ESANN'2006 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2006, d-side publi., ISBN 2-930307-06-4.

Nonlinear dynamics in neural computation

Tjeerd olde Scheper and Nigel Crook

School of Technology - Department of Computing
Oxford Brookes University, Wheatley Campus, Oxford - United Kingdom

Abstract. This tutorial reports on the use of nonlinear dynamics in several different models of neural systems. We discuss a number of distinct approaches to neural information processing based on nonlinear dynamics. The models we consider combine controlled chaotic models with phenomenological models of spiking mechanisms as well as using weakly chaotic systems. The recent work of several major researchers in this field is briefly introduced.

1 Introduction

The use of nonlinear dynamics in models of neural systems has been studied for over a decade. Both experimentalists as well as theorists have investigated and proposed different mechanisms which would allow nonlinear dynamics to be used [1, 2]. Although the existence of chaos in neuronal systems appears to be not in doubt [3], the possible role of chaos is still under discussion [4, 5, 6]. In particular, the possible use of chaos at the core of information processing has been considered to be potentially useful [7, 8]. Even though much is now known about chaotic systems, their synchronisation and control, the next step of relating information to a stable state contained in a (controlled) chaotic system appears elusive. (For a detailed explanation of chaotic control and synchronisation see [9, 10, 11, 12, 13]). In this tutorial, we will explore several systems which provide support for the use of controlled chaotic systems as dynamic filters and transient information processing.

2 Emergent behaviour

Some recent developments in chaotic neural models is the application of controlled chaotic systems in autonomous models. The introduction of purely chaotic systems in any neural model is feasible, however, these tend to become either indistinguishable from stochastic systems or have only a particular feature of the chaotic model which is included in the resulting dynamics. Controlling specific unstable periodic orbits, upon presentation of input, such that they are reliably correlated to that particular input seems to be complicated. In many cases targeting the control towards a particular solution requires non-biologically relevant mechanisms.

Instead of applying control of a chaotic system upon input, the control can be employed continuously, in other words, the chaotic system is always under some form of control. The system becomes therefore stable periodic, even though the controlled model is only unstable periodic. The possible advantages are that the system is only semi-stable, i.e. only during the time that control is effective has it stable properties. When the control is not effective, for example when the control function is close to zero, the system does not exhibit chaotic properties but can be perturbed into different trajectories.

2.1 Dynamic patterns

To show how a dynamic behaviour may emerge from controlled chaoticly driven neurons a neuron model been derived from the Hindmarsh-Rose (HR) model [14] but includes a slow recurrent equation which represents the slow calcium exchange between intracellular stores and the cytoplasm [15]. This makes the modified Hindmarsh-Rose model (HR4) more like a chaotic Hodgkin-Huxley (HH) model of stomatogastric ganglion neurons [15]. In addition to the slow calcium current, an additional inactivation current has been added to this model, which competes with the third current to return the system to the equilibrium state. The third equation of the HR4 model is complemented with a fifth equation resulting in the five dimensional Hindmarsh-Rose model (HR5). The effect of the faster inactivation current z_f (4), compared to the slower inactivation current as used in HR4, is that the system tends to burst less. The faster current makes the system return quickly towards the equilibrium where only a larger (re)activation current can cause the system to burst. In this model, the HR5 system allows the temporal separation of spikes by increasing the refactory period. Parameter values are $a = 1$, $b = 3$, $c = 1$, $e = 1$, $f = 5$, $g = 0.0275$, $u = 0.00215$, $s = 4$, $v = 0.001$, $k = 0.9573$, $r = 3.0$, $m = 1$, $n = 1, s_{f_1} = 8$, $s_{f_2} = 1$, $n_f = 4, d_f = 0.5$, with rest-potential $x_0 = 1.605$ and variable input I. With these parameter values the model is stable in the resting potential but shows low dimensional chaos in the bursting patterns.

