# Neural Gas for Sequences

Marc Strickert and Barbara Hammer
Department of Mathematics/Computer Science
D-49069 Osnabrück, Germany
{marc,hammer}@informatik.uni-osnabrueck.de

Keywords: sequence processing, temporal Kohonen map, recursive SOM,
SOM for structured data, NG with context, HSOM

***Abstract—*** For unsupervised sequence processing, standard self organizing maps (SOM) can be naturally extended by recurrent connections and explicit context representations. Known models are the temporal Kohonen map (TKM), recursive SOM, SOM for structured data (SOMSD), and HSOM for sequences (HSOM-S). We discuss and compare the capabilities of exemplary approaches to store different types of sequences. A new efficient model, the Merge-SOM (MSOM), is proposed, combining ideas of TKM and SOMSD, and which is well suited for processing sequences with dynamic multimodal densities.

## 1 Introduction

Kohonen's self-organizing map (SOM) is a very intuitive and powerful tool for mining high dimensional data sets [8]. The computational results essentially depend on two priorly chosen ingredients: the metric for data comparison, and the grid topology of the self-organizing map. SOM yields excellent results if the intrinsic dimensionality of data is low and the input features are scaled appropriately to match the Euclidean metric. In other cases, several alternatives to overcome the problem of potential topology mismatches in more complex situations have been suggested, e.g. the usage of non-Euclidean grids [11], the adaptation the of the lattice dimensionality during training [2], or data optimal dynamic neuron connectivity [10]. Moreover, approaches which adapt the metric according to auxiliary information have been proposed for handling high dimensional and heterogeneous data [7]. Unfortunately, the paradigms of self-organization cannot be easily transferred to non-vectorial data like time series or more general graph structures. The yet most common workaround for neural time-series processing is the temporal unfolding by embedding the data into a finite dimensional vector space [9, 12]; however, the Euclidean metric and standard rectangular lattices are seldom appropriate for matching the resulting possibly complex data topology. Other approaches recursively model sequential data [3, 4, 13, 14, 15], see e.g. [1] for an overview.

Unsupervised neural sequence processors with recurrent connections compare single sequence elements recursively. Thereby, the similarity structure of entire sequences emerges by the integration of the recursive comparison. In contrast to their supervised counterparts, a variety of models exists for unsupervised recurrent self-organizing networks: the temporal Kohonen map (TKM), the recurrent SOM (RSOM), recursive SOM (RecSOM), and SOM for structured data (SOMSD), to name just a few [3, 4, 14, 15]. It is not clear which model performs best in certain situations, and the representation capacities of those approaches, their similarities and differences are only partially understood. Recently, a general framework has been proposed to cover these approaches in a unified notation [5, 6]: they obey the same recursive dynamics, but they differ in their way of internally representing the temporal context. A crucial point for understanding the models is addressed by the question about the representational capabilities and the encoding characteristics of sequences by the used temporal context. Important theoretical aspects, such as the dynamics of training, the resulting metric structure, and the notion of topology preservation can be investigated within the scope of that question. First steps can be found in [5, 6]. Here, we will consider the capacity of several context models for representing various types of sequences. Just to mention it, the more elaborate the context models, the better the dealing with complex sequences. The capacity of simple models like TKM is restricted to short sequences over a finite alphabet, which can be represented by the models. However, due to the algorithms' just implicit learning of context, to be explained in the next section, it is likely to be affected by noise and topological mismatches. We will argue that the range of context representation cannot be used in an optimal way because of this implicit definition. As an alternative, we propose a new simple model which yields the same encoding strategy but which uses an explicit context representation, a separate adaptation of the context, and which therefore provides a better exploitation of the possible range.

Now, we introduce some recursive unsupervised models for sequence processing and discuss their respective notion of context. After that, we propose a new simple alternative to TKM. In three benchmarks we demonstrate its representation capabilities with different types of time series.

