Stable, oscillatory and chaotic regimes in the dynamics of re-entrant neural networks

Aldo M. Frigo

Lehigh University

Follow this and additional works at: http://preserve.lehigh.edu/etd

Recommended Citation
AUTHOR:
Frigo, Aldo M.

TITLE:
Stable, Oscillatory and Chaotic Regimes in the Dynamics of Re-Entrant Neural Networks

DATE: May 30, 1993
Stable, Oscillatory and Chaotic Regimes in the Dynamics of Re-entrant Neural Networks

by

ALDO M. FRIGO

A Thesis
Presented to the Graduate and Research Committee
of Lehigh University
in Candidacy for the Degree of
Master of Science
in
Electrical Engineering

Lehigh University
May 21th, 1993
This thesis is accepted and approved in partial fulfillment of the requirements for the Master of Science.

\[5-21-93\]
Date

Thesis Advisor

Chairperson of Department
Table of Contents

Abstract

1 Introduction
   1.1 The physiology of perception
      1.1.1 Methodological considerations
      1.1.2 Neurophysiological results
      1.1.3 Background activity
   1.2 Chaos and order in nonlinear systems
      1.2.1 Properties of strange attractors
   1.3 Evidence of chaos in biological neural systems

2 The re-entrant network paradigm
   2.1 Processing element architecture
   2.2 Network architecture

3 Networks with linear transfer function
   3.1 Solutions of homogeneous linear equations
   3.2 Relation between homogeneous linear difference and
differential equations with constant coefficients
   3.3 Phase planes of linear networks
   3.4 Observations and conclusions

4 The Hopfield network
   4.1 The Energy function
   4.2 A practical example
   4.3 Observations and conclusions

5 Small networks with nonlinear elements
   5.1 Networks with Hypertangent transfer function
   5.2 Networks with Sigmoid transfer function
6 Large networks with nonlinear elements
   6.1 Networks with Hypertangent transfer function 60
   6.2 Networks with Sigmoid transfer function 81

7 Re-entrant networks with Hebbian learning 96
   7.1 Learning applied to large networks with Hypertangent transfer function 97
   7.2 Learning applied to large networks with Sigmoid transfer function 101

8 Conclusions 103

References 104

Vita 105
List of figures

| Figure 1.1 | Single unaveraged EEG traces. Topographic contour plot. | 4 |
| Figure 1.2 | Contour plots that emerge from EEG traces. | 5 |
| Figure 1.3 | EEG traces for four classes of states | 6 |
| Figure 1.4 | Evolution of the logistic map | 9 |
| Figure 1.5 | Evolution of the logistic map | 10 |
| Figure 1.6 | Evolution of the logistic map | 10 |
| Figure 1.7 | Bifurcation diagram of the logistic map | 11 |
| Figure 1.8 | Phase plane of pendulum's dynamic | 12 |
| Figure 1.9 | EEG traces of OB and PC. Resulting phase plane | 15 |
| Figure 2.1 | Model of artificial neuron | 17 |
| Figure 2.2 | Flow chart of artificial neuron architecture | 18 |
| Figure 2.3 | Sigmoid and Hypertangent curves | 19 |
| Figure 2.4 | Architecture of re-entrant network | 21 |
| Figure 3.1 | Two neuron linear re-entrant network | 22 |
| Figure 3.2 | Phase plane of two neuron network with real roots | 29 |
| Figure 3.3 | Phase plane of two neuron network with complex roots | 31 |
| Figure 4.1 | The Hopfield network | 33 |
| Figure 4.2 | The Hopfield processing element | 33 |
| Figure 4.3 | Energy terrain contour map | 34 |
| Figure 5.1 | Single layer re-entrant network | 39 |
| Figure 5.2 | Phase plane of Network 1 | 41 |
| Figure 5.3 | Phase plane of Network 1 with two initial conditions | 42 |
| Figure 5.4 | Phase plane of Network 2 | 44 |
Figure 5.5 Phase plane of Network 2 with three initial conditions 45
Figure 5.6 Phase plane of Network 3 46
Figure 5.7 Phase plane of Network 4 47
Figure 5.8 Phase plane of Network 4 with two initial conditions 48
Figure 5.9 Phase plane of Network 5, Network Gain = 13 49
Figure 5.10 Phase plane of Network 5, Network Gain = 14 49
Figure 5.11 Phase plane of Network 5, Network Gain = 15 50
Figure 5.12 Phase plane of Network 5, Network Gain = 16 50
Figure 5.13 Phase plane of Network 5, Network Gain = 17 51
Figure 5.14 Phase planes of Network 5, Network Gain = 16 52
Figure 5.15 Phase planes of Network 5 with two initial conditions 53
Figure 5.16 Phase plane of Network 7 54
Figure 5.17 Trajectories of Network 6, two initial conditions 56
Figure 5.18 Pase plane of Network 7 57
Figure 6.1 Phase plane generated by Simulation 1, NG = 3.5, TFG = 1.0 61
Figure 6.2 Phase plane generated by Simulation 2, NG = 3.8, TFG = 1.0 62
Figure 6.3 Phase plane generated by Simulation 2, NG = 3.8, TFG = 1.0 63
Figure 6.4 Phase plane generated by Simulation 2, different input vector 64
Figure 6.5 Same, time domain 65
Figure 6.6 Same, frequency domain 65
Figure 6.7 Same, phase spectrum 66
Figure 6.8 Output of a neuron generated by Simulation 2 67
Figure 6.9 Frequency spectrum 67
Figure 6.10 Phase spectrum 68
Figure 6.11 Phase plane generated by Simulation 3, NG = 5.0, TFG = 1.0 69
Figure 6.12  Phase planes generated by Simulation 3, NG = 5.0, TFG = 1.0
Figure 6.13  Output of a neuron in Simulation 3, time domain
Figure 6.14  Frequency spectrum
Figure 6.15  Phase plane generated by two different input vectors
Figure 6.16  Phase planes generated by Simulation 4, NG = 3.8, TFG = 1.5
Figure 6.17  Output of a neuron in Simulation 4, time domain
Figure 6.18  Phase planes generated by Simulation 5, NG = 3.9, TFG = 1.7
Figure 6.19  Phase planes generated by Simulation 5, NG = 3.9, TFG = 1.7
Figure 6.20  Output of a neuron in Simulation 5, time domain
Figure 6.21  Frequency spectrum
Figure 6.22  Two overlapped phase planes generated with two input vectors
Figure 6.23  Phase planes generated by Simulation 6, NG = 2.2, TFG = 1.7
Figure 6.24  Phase planes generated by Simulation 6, NG = 2.2, TFG = 1.7
Figure 6.25  Output of a single neuron in time domain
Figure 6.26  Output of a single neuron in time domain
Figure 6.27  Output of a single neuron in time domain
Figure 6.28  Phase plane generated by Simulation 4, NG = 4.4, TFG = 10.0
Figure 6.29  Phase planes generated by Simulation 4, NG = 4.4, TFG = 10.0
Figure 6.30  Output of a single neuron in time domain, frequency spectrum
Figure 6.31  Frequency spectrum
Figure 6.32  Phase spectrum
Figure 6.33  Output of a single neuron in Simulation 5, time domain
Figure 6.34  Phase plane generated by Simulation 6
Figure 6.35  Phase planes generated by Simulation 6
Figure 6.36  Phase plane generated by Simulation 6 with two input vectors
Figure 6.37  Output of a single neuron in Simulation 7, time domain  93
Figure 6.38  Output of a single neuron in Simulation 8, time domain  94
Figure 7.1   Phase plane of three limit cycles after learning       99
Figure 7.2   Phase plane of three limit cycles after learning       100
Figure 7.3   Single neurons discriminates between input vectors   102
List of tables

Table 6.1  Summary of simulation results in section 6.1  81
Table 6.2  Summary of simulation results in section 6.2  95
Table 6.3  Three simulation sequences  95
Abstract

The purpose of this Thesis is to investigate whether dynamic behavior of re-entrant artificial neural networks with initial random weight distributions can be used to recognize patterns. The motivation arises not so much from trying to design new architectures, even though such architectures may result from this effort, but to better understand the pattern recognition mechanism that takes place in the brain. In particular the work is inspired by investigations by Prof. W. Freeman at U.C. Berkeley of odor perception in the olfactory bulb of rabbits. Here specific odors cause specific dynamic patterns of excitation that can be observed on the surface of the bulb.

My reentrant networks have a similarity to Hopfield networks, but the weight matrix is deliberately non-symmetric and random. Using the difference equation approach, stable, oscillatory (periodic or quasi-periodic) and chaotic regimes, similar to those observed in the brain, are generated by our model.

Dependence to bifurcation parameters and initial conditions is also investigated. The results of our computer simulations show that our model is sensitive to the smallest changes in bifurcation parameters. Sensitivity to initial conditions (input pattern) vary with the dynamic regime of the system. The network shows almost no sensitivity to initial conditions until chaos emerges. This suggests a possible role of chaos in biological pattern recognition as a "memory search". Our simulations show that near the threshold of chaos different input vectors can lead to different attractors when Hebbian learning is applied.
1 Introduction

The superior performance of biological systems in pattern recognition, and its modeling, is today the subject of intense research world wide. In the past few years some results of Neuroscience have received great attention in the engineering community because of potential applications in the design of massively parallel computer architecture. Although theories, ideas and expectations abound, the most fundamental questions regarding biological computation still have to be answered.

Within a few milliseconds a stimulus coming from sensory receptors is accurately recognized and distinguished from among millions of others. What is the underlying biological structure that makes this possible? Which are the fundamental mechanisms of memory? How are neural signals addressed in memory? Up to today we can only vaguely answer these and other questions.

Recent research has led to the discovery in the brain of chaos, a behavior of complex nonlinear systems that seems random but actually has some hidden order. Chaos is believed to be the very property that makes perception, and possibly other brain functions, possible.