$$\frac{dx}{dt} = a\,y + b\,x^2 - c\,x^3 - d_s\,z_s - d_f\,z_f + I \tag{1}$$

$$\frac{dy}{dt} = e - f\,x^2 - m\,y - g\,w \tag{2}$$

$$\frac{dz_s}{dt} = u\,(s_s\,(x + x_0) - n_s\,z_s) \tag{3}$$

$$\frac{dz_f}{dt} = u\,((s_{f_1}\,(x + x_0) - s_{f_2}\,x^2) - n_f\,z_f) \tag{4}$$

$$\frac{dw}{dt} = v\,(r(y + l) - k\,w) \tag{5}$$

To introduce controlled chaotic behaviour in either the four dimensional HR4 system or the five dimensional HR5, a scaled and inverted Rössler system has been used [16]. This is necessary because the normal Rössler model has a different time scale from the HR4 model but the scaled variables are proportional to the normal Rössler parameter values. It is possible to map the time scale of the modified Rössler (R3) model to fit the time scale of the HR4 model and use the R3 system to generate patterns. In addition to the scaling, the u_r variable has been inverted to enable the convenient use of this variable as the drive for the HR4 model. Parameter values are $a_r = \frac{1}{75}$, $b_r = \frac{1}{15}$, $c_r = \frac{1}{15}$, $d_r = \frac{1}{50}$, $k_r = -0.57$, $w_r = -\frac{1}{75}$ and $p_r = -1$.

$$\frac{d\,x_r}{d\,t} = -b_r\,y_r - d_r\,u_r \tag{6}$$

$$\frac{d\,y_r}{d\,t} = c_r\,x_r + a_r\,y_r \tag{7}$$

$$\frac{d\,u_r}{d\,t} = p_r\,u_r\,x_r + k_r\,u_r + w_r \tag{8}$$

The R3 system is controlled into an unstable periodic orbit using a chaotic rate control mechanism [17]. This mechanism allows the system to exhibit different periodic orbits by limiting the rate of change of equation (8). The rate control variable σ is only different from 1 if the variables x and u are diverging rapidly, i.e. when the chaotic manifold is stretching or folding. Equation (8) is modified to (10) as shown below. The rate control parameter μ determines the strength of the rate limiting function and the parameter ξ can have different values but is usually $-2 \leq \xi < 0$. This chaotic control mechanism is very effective at stabilising different unstable periodic orbits, but not for any given value of μ and ξ. Typically used values are $\mu = 6$ and $\xi = -1$ or $\xi = -2$.

$$\sigma(x,u) = e^{\dfrac{\xi(xu)}{(u+x+\mu)}} \tag{9}$$

$$\frac{d\,u_r}{d\,t} = \sigma(x_r,u_r)\,p_r\,u_r\,x_r + k_r\,u_r + w_r \tag{10}$$

To demonstrate how these neuron models may exhibit emergent behaviour, two neurons are connected via an electrical synapse with a constant weight. Both neurons are driven by a controlled chaotic Rössler system stabilised into the same periodic orbit. Additionally, the first neuron receives periodic input of a square pulse at varying frequency. In the figures below, is shown the results of driving the mini-network with a period of 40 Hz and 33.3 Hz respectively. In all cases the chaotic control of the Rössler system is disabled at the beginning of the experiment to demonstrate the purely chaotic firing pattern and enabled at 500 ms. The control stabilises the system into a periodic orbit within a few timesteps. The periodic external pulse to the first neuron is enabled throughout.

With an external input period of 40 Hz the first neuron fires aperiodically before the control is enabled. After the control of the chaotic drive is enabled, the first neuron fires in a seemingly multi-orbit which is almost stable (figure 1*(a)*). However, the second neuron which has the same controlled chaotic drive as the first but receives input only from the first neuron ceases to fire (figure 1*(b)*). Changing the input frequency to 33.3 Hz, but leaving all else the same, the first neuron exhibits a clear multi-orbit after control is enabled (figure 1*(c)*) with a period of 1290.9 ms. The second neuron has a different orbit with a similar period but fires only three times in one period (figure 1*(d)*). Note that at 16.25 s, the second neuron fires four times but this is only a transient and it will settle into the three spike pattern at 2 s (not shown). Even though the neurons appear to be only semi-stable, in the sense that an element of noise or a transient element is present in the results, the different emergent behaviour of the second neuron is due to the response of the controlled chaotic neuron to the different external input frequencies when filtered by the first neuron.

