Time series prediction by estimating Markov probabilities through topology preserving maps

G. Dangelmayr[a], S. Gadaleta[a], D. Hundley[b] and M. Kirby[a]

[a]Colorado State University, Mathematics Department,
Fort Collins, Co 80523, USA
[b] Whitman College, Mathematics Department,
Walla Walla, WA 99362, USA

Proc. SPIE Vol. 3812, Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation II, pages 86-93, Eds. B. Bosacchi and D. B. Fogel and J.C. Bezdek (1999)

ABSTRACT

Topology preserving maps derived from neural network learning algorithms are well suited to approximate probability distributions from data sets. We use such algorithms to generate maps which allow the prediction of future events from a sample time series. Our approach relies on computing transition probabilities modeling the time series as a Markov process. Thus the technique can be applied both to stochastic as well as to deterministic chaotic data and also permits the computation of 'error bars' for estimating the quality of predictions. We apply the method to the prediction of measured chaotic and noisy time series.

Keywords: Time series prediction, transition probabilities, topology preserving maps, Markov processes

1. INTRODUCTION

The concepts of nonlinear dynamical systems have not only shown to be useful for analyzing specific systems of ordinary differential equations or iterated maps, but also offered new techniques for time series analysis. A variety of experiments have shown that in many instances a recorded time series can be viewed as driven by a deterministic dynamical system with a low dimensional chaotic attractor. This perspective is particularly appealing since quantities like the correlation dimension, Lyapunov exponents or, more generally, the Kolmogorov-Sinai entropy then become objective measures for characterizing the data set.[1–3] Moreover, the time delay method[4,5] has become a powerful tool for calculating these and other quantities of interest, regardless of whether the time series was obtained from numerical simulations or from experiments. It is therefore not surprising that the methods of nonlinear dynamical systems had a strong influence on the development of new concepts in the field of time series analysis.

One of the major goals of time series analysis is prediction, i.e., forecasting the future development of the series from observations in the past. The first approach to predicting deterministic dynamical systems on the basis of the time delay method relies on nearest neighbors in the embedding space and a local linear model derived by fitting a hyperplane through the data.[6,7] Later this method was refined by incorporating a singular value decomposition.[8] In a broader context, the task of prediction may be viewed as a mapping problem, namely, to approximate a mapping that associates to a vector of previous observations (the source vector) another vector of future events (the target vector). Such mapping problems have been successfully solved by means of neural network learning algorithms, specifically via the backpropagation algorithm[9] which is designed to learn a smooth interpolation between the source space and the target space from a training sample. In the meantime variants of the backpropagation algorithm, suitable for time series analysis, have been developed and applied to experimental data.[10,11]

However, most of the nonlinear approaches to time series analysis cited above suffer from the following drawbacks. First, if the series is generated by a deterministic dynamical system, the methods do not allow for a geometrical description of the dynamics which yields information about the structure of the chaotic attractor. In particular,

Further author information:
G.D.: E-mail: gerhard@math.colostate.edu
S.G.: E-mail: sabino@math.colostate.edu
D.H.: E-mail: hundledr@whitman.edu
M.K.: E-mail: kirby@math.colostate.edu

the dimensionality of the attractor and its embedding in a smooth, nonlinear manifold is not addressed. In general, embeddings in Euclidean spaces may be viewed as encapsulating the nonlinear manifold. The intrinsic dimensionality of the dynamics is in general smaller then the dimension of the space needed to correctly embed the dynamics. (For example, the intrinsic dimension of a periodic orbit is one, but generically a three–dimensional space is needed for its correct embedding.) The problem of reducing the dimensionality of dynamical systems is extremely important and is an active area of research.[12,13] The second drawback concerns the assumption of a deterministic dynamical system itself, since experimenters usually work with noisy data instead. Moreover, many time series, like for instance foreign currency exchange rates or the heart beat, seem not to fit into the class of deterministic dynamical systems, but rather appear to be generated by a stochastic process. It is therefore desirable to have an algorithm which on the one hand can be applied to deterministic as well as to stochastic time series, and on the other hand allows to fit a smooth manifold through the data, regardless of whether the time series is driven deterministically or stochastically.

In this article we propose vector quantization techniques based on neural network learning algorithms as tools for time series analysis with special emphasis laid on prediction. In general such algorithms characterize statistically distributed data vectors by mapping them to a set of weight vectors defined in the same space, but parametrized by multi-indices on a lattice of possibly lower dimension. The learning process is designed such that the mapping is topology preserving, i.e., neighboring data vectors are mapped to neighboring indices. We call such a map a topology preserving map (TPM). If the index dimension is smaller than the dimensionality of the data, the weight vector distribution can be viewed as approximation of a low dimensional, nonlinear manifold that optimally fits to the statistical distribution of the data. In this case, TPM plays the role of a nonlinear transformation to a principal manifold, as opposed to the linear principal components derived via singular value decomposition. On the other hand, if the dimensions of data and index–lattice are the same, the weight vectors approximate the full distribution of the data and typically much fewer weights than data are needed. In both cases TPM serves for data compression. Alternatively, if one chooses more weight vectors than data, TPM serves for interpolation. We utilize two kinds of algorithms for learning a TPM: Kohonen's algorithm[14] and the Neural-Gas algorithm.[15,16]

