ViSOM—A Novel Method for Multivariate Data Projection and Structure Visualization

Hujun Yin

***Abstract*—When used for visualization of high-dimensional data, the self-organizing map (SOM) requires a coloring scheme such as the U-matrix to mark the distances between neurons. Even so, the structures of the data clusters may not be apparent and their shapes are often distorted. In this paper, a visualization-induced SOM (ViSOM) is proposed to overcome these shortcomings. The algorithm constrains and regularizes the inter-neuron distance with a parameter that controls the resolution of the map. The mapping preserves the inter-point distances of the input data on the map as well as the topology. It produces a graded mesh in the data space such that the distances between mapped data points on the map resemble those in the original space, like in the Sammon mapping. However, unlike the Sammon mapping, the ViSOM can accommodate both training data and new arrivals and is much simpler in computational complexity. Several experimental results and comparisons with other methods are presented.**

***Index Terms*—Dimension reduction, multidimensional scaling, multivariate data visualization, nonlinear mapping, self-organizing maps (SOMs).**

I. Introduction

SAMMON [1] pioneered the nonlinear mapping algorithm for visualization of multivariate data, when the linear principal component analysis (PCA) was the primary tool for dimension reduction. The objective of the Sammon mapping is to minimize the differences between interpattern distances in the original space and interpattern distances in the projected space. The projection of data from an invisible high-dimensional space to a low perceptible one (usually two dimensions) can reveal the data structures and cluster tendency. Sammon mapping has been shown to be superior to PCA for data structure analysis [1]. However, the Sammon algorithm is a point-to-point mapping, which does not provide the explicit mapping function and cannot accommodate new data points [1], [2]. For any additional data, the projection has to be recalculated from scratch based on all data points. This proves difficult or even impossible for many practical applications where data arrives sequentially, the quantity of data is large, and/or memory space for the data is limited. The Sammon mapping does not have any generalization ability [2].

Neural networks present another approach to nonlinear data analysis. They are biologically inspired learning and mapping methods and can learn complex nonlinear relationships of variables in a system from sample data. Mao and Jain have given a comprehensive overview on this subject [2]. Kohonen's SOM is an abstract mathematical model of the mapping between nerve sensory (especially retina) and cerebral cortex (especially visual cortex) [3], [4]. As the inputs to the SOM are often drawn from a high-dimensional space, the algorithm has been used as a visualization tool for dimensional reduction (e.g., [5], [6]). One of greatest properties of the SOM is that it is a topology preserving mapping, i.e., close points in the input space will be mapped to nearby neurons in the map space. Such properties can be employed in visualizing *relative* or *qualitative* mutual relationships among the input. The SOM is also an abstraction process and it usually uses fewer representatives for often a large number of data points. Its distribution and convergence properties show that the SOM, when the neighborhood diminishes, is naturally an optimal vector quantizer (VQ) in minimizing the mean-square error between reference vectors and data points they represent [7]–[9]. The algorithm has found a wide range of applications in VQ, pattern classification, data mining and visualization, knowledge discovery, and information retrieval.

When the SOM is used for visualization, however, as the inter-neuron distances are not directly visible or measurable on the map, one has to use a coloring scheme such as the U-matrix [5], [6], or interpolation [10], to mark relative distances between the weights of neighboring neurons referred in the input space. Even so, the structures of data clusters may not be apparent and often appear distorted. For example, if a square SOM is used to visualize data drawn from two separate two-dimensional (2-D) spherical Gaussians, then one half of the lattice will represent one of the Gaussians, in rectangular, triangle, or other shape but never in spherical. The other half will be similar. If we had not known the distributions of the data, it would be impossible to depict them from the learned map. The SOM does not *directly* apply to multidimensional scaling, which aims to reproduce proximity in (Euclidean) distance on a low-dimensional visualization space [11], [12].

In this paper, a new algorithm, namely the visualization induced self-organizing map (ViSOM), is proposed. The ViSOM projects the high-dimensional data in an unsupervised manner as does the SOM, but constrains the lateral contraction force and hence regularizes the inter-neuron distance to a parameter that defines and controls the resolution of the map. It preserves the data structure as well as the topology as faithfully as possible. The ViSOM is a nonlinear projection but of a simple computational structure. The analysis and experimental results show that the ViSOM may offer attractive advantages over the commonly used SOM, PCA, and Sammon mapping. These key projection algorithms are briefly reviewed in the following section for the purpose of structure visualization. The details of the ViSOM are then presented in Section III, followed by illustrative examples demonstrating the principle of the algorithm and comparisons with other methods. Conclusions are given in Section V.

