

Non-linear modelling and chaotic neural networks

Antonia J. Jones, Steve Margetts
Department of Computer Science
Cardiff University
Cardiff CF24 3XF, U.K.

Peter Durrant
Universal Solutions
McCann-Erickson
London WC1N 1EX, U.K.

Alban P.M. Tsui
Cognos Ltd.
8-10 Haymarket
London SW1Y 4PE, U.K.

Abstract

This paper proposes a simple methodology to construct an iterative neural network which mimics a given chaotic time series. The methodology uses the Gamma test to identify a suitable (possibly irregular) embedding of the chaotic time series from which a one step predictive model may be constructed. This model is then iterated to produce a close approximation to the original chaotic dynamics.

Having constructed such networks we show how the chaotic dynamics may be stabilised using time-delayed feedback, which is a plausible method for stabilisation in biological neural systems.

Using delayed feedback control, which is activated in the presence of a stimulus, such networks can behave as an associative memory, in which the act of recognition corresponds to stabilisation onto an unstable periodic orbit.

We briefly illustrate how two identical dynamically independent copies of such a chaotic iterative network may be synchronised using the delayed feedback method. Although less biologically plausible, these techniques may have interesting applications in secure communications.

1. Introduction

This work draws its inspiration from [7]. On the basis of studies of the olfactory bulb of a rabbit Freeman suggested that in the iterative ‘rest state’ the dynamics of this neural cluster is chaotic, but that when a familiar scent is presented the neural system rapidly simplifies its behaviour and the dynamics becomes more orderly, more nearly periodic than when in the rest state. We call this the *retrieval behaviour* since it is analogous to the act of recognition. This suggests an interesting model of recognition in biological neural systems which is quite different from earlier attempts to use neural networks for pattern recognition or as associative memories.

To construct such a system we have to consider how best to construct neural models which exhibit chaotic dynamics.

Neural network models which are dynamical systems are not new. The classical example is the Hopfield network [9], for which the simplest case considers nodes whose outputs are zero or one and where memories are associated with specified (preferably uncorrelated) point attractors. However, such a model cannot meet our needs. The state space is finite, consisting of fixed length vectors whose components are zero or one, and hence *chaos*, in the classical sense of dynamical systems, with its infinitely rich variety of modalities will never be exhibited. Indeed for a symmetric Hopfield network the dynamics are essentially trivial: starting from any initial state the network will simply iterate to a fixed point.

In contrast if the dynamics are chaotic then unstable periodic orbits are dense on the chaotic attractor¹ and there are infinitely many of them. Thus an associative memory such as described by Freeman, for which the computations are performed to an *arbitrary precision*, could in principle accommodate infinitely many memories. At any rate such a system is not subject to the conventional Hopfield upper bound of $0.15n$, where n is number of neurons [2]. Of course, for the Hopfield net the situation is rather different. In the Hopfield model memories are associated with specified point attractors, whereas in the Freeman paradigm memories would be associated with unstable periodic behaviours which could not be specified *ab initio*. Plainly we need to work with network models having continuous node outputs rather than the discrete outputs of the classical Hopfield model.

Chaotic dynamics have been observed in many artificial neural systems, either in continuous-time systems [12] or discrete-time systems [20]. An early example of a neural system which displays chaos at the *neural* level is the *voltage-controlled oscillator neuron* (or VCON) of [10]. This model, in contrast to all-or-none neuron models, generates voltage spikes that phase-lock to oscillatory stimulation, similar to the phase-locking of action potentials to oscillatory voltage stimulation observed in Hodgkin-Huxley preparations of squid axons [8].

¹See footnote 1 on page 1199 of [14].

In this paper, building on existing knowledge of smooth non-linear modelling techniques, we propose a methodology, suggested in part by Takens theorem [18], to build chaotic neural networks based on an iterative model of a conventional feedforward network trained to accurately model a given chaotic time series. We show how such networks can readily be constructed and give a simple example.

1.1. Controlling neural chaos

Having seen how to exhibit neural chaos the next question becomes how to control it? Most such methods, such as the OGY method, require careful and systematic analysis of the chaotic dynamical behaviour and prior specification of the target unstable fixed point, which is difficult and computationally expensive, before successful control can be achieved. Moreover, such control techniques are *external* to the system being controlled, whereas for a neural system to behave as described by [7] the control should be *intrinsic* to the neural dynamics.

The dynamics of large neural ensembles are high-dimensional, and whilst the OGY technique is an effective tool for the control of low dimensional chaos it needs further elaboration for effective control of higher dimensional systems. Indeed, for higher dimensional systems it may be that other types of control procedures will prove far more effective.