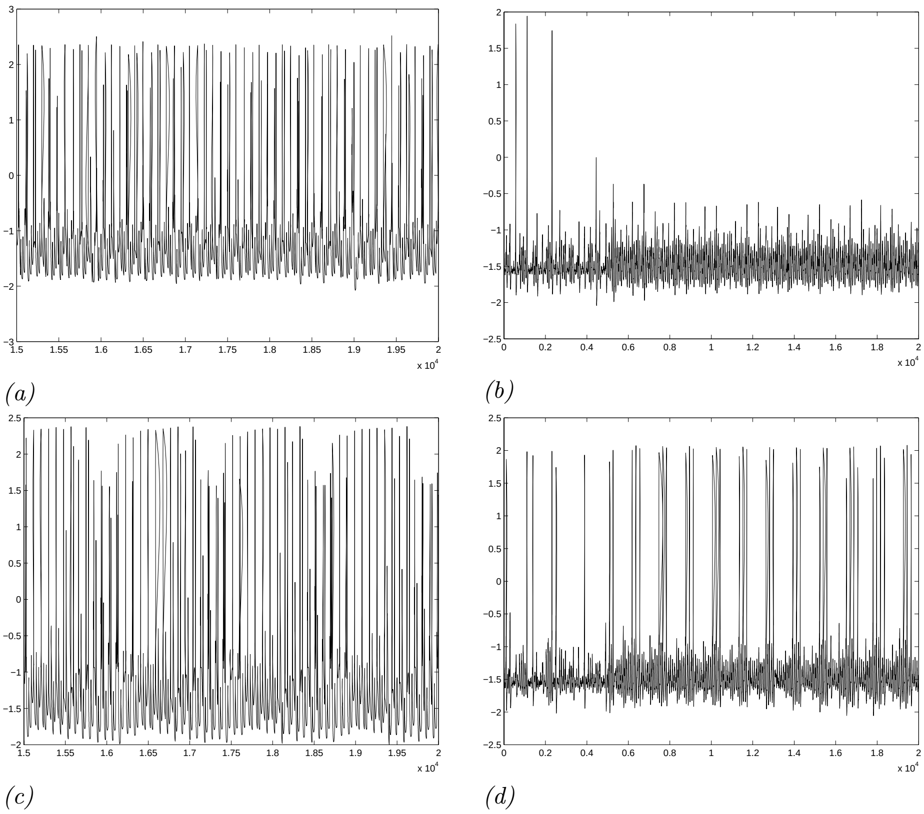

Fig. 1: *(a)* Voltage of the first neuron after chaotic control is enabled with external input of 40 Hz. *(b)* Voltage of the second neuron when control is enabled at 500 ms, input to the first neuron is 40 Hz. *(c)* Voltage of the first neuron after chaotic control is enabled with external input of 33.3 Hz. *(d)* Voltage of the second neuron after chaotic control is enabled with input to the first neuron of 33.3 Hz.

2.2 Membrane Computational Units

One aspect of neural modelling which has been considered to be less relevant to information processing is the signal conductance along the membrane. Using cable models and compartmental models, the possible unique properties of the membrane itself as computational unit are neglected. If we consider the membrane as a dynamic system with localised adaptation, we can formulate a membrane unit consisting of several components, such as ion channels and receptors, which together may act as a computational unit [18, 19]. With the aim of simulating computational processes within a membrane computational unit (MCU), we have build a phenomenological unit based on the Hindmarsh-Rose and Rössler models used above. Each model of an MCU has different components that may act together to produce a system which is capable of complex emergent behaviour. It generally consists of a spike generation component and an optional controlled chaotic drive component, i.e. an HR5 or HR4 model with or without R3 system.

By linking five computational units together a model may be built which synchronised two seperate inputs (SyncMCU). Two units, HR4R3-1 and 2, are made from four dimensional HR4 systems, driven by a controlled scaled Rössler system R3. Another unit, HR5-AND, consists of a single HR5 system, without a controlled chaotic drive, but electrically connected to units HR4R3-1 and 2. A fourth unit, HR4-ANDNOT, consists of a four dimensional HR4 system but with a scaled R3 drive. It receives input from units HR4R3-1 and 2. Lastly, the fifth unit, HR4, is a normal HR4 system without R3 drive, that only receives input from unit HR5-AND. All the R3 drive systems are controlled in the same unstable periodic orbit but the driving scalar is small such that by itself it does not cause the system to fire. The R3 systems may therefore act as a localised subcellular clock that can be in or out of sync with other units.