2. METHOD

2.1. Embedding of time series

Let $x(t)$ $(t = 0, 1, 2, \ldots)$ be a scalar time series which has to be analyzed. Choosing a delay time $\tau \in \mathbf{N}$ and an embedding dimension n we form the usual lag vector $\mathbf{x}(t)$ as

$$\mathbf{x}(t) \;=\; (x(t), x(t-\tau), \ldots, x(t-(n-1)\tau)) \;\in\; \mathbf{X} \;\equiv\; \mathbf{R}^n \;. \tag{1}$$

Many different methods exist to determine sufficient values for n and τ from an observed time series. n can for example be computed with the method of false nearest neighbors developed by Abarbanel and Kennel.[17] Algorithms to determine τ rely mostly on the computation of generalized dimensions (see e.g.[18]).

Assume we wish to predict, given the history $\mathbf{x}(t)$, the next m future events. In the context of mappings this means that we have to associate to $\mathbf{x}(t)$ a vector

$$\mathbf{y} \;=\; (y_1, \ldots, y_m) \;\in\; \mathbf{Y} \;\equiv\; \mathbf{R}^m \;, \tag{2}$$

which should be as close as possible to the actual values $(x(t+\tau), \ldots, x(t+m\tau))$. Our intention is, however, not to approximate a deterministic mapping $\mathbf{x}(t) \to \mathbf{y}$, instead, we treat $\mathbf{x}(t)$ and $\mathbf{y}$ as stochastic vectors and approximate both the probability density $p(\mathbf{x})$ and the transition probability $p(\mathbf{y}|\mathbf{x})$. Here, $p(\mathbf{x})\varepsilon^n$ is the probability of finding $\mathbf{x}(t)$ in a small cube of size ε centered around $\mathbf{x}$ and $p(\mathbf{y}|\mathbf{x})\varepsilon^m$ is the probability that $(x(t+\tau), \ldots, x(t+m\tau))$ is located in a small cube centered around $\mathbf{y}$ if $\mathbf{x}(t)$ is known. Prediction is then made, based on the approximated transition probabilites, either by a maximum likelihood decision, referred to as maximum likelihood predictor (MLP), or by a random selection of $\mathbf{y}$ according to the transition probability, referred to as random Markov predictor (RMP). Both predictors are used in our simulations. For short term predictions MLP usually gives better results whereas for generating longer time sequences (iterated prediction) RMP yields better performance on average. For stochastic or strongly noisy systems the approximated transition probabilities themselves are of particular interest since they give information about the underlying process.

Note that there are no specific model assumptions in this formulation except stationarity of the data. Thus we can treat deterministic non–chaotic as well as deterministic chaotic and stochastic systems. In the first case the

true transition probabilities are delta functions, in the second case they are determined by the positive Lyapunov exponents and in the third case we have genuine transition probabilities.

In principle the two tasks of approximating $p(\mathbf{x})$ and $p(\mathbf{y}|\mathbf{x})$ can be combined, but we found it computationally more convenient to approximate first $p(\mathbf{x})$ and then $p(\mathbf{y}|\mathbf{x})$.

2.2. Compression of embedded data

After the data is correctly embedded we approximate its probability distribution with a topology preserving map. In Kohonen's approach the TPM is represented by an array of formal neurons, arranged on a finite, k–dimensional lattice $\mathbf{L}$ with $k \leq n$. To each lattice point $\mathbf{r} \in \mathbf{L}$ a weight vector $\mathbf{w}(\mathbf{r}) \in \mathbf{X}$ is associated. The total number of lattice points, $M = \sharp\mathbf{L}$, should satisfy $1 << M \leq T$ (often $M/T << 1$ is sufficient), where T is the total number of data vectors

$$\mathbf{x} \in \mathbf{X}_T = \{\mathbf{x}(t) \mid \tau \leq t < T + \tau\} \subset \mathbf{X} , \tag{3}$$

which are used for the training. The dimensionality k of $\mathbf{L}$ may be either selected by trial and error or optimally fixed before the learning by means of techniques such as the topographic product.[19] k should be large enough so that the existence of a topology preserving map from $\mathbf{X}_T$ to $\mathbf{R}^k$ is assured. To $\mathbf{x} \in \mathbf{X}$ and a particular weight distribution $\mathbf{W} = \{\mathbf{w}(\mathbf{r}) \mid \mathbf{r} \in \mathbf{L}\} \subset \mathbf{X}$ the mapping $\mathbf{x} \to \mathbf{r}_0(\mathbf{x}; \mathbf{W})$ is defined by the condition

$$\|\mathbf{x} - \mathbf{w}(\mathbf{r}_0(\mathbf{x}; \mathbf{W}))\| = \min\{\|\mathbf{x} - \mathbf{w}(\mathbf{r})\| \mid \mathbf{w}(\mathbf{r}) \in \mathbf{W}\} . \tag{4}$$

