Lecture on

Self-Organizing Maps

Bruno Silva (bruno.silva@estsetubal.ips.pt)

Self-Organization on the Brain

- The various areas of the brain, especially the cerebral cortex, are organized according to different sensory modalities: areas performing specialized tasks. These regions are called "maps".
- Primary sensory areas comprise only 10% of the cortical area, the rest being left for other associative areas;
- The main structures of the brain networks are determined genetically, but there exists evidence of sensory projections affected by experience.
- There is spatial order and organization in the way the brain behaves.

The Self-Organizing Map (SOM)

- This model proposed by Kohonen in 1982 captures the essential features of computational maps on the brain in a very simple, yet powerful, algorithm.
  - Sometimes also referred as Kohonen's Feature Map.

- The SOM is a very popular ANN algorithm based on competitive and unsupervised learning.

- Extensive research has been made with applications ranging from full text and financial analysis, pattern recognition, image analysis and process monitoring.

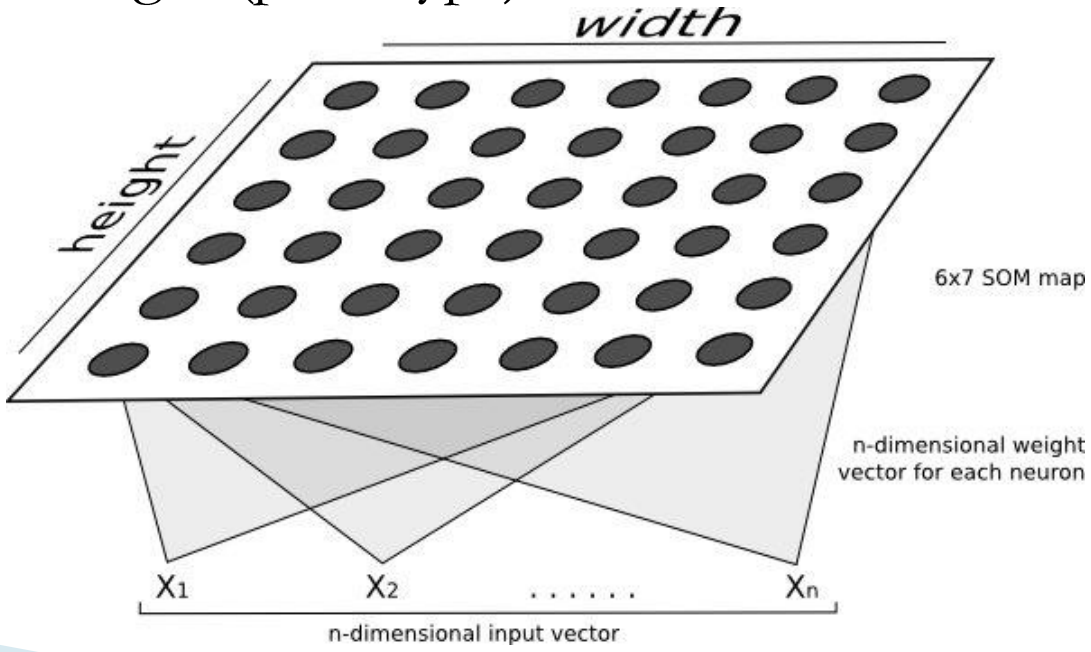
The Self-Organizing Map (SOM)

- The SOM is able to project high-dimensional data in a lower dimension, typically 2D, while preserving the relationships among the input data.

- The 2D pattern map is useful in
  - Analyzing and discovering patterns in the input space;
  - Detecting non-linear correlations between features;
  - Identifying clusters;
  - Classifying new observations.

The Self-Organizing Map (SOM) Architecture of the Network

- Consists in two layers (feed-forward):
  - Input layer;
  - Output (competitive) layer.
- K neurons arranged in a 2D lattice;
- Neighboring relations between neurons;
- Each neuron contains a weight (prototype) vector;



The Self-Organizing Map (SOM) Overview of functioning

- Normalization of training data (we'll see why).
- Network initialization (may be randomly, sample-based or "linearly").
- Emergence of self-organization is achieved by three essential processes:
  - **Competition**: When an input pattern, from the training set, is presented to the network, a discriminant function is computed for all weight vectors. This function provides the basis for competition among the neurons and is commonly given by a metric distance. The neuron with the most similar weigh vector to the input pattern is called the Best Matching Unit (BMU) and is the "winner" of the competition.