## 2 SOMs for sequence processing

The self organizing map is given by a set of neurons $N = \{1, \dots, K\}$ equipped with weights $w_i \in \mathbb{R}^n$, and a topology of the neural arrangement with a function $d_N : N \times N \to \mathbb{R}$. Often, neurons are connected in a two or low-dimensional grid and $d_N$ denotes the distance of two neurons in that grid. A similarity measure on the weight space $\mathbb{R}^n$ is fixed; for simplicity, we will assume the standard squared Euclidean metric $|\cdot|^2$. The goal of training is to find weights which represent a given set of data points in $\mathbb{R}^n$ in such a way that the topology of the neural grid matches the topology of the given data: data points are iteratively presented, and the weights of the closest neuron and its neighbors in the grid are adapted towards the given point. Popular alternatives to the standard SOM are the neural gas (NG) and the hyperbolic SOM (HSOM). The HSOM model operates on a lattice with exponentially increasing neighborhood size [11]; NG adapts all neurons according to their rank with respect to the distance from the current data point, i.e. according to a data-driven optimal lattice structure [9] without topology constraints.

Several popular models extend SOM by means of recurrent connections: assume a sequence of data points in $\mathbb{R}^n$, where $a_t$, an entry $t$ of the sequence at time $t$, is presented:

**TKM** recursively computes the distance of neuron $j$ from $a_t$ as

$$\begin{aligned}\tilde{d}_j(a_t) &= (1-\alpha)\cdot|a_t-w_j|^2+\alpha\cdot\tilde{d}_j(a_{t-1})\\ &= \textstyle\sum_{i\geq 0}(1-\alpha)\cdot\alpha^i\cdot|a_{t-i}-w_j|^2\end{aligned}$$

$\alpha\in(0,1)$ is fixed and denotes the influence of the context compared to the currently presented training pattern. As for standard SOM, the neuron with smallest distance becomes winner and the weights in its inclusive neighborhood are adapted towards the given training pattern $a_t$. Temporal context for TKM is represented by the activation of the winner neuron in the previous time step. Obviously, this context choice can also be combined with hyperbolic lattice structures and the data dependent neighborhood of NG.

**RecSOM** uses a much richer notion of context [15]: besides a weight $w_i\in\mathbb{R}^n$, each neuron $i$ possesses a vector $c_i\in\mathbb{R}^N$ representing the temporal context as the activation of the entire map in the previous time step. Here,

$$\tilde{d}_j(a_t)=\alpha\cdot|a_t-w_j|^2+\beta\cdot|C_t-c_j|^2$$

whereby $\alpha$, $\beta>0$ are fixed values to control the influence of the context compared to the presented sequence element. The context is given by $C_t:=(\exp(-\tilde{d}_1(a_{t-1})),\ldots,$ $\exp(-\tilde{d}_N(a_{t-1})))$. Training adjusts both weights and contexts of neurons in the inclusive neighborhood of the winner towards the current values $a_t$ and $C_t$. This context type can be used also for hyperbolic lattices or NG-topology. However, this context model is costly because its dimensionality coincides with the number of neurons in the map.

**SOMSD** has been proposed for general tree structures [4], and we only describe the special case of sequences. Like for RecSOM, additional context vectors $c_i\in N$ for each neuron are used, which store computationally inexpensive information about the previous map state: the winner in the previous step. Here

$$\tilde{d}_j(a_t)=\alpha\cdot|a_t-w_j|^2+\beta\cdot d_N(i_{t-1},c_j)$$

whereby $i_{t-1}$ denotes the index of the winner in step $t-1$ and $d_N$ the measuring function for distances in the grid, usually the squared Euclidean distance of the neurons' coordinates in their low-dimensional lattice. During training, weights and contexts of the winner's inclusive neighborhood are adapted towards $a_t$ and $i_{t-1}$, respectively. Thereby, the grid topology and indices of neurons are embedded into a real-vector space. Since the distance between context vectors is defined by the lattice dependent measure $d_N$, this approach must be carefully revised to handle alternative lattices. It can be expected that regular grids produce topological mismatches if complex sequences are dealt with.

