Time Series Prediction and Neural Networks

R.J.Frank, **N.Davey**, **S.P.Hunt**
Department of Computer Science, University of Hertfordshire, Hatfield, UK.
Email: { R.J.Frank, N.Davey, S.P.Hunt }@herts.ac.uk

Abstract

Neural Network approaches to time series prediction are briefly discussed, and the need to find the appropriate sample rate and an appropriately sized input window identified. Relevant theoretical results from dynamic systems theory are briefly introduced, and heuristics for finding the appropriate sampling rate and embedding dimension, and thence window size, are discussed. The method is applied to several time series and the resulting generalisation performance of the trained feed-forward neural network predictors is analysed. It is shown that the heuristics can provide useful information in defining the appropriate network architecture.

1 Introduction

Neural Networks have been widely used as time series forecasters: most often these are feed-forward networks which employ a sliding window over the input sequence. Typical examples of this approach are market predictions, meteorological and network traffic forecasting. [1,2,3]. Two important issues must be addressed in such systems: the frequency with which data should be sampled, and the number of data points which should be used in the input representation. In most applications these issues are settled empirically, but results from work in complex dynamic systems suggest helpful heuristics. The work reported here is concerned with investigating the impact of using these heuristics. We attempt to answer the question: can the performance of sliding window feed-forward neural network predictors be optimised using theoretically motivated heuristics? We report experiments using four data sets: the sequence obtained from one of the three dimensions of the Lorenz attractor, a noisy version of the same data, ATM network traffic data and a series of 5400 tree ring measurements.

2 Time Series Prediction

A time series is a sequence of vectors, $\boldsymbol{x}$(t), t = 0,1,… , where t represents elapsed time.
For simplicity we will consider here only sequences of scalars, although the techniques considered generalise readily to vector series. Theoretically, *x* may be a value which varies continuously with t, such as a temperature. In practice, for any given physical system, *x* will be sampled to give a series of discrete data points, equally spaced in time. The rate at which samples are taken dictates the maximum resolution of the model; however, it is not always the case that the model with the highest resolution has the best predictive power, so that superior results may be obtained by employing only every *nth* point in the series. Further discussion of this issue, the choice of time lag, is delayed until section 3.4, and for the time being we assume that every data point collected will be used.

Work in neural networks has concentrated on forecasting future developments of the time series from values of *x* up to the current time. Formally this can be stated as: find a function $f : \Re^N \to \Re$ such as to obtain an estimate of *x* at time t + d, from the N time steps back from time t, so that:

$$x(t+d) = f(x(t), x(t-1), \ldots, x(t-N+1))$$
$$x(t+d) = f(\mathbf{y}(t)) \text{ where } \mathbf{y}(t) \text{ is the N-ary vector of lagged } x \text{ values}$$

Normally d will be one, so that *f* will be forecasting the next value of *x*.

2.1 Neural Network Predictors

The standard neural network method of performing time series prediction is to induce the function *f* using any feedforward function approximating neural network architecture, such as, a standard MLP, an RBF architecture, or a Cascade correlation model [8], using a set of N-tuples as inputs and a single output as the target value of the network. This method is often called the *sliding window technique* as the N-tuple input slides over the full training set. Figure 1 gives the basic architecture.

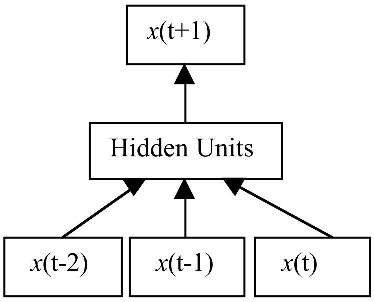


Figure 1: The standard method of performing time series prediction using a sliding window of, in this case, three time steps.

As noted in [4] this technique can be seen as an extension of auto-regressive time series modelling, in which the function *f* is assumed to be a linear combination of a fixed number of previous series values. Such a restriction does not apply with the non-linear neural network approach as such networks are general function approximators.

