An Algorithm for Generalized Principal Curves with Adaptive Topology in Complex Data Sets

G. Balzuweit[1], R. Der[1], M. Herrmann[2], M. Welk[1]

[1] Universität Leipzig, Inst. für Informatik,
Augustusplatz 10/11, D-04109 Leipzig, Fed. Rep. of Germany
der@informatik.uni-leipzig.de
[2] RIKEN, Lab. for Information Representation,
2-1 Hirosawa, Wako-shi, 351-01 Saitama, Japan
michael@murasaki.riken.go.jp

Abstract. Generalized principal curves are capable of representing complex data structures as they may have branching points or may consist of disconnected parts. For their construction using an unsupervised learning algorithm the templates need to be structurally adaptive. The present algorithm meets this goal by a combination of a competitive Hebbian learning scheme and a self-organizing map algorithm. Whereas the Hebbian scheme captures the main topological features of the data, in the map the neighborhood widths are automatically adjusted in order to suppress the noisy dimensions. It is noteworthy that the procedure which is natural in prestructured Kohonen nets could be carried over to a neural gas algorithm which does not use an initial connectivity. The principal curve is then given by an averaging procedure over the critical fluctuations of the map exploiting noise-induced phase transitions in the neural gas.

1 Introduction

Analogous to principal components, principal curves represent essential data features if the underlying functional relation is non-linear. Each point of a principal curve represents the local centroid of the data in the orthogonal directions and provides thus an estimation of a one-dimensional relation in noise-corrupted data. To construct principal curves constitutes an elementary task in non-linear statistics which may be approached in simple cases [2, 3, 4] non-parametrically using suitably modified Kohonen maps with a chain-like topology. Whereas in statistics the centroid condition is accompanied by a loosely defined smoothness requirement for the principal curve, in self-organizing maps both smoothness and local averaging are combined in a most natural way.

In the present contribution we will address this task in a more general manner. The algorithm proposed here goes beyond Kohonen maps as network *structures* are generated in a self-adaptive manner. This allows to extract *generalized* principal curves which may have branching points or may be disconnected, while exhibiting the typical averaging properties in the non-exceptional regions. Thus, non- or multiply connected data structures can be represented without any prior knowledge of the number of components and their respective structure. In this framework a representation of principal curves is given in terms of a discrete set of reference vectors which are updated by a special self-organizing map algorithm which combines adaptation schemes for the reference vectors, for the map parameters and for the network structure. The smoothness condition is included by intrinsic elasticity properties of self-organizing maps which are adaptively modified in dependence on the local variance of the noise. Structure adaptation is performed by a simple Hebbian learning rule, and the reference vectors forming the principal curve arise as averages over fluctuations of the maps at the border of stability.

In the next section the main achievement of the present work, an adaptation scheme for the network topology, is derived. The following sections present the algorithm and results for several typical examples.

2 Collaboratively learning vector quantizers

In on-line learning vector quantizers, reference vectors w_r are adjusted towards a current input datum $v \in \mathcal{V}$,

$$w_r(t+1) = w_r(t) + \varepsilon h_r(v, \mathbf{w})(v - w_r(t)), \tag{1}$$

where $\mathbf{w} = (w_r | r \in \mathcal{A})$, ε is the learning rate, w_r is a vector in the input space $\mathcal{V}$, the range of the indices r is a discrete output (code) space $\mathcal{A}$, and $h_r(v, \mathbf{w})$ involves the interaction among the units. In Kohonen's algorithm the interaction is formulated by the neighborhood function

$$h_r(v, \mathbf{w}) \equiv h_{rs} = \exp\left(-\frac{\|r - s\|^2}{2\sigma^2}\right), \quad s = \arg\min_r \|w_r - v\|, \tag{2}$$

