Self-Organizing Maps for Exploratory Data Analysis and Multidimentional Data Visualization

Victor Lobo

Computation World 2010

Lisbon

Appetizer: Why use a SOM ? (1/3)

- Large amounts of data

  ⇒ Need for powerful analysis tools

  - **Visualize** and "feel" multidimensional data
  - **Cluster** the data to simplify it
  - **Explore** the data structure

- SOM (Kohonen networks) can be put to better use than they are by most researchers…

Why do I really like them ?

- Visual insight into multidimensional data.
- Easy to use.
- Good results in many problems.

Sumary

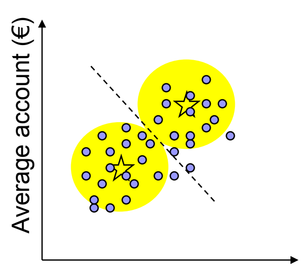
- What can I do with a SOM ?
- What is a SOM ?
  - Historical perspective & basic principles
- Mathematical formalization.
- How can I see results ?
  - Exemples
- Available Software
- Maritime Applications

What can I do with a SOM?

- Define and Detect clusters
- Visualize multidimensional data
- Explore data
- Other…

Define clusters (*k*-mean clustering)

- Market segmentation
- Localization

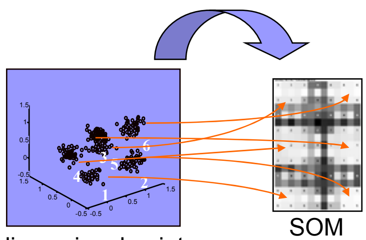


Clients of a Bank ⇒ Account managers


Shop location ⇒ Warehouse location

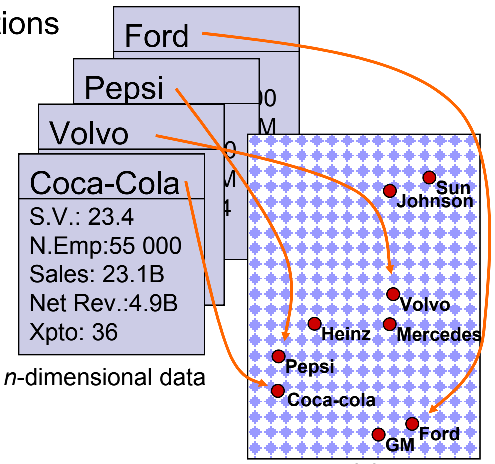
Visualize multidimensional data

- Project a *n*-dimensional space onto 1 or 2 dimensions
- Maintain topological relations
- Detect similarities
- Detect outliers



3-dimensional points

SOM
(bi-dimensional)

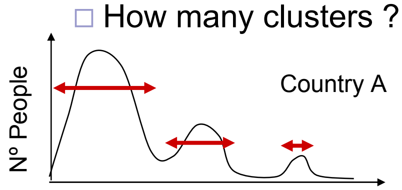


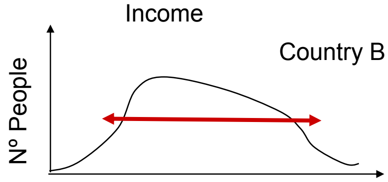
n-dimensional data

SOM

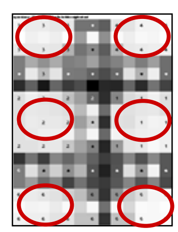
Detect clusters

- Explore data
- Identify the data structure
  - How many clusters ?, How big ?...

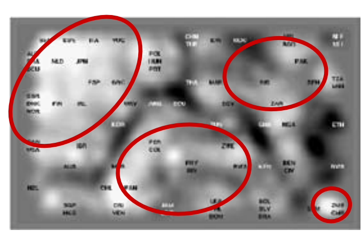




Distribution of income in 2 countries



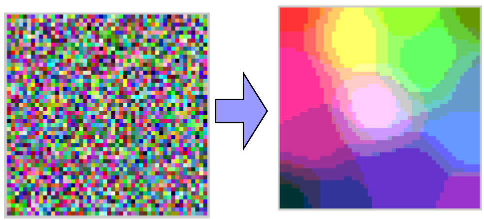
3D points located on
6 vertices of a cube



Socio-economic indicators in
various countries

Other types of problems…

- TSP, Robot contol, data sorting, data interpolation, data classification, feature extraction, sampling, alarm detection, etc, etc, etc



Sorting colours

Travelling salesman problem

What is a SOM ?

- Historical perspective
- Basic principles and overview
- The maths

Historical perspective

Historical Perspective

- Prof. Tuevo Kohonen (Technical University of Helsinki)
  - 1970s - Associative Memory
  - 1982 - First papers on SOM
  - 1988 - Book on SOM, SOM paper in IEEE
  - 1990s - Widespread use
  - 1995,1997,2001 – "Self Organizing Maps" Book
  - Workshop on SOM (WSOM) conferences (next in 2011)
- Motivation and inspiration
  - Vector Quantization methods
  - Associative Memories
  - Preserve topology over the mapping: nearby patterns should be mapped to nearby neurons

Biological inspiration (Just interesting…)

- Biological systems have self-organization

- There is evidence of:
  - Layered structure in the brain
  - Information is spatially organized in the Brain
  - Similar "Concepts" are stores in adjacent areas
  - Experimental work with animals suggests the is na organization similar to SOM of patterns in the visual cortex

