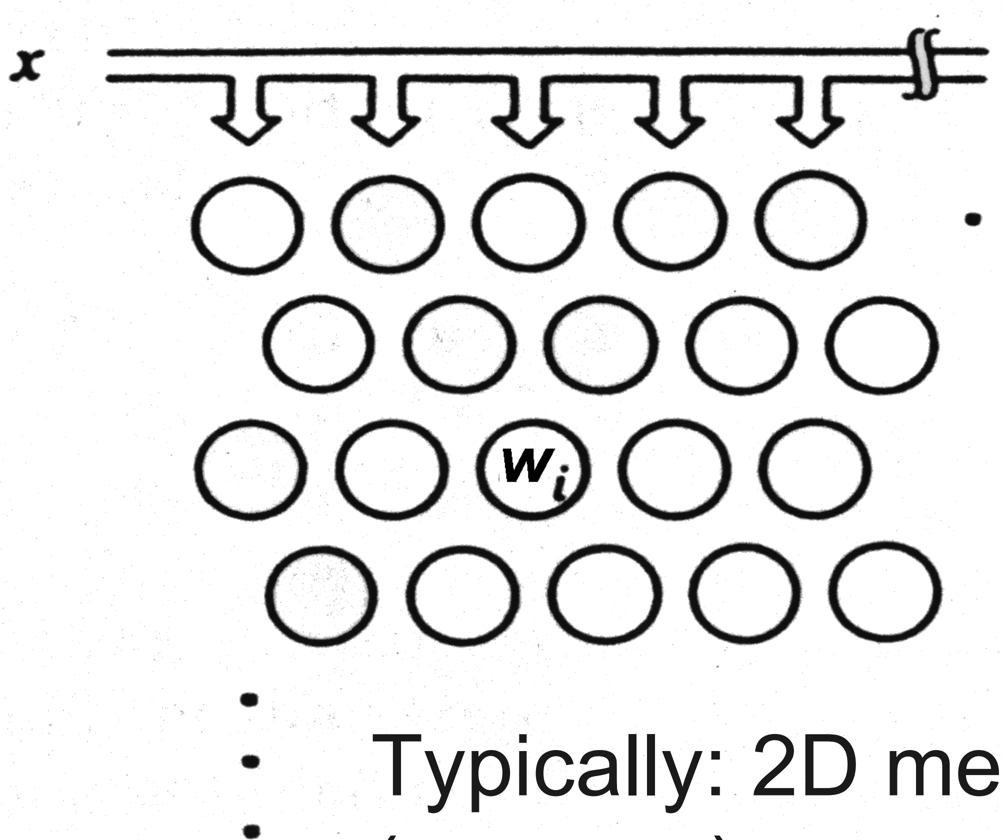
1D SOM of 4 neurons

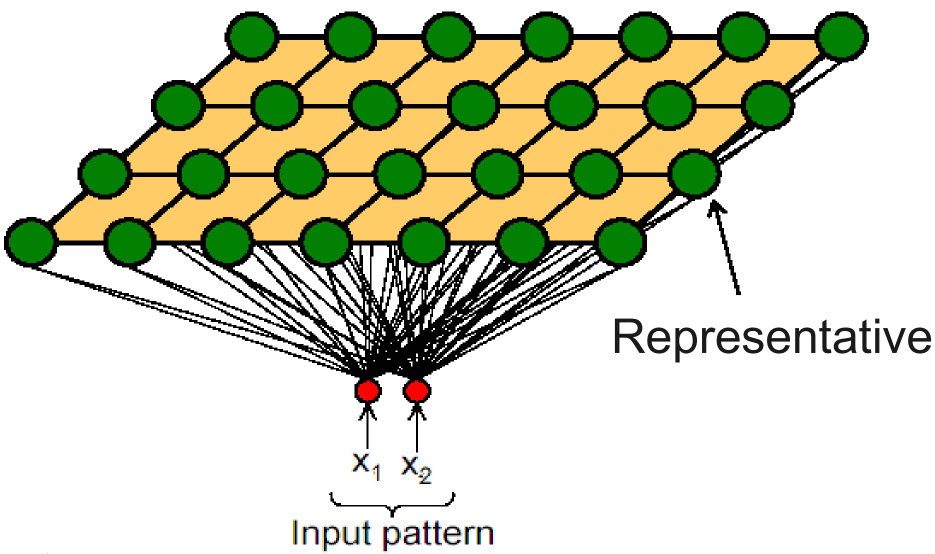
Why the Different Result?

- SOM works with neighbourhood.
- Representatives influence each other.
- They form “elastic”:
  - chain for 1D SOM,
  - mesh for higher dimensions.



SOM Architecture 1/3





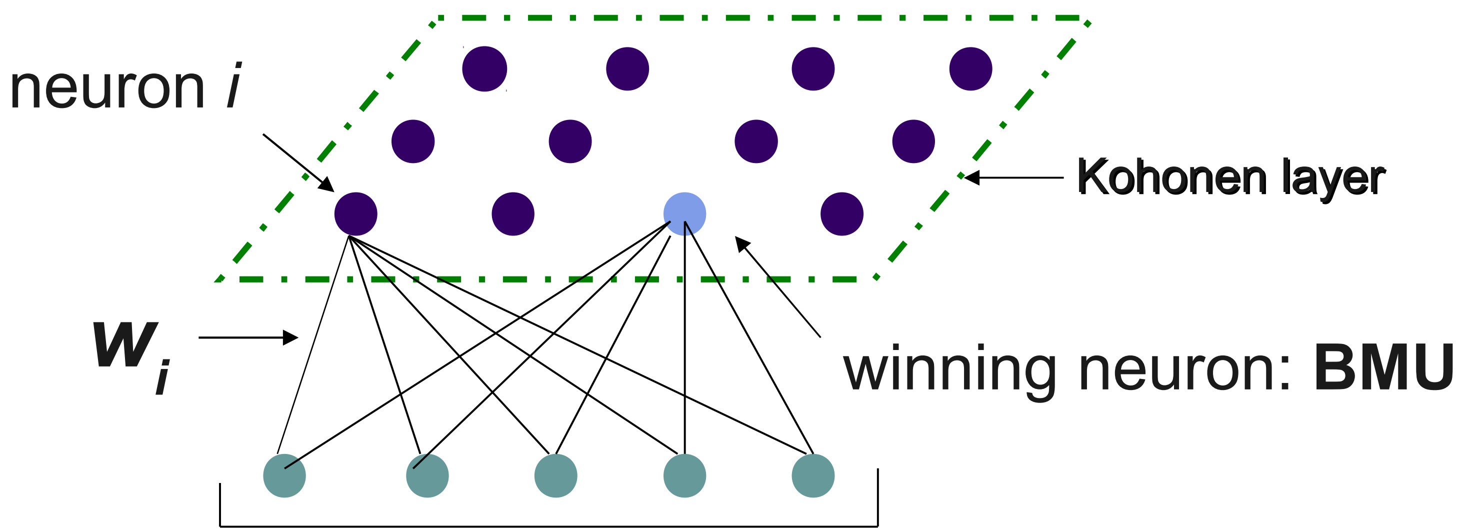
Typically: 2D mesh of representatives (neurons)

SOM Architecture 2/3

- Arrangements:
  - 1D linear quite often,
  - 2D mesh most frequently,
  - 3D (and higher dimensions) exceptionally – problematic visualization.
- The arrangement defines **neighbourhood** of a neuron.
- Kohonen suggests: rectangular SOM!

SOM Architecture 3/3

- Input vector ***x*** has a dimension *N*.
- Each neuron has a weight vector ***w*** of the same dimension *N*.
- Weight vectors of all neurons are compared ***x***.
- The most similar is chosen → BMU (Best Matching Unit).
- BMU becomes a representative of vector ***x***.



SOM Neuron 1/2

Evaluates the similarity of input vector ***x*** and weight vector $\boldsymbol{w}_i$.

Similarity: i.e. Euclidean

The most similar neuron to a input vector is chosen (BMU):

$$j^* = \arg\min_i \left\{ \left\| x - w_i \right\| \right\} ,$$

SOM neuron is a representative of a cluster.

SOM Neuron 2/2

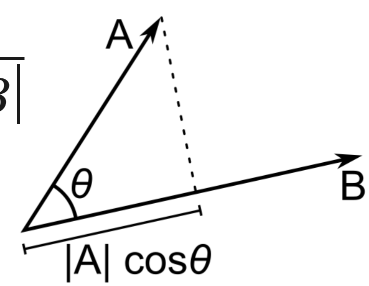
- Note, we don't have to use Euclidean distance.
- We can use directional similarity expressed by the dot product:

$$j^* = \arg\max_i \left\{ x^T(t)\, w_i(t) \right\}.$$

Why max here?

Note:

$$\cos\theta = \frac{A.B}{|A||B|}$$



http://en.wikipedia.org/wiki/Dot_product

Learning SOM

- Initialization (random weights).
- Apply input pattern $\boldsymbol{x} = (x_1, x_2, \ldots x_N)$.
- Compute distances.
- Select BMU – neuron j.
- Adjust weights for all neurons i:

$$w_i(t+1) = w_i(t) + \eta_{ij}(t)\left[x(t) - w_i(t)\right]$$

Neighbourhood function

- Continue with next pattern.

Example

$$\mathbf{X} = \begin{bmatrix} 0.52 \\ 0.12 \end{bmatrix}$$

$$\mathbf{W}_1 = \begin{bmatrix} 0.27 \\ 0.81 \end{bmatrix} \qquad \mathbf{W}_2 = \begin{bmatrix} 0.42 \\ 0.70 \end{bmatrix} \qquad \mathbf{W}_3 = \begin{bmatrix} 0.43 \\ 0.21 \end{bmatrix}$$

$$d_1 = \sqrt{(x_1 - w_{11})^2 + (x_2 - w_{21})^2} = \sqrt{(0.52 - 0.27)^2 + (0.12 - 0.81)^2} = 0.73$$

$$d_2 = \sqrt{(x_1 - w_{12})^2 + (x_2 - w_{22})^2} = \sqrt{(0.52 - 0.42)^2 + (0.12 - 0.70)^2} = 0.59$$

$$d_3 = \sqrt{(x_1 - w_{13})^2 + (x_2 - w_{23})^2} = \sqrt{(0.52 - 0.43)^2 + (0.12 - 0.21)^2} = 0.13$$

The third vector is the winner (BMU).

