

Chaos and information in dynamic neural networks

Tjeerd Olde Scheper (2002)

<https://radar.brookes.ac.uk/radar/items/e2a920c8-ff78-4ad6-adf3-8217d18c3b96/1/>

Note if anything has been removed from thesis: published papers at end of thesis

Copyright © and Moral Rights for this thesis are retained by the author and/or other copyright owners. A copy can be downloaded for personal non-commercial research or study, without prior permission or charge. This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the copyright holder(s). The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the copyright holders.

When referring to this work, the full bibliographic details must be given as follows:

Olde Scheper, T. (2002) *Chaos and information in dynamic neural networks*, PhD, Oxford Brookes University

Chaos and Information in Dynamic Neural Networks

Tjeerd Victor Siebe Maria olde Scheper

School of Computing and Mathematical Sciences

Oxford Brookes University

Thesis submitted in partial fulfillment of the
requirements of the award of Doctor of Philosophy
as awarded by Oxford Brookes University

March 2002



IMAGING SERVICES NORTH

Boston Spa, Wetherby

West Yorkshire, LS23 7BQ

www.bl.uk

**PAGES NOT SCANNED AT
THE REQUEST OF THE
UNIVERSITY**

**SEE ORIGINAL COPY OF
THE THESIS FOR THIS
MATERIAL**

Contents

List of Figures	7
List of Tables	10
Abstract	11
1 Introduction	13
1.1 Welcome	13
1.2 Dynamic neural network hypothesis	17
1.3 Description	17
1.4 Dedication	18
2 Chaos	20
2.1 Introduction	20
2.2 Dynamics	21
2.3 Stability and attractors	24
2.4 Periodic solutions and stability	33
2.5 Poincaré map	37
2.6 Bifurcation	39
2.7 Measures and ergodicity	46
2.7.1 Lebesgue measure	46

2.7.2	Natural measure	48
2.7.3	Borel measure	50
2.7.4	Ergodic theorem	51
2.8	Quasiperiodicity	52
2.9	Chaos	55
2.10	Dynamical characterizations	57
2.10.1	Embedding	57
2.10.2	Dimensions	59
2.10.3	Lyapunov exponents	68
2.10.4	Topological and metric entropy	71
2.11	Control and synchronization	73
2.11.1	Unstable Periodic Orbits	74
2.11.2	Chaotic control by the OGY method	75
2.11.3	Chaotic control by external periodic force	79
2.11.4	Chaotic control by time delay feedback	80
2.11.5	Chaotic control by chaos	85
2.11.6	Synchronization of chaos	87
2.11.7	Conclusion	90
3	Literature study	92
3.1	Introduction	92
3.2	Biological references	97
3.2.1	Neuronal networks	99
3.2.2	Memory	101
3.2.3	Biological dynamic behaviour	102
3.2.4	Biological models	103

3.3	Chaotic neural networks	104
3.3.1	Stability of chaotic networks	104
3.3.2	Periodic behaviour of chaotic networks	108
3.3.3	Chaotic network models	110
3.3.4	Synchronization of chaotic networks	115
3.4	Conclusion	116
4	Modelling environment	118
4.1	Introduction	118
4.2	Design	119
4.2.1	Object dependencies	120
4.3	Algorithms	122
4.3.1	Numerical algorithms	122
4.3.1.1	File parsing and evaluation	123
4.3.1.2	Numerical integration	128
4.3.2	Signal analysis algorithms	132
4.3.3	User interface algorithms	134
4.4	Tests	134
4.4.1	Numerical tests	135
4.4.2	Chaos	137
4.4.3	Bugs	140
4.5	File formats	140
4.5.1	Equation files	140
4.5.2	Network files	145
4.5.2.1	Network setting files	146
4.5.2.2	Network definition files	148

4.5.2.3	Network equation files	150
4.6	Conclusion	153
5	Experiments	154
5.1	Introduction	154
5.2	Results	159
5.2.1	Rössler model	159
5.2.1.1	Rössler network model	161
5.2.1.2	Gate models	165
5.2.1.3	Single Rössler model	173
5.2.1.4	Delayed feedback model with input	182
5.2.2	Lorenz models	188
5.3	Discussion	195
5.4	Future work	196
6	Biological experiments	197
6.1	Introduction	197
6.2	Methodology	200
6.2.1	Subjects	201
6.2.2	Training procedure	201
6.3	Taste aversion learning experiments	202
6.3.1	Long time retention	202
6.3.2	Specificity	205
6.3.3	Necessity of protein synthesis	207
6.3.4	Transfer experiments	208
6.4	Discussion	212

6.5 Conclusion	216
7 Discussion	217
7.1 Introduction	217
7.2 Experiments	219
7.3 Dynamic neural networks	222
7.4 Consequences	223
7.4.1 Symbolism	223
7.4.2 Genetics	224
7.4.3 Evolution	224
7.4.4 Consciousness	225
7.5 Conclusion	226
A Addendum	227
A.1 Supported functions	227
A.2 File list	229
A.3 Evaluation function	234
B Publications	237
B.1 ICSC/IFAC Symposium on Neural Computation (NC'98) . . .	237
B.2 Theory and Mathematics in Biology and Medicine - Society of Mathematical Biology Conferenec 1999	244
B.3 Recent Advances in Soft Computing 1999	247
B.4 Third World Conference of Nonlinear Analysts, 2000	255
B.5 Recent Advances in Soft Computing 2000	258
B.6 International Journal of Information Sciences Special Issue .	268

B.7 Ninth European symposium on Artificial Neural Networks (ESANN 2001)	282
--	-----

References	289
-------------------	------------

List of Figures

2.2.1 Trajectory and attractor	32
2.2.2 The Duffing model	33
2.3.2 Stable state of the Duffing model	35
2.3.4 Types of critical points	36
2.3.5 Stability of critical points by eigenvalues	39
2.3.6 Stability chart	39
2.3.7 Stability of three dimensional critical points by eigenvalues	41
2.3.8 Limit cycle of the Duffing model	34
2.5.3 A Poincaré map	38
2.6.10 Bifurcation diagram	40
2.6.11 Types of bifurcation	43
2.6.12 Period doubling	46
2.7.12 The infinite time energy set	48
2.8.14 Quasiperiodicity	54
2.9.16 Strange attractor of the Duffing model	56
2.10.15 The primary dimension	58
2.10.17 Dynamic analysis of the Duffing model	59
2.10.18 Sketching and coding of a strange attractor	60
2.10.19 The Lyapunov exponent	60

List of Figures

2.2.1	Trajectory and attractor	22
2.2.2	The Duffing model	23
2.3.3	Stable state of the Duffing model	25
2.3.4	Types of critical points	28
2.3.5	Stability of critical points by eigenvalues	29
2.3.6	Stability chart	31
2.3.7	Stability of three dimensional critical points by eigenvalues	32
2.4.8	Limit cycle of the Duffing model	34
2.5.9	A Poincaré map	39
2.6.10	Bifurcation diagram	40
2.6.11	Types of bifurcation	42
2.6.12	Period doubling	45
2.7.13	The middle third cantor set	48
2.8.14	Quasiperiodicity	54
2.9.15	Strange attractor of the Duffing model	56
2.10.16	The pointwise dimension	63
2.10.17	Dynamic analysis of the Duffing model	66
2.10.18	Stretching and folding of a strange attractor	68
2.10.19	The Lyapunov exponent	69

2.11.20	Stabilising an UPO with the OGY method	77
2.11.21	Continuous control of chaos	81
2.11.22	An unstable periodic orbit (UPO)	83
2.11.23	Chaotic control of chaos	86
4.2.1	Object dependencies	121
4.3.2	Evaluation stack diagram	125
4.4.3	Chaotic trajectory hopping	139
4.5.4	Equation file example	141
4.5.5	Network file example	146
4.5.6	Definition file example	148
4.5.7	Network equation file example	151
5.2.1	Rössler model	160
5.2.2	Rössler model delayed feedback control on y	161
5.2.3	Rössler network model with adaptive delay feedback control	163
5.2.4	Scheme of gate mechanism.	165
5.2.5	Chaos in gate 1 model	167
5.2.6	Gate 1 model double delayed feedback control on x result .	168
5.2.7	Gate 2 model result	170
5.2.8	Gate 3 model result of $\tau_1 = 7, \tau_2 = 10$	171
5.2.9	Rössler model delayed feedback control on y	174
5.2.10	Rössler model delayed feedback control on y results	176
5.2.11	Poincaré sections of y at $x = 0$	177
5.2.12	Rössler model delayed feedback control on y with modified γ	179

5.2.13	Rössler model delayed feedback control on z results with modified β	180
5.2.14	Rössler model with only external input	184
5.2.15	Rössler model delayed feedback control with external input ($\tau = 5.85$)	186
5.2.16	Rössler model with delayed feedback control and external input ($\tau = 10$)	187
5.2.17	Synchronization of two Lorenz models	190
5.2.18	Two connected Lorenz models with variable input on r . . .	194
6.3.1	Taste aversion learning	203
6.3.2	Short term retention of memory	204
6.3.3	Long term retention of memory	204
6.3.4	Taste aversion specificity	206
6.3.5	Taste aversion specificity continued	206
6.3.6	Generalization of taste aversion learning	207
6.3.7	Dependence on protein synthesis	209
6.3.8	Dependence on protein synthesis continued	209
6.3.9	Scheme of transfer experiments.	210
6.3.10	Transfer experiment.	211

List of Tables

2.6.1	Types of bifurcations of one dimensional maps	44
2.11.2	Synchronization of chaos	89
4.3.1	Variable encoding	124
5.2.1	Rössler network model with adaptive delay results	164
5.2.2	Results gate 1 model	168
5.2.3	Gate 2 model results	169
5.2.4	Results gate 3 model	172
5.2.5	Model results	175
5.2.6	Model results	178
5.2.7	Model results	178
5.2.8	Period doubling of γ	180
5.2.9	Results z feedback control with modified β	181
5.2.10	Frequency pattern	184
A.1	List of functions	227
A.2	List of files	229

Abstract

This research attempts to identify and model ways to store information in dynamic, chaotic neural networks. The justification for this research is given by both biological as well as theoretical motivations [2, 27, 29, 46, 60, 107].

Firstly, there seems to be substantial support for the use of dynamic networks to study more complex and interesting behaviour. The artificial neural networks (ANN) have specific properties that define its order, such as size, type and function. Simply extending the ANN with complex non-linear dynamics does not improve the memory performance of the network, it modifies the rate at which a global minimum may be located, if such a state exists. Using non-linear differential equations may add more complexity to the system and thereby increase the possible memory states.

Secondly, even though chaos is generally undesirable, it has important properties that may be exploited to store and retrieve information [98]. These are the space filling, the possibility of control via delayed feedback, synchronization and the sensitive dependence on initial conditions.

It is demonstrated in this thesis that by using delayed feedback, Unstable Periodic Orbits (UPO) may be stabilized to reduce the complexity of a

chaotic system to n -periodic behaviour. This is a well known effect of delayed control in many types of chaotic models (e.g. Rössler equation), and the periodicity of the resulting orbit is determined by the model parameter as well as the delay (τ) and the feedback strength (K) of the control function (F). Even though a theoretical infinite number of UPOs exist within a chaotic attractor only some can practically be stabilized. Furthermore, it is shown that input added to the delayed feedback controlled system allows different orbits to be stabilized. The addition of multiple delays changes the number and types of orbits that are available for stabilization. The use of synchronization between similar sets of chaotic systems may be used to target specific orbits.

1.1 Welcome

This thesis describes the Research Student project I have been working on at the last four years at Oxford Brookes University. It consists of two main different parts that bring together this project. The project began as a search for novel and possibly biologically relevant mechanisms of information storage in networks. A thorough literature study as part of the research demonstrated that a possible mechanism is the use of uncontrolled chaotic state bifurcations in the dynamic behaviour of a network. The

Chapter 1

Introduction

“Science is built up with facts, as a house is with stones. But a collection of facts is no more a science than a heap of stones is a house.”

Jules Henri Poincaré (1854-1912)

La science et l’hypothèse.

1.1 Welcome

This thesis describes the Research Student project I have been working on for the last four years at Oxford Brookes University. It consists of several different parts that bring together this project. The project began as a search for novel and possibly biologically relevant mechanisms of information storage in networks. A thorough literature study as part of the research demonstrated that a possible mechanism is the use of controlled chaos to store information in the dynamic behaviour of a network. The in-

formation storage and processing by the dynamic behaviour has been made the central premise of the project. Furthermore, in collaboration with experimental biologists, experiments have been devised to provide biological support for this hypothesis. These experiments were suggested by me, but executed by independent researchers at the School of Biological and Molecular Sciences at Oxford Brookes University. Possible future collaborations with other researchers are currently being considered.

My previous work, that involved signal analysis using non-linear tools and chaos analysis of epileptic patient EEGs, showed that extremely complex behaviour may be produced by a reduced and pathological system. This type of dynamic behaviour is not exploited by neural network models. This gave rise to the idea how new models may be constructed that show complex behaviour but are still capable of information storage and processing but at the same time simple enough to that we understand may their dynamics. A chaotic system is generally undesirable because of its nature and the limited analytical tools available. Chaotic systems have, on the other hand, some very useful properties that may be exploited. These are the space filling, the possibility of control via delayed feedback, synchronization and the sensitive dependence on initial conditions. Several models have been studied to understand how these properties may be employed to store information and some success has been made.

In the literature section several different approaches to build neural network models that may store information in the chaotic behaviour of the system, are described. These models have in common that they rely on the existence of unstable periodic orbits (UPOs) contained within the chaotic at-

tractor. These can be stabilized by controlling the chaos, either by making small perturbations to an appropriate variable or by applying delayed feedback. The second methods seems to be closer to possible biological mechanisms that these model attempt to describe. By using delayed feedback with a delay τ and strength K , it is possible to select a specific orbit within the chaotic attractor. This is proposed to be the “recognized state”, while the normal chaotic state is called the “undetermined state”. The existence of an UPO may be influenced by adapting the parameter space. Thus by making small changes to a selected parameter, UPOs may be found, depending on the period of the external input [18, 109].

Biologically plausible rules for this learning behaviour are the subject of this research. It is thought that when a system is presented with a new input, this does not result in stabilization of a specific orbit (unless it is sufficiently similar); subsequently, a hypothetical learning algorithm may add additional delayed feedbacks (various τ) with appropriately chosen strengths K . Then the input may select any of the “new” orbits which may be fine tuned to correspond with the input. This would involve making small changes to K , τ and the system parameters. One of the interesting features of controlled feedback is the fact that the relation between K and controlling feedback function F is not constant but has a global minimum related with specific input [79, 81]. This may be interpret as follows, the weight of the delayed feedback and the delay length do not linearly depend on each other. Rather an optimum value exists for K and τ to stabilize a specific orbit. This implies that weights do not necessarily need to be strengthened when two neurons are active but may be weakened to opti-

mise the connection. To determine the length of delay τ an adaptation rule has been devised. This equation allows the system to adapt the delay automatically with a certain rate. Unstable periodic orbits that are available for stabilization within the current parameter set may easily be localized.

Another aspect of this research is the connection of different networks and direction of the information flow. This has been studied using models to discover if specific information may be targeted using delayed feedback as well as the possible existence of a pre-attentive state induced by synchronization. This state may then preselect a specific subsection of the phase space in which an Unstable Periodic Orbit may be stabilized.

To study these different properties, a systematic approach has been chosen. An extensive literature study demonstrated that there is considerable support for the dynamic information storage hypothesis. Also, in collaboration with biologists experiments have been conducted by them to support this idea. Software development to study the chaotic system has been a substantial part of the project.

In the discussion section is described the result of this study. The experiments demonstrate that it is possible to stabilize a periodic orbit, that is associated with a “recognized state”, with delayed feedback; that the external input may influence the ability of the feedback to stabilize orbits and that with multiple feedback and variations in the parameter space the unstable periodic orbits that are available for stabilization may be changed. Also some indications exist that suggest that synchronization between individual chaotic systems may target specific orbits.

1.2 Dynamic neural network hypothesis

A chaotic neural network is capable of changing its dynamics from chaos to n -periodic when presented with input. This represents a recognition state of the network. Input can result in the network changing its behaviour corresponding to different recognition states. However, the problem remains that the network does not by itself adapt its behaviour to learn a specific input but merely responds to the dynamics imposed by its parameter set. Modification of the parameter set may have severe consequences for the plasticity of the network and could negate all previously recognized states. Therefore, it is suggested that the parameters may be modified within certain boundaries to optimise certain recognition states but that only an extension of the parameter space can provide the necessary increase in complexity to store additional pieces of information.

It is proposed that a network may stabilize different recognition states by multiple continuous feedback control. The existence of different feedback delays provides the system with the required extension of its parameter space. Selection of one of the feedback delays by the input results in the change in dynamics associated with the recognized state.

1.3 Description

The completed work consists of an introduction to dynamical systems and a mathematical study of the mechanisms that form the basis of stabilizing the periodic orbit of a chaotic attractor (Chapter 2 on page 20), an extensive literature study of mathematical and biological relevant work (Chapter 3 on

page 92), software development to study the dynamic behaviour of chaotic networks (Chapter 4 on page 118), collaboration with biological experimentalists to develop experiments that attempt to demonstrate the principle of dynamic information storage and processing (Chapter 6 on page 197) and the modelling of different dynamical aspects of chaotic neural networks (Chapter 5 on page 154). This is concluded with a discussion (Chapter 7 on page 217). The addendum contains a file list of the developed software, available functions and an explanation of the evaluation algorithm. This is followed by published material and a list of references.

1.4 Dedication

Even though I would very much like to *date caesari quae sunt caesaris*, sometimes it is not always entirely clear whom to credit for what. So I will just thank a lot of people and hope that I have not forgotten anyone. Thanks to Matthijs Verhage for helping me as a scientist and a friend and giving me a job when I needed one. Fernando Lopes da Silva for always being available for help and discussions and for his kindness to a pushy undergraduate student from a different university. Jan-Pieter Pijn for his tutelage and discussions, especially for showing me, as a medical biologist, how to think more like a physicist. I have missed our discussions of life, the universe and everything during our car drives back to Amsterdam. Jaap van Pelt for his guidance and encouragements. My supervisors Nigel Crook and Chris Dobbyn for their help and for giving me a chance to make a small contribution to “push back the boundaries of science”, as well as everybody

Chapter 2

Chaos

"I consider chaos a gift"

Septima Clark

2.1 Introduction

It has been known for a long time that simple systems may exhibit very complex behaviour. Over a hundred years ago, Henri Poincaré studied the problem of the behaviour of three celestial bodies. By considering the behaviour of the orbits arising from an initial set of points, he was able to show very complex behaviour other than the normal stability types. The type of behaviour he found is now called chaotic behaviour. He produced many more results but the new field did not open up fully until the nineteen sixties when, with the arrival of computer modelling, much more could be done to investigate this strange phenomenon. Since the late seventies and eighties many new contributions have been made to the subject. Chaos has been found in nearly all different branches of nonlinear modelling. Externally

forced nonlinear oscillators in mechanics have typically been found to have a large chaotic parameter range, where regular behaviour is exceptional. In chemistry and theoretical biology where the large set of variables and interactions seem to lead inevitably to chaos, the modelling produces ample examples of it. Signal analysis of biological, chemical, epidemiological and even economic and monetary signals showed that many of these signals are chaotic. Even so, it appears that chaos is still an exceptional phenomenon in the field of dynamics and is considered undesirable because of the limited analytical tools available to analyse it, as well as the inherent difficulties in the determination of its existence. This chapter introduces some necessary basic concepts and theory behind chaos and its use in dynamic neural networks.

2.2 Dynamics

Generally speaking the field of dynamical systems is a mathematical study into the way everyday observations behave in time or space. To understand how this may be achieved many concepts need to be defined and explained. The study of dynamics (as opposed to statics) has been performed for many centuries and not until the sixteenth century was statics understood as a subset of dynamics. Furthermore, it needs to be stressed that dynamic behaviour, or the way something changes, is elementary to most modern disciplines, but for some this has not been recognized as such.

Differential equations are a convenient way to study dynamics and are therefore much discussed in this subject. A generic higher order differen-

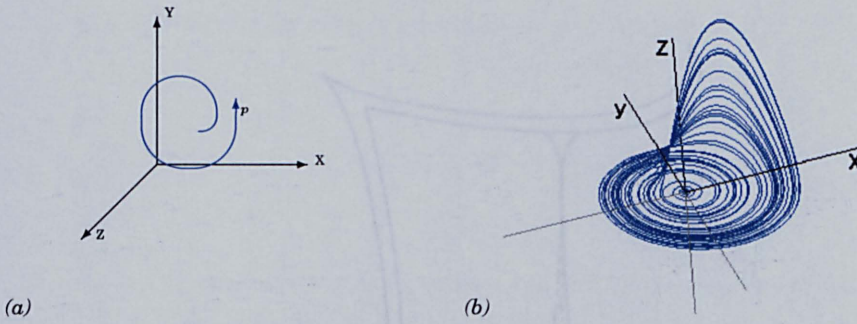


Figure 2.2.1: (a) A trajectory p in the phase space XYZ. (b) The Rössler attractor.

tial equation may be rewritten as a set of first order differential equations. When each of N variables is encompassed into a N -dimensional space, this is called the phase space. As the behaviour evolves over time it describes a path, the trajectory, in phase space (see figure 2.2.1). The variable changes in time are the evolution and because of the uniqueness of a solution, trajectories never intersect except in a stable attracting state. Where the solution is an equilibrium, a critical point may be expected. Eventually the trajectory may settle into a specific area or subspace of the phase space, an attractor, after all transients have died out. In dissipative systems the attractor can be classified in five different groups. Each different attractor will be discussed shortly.

A phase space is defined as follows (where \dot{x} denotes $\frac{dx}{dt}$), $\dot{x} = f(x)$ given with $x \in D \subset \mathbb{R}^n$, then D is called the phase space within the rational space \mathbb{R}^n . If a solution tends in the limit $t \rightarrow \infty$ towards an equilibrium solution, the trajectory tends to go to a critical point. If a neighbourhood $\Omega_a \subset \mathbb{R}^n$ exists around a critical point $x = a$ of the equation $\dot{x} = f(x)$ such that $x(t_0) \in \Omega_a$ and $\lim_{t \rightarrow \infty} x(t) = a$ then this point is called a positive attractor. If this property exists for the point $x = a$ when $\lim_{t \rightarrow -\infty} x(t) = a$, then this point

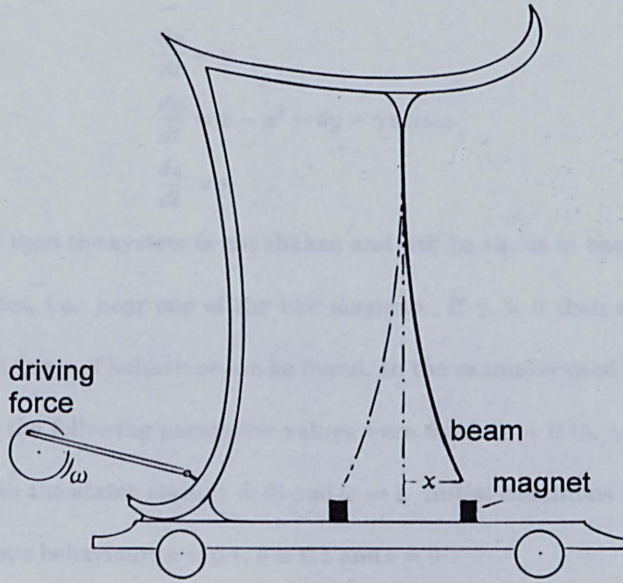


Figure 2.2.2: The physical model described by the Duffing equation. The beam may move freely between the two magnets, by applying an external periodic force, periodicity and chaos may be found in the dynamics of this model.

is called a negative attractor.

Duffing's model

To illustrate the different types of attractors, a physical model of a metal beam suspended over two magnets will be used (see figure 2.2.2). The beam may move freely in the horizontal (i.e. x) direction and is magnetically attracted by each of the two magnets. The behaviour of the beam is characterized by the following equation

$$\frac{d^2y}{dt^2} + \theta \frac{dy}{dt} + (y^3 - y) = \gamma \sin \omega t \quad (2.2.1)$$

where the first two terms describe the inertia and friction of the beam and the third term describes the magnetic and elastic forces. The sine term on the right hand side of the equation describes the external force that shakes the apparatus. This nonlinear equation may be rewritten

as a set of first order differential equations as in equation (2.2.2).

$$\begin{aligned}\frac{dx}{dt} &= y \\ \frac{dy}{dt} &= x - x^3 - \theta y - \gamma \sin \omega z \\ \frac{dz}{dt} &= 1\end{aligned}\tag{2.2.2}$$

If $\gamma = 0$ then the system is not shaken and will be stable in one of the two states, i.e. near one of the two magnets. If $\gamma > 0$ then several different types of behaviour can be found. In the examples used in this chapter the following parameter values were used: $\theta = 0.15$, $\gamma = 0.3$ (except in the stable state, $\gamma = 0$) and $\omega = 1$. Initial conditions are for the chaotic behaviour: $x = 0.1$, $y = 0.1$ and $t = 0$

2.3 Stability and attractors

To investigate if a certain point, that is assumed to be critical, is stable, it is possible to look at the behaviour of the system very close to that point. This is done by assuming that very close to the critical point the behaviour of the system for each variable is linear. By calculating the characteristic values at the critical point it can be determined if the point is stable. Also, if a system is in the stable point it will return to that point after a small perturbation, unless the perturbation drives the system to the attracting basin of a different stable attractor. However, in some points, e.g. saddle nodes, the behaviour depends on the size and direction of the perturbation. An unstable point does not attract the evolution but simply drives it away from that point.

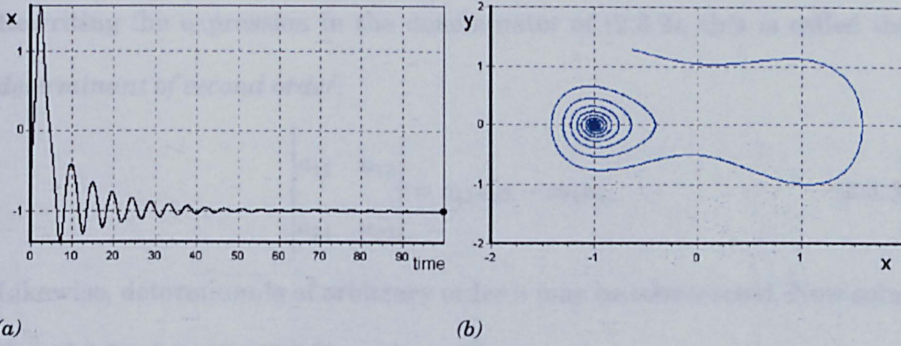


Figure 2.3.3: (a) Stable state, position of the beam x versus evolution, (b) position of the beam x versus the velocity y .

Duffing's model

This is shown in figure 2.3.3 where, depending on the initial position of the beam, it will settle near one of the two magnets in the model. The value of parameter $\gamma = 0$ determines when the system bifurcates from stable to more complex dynamics. A stable point, because it is a single point with no size, has a dimension of zero.

An important concept in this context are the determinant and eigenvalues. These are used to determine how many solutions exist and the stability of each of those solution for a set of linear equations (see [57, Kreyszig]). For example,

$$a_{11}x_1 + a_{12}x_2 = b_1 \quad (2.3.1)$$

$$a_{21}x_1 + a_{22}x_2 = b_2$$

by multiplication and addition to find the roots, the following solution is obtained.

$$x_1 = \frac{b_1 a_{22} - b_2 a_{12}}{a_{11} a_{22} - a_{21} a_{12}}, \quad x_2 = \frac{a_{11} b_2 - a_{21} b_1}{a_{11} a_{22} - a_{21} a_{12}} \quad (2.3.2)$$

Rewriting the expression in the denominator of (2.3.2), this is called the *determinant of second order*.

$$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{21}a_{12} \quad (2.3.3)$$

Likewise, determinants of arbitrary order n may be constructed. Now solution (2.3.2) of system (2.3.1) can be written as

$$x_1 = \frac{D_1}{D}, \quad x_2 = \frac{D_2}{D} \quad (D \neq 0) \quad (2.3.4)$$

where

$$D = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}, \quad D_1 = \begin{vmatrix} b_1 & a_{12} \\ b_2 & a_{22} \end{vmatrix}, \quad D_2 = \begin{vmatrix} a_{11} & b_1 \\ a_{21} & b_2 \end{vmatrix} \quad (2.3.5)$$

If b_1 and b_2 are zero the system is called *homogeneous* and has, in this case, at least the solution $x_1 = 0, x_2 = 0$. If and only if $D = 0$ other solutions may exist. If either or both b_1 and b_2 are non-zero the system is called *nonhomogeneous* and if $D \neq 0$ it has exactly one solution that may be found from (2.3.5).

The basic concept of the eigenvalues is as follows. Supposing that $A = [a_{jk}]$ is a matrix of n by n , consider the vector equation

$$Ax = \lambda x \quad (2.3.6)$$

where λ is a number. For any value of λ the zero vector $x = 0$ is a solution of (2.3.6). Any value of λ for which (2.3.6) has a solution $x \neq 0$ is called an *eigenvalue* (or *characteristic value* or *latent root*). Those solutions are called *eigenvectors* or *characteristic values* of A corresponding with the eigenvalue λ . All eigenvectors that correspond with the eigenvalue of A , including 0,

form the vector space *eigenspace*. Note that eigenvalues can be real and complex.

The relation between eigenvalues and the determinant is that any n by n matrix has at least one and at most n distinct eigenvalues. From (2.3.6) the following relation can be written, with \mathbf{I} as the identity matrix

$$(\mathbf{A} - \lambda \mathbf{I})\mathbf{x} = 0 \quad (2.3.7)$$

A homogeneous system of linear equations has a nontrivial solution if and only if the corresponding determinant of the coefficients is zero, therefore

$$D(\lambda) = \det(\mathbf{A} - \lambda \mathbf{I}) = \begin{vmatrix} a_{11} - \lambda & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} - \lambda & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} - \lambda \end{vmatrix} \quad (2.3.8)$$

Here $D(\lambda)$ is called the *characteristic determinant* and the relation (2.3.8) is called the *characteristic equation* that corresponds with the matrix \mathbf{A} . By developing the $D(\lambda)$ a polynomial of n^{th} degree in λ has been obtained (the *characteristic polynomial* corresponding to \mathbf{A}). From this it is shown that *the eigenvalues of a square matrix \mathbf{A} are the roots of the characteristic equation that correspond with it* (2.3.8).

The type of an isolated critical point P_0 can be determined in most practical cases by *linearization*. Assuming that P_0 is $(0,0)$ and that $F(x,y)$, $G(x,y)$ are continuous and have continuous partial derivatives in the neighbourhood of P_0 . Because P_0 is critical, $F(0,0) = 0$ and $G(0,0) = 0$ and have therefore no constant terms, so the following two dimensional nonlinear

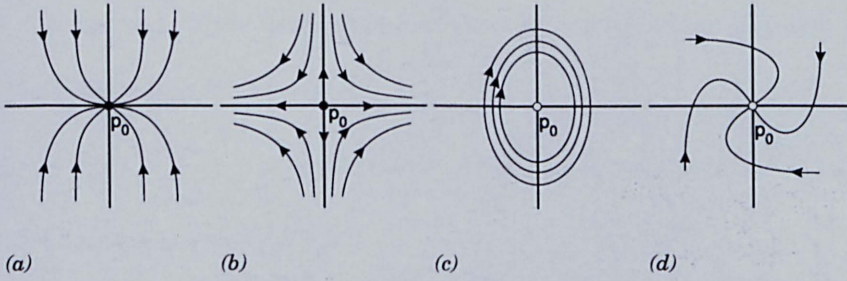


Figure 2.3.4: (a) Stable attractive node, (b) saddle point, (c) center, (d) and spiral or focal point.

system can be written

$$\begin{aligned}\dot{x} &= a_1x + b_1y + F_1(x, y) \\ \dot{y} &= a_2x + b_2y + G_1(x, y)\end{aligned}\tag{2.3.9}$$

If $a_1b_2 - b_1a_2 \neq 0$ then it can be proven that the type and stability of P_0 is the same as that of the critical point $(0, 0)$ of the *linear system* obtained by linearization of (2.3.9) by dropping F_1 and G_1 .

$$\begin{aligned}\dot{x} &= a_1x + b_1y \\ \dot{y} &= a_2x + b_2y\end{aligned}\tag{2.3.10}$$

The linearization assumption implies that both F_1 and G_1 are small near the point P_0 (however, two exceptions exist, if the λ_1 and λ_2 are equal or pure imaginary, apart from the types of linear points, the nonlinear system may have a *spiral point*). Substituting the generic solution $x = Ae^{\lambda t}$ and $y = Be^{\lambda t}$ into (2.3.10), the following homogeneous algebraic equation is obtained

$$\begin{aligned}(a_1 - \lambda)A + b_1B &= 0 \\ a_2A + (b_2 - \lambda)B &= 0\end{aligned}\tag{2.3.11}$$

If and only if the determinant of the coefficients is zero, the non trivial solutions A and B exist, i.e.

$$\lambda^2 - (\mu_1 + \mu_2)\lambda + \mu_1\mu_2 = 0 \quad (2.3.12)$$

Eliminating γ from (2.3.12)

$$-\lambda^2 - (\mu_1 + \mu_2)\lambda + \mu_1\mu_2 = 0 \quad (2.3.13)$$

Now using the following standard results

then if λ_1 and λ_2 are the roots of the characteristic equation of (2.3.12), it follows from that

Therefore, the sum and product of the roots are independent of μ .

Thus, the behavior of the system (2.3.9) is determined by the eigenvalues λ_1 and λ_2 .

Figure 2.3.5: Stability eigenvalues of the system (2.3.9); (a) Node, $\lambda_1, \lambda_2 < 0$ positive attractor (I), $\lambda_1, \lambda_2 > 0$ negative attractor (II); (b) Saddle node, λ_1 and λ_2 are real with different signs; (c) Focus, λ_1 and λ_2 are complex conjugate $\lambda_{1,2} = \mu \pm \omega i$ with $\mu\omega \neq 0$, if $\mu < 0$ positive attracting focus (II), if $\mu > 0$ negative attracting focus (I); (d) Centre, purely imaginary eigenvalues $\lambda_{1,2} = \pm \omega i$ with ω real.

If and only if the determinant of the coefficients is zero, the non trivial solutions A and B exist, i.e.

$$\lambda^2 - (a_1 + b_2)\lambda + a_1b_2 - b_1a_2 = 0 \quad (2.3.12)$$

Eliminating y from (2.3.12)

$$x^n - (a_1 + b_2)x' + (a_1b_2 - b_1a_2)x = 0 \quad (2.3.13)$$

Now using the following standard notation

$$p = a_1 + b_2, \quad q = a_1b_2 - b_1a_2, \quad \Delta = p^2 - 4q \quad (2.3.14)$$

then if λ_1 and λ_2 are the roots or characteristic values or eigenvalues of (2.3.12), it follows from that

$$\begin{aligned} \lambda^2 - p\lambda + q &= \\ (\lambda - \lambda_1)(\lambda - \lambda_2) &= \end{aligned} \quad (2.3.15)$$

$$\lambda^2 - (\lambda_1 + \lambda_2)\lambda + \lambda_1\lambda_2 = 0$$

Therefore, p is the sum and q the product of the roots, and Δ is the *discriminant*. Using this, the behaviour of the solutions A and B near P_0 can be characterized

$$\begin{aligned} &\text{if } p < 0 \text{ and } q > 0 \rightarrow \text{stable and attractive} \\ &\text{if } p \leq 0 \text{ and } q > 0 \rightarrow \text{stable} \\ &\text{if } p > 0 \text{ and } q < 0 \rightarrow \text{unstable} \end{aligned} \quad (2.3.16)$$

Also P_0 is

$$\begin{aligned} &\text{a node if } p < 0 \text{ and } q > 0 \\ &\text{a saddle point if } q < 0 \\ &\text{a center or centre if } p = 0 \text{ and } q > 0 \\ &\text{a spiral point or focus if } p \neq 0 \text{ and } \Delta < 0 \end{aligned} \quad (2.3.17)$$

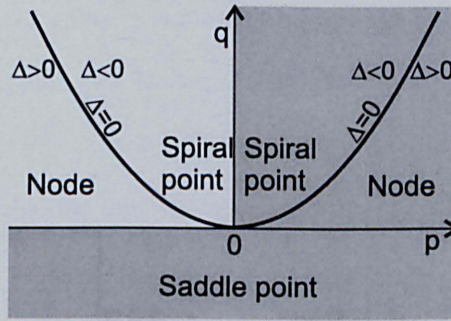


Figure 2.3.6: Stability chart of the system (2.3.9), the unstable regions are shaded.

These criteria can be obtained as follows. If $q = \lambda_1 \lambda_2 > 0$, both the roots (λ_1, λ_2) are both positive or both negative or complex conjugate. If in addition to that $p = \lambda_1 + \lambda_2 < 0$, both roots are negative or have a negative real part. Therefore P_0 is stable and attractive, a similar reasoning may be done for the other two types in (2.3.16).

If the *discriminant* $\Delta < 0$ then the roots are complex conjugate (e.g. $\lambda_1 = \mu + i\omega$ and $\lambda_2 = \mu - i\omega$). If $p = \lambda_1 + \lambda_2 = 2\alpha < 0$, as well, then P_0 is a spiral point that is stable and attractive. If $p = 0$, then $\lambda_2 = -\lambda_1$ and $q = \lambda_1 \lambda_2 = -\lambda_1^2$. If $q > 0$ also, then $\lambda_1^2 = -q < 0$, so that both λ_1 and λ_2 are imaginary. This makes P_0 a center around which periodic solutions exists. Figure 2.3.4 on page 28 illustrates the types of dynamics that are possible. In 2.3.5 on page 29 the different types of stability is plotted with eigenvalues. The resulting stability diagram is plotted in 2.3.6.

In the case that the dimensionality of (2.3.9) is higher, e.g. $n = 3$, the number of possible cases increases quickly. The eigenvalues of a three-dimensional linear system, λ_1, λ_2 and λ_3 can be real or one real and two complex conjugate. If there are three real eigenvalues as in figure 2.3.7,

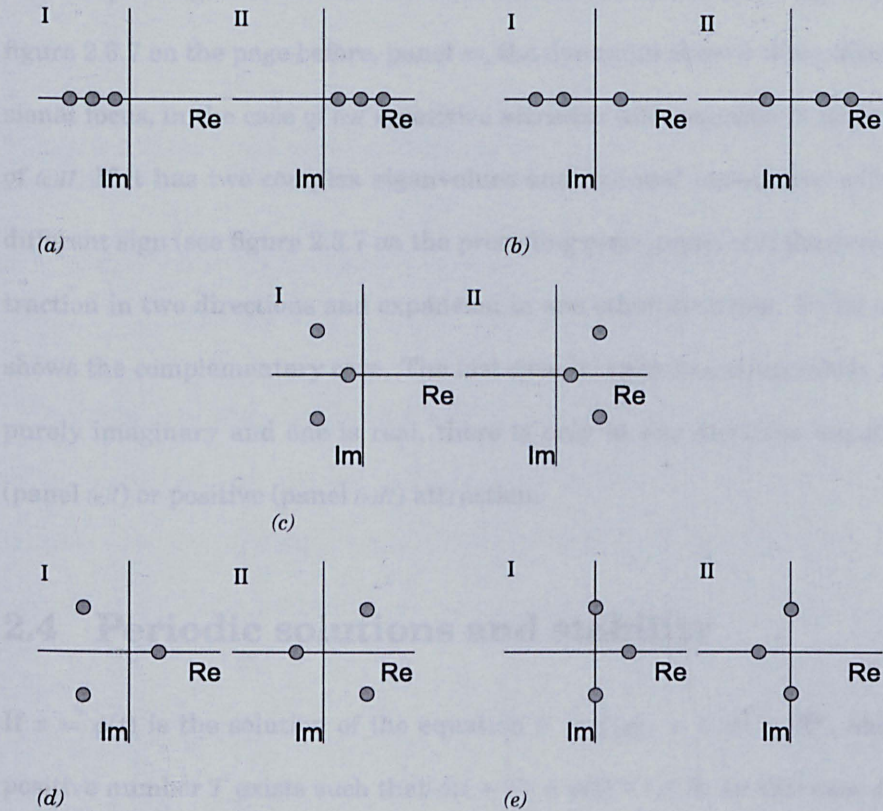


Figure 2.3.7: Stability eigenvalues of the three dimensional system (2.3.9); (a) Three dimensional node with three real values $\lambda_1, \lambda_2, \lambda_3 < 0$ or $\lambda_1, \lambda_2, \lambda_3 > 0$; (b) Saddle node, one of the three values is of different sign than the other two; (c) Three dimensional focus, λ_2, λ_3 and λ_1 with λ_1 and $Re\lambda_2, \lambda_3$ the same sign, also possible is the case $Re\lambda_2, \lambda_3 < \lambda_1 < 0$; (d) Two complex eigenvalues λ_2, λ_3 with λ_1 and $Re\lambda_2, \lambda_3$ of different sign, the case in panel I has attraction in one direction and expansion in two other directions, panel II has complementary behaviour; (e) Two eigenvalues purely imaginary, only in one direction is attraction or expansion.

panel (a), the critical point is a three dimensional node, attracting in the case of (a)I and repelling in the case (a)II. Panel (b) of figure 2.3.7 on the preceding page shows the case of a saddle node. If the critical point has two complex eigenvalues and one real eigenvalue of the same sign as in figure 2.3.7 on the page before, panel (c), the dynamics show a three dimensional focus, in the case of (c)I a positive attractor and negative in the case of (c)II. If it has two complex eigenvalues and one real eigenvalue with a different sign (see figure 2.3.7 on the preceding page, panel (d)I) there is attraction in two directions and expansion in one other direction. Panel (d)II shows the complementary case. The last case is when two eigenvalues are purely imaginary and one is real, there is only in one direction negative (panel (e)I) or positive (panel (e)II) attraction.

2.4 Periodic solutions and stability

If $x = \phi(t)$ is the solution of the equation $\dot{x} = f(x)$, $x \in D \subset \mathbb{R}^n$, and a positive number T exists such that $\phi(t + T) = \phi(t) \forall t \in \mathbb{R}$. In this case $\phi(t)$ is a periodic solution of the equation with period T , although the solutions with period $n T$ where $n = 2, 3, \dots$ exist as well (sometimes the case where $T \geq 0$ is included in the periodic solution). Because the solution will have the same value after T period, the solution is a *cycle*.

Duffing's model

Figure 2.4.8 on the next page illustrates the periodicity of the Duffing model where, depending on the initial position of the beam, it will oscillate between the two magnets in the model. Displayed is the limit cycle

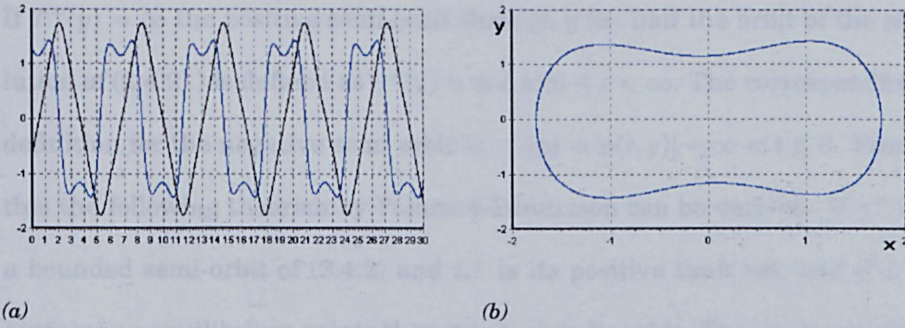


Figure 2.4.8: (a) Limit cycle, position of the beam x and velocity y versus evolution, (b) position of the beam x versus the velocity y .

without transients and parameter values $\theta = 0.15$, $\gamma = 0.3$ and $\omega = 1.0$.

