Detecting chaos with neural networks

D. M. WOLPERT AND R. C. MIALL

University Laboratory of Physiology, University of Oxford, Parks Road, Oxford OX1 3PT, U.K.

SUMMARY

Predictions of the future values of a time series can be used to try to distinguish chaotic from noisy signals. We show that neurally inspired networks provide a powerful tool for this task, in that they can reliably distinguish between predictable, chaotic and noisy records. We have used these neural networks to analyse the 'standard' time series of measles and chickenpox cases in New York City. Applied to these real time series these new methods perform as well as a recently developed technique. We also employed feedforward and recurrent networks to analyse several sets of artificially generated data and found that they exhibit advantages over other recent techniques. These networks were able to generalize the rules governing the generation of the time series from very few data points and could also forecast series generated by non-stationary dynamics.

INTRODUCTION

The future behaviour of biological systems can be difficult to predict from a time series of previous measurements. There are two possible sources of the uncertainty. First, there may be truly random fluctuations in the system (true noise); and secondly the system, while being deterministic, may be so highly sensitive to initial conditions that long-term prediction is impossible (i.e. 'chaotic'). Rapp *et al.* (1985) point out that it is important to distinguish between chaotic and noisy systems in biology for two reasons. First, an apparently disordered system that is found to be chaotic may be following a very simple, potentially discoverable, dynamical law. Second, the irregular behaviour demonstrated by a chaotic system may be controllable. Chaotic activity is generated by deterministic systems, and deterministic systems respond to intervention in a deterministic fashion. Therefore changes in the parameters or perturbations of the system could produce transitions from chaos to ordered behaviour.

Recently, techniques to detect chaos have been developed, which rely on predicting the future from the past (Grassberger & Procaccia 1983; Farmer & Sidorowich 1987; Casdagli 1989; Sugihara & May 1990). Sugihara & May (1990) have demonstrated how such predictions can be used to try to distinguish chaos from noise: chaotic series are characterized by short term predictability and long term unpredictability. In other words, in a chaotic series the accuracy of predictions a number of time steps into the future (T_p) progressively falls as T_p increases, whereas predictions of a truly random uncorrelated noise signal start and remain inaccurate. Therefore if the series is chaotic, the correlation coefficient (ρ) of predicted against actual data shows a characteristic decline with increasing T_p. The rate of decline of ρ with increasing T_p is dependent on the Lyapunov exponent of the chaotic time series, whose units are bits of information per time step. This is a measure of the average rate of loss of predictive power per time step (Wolf 1986). In a predictable series in which, for example, it is as easy to predict one step ahead as ten, ρ is high and remains high as T_p increases.

These prediction methods have some limitations. First, autocorrelated or 'coloured' noise can produce a ρ–T_p plot very similar to that obtained for chaos. However, Sugihara & May (1990) suggest that such a series may tentatively be considered as chaotic rather than autocorrelated noise if the nonlinear prediction method that they introduced can produce a correlation coefficient significantly higher than the best fitting auto-regressive linear predictor. Second, as the predictions are time-independent they are unable to accurately predict ahead in series generated by non-stationary processes. It is possible that many biological phenomena could be generated by such non-stationary dynamics. For example, the parameters governing the dynamics of infectious disease spread may in some cases depend on some unobserved but changing variable. In other words, the equations describing the disease dynamics may themselves change in a predictable or even chaotic manner. Recurrent neural networks may offer ways to overcome this limitation.

Neural networks are a collection of identical, very simple computational units. These units can have input connections both from the external world and from other units within the network. Each unit can have outputs to other units in the network and/or to the external world. Each connection has a strength associated with it, which is called the connection or synapse weight and is usually modifiable. Each unit also has a modifiable bias value associated with it. The output of each unit depends both on its own bias and on the weighted sum of all its inputs, transformed via some simple linear or nonlinear function called its activation function. To train a network at a particular

Proc. R. Soc. Lond. B (1990) **242**, 82–86
Printed in Great Britain

task, some form of learning rule is used to modify the connection weights and unit biases iteratively until the network achieves the desired response, or reaches some equilibrium state. These networks have the ability to learn tasks by example.