Manuscript received June 8, 2000; revised March 21, 2001.

The author is with the Department of Electrical Engineering and Electronics, University of Manchester Institute of Science and Technology, Manchester M60 1QD, UK.

Publisher Item Identifier S 1045-9227(02)00350-8.

II. Data Projection Methods

Searching for better and suitable data projection methods has always been an integral objective of pattern recognition and data analysis. Such a method will enable us to observe and detect underlying data distributions, patterns, and structures. A great deal of effort has been devoted to this subject and a number of useful methods have been proposed and consequently applied to various applications. In the following, we give a brief review of some key methods and algorithms currently used in data visualization and analysis.

A. PCA

PCA is a classic linear data analysis method aiming at finding orthogonal principal directions from a set of data, along which the data exhibit the largest variances. By discarding the minor components, the PCA can effectively reduce data variables and display the dominant ones in a linear low-dimensional subspace. It is the optimal linear projection in the sense of the mean-square error between original points and projected ones, i.e.,

$$\min \sum_{\mathbf{x}} \left[\mathbf{x} - \sum_{j=1}^{m} (\mathbf{q}_j^T \mathbf{x}) \mathbf{q}_j \right]^2 \tag{1}$$

where $\mathbf{x} = [x_1, x_2, \ldots x_n]^T$ is the n-dimensional input vector, $\{\mathbf{q}_j, j = 1, 2, \ldots m, m \leq n\}$ are orthogonal vectors representing principal directions. They are the first m principal eigenvectors of the covariance matrix of the input. The second term in the above bracket is the reconstruction or projection of $\mathbf{x}$ on these eigenvectors. The term $\mathbf{q}_j^T \mathbf{x}$ represents the projection of $\mathbf{x}$ onto the jth principal dimension. Traditional methods for solving the eigenvector problem involve numerical methods. Though fairly efficient and robust, they are not usually adaptive and often require the entire data set. Several Hebbian-based learning algorithms and neural networks have been proposed for performing PCA such as Oja's subspace network [13], Sanger's generalized Hebbian algorithm [14], and Rubner and Tavan's network [15]. The limitation of linear PCA is obvious, as it cannot capture nonlinear relationships defined by higher than the second-order statistics. If the input dimension is much higher than two, the projection onto a linear principal plane will provide only limited visualization power.

The extension to nonlinear PCA, however, has not been straightforward, due to the lack of a unified mathematical structure and an efficient and reliable algorithm, in some cases due to excessive freedom in selection of representative basis functions [16]. Principal curves and principal surfaces [17], [12] were primary nonlinear extensions of PCA, but a valid algorithm is required for a good implementation. Several networks have been proposed for nonlinear PCA such as, the five-layer feedforward associative network [18], the generalized Hebbian algorithm-based method [19], and the kernel-based principal component analysis [20].

B. Sammon Mapping

A traditional subject related to dimension reduction and data projection is multidimensional scaling (MDS), which includes PCA as one of the projection methods. The MDS searches for a mapping to a low (usually two-) dimensional plot, on which inter-point distances of projected points resemble those inter-point distances in the original space [11], [12]. A general fitness function or so-called *stress* can be described as

$$S = \frac{\sum_{i,j} [d_{ij} - f(\delta_{ij})]^2}{\sum_{i,j} d_{ij}^2} \tag{2}$$

where d_{ij} represents the distance (usually Euclidean) between mapped points i and j in the new space, δ_{ij} represents the proximity or dissimilarity of points i and j in the original space. $f(.)$ is a monotonic transformation function.

There is no exact and unique procedure to find the projection. Instead, the MDS relies on an optimization algorithm to search for a configuration that gives as low stress as possible. A gradient method is commonly used for this purpose. Inevitably, various computational problems such as local minima and divergence may occur to the optimization process. The methods are also often computationally intensive. The final solution depends on the starting configuration and parameters used in the algorithm [21]. Sammon proposed a recursive learning algorithm using the Newton optimization method for the optimal configuration [1]. It converges faster than the simple gradient method, but the computational complexity is even higher. It still has the local minima and inconsistency problems. For the Sammon mapping intermediate normalization (of the original space) is used to preserve better local distributions and the transformation function is simply the (Euclidean) distance between points i and j in the original space. The Sammon stress is then expressed as

$$S_{\text{Sammon}} = \frac{1}{\sum_{i<j} d_{ij}^*} \sum_{i<j} \frac{[d_{ij}^* - d_{ij}]^2}{d_{ij}^*} \tag{3}$$

where d_{ij}^* is the distance between point i and j in the original input space.