Example contd.

Let's move the neuron closer to the input pattern: $w_{ij}(t+1) = w_{ij}(t) + \eta(t)\left[x_i(t) - w_{ij}(t)\right]$

$$\Delta w_{13} = \eta\,(\mathrm{t})\,(x_1 - w_{13}) = 0.1\,(0.52 - 0.43) = 0.01$$

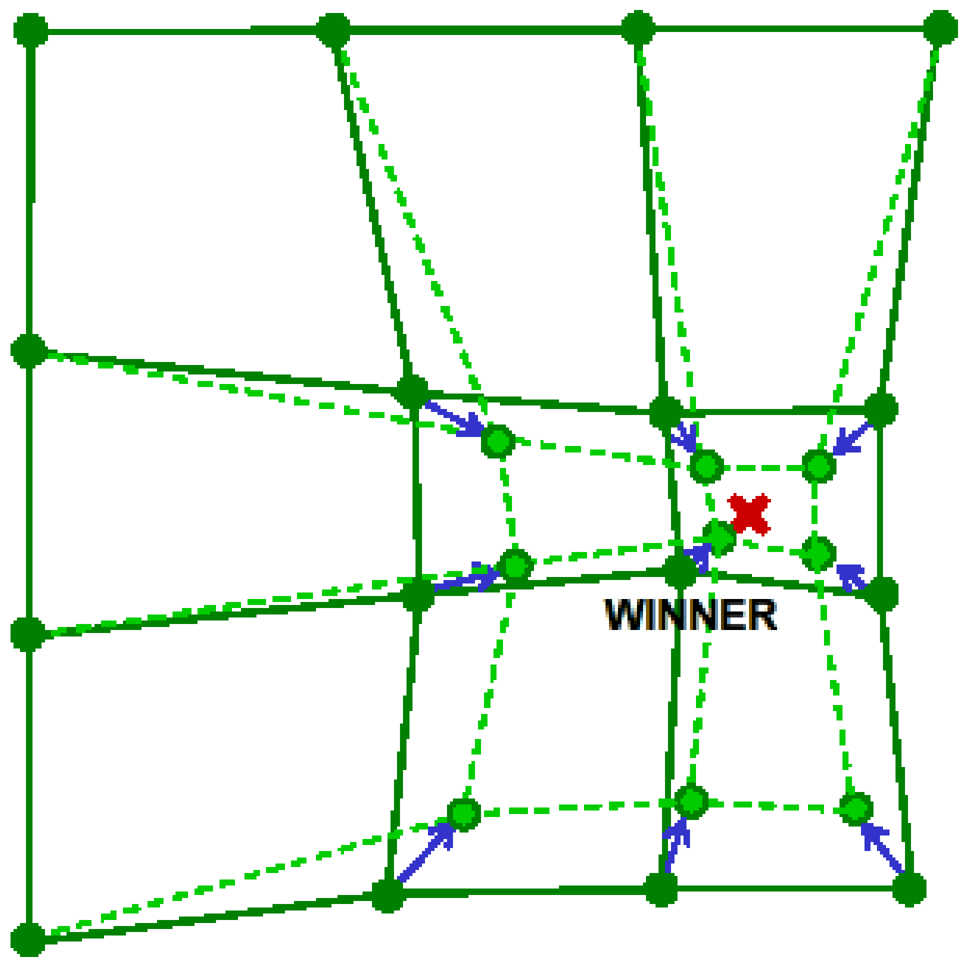
$$\Delta w_{23} = \eta\,(\mathrm{t})(x_2 - w_{23}) = 0.1\,(0.12 - 0.21) = -0.01$$

$$\mathbf{W}_3(p+1) = \mathbf{W}_3(p) + \Delta\mathbf{W}_3(p) = \begin{bmatrix} 0.43 \\ 0.21 \end{bmatrix} + \begin{bmatrix} 0.01 \\ -0.01 \end{bmatrix} = \begin{bmatrix} 0.44 \\ 0.20 \end{bmatrix}$$

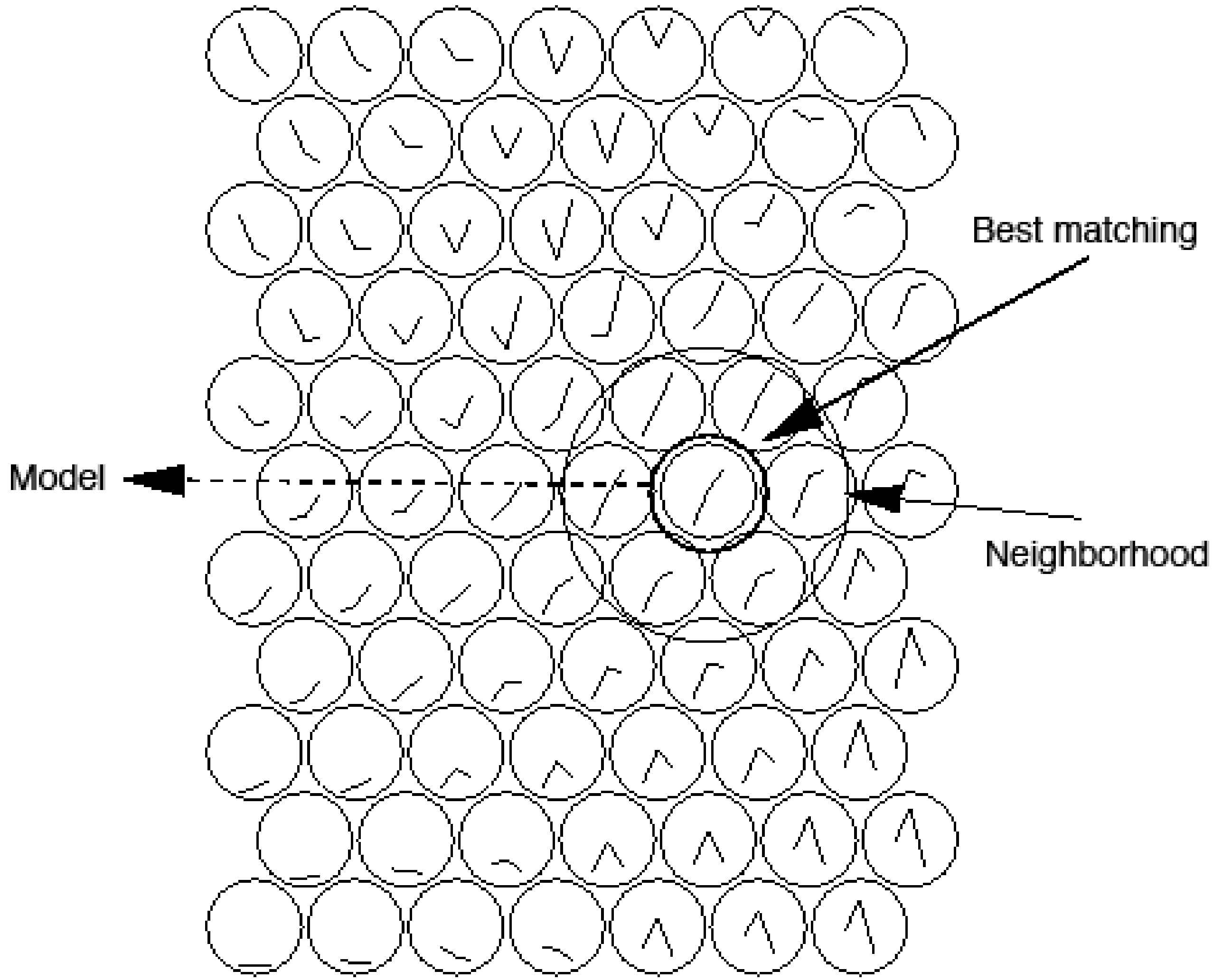
We adjusted only BMU weights.

Here, the winner takes all.

What About Updating Also Neurons in the Neighbourhood?

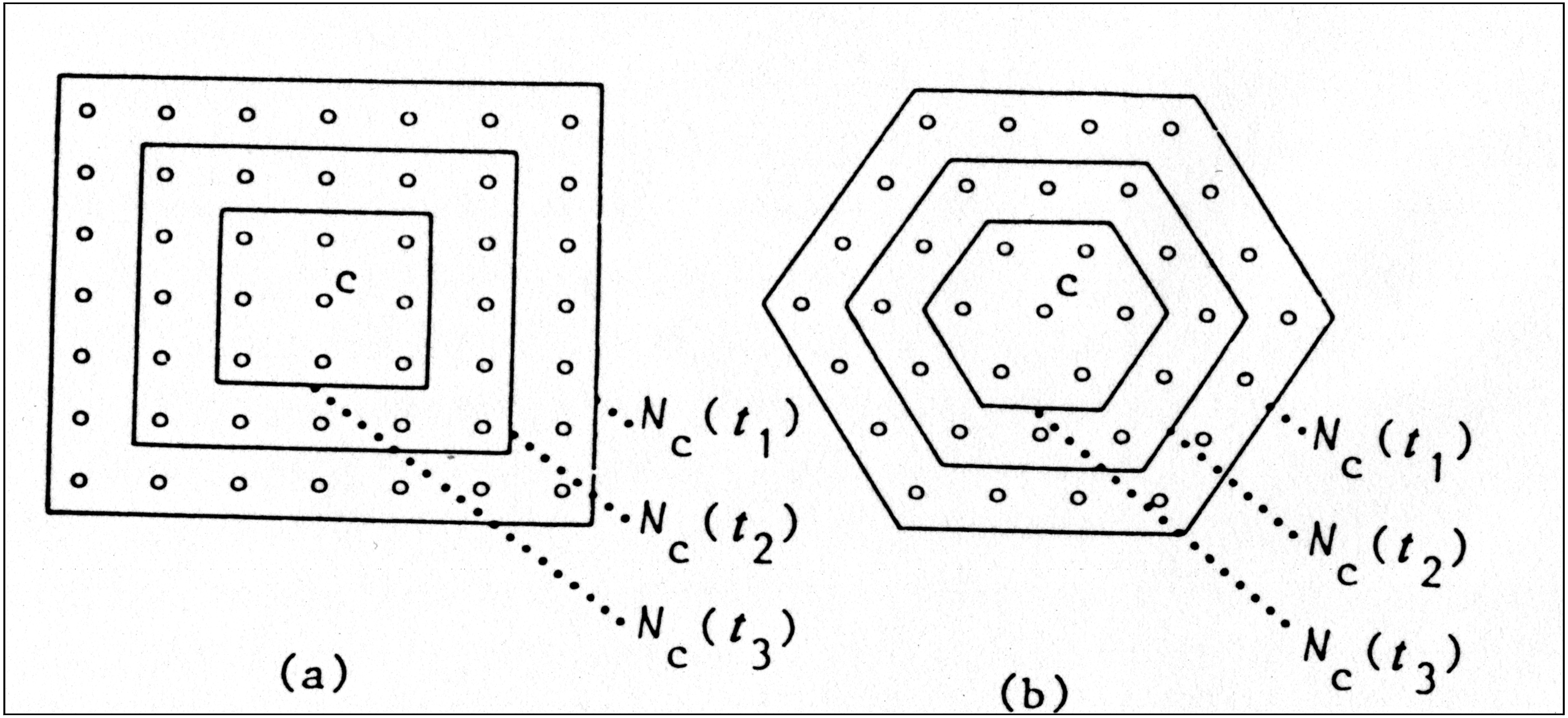


Neighbourhood for Dot-Product SOM



Timo Honkela (Description of Kohonen's Self-Organizing Map)

Common Neighbourhoods



square hexagonal

T. Kohonen: Self Organizing Maps

Learning SOM II

- The neighbourhood plays important role when learning SOM:
  - topological arrangement,
  - neighbour distances.
- Neighbourhood changes in time:
  - its “diameter” decreases (to zero).
- The change is realized by neighbourhood function $\eta(t)$.