where $\|r - s\|$ is the distance between neurons r and s according to the topology of the neuron space. For a topology preserving mapping, this topology has to be known, which is the main disadvantage of Kohonen's algorithm. Instead, one has to infer this topology from the data. The *neural gas* algorithm [6] is suitable for this purpose while otherwise (cf. the following section) exhibiting similar properties as Kohonen's learning rule. Namely, instead of (2) the neighborhood function relies on the rank $R(r)$ in the ordered sequence of distances $\|w_r - v\|$. For the best matching unit s_0 we have $R(s_0) = 0$, the second-best has $R(s_1) = 1$ and so on. Thus,

$$h_r(v, \mathbf{w}) = \exp\left(-R(r)/\sigma\right) \tag{3}$$

and the weights are learned according to (1) while decreasing σ. Eventually, the topology is represented in terms of a connectivity matrix arising from a simple Hebbian learning rule, cf. eq. (5) below.

However, the topology learnt in this way is that of the noisy data and not of the generalized principal curve to be constructed. A principal curve is obtained from the *neural gas* by keeping the interaction width σ sufficiently high so that the weights are forced into chain-like structures. The algorithm given below will self-consistently adapt these widths locally to obtain the desired principal curve representation of the data set.

3 Phase transitions in the neural gas

In self-organizing maps evolving according to Kohonen's algorithm, phase transitions due to dimensional conflicts between the input topology and that of the output lattice of neurons are well investigated. A generic example (cf. [7]) is the mapping of a rectangular data distribution onto a one-dimensional lattice of neurons.

Viewing the rectangle as a straight line corrupted by noise, the height s of the rectangle measures the variance of the noise. For low noise, $s < s_c = 2.02\sigma$ where σ is the neighborhood width, the chain of neurons is mapped onto the original data distribution, i. e. the straight line.

At $s = s_c$ the noise induces a phase transition to a new stable configuration corresponding to a folding of the map into the input space. There is a coexistence between two phases of similar shape where one of the two phases still conserves topology whereas the other one is topology violating. The latter is the more stable one so that the phase transition is signaled usually by the topology violations.

In the neural gas scenario there is no fixed topology (like that of the chain in the example above). Nevertheless one observes phase transitions very similar to the one described above. The point is that for sufficiently large σ the neurons are mapped chain-like into the rectangular input space. This is an immediate consequence of the cooperativity in learning introduced by the neighborhood function. Now, for given neighborhood width σ there is a critical noise level $s_c = s_c(\sigma)$ such that for $s < s_c$ the neurons are mapped onto the median of the rectangular data distribution while for $s > s_c$ we observe a wave like structure very much similar as the one observed with the Kohonen map. Of course there is no topology violation now, since there is no neighborhood between the neurons defined. However it can be observed that in the "folding" phase the first and second winner are no longer neighbors in input space. There are further neurons lying between the two in the sense that their (Euclidean) distance from both the first and the second winner is smaller than the distance between the first and second winner themselves. The number of these neurons "in between" will serve in the following for fixing the local value of the neighborhood function optimally.

4 Topology adaptation

We start with a modification of the neural gas algorithm with real-valued connectivity matrix C. The connection value C_{rs} expresses the current belief on whether neurons r and s are connected; it is increased each time that r and s are the first and second winner while in each step all connections are weakened simultaneously. In evaluation, a connection is considered active according to a threshold condition. The change of topology caused by a new connection being created is used as a criterion for topology violation which controls the learning of an individual neighborhood width. The ranks $R(r)$ are assigned to the units in each step in the order of their distances $\|w_r - w_{s_0}\|$ from the first winner (rather than the input vector), thus enforcing a topological aspect in weight learning.

In topology adaptation, the neighborhood function (3) displays a disappointing behavior insofar as single points tend to "escape" from the adapted topology scheme. Therefore another neighborhood function was derived from (2) by replacing $||r - s||$ by the rank $R(r)$.

Another sort of instabilities still emerges as a result of the ranking by Euclidean distances. This will support clustering of neurons since for instance in a cluster of two, the stimuli which make the two the first and second winner will always move them closer together. Note that, e.g., for a linear equidistant (chain-like) arrangement of units the rank alternates (5–3–1–0–2–4–6), thus making the immediate neighbors of the winner learning with different strength, whereas under (2) both are co-learning with the same strength.