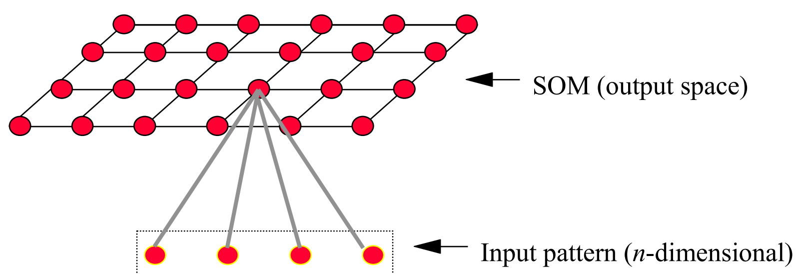
Overview of SOM

Some general ideas:

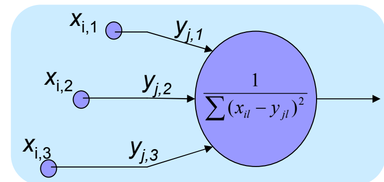
- Neural Network
  - Set of neurons, or UNITS
  - Unsupervised learning (unlike most Neural Nets)
- TRAINING the neural net
  - The net is TRAINED, i.e., its parameters are adjusted (incrementally) according to the available data
- USING the neural net
  - After training the network, we can do many things with it: make predictions, detect clusters, etc,etc

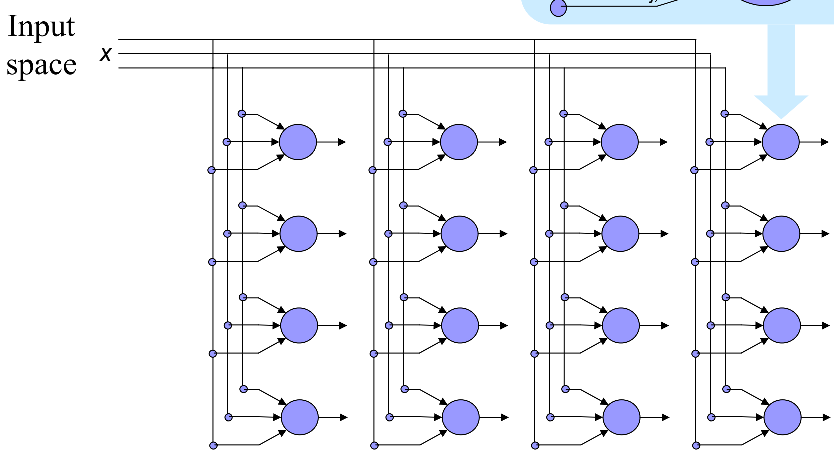
Basic SOM

- Neurons (**units**) are set on a 2-dimensional grid
  - It may be 1-dimensional (line) ou *m*-dimensional …
- One single layer
- Competitive learning (almost "winner-take all")

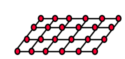


Basic SOM (another vision...)





Input *vs* output space



- **INPUT space** = *n*-dimensional space of the data
- **OUTPUT space** = space defined by the grid of units (usually 2)
- Each unit (neuron) is a point in the output space (defined by the grid position), and a **point in the input space** (defined by the weight vector), just as the data patterns

Training the network

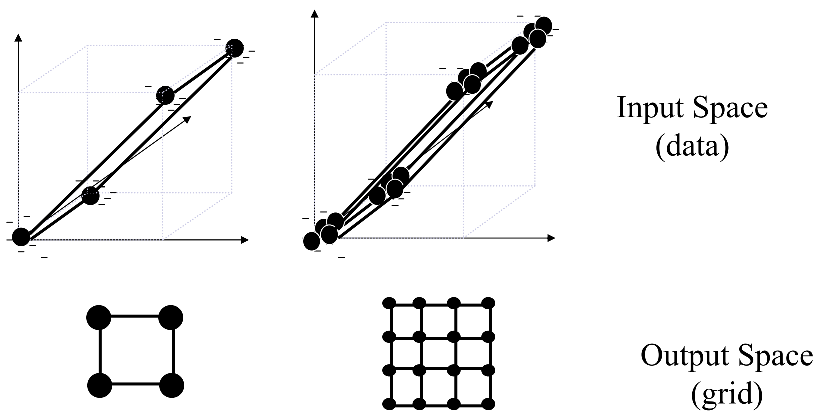
- Units are **pulled** to the positions of the data, **dragging** with them their grid neighbours
- SOM ≈ **rubber sheet**, stretched and twisted so that it passes in (or near) the places where the data patterns are

BMU- Best Matching Unit

- Data patterns are compared with all network units; the closest is considered its **BMU**.-Best Matching Unit.
- That the data pattern is thus **mapped** to the position of it's **BMU**, or for short, is mapped to its **BMU**.
- **The BMU is updated** (so that it resembles even more the data pattern that it maps), and it's neighbors in the grid are also updated.
- There is always a slight difference between the data patterns and the BMUs that represent them. That difference is the **quantization error**.