The desired weight distribution $\mathbf{W}$ for the data (the TPM) is then determined by the following learning algorithm. Choose a sequence $\mathbf{x}_\nu$ from the training set, $\mathbf{x}_\nu \in \mathbf{X}_T$, $\nu = 0, 1, 2, \ldots$, and update the weight distribution $\mathbf{W}_\nu = \{\mathbf{w}_\nu(\mathbf{r})\}$ at each learning step ν by the rule

$$\mathbf{w}_{\nu+1}(\mathbf{r}) - \mathbf{w}_\nu(\mathbf{r}) = \varepsilon_\nu h(\|\mathbf{r} - \mathbf{r}_0(\mathbf{x}_\nu; \mathbf{W}_\nu)\|, \sigma_\nu)(\mathbf{x}_\nu - \mathbf{w}_\nu(\mathbf{r})) , \tag{5}$$

where the initial distribution $\mathbf{W}_0$ is arbitrary. Here, the learning is controlled by $\varepsilon_\nu > 0$ and $\sigma_\nu > 0$ which both decrease to zero for large ν, and $h(\xi, \sigma)$ is an unimodal function of ξ with a peak at zero and width σ. A convenient choice is a Gaussian, $h(\xi, \sigma) = \exp(-\xi^2/2\sigma^2)$. The update rule (5) can be interpreted as a Hebb–type synaptic modification with active loss term. The sequence $\mathbf{x}_\nu$ of input presentations may be chosen at random from $\mathbf{X}_T$ or simply be defined by running repeatedly through the sequence of lag vectors $\mathbf{x}(t)$. It is worth noting that the TPM–learning algorithm is very suitable for parallelization, since the updates involve only neighborhoods of the 'winners' $\mathbf{r}_0$. It is therefore possible to present simultaneously a number of input vectors, provided their pairwise distances are sufficiently large.

The outcome $\mathbf{W}$ ($\mathbf{W}_\nu \to \mathbf{W}$) of the learning process (5) yields a weight distribution $\mathbf{w}(\mathbf{r}) \in \mathbf{W}$ which approximates the point density of the training set $\mathbf{X}_T$.[14] In addition, $\mathbf{W}$ is topologically ordered in the sense that neighboring vectors $\mathbf{x} \in \mathbf{X}$ are mapped to neighboring indices $\mathbf{r} \in \mathbf{L}$ via $\mathbf{r}_0(\mathbf{x}; \mathbf{W})$. According to the first property, we define the 'receptive fields' $A(\mathbf{r})$ by

$$A(\mathbf{r}) = \{\mathbf{x} \in \mathbf{X} \mid \mathbf{r} = \mathbf{r}_0(\mathbf{x}; \mathbf{W})\} \tag{6}$$

and note that in the ideal case $\mathbf{w}(\mathbf{r}) = < \mathbf{x} | \mathbf{x} \in A(\mathbf{r}) >$ with the brackets denoting averages. During the learning process we also approximate the probabilities $P(\mathbf{r})$ of finding $\mathbf{x}$ in $A(\mathbf{r})$ by $\mu_\nu(\mathbf{r})/\nu$ for large ν, where $\mu_\nu(\mathbf{r})$ is the number of times $\mathbf{r}$ was selected as a winner during the past ν input presentations. $P(\mathbf{r})$ represents an approximation to $p(\mathbf{x})$, thus averages of functions $f(\mathbf{x})$ are computed as

$$< f(\mathbf{x}) > \simeq \sum_{\mathbf{r} \in \mathbf{L}} P(\mathbf{r}) f(\mathbf{w}(\mathbf{r})) . \tag{7}$$

Kohonen's algorithm requires the choice of the lattice dimension in advance. Technically this dimension should be equal to the dimension of the embedded data set. Since this dimension might not be an integer number and, moreover, is usually unknown, this requirement can seriously limit the perfomance of Kohonen's map. An improvement is offered here by the Neural-Gas algorithm in which reference vectors are not constraint to a lattice. For details on this algorithm we refer to the literature.[15,16]