3 THEORETICAL CONSIDERATIONS

Time series are generally sequences of measurements of one or more visible variables of an underlying dynamic system, whose state changes with time as a function of its current state vector **u**(t):

$$\frac{d\mathbf{u}(t)}{dt} = G(\mathbf{u}(t))$$

For the discrete case, the next value of the state is a function of the current state: $\mathbf{u}(t+1) = F(\mathbf{u}(t))$

Such dynamic systems may evolve over time to an attracting set of points that is regular and of simple shape; any time series derived from such a system would also have a smooth and regular appearance. However another result is possible: the system may evolve to a chaotic attractor. Here, the path of the state vector through the attractor is non-periodic and because of this any time series derived from it will have a complex appearance and behaviour.

In a real-world system such as a stock market, the nature and structure of the state space is obscure; so that the actual variables that contribute to the state vector are unknown or debatable. The task for a time series predictor can therefore be rephrased: given measurements of one component of the state vector of a dynamic system is it possible to reconstruct the (possibly) chaotic dynamics of the phase space and thereby predict the evolution of the measured variable? Surprisingly the answer to this is yes. The *embedding theorem* of Mañé & Takens [5] shows that the space of time lagged vectors **y** with sufficiently large dimension will capture the structure of the original phase space. More specifically they show that if N, the arity of **y**, is at least twice the dimension of the original attractor, “then the attractor as seen in the space of lagged co-ordinates will be smoothly related” [5] to the phase space attractor. Of course this does not give a value for N, since the original attractor dimension is unknown, but it does show that a sufficiently large window will allow full representation of the system dynamics. Abarbanel et al. [5] suggest heuristics for determining the appropriate embedding size and time lag, and these are discussed below.

3.1 Heuristics for window size estimation

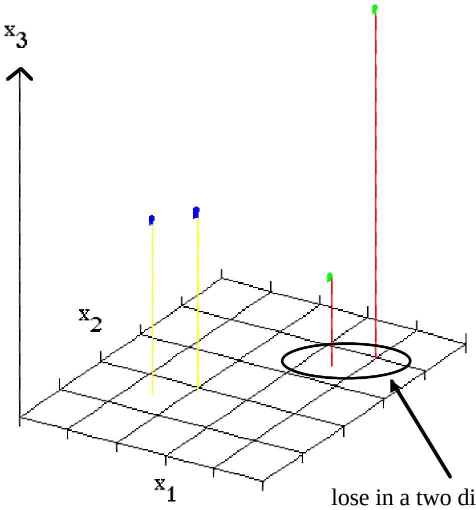
Having a sufficiently large time delay window is important for a time series predictor - if the window is too small then the attractor of the system is being projected onto a space of insufficient dimension, in which proximity is not a reliable guide to actual proximity on the original attractor. Thus, two similar time delay vectors $\mathbf{y}^1$ and $\mathbf{y}^2$, might represent points in the state space of the system which are actually quite far apart. Moreover, a window of too large a size may also produce problems: since all necessary information is populated in a subset of the window, the remaining fields will represent noise or contamination. Various heuristics can be used to estimate the embedding dimension, and here we use the false nearest neighbour method and the singular-value analysis [5]

3.2 False Nearest Neighbour Method

In order to find the correct embedding dimension, N, an incremental search, from N = 1, is performed. A set of time lagged vectors $\mathbf{y}_N$, for a given N, is formed. The nearest neighbour relation within the set of $\mathbf{y}_N$’s is then computed. When the correct value of N has been reached, the addition of an extra dimension to the embedding should not cause these nearest neighbours to spring apart. Any pair whose additional separation is of a high relative size is deemed a false nearest neighbour pair. Specifically, if $\mathbf{y}_N$ has nearest neighbour $\tilde{\mathbf{y}}_N$, then the relative additional separation when the embedding dimension is incremented is given by:

$$\left|\frac{d(\mathbf{y}_N,\tilde{\mathbf{y}}_N)-d(\mathbf{y}_{N+1},\tilde{\mathbf{y}}_{N+1})}{d(\mathbf{y}_N,\tilde{\mathbf{y}}_N)}\right|.$$

When this value exceeds an absolute value (we use 20, following [5]) then $\mathbf{y}_N$ and $\tilde{\mathbf{y}}_N$ are denoted as false nearest neighbours. The idea is illustrated in Figure 2.



lose in a two dimensional
mbedding but not in three
so false nearest neighbours

Figure 2: Genuine and False nearest neighbours. In two dimensions both pairs of points appear to be nearest neighbours, but in three dimensions the right hand pairing is revealed as false.