This configuration may act as a detector of desynchronisation of two input signals. Given an additional external input to the units HR4R3-1 and 2, which are combined in unit HR5-AND and then passed on to unit HR4, the unit HR4-ANDNOT will detect if unit HR4R3-2 fires but HR4R3-1 does not. Note that if they both fire, HR4-ANDNOT does not fire unless it has fired recently. We can now use this to attempt to synchronise unit HR4R3-2 with unit HR4R3-1 even if they have completely different periods.

To enable unit HR4-ANDNOT to synchronise the units HR4R3-1 and 2, a synchronisation function is defined as

$$\frac{d\,S}{d\,t} = \kappa_1(x_r^1 - x_r^2)\theta(x) - \kappa_2 S \tag{11}$$

where κ_1 and κ_2 are the growth and decay parameters, x_r^n are the x_r variables of the controlled chaotic scaled Rössler systems of the units that are synchronised. The function $\theta(x)$ is a threshold function on the x variable of the HR4 system of the unit HR4-ANDNOT. Parameters for (11) are $\kappa_1 = -0.75, \kappa_2 = 0.5$ with the threshold set at -0.5.

In the synchronised case, as shown in figures 2*(c)* and *(d)*, the emerging patterns are corrected by the synchronisation pulses HR4-ANDNOT unit and is much less noisy than in the unsynchronised case (not shown).

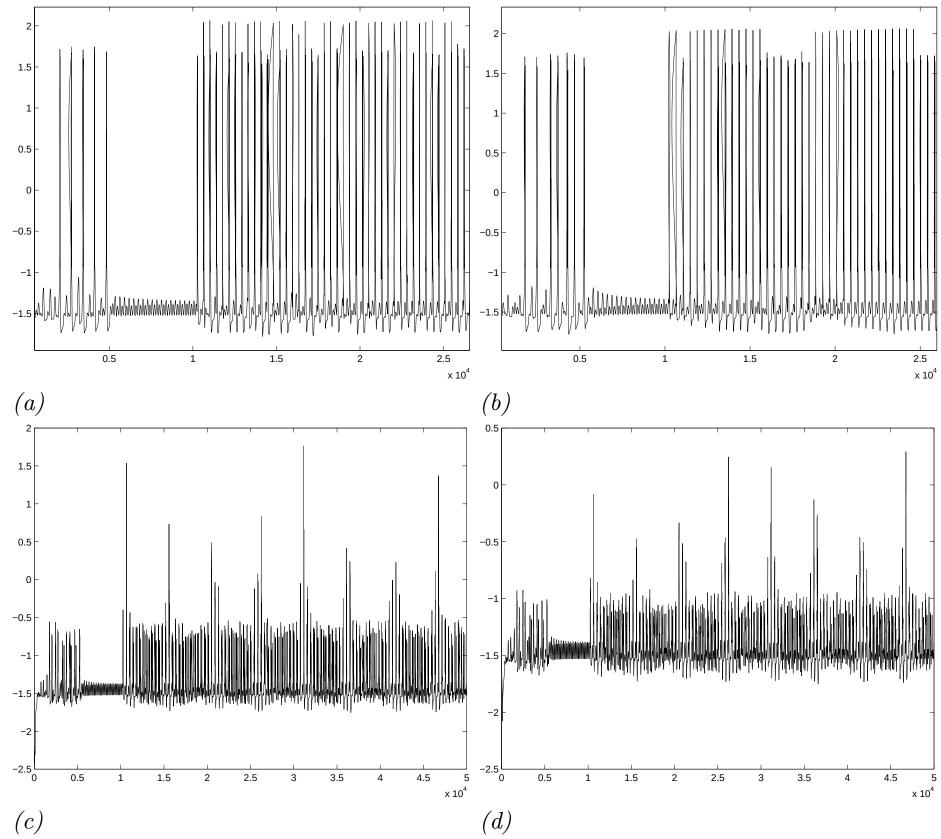

Fig. 2: SyncMCU model with synchronisation; *(a)* x variable of HR4R3-1; *(b)* x variable of HR4R3-2; *(c)* x variable of HR5-AND; *(d)* x variable of HR4-ANDNOT

3 Transient computation

The term *transient computation* describes an approach to information processing in which time dependent input signals cause a deviation in the dynamics of a system. To enable computation, this deviation or transient must in some sense be proportional to the input signal that caused it (see *separation* (SP) and *approximation* (AP) properties below). Devices which perform transient computation in this way have recently received much interest. Most notable among these in the context of neural computation are the liquid state machine (LSM) developed by Maass [5], the echo state machine (ESM) developed by Jaeger [20, 21], and the nonlinear transient computation machine (NTCM) developed by Crook [6].