Example 1: 3D to 2D mapping

- 3D points around 4 corners of a cube

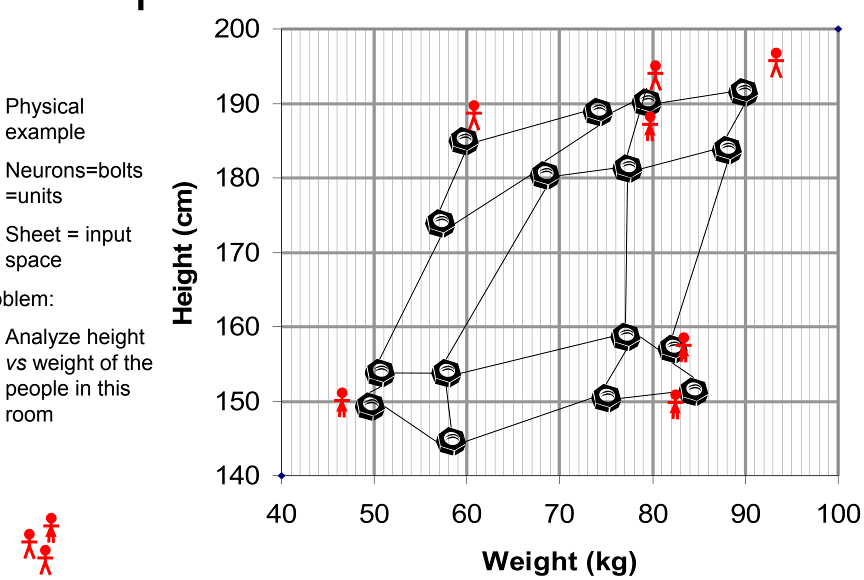


Example 2:

- Physical example
- Neurons=bolts =units
- Sheet = input space

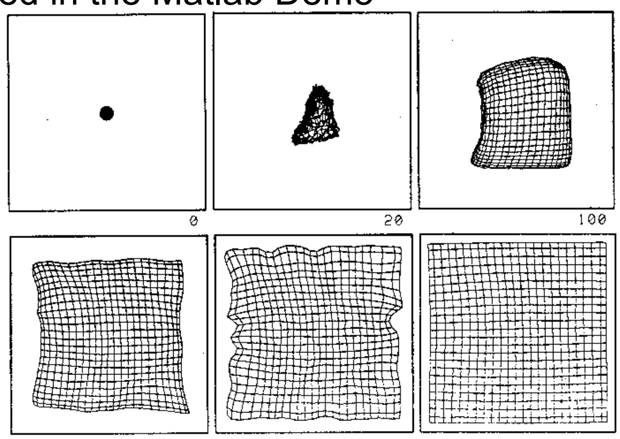
Problem:

- Analyze height *vs* weight of the people in this room



Example 3: 2D to 2D mapping

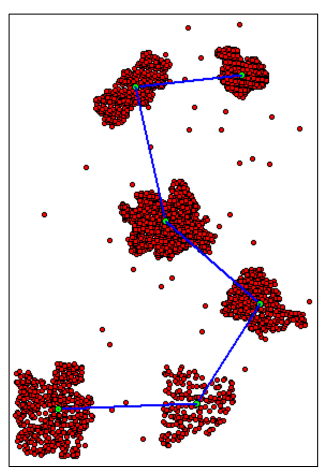
- Data uniformly distributed in the square
- Used in the Matlab Demo



[Kohonen 95]

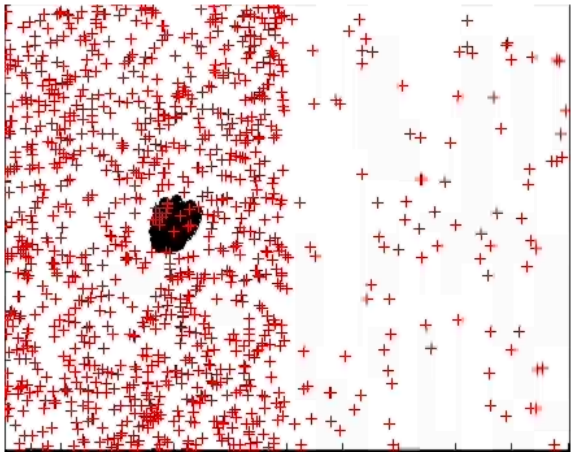
Example 4: 2D to 1D mapping

- Animation
  - Red dots=data
  - Green dots=units
  - Blue lines=grid



Example 5: 2D to 2D

- Different densities

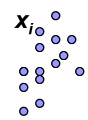


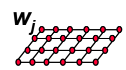
(animation)

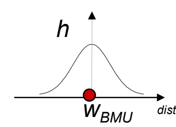
Mathematical formalization

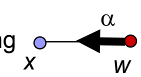
Data, network and initialization

- ***Let:***
  - $\boldsymbol{X} = \{ \boldsymbol{x}_1, \boldsymbol{x}_2,..\boldsymbol{x}_n \}$ ***m-dimensional training dataset***.
    - $\boldsymbol{x}_i = [\, x_i^1, x_i^2, \ldots x_i^m]^T$, where x_{ij} are real-valued scalars.
  - $\boldsymbol{W}$ be a grid with $p \times q$ units $\boldsymbol{w}_i$
    - $\boldsymbol{w}_j = [\, w_j^1, w_j^2, \ldots w_j^m]^T$
    - Initial $\boldsymbol{w}_j$ *chosen randomly in the "data area"*
  - $h(\boldsymbol{w}_i,\boldsymbol{w}_j,r)$ a real-valued funcion (neighborhood function)
    - When $|| \boldsymbol{w}_i\text{-}\boldsymbol{w}_j ||_{(na\ grellha)} \rightarrow \infty$, $h(\boldsymbol{w}_i,\boldsymbol{w}_j,r) \rightarrow \boldsymbol{0}$
    - r sets the radius *(area of influence)*
  - α be the learning rate
    - $0 \leq \alpha \leq 1$
    - Initialized with a large value, and decreases to zero during training