1.1 The physiology of Perception: Freeman's work

One of the people who did the most interesting research on the physiology of perception, and whose work, among others, inspired this thesis, is Professor Walter Freeman of the University of California at Berkeley. Freeman and his group have concentrated for the last thirty years on the study of perception, in particular on the study of the olfactory system of rabbits [1] [2].
According to Freeman perception cannot be understood solely by examining properties of individual neurons, it is instead the result of the cooperative activity of millions of neurons. Freeman also came to the conclusion that chaos plays a crucial role in the olfactory bulb of rabbits, a system that has to react and adapt to an environment subject to unpredictable and violent fluctuations. He hypothesizes that chaotic behavior serves as a "ground state" for the perceptual apparatus.

1.1.1 Methodological considerations

The olfactory bulb of rabbits was studied by analyzing the spatial and time dimensions of EEG (Electroencephalogram) activity near the cortical surface. Each rabbit had a two dimensional array of 64 electrodes implanted permanently in the left olfactory bulb. The 64 EEG traces were amplified, filtered and measured in brief time epochs. The rabbits were then trained to recognize some particular odors.

The olfactory bulb was chosen because it is the simplest and phylogenetically the most stable sensory system, it is also the best understood in its structure and function. The rabbit was chosen because its head is sufficiently large to accommodate all the electrodes and at the same time the bulb is small enough so that the electrodes cover a sufficient portion of its area (about 20%).

1.1.2 Neurophysiological results

The EEG traces showed that the odorant-specific information was found to exist in the spatial patterns of the amplitude of the oscillations (see Fig. 1.1). These patterns were extended to all 64 channels and, by inference, to the entire bulb. All that
distinguished one odorant EEG pattern from another was the spatial configuration of
the average intensity or amplitude of the waveform. No variation in phase, frequency
or amplitude modulation was found in the traces that contained any odorant-specific
information.

Fig. 1.1 Left: a display of single unaveraged EEG traces is shown comprising a single odor
burst from one trial set. Right: the topographic contour plot of the average root mean square
value of 10 EEG recordings. The wave is nearly the same in each recording, except that the
amplitude varies. The spatial pattern of amplitude indicates the identity of the odor [3].

The plots in Fig. 1.1 were the first demonstration of the existence of sensory-specific
information in the spatial dimensions of neural activity in the cerebral cortex. The key
property is that these patterns remain constant after familiarization unless the rabbit is
exposed to a new odor, see Fig. 1.2.

In his research Freeman paid particular attention to the presence of background
activity in the brain. It is well known that some degree of activity in the cortical tissue
appears to be always present, even when the rabbit is resting for example. In other
words, there is no zero-activity "ground state". The EEG traces in Fig. 1.3 show
instead that the ground state of the neural tissue consists of a low amplitude chaotic
waveform called background activity or background noise. This chaotic behavior is extremely robust; it can only be stopped by near-lethal anesthesia or by surgical isolation of parts of the tissue. Its origin can be related to the dense local feedback of the neural tissue, as we demonstrated with our simulations.

Fig. 1.2 Left: contour plot that emerges from the EEGs of a rabbit conditioned to recognize the scent of sawdust. Middle: the rabbit has been conditioned to recognize a new smell. Right: re-exposure to sawdust led to the emergence of a new sawdust plot.

1.1.3 Background activity

Freeman proposes that the background activity has a well defined purpose in the perceptual apparatus. He suggests that the chaotic behavior of the background activity is essential in order for the animal to learn new odors and to recognize familiar ones. In other words, the background activity provides a continued readiness to respond to completely new stimuli as well as to familiar ones, without the requirement for an exhaustive sequential memory search.
Notice in Fig. 1.3 that the EEG signal is noisy during the "Waking Rest" and "Motivation" states but is not noisy in the trace in Fig. 1.1 corresponding to recognition.

In order to understand the role of chaos in neural dynamics we first have to understand the properties and the nature of chaos. The study of complex dynamic systems, chaos and non-linear phenomena in general is a very difficult task due to the inadequacy of traditional mathematical tools and to the lack of new ones.

1.2 Chaos and order in nonlinear systems

This paragraph is an introduction and a survey to order and chaos, it will deal with the many phenomena that are associated with the dynamic of nonlinear systems. Only recently was it realized that a given deterministic system, under slightly different
conditions, can display either a complex behavior that appears random or a tendency toward order.

In a famous paper, published in 1963, Lorentz [13] showed how a simple quadratic equation, called the \textit{logistic equation}, could produce such random-like behavior just by varying a parameter. This phenomenon, called chaos, is indistinguishable from random processes (like white noise) in appearance and in statistical properties, but it is deterministic and not stochastic. Chaos is also said to be low-dimensional, whereas random processes result from the addition of a large number of independent contributions.

Besides the old concepts of limit cycles and attractors a number of new concepts (strange attractors, Liapunov exponents, information entropy, self-organization) belong to a modern branch of mathematics called \textit{fractal geometry} and have been introduced in order to describe and study nonlinear phenomena. The main difficulty is that it is not possible to find analytical solutions to most Hamiltonian systems. The only exceptions are:

1. All systems with only one degree of freedom
2. All systems with linear equations of motion
3. All nonlinear systems that can be separated into uncoupled one-dimensional systems

Clearly most natural phenomena are inherently nonlinear and coupled, this is also valid for biological neural networks.

One dimensional logistic maps provide a good starting point to study and observe the many characteristics of nonlinear systems because of their relative simplicity. In particular, mapping the logistic equation clearly illustrates the mechanisms of
bifurcation of solutions and the folding and stretching required for chaos in a limited phase space.

The logistic equation is given by the following difference equation

\[ f_\mu(x) = x_{n+1} = \mu x_n (1-x_n) \]  \hspace{1cm} (1.1)

where

\[ x_n \in [0,1] \]
\[ x_0 = 0.2 \]

and \( \mu \) is a variable parameter.

The plot in Figure 1.4 shows the evolution of the logistic map for \( \mu = 2 \), the map has three parts: the parabolic curve \( y = \mu x (1-x) \), the diagonal line \( x_{n+1} = x_n \) and a set of lines connecting the successive iterations of the map. The first point, \( x_1 \) is found where the line \( x_0 = 0.2 \) meets the quadratic curve. The next point is found by moving laterally to the \( x_{n+1} = x_n \) diagonal the process is repeated until \( x \) settles to a steady state where \( x_{n+1} = x_n \). The initial condition for the system is said to converge in this case to a point attractor.

Figure 1.5 plots the logistic map for \( \mu = 3.3 \), as one can see the system oscillates indefinitely between two points. It is said that a bifurcation has taken place, the attractor in this case is called a limit cycle. If the parameter \( \mu \) is increased to 3.9 the behavior of the map becomes chaotic, as shown in Figure 1.6. The system is said to be governed by a strange attractor or chaotic attractor. Attractors are commonly visualized in two or three dimensional phase planes, as we will see, their characteristics don't change with the number of dimensions of the system.

Figure 1.7 shows the bifurcation diagram of the logistic map. The long term values of \( f_\mu(x) \) are plotted in the y-axis for 3 < \( \mu < 4 \). An important observation can be
readily made from this figure. A dynamic nonlinear system can change instantaneously from stable to oscillatory to chaotic behavior by "tuning" a single parameter. These changes are usually referred to as phase transitions or bifurcations.

In the study of Hamiltonian systems a standard method, which we also adopted, is to consider the orbit of the system as projected on a two-dimensional subspace. These systems can be classified into two categories: conservative and dissipative. All systems in which energy is conserved also conserve the volume in phase space. This volume is defined as the area of the hyper surface enclosed by the orbit. In 3 dimensions the 2 dimensional hyper space is a true surface with true area. In more than 3 dimensions a hyper space more appropriately is said to have "volume". The conservation of volume is known as the Liouville theorem.

Fig. 1.4 Evolution of the logistic map for $\mu = 2$. The equilibrium value is $x = 0.5$
Fig. 1.5 The logistic map for $\mu = 3.3$ showing an oscillation between $x = 0.48$ and $x = 0.83$.

Fig. 1.6 Iteration of the logistic map for a chaotic state at $\mu = 3.9$. 
In dissipative systems on the other hand the volume of phase space is no longer conserved, it may shrink to a point or reduce its dimensionality. For example in a three dimensional dissipative system the orbit may be converge to a plane (thus reducing the dimensionality from three to two) and eventually to the origin.

Besides the Logistic Equation there are many other examples of very simple systems that exhibit nonlinear behavior and become chaotic if the nonlinearity is sufficiently strong. The driven pendulum as described by the equation

$$\frac{d^2 \theta}{dt^2} + \omega_0^2 \sin \theta = g \cos \omega_D$$

$$\omega_0 = \sqrt{\frac{g}{l}}$$

(1.2)
is one of them. For small amplitudes we can approximate \( \sin \theta \approx \theta \) and the analytical solution is

\[
\theta = \theta_0 \cos(\omega_0 t + \phi)
\]

however for larger amplitudes no explicit solutions exist. In this case a great deal of insight and qualitative understanding can be obtained from the phase portrait of velocity versus angle (or position).

![Phase plane of pendulum's dynamics](image)

Fig. 1.8 Phase plane of pendulum's dynamics

However, the orbit describing motion in the phase space for the free oscillator with a damping factor:

\[
\frac{d^2 \theta}{dt^2} + \gamma \frac{d\theta}{dt} + \omega_0^2 \sin \theta = 0
\]  

(1.3)
converges to a fixed point, in this case the origin. The volume in phase space is said to contract and its dimensionality is reduced.

1.2.1 Properties of strange attractors

We have seen that nonlinear systems are governed, in their dynamical evolution, by attractors. These can be point attractors, for stable systems; oscillatory attractors, also known as limit cycles; and finally strange attractors, or chaotic attractors. In all these cases we can talk about basins of attraction, regions of phase space where all trajectories "fall" asymptotically towards the attractor.