Autonomous two dimensional systems with a phase space of \mathbb{R}^2 are called plane (planar) systems. Closed orbits in the planar system do not intersect because of uniqueness. A theorem by Bendixson says that if the domain of $D \subset \mathbb{R}^2$ is continuous or simply connected then f and g of

$$\dot{x} = f(x, y) \quad (2.4.1)$$

$$\dot{y} = g(x, y)$$

in the domain $D \subset \mathbb{R}^2$, are continuously differentiable in the domain D . The equation (2.4.1) has only periodic solutions if $\nabla \cdot (f, g)$ changes sign in D or if $\nabla \cdot (f, g) = 0$ in D . This theorem can not be generalized to systems of \mathbb{R}^n with $n \geq 3$. Next consider the second order autonomous system

$$\dot{x} = f(x) \quad (2.4.2)$$

where $f : D \rightarrow \mathbb{R}^2$ and $D \in \mathbb{R}^2$ is continuous and contains the origin $x = 0$. If we set $\phi(t, y)$ to be the solution of (2.4.2) beginning at $\phi(0, y) = y$ and the maximal interval in forward time is $[0, T^+(y))$ and reverse time $(-T^-(y), 0]$.

If $T^+(y) = \infty$ the positive semi-orbit through y (as half the orbit of the solution of (2.4.2)) is defined as $\gamma^+(y) = \phi(t, y) | 0 \leq t < \infty$. The corresponding definition for the negative semi orbit is $\gamma^-(y) = \phi(t, y) | -\infty < t \leq 0$. From this the following theorem by Poincaré-Bendixson can be derived. If γ^+ is a bounded semi-orbit of (2.4.2) and L^+ is its positive limit set, and if L^+ contains no equilibrium points then it is a periodic orbit. The same goes for the negative semi orbit. Also, if the positive limit points of γ^+ denoted by $\omega(\gamma^+)$ (ω -limit set) is not equal to γ^+ (i.e. $\omega(\gamma^+) \neq \gamma^+$) the periodic orbit is called a limit cycle.

To say something about the stability of a periodic orbit, consider the following system

$$\dot{x} = f(t, x), \quad x \in \mathbb{R}^n, \quad t \in \mathbb{R} \quad (2.4.3)$$

where $f(t, x)$ is continuous in both t and x . If it is assumed that $x = 0$ is a critical point of the vector function $f(t, x)$ then $f(t, 0) = 0, t \in \mathbb{R}$. In non autonomous equations, critical points are rare, but not in autonomous systems. A neighbourhood $D \subset \mathbb{R}^n$ of (2.4.1) with $x = 0$, the solution that begins at $t = t_0$ in $x = x_0 \in D$ is indicated by $x(t; t_0, x_0)$. The solution $x = 0$ is then called stable in the sense of Lyapunov (Lyapunov-stable) if for each $\epsilon > 0$ and $t_0, \delta(\epsilon, t_0) > 0$ can be found such that $\|x_0\| \leq \delta$ yields $\|x(t; t_0, x_0)\| \leq \epsilon$ for $t \geq t_0$. If a solution can not be determined to be stable in the sense of Lyapunov then the solution is unstable. Now consider the same system (2.4.3) with periodic solutions $\phi(t)$, it is Lyapunov-stable if for each t_0 and $\epsilon > 0$ a $\delta(\epsilon, t_0) > 0$ can be found such that

$$\|x_0 - \phi(t_0)\| \leq \delta \Rightarrow \|x(t; t_0, x_0) - \phi(t)\| \leq \epsilon \quad (2.4.4)$$

for $t \geq t_0$. However, a periodic solution that is Lyapunov-stable is an exceptional case, this implies that orbits that start in a neighbourhood of the periodic solution remain in the neighbourhood, but in the sense that phase points that begin close together, stay together. Usually, periodic solutions are tested using linear perturbation.

A separate case that needs to be mentioned are equations with periodic coefficients of the type

$$\dot{x} = A(t)x \text{ with } t \in \mathbb{R} \quad (2.4.5)$$

where A is a continuous $n \times n$ matrix with period T . This means that $A(t+T) = A(t)$, $t \in \mathbb{R}$. Equations of this type are, for example, $\dot{x} = a(t)x$ with $a(t) = \sin^2 t$ and $t \in \mathbb{R}$. The fundamental result for those equations is given by the Floquet theorem, namely that the fundamental matrix of (2.4.5) can be written as the product of a generally non-periodic matrix and a T -periodic matrix. Thus, the fundamental matrix $\Phi(t)$ can be written as the product of the two $n \times n$ matrices $P(t)$ and C

$$\Phi(t) = P(t)e^{Ct} \quad (2.4.6)$$

where C is a constant $n \times n$ matrix and $P(t)$ a periodic matrix with period T . The fundamental matrices $\Phi(t)$ and $\Phi(\tau) = \Phi(t+T)$ are linear dependent and there exists therefore a nonsingular $n \times n$ matrix F

$$\Phi(t+T) = \Phi(t)F \quad (2.4.7)$$

For this a constant $n \times n$ matrix B exists such that

$$F = e^{BT} \quad (2.4.8)$$

The matrix F is called the *monodromy matrix* of (2.4.5). The eigenvalues ρ of F are the *Floquet multipliers* or *characteristic multipliers*. For every complex number λ that satisfies $\rho = e^{\lambda T}$ is called a *Floquet exponent* or *characteristic exponent*. The imaginary parts of the floquet exponent λ are not determined uniquely since $\frac{2\pi i}{T}$ can be added to them. The floquet multipliers are determined uniquely such that the exponents λ are chosen to coincide with the eigenvalues of the matrix B . No general methods are available to calculate the matrix $P(t)$ or the floquet exponents and multipliers. Therefore, each equation requires a special study. The stability of the trivial solutions and the existence of periodic solutions are determined by the eigenvalues of matrix B . If and only if $Im(\lambda) \neq 0$ and $Re(\lambda) = 0$ then T -periodic solutions exist with modulus $\rho = 1$. If $Re(\lambda) < 0$ then the trivial solution is asymptotically stable with modulus $\rho < 1$. If $Re(\lambda) \leq 0$ then the trivial solution is stable, provided that $Re(\lambda) = 0$ has a multiplicity of one.

2.5 Poincaré map

A useful tool to understand the stability of periodic orbits is the return map or Poincaré map. An n^{th} order continuous autonomous system is replaced by an $(n - 1)^{th}$ order discrete system. If γ is the periodic orbit of the n^{th} order system

$$\dot{x} = f(x) \text{ in } \mathbb{R}^n \quad (2.5.1)$$

and p is a point on γ , H is an $(n - 1)$ dimensional hyperplane at the point p . A hyperplane is defined as a surface $a^T(x - p) = 0$ for some $a \in \mathbb{R}^n$. It is supposed that the hyperplane H is transversal to γ at p or $a^T f(p) \neq 0$. For

example, if $n = 2$ such a hyperplane is any line through the point p that is not a tangent of γ at p . Next, let $S \subset H$ be a local section such that $p \in S$ and $a^T f(x) \neq 0$ for all $x \in S$. The trajectory that begins at p will cross S at p at the period T of the periodic orbit. Because solutions are continuous with respect to initial states, all trajectories that begin on S in a sufficient small neighbourhood of the point p will intersect S in the vicinity of p after period T . Now if $U \subset S$ is a sufficiently small neighbourhood of p such that the orbit γ intersects U only at p , the Poincaré map $g : U \rightarrow S$ is defined for a point $x \in U$ as

$$g(x) = \phi(\tau, x) \quad (2.5.2)$$

with $\phi(t, x)$ is the solution of (2.5.1) that begins at x at time $t = 0$ and $\tau = \tau(x)$ is the time that is required for the trajectory that begins at x to return to the hyperplane S . τ does not generally depend on x and therefore does not need to be equal to T , the period of γ . Also, the Poincaré map is only locally defined, so it is not necessarily defined for all $x \in S$. The solution of the discrete $(n - 1)$ dimensional system

$$x^{(k+1)} = g(x^{(k)}) \quad (2.5.3)$$

is given by the sequence $x^{(k)}$ as long as $x^{(k)} \in U$, then $x^{(k+1)} = g(x^{(k)})$. Even though the vector x is in n dimensions, the solution that is formed by (2.5.3) is limited to the $(n - 1)$ dimensional hyperplane H (see figure 2.5.9 on the following page).

This gives the opportunity to rephrase the stability of periodic solutions as stability of the Poincaré map in the neighbourhood of a fixed point p . The system (2.5.1) has a periodic solution $\phi(t)$, a transversal S and Poincaré

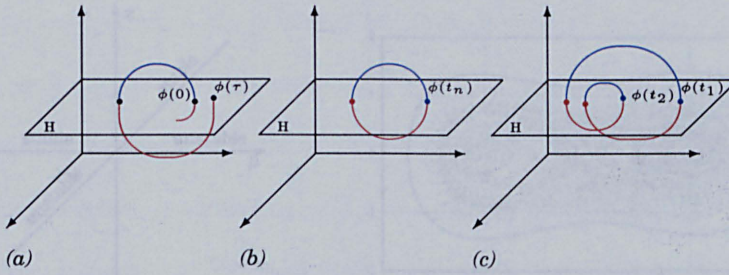


Figure 2.5.9: A return or Poincaré map. (a) The surface of section H is intersected by the trajectory. (b) Period-1 orbit and (c) period-2 orbit.

map U with fixed point p . The solution $\phi(t)$ is stable if for each $\epsilon > 0$ a $\delta(\epsilon)$ may be found such that

$$\|x_0 - p\| \leq \delta, \quad x_0 \in S \Rightarrow \|U^n(x_0) - p\| \leq \epsilon, \quad n = 1, 2, 3, \dots \quad (2.5.4)$$

This is sometimes referred to as orbitally stable.

2.6 Bifurcation

When system parameters are modified they change the behaviour of the dynamical system. This can sometimes be a small change in the behaviour but the qualitative behaviour can remain the same. Sometimes, however, the behaviour may change profoundly. Different types of attractors may appear and disappear. The change of the nature of a critical point when a parameter passes a certain value is called a bifurcation. This is generally used in literature to refer to stability changes and structural changes of a system. This may be readily recognized when a system's equilibrium solution is plotted as a function of a bifurcating system parameter (e.g. left panel (a) of figure 2.6.10 on the next page).

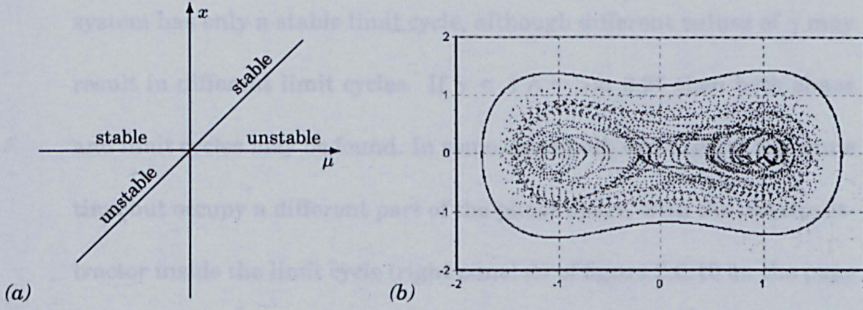


Figure 2.6.10: (a) A bifurcation diagram. (b) Co-existence of chaos and a limit cycle in the Duffing equation.

Consider, for example, the equation

$$\dot{x} = \mu x - x^2 \quad (2.6.1)$$

this equation has an trivial equilibrium solution $x = 0$ and another $x = \mu$. These two solutions coalesce if $\mu = 0$, for any other value of μ a non-trivial solution branches off from $x = 0$. When the value of μ changes from, e.g., negative to nil to positive, the solution changes, hence $\mu = 0$ is called a bifurcation point of (2.6.1). The corresponding bifurcation diagram is plotted in the left panel of figure 2.6.10. Using the implicit function theorem, the bifurcation point of (2.6.1) may be predicted. The solutions x of the equation

$$F(\mu, x) = \mu x - x^2 = 0 \quad (2.6.2)$$

exist and are unique if $\frac{\partial F}{\partial x} \neq 0$, therefore $\mu - 2x \neq 0$. Obviously, the value $\mu = 0, x = 0$ where no uniqueness is guaranteed is a good possibility of being a bifurcation point [111].

Duffing's model

In the model described by the Duffing equation the following dependency exist with the previously used parameter values. If the γ parameter is zero (2.2.2), the system has two stable points, if $\gamma > 1$ then the

system has only a stable limit cycle, although different values of γ may result in different limit cycles. If $\gamma < 1 \wedge \gamma > \approx 0.26$ then both chaos and limit cycles may be found. In some cases both can exist at the same time but occupy a different part of the phase space, with the chaotic attractor inside the limit cycle (right panel (b) of figure 2.6.10 on the page before).

In general, to say something about the behaviour of a system it is necessary to determine possible bifurcations of the equilibrium solutions of the equation

$$\dot{x} = A(\mu)x + f(\mu, x) \quad (2.6.3)$$

with $\mu \in \mathbb{R}^n$, $x \in \mathbb{R}^n$, $\frac{\partial f}{\partial x} \rightarrow 0$ as $\|x\| \rightarrow 0$. To be able to apply this, a classification of the possible bifurcations would be useful. For low dimensional systems such as $n = 1$ and $n = 2$ many results exist; some results for $n \geq 3$ exist but for higher dimensions new types of bifurcations are identified regularly. So, no complete classification of bifurcations is available, but some of the more relevant types is discussed. (Note that for a thorough study of bifurcations an introduction to normalization is in order. Please see any of the textbooks cited at the end of this chapter for a thorough discussion).

For bifurcations of equilibria with one parameter the following cases exist; saddle-node, trans-critical, pitch-fork and Hopf. These are called *co-dimension one bifurcations* and are relatively simple. To demonstrate the use of bifurcation diagrams, consider again equation (2.6.3) with dimension n but now suspended in a $(n + 1)$ dimensional system by adding μ as new

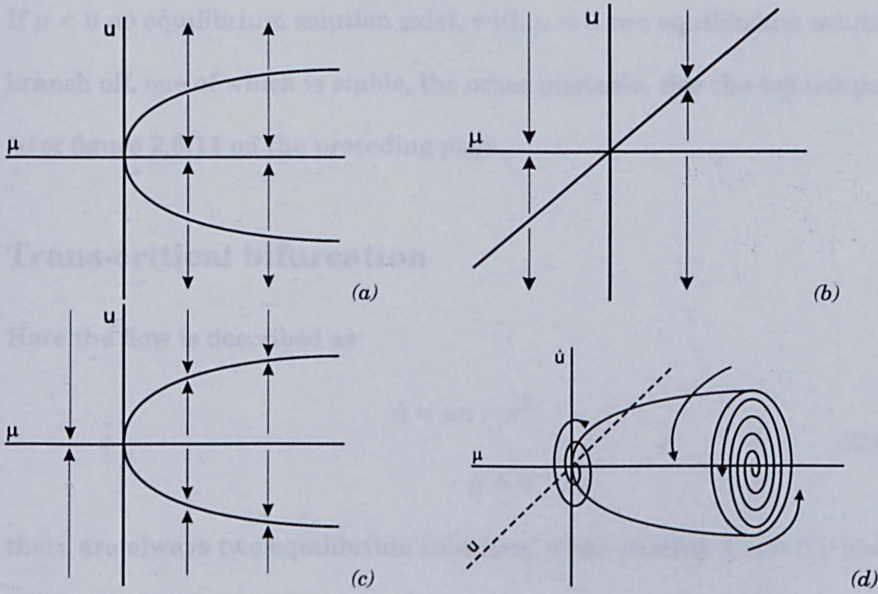


Figure 2.6.11: (a) Saddle-node bifurcation; (b) Transcritical bifurcation; (c) Supercritical Pitch-fork bifurcation; (d) Hopf bifurcation.

variable (as opposed to parameter).

$$\begin{aligned} \dot{x} &= A(\mu)x + f(\mu, x) \\ \dot{\mu} &= 0 \end{aligned} \quad (2.6.4)$$

with $\frac{\partial f}{\partial x} \rightarrow 0$ as $\|x\| \rightarrow 0$ and the possibility of a bifurcation of the solution $x = 0$. If the matrix A has eigenvalues p with real part zero for a certain value of μ then equation (2.6.4) has a $(p + 1)$ -dimensional centre manifold W_e . Normalization and reformulation are required to study this manifold W_e . In the simplest case the manifold W_e is 2-dimensional [111].

Saddle-node bifurcation

In the centre manifold the flow is described as

$$\begin{aligned} \dot{u} &= \mu - u^2 \\ \dot{\mu} &= 0 \end{aligned} \quad (2.6.5)$$

If $\mu < 0$ no equilibrium solution exist, with $\mu = 0$ two equilibrium solutions branch off, one of which is stable, the other unstable. See the top left panel (a) of figure 2.6.11 on the preceding page.

Trans-critical bifurcation

Here the flow is described as

$$\begin{aligned}\dot{u} &= \mu u - u^2 \\ \dot{\mu} &= 0\end{aligned}\tag{2.6.6}$$

there are always two equilibrium solutions, when passing $\mu = 0$ the stability changes type (and $(0, 0)$). See top right panel (b) of figure 2.6.11 on the page before.

Pitch-fork bifurcation

Here the flow is described as

$$\begin{aligned}\dot{u} &= \mu u - u^3 \\ \dot{\mu} &= 0\end{aligned}\tag{2.6.7}$$

If $\mu \leq 0$ there is one equilibrium solution $u = 0$ that is stable. If $\mu > 0$ there are three equilibrium solutions, with $u = 0$ unstable and the two solutions that have branched off at $\mu = 0$ are stable. This pitch-fork bifurcation is supercritical, if the term $-u^3$ is replaced with $+u^3$ the figure is reflected in the u -axis and the bifurcation is called subcritical. See bottom left panel (c) of figure 2.6.11 on the preceding page.

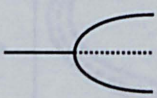
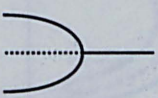


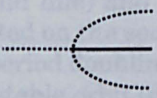
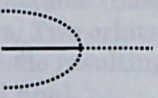
Type of bifurcation	Forward	Backward
(a) Period doubling		
(b) Tangent		
(c) Inverse period doubling		

Table 2.6.1: Generic types of bifurcations of differentiable one dimensional maps. Dashed lines indicate unstable orbits, solid line stable orbits. (a) Period doubling; (b) Tangent bifurcation; (c) Inverse period doubling.

Hopf-bifurcation

This type of bifurcation of periodic solutions is called Hopf-bifurcation after Andronov and Hopf who identified this type of bifurcation explicitly after first being discovered by Poincaré. (He called it a periodic solution “of second type”, solutions of the “first type” are obtained by the continuation method.) Consider the case of equation (2.6.3) where the matrix $A(\mu)$ has two purely imaginary eigenvalues for a value of μ , and the other eigenvalues have a non-vanishing real part. For example, the van der Pol-equation with the value of $\mu \rightarrow 0$

$$\ddot{x} + x = \mu(1 - x^2)\dot{x} \tag{2.6.8}$$

by rescaling $u = \sqrt{\mu}x$

$$\ddot{u} + u = (\mu - \mu^2)\dot{u} \tag{2.6.9}$$

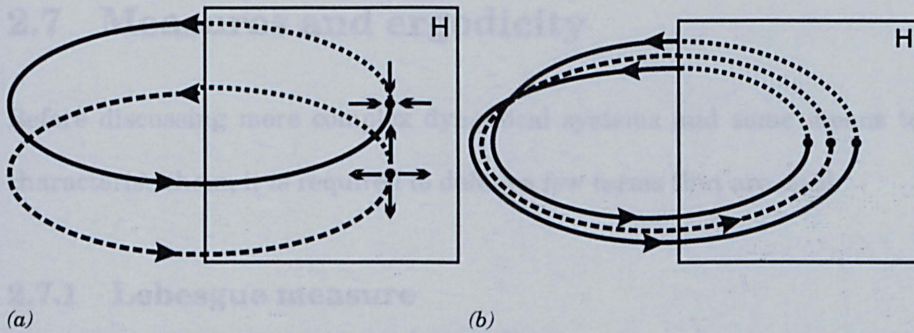


Figure 2.6.12: Period doubling cascade from (a) to (b) with Poincaré section H . (a) Two orbits, one stable (solid line) and one unstable (dashed line), with saddle node directions indicated on the section; (b) Two orbits after the stable orbit from (a) underwent a period doubling cascade resulting in a two period orbit (solid line) and an unstable orbit (dashed line).

The equation (2.6.9) for x has for $\mu > 0$ one periodic solution that has for small values of μ the amplitude $2 + O(\mu)$. Therefore for u the periodic solution branches off with the amplitude $2\sqrt{\mu} + O(\mu^{\frac{3}{2}})$. This behaviour of solutions is typical for Hopf-bifurcations. (See bottom right panel of figure 2.6.11 on page 42).

For chaotic systems the transition from one of the more classical dynamic states to chaos, is complex and has several possible pathways, e.g. periodic doubling cascade and chaotic intermittency. Some generic types of bifurcation is shown in figure 2.6.1 on the preceding page. The effect of a period doubling bifurcation is demonstrated in figure 2.6.12 where a stable one orbit bifurcates into a new stable two orbit and an unstable one orbit that is a continuation of the previous orbit (the forward type (a) bifurcation in figure 2.6.1 on the preceding page).

2.7 Measures and ergodicity

Before discussing more complex dynamical systems and some means to characterise them, it is required to define a few terms that are used.

2.7.1 Lebesgue measure

An *open interval* (a, b) is the set of all real x such that $a < x < b$. A *closed interval* $[a, b]$ is the set of all real x such that $a \leq x \leq b$ with $a < b$. An *interior point* P of a set S is a point such that there exists a neighbourhood of size ϵ around the point $(P - \epsilon, P + \epsilon)$ contained entirely in S . A point P is a *boundary point* of S if any ϵ -neighbourhood of P possesses points that are in S as well as points that are not in S . A point is a *limit point* of the set S if every ϵ -neighbourhood of P contains at least one other point in S (apart from P itself). This is equivalent to an infinite sequence of distinct points x_1, x_2, \dots that are all in S such that $\lim_{n \rightarrow \infty} x_n = P$. If a set contains all its limit points, then this set is said to be a *closed set*. The *closure* \bar{S} of a set S is the set S plus its limit points. And a set S is said to be *open* if all its points are interior points.

Every nonempty bounded open set S_1 can be represented as the sum of a countably infinite or a finite number of disjoint open intervals whose ends do not belong to that set S_1 .

$$S_1 = \sum_k (a_k, b_k) \quad (2.7.1)$$

Then the *Lebesgue measure* of the **open set** S_1 is

$$\mu_L(S_1) = \sum_k (b_k - a_k) \quad (2.7.2)$$

Also a nonempty set S_2 is a closed interval or can be obtained from a closed interval by removing a finite or countably infinite group of disjoint open intervals whose end points belong to S_2 .

$$S_2 = [a, b] - \sum_k (a_k, b_k) \quad (2.7.3)$$

Then the *Lebesgue measure* of the **closed set** S_2 is

$$\mu_L(S_2) = (b - a) - \sum_k (b_k - a_k) \quad (2.7.4)$$

A set has *Lebesgue measure zero* if for any $\epsilon > 0$ the entire set can be covered by a countable union of intervals such that the sum of the length of the intervals is smaller than ϵ . For N -dimensional sets in Cartesian space the analogous definition is that for any $\epsilon > 0$ the set can be covered by a countable union of N -dimensional cubes whose total volume is less than ϵ [73].

A *Cantor set* is a closed set that consists entirely of boundary points, each of which is a limit point of the set. They are uncountable and can have either zero Lebesgue measure or positive Lebesgue measure. A famous example of a Cantor set is the *middle third Cantor set* that is constructed as follows. From a closed interval $[0, 1]$ the open middle third interval $(\frac{1}{3}, \frac{2}{3})$ is removed, this leaves the two intervals $[0, \frac{1}{3}]$ and $(\frac{2}{3}, 1]$. Subsequently, remove the open middle thirds of the last two intervals and this leaves the four intervals $[0, \frac{1}{9}]$, $[\frac{2}{9}, \frac{1}{3}]$, $[\frac{2}{3}, \frac{7}{9}]$ and $[\frac{8}{9}, 1]$ of length $\frac{1}{9}$ each (see figure 2.7.13 on the next page). Continue indefinitely and the remaining set is the middle third Cantor set that has zero Lebesgue measure, since at the n^{th} stage of construction the total length of the remaining intervals is $(\frac{2}{3})^n$ and the length goes to zero as $n \rightarrow \infty$ [73].

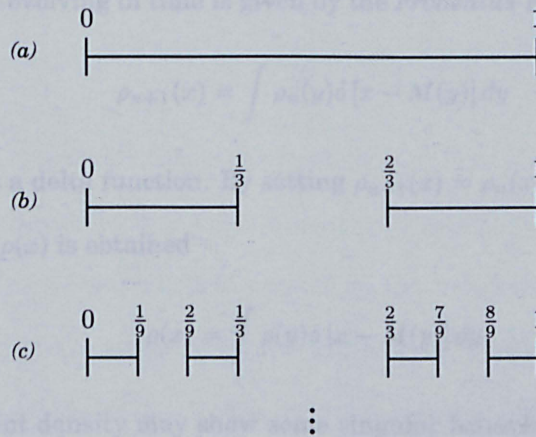


Figure 2.7.13: Construction of the middle third Cantor set, the first three steps are shown, this is continued indefinitely.

2.7.2 Natural measure

In general, a dynamical system may have an abundance of invariant measures most of which possess some complex behaviour. Deciding whether or not a measure reflects the asymptotic behaviour of orbits significantly is a matter to which a lot of study is devoted. For given smooth dynamical systems it can be shown that it has the following smooth positive invariant measure [50].

Imagine a map $M(x)$ with an infinite number of initial conditions along the x -axis, each of which may be started from. The distribution of those initial conditions has a smooth *density* $\rho_0(x)$ such that the fraction of the initial conditions in an interval $[a, b]$ is $\int_a^b \rho_0(x) dx$. After applying the map once to each of all the initial conditions a new density $\rho_1(x)$ is generated. Repeated application of the map will result in subsequent densities $\rho_2(x), \rho_3(x), \dots$

The density evolving in time is given by the *Frobenius-Peron* equation

$$\rho_{n+1}(x) = \int \rho_n(y) \delta [x - M(y)] dy \quad (2.7.5)$$

where $\delta(x)$ is a delta function. By setting $\rho_{n+1}(x) = \rho_n(x) = \rho(x)$ the *invariant density* $\rho(x)$ is obtained

$$\rho(x) = \int \rho(y) \delta [x - M(y)] dy \quad (2.7.6)$$

The invariant density may show some singular behaviour that is common to a typical chaotic one-dimensional map such as the logistic map. It can be expected to become very discontinuous after a few iterations and probably has a dense countable set of x -values at which the invariant density $\rho(x)$ is infinite.

Due to this unfortunate property of the invariant density, it is preferred to use the corresponding measure μ . In general a measure is more generally applicable. A *probability measure* μ for a bounded region R assigns to the set itself the number 1 ($\mu(R) = 1$), positive numbers to any set in R and is countably additive. This means that given any countable group of disjoint (i.e. non overlapping) sets S_i contained in R , the measure of the union of these sets is equal to the sum of the measures of the sets

$$\mu \left(\bigcup_i S_i \right) = \sum_i \mu(S_i) \quad (2.7.7)$$

Define $M^{-1}(S)$ as the set of points of the map M that map onto S after one iteration. If the map M is not invertible $M^{-1}(S)$ is still defined (e.g. if M is the $2x$ modulo 1 map and $S = (\frac{3}{4}, 1)$ then $M^{-1}(S) = (\frac{3}{8}, \frac{1}{2}) \cup (\frac{7}{8}, 1)$). The measure μ is *invariant* if the measure μ is the same for the M on S as the

inverted map on S

$$\mu(S) = \mu(M^{-1}(S)) \quad (2.7.8)$$

If the map is invertible this is the same as $\mu(S) = \mu(M(S))$.

The chaotic attractor of a one-dimensional map has a *basin of attraction* B . This basin of attraction is defined as the closure of the set of initial conditions that are attracted by the attractor. Now given the points of an interval S , the time a trajectory spends, starting from an initial condition x_0 in the basin B , in this interval S in the limit when the orbit length goes to infinity is called $\mu(S, x_0)$. If for every x_0 in the basin of attraction B , $\mu(S, x_0)$ is the same, except for a set of x_0 values of Lebesgue measure zero, then $\mu(S, x_0)$ is called the *natural measure* of S . This means that the natural measure of S is $\mu(S)$ and its value is the common value assumed by $\mu(S, x_0)$ for all the x_0 in B , except for a set of Lebesgue measure zero. Generally, in smooth one-dimensional maps with chaotic attractors natural measures and densities can often be proved. In higher dimensional systems this is still an open problem, but numerically the existence of a natural measure seems fairly clear [73].

2.7.3 Borel measure

There exists an invariant measure for any topological dynamical system on a compact metrizable space [50]. Without going into details we can define a Borel measure μ on a space \mathcal{B} such that $\mu(B) < \infty$ when B is compact. This measure has as main property that it is regular, i.e. for every $B \in \mathcal{B}$ a $\mu(B) = \sup\{\mu(K) | K \subset B_{compact}\}$ exists. Furthermore every continuous function $f : X \rightarrow \mathbb{R}$ is Borel measurable.

To demonstrate that the invariant Borel probability measure exists we take the function $f : X \rightarrow X$ with $x \in X$ and a dense countable set $\{\phi_1, \phi_2, \dots\}$ in space $C(X)$. For each m the sequence $\frac{1}{n} \sum_{k=0}^{n-1} \phi_m(f^k(x))$ is bounded, therefore a convergent subsequence n_k with $k = 1, 2, \dots$ can be found for each $m = 1, 2, \dots$ and the limit

$$J(\phi_m) = \lim_{k \rightarrow \infty} \frac{1}{n_k} \sum_{l=1}^{n_k-1} \phi_m(f^l(x)) \quad (2.7.9)$$

exists. If we let ϕ be an arbitrary continuous function, fix $\epsilon > 0$ and take ϕ_m such that $\sup_{x \in X} |\phi(x) - \phi_m(x)| < \epsilon$ then

$$J(\phi) = \lim_{k \rightarrow \infty} \frac{1}{n_k} \sum_{l=1}^{n_k-1} \phi(f^l(x)) \quad (2.7.10)$$

This functional $J : C(X) \rightarrow \mathbb{R}$ is linear, bounded, positive and invariant. This may be represented by $J(\phi) = \int \phi d\mu_x$ where μ_x is an invariant Borel probability measure [50, 25].

2.7.4 Ergodic theorem

The Birkhoff ergodic theorem deals with transformations of abstract spaces preserving a probability measure. From the section on the natural measure it may be concluded that given a smooth function $f(x)$ and an attractor with a natural measure μ , the average over time of the orbit of $f(x)$ that originates from an initial condition in the basin of the attractor is the same as its natural-measure-weighted average over x (except for a set of Lebesgue measure zero).

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum_{n=0}^T f(M^n(x_0)) = \int f(x) d\mu(x) \quad (2.7.11)$$

where $d\mu(x) = \rho(x)dx$ when a density exists. More generally, any invariant probability measure μ (not necessarily the natural measure) is *ergodic* if it

can *not* be decomposed such that

$$\mu = p\mu_1 + (1 - p)\mu_2 \quad (2.7.12)$$

with $1 > p > 0$ and $\mu_1 \neq \mu_2$ which are two other invariant probability measures. The ergodic theorem states that if $f(x)$ is integrable and μ is an ergodic probability measure then the set A of x_0 values for which the limit of (2.7.11) exists and the equations (2.7.11) holds, has μ measure of 1, i.e. $\mu(A) = 1$. On the other hand, the set of x_0 values for which the equation (2.7.11) does not hold has a μ measure of 0. Further details may be found in [73, 50, 10, Ott, Katok and Birkhoff].

2.8 Quasiperiodicity

Quasiperiodicity is a form of dynamic motion different from the other types, i.e. steady states, periodic and chaotic motion. It is very important in Hamiltonian systems (these are a generalization of the study of differential equations of celestial mechanics) and in dissipative systems quasiperiodic attracting motion occurs frequently. To determine the difference between periodic motion and quasiperiodicity, consider a system of differential equations with a limit cycle. For orbits on the attractor a variable $f(t)$ will oscillate periodically with time, such that some smallest time $T > 0$ exists for $f(t) = f(t + T)$. Therefore, the Fourier transform of $f(t)$ at frequency ω

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(t)e^{i\omega t} dt \quad (2.8.1)$$

consists of the delta function spikes of different strength that are located at integer multiples a_n with $n = 1, 2, \dots$ of the fundamental frequency $\Omega = \frac{2\pi}{T}$,

$$\hat{f}(\omega) = 2\pi \sum_n a_n \delta(\omega - n\Omega) \quad (2.8.2)$$

Basically, quasiperiodic dynamics can be considered to be a mixture of periodic motions of several different fundamental frequencies [73], see for example figure 2.8.14 on the following page. When the number of fundamental frequencies that form the quasiperiodic motion is N , referred to as N -frequency quasiperiodicity; a dynamic variable $f(t)$ can be expressed in terms of a function of N independent variables $G(t_1, t_2, \dots, t_N)$, where G is periodic in *each* of its N independent variables.

$$G(t_1, t_2, \dots, t_i + T_i, \dots, t_N) = G(t_1, t_2, \dots, t_i, \dots, t_N) \quad (2.8.3)$$

The function G has for each of its N variables a corresponding period T_i . These N frequencies $\Omega \equiv \frac{2\pi}{T_i}$ are *incommensurate*, which means that no single frequency Ω_i can be expressed as a linear combination of the other frequencies using coefficients that are rational numbers. The form

$$m_1\Omega_1 + m_2\Omega_2 + \dots + m_N\Omega_N \neq 0 \quad (2.8.4)$$

holds for any possible set of integers m_1, m_2, \dots, m_N with the exception of the trivial solution $m_1 = m_2 = \dots = m_N = 0$. Now the dynamic quasiperiodic variable f may be obtained from the function G by setting all its N variables equal to t , $t_1 = t_2 = \dots = t_N$ so

$$f(t) = G(t, t, \dots, t) \quad (2.8.5)$$

Because G is periodic it can be expressed as a Fourier series of the form

$$G = \sum_{n_1, n_2, \dots, n_N} a_{n_1 \dots n_N} e^{i(n_1\Omega_1 t_1 + n_2\Omega_2 t_2 + \dots + n_N\Omega_N t_N)} \quad (2.8.6)$$

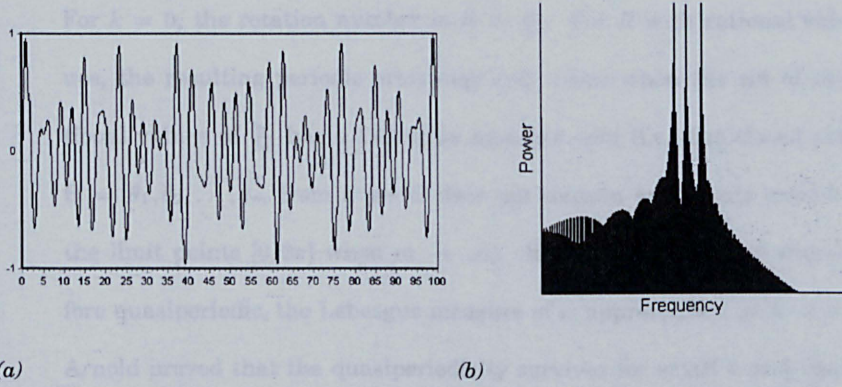


Figure 2.8.14: (a) A quasiperiodic time series constructed from three basic frequencies ($\sqrt{2}$, $\sqrt{3}$ and $\sqrt{5}$); (b) Power spectrum of the time series, showing the three basic frequencies.

Thus from (2.8.5) and (2.8.6) the following expression for f may be obtained

$$\hat{f}(\omega) = 2\pi \sum_{n_1, n_2, \dots, n_N} a_{n_1 \dots n_N} \delta[\omega - (n_1 \Omega_1 + n_2 \Omega_2 + \dots + n_N \Omega_N)] \quad (2.8.7)$$

This explains why the Fourier transform of the dynamical variable $\hat{f}(\omega)$ consists of delta functions of integer linear combinations of the fundamental N frequencies $\Omega_1, \dots, \Omega_N$.

Circle map

An example that is often used in quasiperiodicity demonstrations is the circle map. The circle map has been introduced by Arnold [73] to study the effect of nonlinear oscillator coupling.

$$\theta_{n+1} = (\theta_n + w + k \sin \theta_n) \text{ modulo } 2\pi \quad (2.8.8)$$

Here $k \sin \theta$ models the effect of the nonlinear oscillator coupling and w lies in the range $[0, 2\pi]$. Although the circle map seems very simple, it reveals a wealth of intricate behaviour (like the logistic map), stable states, periodicity and quasiperiodicity are possible. The rotation number is defined to be

$$R = \frac{1}{2\pi} \lim_{m \rightarrow \infty} \frac{1}{m} \sum_{n=0}^{m-1} (w + k \sin \theta_n) \quad (2.8.9)$$

For $k = 0$, the rotation number is $R = \frac{w}{2\pi}$. For R with rational values, the resulting periodic orbit may only occur when the set of rational values of $\frac{w}{2\pi}$ has a Lebesgue measure zero (i.e. the closed set $\Theta = \theta_1, \theta_2, \dots, \theta_m$ from a set w does not contain any points outside the limit points $[0, 2\pi]$ when $m \rightarrow \infty$). If R is irrational and therefore quasiperiodic, the Lebesgue measure of w approaches 1 as $k \rightarrow 0$. Arnold proved that the quasiperiodicity survives for small k and that the set of w values yielding quasiperiodicity is a Cantor set of positive Lebesgue measure. The Cantor set is an uncountable closed set that consists entirely of boundary points each of which is a limit point of the set [73].

The theory of quasiperiodicity is very important in Hamiltonian systems, such as frictionless systems and mixing of fluids. Well-known objects exist in phase space of integrable Hamiltonian systems, such as n -tori. For perturbation of these systems the famous KAM theorem has been devised by Kolmogorov, Arnold and Moser, to determine the prevalence of integrability. This may lead to chaotic maps as well as quasiperiodicity. See for a detailed explanation, e.g. [73, 50, Ott or Katok].

2.9 Chaos

The definition of chaos is complex but is well described in the case of models. Signal analysis of a chaotic signal is more complex and the identification of chaos is only assured in specific types of signals, the difference between noisy and chaotic signals may be lost if a signal is indistinguishable from

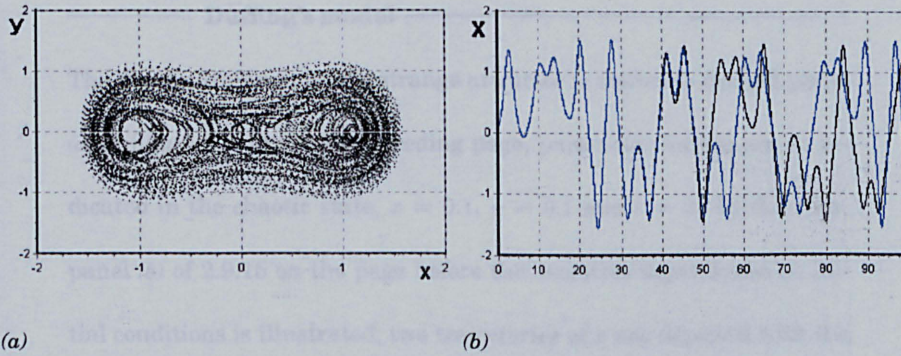


Figure 2.9.15: (a) The strange attractor of the Duffing model, (b) dependence on initial conditions of the Duffing model, numerical divergence exists almost instantaneously, but becomes visible at approximately time step 35.

a random noise. There exist, however, some measures which may indicate the possibility of chaos in the case of low-dimensional chaotic signals. High dimensional chaos is often too complex to enable a numerical algorithm to separate it from noise. The attracting set of a dissipating chaotic system or chaotic attractor is usually referred to as a strange attractors because of the fractal dimension of the attractor, i.e. the attractor has a dimension that is not an integer. The right panel (b) of figure 2.2.1 on page 22 shows the strange attractor of the Rössler equation [90], where at the top of the z -axis the stretching of the attractor, as demonstrated by two trajectories first close together but later expanding away, can be seen. At the x, y plane the folding of the trajectory back into the z -direction is recognisable. Another property of a chaotic system is the dependence on initial conditions. With a very small difference in initial conditions, two identical chaotic systems may diverge away from each other at an exponential rate. The shape of the attractor will remain the same, but the two systems will traverse the phase space differently.

Duffing's model

The chaotic Duffing model's strange attractor is shown in the left panel (a) of figure 2.9.15 on the preceding page, parameter values are as indicated in the chaotic state, $x = 0.1$, $y = 0.1$ and $t = 0$. In the right panel (b) of 2.9.15 on the page before the sensitive dependence on initial conditions is illustrated; two trajectories of x are depicted with the same parameter values but with different initial conditions. For one trajectory the initial condition is $x = 0.1$ and for the other $x = 0.10001$. The initial values of y and t are the same in both cases. (Note that even though the difference is still fairly large, i.e. 0.01%, the sensitive dependence on initial conditions is valid, but with a smaller initial difference between the two trajectories, the divergence would take much longer before the effect becomes visible.)

