Cartogram Representations of Self-Organizing Virtual Geographies

Àngela Martín

Dept de Llenguatges i Sistemes Informàtics

Universitat Politècnica de Catalunya

A thesis submitted for the degree of

Master in Computer Science

Thesis director: Alfredo Vellido, PhD

September 2013

Abstract: Model interpretability is a problem for multivariate data in general and, very specifically, for dimensionality reduction techniques as applied to data visualization. The problem is even bigger for nonlinear dimensionality reduction (NLDR) methods, to which interpretability limitations are consubstantial.

Data visualization is a key process for knowledge extraction from data that helps us to gain insights into the observed data structure through graphical representations and metaphors. NLDR techniques provide flexible visual insight, but the locally varying representation distortion they generate makes interpretation far from intuitive.

For some NLDR models, indirect quantitative measures of this mapping distortion can be calculated explicitly and used as part of an interpretative post-processing of the results. In this Master Thesis, we apply a cartogram method, inspired on techniques of geographic representation, to the purpose of data visualization using NLDR models. In particular, we show how this method allows reintroducing the distortion, measured in the visual maps of several self-organizing clustering methods.

The main capabilities and limitations of the cartogram visualization of multivariate data using standard and hierarchical self-organizing models were investigated in some detail with artificial data as well as with real information stemming from a neuro-oncology problem that involves the discrimination of human brain tumor types, a problem for which knowledge discovery techniques in general, and data visualization in particular should be useful tools.

Contents

Chapter 1

Introduction

It is not uncommon to underestimate the value of simply looking at data in the process of scientific knowledge extraction. Data visualization can be considered as a paramount element in the exploratory phase of any data mining process (1). This task is often about transforming non-visual quantitative information into visual information. Visualization, as a *natural pattern recognition* process, is meant to guide us to draw explanatory or exploratory inferences. In fact, an adequate visualization can become an inductive hypothesis about the analyzed data.

With high-dimensional data, visualization cannot be done directly, but requires some form of dimensionality reduction, be it through feature selection or feature transformation and extraction. This is not a rare situation: high-dimensional data sets are becoming commonplace in many of today's most active scientific research endeavors. Examples of this are, to name a few, bioinformatics, natural language processing, or web mining.

Nonlinear dimensionality reduction (NLDR) (2) is a powerful, flexible and ultimately useful strategy for data modeling and exploration. By its own nature, it is also, indeed, a natural way to deal with high-dimensional data, for which readily intuitive insights about inner structure are hardly ever available.

One of the model families belonging to the wide palette of NLDR techniques is that which includes manifold learning methods. These methods attempt to represent multivariate data assuming they can be approximated reasonably well by low-dimensional manifolds covering the most densely populated areas where data reside. When data modeling focuses on exploration, these manifolds are chosen to be 2-dimensional to provide the model with data visualization capabilities. Note that the observed data might well have intrinsic dimensionalities higher than

2, but, when the goal of our analyses is exploratory visualization, a trade-off between representation faithfulness and feasibility must be reached.

A general drawback limiting the use and popularization of NLDR has to do with the interpretability of the resulting data representation. Even manifold learning models, which can represent high-dimensional data in one, two, or three dimensions, are not necessarily that straightforward to interpret. This is because their new coordinates of representation usually are a complex transformation of the observed ones. Different parts of the original observed data space may undergo different levels of compression and stretching as part of the mapping process. Consequently, the data representation may suffer strong distortion, even if assuming that no serious manifold discontinuities occur.

The potential lack of interpretability is the price paid by NLDR methods for their superior ability to faithfully represent multivariate data (3). Linear dimensionality reduction methods are less flexible in the transformation they entail and, therefore, their representation of high-dimensional data can be less faithful. Compensating for this, their subset of representation coordinates can be expressed as a linear combination of the original coordinates (that is, of the observed data attributes), which often makes these models easy to interpret without resorting to not-too-intuitive post-processing procedures. This comes to explain the popularity, for instance, of a method such as Principal Component Analysis (PCA) (4), which, despite its shortcomings and the fact that it was defined over a century ago, is still widely used.

1.1 Objective of the Thesis

The limitations in interpretability, described in the previous section, are consubstantial to NLDR methods. This highlights the importance, as a research goal, of defining novel approaches to circumvent such problem. One possible approach, and the one which is at the core of this Master's Thesis, is making the nonlinear distortion introduced by the models visually explicit. For some NLDR models, indirect quantitative measures of this distortion can be calculated explicitly and used as part of an interpretative post-processing of the results.

More specifically, we will draw inspiration from a technique originally devised for geographic information, known as *cartograms*. Geographers have traditionally been at the forefront of information visualization. Cartograms are geographic maps in which the sizes of regions such as countries or provinces appear in proportion to profiling quantities such as, for instance, their population and their economic indicators.

They have a limitation in that, to scale regions while not losing their continuity (that is, while preserving the integrity of their borders), their shapes must be distorted in one way or another, potentially resulting in maps that are not obvious to read. A technique inspired on physics theory was recently proposed in (5) for building robust cartograms. It retains the interpretability of maps while distorting them, but without suffering other drawbacks, like the undesired overlapping of regions.

The extrapolation proposed in this thesis consists in substituting the geographical maps by the *virtual geographies* created by some NLDR models with their data-driven, group-defining borders and underlying quantities such as nonlinear distortion. These virtual geographies are then to be represented by means of cartograms, adapting the techniques described in (5) to the peculiarities of NLDR methods. We argue that this type of representation will at least partially improve the interpretability of the NLDR methods, thus easing their use. This is because the resulting cartogram-based data representation is a transformation that encodes some intended inferential bias when viewed by the analyst, namely the inferential bias of a surface-size correlate of the NLDR mapping distortion.

NLDR methods are manifold and the subject of intense research over the last decade (2). It thus makes no sense trying to cover the complete palette of techniques at hand. The experiments reported in this thesis focus on one of the most popular families in NLDR, namely Self-Organizing models.

1.2 Structure of the Thesis

Beyond the introduction, the current document is structured in the following chapters:

- **Background**: In this chapter we provide a general introduction to NLDR methods for visualization with special attention to Self-Organizing Maps (SOM), a family of models proposed in the early ’80s by Prof. Teuvo Kohonen (6), and to some of its variants. These variants include the Emergent SOM, Neural Gas, the Growing SOM and the Hierarchical SOM, concluding with the Growing Hierarchical SOM, which is the focus of the experimental chapters. We then review the general theme of distortion measures in NLDR methods (including the concepts of U-matrix, P-matrix and U*matrix).

- **Cartogram Representation of Mapping Distortion in SOM-Based Methods**: In this chapter, the *Cartogram*-based visual representation of multivariate data modelled by

NLDR methods is introduced. This includes a reasonably detailed account of distortion and cartogram methods. We show how to apply cartograms to the representations of NLDR distortion measures in a visualization space, providing some specific examples.

- **Experiments with Synthetic Data**: This chapter contains the core experimental contribution of the thesis, and it is a somehow detailed investigation of the cartogram representation of SOM-based data visualizations. The evaluation is performed through a wide array of experiments, whose results are reported and discussed at length.

- **Experiments with Real Medical Data**: Once evaluated using synthetic data, the cartogram representation is further illustrated using real data stemming from a neuro-oncology problem. It involves the discrimination of human brain tumour types, a problem for which knowledge discovery techniques in general (7), and data visualization in particular (8) are useful tools.

- **Conclusions**: This thesis concludes with a brief summary of its contributions and some directions for future research.

Chapter 2

Background

This chapter provides the technical background that will be necessary not only to understand the following chapters, where we report our proposed developments and test them experimentally, but also to provide readers with a wider framework in which to locate these developments. We start by considering the problem of data visualization in general, and then we move to the more specific problem of nonlinear dimensionality reduction. This is followed by an overview of the well-known SOM algorithm and some of its main variants. The chapter closes with a description of the quantitative measurement of mapping distortion in NLDR models.

2.1 Data Visualization through Dimensionality Reduction

As stated in the introduction, data visualization is a key component of data exploration and, as such, of the data mining process as a whole. An appropriate visualization of multivariate data should help the analyst to draw explanatory or exploratory inferences. As a result, it could become the source of inductive hypotheses about the analyzed data (1).

The origins of computer-based data visualization are to be found in the early days of computer graphics, back in the 1950's, when the first graphs and figures were generated by computers. The rapid increase in computing power in the following decades and the advent of the personal computer allowed data analysts and statisticians to work with larger data sets and, in the 1990's, saw the first formalization of *Information Visualization* as a research field of its own.

More recently, visualization has grown and advanced to deal with data from very heterogeneous sources, including the increasingly large data collections to be found in business and finance, administration, digital media (including social networks), etc. The concept of *Data Visualization* has thus become commonplace in the scientific and information visualization fields. Today, data visualization has become a very active area of research with manifold applications (9).

The pervasiveness of digital information storage systems makes the collection of large databases (containing text, numerical information and multimedia information) commonplace in many real world problems. An example of that, with rapidly growing social implications, is the existence of large databases in medicine and bioinformatics (fueled by increasingly sophisticated measurement techniques and devices), demographics, finance (with increasingly complex products), or the Internet.

In many of these domains, one of the main characteristics of the data is their considerably high dimensionality (sometimes coupled with data sparseness understood as a phenomenon that concerns small data samples). Clear examples of this are image and video databases, where the data consist of a set of objects, and the high dimensionality is a direct result of trying to describe the objects via a collection of features. The application of data modeling techniques to this type of data is less than obvious and their meaningful visualization is equally challenging.

This situation invites us to consider the development of suitable, context-specific techniques to reduce the data dimensionality, so that the data can be represented in low-dimensional spaces. The problem of finding the intrinsic dimensionality of one such data set is, by itself, an area of active research. Low-dimensional data representations should have the dimensionality that corresponds to the minimum number of variables required to explain the observed properties of the original data (10). Unfortunately, the human eye is limited to cope with a handful of features when the goal is inductive inference from visualization.

The dimension of the data embedding can be a key parameter specially for manifold projection methods: if the dimension is too small, important data features may “collapse” onto the same dimension, and if the dimension is too large, the projections become noisy and, in some cases, unstable. There is no general consensus, however, on how this dimension should be determined (11).

The intrinsic dimension (ID) of a data set is usually defined as the minimal number of parameters (or latent variables in latent models) needed to describe it. ID is helpful for data visualization (and also for classification), as it should clarify the number of variables required to represent the data adequately (12). More often than not, data sets have redundant variables and this variables can be removed without much loss of relevant information; alternatively, many features are highly correlated and we can extract new alternative features that summarize the observed ones. This is the field of work of dimensionality reduction (DR), which is the key for the analysis of high-dimensional data from a visualization viewpoint.

In a general context, consider a data set represented by a matrix in which each row represents a set of attributes or dimensions that describe a particular instance of a measured phenomenon. The problem of DR for this matrix could be defined as follows. Assume we have a data set represented by a $N \times M$ matrix X, consisting of N data vectors x_i where $i = (1, \ldots, M)$ and x_i is the ith row of the data matrix X.

Assume also that this data set has an ID of value m (where $m < M$, and often $m \ll M$). can be related with others and they are'nt necessaries to the new manifold with dimensionality m, which is embedded in the M-dimensional space. DR techniques are meant to transform data X with dimensionality M into a data set Y with dimensionality m, while trying to preserve the main characteristics of the observed data as much as possible.

The low-dimensional counterpart of x_i is denoted by y_i. Even if the ID can be used to obtain a parsimonious while faithful lower-dimensional representation of our data, this may not be enough for data visualization purposes. If that is the case, a trade-off between representation faithfulness and feasibility must be reached and we will have to assume that a very low-dimensional transformation of the analyzed data may entail losing some relevant information.

The existence of an ID for a given data set should reveal the existence of topological structure in the data (13). However, neither the ID nor the topological properties are likely to be easily identified from just a finite set of data points. Many DR *projection techniques* search the best subspace in which to project the data by minimizing some type of projection error. These techniques could roughly be categorized into two groups: Linear DR and NLDR methods (12).

The use of linear DR methods is widespread in all types of scientific research fields. One of their obvious advantages is that the subset of representation coordinates they produce can be expressed as a linear combination of the original observed data attributes. This makes these models easy to interpret because they convey new knowledge in a way that does not require much post-processing of the results. Also, they have a non-trivial advantage over NLDR techniques: when data are well separated from a particular projection (revealing grouping structure), such gap cannot close when the dimensionality of the projection is increased, so that is easy to infer that it is the result of true separation in the observed data.

Linear methods, in any case, are less flexible in the transformation they entail and, therefore, their representation of high-dimensional data can be less faithful than that provided by NLDR techniques. A very popular linear DR method is PCA, which is typically applied in practice using biplots (4). This approach has clear and well-reported limitations, such as its sensitivity to the presence of uninformative noise, the lack of a robust criterion for choosing the adequate number of PCs. Often, these are compensated by the interpretability properties described in the previous paragraph.

Other popular linear method that is often used for data visualization, despite the fact it was mostly defined as a classifier technique is Linear Discriminant Analysis (LDA), which computes a data transformation (projection) by minimizing the within-class variance and maximizing the between-class variance simultaneously, achieving, class discrimination as a result. The optimal transformation in LDA can be readily computed by applying an eigen-decomposition of the scatter matrices (14).

Many relevant contributions to multi-variate data (MVD) visualization have stemmed from the field of nonlinear DR (2) and, more in particular, from spectral-based methods (15, 16) and techniques of the manifold learning family. These include methods for the quantification and visualization of the quality of the DR process (17). Manifold learning attempts to describe (usually high-dimensional) MVD through nonlinear low-dimensional manifolds embedded in the observed data space. These manifolds generate a model by “wrapping around” data while usually preserving their continuity and smoothness properties.

Almost as popular in nonlinear DR for visualization as PCA is in linear DR, SOM (6) and its many variants attempt to model MVD through a discrete version of a manifold consisting of a topologically-ordered grid of cluster centroids. The nonlinearity of these methods entails the existence of different levels of local distortion in the mapping of the data from the observed space into the visualization space. Given that most of these methods rely upon the definition of inter-point distances (Euclidean being the most commonly used) in the metric spaces they deal with, there is no guarantee that the inter-point distances in the observed data space will be uniformly reflected in the visualization space. In other words, points which are distant in the observed data space may end up being represented as closely located in the visualization space and the other way around.

These manifold stretching and compression effects can be understood as geometrical distortions introduced by the nonlinear mapping (18). Such effects can also be seen as a local magnification process. The data representation flexibility provided by nonlinear DR methods often makes them more faithful models of the observed MVD than linear ones. The price that these methods must pay for such ability is the usually less straightforward interpretability of the visualizations they provide (3), given that the coordinates of visual representation are no longer linear combinations of the original data attributes.

This limitation of NLDR methods makes the definition of approaches to attenuate it a research goal on its own right. The following chapters propose a method for explicitly reintroducing the geometrical distortion created by a nonlinear DR manifold learning model into its low-dimensional representation of the MVD. For that, we draw inspiration from a technique originally devised for the analysis of geographic information, namely density-equalizing maps, or cartograms (5). It will be applied to variants of the SOM algorithm, which is summarily introduced in the following paragraphs.

2.2 Self-Organizing Maps

The most emblematic and popular NLDR method is undoubtedly Kohonen's SOM, a type of artificial neural network with two layers, input and output, that is trained using unsupervised learning procedures to generate a low-dimensional (typically two-dimensional) output visualization space.

SOM has extensively been used in numerous applications of data analysis and can be interpreted intuitively as a kind of nonlinear but discrete PCA where a hyperplane is fitted to the data cloud, in such a way that points are encoded through centroids residing on that hyperplane. One way to put this idea in practice consists on replacing the continuous hyperplane with a discrete (and bounded) representation. For example, a grid or lattice defined by a finite number of points.

The task it performs is easy to understand. SOM simultaneously performs the combination of two subtasks: vector quantization (VQ) and topographic mapping (i.e., dimensionality reduction). VQ can be understood as a way to reduce the size of a data set by replacing data points by representative prototypes of the data. Therefore, VQ and DR are somewhat complementary techniques and is noteworthy that several DR methods use vector quantization as preprocessing. In practice, VQ is achieved by replacing the original data points with a smaller set of points called centroids, prototypes or weight vectors in the output space, sometimes called also the codebook (2).

These are linked to a grid of units(neurons). Each of the units on the map has assigned a weight vector, which is of the same dimensionality as the vectors in the input space. During the training process (the learning), the vectors from the input space are presented to the SOM, and the unit with the most similar weight vector to this input (the one whose distance to the input vector is the lowest) is selected as winner. After that, the weight vector of this unit, and the neighboring ones, are adapted as much as possible to the input, that is, their distance in the input space is reduced. For this, SOM incorporates a nonlinear *neighborhood function*. This is not uniquely defined, although Gaussian type functions are the most commonly used.

As a result of this training process, the output space will be arranged in a way that fits the input space as closely as possible.

An important property of the SOM is that the mapping preserves the topology of the input data, that is to say, elements which are located close to each other in the input space will commonly be closely located in the output space, while the dissimilar will be mapped on opposite regions of the map (20), taking in consideration the physical arrangement of the nodes. This feature makes the SOM very useful in data analysis and data visualization where a common goal is to represent data from a high-dimensional space in a low-dimensional space so as to preserve the internal structure of the data in the input space. Preserving neighborhood in the mapping makes the exploration of the output visualization space and the investigation of the structure hidden in the high-dimensional data, such as clusters, a more intuitive undertaking.

The SOM algorithm does not make any assumption about the input data distribution. This algorithm uses a set of neurons, often arranged in a 2D rectangular or hexagonal grid (output space), to form a discrete topological mapping of an input space $X \in \mathbb{R}^m$. At the start of the learning, all the weights $w_{r1}, w_{r2}, ..., w_{rm}$ (where w_{ri} is the weight vector associated to neuron i and is a vector of the same dimension, n, that the input space) are initialized to small random numbers or to conform a linear shape, being m the total number of neurons and ri the weight vector of neuron i on the grid. Then, the standard algorithm repeats the steps described in the following subsection (21).

2.2.1 The SOM Algorithm

1. At each time t, present an input, *x(t)*, select the winner,

2. Calculate the best matching unit (BMU) in terms of the Euclidean norm of the difference between two vectors, being Ω the set of neuron indexes and $w_{\overrightarrow{k}(t)}$ the weight vector

associated to the neuron k at time t, it is to say, *x(t)*.

$$bmu = \arg\min_{k \in \Omega} \| x(t) - w_{\vec{k}(t)} \| \tag{2.1}$$

3. Updating the weights of winner neuron and its neighbors:

$$\Delta(t) = \alpha(t)\eta(bmu,k,t)[x(t) - w_{bmu}(t)] \tag{2.2}$$

$$w_{bmu}(t+1) = w_{bmu}(t) + \Delta(t) \tag{2.3}$$

where $\eta(bmu,k,t)$ is the neighborhood function. If selected Gaussian then:

$$\eta(bmu,k,t) = \exp[-\frac{\|bmu-k\|^2}{2\sigma(t)^2}], \tag{2.4}$$

with σ representing the effective changing range of the neighborhood. The coefficient $\alpha(t), t > 0$ is a scalar-valued learning rate, that decreases monotonically satisfying:

(i) $0 < \alpha(t) < 1$

(ii) $\lim_{t \longrightarrow \infty} \sum \alpha(t) \to \infty$

(iii) $\lim_{t \longrightarrow \infty} \sum \alpha^2(t) < \infty$, or a less restrictive: $\lim_{t \longrightarrow \infty} \sum \alpha(t) \to 0$

4. Repeat from step 1 until some convergence criterion is met, with decreasing neighborhood kernel (21, 22).

One must be sure that the mapping has been correctly estimated. For this purpose, there are different measures to quantify the goodness of a map. The accuracy of the maps in preserving the topology, or neighborhood relations, of the input space has been also measured in various ways. During the training, we have to make assumptions about several parameters of the map, such as learning parameters, map topology and map size. These features influence the final map, thus it is very important to choose these parameters carefully in order to reach the appropriate one.

Once different choices have been tested, some measure can be used to evaluate the quality of the map and select the optimal one to represent the data. Several measures have been used to evaluate the quality of a SOM. A very used measurement is the *quantization error* (QE) which measures the approximation quality of the map, i.e. the distance between each data vector and its BMU.

Normally, the number of data items is far greater than the number of neurons and the precision error is always different from zero. The average of QE (MQE) is calculated as shown in Eq. 2.5, where N is the number of data-vectors and $bmu_{\overrightarrow{x}_i}$ is the best matching prototype of the corresponding $\overrightarrow{x}_i$ data-vector.

$$qe = \frac{1}{N} \| \overrightarrow{x}_i - bmu_{\overrightarrow{x}_i} \| \tag{2.5}$$

Thus, the optimal map is expected to yield the smallest MQE. A smaller quantization error signifies that the data vectors are closer to their prototypes. But, what happens with the topological preservation? (23).

Topology preservation (TP) has, however, turned out to be a quite difficult to define for a discrete grid. There seem to exist two different approaches for measuring the degree of TP. In the first approach, the relations between the reference vectors and the relations between the corresponding units on the map lattice are compared as the topographic product does.

An alternative approach for measuring TP is to use input samples to determine how continuous the mapping from the input space to the map grid is. One of the most extended indices for this purpose is the *topographic error* (TE). It is also one of the errors proposed by Kohonen himself. This error measures the proportion of all data vectors for which first and second BMUs are not adjacent vectors. So the lower TE is, the SOM that preserves the topology.

The TE is calculated as

$$te = \frac{1}{N} \sum_{i=1}^{N} u(\overrightarrow{x_i}), \tag{2.6}$$

where the function $u(\overrightarrow{x_i})$ is 1 if $\overrightarrow{x_i}$ data vector's first and second BMUs are adjacent and, 0 otherwise (23).

One form of representation of a SOM that allows to get a more suitable picture of the vector distribution and the distortion of the space undergone in the mapping process is the *U-matrix* (unified distance matrix), which is a technique that shows the distances between neighboring prototype vectors. It is the most common method associated with SOM although alternative methods have been proposed: the gradient field has some similarities with the *U-matrix*, but applies smoothing over a larger neighborhood and uses a different style of representation. The

U-matrix, normally uses the Euclidean distance between the codebook vector of the neighboring neurons in a gray scale image and for each unit in the output space, there is a corresponding element u_{ij} in the *U-matrix*. In some cases, it is also common to use the *U-matrix* in such a way that the value of a particular node is the average distance between the node and its closest neighbors.

Other simpler visualization techniques take into account the distribution of the data too, for example hit histograms, but the *U-matrix* contains therefore a geometrical correct approximation of the vector distribution in the Kohonen net. To get a visual impression of how this distribution is, the better way to display the *U-matrix* is in no more than two or three dimensions (24).

2.2.2 *U-matrix*

A SOM is a self-organizing projection from the high dimensional data space onto a low-dimensional grid of neuron locations. The grid of neurons is usually embedded in a two dimensional manifold. This space is called a map with a geographical interpretation in mind. The learning algorithm of the SOM is designed to preserve the neighborhood relationships of the high dimensional space on the map. Therefore the map can be regarded as a “roadmap” of the data space.

The U-Matrix is constructed on top of such map. Let n be a neuron on the map, $NN(n)$ be the set of immediate neighbors on the map, $\mathbf{w}(n)$ the weight vector associated with neuron **n**, then $U_{height(n)} = d(\mathbf{w}(n), \mathbf{w}(m))/m \in NN(n)$, where $d(\mathbf{x,y})$ is the distance used in the SOM algorithm to construct the map. The *U-matrix* is a display of the $U_{heights}$ on top of the grid positions of the neurons on the map (25). A *U-matrix* is usually displayed as a grey level picture or as three dimensional landscape, in both cases displaying the local distance structure of the data set. Properties of the U-matrix include:

- The position of the projections of the input data points reflect the topology of the input space.
- Weight vectors of neurons with large U-heights are very distant from other vectors in the data space.
- Weight vectors of neurons with small U-heights are surrounded by other vectors in the data space.

- Projections of the input data points are typically found in depressions.
- Outliers in the input space are found in “funnels”.
- “Mountain ranges” on a U-Matrix point to cluster boundaries.
- “Valleys” on a U-Matrix point to cluster centers.

Using the SOM/*U-matrix* methods for clustering has the advantage of disentangling nonlinear complex cluster structures. *U-matrices* have been used in a number of applications to detect new and meaningful knowledge in data sets. To name a few: sea level prediction, DNA microarray analysis, customer segmentation in mobile phone markets, stock portfolio selection (25).

2.2.3 *P-matrix*

The concept of *P-matrix* was introduced as an extension of the *U-matrix* (26) to represent, instead of the local distances, density values in data space measured at the neurons weights are used as height values. The P_{height} of a neuron n, with associated weight vector $\mathbf{w}(n)$, is defined as: $P_{height(n)} = p(\mathbf{w}(n), X)$, where $p(\mathbf{x,X})$ is an empirical density estimation at point $\boldsymbol{x}$ in the data space $\boldsymbol{X}$.