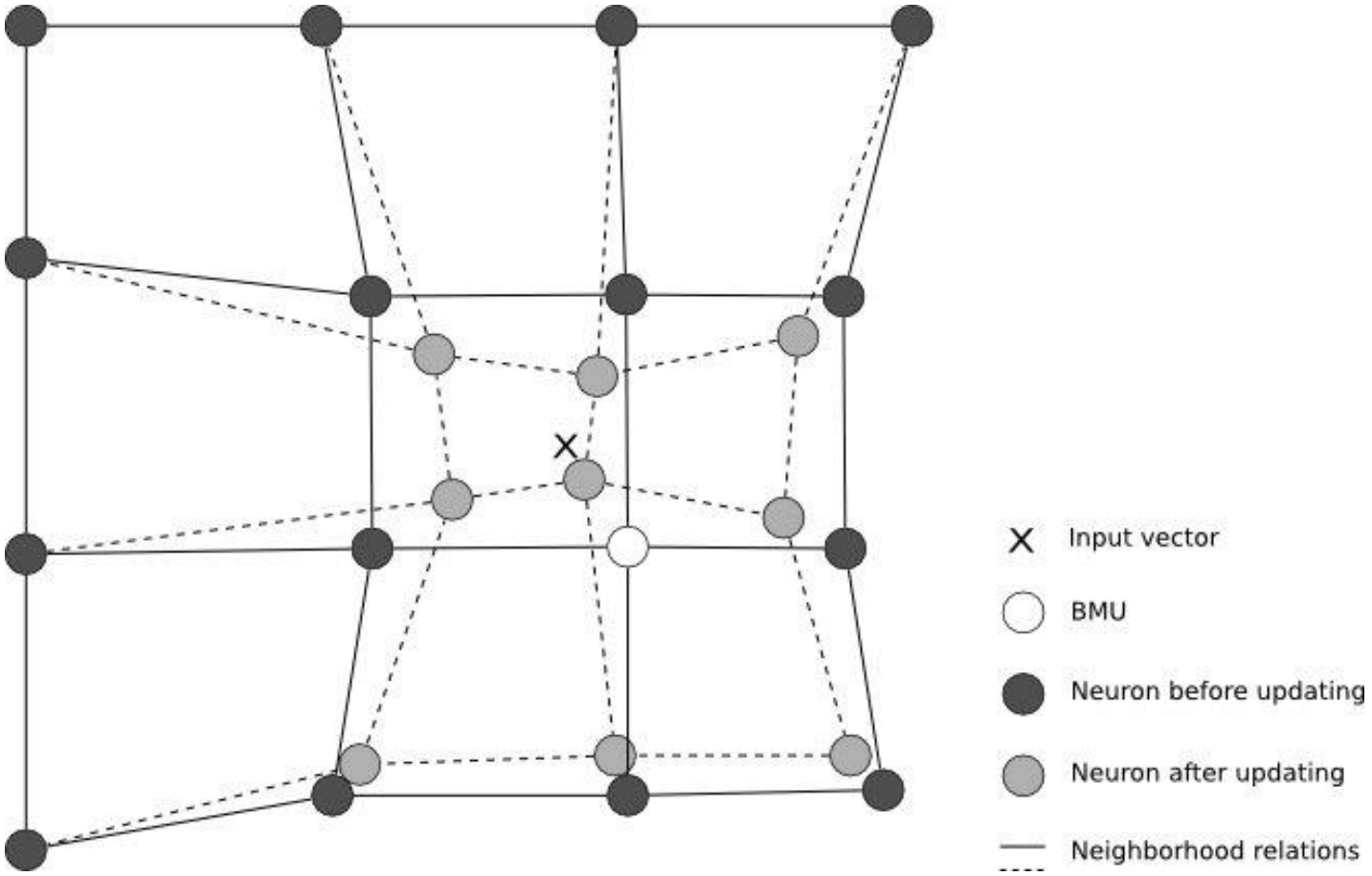
The Self-Organizing Map (SOM) Overview of functioning

- **Cooperation**: The BMU determines the spatial location of a topological neighborhood of "excited" neurons, thereby providing the basis for cooperation among such neighboring neurons. This neighborhood will depend on the type of lattice used and neighborhood radius.

- **Adaptation**: This last process enables the excited neurons to increase their resemblance to the input pattern through suitable adjustments made to their weight vectors. This adjustments decrease with lateral distance from the BMU. Consequently, the response of the winning neuron to a subsequent similar input is enhanced.

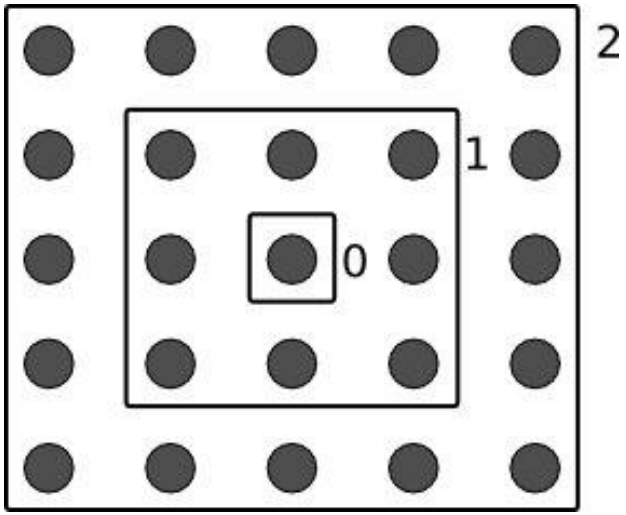
- The network must be exposed to a sufficient number of input patterns to ensure that the self-organization process reaches a stable state. Usually we recycle the input patterns.