2.3. Computation of transition probabilities

When the weights $\mathbf{w}(\mathbf{r})$ are computed, the next step is to approximate the transition probabilites $p(\mathbf{y}|\mathbf{x})$. Here we decompose the space $\mathbf{Z} = \mathbf{Y} \times \mathbf{X}$ of lag vectors of length $m+n$ as $\mathbf{Z} = \mathbf{Y} \times \cup_{\mathbf{r}} A(\mathbf{r})$ and replace the $\mathbf{x}$–component of $\mathbf{z} = (\mathbf{y}, \mathbf{x}) \in \mathbf{Z}$ by its 'winner weight' $\mathbf{w}(\mathbf{r})$, determined via $\mathbf{x} \in A(\mathbf{r})$. For each $\mathbf{r} \in \mathbf{L}$, then, a second TPM $\mathbf{V}(\mathbf{r}) = \{\mathbf{v}(\mathbf{s}|\mathbf{r}) \mid \mathbf{s} \in \mathbf{L}_c(\mathbf{r})\} \subset \mathbf{Y}$ is learned. Here, $\mathbf{L}_c(\mathbf{r})$ is again a finite, now k_c–dimensional integer lattice with $k_c \leq m$ that consists of $M_c(\mathbf{r}) = \sharp\mathbf{L}_c(\mathbf{r})$ lattice points. The total number M_c of weights reserved for the transition lattice, $M_c = \sum_{\mathbf{r}} M_c(\mathbf{r})$, is distributed over the individual lattices $\mathbf{L}_c(\mathbf{r})$ in proportion to the probabilities $P(\mathbf{r})$, i.e., $M_c(\mathbf{r})/M_c \simeq P(\mathbf{r})$. For a given weight distribution $\mathbf{V}(\mathbf{r})$ the mapping $\mathbf{y} \to \mathbf{s}_0(\mathbf{y}|\mathbf{V}(\mathbf{r}))$ is then defined as before by

$$\|\mathbf{y} - \mathbf{v}(\mathbf{s}_0|\mathbf{V}(\mathbf{r}))\| \;=\; \min\{\|\mathbf{y} - \mathbf{v}(\mathbf{s}|\mathbf{r})\| \;\mid\; \mathbf{v}(\mathbf{s}|\mathbf{r}) \in \mathbf{V}(\mathbf{r})\} \tag{8}$$

which corresponds to a decomposition of $\mathbf{Y}$ into a set of receptive fields

$$B(\mathbf{s}|\mathbf{r}) \;=\; \{\mathbf{y} \in \mathbf{Y} \;\mid\; \mathbf{s} = \mathbf{s}_0(\mathbf{y}|\mathbf{V}(\mathbf{r}))\}\;. \tag{9}$$

The desired family of TPM's $\{\mathbf{V}(\mathbf{r}) \mid \mathbf{r} \in \mathbf{L}\}$ for the transition probabilities is determined by a learning algorithm based on a training set that consists of enlarged lag vectors

$$\mathbf{z}(t) \;=\; (x(t+m\tau, \ldots, x(t), x(t-\tau), \ldots, x(t-n\tau)) \;\in\; \mathbf{Z}_T \;\subset\; \mathbf{Z}\;. \tag{10}$$

Here we define, for each index $\mathbf{r}$, the training set $\mathbf{Y}_T(\mathbf{r})$ as the set of all $\mathbf{y}$–components in $\mathbf{z} = (\mathbf{y}, \mathbf{x}) \in \mathbf{Z}_T$ for which $\mathbf{x} \in A(\mathbf{r})$. Then, choosing a learning sequence $\mathbf{y}_\nu \in \mathbf{Y}_T(\mathbf{r})$, at each learning step ν the distribution $\mathbf{v}_\nu(\mathbf{s}|\mathbf{r}) \in \mathbf{V}_\nu(\mathbf{r})$ is updated according to

$$\mathbf{v}_{\nu+1}(\mathbf{s}|\mathbf{r}) - \mathbf{v}_\nu(\mathbf{s}|\mathbf{r}) \;=\; \varepsilon_\nu^{'} h'(\|\mathbf{s} - \mathbf{s}_0(\mathbf{y}_\nu/\mathbf{V}_\nu(\mathbf{r}))\|, \sigma_\nu^{'})(\mathbf{y} - \mathbf{v}_\nu(\mathbf{s}|\mathbf{r}))\;, \tag{11}$$

where $\varepsilon_\nu^{'}$, $\sigma_\nu^{'}$ and h' are analogous to ε_ν, σ_ν and h.

We note that the approximation of the probability densities can be combined with projections to principal manifolds. This is not a priori possible for other algorithms based on hidden Markov dynamics and their implementation via backpropagation.[20,21] Moreover, the TPM–approach appears computationally more efficient than the latter and also has the advantage that averages are easily computed via (7) and the analogous formula

$$< g(\mathbf{y})|\mathbf{x} > = \sum_{\mathbf{s} \in \mathbf{L}_c(\mathbf{r})} P(\mathbf{s}|\mathbf{r}_0(\mathbf{x}; \mathbf{W}))g(\mathbf{v}(\mathbf{s}|\mathbf{r}_0(\mathbf{x}; \mathbf{W})))\;. \tag{12}$$

The transition probabilities $P(\mathbf{s}|\mathbf{r})$ which approximate $p(\mathbf{y}|\mathbf{x})$ are calculated during learning $\mathbf{V}(\mathbf{r})$ in the same way as the probabilities $P(\mathbf{r})$:

$$P(\mathbf{s}|\mathbf{r}) = \frac{\text{Number of } \mathbf{y} \text{ in } B(\mathbf{s}|\mathbf{r})}{\text{Number of } \mathbf{x} \text{ in } A(\mathbf{r})}. \tag{13}$$

Clearly, these approximations can be done with any vector quatization technique. For the applications in Section 3 we choose $m = 1$.