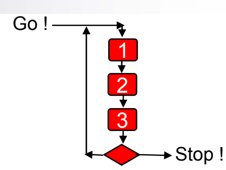








Training Algorithm



- For all $\boldsymbol{x}_i \in \boldsymbol{X}$:

1) **Calculate** the distance between $\boldsymbol{x}_i$ and all units $\boldsymbol{w}$

$$(d_{i,j} = || x_i - w_j ||)$$

2) **Choose** the BMU

$$w_{bmu} : d_{i,bmu} = \min(d_{i,j})$$

3) **Update** each unit according to the learning rule

$$w_j = w_j + \alpha \;\; h(w_{bmu}, w_j, r) \;\; || x_i - w_j ||$$

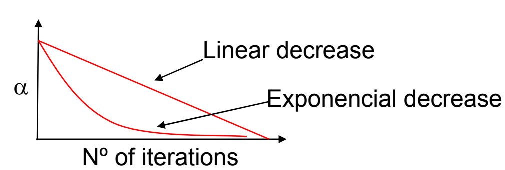
Repeat the process, reducing α and r, using all the training data several times, until a stopping crieteria is reached.

Main decisions:

- How many units ? What type of grid ? How big ? How wide ? How high ?
- How many iterations?
- What type of neighborhood function ? Which initial value for *r* ? Which final value ?
- Which initial value for α ?

Learning rate α

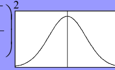
- $0 \leq \alpha \leq 1$
- Defines the ***plasticity*** of the network
  - Large values $\Rightarrow$ The network moves fast, and adapts quickly
  - Small values $\Rightarrow$ The network moves slowly, and stabilizes
- Start with large values, and reduce them to zero during training



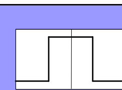
Neighborhood function

- Shape
  - Gaussian
  - Rectangular (bubble)
  - Ramp
  - Others

$$h_g(w_{pq}, w_{mn}, r) = e^{-\frac{1}{2}\left(\frac{\sqrt{(p-n)^2+(q-m)^2}}{r}\right)^2}$$



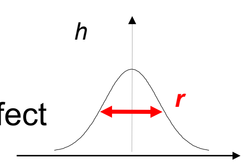
$$h_s(w_{pq}, w_{mn}) = \begin{cases} 1 \Leftarrow \sqrt{(p-n)^2+(q-m)^2} \le r \\ 0 \Leftarrow \sqrt{(p-n)^2+(q-m)^2} > r \end{cases}$$



- Responsible for topological ordering
  - Forces “lateral connections” between units that are neighbors in the grid

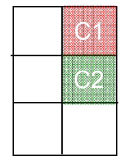
Raduis *r* of the neighborhood function

- Function of time (or number of iterations) r(t)
  - Large radius ⇒ Many units are updates ⇒ Allows *unfolding*
  - Small radius ⇒ Only close neighbors are updated ⇒ Fine-tuning
- Initial values for *r*
  - 1st phase – similar to the network size
  - 2nd phase –radius of the clusters we hope to obtain
- Final value for *r*
  - 0 ⇒Good fit (*k*-means)
  - 1 ⇒Keeps ordering, but has border effect

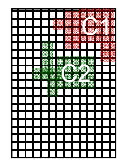


Size of the SOM

- *k*-means SOM (KSOM)
  - Few units
  - 1 unit for each expected cluster

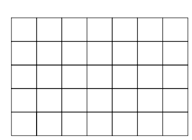


- Emergent SOM (ESOM)
  - Many units for each expected cluster
  - Allows representations of “complicated” and “varied” clusters
  - Allows us to understand the data structure, detect the number of clusters, etc

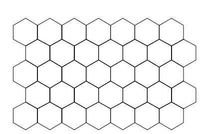


Dimension and type of grid

- Unidimensional grid (line)
  - Subsitute for *k*-means
  - Allows ordering of the data
- 3D or *n*-dimensional grid
  - Hard to visualize
- 2-dimensional grid
  - Most used
  - Square grid
    - Easy to work with
  - Hexagonal or triangular grid
    - Induces less distortion



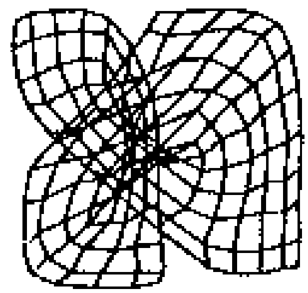
Square grid



Hexagonal grid

Number of iterations and unfolding

- Number of iterations
  - Epochs *vs* individual datum
  - When in doubt… choose more!
- Unfolding problems
  - There are many local minima
  - Topological error metrics
- Solutions
  - Several initializations and runs
  - 2 phases (unfolding+fine tuning)
  - Look at the topological and quantization errors



[Ritter 92]

Theoretical aspects

- Energy function that is minimized: [Hertz 91]

$$V(w) = \frac{1}{2}\sum_x \sum_i \Lambda(i,i^*)\left|\vec{x} - \vec{w}_i\right|^2 = \frac{1}{2}\sum_x \sum_k M_{x,k} \sum_i \sum_j \Lambda(i,k)\left(x_j - w_{ij}\right)^2$$

  - Highly non-linear, not global due to the concept of BMU
  - Reasonable results for 1-dimensional net and data [Cottrell]
  - Reasonable approximations for 2D [Ritter]
  - There are "well behaved" update rules [Heskes]
  - There "well founded" alternatives [Bishop]

$\Big\}$ Not used much

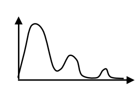
- Magnification factor
  - Unit density $\propto$ (data density)^k, with k<1
  - There is a "magnification" in the representation of low density areas

How can I see results?