A key property of attractors is that, within the basin of attraction, solutions always converge to the same point or limit cycle independently upon the initial condition. On the other hand, nonperiodic chaotic solutions are extremely sensitive to initial conditions. This sensitivity is known to be exponential and it can be characterized in a qualitative way by the Liapunov exponent $\lambda(x_0)$. Consider the chaotic motion generated by the one dimensional map

$$x_{n+1} = f(x_n)$$

(1.4)

and consider two neighboring initial conditions, $x_0$ and $x_0 + \varepsilon$. The Liapunov exponent measures the exponential separation of solutions after $N$ iterations, it follows that

$$\varepsilon e^{\lambda(x_0)} = |f^N(x_0 + \varepsilon) - f^N(x_0)|$$

(1.5)
At the limits $\varepsilon \to 0$ and $N \to \infty$

$$\lambda(x_0) = \lim \frac{\left| f^N(x_0 + \varepsilon) - f^N(x_0) \right|}{\varepsilon} = \lim \frac{1}{N} \log \left| \frac{df^N(x_0)}{dx_0} \right|$$

(1.6)

1.3 Evidence of chaos in biological neural systems

Cortical tissue can certainly be considered a complex and nonlinear dynamic system. It is a complex system because of the large density of neurons and of interconnections among them (between one and two-thousand synapses for each neuron). It is nonlinear because of the sigmoidal response of axonal firing to input potential in each neuron.

It is important to emphasize that cortical tissue is characterized by massive local feedback connectivity. The dynamics that such a structure can exhibit are comparable with those of the Logistic Equation. It is not surprising then that evidence of chaos, such as low dimensional strange attractors and bifurcation events (very similar in principle to those observed in the Logistic Map), has been found in EEG traces of humans [6] [9] and rabbits [7] [8].

A steady non oscillating state can only be observed under near lethal anesthesia, coma or brain death. As it was already mentioned, the olfactory bulb and the prepyriform cortex cannot stay at equilibrium under any normal physiological condition, their ground state is represented by ceaseless chaotic oscillations called background activity.

A bifurcation takes place when the system undergoes a major transition in its dynamics equivalent to, for example, the transition from sleep to wake. Other
bifurcations take place in the transitions from waking rest to motivation activity and from motivation to seizure. The system and its governing equations are the same, but the solutions change radically. In the olfactory bulb these phase transitions or bifurcations are directly related to the breathing cycle and to an arousal state in connection with stimuli coming from the brain stem and from long-range feedback connections to other parts of the brain such as the prepyriform cortex (PC) and the anterior olfactory nucleus (AON).

We cannot prove mathematically that this brain activity is chaotic, on the other hand there are a number of observations that strongly indicate that this is indeed the case. Figure (1.9) shows the EEG traces of the olfactory bulb and of the prepyriform cortex on the left and the resulting phase plane on the right side. As one can see the phase plane is not random, but reveals the presence of an attractor.

It was also discovered that no matter how noisy the EEG traces of the olfactory bulb might look like, the instantaneous frequency is always the same everywhere on the surface. Freeman also attempted a calculation of the attractors dimensionality, which seems to be quite low (between 4 and 7) indicating the presence of deterministic chaos.

Fig. 1.8 Left: EEG traces of OB and PC. Right: the resulting phase plane reveals the presence of a strange attractor.
2 The re-entrant network paradigm

We investigated the spatio-temporal neurodynamics of pattern recognition in biological systems with the aid of computer simulations. The Artificial Neural Network model we designed for this purpose may be considered a nonsymmetric variant of the paradigm used by Hopfield [10]. With this model we wanted to analyze, numerically and analytically, the correlation between spatial patterns of cortical activity and the pattern recognition ability of biological brains, and whether the same strategy may be used by artificial neural networks. Our goals can be summarized in the following points:

1. Obtain qualitative and possibly quantitative insight into the mechanisms and capabilities of this particular paradigm.
2. Determine new design principles that model the superior performance of biological systems in pattern recognition.
3. Artificially reproduce the stable, oscillatory and chaotic regimes observed in the olfactory bulb. Analyze their dynamic behavior under different design parameters.
4. Generate the patterns of excitation that were observed in the EEG arrays.
5. Investigate the role of chaos as a learning and memory-search mechanism.

2.1 Processing element architecture

All neurons in the artificial neural network model share the same architecture based on the well known McCulloch and Pitts model (see Fig. 2.1). Neurons, or Processing Elements, also share the same parameters, with the exception of the Bias. Figure 2.2
shows the flow chart of the software used to model a single neuron. As one can see we designed the software to allow for a large degree of variability and versatility.

There are five steps involved in the processing of a neuron.

STEP 1. SUMMATION FUNCTION: The first is the computation of the weighted sum of the inputs. The synaptic strengths, or weights $w$, can be either positive (excitatory) or negative (inhibitory). From Figure 2.1, the Sum is calculated as

$$\text{Sum} = \sum_{k=0}^{m} w_k x_k$$  \hspace{1cm} (2.1)
STEP 2. TRANSFER FUNCTION: In this second step a function of the weighted sum is calculated. The transfer function can be Linear, Sigmoid or Hypertangent.

Linear: \[ T = I \] (2.2a)

Sigmoid: \[ T = \left(1 + e^{-I \cdot \text{Gain}}\right)^{-1} \] (2.2b)

Hyperbolic Tangent: \[ T = \frac{e^I - e^{-I}}{e^I + e^{-I}} \] (2.2c)

\[ I' = I \cdot \text{Gain} \] (2.2d)

where:

- \( I \) current Sum
- \( T \) Transfer Function
- \( \text{Gain} \) parameter that changes the steepness of the function
STEP 3. SCALING: After the transfer function the result is scaled with a simple linear transformation of the form

\[
\text{SCALED OUTPUT} = \text{SCALE FACTOR} \cdot T + \text{OFFSET} \quad (2.3)
\]

STEP 4. OUTPUT FUNCTION: We used the direct output function

\[
\text{OUTPUT} = \text{SCALED OUTPUT} \quad (2.4)
\]

STEP 5. LEARNING RULE: Variable synaptic weights can be modified according to a learning rule. Several Hebbian learning rules have been implemented in the program and more can easily be added to the code.

2.2 Network architecture

Network architecture deals with the connectivity of the processing elements, or neurons, and with global parameters and strategies of the network. Our neural model and some of the connectionist models, such as the Hopfield network [10] and the ART family of networks [11], converge in several respects. Both rely on parallel, distributed
processing among fully interconnected processing elements in interacting networks to produce behavior. Both also rely on self-organization and on organized feedback among processing elements within the system. For these reasons our network architecture has some similarities with the Hopfield net, as it will be discussed in the next chapter.

There are, however, some important dissimilarities. Our model lacks the symmetry of the Hopfield paradigm and furthermore its dynamic is controlled by global bifurcation parameters. Global bifurcation parameters are variable parameters that model the thalamo-cortical interactions, those responsible for the state of arousal of the animal. There are two of these variables in our program; one is called the Network Gain, the other is the Transfer Function Gain. The Network Gain defines the scale factor (Eq. 2.3) of each neuron in the network, it models the well known increase in amplitude of axonal excitation under characteristic stimuli coming from the thalamus and brain stem [3]. The Transfer Function Gain models the change in sigmoidal response of axonal firing to dendritic potential caused by thalamo-cortical interactions. The Transfer Function Gain is the Gain variable of Eqs. 2.2b and 2.2c, all the neurons in the network share the same value.

Figure 2.3 portrays the network's architecture, excitations flow from Layer 1 (the input layer) up to the last layer and then feeds back to the input layer. The simulation program performs computation of one neuron at a time starting with the leftmost Processing Element in the first layer (PE11). The program then calculates the output of the next PE on the right and so on until the last PEs on the layer, the same process is repeated for the PEs of all the other layers. The output of the last layer becomes the new input of the input layer during the next iteration.
Fig. 2.4 Architecture of the Re-entrant Network
3 Networks with linear transfer function

Consider the linear two-neuron re-entrant network in Fig. 3.1

The outputs are:

\[ x_{n+1} = w_{11}x_n + w_{12}y_n \]
\[ y_{n+1} = w_{21}x_n + w_{22}y_n \]  

(3.1)

This is a system of *homogeneous linear difference equations with constant coefficients*, a solution (that is a function which satisfies the equation) can easily be found for each variable.

It is important to note that dynamic systems are described mathematically by differential equations, not difference equations. Nevertheless in the linear case it can be proven that every system of difference equations "samples" solutions of an underlying system of differential equations (see paragraph 3.2). It follows that a system of
difference equations really describes dynamic systems observed at discrete time intervals. For the nonlinear case no such proof exists, but workers in the field believe that the same applies here. See Sec. 3.2.