**HSOM-S** transfers the idea of SOMSD to hyperbolic lattices (or more generally arbitrary grids given by a triangulation of a two-dimensional manifold) which might better match the data topology [13]. The hyperbolic space is thereby locally approximated by Euclidean geometry within triangles. For a stable learning, the context influence is augmented from 0, which yields initial weight-driven SOM ordering, to a desired value.

Obviously, RecSOM offers the richest notion of context – temporal context is represented by the activation of the entire map in the previous time step – but also constitutes the computationally most demanding model. SOMSD and HSOM-S still use global map information, but in a compressed form; storing only the location of the winner is much more efficient, albeit somehow lossy, in comparison to the whole activity profile. However, context similarity depends on a priorly chosen grid topology and a data optimal NG topology cannot be used. TKM context is given by the activation of only the neuron itself. The context is just implicitly represented within the weights of neurons in $\mathbb{R}^n$. Hence, the domain of its sequence representations is restricted to the domain of sequence entries. The question now occurs which sequences can be represented by which context types; given an explicit context type, a specific type of time series (i.e. continuous or discrete entries), a fixed time horizon $T$ (possibly infinite), and also given enough neurons: can we then find a map using this context such that every part of a time series of length $T$ (possibly after discretization of its elements) is represented in the map by a separate winner? Moreover, which data-driven explicit encoding of the context is generated by the map?

Since a finite number of neurons is available, only a finite time horizon $T$ and a data discretization can be stored in the map, and we therefore focus our considerations to discrete sequence entries $S\subset\mathbb{R}^n$. [1] SOMSD and HSOM-S can represent finite length discrete sequences as follows: assume that $i$ is the index of a winner which represents sequence $[a_1,\ldots,a_t]$. Then neuron $j$ with $w_j=a_{t+1}$ and

---

[1] Even a continuum of neurons would be subject to restrictions: the representation space for sequences is given by $\mathbb{R}^{n+r}$ where $r=0$ for TKM and $r$ is given by the lattice dimensionality for SOMSD or HSOM-S. It is a well known fact from topology that continuous and injective mappings from a high dimensional vector space to lower dimensions do not exist (Theorem of Borsuk). Hence, an upper bound on the time horizon $T$ of representable continuous sequences exists which is independent of the number of neurons for these models. For RecSOM the context would be infinite dimensional if a continuum of neurons was dealt with.

$c_j = i$ is best matching for the sequence $[a_2, \ldots, a_{t+1}]$. Sequences of length $t$ can therefore be internally represented by the index of the winning neuron, i.e. their location in the map. Experiments in [4] demonstrate that this form of representation is found by Kohonen's learning rule. In particular, we can find for every finite horizon $T$ a map (with a sufficient number of neurons) of which the neurons represent *all* sequence parts of length $T$ over a finite alphabet. Note, that the issue of topology preservation is excluded from this argumentation. The neighborhood connectivity of sequence data is an exponential function of the length if arbitrary element concatenation is allowed. Obviously, this data topology is incompatible with a standard grid structure; as a result numerous neurons in a trained map do not become winner or they develop ambiguous contexts. As we will show in our experiments, quite a large number of idle neurons can also be observed even for the exponential neighborhood of HSOM-S.

RecSOM stores sequences by modeling the context as the exponentially transformed activation of the entire map. It can be expected that the activation profile of a trained RecSOM displays unimodal activation, or an activation with only a few modes, corresponding to the fact that one specified neuron is activated by the given sequence entry and only some possible predecessors. Hence, this context enables a more detailed representation than the winning neuron only; especially complex sequences with high information gain can be well handled by this kind of context without producing many idle neurons.