3.3 Singular-Value Analysis

Firstly a relatively large embedding size is chosen. The eigenvalues of the covariance matrix of the embedded samples are then computed. The point at which the eigenvalues reach a floor may then indicate the appropriate embedding dimension of the attractor. The analysis gives a *linear* hint as to the number of active degrees of freedom, but it can be misleading for non-linear systems [5].

3.4 Sampling Rate

Since it is easily possible to over-sample a data stream, Ababarnel et al. [5] suggest computing the average mutual information at varying sampling rates, and taking the first minimum as the appropriate rate. The mutual information acts as the equivalent of the correlation function in a non-linear domain. A minimum shows a point at which sampled points are maximally decorrelated – a small change of sampling rate in either direction will result in increased correlation. See [5] for further details.

4 EXPERIMENTS

We examine the relationship between embedding dimension and network performance for four data sets.

4.1 Lorenz Data

The first data sets are derived from the Lorenz system, given by the three differential equations:

$$\frac{dx}{dt} = \sigma(y_t - x_t) \quad \frac{dy}{dt} = -x_t z_t + r(x_t - y_t) \quad \frac{dz}{dt} = x_t y_t - bz_t$$

We take parameter settings r = 45.92, b = 4.0 and σ = 16.0 [5], and use 25,000 x-ordinate points derived from a Runge-Kutta integrator with time step 0.01. Two data sets were produced one with the original data and the other a noisy version in which a large amount of uniform noise was then added, noise at ±10%. The effect of the noise can be seen in Figures 3 and 4:

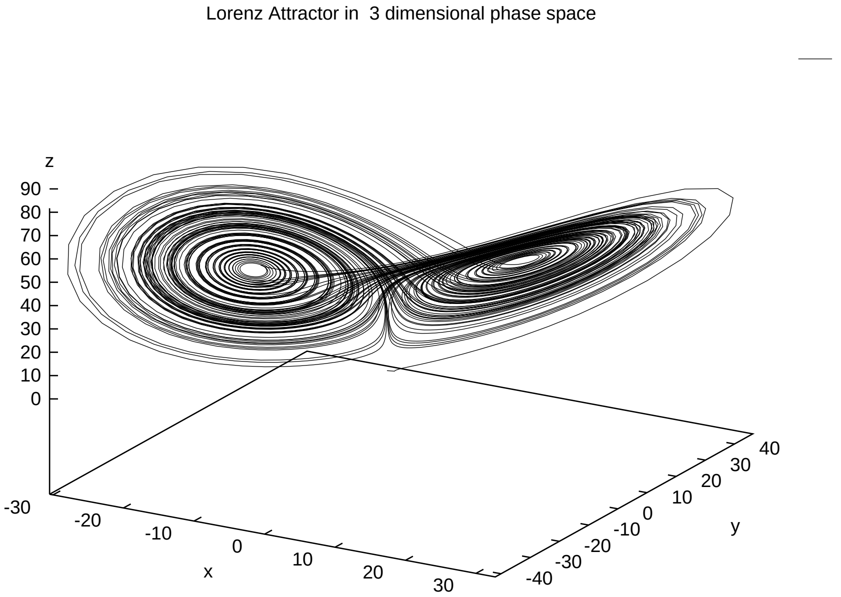


Figure 3: The Lorenz Attractor in 3 dimensional phase space x(t), y(t),z(t)