3.1 Solutions of homogeneous linear difference equations

Let us begin by introducing the $\Delta$ and the $E$ operator: since

$$\Delta y_k = y_{k+1} - y_k$$ (3.2)

we define

$$E^p y_k = y_{k+p}$$ (3.3)

and

$$\Delta = E - 1$$ (3.4)

A linear difference equation of order $n$ is then a difference equation having the form

$$a_0(k)E^ny_k + a_1(k)E^{n-1}y_k + \ldots + a_n(k)y_k = R(k)$$ (3.5)

where $a_0(k) \neq 0$. This equation can also be written as

$$\left[ a_0(k)E^n + a_1(k)E^{n-1} + \ldots + a_n(k) \right] y_k = R(k)$$ (3.6)

or simply

$$\phi(E)y_k = R(k)$$ (3.7)
where the linear operator \( \phi(E) \) is given by

\[
\phi(E) = a_0(k)E^n + a_1(k)E^{n-1} + \ldots + a_n(k) .
\]  

(3.8)

In the case of our network \( a_0(k), a_1(k), \ldots, a_n(k) \) are all constants (because the weights don't change) and \( R(k) \) is zero. This is the case of the **homogeneous linear difference equation with constant coefficients** and can be written as:

\[
(a_0E^n + a_1E^{n-1} + \ldots + a_n)y_k = 0
\]

or

\[
\phi(E)y_k = 0
\]

(3.9)

(3.10)

If we assume that \( y_k = \lambda^k \) is a solution, then upon substitution we obtain:

\[
a_0\lambda^{nk} + a_1\lambda^{n(k-1)} + \ldots + a_n\lambda^k = 0
\]

or

\[
(a_0\lambda^n + a_1\lambda^{n-1} + \ldots + a_n)\lambda^k = 0
\]

(3.11)

(3.12)

It follows that \( \lambda^k \) is a solution if \( \lambda \) is a solution of the **auxiliary equation**

\[
a_0\lambda^n + a_1\lambda^{n-1} + \ldots + a_n = 0
\]

(3.13)

Equation 3.12 can be written in **factored form** as

\[
a_0(E - \lambda_1)(E - \lambda_2)\ldots(E - \lambda_n)y_k = 0
\]

(3.14)
Solutions to 3.14 depend on the nature of the roots $\lambda_1, \lambda_2, \ldots, \lambda_n$. Two cases are possible.

Case 1. Roots are all real and distinct

In this case $\lambda_1, \lambda_2, \ldots, \lambda_n$ are all solutions of 3.9 and since constant multiples and sums of the solutions are also solutions we have the solution

$$y_k = c_1\lambda_1^k + c_2\lambda_2^k + \cdots + c_n\lambda_n^k$$ (3.15)

Case 2. Some of the roots are complex

In this case for the coefficients of 3.9 to be real the solutions must be conjugate complex numbers. This means that $(\alpha + \beta i)$ and $(\alpha - \beta i)$ are solutions, then

$$y_k = K_1(\alpha + \beta i) + K_2(\alpha - \beta i)$$ (3.15)

is also a solution. In real form

$$y_k = \rho^k (c_1 \cos k\theta + i \sin k\theta)$$ (3.16)

with

$$K_1 = \frac{1}{2}(c_1 + ic_2)$$

$$K_2 = \frac{1}{2}(c_1 - ic_2)$$ (3.17)

We can now return to the linear network we started from and rewrite Eq. 3.1 as
\[ \begin{align*}
Ex_n &= x_{n+1} = Ax_n + By_n \\
Ey_n &= y_{n+1} = Cx_n + Dy_n
\end{align*} \tag{3.18} \]

solve for \( x \) and \( y \)

\[ \begin{align*}
[E - A]x_n - By_n &= 0 \\
[E - D]y_n - Cx_n &= 0 \\
[E - D][E - A]x_n - B[E - D]y_n &= 0 \\
[E^2 - (A + D)E + AD]x_n - BCx_n &= 0 \\
[E^2 - (A + D)E + (AD - BC)]x_n &= 0
\end{align*} \]

it follows that

\[ \lambda_{1,2} = \frac{(A + D) \pm \sqrt{(A + D)^2 - 4(AD - BC)}}{2} \tag{3.19} \]

3.2 Relation between homogeneous linear difference and differential equations with constant coefficients

A relation between a system of difference equations, like Eq. (3.1), and its corresponding set of differential equations can be found as follows [4]:

26
\[ x_1(n+1) = b_1 x_1(n) + b_2 x_2(n) + \ldots \]
\[ x_2(n+1) = b_{21} x_1(n) + b_{22} x_2(n) + \ldots \]

or

\[ x(n+1) = B x(n) \]

the solutions are

\[ x(n) = \Phi(n,n_0) x(n_0) \]
\[ x(t) = \Phi(t,t_0) x(t) \]

where the transition matrices are defined as

\[ \Phi(n,n_0) = B^{(n-n_0)} \]
\[ \Phi(t,t_0) = e^{A(t-t_0)} \]

It follows that in order for the two sets of equations to describe the same system the A and B matrices must satisfy the following equality

\[ B = e^{A \Delta t} \]  

or, in our case

\[ B = e^A = I + A + \frac{1}{2!} A^2 + \frac{1}{3!} A^3 + \ldots \]
3.3 Phase planes of linear networks

In the following two examples appropriate values for the weights have been chosen to construct two neuron networks that have either real or complex solutions to Eq. 3.19.

Case 1. Network with real roots

The first plot, Fig. 3.2, portrays the phase plane of a network having real solutions. This phase plane shows a network that oscillates between positive and negative values, the network has the following synaptic weight strengths:

$$w_{11} = -0.095287$$
$$w_{12} = -0.097585$$
$$w_{21} = -0.080838$$
$$w_{22} = 0.084081$$

and the following parameters:

$$Gain = 7.7$$
$$x_0 = 0.45$$
$$y_0 = -0.25$$
The plot in Fig. 3.2 has been obtained from numerically simulating the solutions of Eq. (3.1). It was also verified that the formal solution, derived from Eq. (3.15),

\[ x_k = c_1 \lambda_1^k + c_2 \lambda_2^k \]

\[ y_k = \frac{x_{k+1} - w_{11} x_k}{w_{12}} \]

with

\[ c_1 = 0.161772 \]
\[ c_2 = 0.288228 \]
\[ \lambda_1 = 0.928762 \]
\[ \lambda_2 = -1.015049 \]
yields identical results, that is identical phase plane.

Case 2. Network with complex roots

A different set of weights can be chosen to create a network whose roots to Eq. 3.18 are imaginary. The phase plane plotted in Fig. 3.3 has been generated by a two neuron network with the following parameters:

\[ \begin{align*}
    w_{11} &= 0.05 \\
    w_{12} &= 0.089443 \\
    w_{21} &= -0.089443 \\
    w_{22} &= 0.05 \\
\end{align*} \]

Gain = 10

\[ \begin{align*}
    x_0 &= 0.45 \\
    y_0 &= -0.25 \\
\end{align*} \]

The network exhibits a more complex oscillation caused by the sinusoidal factors. Like in the previous case, this phase plane can be produced by direct numerical simulation of Eq. (3.1) or by plotting the formal solution:

\[ \begin{align*}
    x_k &= \rho^k (c_1 \cos k\theta + c_2 \sin k\theta) \\
    y_k &= \frac{x_{k+1} - w_{11} x_k}{w_{12}} \\
\end{align*} \]

with
\[ \rho = 1.024698 \]
\[ \theta = 1.061058 \]
\[ c_1 = 0.45 \]
\[ c_2 = -0.25 \]

Fig. 3.3 Phase plane of a two-neuron network with complex roots

Notice that the temporal sequence of points in Fig. 3.3 conforms to a single expanding logarithmic spiral being "sampled" six times per revolution.
3.4 Observations and conclusions

The study of re-entrant networks with linear transfer functions proved to be an important and necessary first step in the design and study of more complex networks. We showed that even the simple two-neuron networks can exhibit diverse oscillatory modes, depending solely on the nature of the weight matrices.

We also showed that the linear network paradigm portrays an underlying dynamic system. Unfortunately in the nonlinear case, when the transfer function is a sigmoid, no such proof exists because of the complex nature of non-linear algebra. Nevertheless we hypothesize that even non-linear systems of difference equations describe a dynamic system.

As we know, real biological neurons are not linear. Our next step will then be to study and experiment with re-entrant networks having non-linear transfer functions. Chapter 4 highlights the major attributes of the Hopfield network. This particular paradigm has many similarities with the one we intend to develop. Specifically, it is re-entrant and non-linear, it is therefore interesting to study it.
4. The Hopfield neural network

The network class, or paradigm, we created to model the biological mechanisms of pattern recognition proposed by Freeman can be considered a modification of the *Hopfield Network*. The typical Hopfield network architecture, shown in Fig. 4.1, is composed of three layers: the input layer, the Hopfield layer and the output layer. The input layer is used to start the system with some initial condition and the output layer in just a buffer that displays the system's state.

The basic Hopfield paradigm is implemented in the Hopfield layer and consists of a set of processing elements, that is neurons (see Fig. 4.2), which compute the weighted sum of the inputs, the output is then a function of this sum. The function is usually a Sigmoid but it can be a Step Function or an Hypertangent. The output of each processing element is coupled back to the inputs of every processing element except itself. Furthermore the weight matrix is symmetrical: the connection weight $w_{ij}$ between processing element $j \rightarrow i$ is the same as connection weight $w_{ji}$ between processing element $i \rightarrow j$. This insures stability of the network. It does not break into oscillations. Due to this constraint on the synaptic weights the original stable Hopfield network very likely will not be found in biological systems.

The inclusion of feedback connectivity in the Hopfield paradigm is also present in all the networks we constructed and distinguishes them from the Back-Propagation or the Perceptron-like networks, which emphasize feed-forward connectivity. Both systems can be described by a set of coupled nonlinear difference equations for which no closed-form solution exists. The main difference between the two paradigms is that the re-entrant networks we devised purposely lack the symmetry of the Hopfield networks, in other words they can be considered *asymmetrical Hopfield networks*.  

33
Symmetry of the connections results in a powerful theorem about the behavior of the Hopfield system, and this makes it useful in practical applications. On the other hand our asymmetric systems show a very rich and complex dynamic, which is more...
representative of biological networks, but which is also a lot more difficult to predict and to analyze, as we will see.

Hopfield proved that the symmetry of connections results in a system that cannot oscillate. The symmetrical system will instead move during its evolution "downhill" along an Energy function, reach a state in which the energy is a minimum, and stop changing with time. This concept can be illustrated graphically by a flow map in a state-space (or phase plane) diagram like the one in Fig. 4.3

![Energy-terrain contour map.](image)

Fig. 4.3 (A) Energy-terrain contour map. (B) Typical flow map of neural dynamics for the Hopfield network with symmetric connections \( w_{ij} = w_{ji} \). (C) More complicated dynamics can occur in the asymmetric configuration.

