REFERENCE ONLY

UNIVERSITY OF LONDON THESIS

Degree MPhil Year 2005 Name of Author MAEI H.L.

HOW CAN REALISTIC NETWORKS PROCESS TIME-VARYING SIGNALS?

by

Hamidreza Maei

Submitted to the Gatsby Computational Neuroscience Unit
in partial fulfillment of the requirements for the degree of

Master of Philosophy of University of London

at the

UNIVERSITY COLLEGE LONDON

October 2005

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

HOW CAN REALISTIC NETWORKS PROCESS TIME-VARYING SIGNALS?

by

Hamidreza Maei

Submitted to the Gatsby Computational Neuroscience Unit
on 10 October , 2005, in partial fulfillment of the
requirements for the degree of
Master of Philosophy of University of London

Abstract

The brain is easily able to process and categorise complex time-varying signals. For example, the two sentences "it is cold in London this time of year" and "it is hot in London this time of year" have different meanings, even though the words "hot" and "cold" appear about 3000 ms before the ends of the two sentences. A network that can perform this kind of processing must, therefore, have a long memory. In other words, the current state of the network must depend on events that happened many seconds ago. This is particularly difficult because neurons are dominated by relatively short time constants – 10s to 100s of ms. Recently Jaeger and Haas [2004] (see also Jaeger [2001]) and Maass et al. [2002, 2004] proposed that randomly connected networks could exhibit the long memories necessary for complex temporal processing. This is an attractive idea, both for its simplicity and because little fine tuning is required. However, a necessary condition for it to work is that the underlying network dynamics must be non-chaotic; that is, it must exhibit negative Lyapunov exponents [White et al., 2004, Bertschinger and Natschläger, 2004]. Real networks, though, tend to be chaotic [Banerjee, 2001a,b], an observation that we have corroborated based on an extension of the analysis used by Bertschinger and Natschläger. Real networks also tend to be very noisy – they exhibit synaptic failures 10-90% of the time in the central nervous system[Walmsley et al., 1987, Volgushev et al., 2004]. The question we ask here, then, is: given the chaotic dynamics and high noise intrinsic to biologically realistic networks, can randomly connected networks exhibit memories that are significantly longer than the time constants of their constituent neurons?

Acknowledgements

I express sincere appreciation to Dr. Peter Latham for his guidance, insight and supervision throughout this research. I am specifically indebted to him for devoting his time for reading the manuscript and giving insightful and valuable advise on English style and structure of the thesis.

I want to thank Gatsby Unit for providing an active scientific environment and giving me opportunity to commence this thesis and also providing financial support through Gatsby Charitable foundation.

I would like to express my gratitude to all those who gave me the possibility to complete this thesis. Thanks go to Dr. John Pelan and Iain Murray for their assistance in network and software problems. Also I would like to thank Dr. Zoubin Ghahramani and Dr. Maneesh Sahani for some scientific discussions. Moreover, I must specifically thank Chu Wei and Alexandra Boss for their friendly helps and also thank Nathaniel Daw and Misha Ahrens.

I can not leave this section without appreciation from my friends and the faculty members at Sharif University of Technology and Brandies University (where I received my BSc. and MA degrees in physics), for providing friendly and active scientific environment.

Lastly, but in no sense of least, I must express my gratitude, thanks and appreciation to my parents for their understanding, motivation and patience. Also, I would like to give my special appreciation to my wife Farzaneh whose love, patience and support enabled me to complete this thesis.

Contents

List of Figures

Chapter 1

Introduction

1.1 Temporal pattern processing

Temporal processing in the brain is important for animal and human survival. For example, insects such as crickets use acoustic signals for mate-finding [Pollack, 1998]; bats use echolocation to locate and identify prey [Goldman and Henson, 1977]; and humans communicate by language to pass information and escape from danger.

The above examples indicate how important temporal processing is. However, the problem of temporal processing in the brain is not yet well understood. This is because:

- Unlike other sensory modalities, such as vision, audition and speech, there is no particular area (or areas) devoted to temporal processing. Such processing takes place in distributed areas in the brain, mixing with other sensory systems [Ivry and Spencer, 2004, Mauk and Buonomano, 2004] (see Fig.1-1). For example, in speech recognition, temporal pattern processing takes place in speech, auditory, and some other interconnected areas [Fitch et al., 1997].

- Most work have been devoted to lesion and imaging studies, and there have been few physiological experiments. Thus, little is known about the underlying activity of neurons that process time-varying signals.

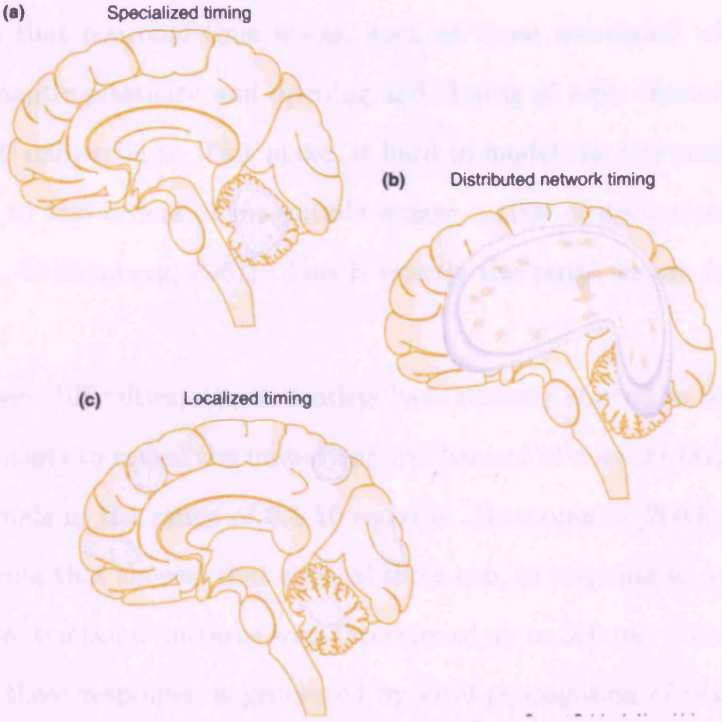


Figure 1-1: Hypothetical ideas about temporal processing in the brain. **a**) shows that temporal processing is devoted to a particular area in the brain. **b**) shows distributed regions that are mixed together to process temporal patterns. **c**)indicates that temporal processing for each particular task is devoted to particular region in the brain. Experimental studies indicate that panel **c** is likely to be the general framework for temporal processing [Ivry and Spencer, 2004, Mauk and Buonomano, 2004]. This picture is adapted from Ivry and Spencer [2004].

The depth of our understanding of temporal processing depends on the time scale of interest [Mauk and Buonomano, 2004, Grothe and Klump, 2000] (see Fig.1-2). For example, for very short time scales such as microseconds, temporal processing is well

understood in terms of delay lines [Carr, 1993]. And for very long scales we know that circadian rhythms, mediated by molecular processes, are involved in temporal processing [Mauk and Buonomano, 2004]. However, on scales of 100s of milliseconds to seconds, it is not yet clear how networks of neurons process temporal signals. One reason is that neuronal time scales, such as those associated with membrane potentials, synaptic plasticity and opening and closing of ionic channels, are on the order of 10-100 milliseconds. This makes it hard to model the representation of time on scales one to two orders of magnitude longer – 100s of milliseconds to seconds [Pollack, 2001, Braitenberg, 1967]. This is exactly the range we are interested in in this thesis.

Despite these difficulties, neuroscientists have recently started to conduct physiological experiments to reveal the underlying mechanism of neurons that process time dependent signals in the range of 0.1-10 seconds. Buonomano [2003] carried out *in vitro* experiments that showed that cortical slices can, in response to a stimulus, produce repeatable temporal patterns with latencies of up to 300 ms. He concluded that the timing of these responses is generated by local propagation of neuronal activity through a circuit or circuits.

In another study, Leon and Shadlen [2003] investigated a very simple kind of temporal processing – interval timing. They recorded from neurons in posterior parietal cortex while monkeys performed a discrimination task in which they were required to make eye movements based on their estimate of elapsed time. They found that: i) neurons in posterior parietal cortex represent elapsed time relative to a remembered duration – previously, it was thought that temporal processing only happens in the basal ganglia and cerebellum [Ivry and Spencer, 2004]; ii) the representation of time could be found in the activity of "one neuron"; iii) the behavioural and neural

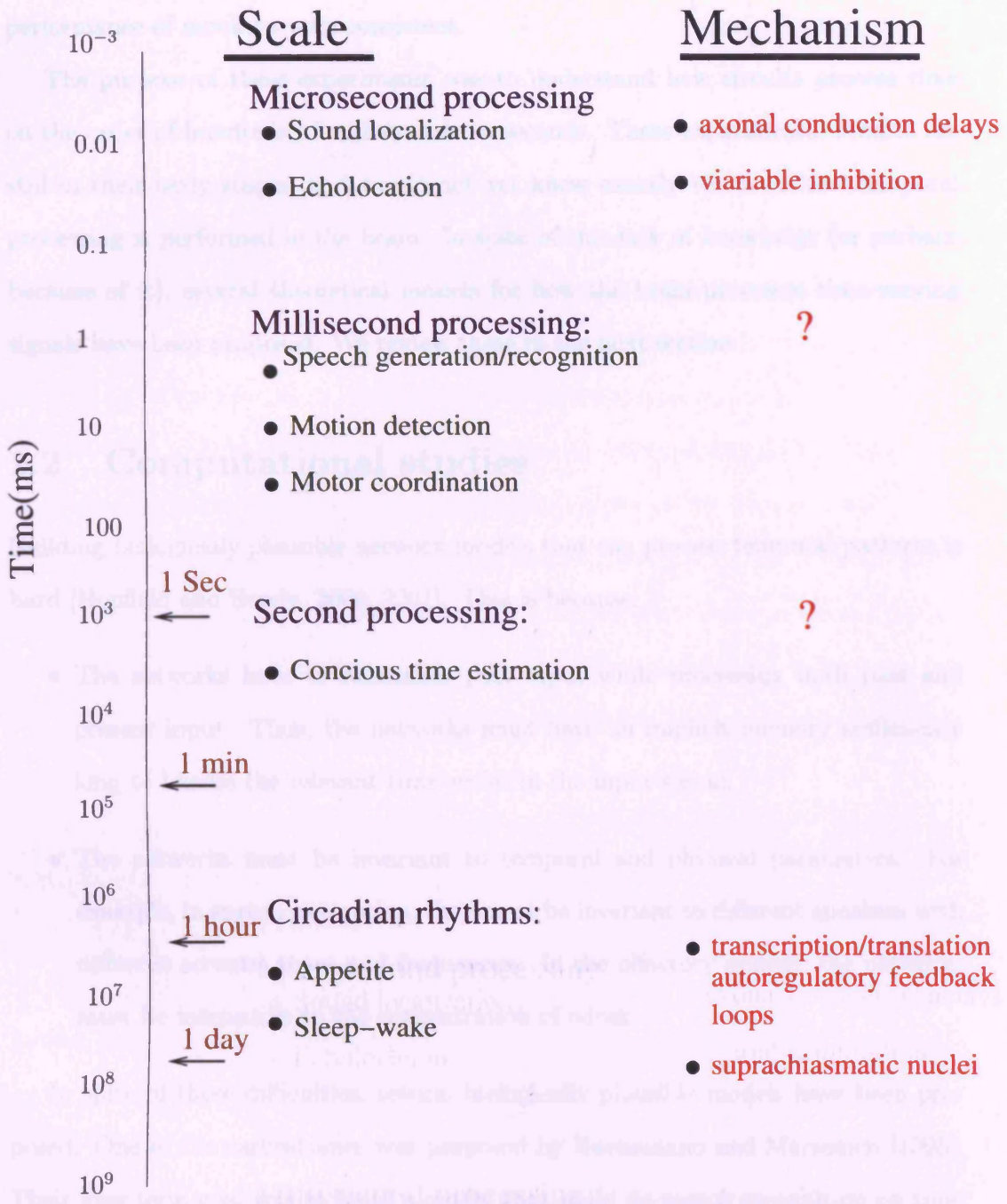


Figure 1-2: Temporal processing is categorised on four time scales [Mauk and Buonomano, 2004]. The temporal scales which are on the order of hundreds of milliseconds to several seconds, such as speech discrimination, are the focus of this thesis. Unfortunately, because of the complexity of neural activities, the mechanisms underlying temporal processing in this regime are not well known. This figure was reconstructed from Mauk and Buonomano [2004].

performance of monkeys were consistent.

The purpose of these experiments was to understand how circuits process time on the order of hundreds of milliseconds to seconds. These experimental studies are still in their early stages, and we do not yet know exactly where or how temporal processing is performed in the brain. In spite of this lack of knowledge (or perhaps because of it), several theoretical models for how the brain processes time-varying signals have been proposed. We review these in the next section.

1.2 Computational studies

Building biologically plausible network models that can process temporal patterns is hard [Hopfield and Brody, 2000, 2001]. This is because:

- The networks have to remember past input while processing both past and present input. Thus, the networks must have an implicit memory sufficiently long to handle the relevant time scales in the input signal.

- The networks must be invariant to temporal and physical parameters. For example, in speech perception, they must be invariant to different speakers with different accents, tones and frequencies. In the olfactory system, the networks must be insensitive to the concentration of odour.

In spite of these difficulties, several biologically plausible models have been proposed. One of the earliest ones was proposed by Buonomano and Merzenich [1995]. Their long term goal was to build a model that could do speech recognition on time scales of hundreds of milliseconds. As a first step in this direction, they considered a simple task, which was to build a network that is selective to temporal intervals. For

this they used a two layer network, with each layer consisting of recurrently connected (with random connectivity) spiking neurons. The principle elements of their network were slow inhibitory post-synaptic potentials (IPSPs) and paired-pulse facilitation (PPF). To determine the performance of the network, they connected read-out neurons to the second layer of the recurrent network and adjusted the weights so that different read-out neurons would recognize different intervals. They used empirical data for the network parameters and showed that, with 200 neurons in each layer, the network could perform temporal interval discrimination on intervals up to 300 ms. Whether this architecture could be used to process temporal signals on longer time scales, say by increasing the number of neurons or layers, is an open question.

Later, Jaeger and Haas [2004](see also Jaeger [2001]) and Maass et al. [2002] proposed independently that randomly connected networks operating on the edge of chaos – that is, with a Lyapunov exponent near zero – could do complex temporal processing. Their proposal came out of the observation that activity in randomly connected networks is a function of past input, so in principle one should be able to look at the activity and infer something about that input. Importantly, "look at the activity" can mean "use a linear read-out" [Buonomano and Merzenich, 1995, Maass et al., 2002].

To implement this proposal, Maass et al. [2002] used a multilayer, randomly connected network with spiking neurons and an architecture similar to that of Buonomano and Merzenich [1995]. The only differences was that they did not use slow IPSPs or PPF, and each neuron made only a few connections (2-10) to other neurons. Using the same analysis as Buonomano and Merzenich [1995] – trained read-out neurons – they showed that such networks could process input on times scales up to ~500 ms. When plasticity was added to the synaptic connections, the time scales

were even longer – up to seconds [Maass et al., 2002, 2004]. Moreover, the network was robust to noise.

The studies mentioned above were largely numerical. To provide more solid theoretical footing, as well as a deeper understanding of temporal processing in general, Bertschinger and Natschläger [2004] used largely analytic techniques to study McCulloch and Pitts's neurons [McCulloch and Pitts, 1943] in randomly connected networks. They showed that such networks can operate on the edge of chaos, and, therefore, exhibit long temporal memory. A key feature of Bertschinger and Natschläger [2004]'s work was that they used a simplified model that they could fully analyse. However, the neurons in their network made only a few connections to other neurons, which is not consistent with the high connectivity observed in the brain. In this thesis we analyse a similar reduced model, but in the more biological regime of high connectivity. We provide rigorous theoretical conditions that tell us when such a network can and cannot exhibit long temporal memory.

1.3 Thesis focus and goals

We are interested in understanding how to build networks that are reasonably biologically plausible and can exhibit long temporal memory, on the order of seconds. Our starting point is the network proposed by Bertschinger and Natschläger [2004], and what we do is extend their analysis in two directions. The first is to consider high connectivity. When we do that, we find that large, randomly connected networks are always chaotic. This implies that such networks cannot operate on the edge of chaos, and, therefore, cannot exhibit long temporal memory.

The implications of this result is that, to perform complex temporal processing,

networks must be sparse, meaning the number of connections per neuron must be fixed as the network size (number of neurons) grows. In this regime, randomly connected networks *can* operate on the edge of chaos, and thus exhibit long temporal memory. However, when we include synaptic failures (our second extension to the work of Bertschinger and Natschläger [2004]), the situation changes: with even a very low probability of failures – much smaller than the order of the inverse of the number of connections per neuron – networks are no longer able to carry out complex temporal processing. Our conclusion, then, is that randomly connected networks are not suitable for processing time-varying input. In the remainder of the thesis we make these ideas more precise.

1.4 Chapter overviews

Figure 1-3 contains a schematic of the outline of the thesis; below we provide a chapter-by-chapter summary.

Chapter 2 This chapter is the core of the thesis. We begin with the reduced model proposed by Bertschinger and Natschläger [2004], and develop a mean-field theory for the model in the limit of large, high connectivity networks. Specifically, we assume that the average number of connections per neuron, K, is proportional to number of neurons, N, so that in the limit $N \to \infty$ and $K \to \infty$, K/N is constant. Our goal is to determine whether such networks can operate at the edge of chaos. If they can, then, according to Jaeger [2001], Maass et al. [2002], Jaeger and Haas [2004], and Bertschinger and Natschläger [2004], they can exhibit long temporal memory. Here, our notion of long temporal memory is the ability to distinguish two signals that are

different in the distant, but not recent, past.

We find, however, that randomly connected networks in the class of Bertschinger and Natschläger [2004], but with K/N fixed as $N \to \infty$, are always chaotic, and thus cannot operate at the edge of chaos. Therefore, they cannot exhibit long temporal memory. We also show that the temporal memory of network is boosted by only $\mathcal{O}(\log N)$ above the time constant of the constituent neurons, so increasing the size of the network does not make it much more powerful.

Chapter 3 In this chapter we ask: why is our network alway chaotic while Bertschinger and Natschläger [2004]'s is not? What we find is that spareness matters: if K is fixed as $N \to \infty$, networks *can* operate on the edge of chaos. However, if K is large – as it is for real networks – the amplitude of the input must be large for networks to operate on the edge of chaos. Thus, the networks are largely input-driven, and the recurrent connectivity does not play much role. Nevertheless, in this regime the temporal memory is boosted by $\mathcal{O}(N)$, implying that large networks can process complex temporal signals.

Chapter 4 In this chapter we show that sparse networks operating on the edge of chaos – the ones we analyzed in Chapter 3 – are not robust to noise. With only a small probability ($\mathcal{O}(\epsilon/K)$, where $\epsilon \ll 1$) of synaptic failures, the temporal memory is drastically reduced, and scaling goes from $\mathcal{O}(N)$ to $\mathcal{O}(\log N)$.

Chapter 5 Finally, we present our conclusions, discussion, and future work.

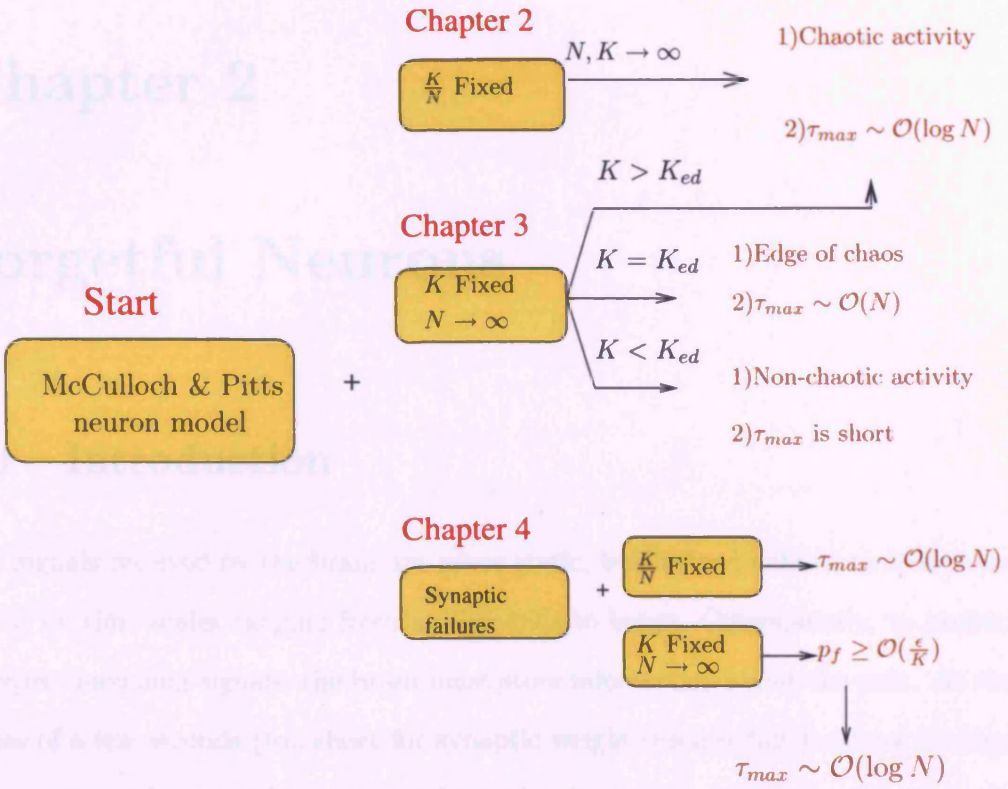


Figure 1-3: Summary of the main chapters. K and N indicate the average number of synaptic connections per neuron and number of neurons, respectively. τ_{max} is the temporal memory of the network, and p_f is the probability of synaptic failures. If $p_f = 0$ and $K = K_{ed}$, we have the condition of being at the edge of chaos, therefore the network exhibits long temporal memory in the order of $\mathcal{O}(N)$. Here, $\epsilon \ll 1$, indicates that an extremely small failure will change the scale of temporal memory,τ_{max}, from $\mathcal{O}(N)$ to $\mathcal{O}(\log N)$.

Chapter 2

Forgetful Neurons

2.1 Introduction

The signals received by the brain are never static, but instead exhibit temporal variations on time scales ranging from milliseconds to hours. Consequently, to properly interpret incoming signals, the brain must store information about the past. At time scales of a few seconds (too short for synaptic weight changes but too long for single neuron processing) it is likely that information is stored in patterns of activity. For example, you are about to leave home to go to your office and suddenly you hear London's forecast: "it will be rainy in London within the next two hours"or "it will be sunny in London within the next two hours" (Fig.2-1). These two sentences drive two different behaviours (taking an umbrella versus not taking one), and so, ultimately, drive very different patterns of activity in your brain.