- U-Matrices (U-MAT)
- Calibration (or labeling)
- Component planes
- Others

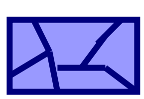
U-Matrices (U-MAT) [Ultsch 93]

- Allows *identification* of clusters

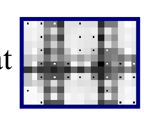


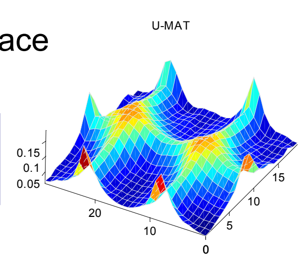
- Computes the distance, in the input space, of neighbors in the output space
- Distance is color coded
  - Low values ⇒ Units close ⇒ cluster
  - High values ⇒ Units far ⇒ empty space

Ideal

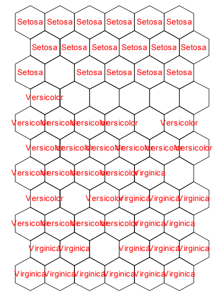


Real U-mat





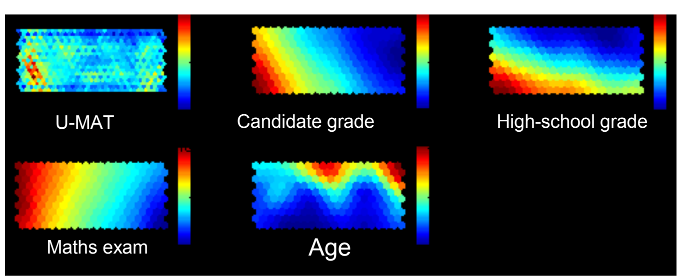
Calibration (or labeling)



- Objective
  - Identify what the cluster ***are***
  - Perform supervised classification
    - LVQ can be better …
- How ?
  - If training data have an associated classe…
  - …their BMUs can inherit those classes

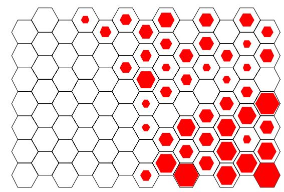
Component planes

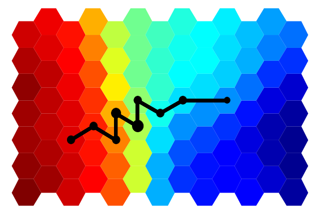
- See how a given variable (component) varies along the map
- See what defines the clusters, that variable contribute to them, how are they correlated



Other visualizations

- *Hits*
  - Identify how many data are mapped to each unit
- Trajectories
  - See how the BMU changes along a data series



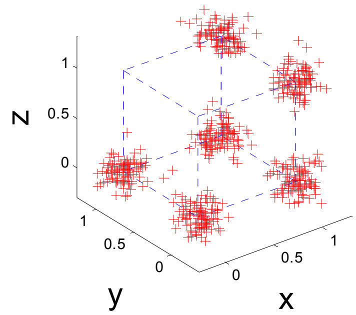


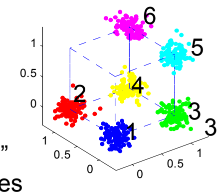
Example

- “Artificial” data

Dataset

- Points in a 3D space
- Generated with some dispersion around 6 corners of a cube
- MATLAB code
  - SOMTOOLBOX

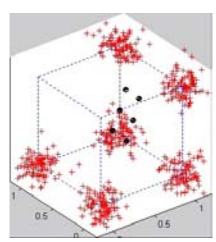




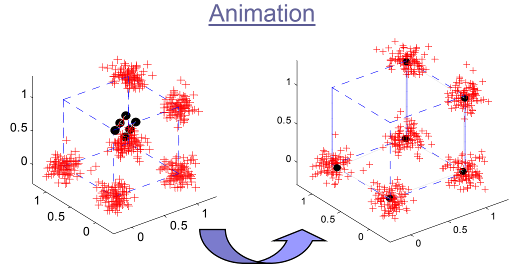
"True" classes

Training a 3x2 map

- Parameters
  - Square grid (3x2)
  - Linear initialization
  - Initial $r = 2$
  - Final $r = 0$
  - Initial $\alpha = 0.1$
  - Num.Iter.=900
    (1,5 epochs)

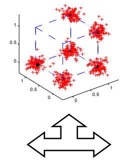


Animation



Visualization in the output space

0.79/0.44/0.61	0.80/0.66/0.62
0.67/0.49/0.43	0.73/0.71/0.41
0.56/0.52/0.29	0.67/0.74/0.25