In addition to being computationally costly and not adaptive, another major drawback of MDS methods including Sammon mapping is the lack of an explicit projection function. Thus for any new input data, the mapping has to be recalculated based on all available data. These methods become impractical for applications where data arrive consecutively and/or predictions are needed for new data points. Mao and Jain have proposed a feedforward neural network to parameterize the Sammon mapping function and a backpropagation algorithm has been derived for training of the network and minimizing the Sammon stress [2].

C. SOM

The SOM is an unsupervised learning algorithm that uses a finite grid of neurons to map or frame the input space. The SOM tends to cluster the data points, find the representatives or features, and minimize the mean-square error between features and the data points they represent

$$\min \sum_{\mathbf{x}} \sum_{c \in \Omega} \sum_{\substack{\mathbf{x} \Rightarrow c \\ k \in \Lambda}} \eta(c, k)(\mathbf{w}_c - \mathbf{x})^2 \tag{4}$$

where $\mathbf{x} \Rightarrow c$ denotes that node c or its weight $\mathbf{w}_c$ is closest to input $\mathbf{x}$. and $\eta(c, k)$ is the neighborhood function of neuron c and its neighbor k. Ω represents the map grid and Λ is the neighborhood region.

As the map is often arranged in a low-dimensional, e.g., 2-D, grid, it can be used for visualization of potentially high-dimensional data. In the SOM, a neighborhood learning is adopted to form a topological ordering among the neurons in the map. The close data points are likely to become projected to nearby nodes. Thus the map can be used to show the relative relationships among the data points. However, the SOM does not directly show the inter-neuron distances on the map. When the SOM is used for visualization, assistance from some coloring schemes such as the U-matrix is needed to imprint the inter-neuron distances so that the boundaries between clusters can be marked. Even so, the cluster structures often appear in distorted and unnatural forms. The SOM is a VQ and tries to accommodate the data points only onto a grid. There will be few or no neurons in regions of low or no data density. The entire data set, regardless of its distribution and structure, will be crammed onto the grid. The SOM can serve as a visualization map to show the relative closeness and relationships among data points and clusters. In many cases, however, a more faithful display of structure shapes and distributions of the data set is more desirable in visualization. In the next section, a new mapping algorithm based on the SOM is proposed to overcome some shortcomings of the SOM for high-dimensional projection and structure visualization.

III. ViSOM

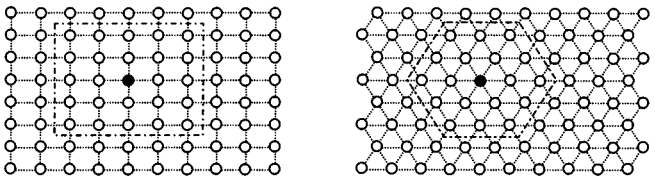
In order for the map to capture the data structure naturally and faithfully, the distance quantity must be preserved on the map, along with topology. Ideally the nodes should be uniformly and smoothly placed in the nonlinear manifold of the data space, so that the inter-neuron distances of any two neighboring neurons are approximately the same. The interneuron distances between a neuron and its far neighbors increase proportionally and regularly according to the structure of the map grid. Then the map can be seen as a smooth mesh embedded into the data space, onto which the data points are mapped and the inter-point distances are approximately preserved.

A. ViSOM Structure and Derivation

The ViSOM uses a similar grid structure of neurons as does the SOM. It usually consists of an array of nodes arranged in a low-dimensional rectangular or hexagonal grid. Fig. 1 shows an example of 2-D maps. Each node c (index $c = (i, j)$ for a 2-D map) has an associated weight vector, $w_c = [w_{c1}, w_{c2}, \ldots w_{cn}]^T$. The dashed encirclement denotes a neighborhood of a winner (marked as shaded), within which the learning takes place.

At time step t, an input $\mathbf{x}(t)$ is drawn randomly from the data set or data space. A winning neuron v can be found according to its distance to the input, i.e.,

$$v = \arg\min_{c \in \Omega} \|\mathbf{x}(t) - \mathbf{w}_c\|. \tag{5}$$



(a) (b)

Fig. 1. 8×10 maps. (a) Rectangular. (b) Hexagonal. Dotted lines indicate grid structure or neighboring relations rather than real connections. Dashed lines denote the neighborhood regions surrounding the winners shown in dark nodes.

Then the SOM updates the weight of the winning neuron according to

$$\mathbf{w}_v(t+1) = \mathbf{w}_v(t) + \alpha(t)[\mathbf{x}(t) - \mathbf{w}_v(t)] \tag{6}$$

where $\alpha(t)$ is the learning rate at time t.