The Self-Organizing Map (SOM) Adaptation Process Illustrated

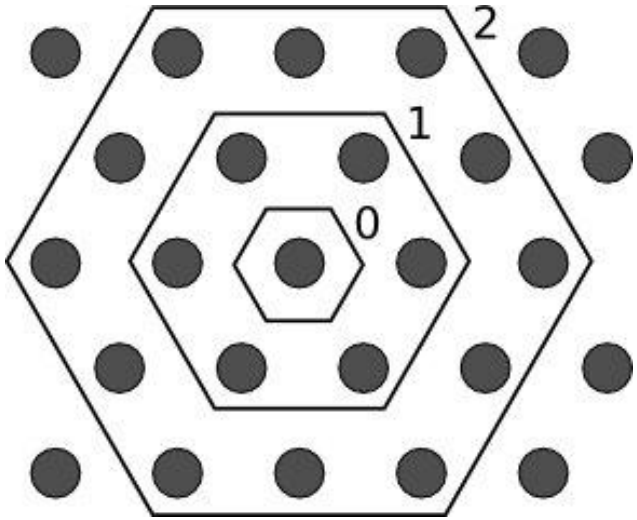


The Self-Organizing Map (SOM)
Types of Lattices

Rectangular



Hexagonal



Neighborhood (at radius 0, 1 and 3) around the BMU.

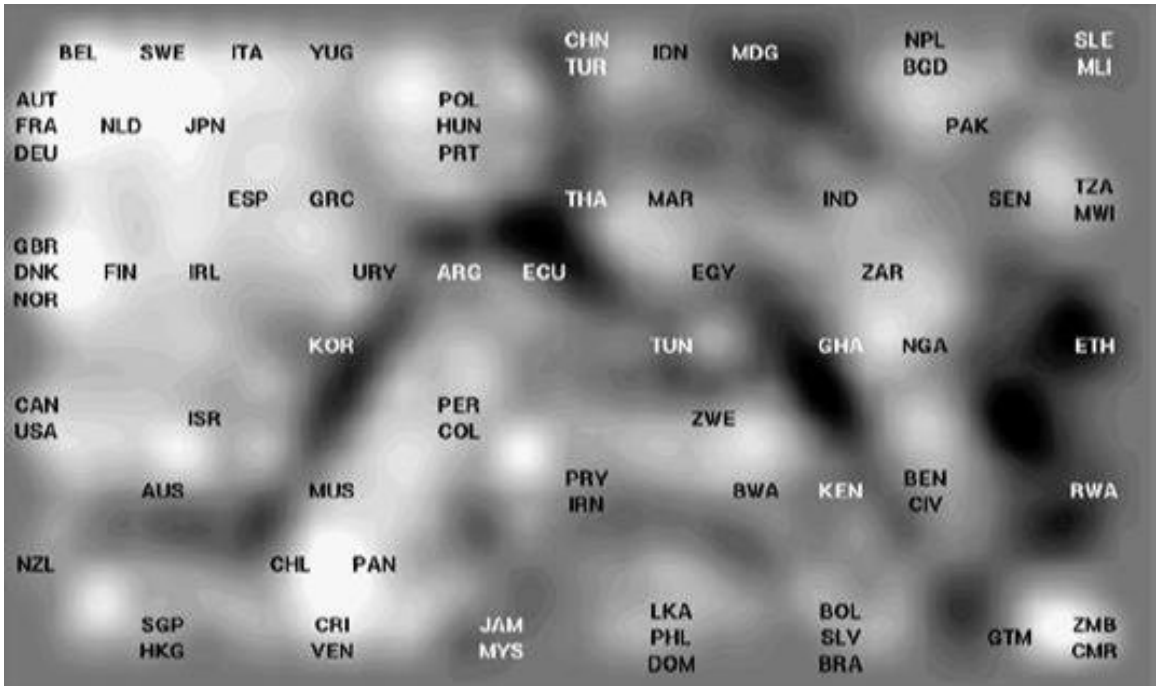
The Self-Organizing Map (SOM) Important Aspects

- Input vectors which are neighbors in the input space are mapped onto neighboring neurons.

- If the dimensionality of the input space and the network differ, only the most important similarity relationships are preserved and mapped onto neighborhood relations on the network
  - The less important ones are not retained.

- The SOM is very robust against "noise" in the input data, because the SOM also maps the input distribution density!

The Self-Organizing Map (SOM) Example – World Poverty Map (1992)



Information about welfare and poverty-related aspects of 77 countries are treated by the SOM algorithm. The map is visualized with a special visualization technique, which we'll discuss later in this lecture.

The Self-Organizing Map (SOM) Formalization of the Algorithm(s)

- There are two major variants of the SOM algorithm:
  - The **classical** – also called sequential or online - algorithm;
  - The **batch** algorithm.
- They differ in the adaptation equation and on when the updates of the weight vectors are made.

- We'll denote the input vector x as:

$$x = [x_1, x_2, ..., x_n] \tag{2.1}$$

- The weight (prototype) vector of neuron k is denoted by:

$$W_k = [w_{k1}, w_{k2}, ..., w_{kn}], \quad k = 1, 2, ..., K \tag{2.2}$$

The Self-Organizing Map (SOM) Formalization of the Algorithm(s)

- A discrete temporal index *t* exists, such that a input vector x(t) is presented to the network at time t and the $W_k(t)$ in the weight vector of neuron k computed for that instant.
- Input patterns are recycled during training: one single pass over the training set is called an **epoch**.