In contrast to these approaches, the range of the implicit context representation of TKM is fixed to the weight range, and just using more neurons does not affect this range. Any information concerning the context in which neuron $i$ should become active has to be stored in the vector $w_i \in \mathbb{R}^n$. For a given sequence $[a_1, \ldots, a_t]$, one can compute the weight $w_i \in \mathbb{R}^n$ in the weight range which yields optimal activation:

$$w_i = \sum_{j \geq 0} \alpha^j a_{t-j} / \sum_{j \geq 0} \alpha^j$$

If sequence entries come from $S = \{1, \ldots, n\}$ a value $\alpha = 0.1^{n+1}$ yields, up to a scaling factor, a representation of the sequence in $w_i$ to base $n+1$. Thus, the recursive computation of the winning neuron is based on an encoding of sequences of finite time horizon in the weights that is similar to fractal codes like the Cantor set. In principle, discrete sequences can be encoded this way in TKM; their number is restricted by the weight range $\mathbb{R}^n$ and by the computation accuracy.

However, such encoding is only partially obtained by standard Kohonen training. We consider two simplified situations: assume that the range of neighborhood adaptation $\sigma$ is very small and that only the winner $i_0$ is adapted in each step with learning rate $\eta > 0$. Then the training steps $\triangle w_i = \eta \cdot \exp(-d_N(i, i_0)/(2\sigma^2)) \cdot (a_t - w_i)$ yield an adjustment of weight $w_i$ of the winner $i$ for sequence $[a_1, \ldots, a_t]$ towards only the last entry $a_t$ (or all entries for which $i$ becomes winner), and the above fractal encoding of the context is not achieved in this situation. Conversely, assume that by the range of the neighborhood adaptation the weight of a neuron $i$ is adapted towards the current sequence element in each step with a constant factor $\eta_0 = \eta \cdot \exp(-d_N(i, i_0)/(2\sigma^2))$. Then Kohonen's learning rule yields the weight $w_i = \sum_{i \geq 0} \eta_0 (1 - \eta_0)^i a_{t-i} + (1 - \eta_0)^t w_i(0)$ after presentation of $[a_1, \ldots, a_t]$ starting from weight $w_i(0)$. This is a fractal encoding, whereby parameter $\eta_0$ has to be chosen according to the respective $\alpha$ of the recursive dynamics. Such encoding, however, is only realized in the special update situation where all sequence entries have an effect on the weight of the winner in step $t$. For small neighborhood range, this situation only takes place, if the winners after each step are located within a similar region of the map.

In addition, even if fractal encoding could be achieved during training, the weight range is presumably not optimal for context encoding: the parameter $\alpha$ determines *both* the influence of the context in comparison to the current value (i.e. the metric) and the location of weights with optimal response (i.e. context representation). On the one hand, $\alpha$ is chosen small to ensure a stable recursive dynamic and to bring forth an appropriate influence of the current entry on the similarity adaptation [15]. On the other hand, an optimal distribution of the context within $\mathbb{R}^n$ would require an $\alpha$ depending on the size of the sequence alphabet, which is not related to stability issues.

## 3 Merge-SOM

We propose Merge-SOM (MSOM), an alternative to TKM, which also encodes contexts within the weight space $\mathbb{R}^n$, based on fractal codes, but which uses an explicit representation of the context and a separate adaptation for which appropriate representations can be achieved with Kohonen's learning rule: For MSOM, each neuron is equipped with a tuple $(w_i, c_i) \in \mathbb{R}^{2n}$ whereby $w_i$ represents the actual weight and $c_i$ the context. The distance of the sequence entry $a_t$ from neuron $j$ is recursively computed by

$$\tilde{d_i}(a_t) = (1 - \alpha) \cdot |a_t - w_j|^2 + \alpha \cdot |C_t - c_j|^2$$

whereby context

$$C_t = \beta \cdot w_{I_{t-1}} + (1 - \beta) \cdot c_{I_{t-1}},$$

is defined as the linear combination (the merging) of the weight $w_{I_{t-1}}$ and the context $c_{I_{t-1}}$ of the last winner in the computation, for which the index is denoted by $I_{t-1}$. A typical merging value for $\beta > 0$ is 0.5. Training iteratively rotates the weight and the context vectors of a neighborhood of the winner towards the values $a_t$ and $C_t$. Hence, like for SOMSD and RecSOM, the context is stored explicitly in the neurons and adapted separately from the weight. As for TKM, context is represented in the weight space. MSOM shares the efficiency of SOMSD and HSOM-S due to this compressed context representation. Since this notion does not refer to the lattice distance, one can use the context of MSOM also with hyperbolic grids as well as the data optimal grid of neural gas. We will use the data dependent topology of NG in the following experiments and refer to the model as MNG.