In the SOM, the weights of the neurons in a neighborhood of the winner are updated by

$$\mathbf{w}_k(t+1) = \mathbf{w}_k(t) + \alpha(t)\eta(v, k, t)[\mathbf{x}(t) - \mathbf{w}_k(t)] \tag{7}$$

where $\eta(v, k, t)$ is the neighborhood function, which is monotonically decreasing with $\|v - k\|$. The second term in (7), especially the updating force $\mathbf{x}(t) - \mathbf{w}_k(t)$, can be rearranged and decomposed into two forces

$$\begin{aligned} \mathcal{F}_{kx} &\equiv \mathbf{x}(t) - \mathbf{w}_k(t) = [\mathbf{x}(t) - \mathbf{w}_v(t)] + [\mathbf{w}_v(t) - \mathbf{w}_k(t)] \\ &\equiv \mathcal{F}_{vx} + \mathcal{F}_{kv}. \end{aligned} \tag{8}$$

This can be shown in Fig. 2. The first force, $\mathcal{F}_{vx}$, represents the updating force from the winner v to the input $\mathbf{x}$, which is the same as that used by the winner in (6). It adapts the neurons toward the input in a direction that is orthogonal to the tangent plane of the winner. While the second force $\mathcal{F}_{kv}$ is a lateral force bringing neuron k to the winner v, i.e., a contraction force. It is this contraction force that brings neurons in the neighborhood toward the winner and thus forms a contraction around the winner on the map at each time step. In the ViSOM, this lateral contraction force is constrained through regularizing the distance between a neighboring neuron to the winner. If the $\mathbf{w}_k$ is far from $\mathbf{w}_v$ under a prespecified resolution, this force remains, otherwise an opposite force applies, until they are in proportion. The scale of the force is controlled by the normalized distance between these two weights.

B. ViSOM Algorithm

The details of the proposed ViSOM algorithm are given below.

1) Find the winner from (5).
2) Update the winner according to (6).
3) Update the neighborhood according to

$$\mathbf{w}_k(t+1) = \mathbf{w}_k(t) + \alpha(t)\eta(v, k, t) \left([\mathbf{x}(t) - \mathbf{w}_v(t)] + [\mathbf{w}_v(t) - \mathbf{w}_k(t)] \frac{(d_{vk} - \Delta_{vk}\lambda)}{\Delta_{vk}\lambda} \right) \tag{9}$$

where d_{vk} and Δ_{vk} are the distances between neurons v and k in the data space and on the unit grid or map, respectively, and λ is a positive prespecified resolution parameter. It represents the desired inter-neuron distance (of two neighboring nodes) reflected in the input space.

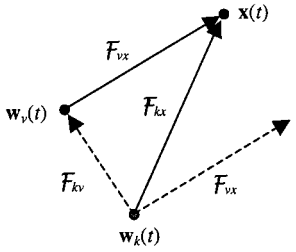


Fig. 2. Decomposition of the SOM updating force.

The smaller the value of λ, the higher resolution the map can provide.

4) Refresh the map by randomly choosing the weights of the neurons as the input at a small percentage of updating times (e.g., 10% iterations). This step can be optional.

The procedure is repeated until the map converges. Although the ViSOM has the same ability of forming topological order among neurons, the initial map does not have to be randomly ordered. This can save a lot of time in untangling the map, which is unnecessary for visualization purposes. Principal components can be used to initialize the map. For example, for a 2-D map, one can place it onto the plane crossed by the first two principal eigenvectors. A refresh phase is added to speed up the map's smooth expansion to the areas where there are few or no data points. The ViSOM is a smooth mesh grid stretching through the data space. The higher the resolution, then more nodes will be needed. Some nodes seldom win from the input. Refreshing is to keep these nodes active and also to regularize the inter-neuron distances among nodes.

The key feature of the ViSOM is that the distances between the neurons on the map (in neighborhoods) reflect the corresponding distances in the data space. That is, the distant measure is preserved on the map. When the map is trained and data points are mapped on the map, the distances between mapped data points on the map will resemble approximately those in the original space (subject to the resolution of the map). This makes visualization more direct, quantitatively measurable, and visually appealing. The size or effective range of the neighborhood function should decrease from an initially large value to a final value, which should not be just the winner. The rigidity of the map is controlled by the ultimate size, σ_f, of the neighborhood. The larger this size is the flatter will be the final map in the data space.

The resolution parameter λ depends on the size of the map, data variance and required resolution of the map. If a high resolution is required, small λ should be used. Consequently a large map is expected, depending on the data range. Although the size of the map and the resolution are usually prespecified, they can both be made adaptive during the training in order for the map to effectively cover the entire data region.

