SOM-Based Data Visualization Methods

Juha Vesanto
Laboratory of Computer and Information Science
Helsinki University of Technology
P.O.Box 5400, FIN-02015 HUT, Finland
Juha.Vesanto@hut.fi

November 19, 1999

Abstract

The Self-Organizing Map (SOM) is an efficient tool for visualization of multidimensional numerical data. In this paper, an overview and categorization of both old and new methods for the visualization of SOM is presented. The purpose is to give an idea of what kind of information can be acquired from different presentations and how the SOM can best be utilized in exploratory data visualization. Most of the presented methods can also be applied in the more general case of first making a vector quantization (e.g. k-means) and then a vector projection (e.g. Sammon's mapping).

Keywords: self-organizing map, data mining, visualization, vector quantization, projection

1 Introduction

Data mining is an emerging area of new research efforts, responding to the presence of large databases in commerce, industry and research. It is also a title for a large number of widely divergent methods ranging from belief networks and relational learning to statistics and neural networks. Data mining is part of a larger framework, Knowledge Discovery in Databases (KDD) [4], whose purpose is to find new knowledge from databases where dimension, complexity or amount of data is prohibitively large for human observation alone. Data mining is an interactive process requiring that the intuition and background knowledge of humans are coupled with the computational efficiency of modern computer technology. For this reason, visualization is a very important part of data mining.

The Self-Organizing Map (SOM) [13] is a neural network algorithm based on unsupervised learning. It is closely related to Principal Curves [6]. The SOM has proven to be a valuable tool in data mining and KDD with applications in full-text and financial data analysis. It has also been successfully applied in various engineering applications in pattern recognition, image analysis, process monitoring and fault diagnosis [14, 27, 28]. The use of the SOM in exploratory data analysis is studied in [2, 10, 18, 23, 29, 31].

The SOM has several beneficial features which make it a useful method in data mining. It implements an ordered dimensionality-reducing mapping of the training data. The map follows the probability density function of the data and is robust to missing data. It is readily explainable, simple and — perhaps most importantly — easy to visualize. Visualization of complex multidimensional data is indeed one of the main application areas of the SOM.

In this paper, different visualization methods of the SOM are shortly introduced. The purpose is to give an idea of what kind of information can be acquired from different presentations and how the SOM can best be utilized in exploratory data visualization. Both old and new methods as well as some new ideas are presented. The new ideas include component plane reorganization (Figure 5) and match-accuracy visualization using response surfaces and sample positioning (Figures 10 and 11).

In the visualizations, a real-world data set is used. The data contain information on the pulp and paper mills of the world [31]. The data set is a difficult one: the input space dimension is 75, and even if there clearly are many clusters in the data, they are highly overlapping.

2 Methods

2.1 Self-Organizing Map (SOM)

The SOM consists of neurons located on a regular low-dimensional grid, usually 1- or 2-dimensional (1D or 2D). Higher dimensional grids are possible, but they are not generally used since their visualization is problematic. The lattice of the grid can be either hexagonal or rectangular. In this paper, the former is used because it is more pleasing to the eye.

Each neuron k is represented by an n-dimensional prototype vector $\mathbf{m}_k = [m_{k1}, \ldots, m_{kn}]$, where n is the dimension of the input space. On each training step, a data sample $\mathbf{x}$ is selected and the nearest unit $\mathbf{m}_c$ (the best-matching unit, BMU) is found from the map. The prototype vectors of the BMU and its neighbors on the grid are moved towards the sample vector:

$$\mathbf{m}_k := \mathbf{m}_k + \alpha(t) h_{ck}(t)(\mathbf{x} - \mathbf{m}_k),$$

where $\alpha(t)$ is learning rate and $h_{ck}(t)$ is a neighborhood kernel centered on the winner unit c. Both learning rate and neighborhood kernel radius decrease monotonically with time.

During the iterative training, the SOM behaves like a flexible net that folds onto the "cloud" formed by input data. The resulting prototype vectors can be interpreted as conditional averages of the data. Because neighborhood on the map grid (h_{ck}) is used to determine the magnitude of update, prototype vectors of neighboring units resemble each other. Therefore, BMUs of similar data samples are close to each other on the map grid.

2.2 Vector quantization and projection (VQ-P)

In general, vector quantization reduces the original data set to a smaller, but still representative, set to work with. At the same time it suppresses noise. Some vector quantization algorithms are listed in Table 1.

In order to visualize multidimensional vectors, a projection from the original high-dimensional input space to at most 3-dimensional output space has to be found. The projection should roughly preserve distances between the original vectors, or at least their topological ordering. Some vector projection algorithms are listed in Table 2.

In this paper, the combination of these two tasks, vector quantization and vector projection, is called *Vector Quantization-Projection (VQ-P)*. In VQ-P, a *set of prototype vectors* forms a codebook which reflects the properties of the data. Together with the projection it forms a *low-dimensional map* of the area covered by the data distribution. Each prototype vector is a *unit of this map*. The aim of visualization is both to understand the mapped area, but also to enable investigation of new data samples with respect to it. The actual visualizations consist of a number of objects — the prototypes and possibly some data samples — and the objective is to give an idea of the properties of the objects and of the relations between them.

The SOM is an example of a method that accomplishes VQ-P. Other such methods can be constructed by first making vector quantization and then a vector projection using, for example, the methods listed in Tables 1 and 2. It is important to notice that most of the visualization methods in Section 3 are not applicable only in case of the SOM, but rather in the more general case of VQ-P.

Of course, the SOM differs from a serial combination of a vector quantization and a vector projection algorithm. In the SOM algorithm, the two tasks interact, while in a serial combination projection is completely subordinate to quantization. Another difference is that the SOM has a regularly shaped projection grid. The regular shape makes it easy to compare different visualizations. It also makes implementation of user interfaces based on the SOM straightforward. On the other hand, the rigid grid has some disadvantages:

- The grid guides the vector quantization process. For example, when the data cloud is discontinuous, interpolating units are positioned between data clusters. In visualization, these may give false cues of the data shape, and should therefore be de-emphasized.
- Although the map is typically rectangular in shape, the axes of the map grid rarely have any clear interpretation.
- The projection implemented by the SOM alone is very crude. The projection of a data sample is usually defined as (the location of) its BMU and it is typically very hard to tell anything about the global shape of the data using the raw map grid alone. This deficit is usually encountered by making a separate vector projection of the prototype vectors using e.g. Sammon's projection (see Section 3.1).