2.10 Dynamical characterizations

2.10.1 Embedding

Quite often one may want to know something about the minimum number of variables (i.e. the dimension) required to describe a dynamical system that may be defined by a time series G . Construct a m -dimensional vector

$$\mathbf{y} = (g(t), g(t - \tau), g(t - 2\tau), \dots, g(t - (m - 1)\tau)) \quad (2.10.1)$$

Provided that m is large enough and the attractor is finite then there exists some dynamical system that describes the evolution of the vector \mathbf{y} . This

vector y is often called the Takens vector. The low-dimensional system that produced the series G is assumed to be smooth and is described by, say,

$$\frac{dx}{dt} = F(x) \quad (2.10.2)$$

where the system x has some dimensionality D . The observed data points G may be regarded as a smooth function of the state variable x and y is therefore a function of x :

$$y = H(x) \quad (2.10.3)$$

The state of the underlying dynamical system is given by x and knowing x is sufficient to evolve the system by (2.10.2). For a dynamical system to evolve the *delay coordinate vectors* y , it is sufficient that the function (2.10.3) is such that a unique system state x' exist where $y' = H(x')$. Therefore the state y' is determined by x' and vice versa, as well as the state x' is uniquely determined by the delay coordinates $y' = H(x')$. The function H must satisfy the condition that $x \neq x'$ and implies

$$H(x) \neq H(x') \quad (2.10.4)$$

If this condition holds then the function H is called the *embedding* of the D -dimensional x -space into the m -dimensional y -space [73, 101]. It was shown by Takens that the minimum embedding dimension m is generically [101, 102]

$$m \geq 2D + 1 \quad (2.10.5)$$

2.10.2 Dimensions

Several different means have been defined to be able to characterise fractal sets. The concept of dimensionality has in the past been used intuitively, it uses the idea of “covering”, i.e. a non-empty bounded set $X \subset \mathbb{R}^m$ in Cartesian space and m -dimensional cubes with sides ϵ . The set X may be covered by the cubes using $M(\epsilon)$ number of cubes. Generally, M will increase with ϵ , unless X has a finite number of points, in which case M becomes constant as $\epsilon \rightarrow 0$. This brings us to the limit capacity or box-counting dimension, usually denoted as D_{cap} or D_0 .

$$D_0 = \lim_{\epsilon \rightarrow 0} \frac{\ln M(\epsilon)}{\ln \frac{1}{\epsilon}} \quad (2.10.6)$$

assuming that this limit exists. D_0 represents a geometric dimension of the set $X \subset \mathbb{R}^m$. It is obvious that if X consists of a finite number of points $D_0 = 0$ and $D_0 \leq m$ [111].

The box-counting dimension does have its limitations if, for example, a set X has an infinite sequence of points on the real line $1, \frac{1}{2}, \frac{1}{3}, \dots$, this will give a box-counting dimension as given by equation (2.10.6) of $D_0 > 0$. Obviously, it ought to be $D_0 = 0$ for this discrete set of points. A different method of estimating the embedding dimension is the so called *Hausdorff* dimension. This does produce the correct result of the above mentioned problem, but for the typical invariant sets used in practice in chaotic dynamics both methods are thought to be equal. This means that for the box-counting and the Hausdorff dimension for most fractal sets the *exact* dimension is unknown and very difficult to determine.

Let us define the *Hausdorff measure*. A is a set in N -dimensional Carte-

sian space, the *diameter* of A is denoted as $|A|$, i.e. the largest distance between two points x and y in A .

$$|A| = \sup_{x,y \in A} |x - y| \quad (2.10.7)$$

Now S_i is a countable collection of subsets of the Cartesian space such that the diameters ϵ_i of S_i are all less or equal to δ

$$0 < \epsilon_i \leq \delta \quad (2.10.8)$$

and the S_i cover A , so $A \subset \bigcup_i S_i$. The quality $\Gamma_H^d(\delta)$ is then defined as

$$\Gamma_H^d(\delta) = \inf_{S_i} \sum_{i=1}^d \epsilon_i^d \quad (2.10.9)$$

This means that the collection of covering sets S_i with diameters less or equal to δ that minimises the sum of (2.10.9) is the set that is desired as the minimized sum $\Gamma_H^d(\delta)$. The d -dimensional Hausdorff measure is defined as

$$\Gamma_H^d = \lim_{\delta \rightarrow 0} \Gamma_H^d(\delta) \quad (2.10.10)$$

The Hausdorff measure generalises the concept of total length, area and volume of simple sets. For example, if the set A is a smooth finite surface in a three dimensional Cartesian space, then Γ_h^2 is just the area of the set, Γ_h^d for $d < 2$ is $+\infty$ and Γ_h^d for $d > 2$ is zero [73]. In general, Γ_h^d is $+\infty$ if d is less than some critical value D_H and Γ_h^d is zero if d is greater than that value. The critical value D_H is called the *Hausdorff dimension* of that set. The value of Γ_h^d at $d = D_H$ can be zero, a finite positive number or $+\infty$. It can be shown that the relation between the box-counting dimension and the Hausdorff dimension is

$$D_0 \geq D_H \quad (2.10.11)$$

So for typical chaotic attractors both dimensions are generally thought to be the same. Some particular sets for which $D_0 \neq D_H$ may be constructed but these do not seem to arise among the invariant sets of typical dynamical systems.

The box-counting dimension gives the scaling of the number of cubes required to cover the attractor. Unfortunately, with strange attractors, the frequency with which any given cube is visited may differ vastly. For small ϵ only a small percentage of the cubes needed to cover the attractor contain the vast majority of the natural measure on the attractor, i.e. only a few cubes are visited by the trajectory continuously. The box-counting dimension counts all cubes equally, regardless of the fact that some cubes are visited more often than others. To take this into account, a generalized box-counting dimension has been formulated by Grassberger, Hentschel and Procaccia [73]. The dimension D_q depends on a continuous index q

$$D_q = \frac{1}{1-q} \lim_{\epsilon \rightarrow 0} \frac{\ln I(q, \epsilon)}{\ln(\frac{1}{\epsilon})} \quad (2.10.12)$$

where

$$I(q, \epsilon) = \sum_{i=1}^{\hat{N}(\epsilon)} \mu_i^q \quad (2.10.13)$$

and the sum is over all the $\hat{N}(\epsilon)$ cubes in a grid of unit size ϵ needed to cover the attractor. Here the $q > 0$ cubes with larger μ_i have a greater effect in determining the value of D_q . If $q = 0$, $I(0, \epsilon) = \hat{N}(\epsilon)$ that is the same as the box-counting dimension definition. In the special case where all the μ_i are equal, $\mu_i = \frac{1}{\hat{N}(\epsilon)}$ and $\ln I(q, \epsilon) = (1-q) \ln \hat{N}(\epsilon)$ which is the box-counting dimension independent from q . So the *information dimension* is defined as

(from (2.10.12))

$$D_1 = \lim_{q \rightarrow 1} D_q = \lim_{\epsilon \rightarrow 0} \frac{\sum_{i=1}^{\hat{N}(\epsilon)} \mu_i \ln \mu_i}{\ln \epsilon} \quad (2.10.14)$$

In general it may be shown that

$$D_{q_1} \leq D_{q_2} \quad \text{if} \quad q_1 > q_2 \quad (2.10.15)$$

so that, for example, D_2 provides a lower bound for D_1 and D_1 provides a lower bound for D_0 [73].

A different concept of dimension that is useful for the study of dynamical systems is the *pointwise dimension* $D_p(\mathbf{x})$. If a N -dimensional ball with radius ϵ is centred at a point \mathbf{x} is denoted $B_\epsilon(\mathbf{x})$ (left panel (a) of figure 2.10.16 on the following page) then the pointwise dimension of a probability measure μ at \mathbf{x} is defined as

$$D_p(\mathbf{x}) = \lim_{\epsilon \rightarrow 0} \frac{\ln \mu(B_\epsilon(\mathbf{x}))}{\ln \epsilon} \quad (2.10.16)$$

It may be argued that given a certain constraint on μ [73], $D_p(\mathbf{x})$ assumes a single common value \bar{D}_p for all locations \mathbf{x} . The \mathbf{x} values yielding this common value form the *core region* of the measure whose box-counting dimension is $D_0(\theta) = D_1$ where $0 < \theta < 1$. Some results imply that the value \bar{D}_p of $D_p(\mathbf{x})$ assumed for “almost every” \mathbf{x} with respect to the measure μ is $D_0(\theta)$ with $0 < \theta < 1$, i.e. all \mathbf{x} except for the set of μ measure zero [73]

$$\bar{D}_p = D_0(\theta) = D_1 \quad (2.10.17)$$

The constraint imposed is that the measure μ is *ergodic* (see section 2.7.4 on page 51). Assuming that the natural measure on a chaotic attractor exists, it is necessarily ergodic because it can be constructed from the long-time limit of the frequency of a single typical orbit visiting regions of phase

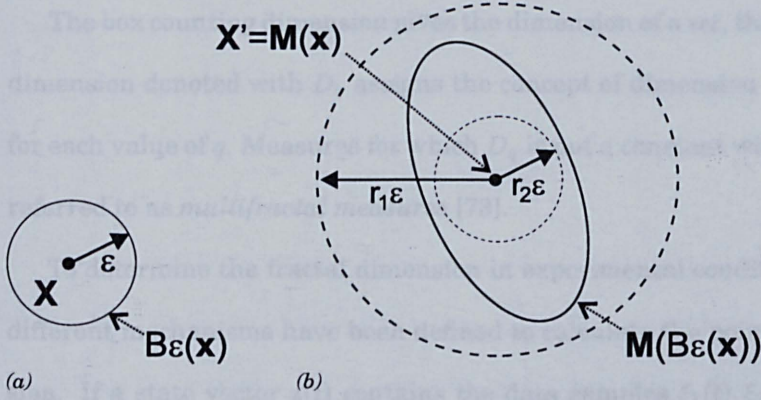


Figure 2.10.16: (a) The ball $B_\epsilon(x)$, (b) map of a small ball $B_\epsilon(x)$.

space. To demonstrate that the pointwise dimension $D_p(x)$ assumes one common value for almost every set of x with respect to the ergodic measure μ , it is argued that

$$D_p(x) = D_p(x') \quad \text{with} \quad x' = M(x) \quad (2.10.18)$$

i.e. the pointwise dimension at x and its first iterate under the map are the same. Because the measure μ is invariant the invertible M has the relation $\mu(B_\epsilon(x)) = \mu(M(B_\epsilon(x)))$. If ϵ is small and the map smooth, the region $B_\epsilon(x)$ is mapped by M to an ellipsoidal region around $x' = M(x)$ (right panel (b) of figure 2.10.16). The constants $r_1 > r_2$ are defined such that the ball $B_{r_1\epsilon}(x')$ contains $M(B_\epsilon(x))$ which in turn contains $B_{r_2\epsilon}(x')$.

Therefore

$$\mu(B_{r_1\epsilon}(x')) \geq \mu(M(B_\epsilon(x))) = \mu(B_\epsilon(x)) \geq \mu(B_{r_2\epsilon}(x')) \quad (2.10.19)$$

From (2.10.16) and (2.10.19) this results in

$$D_p(x') = \lim_{\epsilon \rightarrow 0} \frac{\ln[\mu(B_{r_1,2\epsilon}(x'))]}{\ln(r_{1,2}\epsilon)} = \lim_{\epsilon \rightarrow 0} \frac{\ln[\mu(B_{r_1,2\epsilon}(x'))]}{\ln \epsilon} \quad (2.10.20)$$

from which follows $D_p(x') \geq D_p(x) \geq D_p(x')$ or $D_p(x) = D_p(x')$.

The box counting dimension gives the dimension of a *set*, the spectrum of dimension denoted with D_q assigns the concept of dimension to a *measure* for each value of q . Measures for which D_q is not a constant with q are often referred to as *multifractal measures* [73].

To determine the fractal dimension in experimental conditions, several different mechanisms have been defined to calculate the pointwise dimension. If a state vector $\mathbf{z}(t)$ contains the data samples $\xi_1(t), \xi_2(t), \dots, \xi_n(t)$; a large number of points on the attractor can be obtained by sampling $\mathbf{z}(t)$ at discrete time intervals T so that $\mathbf{z}_0 = \mathbf{z}(t_0), \mathbf{z}_1 = \mathbf{z}(t_0 + T), \mathbf{z}_2 = \mathbf{z}(t_0 + 2T), \dots, \mathbf{z}_K = \mathbf{z}(t_0 + KT)$. One of the points \mathbf{z}_j is then selected as reference point \mathbf{z}_* and the distances $d_k = |\mathbf{z}_k - \mathbf{z}_*|$ are then calculated from \mathbf{z}_* to all the other K points. The resulting distances d_k are sorted in ascending order by size into an array $d_1 \leq d_2 \leq \dots \leq d_i \leq \dots \leq d_K$. The distance at d_i gives a value of $\epsilon = d_i$ such that

$$\mu(B_\epsilon(\mathbf{Z}_*)) \simeq \frac{i}{K} \quad (2.10.21)$$

The quantity $\ln \frac{i}{K}$ may then be plotted as a function of ϵ . In some range of ϵ the points may lie approximately on a straight line. The slope of this straight line is fitted to produce a dimensional estimation. This method is sensitive to noise, number of data points etc. .

For experimental estimation of a dimension the method of the *correlation dimension* or D_2 as proposed by Grassberger and Procaccia in 1983, is very useful and often used. Recall from (2.10.12) and (2.10.13) that for D_2

$$D_2 = \lim_{\epsilon \rightarrow 0} \frac{\ln I(2, \epsilon)}{\ln \frac{1}{\epsilon}} \quad (2.10.22)$$

$$I(2, \epsilon) = \sum_{i=1}^{\tilde{N}(\epsilon)} \mu_i^2$$

that must be estimated for different values of ϵ . The set of points \mathbf{z}_k of the attractor are then used to compute the *correlation integral* $C(\epsilon)$:

$$C(\epsilon) = \lim_{K \rightarrow \infty} \frac{1}{K^2} \sum_{i,j}^K H(\epsilon - |\mathbf{z}_i - \mathbf{z}_j|) \quad (2.10.23)$$

where the function H denotes the unit or Heaviside step function ($H(z < 0) = 0$ and $H(z \geq 0) = 1$). So the sum in (2.10.23) gives the number of point pairs that are separated by a distance less than ϵ . From this “integral” the dimension may be estimated by scaling $C(\epsilon)$ to ϵ . Rewrite (2.10.23) as

$$C(\epsilon) = \lim_{K \rightarrow \infty} \frac{1}{K} \sum_j^K \left[\frac{1}{K} \sum_i^K H(\epsilon - |\mathbf{z}_i - \mathbf{z}_j|) \right] \quad (2.10.24)$$

then for large K the sum over i is the natural measure in the ball $B_\epsilon(\mathbf{z}_j)$, i.e. more points are included of the whole attractor, so

$$C(\epsilon) = \lim_{K \rightarrow \infty} \frac{1}{K} \sum_j^K \mu(B_\epsilon(\mathbf{z}_j)) \quad (2.10.25)$$

This is an average over the orbit of the quantity $\mu(B_\epsilon(\mathbf{z}_j))$ in the position \mathbf{z} . It can be shown [73] that (2.10.25) is the same as the natural measure weighted average over that \mathbf{z} so

$$C(\epsilon) = \lim_{K \rightarrow \infty} \frac{1}{K} \sum_j^K \mu(B_\epsilon(\mathbf{z}_j)) = \int \mu(B_\epsilon(\mathbf{z})) d\mu(\mathbf{z}) \quad (2.10.26)$$

From the definition of $I(2, \epsilon)$ (2.10.22) it can be noted that if \mathbf{z} is in cube i , $\mu(B_\epsilon(\mathbf{z})) \sim \mu_i$. By replacing one of the μ_i in (2.10.22) with $\mu(B_\epsilon(\mathbf{z}_j))$ it shows that it is roughly an average over the natural measure, which is why

$$I(2, \epsilon) \sim C(\epsilon) \quad (2.10.27)$$

This shows that the correlation integral estimate $C(\epsilon)$ is similar to the correlation dimension D_2 .

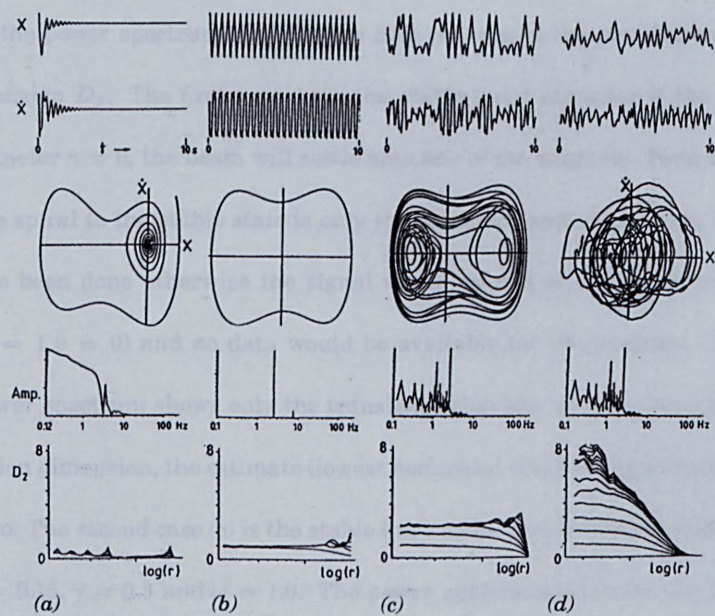


Figure 2.10.17: Dynamic analysis of the Duffing equation, top row depicts the beam position x versus time t , second row depicts the speed y (derivative of x) versus time, third row depicts the phase space x versus y , fourth row depicts the power spectrum, fifth row the correlation dimension; Column (a) the analysis of the stable point attractor; Column (b) analysis of the stable limit cycle; Column (c) analysis of the chaotic attractor; Column (d) analysis of the control.

Duffing's model

The Duffing model has been analysed using several different tools as shown in 2.10.17 on the page before. The first two rows show the time series of the position of the beam x and the first derivative $\dot{x} = y$ of the model in different states, with different parameter values. The third row shows the corresponding phase space of x and y . The fourth row is the power spectrum of x and the fifth row shows the correlation dimension D_2 . The first case (a) is the stable point attractor if the parameter $\gamma = 0$, the beam will settle onto one of the magnets. Note that the spiral to the stable state is only the transient and not a focus, this has been done otherwise the signal would consist of the stable state ($x = 1, y = 0$) and no data would be available for the analysis. The power spectrum shows only the transient behaviour as does the correlation dimension, the estimate (lowest horizontal line) is approximately zero. The second case (b) is the stable limit cycle with parameter values $\theta = 0.15, \gamma = 0.3$ and $\omega = 1.0$. The power spectrum identifies the two basic frequencies that form the orbit. The correlation dimension shows an estimate of $D_2 = 1$. The third case (c) is the chaotic attractor with the same parameter values as in case (b) (coexistence of chaotic attractor and limit cycle). The power spectrum demonstrates the complexity of the chaotic attractor. The correlation dimension estimates a fractal dimension of $D_2 = 2.56$. The last case (d) is a control that has been constructed from the chaotic signal by randomising the phase spectrum of the Fourier transform and recompiling the signal with an inverse transform. This introduces a random component in the system without

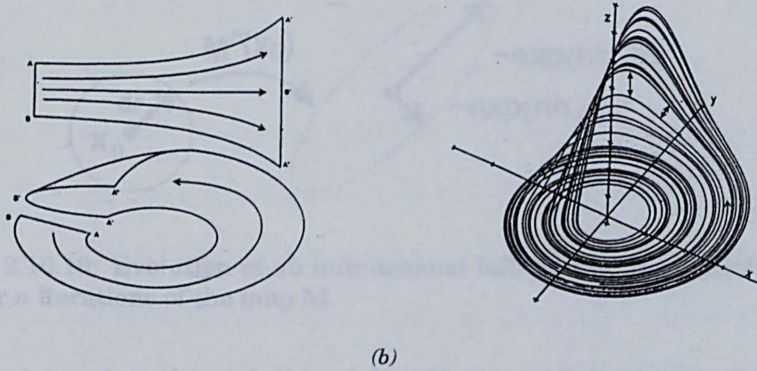


Figure 2.10.18: (a) Stretching and folding (or contraction) of an unstable manifold, three close by trajectories diverge with exponential rate. Folding of a manifold with the direction of evolution; (b) The Rössler attractor with both stretching and folding.

changing its properties. Note that the power spectrum is the same as the original chaotic power spectrum, but the D_2 estimate does not produce any results (maximum embedding dimension is eight) due to the increased complexity of the signal.

2.10.3 Lyapunov exponents

Lyapunov exponents give a means of characterizing the rate at which an attractor, and other invariant sets, stretches and contracts and is a convenient indicator of the sensitivity to small orbit perturbations (see figure 2.10.18). For example, consider a map M , with initial condition x_0 and the orbit x_n ($n = 0, 1, \dots$). Then consider the infinitesimal displacement from x_0 in the tangent direction y_0 , then the evolution of that vector is given by

$$y_{n+1} = DM(x_n) \cdot y_n \quad (2.10.28)$$

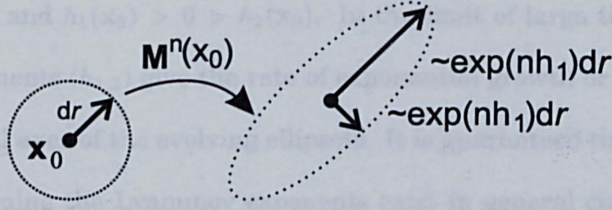


Figure 2.10.19: Evolution of an infinitesimal ball around the initial point x_0 after n iterations of the map M .

This determines the evolution of the infinite small displacement of the orbit from the unperturbed orbit x_n . The direction of the infinitesimal displacement from x_n is given by $\frac{y_n}{|y_n|}$. This fraction is the factor by which the displacement grows $|y_n| > |y_0|$ or shrinks $|y_n| < |y_0|$. From (2.10.28) define

$$y_n = DM^n(x_0) \cdot y_0 \quad (2.10.29)$$

$$DM^n(x_0) = DM(x_{n-1}) \cdot DM(x_{n-2}) \cdot \dots \cdot DM(x_0)$$

The Lyapunov exponent is defined for an initial condition x_0 and initial orientation of the infinitesimal displacement as given by $u_0 = \frac{y_0}{|y_0|}$ as

$$\begin{aligned} h(x_0, u_0) &= \lim_{n \rightarrow \infty} \frac{1}{n} \ln \left(\frac{|y_n|}{|y_0|} \right) \\ &= \lim_{n \rightarrow \infty} \frac{1}{n} \ln |DM^n(x_0) \cdot u_0| \end{aligned} \quad (2.10.30)$$

If the map has a dimension of N , then for a given x_0 there will be N or less distinct Lyapunov exponents. If the map is one dimensional then there is only one Lyapunov exponent. The initial orientation of the vector u_0 determines which Lyapunov exponent applies to the map.

If a number of initial conditions are defined in a small ball around a x_0 and subsequently, each initial condition is evolved under the map M for a certain number of n iterates. With the initial ball radius to be infinitesimal, the ball evolves into an ellipsoid as depicted in figure 2.10.19 for the

case $N = 2$ and $h_1(\mathbf{x}_0) > 0 > h_2(\mathbf{x}_0)$. In the limit of large time the Lyapunov exponents $(h_{1,2})$ give the rate of exponential growth or shrinking of the principal axes of the evolving ellipsoid. It is guaranteed that the limits used in defining the Lyapunov exponents exist in general circumstances. If μ is an ergodic measure, the Lyapunov exponent values $h_i(\mathbf{x}_0)$ are the same set of values for almost every \mathbf{x}_0 with respect to that measure μ . This implies for the natural measure that the Lyapunov exponents are also the same set of values for all \mathbf{x}_0 in the basin of attraction of the attractor, except for a set of Lebesgue measure zero. Therefore, the Lyapunov exponents are usually used without reference to the specific initial conditions. An attractor is defined to be *chaotic* if it has a positive Lyapunov exponent ($h_i > 0$), in this case infinitesimally separated initial conditions will move apart exponentially in time and are growing on average as e^{nh_1} . Thus, the condition $h > 0$ also implies *exponential sensitivity to initial conditions* for the attractor. It is noted that the exponential separation only holds for distances small compared to the attractor size.

For a periodic map M with $\mathbf{x}_0^* \rightarrow \mathbf{x}_1^* \rightarrow \cdots \rightarrow \mathbf{x}_p^* = \mathbf{x}_0^*$, the Lyapunov exponents for the periodic orbit are

$$h_i = \frac{1}{p} \ln |\lambda_i| \quad (2.10.31)$$

where λ_i are the eigenvalues of the matrix DM^p evaluated at one of the points $\mathbf{x} = \mathbf{x}_j^*$. A chaotic attractor may have embedded in it an infinite number of unstable periodic orbits, and each of those orbits yields typically a set of Lyapunov exponents that are different from those that apply for almost every initial condition with respect to the Lebesgue measure in the basin. They are, together with their stable manifolds, part of the zero

Lebesgue measure set that does not yield the Lyapunov exponents of the chaotic attractor.

The Lyapunov exponents of a continuous system of autonomous first order differential equations $\frac{dx}{dt} = F(x)$, are similarly defined as those of the map with some modifications. Equation (2.10.30) is replaced by

$$\begin{aligned}
 h(x_0, u_0) &= \lim_{t \rightarrow \infty} \frac{1}{t} \ln \frac{|y(t)|}{|y_0|} \\
 &= \lim_{t \rightarrow \infty} \frac{1}{t} \ln |\hat{O}(x_0, t) \cdot u_0| \\
 \text{where } \frac{dy(t)}{dt} &= DF(x(t)) \cdot y(t), \\
 x_0 &= x(0), \\
 y_0 &= y(0), \\
 u_0 &= \frac{y_0}{|y_0|}
 \end{aligned} \tag{2.10.32}$$

and where $\hat{O}(x_0, t)$ is the matrix solution of the equation

$$\frac{d\hat{O}}{dt} = DF(x(t)) \cdot \hat{O} \tag{2.10.33}$$

with the initial condition $\hat{O}(x_0, 0) = 1$. From (2.10.32), $y(t) = \hat{O}(x_0, t) \cdot y_0$ and the \hat{O} has the same role as the DM^n in the map derivations. For a chaotic attractor of a flow there exists one Lyapunov exponent that is zero, corresponding to an infinitesimal displacement along the flow.

2.10.4 Topological and metric entropy

The metric and topological entropy are two quantities that, like the Lyapunov exponent, can function as a means of quantifying chaos. They are positive for chaotic systems and zero for non-chaotic systems. The entropies have not been very useful in practice, though very important in theory, because of the difficulties in determining their values. Therefore, only a short

discussion of their properties will follow. Please see [73, 50, Ott or Katok] for an in depth discussion.

The topological entropy represents the exponential growth rate for the number of orbit segments that are distinguishable with arbitrary fine but finite precision. A map $f : X \rightarrow X$ is continuous and is compact with a distance function d . A sequence of increasing distance d_n^f , starting from d_1^f is defined as

$$d_n^f(x, y) = \max_{0 \leq i \leq n-1} d(f^i(x), f^i(y)) \quad (2.10.34)$$

The d_n^f measures the distance between two orbit segments. An open ball is defined as $B_f(x, \epsilon, n) = \{y \in X | d_n^f(x, y) < \epsilon\}$. Now a set $E \subset X$ is called (n, ϵ) -spanning if $X \subset \bigcup_{x \in E} B_f(x, \epsilon, n)$ and $S_d(f, \epsilon, n)$ is the cardinality of a minimal (n, ϵ) -spanning set. This quantity $S_d(f, \epsilon, n)$ is equal to the minimal number of initial conditions whose behaviour up to time n approximates the behaviour of any initial condition up to ϵ . The exponential growth rate is

$$h_d(f, \epsilon) = \lim_{n \rightarrow \infty} \frac{1}{n} \log S_d(f, \epsilon, n) \quad (2.10.35)$$

The $h_d(f, \epsilon)$ does not decrease with ϵ . From this the *topological entropy* $h_d(f)$ is defined as

$$h_d(f) = \lim_{\epsilon \rightarrow 0} h_d(f, \epsilon) \quad (2.10.36)$$

and does not depend on the metric d [50].

The metric entropy is also often referred to as the K-S entropy after Kolmogorov and Sinai who introduced the concept. It can be thought of as a number measuring the time rate of the creation of information as a chaotic orbit evolves. This means that if the initial state is known with limited

precision, it is possible to get to know more about the initial condition once the orbits diverge. The definition of the metric entropy revolves around a partitioning of a map into a number of disjoint compartments W_i . Subsequently, these are stepwise reduced into smaller sized partitions until for a given invariant probability measure μ the metric entropy can be defined, using the uniform norm, as

$$h(\mu) = \sup_{W_i} h(\mu W_i) \quad (2.10.37)$$

It is useful to note that the metric entropy is at most the sum of the positive Lyapunov exponents [73]

2.11 Control and synchronization

Chaos has often been considered undesirable. It tends to complicate analysis and function. However, making large modifications of a system parameter to reduce chaotic behaviour may be undesirable since it may well change the behaviour of the system in unacceptable ways. Rather than having to redefine the system, in most cases it is possible to change the behaviour of the system with only small changes to one system parameter that is available. The sensitive dependence of chaos and the ability to use it to control the chaotic state is one of the underlying reasons to study chaos in neural systems. In particular the ability of chaotic neuronal systems to recognise variations in input and rapidly change the temporal behaviour of the system. The methods that may be employed to control chaotic systems are discussed in this section.

2.11.1 Unstable Periodic Orbits

The natural measure of a chaotic attractor of an invertible, hyperbolic, two-dimensional map may be expressed in terms of the infinite number of unstable periodic orbits embedded within the attractor. This can be shown as follows.

Let \mathbf{x}_{jn} denote the fixed points of an n -times iterated map $M^n(\mathbf{x}_{jn}) = \mathbf{x}_{jn}$ and let $\lambda_1(\mathbf{x}_{jn}, n)$ denote the magnitude of the expanding eigenvalue of the Jacobian matrix $DM^n(\mathbf{x}_{jn})$. Here each \mathbf{x}_{jn} is on a periodic orbit whose period is n or a factor of n . The natural measure of an area S is then given by

$$\mu(S) = \lim_{n \rightarrow \infty} \sum_{\mathbf{x}_{jn} \in S} \frac{1}{\lambda_1(\mathbf{x}_{jn}, n)} \quad (2.11.1)$$

The summation is over all the fixed points of M^n in S . The equation (2.11.1) can be interpreted as a small region about \mathbf{x}_{jn} (for large n) that covers a fraction $\frac{1}{\lambda_1(\mathbf{x}_{jn}, n)}$ of the natural measure such that orbits that originate from this small region closely follow the orbit originating from \mathbf{x}_{jn} for n iterations [73]. How close subsequent orbits follow the original orbit from \mathbf{x}_{jn} depends on the rate of expansion of the eigenvalue in that point. Several consequences may be derived from equation (2.11.1) that may tell us something about the properties of the attractor. For more information please see [50, 49, 7, Katok, Katok and Auerbach et al.].

Determining several of these orbits and subsequently selecting the orbit that improves system performance allows the stabilization of an unstable periodic orbit by making appropriate pre-programmed modifications to a chosen parameter. If the changes are small enough the orbits will not have

completely different properties than the unchanged attractor.

2.11.2 Chaotic control by the OGY method

It is the goal of this control method to limit the dynamics of the system to any one of the unstable periodic orbits. Generally it is necessary to study the system to determine the existence of an orbit that has the properties that are required. This orbit may then be stabilized with the following method. Control requires making a *small* time-dependent perturbation to a controllable system parameter. This has been numerically demonstrated by [72, 73, 92, Ott, Grebogi and Yorke] and is referred to as the OGY method of chaos control. This method assumes that a parameter p in the multidimensional system $\frac{dx}{dt} = F(x, p)$ is available for external modifications. Here x indicates the matrix of system variables on which the operating matrix F applies. Firstly, one constructs a Takens vector by using delay coordinates of a system variable $z(t)$, let this vector be $Z(t) = \{z(t), z(t - T), z(t - 2T), \dots, z(t - MT)\}$. Subsequently, to find the periodic orbits and their stability properties, a Poincaré section is made by embedding the neighbouring points of an orbit of x into Z . This produces many points on the surface when $Z(t)$ crosses the section. These points are denoted $\xi_1, \xi_2, \dots, \xi_k$ for the k subsequent points and form a map Ξ . By taking $M = D - 1$, where D is the system dimension, the stability of the orbit may be studied. If a suitable orbit, one that may be stabilized by small perturbations at appropriate moments, is available, the next step is to define the boundaries between which a perturbation may be made. So, let the range between which p may be modified, around a nominal point p_0 , and let

$p' > p_0 > -p'$. On the Poincaré section the experimentally determined fixed point of the chosen orbit (ξ_f) , the stable and unstable eigenvalues λ_s and λ_u (with $|\lambda_u| > 1 > |\lambda_s|$) have corresponding unit vectors \mathbf{u}_s and \mathbf{u}_u in the stable and unstable directions. If the parameter p is modified slightly from p_0 to \bar{p} the coordinates of the fixed point ξ_f change to $\xi_f(\bar{p})$. If \bar{p} is small the approximation $\mathbf{g} \equiv \frac{\partial \xi_f}{\partial p}|_{p=0} \cong \bar{p}^{-1} \xi_f(\bar{p})$ may be made that allows the vector \mathbf{g} to be determined. In the section surface near the point where $\xi = 0$ the next point of the map Ξ is linearly approximated by $\xi_{n+1} - \xi_f(p) \cong \mathbf{M} \cdot [\xi_n - \xi_f(p)]$ with \mathbf{M} a 2×2 matrix. Using $\xi_f(p) \cong p\mathbf{g}$ and p_n small and of the same order as ξ_n , for every p_n depending on ξ_n [72]:

$$\xi_{n+1} \cong p_n \mathbf{g} + [\lambda_u \mathbf{u}_u \mathbf{f}_u + \lambda_s \mathbf{u}_s \mathbf{f}_s] \cdot [\xi_n - p_n \mathbf{g}] \quad (2.11.2)$$

Here \mathbf{f}_u and \mathbf{f}_s are contravariant basis vectors defined as $\mathbf{f}_s \cdot \mathbf{u}_s = \mathbf{f}_u \cdot \mathbf{u}_u = 1$ and $\mathbf{f}_s \cdot \mathbf{u}_u = \mathbf{f}_u \cdot \mathbf{u}_s = 0$. Now assuming that a ξ_n falls near the desired fixed point at $\xi = 0$ a p_n is chosen so that ξ_{n+1} falls on the stable manifold and $\mathbf{f}_u \cdot \xi_{n+1} = 0$. If ξ_{n+1} falls on the stable manifold of $\xi = 0$ the parameter perturbations may be set to zero and the trajectory will approach the fixed point with rate λ_s . Therefore, for sufficiently small ξ_n , p_n can be determined by:

$$p_n = \frac{\lambda_u}{(\lambda_u - 1)} \frac{(\xi_n \cdot \mathbf{f}_u)}{(\mathbf{g} \cdot \mathbf{f}_u)} \quad \text{with} \quad \mathbf{g} \cdot \mathbf{f}_u \neq 0 \quad (2.11.3)$$

This is used if the magnitude of (2.11.3) is less than p' , else p_n is set to zero.

The control is activated ($p_n \neq 0$) only when ξ_n falls in the strip

$$|\xi_n^u| < \xi' \quad \text{with} \quad \xi_n^u = \mathbf{f}_u \cdot \xi_n \quad (2.11.4)$$

$$\text{and from (2.11.3)} \quad \xi' = p' | (1 - \lambda_u^{-1}) \mathbf{g} \cdot \mathbf{f}_u |$$

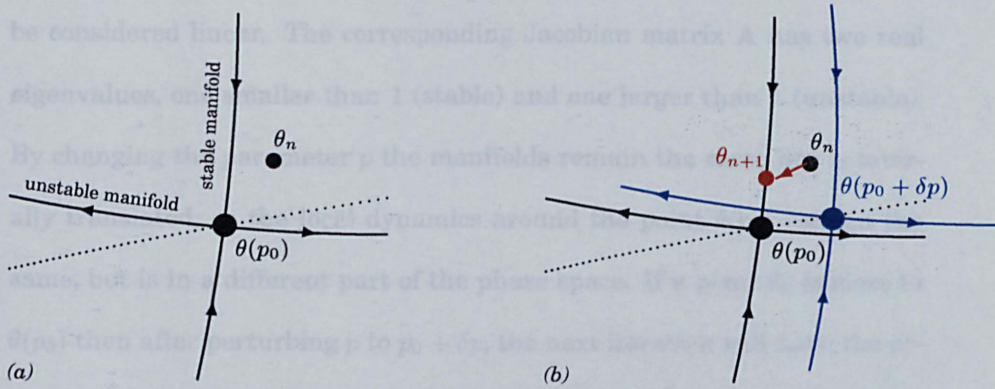


Figure 2.11.20: (a) Stable and unstable manifolds around the point $\theta(p_0)$ with the perturbation line along which θ may be shifted by modifying parameter p . (b) The effect of perturbing p to $p_0 + \delta p$, on the right side of the unstable and stable manifolds of p_0 are the corresponding manifolds of $p_0 + \delta p$.

This strip is along the stable manifold and the width is determined by p' . Starting at initial conditions, an orbit will go through the chaotic attractor until ξ_n falls inside (2.11.4). This activates the control and causes p_n to be modified according to (2.11.3). Because equation (2.11.2) is linear the control may fail to move the orbit onto the stable manifold, the orbit will then leave the boundaries indicated by p' and wander chaotically as if no control is active. In exceptional cases equation (2.11.3) may be activated when the trajectory enters within the control boundaries but is still far from zero, the wrong periodic orbit may be stabilized that also visits the strip.

To illustrate how modifying p stabilises an unstable period-1 orbit, imagine a special case two dimensional map Θ where the orbit has one stable and one unstable direction [94, 93] (see figure 2.11.20). Two curves through a point $\theta(p_0)$ form a stable and unstable manifold (e.g. a saddle node). By assuming a small enough neighbourhood around $\theta(p_0)$ the manifolds may

be considered linear. The corresponding Jacobian matrix A has two real eigenvalues, one smaller than 1 (stable) and one larger than 1 (unstable). By changing the parameter p the manifolds remain the same but is laterally translated, so the local dynamics around the point $\theta(p_0)$ remain the same, but is in a different part of the phase space. If a point θ_n is close to $\theta(p_0)$ then after perturbing p to $p_0 + \delta p$, the next iteration will have the orbit attracted to $\theta(p_0 + \delta p)$ parallel to the stable manifold and repelled from $\theta(p_0 + \delta p)$ parallel to the unstable manifold. If the value of δp is chosen appropriately, the next iteration θ_{n+1} will fall on the stable manifold of $\theta(p_0)$. Subsequently, the value of p is returned to p_0 and the orbit will remain on the stable manifold of $\theta(p_0)$ and approaches $\theta(p_0)$ [94].

The rate at which the stabilized orbit is approach by the chaotic transient $\langle \tau \rangle$ depends sensitively on initial conditions and has an exponential probability distribution $P(t) \sim \exp[-\frac{\tau}{\langle \tau \rangle}]$ for large τ and random initial conditions. The average length of the transient increases when p' decreases and has a power-law relation $\langle \tau \rangle \sim p'^{-\gamma}$ for small values of p' . The exponent γ is given by $\gamma = 1 + \frac{1}{2} \frac{\ln |\lambda_u|}{\ln |\lambda_s|}$. (For a complete derivation of γ see [72, 92].)

This demonstrates how to use a simple method of control and it is important to know which UPO is stabilized and how the boundaries of p' are determined. There exist several points that need to be taken into consideration if it is desired to apply the OGY method to experimental data [92]. Some systems are more difficult to control than others, also the presence of noise or sampling errors may introduce problems. A famous example of an application of chaos control in an empirical system is the control of

cardiac chaos by [31, Garfinkel et al.]. They used a modified version of the OGY method (“proportional perturbation feedback” or PPF) where the state point ξ_n is moved onto the stable manifold and therefore move towards the unstable fixed point. The OGY method moves the stable manifold to the current state point as explained above. They successfully controlled a cardiac arrhythmia into periodic beating using this method of control.

The OGY method of chaotic control is based on low dimensional chaotic models and assumes a unstable fixed point is accessible in the Fourier space. In the case of higher dimensional systems a small perturbation may result in the introduction of higher dimensional transients, to solve this a method is described by [7, Auerbach et al.] that allows only a few variables to be used in the control method depending on the local behaviour of the orbit that is desired. Further difficulties with chaotic control usually depend on the identification and prediction of unstable periodic orbits and the choice of parameter required to control the system. A method to apply an OGY type of control in large sets of models, e.g. network, either in delayed or undelayed systems, is described by [61, 92, Lourenco et al.].

2.11.3 Chaotic control by external periodic force

Apart from the OGY method of chaotic control it is possible to control chaos in several different ways, using an external periodic perturbation, delayed feedback, chaotic control of chaos [47, 53, 79, 80, 81, 92] and synchronization of chaos [76]. These different mechanisms of control seem to be more relevant as candidates of mechanisms in which a network of neurones may control its dynamics. The mechanisms generically do not require specific

knowledge of the model or of the controlling parameter, although some is require to construct the model correctly. The first method to be considered is the method where external periodic perturbation is used to control chaos (top panel (a) of figure 2.11.21 on the next page). This may be regarded as a direct application of control to a dynamical system $\dot{x} = f(x)$, where $x \in \mathbb{R}^n$ is controllable if a control function $u(t)$ exists so that $\dot{x} = f(x) + u(t)$ allows the movement of the trajectory from a point x_0 at time t_0 to the desired point x within finite time. This is a classical application of dynamic control that forces the system to become periodic provided that the period T of the external force $\frac{2\pi}{\omega}$ is equal to a period of one of the unstable periodic orbits. This method is immune to small parameter variations and is thus noise resistant, which makes it a convenient method for experimental systems [47].

2.11.4 Chaotic control by time delay feedback

The time delayed feedback method of control is depicted in the second panel (b) of figure 2.11.21. The general mechanism of delayed feedback control is as follows [92]. A system state vector $x(t)$ has a scalar quantity $g(x(t))$ depending on the accessibility of the state to measurements. If an unstable periodic orbits $\xi(t) = \xi(t + T)$ exists in the system with period T , the stabilising control force $F(t)$ is then

$$\dot{x}(t) = f(x(t), KF(t)) \quad (2.11.5)$$

where K is the amplitude scalar of the control force $F(t)$. Let us consider the case of $K = 0$, the existence of the unstable orbit $x(t) = \xi(t)$ has the

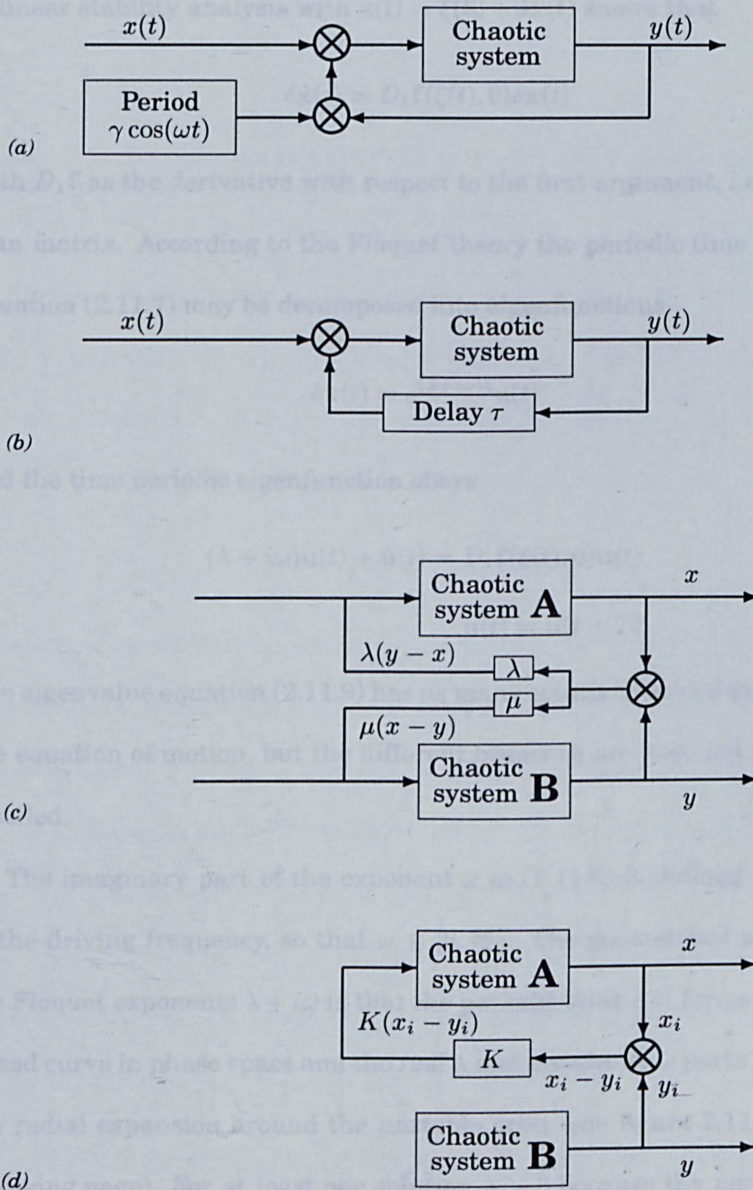


Figure 2.11.21: Control of chaos via (a) external periodic perturbation, (b) time delay feedback, (c) chaotic control of chaos and (d) synchronization by continuous control.

trivial constraint

$$\dot{\xi}(t) = \mathbf{f}(\xi(t), 0) \quad (2.11.6)$$

A linear stability analysis with $\mathbf{x}(t) = \xi(t) + \delta\mathbf{x}(t)$ shows that

$$\delta\dot{\mathbf{x}}(t) = D_1\mathbf{f}(\xi(t), 0)\delta\mathbf{x}(t) \quad (2.11.7)$$

with $D_1\mathbf{f}$ as the derivative with respect to the first argument, i.e. the Jacobian matrix. According to the Floquet theory the periodic time dependent equation (2.11.7) may be decomposed into eigenfunctions

$$\delta\mathbf{x}(t) = e^{(\lambda+i\omega)t}\mathbf{u}(t) \quad (2.11.8)$$

and the time periodic eigenfunction obeys

$$(\lambda + i\omega)\mathbf{u}(t) + \dot{\mathbf{u}}(t) = D_1\mathbf{f}(\xi(t), 0)\mathbf{u}(t) \quad (2.11.9)$$

$$\mathbf{u}(t) = \mathbf{u}(t + T)$$

The eigenvalue equation (2.11.9) has as many solutions as the dimension of the equation of motion, but the different branches are here not separately labelled.