2.4. Forecasting

Given the approximations $P(\mathbf{s}|\mathbf{r})$, forecasting is generated by setting $\mathbf{y}(\mathbf{x}) = \mathbf{v}(\mathbf{s}_m(\mathbf{r})|\mathbf{r})$ for $\mathbf{x} \in A(\mathbf{r})$, where $\mathbf{s}_m(\mathbf{r})$ is determined either by the maximal transition probability (MLP),

$$P(\mathbf{s}_m(\mathbf{r})|\mathbf{r}) \;=\; \max\{P(\mathbf{s}|\mathbf{r}) \;\mid\; \mathbf{s} \in \mathbf{L}_c(\mathbf{r})\}\;, \tag{14}$$

or chosen at random (RMP). The averages (12) can then be used to compute error bars $\Delta(\mathbf{r})$ for the predicted values $\mathbf{y}(\mathbf{x})$,

$$\Delta^2(\mathbf{r}) \;=\; \sum_{\mathbf{s} \in \mathbf{L}_c(\mathbf{r})} P(\mathbf{s}|\mathbf{r})\, \|\mathbf{v}(\mathbf{s}|\mathbf{r}) - \mathbf{v}(\mathbf{s}_m(\mathbf{r})|\mathbf{r})\|^2\;, \tag{15}$$

which depend only on the data and not on specific assumptions like $\mathbf{x}$–dependent Gaussian error distributions as was proposed in.[22]

As an error measure for the predictions we use the *normalized mean square error* (NMSE) defined by

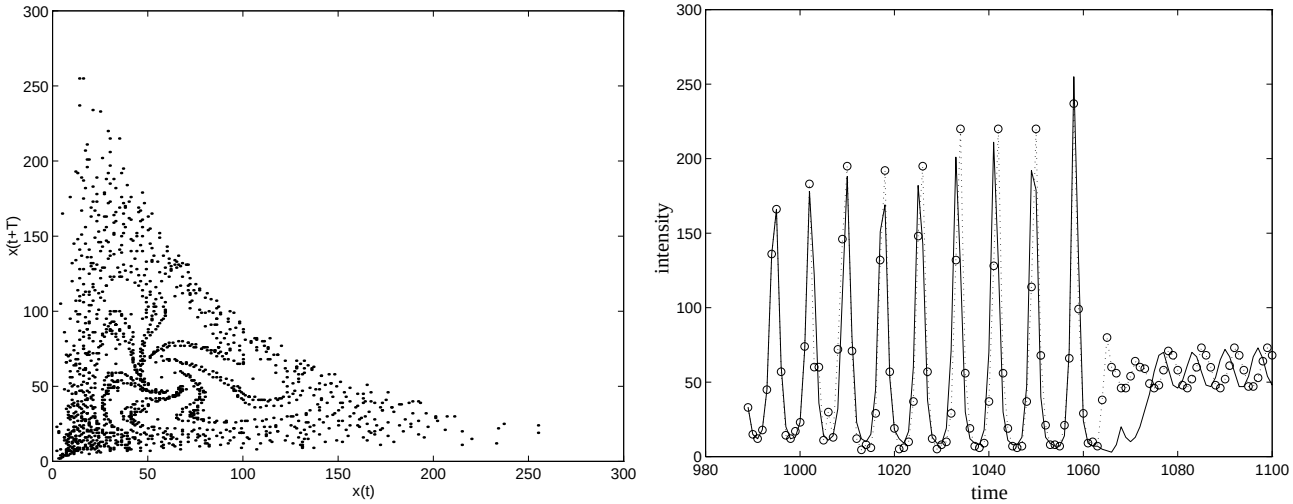
$$\mathrm{NMSE}(N) = \frac{1}{\sigma_T^2 \cdot N} \sum_{k \in T} (x_k - \hat{x}_k)^2, \tag{16}$$

where x_k is the true value, $\hat{x}_k$ its prediction, T the test data containing N points and σ_T^2 its variance. We note that for the applications in Section 3 we exclusively tested the predictions on a data set T which was not contained in the training set from which $\mathbf{X}$ was constructed.