Neural networks are a universal class of model in that a sufficiently large network can implement any function to any desired degree of accuracy (Hornik *et al.* 1989). By presenting a network with samples from a complex time series and training it to output subsequent values, the network can therefore be trained to approximate the dynamics which underlie the series. The network, once trained, can then be used to predict ahead in the time series, and can generalize from the training data to parts of the series that it has not been exposed to. The series can then be diagnosed as chaotic or noisy by assessing the ability of the network to predict accurately several samples ahead. Our work therefore builds on and extends the previous work of Lapedes & Farber (1987), who demonstrated the use of feedforward networks trained by back-propagation for predicting ahead in time series. We show that the network approach is as effective as the method used by Sugihara & May (1990) and that the networks can operate even under non-stationary conditions.

NETWORK ARCHITECTURE

The aim was to use neural networks to predict ahead from past values of a series. The input to the network was therefore the last E points in the series and the desired output was a prediction of the next point in the series. The network therefore has an embedding dimension E. This is equivalent to fitting a surface to the data in $E+1$-dimensional space. Prediction of a test set of data is equivalent to interpolation of this surface.

A feedforward, three-layered neural network with sigmoidal activation function (Rumelhart & McClelland 1986) was used for most of the time series predictions reported in this paper (figure 1*a*). Each unit was associated with a bias and each connection between units had a weight. The weights and the biases

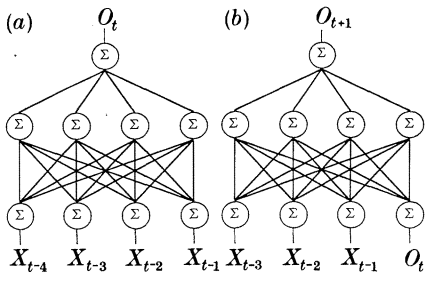

Figure 1. (*a*) An example of the type of network used in the majority of the predictions made in this paper. It has four inputs, four hidden units and one output (a 4–4–1 network). This is a fully connected feedforward network with a sigmoidal activation function. (*b*) The inputs used to generate a prediction two time steps ahead ($T_p = 2$). O_t is the prediction one time step ahead produced by the network in figure 1 (*a*). Correlation coefficients were then calculated between the predicted and target values for the second half of the series, as T_p was varied from 1 to 16.

were positive or negative real numbers. There was one input unit for each of the E embedding variables $\{X_{i-E}, X_{i-E+1} \ldots X_{i-1}\}$. The network therefore generated one output O_i in response to inputs from a window of size E sampled from the time series. The window was shifted through the series point by point to generate the full output series.

Each time series investigated was divided into two equal halves. The first half $\{X_1, X_2 \ldots X_n\}$ was used as a training set and the second half $\{X_{n+1}, X_{n+2} \ldots X_{2n}\}$ as a test set. The input window was repeatedly passed over the first half, sequentially exposing the network to the time series data. The network produced an output time series $\{O_1 \ldots O_n\}$ on each pass through the data, and the weights were then updated. The generation of one output series followed by the update of all weights and biases was regarded as one learning trial.

The number of input and hidden units was varied from 0 (i.e. no hidden layer) to 12; the optimal number for each data set is difficult to specify. If too few, the predictions were poor, but if too many, the networks failed to generalize sufficiently. Using a large network is analogous to high-order polynominal regression, which, while fitting the training set well, will show large excursions between training points and therefore will lead to poor interpolation on the test set. To prevent this overfitting of the training data and thereby provide good generalization we restricted the number of units in the network. The networks presented here are the best of those tested; because of time constraints we were not able to test all combinations of input and hidden unit numbers. The network configuration for each example is mentioned in the figure legends.

TRAINING RULE

The predictive performance of the network was maximized by a form of random evolutionary hill-climbing. This is a technique for modifying the connection weights of the network by an iterated search through the weight-space. Baba (1989) has shown that this type of evolutionary hill-climbing, which was adapted from work by Solis & Wets (1981), can be faster than back-propagation (Rumelhart & McClelland 1986) in certain circumstances and he claims that this technique is more likely to find the global optimum than back-propagation.