Both LSM and ESM approaches to transient computation use a recurrently connected pool or *reservoir* of neurons which perform a temporal integration of input signals. An important aspect of these neural reservoirs is that they constitute a *fading memory*; that is, input signals have a residual effect on the dynamics of the reservoir which fades with time. The neural dynamics which ensue after the input is presented to the reservoir are referred to as the *liquid state* in LSMs or *echo state* in ESMs. There are two properties of these dynamic states which are both necessary and sufficient for machines that perform real-time computation using transient dynamics: They are the *separation* property (SP) and the *approximation* property (AP) [5]. The separation property guarantees that two different inputs to the reservoir will result in two different transients in the dynamics of the reservoir. Specifically, it assures that the degree of separation in the corresponding transients in the dynamics of the reservoir is proportional to the differences in the inputs.

Maass et al tested the separation property of the LSM through a series of experiments using large numbers of randomly generated Poisson spike trains in pairs $u(\cdot)$ and $v(\cdot)$ [5]. Each spike train was presented as input to the M neurons in the reservoir in separate trials. The transients in the dynamics of the reservoir $x_u^M(\cdot)$ and $x_v^M(\cdot)$ caused by $u(\cdot)$ and $v(\cdot)$ respectively were recorded in each case. The average distance $\|x_u^M(t) - x_v^M(t)\|$ between the two transients in each pair was then plotted as a function of time. The measure of distance $d(u, v)$ between spike trains u and v is calculated by converting each spike train to a convolution of Gaussians and using the L_2 -norm as a measure of distance between them. The convolution of Gaussians is constructed by replacing each spike in the spike train by a Gaussian curve centered on the spike time using the kernel $exp(-(t/\tau)^2)$ where $\tau = 5ms$. The Gaussians are summed to produce a continuous curve over the length of the spike train.

The results given in [5] clearly show that the distance between the transients evoked by input $u(\cdot)$ and $v(\cdot)$ is proportional to the distance between $u(\cdot)$ and $v(\cdot)$ and is well above the level of noise (i.e. when $d(u, v) = 0$ and the differences in the transience for $u(\cdot)$ and $v(\cdot)$ are caused solely by the differences in the initial conditions of the reservoir). These results confirm that the LSM possesses the required separation property SP.

The second necessary and sufficient condition for machines which perform computations on dynamic transients is that they possess an *approximation* property AP [5]. This property is concerned with the ability of the output mechanism of the LSM to differentiate and map internal states of the reservoir to specific target outputs. The output component of the LSM is a *memoryless* readout map f^M which transforms the state of the reservoir $x^M(t)$ to the output signal $y^N(t) = f^M(x^M(t))$ at each time step t. The readout map is implemented as a set of N readout neurons, each of which is configured to signal the presence of a recognised input pattern. Each readout neuron receives weighted instantaneous input from all the neurons in the reservoir. The weights are devised using a simple perceptron-like learning mechanism.

The readout mechanism is considered to be memoryless because it does not have access to previous states of the reservoir caused by earlier inputs $u(s)(s < t)$ to the LSM. However, because the reservoir naturally acts as a fading memory, echoes of these previous states are contained in the current state $x^M(t)$ of the reservoir, and hence are available to the readout mechanism at each time step.

Maass et al demonstrate the approximation property AP of the LSM both through theoretical results and experimental evidence [5]. One of these experiments involved the classification of five prototype patterns each consisting of 40 parallel Poisson spike trains. Five readout modules were constructed each consisting of 50 integrate-and-fire neurons. Each module was trained to respond to one of the five prototype patterns. The training was done using 20 noisy versions of each of the prototypes. During training, the initial state of the neurons in the reservoir was randomised at the beginning of each trial. The results presented in [5] demonstrate that the readout modules produces responses which correctly differentiate between the five prototype patterns, thereby demonstrating that the LSM possesses the approximation property AP.

Importantly, Maass et al present theoretical justifications to suggest that there are no serious a-priori limits for the computational power of LSMs on continuous functions of time [5].