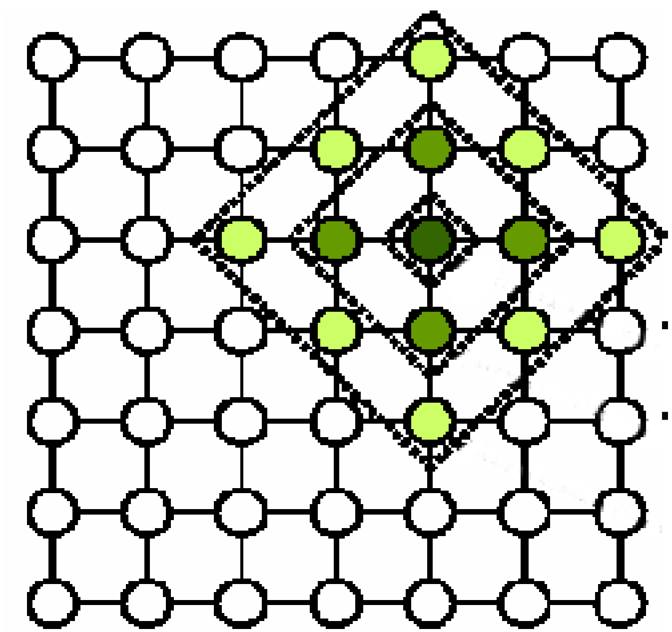
Gaussian Neighbourhood

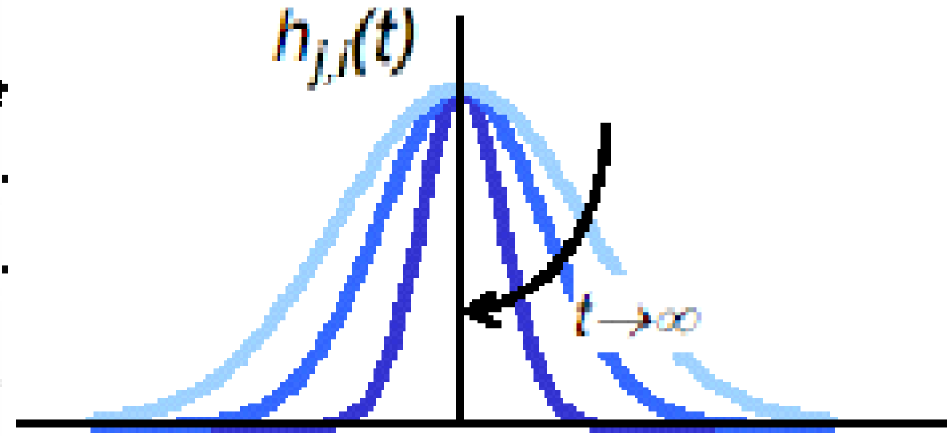
- Neighbourhood function for neuron *i*.

$$\eta_{ij^*}(t) = \alpha(t)\,.\exp\left(-\frac{\left\|r_{j^*} - r_i\right\|^2}{2\sigma^2(t)}\right),$$

- Where j^* is the BMU,
  r the position of neuron in map,
  and function $\alpha(t)$: learning rate.
- The *exp* expression represents neighbourhood shape.

Gaussian Neighbourhood





Distance related learning

Neighbourhood Related Functions

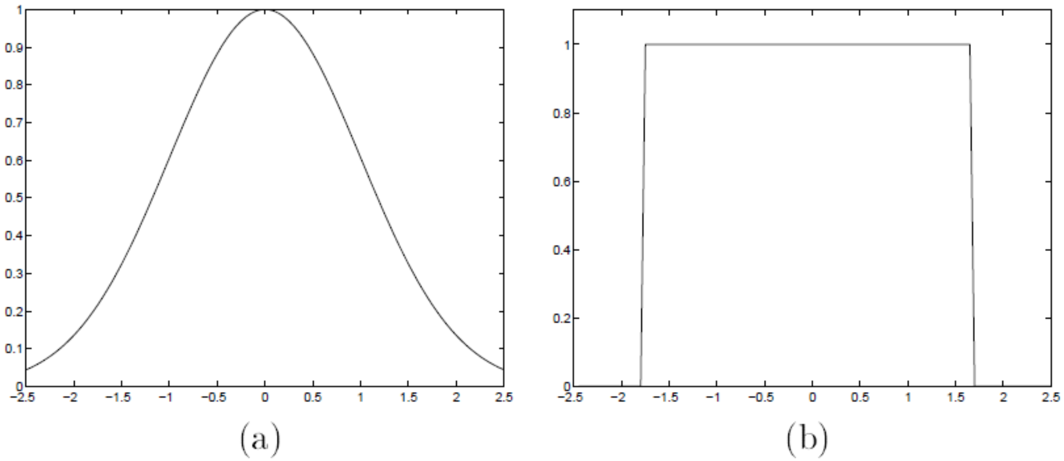


Figure 2.6: Neighborhood function values

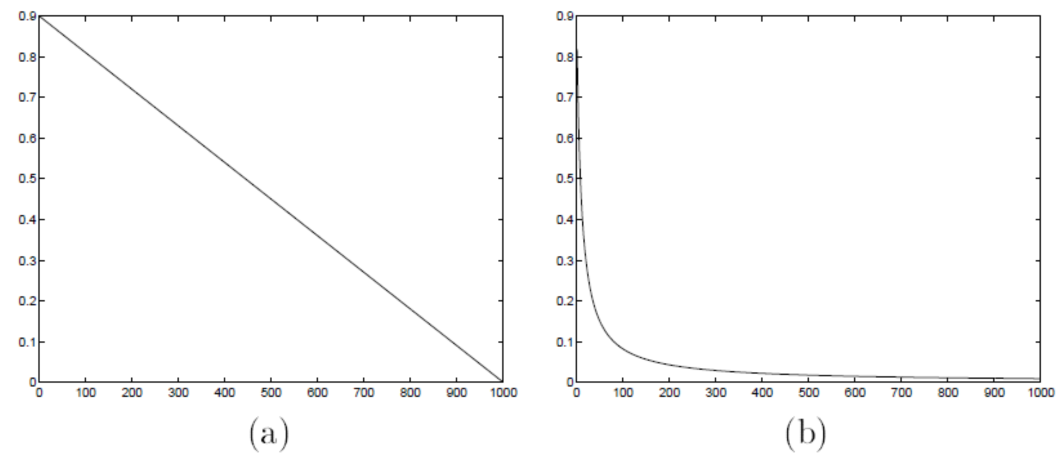
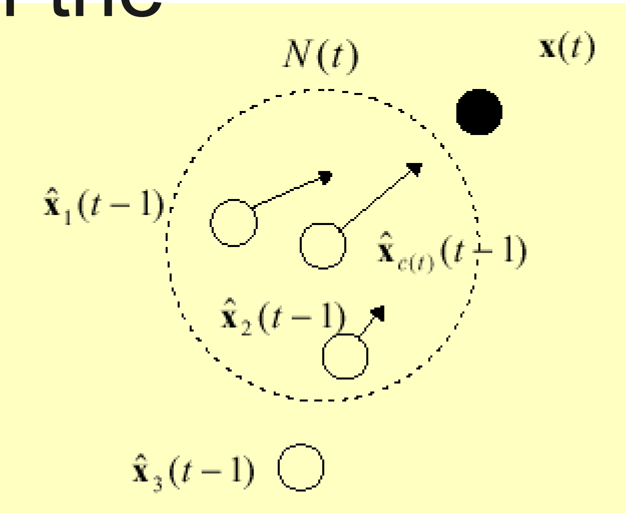


Figure 2.7: Learning rates as functions of time

Hollmen '96, MSc.

Learning Process

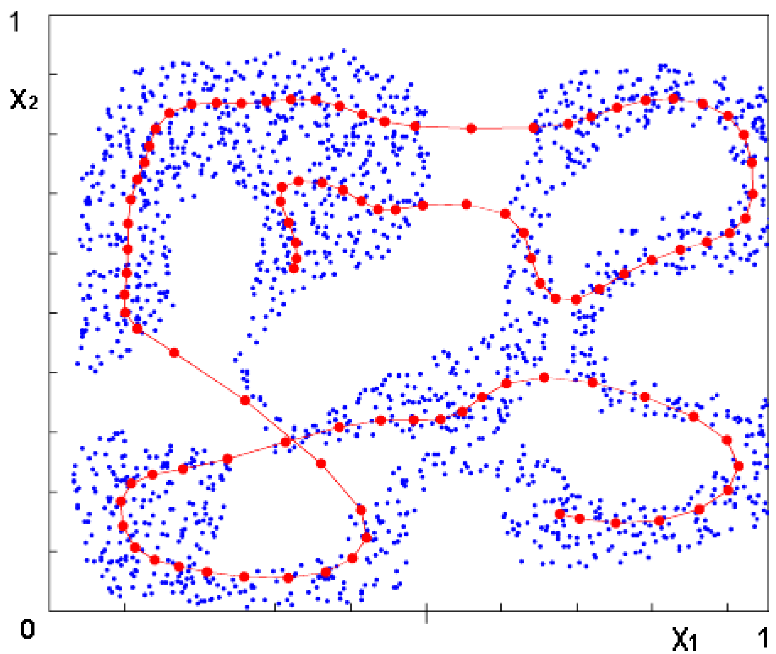
- During the learning the BMU (and its neighbours) is adapted to get closer to the input pattern which have caused its activation.
- Neurons are moving towards the input pattern.
- What influences the magnitude of the approach?