IV. Experimental Results

To demonstrate the applications of the proposed ViSOM and its distance-preserving and structure-revealing properties, several experiments and results are presented. The examples chosen are for the purpose of illustration. The results of other mapping methods such as PCA, SOM, and Sammon mapping are also presented for comparison.

A. Two Illustrative Data Sets

The first data set consists of 24 2-D points. The scatter of the points is shown in Fig. 3 as circles. An SOM and a ViSOM, both of a 50-neuron chain, were trained on this data set. The initial weights for both chains were set randomly. Both have successfully untangled the chain and formed a topology preserved mapping. After a few thousand iterations, their positions in the data space are shown in Fig. 3(a) and (b), respectively. The final neighborhood size was set to six in the ViSOM. The resolution parameter, λ, was set to 0.16. The total length that this ViSOM chain can cover is approximately $0.16 \times 50 = 8$ units. In this simple case, the refresh step was not used as the data are not too scarce compared to the number of neurons.

As can be seen, both SOM and ViSOM can place a nonlinear map into the input space, but their results differ dramatically. As the SOM is similar to a VQ, it tends to place neurons around data points regardless of the distances between them. The ViSOM, however, preserves the distances between neurons, so that the distances between mapped points on the map will resemble those in the data space. If the trained chains are used to visualize the data, then the ViSOM gives a much better presentation of data distribution and a more accurate projection of the data.

In the second data set, 100 points were randomly drawn from each of two Gaussian sources. Their mean vectors are $[1.2, 1.0]^T$ and $[0.5, 0.5]^T$, respectively, and their covariance matrices are $\left[\begin{smallmatrix} 4.0 & -0.9 \\ -0.9 & 0.3 \end{smallmatrix}\right]$ and $\left[\begin{smallmatrix} 2.0 & 0.2 \\ 0.2 & 3.0 \end{smallmatrix}\right]$, respectively. The scatter of these points is shown in both Fig. 4(a) and (b) as circles.

A 20×20 SOM and a 20×20 ViSOM were applied to learn the data. Both networks used random initial states, and both used exponential neighborhood functions, i.e., $\exp(-dis(c, v)^2/2\sigma^2)$, where c and v denote node c and winner v, respectively, $dis(.)$ denotes the index distance between these two nodes, and σ represents the neighborhood size. σ was decreased from a large size (e.g., ten for this case) to eventually one for the SOM and four for the ViSOM. The resultant maps after 10 000 iterations are shown in Fig. 4(a) and (b). The ViSOM places a fine mesh grid through data points and extrapolates smoothly. The projections of the data points on both maps are shown in Fig. 4(c) and (d). They are the maps used for the visualization purpose. It shows clearly that the ViSOM outperforms the SOM in capturing and showing the data and its structure. The mapped data points on the SOM do not reveal the distribution and structure of the data, while those on the ViSOM preserve their original distribution well. The map's resolution parameter λ was set 0.5 in this example. Larger maps should be used for more accurate projections.

B. Iris Data Set

The well-known Fisher's Iris data set is made of 150 4-D points from three categories, each of which has 50 points [22]. Many data clustering and visualization experiments have used this data set as a benchmark. A 100×100 hexagonal ViSOM has been applied to the data set and the result, i.e., the projected data on the map, is shown in Fig. 5(d). For comparison, a SOM of the same size and structure has also been used to map the data and the result is presented in Fig. 5(c), together with