Ultimately the choice of VQ-P method depends on the needs of the application. If computational cost of projection and distortion caused by noise are considered negliable, the vector quantization step can be discarded altogether.

2.3 Multiple visualizations and linking

When visualizing data sets, one of the problems is that they usually contain so much information that it is impossible to show it all in a single figure or visual-

Table 1: *Some vector quantization algorithms. All listed algorithms are iterative and are based on minimizing the reconstruction error of the original data set using a certain number of cluster centers. The difference between the algorithms is the way the cluster centers are updated on each adaptation step.*

Algorithm	Notes
k-means [1]	Only best matching (closest) cluster center of the sample vector is updated.
maximum entropy [19, 25]	All cluster centers are updated according to their distance to the sample vector.
neural gas [19]	All cluster centers are updated according to their ranking order in distance to the sample vector.
SOM	The cluster centers are updated according to their distance from the BMU of the sample vector on the map grid (h_{ck}).

ization. The number of *visual dimensions* determines how many different kinds of information can be efficiently inserted into one visualization. Typical visual dimensions include position, size and color (or texture). Shape and motion are other visual dimensions, but their use is more problematic. In virtual reality environments other senses come into play as well. Information can be relayed through audio, or through touch.

Due to the limited number of visual dimensions, one has to use *multiple visualizations* instead of only one. Key issue then becomes how to link the various visualizations with each other. Which part of each visualization correspond to the same object? Where is a specific object located in the various visualizations? This *linking* is very important. The link between objects should be obvious to the observer.

In the case of SOM, linking is usually done using position, which can be either similar or identical. In the former case, each object is in the same position with regard to the center of individual visualizations, as in Figure 3(a). In the latter case, visualizations are positioned on top of, or inside, each other. In Figure 10(b), each red circle corresponds to the map unit at which it is centered. Identical position can obviously be used for only a limited number of visualizations. Similar position has no such limitations, but it has limited visual accuracy. An effective new technique is to do linking using color, as in Figure 1.

3 SOM visualization

In this section, different kinds of methods for visualizing data using the SOM and other VQ-P methods (cf. Section 2.2) are presented. The methods can be divided to three categories based on the goal of the visualization. The first category is the task of getting an idea of the overall shape and possible cluster structure of the data set. The second category comprises the wide field of analysing the prototype vectors, for example characteristics of clusters and correlations between vector components. The third category is examination of new

Table 2: *Some vector projection algorithms. In the energy functions X_{ij} denotes the distance between vectors i and j in the input space, and Y_{ij} in the output space. In the case of SOM, X_{ik} denotes the distance between vector i and prototype k. The output space distance Y_{ck} is measured between the BMU of the data sample c and all other map units; h_{ck} is the neighborhood kernel. Note that in the first three algorithms input space distances X_{ij} are fixed, but in the case of SOM they change, since the SOM training algorithm moves the prototype vectors to minimize the quantization error. Optimization of the projection is only secondary goal. Note also that the mathematical treatment of the SOM has proven to be very difficult. The energy function given for the SOM holds only for the case of discrete data set and fixed neighborhood kernel, but it does give a qualitative idea of the situation in the general case.*

Algorithm	Energy function	Notes
metric Multi-Dimensional Scaling (MDS) [10, 24]	$\sum_i \sum_{j<i} (X_{ij} - Y_{ij})^2$	Distances in the input space are approximated by distances in output space (typically 2D plane).
Sammon's projection [20]	$\sum_i \sum_{j<i} \frac{(X_{ij} - Y_{ij})^2}{X_{ij}}$	Local distances in the input space are emphasized. Sammon's projection has an inherent instability: the $\frac{1}{X_{ij}}$ term when $X_{ij} \to 0$. This can be corrected by using e.g. $\frac{1}{\max(c, X_{ij})}$ in its stead, where c is a suitably selected constant.
Curvilinear Component Analysis (CCA) [3]	$\sum_i \sum_{j<i} (X_{ij} - Y_{ij})^2 f(Y_{ij})$	Local distances in the output space are emphasized. $f(Y_{ij})$ is a function monotonically decreasing with output space distance.
SOM	$\sum_i \sum_k X_{ik}^2 h_{ck}(Y_{ck})$	Input space distances X_{ik} are measured between prototypes (k) and data vectors (i) rather than between all pairs of data vectors. Similarly input space distances are measured between map units (Y_{ck}).

data samples for classification and novelty detection purposes.

3.1 Structure and shape

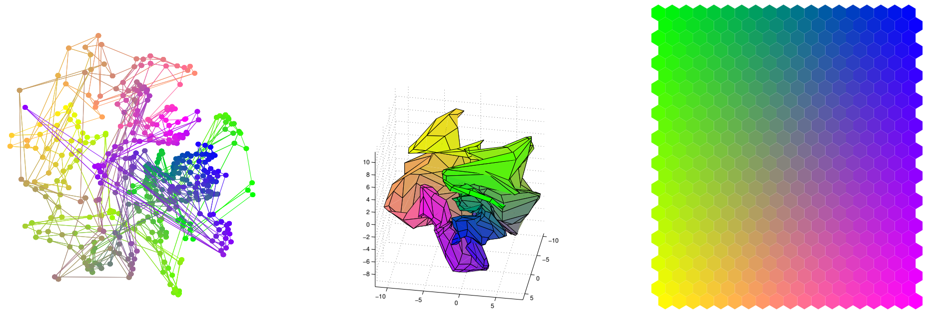
The prototype vectors form a "data cloud" in the input space. The structure of this cloud is very interesting in terms of understanding the data. What, and where, are the *clusters and outliers*? What is the *shape* of the data cloud? Data projection methods are very useful in answering these questions (see Figure 1). An efficient approach to show clusters on the SOM is to use distance matrix techniques (see Figure 2). Both approaches are discussed below.