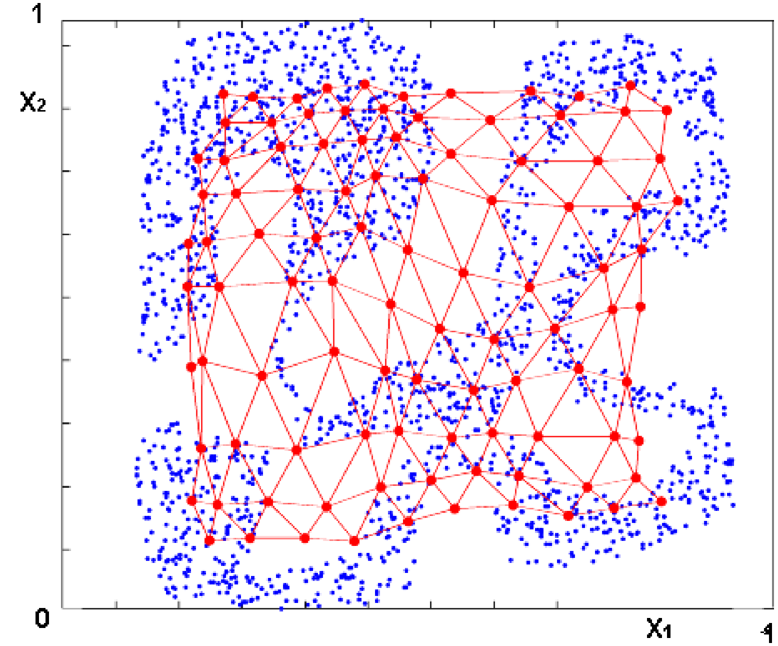
SOM Applications

- To visualize data.
- To cover the input space by representatives.