3. RESULTS

3.1. Laser-data

As a first data set we used the standard laser time series which was used in the Santa Fe time series prediction competition.[23] The training data consists of 1000 points. We embedded the data with $n = 4$ and $\tau = 2$. A 2-dim projection of the embedded data is shown in Fig. 1a). We used a 2-dim 25×25 Kohonen net to approximate $p(\mathbf{x})$ and 1000 reference vectors to approximate $p(\mathbf{y}|\mathbf{x})$. The test data consisted of 100 data points starting at the end of the training set. In Fig. 1b) we show an iterated prediction over the whole test set (dots) together with the true data.



a) b)

Figure 1: a) Two-dimensional projection of an embedding of the laser-data with $n = 4$ and $\tau = 2$. b) Iterated prediction over the entire test-set consisting of 100 points (NMSE=0.2473). –o–: predicted data, ——: true data.

The given data set of 1000 points is too short to approximate true transition probabilities. We therefore used a longer data set of 25.000 points (see also[23]) from which 12.000 points where used as a training set. As a test set we used the points 2270-2310 of the remaining set. Fig. 2a) shows the point-prediction (dots) on the test set together with the true data. We used a 15×15 Kohonen net in this application. Fig. 2b) shows the corresponding approximated transition probabilites for the first 20 points together with the true values (x) and the predictions with maximal probability (o). The NMSE of the prediction was 0.0701. The width of the transition probability distribution is a measure of the quality of the prediction. Clearly, predictions with a broader distribution have a larger error.

3.2. EEG-data

As an example of a time series that is more difficult to annalyze we used data from an EEG measurement which was made available to us by C. Anderson.[24] The training set consisted of 2500 data points of which the first 2000

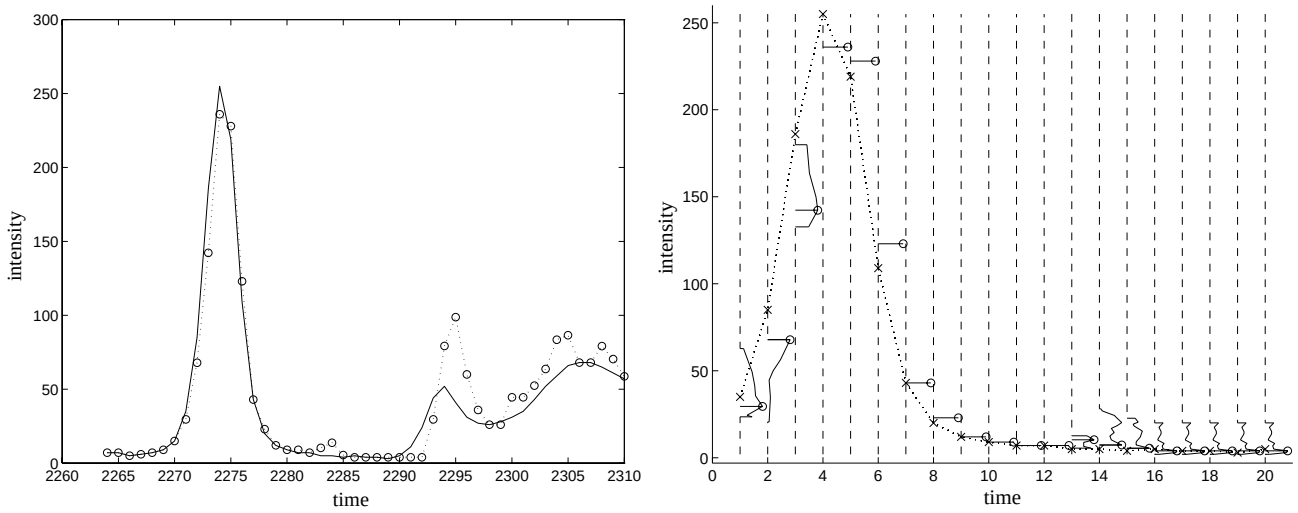


Figure 2: a) Point-prediction (NMSE=0.0701) of 40 points of the larger laser-data set. b) The approximated transition probabilites for the first 20 points of the test set. x: true data, o: prediction with maximal transition probability.

where used for training. The data was embedded with $n = 5$ and $\tau = 10$. Fig. 3a) shows the data set and Fig. 3b) a two-dimensional projection of the embedded data. The embedded data show no appearent structure which indicates stochasticity. To approximate the transition probability distribution we used the Neural-Gas algorithm with 225 reference vectors for the first quantization and 675 reference vectors for the second quantization. The resulting iterated prediction (NMSE=1.2853) of a test set consisting of 100 points is shown in Fig. 4. Even though the NMSE is larger then one we see that the qualitative structure of the time series is reproduced.