Delayed feedback to control *continuous* dynamical systems exhibiting chaos was first suggested in [15]. We use a modified version of this approach to stabilise an iterative neural model (previously trained to generate chaotic behaviour in the ‘rest state’) in the presence of an input stimulus. One of the attractions of delayed feedback stabilisation is that it has a very low computational overhead and so is extremely easy to implement in hardware. It would also be very easy to implement in biological neural circuitry and so offers one plausible mechanism whereby such stabilisation might occur.

The particular unstable periodic orbit which is stabilised depends quite strongly on the precise character of the applied stimulus. Thus the system can act as an associative memory in which the act of recognition corresponds to stabilising onto an unstable periodic orbit which is characteristic of the applied stimulus. The entire artificial system therefore exhibits an overall behaviour and response to stimulus which precisely parallels the biological neural behaviour observed by Freeman.

1.2. Synchronisation

The idea of synchronising two independent copies of identical chaotic dynamical systems has been of increasing

recent interest. Results shown by Skarda and Freeman [17] support the hypothesis that neural dynamics are heavily dependent on chaotic activity. Nowadays it is believed that synchronization plays a crucial role in information processing in living organisms and could lead to important applications in speech and image processing [13]. Moreover due to the important role that secure communications plays in industrial and banking communications, the potential application of chaotic synchronization to secure communications is receiving increased attention. We show how two identical copies of a chaotic neural system may be synchronised using a variation of the method used for control - these results are based on the work of [3].

2. The Gamma test

The Gamma test is a non-linear modelling analysis tool. Suppose we are given an input/output data set of the form $(\mathbf{x}(i), y(i))$ ($1 \leq i \leq M$), where $\mathbf{x}(i) \in \mathbb{R}^m$ and without loss of generality $y(i) \in \mathbb{R}$. If y is a vector we treat each component separately and at very little extra computational cost obtain an individual estimate of the Gamma test result for each output. Assume the data is described by an underlying model of the form

$$y = f(\mathbf{x}) + r = f(x_1, \dots, x_m) + r \quad (1)$$

where f is a smooth function with bounded partial derivatives, and r is a stochastic variable with mean zero and bounded variance.

The Gamma test [1],[11] is designed to estimate the variance of r , $\text{Var}(r)$, i.e. that part of the variance of the output which cannot be accounted for by the existence of some underlying smooth model f (even though f is unknown). A single run of the Gamma test has a time complexity of $O(M \log M)$ (using, for example, a k-d tree) and depending on the input dimension m , executes in a few seconds for data sets of a reasonable size (i.e. $M \leq 1000$). This algorithm requires certain pre-conditions. We assume that training and testing data are different sample sets in which:

- Input and output variables take continuous values
- The training set inputs are non-sparse in input-space
- Each output is determined from the inputs by a smooth deterministic process which is the same for both training and test sets
- Each output is subjected to statistical noise whose distribution may be different for different outputs but which is the same in both training and test sets for corresponding outputs

Here we shall give a brief description of the algorithm but not enter into the details of the theoretical discussion of its range of applicability. For a wide class of sampling distributions in input space, whose support have positive measure and satisfy some reasonable side conditions, theoretical proofs of the algorithm appear feasible and details will be discussed elsewhere [5]. However, the test seems quite robust with respect to the probability density function φ of the sampling distribution in input space and to the precise nature of its support. This is fortunate, because many of the more interesting applications involve chaotic (possibly noisy) time series, and with typical chaotic time series the support of φ (in the embedding space) has measure zero and a Hausdorff dimension which is often substantially less than m . For such cases the test works well and the number of data points M required to obtain an accurate estimate of $\text{Var}(r)$ is substantially less than might otherwise be the case (e.g. if the sampling distribution were uniform over input space).

A single run of the Gamma test returns two numbers. The first (Γ = vertical intercept) is an estimate of $\text{Var}(r)$. The second (A = slope) is one estimate of the complexity of the surface described by the unknown f . This slope estimate is more meaningful if the outputs have been standardized to $[0, 1]$. It is helpful to think of the slope estimate as roughly the average of $(1/4)|\nabla f|^2$ over the input space. If f is a very ‘bumpy’ surface then A will be large. If $A \approx 0.25$ then the function f is essentially linear. More data points (i.e. larger M) are required to get an accurate estimate of A than are required for Γ .