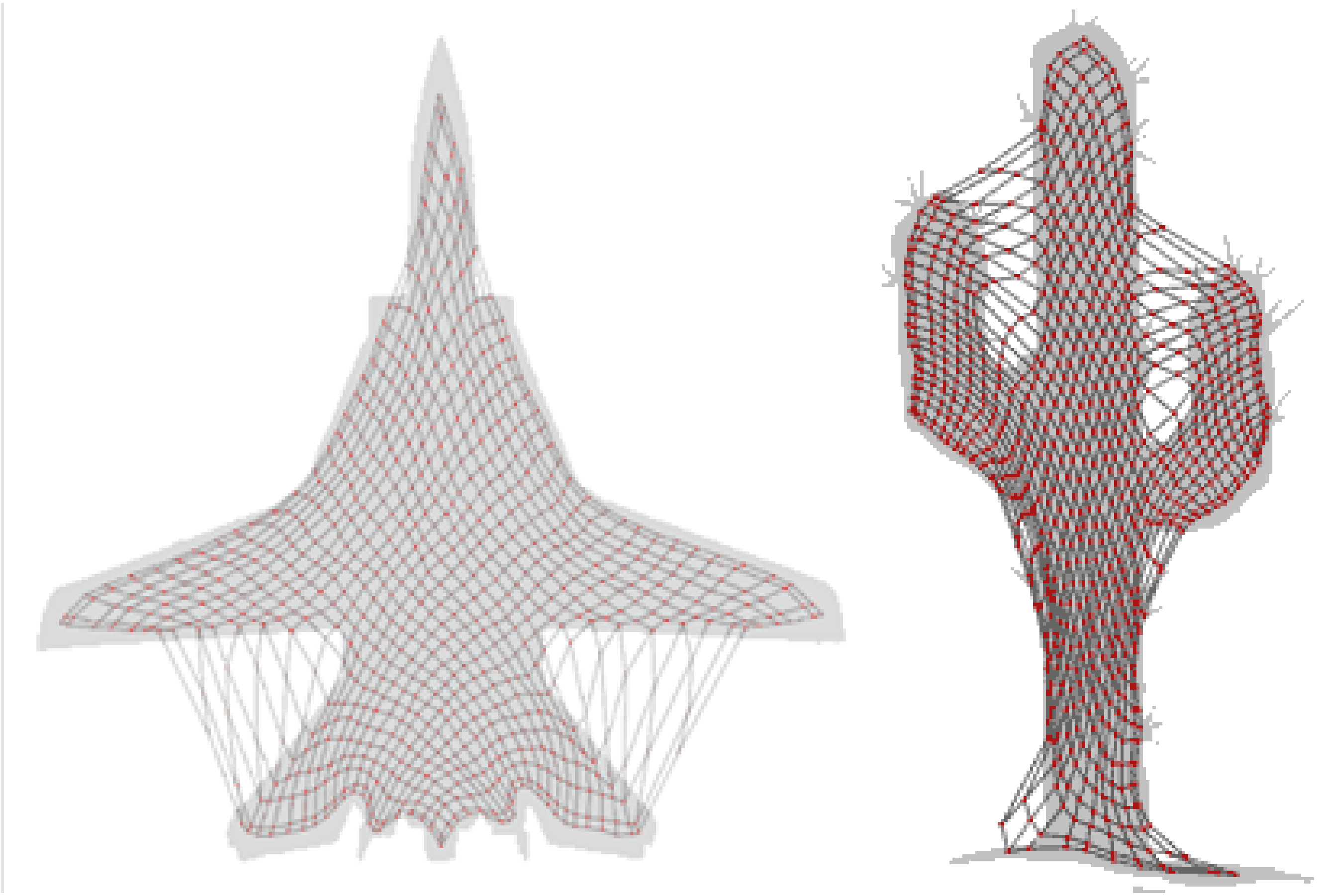
1D



2D

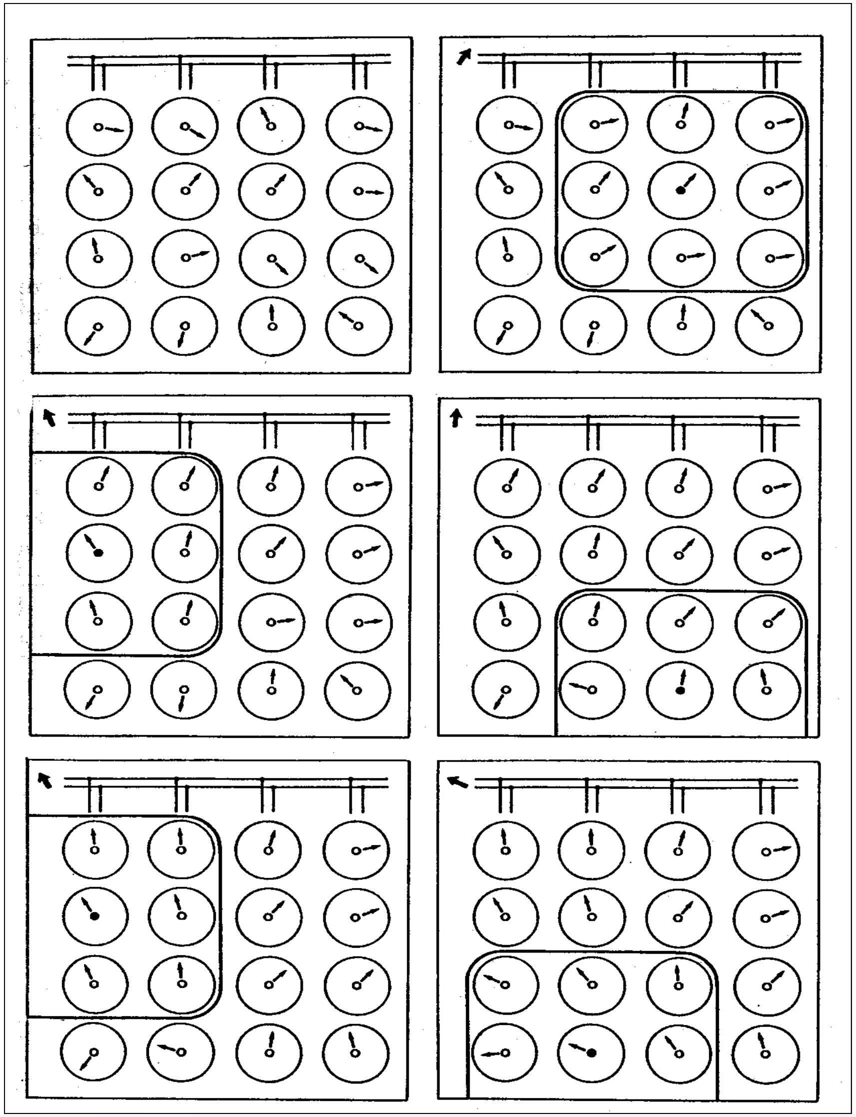


Or …



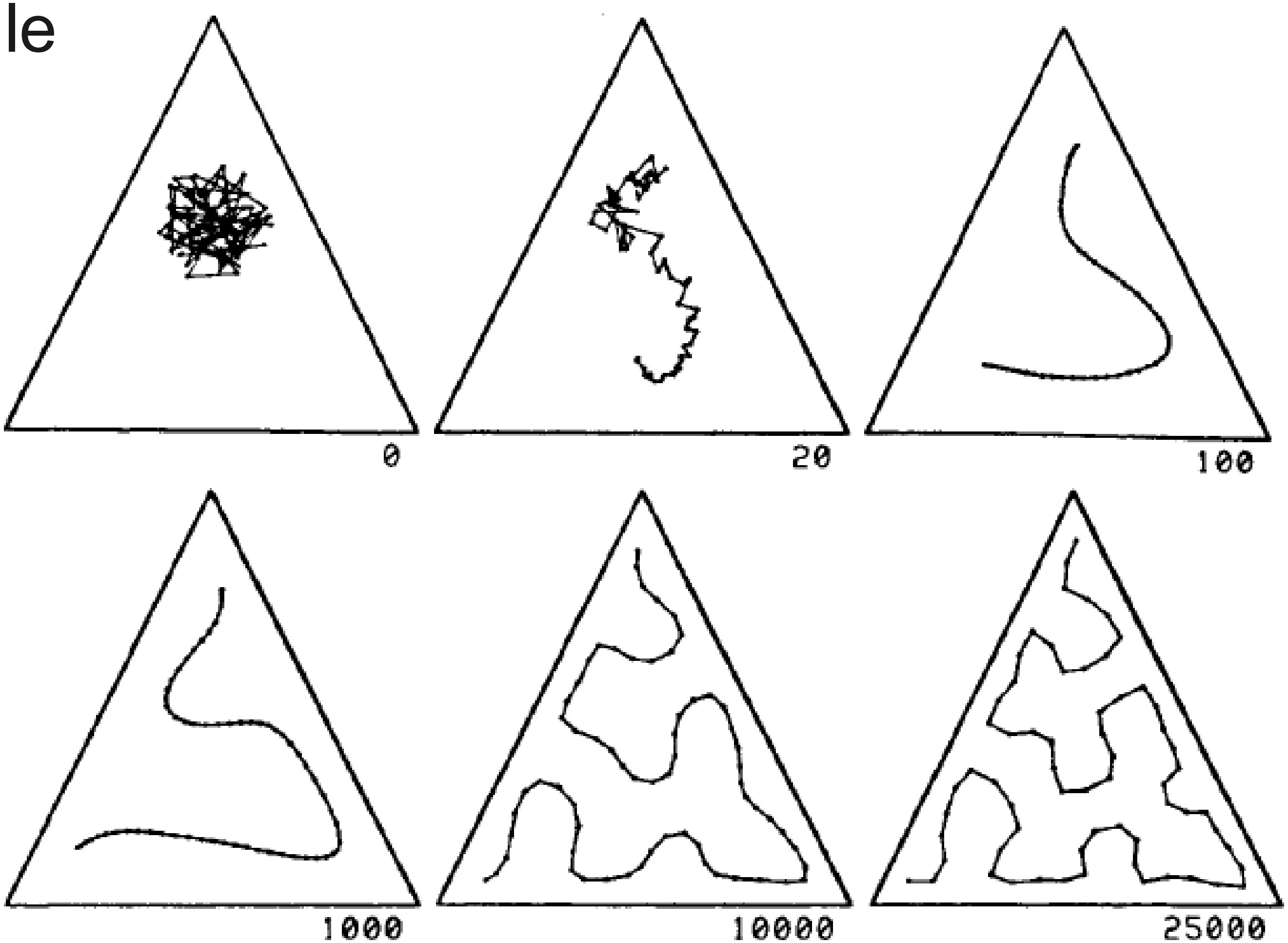
Slide by Johan Everts

Example: Learning Dot-Product SOM



More Examples

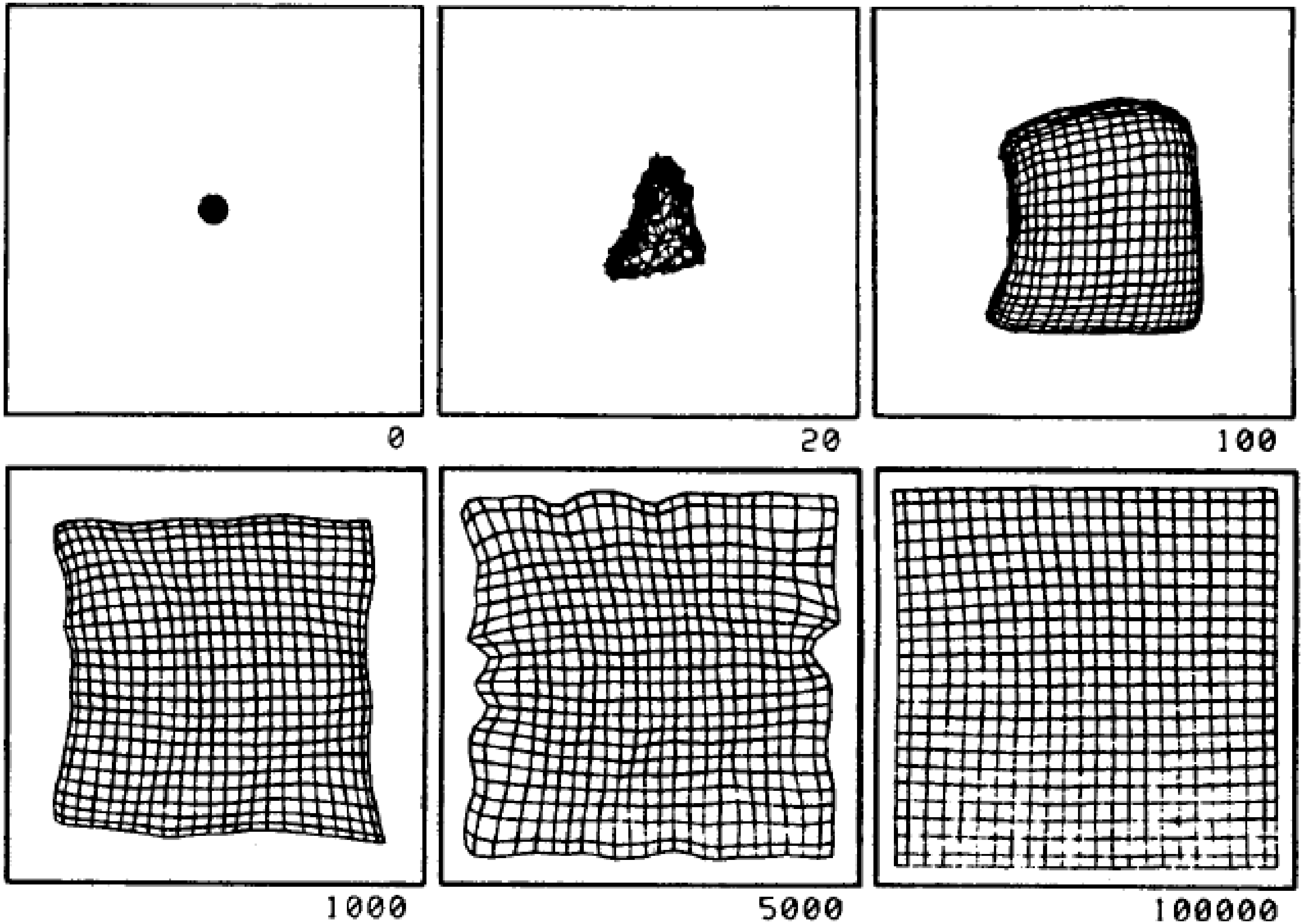
Covering a triangle by 1D SOM.



T. Kohonen: Self Organizing Maps

More Examples contd.

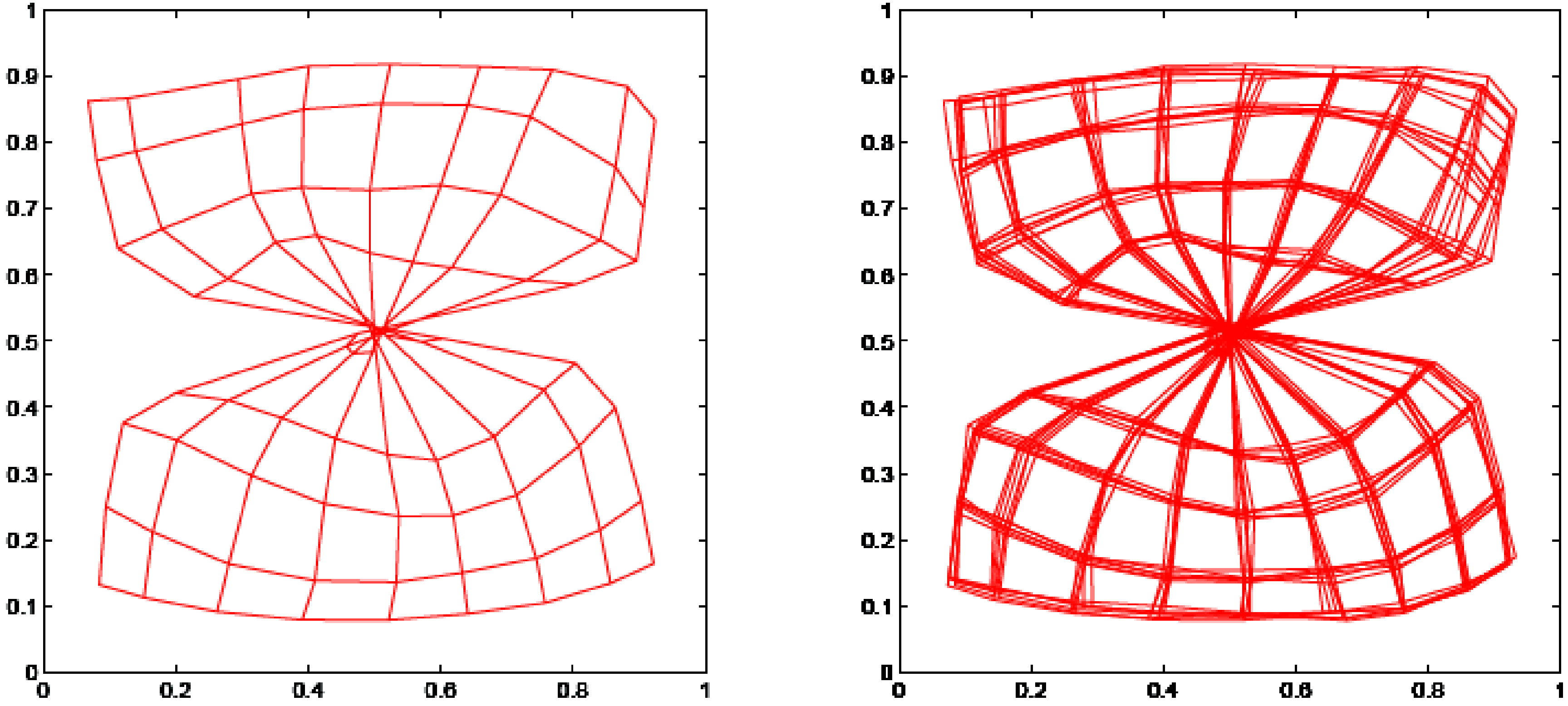
Covering a square by 2D SOM.