An alternative approach to transient computation is presented by Crook [6]. Instead of using large pools of recurrently connected neurons, this approach, referred to as the nonlinear transient computation machine (NTCM), uses just two neurons whose internal dynamics are weakly chaotic. This means that nearby points in the phase spaces of these neurons will diverge at a relatively low exponential rate. Consequently, the transients caused in the neuron's dynamics by similar inputs will initially evolve in a similar way. Only later in the evolution will these transients begin to diverge significantly. The fact that these neurons are weakly chaotic has important consequences on their ability to handle noise [22]. More significantly, it has been shown by Bertschinger et al [23, 24] that systems that are on the *edge of chaos* possess extensive computational capabilities (see below).

The NTCM is a novel device for computing time-varying input signals. It consists of two coupled neurons, one of which acts as a pacemaker (denoted N_P) and the other provides the locus of the transients (denoted N_T). The purpose of the pacemaker (N_P) is to lead the transient neuron (N_T) into a periodic firing pattern through synchronisation. While external input is being presented to N_T, the coupling from N_P is temporarily removed. The external input perturbs the internal state of N_P which will subsequently evolve along a transient away from the periodic firing pattern induced by N_P. This transient is reflected in the output spike train of N_T. After the input has been presented the coupling with N_P is gradually restored and as N_T begins to converge back to the original synchronised periodic firing pattern, the effects of the external input on its internal dynamics fade and eventually disappear. In this way the NTCM possesses a fading memory similar to that found in LSMs and ESMs. Details of the NTCM model are presented in [22, 6]. This tutorial will focus primarily on the experimental evidence that demonstrates that the NTCM has the separation (SP) and approximation properties (AP) which are both necessary and sufficient for real-time computation using transient dynamics.

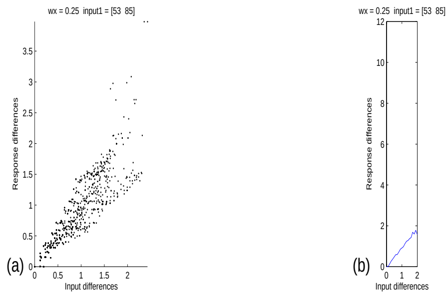

Fig. 3: The separation property for (a) 1000 random inputs (b) average of the 1000 random inputs.

The following set of experiments demonstrate that the NTCM possesses the property of separation SP. In these experiments the NTCM is presented with a randomly generated spike train S_0 of duration 100 time steps and consisting of between 2 and 4 spikes. The corresponding spike output of N_T is recorded during the time window [1..100]. 1000 randomized versions of S_0 are then presented to the NTCM and the response of N_T in each case is recorded for the same time window. The random versions of S_0 were constructed by introducing random *jitter* to the timing of the spikes in S_0. The jitter involved shifting the timing of spikes by from ± 1 to ± 20 time steps. The results from some of these experiments are presented in Figure 3. The x axis of each graph represents the distance of the randomized input spike trains from S_0 calculated using the L_2-norm of the convolution of Gaussians approach reported earlier. The corresponding value in the y axis is the distance of the response of N_T to the randomized spike train from the response evoked by S_0. The times of the spikes for S_0 are shown in the header of each graph.

The results in Figure 3 show that increases in the distance between the multiple spike input patterns given to the NTCM effect proportional increases in the distance between the corresponding output spike trains of N_T. This suggests that the property of separation SP holds for the NTCM.

The approximation property AP of the NTCM is demonstrated by adding a layer of readout neurons to the model. A unique feature of the readout mechanism used here is that not only will they signify the presence of a recognized input pattern, but they will also give a rough indication of the level of noise present in that pattern. This is done using the NTCM's ability to be both noise robust and noise sensitive within the same output spike train as discussed in [22].

The readout set is constructed using three Spike Response Model (SRM) neurons [25], each sensitive to a particular sub-range (or zone) of the spike train emitted by N_T. The first SRM is sensitive to spikes in the first 100 time steps of N_T's spike train. The second is responsive to spikes within the [50..150] time step window. The third is responsive to spikes in the [100..200] window. In [6] has already been shown that the first 100 time steps of N_T's spike output is quite robust to noise and that as the spike train evolves it becomes increasingly more sensitive to noise. In the present model this would mean that all three readout neurons should fire if the input closely matches the recognized pattern. As noise is introduced in the input, the third readout neuron will cease to respond but the other two should recognize the pattern. As the noise is increased further the second readout neuron will also cease to respond but the first neuron should continue to fire.