For each neuron *n* of a SOM, the *P-matrix* displays the density measured in the data space at point ***w(n)***, where *w(n)* is the weight vector associated with neuron *n* of the ESOM or SOM. In principle, any density estimation, which works for the input data set of the SOM can be used. A commonly used density estimation is the *Pareto Density Estimation* (PDE). PDE calculates the density at some point *x* as the number of points inside a hypersphere (Pareto sphere) around $\boldsymbol{x}$. The radius of the hypersphere is called the *Pareto radius*. It has been shown that PDE leads to a meaningful density estimation and it fits nicely into the SOM *U-Matrix* calculation. A *P-matrix* is defined in the same manner as an *U-Matrix*. The *U-matrix* reveals the (local) distance structures, while the *P-matrix* gives insights into the density structures of a high dimensional data set. The elements of a *P-matrix* are called *P-heights*.

Properties of a *P-matrix* include:

- The position of the projections of the data on the SOM reflect the topology of the input space. This is inherited from the underlying SOM algorithm.
- Neurons with large *P-heights* are situated in dense regions of the data space

- Neurons with small *P-height* are “lonesome” in the data space
- Outliers in the input space are found in “funnels”.
- “Ditches” on a *P-matrix* point to cluster boundaries
- “Plateaus” on a *P-matrix* point to regions with equal densities one can see, that many, but not all, properties of the *P-matrix* are the inverse of an *U-matrix* display. In contrast to the *U-matrix*, which is based on the distance structure of the data space, the *P-matrix* is based on the data density structure and this gives a new and complementary insight into the high dimensional data space (25).

2.2.4 U*-matrix

In dense regions of the data space, the local distances depicted in an *U-matrix* are presumably distances measured inside a cluster. In this populated regions of the data space, however, the distances matter. In this case, the *U-matrix* heights correspond to cluster boundaries. This leads to the definition of an *U*-matrix* which combines the distance based *U-matrix* and the density based *P-matrix*. The *U*-matrix* is derived from an *U-matrix* as follows:

- When the data density around a weight vector of a neuron is equal to the average data density, the heights shown in an *U*-matrix* should be the same as in the corresponding *U-matrix*.
- When the data density around a weight vector of a neuron is big, local distances are primarily distances inside a cluster. In this case the *U*-matrix* heights should be low.
- When the data density around a weight vector of a neuron is lower than average, local distances are primarily distances at a border of a cluster. In this case the *U*-matrix* heights should be higher than the corresponding U-height. This leads to the following formula: let *U-height(n)* denote the *U-height* of a neuron *n*, *mean(P)* denote the mean of all *P-heights*, *min(P)* the minimum of all *P-heights*, then the *U*-height* of an *U-Matrix* for neuron *n*, the *U*-height(n)*, is calculated as:

$$U*-height(n) = U-height(n)*ScaleFactor(n) \tag{2.7}$$

with

$$ScaleFactor(n) = \frac{(Pheigt(n) * mean(P)}{mean(P) * max(P)} + 1 \tag{2.8}$$

From the previous definition it follows that:

- $P - height(n) = mean(P - heights) \Longrightarrow U * -height(n) = U - height(n)$
- $P - height(n) < mean(P - heights) \Longrightarrow U * -height(n) > U - height(n)(intercluster)$
- $P - height(n) > mean(P - heights) \Longrightarrow U * -height(n) < U - height(n)(intracluster)$
- $P - height(n) = max(P - heights) \Longrightarrow U * -height(n) = 0(intracluster)$[1]

2.2.5 Batch-SOM

The *Batch-SOM* algorithm is a variant of SOM the that, unlike basic SOM, updates the weight vectors only at the end of a learning iteration (i.e. after the complete training set has been imputed). It does so using the equation:

$$w_k(t_f) = \frac{\sum_{t=t_0}^{t_f} h_{ck}(t)x(t)}{\sum_{t=t_0}^{t_f} h_{ck}(t)} \tag{2.9}$$

where $W_k(t_0)$ is the neuron weight vector calculated at the end of the previous stage. The neighborhood function remains as presented in Eq. 2.2. The learning-rate factor, the coefficient $\alpha(t)$ does not exist in the Batch method and t_0 and t_f mean, respectively, the beginning and end of the current training stage. The winner neuron will be found with the following equations:

$$d_k(t) = \|x(t) - W_k(t_0)\|^2 \tag{2.10}$$

$$d_c(t) = \min_k d_k(t) \tag{2.11}$$

Thus Batch-SOM algorithm can be summarized as follows:

[1] for a discussion of the ScaleFactor see Appendix 1 in (25)

1. Find the BMU for an input vector x(t), using the Eq. 2.10 and 2.11 and accumulate numerator and denominator of Eq. 2.9 for all neurons.
2. Update neuron weights with the Eq. 2.9.
3. Repeat from step 1 until some convergence criterion is met, with a narrowing neighborhood function.

In large problems, the ***Batch*** algorithm has the advantage to be an order of magnitude faster that the ***Sequential*** SOM. A particular difference between both methods is that the weights are not updated immediately and there is no dependency on the order that input vectors are presented to the network. This also eliminates the concern that the last input vectors presented influence the final results (22). However, several researchers clearly state that on-line training is faster that batch training, especially in pattern recognition problems with large training sets. Some researchers have noticed the superiority of on-line training for redundant data too (26).

2.3 Variants of the Standard SOM Algorithm

2.3.1 Emergent SOM

The Emergent SOM (ESOM) is a variant of the basic SOM that also takes a set of high dimensional data points and maps them onto a low dimensional grid, called map. This grid is less restrictive than in the basic algorithm and consists of an almost arbitrarily large number of prototypes, typically in the thousands. An ESOM thus differs from a traditional SOM in that a very large number of neurons (at least a few thousands) are used.

In rectangular grids, the number of immediate neighbors of a neuron is 4, while in hexagonal grids there are 6 immediate neighbors. Nevertheless, at the borders of the grid, the number of immediate neighbors is less as shown in Fig. 2.1.

In these map spaces, border effects occur, increasing the probability of topology errors. To avoid such border effects, grids can be embedded in a finite but boundless space such as, e.g., a sphere or a toroid. In a toroid, the top row is connected to the bottom row and the left column to the right column within the lattice. But, the concept of border less maps (e.g. toroid maps (27) to avoid border effects is rarely used. Emergent phenomena involve, by definition, a large number of items, where large means at least a few thousands. This is why large SOMs called ESOM were defined to emphasize the distinction.

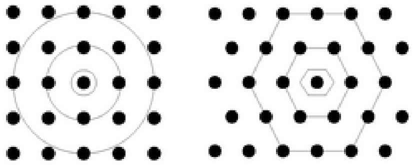

Figure 2.1: Rectangular grid (left) and hexagonal grid (right)

The 2-D grids can be square or rectangular. If the number of rows and columns is equal, the map is called square, otherwise rectangular. The ratio of rows to columns is proposed to be chosen according to the ratio of the first and second eigenvalues of the covariance matrix. An experimental analysis of topology errors showed, however, that it is convenient to chose the ratio of rows and columns to be different from unity even when no dominant direction of variance exists. In (28), the authors recommend the following ESOM architecture: boundless toroid grids with at least 4,000 neurons and a ratio of rows and columns different from unity. This is reported to avoid border effects, topology errors (the topology preservation of the SOM projection is of little use when using small maps), and enable an intuitive undistorted visualization.

When using supervised neural nets, e.g. Multi Layer Perceptrons, a common concern is the model size. Too small neural nets have low accuracy, while too large nets are prone to overfitting. However, this is not the case with ESOM. Using larger maps does not really increase the degrees of freedom in the same sense, because the neurons are restricted by the topology preservation of the map. Using large maps should rather be viewed as increasing the resolution of the projection from the data space onto the map (28).

2.3.2 Growing SOM

The Growing Self Organizing Map (GSOM) model is an architecturally dynamic variant of the SOM and was developed to solve some of its weaknesses, which included the problems of identifying the correct dimensions (height and width) of the rectangular map. The GSOM preserves a rectangular grid, which keeps representation straightforward. The difference with an SOM is that the number of rows and columns of the map grid is a variable whose optimal value is to be estimated.

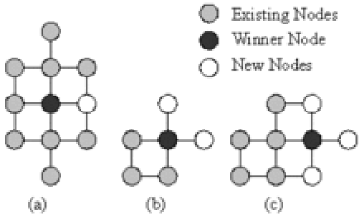

Figure 2.2: Growing-SOM and New nodes in GSOM.

Growing (or incremental) models have no predefined structure. At first, this makes them more complicated than networks with static structure, whose topology must be chosen a priori and does not change during parameter adaptation. For growing networks, however, suitable node insertion strategies have to be defined, as well as criteria to stop the growth.

GSOM starts with a minimal number of nodes (usually 4) and grows only from the boundary of the network to adapt the data set. The new nodes are grown only from the boundary of he network as illustrated in Fig. 2.3.

The size of a GSOM is controlled by a parameter called growth threshold(GT), which is defined as:

$$GT = -D * ln(SF), \tag{2.12}$$

where D is a dimensionality of the data an SF is a user-defined spread factor, a term unique to the GSOM that takes values between 0 and 1, being 0 for minimum growth and 1 the maximum growth.

The GSOM algorithm can summarized in the following steps:

1. Initialize the weight vectors of the starting nodes with random numbers closer to 0.5.

2. Calculate the Growth Threshold (GT) of the inputs with Eq. 2.12. The SF determines the level of spread required by the map. A higher SF value will give a wider spread and more detailed clusters while a lower SF gives more summarized clusters.

3. Present one of the inputs to the network.

4. Find the winner node q from the current nodes using the following expression:

$$q = \arg\min_i \|x - w_i\|, \tag{2.13}$$

where $\|x - w_i\|$ is the distance between the input x and the weight vector of node i, w_i measured in the D dimensional space.

5. Calculate the $Error = \|x - w_q\|$

6. If $Error > GT$ and q is a boundary node, grow new nodes from q such that its neighborhood is completely filled. An example can be found in Fig. 2.3.

7. Else perform a weight adaptation to the winner and its neighbors similar to that of the standard SOM.

8. Repeat steps 3-7 until all the data items have been imputed to the network and the specified number of algorithm iterations have elapsed (29).

At inception, GSOM was mostly applied to small dimensional data sets but, recently, it has also been evaluated on very high-dimensional data (29).

2.3.3 Hierarchical SOM

Many methods based in SOM have attempted to expand its capabilities beyond the provision of a single data mapping. They include Multi-Layer SOMs, Multiresolution-SOMs, Multi-stage SOMs, Fusion SOMs and Tree-SOMs. In one way or another, all this techniques remit to the concept of hierarchical clustering.

A Hierarchical SOM (HSOM) is a method that includes several layers of SOMs and where the output of each one is used to feed another. That is to say, when a data point is present at the first level of the hierarchy, it can be present at second level giving the index and coordinates of its BMU, the QE and all parameters of activation for the first level. The important issue, is that the output of first level is used to train the second level.

Many configurations are possible for a HSOM. They may vary in the number of levels used, in the way the connections are established and even in the information sent through each connection (30).

There are two main reasons that may motivate using a HSOM instead of a standard SOM:

- A HSOM can require less computational effort than a standard SOM to achieve certain goals. This can be made in two ways:

1. Reducing the dimensionality of the inputs to each individual SOM using several SOMs, each using a subset of the components of each input vector.
2. Reducing the number of neurons in each SOM, given the reduced the number of variables imputed. This way, the distance functions used for training the different SOMs will be simpler and faster to compute.

- HSOMs are often used in application fields where a structured decomposition in smaller problems is convenient. That is, when hierarchical cluster structure may be expected in the data, which is a commonplace situation. An HSOM can therefore be better suited to model a problem that has, by its own nature, some kind of hierarchical structure (30).

2.3.4 Neural Gas

Neural gas (NG) is an artificial neural network inspired by the SOM (31). It is a simple algorithm that combines vector quantization with *soft competition* between the units. It was coined "neural gas" because of the dynamics of the feature vectors during the adaptation process, distribute like a gas in the data space (32).

The NG algorithm aims to find optimal data representations based on feature vectors. Given a set of data vectors x from the input space and a finite number of feature vectors or prototypes $m_k, k = 1, \ldots, N$, in each training step the Euclidean distances between a randomly selected input vector x_i from the training set x and all prototypes m_k are calculated as

$$d_{ik} = \|x_i - m_k\|^2 = (x_i - m_k)^T * (x_i - m_k) \tag{2.14}$$

The vector of these distances is d. Each prototype k is assigned a rank $r_k(d) = 0, \ldots, K-1$, where a rank of 0 indicates the closest and a rank of *K-1* the most distant prototype to x. Then, at time *t+1* the m_k is adapted according to the following learning equation:

$$m_k^{t+1} = m_k^t + \varepsilon * h_\rho[r_k(d)] * (x_i - m_k^t), \tag{2.15}$$

where

$$h_\rho(r) = \exp^{(-r/\rho)} \tag{2.16}$$

is a monotonically decreasing function of the ranking that adapts all the prototypes, with a factor exponentially decreasing with their rank. The scope of this influence is determined by the neighborhood range ρ. The learning process is also affected by a global learning rate ε (33). The values of ρ and ε decrease exponentially from an initial positive value ($\rho(0)$, $\varepsilon(0)$) to a smaller final positive value ($\rho(T)$, $\varepsilon(T)$) according to

$$\rho(t) = \rho(0) * [\rho(T)/\rho(0)]^{(t/T)} \tag{2.17}$$

and

$$\varepsilon(t) = \varepsilon(0) * [\varepsilon(T)/\varepsilon(0)]^{(t/T)} \tag{2.18}$$

After sufficient adaptation steps, the feature vectors cover the data space with minimum representation error. This method is applied when data compression or vector quantization is an issue, for example in speech recognition and image processing (32).

2.3.5 Growing Hierarchical SOM

As previously mentioned, the standard SOM learning algorithm suffers from some structural limitations. The first, as stated in the introduction, is that the standard model provides a single *flat* cluster partition. This may not suffice to correctly characterize real data in many practical situations, as these data are expected to show grouping structure at different levels of detail that can only be correctly described through hierarchically organized partitions. This led, quite early on, to the definition of the hierarchical SOM (34). For a review on hierarchical models for clustering and visualization, see, for instance, (35).

A second limitation of the standard model is its fixed, static architecture, which must be defined prior to data observation. In order to find adaptive SOM architectures that are suited to the specificity of the data, thus avoiding tedious processes of trial-and-error, several authors have designed grid-growing algorithms, usually known as GSOM, which automatically define the adequate size of map in the form of an appropriate height-to-width ratio (36, 37).

To overcome these limitations of the basic standard model, Rauber and colleagues (38) proposed a simultaneously growing and hierarchical artificial neural network architecture, called GHSOM. It uses a hierarchical structure of multiple layers, where each layer consists of a number of independent SOMs.

Starting from a top-level map, each individual map, similar to the growing grid SOM model, grows in size to represent a subset of data at a specific level of detail. After a certain improvement regarding the granularity of data representation is reached, the units are individually analyzed for data diversity. Those units representing too-diverse input data are then expanded

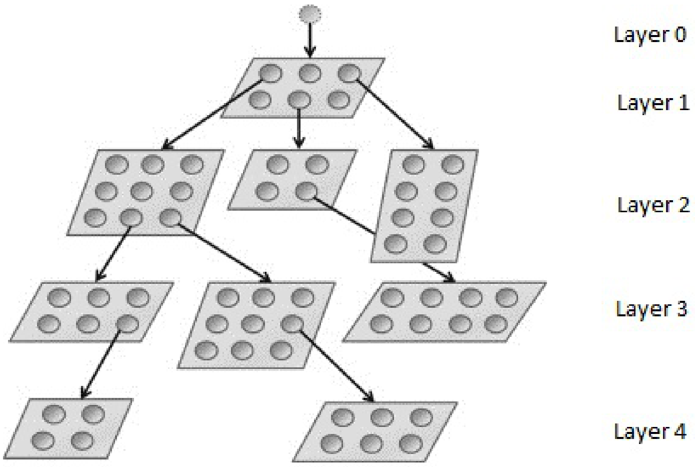

Figure 2.3: Illustration of the architecture of the GHSOM model (Adapted from (38)).

to form a new small growing SOM at a subsequent layer in the hierarchy, where the subset of data is represented in more detail. These new maps again grow in size until a specified improvement of the quality of data representation is reached. Units representing an already rather homogeneous enough set of data, will not require any further expansion into subsequent layers. The resulting GHSOM should therefore reflect, by its very architecture, the hierarchical structure inherent in the data, allocating more space for the representation of inhomogeneous areas in the input space. The topology of GHSOM is illustrated in Fig. 2.3.

In GHSOM, each unit determines its activation according to the Euclidean distance between its weight vector and the input pattern. The unit with the smallest distance is denoted as the winner, and several units in the vicinity of the winner are adapted.

This training process is repeated for a fixed number N of training iterations. After N training iterations, the unit with the largest deviation between its weight vector and the input patterns represented by this very unit is selected as the *error unit* **e**. Then, either a new row or column of units is interpolated between **e** and its most dissimilar neighbor ***d***. The weight vectors of these new units are initialized as the average of their neighbors.

Although the training process is very similar to the GSOM model, it uses a decreasing learning rate and a decreasing neighborhood range, instead of a fixed value. After growing the map, calculate the mean *mqe* of all units (MQE) in the current map. A map grows until its MQE is reduced to a predefined fraction (the growing-stopping criterion) of the *mqe* of the unit in the preceding layer of the hierarchy. In other words, the MQE of each map in the current layer should be smaller than a certain fraction value (τ_1) of the unit in the preceding layer. The lower the value of the quantization error, the better the map has been trained

$$MQE_m < \tau_1 * mqe_u,$$

where m denotes the units in the current map and u the mapped unit in the preceding layer.

After that, the next step is determine the depth of each topic in the current layer according to a predefined fraction (τ_2) of the *mqe* of the parent unit of the precedent layer and such the *mqe* of each unit in the current layer should be smaller than a certain fractional value of the unit in layer 0.

$$MQE_i < \tau_2 * mqe_0,$$

where i denotes the unit in the current layer. The stopping criterion of any unit in the hierarchy is always compared with layer 0. Details of this procedure can be found in (24).

With this, the GHSOM algorithm can be summarized as:

Start with one unit to expand initialized to the average of the input vectors, level 0. Then loop until no more units to expand and:

1. For each unit to expand create new 2×2 SOM.
2. Train SOM on data assigned to parent unit and:
   (a) Insert new row or column? If yes: insert new row/column and go to step 2.
   (b) Hierarchically expand units of map? If yes: add units to expand list.

where insert row/column if $MQE_m > \tau_1 mqe_0$, where mqe_0 is the MQE of a map unit m_0 representing the mean of all instances covered by the parent unit:

$$m_0 = \sum_i X_i/n$$

and expand unit if $MQE_i > \tau_2 mqe_0^*$, where mqe_0^* is the mean quantization error of all data set which generated the new map either the parent unit or all input data set as the beginning (in contrast to mqe_0, which is the mean quantization error of the map unit)(**?**). Generally τ_1, τ_2 are chosen such that $1 > \tau_1 >> \tau_2 > 0$.

2.3.5.1 U-matrix for the GHSOM

The U-Matrix is built from the SOM (and generalized to GHSOM) map prototypes as follows: Let n be a neuron on the map; $NN(n)$ be the set of immediate neighbors of n on the map; and $\mathbf{w}_n$ the prototype or weight vector associated with neuron n. Then $U_{height(n)} = \sum_{m \in NN(n)} d(\mathbf{w}_n - \mathbf{w}_m)$, where $d(\cdot)$ is the Euclidean distance used in the SOM algorithm to construct the map. The U-Matrix is a display of U_{height} on top of the grid positions of the neurons on the map, frequently in the form of a grey-level color-coded picture. Among its properties, we find that: the prototypes of neurons with large U_{height} values are very distant from other prototypes in the data space (high mapping distortion) and, correspondingly, the prototypes of neurons with small U_{height} values are closely surrounded by others in the data space (low mapping distortion); projections of the input data points are typically found in areas of low U_{height} values; high-valued U_{height} areas on a U-Matrix are an indication of cluster boundaries, while low-valued U_{height} areas on a U-Matrix are an indication of cluster center locations.

In the experiments reported on the following chapters, the U-matrix values are used as approximations of the mapping distortions generated by GHSOM at each stage of the hierarchy.

Chapter 3

Cartogram Representation of Mapping Distortion in SOM-Based Methods

3.1 Nonlinear Mapping Distortion

A possible generic definition of distortion in NLDR mapping could be the departure from natural, normal or original shape or size due to the application of a NLDR technique over a set of elements from a multidimensional space in the process of representing it in a low-dimensional visualization space.

Mapping distortion can be quantified and is popular criterion for assessing the quality of a data mapping process. NLDR techniques usually attempt to minimize the unavoidable distortion they introduce in the mapping of the high-dimensional data from the observed space onto lower-dimensional spaces. For a more faithful interpretation of models, a large number of distortion measures have been proposed and adapted to visualization techniques for different NLDR methods.

While reducing dimensionality, NLDR methods generate heterogeneous levels of local mapping distortion that potentially lead to a loss of information that, in one way or another, we aim to palliate in the visualization space.

Stretching or compressing a space affects the preservation of pairwise distances between points. If dealing with continuous techniques, we can apply, for instance Shepard diagrams, which plot pairwise distances of points as coordinates; for example, if x is the distance in the observed space and y is the correspondent distance in the projected space, the output of this diagram that represents x against y should be easy to interpret, as points in the diagonal represent those whose distance remained the same after projection.

Other less obvious measures proposed in (39) intend to estimate distortion focusing on the preservation of the neighborhood projection of each data point. Both measures could be an appropriate tool to compare different DR methods (or different projections performed using the same method), but they do not provide visualization or locally evaluate distortion.

An interesting contribution in (18) identifies different types of distortion, classified as geometrical and topological (including: manifold compression, stretching, gluing and tearing) and proposes the use of *Voronoi diagrams* and color scales to visualize manifold-based measures such as point-based, segment-based and triangle-based measures.

As stated above, the nonlinearity of DR methods such as SOM entails the existence of local distortion in the mapping of the data from the observed space onto the visualization space. This fact limits the direct interpretation of the visual data representation and there have been efforts to provide visual solutions to this limitation by defining and visualizing DR quality measures that, embedded in the method, can be associated to each map.

In SOM methods, which concern most of the thesis, a proposed distortion measure for a discrete data set can be written as:

$$E_d = \sum_{i=1}^{n} \sum_{j=1}^{m} h_{b_i j} \|x_i - m_j\|^2 \tag{3.1}$$

where n is the number of training samples, and m is the number of map units. The neighborhood function $h_{b_i j}$ is centered at unit b, which is the BMU of vector x_i, a sample vector evaluated for unit j and $h_{b_i} = argmin_j \|x_i - m_j\|^2$ (40).

This equation can be used for measuring the quality of a given SOM. One of the major advantages is that the error can be decomposed into two parts:

$$E_d = \sum_{i=1}^{n} H_{b_i} \|x_i - n_{b_i}\|^2 + \sum_{i=1}^{m} N_i \sum_{j=1}^{m} h_{ij} \|n_i - m_j\|^2 \tag{3.2}$$

where $H_{b_i} = \sum_{j=1}^{m} h_{b_i j}$

The first term, measures the quantization quality as the variance of the data vectors within each Voronoi set but, if the neighborhood function values for each map unit are normalized to unity such that $H_{b_i} = 1, \forall i$, then it corresponds to the classical vector quantization error. In this case, the equation, is slightly different from the measure typically used to calculate quantization quality of a SOM. Normally, to calculate the average quantization error the Euclidean distance is used, that is to say, $\sum_{i=1}^{n} \|x_i - b_i\|$ but, to calculate the distance between vectors many others can and are used too, such as the Minkowski, Hausdorff distances, etc. (41).

The second term measures the topological quality of the SOM. More specifically, it is a measure of closeness of prototype vectors on the map grid and it measures the goodness of the map topology. Both errors grow in inverse form, that is, if we want to decrease one it, we are bound to increase the other.

A widespread method to measure the distortion in SOM and proposed in (24) is the *Unified distance Matrix* or U-matrix. There are others as well such as P-matrix, the inverted P-matrix and the U*-matrix which allow the visualization in the latent space of the pairwise distances between corresponding points in the original data space. The values of these distances can be visually and intuitively represented with a color map together with the SOM topographic grid. Nevertheless, this representation os limited by issues of color contrast perception and overlapping with the own data projection representation.

The Cartogram representation described in the next sections is meant to at least partially overcome this limitation.

3.2 Cartograms

Cartograms are cartography maps in which specific geographical areas, delimited by artificial borders, are locally distorted by stretching or compression in proportion to locally-varying underlying quantities of interest, such as, for instance population density. In two dimensions, this distortion takes the form of a continuous transformation from an original plane to a transformed one, in which a vector x = (x_1, x_2) belonging to the former is mapped onto the latter according to $x \rightarrow T(x)$. The Jacobian of the transformation is proportional to a *distorting variable* **d**:

$$\frac{\partial T x_1}{\partial x_1} \frac{\partial T x_2}{\partial x_2} \propto d \tag{3.3}$$

A computationally tractable approach to this map distortion process requires the discretization of the plane to conform a regular grid of points. The distorting variable is assumed to take a uniform value over each of the plane fragments defined by such grid. To avoid the potential loss of connectivity between the plane fragments, a method for cartogram building based on the physics principle of linear diffusion processes was recently proposed in (5). In this method, the distorting variable **d** is let to *diffuse* over the map *over time* so that the final result, for $t \to \infty$, is a map of uniform distortion in which the original locations are displaced while preserving the integrity of the existing borders.