The imaginary part of the exponent ω in (2.11.8) is defined as modulo of the driving frequency, so that $\omega \in [0, \frac{2\pi}{T}]$. The geometrical meaning of the Floquet exponents $\lambda + i\omega$ is that the periodic orbit $\xi(t)$ forms an almost closed curve in phase space and the real λ and imaginary ω parts determine the radial expansion around the unstable orbit (see figure 2.11.22 on the following page). For at least one solution $\lambda > 0$ because the periodic orbit is unstable. Only those solutions that have $\lambda > 0$ are considered in the control.

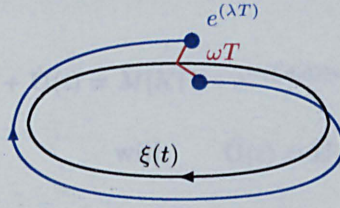


Figure 2.11.22: An unstable periodic orbit $\xi(t)$ and a neighbouring trajectory in phase space. Real and imaginary parts of the Floquet exponent are denoted λ and ω , T is the period of the UPO.

Delayed feedback control is useful, especially when the existence of an UPO is unknown, by delaying the trajectory for the length of one or more periods nT with $n = [1, 2, 3, \dots, k]$. The difference of the delayed trajectory $\mathbf{x}(t - \tau)$ and the trajectory at time $\mathbf{x}(t)$ can be used as the control force.

$$F(t) = g(\mathbf{x}(t)) - g(\mathbf{x}(t - \tau)) \quad (2.11.10)$$

This control force $F(t)$ is coupled to the system via one of the external parameters that is available (see second panel (b) of figure 2.11.21). When control is achieved the control force (2.11.10) becomes very small, often small periodic. Because it is in most cases not known how the control force controls the system within the internal degrees of freedom, it is not always clear by which mechanism the UPO is stabilized. Stability analysis of the general differential equation (2.11.5) and the difference equation (2.11.10) give some indication how this is generally achieved.

$$\delta \dot{\mathbf{x}}(t) = D_1 \mathbf{f}(\xi(t), 0) \delta \mathbf{x}(t) + K D_2 \mathbf{f}(\xi(t), 0) \{ D_g(\xi(t)) [\delta \mathbf{x}(t) - \delta \mathbf{x}(t - \tau)] \} \quad (2.11.11)$$

with $D_2 \mathbf{f}$ indicates the scalar derivative of the second argument. Applying the Floquet decomposition of (2.11.8), using capitalized letters for corre-

spending quantities

$$(\Lambda + i\Omega)\mathbf{U}(t) + \dot{\mathbf{U}}(t) = M[K(1 - e^{-(\Lambda+i\Omega)\tau})]\mathbf{U}(t) \quad (2.11.12)$$

with $\mathbf{U}(t) = \mathbf{U}(t + T)$

and where

$$M[\kappa]\mathbf{U} = D_1\mathbf{f}(\xi(t), 0)\mathbf{U} + \kappa D_2\mathbf{f}(\xi(t), 0)[D_g(\xi(t))\mathbf{U}] \quad (2.11.13)$$

The control can only affect the equation (2.11.12) through the parameter $\kappa = K(1 - e^{-(\Lambda+i\Omega)\tau})$, due to the choice of control. The stability properties or Floquet exponents of the controlled orbit $\Lambda + i\Omega$ are completely determined by equation (2.11.12) for the matrix (2.11.13). These Floquet exponents depend on parameter κ and are denoted $\Gamma(\kappa)$. From (2.11.12) the exponents of the orbit obey the constraint

$$\Lambda + i\Omega = \Gamma(K(1 - e^{-(\Lambda+i\Omega)\tau})) \quad (2.11.14)$$

It is generally difficult to obtain a closed analytical expression for Γ , but by the definition of (2.11.13) the boundary condition

$$\Gamma(0) = \lambda + i\omega \quad (2.11.15)$$

is satisfied, because the system is then reduced to the uncontrolled system (2.11.9). When the real part of the exponent Λ changes its sign on the control amplitude, the orbit becomes stable, the corresponding frequency at the particular value of K has to be finite, that is $\Omega \neq 0$. Additionally, torsion is a condition for an orbit to become stable. The control force is, roughly, proportional to the difference of the distance between the endpoints of the trajectory (see figure 2.11.22 on the page before) $x(t) - x(t - \tau)$. This distance would become zero if torsion was not present in the control, a finite

frequency is therefore required for the stabilization. Whenever some eigenvalues are unstable, the real part has to become negative to achieve stabilization. Frequencies with $\omega = 0$ and $\omega = \frac{\pi}{T}$ are stable with respect to perturbations, so that unstable orbits without the torsion (such as $\omega = 0$) can not be stabilized by small control amplitudes of K . Also, orbits with an odd number of positive Floquet exponents (such as $\Lambda > 0$ and $\Omega = 0$) can not be stabilized at all, because only pairs of eigenvalues may generate a finite frequency [92]. This explains the failure of the delay method to stabilise periodic orbits of the Lorenz model (see [92, 64]).

2.11.5 Chaotic control by chaos

The chaotic behaviour of one system may be controlled by coupling the system to another system that may be chaotic as well (panel (c) in figure 2.11.21 on page 81). The scheme by Pyragas to continuous control chaos is used to enhance the predictability of the coupled system. The two systems are coupled through the linear operators λ and μ . All the state variables of the systems A and B can be measured so that the output signals $x(t)$ and $y(t)$ can be used to connect the systems together with the control functions

$$\begin{aligned} F_1(t) &= \lambda(x(t) - y(t)) = \lambda D_1(t) \\ F_2(t) &= \mu(y(t) - x(t)) = \mu D_2(t) \end{aligned} \tag{2.11.16}$$

The values of the operators determine whether unidirectional ($\lambda > 0$ or $\mu > 0$) or bidirectional ($\lambda, \mu > 0$) coupling exists. Some rigorous conditions have been defined under which chaotic attractors of the systems are equivalent or the evolution of one of the systems is forced to take place on the attractor of the other (conform the pacemaker-slave system) [47]. For ex-

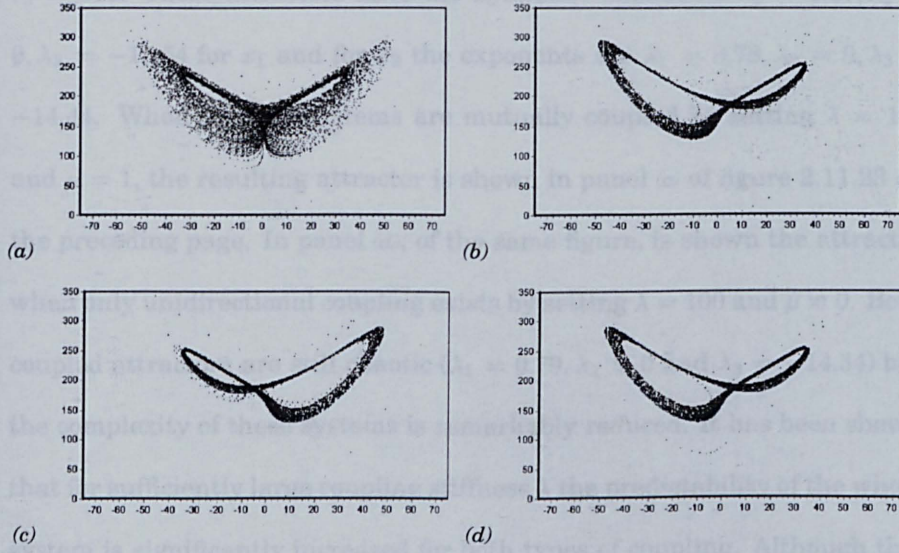


Figure 2.11.23: Chaotic control of chaos; (a) Uncontrolled Lorenz attractor; (b) Uncontrolled attractor with $r_2 = 211$; (c) Bidirectionally coupled attractor $\lambda = 100$ and $\mu = 1$; (d) Unidirectionally coupled attractor $\lambda = 100$ and $\mu = 0$.

ample, consider these two mutually coupled Lorenz models,

$$\begin{aligned}
 \dot{x}_1 &= -\sigma x_1 + \sigma y_1 + \lambda(x_2 - x_1) \\
 \dot{y}_1 &= -x_1 z_1 + r_1 x_1 - y_1 + \lambda(y_2 - y_1) \\
 \dot{z}_1 &= x_1 y_1 - b z_1 + \lambda(z_2 - z_1) \\
 \dot{x}_2 &= -\sigma x_2 + \sigma y_2 + \mu(x_1 - x_2) \\
 \dot{y}_2 &= -x_2 z_2 + r_2 x_2 - y_2 + \mu(y_1 - y_2) \\
 \dot{z}_2 &= x_2 y_2 - b z_2 + \lambda(z_1 - z_2)
 \end{aligned} \tag{2.11.17}$$

where $\sigma, r_{1,2}$ and b are constants. The result of this coupling is shown in figure 2.11.23. Here in panel (a) is shown the uncoupled chaotic Lorenz attractor of x_1 with $\lambda, \mu = 0, \sigma = 10, r_1 = 197.4, b = \frac{8}{3}$. In panel (b) is the uncoupled attractor of x_2 with the same parameter values as for x_1 but with

$r_2 = 211$. These attractors have the Lyapunov exponents $\lambda_1 = 1.87, \lambda_2 = 0, \lambda_3 = -15.54$ for x_1 and for x_2 the exponents are $\lambda_1 = 0.78, \lambda_2 = 0, \lambda_3 = -14.44$. When the two systems are mutually coupled by setting $\lambda = 100$ and $\mu = 1$, the resulting attractor is shown in panel (c) of figure 2.11.23 on the preceding page. In panel (d), of the same figure, is shown the attractor when only unidirectional coupling exists by setting $\lambda = 100$ and $\mu = 0$. Both coupled attractors are still chaotic ($\lambda_1 = 0.79, \lambda_2 = 0$ and $\lambda_3 = -14.34$) but the complexity of these systems is remarkably reduced. It has been shown that for sufficiently large coupling stiffness λ the predictability of the whole system is significantly increased for both types of coupling. Although this type of “control” does not stabilise an unstable periodic orbit, it modifies normal chaotic behaviour significantly and changes the trajectory space. It has also been found that some types of chaotic coupling results in periodic behaviour or steady state locking of the system [47].

2.11.6 Synchronization of chaos

Although a chaotic trajectory is not asymptotically stable, it is possible to synchronise two chaotic systems as has been shown by [76, Pecora and Carroll]. Their idea is to divide a system into subsystems, one of which becomes a driving subsystem and the other the response subsystem. This is done as follows. An n dimensional dynamical system is described by

$$\dot{u} = f(u) \quad (2.11.18)$$

$$u = u(v, w)$$

and can be divided into two arbitrary subsystems

$$\dot{v} = g(v, w)$$

$$\dot{w} = h(v, w)$$

with

$$v = (u_1, u_2, \dots, u_m) \quad (2.11.19)$$

$$g = (f_1(u), f_2(u), \dots, f_m(u))$$

$$w = (u_{m+1}, u_{m+2}, \dots, u_n)$$

$$h = (f_{m+1}(u), f_{m+2}(u), \dots, f_n(u))$$

Now create a new subsystem w' that is identical to the w subsystem and substitute the set of variables v for the corresponding v' in the function h

$$\dot{v} = g(v, w)$$

$$\dot{w} = h(v, w) \quad (2.11.20)$$

$$\dot{w}' = h(v, w')$$

Examine the difference $\Delta w = w' - w$, the subsystem components w and w' will synchronize only if $\Delta w \rightarrow 0$ as $t \rightarrow \infty$. In the infinitesimal limit this leads to the variational equations for the subsystem

$$L = D_w h(v(t), w(t)) L \quad (2.11.21)$$

where $D_w h$ is the Jacobian of the w subsystem with respect to w only. The behaviour of equation (2.11.21) depends on the Lyapunov exponents of the w subsystem, referred to as sub-Lyapunov exponents. The subsystems w and w' will synchronize only if the sub-Lyapunov exponents are all negative [76]. It has been demonstrated that the boundary between possible synchronization and non-synchronization is strictly connected with the transition from chaotic to hyper-chaotic behaviour that is characterized by at

System	Drive	Response	Sub-Lyapunov exponents
Rössler	x	(y, z)	$(0.2, -8.89)$
$a=0.2, b=0.2, c=9.0$	y	(x, z)	$(-0.056, -8.81)$
	z	(x, y)	$(0.1, 0.1)$
Lorenz	x	(y, z)	$(-1.81, -1.86)$
$\sigma = 10, b = \frac{8}{3}, r = 60.0$	y	(x, z)	$(-2.67, -9.99)$
	z	(x, y)	$(0.0108, -11.01)$

Table 2.11.2: Drive and response subsystems for the Lorenz and Rössler systems with corresponding sub-Lyapunov exponents. Taken from [76].

least two positive Lyapunov exponents [47]. In the table 2.11.2 is shown the synchronization behaviour for the Rössler and Lorenz systems. For the Rössler system it is possible to use the y component to drive an (x', z') response system and attain synchronization with the (x, z) components of the driving system. As can be seen in table 2.11.2 only the y drive will result in synchronization. For the Lorenz system both the x as well as the y subsystem drive will result in synchronization.

Different schemes for synchronization of chaotic systems have been devised and this is at this moment a subject of research. One scheme that is of particular interest is the synchronization by continuous control. This is a scheme where two chaotic systems are coupled unidirectionally in a way similar to the Pyragas feedback control (see panel (d) in figure 2.11.21 on page 81). So two systems described as

$$\begin{aligned}\dot{x} &= f(x, a) \\ \dot{y} &= f(y, a)\end{aligned}\tag{2.11.22}$$

where $x, y \in \mathbb{R}$ and $a \in \mathbb{R}$ which describes the control parameters. The

measurable signals $x_i(t)$ from system A and $y_i(t)$ from system B (where $i \in 1, 2, \dots, n$) are used to determine a difference $D(t)$ between the two signals as

$$F(t) = K(x_i(t) - y_i(t)) = KD(t) \quad (2.11.23)$$

which is applied to one of the chaotic systems as negative feedback. The parameter K is an adjustable weight of perturbation. The feedback does not change the solution of the perturbed system. When synchronization is achieved $F(t)$ becomes very small, often small periodic, so the chaotic systems A and B become practically uncoupled. Continuous chaos control systems are limited in their success rate of synchronization, only if the number of positive Lyapunov exponents of the composite coupled system is equal to the number of positive Lyapunov exponents of the component system is synchronization possible. If instead of only one state variable is feed back into the couple system, multiple state variables are used, a broader band of parameter space of K , where synchronization is possible, may be found [47].

2.11.7 Conclusion

All these different ways of controlling chaos, or change the behaviour and complexity of chaotic systems, seem to point to the fact that chaos is in reality a dynamical state that is not exceptional or unusable, but instead is common in nature and may be used to change the behaviour of very complex systems without making large changes to the system. This is ideal for dynamic neural networks whose individual units can not change the state of the entire network. Also, the combination of several different mechanisms,

control, synchronization and entrainment can occur concurrently without disrupting the system. Because in a dynamic neural network the individual units are the same or similar (with different parameter values) all of the preconditions for control and synchronization are satisfied. Indeed in most papers mentioned in this chapter, the tentative use of control and synchronization in neural networks have been indicated by the authors.

Literature study

"The whole is more than the sum of its parts."

Aristotle (ca. 350 BC)

Metaphysics 1014b30-31a

2.1 Introduction

This chapter is structured to introduce the several main categories that are relevant when studying the role of chaos and information storage in dynamic neural networks. Firstly, the biological references, in two sections, extend the neural justification for the assumptions used in this thesis and the biological relevant papers on the organization and behaviour of the brain. Secondly, some mathematical papers about the mechanisms involved in controlling chaotic systems, models of chaotic information storage and general properties of certain types of dynamic behaviour. This extends the discussion on chaos and control of chaos in the previous chapter (Chapter 1 on page 20). Thirdly, models of chaotic signal networks are discussed that

Chapter 3

Literature study

“The whole is more than the sum of its parts.”

Aristotle (ca 330 BC)

Metaphysica 10f-1045a

3.1 Introduction

This chapter is structured to introduce the several main categories that are relevant when studying the role of chaos and information storage in chaotic neural networks. Firstly, the biological references, in two sections, experimental justification for the assumptions used in this thesis and the biological relevant papers on the organization and behaviour of the brain. Secondly, some mathematical papers about the mechanisms involved in controlling chaotic systems, models of chaotic information storage and general properties of certain types of dynamic behaviour. This extends the discussion on chaos and control of chaos in the previous chapter (Chapter 2 on page 20). Thirdly, models of chaotic neural networks are discussed that

are related to this work. A general discussion of the presented literature concludes this chapter.

Several papers and books have been published recently that discuss the function of the brain and neural, as well as neuronal, networks. I would like to make the distinction between model networks (here called neural networks) and biological networks of neurones (here called neuronal networks). The goal is, generally, to explain why the current understanding and consequently, models, of brain function, information storage and processing, seems to be inaccurate or incomplete. In particular [29, Freeman] has suggested that information is not stored in the weight matrix of the synaptic connections as is currently thought by many, rather in the dynamical state of the network. He thought that several rules emerge from his research as hypotheses:

1. A global attractor exists with different “wings” (i.e. subspace) for each class of input.
2. Perception is said to occur when input constrains the global dynamics to one such “wing”.
3. Output is specified by the “wing” over a spatial periodic pattern.
4. Spatial integration by the target area is performed on the output.
5. Learning involves the formation of new “wings” and modification of existing attractors.

An alternative view suggested by [105, Tsuda et al.] which they called “the hermeneutic approach” stresses the importance of the interpretation of

neuronal networks and information. They present an argument that justifies this approach that considers the short time between presentation of an input pattern and the fast recollection of associated memories. This mechanism requires a random access search of all stored memories to access the input associated memories. Although their model stores information in the connection matrix and relies on a steady state mechanism to represent the recalled memories, it is the dynamic behaviour of the system that determines the recollection. Interestingly, [46, Kaneko and Tsuda] have subsequently argued that the reductionist approach, of steady state models, is severely limited when attempting to understand the mechanisms involved in information storage. Their reasoning is that, due to the complex non-linear nature of the brain, chaos is ubiquitous. This would be extremely undesirable if the system is sensitive for small perturbations, so the occurrence of complex dynamics is not incidental. The authors argue for a logic that “grabs a complex system without reduction to an ensemble of simple elements” [46]. This implies that a system may be complex but might depend on unique key elements which determine global behaviour and from this the complex dynamics emerges (with “emergence defined as the appearance of upper level description without the explicit instruction for it” occurring in the model, analogous to the mathematical interpretation of emergent dynamics). In other words, a complex system may contain some unique non-linear property that is identifiable, without reducing the system to predefined simple elements that do not produce the required complexity.

Other papers that support the dynamics hypothesis have been published. For example, [60, Liljenström] argues that biological dynamics have

often been neglected in ANN. He purports that new advances in ANN indeed depend on new developments in dynamic behaviour in ANN. He models the neurodynamics using an activity network with sigmoid functions and time delay. A Gaussian noise term is added to simulate spontaneous neuronal activity and it uses a Hebbian learning rule with a specific learning rate constant. With properly chosen parameter values, this network may be regarded as a network of connected coupled oscillators. Using excitatory and inhibitory connections it is demonstrated that periodic as well as chaotic dynamics are exhibited in this model.

In a paper by [8, Babloyantz et al.] a direct approach is used to model a network of interconnecting chaotic layers of oscillating elements. An external pacemaker stabilises an unstable periodic orbit in the first layer, this is referred to as the “attentive state” of the device. The second layer functions as the output layer and the average value of the squared amplitude of the second layer is used as the resulting output. Presenting different binary input patterns results in different output values, periodic, chaotic and discontinuous stable. Direction detection and motion detection are all possible with the device. The authors believe it to be a reasonable approximation of a neuronal network that incorporates some key features, such as layer interaction and spatiotemporal activities. However, I think that their model demonstrates only the basic properties of a dynamic network. Motion and direction detection are often possible with a distributed system and the most interesting contribution of this device is that this type of detection is also possible in a dynamic state. The importance and contribution of the chaotic equations to this device are unclear.

The role of bifurcations within dynamic neural networks is still a matter of debate. Ross Ashby, for example, has proposed that a system can move from a stable state via a bifurcation to a new stable state as an adaptive step [89]. A change in the system, e.g. due to external feedback, may trigger a bifurcation and the system will move to the new stable state. In chaotic neural networks the system is unstable but locally stabilized with delay feedback. A parameter change causing a bifurcation to a stable state would destroy the chaotic attractor. Bifurcations that cause a change in the strange attractor could be important by allowing the system to stabilise different orbits (see also 2.11.4).

These papers indicate the current interest that exists in dynamic neural networks. It is assumed that the proposed methodology and models are relevant to biology. However, little is known about the possible biological mechanisms due to the complex nature of the brain. There is strong support for the dynamic theory of neuronal dynamics, based on experimental data [27, 28, 29, Freeman], [99, So et al.] and [106, 107, Tsuda et al.] .

Lastly, the dynamic theory of neuronal networks has received some attention from information theorists. For example, [96, Siegelmann] writes that the brain is a possible non-Turing machine. The Church-Turing thesis (C-T) says that no possible abstract digital device can have higher capabilities than Turing machines [96]. It identifies the notion of an algorithm (in mathematical modelling) with a computation in the Turing model and the efficiency of problem solving with polynomial time Turing computation. Several suggestions have been made that the Turing model does not necessarily describe all possible computation in nature [96]. Evolving compu-

tational machines, such as dynamic neural networks as analogue model of computation, can tune their internal constants and parameters on a continuum where the values are not measurable by an external observer.

3.2 Biological references

This part describes relevant papers that have been published about the biological mechanisms involved in memory storage and processing. Obviously, the amount of information published is too large to discuss here in detail. However, several key papers are included that support the main hypothesis of dynamical neural networks presented in this thesis. Few relevant scientific papers were found that challenge this hypothesis, so only a positive evidence for this theory is currently available.

A commentary published by [63, McKenna et al.] is a good introduction to the biological evidence available that supports the dynamic neuronal network hypothesis. It presents a basic introduction to some of the non linear tools available to study complex dynamics (more recently, some new tools have been developed and more is known about the capabilities of each, e.g. [50, 92]). It also demonstrates different single cell dynamics of several organisms, including stable points, limit cycles and chaos; furthermore, transitions over bifurcation points are shown to occur. This offers further support to the original theory proposed by [27, 28, 29, Freeman]. An independent paper by [33, Guevara et al.] states that complex dynamics, assumed to be chaos, are found in non linear feedback systems processing time delays in the absence of noise, such as in forced neural oscillators. This

is demonstrated with a forced Hodgkin-Huxley (HH) model of hippocampal inhibition circuit. It is known that a forced HH model can show complex behaviour, because the unforced model attempts to return in a transient state to the stable rest point (e.g. [16, 104]). The oscillator forces the model away from the stable point and thereby is capable of perturbing the system continuously. Depending on the current state of the HH model this will result in the characteristic firing and slow return to stable state. However, the perturbing forcing oscillator prevents the normal state to return, and keeps forcing the system to fire at different moments in state space. This results in various types of periodic oscillations depending on the strength and periodicity of the forcing term. Although this model does not by itself produce chaotic dynamics, due to its stable system, a set of coupled oscillators or a stochastic term might achieve this. Experimentally, the paper refers to studies of pacemaker neurons that exhibit complex dynamics in certain cases. The importance of this paper is not the modelling but the recognition that chaos is a natural dynamic state of neuronal systems and that chaos could have a function in normal and diseased neuronal systems. An important book, about the complexities of the brain and neural network modelling of experimental psychology is “The Computational Brain” by [16, Churchland and Sejnowski] that demonstrates the computational aspect of the neuronal network. It is one of the standard textbooks and has many examples of experimental psychological experiments. A more recent book on experimental psychology and traditional neural networks is [88, Rolls and Treves]. This introduces traditional neural networks and its relevance for the experimentalist. An appendix discusses recurrent neural networks

and some possible uses for this type of model.

3.2.1 Neuronal networks

Discussing the evidence that exists for complex dynamics in neuronal networks it is noted that again most papers on this particular subject are produced by [12, 27, 28, 29, 30, 51, 115, Freeman et al.]. His group is very active in modelling the olfactory system. They have performed extensive studies of the Olfactory System (OS) in different animals, and have devised very complex ODE models of the neurons in the OS, based on activity and transfer functions [115]. Parameters of these models are determined numerically within physiological constraints. Each neuron has a “synaptic” transfer function, usually linear, with a gain coefficient K , the system parameters are chosen to be in chaotic range. The network of neuronal gains as $K_m[i,j]$ is used as weight matrix of the neural network to train the system. A selective strengthening rule, modified Hebbian learning, is used to train the network. Presenting different binary inputs results in specific firing patterns, associated with the input. Untrained information results in a different pattern, unassociated with any of the other inputs. This model does not change dynamics from chaos, but shows different “bursting” dynamics when input is changed. It is assumed by the authors that different subspaces are occupied by the model that represent the learned input patterns. They argue that the scalability and the independence on initial conditions makes the model accurate. However, this is not necessarily the case, because they can not demonstrate that the “trained” patterns are unique to the system, or that the system has knowledge of the difference between the

patterns. One might argue that the system simply switches from one attracting state to a different attractor altogether and the “bursting” is part of the transient.

Subsequent models such as presented in [12, Chang and Freeman] are gradually more complex and have increased in size. Using a backpropagation algorithm they attempted to make the network learn and optimise for a behaviour in accordance with biological experiments. The resulting networks produce similar EEG patterns as found in their experiments. A different paper by [51, Kay et al.] attempts to identify how the OS moves from one attracting state to another during odour recognition. This is done by specific signal analysis of experimental data. They conclude that inter-area communication exists and that the system moves into an “attentive state” as part of the recognition process as proposed by [8, Babloyantz et al.]. Furthermore, they support the notion as presented by [107, Tsuda] of chaotic itinerancy. This means that the system evolves through the phase space and if a recognized state is presented the corresponding attractor will be formed due to the input and the system will settle in this state. This should correspond with the recognized memory of the presented input. Lastly, in the paper by [30, Freeman] a hypothesis is presented that noise in the microscopic activity of neurons acts as a carrier of the system state. According to this paper, a stimulus induces an instantaneous state transition to one of the possible recognized attractors. The resulting state is broadcast to different areas by the carrier noise and is extracted at the target. Thus, the chaotic patterns of activity are caused by noise controlled chaos. Little experimental or modelling support is presented for

this view, different oscillations in the EEG such as beta waves are identified as the inter-area communication. These papers are all very interesting because of the underlying theory that is explored. Several criticisms may be held against this theory. Firstly, there is little understanding how the multiple states (“wings”) are formed, even though some attempt has been made with some extremely complex models (some consist of 600 first order ODEs and 7000 parameters). It is impossible to understand the parameter space sufficiently to rule out any of the possible dynamical mechanisms that may produce the desired result. Also, the basic pretrained weight matrix $K_m[i,j]$ still functions as a stable state memory, with dynamic behaviour that switches from one state to an other. This results in the system producing a resulting dynamic behaviour that is associated with the input only by inference. The system itself still has no knowledge of its state and therefore is unable to learn or respond to the input in a relevant manner. The chaotic itinerancy is an interesting idea that requires further study but it depends on external observation of the system to determine its state.

3.2.2 Memory

What is currently known about the plasticity and storage of memories? This is the subject of much research, especially in the field of psychology. But little is known about the mechanisms and even the exact nature of memory mechanics. One of our collaborators on this subject is Dr. Steve Ray of the School of Biological and Molecular Sciences. He has been very active in this field and has been working on the nature of memories. Several of his papers attempt to demonstrate that memories are a dynamic part of a

neuronal network [85]. To this end, he has been performing experiments as described in **Chapter 6 Biological experiments** that clearly show that information that has been trained into the subjects is preserved over the destruction of the network as well as modifiable with new information presented later. A recent paper [86, Ray] shows that even after metamorphosis in flies, that involves complex neurogenesis and neurolysis to change into a new shape, memories are preserved. In this case memories were associated with certain behaviour which remained after metamorphosis. Hence, the neurodynamics that produced that behaviour have been preserved. If the network was dependent on the exact spatiotemporal organization, such as would be the case in a traditional neural network, the information would be lost. Therefore, it may be hypothesized that by simply communicating the information a new network may acquire the information without preserving the original organization of the network.

3.2.3 Biological dynamic behaviour

Evidence that complex dynamics is indeed present in a biological non-pathological network is presented by several papers. At many different levels may complex dynamics be found. For example, [26, Faure et al.] demonstrate chaos and periodicities to be present in the synaptic “noise” of a central neuron. Also spontaneous chaotic activity in molluscan neurones and periodically forced axons have been found [34, 24, 1]. From personal experience and many references, complex dynamics including high and low dimensional chaos have been found in EEG of both healthy and epileptic subjects (e.g. [66]). In a letter to the journal Nature [17, Cobb et al.] show

that synchronization of pyramidal neurons is provided by the inhibitory interneurons in the hippocampus. This indicates that entrainment and synchronization of the complex non linear systems indeed occur and have a possible important function in information storage and processing.

3.2.4 Biological models

Faced with the evidence presented in the previous paragraph it is not very hard to imagine how this may be of relevance for biologists. It is, however, very difficult to study and implement in such a way that further understanding of biological processes may be gained. Constructing models has been of great value in increasing the understanding of those processes, however the models are generally fairly complex and require much effort to complete and understand. Some of these models have been landmark models, such as the Hopfield model [40, 41] of neural networks and the models made by Traub [104] that model networks of neurons of the hippocampus using Hodgkin-Huxley type channel dynamics. These are landmark models because they are either novel models that describe complex behaviour [40, 41] or more complete models that have been used only on small scale examples [104]. Other models, such as [56, Krakauer] model different specific behaviour, like the spatial memory in bees using a feedforward perceptron model. This illustrates the capabilities of the ANN models with specific tasks in mind.

An example of a model that uses the Hodgkin-Huxley equations to reproduce specific bursting patterns is the chaos produced by [55, Komendantov et al.]. This describes specific waves of the intracellular calcium, that may

show multi-orbits and even chaos. It is claimed that an excitatory system with complex electrical activity that exhibits slow and fast components has, as a fundamental property, chaotic behaviour. This lets the authors claim that the “random process” observed may be an essential role in neuronal networks.

3.3 Chaotic neural networks

In order to model chaotic network behaviour, it is necessary first to understand the system dynamics and the ability of the system to generate the required behaviour. Since the early eighties, a much better understanding of the mathematical foundations of chaos has been reached, and it is no longer regarded as vague and catastrophic but as something useful and possibly elementary to many branches of science. Several aspects of chaotic dynamics appear to be relevant and useful for models of neural and neuronal networks. These include control, synchronization, space filling and sensitive dependence in initial conditions and several researchers have explored these aspects of chaos.

3.3.1 Stability of chaotic networks

By its nature a chaotic network is unstable, but deterministic. If a network model is constructed from a well known model, such as the McCulloch-Pitts neurons, the network is stable. To achieve a chaotic system one would need to use a chaotic model or modify a stable model to become unstable in a predictable manner. One such an attempt is described by [112, Wang] by

using a modified two state McCulloch-Pitts neuron network. This model has been defined for both autonomous and non-autonomous systems. For the autonomous model the system is defined as follows. The state of neuron i is S_i and the total input $h_i(t)$ for that neuron is given by

$$S_i(t + \Delta t) = \begin{cases} -1 & : x < 1 \\ +1 & : x \geq 0 \end{cases}$$

$$h_i(t) = \gamma_1 \sum_{j=1}^N T_{ij} S_j(t) + \gamma_2 \sum_{j,k=1}^N T_{ijk} S_j(t) S_k(t) + \eta_j \quad (3.3.1)$$

$$T_{ij} = C_{ij} \sum_{\mu=1}^P S_i^\mu S_j^\mu$$

$$T_{ijk} = C_{ijk} \sum_{\mu=1}^P S_i^\mu S_j^\mu S_k^\mu$$

where the T_{ij} and T_{ijk} are the modified Hebbian synaptic efficacies. The vector S^μ is the μ^{th} stored pattern of p stored patterns. The variables C_{ij} and C_{ijk} are chosen randomly according to a certain probability, and is introduced to assure incomplete connectivity. Lastly, the variable η_i is a random Gaussian background noise that introduces a possible fractal element into this system. Using specific values for the parameters it can be shown that this parallel network can exhibit periodic and chaotic behaviour through a period doubling bifurcation. It is demonstrated by [112, Wang] that this model is capable of switching periodic behaviour in a “noise” parameter that may have one of two possible values. The resulting behaviour of the network corresponding with the two values can be induced by modifying the randomness in the system, which results in the occurrence of a periodicity corresponding with one of the two values or neither. The crucial element in this experiment is that the randomness can not only promote

chaotic behaviour but can also control or suppress chaos. The theoretical results of this model clearly show that a neural network may be noisy but this does not preclude the storage and reproduction of a memory and may even enhance the system. It could be that the control of the system is due to synchronization of the units in the network in a specific region of the phase space.

A type of network that is particularly interesting for this thesis is a neural network with delay. Several analyses of this type of network have been performed. One stability analysis of an analogue neural network with delay has been done by [62, Marcus and Westervelt]. This network consists of a Hopfield network [41] with the addition of a delay term. It was found that even though no oscillations are possible in the normal Hopfield network, a network with delayed output can undergo sustained oscillations when the ratio of delay to relaxation time of the network exceeds a critical value. The authors also found that for an all-inhibitory network the potential for oscillation increases significantly; furthermore they found that the application of the Hebb rule does not produce oscillations for arbitrary long delays. It is possible to enable such a network to oscillate by limiting the interconnections to a few connection strengths. A subsequent paper by [69, Olien et al.] shows that regions of multistability and co-existing limit-cycles may be found in the delayed Hopfield network provided that the connection matrix and the maximum slope of the transfer function are restricted to certain values. It is noted by the authors that the most interesting type of bifurcations only occur when the networks contains self-connections.

A third type of network architecture is the specific organization of non-

spatial networks. The stability of these networks has been analysed and shows specific behaviour due to the type of model used. A model used by [21, Destexhe] is derived from passive membrane conductance with sigmoid transfer functions. The architecture of this model consists of pairs of excitatory and inhibitory neurons and with only the excitatory neurons connected within the network. A reduced system of one pair is used to perform calculations of uniform solutions of the network. The system is discretized from the delayed differential equations to simplify analysis. The complete network has previously been shown to have a uniform periodic solution. The analysis shows that the periodic solution was only stable for small networks ($N \leq 9$) and became unstable for larger networks ($N > 9$). Similarly, a two and three neuron model studied by [13, Chapeau et al.] shows for this “minimal” delayed neural network that oscillations and probably chaos are possible within small networks. Furthermore, it is shown by [65, Noonburg] that in a non-delayed network of n neurons, it is possible to get oscillations provided that the transfer function gain is very high and that the weight of the intercellular connections can be modified rapidly around the threshold value.

These models describe the instability of specific types of neural network models, it can be concluded that the occurrence of many different types of dynamic behaviour may be introduced into a model by carefully choosing the parameter space. It is also important to note that although symmetry is prevalent it can still imply oscillations. The use of a delay within these types of networks clearly extends the possible dynamic behaviour of the neural networks.

3.3.2 Periodic behaviour of chaotic networks

Once the chaotic network is constructed and confirmed to be chaotic, the network has to be controlled or limited to become periodic instead. One of the easiest ways to control a neural network is using the periodic external force (see section 2.11.3 on page 79 in chapter 2). A discrete neural network was controlled by [100, Solé et al.] using a periodic feedback algorithm. The periodic control can stabilise two or three period orbits in a discrete two neuron model. The strong connectivity of this particular model ensures that when the system is extended to three or four fully connected neurons, the control on only one neuron can stabilise a periodic orbit of the whole network. A network model developed by [11, Bondarenko] controls a continuous network using an external periodic force with varying frequencies and amplitudes of the external force. It is shown that stabilization is possible and also “anti-control”, producing higher dimensional chaos than the uncontrolled chaotic system.

A single system described by [97, Sinha] attempts to control a network described by the second order equation

$$x_i(n+1) = G \left(\sum_{j=1}^N W_{ij}x_j(n) - K_{Ei}G(K_{IE}x_i(n-1) + I_i(\omega + \delta)) + I_i(\omega) \right) \quad (3.3.2)$$

that resembles a Hopfield-like network with delay dynamics. Using external input with varying amplitude, specific patterns may be visited by the trajectory to represent stored information. The model demonstrates the possible targeting of a specific area in phase space for memory recollection.

Several different methods to control an unstable periodic orbit in a chao-

tic network are discussed by [61, Lourenco and Babloyantz]. The various methods are not extensively compared, but a modified version of the OGY method of control (see section 2.11.2 on page 75 in chapter 2) was investigated for its ability to control a chaotic map. The authors argue against the use of a OGY-like type of control because of the computational power required when using it in larger networks. A delayed feedback mechanism is preferred due to its ability to control a system by making continuous small perturbations to a system parameter.

A completely different approach to elicit a periodic response out of a chaotic neural network has been done by [108, 109, 110, Tsui and Jones]. They use a feedforward neural network whose output is used as delayed feedback control in order to stabilise an unstable periodic orbit. The standard feedforward network of 2 input neurons, 2 output neurons and 10 neurons in 2 hidden layers uses a sigmoidal transfer function and is trained on the Ikeda map

$$z_{n+1} = \gamma + Rze^i \left(\kappa - \frac{\alpha}{1 + |z|^2} \right) \quad (3.3.3)$$

with $\alpha = 5.5$, $\gamma = 0.85$, $\kappa = 0.4$, $R = 0.9$ and $z \in \mathbb{C}^n$. Setting $x_n = \text{Re}(z_n)$ and $y_n = \text{Im}(z_n)$, the external input S_n and the delayed feedback P_n is added to the system mapping F as

$$(x_{n+1}, y_{n+1}) = F(x_n + S_n, y_n + P_n) \quad (3.3.4)$$

Using this it is possible, after training the network, to stabilise the system. Different input signals will result in the same or different stable orbits. Applying only the input, without control, is not generally sufficient to stabilise orbits. However, it appears that in some cases a stable orbit may be

found after several regimes of input and control are tried. Robustness of the control is demonstrated by applying additional Gaussian noise.

3.3.3 Chaotic network models

This part describes some interesting models proposed by different authors, that have different approaches to modelling a chaotic neural network with control. A very useful model is the model proposed by [2, Aihara et al.], that describes a single neuron model with chaotic dynamics, including graded responses, relative refractoriness and spatio-temporal summation of inputs. The model is derived from the Caianiello's neuronic equation, of which the McCulloch-Pitts neuron is a special case. A modified version of the same equation was used by Nagumo and Sato to develop the following model to produce the output $x(t + 1)$

$$x(t + 1) = u \left(A(t) - \alpha \sum_{r=0}^t k^r x(t - r) - \theta \right) \quad (3.3.5)$$

where u is a unit step function or sigmoid function, $A(t)$ is the input strength at time t and k is the damping of the refractor rate. Aihara et al. defined a new internal state $y(t + 1)$ as

$$y(t + 1) = A(t) - \alpha \sum_{r=0}^t k^r x(t - r) - \theta \quad (3.3.6)$$

subsequently, (3.3.5) and (3.3.6) can then be simplified into

$$y(t + 1) = ky(t) - \alpha u(y(t)) + a(t) \quad (3.3.7)$$

$$x(t + 1) = u(y(t + 1)) \quad (3.3.8)$$

$$\text{where} \quad a(t) = A(t) - kA(t - 1) - \theta(1 - k) \quad (3.3.9)$$

If the input into the network is periodic with constant amplitude A then (3.3.9) may be used as $a = (A - \theta)(1 - k)$. The response characteristic of the equations (3.3.7) and (3.3.8) form complete devil's staircases, this means that chaotic solutions exist only at a self-similar Cantor set of the parameter values with zero Lebesgue measure. It is shown by Aihara et al. that this model may be used to produce a chaotic neural network with a small positive largest Lyapunov exponent.

The Aihara model has been used by [58, Kushibe et al.] to build a network that can target a specific embedded memory by associating it with a desired output. The incomplete target enables the system to locate a memory by chaotic searching avoiding local minima. Even when random input patterns are used, the system does not fall into a local minimum but will reproduce a memory pattern provided that at least 20% of the target pattern is available.

Mappings are often used in modeling discrete chaotic neural dynamics. A good example of this type of model is a map model by [75, Pasemann and Stollenwerk] that is described by a recurrent two neuron model as

$$x_{n+1} = \vartheta_1 + w_{11}\sigma(x_n) + w_{12}\sigma(y_n) \quad (3.3.10)$$

$$y_{n+1} = \vartheta_2 + w_{21}\sigma(x_n) + w_{22}\sigma(y_n) \quad (3.3.11)$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (3.3.12)$$

where x_n represents the state of inhibitory neuron and y_n the state of an excitatory neuron. The variables are set as $\vartheta_1 = -2, \vartheta_2 = 3, w_{11} = -20, w_{12} = -6, w_{12} = 6$ and $w_{22} = 0$, i.e. no auto inhibition of y_n . Applying a modified OGY form of control using linear least squares estimations of Δx_{n+k}

on the parameter ϑ_1 , this model may stabilise different periodic orbits with the current parameter set. Further modifications are made to establish a process of self-control, this comprises of four control neurons used as a one layer feedforward network. The combination of the model and the control network allow stabilization of all available periodic orbits. The authors then use external or dynamic noise to switch between different orbits.