MNG as an instance of MSOM stores sequences in the same way as TKM, as can be seen as follows: assume that discrete sequences are used and that weights and contexts are already specialized, i.e. the weight $w_i$ of the current winner approximates the presented sequence entry $a_t$ and the context $c_i$ nearly coincides with the respective context $C_t$ computed at this time step in the map. For a given time step $t$, context $C_t$ can then be unfolded to

$$\begin{aligned} C_t &= (1-\beta)\cdot w_{I_{t-1}} + \beta \cdot c_{I_{t-1}} \\ &\approx (1-\beta)\cdot a_{t-1} + \beta \cdot C_{t-1} \\ &= \dots \\ &\approx \sum_{i\geq 1}(1-\beta)\beta^{i-1}a_{t-i} \end{aligned}$$

Thus, context representation is adapted towards a fractal encoding of sequences by Kohonen's learning rule. Note that the parameter $\beta$, which determines the encoding, can be chosen independently of the parameter $\alpha$ which determines the influence of the context on the similarity of sequences. As an example, if a binary sequence is used, $\beta$ can be chosen close to $0.5$ to achieve an optimal covering of the range for the context representations.

In order to obtain a good value for the context influence $\alpha$ in the distance calculation, we exploit the entropy of the neural activity: after the random initialization of weights and contexts, an increase of the neuron specialization takes place during training, thus, leading to an overall decrease of the neuron activation entropy. By augmenting the context influence $\alpha$ we get a richer state space for the distance computation, which compels a more detailed winner selection. In this way, the map activation entropy grows. Our control strategy for $\alpha$ is to adapt it dynamically at run time: we increase $\alpha$ in case of an entropy decay trend and decrease $\alpha$ otherwise, while integrating local fluctuations in the map activations by means of a momentum term. Empirically, about 1000 entropy values turned out to be sufficient for mastering a complete training task.

# 4 Experiments

Three training experiments were carried out to test the performance of MNG in comparison to alternative unsupervised context models. The first set is the Mackey-Glass continuous 1D time series with quasi-periodic oscillations, exhibiting a temporally changing cyclic structure. In a second task, a discrete sequence from a binary automaton was used to learn the characteristic words. Our third series is a 3D space filling curve with continuous physiological observations, and it was derived from the Santa-Fe time series contest set known as data set B-1991.

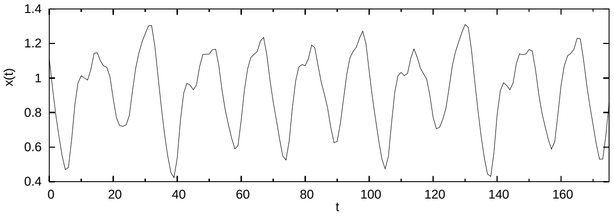

Figure 1: Mackey-Glass time series.

**Mackey-Glass Series:** The chaotic Mackey-Glass time series shown in figure 1 is the first learning experiment. The parameters for this differential equation model $\frac{\mathrm{d}x}{\mathrm{d}\tau} = bx(\tau) + \frac{ax(\tau-d)}{1+x(\tau-d)^{10}}$ are $a = 0.2$, $b = -0.1, d = 17$, which, for comparison purposes, are the same as given in other publications [15]. Three models have been trained: standard NG, the hyperbolic SOM for sequences (HSOM-S), and the merging neural gas (MNG), each of which containing 100 neurons. The temporal quantization error for the trained maps has been used as performance index: first, the receptive field is computed as the average sequence of a fixed time span (here, 30 steps) for which a neuron is winner. The quantization error then calculates the variances for the deviation of the current pattern sequence from the winner's receptive field separately for each time entry. All models got a number of $1.5^5$ presentations for training, and the same number was taken for obtaining the quantization errors, of which the results are shown in figure 2. The graphs marked with $(*)$ show additional results from Voegtlin [15]. Model parameters have been chosen optimal for each approach. Different types of dynamics can be observed: (a) strongly oscillating errors for SOM and NG that do not model temporal context and thus follow the quasi-periodicity of the presented series, (b) the anti-cyclical behavior of the recurrent SOM reflects some temporal representation at the expense of other parts of the context[2], (c) and the increasing and weakly oscillating error curves displayed for the context processing RecSOM, HSOM-S and MNG. These latter models show similar properties with