As part of this diffusion, the *current density* C follows the gradient of the distortion ∇d and can be written as product of the current flow velocity **v** and the distortion itself, so that **C** = -∇**d** = **v**(**x**, t)**d**(**x**, t). The diffusion equation takes the form

$$\nabla^2 d - \frac{\partial d}{\partial t} = 0 \tag{3.4}$$

which has to be solved for distortion **d**(**x**, t), assuming that the initial condition corresponds to each map fragment being assigned its value of the distorting variable. Thus, the distortion diffusion velocity can be calculated as:

$$\mathbf{v}(\mathbf{x}, t) = -\nabla \mathbf{d} \tag{3.5}$$

The calculation of the cartogram involves solving Eq. 3.4 for **d**(**x**,t) starting from the initial condition in which **d** is equal to the density of the region of interest and then calculating the corresponding velocity field from Eq. 3.5. The cumulative displacement **x**(t) of any point on the map at time **t** can be calculated by integrating the velocity field:

$$\Delta x = \int_0^t v(\mathbf{x}, t')dt' \tag{3.6}$$

In the limit $t \to \infty$ the set of such displacements for all points on the original map defines the cartogram (5). Details of this procedure as applied to other NLDR methods can be found in (42).

3.2.1 Cartogram representation for NLDR methods

In this section, we describe and assess cartogram representation as a tool for increasing the interpretability and usability of MVD visualization. In particular, we describe our proposal and preliminary assessment of a cartography-inspired method of cartogram representation of mapping distortion that should help to intuitively interpret the data visualizations generated by NLDR methods.

For techniques such as the SOM and the GHSOM, this distortion can be quantified on the map using a $U-matrix$, $P-matrix$, or $U*-matrix$. This topic is the main technical goal of the current master thesis. The visualization of the map of the NLDR manifold learning methods that we use is transformed into a cartogram taking into account these two points:

1. The square (or hexagonal) regular grid formed by the lattice of latent points u_k in the SOM-based maps can be used to define the map internal boundaries.

2. It is assumed that the level of distortion in the space beyond this square is uniform and equal to the mean distortion over the complete map, that is $1/K\sum_{k=1}^{K} J(u_k)$, where J is the Jacobian of the transformation of the considered method. Likewise, we assume that the level of distortion within each of the squares (hexagons) associated to u_k is itself uniform.

The cartograms reveal the internal relationships of the data and an advantage of this cartogram-based method is its portability, as it should be easy to implement for different representation architectures and with alternative NLDR visualization techniques for which distortion can be quantified. In the next chapters we reports the evaluation experiments carried out with this method.

3.2.2 Cartograms for SOM

As previously mentioned, a SOM consists of a layer (map) of units (or neurons) arranged in a low dimensional regular grid (often 2D). Each of these neurons k ($k = 1,\ldots,K$) is assigned a d-dimensional reference vector y_k. Summarily, the algorithm proceeds by finding, for each input data point x_j ($j = 1,...,N$) the best matching unit (BMU) y_{k_j} of index k_j computed as $k_j = argmin_k d(x_j, y_k)$. The distance d(.,.) is often chosen to be the Euclidean one $L_2(x_j, y_k) = \|x_j - y_{k_j}\|$. The locations of the reference vectors are iteratively updated to fit data points according an evolving learning rule. Details were explained in the previous chapter.

In the previous chapter, we also described some measures that quantify, in an approximate manner, the distortion introduced by the SOM mapping. They include the *U-matrix*, the *P-matrix* and the *U*-matrix*. These measures of distortion of the basic SOM algorithm can be visualized using cartograms (41).

The visualization of the *U-matrix* on the SOM map may inform us of the existence of data clusters and the sparsely populated spaces that separate them, as they undergo different levels of distortion: low in dense areas, while high in empty ones. This direct visualization is not always intuitive. Instead, the cartogram-based representation of the SOM map retains its simplicity while visually factoring out the nonlinear distortion as measured by the U-matrix and U*-matrix. In the following experiments, the SOM maps are transformed into a cartogram by using the rectangular grid, defined by the squares centered on the nodes, and assuming that the level of distortion in the space beyond this rectangle is uniform and equal to the mean distortion over the complete map.

The P-matrix displays the density measured in the data space at prototype or weight vector $\mathbf{w}(n)$ and its inverse can be used for cartogram visualization. This inverted P-matrix displays empty data space areas associated to neurons with large $P_{heights}$, whereas neurons with small $P_{heights}$ are allocated in dense regions of the data space. The idea is the *P-matrix* can be interpreted as the *U-matrix*, that is to say, the nearest vectors have less density and the farthest more density. The algorithm to invert the *P-matrix* is:

1. Calculate *P-matrix*.
2. Search the minimun value of *P-matrix*.
3. Search the maximun value of *P-matrix*.
4. Calculate the inverted *P-matrix* as fallow.
   - For each row of *P-matrix*,
     1 For each column of *P-matrix*:
     inverted *P-matrix*(row,column)=absolute-value(*P-matrix*(row,column) - maximun value of *P-matrix*).
     2 Inverted *P-matrix*(row,column) = minimum value of *P-matrix* + inverted *P-matrix*(row,column). (at this point, the minimum value of the inverted *P-matrix* is where the minimum of *P-matrix* was).

The *U*-matrix* combines the distance based *U-Matrix* and the density based *P-Matrix*. The visualization of the *U*-matrix* on the SOM map may inform us of the existence of data clusters more concentrated than the *U-matrix* and it exhibits more structure of the data set than the *U-Matrix*. Again, it can be used as a distortion measure for cartogram representation.

Chapter 4

Experiments with Synthetic Data

The cartogram-based representation method described in the previous chapter is meant to merge the powerful modeling capabilities of self-organizing NLDR methods and the explicit measurement of the local nonlinear distortion they generate. In doing so, it is aimed to provide an intuitive and compact visualization tool for the exploration of MVD data.

In the case of the SOM and GHSOM, used here to illustrate the cartogram method, the direct visualization of the distortion in the form of the *U-matrix*, mainly for GHSOM, and the *P-matrix*, the inverted P-matrix and the *U*-matrix* for SOM, can provide insight into the possible existence of densely populated data areas (or data clusters) and the sparsely populated areas that separate them.

This is the result of model prototypes being more densely located in densely data-populated areas (thus associated with low values of distortion) and more sparsely located in emptier spaces (thus associated with comparatively high values of distortion), reflecting, overall, heterogeneous levels of distortion as a result of the nonlinear mapping.

The cartogram-based representation of the SOM and GHSOM visualization space in which the observed data are mapped is expected to provide visual insight into the cluster structure of the data. The following and somehow detailed experiments, in which both artificial (in this chapter) and real (in the next one) data sets were analyzed, have the objective of assessing these expectations and, as a result, provide the data analyst with some general guidelines about the interpretation of the cartogram-based visual representation.

More in detail, the cartogram visual representation of U-matrix, P-matrix, Inverted P-matrix and U*-matrix using using SOM was first investigated using artificial data sets of simple statistical properties, so that the impact of their varying characteristics could be adequately interpreted. The first set of experiments thus involves data and model architectures of varying characteristics. Variations were of three main types:

- **Varying number of input data clusters**.
- **Varying model architectures**, concerning lattice size and relative dimensions, and geometry of the inter-neuron connections.
- **Varying input data dimensions**, from low-dimensional observed data to moderately high-dimensional input data.

Initialization of the method

Several parameters must be defined for model training, including the map size (number of map units), the neighborhood function, the radius of the neighborhood, and the learning coefficient. For all of them, there are loose rules that, at least, provide the analyst with rough adequate estimations.

- The number m of neurons conforming the map can be approximated as:

  $$m = 5 \times \sqrt{n}$$

  where n is the number of input vectors. An alternative criterion proposed in (43) is to calculate m as approximately the 10% of the number of input vectors.

- The map shape must be rectangular approximately according to the ratio between the two biggest eigenvalues of the covariance matrix of input vectors (6) defined as:

  $$\psi \approx \frac{1}{N}\sum_{t=1}^{N}(x(t)-\bar{x})(x(t)-\bar{x})^T \tag{4.1}$$

  where

  $$\psi_{i,j} = \begin{bmatrix} \psi_{1,1} & \psi_{1,2} & \cdots & \psi_{1,j} \\ \psi_{2,1} & \psi_{2,2} & \cdots & \psi_{2,j} \\ \cdots & \cdots & \cdots & \\ \psi_{i,1} & \psi_{i,2} & \cdots & \psi_{i,j} \end{bmatrix}$$

  and $x(t)$ is an input vector; $\bar{x}$ is the mean of the input vectors; i in the number of input vectors and j their dimension.

- A standard Gaussian neighborhood function is chosen.
- The neighborhood radius as well as learning ratio are monotonically decreasing functions over time (training iterations). The starting radius depends on the map size but the final radius is always 1.

For the following experiments further pre-processing entailed:

- That data sets underwent normalization and scaling.
- The weight vectors were linearly initialized, a procedure that is computationally more complex than random initialization but consistent for different runs of the algorithm. Eigendecomposition of the autocorrelation matrix of input vectors is performed, and the 2 eigenvectors with largest eigenvalues are used to span a 2-dimensional subspace and initialize the model vectors.
- Models were left to converge in the training process (with training stopping according to a minimum quantization error criterion). Some experiments report results in which the models were still undertrained, for illustrative purposes. We also explored a default provided by expression:

  $$\frac{map\_units\_number}{input\_vectors\_number} \tag{4.2}$$

  multiplying this ratio by 40 for the first phase training, by 160 for the second and then adding both (44). If the epochs number is low the model can underfit the data, whereas it can overfit if it is too big. After preliminary experimentation, the value taken by the learning parameter was 0.5.

Computational complexity

According the algorithm described in Sec. 2.2.1, there are $m \times n \times d$ operations (additions, subtractions, multiplications, divisions or exponents) in each epoch, where m are the map units, n the input vectors and d the dimension of all vectors. Then if $n \gg m$ the computational complexity of one epoch of sequential-SOM is about $\theta(nmd)$. We chose the sequential algorithm because it spends less memory than the batch algorithm despite the fact that the computational complexity of the latter is about half of the former, but the speed of convergence to the final solution is often less effective for the batch version.

4.1 Varying the Number of Data Clusters

These experiments involve the analysis of artificial data. A total of 1,500, 1,800 and 1,600 3-D points were, in turn, randomly drawn from 3, 6 and 8 identical spherical Gaussian distributions, respectively including 500, 300, and 200 points each, all with unit variance. We choose 3-D data for this batch of experiments in order to explicitly allow the direct visualization of the reference (or weight) vectors, also known as prototypes in the observed data space. In that way, we will better understand the effects of the nonlinear dimensionality reduction operated by SOM. At the end we can see the quality measure for each model and the order of complexity.

The SOM was implemented in *MATLAB*®, using the publicly available SOM-Toolbox[1] and Computer software for making cartograms (Cart)[2] using the technique described in the paper (45). As previously stated , data were pre-processed with both normalization and scaling. Given the nature of the data, square grids of 20 × 20 size were used.

In every case, the reported visualization maps include the standard U-matrix map as well as the P-matrix, the inverted P-matrix and the U*-matrix, together with the direct visualization of the original data visualization overlapped by the lattice of prototypes.

4.1.1 Experiment with three clusters

A total of 1,500 3-D points were randomly drawn from 3 identical spherical Gaussians distributions (500 points each), with centers sitting at the vertices of a triangle. They were fitted with a sequential SOM algorithm. A preliminary test involved underfitting the data by setting only a small number of iterations. The result is shown in Fig. 4.1. The rationale for this experiment was the exploration of the effect of model undertraining on the mapping distortion and, therefore, on its cartogram representation.

These results must be compared to those of the same model, but this time trained until convergence. They are displayed in Fig. 4.2.

Discussion

The fact that input data are 3D allows us to visualize directly not only the data themselves, but also the model weight vectors or prototypes. This helps us to investigate the fitting behavior of the model. The top-right corner plots of Fig. 4.1 and 4.2 display the data points as crosses

[1] www.cis.hut.fi/somtoolbox

[2] www-personal.umich.edu/ mejn/cart/

and the prototypes as black dots; the latter are linked by their square-shaped connections in the visualization space.

It is clear from the comparison of both plots that, in the case of the undertrained SOM, the regular square grid is far less distorted than for the fully trained SOM. It is also apparent that many more SOM prototypes are occupying the empty inter-cluster space in the undertrained model than in the fully trained one. This could be expected, given the the training process is meant lo "lead" prototypes towards dense data areas (and many prototypes are "squashed" in data-dense regions), which, in the case of this experiment, are obviously located.

How is all this reflected in the distortion measures? Again, the comparison between the color-coded U-matrix maps in Fig. 4.1 and 4.2 (top row-left plots, labeled as "distance matrix") is quite telling. In the undertrained model, only the areas immediately surrounding the clusters are clearly distorted, but this distortion has not yet reached the central data-empty areas. Instead, distortion has reached most of the empty areas in the fully trained model.

The cartogram representations of these U-matrices (2^{nd} row-left plots in 4.1 and 4.2) add further clarity to the visualization of the distortion. While the undertrained model does seem to be similarly distorted in the data-occupied areas and in the empty central space, the fully trained model shows a clearly different distortion in data-dense and empty areas.

Note also that the implicit encoding of the distortion in the form of grid stretching and compression yields a more transparent visual representation that allows us to clearly overlay the data projections themselves (as solid areas of different gray hues). The SOM prototypes (and thus SOM map areas) to which data points have been assigned are clearly isolated reflecting the three-cluster structure of the data.

The message conveyed by the rest of distortion measures is consistent with all the previous discussion. The P-matrix (2^{nd} row-right plot in 4.1) is not quite capturing the data densities in the undertrained model. This effect obviously has a negative impact on the corresponding inverted P-matrix and U*-matrix cartogram representations (3^{rd} plots in 4.1). Instead, the P-matrix (2^{nd} row-right plot in 4.2) neatly captures the higher data densities right in the cluster centers and the cartogram provides an expressive visualization of this result. This has a positive impact on the corresponding inverted P-matrix and U*-matrix cartogram representations (3^{rd} plots in 4.2), which are very similar to the U-matrix cartogram.

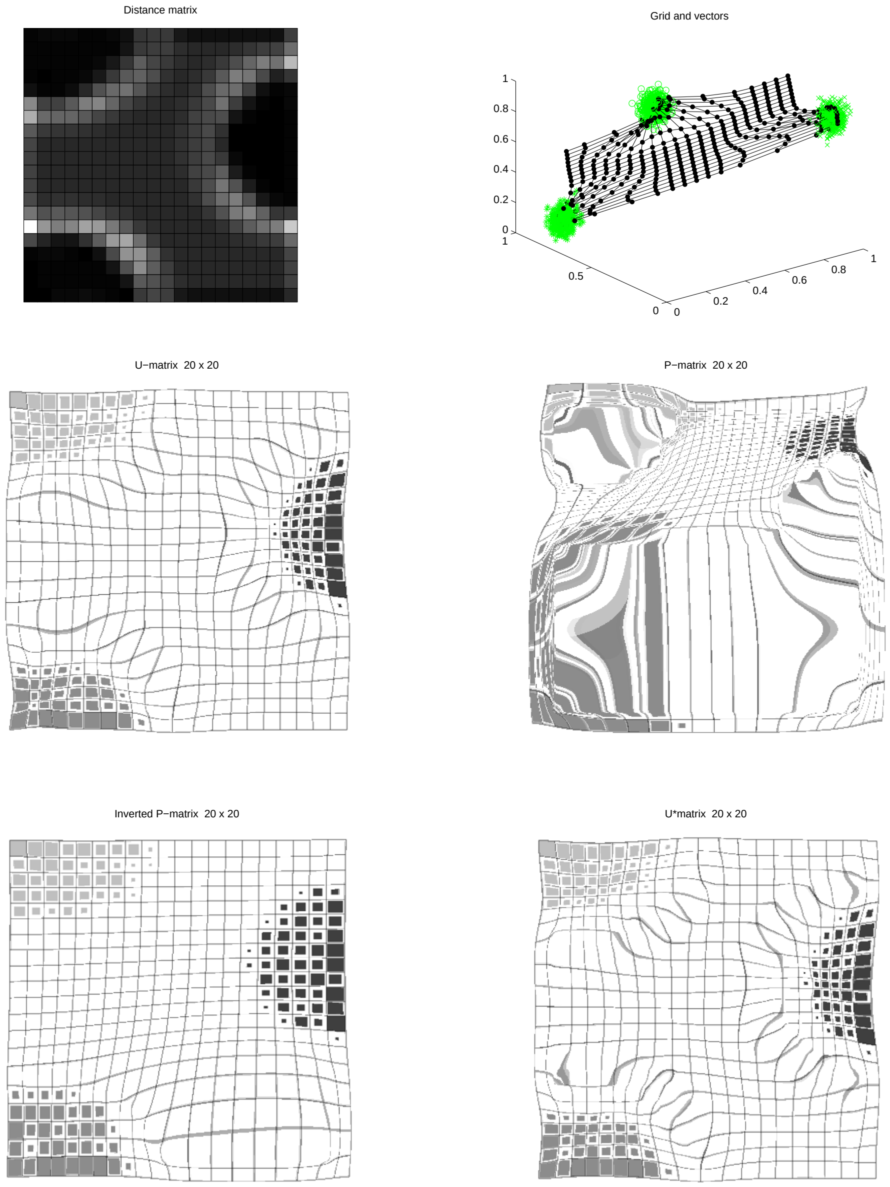

Figure 4.1: Visualization maps for the undertrained SOM model. *Top row*: left) Color-coded distance-matrix, indicating the level of mapping distortion for each of the SOM map units in the square 20×20 lattice (in a gray scale, dark gray indicates low distortion and, therefore, concentration of input data points, while light gray indicates high distortion and, correspondingly, low concentration of data points or borders between clusters) in which three isolated and equally-spaced clusters (of 500 3-D data points); right) SOM grid of prototypes (black dots) overlaid to the data points (cross symbols). *Middle row*: left) cartogram of the U-matrix with the three data clusters codified in different shadows of grey; right) cartogram of the P-matrix. *Bottom row*: left) cartogram of the inverted P-matrix; right) cartogram of the U*-matrix.

The same model, fully trained

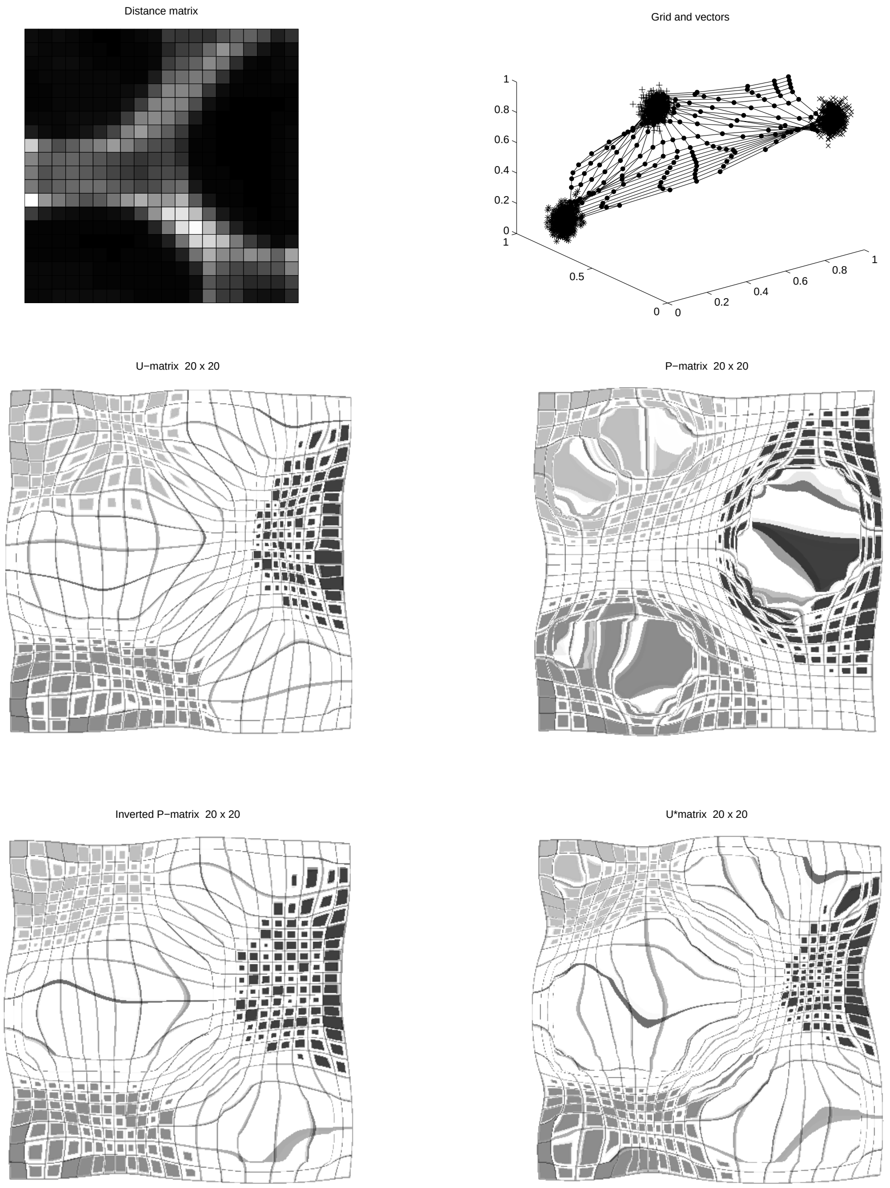

Figure 4.2: Visualization maps for the fully trained SOM model, using the same data set consisting of three clusters. Display as in Fig. 4.1.

4.1.2 Experiment with six clusters

For this second experiment, a total of 1,800 3-D points were randomly drawn from 6 identical spherical Gaussian distributions (300 points each) and again trained with the sequential-SOM algorithm. The rationale for increasing the number of clusters is finding out whether this increase has an impact on the calculation of the mapping distortion and whether this distortion is still clearly captured by its cartogram representation.

No undertraining experiment was performed this time. Thus, the reported results correspond to a SOM fully trained to convergence. They are displayed in Fig. 4.3.

Discussion

The grid of prototypes (top row-right plot) is quite distorted in the empty inter-cluster spaces, while quite densely concentrated in the dense data clusters themselves.

This varying local levels of distortion are neatly reflected by the U-matrix distortion measure (top row-left plot), with sharply defined light/dark areas, and also by all the rest of the distortion measures as described through cartograms (U-matrix, P-matrix, inverted P-matrix and U*-matrix in the rest of plots in Fig. 4.3).

The separation between clusters is less evident than in Fig. 4.2, because the dimensions of the grid remain the same while the number of clusters has increased. The borders between clusters are far more narrow now and, interestingly, they are far better reflected through the U-matrix cartogram, which isolates clusters more clearly by magnifying the between-cluster spaces. In comparison, the rest of distortion measures only partially manage to reflect distortion correctly, mostly due to the failure of the P-matrix to estimate relative data densities.

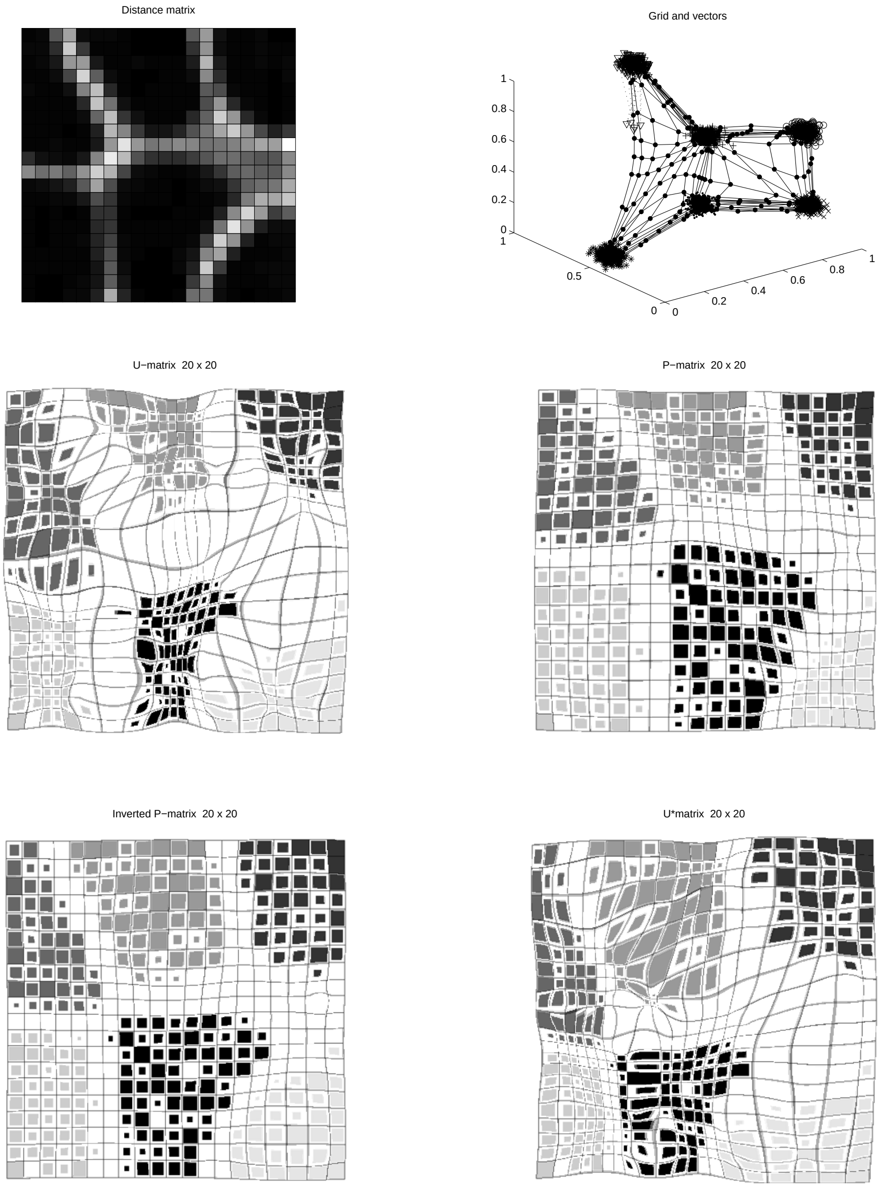

Figure 4.3: Visualization maps for a fully trained SOM model, using a six cluster data set. Display layout as in previous figures.