In these experiments the model is constructed by first presenting a prototype pattern to be recognized as input to the NTCM. The spike train output of N_T is then used to construct multiple time-delay connections from N_T to each of the readout neurons. The delays in these connections are devised so that the specific timings of the output spikes that occur within the sensitive zone of each readout neuron for this prototype input pattern have a coincident above-threshold effect on that readout neuron.

The prototype patterns consist of five independent spike trains, each containing up to 4 randomly timed spikes within the period [1..100]. Noisy versions of the prototypes were constructed by adding jitter to the timing of each spike. The jitter was determined using white Gaussian noise with a mean of 0.

The results of these experiments are presented in detail in [6, 22]. The results demonstrate that readout mechanism consistently responded correctly to the jittered versions of the prototype pattern even in the presence of strong noise. Through these and other similar experiments the readout mechanism of the NTCM consistently demonstrates an ability to differentiate and map transients of the N_T to specific target outputs, thereby indicating that the model possess the required approximation property.

The relationship between the computational power of a system and its stability has been the subject of much debate in recent years [26, 27, 28, 29, 30, 31, 24]. Some have argued that the computational properties of a system become optimal as the dynamics of the system approach *the edge of chaos*; most notably Langton [28] and Packard [29] did some early work on this with cellular automata. Packard studied the frequency of evolved cellular automata rules as a function of Langton's λ parameter [28]. For low values of λ the rules are attracted to a fixed point. As λ is increased, the rules settle down to form periodic patterns. As λ is further increased and it approaches a so called critical value λ_c, the rules became unstable and tended to have longer and longer transients. Packard concluded from his results that the cellular automata rules which are able to perform complex computations are most likely to be found at the near critical value λ_c where the rule dynamics were on the edge of chaos. Mitchell et al [30] subsequently showed that Packard's conclusions from his particular experimental results were unfounded.

The debate about computation at the edge of chaos has recently been revisited by Natschläger et al [24] who studied the relationship between the computational capabilities and the dynamical properties of randomly connected networks of threshold gates. They proposed a measure of complexity which was maximal for a network at the point of transition in its dynamics from periodic to chaotic. Experimental results showed that this complexity measure was able to predict the computation capabilities of the network extremely accurately. Specifically, the measure showed that only when the dynamics of the network were near the edge of chaos was it able to perform complex computations on time series inputs.

4 Conclusion

This tutorial has given an overview of recent work which places nonlinear dynamics at the heart of neural information processing. Naturally, it has not been possible to cover all of the research that is being done in this area. For example, we have not included the work of those who use chaos as a basis for neural itinerancy; which is a process involving deterministic search through memory states [32]. Neither have we reported on the use of the bifurcating properties of specific chaotic systems as a means of switching between neuronal states [33]. However, we have attempted to include samples of work which focus at different neural levels (membrane, cell and network), and have we tried to give a flavour of the direction in which we see the field is moving.

References

[1] W.J. Freeman. Neural networks and chaos. *Journal of Theoretical Biology*, 171:13–18, 1994.

[2] K. Aihara, T. Takabe, and M. Toyoda. Chaotic neural networks. *Physics Letters A*, 144(6,7):333–339, 1990.

[3] M.A. Arbib. *Neural organization.* A Bradford Book, 1998. ISBN 0-262-01159-X.

[4] B. Biswal and C. Dasgupta. Neural network model for apparent deterministic chaos in spontaneously bursting in hippocampal slices. *Physical Review Letters*, 88(8):1–4, 2002.

[5] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

[6] N.T. Crook. Nonlinear transient computation. *Submitted to Neural Computation*, 2005.

[7] F. Pasemann and N. Stollenwerk. Attractor switching by neural control of chaotic neurodynamics. *Network: Computational Neural Systems*, 9:549–561, 1998.

[8] T.V.S.M. olde Scheper, N.T. Crook, and C. Dobbyn. Chaos as a desireable stable state of artificial neural networks. In M. Heiss, editor, *International ICSC/IFAC Symposium on Neural Computation (NC'98)*, pages 419–423, Vienna University of Technology, Austria, September 1998. ICSC, ISCS Academic Press.

[9] T. Kapitaniak. *Controlling chaos.* Academic Press Ltd., 1996. ISBN 0-12-396840-2.

[10] A. Katok and B. Hasselblatt. *Introduction to the Modern Theory of Dynamical Systems*, volume 54 of *Encyclopedia of Mathematics and its Applications.* Cambridge University Press, 1995. ISBN 0-521-57557-5.