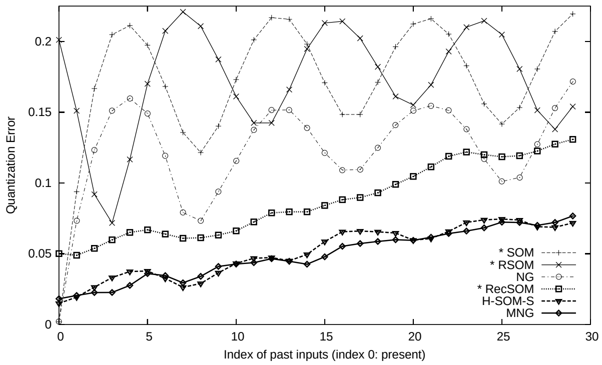

Figure 2: Temporal quantization errors for different models for the Mackey-Glass series.

---

[2] Recurrent SOM (RSOM) integrates over the vectors instead of only the distances of weights and sequence entries [14]. For 1D time series as above, the result is similar to TKM; it can be computed that the optimal encoding of sequences coincides for TKM and RSOM.

respect to their growing uncertainty of the past, but RecSOM requires an additional dimensionality of 100 for each neuron and two more context learning parameters compared to standard SOM; HSOM-S neurons only need to store the compact 2D-coordinate of the last winning location on the map, and the model merely requires one context control parameter, set to $\beta = 3\%$ influence here – nevertheless, this compact context representation produces the same small quantization error as RecSOM; MNG, which uses the fractal encoding instead of global map activity information, has got a total dimension of 2 for each neuron, and the parameter $\beta$ was set to $= 0.75$, while the entropy controlled $\alpha$ started at $\alpha = 0$, taking a final value of $\alpha = 0.82$. The trained MNG context values cover the subinterval $(0.7, 1.15)$ of the weights interval $(0.4, 1.35)$. All the neurons are well spread in the context vs. weight plane, thus, exploiting the maximum possible space for context representation pretty well, and fractal encoding proves to be adequate in this case.

**Binary Automaton:** A further task focused on learning the most likely binary sub-words of a 0/1-sequence generated by a binary automaton with the transition probabilities $P(0|1) = 0.4$ and $P(1|0) = 0.3$. The two questions of interest are: how long are the sequences that still lead to unambiguous winner selection and how many neurons become specialized at all? Figure 3 shows the resulting receptive fields for MNG neurons as a tree, compared to the 100 most likely sequences produced by the automaton. Training parameters were $\beta = 0.45$ and $\eta = 0.025$, and the initial value of $\alpha = 0$ was steered to a final value of $0.37$ after $10^6$ pattern presentations. Due to the training's intrinsic moving target on-context-specialization problem, two neurons fell into idle state. As expected for zero context influence in the beginning, the weight specialization to almost crisp values of $0$ and $1$ could be observed; the contexts, however, fill the interval $(0, 1)$ with a structured density pattern. As a result, many neurons developed their own receptive fields, with the exception of transient neurons, indicated by bullets on the interior tree nodes, for which the descendants represent still longer sequences; after all, a total number of 63 longest words can be discriminated by MNG. For comparison, specialization dynamic for HSOM-S is shown in figure 3 with finally 6% of context influence; this model requires many interior nodes for representing both the possible context transitions and idle nodes that are used for the separation of incompatible adjacent regions on the 2D hyperbolic neural grid, producing a total number of 28 distinct longest represented sequences, but the overall map utilization is smaller than for the fractal encoding scheme.