4.1 Energy Function

One of the key contributions of Hopfield was the formulation of the dynamics of his network in terms of energy surface (spin glass physics). In a simplified way the system in Fig. 3.1, with the step function output, has energy

\[
E = -\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{m} w_{ij} x_i x_j
\]  

(4.1)
The portion of $E$ effected by the state $x_i$ of a particular processing element is given by:

$$E_i = -\frac{1}{2} \sum_j (w_{ij}x_j)$$

(4.2)

If $x_i = 1$,

$$E_i = -\frac{1}{2} \sum_{j \neq i} (w_{ij}x_j)$$

(4.3)

and if $x_i = 0$,

$$E_i = 0$$

Therefore the change in energy due to one of the processing elements changing states is:

$$\Delta E_i = -\frac{1}{2} \Delta x_i \sum_j (w_{ij}x_j)$$

(4.4)

Looking at equation 4.4, if $x_i$ is currently 0, it is because the weighted sum of the inputs is negative. For $x_i$ to become 1 ($\Delta x_i = 1$), the weighted sum must become positive. The result is that the energy decreases.

Likewise, if the weighted sum is positive, $x_i$ will be positive. For $x_i$ to become 0 ($\Delta x_i = -1$), the weighted sum must become negative. Again the result is that the energy of the system decreases. Thus, an element in the network will only change state if and only if it will decrease the overall energy of the network. As a result, if the network is
placed in a random initial state, it is guaranteed to converge to a minimum, but not necessarily the global minimum.

The energy minimization property of these networks can be used to find solutions to optimization problems, such as the job shop scheduling problem and the Traveling Salesman Problem, in which $E$ is proportional to a cost function. In order to accomplish this goal two problems must be solved. The first is to find a mapping between the problem domain and the final states of the neural network. The second problem is to construct a priori synaptic weight matrix that shapes an energy surface whose minima correspond to optimal solutions of the problem.

4.2 A practical example

To give an example of the computing power of the Hopfield paradigm I will mention a project I did for Professor Hassan Barada at Lehigh University. Professor Barada asked me to solve a scheduling problem in which $n$ processes must be allocated to $m$ processors and, given a dependency diagram, a cost function must be minimized. The problem has optimal (highest throughput, lowest cost) and sub-optimal solutions. In general, given $n$ processes and $m$ processors there are $(n!)^m$ possible schedules (without considering dependencies). With just 7 processes and 2 processors, for example, there are already approximately 25 million possible schedules, a more realistic examples would have hundreds of processes and tens of processors.

Because of this combinational explosion resource allocation belongs to the large class of $np$-complete (non deterministic polynomial time complete) problems. This means that an optimal solution, or global minimum, is very difficult to find and in many cases even unnecessary. The Hopfield network instead has the characteristic to
converge rapidly to a very good solution. In my experiments I tried to allocate 50 processes to 8 processors, it took usually less than 10-20 iterations for the network to converge to an almost optimal solution.

4.3 Observations and conclusions

The discussion on the energy function of a network has been simplified using the step output function, thus avoiding nonlinear terms in the equations. Nevertheless the energy terrain model of Fig. 4.3a can be applied also to networks with nonlinear elements or biological neural networks.

The symmetry of the Hopfield paradigm has no biological foundation, it is merely a way to avoid oscillatory and chaotic behavior of the system. Freeman, on the other hand, showed that oscillations and chaos are possibly the very mechanisms that make perception possible.
5. Small networks with nonlinear elements

This chapter explores the dynamic characteristics of small single-layer networks having nonlinear transfer functions. Unlike Chapter 3, where computer simulations of linear networks complemented an analytical analysis, the results presented in this chapter are strictly experimental. This is a consequence of the nonlinearity of the systems.

We experimented with both Sigmoid and Hypertangent nonlinear transfer functions (see Eqs. 2.2b, 2.2c), with different sizes of networks and different weight distributions. Each network created was tested with different input vectors, gains and biases. Our tests show that of all these parameters the one that most characterizes the dynamic behavior of the networks is the choice of Transfer Function.

![Single layer re-entrant network](image)

**Fig. 5.1** Single layer re-entrant network
In general the networks presented in this chapter share the characteristic of having a single layer and of being limited to a maximum of ten neurons (in most of the cases just two). The architecture is otherwise the same as the one of Figure 2.3. The output of each neuron is given by:

\[ x_i(k+1) = NG \cdot f(TFG \sum_{j=1}^{n} x_j(k)w_{ij} + B) \]  

where

- \( x_j(k) \) is the output of the \( j \)th neuron after \( k \) iterations
- \( NG \) is the scale factor of Eq. (2.3) also called the Network Gain
- \( f() \) is one of the nonlinear transfer functions
- \( w_{ij} \) is the synaptic weight between the axonal output of the \( j \)th neuron and the input of the \( i \)th neuron
- \( TFG \) is the Transfer Function Gain of Eq. (2.2c)
- \( B \) is an offset parameter called the network Bias

The Bias is usually a negative value and has the effect of shifting the transfer function to the right of the origin. This means that the bias can be used to set a threshold for axonal firing with respect to total input potential.

### 5.1 Networks with Hypertangent transfer function

The Hypertangent curve has the characteristic of being almost linear near the origin. It is therefore interesting to observe the similarities and the differences between the linear networks studied in Chapter 2 and the same networks with the Hypertangent Transfer Function. In general we should expect these networks to behave in a similar
fashion near the origin, the behavior of the nonlinear networks as they reach saturation is more difficult to predict. The main difference is that, unlike the linear case, the networks with Hypertangent transfer function exhibit a limited amplitude (the output is limited between $\pm 1$). This section presents five networks, the first four have two neurons and the last has three. They are representative of a large number (about 20) of simulations that were actually performed.

Network 1: the first network presented has the same weight distribution and same initial conditions as the linear network described in Case 1 of Paragraph 3.3.

![Fig. 5.2 Phase plane of Network 1](image-url)
The following parameters are used:

- **Network Gain** = 10
- **Transfer Function Gain** = 1
- **Bias** = 0.

Every point in the phase plane represents a value of $x_1(k)$ vs. $x_2(k)$ starting from $k = 0$, the initial conditions. Just like in the linear case the network starts oscillating near the origin with increasing amplitude. Because of the limiting effect of the Transfer Function the network converges to a stable oscillation between point "A" and "B". These two points define the attractor of the system.

![Phase plane of Network 1 with two initial conditions, 1 and 2](image)

*Fig. 5.3* Phase plane of Network 1 with two initial conditions, 1 and 2
Figure 5.3 shows that the same network with different input vectors, 1 and 2, converges to the same final state. The initial conditions are emphasized with circles and two different colors are used for clarity.

Network 2:

This two neuron network has a dynamic behavior comparable to that observed in Paragraph 3.3, Case 2. The phase plane in Figure 5.4 evolves from the origin and it spirals outwards until the average gain becomes unity due to nonlinear limitations. At this point the system oscillates in a limit cycle, this limit cycle appears to be the only attractor of the phase space.

Figure 5.5 shows the overlapping plots of the phase planes generated using three different initial conditions, 1, 2 and 3. It can be seen that regardless of the starting point the network converges to the same attractor.

The following parameters were used for Network 2:

\[ w_{11} = 0.017507 \]
\[ w_{12} = 0.064206 \]
\[ w_{21} = -0.055288 \]
\[ w_{22} = -0.053389 \]

Network Gain = 20
Transfer Function Gain = 1
Bias = 0.

-In general the value of Bias is zero if not otherwise specified.
Fig. 5.4 Phase plane of Network 2
Fig. 5.5 Phase planes generated by Network 2 with three different input vectors
Network 3:
This network has a limit cycle attractor similar to that of the previous network. Its phase plane is presented because it has a better visible dynamic evolution. Like in the previous case the system converges to the same attractor regardless of the input vector. The following parameters were used:

\[ w_{11} = 0.083273 \]
\[ w_{12} = -0.039973 \]
\[ w_{21} = 0.008703 \]
\[ w_{22} = 0.070092 \]

Network Gain = 14
Transfer Function Gain = 1

Fig. 5.6 Phase plane of Network 3
Network 4:

Most of the two-neuron networks have the tendency to converge to a point attractor, that is to settle to a stable state after a short transient time. In this particular case the phase plane evolves from near the origin and it converges to a point in the fourth quadrant. The following parameters were used for Network 4:

\[ w_{11} = 0.079003 \]
\[ w_{12} = -0.021258 \]
\[ w_{21} = -0.074567 \]
\[ w_{22} = 0.046002 \]

Network Gain = 9.5
Transfer Function Gain = 1

Fig. 5.7 Phase plane of Network 4
This network configuration, unlike the ones seen so far, has two point attractors. Figure 5.8 shows the phase planes generated using two different initial conditions, 1 and 2. The two point attractors are symmetric to each other with respect to the origin.

![Phase planes of Network 4 generated by two different initial conditions](image)

**Fig. 5.8** Phase planes of Network 4 generated by two different initial conditions

Network 5:
This section concludes the report on small networks with Hypertangent transfer function with the study of a three-neuron network. The effects of the Network Gain parameter on the network dynamic are reported for this system. The results of running the network with different initial conditions are also presented.
Fig. 5.9 Phase plane of Network 5: Network Gain = 13

Fig. 5.10 Phase plane of Network 5: Gain = 14
Fig. 5.11 Phase plane of Network 5: Network Gain = 15

Fig. 5.12 Phase plane of Network 5: Network Gain 16
Figures 5.9 to 5.13 show four phase planes \((x_1(k) \text{ vs. } x_2(k))\) of Network 5 that were obtained by running the network with increasing Network Gain parameters. These diagrams indicate that the system's dynamic is dramatically affected by changes in gain. This is an indication that the Network Gain is indeed a bifurcation parameter.