- The discriminant function (to find the BMU) used is usually the Euclidean distance from x(t) to $W_k(t)$:

$$d_k(t) = \|x(t) - W_k(t)\| \qquad (2.3)$$

The Self-Organizing Map (SOM) Formalization of the Algorithm(s)

- The BMU is the neuron that holds the smallest distance:

$$d_c(t) = \min_k \; d_k(t) \tag{2.4}$$

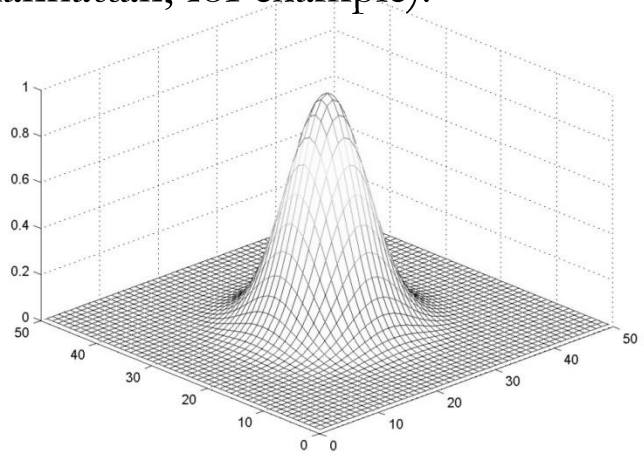
  - If multiple minima occurs, than one can be selected randomly.

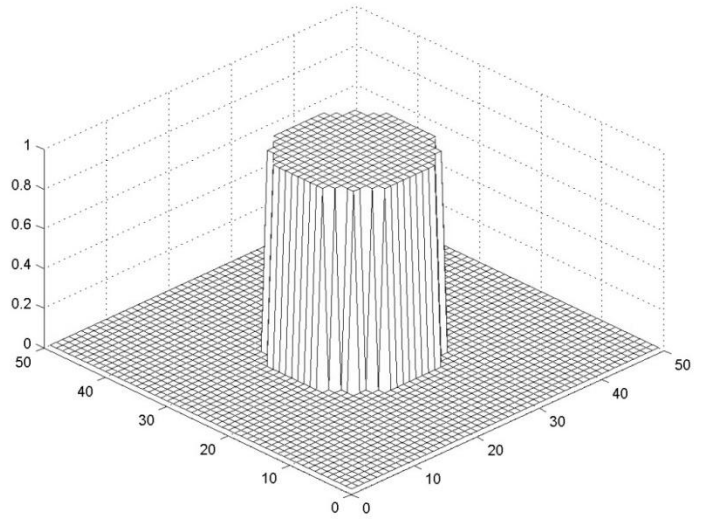
- The winning neuron locates the center of a topological neighborhood of excited neurons. Since the updates decrease with lateral distance a neighborhood kernel (function) – $h_{ck}(t)$ is computed.
  - Usually the Gaussian function is used, although the Bubble function is another possibility.

The Self-Organizing Map (SOM) Formalization of the Algorithm(s)

The values are calculated based on the distance between the BMU and the neighboring neuron k in the lattice.

Euclidean distance is commonly used, although others can be used (Manhattan, for example).





$$h_{ck}(t) = e^{\frac{-\|r_k - r_c\|^2}{\sigma(t)^2}}$$

$$h_{ck}(t) = \begin{cases} 1 & \text{if } \|r_k - r_c\|^2 < \delta(t) \\ 0 & \text{otherwise} \end{cases}$$

σ (sigma) represents the effective width of the neighborhood at time t.

The Self-Organizing Map (SOM) Formalization of the Algorithm(s)

- The topological neighborhood decreases monotonically over time, from a value no less than half the largest diagonal of the map to a value that encompasses the immediate neighbors (radius 1).

- **This is a necessary condition for convergence!**

The Self-Organizing Map (SOM)
The Classic Algorithm

- In the classic algorithm the weight vectors are updated immediately upon the presentation of an input pattern, using the following equation:

$$W_k(t+1) = W_k(t) + \alpha(t)h_{ck}(t)[x(t) - W_k(t)] \qquad (2.6)$$

which is applied to all neurons that lie in the neighborhood of the winning neuron c (BMU).

- α is the learning-rate factor ($0 < \alpha(t) < 1$) that also decreases monotonically over time.

The Self-Organizing Map (SOM)
The Classic Algorithm

Initialize the weight vectors, $W_i \in R^n$

$t = 0$

for *epoch*=1 to N_{epochs} **do**

  for *input*=1 to N_{inputs} **do**

    $t = t + 1$

    for k=1 to K **do**

      Compute distance d_k, using Equation (2.3)

    end for

    Compute BMU for current input, using Equation (2.4)

    for k=1 to K **do**

      Update weight vectors W_k using Equation (2.6)

    end for

  end for

end for

The Self-Organizing Map (SOM)
The Batch Algorithm

- In the Batch variant of the SOM algorithm the updates are deferred to the end of a learning epoch, i.e., the presentation of the whole training set and the new weights are computed using:

$$W_k(t_f) = \frac{\sum_{t'=t_0}^{t'=t_f} \tilde{h}_{ck}(t')x(t')}{\sum_{t'=t_0}^{t'=t_f} \tilde{h}_{ck}(t')} \tag{2.7}$$

$$\tilde{d}_k(t) = \|x(t) - W_k(t_0)\| \tag{2.8}$$

$$d_c(t) = \min_k \tilde{d}_k(t) \tag{2.9}$$

where t_0 and t_f stand for, respectively, the beginning and end of the current epoch.