4.1.3 Experiment with eight clusters

For the last experiment of this series, a total of 1,600 3-D points were randomly drawn from 8 identical spherical Gaussian distributions (200 points each) and trained with the same sequential-SOM algorithm as previous experiments.

Again, no undertraining experiment was performed. Thus, the reported results correspond to a SOM fully trained to convergence. They are displayed in Fig. 4.4.

Discussion

The grid of prototypes (top row-right plot) is once again quite distorted in the empty inter-cluster spaces, while quite densely concentrated in the dense data clusters themselves. The abundance of clusters, though, makes the stretching of the grid of prototypes more complex.

The varying local levels of distortion are still reflected by the U-matrix distortion measure (top row-left plot), although the light/dark areas are less sharply defined this time.

The separation between clusters is again less evident than in Fig. 4.2 and the borders between clusters are more irregular, indicating that the SOM model has managed to separate some clusters better than others.

This affects all the distortion measures as described through cartograms (U-matrix, P-matrix, inverted P-matrix and U*-matrix in the rest of plots in Fig. 4.4). In fact, it could be concluded that the cartogram representation in standard SOM might perhaps be used with great caution when many distinct clusters are present in the data set. Alternatively, when many clusters might be expected, cartograms might better be used with larger maps, using strategies such as those of the ESOM.

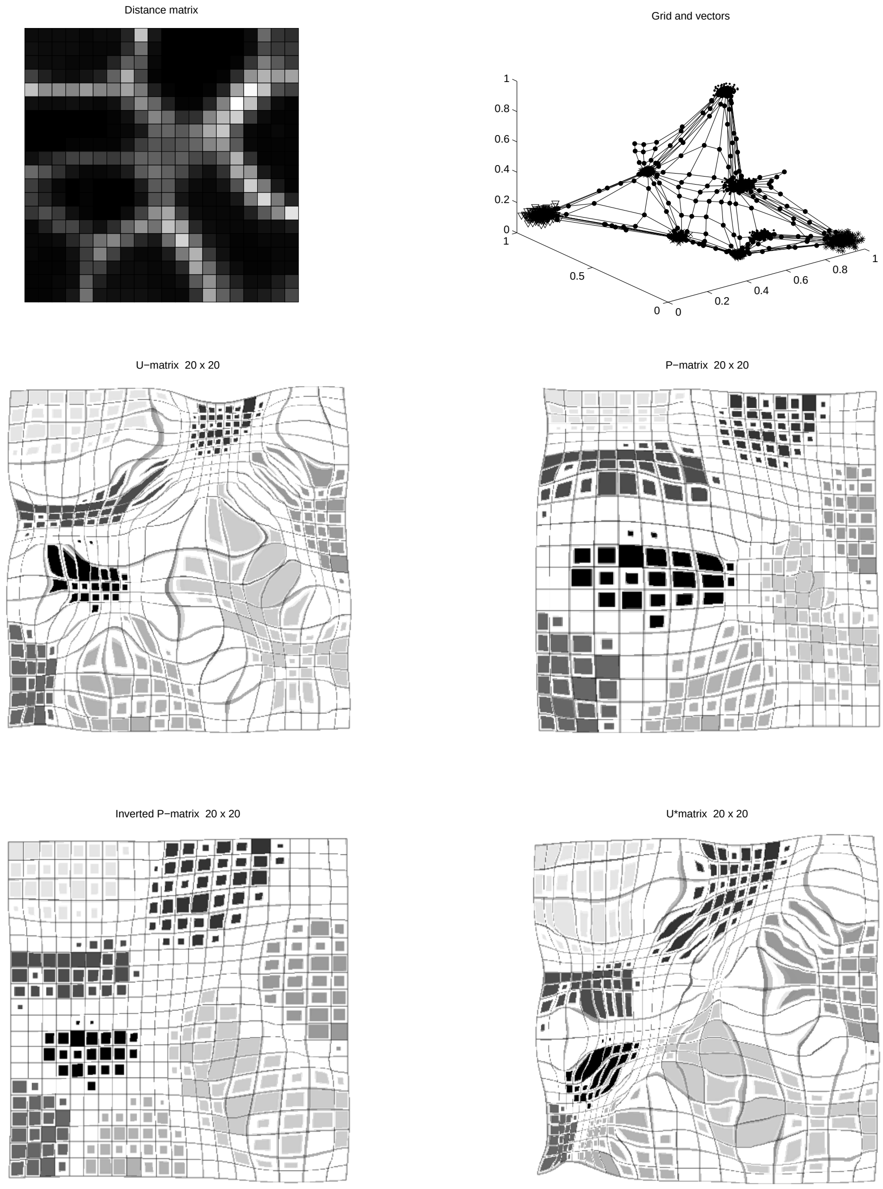

Figure 4.4: Visualization maps for a fully trained SOM model, using an eight cluster data set. Display layout as in previous figures.

The results in the Table 4.1 show that, as we increase the number of clusters, the TE increases. The QE, though does not follow a clear trend. The complexity and $\theta(nmd)$ is of one epoch.

♯ points	Dimension	Clusters	Points/Cluster	QE	TE	Complexity	$\theta(nmd)$
1500	3	3	500	0.0172	0.1153	1,800,000	$\theta(1500 \times 400 \times 3)$
1800	3	6	300	0.0178	0.1300	2,160,000	$\theta(1800 \times 400 \times 3)$
1600	3	8	200	0.0119	0.1588	1,920,000	$\theta(1600 \times 400 \times 3)$

Table 4.1: Results for rectangular lattices.

4.2 Varying the Dimension of Input Data

The rationale for the next set of experiments is the investigation of the impact of the observed data dimensionality on the mapping of the data in terms of the magnitude of the local distortion and its impact on the cartogram visual representation of this distortion.

Three experiments are considered for 3-D, 10-D and 15-D data sets.

4.2.1 3-D input data

For all the experiments in this section, we set the architecture of the SOM to fixed 20 × 20 maps. Therefore, the experiment for 3-D data is exactly the same reported in section 4.1.1 and the results of the next subsections are to be compared with those reported in Fig. 4.2

4.2.2 10-D input data

The only difference with the previous experiment is the dimensionality of the observed data. Thus, a total of 1,500 10-D points were randomly drawn from 3 identical spherical Gaussians distributions (500 points each) and with centers sitting at the vertices of a triangle and trained with the sequential-SOM algorithm.

Results are displayed in Fig. 4.5. The only difference with previous displays is that the data and the prototypes defined by SOM to fit them cannot be displayed directly any longer.

Discussion

The gray-coded U-matrix (Fig. 4.5, top row) conveys very similar information to that of the 3-D data reported in Fig. 4.2. The main difference is that the range of values is more extreme, indicating that the empty areas are more distorted in the higher dimensional space.

This is very explicitly captured by all the cartograms in the rest of plots of Fig. 4.5. The cartogram of the U-matrix sharply delimits the little-distorted hyper-spherical shapes of the original clusters while homogeneously showing the high distortion of the inter-cluster spaces.

The U-matrix reveals itself as the most visually informative distortion measure to use with cartograms, and somehow making more intuitive, by extension, the inverted P and U* matrices.

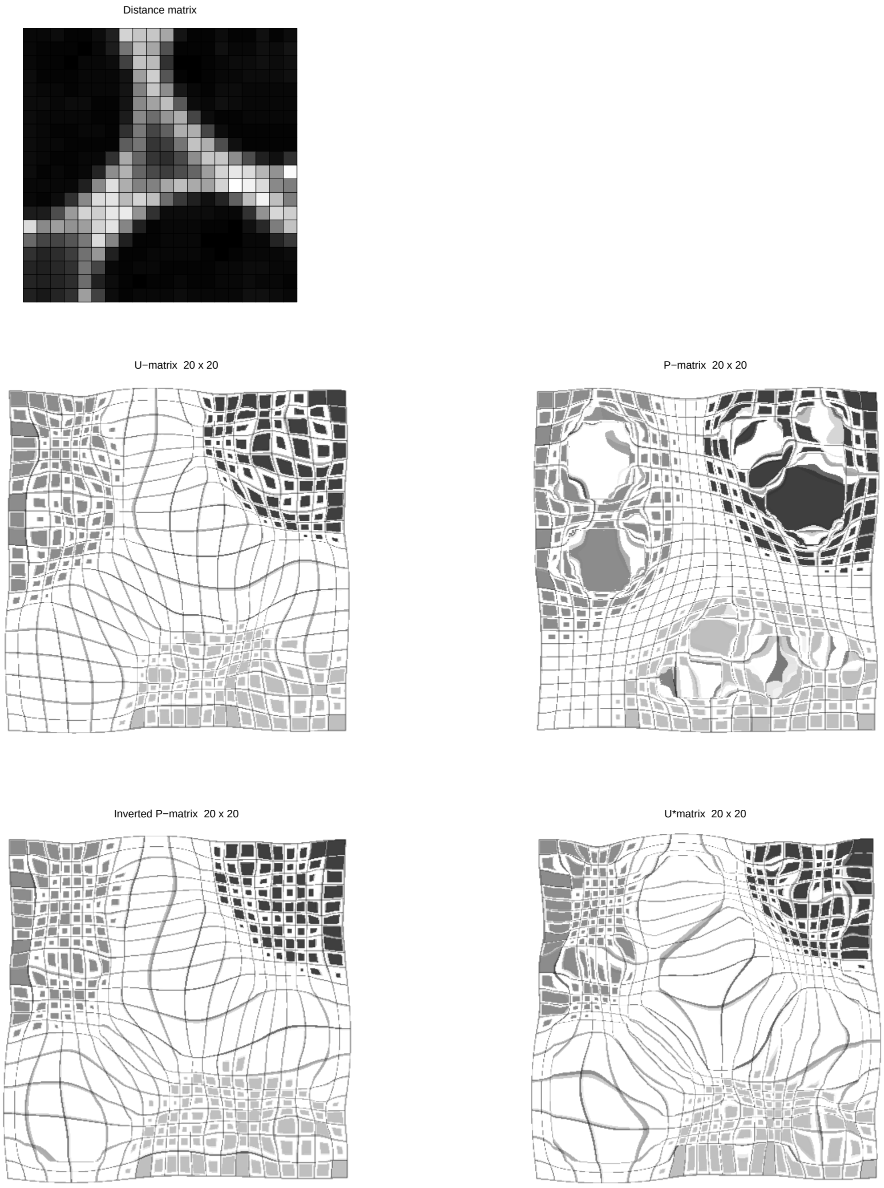

Figure 4.5: Visualization maps for the fully trained SOM model of 10-dimensional data organized in three clusters. *Top row*: Colour-coded distance-matrix, indicating the level of mapping distortion for each of the SOM map units in the square 20×20 lattice. *Middle row*: left) cartogram of the U-matrix with the three data clusters codified in different shadows of grey; right) cartogram of the P-matrix. *Bottom row*: left) cartogram of the inverted P-matrix; right) cartogram of the U*-matrix.

♯ Points	Dimension	Clusters	Points/Cluster	QE	TE	Complexity	$\theta(nmd)$
1500	3	3	500	0.0172	0.1153	1,800,000	$\theta(1500 \times 400 \times 3)$
1500	10	3	500	0.1044	0.1553	6,000,000	$\theta(1500 \times 400 \times 10)$
1500	15	3	500	0.1720	0.1527	9,000,000	$\theta(1500 \times 400 \times 15)$

Table 4.2: Results for rectangular lattices.

4.2.3 15-D input data

A total of 1,500 15-D points were now randomly drawn from 3 identical spherical Gaussians distributions (500 points each) and with centers sitting at the vertices of a triangle and trained with the sequential-SOM algorithm.

Results of the last experiment of this section are displayed in Fig. 4.6.

Discussion

The gray-coded U-matrix (Fig. 4.6, top row) conveys results that are almost identical to those of the previous experiments.

Again, the different local levels of distortion are very explicitly captured by all the cartograms in the rest of plots of Fig. 4.6. Some cautious preliminary conclusion is that, at least for data sets with well-separated cluster structure, the increase of data dimensionality not only does not hamper the visual exploration of the cluster structure of the data through cartogram-based visualization of the mapping distortion, but in fact makes it clearer.

Further research should extend the reported one in two directions: First, data of truly large dimensionality should also be investigated and, second, more complex data with less obvious cluster structure should be put to the same tests.

The Table 4.2 summarizes some mapping quality data, showing that, as the number of variables of the input data (dimension) increases, the QE correspondingly increases, whereas the TE does not follow a clear increasing trend. The complexity and $\theta(nmd)$ is of one epoch.

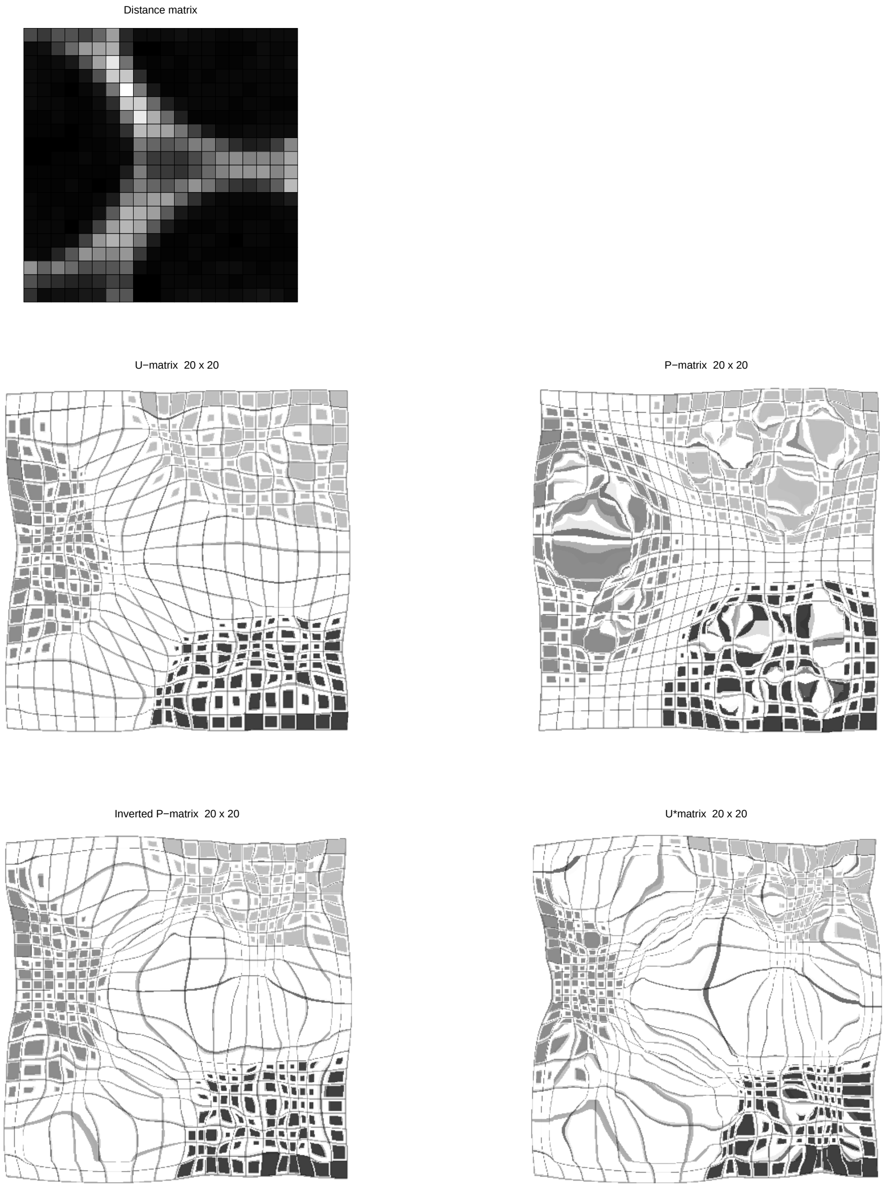

Figure 4.6: Visualization maps for the fully trained SOM model of 15-dimensional data organized in three clusters. Display as in the previous figure.

4.3 Varying the SOM Architecture: Grid Dimensions and Lattice Type

The rationale for the following set of experiments is finding out what is the impact, if any, of varying the characteristics of the fixed architecture of the standard SOM model.

The SOM architecture was varied in the experiments according to different characteristics. All throughout the experiments in this section, square lattices (that is, lattices in which only vertical and horizontal inter-unit connections -4 per unit- are allowed) will be compared with hexagonal lattices (in which diagonal inter-unit connections -6 per unit- are allowed); rectangular grids (different number of rows and columns) will be compared to square ones (same number of rows and columns); and, finally, square grids of different sizes (10×10, 20×20, 30×30 and 50×50,) will be compared.

Once again, in these experiments 3 clusters of 1,500 3-D input vectors were used.

4.3.1 Experiments with varying grid dimensions

4.3.1.1 Rectangular 10×20 grid with rectangular lattice

A first experiment was performed using a 10×20 layout for the SOM map, which is the default one obtained using the formula in Eq. 4. The lattice has rectangular connections, that is, only horizontal and vertical inter-unit connections.

The map, while small, perfectly reflects the three clusters of the data. Results can be seen in Fig. 4.7

The same experiment is repeated with an hexagonal lattice, and its results are reported in Fig. 4.8

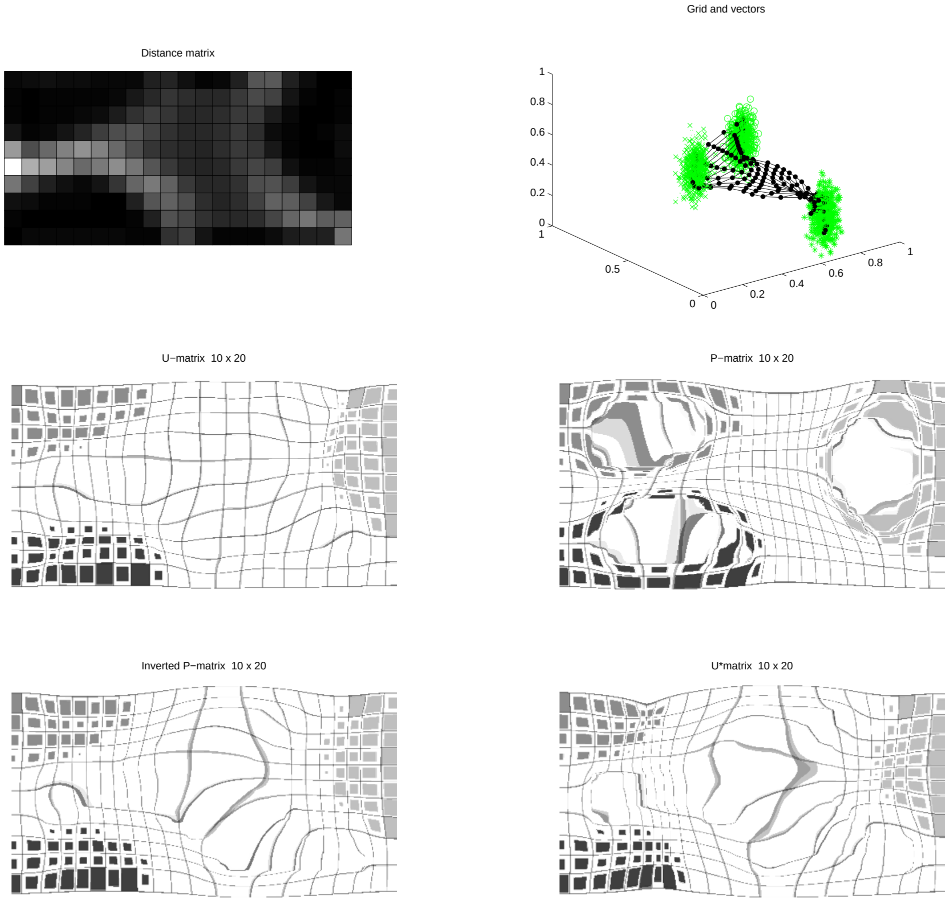

Figure 4.7: Visualization maps for the fully trained SOM model of 3-dimensional data organized in three clusters, using a 10×20 grid with rectangular layout. Display as in previous figures.

4.3.1.2 Rectangular 10×20 grid with hexagonal lattice

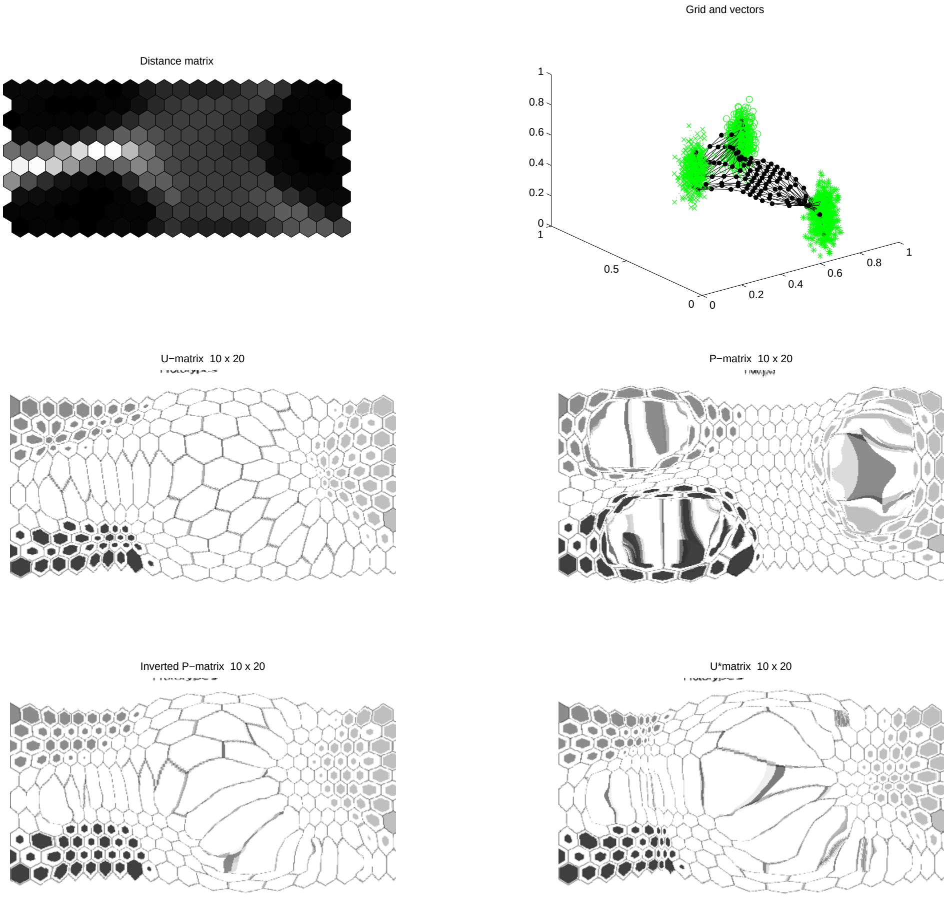

Figure 4.8: Visualization maps for the fully trained SOM model of 3-dimensional data organized in three clusters, using a 10×20 grid with hexagonal layout. Display as in previous figures.

4.3.1.3 Rectangular 20×10 grid with rectangular lattice

The next two experiments replicate the previous two, but using a symmetrical 20×10 layout for the SOM map, which is equally the default obtained using the formula in Eq. 4. The lattice is rectangular (results can be seen in Fig. 4.9) for the first experiment.