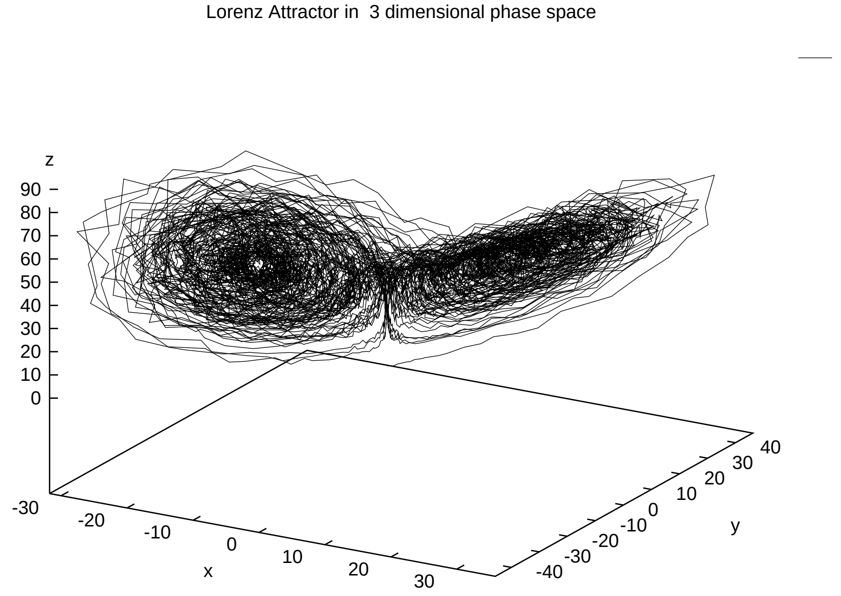


Figure 4: The Lorenz Attractor in 3 dimensional phase space x(t), y(t),z(t) Noise level =0.1

The mutual information analysis for both sets of Lorenz data is summarised in Figures 5 and 6. In either case the first minimum occurs at about a time lag of 13, and therefore a sampling rate of every 13 point was chosen for both data sets.

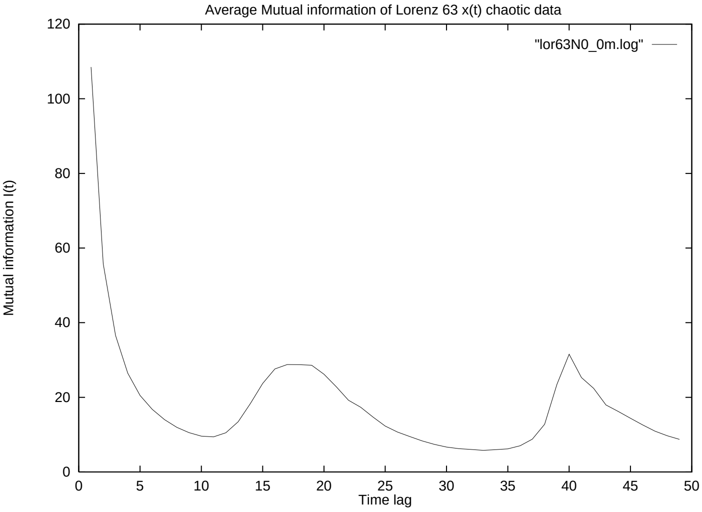


Figure 5: The mutual information graph for the Lorenz data with Noise level = 0.0

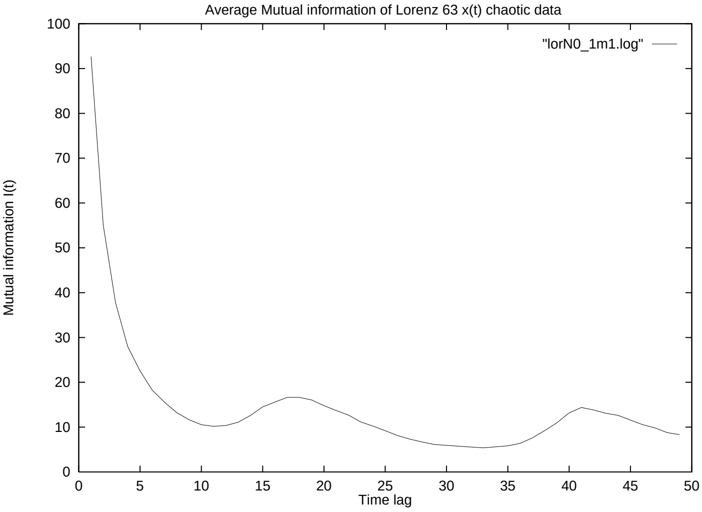


Figure 6: The mutual information graph for the Lorenz data with Noise level = 0.1