The question that we want to address in this chapter is: how can circuits in the brain distinguish different time-varying signals, such as the two sentences concerning the weather in London? We are particularly interested in signals that differ in the

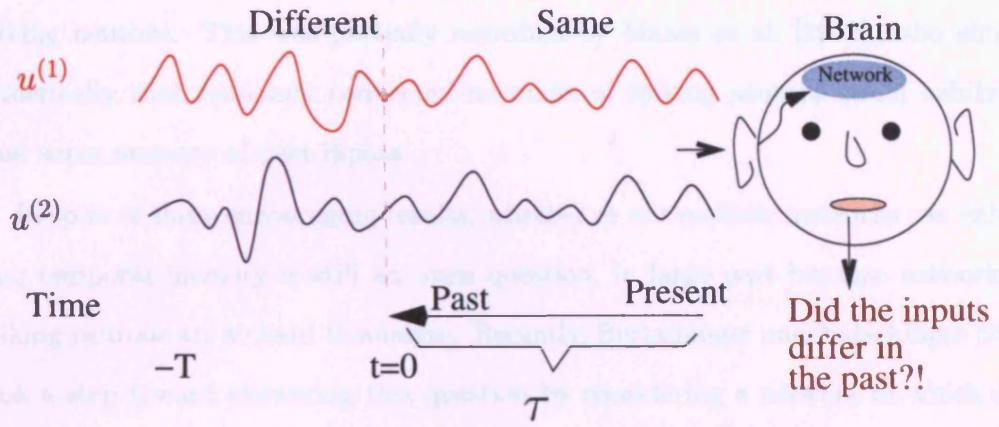


Figure 2-1: Schematic of signal discrimination. A person is trying to distinguish two signals at time τ. The signals differ only at times $t < 0$, so for large τ the task is difficult.

distant, but not recent, past. A network that performs this kind of processing must, therefore, have a long temporal memory. In other words, the current state of the network must depend on events that happened many seconds ago.

Building a network whose current state depends on input that occurred seconds to tens of seconds ago would seem difficult, since neurons are dominated by relatively short time constants – 10s to 100s of ms. However, a network time constant can be considerably longer than its single neuron time constants; all that is required is sufficient positive feedback to make the network almost self-sustaining. As with the well-studied neural integrator, such feedback can boost the single neuron time constants by large factors, one to two orders of magnitude (Seung [1996], Koulakov et al. [2002]).

Jaeger [2001] used just this method – positive feedback – to achieve long time constants in randomly connected networks. His analysis, however, focused on analog neurons, and it was not immediately clear whether it would apply to networks of

spiking neurons. This was partially remedied by Maass et al. [2002], who showed numerically that randomly connected networks of spiking neurons could exhibit at least some memory of past inputs.

In spite of these encouraging results, whether or not realistic networks can exhibit long temporal memory is still an open question, in large part because networks of spiking neurons are so hard to analyse. Recently, Bertschinger and Natschläger [2004] took a step toward answering this question by considering a network in which such analysis was possible. What they found was that networks were able to exhibit long memories if they operated on the edge of chaos; i.e., with a Lyapunov exponent near 0.

While their analysis provided a great deal of insight into the mechanism by which networks process time-varying input, they considered networks with unrealistically low connectivity (each neuron made only a few connections). Here we consider the more realistic case of high connectivity. When we do that, we find a number of differences, the most important being that networks, at least networks in the class they considered, are guaranteed to be chaotic. This finding suggests that randomly connected networks may not be able to exhibit memory much longer than the time constants of their constituent neurons.

2.2 Signal detection: memory in activity patterns

In this section we ask: what properties must a network have if it is to distinguish time-varying inputs that are very close in the recent past but not in the distant past? To answer this we take a state-space approach: we assume that the activity of each neuron is described by one variable. Therefore, for N neurons, the network activity

can be depicted in N-dimensional space. The activity pattern of the network at any time corresponds to a single point in this space, and the evolution of activity patterns over time is represented by trajectories, as illustrated in Fig.2-2.

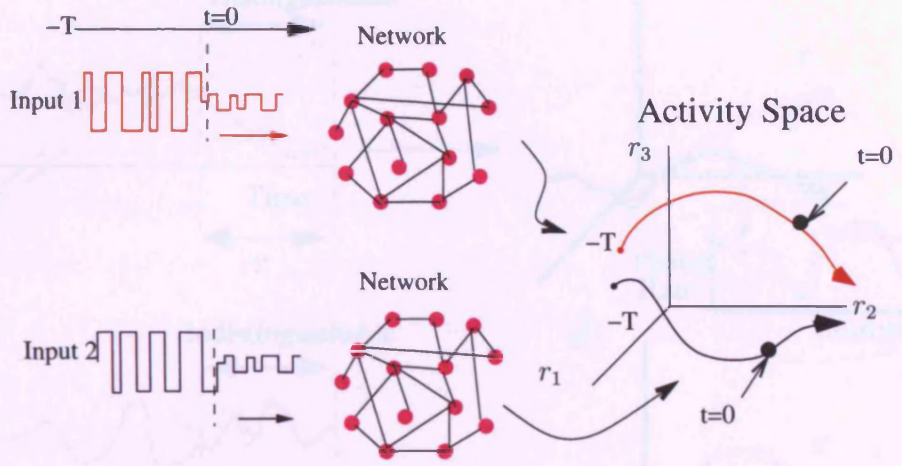


Figure 2-2: The neural activity in the presence of two inputs on two trials. The purple filled circles indicate neurons. In trial 1, the network is initialised in some state at time $-T$, and receives input 1. In trial 2, the same network is initialised to a different state, also at time $-T$, and receives input 2. The two inputs differ at $t < 0$ but are the same at $t > 0$. The activity of N neurons in state space (shown here in three dimensions) is depicted for the two trials by red and blue curves, respectively.

The trajectories depend on input to the network, and if different inputs push trajectories to two very different places (different patters of activity), then the network can distinguish them easily (Fig.2-3a, b). If the trajectories end up close together, then the network has a hard time (Fig.2-3c, d).

To distinguish many different signals, a network must be able to distinguish inputs that are fairly close. A natural measure of close is the mean square error between two signals. However, as shown in Fig.2-3, this is a bad measure for time-varying signals: the mean square error in the inputs can be small even though the network can easily distinguish them (Fig.2-3a, b), and the mean square error can be large when the

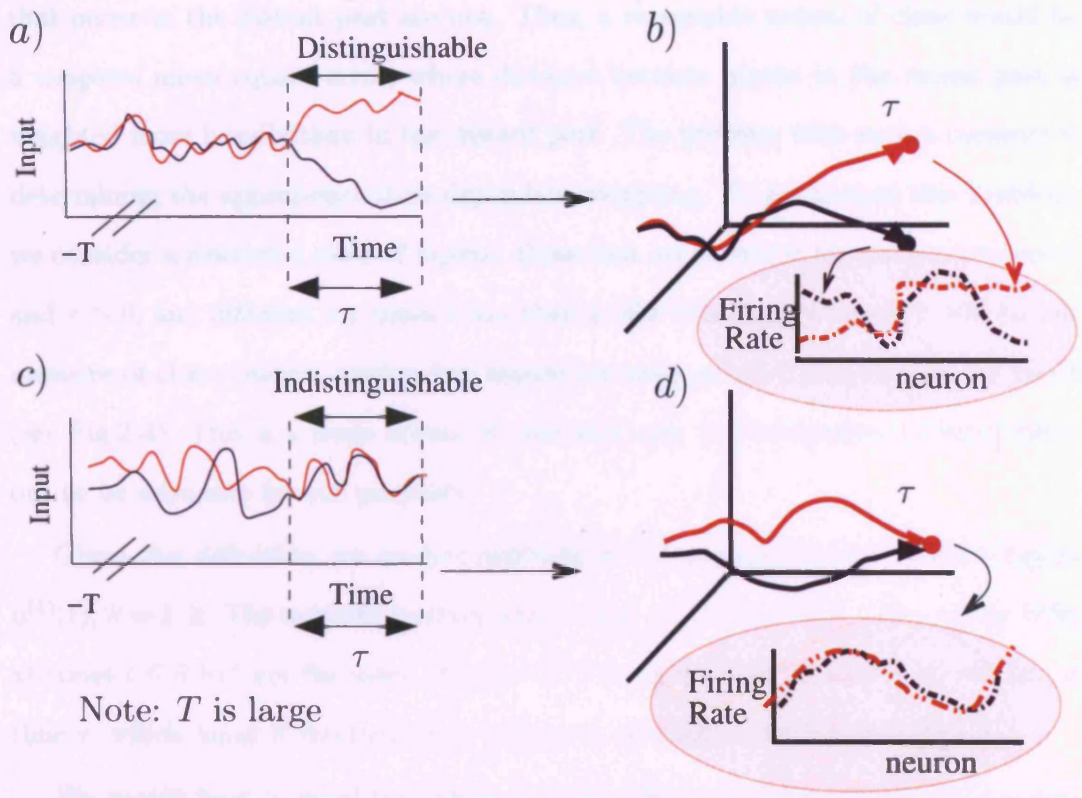


Figure 2-3: Mean square error does not correlate with the ability to distinguish inputs. **a**. The inputs are close in the distant past but different in the recent past, and have small mean square error. **b**. Network trajectories under the input shown in panel a. At time τ the trajectories are well separated, so these two inputs are easy for the network to distinguish. **c**. The inputs are different in the distant past but close in the recent past. Since the past is long (indicated by the two parallel lines crossing the horizontal axis), the mean square error between the inputs is large. **d**. Network trajectories under the input shown in panel b. At time τ the networks are *not* well separated, so these two inputs are hard for the network to distinguish.

inputs are indistinguishable (Fig.2-3c, d). The problem, of course, is that differences in input that occur in the recent past are easy to distinguish, whereas differences that occur in the distant past are not. Thus, a reasonable notion of close would be a *weighted* mean square error, where distance between inputs in the recent past is weighted more heavily than in the distant past. The problem with such a measure is determining the appropriate time-dependent weighting. To get around this problem, we consider a restricted class of inputs: those that are identical for times t between 0 and $\tau > 0$, and different for times t less than 0. For this class of input, τ will be our measure of close: large τ implies two inputs are close; small τ implies they far apart (see Fig.2-4). This is a crude notion of close and later we could refine it, but it turns out to be adequate for our purposes.

Given this definition, we analyse networks in the presence of two different inputs $u^{(k)}(t)$, $k = 1, 2$. The network receives either input $u^{(1)}(t)$ or $u^{(2)}(t)$. The inputs differ at times $t \leq 0$ but are the same after $t = 0$. The task of the network is to tell us, at time τ, which input it received. As τ increases the task become more difficult.

We should bear in mind that observing two different patters of activity in a network does not necessarily means that it received two different inputs. For example, if the network started with two different initial conditions, it would exhibit two different patterns of activity, at least for a while. Therefore, we need to consider the effects of different initial conditions on the activity patterns. To do that, instead of studying just two trajectories with two given initial conditions, we study two groups of trajectories with different initial conditions. Each group corresponds to one input.

For the rest of our study, then, we will assume that initial conditions form some ball in activity space. The final position of ball depends on which input, $u^{(1)}(t)$ or $u^{(2)}(t)$, the network received. Thus, the question of how to build networks that are

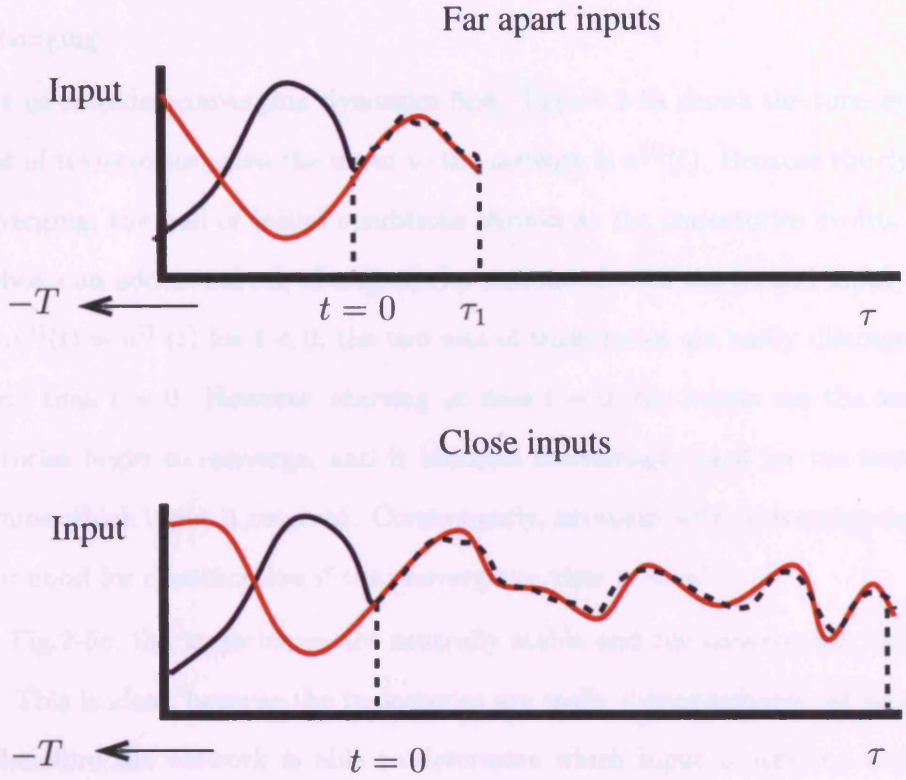


Figure 2-4: Measure of "close". In the upper panel, τ_1 is small, which makes it easy for the network to distinguish the two inputs (red and blue lines). Thus, the inputs are effectively far apart. In the lower panel, τ is large, which makes it hard for the network to distinguish the two inputs. Thus, they are effectively close. In general, larger τ implies closer inputs.

good at classifying inputs reduces to the question: for what kind of networks do the two balls (each driven by a different input) end up far apart in activity space? Although we cannot answer this question in all generality, we can study it in the three main regimes that dynamical systems tend to fall into: converging, neutrally stable, and diverging.

Let us consider converging dynamics first. Figure 2-5a shows the time evolution of a set of trajectories when the input to the network is $u^{(1)}(t)$. Because the dynamics is converging, the ball of initial conditions shrinks as the trajectories evolve. Figure 2-5b shows an additional set of trajectories associated with the second input, $u^{(2)}(t)$. Since $u^{(1)}(t) \neq u^{(2)}(t)$ for $t < 0$, the two sets of trajectories are easily distinguishable up until time $t = 0$. However, starting at time $t = 0$, the inputs are the same, the trajectories begin to converge, and it becomes increasingly hard for the network to determine which input it received. Consequently, networks with converging dynamics are not good for classification if the convergence time is small.

In Fig.2-5c, the trajectories are neutrally stable and the convergence time is infinity. This is ideal, because the trajectories are easily distinguishable for long times, and therefore the network is able to determine which input it received even when $\tau \gg 0$. Networks that operate in this regime are optimal for distinguishing signals.

Beyond neutrally stable is the diverging regime. This regime might seem to be good also, since trajectories that are farther apart are easier to distinguish. However, because the state-space is bounded, the trajectories cannot diverge forever, and eventually they start mixing. This is illustrated as intertwined regions in Fig.2-5d. Here we call the two balls *blobs*. After $t = 0$, the blobs receive the same input, and they become closer and closer. Eventually they converge to one blob, known as a chaotic attractor [Glass, 1995, Glass and Mackey, 1988, Grebogi et al., 1987, Eckmann and

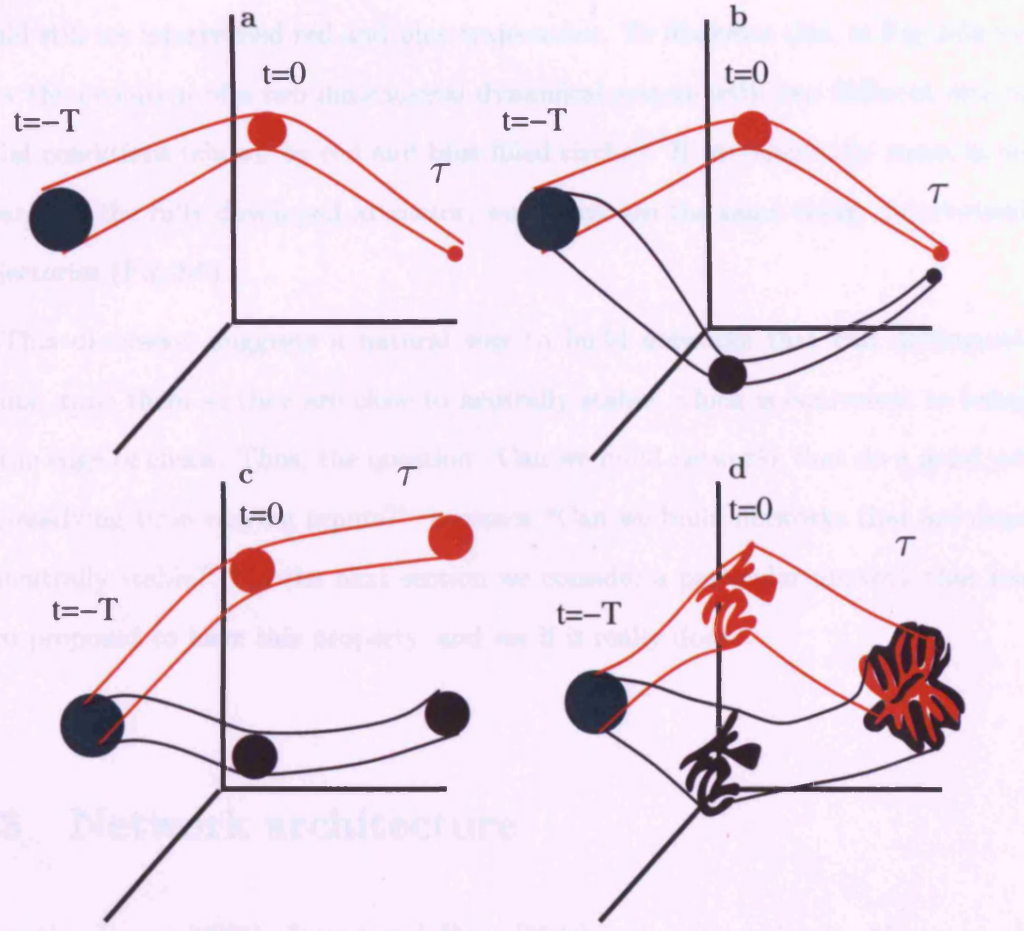


Figure 2-5: Schematic of activity versus time. **a,b**. Converging trajectories. **c**. Neutrally stable trajectories. **d**. Diverging trajectories. The cyan ball at $t = -T$, indicates the initial conditions. The red ball at $t = 0$ represents the endpoints of trajectories that evolved under the input $u^{(1)}(t)$ between $-T$ and 0. The blue ball at $t = 0$ corresponds to the analogous case, except the input is $u^{(2)}$. The balls at $t = \tau$ are the images of the balls at $t = 0$ under the network dynamics. Note that diverging trajectories (panel d) imply chaos, for which there is strong mixing in activity space.

Ruelle, 1985]. In this blob the red and blue trajectories are intertwined, making it very hard to distinguish them. Importantly, even if we were to zoom in on a blob, we would still see intertwined red and blue trajectories. To illustrate this, in Fig.2-6a we show the evolution of a two dimensional dynamical system with two different sets of initial conditions (shown by red and blue filled circles). If we repeatedly zoom in on a patch of the fully developed attractor, we always see the same thing: intertwined trajectories (Fig.2-6).

This discussion suggests a natural way to build networks that can distinguish inputs: tune them so they are close to neutrally stable, which is equivalent to being on the edge of chaos. Thus, the question "Can we build networks that do a good job at classifying time-varying inputs?" becomes "Can we build networks that are close to neutrally stable?". In the next section we consider a particular network that has been proposed to have this property, and see if it really does.

2.3 Network architecture

Recently, Jaeger [2001], Jaeger and Haas [2004] and, independently, Maass et al. [2002], proposed that randomly connected networks could exhibit the long temporal memory necessary for complex temporal processing. To investigate this proposal analytically, Bertschinger and Natschläger [2004] considered the following highly reduced network model: Each neuron of N neurons in a network is connected to exactly K randomly chosen neurons. The activity of neuron i at time t is represented by two states, $x_i(t) = \{1, -1\}$; that is, firing/non-firing, respectively. The neurons have the following parallel dynamics update rule:

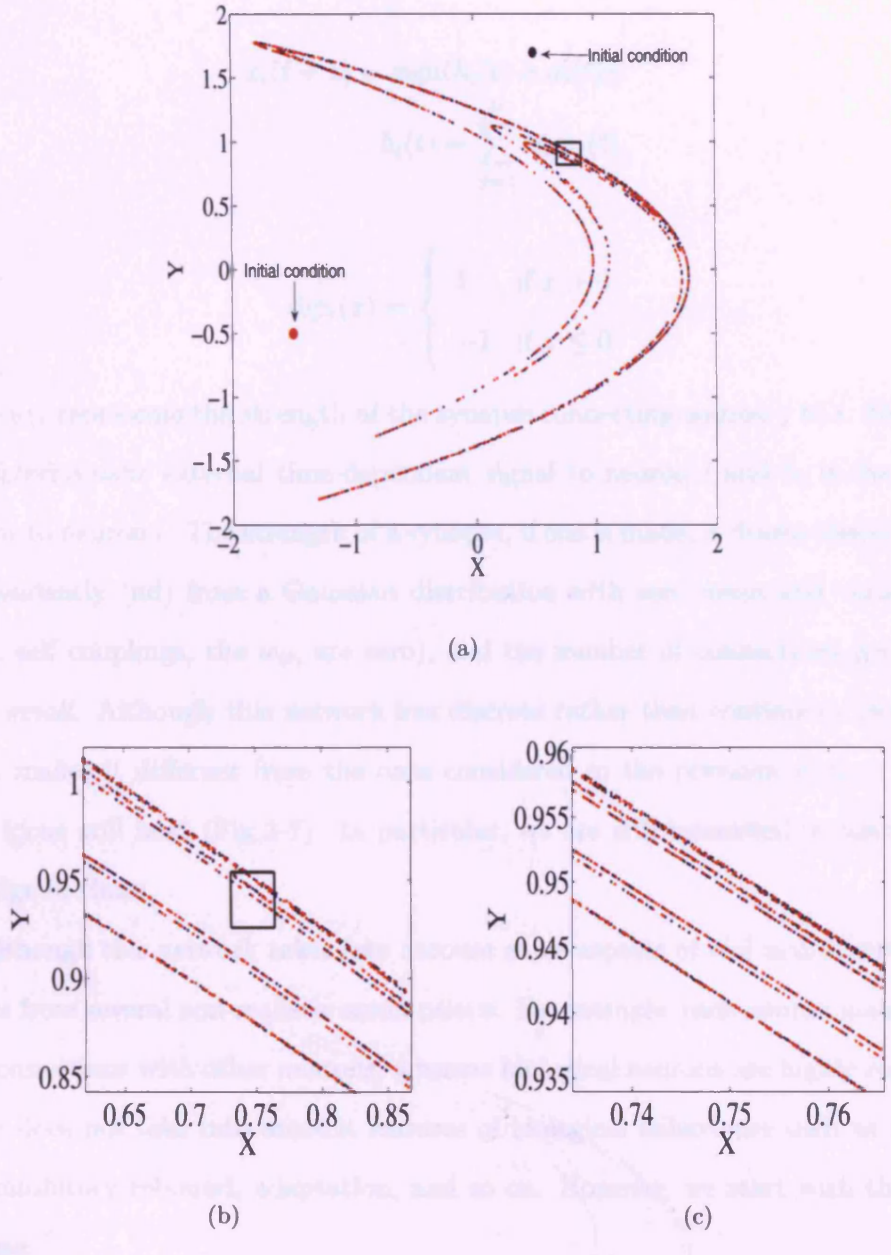


Figure 2-6: Schematic of a chaotic attractor (Henón attractor [Grebogi et al., 1987]). The evolution equation is $X(t+1) = a - X^2(t) + bX(t)$ and $Y(t+1) = X(t)$, where $a = 1.4$ and $b = 0.3$. **a**. The full attractor: two balls of initial conditions (red and blue circles) expand (after 30,000 iterations) into two sets of distinct blue and red dots. **b**. Enlarged view of the patch shown in panel a. **c**. Enlarged view of the patch shown in panel b.

$$x_i(t+1) = \text{sign}(h_i(t) + u_i(t)), \tag{2.1}$$

$$h_i(t) = \sum_{j=1}^{N} w_{ij} x_j(t), \tag{2.2}$$

$$\text{sign}(x) = \begin{cases} 1 & \text{if } x > 0 \\ -1 & \text{if } x \leq 0 \end{cases},$$

where w_{ij} represents the strength of the synapse connecting neuron j to i. Here, $u_i(t)$ is a *deterministic* external time-dependent signal to neuron i and h_i is the internal current to neuron i. The strength of a synapse, if one is made, is drawn identically and independently (iid) from a Gaussian distribution with zero mean and variance σ^2_{BN} (note, self couplings, the w_{ii}, are zero), and the number of connections per neuron, K, is *small*. Although this network has discrete rather than continuous dynamics – which makes it different from the ones considered in the previous section – all the main ideas still hold (Fig.2-7). In particular, we are still interested in operation on the edge of chaos.