A comparable model as the Pasemann system is used by [54, Klotz and Bräuer]. The system is described by

$$x_i(t+1) = f\left(\sum_{j=1}^n w_{ij}x_j(t)\right) \quad (3.3.13)$$

$$f(z) = \frac{1}{1 + e^{-4\sigma z}} \quad (3.3.14)$$

From this several different network configurations can be constructed. A two neuron model with appropriate weights demonstrates two chaotic attractors. A four neuron model with time delay is constructed to show one chaotic attractor. Introducing an external input on one of the neurons reduces the system to one half of the attractor depending on the sign of the input. This model is then used by the authors to create an XOR function network. It shows chaotic behaviour when the two inputs are both zero or one. It will reduce to one of the two halves of the attractor when one of the two inputs is one.

As in the previous model, instead of stabilising a periodic orbit or state a particular subspace within the attractor or even different attractors may be used as an encoding of information. Itinerating of a chaotic state among different attractors is used in a large model by [42, Hoshino et al.]. This model is a network of a three neuron module as described by [13, Chapeau-

Blondeau et al.]. This system is capable of chaotic and periodic oscillations within a certain distance of unit firing. This may then be modified by parameter changes in the system to represent different learning patterns. The system is not controlled but merely illustrates the ability to associate information with specific chaotic or periodic behaviour in the parameter domain.

A compartmental neuronal model approach has been used by [22, Destexhe] to model a network of delayed neurons. Within certain boundaries of the critical parameter values, particularly the weights, periodic oscillations, spiral waves and spatiotemporal chaos was found. None of these dynamics can be produced without the delay, although the system is not particularly sensitive to the delay length. It is argued that the spatiotemporal phenomenon observed is useful to optimise information transfer within the network, even though the model does not demonstrate this.

A recurrent neural network model has been constructed by [3, Andreyev et al.] that is trained to store a piecewise linear map. The exact nature of the map, i.e. the transition from one map state to another, is defined by the encoding of the desired information. This will result for n pieces of information of a repeated information block of length $n + 1$. After training of the network with different blocks of information, the trajectory of the system will visit all regions of the stored patterns during iteration. Presentation of an input pattern will then stabilise a particular period associated with one of the information blocks. This model shows a useful method of encoding information temporally into a neural net that then may be searched chaotically for the correct pattern.

In papers by [113, 114, Watanabe et al.] the effect of coincidence detection in a network is investigated. A simple continuous neuron model is described with an exponential decay and threshold. The internal state is described as $a(t_{n+1}) = s_{n+1} + a(t_n)e^{\frac{-T_{n+1}}{\tau}}$ and a global negative feedback is calculated to reduce or increase the threshold value. This model can demonstrate coincidental firing of neurons in an almost synchronized manner. A more complex neural network model with coincidence detector is also described. This second model is continuous and includes a time delay and exponential decay of the activation. A global negative feedback with time decay τ_g is applied to the network. The model is described as

$$\tau \frac{da_i(t)}{dt} = -a_i(t) + S^{ext} \sum_n \delta(t - t_{i,n}^p) + \sum_{j=1}^N w_{ij} x_j(t - d_{ij}) \quad (3.3.15)$$

$$\tau_g \frac{dr(t)}{dt} = -r(t) + R \sum_{j=1}^N x_j(t) \quad (3.3.16)$$

If

$$h(t) \equiv a_i(t) - (\theta + r(t)) \geq 0 \quad (3.3.17)$$

then

$$x_i(t') = \delta(t' - g(h(t)) - t) \text{ and } a_i(t') = 0 \quad (3.3.18)$$

otherwise

$$x_i(t) = 0 \quad (3.3.19)$$

where S^{ext} is an external input pulse strength, R is the global feedback strength, $x_i(t)$ is the output of neuron i at t , $t_{i,n}^p$ is the time for external pulse n to arrive at neuron i , w_{ij} and d_{ij} are the synaptic weight and delay

from neuron j to neuron i , and N the total number of neurons. All synaptic weights are set to 1 with the exception of autoconnections $w_{ii} = 0$. The external pulse is required for the activity of the network to be maintained. If the external pulse generation is increased over time the system will become unstable, due to the fact that the monotone increase of the pulse does not allow the system to stabilise onto a coinciding pulse. If the pulse is varied but not monotone, coincidental pulses may be found and the system becomes periodic.

3.3.4 Synchronization of chaotic networks

Some work has been done on the synchronization of chaotic neural networks. In a paper by [95, Shuai and Wong], the authors use two fully connected chaotic networks described by

$$S_i(t+1) = f(h_i(t)) + \nu_0 R \quad (3.3.20)$$

$$h_i(t) = \sum_{j=1}^N J_{ij} S_j(t) \quad (3.3.21)$$

$$f(x) = \tanh(\alpha x) e^{-\beta x^2} \quad (3.3.22)$$

where $S_i(t+1)$ is the state of neuron i of N neurons, J_{ij} is the connection weight of neuron i onto neuron j and R is a random noise function with scalar ν_0 . Using appropriate weights the systems are hyperchaotic and may be modified such that when a random noise is introduced in both systems this results in synchronization if the amplitude ν_0 of the noise in the first neuron is ≥ 1.09 . This is proved by calculation of a conditional Lyapunov exponent. A phenomenon known as finite-precision synchronization of chaos may be found when $\nu_0 = 1.2$, the Lyapunov exponent of the first neuron is

positive but subsequent units are negative. They also show that for a fully connected neural network, a stochastic synchronization state can always be achieved for any neuron driven by noise. If a network is not fully connected, synchronization can not always be obtained independently of the amplitude of the noise. For two networks with slightly different parameters, with one as a driving system, weakly noise induced synchronization may be found.

For this reason several authors have investigated the use of synchronization as a means of secure communication. The general idea is to encrypt a binary message by means of a chaotic masking system or by a modulation/detection system [19]. The system hides the message within a chaotic model and the received message will synchronize the receivers chaotic system allowing the extraction of the message. See for a full discussion [19, 48, 70, 71, 76, Cuomo et al., Kapitaniak et al., Oliviera et al., Oliviera et al. and Pecora et al.]

3.4 Conclusion

Several researchers have made attempts to use chaos for storing and retrieving information. This has resulted in different types of models and experiments, all of which seem to support the concept of chaos as an efficient means of traversing the phase space to locate a specific subspace, unstable periodic orbit or attractor. The mechanism for information storage is therefore indicated but little useful information has actually been stored. Considerable further research is therefore required to enable the encoding, storage and retrieval of information for chaotic neural networks

to become useful and comparable with artificial neural networks. The most remarkable aspect of the dynamic information theory seems to be not the fairly large body of support but the absence of the contrary.

Chapter 4

Modelling environment

4.1 Introduction

To study the dynamics of a chaotic neural network it was necessary to develop a software package that can do this. The package was structured such that the chaotic differential equations (ODEs) were given in a text file that was passed to the package from an external script. This was then used to calculate the dynamics of the network. Due to the nature of the package, it was possible to perform simulations.

Chapter 4

Modelling environment

“As soon as we started programming, we found out to our surprise that it wasn’t as easy to get programs right as we had thought. Debugging had to be discovered. I can remember the exact instant when I realized that a large part of my life from then on was going to be spent in finding mistakes in my own programs.”

Maurice Wilkes

4.1 Introduction

To study the dynamics of a chaotic neural network, it was deemed necessary to develop a software package that can do the numerical integrations. The package was structured such that the Ordinary Differential Equations (ODEs) were given in a text file that was parsed by the system into an internal format. This was then used to calculate the evolutions of the network. Due to the nature of the proposed models, no scientific or commercial packages were available to perform the modelling.

The system was developed on PC under Microsoft Windows®95/98. This was done because these were the most easily available systems, and because of the emergence of the Borland CBuilder®(1-5) C++ development environment that facilitated the development of the user interface significantly. This choice, however, required the program design to be severely limited in portability and flexibility to updates. Even though the system has been designed in object oriented manner, a reasonably large part is still platform dependent.

All the software has been designed and written by myself, with the exception of the following

- The signal analysis toolkit, developed by the Intel® corporation. This was used because it has processor optimized code for Fourier analysis and correlation [43].
- The skeleton of the equation parser was provided by Harald Helfgott [36], this has been modified and extended by me .
- The code for the box counting dimension was taken from the TISEAN Nonlinear Time Series Analysis toolkit [35]. This was modified and optimized by me.

4.2 Design

In designing the system I was able to use my experience gained when developing a simpler ODE integrator for the Netherlands Institute for Brain Research (NIBR), to model intercellular circadian rhythm generators [67, 68]. The basic integrator algorithm (see section 4.3 on page 122) was then taken

from [78, Press et al.]. This is however copyrighted and free use is only allowed for academic research. To enable free use and, possible future distribution of the system, the algorithm was redesigned from the basic equations. Some constants and other parameters were retained from [78, Press et al.], these are however not part of their code and may be considered to be in the public domain. Other modelling algorithms were either freely available or designed specifically. Data structures were designed in an object oriented way. To handle the complex interactions of the system, objects were stored in templated vectors. These were provided by the Standard Template Library (STL), and have efficient procedures for memory management, and insertion and deletion of objects. The calculated data is stored in double format that is a 64 bit data format with 15-digit precision.

4.2.1 Object dependencies

The objects were defined in a modular manner, with logical grouping of related code. The interface modules and the numerical modules were separated, but required a single module to handle the mutual dependencies. This module, called *Server*, is a class object of type *ServerObject*. It is created and owned by the main window object and it provides protected access to the integration objects which it owns (see figure 4.2.1 on the following page). The server object owns the *Integrator* and *NetIntegrator* objects that are separate threads from the main program. These two objects implement the integration routines for the single model *Equation* object and the network *NetEqObject* objects respectively. Each *NetEqObject* corresponds with a unit, an array of which is owned by the *UnitWeb* object.

The Server object also handles the updating of the various windows and the graphs. The equation parser and evaluation is left to an object-oriented formal language.

4.3 Algorithms

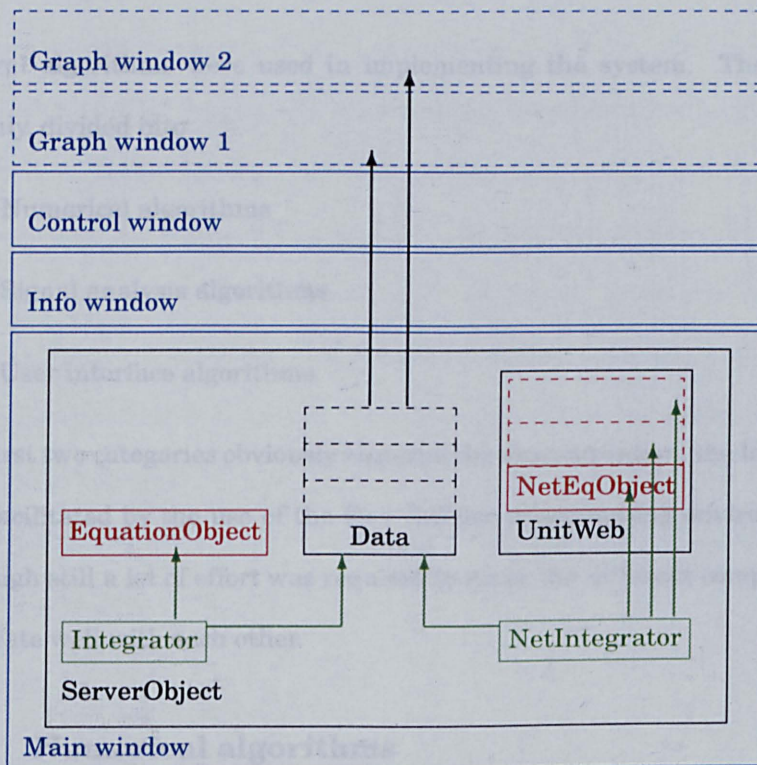


Figure 4.2.1: Object dependencies. The “ServerObject” is owned by the main window, which is the entry point of the program. This object owns the two integration threads and the “EquationObject”. It also owns the data arrays (“Data”) and the network arrays (“UnitWeb”), which in turn owns the network equation objects.

The `Server` object also handles the updating of the various windows and the graphs. The equation parser and evaluation is part of an object called `formuClass`.

4.3 Algorithms

Several algorithms were used in implementing the system. These are, roughly, divided into

1. Numerical algorithms
2. Signal analysis algorithms
3. User interface algorithms

The first two categories obviously required the most attention, the last item was facilitated by the use of the C++ Builder programming environment, although still a lot of effort was required to make the different components integrate well with each other.

4.3.1 Numerical algorithms

The development of the required numerical algorithms was subdivided into two major parts. The first concerning the parsing and evaluation of the model equations and the second part with the numerical integration of these equations. The model equations are provided in proprietary format text files. The format of these files is described in the subsection 4.5.1 on page 140 and subsection 4.5.2 on page 145.

4.3.1.1 File parsing and evaluation

The parsing of the equation files is divided into two parts. The first part reads in the files and prepares the data for the parsing of the equations. For the equation files this is done in the module `equation.cpp` and for the network files this is done in the modules `Settings.cpp` and `Neteq.cpp`. The actual equation parsing is done in the module `formulc.cpp`. The skeleton of an algorithm to parse an equation was based on some sample code provided by Harald Helfgott [36]. This has been improved and extended by me to provide the required functionality. The equation parser converts the equations into a proprietary format that consists of an array of unsigned integers. Each integer encodes for a variable or operation that corresponds to the Reverse Polish Notation. This enables the system to evaluate the equation at runtime.

To demonstrate how the encoding and evaluation works, consider the Duffing equation,

$d=0.15$

$g=0.3$

$o=1$

$x'=y$

$y'=x-x^3-(d*y)+g*\cos(o*t)$

$t'=1$

$x=-1$

$y=1$

$t=0$

Variable	Value	Index number
d	0.15	0
g	0.3	1
o	1	2
x	-1	3
y	1	4
t	0	5

Table 4.3.1: Variable table with values and index number in variable array.

The parser will translate the variables as is shown in table 4.3.1. The encoded variables are then used to build the reverse polish notation of the equation in binary format, each encoding occupying a 32 bit unsigned integer. Because the value zero (0) is used to indicate an undefined variable, at most $2^{32} - 1$ variables are supported. Differential equations are denoted by the presence of a apostrophe (') after the variable on the left hand side, if it is omitted the equation is considered to be a difference function. The first equation $x' = y$ is translated as

V3=V4.

The second equation $y' = x - x^3 - (d * y) + g * \cos(o * t)$ is translated as

V4=V3V3D0^-V0V4*-V1V2V5*F1*+.

And the third equation $t' = 1$ is translated as

V5=D1.

Here each character, number or symbol represents a variable, value or operation in reverse polish notation. The characters (D, V, E, F, T, M) represent the encoding, where V stands for a variable, followed by the index number

of the variable in the variable array. The character D represents an absolute value (in double format), followed by the index number in the value array. The character E represents an external unit, with the first subsequent number as the index in the unit array and the second subsequent number as the index in that unit's variable array. The character F represents one of the predefined functions, with the following number the index in the function array. The character T represents a delayed function with the following index number of the delay array. The last character M represents the negate operation. The supported symbols $+$, $-$, $*$, $/$, $^$ represent the basic operations, addition, subtraction, multiplication, division and power.

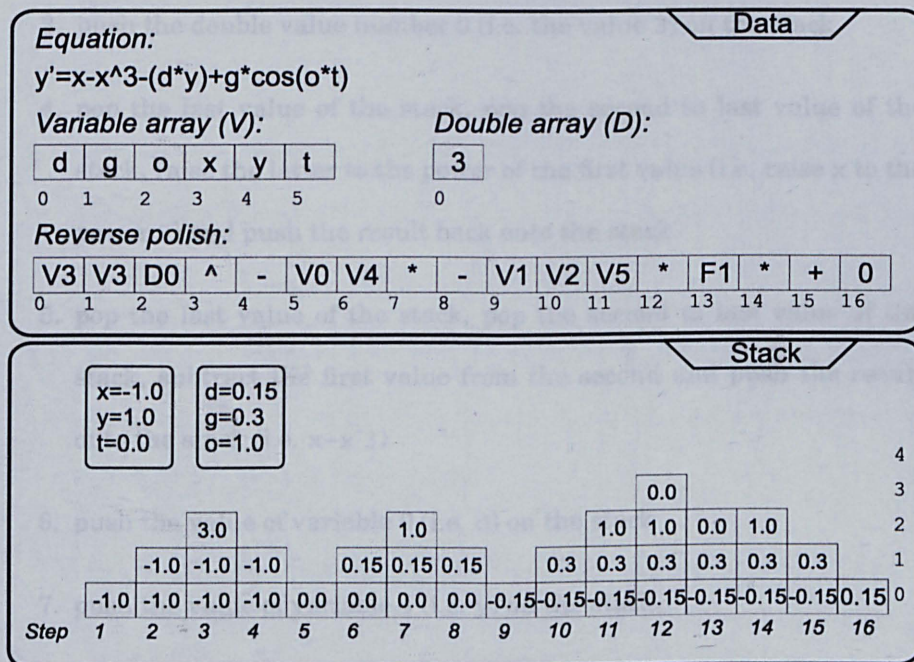


Figure 4.3.2: Diagram showing the internal data representation of the second Duffing equation in the top panel. The bottom panel diagram shows the evaluation steps on the stack with the current values of the variables indicated in the subpanels. The steps at the bottom of the panel correspond with the steps in the text on the following page.

At runtime the encoded equation is evaluated using a result stack. Each character is in reality a 32 bit unsigned integer in a variable size array that is evaluated from left to right. The code of the evaluation function is shown in the appendix A.3 on page 234. To demonstrate how the function works, the encoded second equation of the Duffing's equation (the variable y) is calculated as described below. In figure 4.3.2 on the page before a schematic representation of the data and the stack is shown.

1. push value of variable number 3 (i.e. x) on the stack
2. push value of variable number 3 on the stack
3. push the double value number 0 (i.e. the value 3) on the stack
4. pop the last value of the stack, pop the second to last value of the stack, raise the latter to the power of the first value (i.e. raise x to the power 3) and push the result back onto the stack
5. pop the last value of the stack, pop the second to last value of the stack, subtract the first value from the second and push the result onto the stack (i.e. $x - x^3$)
6. push the value of variable 0 (i.e. d) on the stack
7. push the value of variable 4 (i.e. y) on the stack
8. pop the last value of the stack, pop the second to last value of the stack, multiply the two values and push the result on the stack (i.e. $d * y$)
9. pop the last value of the stack, pop the second to last value of the

- stack, subtract the first value from the second and push the result onto the stack (i.e. $x - x^3 - (d*y)$)
10. push the value of variable 1 (i.e. g) on the stack
11. push the value of variable 2 (i.e. o) on the stack
12. push the value of variable 5 (i.e. t) on the stack
13. pop the last value of the stack, pop the second to last value of the stack, multiply the two values and push the result on the stack (i.e. $o*t$)
14. pop the last value of the stack and execute the function number 1 (i.e. $\cos(o*t)$)
15. pop the last value of the stack, pop the second to last value of the stack, multiply the two values and push the result on the stack (i.e. $g*\cos(o*t)$)
16. pop the last value of the stack, pop the second to last value of the stack, add the two values together and push the result onto the stack (i.e. $x - x^3 - (d*y) + g*\cos(o*t)$)

At the end of this evaluation the stack contains just one value, which is the result of the equation. The resulting value is then added to the variable y . If the function is not a differential function but a difference function, the calculated result replaces the value of the variable.

4.3.1.2 Numerical integration

Several numerical integration algorithms are used in the system. They will be shortly described, the reader is referred to the reference section for more extensive discussion of integration algorithms and their advantages and disadvantages.

Forward Euler

The Forward Euler integration algorithm [74] is the simplest algorithm that attempts to approximate the initial value problem solution $\xi_t(x_0, t_0)$ of the continuous time system.

$$\dot{x} = f(x, t) \quad (4.3.1)$$

$$x(t_0) = x_0$$

by calculating a number of points x_1, x_2, \dots, x_k from the initial conditions x_0, t_0 , that ought to be $x_k \approx \xi_{t_k}(x_0, t_0)$. The uniformly spaced steps from which the next point is calculated is denoted by $h > 0$ such that $t_{k+1} = t_0 + kh$.

The state of the system (4.3.1) at time t_k is approximated by the k^{th} derivative of x as

$$\dot{x}(t_k) \approx \frac{x_{k+1} - x_k}{h} \quad (4.3.2)$$

This approximation becomes then

$$x_{k+1} = x_k + hf(x_k, t_k) \quad (4.3.3)$$

The accuracy of this integration algorithm depends on the size of the step h . Smaller steps will result in more frequent approximations and as $h \rightarrow 0$ equation (4.3.2) will approach (4.3.1).

Runge-Kutta

The algorithm described by the Runge-Kutta set of integration algorithm is based on the local approximation of the solution $\xi_t(x_h)$ by its Taylor series expansion. The Taylor series expansion is an approximation by ν -terms of the derivative. The number of the order of the Runge-Kutta algorithm indicates the number of Taylor expansion terms used in the approximation. The second order Runge-Kutta also known as midpoint method, though not implemented in the system, illustrates the use of this algorithm nicely. A whole family of the second order Runge-Kutta exists, but generally they work along the following lines, here described by the modified Euler-Cauchy algorithm [74]

$$x_{k+1} = x_k + hf \left(x_k + \frac{h}{2} f(x_k, t_k), t_k + \frac{h}{2} \right) \quad (4.3.4)$$

This works in two steps, firstly it moves a half step forward in time $t_k + \frac{h}{2}$ using the forward Euler:

$$\hat{x} = x_k + \frac{h}{2} f(x_k, t_k) \quad (4.3.5)$$

then this value \hat{x} is used to approximate the vector field and a forward Euler iteration is used:

$$x_{k+1} = x_k + hf \left(\hat{x}, t_k + \frac{h}{2} \right) \quad (4.3.6)$$

The fourth order Runge-Kutta is an extension of this, using intermediate points to calculate the state. Again, several variations exist, but the

most common fourth order Runge-Kutta is

$$\begin{aligned}
 k_1 &= f(x_k, t_k) \\
 k_2 &= f\left(x_k + \frac{h}{2}k_1, t_k + \frac{h}{2}\right) \\
 k_3 &= f\left(x_k + \frac{h}{2}k_2, t_k + \frac{h}{2}\right) \\
 k_4 &= f(x_k + hk_3, t_k + h) \\
 x_{k+1} &= x_k + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)
 \end{aligned}
 \tag{4.3.7}$$

The derivation of (4.3.7) is extensive, but it is easy to understand. The k_i 's where $i = 1, 2, 3, 4$, represent an approximate value of the vector field. The first approximation k_1 is the value of the vector field at x_k , the second k_2 is the approximate value a half-step later at time $t_k + \frac{h}{2}$. This is the same as a forward Euler step with step size $\frac{h}{2}$. The third approximation k_3 is also at time step $t_k + \frac{h}{2}$ but estimated using the slope of k_2 . This is comparable to a backward Euler half step. The fourth step k_4 is the value of the vector field at t_{k+1} and is obtained using the value k_3 . This is a modified Euler-Cauchy step. The four approximations k_1, k_2, k_3, k_4 are averaged to provide the approximation of the vector field used to predict x_{k+1} [74].

Variable step Runge-Kutta

In many cases it is desirable to have an integration routine that can control its own progress by adapting the step size according to some error function. This will allow the integration to use larger steps when possible, thereby increasing the efficiency significantly. One good method is the Runge-Kutta-Fehlberg algorithm that compares the results of fourth and fifth order Runge-Kutta algorithms to estimate an error value. This is then used to accept the step or adjust the step size as well as to predict the next

step size. The general form of a fifth order Runge-Kutta is

$$\begin{aligned}
 k_1 &= f(x_k, t_k) \\
 k_2 &= f(x_k + a_{21}hk_1, t_k + b_2h) \\
 k_3 &= f(x_k + a_{31}hk_1 + a_{32}hk_2, t_k + b_3h) \\
 k_4 &= f(x_k + a_{41}hk_1 + a_{42}hk_2 + a_{43}hk_3, t_k + b_4h) \\
 k_5 &= f(x_k + a_{51}hk_1 + a_{52}hk_2 + a_{53}hk_3 + a_{54}hk_4, t_k + b_5h) \\
 k_6 &= f(x_k + a_{61}hk_1 + a_{62}hk_2 + a_{63}hk_3 + a_{64}hk_4 + a_{65}hk_5, t_k + b_6h) \\
 x_{k+1} &= x_k + c_1k_1 + c_2k_2 + c_3k_3 + c_4k_4 + c_5k_5 + c_6k_6
 \end{aligned} \tag{4.3.8}$$

The error estimate is then calculated using

$$\Delta \equiv x_{n+1} - x_{n+1}^* = \sum_{i=1}^6 (c_i - c_i^*)k_i \tag{4.3.9}$$

where x_{n+1}^* and c_i^* indicate the fourth order Runge-Kutta estimate and its constant respectively [74, 78]. In the program the values of the constants a_i , b_i, c_i and c_i^* are taken from [78, Press et al.] and are those found by Cash and Karp that form a more efficient method and have better error properties. It is also required to consider some boundary effects when estimating the new step value h , for example scaling and desired accuracy.

Multi step and other algorithms

There exist many other powerful algorithms for ODE integration. One algorithm that has not been implemented but may prove useful is a multi step algorithm such as Gear's algorithm. These algorithms reuse previous calculated points of the trajectory to estimate the current state. A m -step algorithm uses the m previous points $x_k, x_{k-1}, \dots, x_{k-m+1}$ and the evalua-

tion at those points to estimate x_{k+1} . Its general form is

$$\begin{aligned}
 x_{k+1} = & a_0 x_k + a_1 x_{k-1} + \cdots + a_{m-1} x_{k-m+1} \\
 & + h(b_{-1} f(x_{k+1}, t_{k+1}) + b_0 f(x_k, t_k) + b_1 f(x_{k-1}, t_{k-1}) \\
 & + \cdots + b_{m-1} f(x_{k-m+1}, t_{k-m+1}))
 \end{aligned} \tag{4.3.10}$$

where the t_{k-i} are evenly spaced with time step h . It has, however, not been proved that multi step algorithms are more efficient than single step algorithms such as Runge-Kutta [74].

Some other algorithms require the availability or easy computability of the Jacobian. Examples are the Newton-Raphson and generalizations of the Runge-Kutta method such as the Rosenbrock method. The Jacobian is not always available, especially in this system where the equations are evaluated at runtime. Estimating the Jacobian using numerical differentiation may introduce additional errors and these numerical integration methods are therefore not used in the program.

4.3.2 Signal analysis algorithms

To study the resulting data produced by the program, several signal analysis tools have been implemented. Some of these implementations are incomplete due to time restrictions. Three main sources for algorithm have been used, these are the Intel® Signal Processing library v4.1 [43], the Tisean code [35] and the Numerical Recipes by [78, Press et al.]. The signal analysis algorithms concerned are

- Fast Fourier transform (FFT), for the Power spectrum and the period estimate.

The FFT implementation used is from the Intel library, replacing the original implementation of the FFT taken from Press et al., there are some concerns about the performance and accuracy of the Press implementation. Also, the Intel library has processor optimized code, increasing the performance.

- Recurrence or return map

The implementation was taken from the original paper by [23, Eckmann et al.], the Tisean package [35] and slightly modified by myself to increase performance.

- Poincaré map

This implementation was taken from the Tisean package with modifications by myself.

- Dimension estimate for the box counting dimension and the correlation dimension

These were taken from the Tisean package, the Procaccia algorithm and from code used previously by me [77].

- Lyapunov exponent

The algorithm for the estimation of the maximal Lyapunov exponent is not entirely straightforward. Several different approaches are possible. The most common version generally used is the algorithm as proposed by Benettin [73, 74] which uses Gram-Schmidt orthonormalization to preserve the changing linear subspace spanned by the evolving vectors. An alternative approach is based on the box counting dimension [35].

4.3.3 User interface algorithms

These involved the design and construction of a user interface to enable the user to interact with the data and the parameter settings of the model and the system. The design of the user interface was facilitated by using the Borland C++ Builder environment. This enabled the use of multiple windows, and ease to manipulate tree like views of the variables and parameters. It also provides the use of a chart component that can plot the data in several different windows. However, it was not until late in this project discovered that in particular cases this library caused the system to “hang”. To circumvent this problem it is required that the charting module of the system is redesign again. The validity of the calculated results is not compromised by this problem, it is merely a cosmetic issue.

The interaction of the data calculation modules, i.e. the integrator and the analysis modules, with the user interface is particularly difficult if the user would like to have real time data visualization. This is solved in the system by using different application threads. This means that the integrators, the data analysis and the user interface are considered to be different processes. Synchronization of the different units is achieved by an exclusive access manager that handles access to the data buffer.

4.4 Tests

Naturally, it is very important that the calculated data is indeed the result of the expected calculation. This means that particular attention has been given to ensure the accuracy and efficacy of the system. The system

has been tested in several different ways, i.e. the numerical procedures were tested as well as the resulting behaviour compared to known dynamic behaviour.

4.4.1 Numerical tests

The numerical tests involved the verification of the proper functioning of the numerical algorithms. Firstly, the calculation was tested using the Duffing equation (see chapter 2 on page 20). Because of my previous experience with this particular model, and the availability of a separate system to compare the results with [77], it confirmed the accuracy of the integration. The signal analysis algorithms were tested with the produced data of the Duffing model, and compared with literature. This confirmed the correct use of these algorithms.

The problem of integration errors produced by the system has been addressed by comparing the integration results with results obtained by integrating with different system parameter settings. Local errors, i.e. an error obtained per algorithm step do not propagate from one step to the next and it can be assumed that the optimization of the algorithm limits the size of this type of error. The round-off error depends on the hardware used but is independent of the integration step. To minimise this type of error, throughout the system double precision (64 bit) was consistently used which is accurate to about fifteen decimal places. The third type of error is the local error introduced by the fact that the algorithm used is based on the Taylor series, and is known as the truncation error. The result of the integration would be exact if the entire infinite Taylor series could be used. It

is algorithm dependent but independent of the hardware used. For the k^{th} order Runge-Kutta algorithm the local truncation error is under suitable conditions

$$\epsilon_t = \alpha_k h^k \quad (4.4.1)$$

where $\alpha_k \in \mathbb{R}$ and depends on k, f and x_k but not on h . This means that the local truncation error decreases with step size and if h is small enough for a certain algorithm, the higher-order algorithms are more accurate than the lower-order algorithm. To ensure that the local truncation error does not play a major part in the dynamic behaviour investigated, each calculation was performed with different values of h .

The global errors are the global effects of the local errors discussed above. Assume that the step size is much smaller than one and that the number of integration steps is $N = \frac{1}{h}$. The global round off error is the accumulation of the local round off error, so if the local round off error is ϵ_r then

$$\epsilon_{gr} = N\epsilon_r = \frac{\epsilon_r}{h} \quad (4.4.2)$$

This means that the global round off error is inverse proportional to the step size, so a larger step size leads to a smaller global round off error. The global truncation error is the accumulation of the local truncation error, if the dependence of α_k in (4.4.1) is negligible then the global truncation error is

$$\epsilon_{gt} = N\epsilon_t = \alpha_k h^{k-1} \quad (4.4.3)$$

so by decreasing the step size, the global truncation error is decreased.

A last word about numerical stability; the chosen numerical integration algorithm may have low round off error and low truncation error, but if it is numerically unstable it is not useful. The integration algorithm assumes that the previous calculated m points x_k, \dots, x_{k-m+1} lie on the trajectory, but this is not usually the case due to the local truncation and round off errors. This effect propagates throughout the integration and is not accounted for by either of the errors. Different integration algorithms as well as different order of the algorithm have different areas of stability. The solution to this problem is to vary the value of h and use different algorithms when a specific problem appears to be unstable, possibly due to the algorithm used. This particular issue is considered to be a minor problem for the practical integration of an ODE [74, 78].

4.4.2 Chaos

It may be considered that the integration of chaotic systems is particularly suspect due to the sensitive dependence on initial conditions of chaotic systems. If an arbitrary small error eventually effects the global behaviour of a system, how can then an integration algorithm with its inherent errors produce useful results. It has been argued that therefore it is meaningless to integrate a chaotic system. However, numerical simulations of chaotic systems are widespread and have been proven to be meaningful and useful. To see why this is the case, consider the following.

An integration routine is not capable of predicting the *exact* state of a chaotic system after long integration. The perturbations in a chaotic system grow initially as $e^{\lambda_1 t}$ with λ_1 as the largest Lyapunov exponent. If the error

per step is ϵ then at time t the error $E(t)$ due to the first step has become $E(t) \approx \epsilon e^{\lambda_1 t}$. This means that even with a small initial error, the error increases exponentially faster than the trajectory convergence itself. This also implies that only a few time integration steps are capable of predicting the state of the system. This seems like a severe limitation on the use of integration of chaotic systems. This is however not a problem for specific applications of numerical integration models for the following reasons.

If the system is being integrated for a fairly short period then the tolerance may be set such that the error is much smaller than the length of the periodic solution, as for example has been shown in the shooting method for locating periodic solutions [74].

Also, the nature of the chaotic attractor is thus that it is attracting in one specific direction. If the local integration error is reasonably small then the points estimated by the integration routine may not lie near the solution but they still approach the attractor. So at point x_k one wants to find the point $\xi_{t_{k+1}}(x_k, t_k)$ on the attractor $\xi_t(x_k, t_k)$ (see figure 4.4.3 on the next page). The effect of the integration error is that the next point x_{k+1} lies on the attractor $\xi_t(x_{k+1}, t_{k+1})$ instead. If the error is sufficiently small than the new attractor to which the system “hopped” is still in the basin of attraction of the chaotic attractor and all the points therefore approach the attractor.

Furthermore, the presence of nearby Unstable Periodic Orbits within the region $\frac{1}{\lambda_1(\mathbf{x}_{jn}, n)}$ (with $\lambda_1(\mathbf{x}_{jn}, n)$ denoting the magnitude of the expanding eigenvalue of the Jacobian matrix of \mathbf{x}) guarantees that subsequent orbits originating from this region follow closely the original orbit (see Chapter 2, subsection 2.11.1 on page 74). This means that if the integration error

4.4.3 Bugs

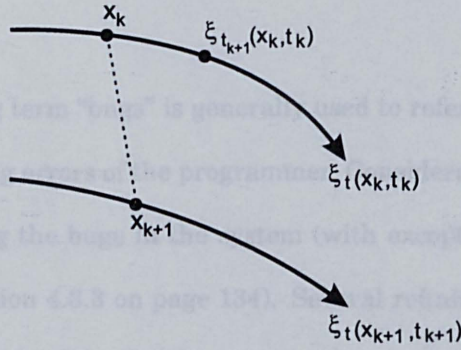


Figure 4.4.3: The phenomenon of trajectory “hopping” due to the local error at the step $k + 1$. From point x_k on the trajectory $\xi_t(x_k, t_k)$ the next point x_{k+1} is estimated to be on the trajectory $\xi_t(x_{k+1}, t_{k+1})$.

causes a jump to a different trajectory and if new trajectory is still within the said region, all subsequent orbits are close by. The resulting trajectory is then wandering between orbits but most trajectories will be so close to each other that the resulting attractor is reasonably approximated.

Another point is that due to the local behaviour of the system, which may be considered to act like a saddle point, any errors introduced in the system which pushes the system onto the unstable manifold, will result in an attracting step back towards the stable manifold. This mechanism is the basis of the OGY system of chaotic control. In the case of the control of chaos, a relatively large (compared to the integration error) perturbation introduces a targeted direction towards a specific trajectory. If the error causes the system to hop to a different trajectory the local manifolds are different to the original manifolds, but not in a different orientation. Therefore, the application of the control perturbation is still valid and the global behaviour is not dissimilar from the original attractor.

4.4.3 Bugs

The programming term “bugs” is generally used to refer to errors in a program due to coding errors of the programmer. Considerable effort has been put in eliminating the bugs in the system (with exception of the problem mentioned in section 4.3.3 on page 134). Several refining and redefinition steps have been part of the development process and special attention has been paid to make the system stable.

One particular problem was discovered when a change in the dynamic behaviour failed to appear when expected at the point where an earlier investigation had indicated it. It was due to a difference in the two programs due to the optimization of the code. Debug versions of the system may produce numerically different results, but the dynamic behaviour remains the same. The optimized version of the program still produces the same dynamic result but the numerical result is different and the timing may be different from the debug version. This is not a particular problem because we are mostly interested in the dynamic behaviour of the system and the changes in the behaviour. Even though the exact prediction of the trajectory is impossible, the general dynamic behaviour is the same as the original attractor.

4.5 File formats

4.5.1 Equation files

These are the files that contain a single model for straight forward integration, the default extension for these files is “.eq”. An example is given in

```
% Lorenz set of equations
<Parameters>
sigma=10
beta=2+2/3
rho=28

<Equation>
x'=-sigma*x+sigma*y
y'=-x*z+(rho*x)-y
z'=(x*y)-(beta*z)

<Initial>
x=5
y=5
z=1

<Options>
Duration=50
Output=1000
Integrator=Cash-Karp-RK
X-axis=x
Y-axis=z
X-range=-20:20
Y-range=0:50
GraphType=Point
3D-plot=x,y,z

<Name>
Lorenz

<Description>
Lorenz' set of equations
```

Figure 4.5.4: An example of an equation file. This implements the Lorenz equation model.

figure 4.5.4 on the preceding page. In all the files, white space is ignored, that is $x = (y * z) / 2$ is the same as $x = (y * z) / 2$. A line beginning with a percent sign %, regardless of white space, is considered as a comment, this can also be used to “comment out” bits of the model while developing a model. Furthermore, parameters and variables do not need to be declared before they are used. If an unknown variable is encountered by the parser it will be created and set to zero. Note that it is the duty of the user to initialise variables in the <Initial> section. There is no limit on the number of equations or variables, apart from memory limitations.

Parameters are set in the first section denoted <Parameters>. These are defined as Identifier=Value, where the identifier can be any length, case sensitive, name not beginning with a number. The value can be a number of any size or a simple evaluation that results in a value, e.g. $2/3$ or $(2 * \pi()) / 3$, therefore variables are not allowed in this section..

The second section describes the model equations, headed <Equations>. Note that this section has to come after the parameters section, otherwise the parser would need more than one pass to resolve undefined names. The format of this section is Identifier[']=Equation, where the identifier is an unique name of a variable that can be of any length. The optional prime symbol (') denotes a differential equation in time, without it the equation is considered to be a difference equation. Note that due to the difference of how the integrator handles differential and difference equations, it is required that difference equations are listed *before* the differential equations. Failing to do so, will result in an error message.

For example,

<Equations>

$x = x + 1$

$y' = x$

In the equation section any number of relations may be made, provided it fits on one line. The normal type of relations and priorities are used, ambiguous expressions may need brackets. For example, to have a value of 1.6666..., one would say $1 + 2/3$, as opposed to the value of 1 in the case of $(1 + 2)/3$.

The system also provides some standard and convenience functions that may be used in an equation (see table 1.1.1 on page 227). These function names are reserved, if the user attempts to use a function as a variable, this will result in an error message. The functions are used by giving the name of the function (in column one of table 1.1.1 on page 227) and the optional arguments in brackets. If a function does not have any arguments empty brackets are required anyway. For example, $x = \text{pulse}(t, 10, 10)$ or $y' = A * \sin(2 * \pi() * t)$.

The third section in an equation file describes the initial values of the variables defined in the equations section called <Initial>. The format of this section is Identifier=Value. Again, the value can be a number or a simple evaluation. If an identifier used in the equations section is not initialized here, it will be set to zero.

The fourth section <Options> allows the user to preset some programs options. When the equations file is opened, the settings will be used to initialise several options. The order of the settings is not important. The options are not required and this section can be empty or left out of the file

altogether. The options are :

- Duration=Value

The value is the end value of the integration steps.

- Integrator=Name

This indicates which integrator to use, the three supported integrating routines are: Euler, Runge-Kutta or variable step Runge-Kutta Cash-Karp-RK.

- Step=Value

Step size to use when integrating, for the variable step Cash-Karp Runge-Kutta integrator, this indicates the initial step size.

- Output=Value

This indicates how often to update the user interface with the calculated data. The value is in milliseconds, when the output value is reached, the integration is paused to enable the user interface to update graphs and lists. This may be a lengthy process, depending on the amount of data generated and the available memory.

- X-axis=Variable

This indicates which variable to use on the horizontal axis of a graph. Although the program supports multiple graphs, only one may be defined in this file. This variable may be any variable defined in the equations section or the predefined variable Evolution that uses the integration steps as x-axis values.

- Y-axis=Variable, Variable, ...

This indicates which variable to plot on the y-axis of the graph. Any

number of variables is allowed, if an undefined variable is specified, the graph will be created, but nothing will be plotted.

- `X-range=Value:Value`

This indicates the begin and end values of the x-axis, if this option is omitted, the x-axis is auto scaled.

- `Y-range=Value:Value`

This indicates the begin and end values of the y-axis, if this option is omitted, the y-axis is auto scaled.

- `GraphType=Type`

This indicates how to plot the data, using a connecting line or plotting the values with a marker. Valid options are `Line` or `Point`.

- `3D-plot=Variable, Variable, Variable`

This will let the program create a three dimensional plot with the specified variable as x,y and z variables respectively.

The fifth section is the `<Name>` section, here the user may specify a name for the model. This name is used as the header of hard copies and exported data. The name has to fit on one line.

The last section is the `<Description>` section. This is an optional section where the user may supply additional information. This may be several lines.

4.5.2 Network files

These files are used to define a network of units, each of which may have been defined by a set of equations. The network files are split in a set

```
% Main settings file

Name=Feedback control two Lorenz attractors
Chart=Lorenz1[x],Lorenz1[z]
Chart=Lorenz2[x],Lorenz2[z]
Start=0
Stop=250.0
Integrator=Runge-Kutta
Update=1000

<Files>
Net=fbctwolorenz.nwf
Equations=fbctwolorenz.nwe
```

Figure 4.5.5: Example of a network settings file.

of three files with the following default extensions, “.nws” for the main network settings file, “.nwf” for the network definition file and “.nwe” for the network equations file.

4.5.2.1 Network setting files

These files (“*.nws”) define the framework of the network. It sets up the integrator and indicates which equation and network definition files to use. Several different options may be set inside this file. The options are listed at the beginning of this file without any header. See for an example figure 4.5.5. The following options are supported:

- Name=Text

This is an one line description of the model.

- Chart=Variable, Variable, ...

This creates a graph with the first variable on the x-axis and all subsequent variables on the y-axis. This option may be repeated for subsequent graphs.

- Start=Value

Integrator start value, if omitted it is set to zero.

- Stop=Value

Integrator stop value, if omitted it is set to 100.

- Step=Value

Integrator step size, if omitted it is set to 0.01.

- Integrator=Name

Indicates which integrator to use, the three supported integrating routines are: Euler, Runge-Kutta or variable step Runge-Kutta Cash-Karp-RK.