Projection methods. Some projection methods are listed in Table 2, but there are many more, see for example [15, 18, 26]. The various projection methods have different strengths. Linear methods, like PCA [1], are computationally light. Iterative nonlinear methods, like Sammon's projection and CCA, are able to handle nonlinear structures in data, but they are computationally intensive, which is why vector quantization is useful when they are utilized. Also, Sammon's projection and CCA emphasize local distances over global ones and are therefore excellent for showing the (local) shape of the data set. Some projection methods, e.g. Linear Discriminant Analysis (LDA) [24], are only applicable if the class memberships of the samples are known.

New interesting projection methods have been proposed by Modha *et al.* [22] and König [16, 15]. Modha proposes e.g. a nonlinear cluster projection algorithm which provides a way to study data sets "through the eyes of the k-means". The method is very simple: two cluster centroids are chosen and the distances to these centroids provide x- and y-coordinates for each data vector. What makes the methods really useful is that Modha also proposes techniques to move smoothly between such projections, thus linking several visualizations together using motion. In [15] König gives a survey of existing and new unsupervised projection techniques with special attention to their computational complexity and structure preservation.

Projections of the SOM. The projection implemented by the SOM is restricted to the junctions of the map grid, and therefore it is very crude. To visualize the shape of the SOM in the input space, the prototype vectors of the map are typically projected separately using e.g. Sammon's projection. A problem is how to link the projected map to other visualizations: SOM visualizations are usually linked using position, but obviously this cannot be used here. The problem can be resolved using color coding of the map grid as in Figure 1.

Projection visualizations can be further enhanced by connecting some selected objects to each other with lines. In principle objects near each other in the projection are also close to each other in the input space. Due to the low output dimension, this doesn't always hold, not even the other way around. Therefore, it is advantageous to complement the position information by connecting objects that are, for example, close to each other in the input space. Particularly useful this is in 3D if proper depth-cues cannot be shown: lines connecting the objects can give the visualization a shape that single objects hanging in space fail to convey. Visualizing the SOM by projecting the prototype vectors to a low dimension and connecting each unit to its neighbors gives the map its characteristic net-like look and makes it very easy to grasp its shape.



(a) 2D projection (b) 3D projection (c) Color coding

Figure 1: *The overall shape of the data cloud can be visualized by making a projection of the prototype vectors to a lower dimension. In this case, two projections of the SOM trained on the case data have been made, one to 2D (a) and one to 3D (b). In Figure (a) each dot corresponds to one map unit, the color of which has been taken from the color coding in Figure (c). Each map unit has been connected to its neighbors with lines. From Figure (a) several clusters can be seen as concentrations of dots. It is also apparent that the 2D projection is inadequate for the task, since the projection has folded badly on itself. On the other hand the 3D-projection in Figure (b) performs quite well. From both projections it can be seen that for example the bottom right and top right corners of the map are actually rather close to each other in the input space.*

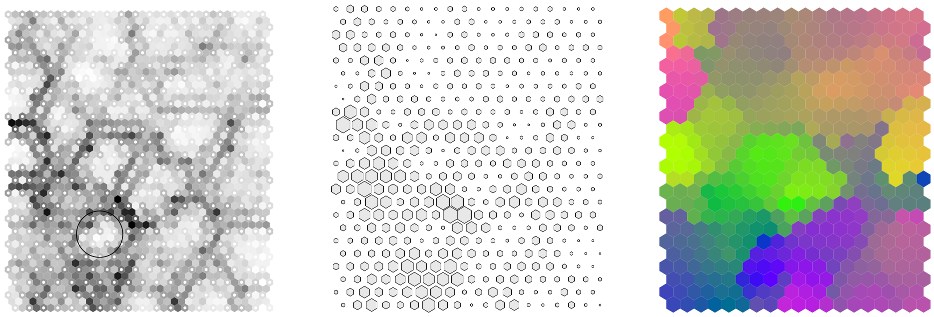
Projections in 3D. In projection tasks 3D gives a huge advantage when compared to conventional 2D-projections. This is easy to understand since position in 3D is much more flexible, and thus more powerful, indicator than position in 2D. For the SOM, additional benefits may result from the fact that the SOM initially projects the data on a 2D grid: the ability to position the map units in 3D may allow correction of some of the errors induced by the rigid grid structure (cf. Figures 1(a) and (b)).

Unfortunately, to be even of slightest value, 3D visualizations must be viewed from multiple viewpoints, or be totally interactive so that they can be rotated at will. Of course, this interactivity is a demanding requirement. However, most mathematical software packages, like Matlab and Mathematica, provide efficient and easy to use 3D visualization. Using Virtual Reality Modeling Language (VRML) enables a much wider audience to access such visualizations.

Distance matrices. While projections give a rough idea of clusters in the data, a different technique is typically used to show the clusters on SOM: distance matrices. The most widely used is the U-matrix [30]: distances of each map unit to each of its immediate neighbors are calculated and visualized using gray shade [9]. An example of a U-matrix is shown in Figure 2(a). Alternatively, the distances can be visualized using the shape and size of each map unit [5], or the appearance of the lines connecting the map units [21]. A simplified approach is to calculate a single value for each map unit - minimum, median or maximum of the distances to neighbors - and use them to control color or size [17]. See for example Figures 2(b) and 6(b).

Distance matrix methods show borders between clusters. Another approach is to visualize the similarity of map units. This can be done by giving a color to each map unit so that similar map units get similar hues [11]. An example of this, using the approach in [7], is shown in Figure 2(c). The hues have been selected by spreading a smooth colormap on the 2D projection of the map in Figure 1(a).

The advantage of these methods over vector projections is that they do not use position to indicate clusters. Thus, the position can be used for linking the figures to other visualizations, and some other visual dimension can be used to show the cluster information. For example in Figure 6(b), hexagon position indicates the individual map unit, size gives distance matrix information and color gives the color coding.