2.1. The Gamma test algorithm

The Gamma test works by exploiting the continuity of the unknown function f . Consider what happens when two data points $\mathbf{x}(i)$, $\mathbf{x}(j)$ are close together. Then since f is smooth we should expect that $f(\mathbf{x}(i))$ and $f(\mathbf{x}(j))$ are also close together. If they are not then it can only be because of the addition of noise to each of these values. The Gamma test (or near neighbour technique) is based on the statistic

$$\gamma = \frac{1}{2M} \sum_{i=1}^M (y'(i) - y(i))^2 \quad (2)$$

where $y'(i)$ is the y value corresponding to the first near neighbour of $\mathbf{x}(i)$. Given data samples $(\mathbf{x}(i), y(i))$, where $\mathbf{x}(i) = (x_1(i), \dots, x_m(i))$, $1 \leq i \leq M$, let $N[i, p]$ be the list of (equidistant) p^{th} nearest neighbours to $\mathbf{x}(i)$. We write

$$\delta(p) = \frac{1}{M} \sum_{i=1}^M \frac{1}{L(N[i, p])} \sum_{j \in N[i, p]} |\mathbf{x}(i) - \mathbf{x}(j)|^2$$

$$= \frac{1}{M} \sum_{i=1}^M |\mathbf{x}(i) - \mathbf{x}(N[i, p])|^2 \quad (3)$$

where $L(N[i, p])$ is the length of the list $N[i, p]$. Here, in the second sum, we use a convenient abuse of notation in which $\mathbf{x}(N[i, p])$ stands for any one of the (equidistant) p^{th} nearest neighbours of $\mathbf{x}(i)$, since the distance is the same for all. Thus $\delta(p)$ is the mean square distance to the p^{th} nearest neighbour. We also write

$$\gamma(p) = \frac{1}{2M} \sum_{i=1}^M \frac{1}{L(N[i, p])} \sum_{j \in N[i, p]} (y(j) - y(i))^2 \quad (4)$$

where the y observations are subject to statistical noise assumed independent of \mathbf{x} and having bounded variance².

Under reasonable conditions one can show that

$$\gamma(p) \approx \text{Var}(r) + A\delta(p) + o(\delta(p)) \quad \text{as } M \rightarrow \infty \quad (5)$$

where the convergence is in probability.

```

 $\delta(p) = 0, \gamma(p) = 0$  for  $1 \leq p \leq p_{max}$ 

for  $i = 1$  to  $M$  do
  generate  $N[i, p]$  {find the  $p_{max}$  near neighbours of  $\mathbf{x}(i)$ }
  for  $p = 1$  to  $p_{max}$  do
     $\delta(p) = \delta(p) + [\mathbf{x}(i) - \mathbf{x}(N[i, p])]^2$ 
     $z(p) = 0$ 
    for  $j = 1$  to  $L(N[i, p])$  do
       $z(p) = z(p) + [y(i) - y(N[i, p][j])]^2$ 
    end for
     $\gamma(p) = \gamma(p) + [z(p)/L(N[i, p])]$ 
  end for
end for

for  $p = 1$  to  $p_{max}$  do
   $\delta(p) = \delta(p)/M$ 
   $\gamma(p) = \gamma(p)/2M$ 
end for

Perform least squares fit on  $(\delta(p), \gamma(p))$  ( $1 \leq p \leq p_{max}$ ) to compute  $y = Ax + \Gamma$ 
return  $(\Gamma, A)$ 

```

Algorithm 1: The Gamma test algorithm.

The Gamma test computes the mean-squared p^{th} nearest neighbour distances between $\delta(p)$ ($1 \leq p \leq p_{max}$, typically $p_{max} = 10$), and the corresponding $\gamma(p)$. Next the $(\delta(p), \gamma(p))$ regression line is computed and the vertical intercept is returned as the *Gamma statistic* (Γ). Effectively this is the limit $\lim \gamma$ as $\delta \rightarrow 0$, which in theory is $\text{Var}(r)$.

²The original version in [1] and [11] used smoothed versions of $\delta(p)$ and $\gamma(p)$ which rolled off the significance of more distant near neighbours. Later experience showed that this complication was largely unnecessary and the version of the software used here (*winGamma* [4] licensed by the University of Wales, Cardiff) is implemented as described above.

2.2. A simple example

We consider the function $y = \sin(4\pi x)$ and sampled with $M = 1000$ points and add to y uniformly distributed noise with mean zero and variance of 0.01. The resulting graph is shown in Figure 1. For increasing M the Gamma statistic is plotted in Figure 2 (we call this an M -test).