Although this network takes into account some aspects of real neural networks, it suffers from several non-realistic assumptions. For example, each neuron makes only a few connections with other neurons, whereas biological neurons are highly connected, and it does not take into account features of biological behaviours such as bursting, post-inhibitory rebound, adaptation, and so on. However, we start with this model because:

- Bertschinger and Natschläger showed that such networks can operate on the edge of chaos (Fig.2-5c); that is, they can distinguish inputs that differ in the

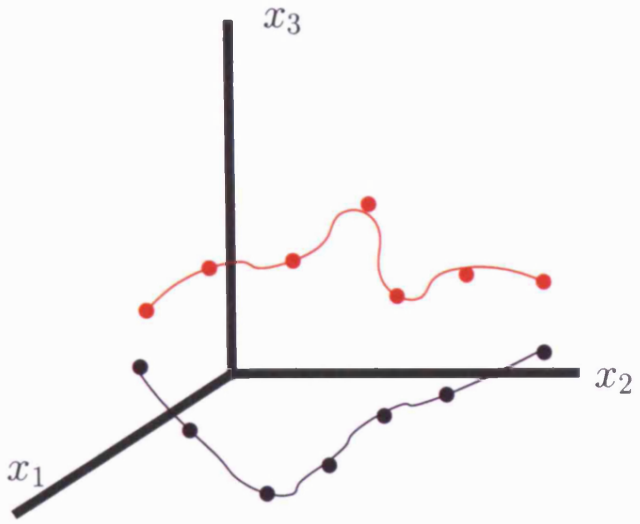


Figure 2-7: Time evolution of a discrete system. Red and blue dots corresponds to patterns of activity at discrete times ($t = 1, 2, ...$). The picture indicates that our intuition about continuous trajectories holds for discrete trajectories.

distant past, which implies a long temporal memory.

- They are reasonably analogous to realistic neurons, in the sense that they sum inputs and "fire" if the sum is large enough.

Thus, even though some biological details were discarded, these networks can still give us insights into how to build realistic networks that exhibits long memory.

In this chapter, we study networks with the same underlying architecture of those proposed by Bertschinger and Natschläger [2004], but in the more biologically realistic regime of high connectivity. Specifically, we assume that neurons do not have a fixed number of connections, and, more importantly, we let the mean number of connections per neuron, K, scale with N. In other words, we fix K/N as we vary N. This is because we want to know how temporal memory scales by increasing N, while neurons are highly connected (K/N fixed). In chapter 3, we will study temporal memory in

sparse connected networks. To simplify calculations, we will consider the large N limit.

To allow the number of connections per neuron to vary, we let the connection matrix, w_{ij}, have the form

$$w_{ij} = \begin{cases} w_{ij}^c & \text{prob} = \frac{K}{N} \\ 0 & \text{prob} = 1 - \frac{K}{N} \end{cases}, \tag{2.3}$$

where the w_{ij}^c's are drawn iid from a distribution (not necessarily Gaussian) with zero mean and variance σ^2/K. Thus, the variance of the weights is

$$\text{Var}[w_{ij}] = \frac{K}{N}\text{Var}[w_{ij}^c] = \frac{\sigma^2}{N}.$$

This network architecture, although still simpler than that of biological networks, is closer to realistic than that of Bertschinger and Natschläger. Our goal in the next section is to determine whether these networks can operate at the neutrally stable regime; that is on the edge of chaos. In the next chapter, we return to a network with low connectivity, which will allow a direct comparison with Bertschinger and Natschläger's network.

2.4 Formal analysis

In this section we provide a formal analysis of large, highly connected networks. Since we want to know how nearby trajectories behave, we start by defining a measure for the distance between trajectories, the normalised Hamming distance [Hamming, 1950],

$$d(t) \equiv \frac{1}{N}\sum_{i=1}^{N}\begin{cases} 0 & \text{if } x_i^1(t) = x_i^2(t) \\ 1 & \text{if } x_i^1(t) \neq x_i^2(t) \end{cases}, \tag{2.4}$$

where $x_i^1(t)$ and $x_i^2(t)$ represent the state of neuron i at time t for two trajectories. For simplicity, from now on we call $d(t)$ the Hamming distance. The maximum value of this distance is 1, when all the states are different, and its minimum is zero, when all the states are the same.

First let us study the distance between two trajectories, $x_i^1(t)$ and $x_i^2(t)$, that have different initial conditions but receive the same input, u. We want to derive a dynamical equation for the Hamming distance; that is, we want to derive an equation for $d(t+1)$ in terms of $d(t)$. To do that, we use Eqs. 2.1 and 2.4, which gives us

$$d(t+1) = \frac{1}{N}\sum_{i=1}^{N}\begin{cases} 0 & \text{if } (h_i^1(t) + u_i(t)) \text{ and } (h_i^2(t) + u_i(t)) \text{ have the same sign} \\ 1 & \text{if } (h_i^1(t) + u_i(t)) \text{and } (h_i^2(t) + u_i(t)) \text{ have opposite signs} \end{cases},$$

where $h_i^k(t) = \sum_i w_{ij} x_j^k(t)$, and $k = 1, 2$ indicates the two trajectories. Alternatively, we can write

$$\begin{aligned} d(t+1) &= \text{prob}\,(h_i^1(t) + u_i(t) \geq 0 \;\&h_i^2(t) + u_i(t) < 0) \\ &+ \text{prob}\,(h_i^1(t) + u_i(t) < 0 \;\&h_i^2(t) + u_i(t) \geq 0) \end{aligned} \tag{2.5}$$

where the probability is over index, i. Since the neurons can take binary values, ± 1, the current, h_i, can be written $h_i = \sum_{\{x_j=1\}} w_{ij}^c - \sum_{\{x_j=-1\}} w_{ij}^c$. Because the w_{ij}^c's are uncorrelated random variables, the central limit theorem tells us that h_i is a Gaussian random variable with zero mean and variance equal to σ^2 ($= K\text{Var}[w_{ij}^c]$; see Eq.2.3); that is, $h_i \sim \mathcal{N}(0, \sigma^2)$. Note that the central limit theorem is valid if we assume that

x_i's are independent than w_{ij}'s. Since, we don't know how to prove this assumption analytically, we have validated it by numerical simulation shown in Fig.2-8.

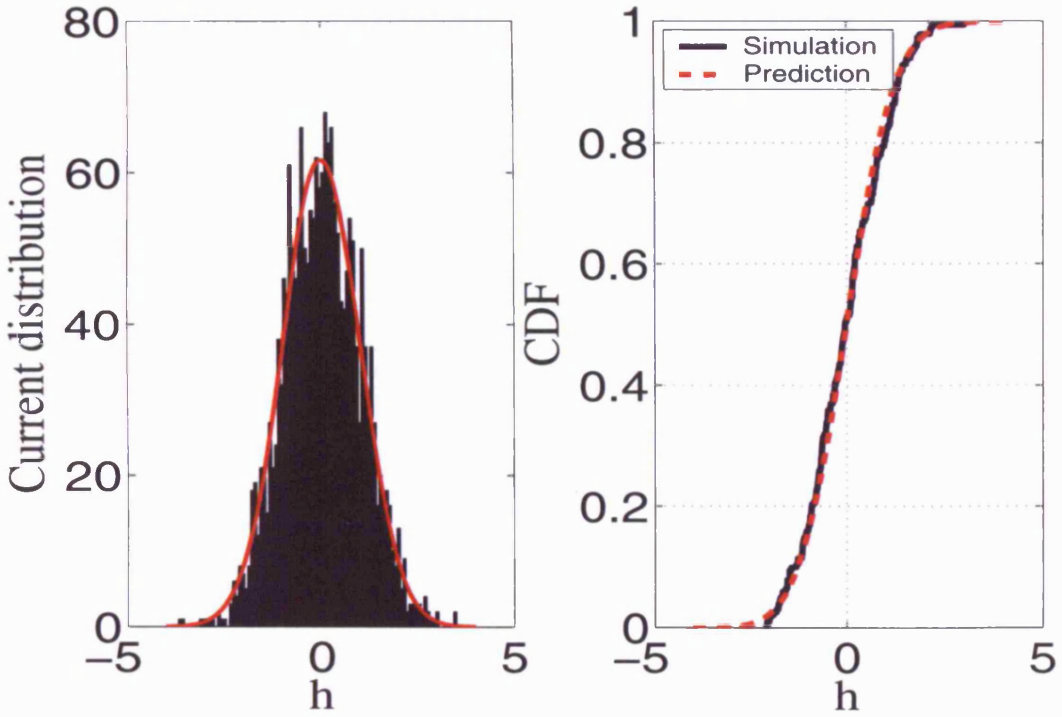


Figure 2-8: The left panel illustrates the distribution of h for 2000 neurons after several time steps. Here, w_{ij}'s, are drawn iid from a probability distribution with zero mean and variance $1/2000$ ($\sigma = 1$). We hypothesis that the depicted histogram has a Gaussian distribution with zero mean and variance 1 (red curve). In the right panel we have plotted the cumulative distribution function (CDF) of the normalized distributions shown in the left panel (prediction and simulation). According to K-S test our hypothesis is valid (P value > 0.1).

To compute the right hand side of Eq.2.5, we need to know the joint distribution of h^1 and h^2 given input, $u(t)$; we denote this distribution $P(h^1, h^2)$. Again, because of the central limit theorem, this distribution is Gaussian. Since w_{ij} is a zero mean random variable, h_i^1 and h_i^2 are also zero mean random variables. Thus

$$P(h^1, h^2) \equiv \frac{1}{\sqrt{\det(2\pi\Sigma)}} e^{-\frac{1}{2}\mathbf{H}\Sigma^{-1}\mathbf{H}^\top}, \tag{2.6}$$

where $\mathbf{H} = (h^1, h^2)$, and Σ, the covariance matrix, is given by

$$\Sigma_{kl} = \frac{1}{N}\sum_{i=1}^{N} h_i^k h_i^l = \sum_{j,j'=1}^{N} x_j^k x_{j'}^l \left(\frac{1}{N}\sum_{i=1}^{N} w_{ij} w_{ij'}\right). \tag{2.7}$$

Because the w's are drawn iid from a distribution with zero mean and variance σ^2/N, in the large N limit, the term $\frac{1}{N}\sum_i w_{ij} w_{ij'}$ is equal to $(\sigma^2/N)\,\delta_{jj'}$ where $\delta_{jj'}$ is the Kronecker delta. We thus have

$$\Sigma_{kl} = \frac{\sigma^2}{N}\sum_{i=1}^{N} x_i^k x_i^l, \tag{2.8}$$

where $k, l = 1, 2$. For clarity, we have dropped the explicit dependence on time, but we should keep in mind that x_i and Σ_{kl} both depend on time.

It is convenient to define a new parameter, $q(t) \equiv 1 - 2d(t)$. Using Eq.2.4, we have

$$q(t) = \frac{1}{N}\sum_{i=1}^{N} x_i^1(t) x_i^2(t). \tag{2.9}$$

The parameter $q(t)$ represents the overlap (dot product) between the states of network for the two different trajectories at time t. In term of q, the covariance matrix has an especially simple form,

$$\Sigma = \sigma^2 \begin{pmatrix} 1 & q \\ q & 1 \end{pmatrix}. \tag{2.10}$$

This is consistent with $\mathrm{Var}[h] = \sigma^2$, which we derived above. Note that all our calculations are strictly valid in the limit $N \to \infty$.

For simplicity, we make the following assumptions about the time varying input, $u_i(t)$:

- $u_i(t)$ is temporally uncorrelated $\forall\ i$.
- Each $u_i(t)$ is drawn iid from a stationary distribution, $P(u)$.

We use temporally uncorrelated inputs so that the present input cannot tell us anything about past inputs. This forces the network to do all the work; basically, it isolates temporal correlations in the network from temporal correlations in the inputs. To see the effect of temporally correlated inputs refer to Okada [1995], Henkel and Opper [1990].

Letting $d(t)$ and $d(t+1)$ be related via

$$d(t+1) = f(d(t)), \tag{2.11}$$

we see from Eq.2.5 that

$$f(d) = \int du P(u) f(d;u,u), \tag{2.12}$$

where

$$f(d;u,u) = \int_{-\infty}^{\infty} dh^1 dh^2 P(h^1,h^2)\Theta(-(h^1+u)(h^2+u)) \tag{2.13}$$

and $\Theta(x)$ is the Heaviside step function: $\Theta(x) = 1$ when $x \geq 0$, and 0 otherwise.

We can write Eq.2.13 in the form

$$f(d;u,u) = \int_{-u}^{\infty} dh^1 \int_{-\infty}^{-u} dh^2 P(h^1,h^2) + \int_{-\infty}^{-u} dh^1 \int_{-u}^{\infty} dh^2 P(h^1,h^2), \tag{2.14}$$

which has the geometrical interpretation illustrated in Fig.2-9. In addition, Eq.2.14, can be written in the form (see Appendix A)

$$f(d;u,u) = \int_{-\infty}^{\infty} \frac{db}{\sigma\sqrt{2\pi d}} e^{-\frac{b^2}{2d\sigma^2}} \int_{-|b|-u}^{|b|-u} \frac{da}{\sigma\sqrt{2\pi(1-d)}} e^{-\frac{a^2}{2(1-d)\sigma^2}}, \tag{2.15}$$

which is very convenient for calculation. And, finally, in Appendix A, we derive the following reduced form for $f(d;u,u)$,

$$f(d;u,u) = \frac{1}{2\sqrt{\pi}} \int_{-\infty}^{\infty} dz e^{-z^2} \left(\text{erf}\left(|z|\sqrt{\frac{d}{1-d}} - \hat{u} \right) + \text{erf}\left(|z|\sqrt{\frac{d}{1-d}} + \hat{u} \right) \right), \tag{2.16}$$

where $\hat{u} \equiv \frac{u}{\sqrt{2(1-d)}\sigma}$ and

$$\text{erf}(z) \equiv \frac{2}{\sqrt{\pi}} \int_0^z dt \exp(-t^2)$$

is the standard error function.

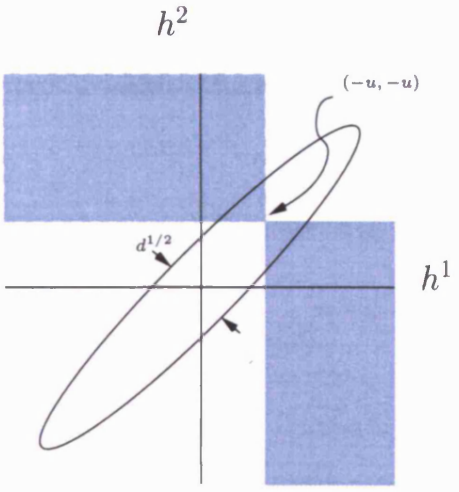


Figure 2-9: The function that determines how $d(t)$ evolves in time, $f(d;u,u)$, is a Gaussian integral over the blue quadrants. The ellipse tilted at 45 degree represents the outline of a Gaussian distribution with covariance matrix given in Eq.2.10.

It turns out that when u is drawn from a Gaussian distribution with zero mean and variance σ_u^2, Eq.2.12 reduces to

$$d(t+1) = \frac{2}{\pi} \arcsin \sqrt{Ad(t)}, \tag{2.17}$$

where $A = \frac{\sigma^2}{\sigma^2+\sigma_u^2}$ (Amari [1974], also see Amari and Maginu [1988]). This expression is derived in Appendix B.

The function $f(d)$ contains all the information about the evolution of $d(t)$. Fig.2-10 shows the behaviour of $f(d)$ versus d when there is no external input ($P(u) = \delta(u)$, which means $A = 1$ in Eq.2.17). The intersections of this function with the line at 45° determine the fixed points, denoted d^*. These points are thus solutions to the equation $f(d^*) = d^*$. Straightforward linear analysis of Eq.2.11 (see Eq.2.24) tells us that $f'(d^*) = \frac{df(d)}{dd}|_{d=d^*}$ determines the stability of the fixed points at $d = d^*$. When $f'(d^*) < 1$, the fixed point is stable, so trajectories converge. However, when $f'(d^*) > 1$, the fixed point is unstable, and trajectories diverge. Figure2-10 contains three fixed points: $d^* = 0$ and 1, which are unstable and $d^* = 1/2$, which is stable*.

Fortunately, we can get a very simple expression for the derivative of $f(d; u, u)$ with respect to d (see Appendix A.2), which is

$$\frac{\partial f(d;u,u)}{\partial d} = \frac{1}{\pi\sqrt{d(1-d)}} e^{-\frac{u^2}{2(1-d)\sigma^2}}. \tag{2.18}$$

In the limit $d \to 0$, $f'(d) \to \infty$. This means that nearby trajectories diverge, and they diverge *very* fast. Just how fast will be shown below.

To understand intuitively this rapid divergence, let us examine what happens when

*When u can take on non-zero values there are only two fixed points, since in this case $d = 1$ is not an equilibrium.

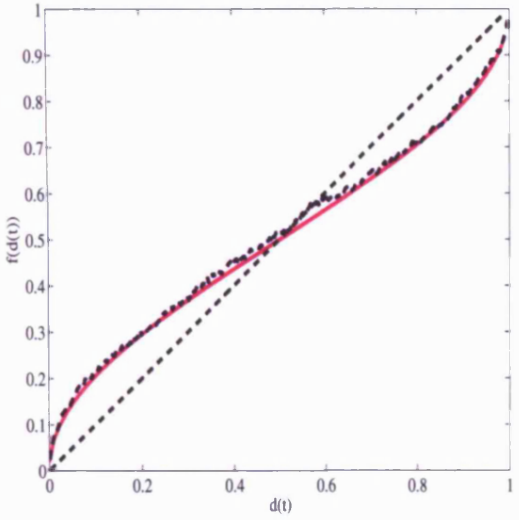


Figure 2-10: Comparison of mean field equation and simulations for $P(u) = \delta(u)$. The magenta curve corresponds to Eq.2.17, the blue dotted line is the result of simulations, and the black dashed line lies at 45°. The intersections of the magenta curve with the line at at 45° are the fixed points of Eq.2.11.

two states differ by only one neuron, which we choose to be neuron j. Specifically, $x_j^1(t) \neq x_j^2(t)$ and $x_{j'}^1(t) = x_{j'}^2(t)$ whenever $j' \neq j$. Under these conditions, $h_i^1(t)$ and $h_i^2(t)$ differ by $\pm 2w_{ij}$. Since w_{ij} is zero with probability $1 - K/N$, for about $N - K$ of the neurons, $h_i^1(t) = h_i^2(t)$. For the other $\sim K$ neurons, however, $h_i^1(t)$ and $h_i^2(t)$ differ by w_{ij}^c. Since $w^c \sim \mathcal{O}(\frac{\sigma}{\sqrt{K}})$, for these $\sim K$ neurons, $h_j^1 = h_j^2 \pm \mathcal{O}(\frac{\sigma}{\sqrt{K}})$. The probability that $h_i^1(t) > u_i(t)$ and $h_i^2(t) < u_i(t)$ is, then, proportional to $\sigma K^{-1/2}$ divided by the total range of the current, h, which is about σ (see Eqs.2.6 and 2.10, and Fig.2-11). Thus, the number of neurons that are different on the next time step is $K\mathcal{O}(K^{-1/2}) \sim N\mathcal{O}(N^{-1/2})$ (where the second "$\sim$" follows because $K/N \sim \mathcal{O}(1)$). Consequently, if $d(0) = 1/N$ (one neuron difference), then on the next time step, $d(1) \sim N^{1/2}/N = N^{-1/2} \sim d(0)^{1/2}$. A single missed or extra-spike on one time step thus causes, on the next time step, missed or extra spikes on $\mathcal{O}(N^{1/2})$ neurons. In

other words, if $d(0) = 1/N$ then $d(1)$ will be $\mathcal{O}(1/N^{1/2}) \sim \mathcal{O}(d(0)^{1/2})$.

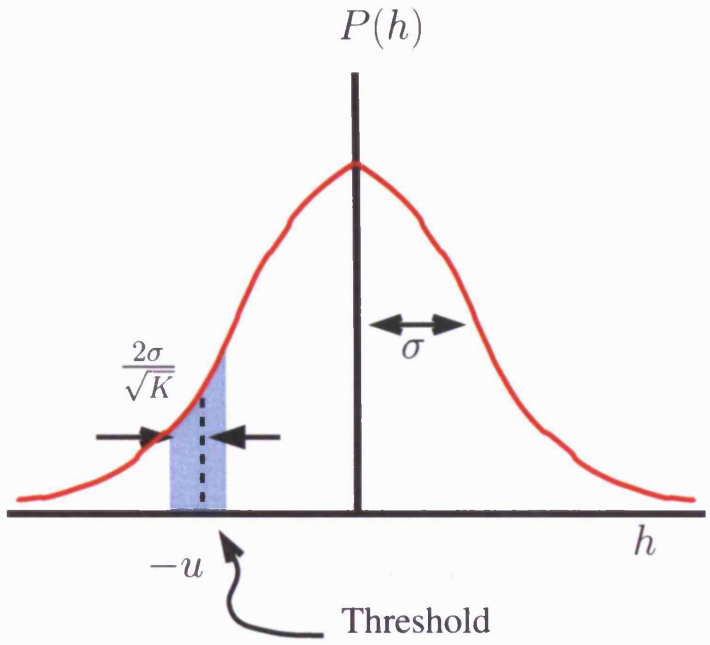


Figure 2-11: The probability distribution of the internal current, h, for one neuron. Each neuron receives its input from K neurons.The blue region is where the current has the most influence on the postsynaptic neuron.