(a) U-matrix (b) Distance matrix (c) Similarity coloring

Figure 2: *The cluster structure of the SOM is typically visualized using distance matrix techniques. Figure (a) shows the U-matrix visualization. The white dots indicate locations of map units and hexagons between them show the actual values of the U-matrix. The gray shade of the hexagon denotes distance to the neighboring map unit. The darker the shade, the bigger the distance. Thus, clusters in the visualization can be seen as light areas with dark borders. There are several such clusters on the U-matrix. With further examination the characteristics of these clusters can be determined. For example, the area circled on the U-matrix could be named as "Groundwood" cluster (cf. Figure 7). Figure (b) gives essentially the same information as Figure (a). It is an averaged version of the U-matrix: the size of each map unit is proportional to its average distance to its neighbors. Figure (c) illustrates the similarity visualization of the map: areas with similar hues are close to each other in the input space.*

3.2 Properties of the prototype vector set

After getting an idea of the overall cluster characteristics of the data, the actual prototype vectors can be studied more closely. In this, component planes play the key role. Each component plane consists of the values of a single vector component in all map units. Examples of component planes are shown in Figures 3(a) and 5. Note that while in these the grid coordinates are used to position the map units, coordinates originating from some projection (cf. Section 3.1) could be used as well. In that case, however, the advantage of the regular shape of the SOM grid is lost.

Spread of values. Simple inspection of a component plane provides an idea of the spread of values of that component. In Figure 3(a), two component planes (total annual paper and pulp productions on a mill) are shown. Corresponding histograms calculated from the original data and from the prototype set are shown in Figure 3(b) (top and bottom, respectively). It can be seen that the prototype vectors cover well the typical data values. They do not cover the very high values, but this is because for these components the extremely high values are very rare and can be considered outliers: for both components less than 3% of the values in the original data are bigger than the maximum value in the prototype set. What's more is that from the component plane visualizations, one can see that for the total pulp production the high values correspond to two different clusters. This kind of information cannot be inferred from the component histogram.

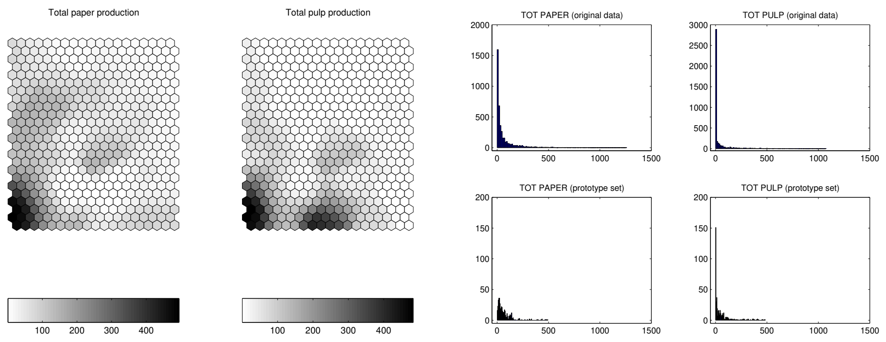
Cluster properties. A very interesting property of a cluster is, of course, what *makes* it a cluster. Methods for visualizing the contribution of the original variables to the cluster structure have been developed by Kaski *et al.* [12]. The importance can be simply measured as the contribution of the observed component (k) to the total distance between two map units:

$$|m_{ik} - m_{jk}| / ||\mathbf{m}_i - \mathbf{m}_j||,$$

where i and j are two neighboring map units, $\mathbf{m}_i$ is a prototype vector and m_{ik} is the observed component. One such contribution, averaged over the neighborhood of each unit, is shown in Figure 4. A measure of the component importance to a particular cluster can be acquired e.g. by observing a chosen cluster and its immediate surroundings. The importance is indicated by the correlation between cluster structure in that area revealed by all variables together (i.e. some distance matrix) and by each component alone.

Correlations. The component planes can also be used for correlation hunting. Correlations between component pairs are revealed as similar patterns in identical positions of the component planes. Pattern matching is something that the human eye is very good at, and it is further enhanced by the regular shape of the map grid. See for example Figure 3(a).

The hunting can be made even easier if the component planes are reorganized so that possibly correlated ones are presented near each other as in Figure 5(b). To do this, the covariance matrix between all component planes is calculated. Each row of this matrix is a data vector corresponding to one component plane.



(a) Component planes

(b) Histograms

Figure 3: *In Figure (a), there are two component planes: total annual paper (on the left) and pulp production (on the right) on a mill. Both components have mainly low values and increasingly smaller number of high values. It can be seen that for annual pulp production, high values correspond to two different clusters: on bottom middle, and on bottom left corner. These clusters can also be seen from the U-matrix visualization in Figure 2(a). Figure (b) shows histograms of the components in the training data (top two) and in the prototype vector set (bottom two). The values in the prototype vector set cover the typical values of the original data set, but not the rarely occuring extremely high values, which can be considered outliers.*

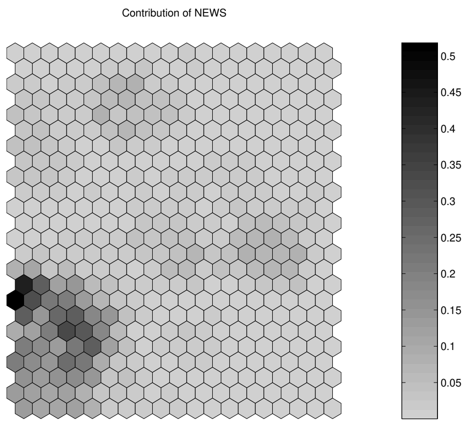
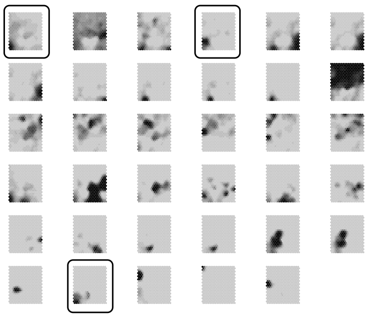
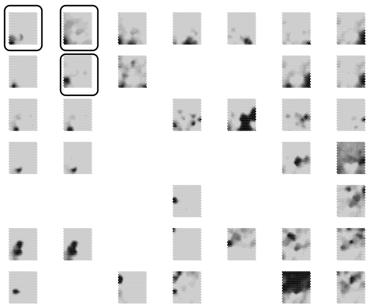


Figure 4: *Contribution of news paper production capacity on the cluster structure of the map. It can be seen that the component has greatest effect on the bottom left corner, which can then be named as the "Newspaper" cluster.*

This data is used to train a SOM with a rectangular lattice. Each component plane is then assigned a place corresponding to the BMU of the respective row of the covariance matrix. To prevent several components from being assigned to the same place, a simple procedure is applied: for any unit with multiple components, the worst matching of them is moved to its next-best-matching unit. This is repeated until no two component planes are in the same location.