The Self-Organizing Map (SOM)
The Batch Algorithm

- The learning-rate factor is explicitly present in the batch update equation and does not need to be parameterized.
  - Also, if a neuron is not affected during an epoch, then $W_k(t+1) = W_k(t)$.
- This variant can only be applied when the whole set of input data is present.

- Since the weights are not updated immediately there is no dependency on the order of the input vectors.

The Self-Organizing Map (SOM)
The Batch Algorithm

Initialize the weight vectors, $W_i \in R^n$

$t = 0$

for *epoch*=1 to N_{epochs} **do**

  Interpolate new value for $\sigma(t)$

  Reset numerator and denominator of Equation (2.7)

  for *input*=1 to N_{inputs} **do**

    $t = t + 1$

    for k=1 to K **do**

      Compute distance d_k, using Equation (2.8)

    end for

    Compute BMU for current input, using Equation (2.9)

    for k=1 to K **do**

      Accumulate numerator and denominator of Equation (2.7)

    end for

  end for

  for k=1 to K **do**

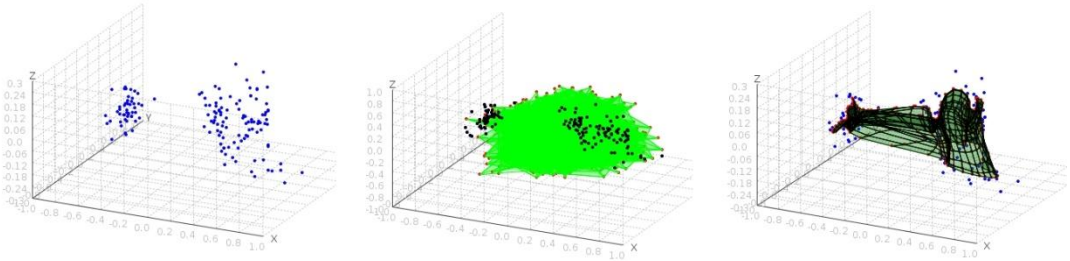
    Update weight vectors W_k with Equation (2.7)

  end for

end for

The Self-Organizing Map (SOM) Ordering and Convergence

- Starting from a state of complete disorder the SOM algorithm gradually achieves an organized representation of the input space, provided that the parameters are chosen properly.



- The adaptation process should be decomposed into two phases: and **ordering phase** followed by a **convergence phase**.

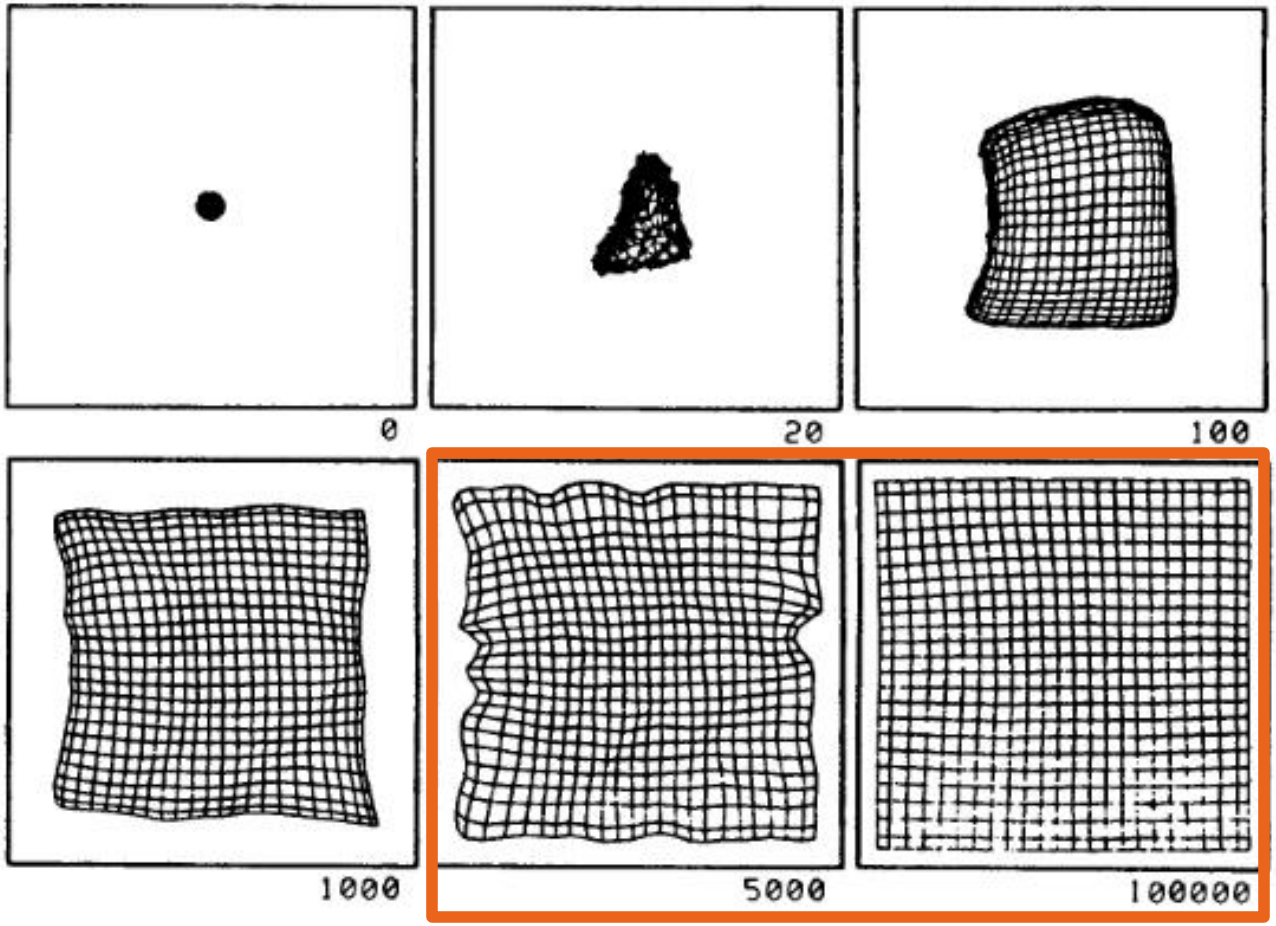
The Self-Organizing Map (SOM) Ordering and Convergence