The fact that $f(d) \sim d^{1/2}$ for small d means small differences in initial condition are amplified. Moreover, the time it takes for small $d(0)$ to become $\mathcal{O}(1)$ is very short: $f(d) \sim d^{1/2}$ implies that $d(t) \sim d(0)^{\frac{1}{2^t}}$, so the time it takes for $d(t)$ to become $\mathcal{O}(1)$ is $\mathcal{O}\Big(\log(\log(1/d(0)))\Big)$. This is *extremely* fast divergence.

Ultimately, the rapid divergence of nearby trajectories is due to the fact that $\mathcal{O}(K^{1/2})$ neurons are always close to threshold (see Fig.2-11). Van Vreeswijk and Sompolinsky [1996] found the same phenomena in a more biologically plausible model. Thus, our very simple model retains some of the features of biologically realistic networks.

2.5 Main scenario: two input separation

In the previous section, we found that large and highly connected networks (in a particular simplified class) are chaotic, which precludes operation at the edge of chaos. We now investigate the consequences of this. We consider the scenario discussed in Sec.2.2, in which the network evolves with different inputs, $u^{(1)}(t)$ and $u^{(2)}(t)$, between $-T$ and 0, and then evolves under the same input, $u(t)$, between 0 and τ. The question we ask is: how large is τ before the state of the network can not tell us which input it received?

We use a time, T, that is large compared to transients, so, from Fig.2-5d, we see that the blobs enter two different chaotic attractors at $t = 0$. Because the inputs are different for times $t < 0$, the attractors are fairly distinguishable at $t = 0$. After time 0 the inputs become the same, the distance between the chaotic attractors (blobs) becomes smaller and smaller, and eventually the two attractors merge. The time it takes for this to happen, denoted τ_{max}, tells us how long the network can distinguish the difference between inputs.

To compute this characteristic time, which, as above, we call temporal memory, we proceed as in Sec.2.4. The only difference is that we consider trajectories that receive different inputs rather than the same input. We denote trajectories for each input, $u^{(k)}$, as $x_i^{(k)}(t)$, where $k = 1, 2$. Note that we have extended our previous notation for labelling trajectories: $x^{(k)}$ refers to a trajectory that receives input k while x^k refers to a trajectory with initial condition k. The Hamming distance, which has a form very similar to Eq.2.4, is written as

$$\begin{aligned} d(t+1) &= \text{prob}\left(h_i^{(1)}(t) + u_i^{(1)}(t) \geq 0 \;\&\; h_i^{(2)}(t) + u_i^{(2)}(t) < 0\right) \\ &+ \text{prob}\left(h_i^{(1)}(t) + u_i^{(1)}(t) < 0 \;\&\; h_i^{(2)}(t) + u_i^{(2)}(t) \geq 0\right), \end{aligned} \tag{2.19}$$

which again can be written in the simple form

$$d(t+1) = f(d(t)), \tag{2.20}$$

where

$$f(d) = \int du^{(1)} du^{(2)} P(u^{(1)}, u^{(2)}) f(d; u^{(1)}, u^{(2)}), \tag{2.21}$$

and

$$\begin{aligned} f(d; u^{(1)}, u^{(2)}) &= \int_{-u^{(1)}}^{\infty} dh^{(1)} \int_{-\infty}^{-u^{(2)}} dh^{(2)} P(h^{(1)}, h^{(2)}) \\ &+ \int_{-\infty}^{-u^{(1)}} dh^{(1)} \int_{-u^{(2)}}^{\infty} dh^{(2)} P(h^{(1)}, h^{(2)}). \end{aligned} \tag{2.22}$$

Equation 2.22 reduces to Eq.2.14, when $u^{(1)} = u^{(2)} = u$. The inputs are again temporally uncorrelated and are drawn iid from the distribution $P(u^{(1)})P(u^{(2)})$. The geometrical interpretation of Eq.2.22 is illustrated in Fig.2-12, and the full expression for $f(d; u^{(1)}, u^{(2)})$ is derived in Appendix A.

Computing τ_{max}: To compute how long it takes for the network to forget about the inputs, we need to compute the time that it takes for the two attractors to converge to one (Fig.2-13). This is the time it takes the distance between the attractors to become approximately equal to the distance between points on each attractor, since at this time the state of the network will provide no information about which input, $u^{(1)}$ or $u^{(2)}$, it received. Thus, the question we address is: when does the Hamming distance between attractors, $d(t)$, reach the Hamming distance between points on the

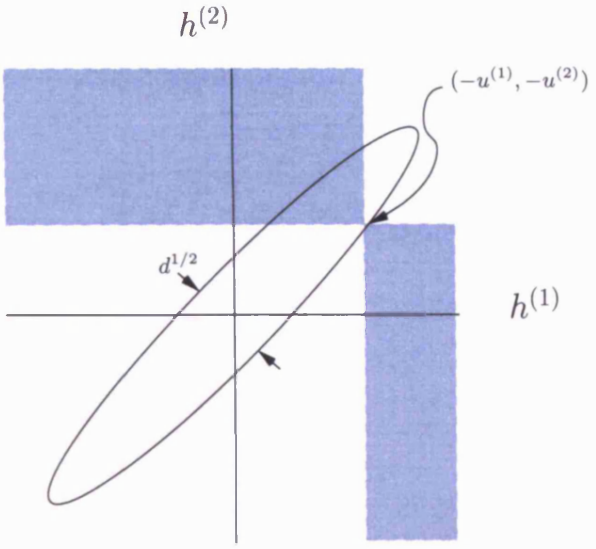


Figure 2-12: The function that determines how $d(t)$ evolves in time, $f(d; u^{(1)}, u^{(2)})$, is Gaussian integral over the blue quadrants. The ellipse tilted at 45 degree represents the outline of a Gaussian with covariance matrix given in Eq.2.10.

attractor, d^*?

In the mean field limit, $N \to \infty$, the answer is never: $d(t)$ will get exponentially close to d^*, but never reach it. This is illustrated in Fig. 2-14. Therefore, in this limit, the network always can distinguish inputs. For finite N, however, there are fluctuations around the equilibrium that are of order $1/\sqrt{N}$. In this case, when the difference between $d(t)$ and d^* becomes in the order of the fluctuations, $\mathcal{O}(1/\sqrt{N})$, the network can no longer tell the difference between trajectories. The time it takes for this to happen, denoted τ_{max}, is the temporal memory of the network. Our aim here is to understand how τ_{max} scales with N.

We can derive an explicit expression for the time evolution of the Hamming distance near equilibrium by linearising Eq. 2.20 around $d = d^*$. This gives us, to first

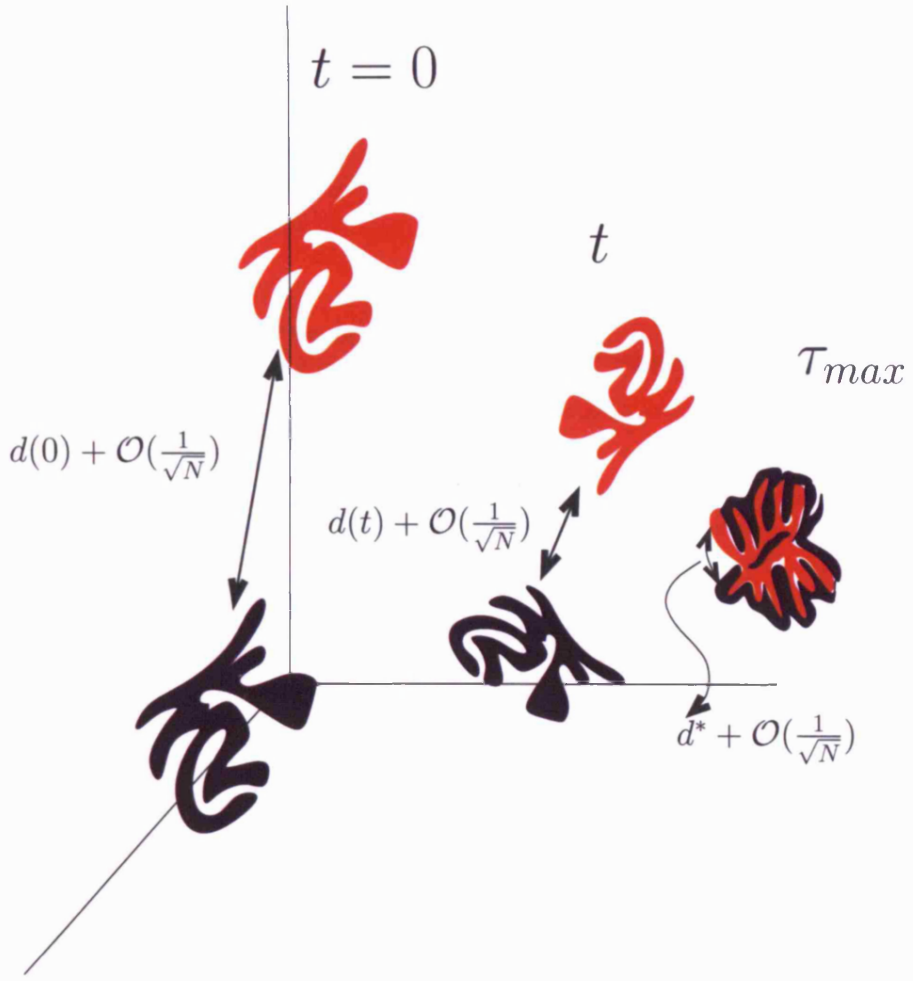


Figure 2-13: The distance between attractors for the large but finite N. The fluctuations, which are $\mathcal{O}(\frac{1}{\sqrt{N}})$, are added to the mean-field distance, $d(t)$, which was derived in the limit $N \to \infty$.

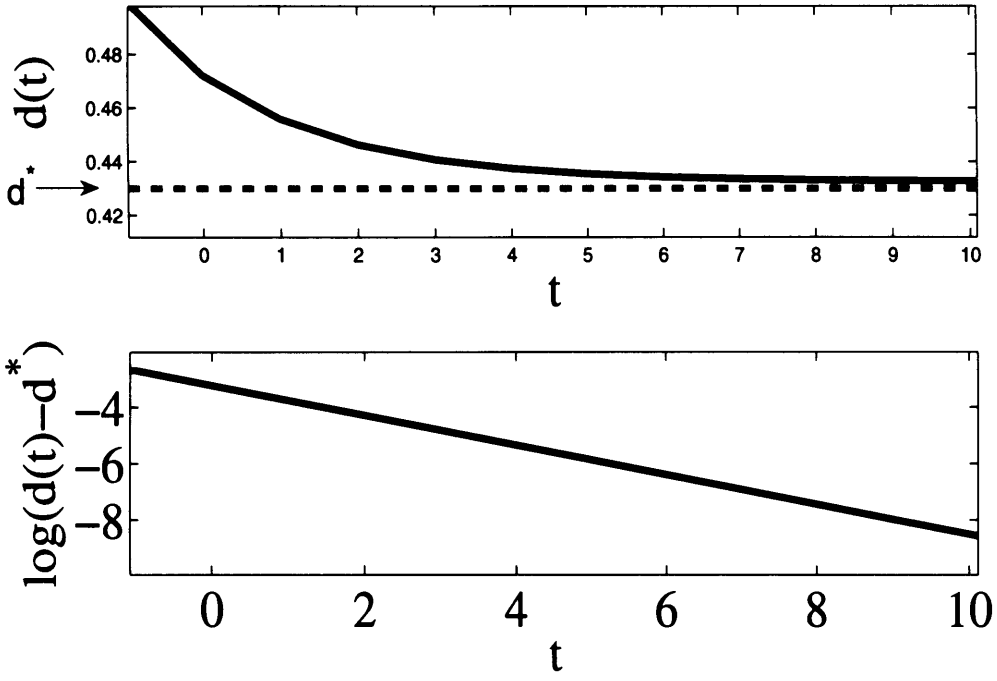


Figure 2-14: The distance between two attractors, $d(t)$, decreases when $t > 0$. However, as we can see in the lower panel, it never reaches the equilibrium distance, d^*.

order in $d(t) - d^*$,

$$d(t+1) = f(d^*) + f'(d^*)(d(t) - d^*). \tag{2.23}$$

Defining $\delta d(t) \equiv d(t) - d^*$ and using $d^* = f(d^*)$, we find that

$$\delta d(t+1) = f'(d^*)\delta d(t). \tag{2.24}$$

Iterating this equation then leads to

$$\delta d(t) = (f'(d^*))^t \delta d(0) = e^{[\log f'(d^*)]t}\delta d(0). \tag{2.25}$$

Equation 2.25 tells us the distance between trajectories in the $N \to \infty$ limit. For finite N, however, there are fluctuations, so we should write

$$\delta d(t) = \delta d(0) e^{[\log(f'(d^*))]t} + \mathcal{O}(1/N^{1/2}). \tag{2.26}$$

(see Fig.2-13). Thus, when $\delta d(0) \exp[\log(f'(d^*))t]$ becomes $\mathcal{O}(1/N^{1/2})$, trajectories that received different inputs become indistinguishable from trajectories that received the same input. Because $\delta d(0) \sim \mathcal{O}(1)$, we conclude that the time it takes for this to happen, $\tau_{\max}$ (the effective temporal memory), is given implicitly by $e^{[\log(f'(d^*))]\tau_{max}} \sim 1/\sqrt{N}$. Solving for $\tau_{\max}$, we have

$$\tau_{max} = -\frac{\log N}{2\log(f'(d^*))} + \text{const}, \tag{2.27}$$

where the constant takes into account the behaviour far from equilibrium. Eq. 2.27 tells us that τ_{max} scales as $\log N$. Consequently, networks cannot increase their temporal memory very much by increasing the number of neurons, which makes them

not very promising as classifiers of time-varying signals.

2.6 Training a linear classifier

Our estimate of τ_{max}, Eq.2.27, was based on general arguments about chaotic dynamics. However, it tells us nothing about practical issues of readout. In this section, we build a linear classifier [Jaeger, 2001, Maass et al., 2002, Bertschinger and Natschläger, 2004], and, based on it, show the same scaling for τ_{max} as in Eq.2.27.

The linear classifier, which has the form of a readout neuron, is given by

$$y(t) = \frac{1}{N}\sum_{i=1}^{N} J_i x_i(t)\,. \tag{2.28}$$

Our goal is to choose the weights so that $y(t)$ is large when a particular input is presented to the network, and small otherwise. As in the previous section, we use $u^{(k)}(t)$ to denote input k. Then, if we want our readout neuron to "detect" $u^{(k)}(t)$ at time τ (which really means we want it to detect the input $(..., u^{(k)}(\tau-2), u^{(k)}(\tau-1), u^{(k)}(\tau))$, the weights should be chosen so that when input $u^{(k)}(t)$ is presented, $y(t=\tau)$ is large and $y(t\neq\tau)$ is small, and when any other input is presented, $y(t)$ is small for all t.

In the general case, finding the optimal weights is a hard problem; it requires that we minimise the probability of making an incorrect choice when classifying patterns, which in turn requires that we know the statistics of those patterns. In the large N limit, however, the problem becomes easier, because we expect to be able to classify activity patterns that are relatively close. Consider two such close patterns, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, which are vectors with components $x_i^{(1)}$ and $x_i^{(2)}$, respectively. Now consider

a weight vector $\mathbf{J}^{(1)}$ whose job it is to detect $\mathbf{x}^{(1)}$, which for this example means maximising the effective distance between $\mathbf{J}^{(1)} \cdot \mathbf{x}^{(1)}$ and $\mathbf{J}^{(1)} \cdot \mathbf{x}^{(2)}$. Since $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are close, $\mathbf{J}^{(1)} \cdot \mathbf{x}^{(1)}$ and $\mathbf{J}^{(1)} \cdot \mathbf{x}^{(2)}$ have approximately the same variance, the optimal weight vector can depend only on the means of $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$. However, since the classifier cannot know ahead of time anything about $\mathbf{x}^{(2)}$, $\mathbf{J}^{(1)}$ can depend only on the mean of $\mathbf{x}^{(1)}$, and in fact must be proportional to it. Thus, letting $J_i^{(k)}(\tau)$ be the set of weights that detects input k at time τ, we have

$$J_i^{(k)}(\tau) = \bar{x}_i^{(k)}(\tau) \tag{2.29}$$

where $\bar{x}_i^{(k)}(\tau)$, the mean value of $x_i^{(k)}(\tau)$, is given by

$$\bar{x}_i^{(k)}(\tau) \equiv \frac{1}{R} \sum_r x_i^{(k)r}(\tau) \tag{2.30}$$

with the sum over r representing a sum over trials.

This is not a very surprising result: if we want the weights as close as possible to an activity pattern on average, we should set the weights to the average activity pattern. As we will see in the next section, this set of weights leads to a network with the same temporal memory as in Eq.2.27.

2.7 Testing a linear classifier

Given the above weights and readout neuron, we can construct a classifier by simply setting a threshold: assuming that the weights are given by Eq.2.29, then if $y(t)$ is above that threshold we say that $u^{(k)}$ was present; otherwise we say that it wasn't. To detect multiple inputs, we simply construct multiple readout neurons, each using

weights associated with different inputs (different ks in Eq.2.29). For simplicity we will consider the case in which there are only two possible inputs, $u^{(1)}$ and $u^{(2)}$; extension to multiple inputs is straightforward. Our goal in this section is to compute the optimal threshold and, given that threshold, compute the probability of correctly detecting the input that was present.

It is convenient to define

$$y_{kl}(t;\tau) = \frac{1}{N}\sum_{i=1}^{N} J_i^{(k)}(\tau)x_i^{(l)}(t)\,. \tag{2.31}$$

This quantity is the value of a readout neuron tuned to detect input $u^{(k)}(t)$ at time τ when input $u^{(l)}$ is presented, and it is useful because it can tell us how to set the thresholds. Loosely, a threshold somewhere between $y_{11}(\tau;\tau)$ and $y_{12}(\tau;\tau)$ could be used to detect input 1, and a threshold somewhere between $y_{22}(\tau;\tau)$ and $y_{21}(\tau;\tau)$ could be used to detect input 2. This is illustrated in Fig.2-15, where we plot $y_{11}(\tau;\tau)$ and $y_{12}(\tau;\tau)$ versus τ. For this particular plot it is easy to set the threshold. However, because the network is chaotic, on a different trial both $y_{11}(\tau;\tau)$ and $y_{12}(\tau;\tau)$ will be different and, consequently, potentially misclassified.

For a given threshold, Γ, the probability of correctly classifying input 1 at time τ, denoted $p_c(t;\Gamma)$, is given by

$$p_c(\tau;\Gamma) = \text{prob}[y_{11}(\tau,\tau) > \Gamma] \times \text{prob}[y_{12}(\tau,\tau) < \Gamma]\,. \tag{2.32}$$

The first term is the probability of correctly detecting input 1 when it is present; the second is the probability of *not* detecting input 1 when input 2 is present. Assuming that inputs 1 and 2 have the same statistics, the probability of correctly classifying input 2 is also $p_c(\tau;\Gamma)$; thus, we don't use different symbols for these two cases, and

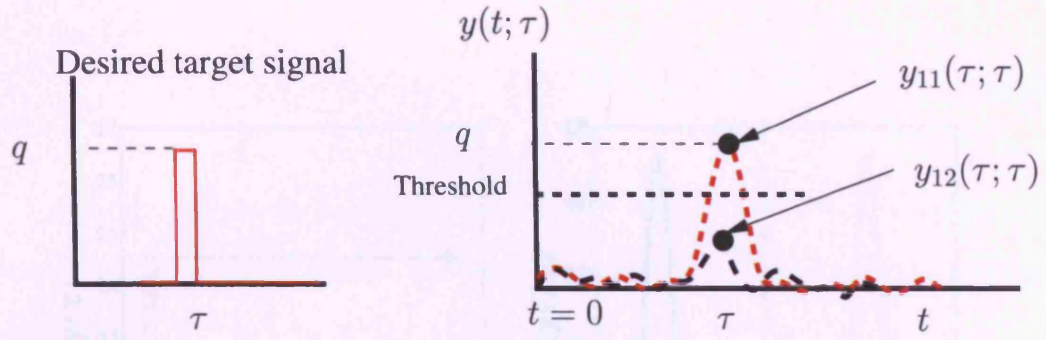


Figure 2-15: Desired and actual activity of a readout neuron trained to detect input 1. The left panel shows the desired activity of $y_{11}(\tau;\tau)$: it is equal to $q(\tau)$ at time τ and zero otherwise. The right panel shows a cartoon of actual values of both $y_{11}(\tau;\tau)$ and $y_{12}(\tau;\tau)$. Both peak at $t = \tau$, but the former has a higher peak, and so is readily distinguishable by setting a threshold (bold dashed line).

simply call $p_c(\tau;\Gamma)$ the probability of correctly classifying inputs. Note that we are focusing on time $t = \tau$. This is because the false positive rate, the probability that $y_{12}(t)$ is above threshold when the correct input was not present, turns out to be negligible.

As in previous sections, we will assume that $u^{(1)}(t)$ and $u^{(2)}(t)$ are different when $t \leq 0$ and the same when $t > 0$. Thus, discrimination gets progressively harder as τ increases. This is illustrated in Fig. Fig.2-16.