(a) Component planes



(b) Reorganized planes

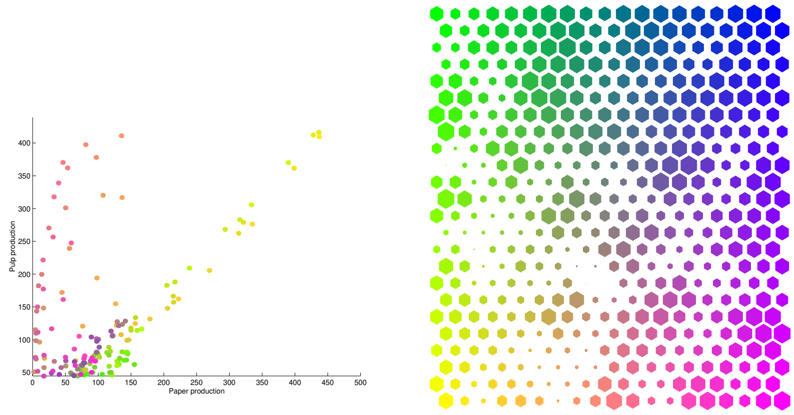
Figure 5: *Correlations between components can be hunted from the component planes visualization in Figure (a) (only the first 35 components of the data set are shown). The task is easier if the planes are reorganized so that component planes which seem to have high correlation are placed near each other, as in Figure (b). For example, this reorganization shows nicely a connection between high total paper production, newspaper production and thermomechanical pulp. These components have been indicated on Figure (a) and (b) by framing them.*

Using component planes in correlation hunting is easy, but also rather vague. However, it is easy to select interesting component combinations for further investigation. A more detailed study of interesting combinations can be done using e.g. scatter plots as proposed by Himberg [7]. Since correlations may differ in different areas of the map, it is useful to link such plots to other visualizations using color. This makes it possible to identify partial correlations with clusters on the map. From the plot in Figure 6, one can easily see how two vector components behave as a function of each other in different parts of the map.

However, it is important to realize that VQ-P algorithms are designed to detect clusters, not correlations between components. The perceived (partial) correlations may actually be artifacts induced by the cluster structure of the map, incorrect folds on the map or by the limited visual accuracy of the linking method (similar position).

3.3 Examining new data with the map

After the properties of the prototype vectors have been examined, new data can be considered. The term "data" can refer to whole data sets as well as to single data samples. This third visualization category tries answer to questions like:



(a) Scatter plot

(b) Color map

Figure 6: *Correlations between vector components can be efficiently visualized using scatter plots. In Figure (a) each dot corresponds to one map unit. The x- and y-coordinates of the dots have been taken from two components of the prototype vectors: total annual paper and pulp production, respectively. To link the scatter plot to other visualizations, each dot is given a color according to the color coding of the map units shown in Figure (b). In addition to color coding, Figure (b) also uses size to indicate clusters on the map: small units correspond to cluster borders. It can be seen that for most units, especially those with yellow color coding, the two components are linearly correlated (indicating that the mills produce their own pulp), but that there are distinct exceptions. There are a number of units for which pulp production is zero. Conversely, the units with orange color have low paper production compared to their pulp production. These mills sell most of their pulp to other mills.*

- Which part of the mapped distribution best corresponds to given data? Or, where on the map are the data samples located?
- How accurate is that correspondence/localization? Does the data belong to the mapped distribution at all?
- How similar are two data vectors (or data sets) in terms of the map?

To answer these questions, the *response* of the prototype set to the data sample has to be quantified and visualized.

Data histograms. The responses are typically based on (Euclidean) distances between data sample and each prototype vector. The nearest prototype vector is the BMU of that sample. The traditional way to show the response has been to simply show the BMU on the map. For multiple vectors, a data histogram is obtained.

There are several ways to visualize this histogram, for example those in Figure 7. The histograms can be used to compare data sets with respect to each other and the mapped area. See for example Figure 8, where two data histograms, corresponding to the pulp and paper mills in the whole world and in Scandinavia, are shown.

The problem with simply pointing out the BMU is that this gives no information of the accuracy of the match. Typically there are several units with almost as good match as the BMU. If the map folds heavily these matches may be located at different parts of the map. Also, no information is given whether the sample is close to the mapped distribution or very far from it.

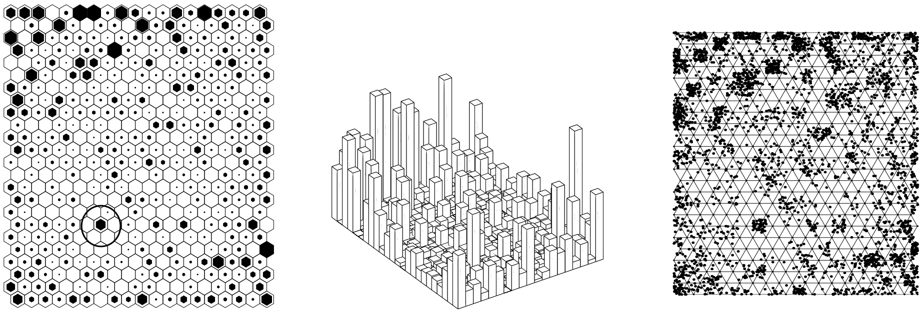
Response surface. Instead of simply pointing out the BMU, the response of all vectors can be shown. The response surface shows the relative goodness of each map unit in representing the data. A simple approach is to use the quantization error $q_i = ||\mathbf{x} - \mathbf{m}_i||$ as the indicator of goodness:

$$g(q_i) = \frac{1}{1 + q_i^2}.$$

Notice that this goodness function has several beneficial properties: $g(0) = 1$, $g(q \to \infty) = 0$, and $g'(q)$ is monotonically decreasing. The result of the last property is that good and poor matches are easy to distinguish: good matches produce sharp spots on the map, while poor matches produce an even response over the whole map. Examples of response surfaces are shown in Figure 9.