After sampling data sets consisting of 1923 points remain and both a nearest neighbour analysis and a singular-value analysis was undertaken, with results given in Tables 1 and 2. The false nearest neighbour results suggest that, in both cases, an embedding of 4 or 5 should be sufficient to represent the attractor. This corresponds well with the theoretical upper bound of about 5, from the embedding theorem. The singular-value analysis shows the contrast between the clean and noisy data very clearly. For the noisy data a “noise floor” for eigenvalues 5 onwards is apparent, but such a floor is, as expected, completely absent from the clean data.

Embedding Dimension	*Percentage of False*	*Nearest Neighbours*
	Clean Data	*Noisy Data*
2	77%	77%
3	3.3%	3.3%
4	0.3%	0.3%
5	0.3%	0.3%

Table 1: The percentage of false nearest neighbours in the Lorenz data set

Eigenvector	*Eigenvalues - Clean*	*Eigenvalues - Noisy*
1	1528.67	1534
2	73.15	74
3	3.12	3.7
4	0.10	0.64
5	0.00	0.59
6	0.00	0.59
7	0.00	0.57
8	0.00	0.56

Table 2: The eigenvalues of the sample covariance matrix with an embedding of 10 for the clean and noisy Lorenz dataset

We train a feedforward neural network with 120 hidden units, using conjugate gradient error minimisation. The embedding dimension, the size of the input layer, is increased from 2-units to 10 units. The data is split into a training set of 1200 vectors and test set of 715 vectors. Each network configuration is trained 10 times with different random starting points, for 3000 epochs. The point at which the test set error reached its lowest point was selected. This point is normally found well before the training set error minimum is reached. The lowest of the 10 test set errors is reported below, but it should be noted that there was little variation across runs. Errors are calculated as relative errors [8], given by:

$$\sqrt{\frac{\sum_t \left(observation_t - prediction_t\right)^2}{\sum_t \left(observation_t - observation_{t+1}\right)^2}}$$

The denominator simply predicts the last observed value, which is the best that can be done for a random walk. A ratio above 1 is a worse than chance prediction.

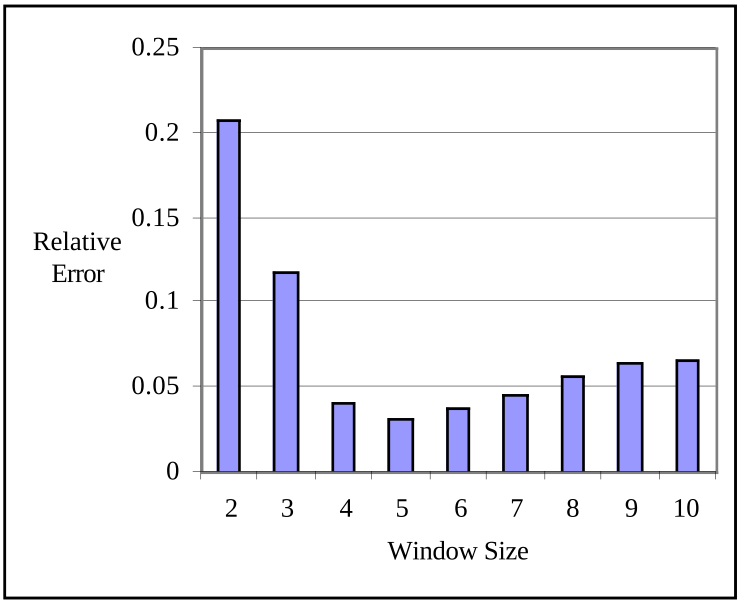


Figure 7: The relative error of the predictor with varying window size for the Lorenz data.. Averaged over 5 runs