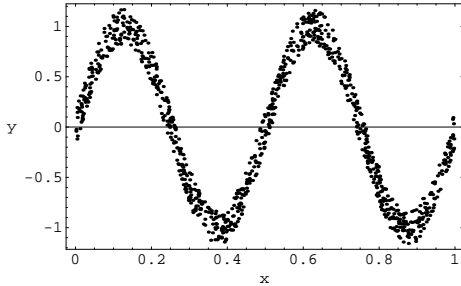


Figure 1. The noisy sine curve.

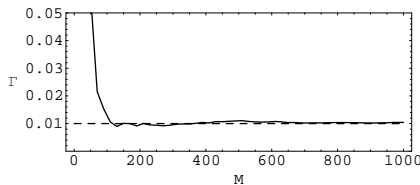


Figure 2. Gamma statistic against M for the noisy sine data.

2.3. An example of model construction

The networks used the sigmoidal function $g(x) = s_F (2/(1 + e^{-x/T}) - 1)$, where $T = 0.833$ and the scale factor $s_F = 1.5$, as the output function, where the activation x is given by

$$x = \sum_{j=1}^n w_{ij} x_j \quad (6)$$

where w_{ij} is the weight from node j to node i and x_j is the output of node j . The networks were trained using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [6] with a slightly modified line search procedure which proved more effective for large training sets.

2.4. Example: The Mackey-Glass equation

The Mackey-Glass equation is a time delayed differential equation which produces a chaotically evolving continuous dynamic system. The version used to generate the data is given by

$$\frac{dx}{dt} + 0.1x(t) = \frac{0.2x(t-\tau)}{1 + x(t-\tau)^{10}} \quad (7)$$

where $\tau = 30$ (N.B. $\tau > 17$). We integrated the equation over $t \in [0, 8000]$ with the initial condition $x(t) = 2$. No noise was added. The Lyapunov exponents from the training time series are $\{0.001, -0.006, -0.027\}$ to 3 decimal places using the technique from [16].

The Mackey-Glass time series data was created by writing out the values of $x(t)$ at $t = 10, 20, 30, \dots, 8000$ ($\Delta t = 10$) giving 800 data points of a chaotic time series. The plot of the first 100 data points is given in Figure 3. Given a reasonable amount of data, predicting a chaotic time series a small time ahead is usually not too difficult. The problem is to predict a long way ahead. Here $\Delta t = 10$ is a modest time ahead.

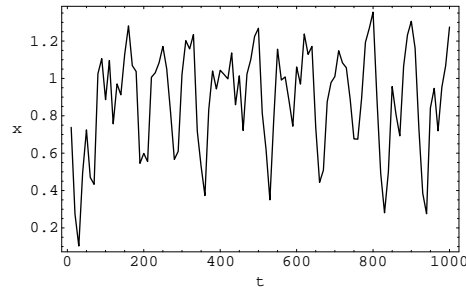


Figure 3. The Mackey-Glass time series.

2.5. Gamma test model identification and analysis

We first perform an increasing embedding up to maximum length 10. The graph is shown in Figure 4 which suggests that it is sufficient to search irregular embeddings up to a maximum length of 10. However since we are using a relatively small data set we shall examine embeddings to maximum length 6. We therefore next examine the prospect of trying to predict $x(t)$ using the last 6 values. There are $2^6 - 1 = 63$ combinations of inputs so it is no problem to do a full embedding search. We find that the best embedding (i.e. the embedding with smallest $|\Gamma|$) is 111100, which means that we predict $x(t)$ using $x(t-3\Delta t)$, $x(t-4\Delta t)$, $x(t-5\Delta t)$ and $x(t-6\Delta t)$. On this basis we generate the results in Table 1 and Figure 5 - Figure 6.

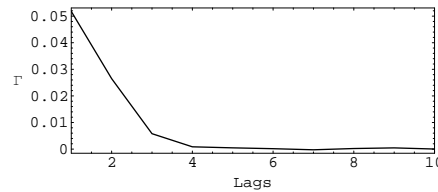


Figure 4. Increasing embedding for the Mackey-Glass time series.

It is interesting to note that the full embedding search obtained the best model by omitting $x(t-1\Delta t)$ and $x(t-$

Order	Γ	A	Embedding
1	0.00033122	0.29804	111100
2	0.00044890	0.26698	101101
3	0.00047976	0.24227	111101
4	0.00055886	0.26742	111110
5	0.00069914	0.23469	101111
6	0.00074670	0.40402	101110
7	0.00077647	0.40516	011110
8	0.00082972	0.25861	011111

Table 1. The first 8 best embeddings for the Mackey-Glass time series data in the ascending order of $|\Gamma|$.