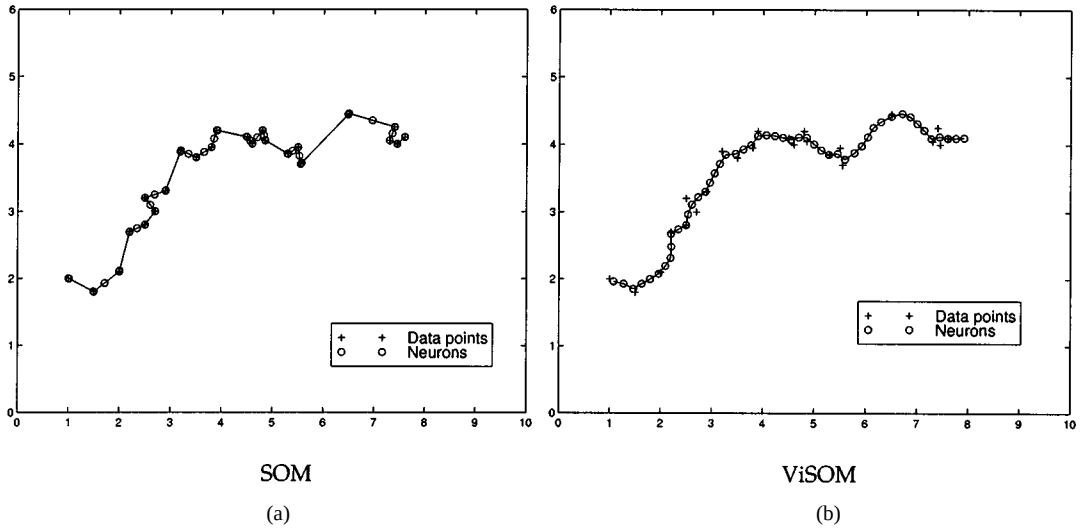


Fig. 3. (a) A 50-node SOM chain in the data space, (b) A 50-node ViSOM chain in the data space, data points are marked with "+," while nodes are marked with circles. $\lambda = 0.16$, $\sigma_f = 6$.

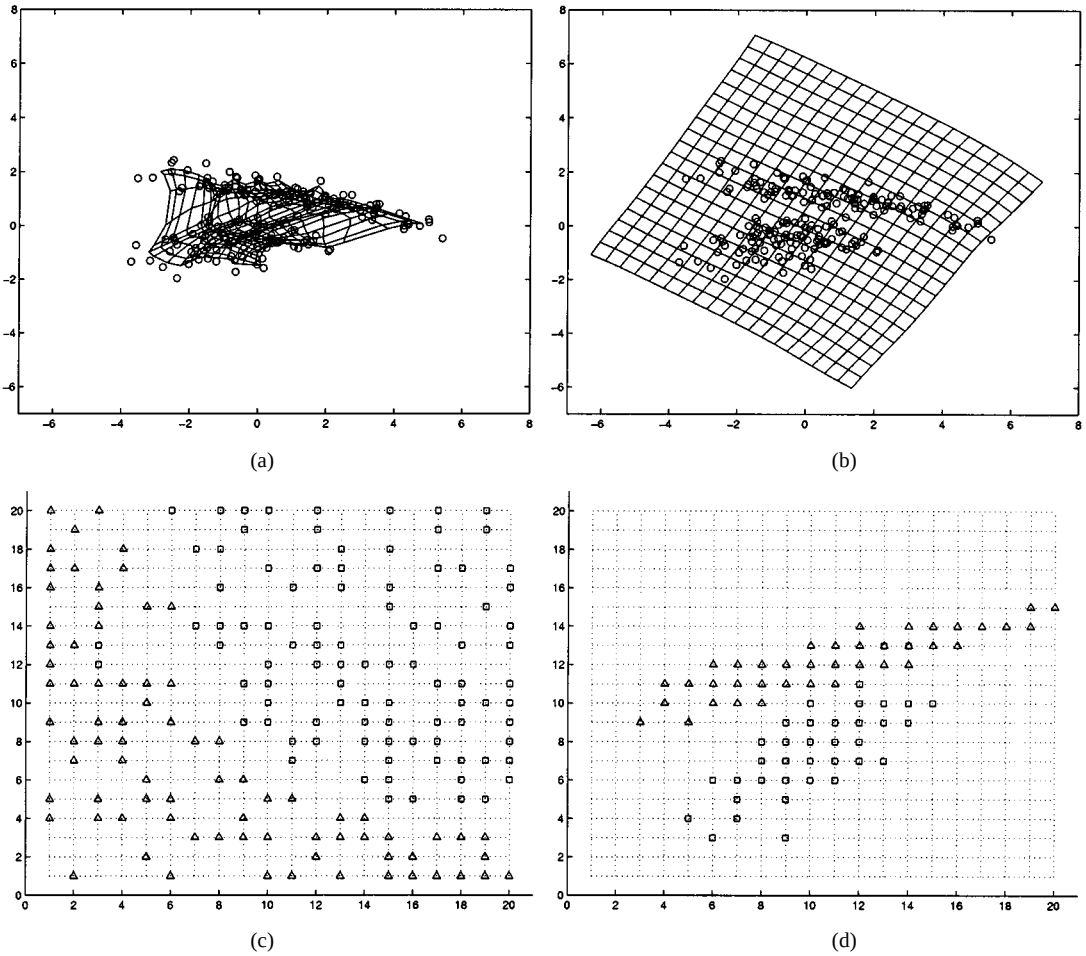


Fig. 4. (a) A trained 20 × 20 SOM in data space. (b) A trained 20 × 20 ViSOM in data space. (c) Appearance of data on the SOM. (d) Appearance of data on the ViSOM. The squares and triangles in (c) and (d) indicate the data points from two Gaussians. $\lambda = 0.5$, $\sigma_f = 1$.

the results of the PCA and Sammon mapping in Fig. 5(a) and (b). The initial states of both SOM and ViSOM were placed on a plane spanned by the first two principal components of the data. The results shown were after 10 000 iterations. The final neighborhood size for the ViSOM was set to four, and resolution $\lambda = 0.075$.