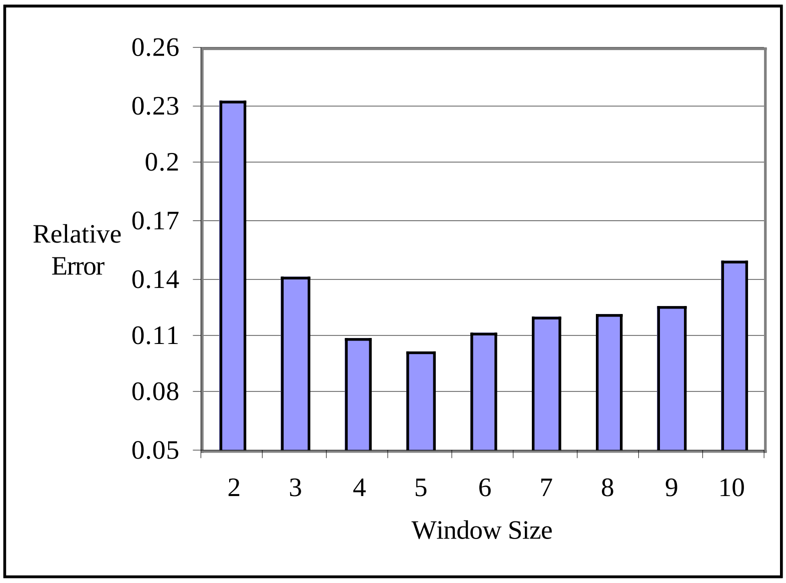


Figure 8: The relative error of the predictor with varying window size for the noisy Lorenz data.. Averaged over 5 runs.

As can be seen in Figures 7 and 8, the relative errors for the noisy Lorenz data are higher than for corresponding non noisy forecaster. However in both cases a window size of 5 gives clearly the best performance, corresponding precisely with the theoretical and heuristic prediction. For the clean data set the error rises as the window increases, and this cannot, here, be due to contamination in the higher dimensions. It is probably due to overfitting on the training set. For the noisy data, the increasing generalisation error could be due to overfitting or to the modelling of noise, or most likely a combination of both.

4.2 Voice traffic demand, over an ATM network

In this experiment, data of 1339 time series points representing network telephony traffic was used [1]. The false nearest neighbour analysis gave the result shown in Table 2. In this case a window of size four was chosen.

One of the characteristics of Telecomms traffic is the superimposing of many cyclical effects. For instance, there are hourly trends corresponding to the business day, daily trends according to which day of the week (some working days are typically busier than others and weekends have very little traffic) trends according to the day of the month (end of month can be busier) and seasonal trends. Each of these trends are cyclical with differing periodicities. To help the forecaster deal with this the day, hour and minute were added to the input using a periodic code, wit two bits for each feature, so that six additional input units are used, to give ten input units in total.

Embedding Dimension	*Percentage of False Nearest Neighbours*
1	100%
2	82.1%
3	7.1%
4	0.8%
5	0.7%

Table 3: The percentage of false nearest neighbours in the ATM data set

4.3 Tree Ring Data

This data set consists of 5405 data points recording tree ring data (measuring annual growth) at Campito mountain from 3435BC to 1969AD, [6], see Figure 9. The mutual information analysis of the data, shown in Figure 10, suggested a sampling rate of 1. There are no well established minimum in the graph, implying that every point is independent.