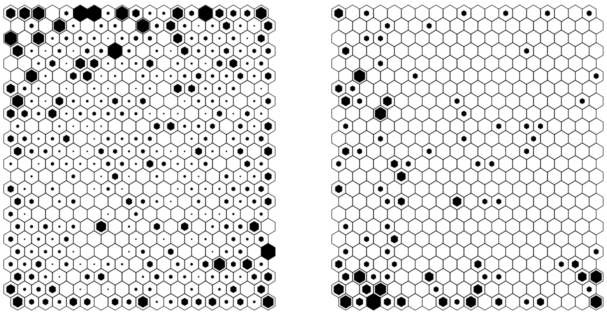
To remove the effect of scale the responses should be examined in the context of the accuracy of the quantization. The accuracy can be measured e.g. as the average quantization error (average distance between each sample and its BMU). If a data independent measure is desirable, the accuracy can be defined as the average distance of each prototype vector from its neighbors. The accuracy a is taken into account by scaling the quantization errors with the accuracy $g(q_i/a)$. This way the responses are comparable even between different maps.

It is tempting to interpret the fuzzy responses as probabilities for a data vector to belong to a certain map unit. However, if this is desired, it is better to use the SOM as a basis for an additive mixture model, and estimate the probabilities through a kernel density estimate, see for example [8].



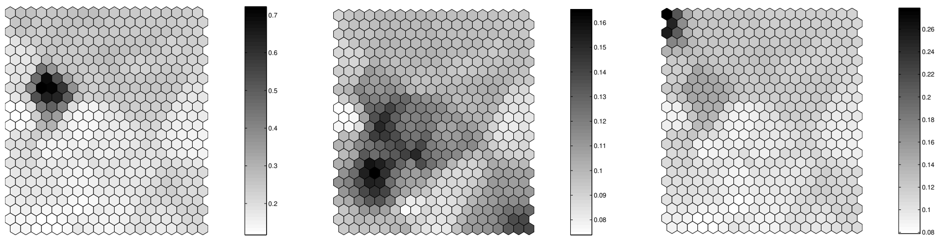
(a) 2D (b) 3D (c) Scatter

Figure 7: *Different ways to visualize data histograms. In Figure (a), the size of the black hexagon is proportional to the value of the data histogram in the corresponding unit. In Figure (b), the height of the bar tells the same. Notice that the 3D-representation is much less informative because things in the front hide things in the back. In Figure (c), each dot represents one data sample. The sample marker has been placed on top of the correspoding BMU with a small random offset to make individual samples distinguishable. All histograms were calculated using the whole training data. It can be seen that there are a number of empty, or interpolating units. There are also a number of concentrations indicating that there are some mills that are very similar to each other. For example the concentration of mills indicated by a circle in Figure (a) contains only mills that produce pulp from groundwood. This cluster is also apparent from the distance matrices in Figure 2(a) and (b).*



(a) All mills (b) Scandinavian mills

Figure 8: *Figure (a) shows the data histogram for all the pulp and paper mills of the world. It is markedly different from the data histogram in Figure (b), which shows the data histogram for Scandinavian mills. Perhaps the most important distinguishing factor of Scandinavian mills is their size: the histogram in Figure (b) corresponds pretty well to the total pulp and paper production component planes (see Figure 3(a)).*



(a) Good match

(b) Poor match

(c) Average match for a set of data

Figure 9: *Response surfaces. In each figure black corresponds to the best response and white to the worst. In a poor match, there are wide areas on the map which correspond to the data vector almost as well as the BMU (large dark areas). Figures (a) and (b) show good and poor matches, respectively. From Figure (a) a clear cluster can also be seen, corresponding to mills producing cartonboard. Figure (c) shows the average response of the data vectors whose BMU is the unit on top left corner of the map. From this figure it can be seen that there's a fold on the map, as the cartonboard area of Figure (a) has a higher response to the set of data vectors than the immediate surroundings of the corner.*

Information from position. Another approach is to position the samples in the visualization so that the accuracy is apparent from either the size or the position of the sample marker. The former approach is used in Figure 10 and the latter in Figure 11.

In Figure 10, the position of the sample marker shows the BMU and its size (height or diameter) the quantization error. The size of the marker has been scaled so that it has a meaningful interpretation. In Figure 10(a), a kernel density estimator has been implemented based on the map. The upper grid marks the 50%-probability boundary of the Gaussian kernel associated with each of the map units. Each bar represents one data sample, and the higher the bar, the less probable such a sample is. In Figure 10(b), the interpretation is more straightforward: the diameter of the circle is scaled by the average distance of each map unit to its neighbors (ie. by distance matrix value at that map unit). If the circle is smaller than the hexagon, the BMU is closer to the data sample than to its neighbors (on the average).