A network's performance is a function of its weights and biases (m variables). These can be represented by a point v in m-dimensional space. The hill-climbing strategy starts by randomly choosing a starting point v_1 within the m dimensional space and evaluating the network performance. Another point v_2 is then randomly chosen in the near vicinity of point v_1. The effect of choosing a neighbouring point is equivalent to changing all the connection weights and biases by small amounts. One moves to v_2 if performance at the point in weight-space defined by v_2 is better than at v_1. The process is then repeated iteratively. This technique guarantees that performance will always increase and will therefore reach at least a local maximum.

Three refinements were made to this basic search

technique (Baba 1989). First, a momentum term was added so that if a search in a particular direction in weight-space yielded a good result then the next search was biased towards the same direction. Second, if a search in one direction yielded a lower correlation, then the next point tried was in the opposite direction. Third, the step size had a Gaussian distribution with a mean of zero and its variance decreased with time (simulated annealing). This allows the search process to settle down to a fine-scale search within an optimal area of weight-space.

We chose to maximize the correlation coefficient between the input and output series as a measure of predictive performance. This allows the network to mimic the shape of the series rather than try to match the actual values. Therefore the network was able to use the range of its sigmoidal activation function which best suited the predictive task. The network was therefore not trained to predict the actual time series data, but rather to produce an output that had a high correlation with the series. If needed, the output could easily be scaled to produce an actual prediction.

ASSESSING NETWORK PERFORMANCE

Once a network had achieved a high correlation coefficient on its predictions on the first half of the series, further learning was prevented by holding all weights and biases fixed. The network was then assessed on the second half of the time series $\{X_{n+1}, X_{n+2} \ldots X_{2n}\}$, which it had not yet seen (the test set). This procedure produced a correlation coefficient for predictions one time step into the future of the test set ($T_p = 1$).

To make predictions two time steps into the future ($T_p = 2$), the output of the network was passed back through the network as one input with the previous inputs shifted by one place (figure 1*b*). The new output was now the prediction for two time steps into the future. By passing the output back through the network T_p times it was possible to obtain predictions for X_{t+T_p} (Lapedes & Farber 1987). The correlations between the predicted and the actual values of the test data were then plotted for different values of T_p.

PREDICTION OF STATIONARY DYNAMICAL EQUATIONS

As an example of a partly predictable and partly noisy series, figure 2*a* shows a time series of 200 samples of a sinusoid of unit amplitude contaminated by uncorrelated white noise, which was uniformly distributed across the range $[-1.5, 1.5]$. Figure 2*b* shows the first 200 points obtained from the chaotic logistic equation $X_{t+1} = 3.8X_t(1-X_t)$ (May 1976). The correlations between the network's forecasts and the actual data series (figure 2*c*) demonstrate that the sinusoid–noise series (open squares) was indeed partly predictable and partly noisy. For the logistic series (filled squares), the rapid decline in correlation coefficient as T_p increased is the hallmark of chaos.

When analysing real-time series data, for example biological populations, it will not always be known which variables are the best predictors of the system's behaviour. In other words, the network needs to learn which inputs are the most useful for prediction of future points. As an extreme example of how the networks can cope with non-predictive inputs, a network was trained to predict ahead in a logistic series. In this case eight inputs were used in the network. Only one of these inputs received sequential values from the logistic series, as above. Unlike the other networks the remaining seven inputs received uniformly distributed random numbers [0, 1] (figure 3*a*). During the training stage the network reduced the importance of the random inputs by reducing the appropriate connections' weights. This demonstrates the network's capacity to extract only the useful information from the available inputs. It also demonstrates how networks can be used to predict time series whose generating rule incorporates unknown time-delays. For example a network was able to learn the generating rule for the double-step logistic series, in which X_{t+1} depends not on X_t but on X_{t-1} (figure 3*b*).