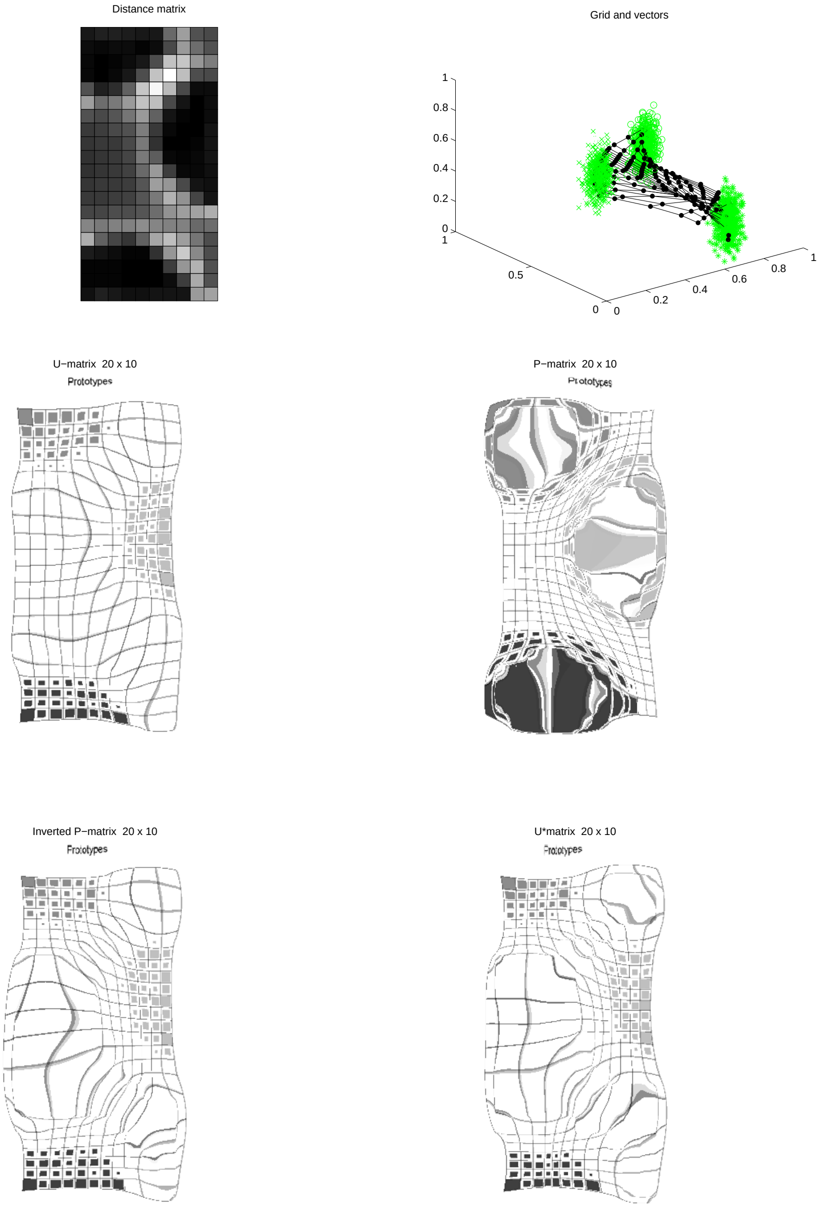

Figure 4.9: Visualization maps for the fully trained SOM model of 3-dimensional data organized in three clusters, using a 20×10 grid with rectangular layout. Display as in previous figures.

4.3.1.4 Rectangular 20×10 grid with hexagonal lattice

Results for the same experiment with hexagonal lattice are shown in Fig. 4.10.

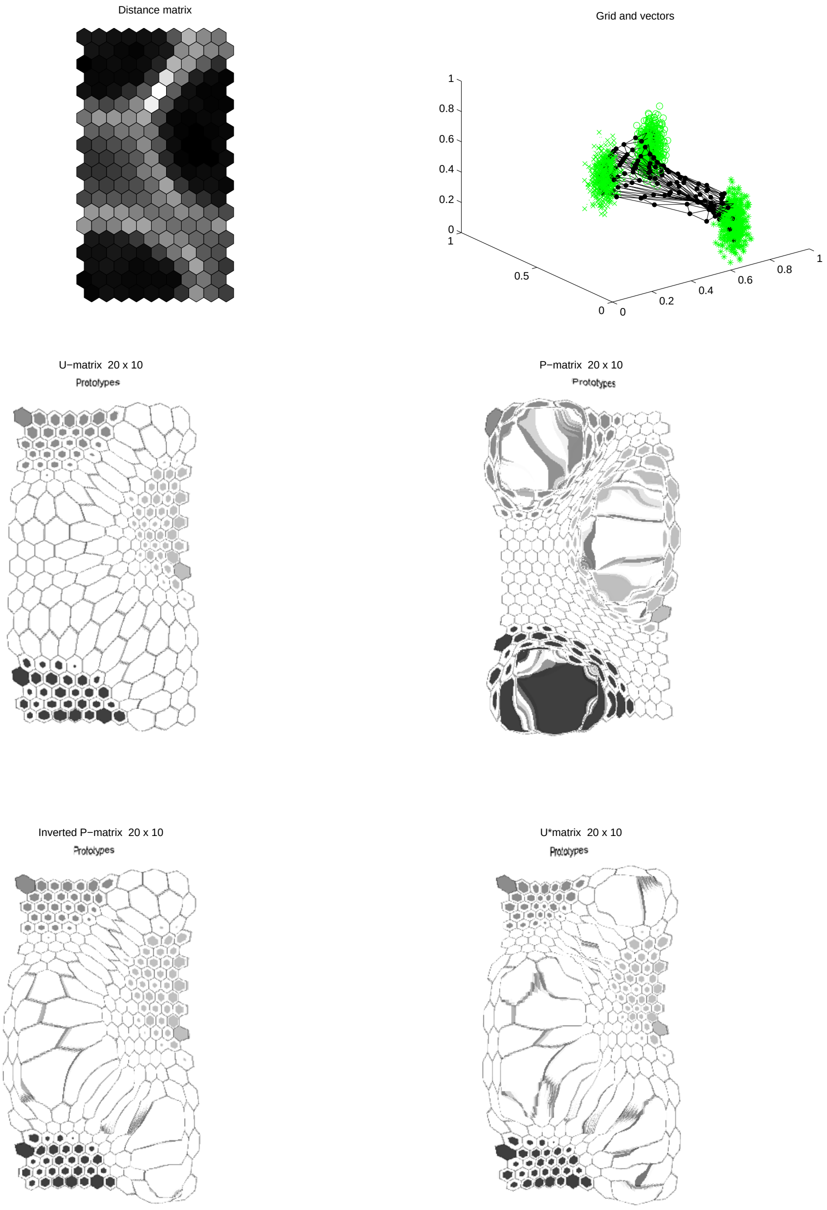

Figure 4.10: Visualization maps for the fully trained SOM model of 3-dimensional data organized in three clusters, using a 20×10 grid with hexagonal layout. Display as in previous figures.

Discussion

The results of the four previous experiments are clear.

- First of all, the four experiments isolate the three clusters perfectly well.
- The exchange of grid dimensions (10×20 by 20×10) has no effect whatsoever in the modeling of the data. In fact, results are almost perfectly symmetrical.
- the use of either a rectangular or an hexagonal lattice has no impact whatsoever in the results, including the distortion measures.
- All distortion measures neatly separate the highly distorted data-empty between-cluster areas from the barely distorted data-rich cluster areas. This distortion is intuitively captured by the cartogram representations of all the distortion measures over the SOM map.

4.3.2 Experiments with varying grid sizes

This subsection considers the effect of increasing grid sizes (that is, increasing number of units in the visualization map) on the cartogram representation.

We start from an overall similar number of units as the latest experiments, using a square 13×13 grid with rectangular lattice (169 units) and increase it to a size of 50×50 (2,500) grid. For this latter size, we also try a further experiment to illustrate the effect of under-training and over-training.

4.3.2.1 Square 13×13 grid with rectangular lattice

The results for the square 13×13 grid map with rectangular lattice are summarized in Fig. 4.11.

4.3.2.2 Square 13×13 grid with hexagonal lattice

Similarly, the results for the square 13×13 grid map with hexagonal lattice are summarized in Fig. 4.12.

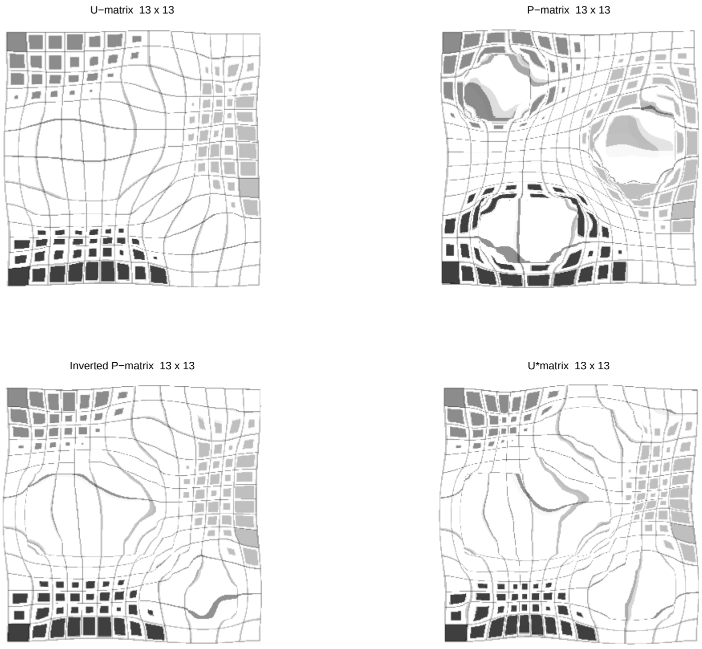

Figure 4.11: Visualization maps for the fully trained SOM model of 3-dimensional data organized in three clusters, using a 13×13 square grid with rectangular layout. Display as in previous figures, but restricted to cartogram representations.

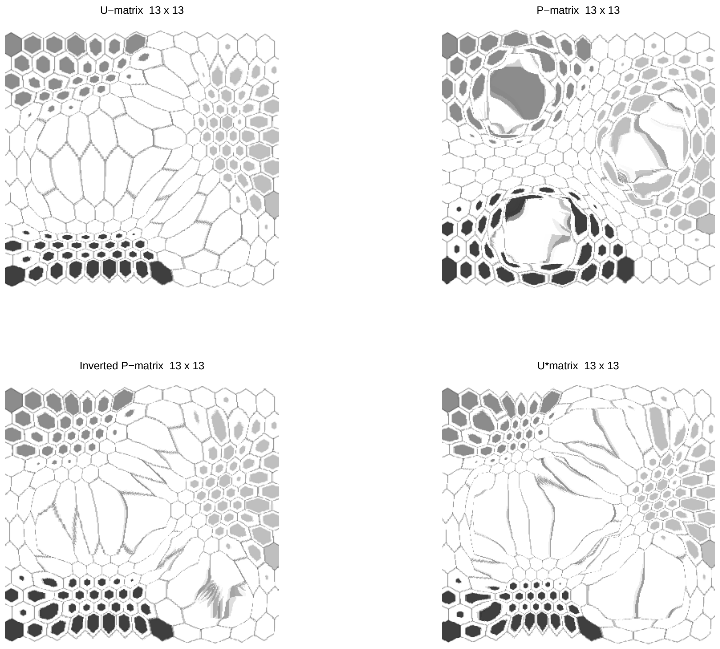

Figure 4.12: Visualization maps for the fully trained SOM model of 3-dimensional data organized in three clusters, using a 13×13 square grid with hexagonal layout. Display as in previous figure.

4.3.2.3 Square 30×30 grid with rectangular lattice

The results for the square 30×30 grid map with rectangular lattice are summarized in Fig. 4.13.

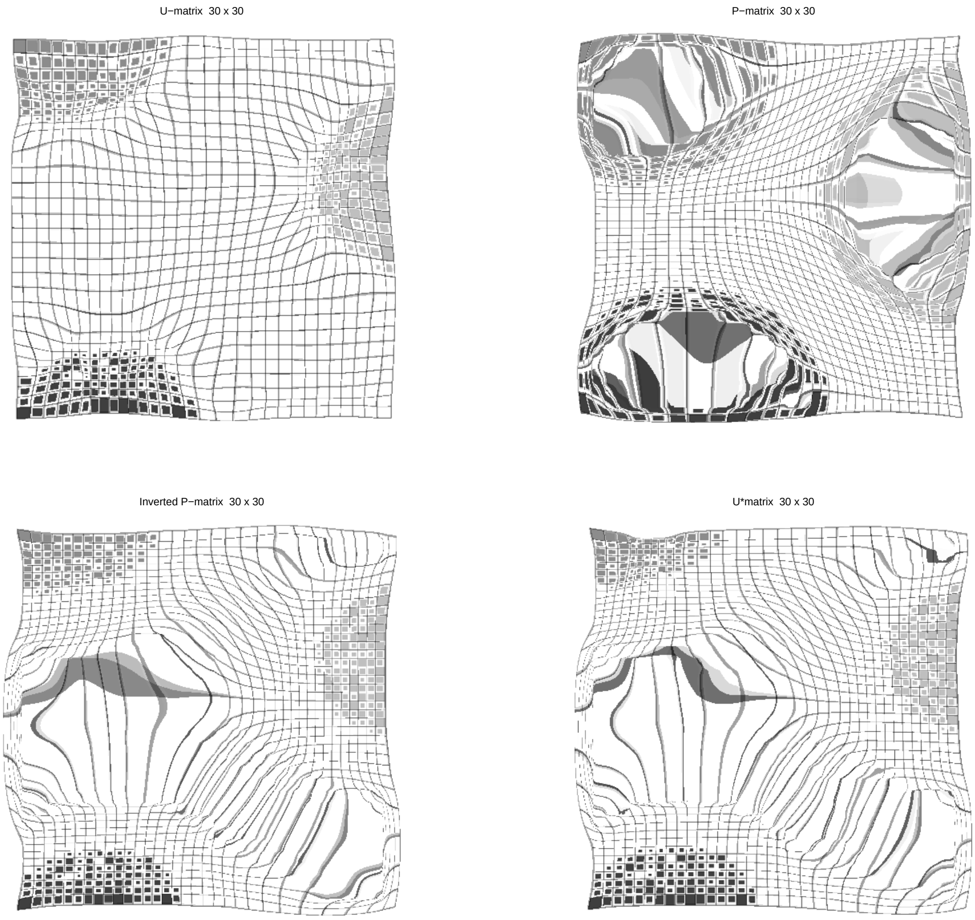

Figure 4.13: Visualization maps for the fully trained SOM model of 3-dimensional data organized in three clusters, using a 30×30 square grid with rectangular layout. Display as in previous figures.

4.3.2.4 Square 30×30 grid with hexagonal lattice

Similarly, the results for the square 30×30 grid map with hexagonal lattice are summarized in Fig. 4.14.

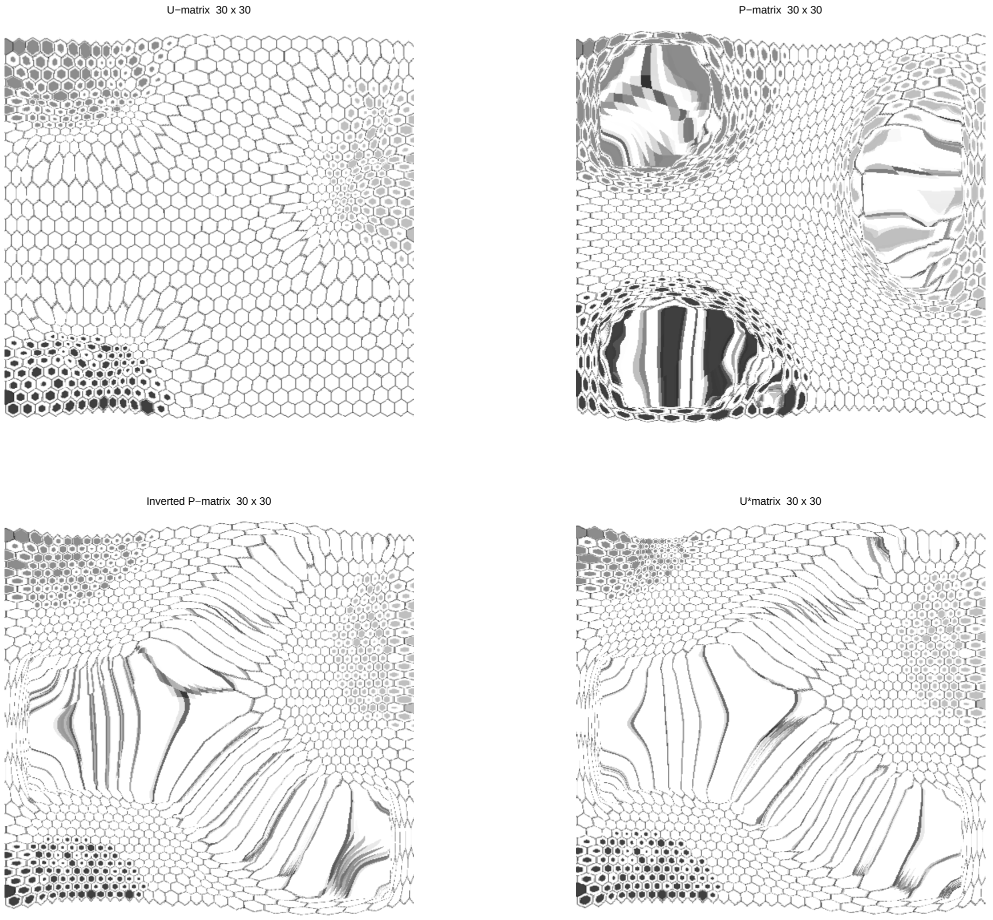

Figure 4.14: Visualization maps for the fully trained SOM model of 3-dimensional data organized in three clusters, using a 30×30 square grid with hexagonal layout. Display as in previous figures.

4.3.2.5 Square 50×50 grid with rectangular lattice: 10 iterations

Now, once reached the maximum map size under experimentation, namely 50×50, we set to illustrate the differences between maps trained for a bare 10 iterations, for 17 iterations, as recommended by default by the SOM toolbox, and then for 110 and 800 iterations.

The results for the square 50×50 grid map with rectangular lattice are summarized in Fig. 4.15.

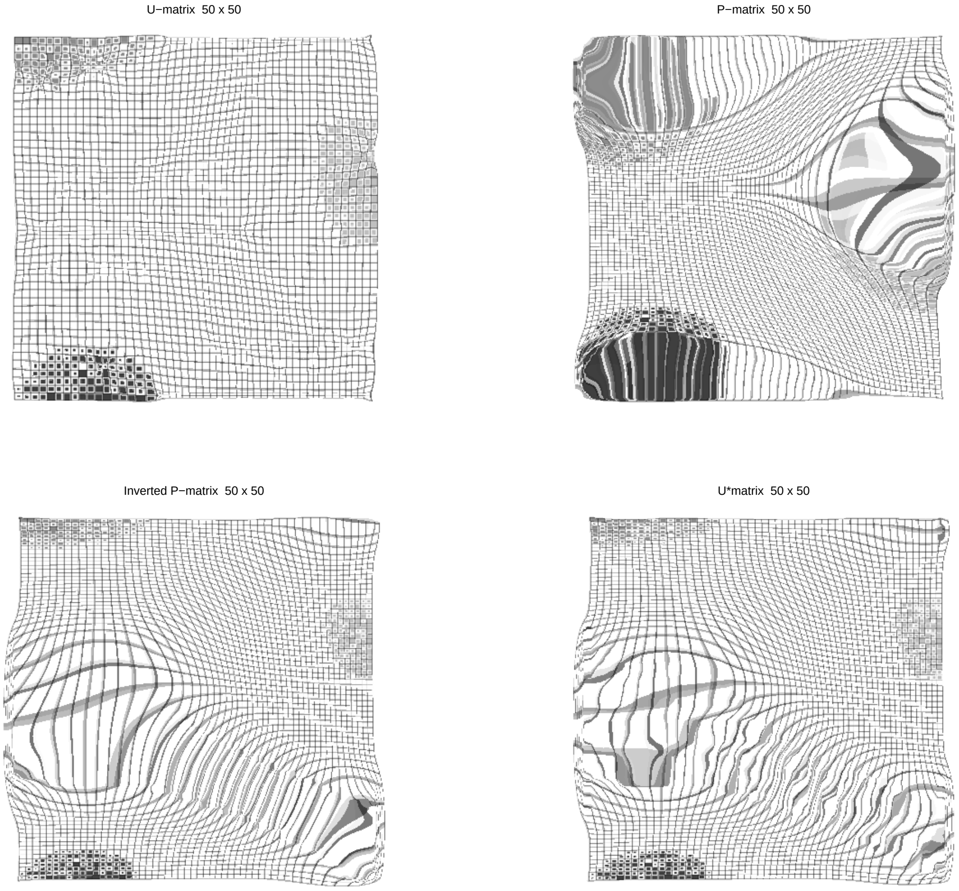

Figure 4.15: Visualization maps for a SOM model, trained for only 10 iterations, of 3-dimensional data organized in three clusters, using a 50×50 square grid with rectangular layout. Display as in previous figures.

4.3.2.6 Square 50×50 grid with hexagonal lattice: 10 iterations

Similarly, the results for the square 50×50 grid map with hexagonal lattice in a SOM trained for 10 iterations are summarized in Fig. 4.16.

4.3.2.7 Square 50×50 grid with rectangular lattice: 17 iterations

The results for the square 50×50 grid map with rectangular lattice in a SOM trained for 17 iterations are summarized in Fig. 4.17.

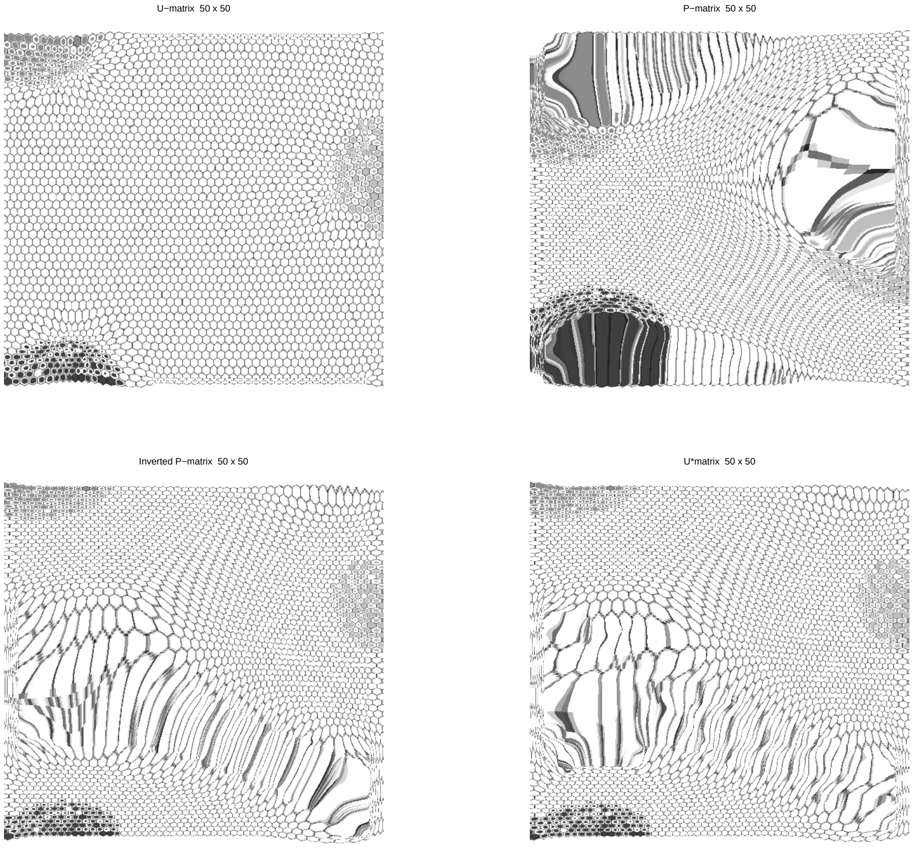

Figure 4.16: Visualization maps for a SOM model, trained for only 10 iterations, of 3-dimensional data organized in three clusters, using a 50×50 square grid with hexagonal layout. Display as in previous figures.

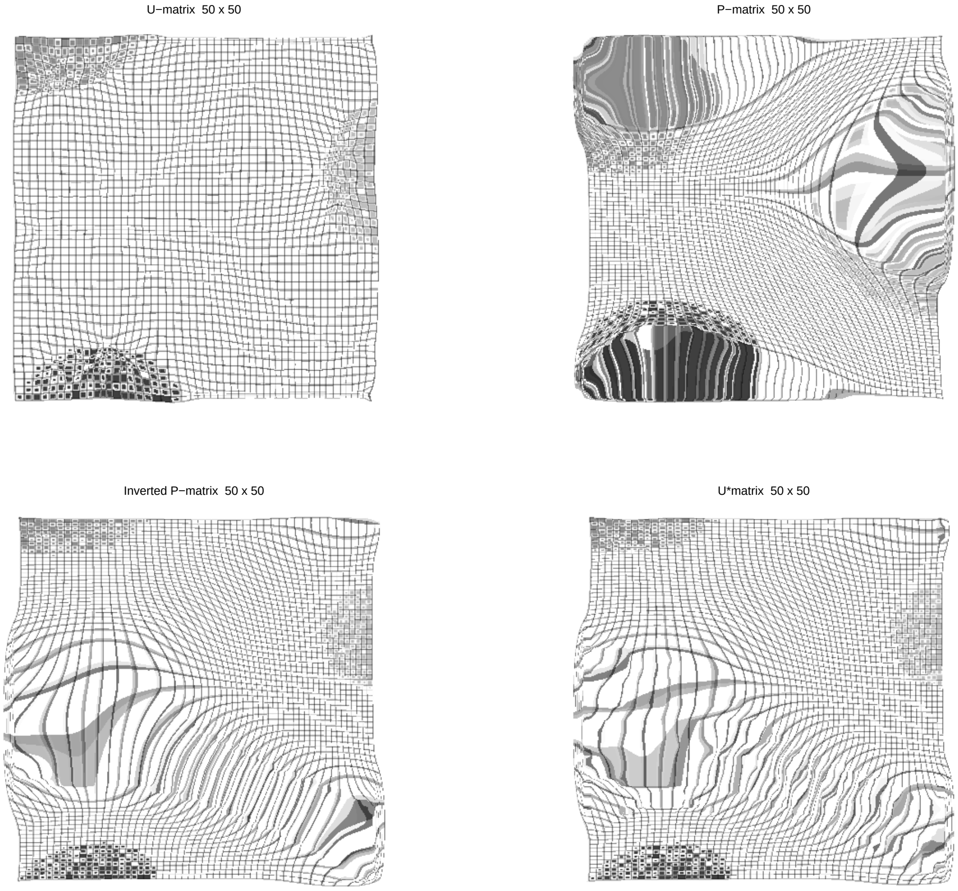

Figure 4.17: Visualization maps for a SOM model, trained for 17 iterations, of 3-dimensional data organized in three clusters, using a 50×50 square grid with rectangular layout. Display as in previous figures.