- Update=Value

Indicates the interval in milliseconds for updating the user interface.

- GraphType=Type

This indicates how to plot the data, using a connecting line or plotting the values with a marker. Valid options are Line or Point.

The only header in this file is the <Files> header. This header describes the name and optional path of the other two files, the network definition file and the equation file. These are required to enable the system to setup a network and load the appropriate equations. The only two lines in this section are

- Net=File

This describes the file and path of the network definition file.

```
% Automatic generated file
% EuNeurone - Version: 1.0.8.238 -
% File: fbctwolorenz.nwf

<Network>
Dimension={20.000000,20.000000,20.000000}
Origin={10.000000,10.000000,10.000000}
Count=2

<Units>
Unit0.Name="Lorenz1"
Unit0.Position={1.000000,1.000000,1.000000}
Unit0.RandomPosition="No"
Unit0.Equations=0
Unit0.Connections={1}
Unit1.Name="Lorenz2"
Unit1.Position={2.0,-2.0,-2.0}
Unit1.RandomPosition="No"
Unit1.Equations=1
Unit1.Connections={0}
```

Figure 4.5.6: An example of a network definition file. This defines a network of two units called “Lorenz1” and “Lorenz2”.

- Equations=File

This describes the file and path of the equations file.

4.5.2.2 Network definition files

These files describe the structure of the network. This file may be written manually or can be created automatically with the network editor provided in the system. If it is created automatically, the first line in the file will be the following comment: “% Automatic generated file”. A network definition file (*.nwf) has two headings, see for an example of this type of file figure 4.5.6. The first header “<Network>” describes the network and the number of units in the network. Valid entries are:

- Dimension={Value,Value,Value}

This entry sets the three dimensional structure of the network, all

units must be contained within this dimensionless space.

- `Origin={Value,Value,Value}`

This sets the origin, i.e. coordinate (0,0,0) within the dimensionalized space.

- `Count=Value`

This sets the number of units within the network.

The second header "<Units>" sets individual settings for each unit. This section has a more complex structure. Each entry consists of the keyword "Unit" followed immediately by the number of the unit whose properties are modified. This is followed by a dot "." and the name of the property that is set. This is again followed by an equal sign "=" and the value or values of the property. For example to set the name of the unit the following line applies to the first unit (starting with zero):

```
Unit0.Name="Lorenz1"
```

Note that the name is used to identify this particular unit in the equations, so it should not contain any white space. Valid properties for any unit are:

- `Name="Name"`

This sets the unit name, this has to be an unique name within the units. It has to be encompassed with double quotes.

- `Position={Value,Value,Value}`

This indicates the position of the unit in the network space. Although it has to be a valid position, it is currently only used to display a three dimensional plot of the network. It is mostly intended for future use.

- `Connections={Value,...}`

This indicates the connections this unit has with other units by their index number. This is intended for future use.

- `Equations=Value`

This is used to indicate which set of equations this unit uses in the equations file. This is obligatory, otherwise the network will not work.

- `RandomPosition="Boolean"` This sets whether the position should be randomized or not. If it is "Yes" then a random position within the network space will be generated. If it is "No" the `Position` property will be used. This is intended for future use.

4.5.2.3 Network equation files

The network equation file ("`*.nwe`") define the equations used for each unit. Only those equations that are required need to be defined. If each unit uses the same set of equations, only one set is required. The difference with the "`*.eq`" equation files is that the network equation files define all the equations of each unit in a network. The "`*.eq`" equation files define only one model that is not part of a network. The format of the network equation files is as follows, each set of equations is encompassed by the keywords `<Set>`, to indicate the begin of a set, and `<EndOfSet>` to indicate the end of a set of equations. For each different model a new set needs to be defined. Inside a set of equations, three subheadings are defined. These are basically the same headers as the headings in the single equation file ("`*.eq`"), see the subsection 4.5.1 on page 140. The subheadings are `<<Parameters>>`, `<<Equation>>` and

```

% Net equation
% fbctwolorenz.nwe

<Set>
<<Parameters>>
sigma=10
beta=8/3
rho=197.4
%lambda=100
lambda=0

<<Equation>>
x'=-sigma*x+sigma*y+lambda*(Lorenz2[x]-x)
y'=-x*z+rho*x-y+lambda*(Lorenz2[y]-y)
z'=(x*y)-(beta*z)+lambda*(Lorenz2[z]-z)

<<Initial>>
x=5
y=5
z=1
<EndOfSet>

<Set>
<<Parameters>>
sigma=10
beta=8/3
rho=211
%mu=1
mu=0

<<Equation>>
x'=-sigma*x+sigma*y+mu*(Lorenz1[x]-x)
y'=-x*z+rho*x-y+mu*(Lorenz1[y]-y)
z'=(x*y)-(beta*z)+mu*(Lorenz1[z]-z)
t'=1

<<Initial>>
x=5
y=5
z=1
<EndOfSet>

```

Figure 4.5.7: A network equation file, describing two sets of Lorenz equations with direct feedback from the other unit. The parameters “mu” and “lambda” can be modified to change the feedback strength.

<<Initial>>, where the double angle brackets indicate a subheader.

The first subheader describes the parameters used in this set of equations in the format `Identifier=Value`. The second section describes the equations in the format `Identifier['']=Equation`. The third section describes the initial values for the variables defined in the equation section. Two special features are available in this file. Firstly, an equation can refer to a variable in a different unit by the units name followed by the unit's variable within square brackets, e.g. `Lorenz1[x]`. Secondly, variables that exist in all units, with the same name, may be summed or multiplied directly within the equation. To do this four keywords are available:

- `SumAll`

Use this to sum all occurrences of a variable in all units and return that value, e.g. `SumAll[x]`.

- `SumAny`

Use this to sum together only the variable in units that have connections to the current unit, e.g. `SumAny[x]`.

- `ProdAll`

Use this to multiply all occurrences of a variable, e.g. `ProdAll[x]`.

- `ProdAny`

Use this to multiply only the variable in units that are connected to this unit, e.g. `ProdAny[x]`.

4.6 Conclusion

A reliable and useful tool has been developed to study single ODE systems and a dynamic network of ODEs. It is acknowledged that the exact estimation of the chaotic trajectory by numerical integration is not possible. However, the dynamic behaviour of the numeric estimation, inside the domain of attraction, is the same as the initial chaotic attractor. Changes in the dynamic behaviour, which is the main interest of this thesis, are still valid. These have been tested and verified by varying the system parameters and the time step.

Huxley type of neurons with added noise to make them excitable-stimulably.

The Hodgkin-Huxley models are generally of the form [eq. 28, 29, 30]

$$C_m \frac{dV}{dt} = -g_L(V - E_L) - g_K n^4 (V - E_K) - g_{Na} m^3 h (V - E_{Na}) \quad (5.1.1)$$

Chapter 5

Experiments

“We are what we think. All that we
are arises with our thoughts. With our
thoughts, we make the world.”

Buddha

5.1 Introduction

This chapter discusses several models, constructed and tested to enable the study of continuous time chaotic models. The first consideration that had to be made was if it was desirable to study a continuous system instead of a discrete system or map. It was decided that the map approach, though simpler and easier to implement, was not the right approach. This limits the set of useable models considerably. Another reason for using a continuous system was the desire to design a neuronal-like network, which naturally is continuous. The obvious choice would be to construct a network of Hodgkin-

Huxley type of neurons with added noise to make them oscillate chaotically.

The Hodgkin-Huxley models are generally of the form [44, 20, 38, 39]

$$i_m = g_L(V - E_l) + g_K n^4(V - E_k) + g_{Na} m^3 h(V - E_{Na}) \quad (5.1.1)$$

where i_m is the current over the membrane and with g as the conductance of an ion (leak conductance $g_L = 0.003 \text{ mS/mm}^2$, potassium conductance $g_K = 0.36 \text{ mS/mm}^2$, sodium conductance $g_{Na} = 1.2 \text{ mS/mm}^2$), V as the membrane potential and E as the reversal potential ($E_l = -54.387 \text{ mV}$, $E_K = -77 \text{ mV}$, $E_{Na} = 50 \text{ mV}$); also known as the equilibrium or Nernst potential. The functions n, m, h represent the gating, i.e. the opening and closing of the ion channels, in the membrane. The shape of the gating functions has been experimentally determined by Hodgkin and Huxley and depends on the opening and closing rate of the different ion channels [39]. They found that for the potassium channel the following relation exists

$$\begin{aligned} \frac{dn}{dt} &= \alpha_n(V)(1 - n) - \beta_n(V)n \\ \alpha_n(V) &= \frac{0.01(V + 55)}{1 - e^{-0.1(V+55)}} \\ \beta_n(V) &= 0.125e^{-0.0125(V+65)} \end{aligned} \quad (5.1.2)$$

The sodium channel has an additional probability h of opening and closing apart from the function m so the activation functions for the sodium channel are

$$\begin{aligned} \frac{dm}{dt} &= \alpha_m(V)(1 - m) - \beta_m(V)m & \frac{dh}{dt} &= \alpha_h(V)(1 - h) - \beta_h(V)h \\ \alpha_m(V) &= \frac{0.1(V + 4)}{1 - e^{-0.1(V+40)}} & \alpha_h(V) &= 0.07e^{-0.05(V+65)} \\ \beta_m(V) &= 4e^{-0.556(V+65)} & \beta_h(V) &= \frac{1}{1 + e^{-0.1(V+35)}} \end{aligned} \quad (5.1.3)$$

These functions are the result of curve fitting on data of voltage clamped local membrane current measurements. They obey the Nernst equation and Ohm's law and are idealizations of the mechanisms that produces changes in the membrane potential. Furthermore, the model is, as it is shown in equations (5.1.2) and (5.1.3), a stable system remaining at the stable resting potential of $\approx -65mV$. When a perturbation is introduced (e.g. an additional current) and the potential is raised above the threshold ($\approx 35mV$), the system will spike (a transient) and return to the stable point. By adding a noisy current into the system the duration of the interval between consecutive spikes can be made noisy.

This applies to a single point on a neurone's membrane. To determine the behaviour of the whole cell, one has to take into account the attenuation and delay of a potential as it propagates over the membrane sheet. This can be approximated by using the *cable equation* [20, 44]. A simplified neuron model that includes the cable equation and also takes into account the dendritic integration of incoming currents is a famous model proposed by Rall [82, 83]. This model allows the injected current to be calculated over the integrated length and radius of a dendritic tree. To more accurately calculate the current flow over different parts of the neuron, one can divide a neuron into multiple parts and model each part with different conductances and dimensions separately. These models are known as *multicompartment models* and can be very extensive and complex. Note, however, that the basic dynamics of the model is solely determined by the channel activation functions.

The Hodgkin-Huxley models and the subsequent developed models de-

rived from channel dynamics and membrane properties are approximations of the physical processes that determine the membrane potential. The nature of this biophysical mechanism under the conditions of patch clamp experiments, is to remain at the stable state and this is indeed what may be seen in the modelling. Leakage and a passing action potential (a spike) causes a perturbation away from the stable point. The system then goes back onto the stable state unless further perturbations are added. The system is ultrastable and therefore not suitable for the study of unstable systems.

By adding noise, the perturbations may occur noisily or even chaotically. It is important to recognize that this does not change the basic dynamics of the Hodgkin-Huxley model (no bifurcation route to chaos exists) nor does it add any useful property to the model that can not be seen without adding noise. Constructing extensive (e.g. multicompartment) models of neurons may possibly tell you something about the physical properties of the neuron under study, it will not tell you anything about the information it carries. For this reason the experiments in this chapter will concern themselves with a network that consists of intrinsically chaotic continuous equations.

To determine the period and the stability of an orbit, the orbit was considered to be stable if the time series was periodic and the periodogram produced the corresponding period significantly. Also, the periodicity of an orbit was determined by a Poincaré section and the corresponding periodogram. The periodogram produces the frequency components of the time series. If a specific period is present the frequency estimate has to be significantly larger than the background to be considered. Note that throughout

the results section, the period estimates may vary between experiments and even between runs. This is due to a combination of effects that are particular to the type of experiments performed in this thesis.

Firstly, the integration of the model is only a estimate of the general trajectory behaviour, this is explained in sections 4.4.1 on page 135 and 4.4.2 on page 137. This means that the qualitative behaviour of the integration is similar to the model, but due to the instability of the integration routine and the nature of chaos numerical results are inaccurate. This may be remedied by varying integration parameters.

Secondly, the periodogram uses a Fast Fourier Transform (FFT) to estimate the frequency components. The bin size of each frequency is estimated on basis of the number of sample points and a four times oversampling rate; e.g. for a typical time series of 100.000 points, the number of frequency bins is 200.000. This results in fairly accurate estimate range but in less “binning” of frequencies whose variance is similar. Therefore the resulting frequency may be accurate within several digits behind the decimal point, but this may mean that for different experiments the same orbit estimated may vary in length.

Lastly, due to the delayed feedback control, similar orbits may be stabilized with different delays and may have a certain amount of “drift”. This is caused by the behaviour of the network models, for example a competition or cooperation between two delay control mechanisms to control the system. When estimating the frequency of an orbit, a subsection of the time-series is used in which transients are considered to be absent. Throughout the results section the resulting period of an n -orbit is given in addition to the

basic period of the orbit.

The following experiments are presented

- A network model based on the Rössler system with adaptive delay (section 5.2.1.1 on page 161).
- Models with different and multiple delay feedback (section 5.2.1.2 on page 165).
- A model of a single Rössler system with varying parameter sets (section 5.2.1.3 on page 173).
- Delayed feedback models with external input (section 5.2.1.4 on page 182).
- A synchronization model of the Lorenz system (section 5.2.2 on page 188).

5.2 Results

5.2.1 Rössler model

The first model to be studied is based on the Rössler system of equations. The model has a different method of controlling the chaotic behaviour than direct delayed feedback. The model was devised as follows. The Rössler system is described as [90]

$$\dot{x} = -y - z \quad (5.2.1)$$

$$\dot{y} = x + \alpha y \quad (5.2.2)$$

$$\dot{z} = \beta + (xz) + \gamma z \quad (5.2.3)$$

with $\alpha = 0.2$, $\beta = 0.2$ and $\gamma = -5.7$. The parameter values are chaotic in a fairly large range of the parameter space. The behaviour of this system is

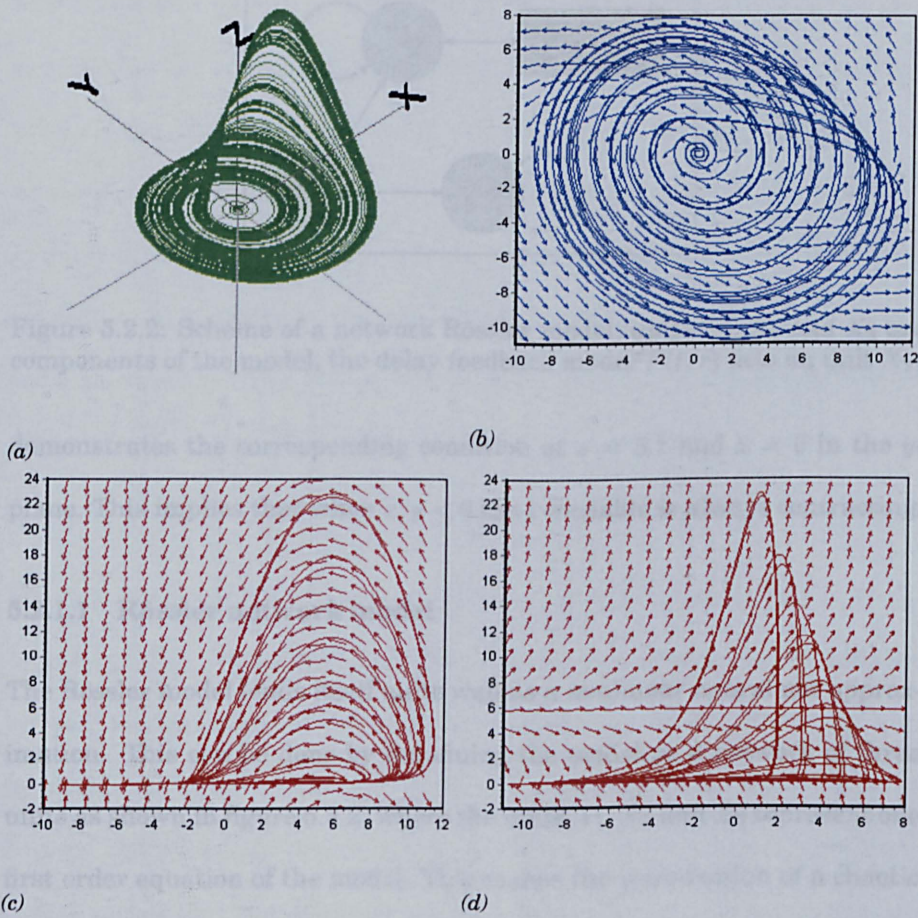


Figure 5.2.1: (a) Three dimensional plot of the Rössler equation; (b) Plot of y versus x ; (c) Plot of z versus x ; (d) Plot of z versus y .

described in phase space as (see figure 5.2.1 panel (a)).

As can be seen by the vector-field in figure 5.2.1 panel (b) and from the equations (5.2.1) and (5.2.2), the model is unstable in the x, y -plane, with a negative attractor at $(0, 0)$. The vectorfield of the x, z -plane demonstrates the z -isocline at $x = 5.7$ (figure 5.2.1 panel (b)), this can also be seen in equation (5.2.3) where the attractor contracts at $x < 5.7$ and expands in the z direction when $x > 5.7$. Furthermore, a similar mirrored condition exists below $z = 0$. In figure 5.2.1 panel (d) the vectorfield of the y, z -plane

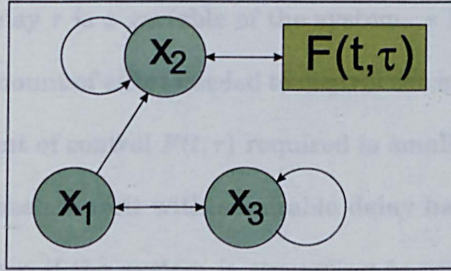


Figure 5.2.2: Scheme of a network Rössler model, units X_1 , X_2 and X_3 are components of the model, the delay feedback model $F(t, \tau)$ acts on unit X_2 .

demonstrates the corresponding condition at $z = 5.7$ and $z = 0$ in the y -plane. This implies that when $x, y < 0$ the z -variable is always contracting.

5.2.1.1 Rössler network model

The Rössler model lends itself quite well as a nonlinear neural net approximation. This can be done by redefining the model as a network of three units as shown in figure 5.2.2, where the units X_1 , X_2 and X_3 represent one first order equation of the model. This makes the introduction of a chaotic nonlinear system into a neural network much easier.

The equation of each unit is therefore

$$X_1 : \dot{x}_1 = w_{12}x_2 + w_{13}x_3 \quad (5.2.4)$$

$$X_2 : \dot{x}_2 = w_{21}x_1 + w_{22}x_2 + F(t, \tau) \quad (5.2.5)$$

$$X_3 : \dot{x}_3 = v + x_1x_3 + w_{33}x_3 \quad (5.2.6)$$

The external feedback force $F(t, \tau)$ is defined depending on the particular type of feedback required. In this model the delayed feedback on x_2 is defined according to [79, 80, 81, Pyragas] as $F(t, \tau) = k(x_2(t) - x_2(t - \tau))$.

Pyragas' method of chaos control has been modified so that the length

of the feedback delay τ is a variable of the system. τ has been made proportional to the amount of effort needed to control the system in its current orbit. If the amount of control $F(t, \tau)$ required is small, then no change is necessary in τ , since an orbit with a suitable delay has been successfully stabilized. However, if the system is struggling to control an orbit, then τ needs to be modified so that the length feedback delay is commensurate with an existing UPO of the system. The equation for adapting τ is as follows:

$$\frac{d\tau}{dt} = \alpha F \frac{dx}{dt} \quad (5.2.7)$$

where α is a constant. Equation 5.2.7 will be minimized when $F(t, \tau)$ becomes small and $\frac{dx}{dt}$ is periodic. This approach of estimating the amount of modification required to stabilise an orbit does however not provides us with a direction of change. Using this mechanism to allow the τ to become self adapting, several experiments were performed with different initial values for τ to locate controllable UPOs. In these experiments the parameter values are according to the original Rössler model (5.2.1), (5.2.2) and (5.2.3).

The result of the experiments with the networked Rössler model with adaptive delay are shown in table 5.2.1 on page 164. The system was started with a specific initial value for τ ; the value of τ then evolved depending on the strength of the feedback into a stable orbit or failed to reach a stable orbit within the time length of the evolution. The orbits that were found are a period one orbit with a period of 5.85 and a period two orbit of ≈ 11.72 . These values are in close agreement with the orbits found by [79, Pyragas] of 5.9 and 11.75. The experiment number 4 in table 5.2.1 produced

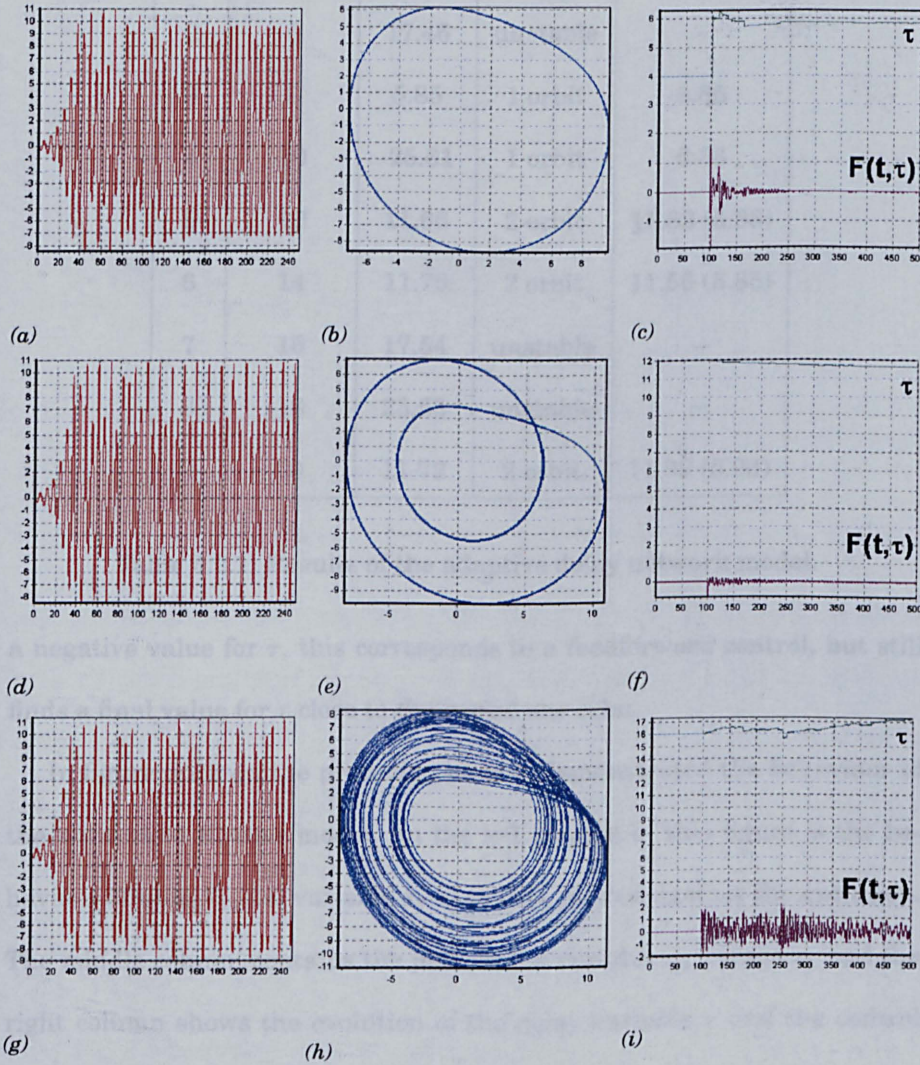


Figure 5.2.3: (a) Unit X_1 versus time, delay enabled at $t = 100$ with initial $\tau = 4$; (b) Plot of unit X_1 versus unit X_2 showing the period one orbit; (c) Value of τ and feedback control $F(t, \tau)$ in time, initial $\tau = 4$ converges to 5.84; (d) X_1 versus time, when initial $\tau = 12$; (e) X_1 versus X_2 showing the period two orbit; (f) τ and feedback in time, the initial $\tau = 12$ converges to 11.82; (g) X_1 in time with initial $\tau = 16$; (h) X_1 versus X_2 showing chaos; (i) τ and control $F(t, \tau)$ in time, initial $\tau = 16$ does not converge.

No.	Initial τ	Final τ	Orbit	Period
1	4	5.84	1 orbit	5.85
2	6	17.46	unstable	—
3	8	5.85	1 orbit	5.85
4	10	-25.61	1 orbit	6.33
5	12	11.66	2 orbit	11.82 (5.88)
6	14	11.76	2 orbit	11.56 (5.85)
7	16	17.54	unstable	—
8	18	23.53	unstable	—
9	20	11.72	2 orbit	11.92 (5.96)

Table 5.2.1: Results of the adaptive delay network model.

a negative value for τ , this corresponds to a feedforward control, but still finds a final value for τ close to the period one orbit.

In figure 5.2.3 on the preceding page is demonstrated the behaviour of the networked Rössler model. In the left column of this figure is the behaviour of unit X_1 , i.e. variable x_1 in (5.2.4), plotted against the evolution. The middle column contains the plots of the variable x_2 versus x_1 and the right column shows the evolution of the delay variable τ and the control function $F(t, \tau)$. The first three graphs in 5.2.3 on the page before (a), (b) and (c) show the behaviour when the initial value of τ was 4. When the control was enabled at $t = 100$, the value of τ quickly changes to ≈ 6 after which it converges to the stable orbit at $\tau = 5.84$. The second row of graphs show the behaviour of the system when the initial value was $\tau = 12$. In this case τ settles onto the second orbit, the period 2 orbit at $\tau = 11.82$. In the last

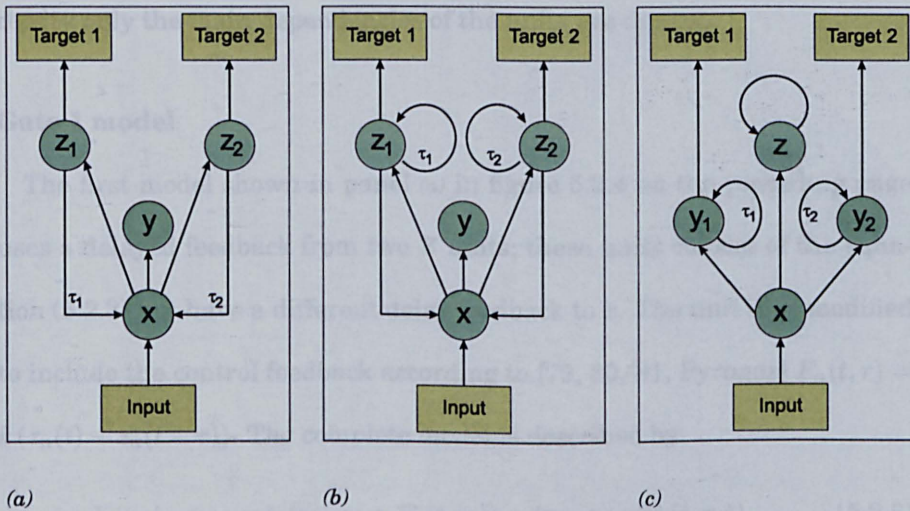


Figure 5.2.4: Schemes of possible gate mechanisms to direct input into different target areas. It shows only the main dependencies of units for clarity. (a) Different feedback lengths τ_1 and τ_2 from Z_1, Z_2 select different areas of output depending on input; (b) Alternative scheme with delayed autofeedback on z_1 and z_2 ; (c) Alternative scheme with auto-feedback on y_1 and y_2 .

row, the behaviour is shown when the initial $\tau = 16$. This does not settle onto a orbit, it may find a orbit if allowed to evolve for a long time, but due to the small changes in τ , it can be very long. The τ evolves according to the direction and strength of the feedback variable and may wander for a long time before converging.

5.2.1.2 Gate models

This section describes a set of experiments that test a hypothesis on how the input on a network of units may select one of two possible paths of output. The idea is that a network of three or four units may be connected such that a single input unit receives external input, one unit may function as a selector unit or as a inhibitor, and one or two units then may function as output unit. This is depicted schematically in the figure 5.2.4 where for

clarity only the main dependencies of the units are drawn.

Gate 1 model

The first model shown in panel (a) in figure 5.2.4 on the preceding page uses a delayed feedback from two Z units; these units consist of the equation (5.2.3) but have a different delay feedback to x . The unit X is modified to include the control feedback according to [79, 80, 81, Pyragas] $F_n(t, \tau) = k(z_n(t) - z_n(t - \tau))$. The complete model is described by

$$X : \dot{x} = w_{12}y + (w_{13}z_1 + F_1(t, \tau_1)) + (w_{14}z_2 + F_2(t, \tau_2)) \quad (5.2.8)$$

$$Y : \dot{y} = w_{21}x + w_{22}y \quad (5.2.9)$$

$$Z_1 : \dot{z}_1 = v + xz_1 + w_{33}z_1 \quad (5.2.10)$$

$$Z_2 : \dot{z}_2 = v + xz_2 + w_{44}z_2 \quad (5.2.11)$$

with the values of the parameters w_{nm} as in equations (5.2.1), (5.2.2) and (5.2.3) unless otherwise stated. As can be seen in the equations (5.2.8) the feedback acts on the variable x instead of the direct feedback of z . This is done to make the model more like a neuronal network. The weights w_{13}, w_{14} and k_1, k_2 determine the efficacy of a particular feedback. Subsequently, the system may then, when presented with appropriate input, stabilise one UPO as made available by the two feedback mechanisms. The resulting output of the units Z_1 and Z_2 may be different, projecting their state to an external output. This possible gating mechanism may be relevant in separating information by redirecting input into different target areas depending on dynamic behaviour.

Firstly, consider the case when only one feedback is active, e.g. $k_2 = 0$. This reduces the system (5.2.8) to a system where x has two direct de-

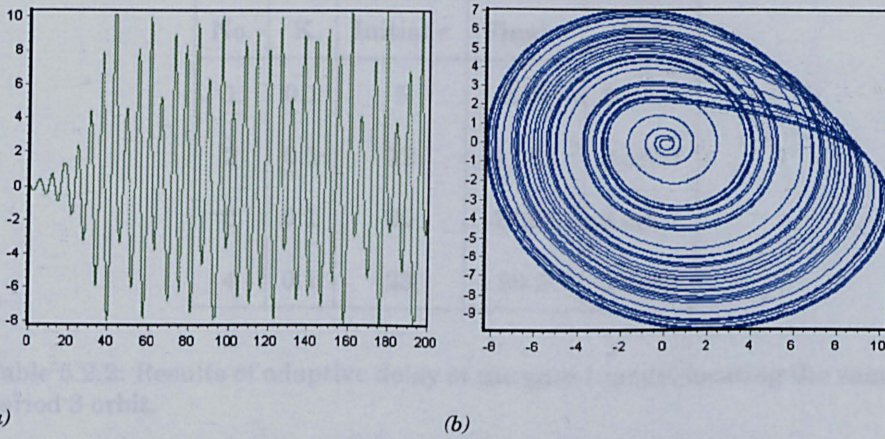


Figure 5.2.5: (a) Plot of x versus evolution showing chaos in gate 1 model; (b) Plot of y versus x of the chaotic gate 1 model.

dependencies on z in addition to the feedback mechanism. This results in a still chaotic system, without any feedback, that is similar to a Rössler with $w_{13} = -2$. This is shown in figure 5.2.5.

Secondly, consider the effect of a delayed feedback with a specific τ . It seems reasonable to assume that the modified system can still be stabilized into an UPO using a delayed feedback. This was tested with different values of τ as well as with the adaptive delay (5.2.7). An overview is given in table 5.2.2 on the following page. The resulting 3 orbit is consistently the same orbit with a period of ≈ 17.4 . If only one orbit is available for stabilization then the competition is not between different orbits but which delay feedback wins.

In figure 5.2.6 on the next page is shown the result of one of many experiments to stabilise one of the two delay feedback lines. In this model $k = 0.1$, $\tau_1 = 17.66$ and $\tau_2 = 5.67$. As can be seen in panel (a) of this figure, after control is enabled at time step 100, the model seems to stabilise into a 1-orbit of 5.98, however this is not at all stable and the model drifts into the

No.	K	Initial τ	Final τ	Orbit
1	0.1	5	5.67	3 orbit
2	0.1	12	11.63	3 orbit
3	0.1	20	17.66	3 orbit
4	0.1	23	29.5	3 orbit

Table 5.2.2: Results of adaptive delay of the gate 1 model locating the same period 3 orbit.

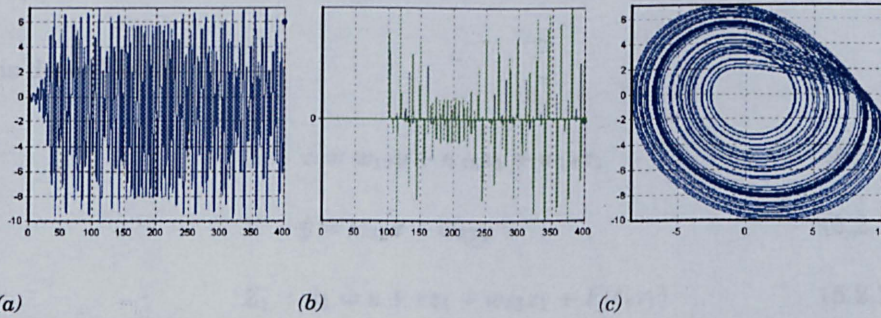


Figure 5.2.6: Gate 1 model results; (a) Plot of x versus evolution of the gate 1 model unable to stay in period 1 orbit; (b) Control functions F_1 and F_2 versus evolution of the same; (c) Phase plot of y versus x showing the period 1 orbit (thick line).

3-orbit. Later the 3-orbit may occasionally be visited by the trajectory but this is not stable and the model appears chaotic. In panel (b) can be seen the feedback F_1 and F_2 , that are smallest and periodic during the unstable 1-orbit, but appear to be more or less chaotic after that. In the last panel (c) of figure 5.2.6 the phase space of y versus x is plotted with the one orbit clearly visible.

Gate 2 model

An alternative gate model, as shown in panel (b) in figure 5.2.4 on page 165

No.	$K_{1,2}$	τ_1	τ_2	Orbit	Period
1	0.2	4	7	unstable	—
2	0.2	5	7	unstable	—
3	0.2	6	7	unstable	—
4	0.2	7	7	semi stable 2 orbit	—
5	0.2	8	7	2/4 intermittent orbit	12.20 (6.02)

Table 5.2.3: Some typical results from the Gate 2 model.

is described by

$$X : \dot{x} = w_{12}y + w_{13}z_1 + w_{14}z_2 \quad (5.2.12)$$

$$Y : \dot{y} = w_{21}x + w_{22}y \quad (5.2.13)$$

$$Z_1 : \dot{z}_1 = v + xz_1 + w_{33}z_1 + F(t, \tau_1) \quad (5.2.14)$$

$$Z_2 : \dot{z}_2 = v + xz_2 + w_{44}z_2 + F(t, \tau_2) \quad (5.2.15)$$

This is essentially the same model as the gate 1 model, but with the delayed feedback onto Z_n itself. Because of the different behaviour of the x and z variables, applying feedback onto x from z may introduce extra complexity. To see if this is the case and whether or not the results from the gate 1 model still apply, a set of experiments were conducted. Some typical results are shown in table 5.2.3, with the results of experiment 5, where $\tau_1 = 8$ and $\tau_2 = 7$, shown in figure 5.2.7 on the next page.

As is clear from the table the gate 2 model does not seem to add any useful dynamics to the system. This is not unexpected since the feedback control is more difficult to establish when a variable varies of a large range as the variable z does. The figure 5.2.7 on the following page shows an

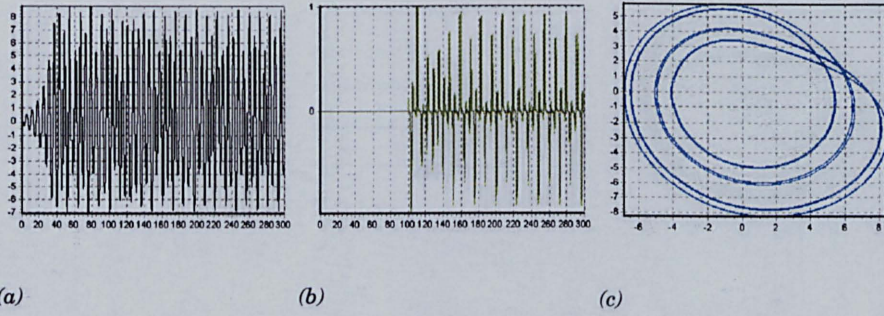


Figure 5.2.7: Gate 2 model results; (a) Time series of x showing intermittent 2 orbits; (b) Control F_1 and F_2 of the same experiment; (c) Phase plot of y versus x , after transients, showing the orbits.

experiment of the gate 2 model with $k = 0.2$, $\tau_1 = 8$ and $\tau_2 = 7$. This shows a time series that is a 2 orbit but may also be a 4 orbit composed of two different UPOs or a intermittence of two 2 orbits. The periodogram is unable to distinguish between the two, and the four orbit is not significant.

Gate 3 model

An different mechanism that is more readily constructed from the model point of view is shown in panel (c) in figure 5.2.4 on page 165. The corresponding model is

$$X : \dot{x} = w_{12}y_1 + w_{13}y_2 + w_{14}z \quad (5.2.16)$$

$$Y_1 : \dot{y}_1 = w_{21}x + w_{22}y_1 + F(t, \tau_1) \quad (5.2.17)$$

$$Y_2 : \dot{y}_2 = w_{31}x + w_{32}y_2 + F(t, \tau_2) \quad (5.2.18)$$

$$Z : \dot{z} = v + xz + w_{44}z \quad (5.2.19)$$

This gate 3 model produces much better results, the parameter values are still the same as in the original Rössler model. The double delay feedback control shows an extensive range of possible orbits that may be stabilized, so no negative effects seem to exist. The table 5.2.4 on page 172

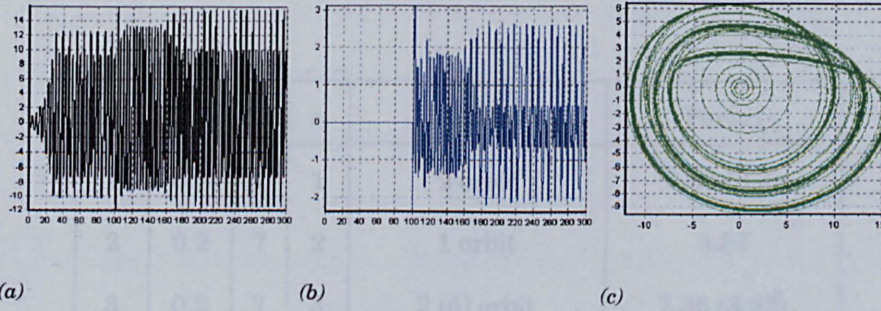


Figure 5.2.8: Gate 3 model; (a) Time series of x of experiment 10 with a temporary stabilising of a 1 orbit, later a two orbit; (b) Control functions F_1 and F_2 behaving exactly the same; (c) Phase plot of y_1 versus x showing both the one and the two orbit.

shows some results with different values for τ_1 and τ_2 , where the combined effect of τ_1 and τ_2 may stabilise different orbits that were not available in a single model. The units do not, however, stabilise into a orbit from only one τ .

A particular interesting experiment is experiment 10 in table 5.2.4 on the next page where $\tau_1 = 7$ and $\tau_2 = 10$. The result is shown in figure 5.2.8, where in panel (a) the timeseries of x is shown. Interestingly, the model seems to stabilise into a 1 orbit (period is 3.7) at $t = 120$ to $t = 160$, after which the gate 3 model switches to a 2 orbit from $t = 185$ onwards (period is 7.36). In panel (b) of the same figure, the feedback F_1 and F_2 behave exactly the same, this means that the resulting behaviour is again not the result of a competition between the two delay feedback control functions but of a collaboration of both delayed feedback control functions. In the last panel (c) the phase plot is shown of y_1 versus x (y_2 behaves the same as y_1), this shows the one orbit in the middle of the 2 orbit including the transient from the one to the two orbit.

The concept of a gating mechanism to direct information into different

No.	$K_{1,2}$	τ_1	τ_2	Orbit	Period
1	0.2	7	1	2 orbit	7.69 (3.8)
2	0.2	7	2	1 orbit	3.57
3	0.2	7	3	2 (5) orbit	7.35 (3.62)
4	0.2	7	4	unstable	—
5	0.2	7	5	2 orbit	7.46 (3.75)
6	0.2	7	6	semi stable 2 orbit	7.69 (3.8)
7	0.2	7	7	unstable	—
8	0.2	7	8	unstable	—
9	0.2	7	9	unstable	—
10	0.2	7	10	1 orbit & 2 orbit	3.7 & 7.36 (3.59)
11	0.2	5	1	stable point	—
12	0.2	5	2	1 orbit	3.26
13	0.2	5	3	2 orbit	7.00 (3.57)
14	0.2	5	4	1 orbit	4.08
15	0.2	5	5	stable point	—
16	0.2	5	6	2 orbit	6.55 (3.23)
17	0.2	5	7	2 orbit	7.40 (3.76)
18	0.2	5	8	1 orbit	4.11
19	0.2	5	9	semi stable 1 orbit	4.25
20	0.2	5	10	2 orbit	7.09 (3.49)

Table 5.2.4: Gate 3 model results.

pathways is still a good idea, however the gate models do not seem to be able of stabilising only one delay feedback. The models are apparently not capable of independent stabilization of one side of the system. Possibly the network needs to be extended to make each side of the system more independent. An interesting result is the ability of different feedback delays to stabilise whole new orbits that are not available for stabilization if only one delay is incorporated.

5.2.1.3 Single Rössler model

A different approach was to study more intensively the behaviour of a controlled Rössler model as a single unit. It was thought important to study the effect of different ways to control the system into a stable UPO and the parameter range within this is possible. To control the behaviour of the z -variable as the output variable, the delayed y -variable on the variable x as the controlling feedback was used. To achieve this, modify (5.2.1) as

$$\dot{x} = -(y - F(y)) - z \quad (5.2.20)$$

$$\text{with } F(y) = k(y(t - \tau) - y) \quad (5.2.21)$$

with the other two equations (5.2.2) and (5.2.3) remaining the same. An example of a stable 2 orbit using this method of control is shown in figure 5.2.9 on the next page.