2. Δt). Why is this? In the original time delay equation the value $x(t)$ depends on the value $x(t-30)$ but $x(t-10)$ and $x(t-20)$ are not needed at all, as the software discovered.

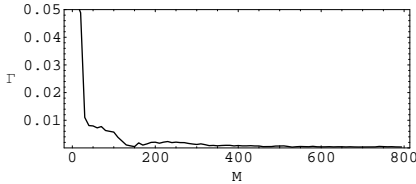


Figure 5. M -test for a Mackey-Glass time series using the embedding 111100.

The embedding 111100 provides a four input/one output set of I/O pairs. The low noise level $\Gamma \approx 0.0003$, combined with the rapid fall off of the M -test graph (Figure 5), and $\Gamma/\text{Var}(y) \approx 0.004$ indicates the existence of a reasonably accurate smooth model. The regression line fit is good with $SE \approx 0.0002$. The slope $A \approx 0.298$ is low enough to suggest a fairly simple non-linear model. Taken together these are clear indicators that it should be quite straightforward to construct a predictive model using around 800 data points with an expected MSE around 0.0003. The scatter plot of Figure 6 contains the typical more or less blank wedge in the lower small δ region, which also supports the conclusion.

2.6. Neural network construction and testing

A neural network with 4 inputs, two hidden layers with eight neurons each and one output neuron (4 – 8 – 8 – 1) was trained using $M = 800$ data points to a MSE of 0.000329877 which is comparable with the Gamma statistic. The plot of $x(n+1)$ against $x(n)$ in Figure 7 shows the original attractor constructed from the training data. Figure 8 shows the analogous result obtained by iterating the trained neural network. Using 100 unseen samples with the same sampling time of $\Delta t = 10$ from the system as test

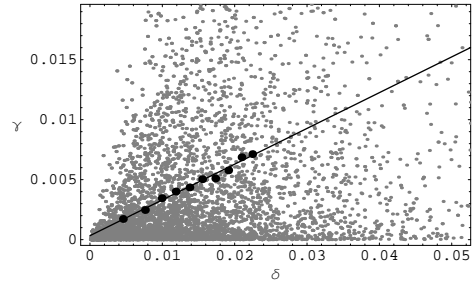


Figure 6. Regression line and scatter plot for the Mackey-Glass time series data using the embedding 111100.

data, we calculated the MSE of the network on this test data to be 0.0004 which is again in line with the Γ statistic. Using the technique in [16] with an embedding of dimension 3 on 800 data points generated by the network, we estimated the Lyapunov exponents to be $\{0.059, -0.044, -0.239\}$ to 3 decimal places. We should note that using such a short time series may produce inaccurate values of the true Lyapunov exponents but applying this technique to the training data facilitates a direct comparison between the trained neural network and the training data.

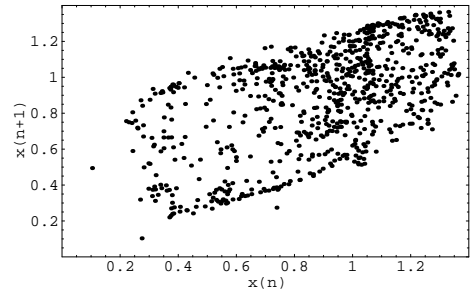


Figure 7. Phase space of $x(n+1)$ against $x(n)$ for the Mackey-Glass time series data.

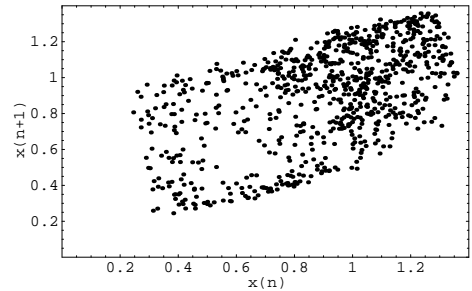


Figure 8. Phase space of the output $x(n+1)$ against $x(n)$ for the iterated Mackey-Glass 4 – 8 – 8 – 1 network.