4.3.2.8 Square 50×50 grid with hexagonal lattice: 17 iterations

Similarly, the results for the square 50×50 grid map with hexagonal lattice in a SOM trained for 17 iterations are summarized in Fig. 4.18.

4.3.2.9 Square 50×50 grid with rectangular lattice: 110 iterations

The results for the square 50×50 grid map with rectangular lattice in a SOM trained for 110 iterations are summarized in Fig. 4.19.

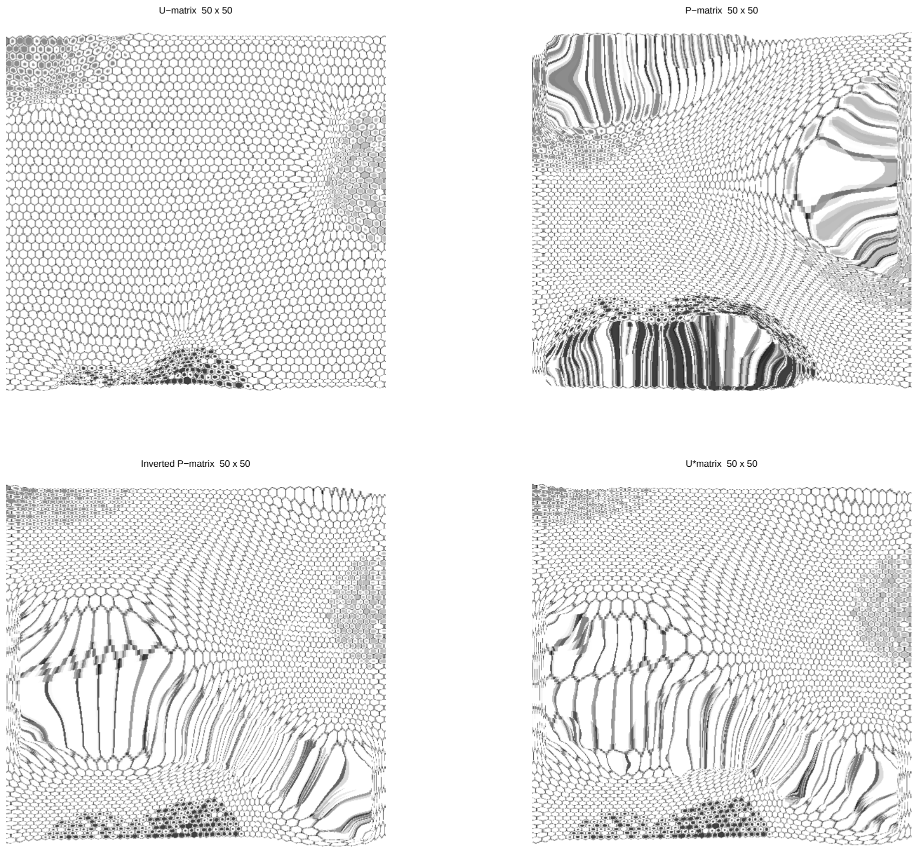

Figure 4.18: Visualization maps for a SOM model, trained for 17 iterations, of 3-dimensional data organized in three clusters, using a 50×50 square grid with hexagonal layout. Display as in previous figures.

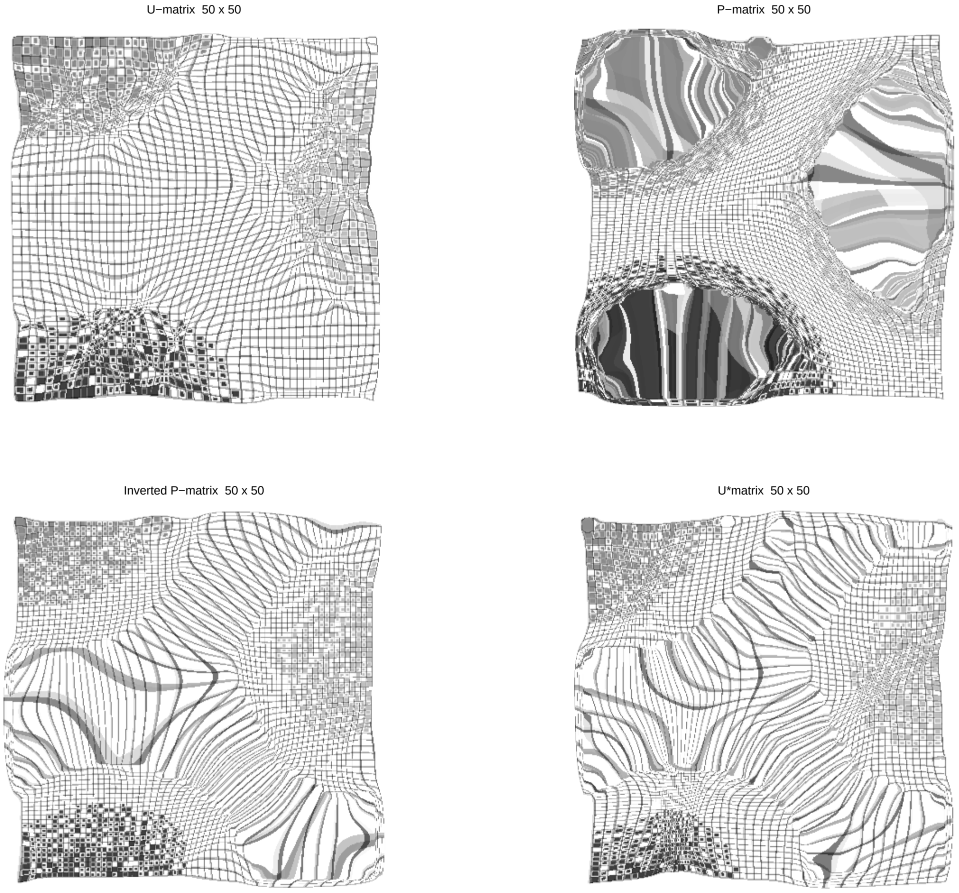

Figure 4.19: Visualization maps for a SOM model, trained for 110 iterations, of 3-dimensional data organized in three clusters, using a 50×50 square grid with rectangular layout. Display as in previous figures.

4.3.2.10 Square 50×50 grid with hexagonal lattice: 110 iterations

Similarly, the results for the square 50×50 grid map with hexagonal lattice in a SOM trained for 110 iterations are summarized in Fig. 4.18.

4.3.2.11 Square 50×50 grid with rectangular lattice: 800 iterations

Finally, the results for the square 50×50 grid map with rectangular lattice in a SOM trained for 800 iterations are summarized in Fig. 4.21.

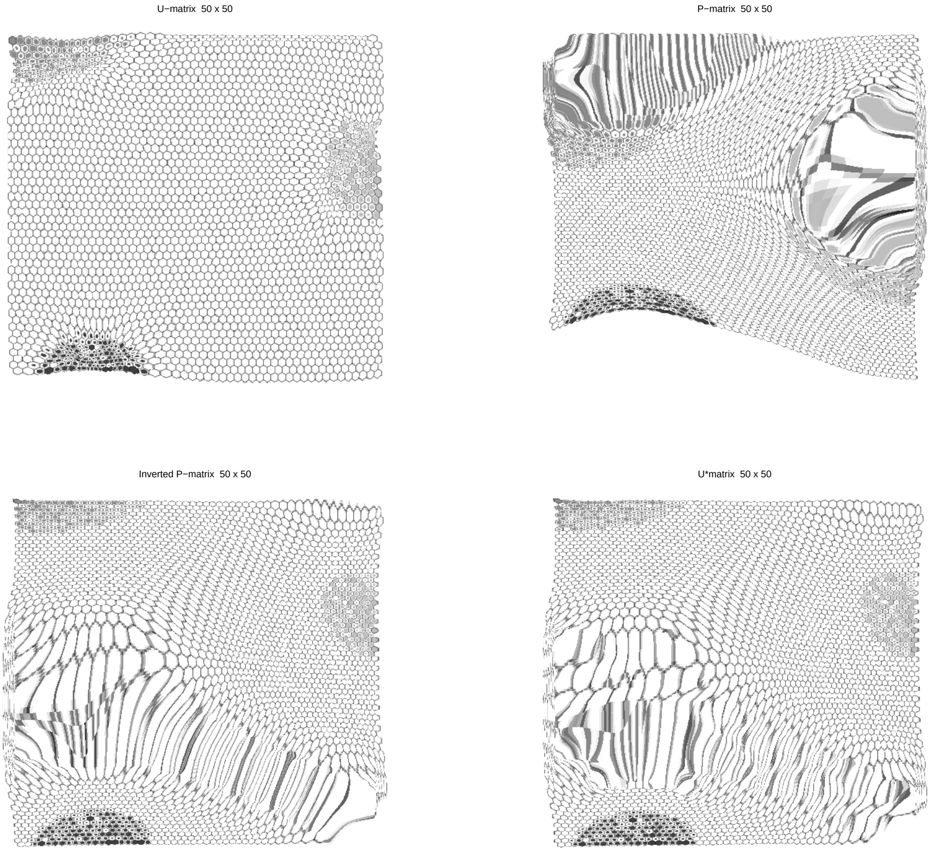

Figure 4.20: Visualization maps for a SOM model, trained for 110 iterations, of 3-dimensional data organized in three clusters, using a 50×50 square grid with hexagonal layout. Display as in previous figures.

4.3.2.12 Square 50×50 grid with hexagonal lattice: 800 iterations

The corresponding final results for the square 50×50 grid map with hexagonal lattice in a SOM trained for 800 iterations are summarized in Fig. 4.22.

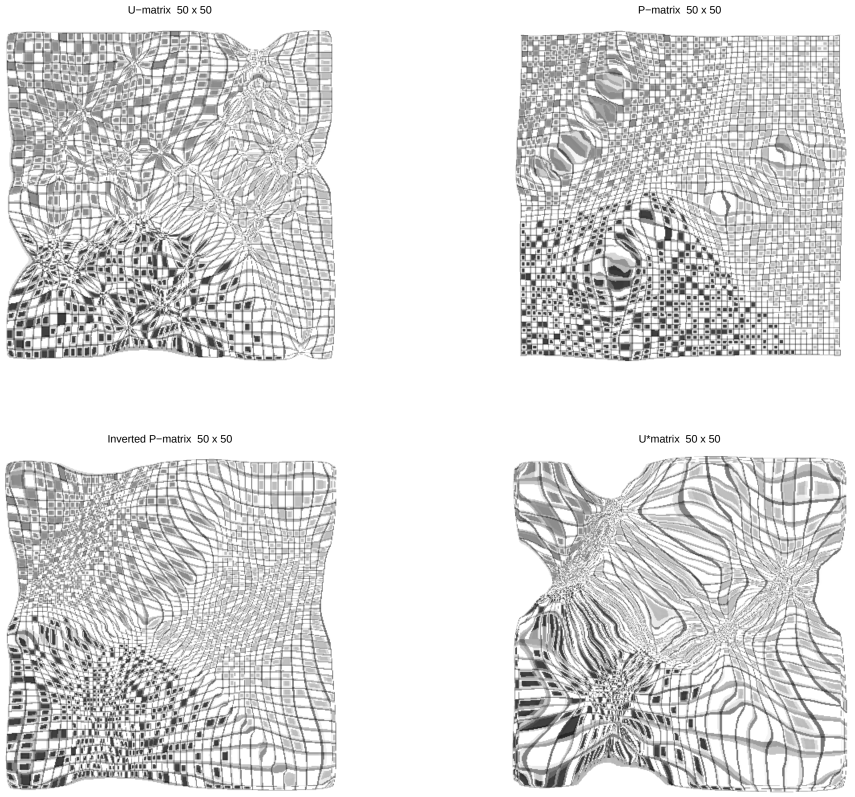

Figure 4.21: Visualization maps for a SOM model, trained for 800 iterations, of 3-dimensional data organized in three clusters, using a 50×50 square grid with rectangular layout. Display as in previous figures.

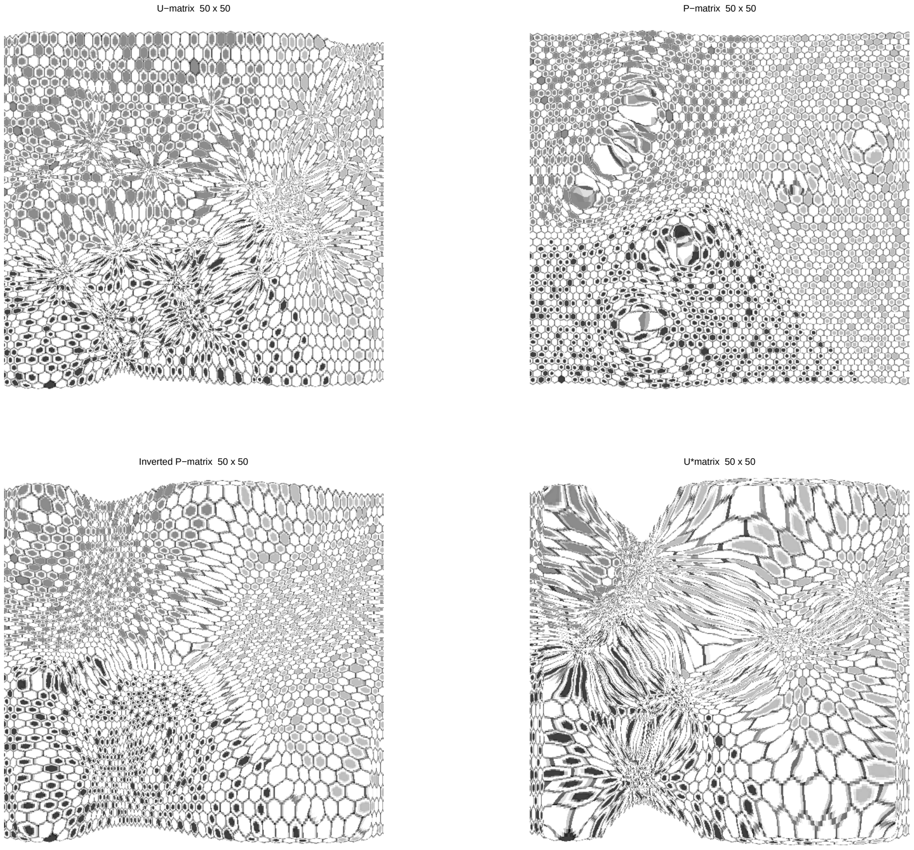

Figure 4.22: Visualization maps for a SOM model, trained for 800 iterations, of 3-dimensional data organized in three clusters, using a 50×50 square grid with hexagonal layout. Display as in previous figures.

4.3.3 Discussion

The results of this last set of experiments with the standard SOM algorithm are quite straightforward. Their interpretation can be summarized in the following points:

- Yet again, there is almost no difference between the results obtained using a square lattice and those obtained using an hexagonal one. This is the case both in terms of the cluster distribution over the representation map and in terms of the local distribution of the mapping distortion, as measured by the *U-matrix*, the *P* and inverted *P* matrices and the *U*-matrix*. This is clearly reflected in the cartogram representations of all these measure and for all map sizes.

- The effect of increasing the map size does not yield qualitatively different . The only differences can be expressed in terms of map resolution (as fewer data points are assigned to each of the map units), which increases the detail in proportion to the map size. Given that increasing the map sizes has an impact on computation times, it could be argued that increasing the map size arbitrarily has no advantage in terms of data and model visual interpretation.

- Again, it has been clearly illustrated that the achievement of convergence in the SOM training process has a strong impact on the quality of the results. Sufficient adaptive training iterations of the algorithm must be performed to achieve an adequate mapping of the data.

- The use of cartogram representations of the distortion measures has been shown to provide the data analyst with a rich and intuitive visual tool for NLDR projection display that reflects the true distribution of the modelled data more faithfully by explicitly reintroducing the local distortion into the visualization map.

The following tables 4.3 and 4.4 summarize the quantization and topographic error results for, in turn, rectangular and hexagonal lattices. The complexity row is of one iteration. The computational complexity is $\theta(Inmd)$, being I total iterations.

Grid	QE	TE	Iter.	Complexity	$\theta(Inmd)$
10×20	0.0293	0.1840	2	9,000,000	$2 \times 1500 \times 200 \times 3$
20×10	0.0304	0.1540	2	9,000,000	$2 \times 1500 \times 200 \times 3$
13×13	0.0296	0.2067	2	760,500	$2 \times 1500 \times 169 \times 3$
30×30	0.0194	0.1260	10	4,050,000	$10 \times 1500 \times 900 \times 3$
30×30	0.0159	0.1460	20	4,050,000	$20 \times 1500 \times 900 \times 3$
50×50	0.0186	0.1513	10	11,250,000	$10 \times 1500 \times 2500 \times 3$
50×50	0.0167	0.1133	17	11,250,000	$17 \times 1500 \times 2500 \times 3$
50×50	0.0158	0.1573	20	11,250,000	$20 \times 1500 \times 2500 \times 3$
50×50	0.0138	0.1300	30	11,250,000	$30 \times 1500 \times 2500 \times 3$
50×50	0.0127	0.1427	40	11,250,000	$40 \times 1500 \times 2500 \times 3$
50×50	0.0117	0.1633	50	11,250,000	$50 \times 1500 \times 2500 \times 3$
50×50	0.0097	0.1560	80	11,250,000	$80 \times 1500 \times 2500 \times 3$
50×50	0.0088	0.1520	100	11,250,000	$100 \times 1500 \times 2500 \times 3$
50×50	0.0083	0.1487	110	11,250,000	$110 \times 1500 \times 2500 \times 3$
50×50	0.0082	0.1727	120	11,250,000	$120 \times 1500 \times 2500 \times 3$
50×50	0.0024	0.1820	800	11,250,000	$800 \times 1500 \times 2500 \times 3$

Table 4.3: Results for rectangular lattices.

4.4 Experiments with the Growing Hierarchical SOM

4.4.1 First experiment

GHSOM was implemented in *MATLAB*®, using the SOM-Toolbox[1], the GHSOM-Toolbox[2] and Computer software for making cartograms (Cart)[3] using the technique described in the paper (45). A standard Gaussian neighborhood function was used and the batch-SOM method. The required U-matrices were calculated and their corresponding cartograms were generated using them and the model grids. For the U-matrix we used, for each unit, the average of the distances to its neighboring units.

We illustrate the cartogram representation of the U-matrix for the GHSOM with a basic preliminary experiment using artificial data. A total of 1,500 3-D points were randomly drawn from 3 spherical Gaussians (500 points each), all with unit variance, and with centers sitting at

[1] www.cis.hut.fi/somtoolbox

[2] www.ofai.at/ elias.pampalk/ghsom/

[3] www-personal.umich.edu/ mejn/cart/

Grid	QE	TE	Iter.	Complexity	$\theta(Inmd)$
10×20	0.0295	0.0360	2	9,000,000	$2 \times 1500 \times 200 \times 3$
20×10	0.0294	0.0687	2	9,000,000	$2 \times 1500 \times 200 \times 3$
13×13	0.0293	0.0613	2	760,500	$2 \times 1500 \times 169 \times 3$
30×30	0.0218	0.0440	6	4,050,000	$6 \times 1500 \times 900 \times 3$
30×30	0.0193	0.0580	10	4,050,000	$10 \times 1500 \times 900 \times 3$
30×30	0.0185	0.0440	12	4,050,000	$12 \times 1500 \times 900 \times 3$
30×30	0.0142	0.0493	30	4,050,000	$30 \times 1500 \times 900 \times 3$
50×50	0.0160	0.0480	17	11,250,000	$17 \times 1500 \times 2500 \times 3$
50×50	0.0175	0.0320	110	11,250,000	$110 \times 1500 \times 2500 \times 3$
50×50	0.0025	0.0680	800	11,250,000	$800 \times 1500 \times 2500 \times 3$

Table 4.4: Results for hexagonal lattices.

the vertices of an equilateral triangle. In this experiment, the three cluster regular structure was chosen as one that should yield a most basic hierarchic structure with a single level, reflecting internally homogeneous clusters in the leaves.

All results are compiled and displayed in Fig. 4.23. They include, for reference, a basic SOM map visualization of the data with a hexagonal grid in the top row. It reflects a neat but narrow separation between clusters (46) as would be expected from the non-overlapping 3-cluster structure of the synthetic data.

The GHSOM estimated a single layer in the hierarchy. This reflects the fact that each cluster has no sub-cluster structure (it is internally homogeneous). This single layer estimates three leaves stemming from the root map (each of them displayed in a different row of the figure). In each, data from each cluster are almost perfectly separated.

Given that each of this three leaves reflects a single cluster with no internal structure, we would expect them to show a locally low-varying mapping distortion. This distortion is captured by the U-matrices (left column) and the corresponding cartograms (right column). The difference between both representations is quite telling: from the color-coded visualization of the U-matrices, the analyst might wrongly infer significant differences in distortion between different areas of the map (that is, internal structure). This is correctly (and intuitively) refuted by the cartograms, which reflect the low variability in distortion, as they retain the original rectangular grid with little change. The existence of neat internal between-sub-cluster gaps would have been reflected in the form of a heavily distorted cartogram.

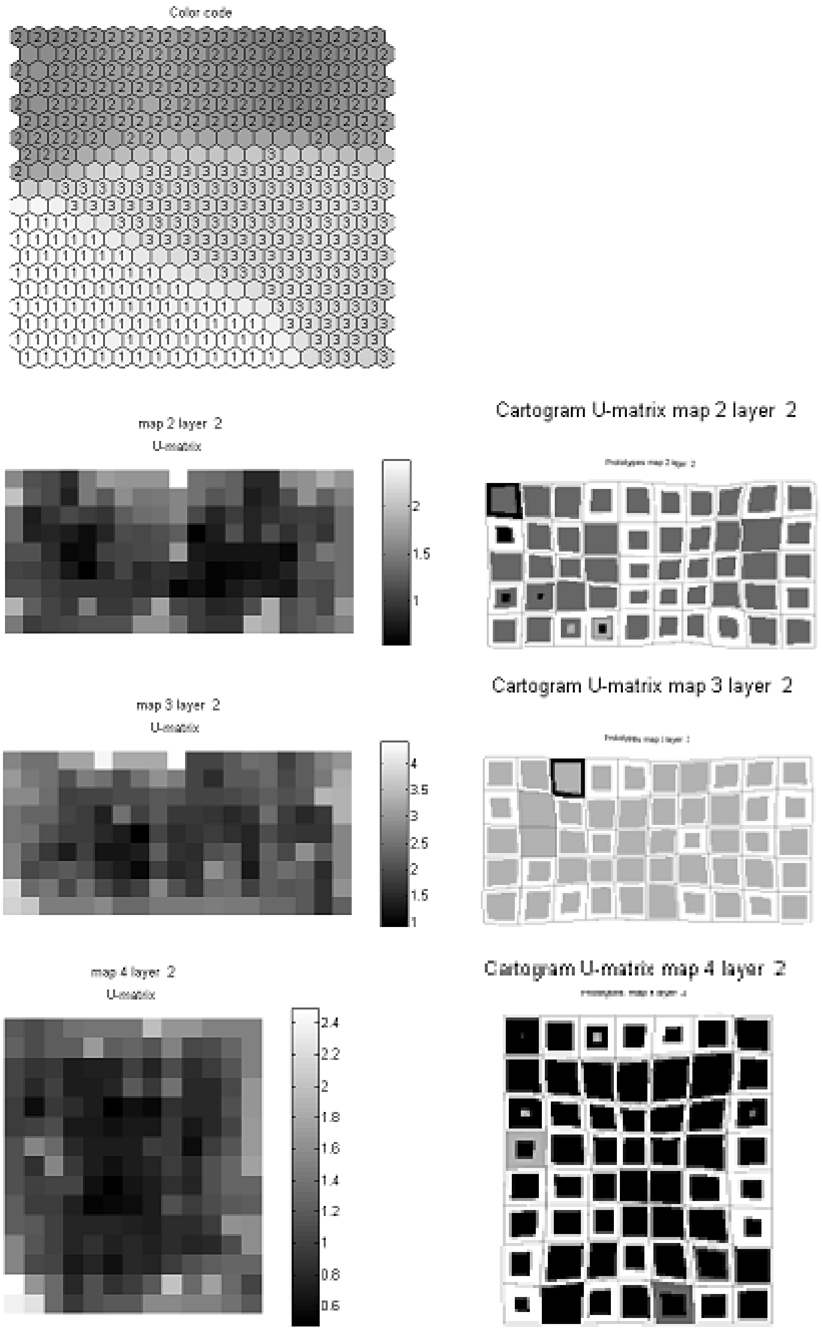

Figure 4.23: *Top row*: Data projection of the three isolated and equally-spaced clusters in the hexagonal 20×20 grid of a trained SOM. The original clusters were labeled 1-to-3 and, thus, the labels 1-to-3 in each unit of the visualization map indicate that clusters have been correctly separated by the model. Each of the following rows correspond to one of the three leaves of the hierarchy. *Left column*: U-matrices corresponding to each of the leaves of the hierarchy, in grey-scale representation coding, quantified in the colorbar on the right of each map. *Right column*: Cartogram coding of the U-matrix values in the grid. Each of the three clusters are codified in different shadows of grey. More than two different shadows overlapping in a SOM unit indicates that data points of different clusters were assigned to the same BMU.