NON-STATIONARY DYNAMICS

The feedforward networks described above performed well on these series as the rules generating X_t

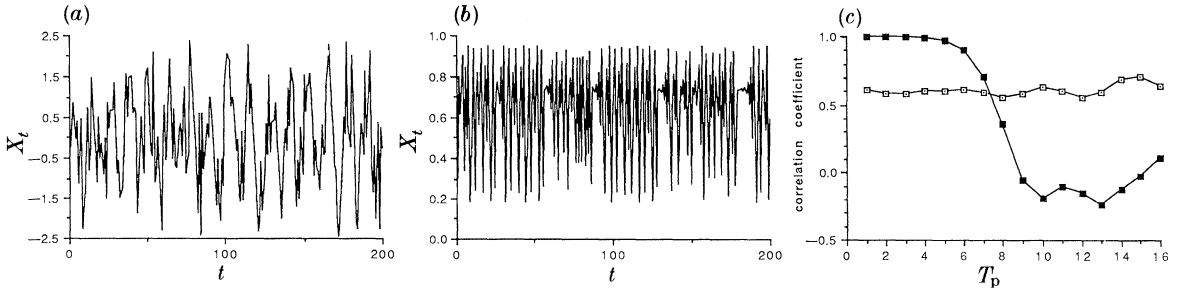

Figure 2. (*a*) The first 200 points in a time series generated by adding uniformly distributed noise (range $[-1.5, 1.5]$) to a regular sinusoid $X_t = \sin(0.5t)$. This is a partly predictable and partly noisy series. (*b*) The first 200 points in the time series generated by the logistic map $X_{t+1} = 3.8X_t(1-X_t)$. This series is an example of a purely deterministic chaotic series. (*c*) By using the method described in Figure 1, correlation coefficients between predicted and observed values in the second half of the series shown in Fig. 2 (*a*, *b*) were calculated as a function of prediction interval T_p. For the sine wave/noise an 8–2–1 network was trained for 400 trials. For the logistic equation a 1–4–1 network was used and trained for 200 trials. There was no fall-off in ρ with increasing T_p for the sinusoid/noise, demonstrating the network's ability to generalize the predictable part of the series, and thereby to maximize the correlation. The fall-off in ρ with increasing T_p for the logistic series demonstrates this characteristic feature of chaotic series.

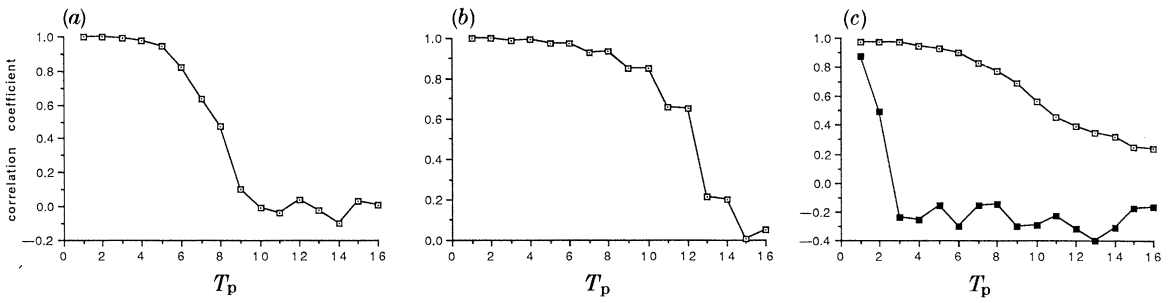

Figure 3. (*a*) Correlation coefficient against T_p for an 8–4–1 network where seven inputs were white noise (range (0, 1)) and the eighth was X_t from the logistic function. The network was trained to predict X_{t+1}. The network was able to predict as well under these conditions as the network in figure 2 (*c*); this demonstrates the network's ability to discount inputs that had no predictive power. (*b*) Correlation coefficient against T_p for the double-step logistic function $X_{t+2} = 3.8X_t(1-X_t)$. A 2–4–1 network was trained on the first 100 points in this series and then assessed on the next 100 points. The correlation coefficient shows a stepwise decline with T_p, confirming that it was a chaotic series, but that it was now as easy to predict two steps ahead as to predict one. (*c*) The plot of ρ–T_p for a non-stationary time series (see text). The line with closed squares was produced by a 2–4–1 feedforward network. The line with open squares was produced by a 1–4–1 recurrent network in which each hidden layer unit was connected to all the other hidden layer units including itself.

did not change with time, depending only on the preceding values of the series. Such feedforward networks are not able to store temporal information, and therefore they cannot be used for time series in which the governing rule does change over time. Networks that have recurrent connections can form reverberating circuits in which activity can be sustained without any external input to the network. Therefore these recurrent networks can learn to store information about time (Williams & Zipser 1989). This ability makes recurrent networks suitable for prediction of non-stationary time series.