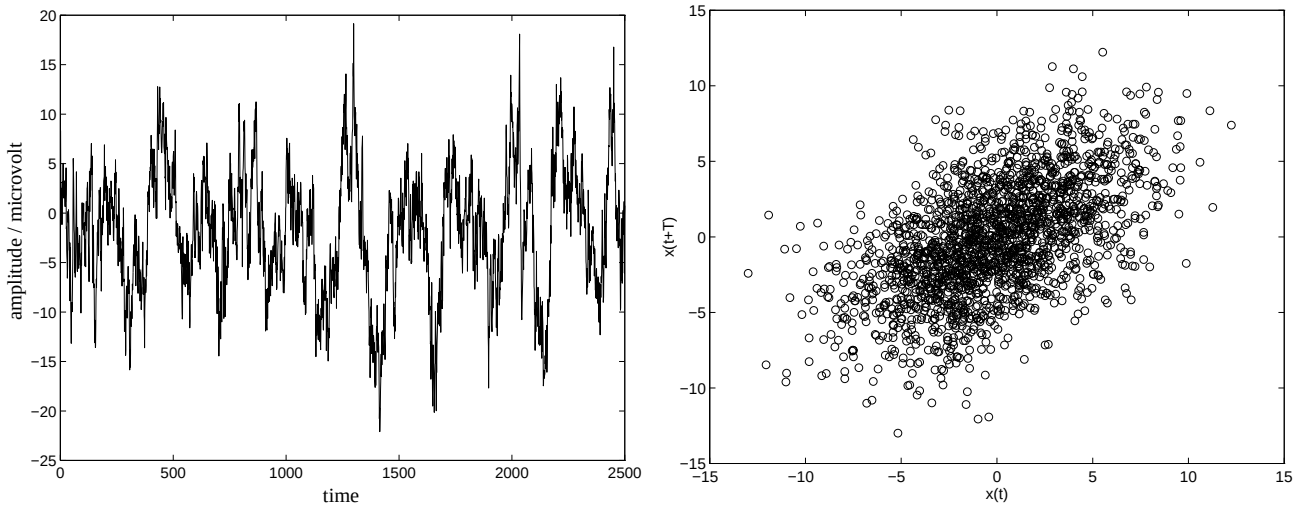


Figure 3: a) The complete EEG-data set. b) Embedding of the first 2000 points of the EEg-data set with $n = 5$ and $\tau = 10$.

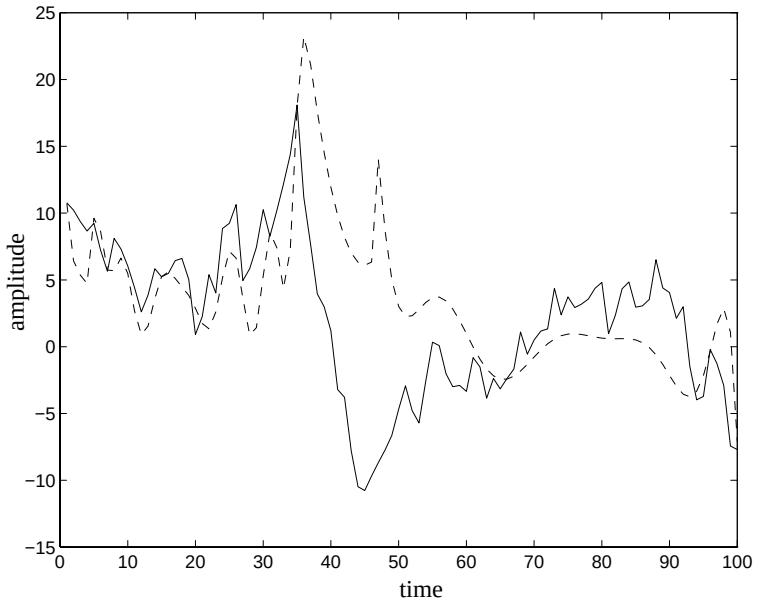


Figure 4: Iterated prediction of the EEG data consisting of 100 points (NMSE=1.2853). ——: True data, - - - : Prediction.

4. CONCLUSIONS AND FUTURE WORK

Topology preserving maps are seen to provide an effective approach for estimating transition probabilities and to permit the construction of stochastic as well as deterministic models of time series and their prediction. In future work, the role of the ordering in the index set and its connection with the *stability* of the prediction in the presence of noise

need to be elucidated. Also, the use of such maps for modeling the dynamics on a low-dimensional manifold should be investigated. In particular, approaches for optimizing the topology of the weights for a given set of data should be examined. For example the application of a spatial-temporal clustering,[25] instead of a pure spatial clustering, of the embedded data could possibly lead to improved results. Such a clustering uses derivative information of the embedded data, in addition to the spatial information, to form a density approximation and was shown to improve the results of a dimensionality reduction algorithm for nonlinear systems.[25]

ACKNOWLEDGMENTS

We would like to acknowledge support for this research under grants from the National Science Foundation DMS-9505863, INT-9513880 and DOD-USAF Office of Scientific Research F49620-99-1-0034. Preliminary work was supported under NSF grant ECS-9312092.