As can be seen in this figure, the delay, enabled at timestep 100, stabilises an UPO with basic period 4.72 and the period 2 orbit of 9.53. When varying the delay τ or the feedback strength K , different orbits may be found as is shown in table 5.2.5 on page 175. If the combined effect of τ and K is strong enough to establish the control, a UPO is stabilized, if it is close

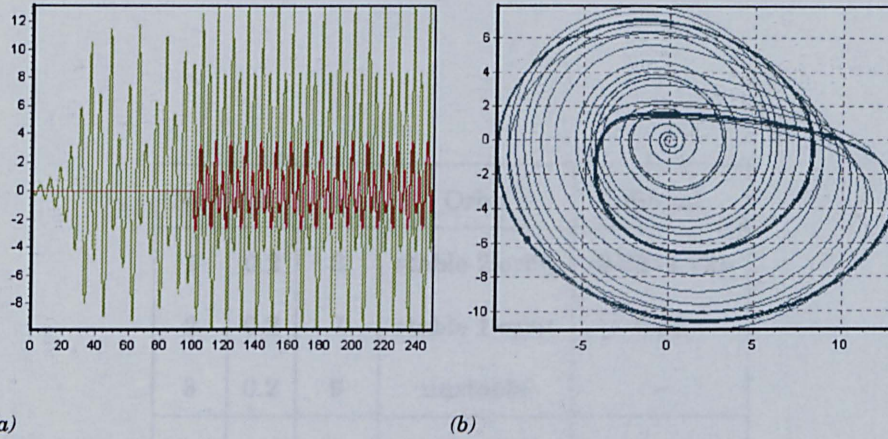


Figure 5.2.9: (a) Plot of delayed feedback controlled Rössler model variable x and $F(y)$, control is enabled at evolution step 100; (b) Plot of the delayed feedback controlled model x versus y .

to a UPO it is not stable: the trajectory drifts around the UPO. Usually, this means that a small modification to τ or K is necessary to stabilise the system. In rare cases the specific orbit may not be stabilized at all. Table 5.2.5 on the next page shows some values of τ and K with the resulting periodic orbit. The period of the UPO is calculated using a periodogram.

This model also demonstrates the mechanism of stabilization, when varying a specific parameter such as τ or K . In panel (a) of figure 5.2.10 on page 176, the control does not stabilise an UPO ($\tau = 6, K = 0.1$). In panel (c) of the same figure, a $2/4$ period is stabilized when τ is increased to $\tau = 7$. The panel (b) shows the intermediate case when $\tau = 6.9$ where the trajectory seems to “condense” into the UPO. This “condensation” is typical for the transition from unstable to a stable UPO. The panel (d) of figure 5.2.10 on page 176 shows the period 2 orbit with $\tau = 8, K = 0.1$. In panel (f) the 4 orbit is shown when $\tau = 9$. Note that this orbit is different from the $2/4$ orbit of $\tau = 7$. The panel (e) shows again the intermediate

No.	K	τ	Orbit	Period
1	0.2	3	stable 2 orbit	9.63 (4.72)
2	0.2	7	stable 1 orbit	5.85
3	0.2	9	unstable	—
4	0.1	3	unstable	—
5	0.1	6	unstable	—
6	0.1	6.6	unstable	—
7	0.1	6.9	unstable	—
8	0.1	7	stable 2 orbit	11.58 (5.85)
9	0.1	7.5	stable 2 orbit	11.08 (5.57)
10	0.1	8	stable 2 orbit	10.53 (5.26)
11	0.1	8.25	stable 2 orbit	10.37 (5.19)
12	0.1	8.5	stable 2 orbit	10.41 (5.15)
13	0.1	8.8	stable 2 orbit	10.35 (5.20)
14	0.1	9	stable 4 orbit	21.14 (5.09)
15	0.1	9.2	unstable	—
16	0.1	9.4	stable 5 orbit	26.04 (5.20)

Table 5.2.5: Effect of τ and K on the delayed feedback of y on x in the Rössler model. The period is the total calculated period of the orbit with the basic period in brackets.

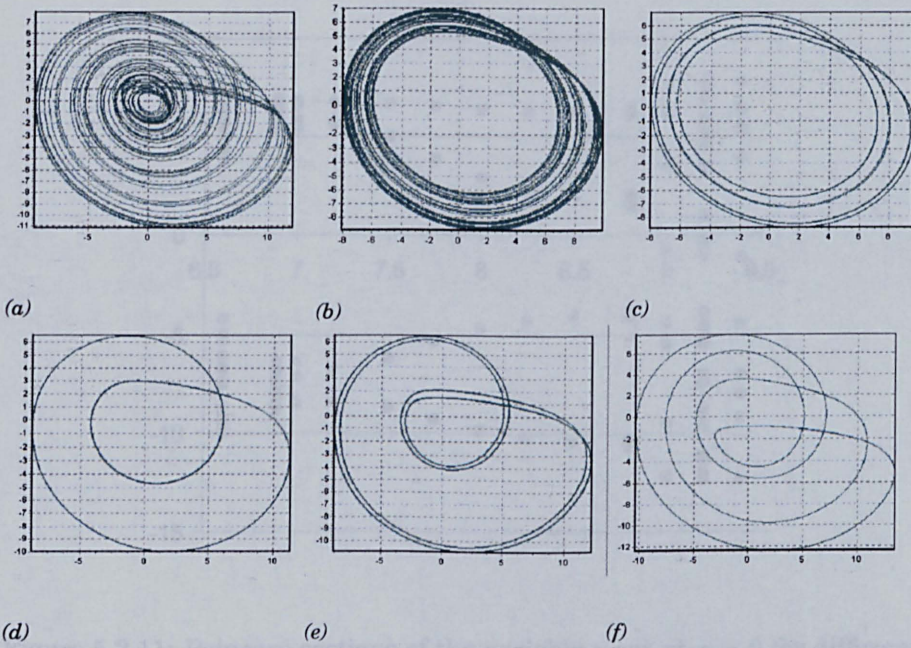


Figure 5.2.10: (a) No stable UPO at $\tau = 6$; (b) Condensation of trajectory into stable orbit shown in (c), $\tau = 6.9$; (c) Stable UPO at $\tau = 7$; (d) Stable UPO at $\tau = 8$; (e) Period doubling from UPO in panel (d) to the UPO in panel (f) with $\tau = 8.8$; (f) Stable UPO with $\tau = 9$.

case of $\tau = 8.8$ where the stable UPO of $\tau = 8$ splits into the stable UPO of $\tau = 9$. This process seems similar to a period doubling where the period of the $\tau = 8$ scenario, equal to ≈ 10.53 , doubles to ≈ 21.14 ; see table 5.2.5 on the preceding page. Also note, in the same table, the chaotic state at the intermediate value of $\tau = 9.2$.

To demonstrate the effect of τ on the stability of the UPO's, a set of thirteen Poincaré sections was made using different values of τ at different runs of the system. This is shown in figure 5.2.11 on the next page where the points indicate the position of the variable y when all transients are eliminated. On the x-axis of this figure is the value of τ while on the y-axis the value of the variable y is given. The dots indicate the point where the

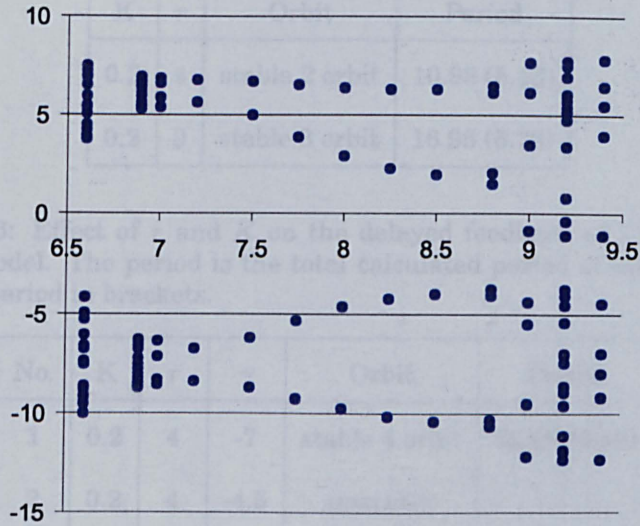


Figure 5.2.11: Poincaré sections of the variable y cut at $x = 0$ for different values of τ on the x -axis ($K = 0.1$).

y variable intersects $x = 0$, for the given model. The range of $\tau \geq 7$ to $\tau < 9$ clearly shows a set of stable 2-orbits with varying amplitude in y , but similar periods (see also table 5.2.5 on page 175). For the values of $\tau < 7$ no UPO can be stabilized for the value $K = 0.1$. A period four orbit is found at $\tau = 9$ and a period 5-orbit for $\tau = 9.4$, the intermediate value of $\tau = 9.2$ is unstable.

A different system can be made by using the variable z as delay feedback variable on the variable x . Because of the different nature of the variable z , the control as well as the dynamics of the control are completely different. To include the delay feedback control of z we change the equation (5.2.1) to $\dot{x} = -y - (z - F(z))$ with $F(z) = K(z(t - \tau) - z)$, this can then be rewritten

K	τ	Orbit	Period
0.2	4	stable 2 orbit	10.98 (5.13)
0.2	9	stable 3 orbit	16.95 (5.73)

Table 5.2.6: Effect of τ and K on the delayed feedback of z on x in the Rössler model. The period is the total calculated period of the orbit with the basic period in brackets.

No.	K	τ	γ	Orbit	Period
1	0.2	4	-7	stable 4 orbit	22.08 (5.48)
2	0.2	4	-4.5	unstable	—
3	0.2	4	-9	stable 3 orbit	16.42 (5.41)
4	0.2	4.3	-5	stable 2 orbit	9.78 (4.83)

Table 5.2.7: Effect of τ and γ on the delayed feedback of z on x in the Rössler model. The period is the total calculated period of the orbit with the basic period in brackets.

as

$$\dot{x} = -y + (K - 1)z + Kz(t - \tau) \quad (5.2.22)$$

Using (5.2.22) to control the orbit of the chaotic system whilst varying the parameters τ and K , only a few stable orbits can be found (see table 5.2.6). Other values for τ were tried but did not produce any results for this value of K .

A subsequent experiment to see if the variable z could be used as a feedback delay involved the change of the parameter value of $\gamma = -5.7$ in (5.2.3). Keeping $\tau = 4$ and $K = 0.2$ in first instance, some results are as in table 5.2.7.

This demonstrates that the basic frequency of the Rössler system (\approx

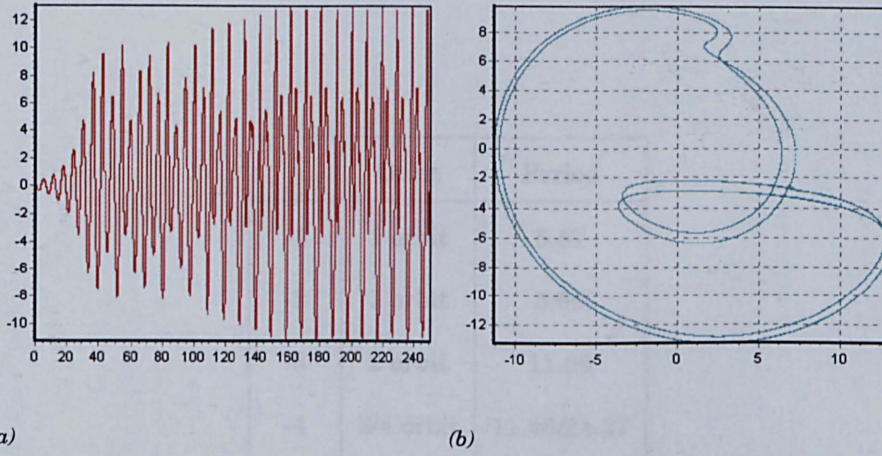


Figure 5.2.12: (a) Plot of delayed feedback control on variable x with $\gamma = -5$, control is enabled at evolution step 100; (b) Plot of the delayed feedback controlled model x versus y .

5.988) is due to the complex interaction of the system and not a mere parameter result. The modified systems used in table 5.2.7 on the page before were tested to be chaotic without control by setting the parameter $K = 0$. The resulting attractors were all sufficiently similar to the attractor when $\gamma = -5.7$. In experiment number 4 in table 5.2.7 on the preceding page the change of γ to -5 did not result in a stable orbit but by modifying τ to 4.3, a UPO could be stabilized. This is shown in figure 5.2.12. It is also interesting in this particular model that it would appear that the UPO is a 4 orbit, the duplicate orbit is however considered not significantly different from the original to be a separate orbit. The γ parameter appears to go through a period doubling cascade route to chaos as demonstrated in table 5.2.8 on the following page.

Let us consider a different parameter in (5.2.3) that describes the behaviour of the variable z within the delayed feedback model on z . This is the β parameter, that acts as a rate constant. If all the parameters are set

γ	Orbit	Period
-1	1 orbit	5.57
-2	1 orbit	5.69
-3	2 orbit	11.66
-4	2/4 orbit	11.46/24.27
-4.25	2/4 orbit	11.56/23.75
-4.5	chaos	—

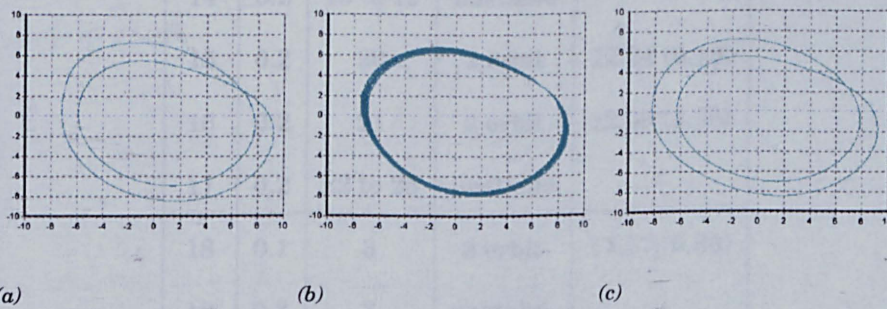
Table 5.2.8: Period doubling cascade of the γ parameter.

Figure 5.2.13: (a) Stable 2 orbit at $\tau = 8$; (b) Possible one orbit, $\tau = 8.5$ intermediate state to the 2 orbit in (c); (c) Stable 2 orbit, different from the orbit in (a), $\tau = 9$.

No.	K	τ	Orbit	Period
1	0.2	1	unstable	—
2	0.2	2	unstable	—
3	0.2	3	3 orbit	17.25 (5.77)
4	0.2	4	2 orbit	11.31 (5.65)
5	0.2	5	5 orbit	28.60 (5.68)
6	0.2	6	unstable	—
7	0.2	7	3 orbit	18.50 (6.11)
8	0.2	8	2 orbit	12.37 (6.11)
9	0.2	8.5	1 orbit	6.03
10	0.2	9	2 orbit	12.37 (6.11)
11	0.2	10	unstable	—
12	0.2	10.85	3 orbit	17.79 (5.93)
13	0.2	12	unstable	—
14	0.2	13 to 19	unstable	—
15	0.2	20	2 orbit	12.24 (6.16)
16	0.2	21	2 orbit	12.09 (5.95)
17	0.2	22 to 25	unstable	—
18	0.1	3	3 orbit	17.57 (5.86)
19	0.3	3	unstable	—
20	0.4	3	2 orbit	11.34 (5.65)

Table 5.2.9: Stability of UPO with varying K and τ of delayed z feedback on x with $\beta = 0.4$.

as in the original Rössler model, but β is increased from 0.2 to 0.4, the lack of UPOs that was found earlier with the delayed z feedback model is resolved as is shown in table 5.2.9 on the page before. An interesting case for these parameter values is to switch from one 2 orbit to a different 2 orbit when passing through the values from $\tau = 8$ to $\tau = 9$. As can be seen in figure 5.2.13 on page 180 panel (a) the orbit is a stable 2 orbit when $\tau = 8$, collapses into an almost complete one orbit when $\tau = 8.5$ (panel (b) in figure 5.2.13 on page 180) and then becomes a different 2 orbit when $\tau = 9$ (panel (c) in the same figure). The increase in the number of possible orbit that can be stabilized is considered to be due to the particular parameter values that combine to result in more orbits that can be stabilized. It is expected that with different parameter values (within the chaotic domain), for τ , K as well as α, β and γ , different sets of orbits may be stabilized. Because it is known that similar chaotic systems may easily synchronise each other [76, 92], models with different parameter values may still collaborate to generate specific output patterns and may even increase the number of orbits available.

5.2.1.4 Delayed feedback model with input

To study the effect of external periodic input on a delayed feedback model, experiments were performed that included an additional periodic input. Adding an external periodic input to one of the variables may also function as a stabilizing control force (see section 2.11.3 on page 79). If this is combined with the delayed feedback control, the stabilization of a UPO may occur if one of the controlling forces causes the system to be near an UPO

and the other does not function as a perturbation.

To demonstrate the effect of an external input combined with delayed feedback the following system was constructed

$$\begin{aligned}
 X_1 : \dot{x}_1 &= w_{12}x_2 + w_{13}x_3 \\
 X_2 : \dot{x}_2 &= w_{21}x_1 + w_{22}x_2 + F(t, \tau) + hP(t) \\
 X_3 : \dot{x}_3 &= v + x_1x_3 + w_{33}x_3
 \end{aligned} \tag{5.2.23}$$

Here $F(t, \tau)$ is again the feedback control of x_2 and $P(t)$ is a cosine input with varying frequency and scalar h . Recall from section 5.2.1.1 on page 161 that a period one orbit exists when $\tau = 5.85$ and a period two when $\tau = 11.85$. If for this system the feedback strength $k = 0$, i.e. no delayed feedback, and the period of the input is set to 5.58 the system does not stabilize into a UPO (panel (a) in figure 5.2.14). Changing the period of the input to 11.85 and the system still does not stabilize any unstable orbit. An external periodic force can stabilize a chaotic system if the period is close to a UPO and the amplitude of the force is sufficient but not too large. However, this is not the case here. This demonstrates that the subsequent results are not due to an external control force. To investigate the effect of changes in the input the two periods of input are combined in the following experiments.

If for the system (5.2.23) $\tau = 5.85$ and the external input is varied using the frequency pattern shown in table 5.2.10 on the following page the resulting behaviour can be seen in panel (a) of figure 5.2.15 on page 186. The system is first stabilized into a period one orbit ($t = 1 \rightarrow 100$), then the input with period 11.85 and amplitude $h = 1$ destabilizes the system ($t = 100 \rightarrow 200$). Subsequently, if the input period is set to the delayed

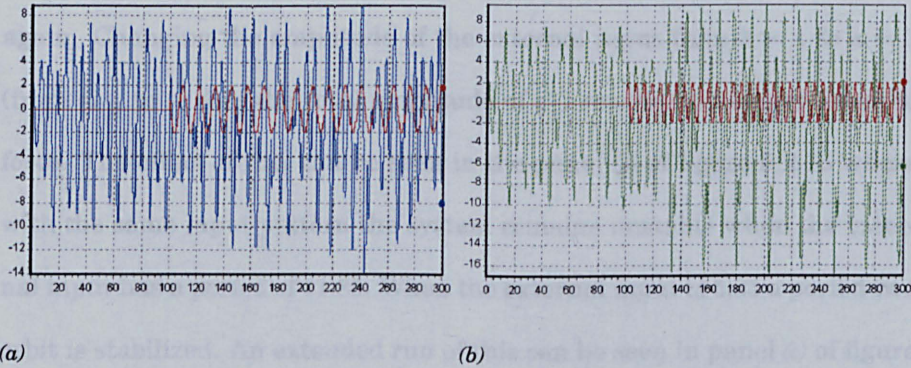


Figure 5.2.14: (a) Plot the Rössler model with external input of period 5.58, x_2 and input $P(t)$ versus time ; (b) Plot of the same model with input period 11.85, x_2 and input $P(t)$ versus time.

Start time	Stop time	External Period
0	100	0
100	200	11.85
200	300	5.85
300	400	11.85

Table 5.2.10: Frequency pattern used in the external input experiments.

feedback τ , i.e. 5.85, then the system returns to a period one orbit but with smaller amplitude possibly due to damping ($t = 200 \rightarrow 300$). The last time window ($t = 300 \rightarrow 400$) with external period 11.85 destabilizes the system again. Changing the amplitude of the external input from $h = 1$ to $h = 2$ (from $\approx \frac{1}{6}$ to $\approx \frac{1}{3}$ of the total amplitude of x_2) results in a larger external force. The effect of this can be seen in the panel (b) of figure 5.2.15 where with the same input pattern the system remains unstable when the external input has a period of 11.85. When the external input is 5.85 a period two orbit is stabilized. An extended run of this can be seen in panel (c) of figure 5.2.15 where after $t = 100$ an external frequency of 5.85 with amplitude 2 is added to x_2 . As can be seen in the panel (d) of the same figure (plotted x_1 versus x_2), the addition of the external input with the same period stabilizes a different period one orbit. An extended run with input period 11.85 and amplitude $h = 2$ stabilizes yet another period one orbit, but with an amplitude of $h = 1$ a period two orbit can be stabilized (not shown).

Next the effect of a periodic cosine input on the system, given that the feedback delay has not stabilized an orbit, is studied. Here, the delay was set to a value that did not result in a periodic orbit $\tau = 10$. The input was set to same frequency pattern and with amplitude $h = 1$ and $h = 2$ respectively. The corresponding results are in panels (a) and (b) of figure 5.2.16. Adding a cosine input with different amplitudes does not stabilize any orbits in this case. An extended run with frequency 11.85 and amplitude $h = 1$ shows that the system does not stabilize into a UPO but behaves as a forced chaotic system (shown in panels (c) and (d) in the figure 5.2.16).

These experiments show that the delayed feedback system (5.2.23) is

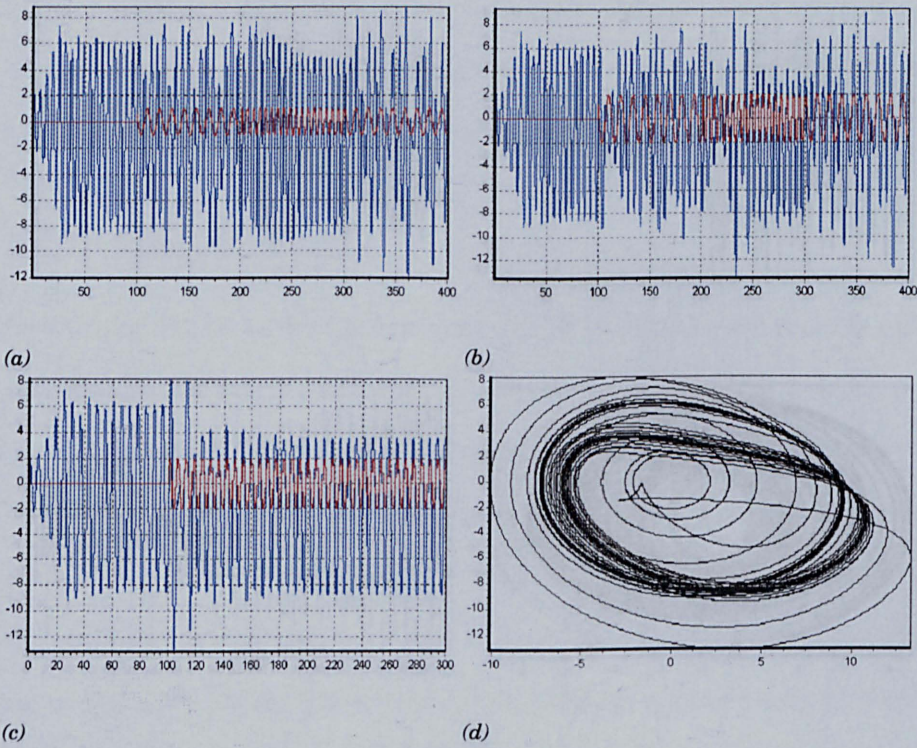


Figure 5.2.15: (a) Plot of delayed feedback controlled Rössler model with external input, x_2 and input $P(t)$ versus time ($\tau = 5.85$, $h = 1$); (b) Plot of the same model but with amplitude $h = 2$; (c) Extended plot of the model with external period 5.85 and $h = 2$; (d) Phase space of the extended run in panel (c) (x_2 versus x_1).

not merely controlled by the external force but is sensitive to the external input. If the input is near the basic frequency of the delayed feedback system (in this case 1.45) with appropriate amplitude then it can be stabilized. If the external input period and amplitude are not equal to the system may turn into a forced periodic perturbed system. The system may

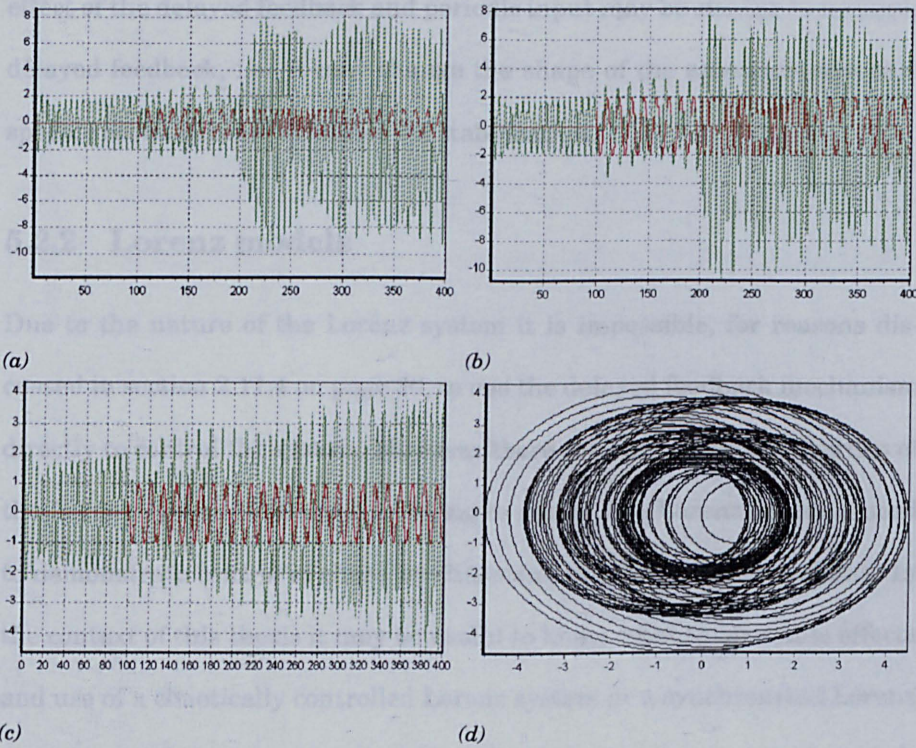


Figure 5.2.16: (a) Plot of delayed feedback controlled Rössler model with external input, x_2 and input $P(t)$ versus time ($\tau = 10$, $h = 1$); (b) Plot of the same model but with amplitude $h = 2$; (c) Extended plot of the model with external period 11.85 and $h = 1$; (d) Phase space of the extended run in panel (c).

not merely controlled by the external force but is sensitive to the external input. If the input is near the basic frequency of the delayed controlled system (in this case 5.85) with appropriate amplitude then UPO's may be stabilized. If the external input period and amplitude are not appropriate, the system may turn into a forced periodic perturbed system. The combined effect of the delayed feedback and periodic input may be similar to multiple delayed feedback, i.e. it may change the shape of the attractor such that specific orbits become available for stabilization.

5.2.2 Lorenz models

Due to the nature of the Lorenz system it is impossible, for reasons discussed in section 2.11.4 on page 80, to use the delayed feedback mechanism directly to control the system. However there are other useful properties of the Lorenz system that are interesting to study. The Lorenz model is used to demonstrate control of chaos by chaos and synchronization of chaos. In the context of this thesis it may be useful to know what the possible effects and use of a chaotically controlled Lorenz system or a synchronized Lorenz system can be. In particular the effect of an external input system on such a delay insensitive system may be useful.

Two Lorenz models can be constructed such that they control each other,

also known as control of chaos by chaos [47], e.g.

$$\begin{aligned}
 \dot{x}_1 &= -\sigma x_1 + \sigma y_1 + \lambda(x_2 - x_1) \\
 \dot{y}_1 &= -x_1 z_1 + r_1 x_1 - y_1 + \lambda(y_2 - y_1) \\
 \dot{z}_1 &= x_1 y_1 - b z_1 + \lambda(z_2 - z_1) \\
 \dot{x}_2 &= -\sigma x_2 + \sigma y_2 + \mu(x_1 - x_2) \\
 \dot{y}_2 &= -x_2 z_2 + r_2 x_2 - y_2 + \mu(y_1 - y_2) \\
 \dot{z}_2 &= x_2 y_2 - b z_2 + \mu(z_1 - z_2)
 \end{aligned} \tag{5.2.24}$$

with parameter values $\sigma = 100, r_1 = 197.4, b = \frac{8}{3}$ and $r_2 = 211$. If this is done by setting $\lambda = 100$ and $\mu = 1$, the two Lorenz attractors will control each other into a reduced attractor, this is discussed in section 2.11.5 on page 85 [47]. The most interesting part of this behaviour is the concept that a chaotic system may reduce the size and shape of an attractor in a similar system with different parameters. The conceptional idea could be a priming of a system into a specific region of the phase space where an UPO might be located to be stabilized. Another option would be the possibility of attractor switching, this process could be referred to in psychological terms as the attentive state.

A similar set of equations can be constructed, to synchronise one attrac-

Figure 5.2.17: (a) Variable x_1 versus x_2 of the Lorenz model, (b) Exact synchronization of x_1 versus x_2 and (c) x_1 versus x_2 in practical synchronization with transient.

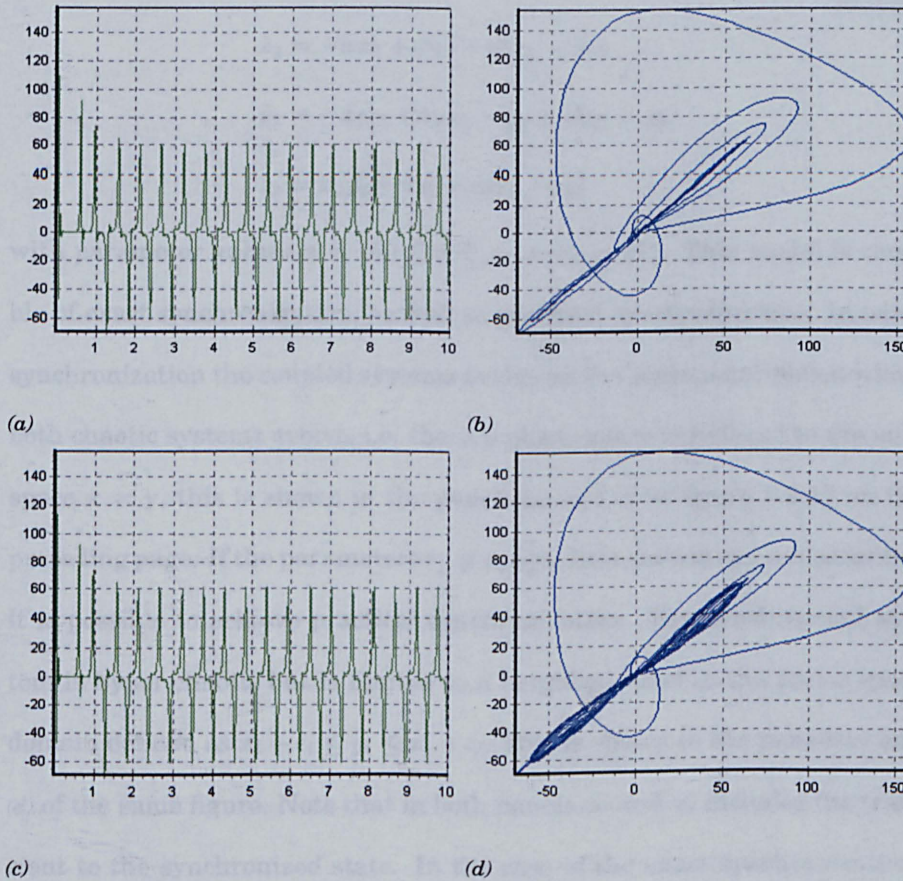


Figure 5.2.17: (a) Variable x_2 versus evolution of the exact synchronization model; (b) Exact synchronization of x_2 versus x_1 including transient; (c) Variable x_2 versus time in practical synchronization; (d) Variable x_2 versus x_1 in practical synchronization with transient.

tor onto another instead of controlling them. This model is, e.g. [47]

$$\begin{aligned}
 \dot{x}_1 &= -\sigma x_1 + \sigma y_1 \\
 \dot{y}_1 &= -x_1 z_1 + r_1 x_1 - y_1 \\
 \dot{z}_1 &= x_1 y_1 - b z_1 \\
 \dot{x}_2 &= -\sigma x_2 + \sigma y_2 + d(x_1 - x_2) \\
 \dot{y}_2 &= -x_2 z_2 + r_2 x_2 - y_2 + d(y_1 - y_2) \\
 \dot{z}_2 &= x_2 y_2 - b z_2 + d(z_1 - z_2)
 \end{aligned} \tag{5.2.25}$$

with parameter values $\sigma = 100, b = \frac{8}{3}, r_1 = r_2 = 211$. This model is capable of exact synchronization, as well as practical synchronization. In exact synchronization the coupled systems evolve on the same manifold on which both chaotic systems evolve, i.e. the x, y phase space is reduced to the subspace $x = y$, this is shown in the panels (a) and (b) of figure 5.2.17 on the preceding page. If the parameters $r_1 \neq r_2$ synchronization can not occur but it is possible to achieve practical synchronization. The synchronized system is hyper-chaotic but is limited to a neighbourhood in the phase space domain defined as $x_i + \epsilon_i \leq y_i \leq x_i + \epsilon_i$. This is shown in the panels (c) and (d) of the same figure. Note that in both panels (b) and (d) includes the transient to the synchronized state. In the case of the exact synchronization, the state is a straight line on $x = y$, in the practical case the synchronized state occupies more space.

Considering the manner in which both mechanisms seem to reduce the phase space similarly, using direct feedback; it may appear to be the case that in a practical situation the difference between a chaotically control target and a synchronized target is impossible to determine. Therefore,

they could be similarly used in the previous models of delayed feedback to first synchronise or chaotically control the system to target a certain subspace of the phase space, and subsequently control the desired UPO. Attempts to model this type of behaviour have not been very successful so far. The Lorenz model has the unfortunate property for these experiments to not only be delay insensitive, but also very robust. This is generally desirable in a system, but not if one wants to exploit the instabilities as intended.

A different aspect of the modelling of a network that selects an UPO on input, is the response of a system on specific input. Some earlier experiments discovered again the robustness of the Lorenz model but did find the ability to modify parameter values on input states. This means that parameters could be modified to reflect the input and thereby change the dynamics of the system (and making the parameter a variable of the system). Several parameters were relatively effective: these are the b and r parameters in the models (5.2.24) and (5.2.25). However, this means that the attractor is changed from one to a different (similar chaotic) attractor, which is not the original intention of this study. Changes in parameters are intended to provide the required dynamics within which the control may select a specific orbit. Other models with this approach were investigated but were not very successful.

A model that showed some interesting dynamics was the following:

$$\begin{aligned}
 \dot{x}_1 &= -\sigma x_1 + \sigma y_1 \\
 \dot{y}_1 &= -x_1 z_1 + \rho x_1 - y_1 \\
 \dot{z}_1 &= x_1 y_1 - b z_1 \\
 \dot{x}_2 &= -\sigma x_2 + \sigma y_2 + d(x_1 - x_2) \\
 \dot{y}_2 &= -x_2 z_2 + r x_2 - y_2 + d(y_1 - y_2) \\
 \dot{z}_2 &= x_2 y_2 - b z_2 + d(z_1 - z_2)
 \end{aligned} \tag{5.2.26}$$

$$\text{with } \rho = r + G(t)$$

where $\sigma = 100$, $b = \frac{8}{3}$, $r = 212$, $d = 1$ and $G(t)$ is a periodic input function that may be varied. In figure 5.2.18 on the next page is shown the behaviour if the input function is varied, first the first 2500 evolution steps with period $\pi = 3.14 \dots$ with amplitude 4, next for another 2500 evolution steps with period $2\pi = 6.28 \dots$ with amplitude 12. The panel (a) shows the last 1000 evolution steps of x_1 and x_2 , to demonstrate the shape and period of the Lorenz equations. The panel (b) shows z_1 and z_2 , note that even though the variable ρ is a variable of y_1 and y_2 the periodicity of the input is clearly amplified in the z_n variables. Panel (c) shows the phase space of y_2 versus y_1 , this demonstrates the periodicity of the model, but also that the change in frequency and amplitude of the input has little effect on these two variables. The last panel (d) of figure 5.2.18 on the following page shows the phase space of z_2 versus x_1 , this is more interesting because the shift in parameter input from π to 2π is clearly visible as an expansion of the attractor in the vertical direction.

5.3 Discussion

In this chapter the results of extensive numerical experiments with a continuous chaotic model are presented. The model has been successfully able to incorporate into a neural network

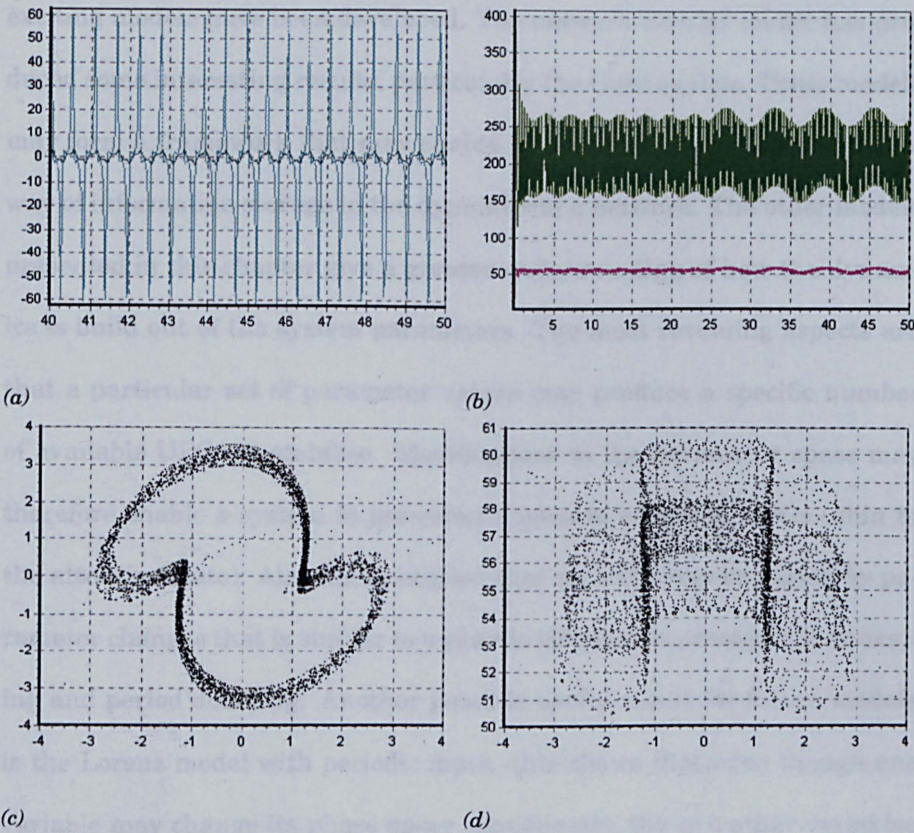


Figure 5.2.18: (a) Time series plot of x_1 and x_2 of model (5.2.26); (b) Time series plot of z_1 and z_2 of the same model; (c) Plot of x_2 versus y_2 with little change in amplitude of both variables; (d) plot of z_2 versus x_2 with an expansion of the space occupied by the attractor after an increase in amplitude and frequency of the input.

5.3 Discussion

In this chapter the results of extensive research into storing information in a continuous chaotic model are presented. Although no actual information has successfully been incorporated into a model, several interesting and exciting models have been developed. The network Rössler model has produced some interesting results, particularly the Gate models. These models may form a framework that may enable us to continue developing a novel way of information storage in the dynamics of a network. The other models presented in this chapter give a greater understanding of how the dynamics is build out of the system parameters. The most revealing aspects are that a particular set of parameter values may produce a specific number of available UPOs to stabilise. Modifications to the parameter space may therefore enable a system to pre-select a specific subset of UPOs (akin to the attentive state). Also the indication that an UPO may be varied by parameter changes that is similar to a chaotic bifurcation process, i.e. annealing and period doubling. Another possible useful result for future models is the Lorenz model with periodic input, this shows that even though one variable may change its phase space significantly, the two other variables remain virtually unmodified. This indicated that a modification to a parameter necessary to stabilise specific orbits may not change the attractor in any other variable.

5.4 Future work

For possible future development of both model and theory, it is intended to focus onto the problem of storage and learning. How may a network respond to input and then change its parameters such that a UPO is stabilized. In particular the variables τ and K may need to be modified according to the input. Subsequent input is then required to train different orbits and reproduce the already available orbits. At the moment input is a simple periodic sine wave, but it may be necessary to study the correct shape of the input function to provide the desired response. Apart from the learning problem, a methodological issue may need to be resolved of the order and scale of parameter changes required to store information.

A completely different issue is the remaining problem of the correctness of the models used. The equations used to model are well known chaotic models, but these may not necessarily provided the correct type of dynamics required. Although some research has been done to redefine a neural network chaotically, these models need to be investigated. Future work may need to be focused in developing a new model with the dynamics required. This may be based on biological networks or abstractions thereof.

Chapter 6

Biological experiments

"In the study of ideas, it is necessary to remember that insistence on hard-headed clarity issues from sentimental feeling, as it were a mist, cloaking the perplexities of fact. Insistence on clarity at all costs is based on sheer superstition as to the mode in which human intelligence functions. Our reasonings grasp at straws for premises and float on gossamers for deductions."

Alfred North Whitehead (1861 - 1947)

6.1 Introduction

This chapter is the result of the collaboration between the neurocomputing group of the school of computing and mathematical sciences (CMS) and the behavioural neuroscience group of the school of biological sciences (BMS). The people involved were Dr. Nigel Crook and myself from CMS and Dr. Steve Ray, Michele Martin, Scott Border, Dr. Ken Howells, Baz Abbass and Len Bagnell from BMS. Initially, the aim of this collaboration was to

develop some simple experiments to test the dynamic information storage and processing hypothesis. This has subsequently become one of the main areas of research studied by Steve Ray's group. The problem of how to prove experimentally that information is dynamically stored and processed in the brain is not trivial. Three different aspects have to be considered when making an experimental choice.

Firstly, it is necessary to consider the biological level at which this problem may be approached. It would be ideal to be able to monitor the activity and adaptation of multiple networked neurons. These may then be analysed to demonstrate the dynamic properties involved in the network. This is a difficult issue. It requires a specialist experimental setup (eletrophysiological systems and nerve growing laboratory) not currently available at Brookes. Possible collaborations with laboratories that may provide the material are being considered.

Can support for a dynamic theory of information storage be provided using an animal behaviour model that is available to us? At first glance this seems not to be the case, since if a simple behaviour effect can distinguish between the synaptic theory and the dynamic theory, it would seem likely that such an effect would already have been identified. However, this is not the case, presumably, because researchers were not looking for the dynamic behaviour of information storage and retrieval. The interpretation of results with the synaptic theory in mind might also induce a certain bias towards that theory. This is how science progresses, but it makes it difficult to introduce radically new and alternative ideas.

Secondly, it is necessary to consider what form the experiments will

take. An ideal experiment would identify the dynamic behaviour in an easy, readily reproduced and reliable experiment, without falling into the trap of behavioural difficulties that might feasibly explain the effects without confirmation of the dynamic contribution. The type of experiment and subject has been decided to be a Taste Aversion (TAV) learning task in the rat. The advantages of this experiment are the reliable training, the ease of experimental reproduction and the one trial learning. One trial learning means that an animal may be taught a specific testable memory in only one exposure to the experimental situation, as opposed to other learning tasks that require repeated exposure to the experimental environment. This by itself might produce associated memories that are impossible to separate from the test memory.