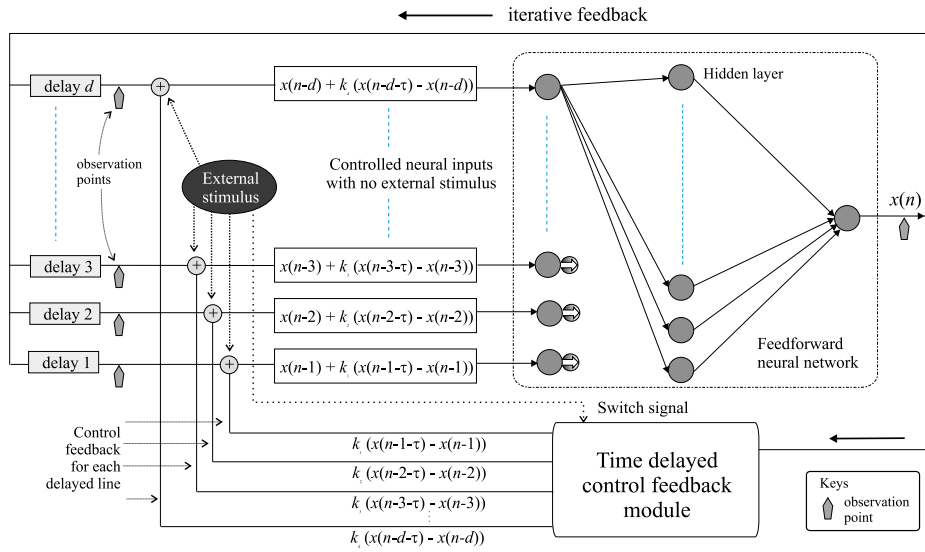


Figure 9. A general scheme for constructing a stimulus-response chaotic recurrent neural network: the chaotic “delayed” network is trained on suitable input-output data constructed from a chaotic time series; a delayed feedback control is applied to each input line; entry points for external stimulus are suggested, with a switch signal to activate the control module during external stimulation; signals on the delay lines or output can be observed at the “observation points”.

2.7. A generic chaotic neural model for stimulus-response: architecture and control

A generic scheme for such a stimulus-response recurrent network is shown in Figure 9. The single output of the network feeds back into the inputs using delay buffers according to the embedding previously determined by the Gamma test experiments. This embedding should contain enough information for predicting the next system state.

For stabilisation control a multiple (gain constant) of the delayed feedback is added to each neural network input specified by the irregular embedding, based on the idea from Pyragas’ delayed feedback control. The control module is shown in Figure 9 and the control perturbation for the i^{th} input at the n^{th} iteration is

$$k_i (x_i(n - i - \tau) - x_i(n - i)) \quad (8)$$

where k_i is a gain constant and τ is the delay time. We imagine that the presence of an external stimulus excites (activates) the control circuitry, which is otherwise inhibited. Thus to achieve a stabilised dynamical regime in response to a stimulus the control is switched on at the same time as the external signal is fed into the input line(s). By varying the external signal in small steps and holding the new setting fixed long enough for the system to stabilise we can observe the response of the network to small changes in stimulus.

In the diagram, τ is the same for each control perturbation but of course, we could set τ to be different on each control line. Stimuli can also be applied by varying k .

2.8. Delayed feedback control

We give an example of the different responses of the system. The response signals of the system can be observed at the output $x(n)$ of the feedforward neural network module or the “observation points” on the delay lines $x(n - 1), \dots, x(n - d)$, as indicated in Figure 9.

2.9. Example: Controlling the Mackey-Glass neural network

We use $\tau = 5$ and $k = 0.414144$ for our control parameters. In this example the control is applied to all delayed feedback lines. With control, but without any external stimulation, the network quickly produces a periodic response as shown in Figure 10, where control is being turned on and off. The particular orbit stabilised depends on the initial conditions.

Figure 11 shows the signal on the output $x(n)$ of the feedforward neural network module with the control signal on all lines using $k = 0.414144$, $\tau = 5$ and with external stimulation s_n added to $x(n - 5)$. This simple example already produces a rich variety of dynamical behaviours

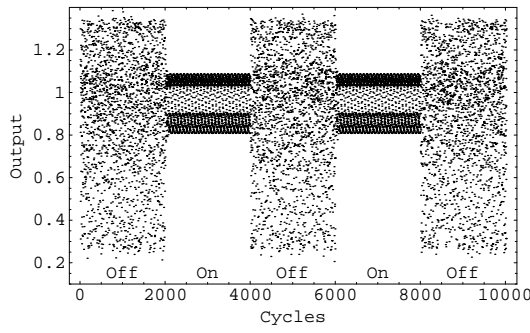


Figure 10. Response signal on $x(n - 6)$ with control signal activated on all inputs using $k = 0.414144$, $\tau = 5$ and without external stimulation.

varying in intriguing ways from low to very high periodic behaviours (see [19] for further examples).