- **Ordering (rough) phase**
  - It is during this first phase of the adaptation process that the topological ordering of the weight vectors takes place. This ordering phase is relatively *short* in comparison to the second phase. Large values for the neighborhood radius and learning rate should be used, such that the neuron's weights initially take large steps all together toward the area of input space where input vectors are occurring. These values then should decrease to their tuning values and, consequently, the neighborhood decreases to encompass only the closest neighbors.

The Self-Organizing Map (SOM) Ordering and Convergence

- **Convergence (fine-tuning) phase**
  - This phase lasts for the rest of the training or adaptation process and is necessary to fine tune the network and therefore provide an accurate statistical quantification of the input space. During this phase the weight vectors converge to their *correct* values. For this, the neighborhood should be fairly small, encompassing only the immediate neighbors. This also applies to the learning rate, such that the magnitude of the weight updates is very small. The convergence phase is usually several times longer than the ordering phase.

The Self-Organizing Map (SOM) Ordering and Convergence



Fine-tuning Phase

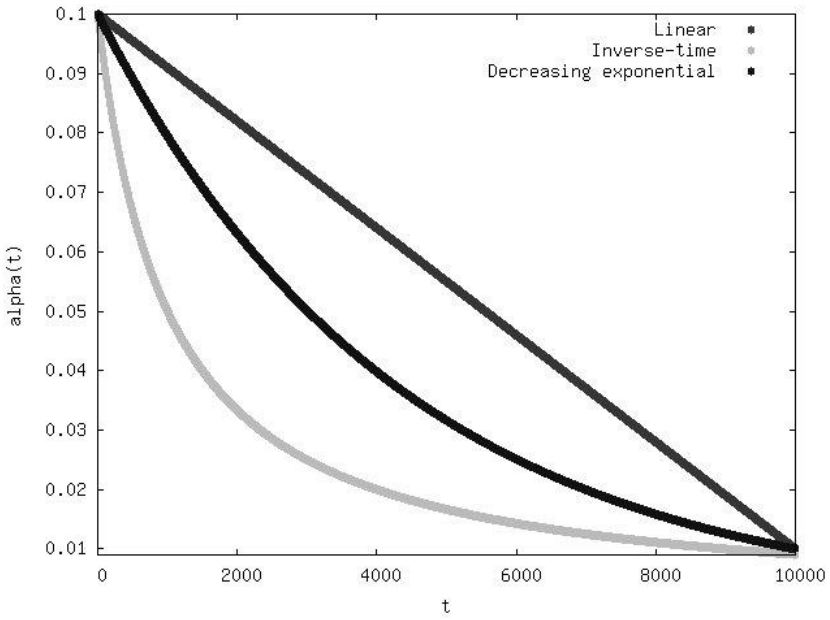
The Self-Organizing Map (SOM) Parameterization

- Appropriate selection of the initial parameters is of great importance to obtain good maps.
- Unfortunately, parameterizing the SOM is an "art" :-(
  - But there are some "rules-of-thumb".
- We'll talk briefly about:
  - The learning rate;
  - The width of the neighborhood kernel;
  - The map's shape;
  - The map's size.

- The number of epochs usually depends on the size of the training set. Small training sets may require more epochs.