Since we want an optimal classifier, we will need to choose Γ to maximise $p_c(\tau;\Gamma)$. This requires knowledge of the distribution of $y_{kl}(\tau;\tau)$. Because $y_{kl}(\tau;\tau)$ is the sum of a large number of random variables, it is Gaussian, so all we need are its mean and variance. The mean is relatively easy to compute; using 2.31, we have

$$\overline{y}_{kl}(\tau,\tau) = \frac{1}{N}\sum_i J_i^{(k)}(\tau)\overline{x}_i^{(l)}(\tau)\,. \tag{2.33}$$

Then, using Eq.2.29 for $J_i^{(k)}(\tau)$ and Eq.2.30 for $\overline{x}_i^{(k)}(\tau)$, the mean becomes

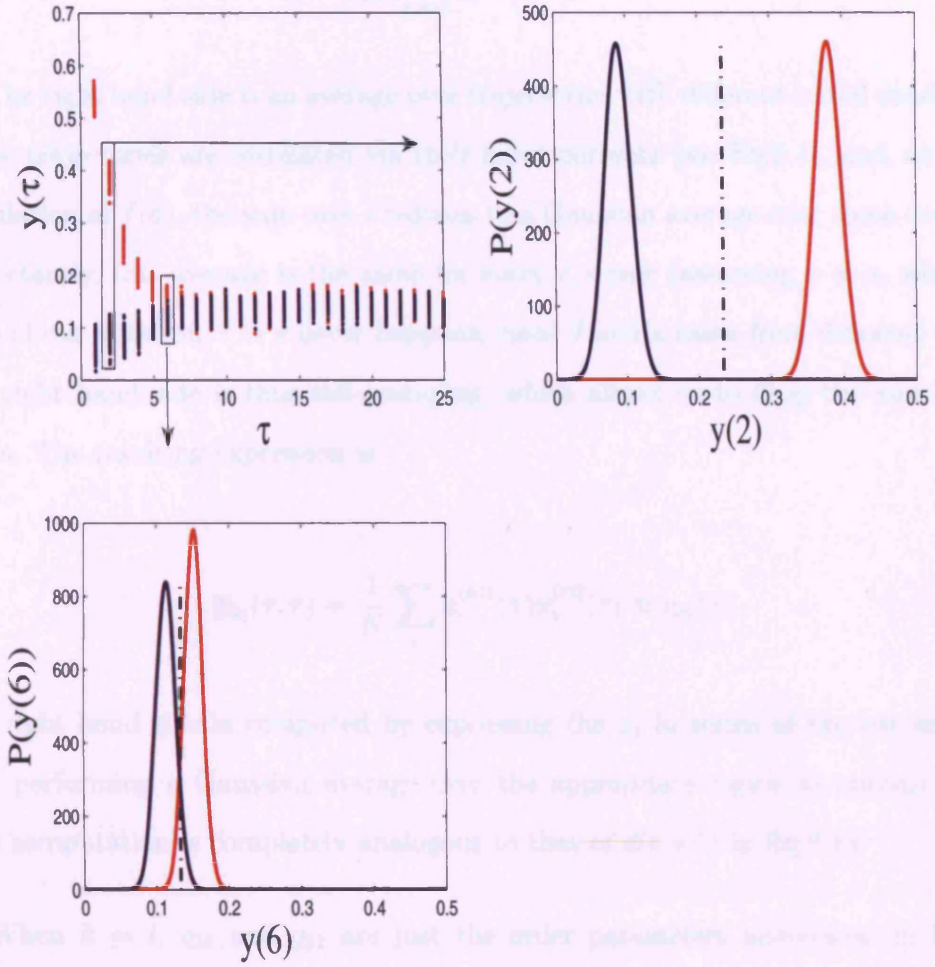


Figure 2-16: The upper left panel shows the distribution of $y_{11}(\tau)$ (red dots) and $y_{12}(\tau)$ (blue dots) for 500 trials and $N = 1000$ neurons. Each vertical line is a collection of dots. The upper right panel shows the probability density of blue and red dots for $\tau = 2$. Here the probability of misclassification is low. The lower panel shows the same distributions when $\tau = 6$. For this larger value of τ – indicating that the inputs have been the same for a longer time – the probability of misclassification is much higher. The black dashed lines is the optimum threshold.

$$\overline{y}_{kl}(\tau,\tau) = \frac{1}{R^2}\sum_{r,s=1}^{R}\frac{1}{N}\sum_i x_i^{(k)r}(\tau)x_i^{(l)s}(\tau)\,. \tag{2.34}$$

The right hand side is an average over trajectories with different initial conditions. These trajectories are correlated via their input currents (see Eq.2.1), and, as in the calculation of $f(d)$, the sum over i reduces to a Gaussian average over these currents. Importantly, this average is the same for every r, s pair (assuming $r \neq s$, which, in spite of our notation, $r = s$ never happens, since J and x came from different trials). The right hand side is thus self-averaging, which allows us to drop the sum over r and s. The resulting expression is

$$\overline{y}_{kl}(\tau,\tau) = \frac{1}{N}\sum_i x_i^{(k)1}(\tau)x_i^{(l)2}(\tau) \equiv q_{kl}(\tau)\,. \tag{2.35}$$

The right hand side is computed by expressing the x_i in terms of the current, and then performing a Gaussian average over the appropriate region in current space. This computation is completely analogous to that of $d(t+1)$ in Eq.2.13.

When $k = l$, q_{11} and q_{22} are just the order parameters introduced in Eq.2.9: $q_{kk} = 1 - 2d_{kk}$ where d_{kk} is the distance between different trajectories that receive input k. When $l \neq k$, on the other hand (say $k = 1$ and $l = 2$), q_{12} is $1 - 2d_{12}$ where d_{12} is the distance between trajectories that receive different input (1 versus 2). Typically d_{12} is larger than both d_{11} and d_{22}, which implies that q_{12} is smaller than q_{11} and q_{22}. This in turn tells us that we should put the threshold slightly above q_{12}. To determine exactly where to put it, however, we need the variance of $y_{kl}(\tau)$. A straightforward calculation (see Appendix C) yields

$$\mathrm{Var}[y_{kl}(\tau,\tau)] = \frac{1}{N}\left[\frac{1}{N}\sum_i J_i^{(k)}(\tau)^2\left(1-\overline{x}^{(l)_i}(\tau)^2\right)\right] \equiv \frac{\sigma_{kl}^2(\tau)}{N}. \qquad (2.36)$$

Note that the term inside the square brackets is $\mathcal{O}(1)$, and thus so is $\sigma_{kl}^2(\tau)$.

We can now write an explicit expression for $p_c(\tau,\Gamma)$ (Eq. 2.32),

$$\begin{aligned} p_c(\tau,\Gamma) &= \int_\Gamma^\infty \frac{dq}{[2\pi\sigma_{11}^2(\tau)/N]^{1/2}} \exp\left[-\frac{[q-q_{11}(\tau)]^2}{2\sigma_{11}^2(\tau)/N}\right] \\ &\times \int_{-\infty}^\Gamma \frac{dq'}{[2\pi\sigma_{12}^2(\tau)/N]^{1/2}} \exp\left[-\frac{[q'-q_{12}(\tau)]^2}{2\sigma_{12}^2(\tau)/N}\right]. \end{aligned} \qquad (2.37)$$

Written in terms of the error function, this expression becomes

$$p_c(\tau,\Gamma) = \frac{1}{4}\left[1+\mathrm{erf}\left(\frac{q_{11}(\tau)-\Gamma(\tau)}{[2\sigma_{11}^2(\tau)/N]^{1/2}}\right)\right]\cdot\left[1+\mathrm{erf}\left(\frac{\Gamma(\tau)-q_{12}(\tau)}{[2\sigma_{12}^2(\tau)/N]^{1/2}}\right)\right]. \qquad (2.38)$$

In the large N limit, the variance of both $y_{11}(\tau)$ and $y_{12}(\tau)$ are small, and τ can be large before the responses start being misclassified – something we have seen already in Eq.2.27. In the large τ limit Eq.2.38 simplifies considerably. This is because $\sigma_{11}(\tau)$ approaches $\sigma_{12}(\tau)$ (and both approach a constant, independent of τ), which in turn implies that the optimal value of Γ is half way between $q_{11}(\tau)$ and $q_{12}(\tau)$. Setting Γ to $\Gamma_{\mathrm{opt}} \equiv (q_{11}(\tau)+q_{12}(\tau))/2$ and defining σ_0^2 as the large τ limit of $\sigma_{kl}^2(\tau)$,

$$\sigma_0^2 \equiv \lim_{\tau\to\infty}\sigma_{kl}^2(\tau), \qquad (2.39)$$

the probability of misclassification becomes

$$p_c(\tau, \Gamma_{\rm opt}) = \frac{1}{4}\left(1 + {\rm erf}\left(N^{1/2}\frac{q_{11}(\tau) - q_{12}(\tau)}{2\sigma_0}\right)\right)^2 . \tag{2.40}$$

What Eq.2.40 tells us is that the larger N is, the closer $q_{11}(\tau)$ can be to $q_{12}(\tau)$ for the same fraction correct. Specifically, defining $\tau_{\rm max}$ via the relation $p_c(\tau_{\rm max}, \Gamma_{\rm opt}) = p_{c0}$ where p_{c0} is some desired fraction correct (say near one), then N and $\tau_{\rm max}$ are related via $N^{1/2}(q_{11}(\tau_{\rm max}) - q_{12}(\tau_{\rm max})) = {\rm constant}$. Since $q_{12}(\tau) - q_{11}(\tau) = 2(d_{11}(\tau) - d_{12})(\tau)$, and $d_{12}(\tau)$ decays exponentially toward $d_{11}(\tau)$ at rate $-[\log f'(d^*)]^{-1}$ where d^* is the limiting value of $d_{11}(\tau)$ (see Eq.2.26), this relation tells us that

$$N^{1/2} e^{[\log f'(d^*)]\tau_{max}} = {\rm constant} . \tag{2.41}$$

Taking the log of both sides, we recover Eq.2.27. In the next section we test quantitatively whether this scaling holds.

2.8 Simulations

We numerically simulated Eqs.2.1 and 2.2, and used the linear readout given in Eq.2.28 to classify inputs. We then computed the fraction of correctly classified inputs, and compared to the prediction given in Eq.2.38. The numerical and analytical results of this computation are illustrated in Fig.2-17. The parameters used to make that figure were as follows: for $t \leq 0$ the two inputs were uncorrelated, and took the values ± 1, each with probability 1/2. Because they were uncorrelated, the probability that $u_i^{(1)}(t) = u_i^{(2)}(t)$ was 1/2. For $t > 0$ the inputs took on the values ± 0.3, again each with probability 1/2, but this time they were 100% correlated: $u_i^{(1)}(t) = u_i^{(2)}(t)$ with probability 1. The fraction correct was computed for different network sizes,

N, with N ranging from 1000 to 16000. We used 500 different initial conditions for training and 500 for testing. We found no false positives.

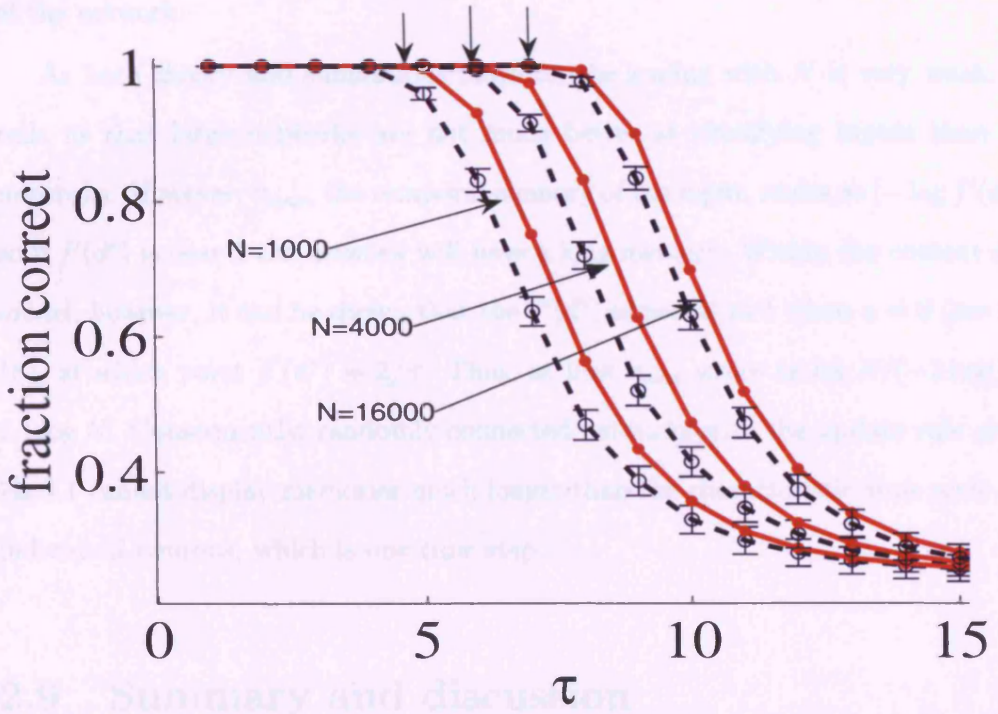


Figure 2-17: For the parameters described in the text, $f'(d^*) = 0.59$, which implies that each quadrupling of N should increase the fraction correct by $-\frac{\log 4}{2 \log 0.59} = 1.31$. This spacing, which is indicated by arrows on the top of the graph, accurately predicts the performance of the network.

As can be seen in Fig.2-17, agreement between the predictions and simulations is good, and improves with N. We can also look at the difference in $\tau_{\max}$ when the network size increases by a factor of 4. From Eq.2.27, we have

$$\Delta\tau_{max} = \tau_{max}^{4N} - \tau_{max}^{N} = -\frac{\log(4)}{2\log(f'(d^*)}. \tag{2.42}$$

For the above parameters, $f'(d^*) = 0.59$, which implies that each quadrupling of

N should increase the fraction correct by 1.31. This spacing, which is indicated in Fig.2-17 by the arrows at the top of the graph, accurately predicts the performance of the network.

As both theory and simulations indicate, the scaling with N is very weak. This tells us that large networks are not much better at classifying inputs than small networks. However, τ_{max}, the temporal memory of the input, scales as $[-\log f'(d^*)]^{-1}$, so if $f'(d^*)$ is near 1 the network will have a long memory. Within the context of this model, however, it can be shown that the $f'(d^*)$ is closest to 1 when $u = 0$ (see Fig.2-18), at which point $f'(d^*) = 2/\pi$. Thus, at best τ_{max} scales as $\log N/(-2\log(\frac{2}{\pi})) \approx 1.1 \log N$. Consequently, randomly connected networks with the update rule given in Eq.2.1 cannot display memories much longer than the characteristic time scale of the individual neurons, which is one time step.

2.9 Summary and discussion

Using a reduced model, we showed that randomly connected networks cannot process time-varying signals much better than their constituent neurons – meaning the temporal memory of the network can be boosted by only $\mathcal{O}(\log N)$ above the time constants of the individual neurons and synapses. This conclusions relied on the observation that our networks are always chaotic, so they cannot access the edge of chaos, where the network time constant diverges to infinity [Bertschinger and Natschläger, 2004, Derrida, 1987].

An issue, of course, is whether this result applies to biologically realistic. This seems plausible, based on the following reasoning: Consider a network in which each neuron sums its input current and emits a spike whenever the total current crosses a

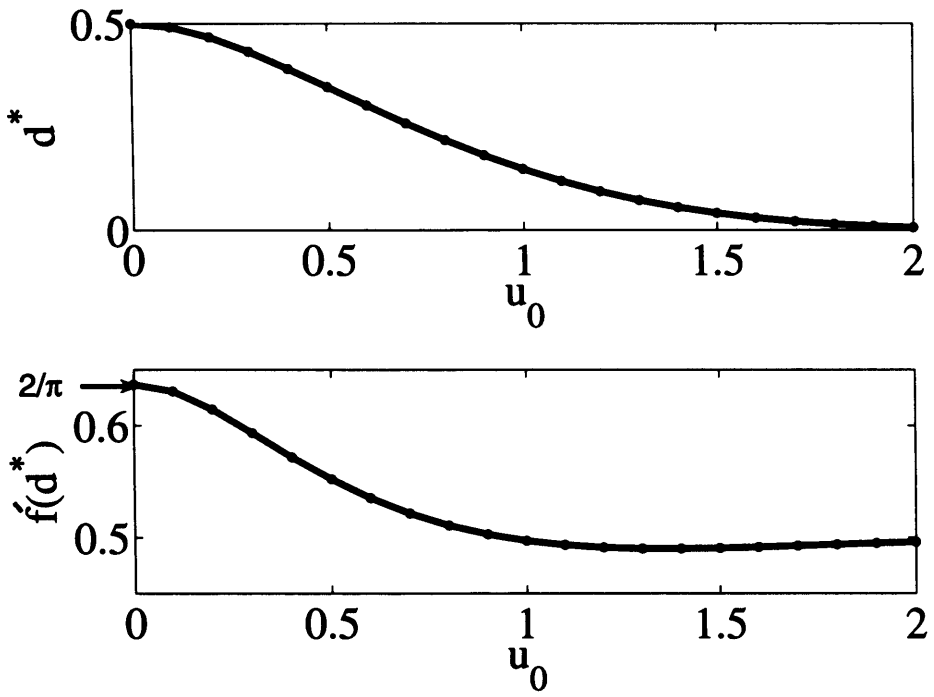


Figure 2-18: Equilibrium distance versus input amplitude, u_0. In both panels, $P(u) = [\delta(u - u_0) + \delta(u + u_0)]/2$. The upper panel shows that by decreasing the amplitude of the input (decreasing u_0), the equilibrium distance, d^*, increases toward 1/2. The lower panel shows that $f'(d^*)$ is largest when $u_0 = 0$.

threshold. If the number of neurons is large, the analysis of Sec.2.4 applies, and the network will always be in a regime in which nearby trajectories diverge. Although this reasoning is suggestive rather than definitive, the basic conclusion that realistic networks are chaotic is consistent with analysis of more realistic networks of spiking neurons [Banerjee, 2001a,b].

In the next chapter we investigate why our network is always chaotic, while Bertschinger and Natschläger's network could operates on the edge of chaos.

Chapter 3

Sparse, randomly connected networks and the edge of chaos

3.1 Introduction

In chapter 2, we considered the class of networks analysed by Bertschinger and Natschläger [2004], and showed that highly connected networks within this class always exhibit chaotic activity. This seems to be inconsistent with Bertschinger and Natschläger [2004], who found that randomly connected networks *could* operate at the edge of chaos, a regime between order and chaos that allows networks to exhibit long temporal memory. In this chapter we address, and resolve, this apparent inconsistency.

A key assumption in our analysis was that the probability of a connection between two neurons is independent of network size. This means that the average number of synaptic connections per neuron, K, is in the order of the number of neurons, N. In this chapter we consider a different scaling: K fixed as N goes to infinity. This is the

scaling considered by Bertschinger and Natschläger [2004].

We will follow exactly the same analysis that we used in chapter 2. Because K is finite, however, our mean field equation will change slightly. Our aim in this chapter is the following:

- Determine under what conditions the network proposed by Bertschinger and Natschläger [2004] can operate at the edge of chaos.
- For networks that operate on the edge of chaos, determine how the temporal memory, τ_{max}, scales with the number of neurons, N.

3.2 Formal analysis

In this section we repeat the analysis of Chapter 2, and derive a time evolution equation for the Hamming distance when the network receives the same input on two trials. The only difference here versus in Chapter 2 is that we fix the average number of connections, K, as N goes to infinity. We will, however, assume that K is large enough that the distribution of the number of connections per neuron is sharply peaked, and we will ignore fluctuations around the peak value. Including fluctuations would have no effect on our results.

Starting with the definition of Hamming distance, $d(t) = \text{prob}(x_i^1(t) \neq x_i^2(t))$, and assuming exactly K connections, $f(d;u,u)$ can be written

$$f(d;u,u) = \sum_{C=0}^{K} \text{prob}(x_i^1(t) \neq x_i^2(t) \mid C;K,u)P(C|d), \tag{3.1}$$

where C is the number of neurons presynaptic to neuron i that have different states on two different trials (see Fig.3-1), and $P(C|d)$ is the probability of having C different

states given d. The latter distribution is binomial,

$$P(C|d) = \binom{K}{C} d^C (1-d)^{K-C}, \tag{3.2}$$

where $\binom{K}{C} \equiv \frac{K!}{C!(K-C)!}$. Let us make the definition

$$\text{prob}(x_i^1(t) \neq x_i^2(t) \mid C; K, u) \equiv P(\text{diff} \mid C; K, u). \tag{3.3}$$

Then, inserting Eqs.3.2 and 3.3 into Eq.3.1, we have

$$f(d; u, u) = \sum_{C=1}^{C=K} \binom{K}{C} d^C (1-d)^{K-C} P(\text{diff} \mid C; K, u). \tag{3.4}$$

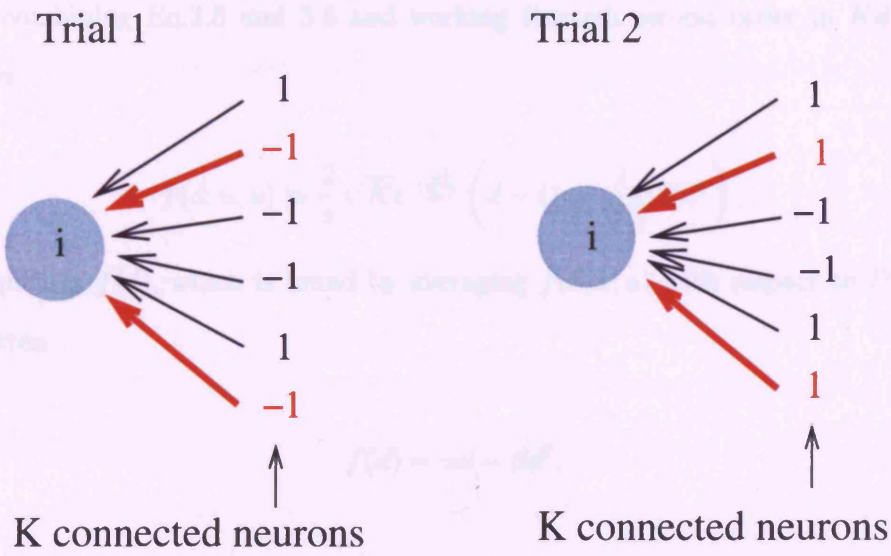


Figure 3-1: Neuron i receives input from $K = 6$ other neurons. The number of neurons that have different states on the two trials, C, is 2. To take a concrete example, if we assume that the Hamming distance, d, is 1/3, then the probability of observing the above configuration is $\binom{6}{2} d^2 (1-d)^4 \approx 0.33$.

In the previous section, K/N was fixed as N went to infinity, which implied that

$Kd \gg 1$. In this limit, the first term in Eq.3.4, $\binom{K}{C} d^C (1-d)^{K-C}$, is very sharply peaked around $C = Kd$. In this chapter, however, K is finite as N goes to infinity. Thus, for small enough d, $Kd \ll 1$. In this limit, $\binom{K}{C} d^C (1-d)^{K-C}$ becomes Poisson,

$$\binom{K}{C} d^C (1-d)^{K-C} \to \frac{(Kd)^C e^{-Kd}}{C!}. \tag{3.5}$$

Because this distribution is a rapidly decreasing function of C, we can evaluate $P(\text{diff} \mid C; K, u)$ in the small C limit. The analysis is almost identical to that used to derive Eq.2.15 (details are given in Appendix D), and we find that

$$P(\text{diff} \mid C; K, u) \approx \frac{2}{\pi} \sqrt{\frac{C}{K}} e^{-\frac{u^2}{2\sigma^2}}. \tag{3.6}$$

Then, combining Eq.3.5 and 3.6 and working through second order in Kd, Eq.3.4 becomes

$$f(d; u, u) \approx \frac{2}{\pi} \sqrt{K} e^{-\frac{u^2}{2\sigma^2}} \left(d - (1 - \frac{1}{\sqrt{2}}) K d^2 \right). \tag{3.7}$$

Consequently $f(d)$, which is found by averaging $f(d, u, u)$ with respect to $P(u)$, can be written

$$f(d) = \alpha d - \beta d^2, \tag{3.8}$$

where

$$\alpha = \frac{2}{\pi} \sqrt{K} \int du \; P(u) e^{-\frac{u^2}{2\sigma^2}}, \tag{3.9}$$

$$\beta = \left(1 - \frac{1}{\sqrt{2}} \right) K \alpha.$$

By definition, nearby trajectories are neutrally stable, and the network operates on the edge of chaos, if $f'(d)|_{d=0} = 1$. Examining Eq.3.8, we see that this happens when $\alpha = 1$. Denoting α_{ed} and β_{ed} the values of α and β at the edge of chaos, we have

$$\begin{aligned} \alpha_{ed} &= 1, \\ \beta_{ed} &= \left(1 - \frac{1}{\sqrt{2}}\right) K_{ed}, \end{aligned} \tag{3.10}$$

where

$$K_{ed} \equiv \frac{\pi^2}{4\left(\int du P(u) e^{-\frac{u^2}{2\sigma^2}}\right)^2}. \tag{3.11}$$

Fig.3-2 shows the three possible regimes: converging, neutrally stable (edge of chaos), and diverging. These correspond to small, intermediate and large K, respectively.