To overcome these instabilities one wishes to assign the neighborhood ranks in analogy to the Kohonen algorithm by searching the C matrix for neighbors. However, matrix searching is of high numerical complexity. This led us to try a neighborhood function based on searching only immediate neighbors in the C matrix and ranking by Euclidean distance from the winner for all other units. It turned out that such a "mixed ranking" grants essentially all the benefits of a full matrix search while preserving the lower numerical complexity of Euclidean ranking.

As can be seen from the figures, extra connections can be observed in some cases near branching points. These represent an uncertainty in the position of the branching points caused by noise. For purposes like object recognition they can be removed in a subsequent step.

5 The algorithm

For each input signal v drawn from a distribution $P(v)$ the first and second winner s_0, s_1 are calculated. The neighborhood parameter σ is initialized at e.g. $N/3$ (with N being the number of units) and is decreased slowly later on:

$$\sigma_r := \max\{\sigma_r - \varepsilon_\sigma \sigma_r, 1\}. \tag{4}$$

The connection values C_{rs} develop by decreasing in each step all links simultaneously, followed by an increase of the link between the first two winners.

$$\Delta C_{ij} = -\frac{1}{N}\varepsilon_c C_{ij} \;\; \forall i,j \quad \text{and} \quad \Delta C_{s_0 s_1} = \varepsilon_c \tag{5}$$

If in this way a new link has been created (i.e. has grown across the threshold) the neighborhood parameter σ_r is increased for all neurons with pointers w_r geometrically between w_{s_0} and w_{s_1}, counteracting to the persistent decrease of σ. We consider those neurons as situated between the first and second winner which are closer to both of them than these are to each other, i.e. for which

$$||w_r - w_{s_0}|| < ||w_{s_0} - w_{s_1}|| \quad \text{and} \quad ||w_r - w_{s_1}|| < ||w_{s_0} - w_{s_1}||. \tag{6}$$

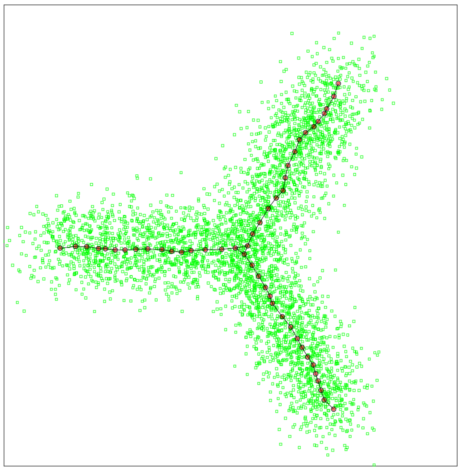

Fig. 1. This Y-shaped data distribution cannot be represented appropriately by a linear chain. Instead, a generalized principal curve with one branching point is constructed by adaptively structuring a network of 50 neurons.

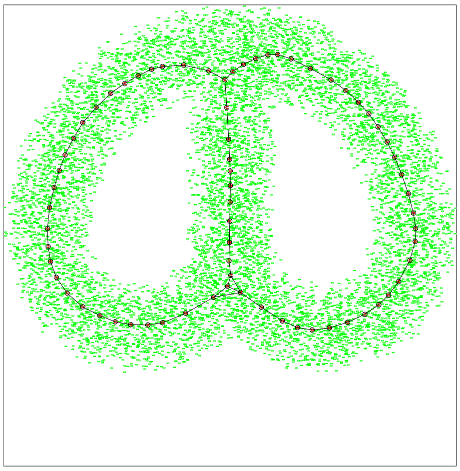

Fig. 2. Representation of a multiply connected data set by a closed generalized principal curve with two branching points; $N = 70$.