T. Kohonen: Self Organizing Maps

Possible Problem: Knots

- This problem is not likely to be corrected by further learning if the *plasticity* is low:



Rojas: Neural Networks - A Systematic Introduction

What is the Cause?

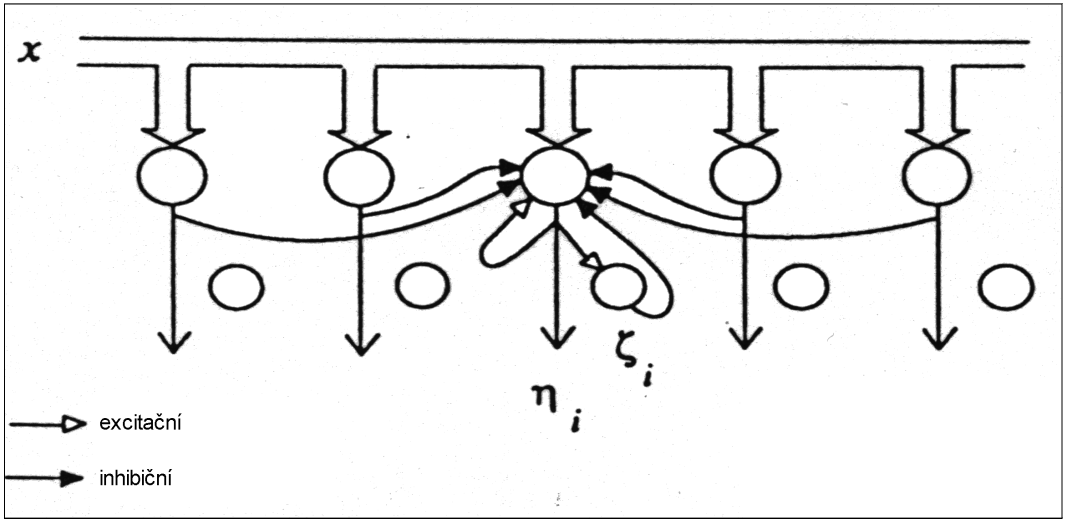
- There are many:
  - Random initialization of weights → we are unable the change bad initial orientation of vectors.
  - Choice of a neighbourhood function.
  - Scheduling of neighbourhood modification in time.
  - Input data of course...

What Can Help?

- Same weights for all neurons initially → each neuron has a same chance to represent a pattern.
- Add random noise to input patterns at start.
- Lateral inhibition...

Lateral Inhibition

- When choosing the BMU we do not pick isolated winner.
- The choice does not depend on an activation of a single neuron but also on activity of its neighbours...



Lateral Inhibition II

$$I_j = I_j^l + I_j^f = d_j + \sum_k g_{jk} I_k$$

j-th neuron response

local response

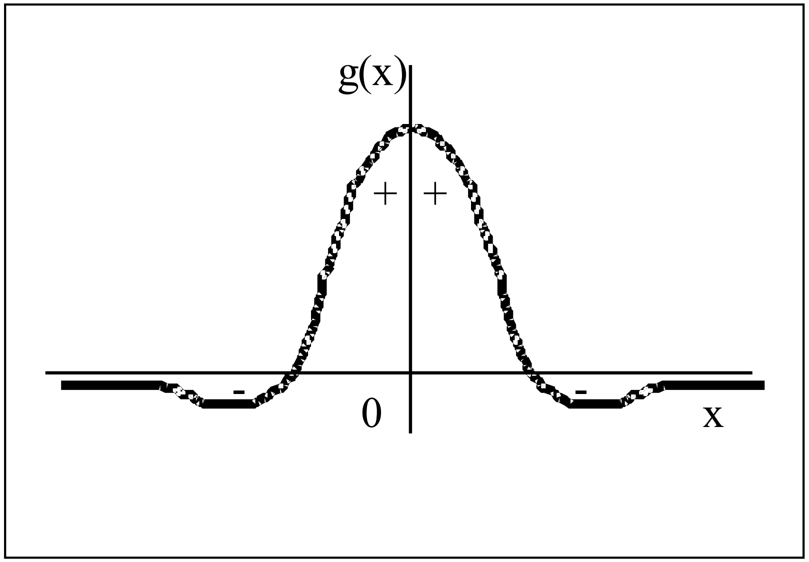
neighbourhood response

distance from input vector

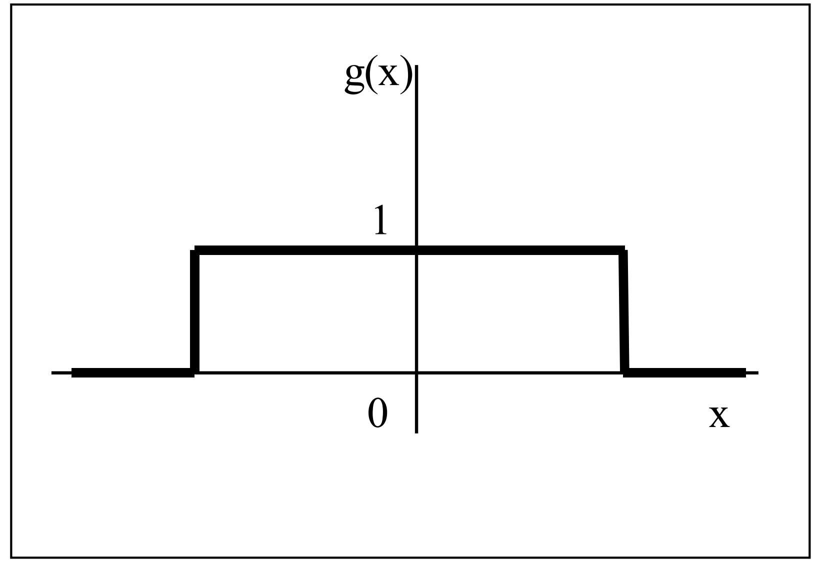
neighbours

lateral inhibition interaction

Lateral Inhibition Functions



biological



simplified

SOM Visualization

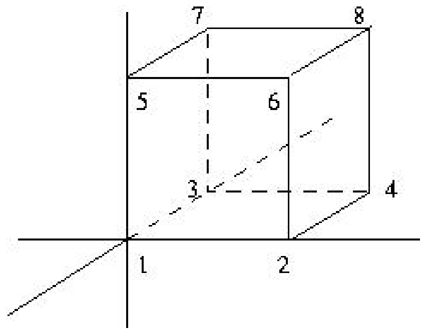
- How to visualize representatives?
- Weight dimension = input vector dimension.
- How to show in 2d?
    - U-matrix,
    - P-matrix,
    - PCA (linear projection),
    - Sammon's projection (non-linear).

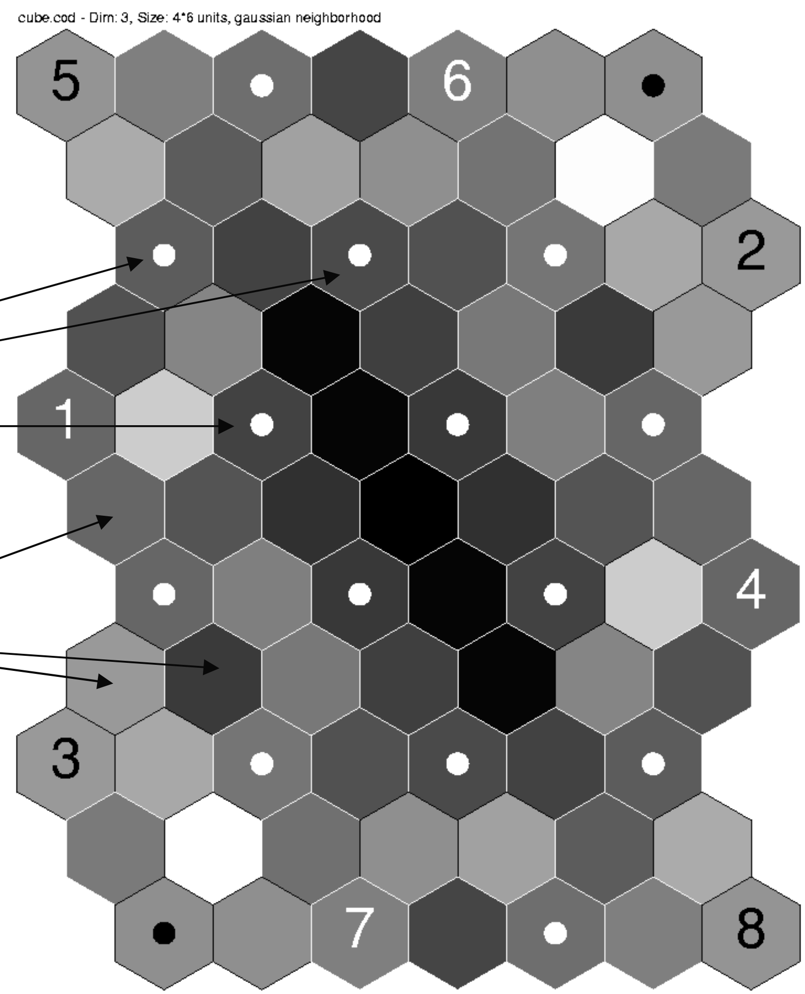
U-matrix (UMAT)

- Visualizes distances between neurons:
  - Dark coloring between neurons → large distance.
  - Light → close in input space.
- Dark gaps separate clusters.
- Neuron colour reflects the distance of its weight vector to all other weight vectors, again:
  - dark → large distance,
  - light → close distance.

U-matrix Example

data





neurons

distance between adjacent neurons

P-matrix (Pareto Density Estimation)

- Shows the number of input space vectors which belong to a sphere centered in the neuron's weight vector.
- Visualizes data density.
- Neurons with high value belong to “dense” areas of input space.
- Neurons with low value are “lonesome”.
- Valleys separate clusters (“plateaus”).