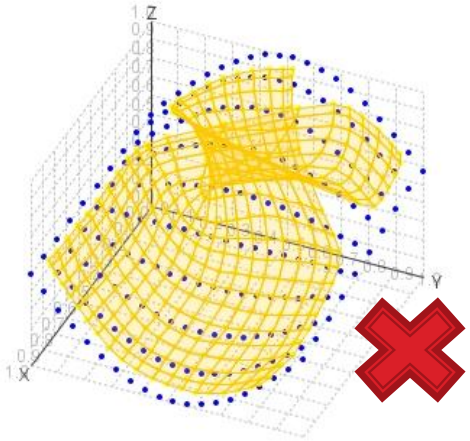
The Self-Organizing Map (SOM) Parameterization – α and σ

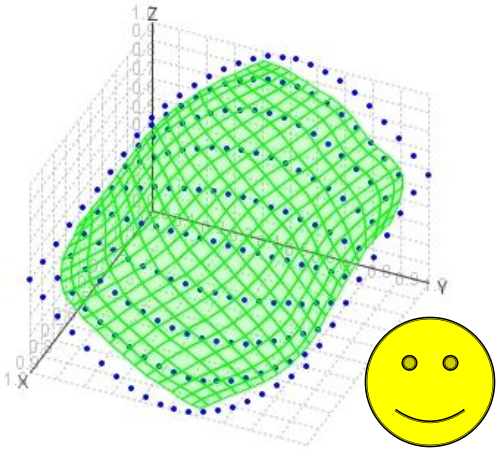
- They should be ruled by a function that decreases monotonically over time.
  - Its exact form is not critical.
- The learning rate is hard to estimate and is "problem-specific". General values can be $0.1 > \alpha > 0.01$. Although higher values may avoid local minima.
- The neighborhood radius should start with a value that spans the whole map down to the immediate neighbors.



The Self-Organizing Map (SOM) Parameterization – α and σ

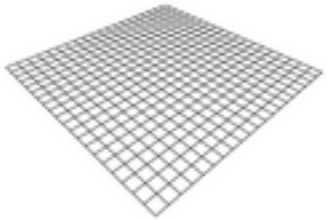
- The properly initialization of the neighborhood radius has a strong impact in the final topology of the map.
  - If one chooses a small initial value, a distorted map can be obtained – which is not desirable!



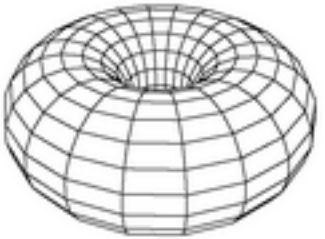


The Self-Organizing Map (SOM) Parameterization – Shape

- The SOM shape determines the neurons' neighboring relations.
  - Usually the sheet-like shape is the most common and straightforward:



  - The toroid topology avoids "border effects" by connecting the top of the map to the bottom and the left to the right:



The Self-Organizing Map (SOM) Parameterization – Size

- First of all, the map should take a rectangular form, rather than square, so the weight (prototype) vectors are oriented along with the input space distribution probability.
- The map size determines the degree of precision and the ability to quantize the input space.
  - If one chooses the number of neurons to be equal to the desired number of clusters, then the SOM behaves like a very robust k-means algorithm.
  - Using large maps allows for the emergence of salient features in the input space. With the use of visualizations techniques one can obtain insight on the number of clusters in the data and correlations between features in the training set.

The Self-Organizing Map (SOM) Quality Measures

- The "quality" of the self-organization can be assessed by two major statistics:
  - **Quantization Error**: computed by determining the average distance of input vectors to the prototype vector of the BMU that represents them.

$$QE = \frac{\sum_{i=1}^{i=N} \|x_i - W_{ci}\|}{N}$$

  - **Topographic Error**: is the most simple measure for topology preservation. For all input vectors the respective BMU and the second-BMU are determined. If these two are not adjacent on the map lattice then this is considered an error. Calculated as (#errors/N) where 0 (zero) means perfect topology preservation.

The Self-Organizing Map (SOM) Visualization Techniques

- One of the greatest advantages of the SOM is that it is highly visual.
- There are two major visualizations for the SOM:
  - **Unified Distance Matrix**