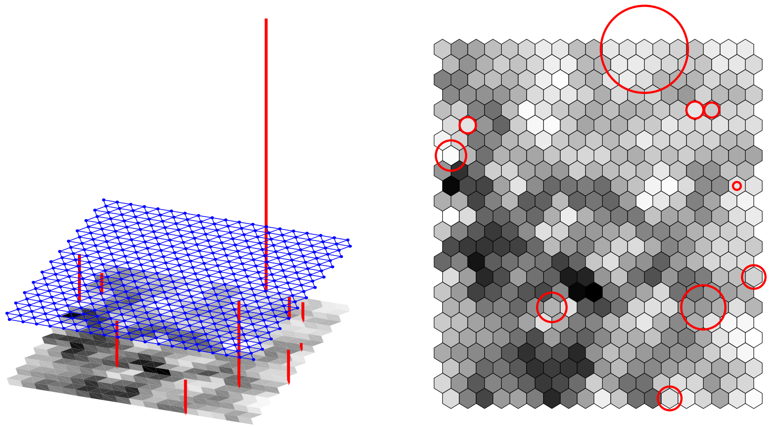
In Figure 11, the position of the sample marker shows the match accuracy: the closer the sample marker is to the map, the more accurate is the match. First, a 2D position for each of the prototype vectors has been designated: CCA-projection in Figure 11(a), unit coordinates on the map grid in Figure 11(b). Then, the position of the marker has been optimized relative to its three best-matching prototype vectors using a CCA-like projection algorithm:

`repeat`

$$\mathbf{dx}' = -\sum_{i=1}^{3}\left(\frac{||\mathbf{x}-\mathbf{m}_i||}{||\mathbf{x}'-\mathbf{m}_i'||} - 1\right)(\mathbf{x}' - \mathbf{m}_i')$$

$$\mathbf{x}' := \mathbf{x}' + 0.1\mathbf{dx}'$$

`while` $steps < maxsteps$ `AND` $||dx'|| > \epsilon$, where $\mathbf{x}$ is the data vector, $\mathbf{m}_i$ are the prototype vectors and $\mathbf{x}', \mathbf{m}'_i$ are their projections in the output space.



(a) Bar

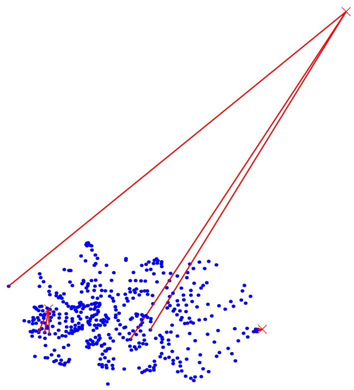
(b) Circle

Figure 10: *Quantization error plots. The BMUs of a set of data vectors are indicated by the markers: the point of the bar in Figure (a) and the center of the circle in Figure (b). The size of the markers is inversely proportional to the accuracy of the match. The bottom texture in both figures is the distance matrix of the map. It can be seen that while most of the samples in the set fit relatively well to the map, there is one that is clearly an anomaly.*

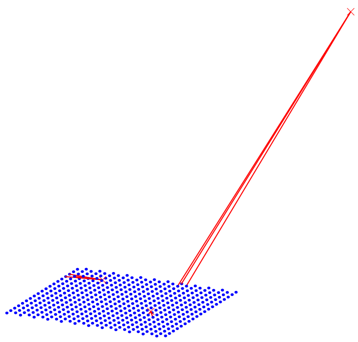
4 Conclusion

The major point of this paper is to bring together the many visualization methods for the Self-Organizing Map and thus bring forth the power of the SOM, and VQ-P methods in general, in visualization of multidimensional numerical data. The SOM combines vector quantization and projection which together provide a map of the data giving a visual insight to the properties of the data: the shape of the data cloud, clusters, their properties and correlations between vector components. Comparing new data with the map helps in classifying the data and gives an indication of whether the new data belongs to the same data distribution as the map was trained with. In the general framework of knowledge discovery, the methods presented in this paper are especially useful in the exploratory phase of knowledge discovery task.

Software. The existence of all these visualization methods does little good without a good implementation in a flexible and user-friendly data mining environment. Most of the visualization methods discussed in this paper have been implemented in *SOM Toolbox*, a software package for Matlab 5 computing environment by Mathworks, Inc. (`http://www.mathworks.com/`). SOM Toolbox is



(a) CCA



(b) Grid

Figure 11: *Three data samples (with low, medium and very high quantization errors) have been positioned in 3D on the 2D-projection of the map. The samples are denoted by red crosses and map units by blue dots. In Figure (a), CCA has been used for the initial 2D-projection. In Figure (b), the map grid has been used. The position of the samples has been optimized using an iterative CCA-like projection procedure using the 2D-positions of the three BMUs of each data sample. The projected samples have been connected to the their BMUs with lines. Further analysis reveals what makes the third sample so strange (in terms of the map): it is one of the four mills (out of over 4000) producing semibleached sulphite pulp, and the only one that concentrates only on that kind of pulp. It can well be considered an outlier.*

available for free from `http://www.cis.hut.fi/projects/somtoolbox/`. SOM Toolbox was primarily constructed as an efficient implementation of the SOM algorithm in Matlab environment, but special attention has been given to its visualization. All figures in this paper have been done using Matlab and the SOM Toolbox.

Future work. Vector quantization divides the data space into subareas corresponding to clusters, special cases and outliers. Using the qualitative information given by visualizations one can select subareas and subspaces from the data and perform quantitative analysis on these rather than blindly on the whole data. Some areas may be discarded as outliers, some variables may be left out or preprocessed in a more appropriate manner. Thus filtered, the data can be reanalysed.

To make this cycle as efficient as possible, one needs automatic postprocessing methods which provide relevant and easily comprehensible information about the map and its subareas. Examples of such postprocessing methods include quantifying the importance of different variables in each subarea [12], linguistic interpretation of the vector quantization [23], as well as simple statistical descriptors. However, the foremost among such methods would be a (semi)automated cluster analysis to help with selecting the subareas themselves.

Ideally, one would feed in a data set and the data mining application would tell what's in it. Using the information given by the application — qualitative as well as quantitative — one could select interesting variables or subareas, preprocess them appropriately and re-enter the analysis stage. This cyclic process would enable the knowledge engineer to "play" with the data and gain ever keener insights about its properties.

Acknowledgments

This work has been carried out within the technology program "Adaptive and Intelligent Systems Applications" of Technology Development Center of Finland (TEKES), and the EU financed Brite/Euram project "Application of Neural Network Based Models for Optimization of the Rolling Process" (NEUROLL). The cooperation of Jaakko Pöyry Consulting is gratefully acknowledged.