In Figures 5.9 the network quickly converges to a two point oscillatory attractor, a case that was already encountered in Network 1. As the Network Gain increases the network converges more slowly to a stable state via collapsing spirals, see Figures 5.10 and 5.11. Figure 5.12 shows that the stable state of the system with a gain of 16 is a limit cycle (the output has two frequency components which accounts for the two cycles). Finally, by further increasing the gain the state reaches a hard limit (Fig. 5.13). Ultimately the oscillations will continue among four fixed points.
Fig 5.14  (a) Phase plane $x_1(k)$ vs. $x_2(k)$ of Network 5 with Gain = 16.  (b) Phase plane $x_1(k)$ vs. $x_3(k)$ of Network 5 with Gain = 16.  (c) Trajectory of phase plane (a).  (d) Trajectory of phase plane (b).
The plots in Figure 5.14 (a) and (b) show two different phase planes of Network 5 with Gain = 16. The respective trajectories, Figure (c) and (d), were obtained by connecting consecutive points. These plots demonstrate how the dynamic evolves in the three-dimensional phase space, from the initial state near the origin to the limit cycle attractor. Notice that the trajectory diagrams reveal that the state alternates between the two closed contours. Therefore, there really is just one limit cycle, even though it may not appear so initially. Figure 5.15 is a Fast Fourier Analysis of the data plotted in Figure 5.14. It clearly shows the two frequency components present.

Network 5 with different input vectors:
Unlike the other networks that have been considered, Network 5 has two different limit cycle attractors to which the network converges depending on the input vector (initial condition). The part of phase space which encloses all the input vectors that converge
to the same attractor is called the basin of attraction for that attractor. In this case the basin of attraction of the attractor analyzed so far is a small region near the origin, all the other points in phase space converge to the second attractor. Figure 5.15 shows both attractors superimposed and color-coded on the same plot.

Fig. 5.16 Phase planes of Network 5 with Network Gain = 16 generated with two different initial conditions
5.2 Networks with Sigmoid transfer function

Unlike the two or three neuron networks with Hypertangent transfer function, the small networks with Sigmoid T.F. do not exhibit a rich and diverse dynamic behavior. These systems are extremely stable and do not exhibit oscillations due to the fact that the transfer function does not have negative values. This section analyses a two-neuron network and a ten-neuron network.

Network 6:
This two-neuron network has been tested with two different input vectors and with the following parameters:

\[
\begin{align*}
    w_{11} &= -0.0976 \\
    w_{12} &= -0.0804 \\
    w_{21} &= 0.0175 \\
    w_{22} &= 0.0642 \\
    \text{Network Gain} &= 10 \\
    \text{Transfer Function Gain} &= 2 \\
    \text{Bias} &= -1.0
\end{align*}
\]

The trajectories in Figure 5.15 show that both input vectors converge to the same final state after a short transient time. The same behavior has been observed in all the small networks with Sigmoid Transfer Function that we have experimented with. These systems converge to a unique point attractor regardless of the parametric space used for Transfer Function Gain, Network Gain and Bias.
Network 7:

Stability is characteristic of these systems and appears to be exceedingly robust. Even increasing the size of the network to ten neurons does not produce new dynamic states. The only point worth noticing in the phase plane in Figure 5.16 is that the network converges to its stable final state through an oscillatory transient state. The same final state is reached regardless of the initial condition, it is therefore a unique point attractor.

Fig. 5.17 Trajectories of Network 6 with two different initial conditions
The phase plane in Figure 5.16 was generated by a single layer, ten neuron network with the following parameters:

Network Gain = 10
Transfer Function Gain = 2
Bias = -1.0

Fig. 5.18 Phase plane of Network 7
6 Large networks with nonlinear elements

This chapter presents a large collection of experimental data obtained from simulations of re-entrant neural networks having 250 neurons. The network and cell architecture (see chapter 2) is the same as that of the small networks that we analyzed in Chapter 5. Here the networks are organized in 10 fully interconnected rows of 25 neurons each. The synaptic weight matrix has a random distribution with values ranging between -0.1 and 0.1. Given a certain weight distribution each network has five variable parameters: bias, network gain, transfer function, transfer function gain and input vector.

We have already seen in the previous chapter that small networks can emulate some of the dynamic characteristics of EEG traces. Small networks were able to recreate oscillatory states and limit cycle attractors similar to those found in the brain. These networks, on the other hand, were not able to generate the kind of chaotic dynamic that Freeman found in the cortical tissue of rabbits.

Large networks, having 250 neurons, are undoubtedly different from two or three-neuron networks in statistical properties, the dynamic behavior of these systems is therefore expected to be quite different too. Furthermore, large nets are more representative of biological cortical tissue (with hundred thousands of neurons per square millimeter).

Each class of activity in rabbits has a characteristic EEG trace associated with it (see Ch. 1 and Fig. 1.3). For example, waking rest is associated with low-amplitude chaotic EEG traces while motivation is associated with strong oscillatory bursts. Before trying to find the functional role for each of these states we should find the mechanisms that make this complex dynamic behavior possible.
We hypothesize that the sudden changes in neural dynamic are caused by thalamo-cortical interactions. The thalamus, we believe, effects the bifurcation parameters of the neural cells, thus dramatically changing the behavior of the entire system. We base this hypothesis on three premises:

1. Cortical tissue appears to have dynamic properties similar to those of complex nonlinear systems. In particular it appears to be able to switch rapidly from oscillatory to chaotic regimes (see Paragraph 1.3).

2. The thalamus can amplify the strength of the synaptic weights and change the slope of the sigmoidal response of axonal firing to input potential [8].

3. In nonlinear systems the slope of the nonlinearity and the overall gain are bifurcation parameters (see Paragraph 1.2).

The main purpose of the experiments described in this and the next chapter is to demonstrate that the re-entrant artificial neural network can behave in similar manner. We want to find if and under which design parameters our model of cortical tissue manifests stable, oscillatory and chaotic dynamic regimes.

Our initial intent was to graphically visualize the network dynamic like Freeman did, with topographic contour plots (like those in Figure 1.2 for example). Phase planes, on the other hand, require much less computing time and convey more information about the system and its dynamic. We therefore made extensive use of phase planes as a visualization tool. Fast Fourier Transform analysis of frequency spectra and phase spectra have also been used.

The networks presented here are representative of a much larger number of simulations that were actually performed. The experimental data is organized in two
sections, one for networks with Hypertangent transfer function and one for networks with Sigmoid transfer function.

6.1 Networks with Hypertangent transfer function

This chapter presents a network whose neurons implement the Hypertangent transfer function. The network is tested with different values of network gain and transfer function gain, the two bifurcation parameters. The synaptic weight matrix is the same for all the tests (there are 6,250 weights in the network, 25 weights for each of the 250 neurons). This section analyses the results of six simulations made on the network.

Simulation 1.
Given a weight matrix randomly generated between -0.1 and +0.1, which defines the network, we experiment with increasing values of network gain and different input vectors. The parameters for this simulation are:

\[
\begin{align*}
\text{Network Gain (NG)} &= 3.5 \\
\text{Transfer Function Gain (TFG)} &= 1.0 \\
\text{Bias} &= 0.0
\end{align*}
\]

The effect of Bias on this network paradigm is not well understood and more research is necessary. Its value remains zero throughout the remaining simulations.

Figure 6.1 shows that for a small Network Gain the system's phase plane is a collapsing spiral. In general the values of the Network Gain variable will not be as high as with small networks of two or three neurons. With many contributions to the summation function the probability that the sum of input potentials will be close to zero
is smaller. In other words the Probability Density Function of the sum changes with the number of terms. This means that a larger network needs a smaller network gain parameter to achieve the same potential energy.

Fig. 6.1 Phase plane generated by simulation 1. NG = 3.5, TFG = 1.0

Simulation 2.

In this simulation the same network, with the same input vector, is tested with a higher network gain:

\[
\text{Network Gain} = 3.8 \\
\text{Transfer Function Gain} = 1.0
\]
Figure 6.2 shows that the system falls very rapidly (only three transient points) into a well defined limit cycle. Other phase planes of the same attractor are shown in Figure 6.3.

So far the behavior of the network is comparable to that of the smaller networks analyzed in Chapter 5, the main difference is that here the system falls almost instantaneously into the attractor. This property is quite astonishing and, given the complexity of the system, against intuitive understanding.

When the same simulation is repeated with a different input vector the network falls into the same limit cycle attractor (compare Figure 6.4 with Figure 6.1). Ideally we
would have liked to see the network converge to different limit cycles for every input vector (initial condition) but this does not seem to be the case. There is only one limit cycle, the basin of attraction consists of the entire phase space.

Fig 6.3 Phase planes generated by simulation 2, NG = 3.8, TFG = 1.0
A limit cycle attractor results from steady oscillations of the two variables being plotted having the same frequency but different phase spectra. The FFT of a vector \( x \) having \( n = 2^m \) values (where \( m \) is an integer) is a vector of \( k = 2^{m-1} + 1 \) values whose \( j \)th element is given by:

\[
a_j = \frac{1}{\sqrt{n}} \sum_k x_k e^{2\pi j k/n}
\]  

(6.1)

Fig. 6.4 Phase plane generated by simulation 2 using a different input vector
Fig 6.5 Output of a neuron generated by simulation 2, time domain

Fig. 6.6 Frequency spectrum of the signal
Figures 6.5, 6.6 and 6.7 are the time, frequency and phase domain plots of a single neuron's output. As one can see, there is a dominating resonant frequency in the spectrum. Figures 6.7 to 6.9 refer to a second neuron in the network, the frequency spectra have the same components (with different magnitudes) but the phase plots are completely different.
Fig. 6.8 Output of a neuron generated by simulation 2, time domain

Fig. 6.9 Frequency spectrum
Simulation 3.