[11] A. Kittel, J. Parisi, and K. Pyragas. Delayed feedback control of chaos by self-adapted delay time. *Physics Letters A*, 198:433–436, 1995.

[12] E. Ott. *Chaos in dynamical systems.* Cambridge University Press, 1993. ISBN 0-521-43799-7.

[13] K. Pyragas. Control of chaos via an unstable delayed feedback controller. *Physical Review Letters*, 86(11):2265–2268, 2001.

[14] J. L. Hindmarsh and R. M. Rose. A model of neuronal bursting using three coupled ?rst order differential equations. *Proc. R. Soc. London*, B221:87–102, 1984.

[15] RD Pinto, P Varona, AR Volkovskii, A Szucs, HDI Abarbanel, and MI Rabinovich. Synchronous behavior of two coupled electronic neurons. *Physical Review E*, 62 (N2 PTB):2644–2656, 2000.

[16] O. E. Rössler. An equation for continuous chaos. *Physics Letters*, 57A(5):397–398, 1976.

[17] Tjeerd olde Scheper. Rate control of chaotic systems. *submitted*.

[18] Lyle J. Graham and Raymond T. Kado. *The Handbook for Brain Theory and Neural Networks*, chapter The neuron's biophysical mosaic and its computational relevance, pages 170–175. MIT Press, 2nd edition, 2002.

[19] Tjeerd olde Scheper. The spike generation processes: a case for low level computation. In *Proceedings of the European Conference on Mathematical and Theoretical Biology (ECMTB)*, 2005.

[20] H. Jaeger and H. Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science*, 304:78–80, 2004.

[21] H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks. Technical Report GMD Report 148, German National Research Center for Information Technology, 2001.

[22] N.T. Crook. Nonlinear transient computation and variable noise tolerance. In M. Verleysen, editor, *Proceedings of 14th European Symposium on Artificial Neural Networks (ESANN'2006)*, Bruges, April 2005. d-side, Belgium.

[23] Nils Bertschinger and Thomas Natschläger. Real-time computation at the edge of chaos in recurrent neural networks. *Neural Comput.*, 16(7):1413–1436, 2004.

[24] Nils Bertschinger, Thomas Natschläger, and Robert A. Legenstein. At the edge of chaos: Real-time computations and self-organized criticality in recurrent neural networks. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 145–152. MIT Press, Cambridge, MA, 2005.

[25] W. Gerstner. Associative memory in a network of 'biological' neurons. *Advances in Neural Information Processing Systems*, 3:84–90, 1991.

[26] J.P. Crutchfield and K. Young. Computation at the onset of chaos. In W. Zurek, editor, *Entropy, Complexity, and the Physics of Information*, SFI Studies in the Sciences of Complexity, VIII, pages 223–269. Addison-Wesley, Reading, Massachusetts, 1990.

[27] S.A. Kauffman and S. Johnsen. Co-evolution of the edge of chaos: Coupled fitness landscapes, poised states, and co-evolutionary avalanches. *J. Theor. Biol*, 149(4):467–505, 1991.

[28] Chris G. Langton. Computation at the edge of chaos: phase transitions and emergent computation. *Phys. D*, 42(1-3):12–37, 1990.

[29] N.H. Packard. Adaptation toward the edge of chaos. In J.A.S.Kelso, A.J. Mandell, and M.F. Shlesinger, editors, *Dynamic Patterns in Complex Systems*, pages 293–301. World Scientific, Singapore, 1988.

[30] M. Mitchell, P.T. Hraber, and J.P Crutchfield. Revisiting the edge of chaos: Evolving cellular automata to perform computations. *Complex Systems*, 7:89–130, 1993.

[31] H. Soula, A. Alwan, and G. Beslon. Learning at the edge of chaos : Temporal coupling of spiking neuron controller of autonomous robotics. In *AAAI Spring Symposia on Developmental Robotics*, Stanford, CA, 2005.

[32] O. Hoshino, N. Usuba, Y. Kashimori, and T. Kambara. Role of itinerancy among attractors as dynamical map in distributed coding scheme. *Neural Networks*, 10(8):1375–1390, 1997.

[33] G. Lee and N.H. Farhat. The bifurcating neuron network 2: an analog associative memory. *Neural Networks*, 15(1):69–84, January 2002.