3.3 Main scenario

We turn now to the question: how does the temporal memory of a network scale with the number of neurons, N? We consider the same class of inputs that we discussed in chapter 2 – inputs that are different between $t = -T$ and $t = 0$ and are the same between $t = 0$ and $t = \tau$. We want to know how large we can make τ before the state of the network cannot tell us which input it received.

To determine how the distance between trajectories evolves, we need to solve our standard equation $d(t+1) = f(d(t))$. At the edge of chaos, $f(d)$ is given by Eq.3.8 with $\alpha = \alpha_{ed}$ and $\beta = \beta_{ed}$. This equation has the approximate solution

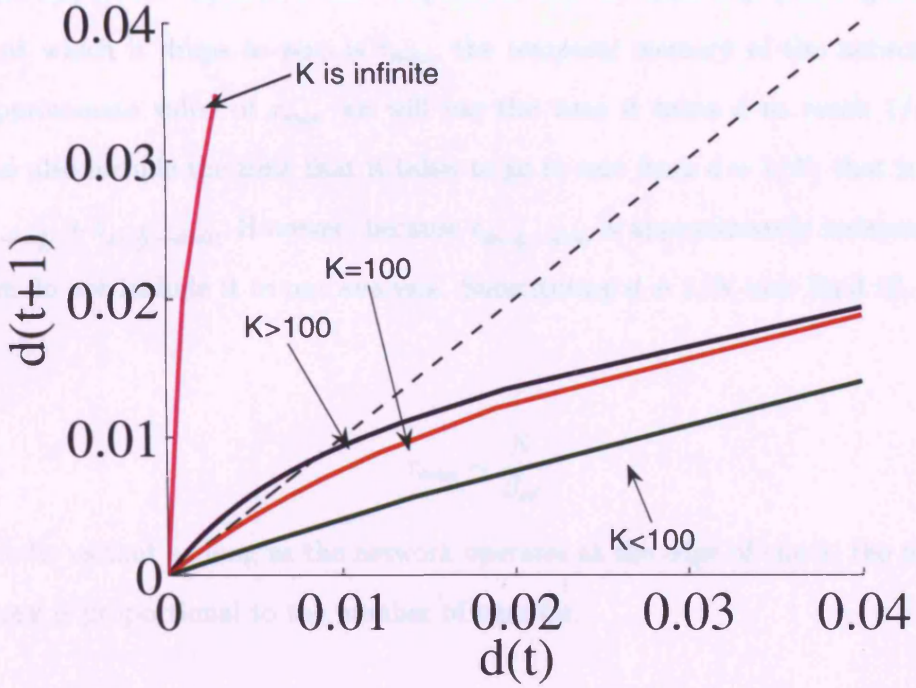


Figure 3-2: Three different dynamical regimes: converging (green), neutrally stable (edge of chaos, red), and diverging (blue and magenta). $K_{ed} = 100$ is the number of connections needed for neutral stability. The magenta line indicates the regime that we obtained in chapter 2 ($K \to \infty$), for which there is a square root singularity at $d = 0$.

$$d(t) = \frac{1}{\beta_{ed} t}. \tag{3.12}$$

To verify that Eq.3.12 is approximately correct, we substitute it into Eq.3.8, and we find that

$$d(t+1) = \frac{1}{\beta_{ed}(t+1)} \approx \frac{1}{\beta_{ed} t} - \frac{1}{\beta_{ed} t^2} = d(t) - \beta_{ed} d(t)^2 = f(d(t)).$$

Eq.3.12 is valid only so long as $d \gg 1/N$. When $d \sim 1/N$, however, fluctuations

become important. In fact, d can drop to zero in one time step (see Fig.3-3). The time at which it drops to zero is τ_{max}, the temporal memory of the network. For an approximate value of τ_{max} we will use the time it takes d to reach $1/N$. We should also include the time that it takes to go to zero from $d = 1/N$; that is $\tau_{max} = t_{d(0)\to d=\frac{1}{N}} + t_{d=\frac{1}{N}\to d=0}$. However, because $t_{d=\frac{1}{N}\to d=0}$ is approximately independent of N, we do not include it in our analysis. Substituting $d = 1/N$ into Eq.3.12, we find that

$$\tau_{max} \sim \frac{N}{\beta_{ed}}. \tag{3.13}$$

This tells us that as long as the network operates at the edge of chaos, the temporal memory is proportional to the number of neurons.

3.4 Simulation results

To validate our analysis, we conducted simulations with the average number of connections per neuron set to K_{ed}. There is, however, a difference in these simulations compared to the ones in Sec. 2.8. In the simulations in Sec. 2.8, we averaged over initial conditions. However, at the edge of chaos, for sufficiently large T all trajectories converge to the same point in activity space (Fig.3-4). This makes $\tau_{\max}$ the same for all initial conditions. Thus, instead of averaging over initial conditions, we average over different realizations of the input, $u_i(t)$.

Specifically, the simulations are carried out as follows: for each realization of the input, we initialise the network to a random configuration, and then drive the network with two different inputs, $u^{(1)}$ and $u^{(2)}$, between $t = -T$ and $t = 0$. The inputs take on the values $\pm u_0$, each with probability 1/2, and $u_i^{(1)}$ and $u_i^{(2)}$ are independent. We

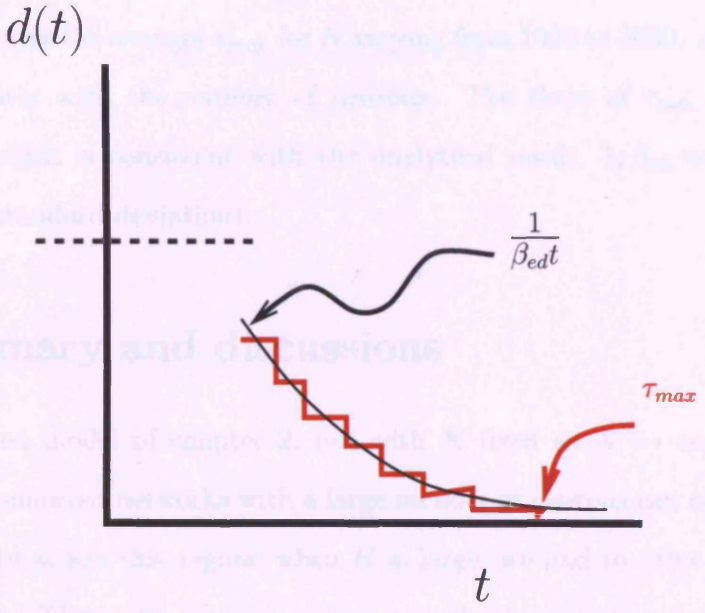


Figure 3-3: Evolution of the Hamming distance when the network operates near the edge of chaos and $d \sim 1/N$. The black curve shows the evolution of $d(t)$ in the absence of fluctuations (Eq.3.12). The red curve shows the actual evolution of $d(t)$.

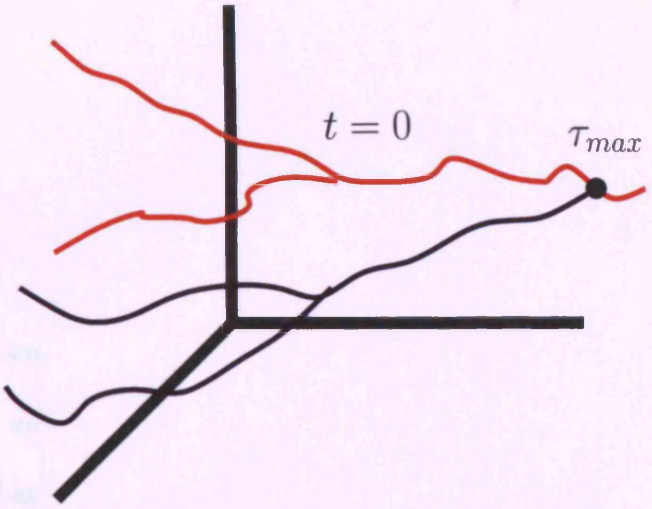
use $K_{ed} = 24$, and, to ensure that the network operates near the edge of chaos, we choose u_0 according to Eq.3.11, which tells us that $u_0 = \sqrt{\log(\frac{4K_{ed}}{\pi^2})} = 1.44$. We choose T large enough to eliminate transients. At time $t = 0$, the two inputs become the same (but still equal to $\pm u_0$ with probability 1/2), and we measure how long it takes the two trajectories to merge. This time is $\tau_{\max}$ for that set of inputs; we then repeat this process for many realizations of the input and compute the average $\tau_{\max}$.

Fig.3-5 illustrates the average τ_{max} for N varying from 1000 to 3000. As predicted, τ_{max} scales linearly with the number of neurons. The slope of $\tau_{\max}$ versus N is 0.135 ± 0.067, which is consistent with the analytical result, $1/\beta_{ed} = 0.148$ (it is within 1/5 of a standard deviation).

3.5 Summary and discussions

Using the reduced model of chapter 2, but with K fixed as $N \to \infty$, we showed that randomly connected networks with a large number of neurons can operate at the edge of chaos. To access this regime when K is large, we had to drive the network with large input. Thus, the network is mainly reflecting what comes into it. In addition, the network was making discriminations based on the difference in activity of a very small number of neurons. This raises issues of robustness. In particular: what happens to this result when there is a small amount of noise, either in the input or in the network? In the next section, we consider the latter.

One realisation of input for red and blue trajectories.



Two realisations of input for red and blue trajectories.

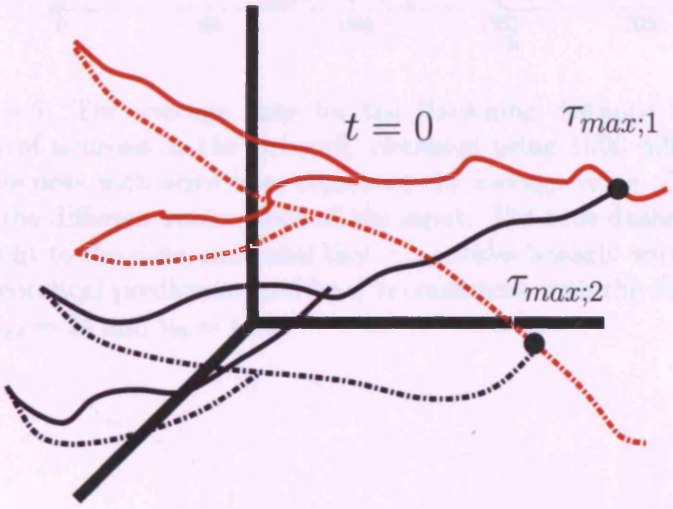


Figure 3-4: Trajectories with one (upper panel) and two (lower panel) realisations of inputs. The upper panel shows the trajectories and inputs used in chapter 2. In the lower panel we added another realisation of input. This input drives the trajectories to another location (dashed trajectories), and produces a different value of τ_{max}. To estimate τ_{max} we repeat this process many times and take an average.

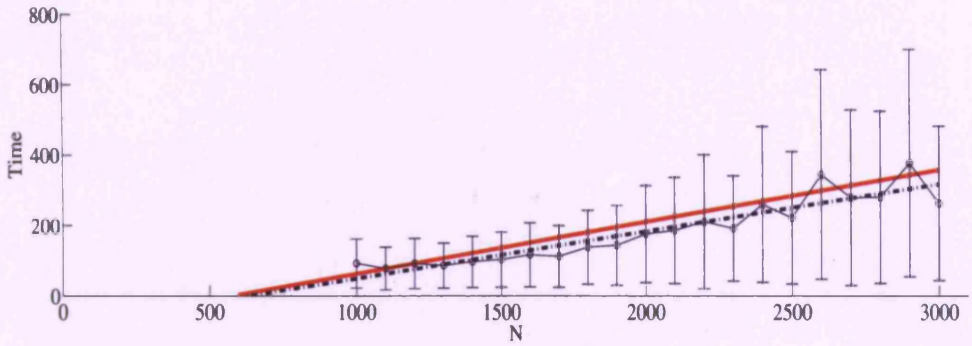


Figure 3-5: The average time for the Hamming distance to reach zero versus the number of neurons in the network, obtained using 1000 different input realisations. The blue dots with error bars represent the average value of τ_{max}, where the average is over the different realizations of the input. The blue dashed line, which is the least squares fit to the data, indicates that τ_{max} scales linearly with the number of neurons. The theoretical prediction (red line) is consistent with the simulations (see text). We used $K_{ed} = 24$ and $u_0 = 1.44$.

Chapter 4

Temporal memory in networks with synaptic failures

4.1 Introduction

In chapters 2 and 3, we considered a reduced network in which the underlying activity was deterministic. In chapter 2, we provided a mean-field theory for highly connected networks and found that such networks could not maintain long temporal memory. Then, in chapter 3, we analysed sparsely connected networks, and found that they could operate at the edge of chaos, and therefore, exhibit long memory. In such sparse networks, we showed that long memory is possible only if the amplitude of the input is large.

Real neurons, however, are not deterministic – they exhibit synaptic failures [Castillo and Katz, 1954, Fatt and Katz, 1951, 1952, Katz, 1959]. If we want to build a model that biologically is plausible, we need to take failures into account. In this chapter, we address the question: how does the probabilistic nature of synaptic trans-

mission affect the temporal memory of the network? We begin by reviewing some basic concepts about failures in biological neurons.

4.2 Synaptic failures; brief review

Probabilistic synaptic release was discovered by Bernard Katz and his colleagues [Catstillo and Katz, 1954, Fatt and Katz, 1951, 1952, Katz, 1959] in the 1950s. Their work resulted in the so-called standard model [Zigmond et al., 1999], in which each presynaptic terminal contains n quanta of neurotransmitter, and the probability that each of them is released when the presynaptic neuron fires is the same. This results in a binomial distribution for the number of quanta released. Katz and colleagues worked in the frog motor endplates, where n is large and the release probability is near 1. Failures, however, turned out to be a ubiquitous feature of the central nervous system [Walmsley et al., 1987, Volgushev et al., 2004].

There is, however, at least one major difference between central synapses and the motor endplates: in the motor endplate n is large, while in central synapses it is small. The probability of failure varies significantly from one brain regions to the next [Stacey and Durand, 2001], and the value in cortex can be as high as 90%, although 30-60% is more typical [Walmsley et al., 1987, Volgushev et al., 2004].

Below we consider the effect of failures on our network. For simplicity we will assume that the number of quanta, n, is 1, and each synapse has the same probability of release.

4.3 Formal analysis: neurons with failures

To incorporate failures into our model, we modify the current, Eq. (2.2), so that it has the form

$$h_i(t) = \sum_{j=1}^{N} \eta_{ij}(t) w_{ij} x_j(t), \tag{4.1}$$

where $\eta_{ij}(t)$ is a random variable that takes on the values 0 and 1. It is 1 with probability p and 0 with probability of $1-p \equiv p_f$ (p_f is the probability of a failure). $\eta_{ij}(t)$ is not quenched; it is drawn i.i.d. from a Bernoulli distribution at each time step.

As in Sec. 2.2, we want to know how the distance between trajectories evolves in time; that is, how $d(t+1)$ depends on $d(t)$. We will start by considering the $K \to \infty$ regime. In this regime, the analysis follows almost exactly that of Sec. 2.4. In fact, the only difference is that the existence of failures modifies the covariance matrix. Again denoting the covariance matrix Σ_{kl}, $k = 1, 2$, we have

$$\Sigma_{kl} = \frac{1}{N} \sum_{i=1}^{N} h_i^k h_i^l = \sum_{j,j'=1}^{N} x_j^k x_{j'}^l \left(\frac{1}{N} \sum_i w_{ij} w_{ij'} \eta_{ij}^k \eta_{ij'}^l \right), \tag{4.2}$$

where we used Eq.4.1 for the currents, h_i^k. Because the η_{ij} are drawn independent of the weights, it follows that

$$\frac{1}{N} \sum_i w_{ij} w_{ij'} \eta_{ij}^k \eta_{ij'}^l = \left(\frac{1}{N} \sum_i w_{ij} w_{ij'} \right) \left(\frac{1}{N} \sum_i \eta_{ij}^k \eta_{ij'}^l \right).$$

We have already seen that the first term on the right hand side is equal to $\sigma^2 \delta_{jj'}/N$. Consequently, we can set j to j'. Once we do that, the second term is p^2 if $k \neq l$ (because η_{ij}^1 and η_{ij}^2 are independent), and p otherwise (because $(\eta_{ij})^2 = \eta_{ij}$). Combining

these results with Eq.4.2, we have

$$\Sigma = p\sigma^2 \begin{pmatrix} 1 & qp \\ qp & 1 \end{pmatrix}, \tag{4.3}$$

where, as in chapter 2, $q = \frac{1}{N}\sum_{i=1}^{N} x_i^{(1)} x_i^{(2)} = 1 - 2d$.

With this new covariance matrix, we can easily compute the function $f(d)$ that relates $d(t+1)$ to $d(t)$. Analogous to Eq.2.11, we define $f(d;p,u,u)$ via

$$f(d;p) = \int du\, P(u) f(d;p,u,u), \tag{4.4}$$

and we have (see Appendix.A)

$$f(d;p,u,u) = \int_{-\infty}^{\infty} \frac{db}{\sqrt{\pi\lambda_b}} e^{-\frac{b^2}{\lambda_b}} \int_{-|b|-u}^{|b|-u} \frac{da}{\sqrt{\pi\lambda_a}} e^{-\frac{a^2}{\lambda_a}}, \tag{4.5}$$

where $\lambda_a = p\sigma^2(1+qp)$ and $\lambda_b = p\sigma^2(1-qp)$. When $p = 1$, this expression reduces to Eq.2.13.

As in chapter 2, when u is drawn from a Gaussian distribution with zero mean and variance σ_u^2, $f(d;p)$ can be calculated analytically (see Appendix B), and the result is

$$f(d;p) = \frac{2}{\pi} \arcsin \sqrt{B\left(\frac{p_f}{2} + (1-p_f)d\right)}, \tag{4.6}$$

where $B \equiv \frac{p\sigma^2}{p\sigma^2+\sigma_u^2}$.

Fig.4-1 shows the evolution of the Hamming distance in the absence of input and $p_f = 0.2$. The plot shows that the analytical and simulation results are consistent. Note also that, unlike in Chapter 2, $d = 0$ is not a fixed point.

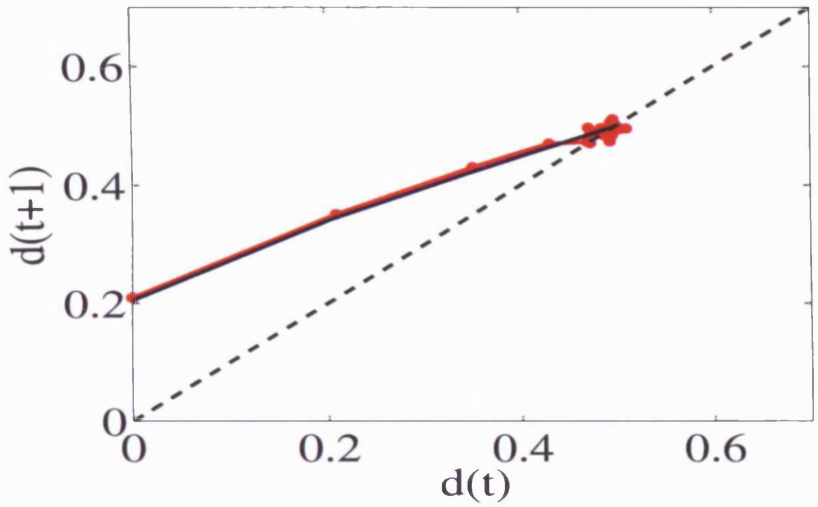


Figure 4-1: Comparison of mean-field equations, Eq.4.6 (blue line), and simulations (red line), for $u = 0$ (and thus $B = 1$). The red dots near the dashed line correspond to the equilibrium distance, which is reached after only a few iterations.

To compute how long it takes for a network of size N to forget which of two inputs it received, we follow exactly the same analysis as in chapter 2. We find that τ_{max} has the same form as in Eq.2.27; that is

$$\tau_{max} = -\frac{\log N}{2 \log f'(d^*; p)} + \text{const.} \tag{4.7}$$

To compute $f'(d; p)$, we use (see Appendix.A.2)

$$f'(d; p, u, u) = \frac{\partial f(d; p.u, u)}{\partial d} = \frac{2p}{\pi\sqrt{1 - q^2p^2}} e^{-\frac{u^2}{p(1+qp)\sigma^2}}. \tag{4.8}$$

Thus,

$$f'(d; p) = \frac{2p}{\pi\sqrt{1 - q^2p^2}} \int du\, P(u) e^{-\frac{u^2}{p(1+qp)\sigma^2}}. \tag{4.9}$$

4.4 Simulations

To validate Eq.4.7, we performed simulations for a range of N, using the procedure described in Sec.2.6. The fraction of correct versus time, τ, is illustrated in Fig.4-2 for networks ranging in size from from $N = 1000$ to $N = 16000$. We used 500 different initial conditions for training section and 500 for testing. The input parameters were the same as in Sec. 2.8: for $t \leq 0$ the two inputs were uncorrelated, and took the values ± 1, each with probability $1/2$. Because they were uncorrelated, the probability that $u_i^{(1)}(t) = u_i^{(2)}(t)$ was $1/2$. For $t > 0$ the inputs took on the values ± 0.3, again each with probability $1/2$, but this time they were 100% correlated: $u_i^{(1)}(t) = u_i^{(2)}(t)$ with probability 1. The probability of failures, p_f, was set to 0.2, and the variance of the input, σ^2, was set to 1. Using Eq.4.7, we have: $\Delta\tau_{max}^{anal} = \tau_{max}^{4N} - \tau_{max}^{N} \approx 0.9$. The spacing, which is indicated in Fig.4-2 by the arrows on the top of the graph, predicts fairly accurately the performance of the network.

As in the deterministic case, both theory and simulations indicate that τ_{max} scales as $\log N$. This tells us that large networks with failures, like their deterministic counterparts, are not much better at remembering past inputs than small networks.