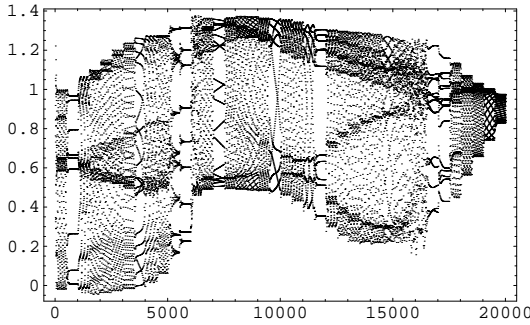


Figure 11. Response signals on network output $x(n)$, with control signal activated on all inputs using $k = 0.414144$, $\tau = 5$ and with constant external stimulation s_n added to $x(n-5)$, where s_n varies from -1 to 1 in steps of 0.05 at each 500 iterative steps after 20 initial transient steps.

2.10. Synchronisation – experimental results

We will now give an experimental result for a synchronisation experiment (using the scheme illustrated in Figure 12) and show how a suitable value for the feedback constant can be determined by examining the maximum Lyapunov exponent of the difference between the two systems [3].

2.11. Example: Synchronisation of two Mackey-Glass neural networks

The graph of maximum Lyapunov exponent against k averaged over 10 sets of initial conditions for the synchronisation of two Mackey-Glass networks is shown in Figure 13. Two copies of the Mackey-Glass neural network were

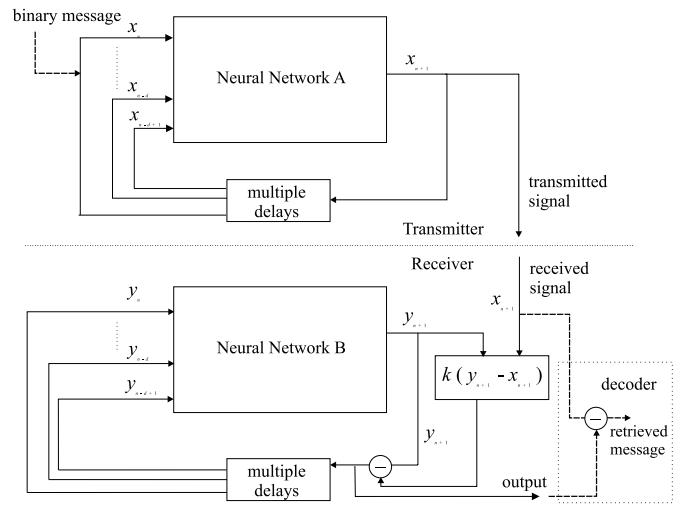


Figure 12. Synchronisation scheme.

synchronised with $k = 1.1$. The difference between the two networks quickly approaches zero as shown in Figure 14.

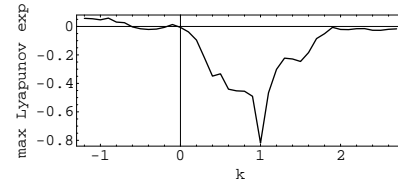


Figure 13. Average maximum Lyapunov exponent against k for the synchronisation of two Mackey-Glass networks.

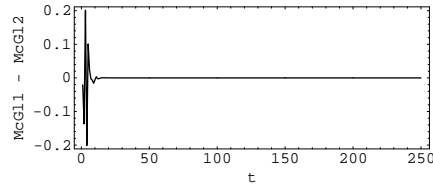


Figure 14. Synchronisation of two Mackey-Glass networks with different initial conditions with $k = 1.1$.

3. Conclusions

We have outlined a systematic methodology which reduces the construction of an iterative neural network, which closely models time series data from a given chaotic system, to a relatively automatic process. This construction method is based on finding a suitable (usually irregular) embedding using the Gamma test. Since we know the target MSE we can start with a simple network and increase the number of hidden nodes until the training algorithm reaches

the required target error. We have found that in practice a slightly modified form of the BFGS algorithm produces quite rapid convergence to the target MSE and compares very favourably with other forms of gradient descent.

Prompted by the work of Freeman [7] we are interested in biologically plausible mechanisms of stabilisation for chaotic neural networks and have illustrated that all the examples considered can be stabilised via time-delayed feedback onto to unstable periodic attractors which are to a large degree characteristic of the applied stimulus. Moreover we have found that these behaviours are relatively robust in the face of noise. We conclude that if stabilisation of chaos is significant in biological information processing then time delayed feedback is one possible mechanism by which this might be achieved. However, one should note that in the model we have described the response of the system to a particular stimulus cannot be specified *ab initio* and it remains an open question as to how such a system might be encouraged to *learn* particular responses without destroying the features of interest (such as the chaotic behaviour).