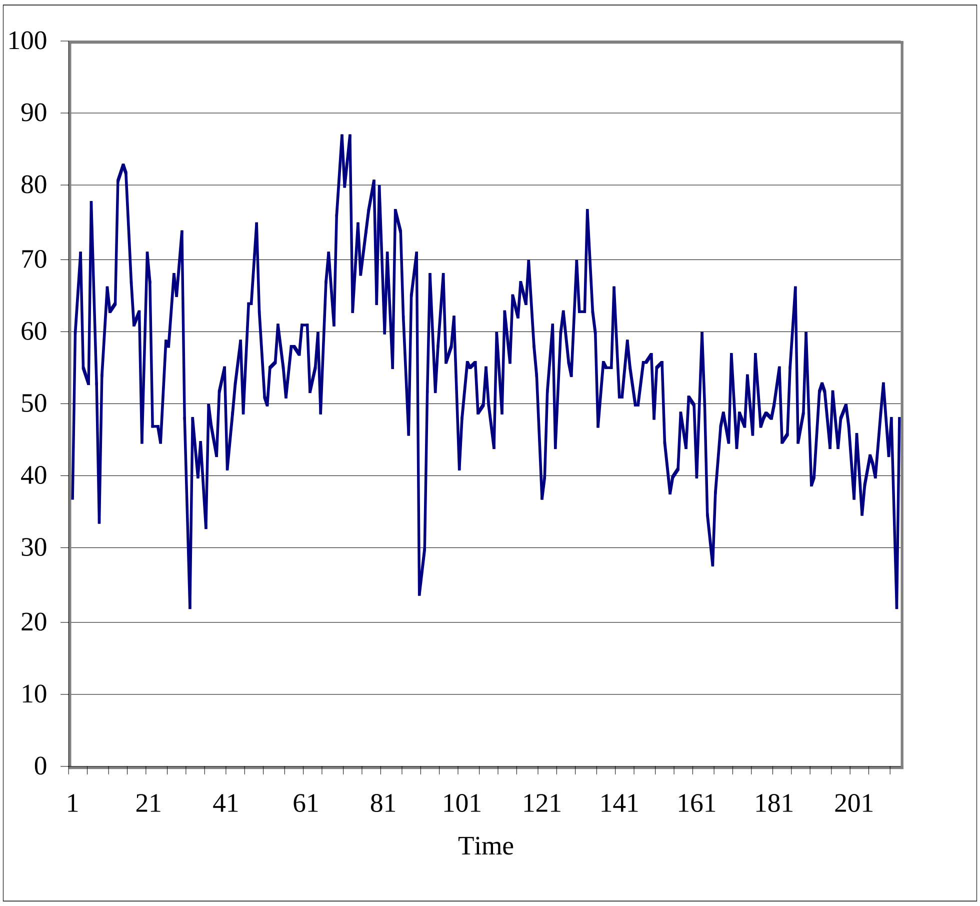


Figure 9: Tree Ring widths taken from the Campito mountain site, here just the first 210 points are shown.

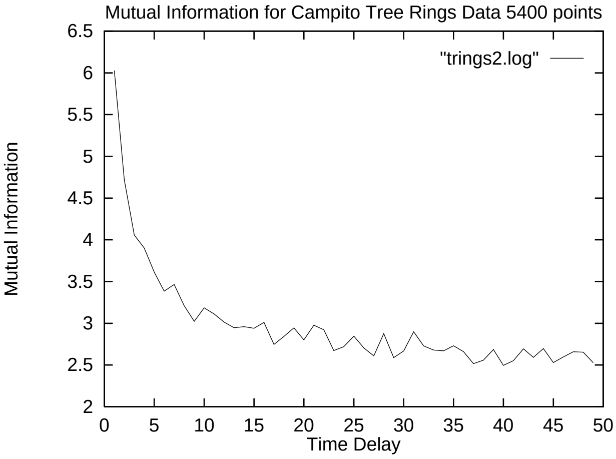


Figure 10: Mutual Information of the tree ring data. No clear minima is present.

The subsequent false nearest neighbour analysis and singular-value analysis are shown in Tables 4 and 5. The results here are not as clear as with the Lorenz data. The false nearest neighbour method suggests an embedding of 5, but the singular-value analysis does not give any strong indication of a suitable embedding dimension

Embedding Dimension	*Percentage of False Nearest Neighbours*
1	100%
2	13.6%
3	69.19%
4	1.6%
5	0.8%

Table 4: The percentage of false nearest neighbours in the Tree Ring data set.

Eigenvector	*Eigenvalue*
1	1209
2	208
3	126
4	92
5	79
6	65
7	55
8	52
9	50
10	45
11	39

Table 5: The eigenvalues of the sample covariance matrix with an embedding of 16 for the tree ring data.