References

[1] C. M. Bishop. *Neural Networks for Pattern recognition.* Oxford University Press, 1995.

[2] G. Deboeck and T. Kohonen. *Visual explorations in Finance using Self-Organizing Maps.* Springer-Verlag, London, 1998.

[3] P. Demartines and J. Hérault. Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets. *IEEE Transactions on Neural Networks*, 8:148 – 154, 1997.

[4] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining.* AAAI Press / The MIT Press, California, 1996.

[5] E. Häkkinen and P. Koikkalainen. The neural data analysis environment. In *Proc. of the Workshop on Self-Organizing Map*, pages 69–74, 1997.

[6] T. Hastie and W. Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84:502–516, 1989.

[7] J. Himberg. Enhancing the som based data visualization by linking different data projections. In *Proceedings of International Symposium on Intelligent Data Engineering and Learning (IDEAL)*, pages 427–434, Hong Kong, October 1998.

[8] L. Holmström and A. Hämäläinen. The self-organizing reduced kernel density estimator. In *Proc. of the 1993 IEEE International Conference on Neural Networks*, pages 417–421, San Francisco, California, March 1993.

[9] J. Iivarinen, T. Kohonen, J. Kangas, and S. Kaski. Visualizing the clusters on the self-organizing map. In Christer Carlsson, Timo Järvi, and Tapio Reponen, editors, *Proc. Conf. on Artificial Intelligence Res. in Finland*, number 12 in Conf. Proc. of Finnish Artificial Intelligence Society, pages 122–126, Helsinki, Finland, 1994. Finnish Artificial Intelligence Society.

[10] S. Kaski. *Data Exploration Using Self-Organizing Maps.* PhD thesis, Helsinki University of Technology, 1997.

[11] S. Kaski, T. Kohonen, and J. Venna. *Visual explorations in Finance using Self-Organizing Maps*, chapter Tips for SOM Processing and Colorcoding of Maps. Springer-Verlag, London, 1998.

[12] S. Kaski, J. Nikkilä, and T. Kohonen. Methods for interpreting a self-organized map in data analysis. In *Proc. of European Symposium on Artificial Neural Networks (ESANN)*, Bruges, Belgium, 1998.

[13] T. Kohonen. *Self-Organizing Maps.* Springer, Berlin, Heidelberg, 1995.

[14] T. Kohonen, E. Oja, O. Simula, A. Visa, and J. Kangas. Engineering applications of the self-organizing map. *Proceedings of the IEEE*, 84(10):27 pages, October 1996.

[15] A. König. A survey of methods for multivariate data projection, visualization and interactive analysis. In *Proc. of the 5th International Conference on Soft Computing and Information/Intelligent Systems (IIZUKA'98)*, pages 55–59, October 1998.

[16] A. König, O. Bulmahn, and M. Glesner. Systematic methods for multivariate data visualization and numerical assessment of class separabilityand overlap in automated visual industrial control. In *Proc. of the 5th British Machine Vision Conference (BMVC'94)*, pages 195–204, September 1994.

[17] M. A. Kraaijveld, J. Mao, and A. K. Jain. A nonlinear projection method based on kohonen's topology preserving maps. *IEEE Transactions on Neural Networks*, 6(3):548–559, May 1995.

[18] J. Mao and A.K. Jain. Artificial neural networks for feature extraction and multivariate data projection. *IEEE Transaction on Neural Networks*, 6(2):296–317, March 1995.

[19] T. M. Martinetz, S. G. Berkovich, and K. J. Schulten. "neural-gas" network for vector quantization and its application to time-series prediction. *IEEE Transaction on Neural Networks*, 4(4):558–569, 1993.

[20] J. W. Sammon, Jr. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, C-18(5):401–409, May 1969.

[21] D. Merkl and A. Rauber. Alternative ways for cluster visualization in self-organizing maps. In *Proc. of the Workshop on Self-Organizing Map*, pages 106–111, 1997.

[22] D. S. Modha, S. Spangler, and S. Vaithyanathan. Multidimensional cluster visualization using guided tours. Technical Report Research Report RJ 10124, IBM Almaden Research Center, San Jose, CA, June 17 1998. http://www.alphaWorks.ibm.com/formula/CViz.

[23] W. Pedrycz and H. C. Card. Linguistic interpretation of self-organizing maps. In *Proc. of International Conference on Fuzzy Systems '92*, 1992.

[24] B. D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, 1996.

[25] K. Rose, F. Gurewitz, and G. Fox. Statistical mechanics and phase transitions in clustering. *Physical Rev. Lett.*, 65(8):945–948, 1990.

[26] W. Siedlecki, K. Siedlecka, and J. Sklansky. An overview of mapping techniques for exploratory pattern analysis. *Pattern Recognition*, 21(5):411–429, 1988.

[27] O. Simula and J. Kangas. *Neural Networks for Chemical Engineers*, volume 6 of *Computer-Aided Chemical Engineering*, chapter 14, Process monitoring and visualization using self-organizing maps. Elsevier, Amsterdam, 1995.

[28] V. Tryba and K. Goser. Self-Organizing Feature Maps for process control in chemistry. In T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, editors, *Artificial Neural Networks*, pages 847–852, Amsterdam, Netherlands, 1991.

[29] A. Ultsch. Self organized feature maps for monitoring and knowledge acquisition of a chemical process. In S. Gielen and B. Kappen, editors, *Proc. ICANN'93, Int. Conf. on Artificial Neural Networks*, pages 864–867, London, UK, 1993. Springer.

[30] A. Ultsch and H.P. Siemon. Kohonen's self organizing feature maps for exploratory data analysis. In *Proc. INNC'90, Int. Neural Network Conf.*, pages 305–308, Dordrecht, Netherlands, 1990. Kluwer.

[31] J. Vesanto. Data mining techniques based on the self-organizing map. Master's thesis, Helsinki University of Technology, June 1997. http://www.cis.hut.fi/projects/ide/publications/.