REFERENCES

1. P. Grassberger and I. Procaccia, "Characterization of strange attractors," *Physical Review Letters* **50**, pp. 346–349, 1983.
2. J.-P. Eckmann, S. Kamphorst, D. Ruelle, and S. Ciliberto, "Liapunov exponents from time series," *Physical Review A* **34**, pp. 4971–4979, 1986.
3. J. Farmer, "Chaotic attractors of an infinite-dimensional dynamical system," *Physica D* **4**, pp. 366–393, 1982.
4. N. Packard, J. Crutchfield, J. Farmer, and R. Shaw, "Geometry from a time series," *Physical Review Letters* **45**, pp. 712–716, 1980.
5. F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence, Lecture Notes in Math. Vol. 898*, D. Rand and L.-S. Young, eds., pp. 366–381, Springer, Heidelberg-New York, 1981.
6. J. Farmer and J. Sidorowich, "Prediciting chaotic time series," *Physical Review Letters* **59**, pp. 845–848, 1987.
7. M. Casdagli, "Chaos and deterministic versus stochastic nonlinear modeling," *J. Roy. Stat. Soc. B* **54**, pp. 303–328, 1991.
8. T. Sauer, "Time series prediction using delay coordinate embedding," in *Time Series Prediction: Forecasting the Future and Understanding the Past*, A. Weigend and N. Gershenfeld, eds., pp. 175–193, Addison-Wesley, Reading, 1994.
9. D. Rummelhart and J. McClelland, eds., *Parallel Distributed Processing, Vol. I*, MIT-Press, Cambridge, MA, 1986.
10. K. Lang, A. Waibel, and G. Hinton, "A time-delay neural network architecture for isolated word recognition," *Neural Networks* **3**, pp. 23–43, 1990.
11. E. Wan, "Time series prediction using a connectionist network with internal delay lines," in *Time Series Prediction: Forecasting the Future and Understanding the Past*, A. Weigend and N. Gershenfeld, eds., pp. 195–217, Addison-Wesley, Reading, 1994.
12. L. Sirovich, "Turbulence and the dynamics of coherent structures, pt i: Coherent structures," *Quarterly of Applied Mathematics* **XLV**, pp. 561–571, 1987.
13. M. Kirby and R. Miranda, "The nonlinear reduction of high-dimensional dynamical systems via neural networks," *Physical Review Letters* **72**, pp. 1822–1825, 1994.
14. T. Kohonen, *Self-Organization and Associative Memory, 3rd Ed.*, Springer, 1989.
15. T. Martinetz, S. Berkovich, and K. Schulten, ""Neural-Gas" network for vector quantization and its application to time-series prediction," *IEEE Trans. on Neural Networks* **4**, pp. 558–569, 1993.
16. T. Martinetz and K. Schulten, "Topology representing networks," *Neural Networks* **7**, pp. 507–522, 1994.
17. H. Abarbanel and M. Kennel, "Local false nearest neighbors and dynamical dimensions from observed chaotic data," *Physical Review E* **47**, pp. 3057–3068, 1993.
18. F. Pineda and J. Sommerer, "Estimating generalized dimensions and choosing time delays: A fast algorithm," in *Time Series Prediction: Forecasting the Future and Understanding the Past*, A. Weigend and N. Gershenfeld, eds., pp. 367–385, Addison-Wesley, Reading, 1994.
19. H. Bauer and K. Pawelzik, "Quantifying the neighborhood preservation of self-organizing feature maps," *IEEE Trans. on Neural Networks* **3**, pp. 570–579, 1992.

20. A. N. Srivastava and A. S. Weigend, "Computing the probability density in connectionist regression," in *Proceedings of the International Conference on Artificial Neural Networks, Sorrento, Italy (ICANN 94)*, M. Marinaro and P. G. Morasso, eds., pp. 685–688, Springer-Verlag, 1994. Also in Proceedings of the IEEE International Conference on Neural Networks, Orlando, FL (IEEE–ICNN'94), p. 3786–3789, IEEE-Press.
21. A. Fraser and A. Dimitriadis, "Forecasting probability densities using hidden Markov models with mixed states," in *Time Series Prediction: Forecasting the Future and Understanding the Past*, A. Weigend and N. Gershenfeld, eds., pp. 265–282, Addison-Wesley, Reading, 1994.
22. D. Nix and A. Weigend, "Learning local error bars for nonlinear regression," in *Advances in Neural Information Processing Systems 7 (NIPS*94)*, G. Tesauro, D. S. Touretzky, and T. K. Leen, eds., pp. 489–496, MIT Press, Cambridge, MA, 1995.
23. A. S. Weigend and N. A. Gershenfeld, eds., *Time Series Prediction: Forecasting the Future and Understanding the Past*, Addison-Wesley, Reading, MA, 1994.
24. http://www.cs.colorado.edu/∼anderson/EEG/eeg.html.
25. D. Hundley, *Local Nonlinear Modeling via Neural Charts*. PhD dissertation, Colorado State University, Department of Mathematics, 1998.