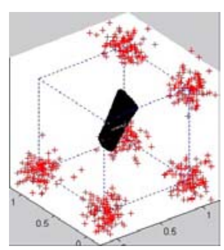


5	6
3	4
1	2

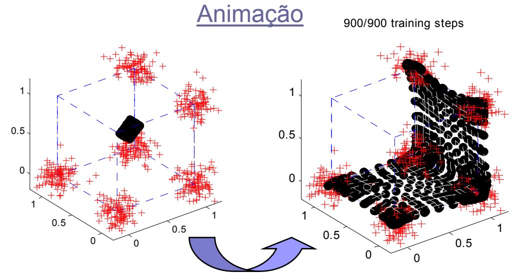
- During training
- After labeling with training data

Training a 30x20 map

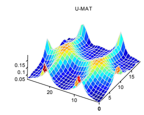
- Parameters
  - Square grid (30x20)
  - Linear initialization
  - Initial r = 15
  - Final r = 0
  - Initial α = 0.1
  - Num.Iter.=900 (1,5 epochs)

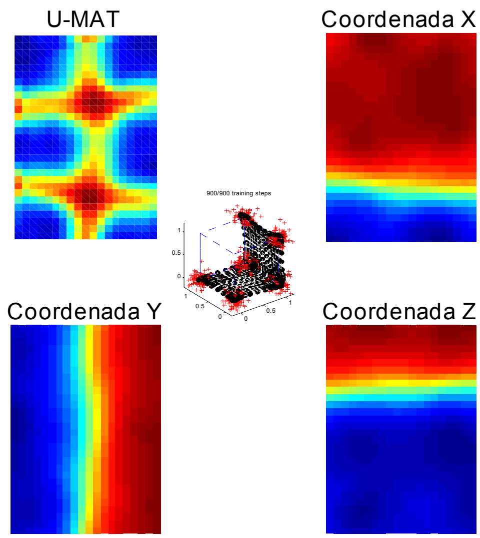


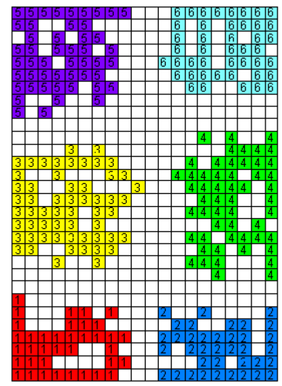
Animação



U-MAT and component planes

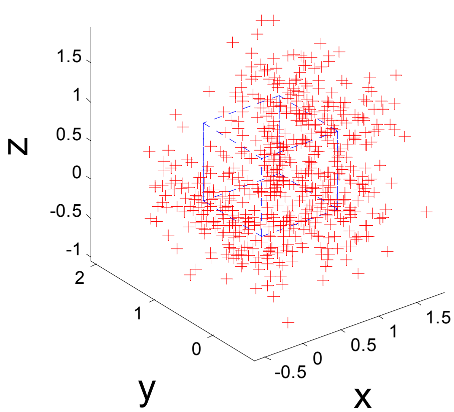


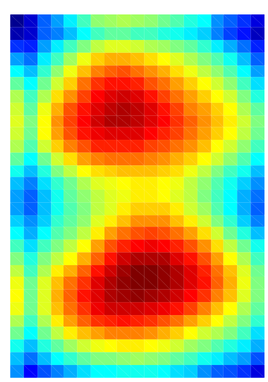




After labeling with training data

With larger variance in the data…





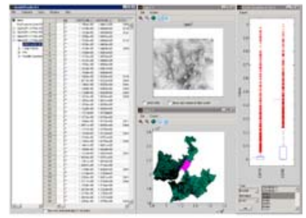
U-MAT

Available Software

Available Software

- SOM-PAK
  - (http://www.cis.hut.fi/research/som_lvq_pak.shtml)
  - C code, compilable in UNIX or MS-DOS
  - Fast, reliable, easy to use
- Somtoolbox for MATLAB
  - (www.cis.hut.fi/projects/somtoolbox)
  - Good visualization, easily changed, “ideal” for R&D
- Many others
  - SAS Enterprise Miner, SPSS-Clementine, IBM Intelligent Miner, Weka, etc...

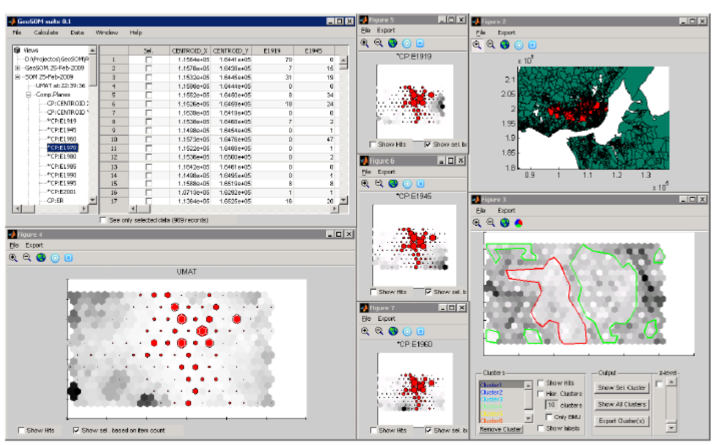
Our implementation of SOM



- GeoSOM-SUITE
  - Based on SOMToolbox
  - User-friendly GUI
  - Imports ArcGIS Shapefiles
  - Allows dyanmically-linked windows
  - Implements SOM and GeoSOM algorithms
  - Many views (Som,c-planes,u-mat,parallelplots)
  - www.isegi.unl.pt/labnt/geosom