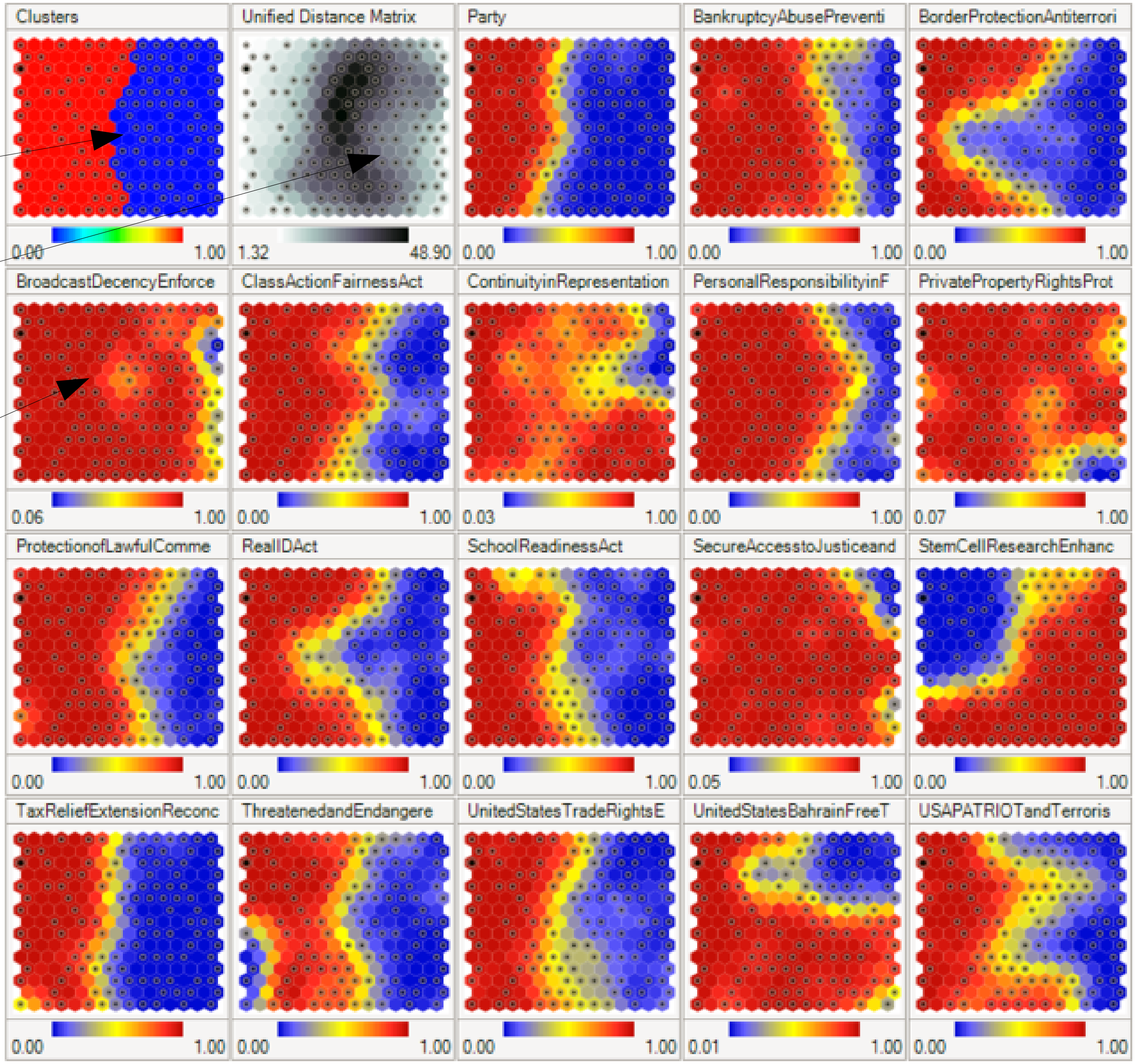
Feature Plots

clustering

UMAP

feature plot shows a value of a single component (feature) of a weight vector

can be used to check if two components correlate



http://en.wikipedia.org/wiki/Self-organizing_map

Drawbacks of UMAT, PMAT

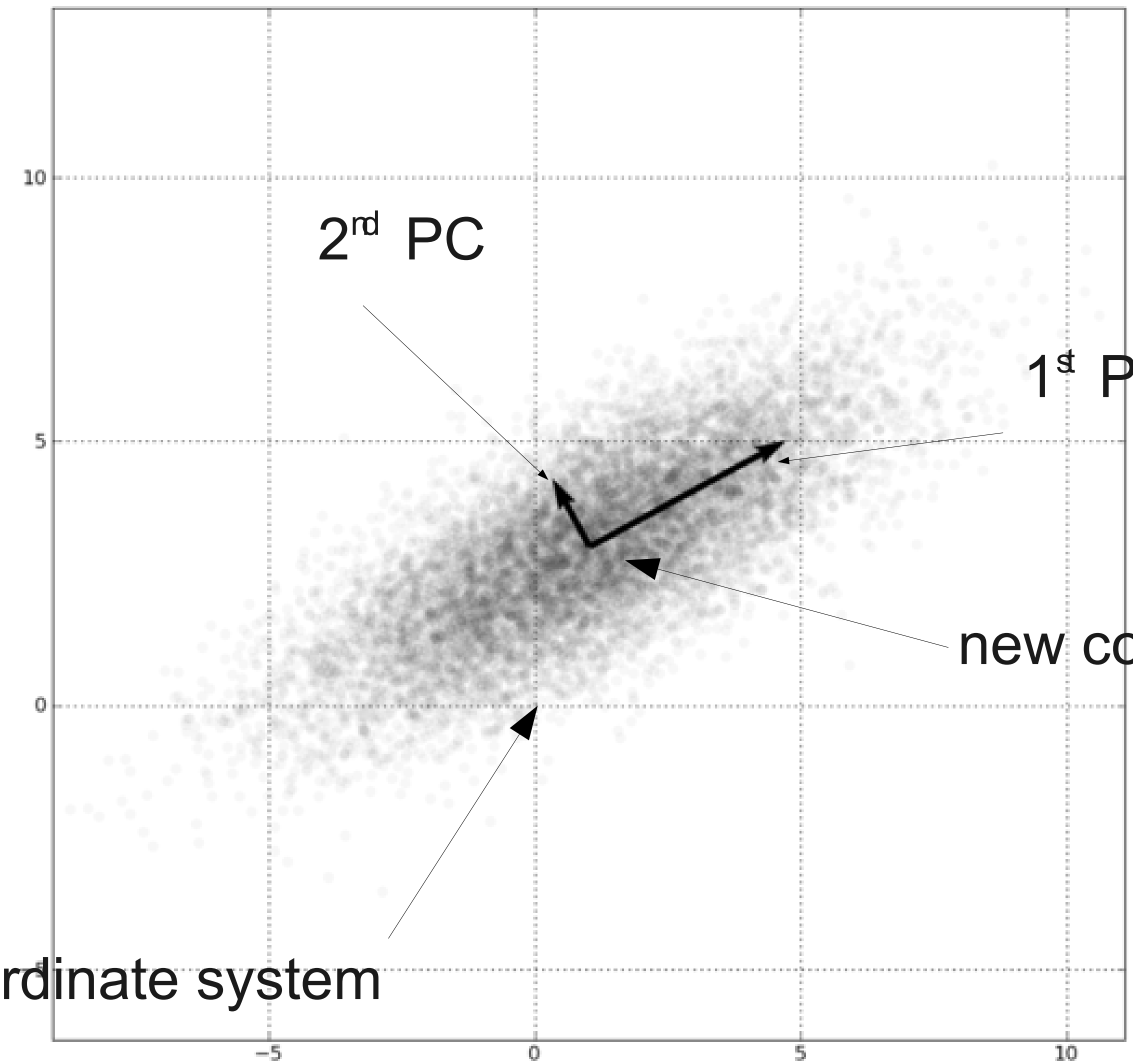
- Only distances between neighbours.
- New learning on the same data may give different results: (i.e. 90 degrees rotation)
- Not intuitive.

- **How can we show high-dimensional data in 2D(3D) keeping notion of original distances?**

PCA

- Principal Component Analysis.
- Linear transformation to a new coordinate system such that:
  - 1st coordinate (principal component)→ greatest variance by any projection of the data
  - 2nd coordinate → 2nd greatest variance
  - etc.
- Dimension reduction → use only *N* first coordinates, **throw the rest away**...

Principal Components Example



http://en.wikipedia.org/wiki/Principal_component_analysis

Sammon's Projection

- Non-linear reduction of higher-dimensional space to lower-dimensional space.
- Tries to preserve distances.

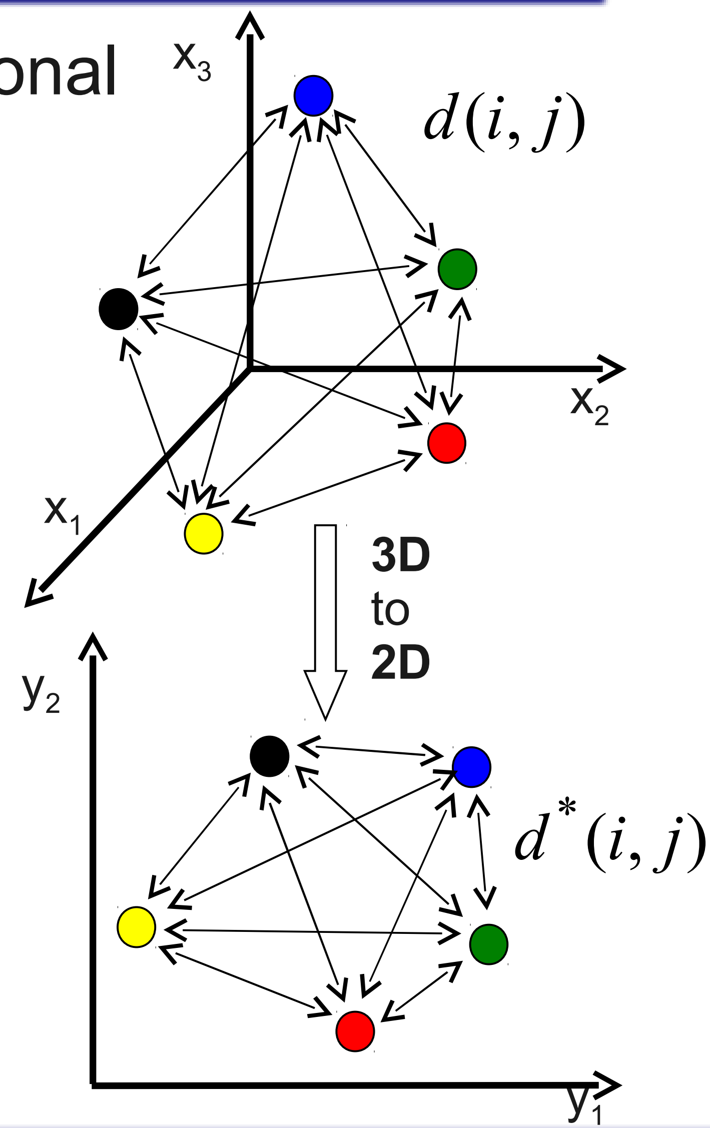
energy function → low for similar distances in both spaces.