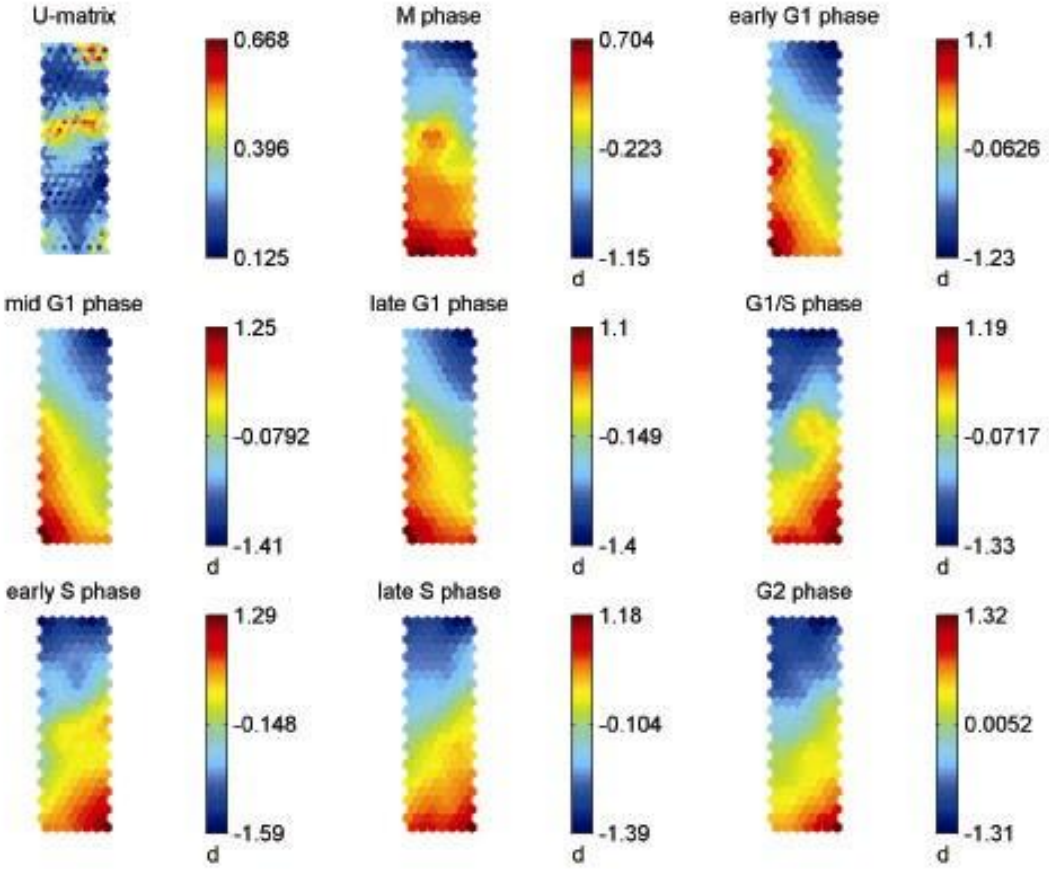
    or simply U-Matrix, visualizes the cluster structure of the SOM. A matrix of distances between the weight vectors of adjacent neurons on the map is formed, after which some representation for the matrix is chosen, for example a color scale. The lighter the color between two map units is, the smaller is the relative distance between their weight vectors. Given this, dark areas on the maps usually identify boundaries between clusters in the underlying data.

The Self-Organizing Map (SOM) Visualization Techniques

- **Component Planes**

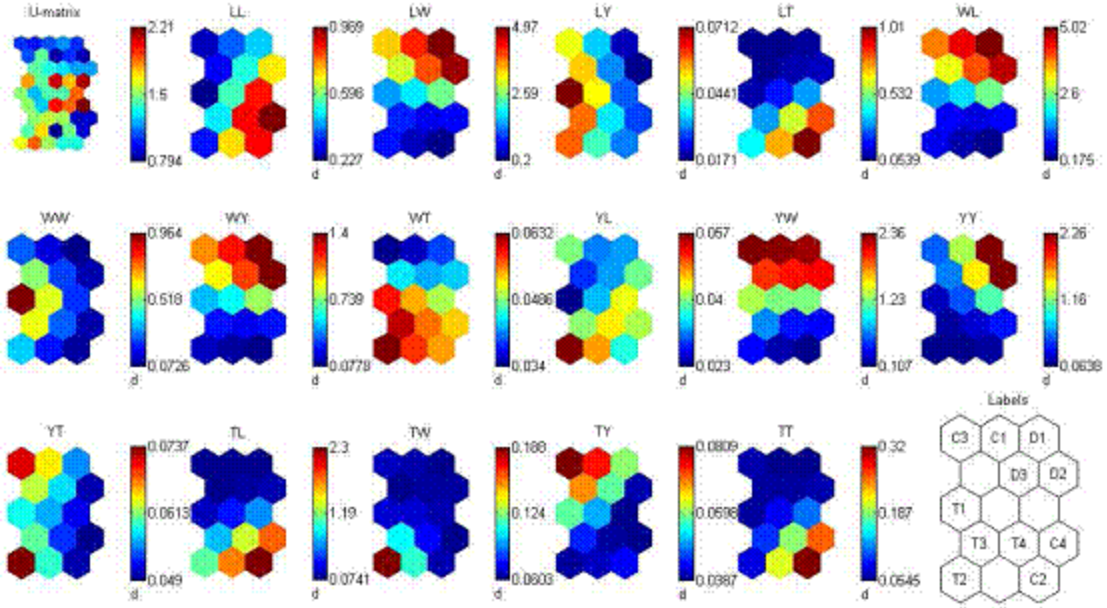
  is a representation that visualizes relative component values in the weight vectors of the SOM. The illustration can be considered as a sliced version of the SOM, where each plane shows the distribution of one weight vector component. Using the distributions, correlations between different components can be studied. Sometimes these component planes can be useful in interpreting the type of samples that belong to a cluster, comparing them to the U-Matrix.

The Self-Organizing Map (SOM) Visualization Techniques - Example



The Self-Organizing Map (SOM) Visualization Techniques - Labeling

- Input samples can have labels attached.
- Labeling the SOM is helpful for certain tasks:
  - Gain insight on the composition of clusters;
  - Use the SOM for classification tasks. A new sample is projected onto the map and the label of the BMU determines the class of the new sample.



The Self-Organizing Map (SOM) References

- T.Kohonen – Self-Organizing Maps (3rd Edition), 2001.Springer

- B. Silva – "A Study of a Hybrid Parallel SOM Algorithm for Large Maps in Data Mining", 2008. Master Thesis FCT-UNL.