4.4.2 Second experiment

For the second experiment, also 1,500 3-D points were randomly drawn, but this time they correspond to 5 spherical Gaussians (300 points each), all with unit variance, but heterogeneously distributed in three groups: one of them consisting of three clusters close to each other but without overlapping, and two consisting on isolated clusters at similar distances to the previous group of three. This configuration is meant to reflect a hierarchical structure that should reveal a first partition between the three groups and a secondary one between the three clusters of the first group. This last level should yield internally homogeneous clusters in the leaves of the hierarchy.

All results are compiled and displayed in Figs. 4.24, 4.25 and 4.26.

Fig. 4.24 displays the hierarchical structure found by GHSOM. It is precisely as expected: it includes a first level in which a cluster 1 dominated map is separated from a cluster 2 dominated one and these, in turn, are separated from the group

The 3 maps of this first level of the hierarchy are shown in 4.25. The first two rows correspond to the maps assigned to clusters 1 and 2. As we might expect theiy are internally homogeneous and, therefore, yield a completely "flat" cartogram, devoid of heterogeneity in the local distortion.

The third map (last row), represents the group containing clusters 3, 4, and 5 and due to the internal heterogeneity of this group. GHSOM finds sufficient heterogeneity as to split into a second level of the hierarchy. The three maps of this second level displayed in 4.26, separate the three clusters quite nicely and, again, yield cartograms that are devoid of heterogeneity in the local distortion.

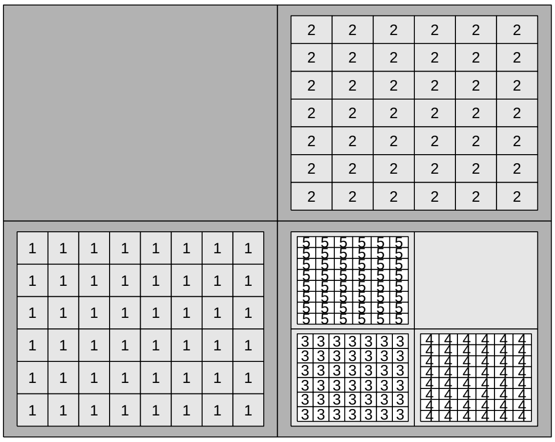

Figure 4.24: GHSOM map partition for the second experiment as reported in the text.

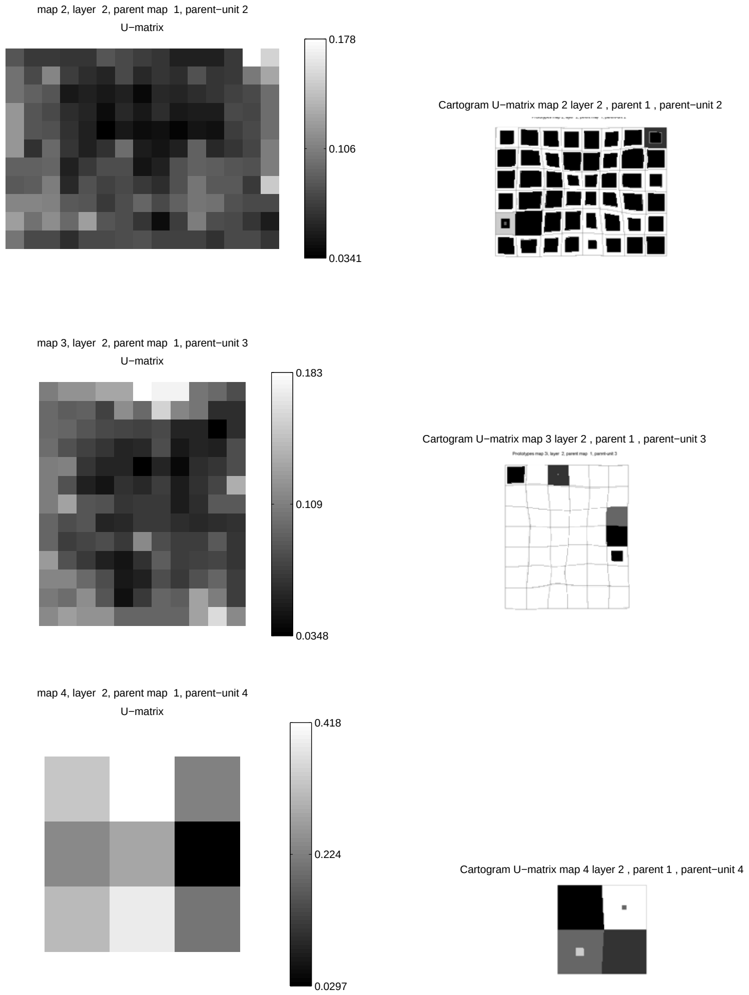

Figure 4.25: GHSOM map partition for the second experiment as reported in the text. First level of the hierarchy. Display as in Fig. 4.23

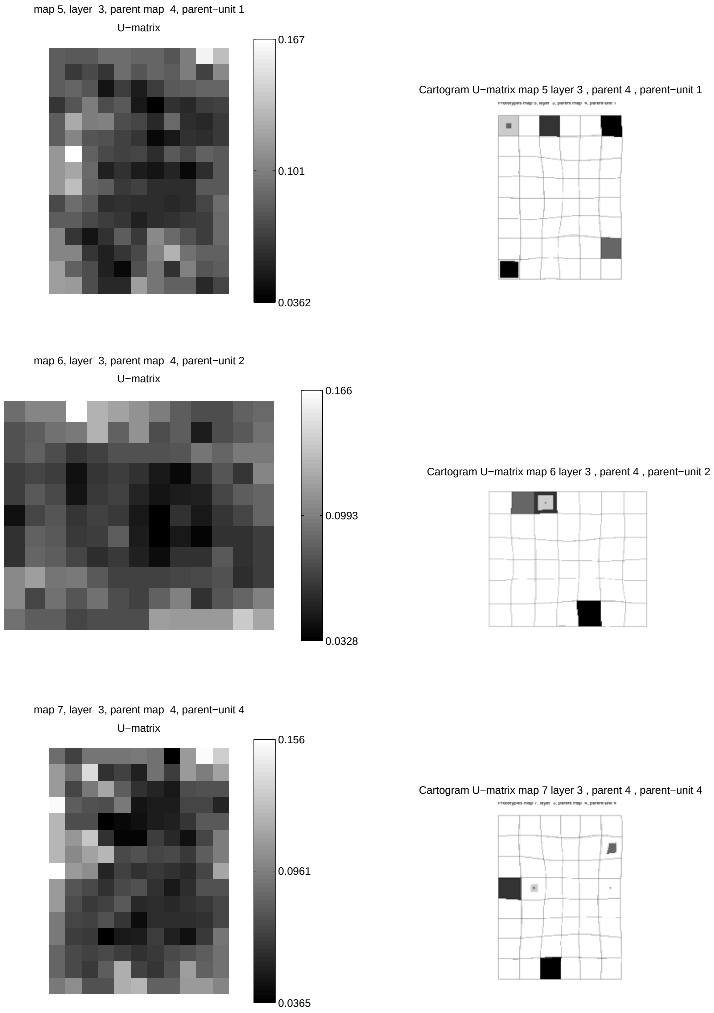

Figure 4.26: GHSOM map partition for the second experiment as reported in the text. Second level of the hierarchy. Display as in Fig. 4.23

Chapter 5

Experimentation with real data

Once the main capabilities and limitations of the cartogram visualization of MVD for SOM-based models have been investigated in some detail with artificial data, we now proceed to illustrate the method using real data stemming from a neuro-oncology problem. It involves the discrimination of human brain tumor types, a problem for which knowledge discovery techniques in general, and data visualization in particular (42) should be useful tools.

5.1 Materials: Neuro-oncology data

The available data are single-voxel[1] (and therefore spatially localized in an area of the brain) proton magnetic resonance spectroscopy (SV-^{1}H-MRS) cases acquired in vivo from brain tumor patients. They are part of the multi-center, international web-accessible INTERPRET project database (47). A total of eight clinical centers from five countries contributed cases to this database.

The spectra provide a metabolic signature of the brain tissue (be it tumor or healthy), as certain metabolites are known to be reflected by resonances at certain frequencies or bands of frequency.

The analyzed data were acquired at long echo time (LET). The echo time is an influential parameter in ^{1}H-MRS data acquisition. The use of LET yields relevant information on fewer metabolites, but with clearly resolved amplitude peaks and little baseline distortion, resulting in a more readable spectrum. The data include 78 glioblastomas, 31 metastases (these are

[1] A voxel is a volume element - a cube - within a grid covering a the global delimited volume under study in the brain.

high-grade, malignant tumors of poor prognosis; importantly in our experiments, both of these pathologies are know to be heterogeneous as expressed by their SV-^{1}H- MRS), 15 normal (healthy) tissue cases (which should have a very homogeneous SV-^{1}H-MRS signature), and 8 abscesses (abnormal masses that may or may not be a byproduct of tumors, but which are often distinct from the tumors themselves).

Clinically-relevant regions of the spectra were sampled to obtain 195 frequency intensity values (data features), spanning approximately from 4.22 down to 0.49 ppm (parts per million) in the frequency range. These frequency intensity values become the input to our self-organizing models.

5.2 Experiments

Two problems were investigated:

- Glioblastomas vs. normal tissue: Both types of brain tissue should be well separated (as they both differ radically in their metabolic composition), but their visualization should reveal that, while normal tissue forms a compact group, glioblastomas lack homogeneity. Atypical cases might be expected (44).
- Metastases vs. normal tissue and abscesses: These types should also be separated due to their different metabolic composition. As previously mentioned, normal tissue forms a compact group and most abscesses should be similar. On the contrary, metastases should not show much homogeneity. Some atypical cases might again be expected (44).

5.2.1 Results and discussion

Glioblastomas vs. normal tissue

Aggressive tumors (glioblastomas) and normal (healthy) tissue data were investigated in this experiment using GHSOM. For this, the model settings included a standard Gaussian neighborhood function; a linear initialization on the weights; $(\tau_1) = 0.55$, $(\tau_2) = 0.003$, and the sequential-SOM method as a basis.

The number of grid nodes is a key parameter as well. The use of a small number of nodes yields high quantization error but well-defined clusters, while a large number of nodes results in low quantization error but, in the most extreme case, a cluster for each data sample, which might not be useful to investigate cluster structure.

Some balance must thus be struck in the choice of this parameter. Several of these choices were investigated in order to obtain the best workable and visually revealing U-matrix. A map size with 9 rows and 6 columns as a approximation of the Eq. 4 was finally selected as the best map size to represent the data. The required U-matrices of each map were calculated and their corresponding cartogram representations were generated.

The results for the first of the problems include SOM (where the map dimensions were approximated according to Eq. 4) and GHSOM modelling. The shape of the map grid was chosen according to the approximate ratio between the two largest eigenvalues of the covariance matrix of the input vectors (of values 661.63 and 384.59).

It is interesting to see that the dimensions of the estimated SOM map are very similar to the ones chosen for the parent map of GHSOM.

Let us first have a look at the SOM results, as reported in Fig. 5.1. As expected, normal tissue has mostly been mapped to a tight and compact area of the map in the bottom left corner, predominating, in fact, in only two units of the grid. The area occupied by normal tissue is neatly separated from the area dominated by glioblastomas by an empty space. Beyond it, glioblastomas occupy most of the rest of the map, but, again as expected, in a very heterogeneous manner. That is, a clear subgroup structure is observed, with empty spaces between groups of glioblastomas. Moreover, the values of the U-matrix reveal that there is a core dense subgroup of them in the middle-right hand side of the maps but more heterogeneous groups in the rest of the areas were they are mapped. In any case, the cartogram representation in the same figure clearly indicates that these local distortions are rather moderate, signifying that the mapping process has probably been rather homogeneous and not too-nonlinear in nature.

Now moving to the first-layer representation generated by the GHSOM, whose results are displayed in Fig. 5.2, we find a fairly similar situation. Normal tissue is again quite isolated (now mostly in four units at the bottom-right corner of the square map) and neatly separated from the rest of glioblastomas. Consistently with the SOM results the local distortion reflected by the cartogram representation suggests an homogeneous and predominantly linear mapping process.

For glioblastomas, a second level of the hierarchy only appears in units of the left hand-side of the map that are also densely populated while undergoing relatively high distortions. Only the most populated unit of normal tissue splits up to a second level of the hierarchy. The maps of this second level of the hierarchy are shown in Fig. 5.3 and its continuation, 5.4. All these maps have almost completely regular cartograms, indicating an almost complete lack of distortion on the local mapping and thus an almost complete lack of internal cluster sub-structure.

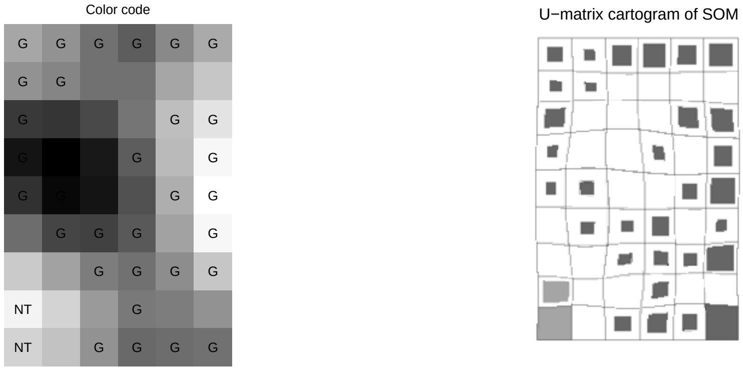

Figure 5.1: Visualization maps corresponding to the trained SOM with a 9 × 6 rectangular grid for the *Glioblastomas vs. Normal Tissue* problem. Left: Color coded distance U-matrix, where dark shades of gray correspond to high distortion values and light shades correspond to low values. Each unit of the map is labeled according to the predominant type of tissue (G for *Glioblastomas*, NT for *Normal Tissue* and unlabeled when no data point has been mapped into that unit). Right: Corresponding cartogram, where the relative size of the solid gray area inside each unit is directly proportional to the ratio of data inputs mapped into that unit. Light gray are predominantly *Normal Tissue* units and dark gray are predominantly *Glioblastomas*.

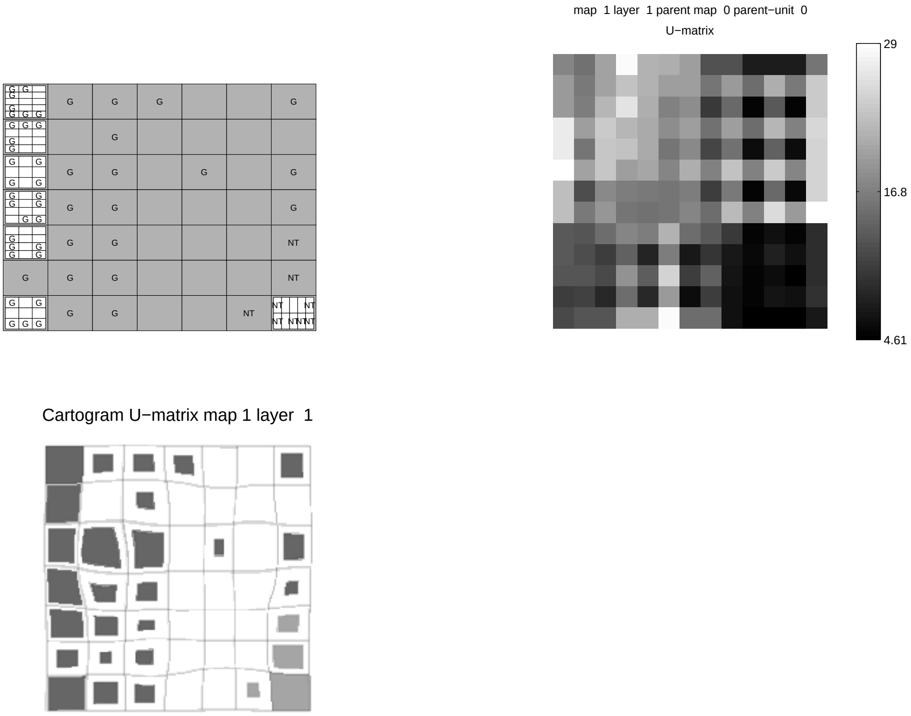

Figure 5.2: Visualization maps corresponding to the first level of the hierarchy of the trained GHSOM for the *Gliblastomas vs. Normal Tissue* problem. Top row, left: Hierarchy map, with labels as in previous figure. Six glioblastoma-dominant units on the left hand-side column (showing the corresponding internal grid) are further split into the second layer. From top to bottom they will be labelled with numbers 1 to 6. A Normal Tissue-dominant unit on the bottom right corner is also split into a second layer of the hierarchy level. All the split units in the second level are shown in Fig. 5.3; right: Color coded complete U-matrix. Bottom row: Corresponding cartogram, displayed as in previous figure.

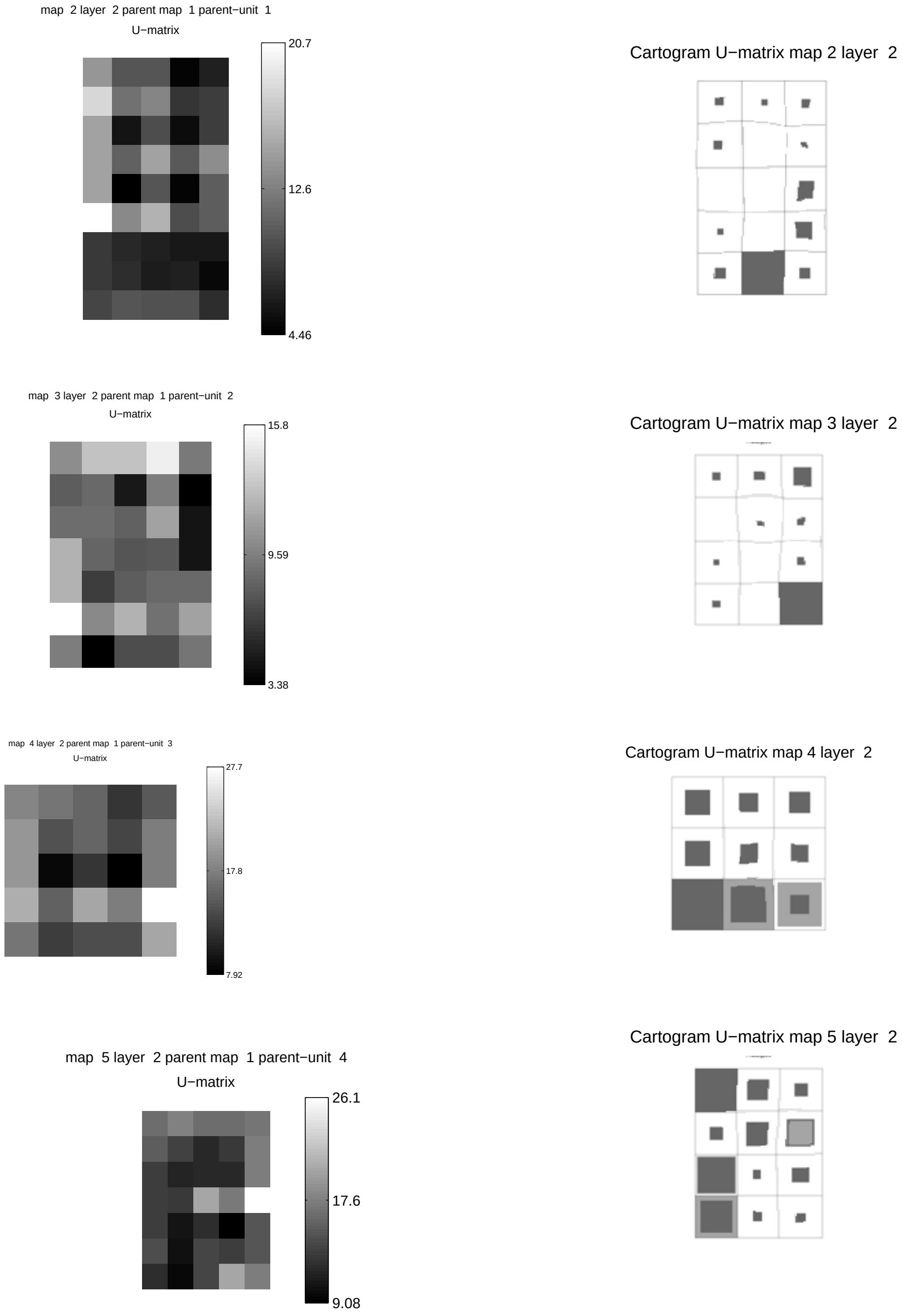

Figure 5.3: Visualization maps corresponding to the second level of the hierarchy of the trained GHSOM for the *Gliblastomas vs. Normal Tissue* problem. Left column: Color coded complete U-matrices; from top to bottom, glioblastoma-dominated units from 1 to 6, as labelled from the previous figure, followed in the last row by the Normal Tissue-dominated unit. Right column: Corresponding cartograms, displayed as in previous figures with the difference that the proportion of cases belonging to each of the two classes is explicitly.

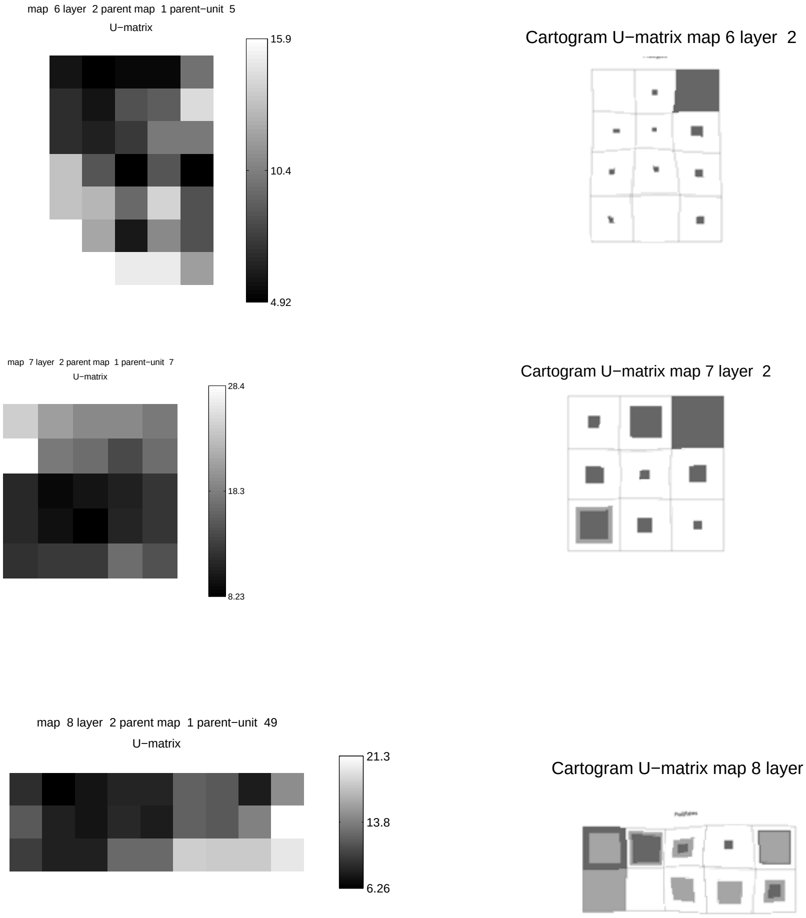

Figure 5.4: Continuation of previous figure.

Assessment of the quality of SOM and GHSOM

As already mentioned in previous experiments, the issue of gauging the quality of SOM-based methods is a complicated one, as the measures are quite data-dependent. Typically, the quality of the map is measured in terms of the training data and two evaluation criteria: resolution (average quantization error, or average distance between data vectors and their BMUs) and topology preservation (topographic error as the proportion of all data vectors for which first and second BMUs are not neighboring units), in turn QE and TE.

Table 5.1 summarizes the quality measures for the two methods.

Method	Map	Units	QE	TE
SOM		9×6	34.938	0
GHSOM	Map 1 level 1 parent-map 0 parent-unit 0	7×7	33.031	0.097
	Map 2 level 2 parent-map 1 parent-unit 1	5×3	84.509	0.473
	Map 3 level 2 parent-map 1 parent-unit 2	4×3	73.226	0.215
	Map 4 level 2 parent-map 1 parent-unit 3	3×3	66.404	0.140
	Map 5 level 2 parent-map 1 parent-unit 4	5×3	61.612	0.376
	Map 6 level 2 parent-map 1 parent-unit 5	4×3	62.992	0.054
	Map 7 level 2 parent-map 1 parent-unit 7	3×3	69.526	0.140
	Map 8 level 2 parent-map 1 parent-unit 49	2×5	84.332	0.215

Table 5.1: QE: quantification error; TE: topological error.