The results of predictions made with feedforward and recurrent networks on a non-stationary series are shown in figure 3*c*. The series was generated by a rule in which each point in the series depended not only on the last point but also on whether t was odd or even:

$$X_{t+1} = 3.8X_t(1-X_t) \quad \text{for all even } t;$$
$$X_{t+1} = (1-X_t) \quad \text{for all odd } t.$$

In this case the recurrent network's performance was superior to that of a feedforward network. The network, without any prior knowledge of the time-dependence of the generating rule, was able to set up an internal counter, which allowed the network to alternate between the two generating rules.

ANALYSIS OF BIOLOGICAL DATA

We have also tested the network prediction method on the measles and chickenpox incidence records for New York City (1928–72) (figure 4*a*, *b*). These time series are considered up to 1963, when immunization altered the dynamics. They now serve as a standard test of new techniques (London & Yorke 1973; Schaffer & Kot 1985, 1986; Sugihara & May 1990). Sugihara & May (1990) first-differenced the chickenpox and measles data before using their simplex method for prediction. For purposes of comparison we have tested the network prediction method on both the raw (figure 4*c*) and first-differenced data (figure 4*d*); the results were qualitatively the same. Our results confirm that

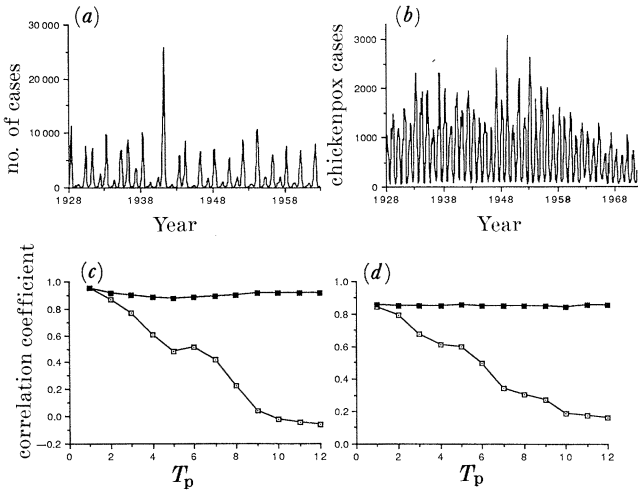

Figure 4. (*a*) The time series of monthly measles cases in New York City between 1928 and 1963 (432 points). Recent data points have not been included as an immunization programme, started in 1963, altered the disease's dynamics. (*b*) The time series of monthly chickenpox cases in New York City between 1928 and 1973 (532 points). (*c*) Correlation coefficients for the measles and chickenpox time series (raw data) against T_p. An 8–4–1 network was used for measles predictions and was trained on the first 216 points for 3500 trials. A 25–0–1 network was used for the chickenpox predictions, and was trained on the first 266 points for 10000 trials. The results show that chickenpox (filled squares) is a predictable series with noise ($\rho = 0.95$ for $T_p = 1$). However for measles (open squares), even though $\rho = 0.95$ for $T_p = 1$, the correlation coefficient shows the fall-off with increasing T_p, typical of chaos. (*d*) Correlation coefficients for the first-differenced measles and chickenpox time series. A 5–3–1 network was used for measles predictions (open squares) and was trained on the first 216 points for 5500 trials. A 25–0–1 network was used for chickenpox predictions (filled squares) and was trained on the first 266 points for 11000 trials. The results are qualitatively similar to those in figure 4 (*c*).

the measles series is chaotic and that the chickenpox series is predictable with a random component.