Another aspect of this model is that since the response may be a high-order periodic behaviour we could imagine a system which responds in a continuing way over a longer time scale. Thus, for example, the smell of a known predator could trigger a gait circuit and produce an evasion behaviour as long as the stimulus remained present.

In a similar way it is conjectured that *synchronisation* of different neural clusters may also play an important role in biological information processing. Again we have shown that synchronisation of two identical chaotic networks can be achieved very easily by means of time delayed feedback.

Finally, irrespective of the interest of chaotic neural synchronisation in biological information processing there is the potential application of these ideas to secure communications. We have taken one of the synchronisation examples and illustrated how a simply encoded binary message may be masked by the chaotic neural carrier and then transmitted and recovered by an identical neural system at the receiver. If chaotic carriers were to prove viable as a method of encryption then the use of neural networks may offer several advantages. In particular, to switch from one chaotic carrier to another could be done without having to reconfigure the underlying hardware.

References

- [1] Aðalbjörn Stefánsson, N. Končar, and A. J. Jones. A note on the Gamma test. *Neural Computing and Applications*, 5(3):131–133, 1997. ISSN 0-941-0643.
- [2] D. J. Amit, H. Gutfreund, and H. Sompolinsky. Information storage in neural networks with low levels of activity. *Physical Review A*, 35:2239–2303, 1997.
- [3] A. R. S. G. de Oliveira. *Synchronization of chaos and applications to secure communications*. PhD thesis, Department of Computing, Imperial College of Science, Technology and Medicine, University of London, 1999.
- [4] P. Durrant. *winGamma: a non-linear data analysis and modelling tool for the investigation of non-linear and chaotic systems with applied techniques for a flood prediction system*. PhD thesis, Department of Computer Science, Cardiff University, Wales, UK, 2000.
- [5] D. Evans. *Data derived estimates of noise using near neighbour asymptotics*. PhD thesis, Department of Computer Science, Cardiff University, Wales, UK, 2001.
- [6] R. Fletcher. *Practical methods of optimization, 2nd edition*. John Wiley, 1987.
- [7] W. J. Freeman. The physiology of perception. *Scientific American*, 1991.
- [8] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology (London)*, 117:500–544, 1952.
- [9] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79:2554–2558, 1982.
- [10] F. C. Hoppensteadt. Intermittent chaos, self-organization, and learning from synchronous synaptic activity in model neuron networks. *Proceedings of the National Academy of Sciences of the United States of America*, 86:2991–2995, 1989.
- [11] N. Končar. *Optimisation methodologies for direct inverse neurocontrol*. PhD thesis, Department of Computing, Imperial College of Science, Technology and Medicine, University of London, 1997.
- [12] K. E. Kürten and J. W. Clark. Chaos in neural systems. *Physical Letters A*, 114:413–418, 1986.
- [13] M. J. Ogorzallek. Taming chaos - part 1: Synchronization. *IEEE Transactions on Circuits and Systems - I: Fundamental Theory and Applications*, 40:693–699, 1993.
- [14] E. Ott, C. Grebogi, and J. A. Yorke. Controlling chaos. *Physical Review Letters*, 64(11):1196–1199, 1990.
- [15] K. Pyragas. Continuous control of chaos by self-controlling feedback. *Physics Letters A*, 170:421–428, 1992.
- [16] M. Sano and Y. Sawada. Measurements of the Lyapunov spectrum from a chaotic time series. *Physical Review Letters*, 72(13):1082–1085, 1985.
- [17] A. Skarda and W. Freeman. How brains make chaos in order to make sense of the world. *Behavioural and Brain Sciences*, 10:161–195, 1987.
- [18] F. Takens. Detecting strange attractors in turbulence. In D. A. Rand and L. S. Young, editors, *Dynamical Systems and Turbulence*, volume 898, pages 366–381. Lecture Notes in Mathematics, Springer-Verlag, 1981.
- [19] A. Tsui. *Smooth Data Modelling and Stimulus-Response via Stabilisation of Neural Chaos*. PhD thesis, Department of Computing, Imperial College of Science, Technology and Medicine, University of London, 1999.
- [20] X. Wang. Period-doubling to chaos in a simple neural network. *Proceedings of IJCNN-91 Seattle*, 2:333–339, 1991.