Metastases vs. normal tissue and abscesses

The second experiment concerned metastases, normal tissue and abscesses using (as in the previous case) a standard Gaussian neighborhood function; a linear initialization of weights; $(\tau_1) = 0.55$, $(\tau_2) = 0.003$, and the sequential-SOM method. A map size with 8 rows and 5 columns was selected as the best map size to represent the data.

The required U-matrices of each map were calculated and their corresponding cartograms were generated.

The results for this second problem again include SOM (where the map dimensions were approximated according to Eq. 4) and GHSOM modeling. The shape of the map grid was chosen according to the approximate ratio between the two largest eigenvalues of the covariance matrix of the input vectors (of values 478.03 and 334.29).

Also for this problem, the dimensions of the estimated SOM map are very similar to the ones chosen for the parent map of GHSOM.

The SOM results, as reported in Fig. 5.5, provide us with a less obvious image than the previous problem. Once again as expected, normal tissue has mostly been mapped to a tight and compact area of the map in the bottom left corner, predominating in four units of the grid. The area occupied by normal tissue is reasonably well separated from the area dominated by metastases and abscesses. Beyond it, metastases and abscesses share the same areas, although a good deal of abscesses occupy the right hand-side of the map. Metastases are almost as heterogeneous as glioblastomas in the previous problem. That is, clear subgroup structure is observed, with empty spaces between subgroups. Yet again, the cartogram representation in the same figure clearly indicates that local distortions measured by the U-matrix are rather moderate, signifying that the mapping process has probably been rather homogeneous and not too-nonlinear in nature.

The maps of the first-layer representation generated by the GHSOM are displayed in Fig. 5.6. Metastases are separated in subgroups, predominantly in the left and central columns of the map. Normal tissue is again quite isolated (now mostly in four units at the bottom-right corner of the square map) and neatly separated from the rest of the data. Abscesses predominate dispersely in right hand-side areas of the map. Consistently with the SOM results the local distortion reflected by the cartogram representation suggests an homogeneous and predominantly linear mapping process.

For metastases, a second level of the hierarchy only appears in units of the left hand-side of the map that are also densely populated (light for glioblastomas in the previous problem). The two normal tissue units split up to a second level of the hierarchy. The maps of this second level of the hierarchy are shown in Fig. 5.7. All these maps have almost completely regular cartograms, indicating an almost complete lack of distortion on the local mapping and thus an almost complete lack of internal cluster sub-structure.

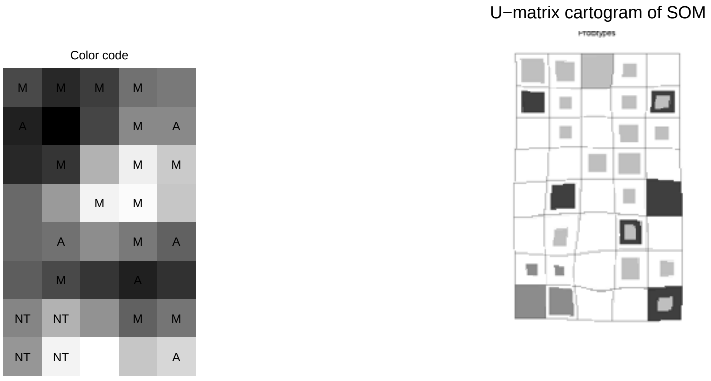

Figure 5.5: Visualization maps corresponding to the trained SOM with a 8 × 5 rectangular grid for the *Metastases vs. Normal Tissue vs. Abscesses* problem. Left: Color coded distance U-matrix, where dark shades of gray correspond to high distortion values and light shades correspond to low values. Each unit of the map is labelled according to the predominant type of tissue (M for *Metastases*, NT for *Normal Tissue*, A for *Abscesses* and unlabelled when no data point has been mapped into that unit). Right: Corresponding cartogram, where the relative size of the solid gray area inside each unit is directly proportional to the ratio of data inputs mapped into that unit. Light gray are predominantly *Metastases* units, medium gray are predominantly *Normal Tissue* and dark gray are predominantly *Abscesses*.

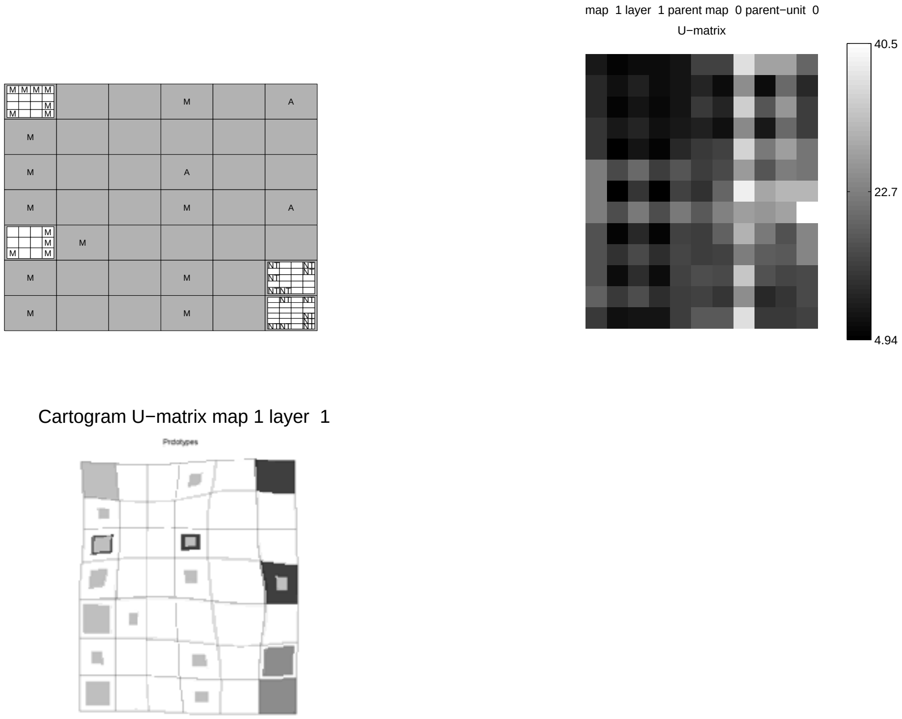

Figure 5.6: Visualization maps corresponding to the first level of the hierarchy of the trained GHSOM for the *Metastases vs. Normal Tissue vs. Abscesses* problem. Top row, left: Hierarchy map, with labels as in previous figure. Two glioblastoma-dominant units on the left hand-side column (showing the corresponding internal grid) are further split into the second layer. From top to bottom they will be labelled with numbers 1 and 2. Two Normal Tissue-dominant units on the bottom right corner are also split into a second layer of the hierarchy level, and labelled as 3 and 4. All the split units in the second level are shown in Fig. 5.7; right: color coded complete U-matrix. Bottom row: Corresponding cartogram, displayed as in previous figure.

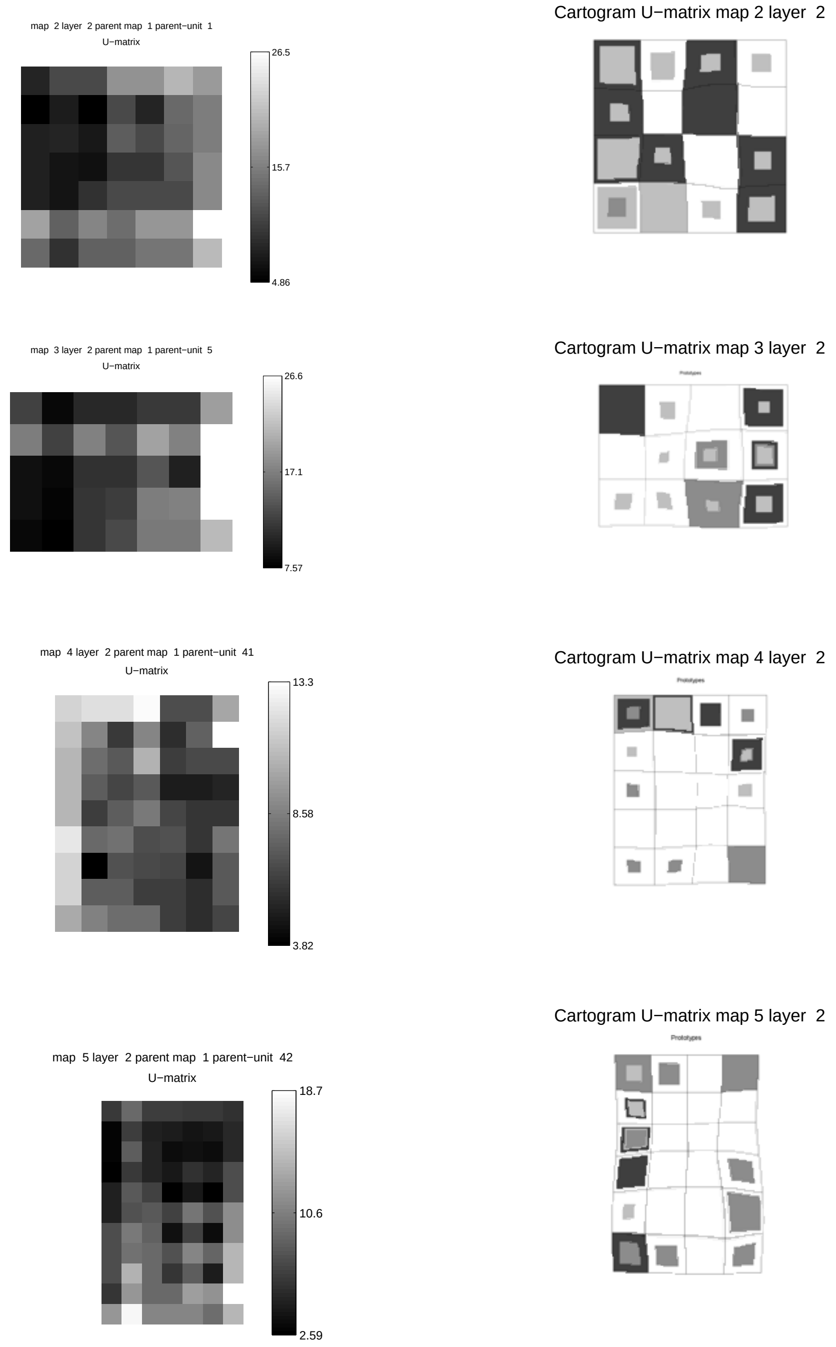

Figure 5.7: Visualization maps corresponding to the second level of the hierarchy of the trained GHSOM for the *Metastases vs. Normal Tissue vs. Abscesses* problem. Left column: Color coded complete U-matrices; from top to bottom, metastases-dominated units 1 and 2, followed by the Normal Tissue-dominated units 3 and 4. Right column: Corresponding cartograms, displayed as in previous figures.

Table 5.2 summarizes the quality measures for the two methods.

Method	**Map**	**Units**	**QE**	**TE**
SOM		8×5	37.985	0.037
GHSOM	Map 1 level 1 parent-map 0 parent-unit 0	7×6	36.663	0.019
	Map 2 level 2 parent-map 1 parent-unit 1	4×4	84.556	0
	Map 3 level 2 parent-map 1 parent-unit 5	3×4	68.957	0.037
	Map 4 level 2 parent-map 1 parent-unit 41	5×4	79.431	0.093
	Map 5 level 2 parent-map 1 parent-unit 42	6×4	80.464	0.037

Table 5.2: QE, quantification error. TE, topological error.

Chapter 6

Conclusions

The visualization of MVD can provide us with inductive reasoning insights that could be difficult to gain from direct deductive reasoning from the raw data. Most available observed data, though, involve information contained in many variables and are thus commonly expressed, in a quantitative manner, through high-dimensional data.

High-dimensional data can only be visually inspected in an indirect fashion. One possibility is that of using projection methods. Linear projection methods, in particular, have been used for decades to this purpose. They are fairly easy to interpret, even though the faithfulness of their representation is limited when data are complexly structured.

Over the last decade, NLDR methods for MVD visualization (2) have provided novel approaches to circunvent this limitation but their adoption is hindered by the difficulty of interpreting the visualizations they provide in terms of the original data attributes and also by the non-uniform distortion they generate.

In this Master Thesis we have adapted a technique, originally defined for the distortion of geographic maps according to underlying attributes, to provide MVD visualization using NLDR models of the self-organizing family, namely, SOM and GHSOM. The latter is extension of the former that includes partially-adaptive architecture and hierarchical mapping structure.

We have shown, through a batch of experiments, that the proposed density-equalizing cartogram representation of the visualization maps of SOM and GHSOM allows explicitly reintroducing the mapping distortion created by the models, so that more intuitive data visualizations are created. The capabilities and limitations of the proposed technique have been assessed through both artificial and real medical data concerning a human neuro-oncology problem.

The contributions of this thesis can be summarized as follows:

- Previous preliminary experiments for the cartogram representation of the U-Matrix in batch-SOM were carried out in (48). For the sequential SOM model, though, the cartogram-based visualization of MVD has for the first time been investigated in some detail using not only the more standard U-matrix as a proxy for local nonlinear distortion, but also the P and, importantly, the inverted P matrices as well as the U*-matrix, which combines the information of the U and the P.
- The cartogram-based method has been defined and investigated for the first time (to the best of the author's knowledge) as applied to the GHSOM model.
- The hierarchical clustering setting has been applied for the first time to the problem of human brain tumour cluster structure exploration from SV-^{1}H-MRS data.
- The cartogram-based method for MVD visualization in NLDR methods has for the first time been used in the aforemention neuro-oncology problem using self-organizing artificial neural networks.

6.1 Proposals for future research

The research carried out for this Master Thesis does not provide closure, by any means, to the investigation in the problem of cartogram-based visual MVD exploration in NLDR methods. You could say that, in fact, this research has opened up some possibilities that could only be investigated in full by following some of the lines that we summarily suggest next:

- All experiments in this thesis have concerned just two self-organizing artificial neural network models, namely the standard SOM and the GHSOM. As stated in Chapter 2, these are two options out of the many self-organizing methods defined in the literature. Cartogram representations could be defined for any of them. We consider that some specially interesting alternatives would include Neural Gas models and alternative Growing SOM variants.
- Beyond alternative self-organizing alternatives, cartogram visualization of nonlinear distortion can be implemented in any other NLDR method that provides prototype-based data mapping. That is, it could be investigated for other manifold learning techniques and even for non-prototype-based methods using, for instance, Voronoi tesselations of the data projections.

- The experiments carried out in Chapter 4 using artificial data have involved the variation of a given number of parameters, such as number of clusters, data dimensionality or model architecture. These variations do not cover all the available possibilities. In fact, future research should involve the simultaneous variation of more than one of these characteristics. For instance, we could investigate the simultaneous effect of increasing the data dimensionality and the number of clusters.

- The aforementioned experiments with artificial data have been limited to data sets of very specific and simple characteristics (fairly symmetric and clearly separated clusters). Further research should consider the use of data set of a wider range of characteristics, such as partially overlapping clusters, clusters of varying densities, or clusters contaminated with uninformative noise, just to list a few possibilities.

Publications resulting from the thesis

Àngela Martín and Alfredo Vellido. Cartogram-Based Data Visualization using the Growing Hierarchical SOM. in Proceedings of the *Decimosexto Congreso Internacional de la Asociación Catalana de Inteligencia Artificial* (CCIA 2013), Vic, Barcelona, Spain, October 2013.

Acknowledgments

I want to express my gratitude to my supervisor, Dr. Alfredo Vellido, whose expertise, effort and patience have made this work possible. I must also acknowledge Mrs. Alessandra Tosi and Mr. David L. García for their help when it was required.

I must also acknowledge my husband and my two children, whose love, encouragement and assistance have made this work more bearable.

Finally, I must also thank my colleagues of the Department of Llenguatges i Sistemes Informàtics of the Universitat Politècnica de Catalunya in Terrassa for their support and understanding.

Bibliography

[1] Vellido, A., Martín, J. D., Rossi, F., and Lisboa, P. J. (2011). Seeing is believing: The importance of visualization in real-world machine learning applications. In Procs. of the 19th European Symposium on Artificial Neural Networks (ESANN 2011), Bruges, Belgium, pp.219-226.

[2] Lee, J. J. A., and Verleysen, M. (2007). *Nonlinear Dimensionality Reduction*. Springer.

[3] Vellido, A., Martín-Guerrero, J. D., and Lisboa, P. (2012). Making machine learning models interpretable. In Procs. of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2012), Bruges, Belgium, pp.163-172.

[4] Jolliffe, I. (2005). *Principal Component Analysis*. John Wiley and Sons(2nd e edition).

[5] Gastner, M. T., and Newman, M. E. (2004). Diffusion-based method for producing density-equalizing maps. *Proceedings of the National Academy of Sciences of the United States of America*, 101(20), 7499-7504.

[6] Kohonen, T. (2001). *Self-Organizing Maps*. Springer (third edition).

[7] Lisboa, P. J., Vellido, A., Tagliaferri, R., Napolitano, F., Ceccarelli, M., Martín-Guerrero, J. D., and Biganzoli, E. (2010). Data Mining in cancer research IEEE *Computational Intelligence Magazine*, 5(1), 14-18.

[8] Cruz-Barbosa, R., and Vellido, A. (2011). Semi-supervised analysis of human brain tumours from partially labeled MRS information, using manifold learning models. *International Journal of Neural Systems*, 21(01), 17-29. 1, 5
1, 3, 8, 9, 89

[9] Post, F. H., Nielson, G. M., and Bonneau, G. P. (Eds.) (2003). Data Visualization: The State of the Art. Springer.

[10] Van der Maaten, L. J. P., Postma, E. O., and Van Den Herik, H. J. (2009). Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10, 1-41.

[11] Levina, E., and Bickel, P. J. (2004). Maximum likelihood estimation of intrinsic dimension. In Advances in Neural Information Processing Systems, pp. 777-784.

[12] Camastra, F., and Vinciarelli, A. (2001). Intrinsic dimension estimation of data: An approach based on GrassbergerProcaccia's algorithm. *Neural Processing Letters*, 14(1), 27-34.

[13] Lee, J. J. A., and Verleysen, M. (Eds.). (2007). *Nonlinear Dimensionality Reduction*. Springer.

[14] Ye, J., and Ji, S. (2010). *Discriminant Analysis for Dimensionality Reduction: An Overview of Recent Developments. Biometrics: Theory, Methods, and Applications*. Wiley-IEEE Press, New York.

[15] Roweis, S. T., and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500), 2323-2326.

[16] Paulovich, F. V., Eler, D. M., Poco, J., Botha, C. P., Minghim, R., and Nonato, L. G. (2011). Piece wise Laplacian-based projection for interactive data exploration and organization. *Computer Graphics Forum*, 30(3), 1091-1100.

[17] Venna, J. (2007). Dimensionality reduction for visual exploration of similarity structures. PhD Thesis, Helsinki University of Technology.

[18] Aupetit, M. (2007). Visualizing distortions and recovering topology in continuous projection techniques. *Neurocomputing*, 70(7), 1304-1330.

[19] Lee, J. J. A., and Verleysen, M. (2007). *Nonlinear Dimensionality Reduction*. Springer.

[20] Mayer, R., Aziz, T. A., and Rauber, A. (2007). Visualising class distribution on self-organising maps. In Procs. of ICANN 2007 (pp. 359-368), Springer.

[21] Yin, H. (2008). Learning nonlinear principal manifolds by self-organising maps. In *Principal Manifolds for Data Visualization and Dimension Reduction* Lecture Notes in Computational Science and Enginee Vol.58, pp. 68-95, Springer.

[22] Silva, B., and Marques, N. C. (2007). A hybrid parallel SOM algorithm for large maps in data-mining. In Procs. of the 13^{th} Portuguese Conference on Artificial Intelligence (EPIA07), workshop on business intelligence, Guimaraes, Portugal, IEEE. 2, 9

[23] Uriarte, E. A., and Martín F. D. (2005). Topology preservation in SOM. *International Journal of Applied Mathematics and Computer Sciences*, 1(1), 19-22. 2, 8

[24] Ultsch, A., and Siemon, H. P. (1990). Kohonenś Self Organizing Feature Maps for Exploratory Data Analysis. In Procs. of the International Neural Network Conference INNC'90, pp. 305-308 3, 9, 29, 30

[25] Ultsch, A. (2004). U*-matrix: a tool to visualize clusters in high dimensional data. Technical Report 36, CS Department, Philipps-University Marburg, Germany. 3, 8, 34

[26] Wilson, D. R., and Martinez, T. R. (2003). The general inefficiency of batch training for gradient descent learning. *Neural Networks*, 16(10), 1429-1451.

[27] Nöcker, M., Mörchen, F., Ultsch, A. (2006). An algorithm for fast and reliable ESOM learning. In Procs. of the 14^{th} European Symposium on Artificial Neural Networks (ESANN 2006), Bruges, Belgium, pp. 131-136. 4

[28] Ultsch, A. (1999). Data mining and knowledge discovery with emergent self-organizing feature maps for multivariate time series. In: Oja, E., Kaski, S. (Eds.) *Kohonen Maps*, pp. 33-46.

[29] Amarasiri, R., Alahakoon, D., and Smith, K. (2004). Applications of the growing self organizing map in high dimensional data. In Procs. of the International Information Technology Conference (IITC 2004), Colombo, Sri Lanka. 4

[30] Henriques, R., Lobo, V., and Basao, F. (2012). Spatial Clustering Using Hierarchical SOM. In Johnsson, M. (Ed.) *Applications of Self-Organizing Maps*, Ch.12. 6

[31] Martinetz, T., Schulten, K. (1991). A "neural-gas" network learns topologies. In Kohonen, T. et al. (Eds.) *Artificial Neural Networks*, pp. 397-402, Elsevier.

[32] García, S., Rowe, D., Gonzílez, J., and Villanueva, J. J. (2005). Articulated object modelling using Neural Gas networks. In Procs. of Visualization, Imaging, And Image Processing: Fifth IASTED International Conference. 6

[33] Pena, M., Barbakh, W., and Fyfe, C. (2008). Topology-preserving mappings for data visualisation. In *Principal Manifolds for Data Visualization and Dimension Reduction*, pp. 131-150, Springer.

[34] Luttrell, S.P. (1989). Hierarchical self-organizing networks. In Procs. of the International Conference on Neural Networks (ICANN'89). London, U.K., pp.2-6.

[35] Vicente, D., and Vellido, A. (2004). Review of hierarchical models for data clustering and visualization. Tendencias de la Minera de Datos en España, Red Española de Minera de Datos.

[36] Rodrigues, J. S., and Almeida, L. B. (1990). Improving the learning speed in topological maps of patterns. In Procs. of the International Neural Networks Conference (INNC'90), pp. 813-816.

[37] Fritzke, B. (1995). Growing grid A self-organizing network with constant neighborhood range and adaptation strength. *Neural Processing Letters*, 2(5), 9-13.

[38] Rauber, A., Merkl, D., and Dittenbach, M. (2002). The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. *IEEE Transactions on Neural Networks*, 13(6), 1331-1341. 6

[39] Venna, J., and Kaski, S. (2006). Local multidimensional scaling. *Neural Networks*, 19(6), 889-899. 6, 7

[40] Vesanto, J., Sulkava, M., and Hollmén, J. (2003) On the decomposition of the self-organizing map distortion measure. In Procs. of the Workshop on Self-Organizing Maps (WSOM'03), (pp. 11-16). 7

[41] Basao, F. and Lobo, V. (2012). Introduction to Kohonenś Self Organizing Maps, Department of Software Technology, Vienna.

[42] Vellido, A., and García, D., Nebot A. (2013) Cartogram visualization for nonlinear manifold learning models. *Data Mining and Knowledge Discovery*, 27(1), 22-54. 8

8

8

8

[43] Merelo, J.J. *Mapa autoorganizativo de Kohonen*, Tutorial, Dpto. Arquitectura y Tecnologa de Computadores Escuela Tcnica Superior de Ingenera Informtica, Granada(Spain), URL: http://geneura.ugr.es/ jmerelo/tutoriales/bioinfo 8, 28

[44] Vellido, A., Romero E., González-Navarro F.F., Belanche-Muñoz L., Julià-Sapé M., Arús, C. (2009) Outlier exploration and diagnostic classification of a multi-centre 1H-MRS brain tumour database. *Neurocomputing* 72(13-15), 3085-3097.

[45] Gastner, M. T., and Newman, M. E. (2004). Diffusion-based method for producing density-equalizing maps. *Proceedings of the National Academy of Sciences of the United States of America*, 101(20), 7499-7504.

[46] Bengio, Y., Buhmann, J.M., Embrechts, M. and Zurada, J.M. Introduction to the special issue on neural networks for data mining and knowledge discovery, *IEEE Transactions on Neural Networks*, 11(3), 545-549, 2000.

10

10, 11

[47] Julià-Sapé, M., Acosta, D., Mier, M., Arús, C., and Watson, D. (2006). A multi-centre, web-accessible and quality control-checked database of in vivo MR spectra of brain tumour patients. *Magnetic Resonance Materials in Physics, Biology and Medicine*, 19(1), 22-33. 11, 17

12

13, 24, 29

[48] Tosi A, Vellido A (2012) Cartogram representation of the batch-SOM magnification factor. In ESANN 2012, Bruges, Belgium, 25-27th of April, pp 203-208.