From the results, it can be seen that the Sammon method is better than the linear PCA, as it reveals structure details better. While the ViSOM is even better in the sense that it has not only preserved the inter-cluster structures but also captured more details of intra-cluster and inter-point distributions. Another important point is that the ViSOM can provide the projection

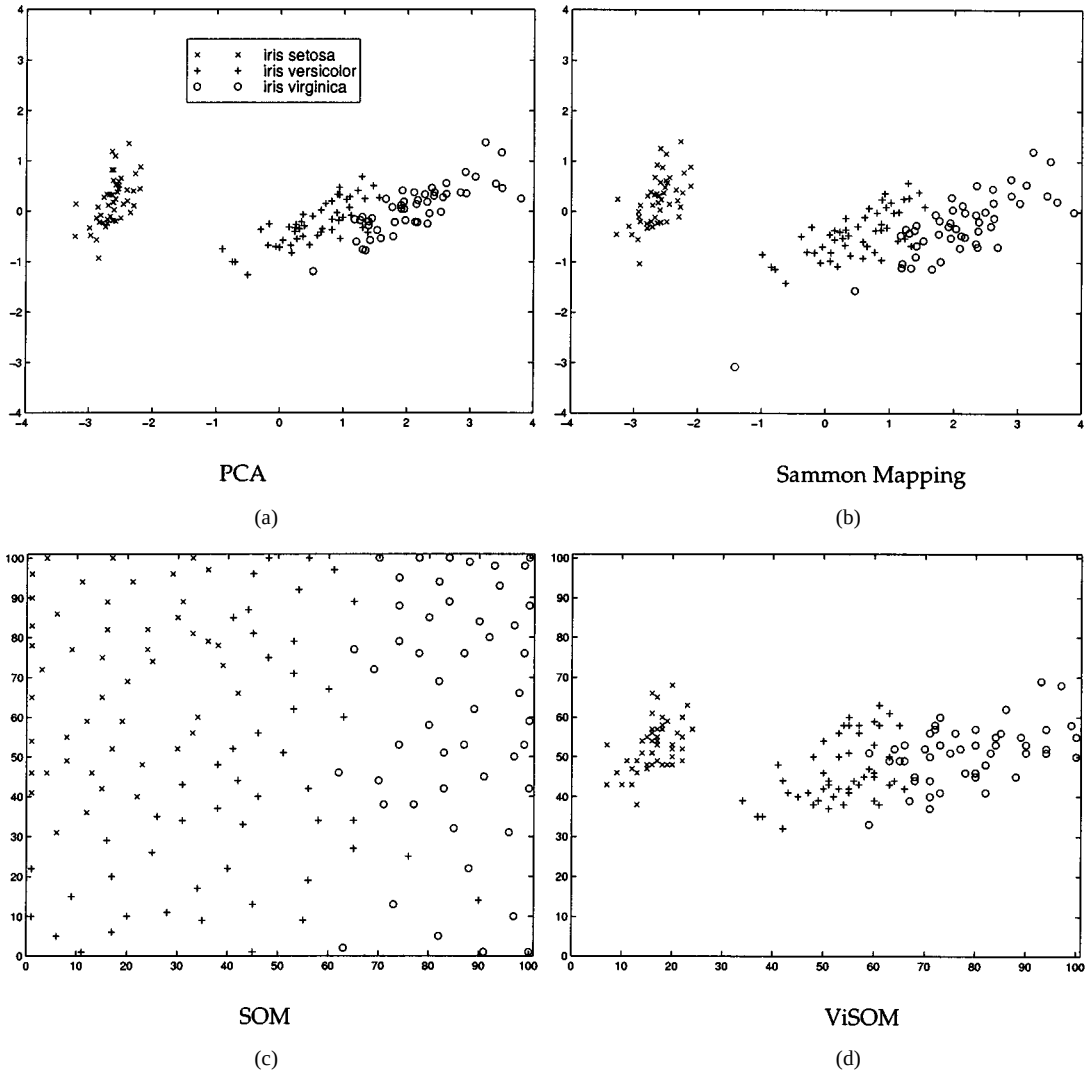


Fig. 5. Mappings of Iris data set: (a) PCA, (b) Sammon Mapping, (c) SOM, (d) ViSOM. Map sizes for SOM and ViSOM are 100×100, and $\lambda = 0.075$, $\sigma_f = 4$.