4.5 How large must p_f be to have short memory?

In this chapter and in Chapter 2, we showed that when K/N is fixed as $N \to \infty$, randomly connected networks always exhibit short temporal memory. However, in Chapter 3 we showed that when we hold K fixed as $N \to \infty$, randomly connected networks *can* exhibit long memory. The issue we address in this section is whether this long temporal memory in the fixed K regime is robust to failures. Or, more

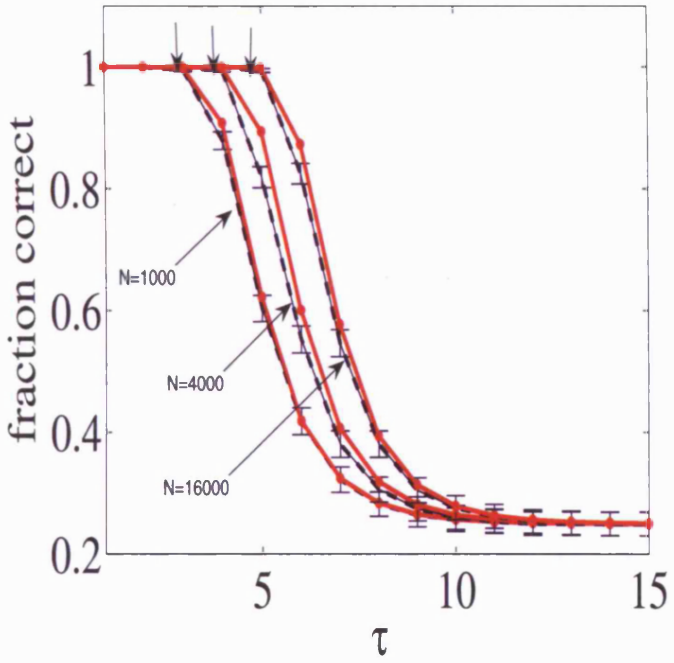


Figure 4-2: For the parameters provided in the text, $f'(d^*) = 0.42$, which implies that each quadrupling of N should increase the fraction correct by $-\frac{\log 4}{2 \log 0.42} = 0.9$. This spacing, which is indicated by arrows on the top of the graph, predicts fairly accurately the performance of the network. Red: analytic predictions. Blue: simulations.

explicitly: how large do we have to make the probability of failures, p_f, before the network exhibits short temporal memory?

To answer this question, we compute $f(d;p)$ in the regime $Kd \ll 1$, which is the long memory regime that we investigated in Chapter 3. Our starting point is Eq.3.4, but with $P(\text{diff}|C;K,u)$ replaced by $P(\text{diff}|C;K,u,p)$,

$$f(d;u,u,p) = \sum_{C=1}^{C=K} \binom{K}{C} d^C (1-d)^{K-C} P(\text{diff}|C;K,u,p)\,, \tag{4.10}$$

where, as in Eq.3.3,

$$P(\text{diff}|C;K,u,p) = \text{prob}(x_i^1(t) \neq x_i^2(t)|C;K,u,p) \tag{4.11}$$

and C is the number of presynaptic neurons that are different on two trials. Averaging Eq.4.10 over u, we arrive at

$$f(d;p) = \int du\; p(u) \sum_{C=1}^{C=K} \binom{K}{C} d^C (1-d)^{K-C} P(\text{diff}|C;K,u,p). \tag{4.12}$$

In Appendix D, we show that in the limit that $C/K \ll 1$ (the limit of interest here, because otherwise $Kd \gg 1$), and $p_f \ll 1$, $P(\text{diff}|C;K,u,p)$ is given by

$$P(\text{diff} \mid C;p,K,u,p) \approx \frac{2}{\pi}\sqrt{\frac{p_f}{2} + \frac{C}{K}}\, e^{-u^2/2\sigma^2}. \tag{4.13}$$

Combining Eqs.4.12 and 4.13, and keeping terms only up to $\mathcal{O}(d^2)$, we find that

$$f(d;p) = A_0 + A_1 d - A_2 d^2 \tag{4.14}$$

where

$$A_0 = \gamma\sqrt{\frac{p_f}{2}}, \quad A_1 = \gamma\sqrt{\frac{p_f}{2}}K\alpha_1, \quad A_2 = \gamma\sqrt{\frac{p_f}{2}}K^2\alpha_2, \tag{4.15}$$

and γ, α_1 and α_2 are given by

$$\gamma \equiv \int du\ P(u)e^{-\frac{u^2}{2\sigma^2}} \tag{4.16}$$

$$\alpha_1 \equiv \sqrt{1+c} - 1 \tag{4.17}$$

$$\alpha_2 \equiv \sqrt{1+c} - (1/2)(1+\sqrt{1+2c})\,. \tag{4.18}$$

with

$$c \equiv \frac{2}{Kp_f}\,. \tag{4.19}$$

It turns out that we can ignore the third term in Eq.4.14, A_2d^2. This is because $A_2d^2/A_1d = Kd(\alpha_2/\alpha_1) \ll 1$, where the "$\ll$" follows because we are in the regime $Kd \ll 1$, and α_2/α_1 is at most $2 - 2^{-1/2}$ (which occurs when c is large; as $c \to 0$, $\alpha_2/\alpha_1 \to c/4$). Thus, through first order in Kd, we have

$$f(d;p) = A_0 + A_1 d\,. \tag{4.20}$$

The equilibrium Hamming distance, d^*, which occurs when $d^* = f(d^*;p)$, is then given by

$$d^* = \frac{A_0}{1-A_1}\,, \tag{4.21}$$

and the slope at d^*, $f'(d^*;p)$, is

$$f'(d^*;p) = A_1 \,. \tag{4.22}$$

What we now show is that if Kp_f is $\mathcal{O}(1)$, then the condition $Kd^* \ll 1$ implies that $1 - f'(d^*;p)$ is also $\mathcal{O}(1)$, which in turn implies that the network has short memory.

Our first step is to solve for γ in terms of Kd^*. Using Eq.4.21, we have

$$Kd^* = \frac{KA_0}{1-A_1} = \frac{\gamma(K/c)^{1/2}}{1-\gamma(K/c)^{1/2}\alpha_1} \,. \tag{4.23}$$

Solving this equation for γ gives us

$$\gamma = \frac{Kd^*}{(K/c)^{1/2}(1+\alpha_1 Kd^*)} \,. \tag{4.24}$$

Then, using Eq.4.22 for $f'(d^*;p)$ and Eq.4.15 for A_1, we find that

$$f'(d^*) = \frac{\alpha_1 Kd^*}{1+\alpha_1 Kd^*} \tag{4.25}$$

Since the temporal memory scales as $1/|\log f'(d^*;p)|$, it follows that the requirement for long temporal memory is that $f'(d^*;p)$ be near 1. From Eq.4.25, we see that this happens only when $\alpha_1 \gg 1$, which in turn requires that $Kp_f \ll 1$ (see Eqs.4.17 and 4.19). More quantitatively, let's say $\alpha_1 Kd^* = 1$, which would imply a temporal memory of order $1/\log 2$, which is short. In the limit that Kd^* is large, $\alpha_1 Kd^* = 1$ when

$$p_f = \frac{2(Kd^*)^2}{K} \,. \tag{4.26}$$

Since K is large and Kd^* is small, what this equation tells us is that an extremely

small failure will rate will drastically shorten memory.

4.6 Conclusions

Using a reduced model, we showed – not surprisingly – that failures have a deleterious effect on the temporal memory of randomly connected networks. Importantly, we found that the long temporal memory regime found in Chapter 3 is destroyed by even a small amount of noise (in our case, failures).

Chapter 5

Conclusion

5.1 Summary and discussion

In this thesis we tried to understand how to build networks that can exhibit long temporal memory, and, of course, how to make them biologically plausible. We focused on randomly connected networks, as such networks require little fine tuning and can exhibit long memory if they operate on the edge of chaos.

The idea of operating on the edge of chaos in randomly connected networks was proposed first by Jaeger [2001], Jaeger and Haas [2004], and then, independently, by Maass et al. [2002]. Importantly, Maass and colleagues showed that a network of spiking neurons can exhibit reasonably long temporal memory – long enough to discriminate different spoken words when they are suitably encoded as input spike trains.

While the simulations with spiking neurons by Maass and colleagues were encouraging, a detailed understanding of operation on the edge of chaos was missing, primarily because spiking networks are so hard to analyse. To remedy this, Bertschinger

and Natschläger [2004] considered a tractable, although simplified, model based on McCulloch and Pitts [1943] neurons. Within the context of this model, they showed that randomly connected neurons can operate at the edge of chaos, and thus have long temporal memory.

While Bertschinger and Natschläger [2004] provided an extremely nice analysis of randomly connected networks, they restricted themselves to a regime in which each neuron makes a small number of connections. This motivated us to extend their analysis to the more realistic regime of high connectivity. We thus developed a mean-field theory for highly connected neurons in large networks, the regime in which biological neurons operate. Precisely speaking, our model was constructed in the limit $N \to \infty$ with K/N fixed, where K is the average number of connections per neuron and N is the number of neurons. We found that such networks can *not* operate on the edge of chaos. Instead, their temporal memory is short, on the order of the time constants of the neurons within the network. Moreover, the temporal memory depends weakly on network size – it increases only as $\log N$ – so large networks do not exhibit much longer memory than small networks. This result is consistent with analysis of more realistic networks [Banerjee, 2001a,b].

To understand why our results differed from Bertschinger and Natschläger [2004], in the sense that their network could operate at the edge of chaos while ours could not, in Chapter 3 we considered a regime in which the average number of connections per neuron is fixed as $N \to \infty$. When we did that, we found that networks could operate on the edge of chaos. Moreover, the temporal memory scaled as N, rather than $\log N$, so large networks could display very long memory. These results are consistent with the findings of Maass et al. [1997], who showed that when the connectivity in a network of spiking neurons decreased, the performance on a classification task greatly

improved.

A feature of these long temporal memory networks was that the input had to be large enough so that it dominated over the recurrent connections. This meant that most neurons were purely input driven, and a very small number was actually involved in memory. We suspected that such networks might not be robust to noise, so in Chapter 4 we considered noise in the form of synaptic failures. Indeed, we found that even a low probability of failures – much less than the order of $1/K$ – had a deleterious effect on the temporal memory, and led to the same short memory and $\log N$ scaling we saw in the high connectivity limit.

5.2 Outstanding future issues

The outstanding question for the future is: how can we build biologically plausible networks that can exhibit long temporal memory – much longer than the time constants of the constituent neurons?

We propose the following:

- Add structure in the connection matrix of the network [White et al., 2004]. This also includes broad classes of connection matrices with some underlying random elements. For example, the connection matrix could be random orthogonal matrix[White et al., 2004] or it could be a block matrix such that its blocks are correlated but elements of each block are uncorrelated (blocks with random elements).

- Add temporal correlation in the inputs. In our studies, we have assumed that the signals received by the network temporally are uncorrelated. However,

considering temporally correlated signals (e.g. speech signals), can increase the temporal memory of the network about the past inputs (see Okada [1995], Henkel and Opper [1990]). This is because the network tells us about the past inputs not only from temporal correlations in the network but also from temporal correlations in the inputs. So one can ask, how $\tau_{\max}$ scales with temporal correlations in the inputs.

- One drawback of the McCulloch & Pitts update rule is that the present state of a neuron depends only on the state at the previous (discrete) time. We can modify this by including a temporal kernel in the update rule, so that the present state depends on the history of the past states; that is, the state of neuron i at time $t+1$ can be written as $x_i(t+1) = \text{sign}\left(\sum_{j=1}^{N}\sum_{\tau=0}^{\infty} w_{ij}\kappa^{\tau}x_j(t-\tau) + u_i(t)\right)$, where $\kappa < 1$ (see Mozer [1993] for review). One can then ask whether κ can substantially increase the temporal memory.

- Finally, one can consider neurons with synaptic plasticity [Maass et al., 1997, 2004]. Natschläger et al. [2001] demonstrated in a specific case that the synaptic dynamics plays a critical role in the network dynamics. Whether it can increase temporal memory, however, is an open question.

Appendix A

Computing the evolution function $f(d; p, u^{(1)}, u^{(2)})$, and its derivative with respect to d

This appendix is divided into two sections. In the first, Sec.A.1, we derive an analytical form for $f(d; p, u^{(1)}, u^{(2)})$. For completeness we include synaptic failures; the "p" in $f(d; p, u^{(1)}, u^{(2)})$ is the probability of synaptic release, which in realistic networks is less than 1. In the second section, Sec.A.2, we compute the derivative of $f(d; p, u^{(1)}, u^{(2)})$ with respect to d.

A.1 The evolution function: $f(d; p, u^{(1)}, u^{(2)})$

Our starting point is Eq.2.22, extended to include failures. This equation can be written succinctly as

$$f(d;p,u^{(1)},u^{(2)}) = \int dh^{(1)}dh^{(2)}P(h^{(1)},h^{(2)};p)\Theta\Big(-(h^{(1)}+u^{(1)})(h^{(2)}+u^{(2)})\Big). \quad \text{(A.1)}$$

The probability distribution for the current, Eq.2.6 (again extended to include failures), is given by

$$P(h^{(1)},h^{(2)};p) = \frac{1}{2\pi\sqrt{\det\Sigma}}e^{-\frac{1}{2}\mathbf{H}\Sigma^{-1}\mathbf{H}^{\top}} \quad \text{(A.2)}$$

where $\mathbf{H} = (h^{(1)},h^{(2)})$ and Σ is given by Eq.4.3 (or Eq.2.10 if there are no failures). It is convenient to make the change of variables

$$\mathbf{H} = \sqrt{2}(a\mathbf{v}_a + b\mathbf{v}_b), \quad \text{(A.3)}$$

where $\mathbf{v}_a \equiv \frac{1}{\sqrt{2}}(1,1)$ and $\mathbf{v}_b \equiv \frac{1}{\sqrt{2}}(1,-1)$ are the eigenvectors of the covariance matrix, Σ. The corresponding eigenvalues, λ_a and λ_b are given by

$$\lambda_a = p\sigma^2(1+qp), \quad \lambda_b = p\sigma^2(1-qp) \quad \text{(A.4)}$$

where, recall, $q = 1-2d$. This transformation implies that

$$\begin{aligned} h^{(1)} &= a+b \\ h^{(2)} &= a-b. \end{aligned} \quad \text{(A.5)}$$

Applying this change of variables to Eq.A.2, we have

$$P(a,b;p) = \frac{1}{2\pi\sqrt{\lambda_a\lambda_b}}e^{-\frac{a^2}{\lambda_a}}e^{-\frac{b^2}{\lambda_b}}. \quad \text{(A.6)}$$

Because we are transforming from $(h^{(1)}, h^{(2)})$ to (a, b) in Eq.A.1, $dh^{(1)}dh^{(2)}$ transforms $|\det J| dadb$ where J, the Jacobian, is given by (see Eq.A.5)

$$J = \begin{pmatrix} \frac{\partial h^{(1)}}{\partial a} & \frac{\partial h^{(1)}}{\partial b} \\ \frac{\partial h^{(2)}}{\partial a} & \frac{\partial h^{(2)}}{\partial b} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{A.7}$$

Consequently, $|\det J| = 2$.

To determine the region of (a, b) space to integrate over, we note that the condition $(h^{(1)} + u^{(1)})(h^{(2)} + u^{(2)}) \leq 0$ (Eq.A.1) implies that $(a + b + u^{(1)})(a - b + u^{(2)}) \leq 0$. This region corresponds to the blue quadrants in Fig.A.1. These quadrants meet at the point (a_c, b_c), where

$$a_c = -\frac{u^{(1)} + u^{(2)}}{2}, \qquad b_c = -\frac{u^{(1)} - u^{(2)}}{2}. \tag{A.8}$$

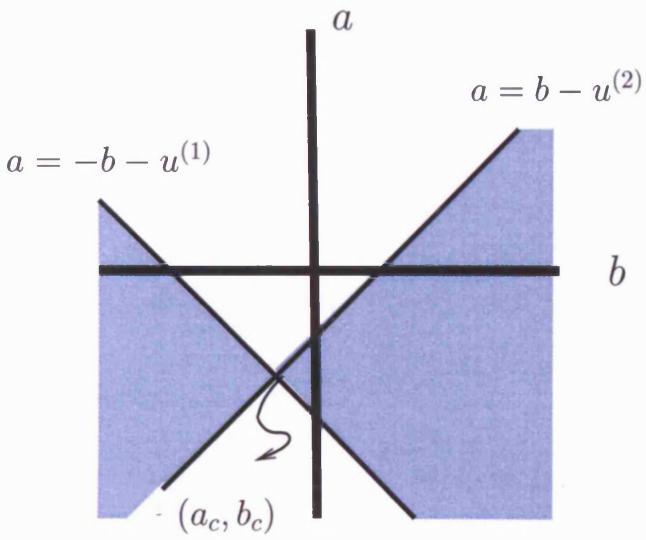


Figure A-1: The function that determines how $d(t)$ evolves in time, $f(d(t); u, u)$, is a Gaussian integral over the blue quadrants.

Combining the fact that $|\det J| = 2$ with the region shown in Fig.A.1, we have

$$f(d;p,u^{(1)},u^{(2)}) = \int_{-\infty}^{\infty} \frac{db}{\sqrt{\pi\lambda_b}} e^{-\frac{b^2}{\lambda_b}} \int_{-|b-b_c|+a_c}^{|b-b_c|+a_c} \frac{da}{\sqrt{\pi\lambda_a}} e^{-\frac{a^2}{\lambda_a}}. \tag{A.9}$$

It is convenient to make the change of variables

$$\rho = \left(\frac{\lambda_b}{\lambda_a}\right)^{1/2}, \quad \hat{b} = \frac{b_c}{\sqrt{\lambda_b}}, \quad \hat{a} = \frac{a_c}{\sqrt{\lambda_a}}, \quad z = \frac{b}{\sqrt{\lambda_b}}, \quad y = \frac{a}{\sqrt{\lambda_a}}, \quad \mathcal{Z} = \rho|z-\hat{b}|,$$

which allows us to simplify Eq.A.9 to

$$f(d;p,u^{(1)},u^{(2)}) = \frac{1}{\pi} \int_{-\infty}^{\infty} dz\, e^{-z^2} \int_{-\mathcal{Z}+\hat{a}}^{\mathcal{Z}+\hat{a}} dy\, e^{-y^2}. \tag{A.10}$$

Using the error function,

$$\mathrm{erf}(z) \equiv \frac{2}{\sqrt{\pi}} \int_0^z dt\, e^{-z^2}, \tag{A.11}$$

Eq.A.10 becomes

$$f(d;p,u^{(1)},u^{(2)}) = \frac{1}{2\sqrt{\pi}} \int_{-\infty}^{\infty} dz e^{-z^2} \left(\mathrm{erf}\left(\mathcal{Z}-\hat{a}\right) + \mathrm{erf}\left(\mathcal{Z}+\hat{a}\right)\right). \tag{A.12}$$

And when $u^{(1)} = u^{(2)} = u$, Eq.A.12 reduces to

$$f(d;p,u,u) = \frac{1}{2\sqrt{\pi}} \int_{-\infty}^{\infty} dz e^{-z^2} \left[\mathrm{erf}\left(|z|\sqrt{\frac{1-qp}{1+qp}} - \bar{u}_p\right) + \mathrm{erf}\left(|z|\sqrt{\frac{1-qp}{1+qp}} + \bar{u}_p\right]\right), \tag{A.13}$$

where $\hat{u}_p \equiv \frac{u}{\sigma\sqrt{(1+qp)p}}$.

For deterministic networks, where $p = 1$, we let $f(d;p,u,u) \rightarrow f(d;u,u)$, and we have

$$f(d;u,u) = \frac{1}{2\sqrt{\pi}} \int_{-\infty}^{\infty} dz e^{-z^2} \left[\text{erf}\left(|z| \sqrt{\frac{d}{1-d}} - \hat{u} \right) + \text{erf}\left(|z| \sqrt{\frac{d}{1-d}} + \hat{u} \right) \right], \tag{A.14}$$

where, consistent with the above definition of $\hat{u}_p$, $\hat{u}_1 = \frac{u}{\sigma\sqrt{1+q}} = \frac{u}{\sigma\sqrt{2(1-d)}}$.

A.2 Computing $\frac{\partial f(d;p,u^{(1)},u^{(2)})}{\partial d}$

To compute the derivative of the evolution function, $f(d;p,u^{(1)},u^{(2)})$, with respect to the Hamming distance, d (see Eqs.4.9 and 4.8) we start by proving the following identity:

$$\frac{\partial P(h^{(1)},h^{(2)};p)}{\partial d} = -p^2\sigma^2 \frac{\partial^2 P(h^{(1)},h^{(2)};p)}{\partial h^{(1)} \partial h^{(2)}} \tag{A.15}$$

where

$$P(h^{(1)},h^{(2)};p) = \frac{1}{2\pi\sqrt{\det \Sigma}} e^{-\frac{1}{2}\mathbf{H}\boldsymbol{\Sigma}^{-1}\mathbf{H}^{\top}}, \tag{A.16}$$

and Σ is defined in Eq.4.3.

The significance of this identity, which we demonstrate below, is the following. Using Eq.A.1, we can write

$$\frac{\partial f(d;p,u^{(1)},u^{(2)})}{\partial d} = -p^2\sigma^2 \int dh^{(1)}dh^{(2)} \frac{\partial^2 P(h^{(1)},h^{(1)};p)}{\partial h^{(1)}\partial h^{(2)}} \Theta(-(h^{(1)}+u^{(1)})(h^{(2)}+u^{(2)}))\,. \tag{A.17}$$

This follows because the only d-dependence on the right hand side is in $P(h^{(1)},h^{(1)};p)$, via its covariance matrix. Integrating Eq.A.17 by parts then leads to

$$\frac{\partial f(d;p,u^{(1)},u^{(2)})}{\partial d} = -p^2\sigma^2 \int dh^{(1)}dh^{(2)} P(h^{(1)},h^{(1)};p) \frac{\partial^2 \Theta(-(h^{(1)}+u^{(1)})(h^{(2)}+u^{(2)}))}{\partial h^{(1)}\partial h^{(2)}}\,. \tag{A.18}$$

It is easy to see that

$$\frac{\partial \Theta(-(h^{(1)}+u^{(1)})(h^{(2)}+u^{(2)}))}{\partial h^{(1)}} = \delta(h^{(1)}+u^{(1)})[-\Theta(h^{(2)}+u^{(2)}) + \Theta(-(h^{(2)}+u^{(2)}))]\,, \tag{A.19}$$

from which it follows that

$$\frac{\partial^2 \Theta(-(h^{(1)}+u^{(1)})(h^{(2)}+u^{(2)}))}{\partial h^{(1)}\partial h^{(2)}} = -2\delta(h^{(1)}+u^{(1)})\delta(h^{(2)}+u^{(2)})\,. \tag{A.20}$$

Inserting this expression into Eq.A.18, we then have

$$\frac{\partial f(d;p,u^{(1)},u^{(2)})}{\partial d} = 2p^2\sigma^2 P(-u^{(1)},-u^{(2)};p). \tag{A.21}$$

To obtain an explicit expression for the right hand side of Eq.A.21, we use Eq.A.16

with the inverse of the covariance matrix given by

$$\Sigma^{-1} = \frac{1}{p\sigma^2(1-q^2p^2)}) \begin{pmatrix} 1 & -qp \\ -qp & 1 \end{pmatrix}, \tag{A.22}$$

and we find that

$$\frac{\partial f(d;p,u^{(1)},u^{(2)})}{\partial d} = \frac{p}{\pi\sqrt{1-q^2p^2}} \exp\left[-\frac{(u^{(1)})^2+(u^{(2)})^2-2qpu^{(1)}u^{(2)}}{2p\sigma^2(1-q^2p^2)}\right]. \tag{A.23}$$

When $u^{(1)} = u^{(2)} = u$, this equation simplifies to

$$\frac{\partial f(d;p,u,u)}{\partial d} = \frac{p}{\pi\sqrt{1-q^2p^2}} \exp\left[-\frac{u^2}{p\sigma^2(1+qp)}\right], \tag{A.24}$$

and when $p = 1$, it simplifies even further,

$$\frac{\partial f(d;u,u)}{\partial d} = \frac{1}{\pi\sqrt{d(1-d)}} \exp\left[-\frac{u^2}{2(1-d)\sigma^2}\right] \tag{A.25}$$

where we used $q = 1 - 2d$.