distance in high dim. space

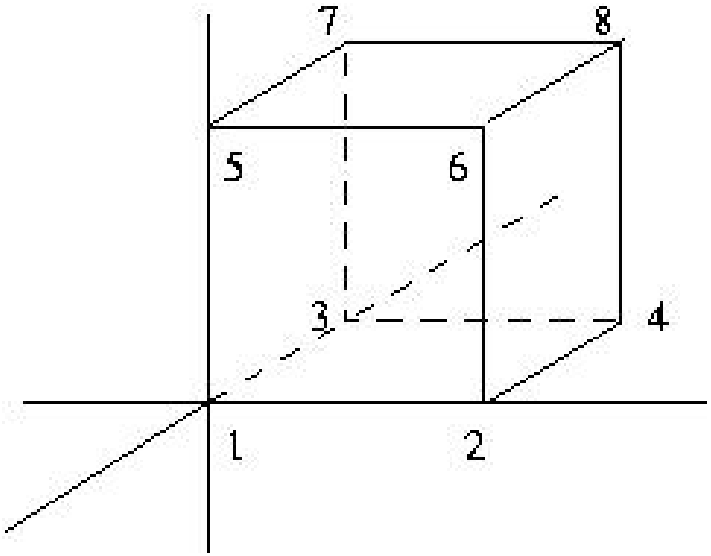
distance in low dim. space

$$E = \frac{1}{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} d(i,j)} \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \frac{\left(d(i,j) - d^{*}(i,j)\right)^{2}}{d(i,j)}$$

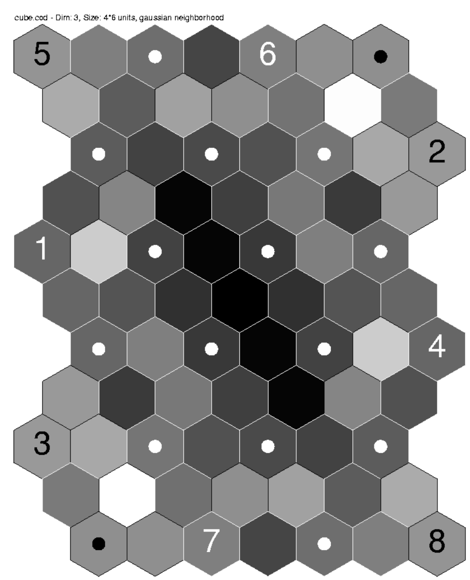
- Energy function is a subject to minimization (originally using gradient descent)



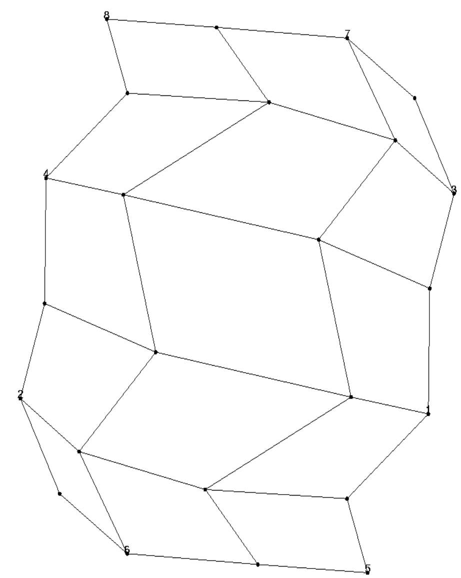
Standard SOM Visualizations



UMAT



Sammon
neuron weights projected to 2D, neighbours connected

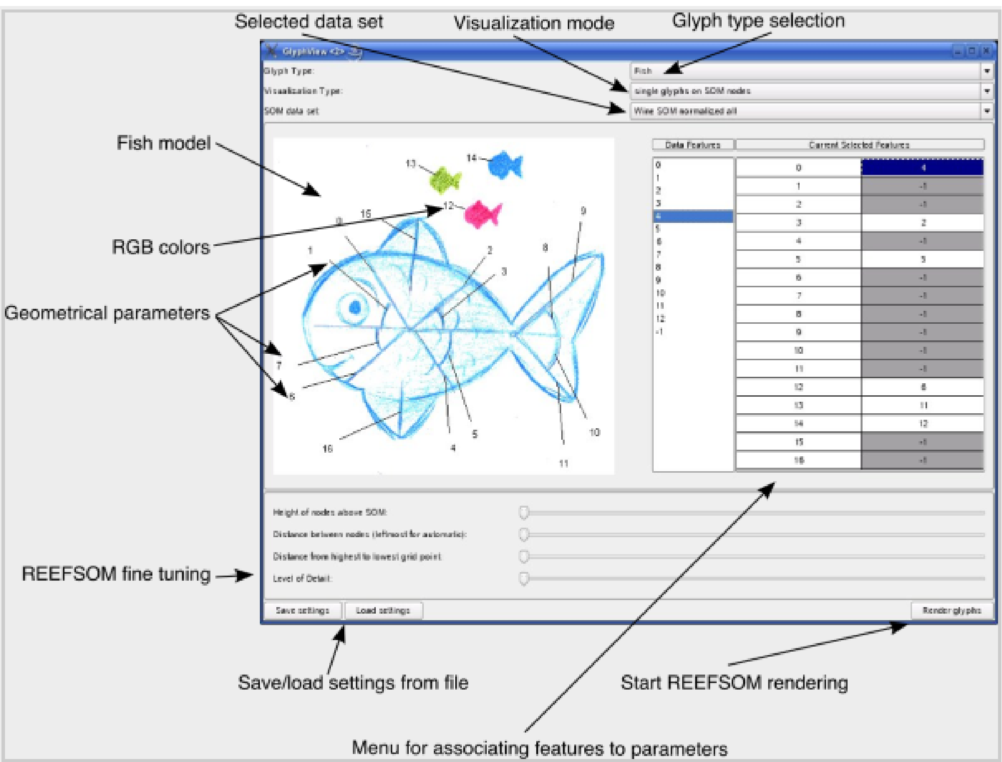


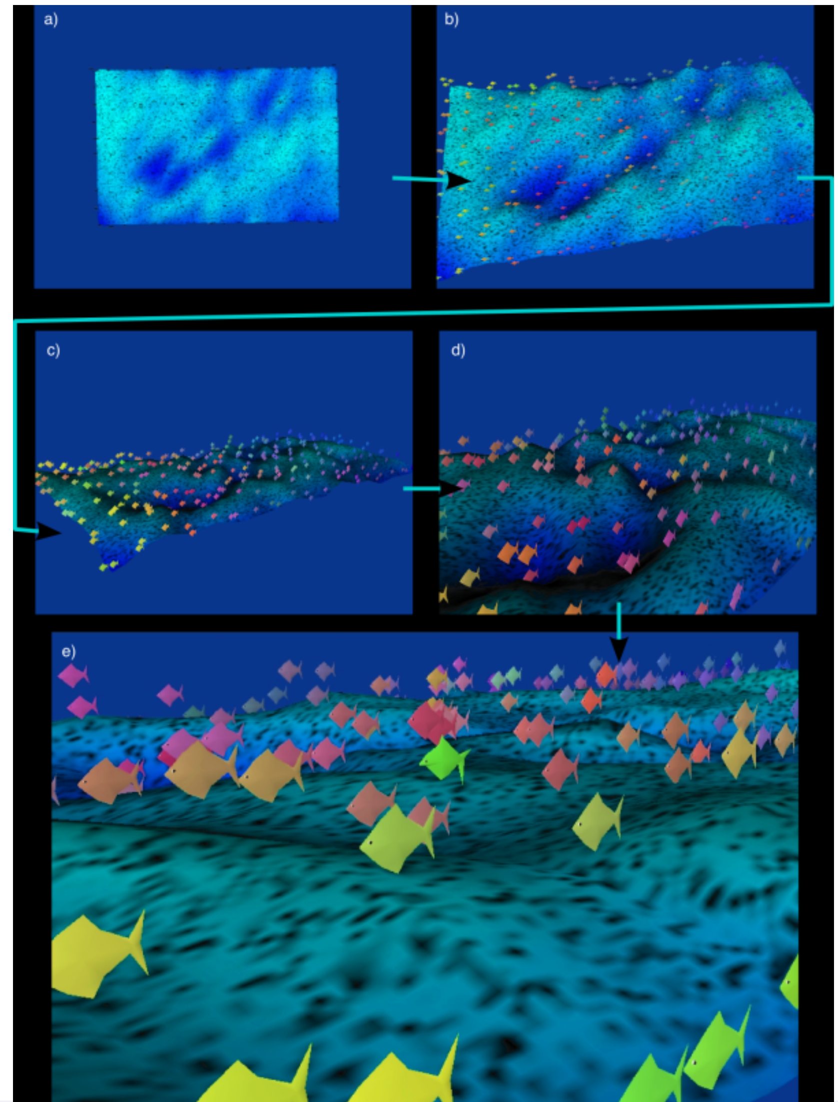
SOMU Applications

- Detection of similar images.
- http://www.generation5.org/content/2007/kohonenImage.asp

ReefSOM

- SOM visualization for non-experts.
- UMAT + glyphs.
- http://www.brains-minds-media.org/archive/305





SOM Evaluation

- VQ – vector quantization, more input vectors mapped into a single neuron → **quantization error or distortion**.
- Compression of an input space dimension.
- Preserves data topology – neighbour vectors (from an input space) are mapped to neighbour neurons (in the mesh) → **topographic error**.

SOM Quantization Error & Distortion

- Quantization Error → average distance between input vector and its BMU (computed over all input vectors).
  - precision of mapping.
- Distortion → count with neighbours:

$$E=\sum_{i\in N}\sum_{j\in I}\eta_{i,bmu(j)}\|w(i)-x(j)\|^2$$

neurons

input vectors

Energy function again!

Topographic Error of SOM

- # of input vectors, for which the winner (BMU) and the second best neuron are not adjacent in the mesh.

Next Lecture

- Universal approximation.
- Kolmogorov's theorem.
- RBF networks.
- GMDH networks.