GeoSOM Suite

- Interactive exploration of data with SOM

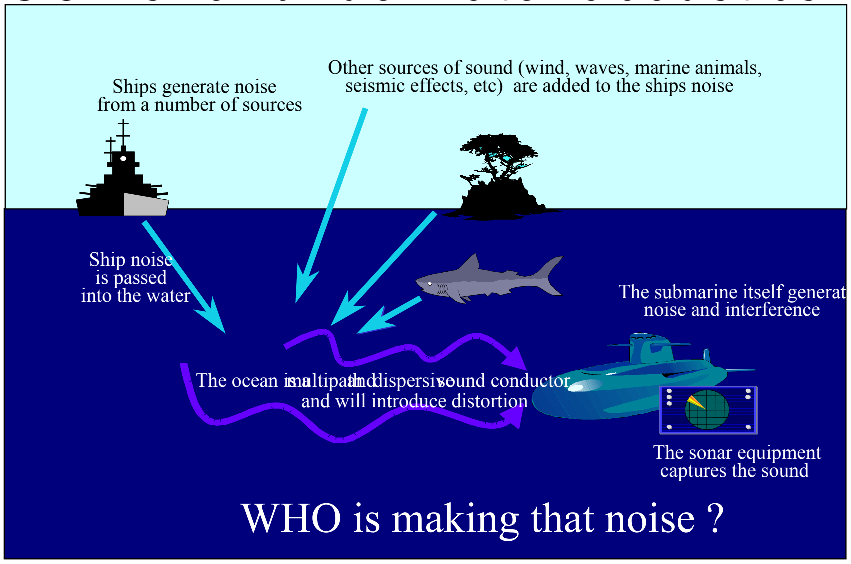


Maritime and GIS Applications

Common applications

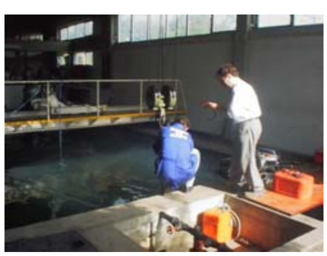
- Clustering and classification
  - Satellite or remotely sensed images, and other data sets
    - Cluster pixels to reduce the number of patters
    - Manually classify a few pixels
    - Use the manually classified pixels to automatically classify the others
    - [Niang 2003][Leloup 2007][Liu 2007][Cavazos 2000][Liu 2002,2007,2008][Tozuka 2008][Solidoro 2007]…
- Control of Underwater Autonomous Vehicles
- Detecting anomalous behavior of ships
- Predicting tides in estuaries
- Studying the movements of fluids

SOMs for underwater acoustics (1/5)



SOMs for underwater acoustics (2/5)

- Data Sources
  - Classified Submarine Squadron data
  - Recordings at the acoustic tank of the navy shipyard
  - Recordings off the Portuguese coast


Recordings in the Shipyard Tank


Recordings at sea (Academy Boats)

SOMs for underwater acoustics (3/5)

Operational Software

Pull-down menus

Spectrogram

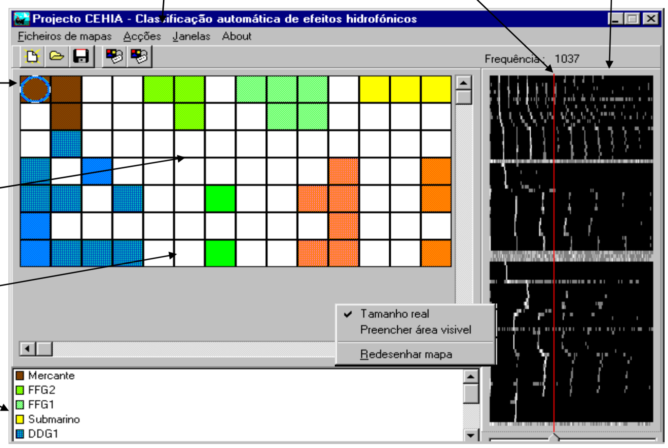
Frequency Pointer

Blinking indicator

Kohonen SOM

Pop-up menus

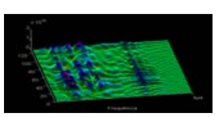
Legend

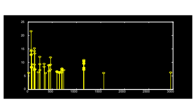


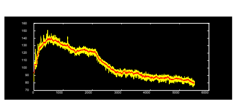
For PC w/ soundcard

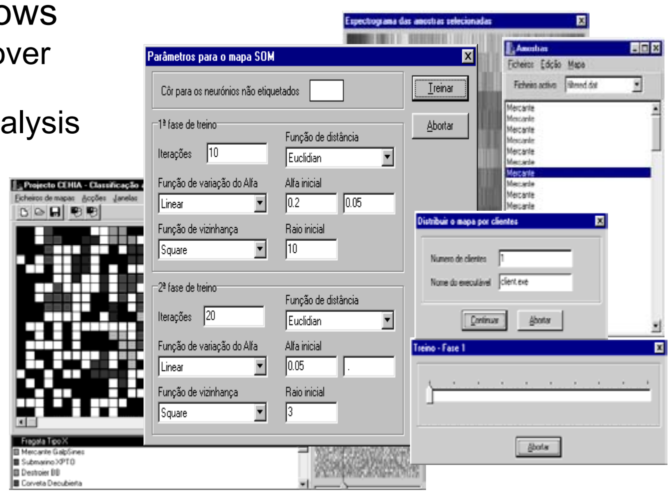
SOMs for underwater acoustics (4/5)