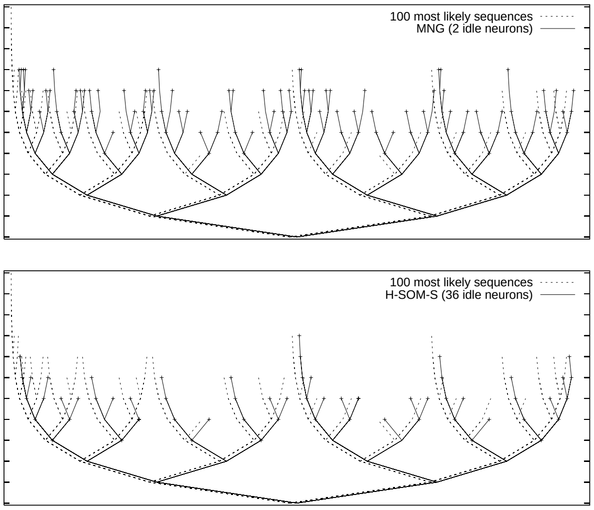

Figure 3: Upper panel, MNG receptive fields for the binary automaton; lower panel, HSOM-S receptive fields.

**Physiological 3-Variate Series (B-1991):** Real life data has been taken from `physionet.org`. The first column contains the heart rate, the second the chest volume, and the third is the blood oxygen concentration. By spline interpolation the number of 32351 samples has been increased to 64701 in order to calculate a smooth order 4 polynomial first derivative that purges the linear trends and value shifts in each component. Averages have been subtracted from the columns and their variances were normalized. Finally, a logarithmic transform $x' = \operatorname{sgn}(x) \cdot \log(|x| + 1)$ has been computed for each column to account for the sharp peaks within the otherwise moderate dynamics. For this preprocessed series, training has been performed with a 20-fold repetition of the series using the 3D inputs. Figure 4 shows the quantization errors for NG, HSOM-S, and MNG with 617 neurons each: the error is the average of the three variances that have been obtained separately for each column the same way as for the Mackey-Glass series. As expected, the NG without context is the worst temporal quantizer, MNG works pretty well until 4 steps in the past and it is then outperformed by HSOM-S. Since the data set is of roughly unimodal density, a curve of concatenated spatial loops around the origin **0**, a long context integration for MNG falls into the center of the dynamic. Thus, distinct clusters cannot easily emerge for larger contexts, making MNG asymptotically resulting in the standard NG quantization performance. HSOM-S does not suffer from this problem, and remains on a smaller error level, because its context representation does not require a folding of historic information into the weight domain, but uses the larger domain of indices of neurons instead. The performance of MNG is not quite as good as HSOM-S in this case.

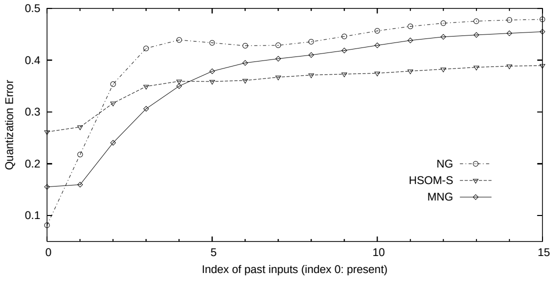

Figure 4: Temporal quantization errors for the preprocessed B-1991 data.

# 5 Conclusions

The mining of sequential data is a challenging task for various interesting areas of application, such as finance, speech processing, or bio-informatics. It is therefore worthwhile transferring powerful data mining tools like Kohonen's self-organizing map, HSOM, or neural gas from the domain of finite dimensional vector spaces to these more complex data structures.