We now show that Eq.A.15 is indeed correct. Our first step is to change variables from $(h^{(1)}, h^{(2)})$ to (a, b), using Eq.A.5. Doing this decouples the joint distribution, and leads to

$$P(h^{(1)}, h^{(1)}; p) = P(a;p)P(b;p), \tag{A.26}$$

where $P(a;p)$ and $P(b;p)$ are Gaussian distributions with zero mean and variances $\lambda_a/2$ and $\lambda_b/2$, respectively (see Eqs.A.4 and A.6). Moreover, it is easy to show that

$$\frac{\partial^2}{\partial h^{(1)}\partial h^{(2)}} = \frac{1}{2}\left(\frac{\partial^2}{\partial a^2} - \frac{\partial^2}{\partial b^2}\right). \tag{A.27}$$

In these new variables, Eq.A.15 becomes, after dividing both sides by $P(a;p)P(b;p)$,

$$\frac{\partial \log P(a;p)}{\partial d} + \frac{\partial \log P(b;p)}{\partial d} = -\frac{p^2\sigma^2}{2}\left(\frac{1}{P(a;p)}\frac{\partial^2 P(a;p)}{\partial a^2} - \frac{1}{P(b;p)}\frac{\partial^2 P(b;p)}{\partial b^2}\right). \tag{A.28}$$

Letting $x = a$ or b, we have

$$\frac{\partial \log P(x;p)}{\partial d} = \frac{1}{4}\left[\frac{4x^2}{\lambda_x^2} - \frac{2}{\lambda_x}\right]\frac{\partial \lambda_x}{\partial d}, \tag{A.29}$$

$$\frac{1}{P(x;p)}\frac{\partial^2 P(x;p)}{\partial x^2} = \left[\frac{4x^2}{\lambda_x^2} - \frac{2}{\lambda_x}\right]. \tag{A.30}$$

Combining these relationships with

$$\begin{aligned} \frac{\partial \lambda_a}{\partial d} &= -2p^2\sigma^2 \\ \frac{\partial \lambda_b}{\partial d} &= +2p^2\sigma^2, \end{aligned} \tag{A.31}$$

it is easy to show that the left and right hand sides of Eq.A.28 are equal. Since Eqs.A.28 and A.15 are identical, the fact that Eq..A.28 is true proves that A.15 is true.

Appendix B

A stimulus distribution for which we can compute $f(d;p)$ analytically

When u is drawn from a zero mean Gaussian distribution, $f(d;p)$, has an especially simple form. Here we derive this form for a network with failures. For definiteness we consider the case in which the inputs are the same on the two trials, meaning $u^{(1)} = u^{(2)} \equiv u$.

Our starting point is Eq.A.1 with $u^{(1)} = u^{(1)} = u$. Integrating this function over u, we have

$$f(d;p) = \int du\, dh^{(1)} dh^{(2)}\, \tilde{P}(u) P(h^{(1)}, h^{(2)}; p) \Theta\Big(-(h^{(1)} + u)(h^{(2)} + u)\Big). \qquad \text{(B.1)}$$

Here we will take u to be a Gaussian random variable with zero mean and variance σ_u^2. Making the change of variables

$$\begin{aligned} x &= h^{(1)} + u \\ y &= h^{(2)} + u\,, \end{aligned} \tag{B.2}$$

we see that x and y are zero mean Gaussian random variables with a covariance matrix, denoted Σ_{xy}, given by

$$\Sigma_{xy} = \begin{pmatrix} p\sigma^2 + \sigma_u^2 & qp^2\sigma^2 + \sigma_u^2 \\ qp^2\sigma^2 + \sigma_u^2 & p\sigma^2 + \sigma_u^2 \end{pmatrix}, \tag{B.3}$$

(see Eq.4.3). With this change of variables, Eq.B.1 becomes

$$f(d;p) = \int dxdy\, P_{xy}(x,y)\Theta(-xy)\,, \tag{B.4}$$

where $P_{x,y}(x,y) \sim \mathcal{N}(0,\Sigma_{xy})$.

We now make a second change of variables,

$$\begin{aligned} \xi &= \tfrac{x+y}{[2(p\sigma^2(1+pq)+2\sigma_u^2)]^{1/2}} \\ \eta &= \tfrac{x-y}{[2(p\sigma^2(1-pq)]^{1/2}}\,. \end{aligned} \tag{B.5}$$

The denominators in these expressions correspond to the standard deviation of $x \pm y$, so ξ and η have unit variance. Also, since both x and y have the same variance, ξ and η are uncorrelated. Finally, it is straightforward to show that $xy \propto \xi^2 - \rho^2\eta^2$ where

$$\rho^2 = \frac{p\sigma^2(1-pq)}{p\sigma^2(1+pq)+2\sigma_u^2}\,. \tag{B.6}$$

Thus, Eq.B.4 becomes

$$f(d) = \frac{1}{2\pi}\int d\xi d\eta\, e^{-(\xi^2+\eta^2)/2}\,\Theta(\rho^2\eta^2 - \xi^2)\,. \tag{B.7}$$

The Heaviside step function restricts this integral to the blue region shown in Fig.B-1, weighted by an isotropic Gaussian distribution. Thus, the integral in Eq.B.7 is just $4\theta_c/(2\pi)$ where $\theta_c = \tan^{-1}\rho = \sin^{-1}(\rho/(1+\rho^2)^{1/2})$ (the second equality follows from the fact that $\tan^2 x = \sin^2 x/(1-\sin^2 x)$). Using Eq.B.6 for ρ, we then have

$$f(d;p) = \frac{2}{\pi}\sin^{-1}\left[B(1-pq)/2\right]^{1/2} \tag{B.8}$$

where

$$B \equiv \frac{p\sigma^2}{p\sigma^2+\sigma_u^2}\,. \tag{B.9}$$

Finally, using $q = 1-2d$ and $p_f = 1-p$, we arrive at our final answer,

$$f(d;p) = \frac{2}{\pi}\sin^{-1}\left[B\left(\frac{p_f}{2}+d(1-p_f)\right)\right]^{1/2}\,. \tag{B.10}$$

For the deterministic case, $p_f = 0$ (and thus $p = 1$), Eq.B.10 reduces to

$$f(d;p) = \frac{2}{\pi}\sin^{-1}(Ad)^{1/2} \tag{B.11}$$

where

$$A \equiv \frac{\sigma^2}{\sigma^2+\sigma_u^2}\,. \tag{B.12}$$

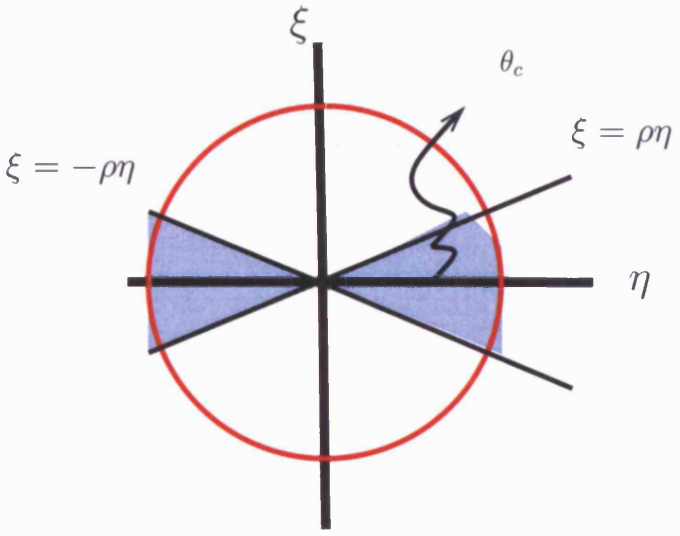


Figure B-1: Geometrical picture of $f(d;p)$ for Gaussian input. $f(d;p)$ is the area of the blue region, which corresponds to $\xi \leq \rho|\eta|$ and in fact extends to infinity, weighted by an isotropic Gaussian distribution. Because the weighting is isotropic, $f(d;p)$ is just equal to $4\theta_c/(2\pi)$ where $\theta_c = \tan^{-1}\rho$.

Appendix C

Computing $\mathrm{Var}[y_{kl}(\tau,\tau)]$

In this appendix, we compute the variance of $y_{kl}(\tau,\tau)$. The mean of this quantity is given in Eq.2.33, so we start with the second moment. Using Eq.2.31 for $y_{kl}(\tau,\tau)$, we have

$$y_{kl}^2(\tau,\tau) = \frac{1}{N^2}\sum_{i,j} J_i^{(k)}(\tau)J_j^{(k)}(\tau)x_i^{(l)}(\tau)x_j^{(l)}(\tau)\,. \tag{C.1}$$

Using the fact that $x_i^{(l)}(\tau)^2 = 1$, this equation may be written

$$y_{kl}^2(\tau,\tau) = \frac{1}{N^2}\sum_{i} J_i^{(k)}(\tau)^2 + \frac{1}{N^2}\sum_{i\neq j} J_i^{(k)}(\tau)J_j^{(k)}(\tau)x_i^{(l)}(\tau)x_j^{(l)}(\tau)\,. \tag{C.2}$$

Then, averaging over trials and subtracting $\overline{y}_{kl}(\tau,\tau)^2$ (see Eq.2.33), we arrive at the expression for the variance,

$$\begin{aligned}\mathrm{Var}[y_{kl}(\tau,\tau)] &= \frac{1}{N^2}\sum_i J_i^{(k)}(\tau)^2[1-\overline{x}_i^{(l)}(\tau)^2] \\ &+ \frac{1}{N^2}\sum_{i\neq j} J_i^{(k)}(\tau)J_j^{(k)}(\tau)[\overline{x_i^{(l)}(\tau)x_j^{(l)}(\tau)} - \overline{x}_i^{(l)}(\tau)\overline{x}_j^{(l)}(\tau)]\,.\end{aligned} \tag{C.3}$$

Since activity on different neurons is only weakly correlated ($\mathcal{O}(1/N)$ at most; see Eq.2.1), it follows that, in the large N limit, $\overline{x_i^{(l)}x_j^{(l)}} = \overline{x}_i^{(l)}\overline{x}_j^{(l)}$. Thus, the second term in Eq.C.3 vanishes, and we recover Eq.2.36: $\mathrm{Var}[y_{kl}(\tau,\tau)] = \sigma_{kl}^2(\tau)/N$ where

$$\sigma_{kl}^2(\tau) = \frac{1}{N}\sum_i J_i^{(k)}(\tau)^2[1-\overline{x}_i^{(l)}(\tau)^2]\,. \tag{C.4}$$

The first term in this expression is just $q_{kk}(\tau)$ (see Eq.2.35), and Eq.C.4 simplifies to

$$\sigma_{kl}^2(\tau) = q_{kk}(\tau) - \frac{1}{N}\sum_i J_i^{(k)}(\tau)^2\overline{x}_i^{(l)}(\tau)^2\,. \tag{C.5}$$

We now use the fact that $J_i^{(k)}(\tau) = \overline{x}_i^{(l)}(\tau)$, the second term in Eq.C.5 becomes

$$\frac{1}{N}\sum_i J_i^{(k)}(\tau)^2\overline{x}_i^{(l)}(\tau)^2 = \frac{1}{R^4}\sum_{r,r',s,s'}\frac{1}{N}\sum_i x_i^{(k)r}x_i^{(k)r'}x_i^{(l)s}x_i^{(l)s'} \tag{C.6}$$

where r, r', s and s' range from 1 to R, with R the number of trials. As in Eq.2.34, the right hand side is an average over trajectories with different initial conditions. And, as discussed immediately after that equation, the right hand side is self-averaging. Thus, ignoring terms for which $r = r'$ or $s = s'$ (which is valid in the large R limit), we can drop the sum over r, r', s and s'. Doing this, and incorporating the expression into Eq.C.5, we arrive at our final expression,

$$\sigma_{kl}^2(\tau) = q_{kk}(\tau) - \frac{1}{N}\sum_i x_i^{(k)1} x_i^{(k)2} x_i^{(l)3} x_i^{(l)4} . \tag{C.7}$$

The second term can be written as a four-dimensional Gaussian integral among correlated variables (analogous to the two-dimensional Gaussian integral in, for example, Eq.2.13). However, it is not easily reducible to a form that can be computed numerically. Instead, we compute it by Monte-Carlo sampling of the correlated Gaussian. This computation is used in Chapter 2.

Appendix D

$P(\text{diff}|C; K, u, p)$ in the limit $C/K \ll 1$

The quantity $P(\text{diff}|C; K, u, p)$ which is defined in Eq.3.3, is the probability that a neuron will have different activity on two different trials when C out of K inputs are different on the two trials, and the neuron receives input u (see Fig.3-1). Consequently, $P(\text{diff}|C; K, u, p)$ is exactly equal to $f(C/K; p, u, u)$. Our goal in this appendix is to compute this quantity in the limit that both $C/K \ll 1$ and p_f are small. Our starting point is Eq.A.9 with $u^{(1)} = u^{(2)} = u$,

$$P(\text{diff}|C; K, u, p) = \frac{1}{\pi p\sigma^2} \frac{1}{(1-q^2p^2)^{1/2}} \int_{-\infty}^{\infty} db\, e^{-b^2/(p\sigma^2(1-qp))} \int_{-|b|+u}^{|b|+u} da\, e^{-a^2/(p\sigma^2(1+qp))} \tag{D.1}$$

where $q = 1 - 2C/K$, we used the fact that when $u^{(1)} = u^{(2)} = u$, $b_c = 0$ and $a_c = u$ (Eq.A.8), and we used the explicit form for λ_a and λ_b (Eq.A.4).

Expressed in terms of C/K and p_f, the variance of b is proportional to $2(1 - p_f)C/K + p_f$. Thus, in the limit that both C/K and p_f are small, $P(b)$ is sharply peaked around 0. This allows us to replace a be u in the exponent, and the integral

over a then yields a factor of $2|b|e^{-u^2/(p\sigma^2(1+qp))}$. Thus, Eq.D.1 becomes

$$P(\text{diff}|C;K,u,p) = \frac{1}{\pi p\sigma^2}\frac{e^{-u^2/(p\sigma^2(1+qp))}}{(1-q^2p^2)^{1/2}}\int_{-\infty}^{\infty} db\, 2|b|e^{-b^2/(p\sigma^2(1-qp))}\,. \tag{D.2}$$

Performing the integral over b, we arrive at

$$P(\text{diff}|C;K,u,p) = \frac{2}{\pi}\left(\frac{1-qp}{1+qp}\right)^{1/2}\exp\left[-\frac{u^2}{p\sigma^2(1+qp)}\right]\,. \tag{D.3}$$

Using $q = 1-2C/K$ and $p = 1-p_f$, we have

$$\begin{aligned} 1-qp &= 2(1-p_f)C/K + p_f \approx 2C/K + p_f \\ 1+qp &= 2-2(1-p_f)C/K - p_f \approx 2\,. \end{aligned} \tag{D.4}$$

Finally, inserting Eq.D.4 into Eq.D.3, we arrive at our main result,

$$P(\text{diff}|C;K,u,p) \approx \frac{2}{\pi}\,(C/K + p_f/2)^{1/2}\,e^{-u^2/2\sigma^2}\,. \tag{D.5}$$

Bibliography

S. Amari. A method of statistical neurodynamics. *Kybernetik.*, 14(4):201–15, 1974.

S. Amari and K. Maginu. Statistical neurodynamics of associative memory. *Neural Networks*, 1(1):63–73, 1988.

A. Banerjee. On the phase-space dynamics of systems of spiking neurons. i: Model and experiments. *Neural Computation*, 13(1):161–193, 2001a.

A. Banerjee. On the phase-space dynamics of systems of spiking neurons. ii: Formal analysis. *Neural Computation*, 13(1):195–225, 2001b.

N. Bertschinger and T. Natschläger. Real-time computation at the edge of chaos in recurrent neural networks. *Neural Computation*, 16(7):1413–1436, 2004.

V. Braitenberg. Is the cerebellar cortex a biological clock in the millisecond range? *Prog Brain Res.*, 25:334–46., 1967.

D.V. Buonomano. Timing of neural responses in cortical organotypic slices. *Proc. Natl. Acad. Sci. USA*, 100(8):4897–902., 2003.

D.V. Buonomano and M.W. Merzenich. Temporal information transformed into a spatial code by a network with realistic properties. *Science*, 267:1028–1030, 1995.

C.E. Carr. Processing of temporal information in the brain. *Annu. Rev. Neurosei.*, 16:223–43, 1993.

D. Catstillo and B. Katz. Quantal components of the end-plate potential. *J Physiol.(London)*, 124(3):560–73, 1954.

B. Derrida. Dynamical phase transition in non-symmetric spin glasses. *J. Phys. A: Math. Gen.*, 20:L721–L725, 1987.

J. P. Eckmann and D. Ruelle. Ergodic theory of chaos and strange attractors. *Rev. Mod. Phys.*, 57:617, 1985.

P. Fatt and B. Katz. An analysis of the end-plate potential recorded with an intracellular electrode. *J. Physiol.(London)*, 115(3):320–70, 1951.

P. Fatt and B. Katz. Spontaneous subthreshold activity at motor nerve endings. *J. Physiol.(London)*, 117(1):109–28., 1952.

R.H. Fitch, S. Miller, and P. Tallal. Neurobiology of speech perception. *Annual Review of Neuroscience*, 20:331–53, 1997.

L. Glass. Chaos in neural systems. *The Handbook of Brain Theory and Neural Networks, M.A. Arbib, Ed.(MIT Press, Cambridge, MA,)*, pages 186–189, 1995.

L. Glass and MC. Mackey. *From Clocks to Chaos: the Rhythms of Life.* Princeton: University Press, 1988.

L.J. Goldman and O.W. Henson. Prey recognition and selection by the constant frequency bat, pteronotus p. parnellii. *Behav. Ecol. Sociobiol.*, 2:411 419, 1977.

C. Grebogi, E. Ott, and J. A. Yorke. Chaos, strange attractors, and fractal basin boundaries in nonlinear dynamics. *Science, New Series*, 238:632, 1987.

B. Grothe and G.M. Klump. Temporal processing in sensory systems. *Curr. Opin. Neurobiol*, 10(4):467–73., 2000.

R. W. Hamming. Error detecting and error correcting codes. *Bell System Tech Journal*, 9:147–160, 1950.

R.D. Henkel and M. Opper. Distribution of internal fields and dynamics of neural networks. *Europhys. Lett.*, 11:403, 1990.

J. J. Hopfield and C. D. Brody. What is a moment? "cortical" sensory integration over a brief interval. *Proceedings of the National Academy of Sciences, USA*, 97 (25):13919–24., 2000.

J. J. Hopfield and C. D. Brody. What is a moment? transient synchrony as a collective mechanism for spatiotemporal integration. *Proceedings of the National Academy of Sciences, USA*, 98(3):1282–7., 2001.

R.B. Ivry and R.M. Spencer. The neural representation of time. *Curr. Opin. Neurobiol.*, 14:225–232, 2004.

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

H. Jaeger and H. Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(4):78–80, 2004.

B. Katz. Mechanisms of synaptic transmission. *Rev. Mod. Phys.*, 31:524–531, 1959.

A.A. Koulakov, Raghavachari S, Kepecs A, and Lisman JE. Model for a robust neural integrator. *Nature Neuroscience*, 5(8):775–82, 2002.

M. Leon and M.N. Shadlen. Representation of time by neurons in the posterior parietal cortex of the macaque. *Neuron*, 38:317–327, 2003.

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.

W. Maass, T. Natschläger, and H. Markram. On the computational power of circuits of spiking neurons. *J. of Physiology (Paris)*, page in press, 2004.

W. Maass, T. Natschläger, and N.T.R. Series. Networks of spiking neurons can emulate arbitrary hopfield nets in temporal coding. *Network: Comput. Neural Syst.*, 8:355–371., 1997.

M.D. Mauk and D.V. Buonomano. The neural basis of temporal processing. *Annual Review of Neuroscience*, 27:307–340, 2004.

W. S. McCulloch and W. H. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.*, 5:115–133, 1943.

M. C. Mozer. Neural network architectures for temporal pattern processing. *In A. S. Weigend & N. A. Gershenfeld (Eds.), Time series prediction: Forecasting the future and understanding the past.Redwood City, CA: Sante Fe Institute Studies in the Sciences of Complexity, Proceedings Volume XVII, Addison-Wesley Publishing.*, pages 243–264, 1993.

T. Natschläger, W. Maass, and A. Zador. Efficient temporal processing with biologically realistic dynamic synapses. *Network: Comput. Neural Syst.*, 12:75–87., 2001.

M. Okada. A hierarchy of macrodynamical equations for associative memory. *Neural Networks*, 8:833–838, 1995.

G. S. Pollack. Neural processing of acoustic signals. *Comparative Hearing: Insects, R. R. Hoy, A. N. Popper, and R. R. Fay, eds. Springer, Berlin.*, pages 139–196., 1998.

G.S. Pollack. Analysis of temporal patterns of communication signals. *Curr Opin Neurobiol*, 11:734–8., 2001.

H.S. Seung. How the brain keeps the eyes still. *Proc. Natl. Acad. Sci. USA*, 93: 13339–13344, 1996.

W.C. Stacey and D.M Durand. Synaptic noise improves detection of subthreshold signals in hippocampal ca1 neurons. *The Journal of Neurophysiology*, 86(3):1104–1112, 2001.

C. A. Van Vreeswijk and H. Sompolinsky. Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science*, 274:1724–1726, 1996.

M. Volgushev, I. Kudryashov, M. Chistiakova, M. Mukovski, Niesmann, J., and U.T. Eysel. Probability of transmitter release at neocortical synapses at different temperatures. *J. Neurosci.*, 92(1):212–20, 2004.

B. Walmsley, F.R. Edwards, and D.J. Tracey. The probabilistic nature of synaptic transmission at a mammalian excitatory central synapse. *J. Neurosci.*, 7(4):1037–46, 1987.

O. White, D. Lee, and H. Sompolinsky. Short term memory in orthogonal neural networks. *Physical Review Letters*, 92(14), 2004.

M.J. Zigmond, F.E. Bloom, S.C. Landis, and R.L. Squire. *Fundamental Neuroscience.* Academic Press., 1999.