σ is increased towards an estimated optimal $\hat{\sigma}_{\mathrm{opt}}$ which is proportional to the number of neurons satisfying (6) with a global proportionality constant.

$$\Delta\sigma_r = \mu_\sigma (\hat{\sigma}_{\mathrm{opt}} - \sigma_r) \text{ if } \sigma_r < \hat{\sigma}_{\mathrm{opt}} \tag{7}$$

Further, the σ_r must not exceed their initial value.

Next, sort the list of distances $||w_r - w_{s_0}||$ in an ascending order and define a modified neighborhood function $h_r(v, \{w\})$ by assigning ranks $R_r = 1$ to all units which are directly connected to the winning neuron, while using the usual ranks otherwise. The update rule of the algorithm finally reads

$$\Delta w_r = \varepsilon \exp\left(-\frac{2R_r^2}{(\sigma_r + \sigma_{s_0})^2}\right)(v - w_r)\,. \tag{8}$$

6 Conclusion

In the present contribution the principal curve problem was solved in a quite general way. This task required an algorithm that combines learning at different levels, in particular of the network structure, while retaining the capabilities of the learning mechanisms at the other levels. In the present algorithm the solution of the principal curve problem was not given in terms of a convergent network state, but as an average over the network's dynamical behavior in the vicinity

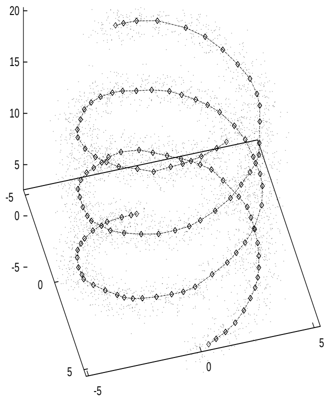

Fig. 3. Solution of a two-spirals problem found in an unsupervised manner by the present algorithm. The constructed principal curve consists of two disconnected chains evolving from the same initial state of the *neural gas* as in the above examples. Here, $N = 120$.

of a topological phase transition. The algorithm is relatively efficient since the most time consuming stage is the averaging process.

Forthcoming work on this topic will address a theoretical analysis of the phase transition in the neural gas algorithm analogous to the analysis in [7] as well as a possible generalization to generalized principal manifolds of dimension greater than one.

ACKNOWLEDGEMENT: One of the authors (RD) gratefully acknowledges the hospitality of RIKEN (Tokyo) he received during a visit in February 1996.

References

1. Der, R., Herrmann, M.: Critical Phenomena in Self-Organized Feature Maps: A Ginzburg-Landau Approach. *Phys. Rev.* **E 49**:6, 5840–5848 (1994).
2. Der, R., Balzuweit, G., Herrmann, M.: Constructing Principal Manifolds in Sparse Data Sets by Self-Organizing Maps with Self-Regulating Neighborhood Width. To appear in *Proc. ICNN'96*, Washington DC (1996).
3. Herrmann, M.: Self-Organizing Feature Maps with Self-Organizing Neighborhood Widths. *Proc. 1995 IEEE Intern. Conf. on Neur. Networks*, 2998–3003 (1995).
4. Kiviluoto, K.: Topology Preservation in Self-Organizing Maps. Subm. to *Proc. 1996 IEEE Intern. Conf. on Neur. Networks* (1996).
5. Kohonen, T.: *The Self-Organizing Map.* Springer-Verlag (1995).
6. Martinetz, T.: *Selbstorganisierende neuronale Netzwerkmodelle zur Bewegungssteuerung.* Gesellschaft für Informatik e.V. (1992).
7. Ritter, H., Martinetz, T., Schulten, K.: *Neural Computation and Self-Organizing Maps.* Addison-Wesley (1992).
8. Villmann, T., Der, R., Herrmann, M., Martinetz, T.: Topology Preservation in SOFMs: Exact Definition and Measurement. To appear in *IEEE Transact. on Neur. Netw.* (1994).
9. Yuille, A. L., Geiger, D.: Winner-take-all Mechanisms. In: Arbib, M. A.: *The Handbook of Brain Theory and Neural Networks.* The MIT Press, 1056–1060 (1995).