In this case a cascade correlation network [7] was used The data was split into two: a training set, of 4000 points and a test set of 1405 points. The network was trained with a maximum of 25 hidden units, and the MSE of the test set was measured after each hidden unit recruitment. In all cases this error was at a minimum with less than the full number of hidden units. The relative error was calculated as by taking the quotient of the MSE with the error produced by a predictor that always chooses the next value to be the current value.

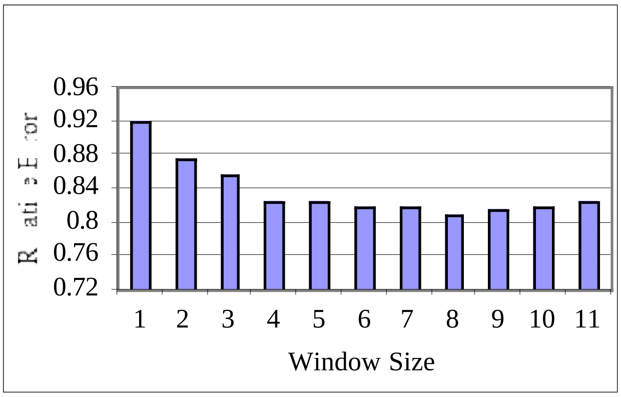


Figure 11: The relative error for the tree ring data test set, for varying window sizes and a trained cascade

correlation network

From Figure 11 it can be seen that the error falls to a minimum with a window of size 8 and then increases a little. The results show some correspondence with the nearest neighbour results and also the rising error is again indicative of a window size becoming too large. It is also noteworthy that the quality of the predictions here is significantly worse than for the Lorenz data.

5 CONCLUSIONS

The results suggest that the window size does have an important effect on the quality of a neural network based forecaster. From the experiments reported here on the Lorenz data it can be seen that optimal performance is clearly obtained at the correct embedding dimension and variation either side of this window size diminishes performance. Moreover this phenomena occurs for both noisy and clean versions of the data suggesting that the absence of noise does not obviate the need for great care in window size selection. A second important conclusion is that the heuristic approaches to embedding size estimation do provide useful guides in selection of appropriate network architecture. However the heuristics must be treated with caution on real world data. From the tree rings experiment it is apparent that the heuristics do not provide perfect guides and that even the optimal configuration may not produce good quality predictions.

References

[1] Edwards, T., Tansley, D, S. W., Davey, N., Frank, R. J.(1997). Traffic Trends Analysis using Neural Networks. *Proceedings of the International Workshop on Applications of Neural Networks to Telecommuncations 3*. pp. 157-164.

[2] Patterson D W, Chan K H, Tan C M. 1993, Time Series Forecasting with neural nets: a comparative study. *Proc. the international conference on neural network applictions to signal processing*. NNASP 1993 Singapore pp 269-274.

[3] Bengio, S., Fessant F., Collobert D. A Connectionist System for Medium-Term Horizon Time Series Prediction. *In Proc. Intl. Workshop Application Neural Networks to Telecoms* pp308-315, 1995.

[4] Dorffner, G. 1996, Neural Networks for Time Series Processing. *Neural Network World* 4/96, 447-468.

[5] Ababarnel H., D., I., Brown R., Sidorowich J., L. and Tsimring L., S., 1993, The analysis of observed chaotic data in physical systems. *Reviews of Modern Physics*, Vol. 65, No. 4 pp1331-1392

[6] Fritts, H.C. et al. 1971. Multivariate techniques for specifying tree-growth and climatic relationships and for reconstructing anomalies in Paleoclimate. *Journal of Applied Meteorology*, 10, pp.845-864.

[7] Fahlman, S.E. and C. Lebiere 1990. The Cascade-Correlation Learning Architecture. In Advances in Neural Information Processing Systems 2 (D. S. Touretsky, ed.) pp 524-532. San Mateo, CA: Morgan Kaufmann.

[8] Gershenfeld N. A. and A. S. Weigend, The Future of Time Series. In Time Series Prediction: Forecasting the Future and Understanding the Past, Gershenfeld A. N.. and A. S. Weigen, eds, pp 1-70. 1993.