- Other windows
  - Training (over PVM)
  - Off-line analysis







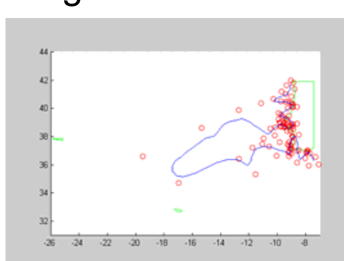


SOMs for underwater acoustics (5/5)

- Main advantages of SOM in this case
  - Explore the available data
    - Is it interesting ?
    - Can we do what we want ?
  - Classify known sounds
  - Detect new sounds
    - Give similarities to known sounds

SOMs for route planning (1/3)

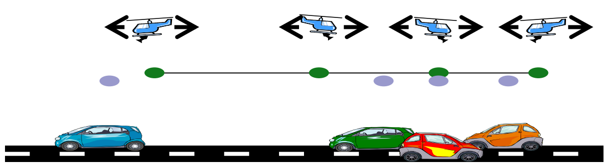
- Route planning
  - 2 to 1-dimensional mapping
  - Training patterns
    - 2-dimensional Coordinates of points of interest
  - 1-dimensional grid of units Patrol routes for ships

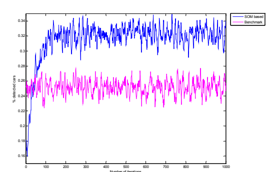


Illegal activities,
and patrol routes

SOMs for route planning (2/3)

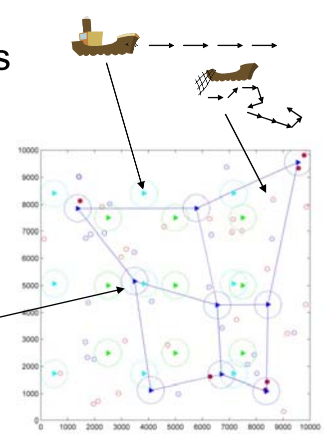
- Routes for multiple sensors, with moving targets
- 1-Dimensional simple case
  - UAVs for monitoring motorways
- Basic idea
  - Keep on training the network as data arrives
  - Keep historical/reference data with lower weight





SOMs for route planning (3/3)

- 2-Dimensional case
  - Fleet of UAV monitoring ships



SOMs for zoning and environment

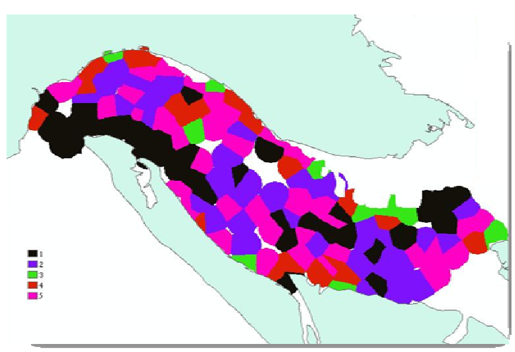
(1/4)

- Define areas for environment control in a very sensitive river estuary
  - Cooperation with École Navale (France)

SOMs for zoning and environment

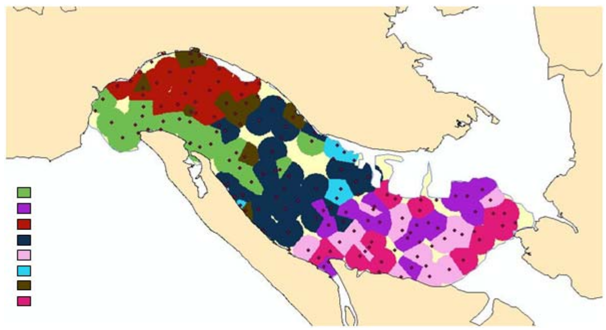
(2/4)

- Previous work
  - Experts decided based on experience
  - Classical k-means (too fragmented)



SOMs for zoning and environment
(3/4)

- Classical SOM
  - Better



SOMs for zoning and environment
(4/4)

- GeoSOM
  - Provided the best balance

Future work and interesting topics

- Clustering trajectories and dealing with time/space issues in SOMs
- Mapping data onto 3D SOMs visualizing them
  - Some progress with geographical data…
- Using SOMs for Piping
  - Routing pipes and cables in a ship
- Theoretical and experimental studies of magnification, convergence, and energy functions

Bibliography and support

- "Self-Organizing Maps", Prof.Tuevo Kohonen
  - Springer-Verlag 2001
- Technical University of Helsinki (www.cis.hut.fi/projects/somtoolbox/links)
  - **Public-domain software, manuals, guides, e documentation**
    - SOM-PAK for DOS, SOM Toolbox for MATLAB
  - **Extensive bibliography**
    - "www.cis.hut.fi/research/som-bibl"
    - 7718 references in December de 2008
- My group's homepage
  - www.isegi.unl.pt/labnt/geosom.html
  - www.isegi.unl.pt/docentes/vlobo

Self-Organizing Maps

V 1.4 V.Lobo, EN 2010