We have discussed various popular recursive self-organizing models for sequence processing in this paper. As pointed out, the models differ with respect to the crucial part of how temporal context is represented in the map. Mainly two types of representations are used: sequence contexts are either explicitly modeled by storing the location of the last winner in SOMSD and HSOM-S, or by keeping more subtle information like the map activation profile in RecSOM; the range of context representation in these models is a different and possibly much larger set than the range of sequence entries. Alternatively, sequences can be directly stored in the weight space using fractal codes. We have pointed out that this context representation is used by TKM, but that it is rather an implicit side effect of the Kohonen training, and only limited context information can actually be represented by TKM. We have proposed an alternative, MNG, which is based on the same idea of context representation, but for which training leads to better adaptation because of the model's explicit notion of the context.

As demonstrated in experiments, the representation capability of MNG is competitive to HSOM-S for interesting data sets, although the latter one uses a richer representation. Moreover, since context in MNG does not refer to the lattice structure, it can be directly combined with both SOM and NG for avoiding potential topological mismatches, and thus produce good solutions with less idle neurons than HSOM-S. Naturally, MNG has only a limited capacity e.g. if space-filling continuous-valued time-series are dealt with, and this is due to the embedding of context in the low-dimensional weight vector domain.

This investigation constitutes a step further towards the design and understanding of canonical, simple, though powerful self-organizing models for the processing of complex data structures, and there are more steps to be taken.

# References

[1] G. Barreto and A. Araújo. Time in self-organizing maps: An overview of models. *International Journal of Computer Research*, 10(2):139–179, 2001.

[2] H. U. Bauer and T. Villmann. The GSOM-algorithm for growing hypercubical output spaces in self-organizing maps. In *Proceedings of WSOM'97, Workshop on Self-Organizing Maps, Espoo, Finland, June 4-6*, pages 286–291. Helsinki University of Technology, Neural Networks Research Centre, Espoo, Finland, 1997.

[3] G. Chappell and J. Taylor. The temporal Kohonen map. *Neural Networks*, 6:441–445, 1993.

[4] M. Hagenbuchner, A. Sperduti, and A. Tsoi. A self-organizing map for adaptive processing of structured data. *IEEE Transactions on Neural Networks*, 14:491–505, 2003.

[5] B. Hammer, A. Micheli, and A. Sperduti. A general framework for self-organizing structure processing neural networks. Technical Report TR-03-04, Dipartimento di Informatica, Università di Pisa, 2003.

[6] B. Hammer, M. Strickert, A. Micheli, and A. Sperduti. A general framework for unsupervised processing of structured data. *Neurocomputing*, to appear.

[7] S. Kaski. Bankruptcy analysis with self-organizing maps in learning metrics. *IEEE Transactions on Neural Networks*, 12:936–947, 2001.

[8] T. Kohonen. *Self-Organizing Maps*. Springer-Verlag, Berlin, 2001.

[9] T. Martinetz, S. Berkovich, and K. Schulten. "Neural-gas" network for vector quantization and its application to time-series prediction. *IEEE Transactions on Neural Networks*, 4:558–569, 1993.

[10] T. Martinetz and K. Schulten. Topology representing networks. *Neural Networks*, 7(3):507–522, 1993.

[11] H. Ritter. Self-organizing maps on non-euclidean spaces. In S. Oja, E. & Kaski, editor, *Kohonen Maps*, pages 97–110. Elsevier, Amsterdam, 1999.

[12] M. Strickert, T. Bojer, and B. Hammer. Generalized relevance LVQ for time series. In *Artificial Neural Networks - ICANN'2001*, pages 677–683. Springer, 2001.

[13] M. Strickert and B. Hammer. Unsupervised recursive sequence processing. In M. Verleysen, editor, *European Symposium on Artificial Neural Networks'2003*, pages 433–439. D-side publications, 2003.

[14] M. Varsta, J. del R. Milan, and J. Heikkonen. A recurrent self-organizing map for temporal sequence processing. In *Proc. ICANN'97, 7th International Conference on Artificial Neural Networks*, volume 1327 of *Lecture Notes in Computer Science*, pages 421–426. Springer, Berlin, 1997.

[15] T. Voegtlin. Recursive self-organizing maps. *Neural Networks*, 15(8-9):979–991, 2002.