function, i.e., the discrete map, so that any new data points can find their places on the map, while the Sammon mapping cannot. Moreover, the Sammon mapping is computationally more costly, since it requires both first- and second-order derivatives of the stress function at each iteration [1], [21]. The standard SOM can also form the mapping function. However, the SOM preserves the topology only and can only provide relative relationships among data points. It is hard for the SOM to reveal both inter-cluster and intra-cluster distribution.

V. Conclusion

In this paper, a new mapping method, ViSOM, is proposed for visualization and projection of high-dimensional data. The ViSOM is similar in structure to the SOM, but constrains the *lateral contraction force* within the updating neighborhood. As a result, the map preserves the inter-neuron distances as well as topology as faithfully as possible. The ViSOM produces a smooth and regularly graded mesh through the data points and enables a quantitative, direct, and visually appealing measure of inter-point distances on the map. Thus it can provide a better visual exhibition of data points and their structure and distribution on the map than the SOM can. The ViSOM is as simple an algorithm as the SOM is.

Acknowledgment

The author would like to thank the anonymous Associate Editor and reviewers for their valuable comments and helpful suggestions.

References

[1] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Trans. Comput.*, vol. C-18, pp. 401–409, 1969.
[2] J. Mao and A. K. Jain, "Artificial neural networks for feature extraction and multivariate data projection," *IEEE Trans. Neural Networks*, vol. 6, pp. 296–317, 1995.
[3] T. Kohonen, "Self-organized formation of topologically correct feature map," *Biol. Cybern.*, vol. 43, pp. 56–69, 1982.
[4] ——, *Self-Organizing Maps*. Berlin, Germany: Springer-Verlag, 1995.
[5] A. Ultsch, "Self-organizing neural networks for visualization and classification," *Inform. Classification*, pp. 864–867, 1993.
[6] M. A. Kraaijveld, J. Mao, and A. K. Jain, "A nonlinear projection method based on Kohonen's topology preserving maps," *IEEE Trans. Neural Networks*, vol. 6, pp. 548–559, 1995.

[7] S. P. Luttrell, "Hierarchical self-organizing networks," in *Proc. Inst. Elect. Eng. Int. Conf. Artificial Neural Networks*, 1989, pp. 2–6.
[8] ——, "A Bayesian analysis of self-organizing map," *Neural Comput.*, vol. 6, pp. 767–794, 1994.
[9] H. Yin and N. M. Allinson, "On the distribution and convergence of feature space in self-organizing maps," *Neural Comput.*, vol. 7, pp. 1178–1187, 1995.
[10] ——, "Interpolating self-organizing map (iSOM)," *Electron. Lett.*, vol. 35, pp. 1649–1650, 1999.
[11] T. F. Cox and M. A. A. Cox, *Multidimensional Scaling*. London, U.K.: Chapman and Hall, 1994.
[12] B. D. Ripley, *Pattern Recognition and Neural Networks*. Cambridge, U.K.: Cambridge Univ. Press, 1996.
[13] E. Oja, "Neural networks, principal components, and subspaces," *Int. J. Neural Syst.*, vol. 1, pp. 61–68, 1989.
[14] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward network," *Neural Networks*, vol. 2, pp. 459–473, 1991.
[15] J. Rubner and P. Tavan, "A self-organizing network for principal component analysis," *Europhys. Lett.*, vol. 10, pp. 693–698, 1989.
[16] E. C. Malthouse, "Limitations of nonlinear PCA as performed with generic neural networks," *IEEE Trans. Neural Networks*, vol. 9, pp. 165–173, 1998.
[17] T. Hastie and W. Stuetzle, "Principal curves," *J. Amer. Statist. Assoc.*, vol. 84, pp. 502–516, 1989.
[18] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AICHE J.*, vol. 37, pp. 233–243, 1991.
[19] J. Karhunen and J. Joutsensalo, "Generalization of principal component analysis, optimization problems, and neural networks," *Neural Networks*, vol. 8, pp. 549–562, 1995.
[20] B. Schölkopf, A. Smola, and K. R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, pp. 1299–1319, 1998.
[21] G. Biswas, A. K. Jain, and R. C. Dubes, "Evaluation of project algorithms," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-3, pp. 701–708, 1981.
[22] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, pp. 178–188, 1936.