The next step is to further increase the network gain variable. Figure 6.11 shows a phase plane resulting from simulating the network with the following parameters:

\[
\text{Network Gain} = 5.0 \\
\text{Transfer Function Gain} = 1.0
\]

The outcome of this simulation is rather astonishing, the dynamic of the network is dominated by a chaotic attractor. Figure 6.12 displays other phase plane projections of the strange attractor. The symmetry of the attractor is probably a result of the symmetry of the Hypertangent function without bias.

Figures 6.13 trough 6.15 portray the signal generated by a single neuron in time domain, its frequency and phase spectra respectively. The noisy attribute of frequency and phase spectra of this deterministic signal is a definite sign of the presence of chaos.
Fig. 6.11 Phase plane generated by simulation 3, NG = 5.0, TFG = 1.0
The signal in Figure 6.13 and its spectra are so noisy that it is useless to compare them with those of another neuron. It is interesting instead to generate a phase plane with a different input vector, and see how the attractor differs from the first one. Figure 6.16 overlaps two phase planes generated with two different input vectors, notice that the overall shape of the attractor is the same but no two points coincide.
The results of this simulation lead to the conclusion that the network gain is indeed a bifurcation parameter of this paradigm.

Fig. 6.13  Output generated by a neuron in simulation 3, time domain

Fig. 6.14  Frequency spectrum
Fig. 6.15 Phase plane generated by two different input vectors
Simulation 4.

The first three simulations have demonstrated the effect of increasing network gain, while all other parameters stayed the same. In the next two simulations we are going to analyze the effect the transfer function gain has on the network dynamic. In this simulation the data is generated using the following parameters:

- Network Gain = 3.8
- Transfer Function Gain = 1.5
- Bias = 0.0

These are the parameters used in the second simulation, only that the transfer function gain has a slightly higher value. Figure 6.16 shows the phase plane generated by the simulation.

Fig. 6.16 Phase plane generated by simulation 4, NG = 3.8, TFG = 1.5
The phase plane in Figure 6.16 corresponds to the phase plane in Figure 6.2, the same variables of the same network are plotted against each other. The only difference is a higher transfer function gain parameter (1.5 instead of 1.0), which corresponds to a steeper slope of the Hypertangent curve. The difference in dynamic, on the other hand, is dramatic. The first implication is that also the slope of the nonlinear transfer function is a bifurcation parameter. The second implication is that we have strong evidence of how low-amplitude chaotic waveforms might be generated in the brain.

Figure 6.16 hides another fascinating phenomena. The simulation was set to run for 8000 iterations but much fewer points are visible on the phase plane. This indicates that points start repeating sometime during the evolution of the system. This assumption is confirmed by the plotting the output of a processing element in time domain. Figure 6.17 reveals that after circa 1200 iterations the network converges to a stable oscillatory state.

Fig. 6.17 Output of a neuron generated by simulation 4, time domain
Simulation 5.

The natural next step is to further increase the transfer function gain. The following parameters are used:

Network Gain = 3.8
Transfer Function Gain = 1.7
Bias = 0.0

As expected an increase of the TFG prevents the system from converging to a stable state.

Fig. 6.18 Phase plane generated by simulation 5, NG = 3.9, TFG = 1.7
It is hardly possible to call the distribution in Figure 6.18 an attractor since it spreads the entire phase space. Figure 6.19 shows that in other phase planes generated by the same simulation there is a concentration near the saturation values of ±1, nevertheless this dynamic pattern is quite astonishing.

![Fig. 6.19 Phase planes generated by simulation 5, NG = 3.8, TFG = 1.7](image)

Figures 6.20 shows that the system does not converge to any stable oscillation even
after 4000 iterations. The frequency spectrum of the signal does not have any dominating component, it is noisy throughout the range. The last figure of this section, Figure 6.22, portrays two color-coded overlapped phase planes generated by the network with two different input vectors. Again the overall shape of the distribution is the same in the two cases but no two points coincide.

**Fig. 6.20** Output of a neuron in simulation 5, time domain

**Fig. 6.21** Frequency spectrum
Fig. 6.22 Two overlapped phase planes generated with two different input vectors
Simulation 6.
In simulation 5 chaos was generated by increasing the transfer function gain and maintaining a constant value of network gain. This simulation shows that by decreasing the network gain the system returns to a stable oscillatory state. This demonstrates how the two bifurcation parameters can independently produce oscillatory and chaotic dynamic regimes. Simulation 6 uses the following parameters:

Network Gain = 2.2
Transfer Function Gain = 1.7

Fig. 6.23 Phase plane generated by simulation 6, NG = 2.2, TFG = 1.7
Figures 6.23 and 6.24 are almost identical to Figures 6.2 and 6.3 respectively, this result is quite surprising. Before running this simulation we were expecting the network to converge to some kind of regular oscillatory pattern, like in simulation 2, but not the same one.

Fig. 6.24 Phase planes generated by simulation 6. NG = 2.2, TFG = 1.7
Table 6.1 summarizes the results obtained in the six simulations of the 25 by 10 neuron re-entrant network.

<table>
<thead>
<tr>
<th>Simulation</th>
<th>Network Gain</th>
<th>Transfer Function Gain</th>
<th>Resulting dynamics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.5</td>
<td>1.0</td>
<td>collapsing spiral</td>
</tr>
<tr>
<td>2</td>
<td>3.8</td>
<td>1.0</td>
<td>limit cycle</td>
</tr>
<tr>
<td>3</td>
<td>5.0</td>
<td>1.0</td>
<td>chaos</td>
</tr>
<tr>
<td>4</td>
<td>3.8</td>
<td>1.5</td>
<td>edge of chaos</td>
</tr>
<tr>
<td>5</td>
<td>3.8</td>
<td>1.7</td>
<td>chaos</td>
</tr>
<tr>
<td>6</td>
<td>2.2</td>
<td>1.7</td>
<td>limit cycle</td>
</tr>
</tbody>
</table>

Table 6.1 Summary of simulation results

6.2 Networks with Sigmoid transfer function

This section presents the results of several simulations performed on a 25 by 10 element network. The processing elements implement the Sigmoid curve as transfer function. Biological neurons also have a sigmoidal response of axonal firing to input potential, this artificial network is therefore more representative of biological neural networks.

This paragraph presents eight simulations, they analyze the behavior of the network under different values of bias, network gain and transfer function gain.
Simulation 1.

We have seen in Chapter 5 that the small networks in which neurons implement the Sigmoid transfer function are very stable, their dynamic is strongly dominated by a point attractor. The same is true in larger networks for a wide range of bifurcation parameters. Simulation 1 shows that if the transfer function gain is kept low (TFG = 1.0 in this case) network gain value as high as NG = 30.0 cannot destabilize the system.

In this simulation the following parameters are used:

Network Gain = 30.0
Transfer Function Gain = 1
Bias = -0.4

Fig. 6.25 Output of a single neuron in time domain

All 250 neurons in the network lock at a constant value after a single iteration regardless of the network gain value. This behavior is quite different from what we
saw in the previous simulations, it suggests that Sigmoid based networks cannot achieve diverse dynamic regimes as easily as the Hypertangent ones. More variables will have to be adjusted in a cooperative fashion.

Simulation 2.
The following parameters induce an oscillatory state, note that it is necessary to greatly increase the transfer function gain:

- Network Gain = 8.0
- Transfer Function Gain = 6.0
- Bias = -0.4

**Fig. 6.26** Output of a single neuron in time domain

All neurons in the network oscillate at the same frequency and are all in phase, only the magnitude of the oscillations change among neurons.
Simulation 3.
By further increasing the transfer function gain and reducing the network gain parameter the network converges once again to a point attractor. This is another indication that the network is quite stable and that all the variable parameters must be properly set to achieve bifurcations. In this simulation:

Network Gain = 4.3
Transfer Function Gain = 10.0
Bias = -0.4

Fig. 6.27 Output of a single neuron in time domain

Simulation 4.
In the following three simulations the transfer function gain is kept constant while the network gain is increased each time. The parameters used in this simulation are:

Network Gain = 4.4
Transfer Function Gain = 10.0
Bias = -0.4

The phase planes in Figure 6.28 and 6.29 visualize a complex limit cycle attractor. Figures 6.30, 6.31 and 6.32 are the time, frequency and phase plots generated by this simulation. While the frequency and phase spectra look like those of a chaotic attractor, the output versus time plot reveals alternating sequences of chaotic and oscillatory bursts.
Fig 6.29 Phase planes generated by simulation 4, NG = 4.4, TFG = 10.0, B = -0.4
Fig. 6.30 Output generated by a single neuron in simulation 4, time domain

Fig. 6.31 Frequency spectrum
Simulation 5.

The results of this fifth simulation are a little puzzling, they confirm the unpredictability and complexity of nonlinear systems dynamic. By increasing the network gain the system returns to a stable oscillatory state instead of further bifurcating. The parameters used in this simulation are the following:

- Network Gain = 4.5
- Transfer Function Gain = 10.0
- Bias = -0.4
Fig. 6.33 Output of a single neuron generated by simulation 5 in time domain

Simulation 6.

By further increasing the network gain the system begins to oscillate in a chaotic mode. The strange attractor portrayed in Figure 6.34 and 6.35 results from running the network with the following parameters:

Network Gain = 5.0
Transfer Function Gain = 10.0
Bias = -0.4
Fig 6.34 Phase plane generated by simulation 6, NG = 5.0, TFG = 10.0, B = -0.4
The time, frequency and phase plots confirm the presence of chaos. The color coded Figure 6.36 indicates that, just like in section 6.1, two different input vectors converge to the same attractor but no two points in the phase plane coincide.
Fig. 6.36 Two overlapped phase planes generated by simulation 6 with two input vectors
Simulation 7.