For the chickenpox data the network's correlation for $T_p = 1$ was significantly higher than that achieved

by the simplex method ($\rho = 0.86:0.82$, respectively; $p < 0.02$, $N = 266$) but not significantly different from an autoregressive linear model ($\rho = 0.84$) (Sugihara & May 1990).

The correlation for measles was not significantly different at $T_p = 1$ than the simplex method ($\rho = 0.85:0.84$) but was significantly higher than the best autoregressive linear model ($\rho = 0.85:0.72$, $p < 0.0005$). However the network's predictions were significantly higher when compared with the simplex method for $T_p = 2$ ($\rho = 0.79: < 0.7$, respectively, $p < 0.0005$, $N = 216$).

CONCLUSION

Successful distinction of noise from chaos through the correlation of predicted and actual data requires a technique that is both accurate and reliable. We have shown that feedforward and recurrent networks provide an effective predictor of both artificial and real data sets. The networks, in particular, were able to make accurate predictions in a non-stationary series. They were able to generalize the generating rule from small data sets, and therefore promise to be useful in fields such as population biology, economics and epidemiology, where by their nature data series tend to be short.

We thank J. F. Stein, R. M. May and G. Sugihara for helpful discussion. D.M.W. is supported by a Medical Research Council Training Fellowship and R.C.M. by the Wellcome Trust.

REFERENCES

Baba, N. 1989 A new approach for finding the global minimum of error function of neural networks. *Neural Networks* **2**, 367–373.

Casdagli, M. 1989 Nonlinear prediction of chaotic time series. *Physica* D **35**, 335–356.

Farmer, J. D. & Sidorowich, J. J. 1987 Predicting chaotic time series. *Phys. Rev. Lett.* **59**, 845–848.

Grassberger, P. & Procaccia, I. 1983 Characterization of strange attractors. *Phys. Rev. Lett.* **50**, 346–369.

Hornik, K., Stinchcombe, M. & White, H. 1989 Multilayer feedforward networks are universal approximators. *Neural Networks* **2**, 359–366.

Lapedes, A. S. & Farber, R. M. 1987 Nonlinear signal processing using neural networks: prediction and system modelling. Technical report LA-UR-87-2662, Los Alamos National Laboratory.

London, W. P. & Yorke, J. A. 1973 Recurrent outbreaks of measles, chickenpox and mumps. *Am. J. Epidemiol.* **98**, 453–467.

May, R. M. 1976 Simple mathematical models with very complicated dynamics. *Nature, Lond.* **261**, 459–467.

Rapp, P. E., Zimmerman, I. D., Albano, A. M., deGuzman, G. C., Greenbaum, N. N. & Bashore, T. R. 1985 Experimental studies of chaotic neural behaviour: cellular activity and electroencephalographic signals. Preprint.

Rumelhart, D. E. & McClelland, J. L. (eds) 1986 *Parallel distributed processing*. Cambridge, Massachusetts: MIT Press.

Schaffer, W. M. & Kot, M. J. 1985 Nearly one dimensional dynamics in an epidemic. *J. theor. Biol.* **112**, 403–407.

Schaffer, W. M. & Kot, M. J. 1986 Differential systems in ecology and epidemiology. In *Chaos: an introduction* (ed. A. V. Van Holden), pp. 158–178. Manchester University Press.

Solis, F. J. & Wets, J. B. 1981 Minimization by random search techniques. *Math. Oper. Res.* **6**, 19–30.

Sugihara, G. & May, R. 1990 Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series. *Nature, Lond.* **344**, 734–741.

Williams, R. J. & Zipser, D. 1989 A learning algorithm for continually running fully recurrent neural networks. *Neural Computation* **1**, 270–280.

Wolf, A. 1986 Quantifying chaos with Lyapunov exponents. In *Chaos: an introduction* (ed. A. V. Van Holden), pp. 272–290. Manchester University Press.

(Received 9 July 1990; accepted 10 August 1990)

Note added in proof (16 October 1990): A. S. Weigend *et al.* (*Int. J. Neural Systems*, in the press) report a similar technique. They used a feedforward network to predict ahead in time-invariant series.