Next the role of bias in network dynamic is investigated. The parameters used in simulation 5 constitute a frame of reference for this and the next simulations. The parameters used in simulation 7 and 8 are the same as simulation 5, only the bias changes:

- Network Gain = 4.5
- Transfer Function Gain = 10.0
- Bias = -0.3

As Figure 6.37 shows, the system is not oscillating anymore but converges to a stable state.

![Figure 6.37](image)

Fig. 6.37 Output of a single neuron generated by simulation 7, NG = 4.5, TFG = 10.0 B = -0.3

Simulation 8.

In this simulation the bias is slightly increased (made more negative) with respect to the parameter used in simulation 5.

- Network Gain = 4.5
Transfer Function Gain = 10.0

Bias = -0.5

Fig. 6.38 Output of a single neuron generated by simulation 8, time domain

Figure 6.38 shows that if the bias becomes too negative the entire network decays to zero excitation. By comparing the results obtained in simulation 5, 7, and 8 it appears that this network paradigm is extremely sensitive to small changes in bias level.

Table 6.2 is a summary of all the simulations presented, the parameters and resulting dynamics. Table 6.3 shows three sequences of simulations, in each row only the indicated variable is changed while the others remain constant.
<table>
<thead>
<tr>
<th>Simulation</th>
<th>Network Gain</th>
<th>Transfer Function Gain</th>
<th>Bias</th>
<th>Resulting dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30.0</td>
<td>1.0</td>
<td>-0.4</td>
<td>point attractor</td>
</tr>
<tr>
<td>2</td>
<td>8.0</td>
<td>6.0</td>
<td>-0.4</td>
<td>oscillation</td>
</tr>
<tr>
<td>3</td>
<td>4.3</td>
<td>10.0</td>
<td>-0.4</td>
<td>point attractor</td>
</tr>
<tr>
<td>4</td>
<td>4.4</td>
<td>10.0</td>
<td>-0.4</td>
<td>chaos/oscillation</td>
</tr>
<tr>
<td>5</td>
<td>4.5</td>
<td>10.0</td>
<td>-0.4</td>
<td>oscillation</td>
</tr>
<tr>
<td>6</td>
<td>5.0</td>
<td>10.0</td>
<td>-0.4</td>
<td>chaos - strange attractor</td>
</tr>
<tr>
<td>7</td>
<td>4.5</td>
<td>10.0</td>
<td>-0.3</td>
<td>point attractor</td>
</tr>
<tr>
<td>8</td>
<td>4.5</td>
<td>10.0</td>
<td>-0.5</td>
<td>zero output</td>
</tr>
</tbody>
</table>

Table 6.2 Summary of simulations presented in section 6.2

<table>
<thead>
<tr>
<th>Network Gain:</th>
<th>3 → 4 → 5 → 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>point attractor</td>
<td>limit cycle</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transfer Function Gain/Network Gain</th>
<th>1 → 2 → 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>point attractor</td>
<td>oscillation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bias:</th>
<th>7 → 5 → 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>point attractor</td>
<td>oscillation</td>
</tr>
</tbody>
</table>

Table 6.3 Three simulation sequences
7 Re-entrant networks with Hebbian learning

In the simulations of the previous section we have tried to use varying initial conditions in order to see whether the system would converge to different attractors, be they chaotic, limit cycles or fixed points. The success was limited, only in rare cases we were able to observe the networks converge to different final states depending on the input vector (see Fig. 5.15 for one such case).

The idea pursued throughout this investigation is whether the re-entrant network paradigm, under certain circumstances, can act like classifiers of patterns. In analogy to the olfactory bulb of rabbits we were looking for different and specific dynamic patterns into which the system would fall upon presentation of different initial vectors.

All self adapting classifiers use Hebbian type learning rules, therefore it was assumed that chances of success would be increased if learning rules were added to our system. If that would enhance the discrimination between various input vectors, at least partial success would be achieved.

It has been noted that memory formation occurs during the peak of a slow global wave that is observed in the brain, the Theta wave [12]. We applied this concept to the control strategy of the learning process. The strategy consists of forcing a chaotic oscillatory state during learning, decreasing the bifurcation parameters when learning is over and let the system settle to a non-chaotic state. Bifurcation parameters therefore are not fixed, they change globally in time inducing different dynamic behaviors and different properties in the network that can be used for different functions (like learning and recognition).

If Hebbian learning will produce strongly different dynamic patterns for different input vectors, our systems would have a certain resemblance of a classifier. Ideally the
dynamic patterns should be stable oscillations or limit cycles, they would be equivalent to the topographic maps of oscillatory magnitude seen by Freeman in the olfactory bulb of rabbits (see Fig. 1.1 and 1.2).

7.1 Learning applied to large networks with Hypertangent transfer function

The learning rule we use in this section is known as Hebb/Anti-Hebb and is a modification of the basic Hebbian rule. Using the nomenclature of Fig. 2.1:

\[ w'_{ji} = w_{ji} + C_1 S(y_j) [2S(x_{ji}) - 1] \]  \hspace{1cm} (7.1)

where:

- \( w_{ji} \) is the synaptic weight connecting the \( i \)th neuron of the previous layer to the \( j \)th neuron in the current layer.
- \( w'_{ji} \) is the synaptic weight after learning step.
- \( C_1 \) is the learning rate.
- \( y_j \) is the output of the current \( j \)th neuron.
- \( x_{ji} \) is the input excitation to weight \( w_{ji} \).
- \( C_2 \) is a threshold value.
- \( S() \) is a step function, \( S() = 0 \) if argument < \( C_2 \), \( S() = 1 \) otherwise.

In words this rule can be stated as follows: the weight changes only if the current output is above threshold. The weight is incremented if the input is above threshold, decrement otherwise.

The Hebb/Anti-Hebb learning rule is applied to the network described in section 6.1 for the number of iterations defined by the Learning Cycles variable. The network gain
during learning is defined by Network Gain 1, after learning it is decrement to Network Gain 2. All other parameters stay the same:

\[
\begin{align*}
\text{Learn Cycles} &= 20 \\
\text{Network Gain 1} &= 5.0 \\
\text{Network Gain 2} &= 3.8 \\
\text{Transfer Function Gain} &= 1.0 \\
\text{Bias} &= 0.0 \\
C1 &= 0.001 \\
C2 &= 0.5 \\
\end{align*}
\]

Figures 7.1 and 7.2 show different phase planes of limit cycles (color coded Black, Blue and Red) generated from three different input vectors. The vectors used are,

- black: 0.45 0.25 0.33 0.19 0.02 0.35 0.17 0.56 0.03 0.26 0.86 0.45 0.15 0.10 0.39 0.29 0.92 0.84 0.66 0.34 0.12 0.23 0.34 0.45 0.56
- red: 0.34 0.25 0.63 0.99 0.62 0.34 0.27 0.66 0.93 0.06 0.96 0.75 0.55 0.30 0.31 0.45 0.95 0.45 0.68 0.94 0.02 0.20 0.74 0.45 0.56
- blue: 0.24 0.25 0.63 0.29 0.23 0.74 0.89 0.67 0.56 0.45 0.98 0.76 0.34 0.23 0.21 0.43 0.23 0.23 0.34 0.54 0.56 0.23 0.12 0.23 0.46

Note that these values are multiplied by the network gain before becoming input vectors.
Fig. 7.1 Phase plane portraying the limit cycles generated by three input vectors with learning
In this simulation we successfully made the network converge to a different final oscillatory state for each input vector. Further testing could be done to see whether every input vector converges to a unique limit cycle or, more probably due to the threshold condition, similar input will converge (cluster) to the same attractor.
7.2 Learning applied to large networks with Sigmoid transfer function

In this section we apply a different learning rule to the network used in section 6.2, the learning strategy is the same as in the previous section. The learning rule is another modification (by Grossberg) of the basic Hebbian rule:

\[ w'_j = w_j + C_1 [C_2 \frac{y_j x_{ji}}{(NG)^2} - w_j] \]  

(7.2)

Here \( C_1 \) determines the value towards which the weight drifts. There is no change when the two terms in the bracket cancel. \( C_2 \) is the learning rate.

It was already stated that the sigmoid function leads to relatively stable networks. Even though we were able to find oscillatory and chaotic configurations, so far we did not find oscillatory states that strongly depended on the input vector, even though small variations were observed. Figure 7.3 however shows that the same network falls into two different stable states depending on the choice of input vector.

The parameters used to generate Figure 7.3 are the following:

- Learning Cycles = 10
- Network Gain 1 = 5.0
- Network Gain 2 = 4.6
- Transfer Function Gain = 10.0
- Bias = -0.4
- \( C_1 = 0.03 \)
- \( C_2 = 0.2 \)
The two input vectors used are the same as in the previous section. The parametric space explored was limited. Therefore it can not be concluded that behaviors as seen (in section 7.1) with Hypertangent transfer functions do not happen with the Sigmoid functions.
8 Conclusions

We have demonstrated various dynamic states in re-entrant nonlinear neural networks. Particularly in large networks we had expected to see predominantly chaotic unstable behavior. We were astonished to see that, on the contrary, many adjustments had to be made before chaos would emerge. Altogether order, in the form of limit cycles and point attractors, seems to predominate even more in large networks of this kind.

We found classification of input vectors to happen without a permanent memory record in the synaptic strength. Of course we are far from explaining the smell specific patterns on the rabbit's olfactory bulb, yet the phenomena observed point in the right direction.

When the threshold from chaos to order is crossed while presenting an input vector, classification can take place despite a random distribution of weights. This also requires a cyclic process as observed in the Theta wave pattern in brains.
References

Vita

Aldo M. Frigo was born in Verona, Italy on November 20, 1966. He graduated from Lehigh University in 1990 with a Bachelor of Science degree in Electrical Engineering. After graduation he worked in the Research and Development center of Porsche GmbH in Germany for one year. He is currently working towards his Master of Science degree in Electrical Engineering at Lehigh. He is a member of IEEE.
END
OF
TITLE