Thirdly, it is necessary to consider the experimental situation. The choice of experiments have to be unambiguous so that any deviation from the accepted behaviour may be attributed to the experiment and not to other complicating conditions. For example, most learning tasks may require repetitive training of the animal, it is therefore impossible to distinguish between the efficiency of learning and the efficiency of retrieval. To deal with this, a stepwise approach was used in which each experiment was evaluated and if required retested with an alternative method to provide collaborating evidence of the test situation. This has been remarkably successful and provided a clear insight into the behaviour of the dynamic memory created in the rats.

As will be shown in the experimental results section 6.3 on page 202, the establishment of the memory was reliably and easily performed. Also,

no extinguishing of the memory occurred over time. Furthermore, it was shown that the trained memory was plastic and phenomena such as generalization and specificity were found. Lastly, it was found that the memory required protein synthesis. Novel, different types of experiments were performed by Dr. Steve Ray et al. involving the transfer of specific brain regions from a donor to an acceptor animal. Because of the destruction of the network involved in making the transfer preparation, the original information due to specific network architecture and connections (i.e. synapses) are lost. Even so, it was conclusively shown that the memory was transferred to the acceptor, even when this process was repeated sequentially with another animal. Although this may be explained with alternative theories, it does not fit the classical synaptic theory, but the findings lend support to the dynamic information hypothesis. A complete discussion may be found at the end of this chapter in section 6.4 on page 212.

6.2 Methodology

The experiments in this chapter all use the relatively simple learning task Taste Aversion Learning [84]. They were performed by Steve Ray and his group. My contribution to the experiments was helping to design the experiments and interpreting the results. Taste aversion learning involves the presentation of a novel food substance to which rats naturally show a preference for, such as carrot, paired in a single training trial with a mild poison such as LiCl. Rats readily learn to avoid the novel food from this single trial and retain this memory over long periods. The conditioned taste

aversion was used to investigate different aspects of rat memory ability following a single training trial. Standard statistical procedures were used to determine significant levels between experimental groups and control groups [87].

6.2.1 Subjects

In all experiments at least 10 Sprague Dawley rats were used per group per experiment and all experiments were at least once replicated. All animals were between 90 and 180 days old at training. All animals were maintained on an 8 hour dark/16 hour light schedule with a constant temperature of $20^{\circ}\text{C} \pm 2^{\circ}\text{C}$. All had food and water ad libitum.

6.2.2 Training procedure

On the day of training all animals were removed from their home cage and placed in an individual cage with no food or water for 1 hour. Each animal then received a fixed amount of sliced carrot (depending on animal weight) for thirty minutes. Immediately after the thirty-minute exposure to carrot each animal received either a 3ml intraperitoneal (IP) injection of 0.3M LiCl (experimental) or a 3ml IP injection of physiological saline (control) and was returned to its home cage. Animals were returned to the test cage 24 hours after the initial training session, and after a similar 1-hour acclimatisation period were again presented with 20 grammes of carrot. Learning would be evidenced by reduced carrot consumption in the experimental animals compared to the control, saline, injected animals. For the cola drink test, animals were presented with a water bottle containing

50ml of cola drink and the amount consumed calculated over a 30-minute access period. This test was included to demonstrate that any reduction in carrot consumption (found in the carrot tests) represented specific learning of the paired association between carrot and illness, rather than a general debilitation of the experimental animals. After testing on both carrot and cola, animals were immediately returned to their home cages and given ad libitum access to food and water.

6.3 Taste aversion learning experiments

The animal behaviour model that was used is the taste aversion test. This is based on the fact that if a food or drink is presented to the test animal and simultaneously a negative experience is associated to the novel experience, the animal will retain this memory and refuse the associated food. So by presenting the test rats with a novel food, e.g. carrot, and just after that injecting them intraperitoneally with LiCl, the rats will feel sick and associated that with the carrot. This effect is shown in figure 6.3.1 on the next page, where the “Carrot + LiCl” group eats the least carrot, the two controls (“Carrot only” and “LiCl only”) eat equally much carrot as the untrained group.

6.3.1 Long time retention

This one trial learning is retained over the whole life of the rat. To demonstrate this, an experiment was performed where animals were tested at several intervals after learning (or not for the control) up to one week. This

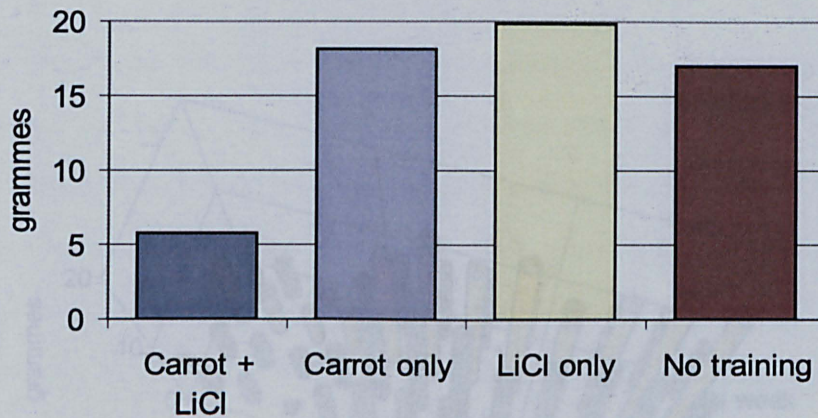


Figure 6.3.1: Taste aversion learning of carrot (in grammes); The experimental group “Carrot + LiCl” is significant lower than the controls.

is much shorter than the average life span (≈ 2 to 3 years) of a rat, but for practical reasons the maximum test duration of one week was chosen. In figure 6.3.2 on the following page is shown the test after 30 minutes, 60 minutes, 90 minutes and one week. Clearly, the experimental group “Carrot + LiCl” eats significantly the least carrot compared with the two control groups, even after one week. Longer time retention of the memory is shown in figure 6.3.3 on the next page, where an experimental group is compared with three controls after one day and after one month. The experiment group still eats the least amount of carrot, they retain the memory of the associated learning. This demonstrates that the one trial learning is a well established memory that does not extinguish after a reasonable amount of time (i.e. $\approx \frac{1}{24}^{th}$ of a rat’s average life time). Personal experience of the experimentalists confirm the retention of this memory for the rat’s lifetime.

6.3.2 Specificity

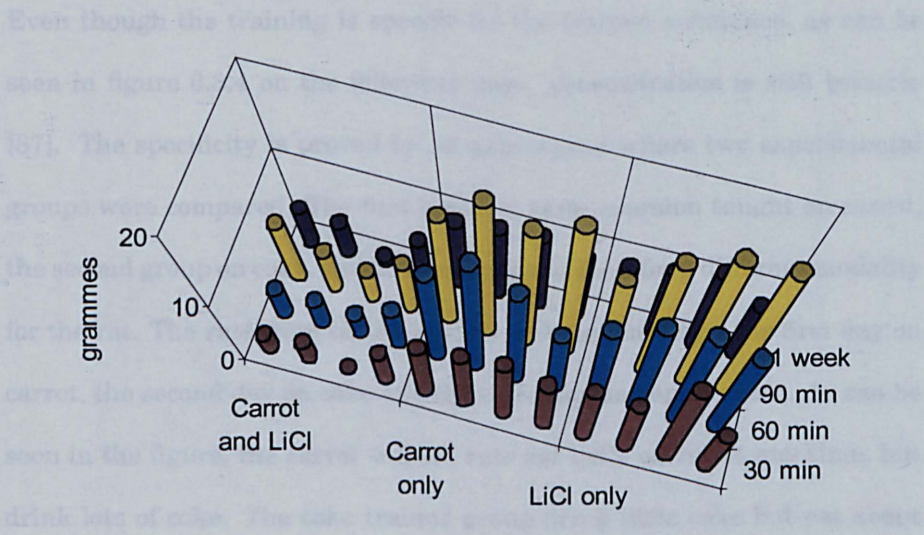


Figure 6.3.2: Retention of taste aversion memory tested after 30 min, 60 min, 90 min and one week (in grammes).

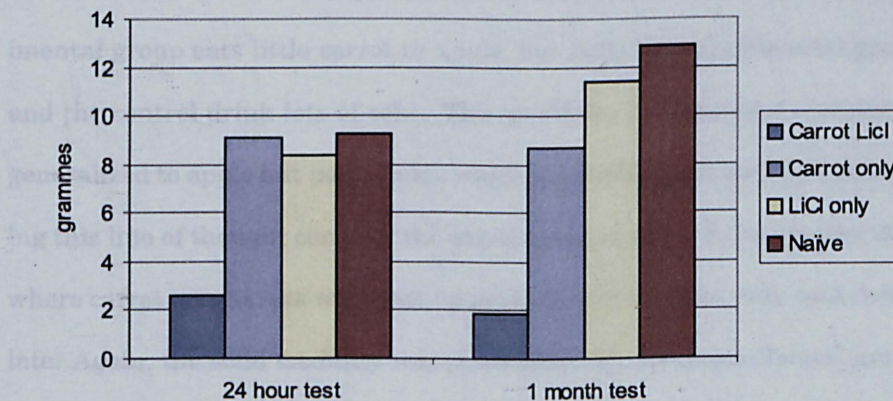


Figure 6.3.3: Retention of taste aversion memory after one day and a month, tested on carrot (in grammes).

6.3.2 Specificity

Even though the training is specific for the trained substance, as can be seen in figure 6.3.4 on the following page, generalization is still possible [87]. The specificity is proved by an experiment where two experimental groups were compared. The first group is taste aversion taught on carrot, the second group on coke, which is a liquid and therefore a different modality for the rat. The rats were tested in three subsequent days, the first day on carrot, the second day on coke and the third day on carrot again. As can be seen in the figure, the carrot trained rats eat little carrot at any time, but drink lots of coke. The coke trained group drink little coke but eat about the same amount of carrot as the controls. So it may be concluded that the modality of the trained substance determines the specificity. To see if this is exactly the case, consider the experiment of figure 6.3.5 on the next page. In this experiment a carrot trained group were compared with a control on the test substances carrot, apple, coke and carrot again. Clearly, the experimental group eats little carrot or apple, but both the experimental group and the control drink lots of coke. The specificity of the carrot training is generalized to apple but not to coke, which is a different modality. Continuing this line of thought consider the experiment of figure 6.3.6 on page 207, where carrot taught rats were testing against carrot, apple, coke and chocolate. Again, the solid modality was generalized by the experimental group to include apple and chocolate as the undesirable substance.

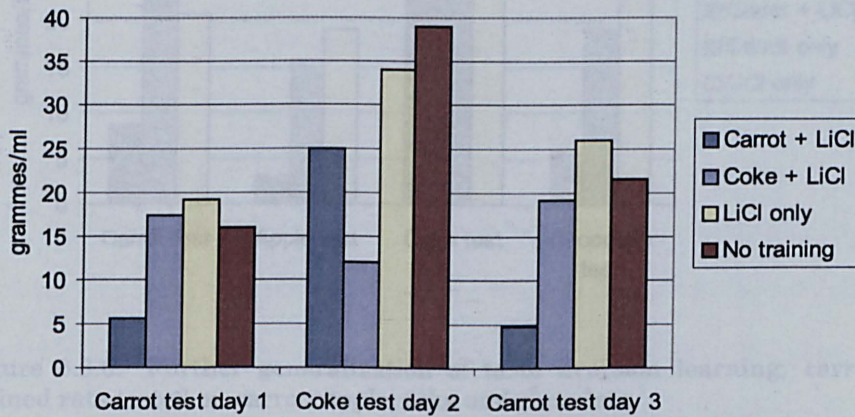


Figure 6.3.4: Specificity of taste aversion learning; three day test, first day test on carrot (in grammes), second day on coke (in ml), third day on carrot (in grammes).

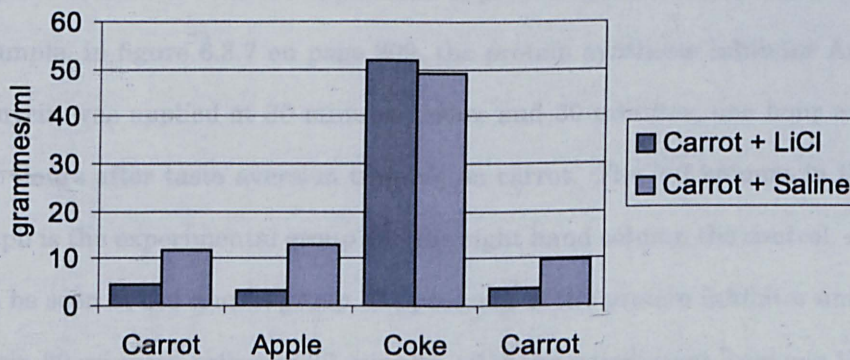


Figure 6.3.5: Continued specificity and generalization of taste aversion learning; carrot trained rats tested on carrot, apple, coke and carrot.

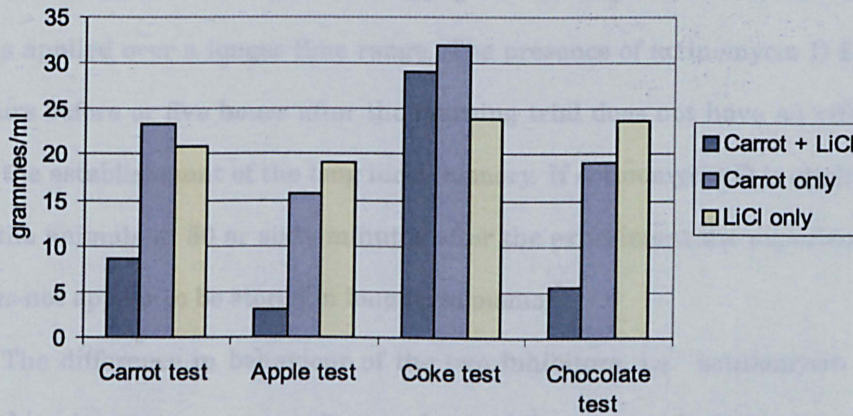


Figure 6.3.6: Further generalization of taste aversion learning; carrot trained rats tested on carrot, apple, coke and chocolate.

6.3.3 Necessity of protein synthesis

It is already well known that for long term retention of information in the long term memory, protein synthesis is required. To confirm that this is the case, several experiments were conducted where taste aversion learning was combined with the application of protein synthesis inhibitors. For example, in figure 6.3.7 on page 209, the protein synthesis inhibitor Anisomycin was applied at 30 minutes before and 30 minutes, one hour and two hours after taste aversion training on carrot. The left column in the graph is the experimental group and the right hand column the control. As can be seen in the control group, the presence of the protein inhibitor anisomycin 30 minutes before or 30 minutes after the experiment prevents the establishment of the taste aversion memory. If the inhibitor is present one or two hours after the experiment the memory is already established, and the animals avoid the carrot.

A different inhibitor actinomycin D was used in a similar experiment

shown in figure 6.3.8 on the following page. In this experiment the inhibitor was applied over a longer time range. The presence of actinomycin D five hours before or five hours after the learning trial does not have an effect on the establishment of the long term memory. If actinomycin D is applied to the animals at 30 or sixty minutes after the experiment the experience does not appear to be stored in long term memory.

The difference in behaviour of the two inhibitors, i.e. actinomycin D blocking long term memory after one hour while anisomycin does not, may be attributed to the different half times of these chemicals. The general effect of the protein synthesis inhibitors appears to be the prevention of the transfer of a memory in short term memory to long term memory. Apparently the storage of the memory requires protein synthesis to enable the brain to retrieve the memory.

6.3.4 Transfer experiments

To take a further step in studying the effects of protein synthesis and the synapses, the following experiments are conducted. Here it was investigated if a memory may be passed on by transfer of protein or mRNA molecules from a donor to a recipient. This type of experiment has been performed in the past with varying success, but not systematically using a specific, easily trainable and testable task, such as taste aversion learning. The procedure is as follows (see figure 6.3.4 on page 210): a test animal, referred to as the donor, is trained (or not as the case may be for the control) on a specific taste aversion task, e.g. on carrot. Subsequently, the donor is *not* tested but after sufficient time has passed to ensure the transfer of

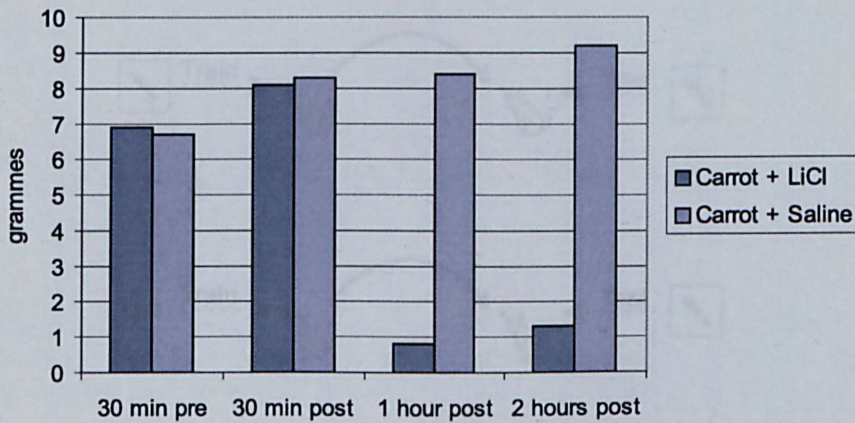


Figure 6.3.7: Taste aversion learning dependence on protein synthesis; At different times before or after learning application of Anisomycin.

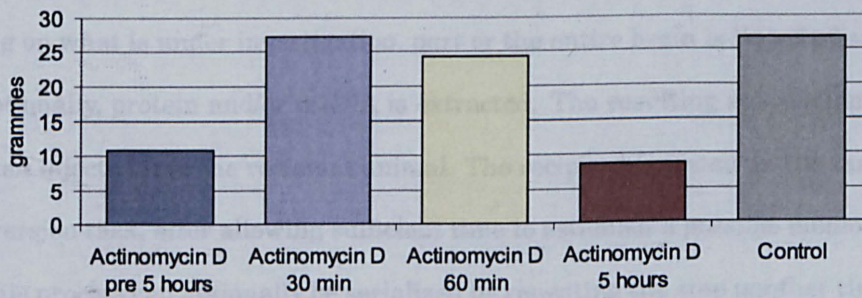


Figure 6.3.8: Taste aversion learning dependence on protein synthesis; application of Actinomycin D at different times before and after taste aversion test.

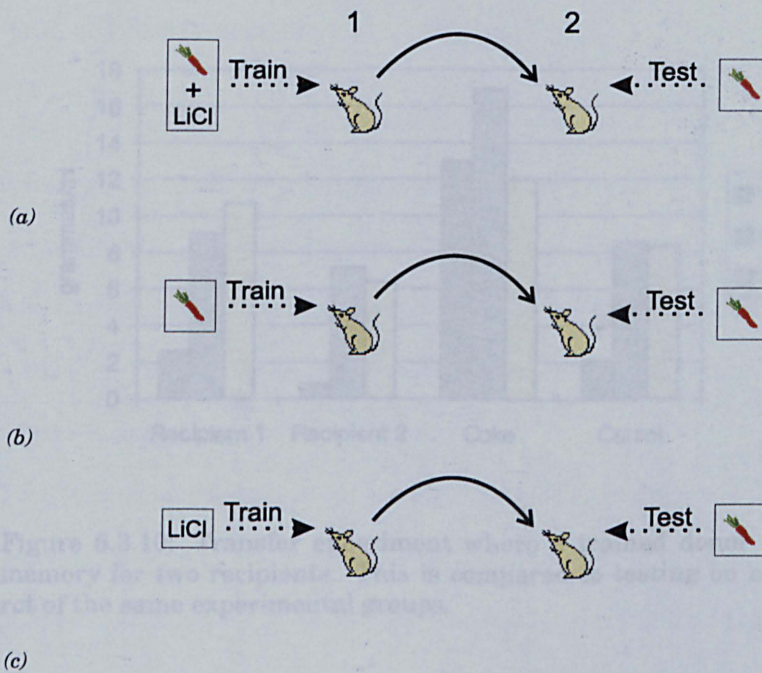


Figure 6.3.9: Transfer experiment scheme. The donor animal in column one is trained on one of three cases in this example. In (a) the donor is trained on carrot and LiCl, in (b) the donor is trained on carrot only and in (c) the donor is trained on LiCl only. After transfer the recipient in column two is then tested on carrot.

the memory to the long term memory, the animal is sacrificed. Depending on what is under investigation, part or the entire brain is liquefied and optionally, protein and/or mRNA is extracted. The resulting suspension is then injected into the recipient animal. The recipient is tested on the taste aversion task, after allowing sufficient time to establish a possible memory. This process can optionally be serialized by repeating the step another time to transfer a memory into a third animal, in this case the second animal is not tested on the memory to prevent the creation of a memory about this particular substance.

The results of a transfer experiment is shown in figure 6.3.10 on the following page. Here the comparison is made between the transfer of a

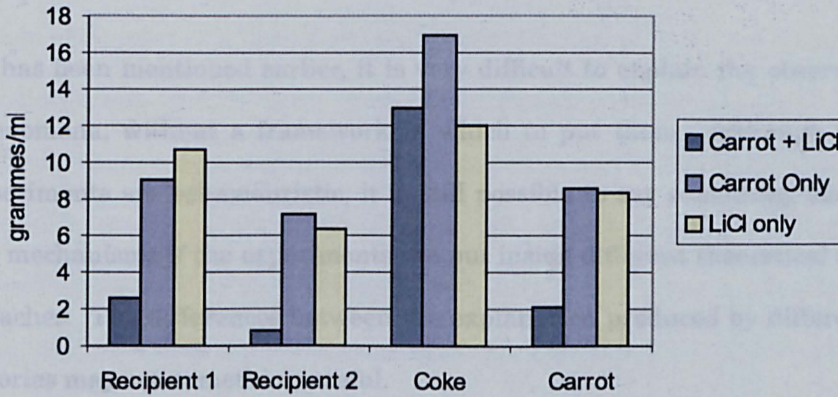


Figure 6.3.10: Transfer experiment where a trained donor provides the memory for two recipients. This is compared to testing on coke and carrot of the same experimental groups.

taste aversion memory from a donor to a recipient. The animals are trained on carrot. The experimental group “carrot and LiCl” is in both the recipients low, compared to the control groups as well as the normal experiment groups “coke” and “carrot”. Apparently, the transfer of a memory by this method is extremely feasible. This does call into question how this mechanism works, both the protein and/or mRNA need to be relocated to the relevant brain area, as well as incorporated into the network in a relevant way. Although the idea of local-coding (also known as the “grandmother cell”, i.e. a single cell which contains the necessary information about your grandmother) has already conclusively been disproved, it does seem similar to this concept [16]. The interpretation of this data is discussed in the next section 6.4 on the next page.

6.4 Discussion

As has been mentioned earlier, it is very difficult to explain the observed phenomena, without a framework in which to put them. Although the experiments are behaviouristic, it is still possible to say something about the mechanisms if the experiments are put inside different theoretical approaches. The differences between the explanation produced by different theories may say something useful.

If the experiments are considered within the synaptic weight theory, one may draw several conclusions. The training and establishing of the taste aversion memory experiments are easy to interpret within this framework, they merely reinforce the required network connection, i.e. synapses, to relate to the information presented (e.g. long term depression and potentiation [16, 38]). The long term retention experiment is more difficult to understand within the context of synaptic learning. If a single or several synapses contain the taste aversion information, how is this then maintained during the entire life span of the animal, considering the appearance of new information in the network as well as possible cell death or other network degeneration. It would appear that if a cardinal cell would die, the information stored in its synapses may be lost. In neural networks severing links between units does not reduce their performance much until a certain limit is reached. Removing connections and subsequent addition of units and connections requires additional training to locate the new global minimum.

Some work has been performed on neural cell death and it may be that information is recovered from the protein of the cell by its neighbours [5].

It has been suggested that the astrocytes may have an important role in this mechanism. The specificity and generalization of the taste aversion learning is also more difficult to understand in the synaptic theory framework. If a taste aversion animal trained on carrot is presented with a novel substance, say chocolate, it significantly eats less chocolate compared to control animals. This may be interpreted as feature extraction so that there is no information stored specifically about the carrot but about the experience of having a solid, novel food and becoming ill. This obviously requires several brain areas, specifically associate memory to become involved in forming the memory. Within the synaptic weight framework, the experiments where the addition of protein inhibitors fails to establish a long term memory, are easily interpreted by saying that the change in synaptic weights requires additional protein synthesis, e.g. ion channels.

The transfer experiments do not fit at all well within the synaptic framework. How would a single protein, injected into a naive recipient, know where to go and in which specific synapse to incorporate or which specific neuron to goad into synthesizing the protein concerned. If this was possible then the information stored is not in the synapse at all but in the protein itself, which has already been discarded as an unlikely mechanism for memory storage [16].

By re-examining the experiments in the framework of the dynamic information hypothesis, it may be possible to clarify some specific points. From this point of view the first experiments might simply establish the memory by stabilizing specific orbits within the network. The ease of training may simply be because the avoidance of noxious substances is fundamental

to survival. The long term retention experiments represent an important property of dynamic information storage, namely the ability to refresh information, without representation of the original information; as well as the ability to adapt memories or plasticity. This can be understood by imagining the network visiting the orbits occasionally due to the nature of the attractor. The network will represent the information associated by the orbit, but within external feedback the network will pass on and not linger in the orbit. Regulation of the feedback control is therefore a crucial element of a dynamic feedback network. If a specific neuron would die, the functionality of the network may be impaired but the information is not lost, only reduced. Reconstruction of the full memory may be achieved by representing the possible orbits (associated with memories) to neurons that are “untrained”. This may reestablish the orbits and the memories can be fully reproduced. The experiments about specificity and generalization may be understood by the capability of a dynamic network to reduce the possible number of orbits to a limited set. This then can work as a generalization principle were similar orbits are grouped together into one representation, the opposite can be achieved by allowing a separate classification input to determine if the input should be generalized or become a new group. The modality of the substance presented to the animals is important but not the actual flavour or smell. So coke presented to a carrot trained rat will not be associated with the taste aversion memory, but chocolate will. The function of proteins in the dynamic network hypothesis is very different from the synaptic weight. Even though the synaptic phenomena still occur, they represent a different functionality of the network. The proteins mod-

ify the synaptic behaviour (and other cell areas) not to change the weights that contain the distributed memory but to modify the dynamic behaviour of this particular neuron in the network that is attempting to establish the memory by stabilising the associated orbit. Protein synthesis is still crucial therefore, since it requires protein synthesis to change the channel and synaptic properties of a neuron such that the dynamic behaviour of the network and the possible parameter space has changed.

The memory transfer experiments are still a complicated problem. Within the dynamic network hypothesis it is possible to express an intuitive mechanism to explain this behaviour although no experimental or theoretical evidence exists. This explanation depends on the premise that in the dynamic hypothesis there is no need for an exact replication of the original network. Because neurons and networks are similar in organization and function, donor proteins that target the particular areas where the memory is going to be stored, need not be incorporated in the exact same location, because it most likely does not exist. However, the possible dynamical behaviour of the recipient network is similar to the original donor. This means that the addition of new proteins incorporated into the network, modify the dynamic behaviour, not exactly the same but closely similar to the original such that at the behavioural level the result is the same. This seems illogical, because one would like the exact memory to be reproduced. It is my belief that this is not true simply because the evolved mechanisms to recover from damage and reconstruct memories are not selected on accuracy per se, but on the fitness of the recovered memory to replace the function of the damaged memory. The new memory should produce a result accurate

Chapter 7

Discussion

"This is my simple religion. There is no need for temples; no need for complicated philosophy. Our own brain, our own heart is our temple; the philosophy is kindness."

Dalai Lama

7.1 Introduction

The aim of my research is to investigate the possibility of storing information dynamically in a neural network. The literature supports this idea and provides several models that attempt to do this. These models concern themselves with the stabilization of periodic orbits within a chaotic attractor. No models currently exist that store any relevant amount of information, but the models give different approaches to achieving that goal. Significantly, no theoretical reasons exist that object to the concept of information storage in chaotic networks.

To test our own models, it was required to develop an extensive software package. The development of this tool took up a major part of the time available. The reliability and theoretical correctness of the software is ensured and subsequent experiments are considered to be accurate.

In addition to the theoretical modelling of chaotic networks, in collaboration with biologists, several behavioural experiments were designed to find experimental evidence of dynamic information storage. These experiments indicate that the storage of information in memory is easily modifiable and transferable.

The models presented in this thesis provide some insight into the possible use of chaos in neural networks and offer a framework for future research in this area. The results indicate that using delayed feedback control, specific unstable periodic orbits may be stabilized, even in alternative network configurations. Localization of a periodic orbit by variations in time delay is demonstrated, the dependence on parameters of the ability of the control algorithm to stabilize specific orbits and the possible use of synchronization between different attractors to modify the available phase space in which orbits may be stabilized are also shown. The delayed feedback control of a network is sensitive to external periodic input which also influences the availability of orbits for stabilization. The stabilization of periodic orbits within a chaotic system is relatively easy, but the use of those orbits as representation of information is much more difficult. Further development of chaotic neural network may need to focus on the creation of available orbits for stabilization instead of the reliance on already available orbits. Interaction between different chaotic systems may determine

which particular set of orbits is available for control, thereby preparing the network for input or changing the sensitivity of the network for input.

7.2 Experiments

The experiments described in chapter 5 on page 154 are based on the concept of a chaotic system that becomes periodic by stabilising an unstable orbit or by synchronization. Earlier approaches to storing information in a chaotic network generally revolve around either a network whose mapping may be controlled into a stable state, or around a single neuron model whose chaotic behaviour is reduced into a stable state or orbit (see chapter 3 on page 92). One of the recurrent issues in constructing a chaotic neural network is the problem of the generation of the required dynamics. By extending a well known stable model with additional non linear components to destabilize the system a model may be constructed but the required dynamics may not exist or may behave inappropriately. Using a system that is already known to have the necessary dynamics, i.e. chaos and controllability, this problem may be averted but this also introduces a new problem that the dynamics are unlikely to be similar to those found in biological networks. The definition of a new chaotic model may be required. An intermediate solution would be to use biological signals or models thereof. These are, however, not responsive to changes in parameter space, although these signals may be used as network input.

Several types of control have been developed to stabilize unstable periodic orbits. Given the biological support for delayed feedback loops and

the fact that delayed feedback is the only control mechanism that does not require extensive knowledge of the orbit to control, the choice was made to use this type of control in the experiments. The successful use of the delayed feedback control is demonstrated in the modelling experiments.

When designing the models used in the experiments, the choice was made to use continuous systems that are known to be chaotic within certain parameters. These are the Rössler model and the Lorenz model. The concept is that even though these systems are not designed for this type of modelling, the basic properties provided by the model are valid and at least some of the resulting behaviour will be similar to chaotic network memory states. It is expected that a more appropriate model, specifically designed for this task, may produce more biologically relevant results. The results as shown in the experiment chapter describe the behaviour of single and networked models. The following conclusions may be drawn.

1. In the case of the single Rössler model and the network Rössler models, the introduction of delayed feedback at several different variables does not impair the basic chaotic properties of the model and the ability to control unstable periodic orbits.
2. In those models the delayed feedback is an extension of the model and the modification of the feedback parameters may result in bifurcations into stable, semi-stable and chaotic states.
3. The Rössler models also show that different parameter values of the feedback control may result in a different number of periodic orbits available for control. This may indicate a topological change of the

global strange attractor.

4. The gate models are exercises in attempting to control a particular orbit depending on input. The results are interesting but the gate models do not produce the single orbit intended but, given appropriate parameter values, result in new orbits available for control. Modifying the weights, to enhance the control of a particular orbit, does not work (unless the other weights are zero). This may be because, as has been shown in particular in synchronization models, even very weak coupling may result in a significant change in the model dynamics. It can therefore be said that the mechanism of a gate-like network may be the inclusion of specific orbits available for control, that are not available without the secondary feedback. These orbits may be part of the same attractor or even of a subspace of a different attractor.
5. The delayed feedback Rössler model with periodic input demonstrates that a controlled chaotic system is still sensitive to input and may stabilize new orbits.
6. The Lorenz models indicate that in a system where synchronization controls a specific variable of the model, other variables may remain unaffected, thereby only changing part of the attractor.
7. The combination of and the difference between the delayed control feedback and chaotic synchronization may be very important in chaotic neural networks. In the models used, as well as possibly in biology, both mechanisms may exist concurrently, their preconditions are satisfied in both situations.

7.3 Dynamic neural networks

This work has gone some way towards supporting the dynamic neural network hypothesis that states that information may be stored dynamically in a neural network. The experiments mentioned in the literature chapter 3 on page 92, already give a good indication of the ground work performed in this field. This thesis has extended this groundwork by demonstrating that UPOs are available for stabilization depending on the parameter set used. The use of multiple delayed feedback within a network extends the number of orbits that may be stabilized.

The biological experiments indicate that memories are not static but can be modified and even transferred. However, the ubiquity of dynamic information processing may not be always required. In some cases it may be possible that less complex information process system may perform the required calculations. In evolution any system that fits the requirements of environmental pressure will suffice. It is therefore feasible that only in certain parts of the brain dynamic information processing is performed and our knowledge of the processes in those areas at cellular level may still be inadequate to observe and understand these processes.

The modelling demonstrates several potentially useful features of chaotic neural networks but considerably further work is needed before a complete system can be developed. The network experiments show that a broad range of dynamics is available for manipulation but may need to be used differently. It may be, for example, that when a network has an orbit available for stabilization this is the ideal case. If a periodic signal is presented as input the network may need to try (or target) several different feedback

controls to change the global attractor to include the required orbit, synchronization may play a role in this.

7.4 Consequences

During the work on the software, the models and this thesis, I have had time to think over the dynamic neural network and the consequences of this hypothesis for the way our brain is considered to function. I have had many interesting conversations with people from a range of different backgrounds. From those conversations I think, several points are worth mentioning within the context of this thesis. These points are partly speculation and are by no means extensively discussed or in any way proven by the work in this thesis. However, they are intended to provide some ideas of what the dynamic neural network hypothesis might imply. Some relevant references are e.g. [4, 5, 16, 29, 32, 88, 92, 96, 104, Arbib, Churchland and Sejnowski, Freeman, Gazzaniga, Rolls, Schuster, Siegelmann and Traub].

7.4.1 Symbolism

One problem with the dynamic network is the lack of apparent symbolic representation of information. Symbols have traditionally been associated with intelligent thought, so how are they represented if no symbolic representation seems to exist in a dynamic neural network. From the point of view of the neurons, symbolism is not required at all, the global dynamic state is created by neurons that are active in collaboration with other neurons. Each individual neuron is not “aware” of this global state, but the

state only makes sense for other neurons to whom the information is presented. These again do not recognize it per se, but respond to the presented input with their own dynamic behaviour. The interactions of neural dynamics is the process of information representation. The attachment of symbols to a certain state is thereafter a process of association.

7.4.2 Genetics

The issue of how much of the development of the brain is genetically predisposed or the result of training is still much debated. The basic organization of a brain is very much predisposed, one brain is organized fairly similar to another, even across species. The difference lies in low level connections and intercellular communication. This may make the storage of specific information for which a predilection exists within certain areas much more likely. A network receptive for specific information, e.g. a skill, may be more likely to learn this skill if both the genetically determined organization as well as “random” components combine to make the network receptive for the required information storage. The dynamic information may be more readily stored if the weights and delays (and other elements that contribute to produce dynamic behaviour) are within the boundaries required to store the information. The availability of receptive networks may be modified by training.

7.4.3 Evolution

The evolutionary aspect of dynamic neural networks may be that a biological system is more responsive to the dynamics in the environment than to

the fixed state. The chemical ensemble of a neuron may exhibit complex molecular behaviour. The control of the biochemical state internally and response of the internal state to external input is complex and well organized, i.e. the evolutionary process has optimized the involved mechanisms. Responding to rapid changes in external input by projecting the internal state dynamically to other neurons is a fundamental behavioural feature of biological neurons. From the dynamic representation of single neurons a global network representation may be formed. The dynamic behaviour of the network is then capable of evolving into more complex networks with rapid development of the number of tasks that may be solved.

7.4.4 Consciousness

It has been accepted, mostly, that the phenomenon of consciousness is an emergent property of brain function. How this is actually achieved by the neuronal networks involved is not determined. The dynamic neural network hypothesis may indicate a way that allows the different ensembles of dynamic behaviour to establish a global state representing a conscious map. The mere existence of the dynamics does not constitute a conscious state, but the absence of the dynamics prevents any possible conscious representation. The establishment of a conscious map may be part of the global consensus of attracting states that exist within different parts of the brain. The different expressions of consciousness, such as awareness and even language may simply be different elements of the map. Each part of the global map associated with consciousness is filled in by the dynamic states of the brain. The presence or absence of parts of the map may be sufficient

to determine different levels within the conscious state.

7.5 Conclusion

The experiments shown in this thesis provide a framework for future work in this area. It has already been demonstrated by the experiments that control and synchronization can be made to interact with each other in ways that may be useful for information processing. It has also been shown that control of specific orbits strongly depends on the parameter values that make orbits available for stabilization. Not only is the dynamic neural network hypothesis a good working hypothesis for future development of advanced neurocomputing algorithms, it can also be used for modelling and interpreting biological processes.

Appendix A

Addendum

A.1 Supported functions

These are the names of the functions supported by the modelling program “Euneurone”. The first column indicates the function name as it is used in the equations, the second column indicates the corresponding standard function if it exists. The third column indicates the number of arguments required for that function. The last column describes the function.

Function name	Standard function	Number of arguments	Description
sin	sin	x	Calculates sine of x
cos	cos	x	Calculates cosine of x
tan	tan	x	Calculates tangent of x
asin	asin	x	Calculates arc sine of x
acos	acos	x	Calculates arc cosine of x

Function name	Standard function	Number of arguments	Description
atan	atan	x	Calculates arc tangent of x
atan2	atan2	x,y	Calculates arc tangent of y/x
sinh	sinh	x	Calculates hyperbolic sine
cosh	cosh	x	Calculates hyperbolic cosine
tanh	tanh	x	Calculates hyperbolic tangent
exp	exp	x	Calculates exponential e to x
ln	log	x	Calculates natural logarithm of x
log	log10	x	Calculates base ten logarithm of x
log2	log2	x	Calculates base two logarithm of x
abs	fabs	x	Calculates absolute value of x, i.e. $ x $
sqrt	sqrt	x	Calculates square root of x
mod	fmod	x,y	Calculates the modulo of x/y
hypot	hypot	x,y	Calculates hypotenuse of a right triangle
pi	none	0	Gives the value Pi (≈ 3.141592653)
rnd	none	0	Returns a random number between 0 and 1, i.e. [0,1]
random	none	x,y	Returns a random number between x and y, i.e. [x,y]
pulse	none	x,y,z	Returns a point on a square pulse at time x, y low time, z high time, period is y+z

Function name	Standard function	Number of arguments	Description
lthres	none	x,y,z	Returns z if $x < y$ else returns zero
hthres	none	x,y,z	Returns z if $x > y$ else returns zero
equal	=	x,y	Returns true if $x = y$ else false
AND	and	x,y	Returns true if x logical AND y else false
OR	or	x,y	Returns true if x logical OR y else false
XOR	xor	x,y	Returns true if x logical exclusive OR y else false
NOT	\neg	x	Returns true if not x else false
delay	none	x,y	Returns value of x at delay y, i.e. $x(t-y)$

A.2 File list

This file list is the complete list of source files of the modelling programme “Euneurone”. The first column in the table is the filename, the second the file size in bytes and the third is a short description of the file.

Filename	File size	Description
about.cpp	2,993	Displays “About” information
about.dfm	1,830	Form definition of “About” box
about.h	1,314	Header of about.cpp

Filename	File size	Description
compwin.cpp	1,177	Displays compilation results window
compwin.dfm	1,034	Form definition for compile results window
compwin.h	1,011	Header of compwin.cpp
connect.cpp	8,782	Displays network in 3D window using OpenGL
connect.dfm	1,282	Form definition of 3D network window
Connect.h	1,526	Header of connect.cpp
controlwin.cpp	6,978	General model start/stop window
controlwin.dfm	2,671	Form definition of control window
controlwin.h	1,551	Header of controlwin.cpp
debug.cpp	13,310	Window for debug output
debug.dfm	1,586	Form definition of debug window
debug.h	4,886	Header of debug.cpp
dynana.h	3,528	Header of analysis functions (DLL)
edit.cpp	4,757	Text window for editing equations
edit.dfm	1,697	Form definition of edit window
edit.h	1,993	Header of edit.cpp
EditConnect.cpp	4,663	Window to edit connections
EditConnect.dfm	1,697	Form definition of edit connect
EditConnect.h	1,857	Header of editconnect.cpp
editseries.cpp	1,556	Window for setting graph series properties

Filename	File size	Description
editseries.dfm	5,092	Form definition of edit series
editseries.h	1,625	Header of editseries.cpp
equation.cpp	20,846	Equation object module
equation.h	3,738	Header definitions of equation object
EuNeurone.cpp	2,721	Main program file
floating.cpp	27,438	Floating window for parameter settings
floating.dfm	5,825	Form definitions of floating window
floating.h	5,413	Header of floating.cpp
fnfit.cpp	1,517	Fits filename in specifies space
fnfit.h	328	Header of fnfit.cpp
formulc.cpp	42,167	Equation parser and evaluation module
formulc.h	4,413	Header of formulc.cpp
GlControl.cpp	36,583	OpenGL component module
GlControl.h	5,960	Header of glcontrol.cpp
graph.cpp	2,250	Graph window
graph.dfm	847	Form definition of graph window
graph.h	2,250	Header of graph.cpp
Integrator.cpp	18,352	Equation integrator module
Integrator.h	2,826	Header of integrator.cpp
main.cpp	51,695	Main window module
main.dfm	37,469	Form definitions of main window
main.h	6,343	Header of main.cpp

Filename	File size	Description
map.cpp	7,804	Graphical state module
map.dfm	1,633	Form definition of map.cpp
map.h	1,947	Header of map.cpp
Neteq.cpp	14,214	Network equations object
Neteq.h	3,338	Header of neteq.cpp
NetIntegrator.cpp	23,671	Network integration object
NetIntegrator.h	3,135	Header of netintegrator.cpp
NewEq.cpp	14,214	New equation file wizzard
NewEq.dfm	33,130	Forn definitions of neweq.cpp
NewEq.h	3,338	Header of neweq.cpp
newseries.cpp	1,391	New series window
newseries.dfm	2,976	Form definition of new series
newseries.h	1,459	Header of newseries.cpp
Plot3d.cpp	4,270	OpenGL 3D graph window
Plot3d.dfm	909	Form definition of plot3d.cpp
Plot3d.h	1,609	Header of plot3d.cpp
poin.cpp	6,026	Poincare plot window
poin.dfm	3,799	Form definitions of poin.cpp
poin.h	2,012	Header of poin.cpp
printchart.cpp	1,915	Print chart dialog
printchart.dfm	1,866	Form definitions of printchart.cpp
printchart.h	1,420	Header of printchart.cpp
recur.cpp	5,410	Recurrence plot window
recur.dfm	1,308	Form definition of recur.cpp

Filename	File size	Description
recur.h	1,820	Header of recur.cpp
scaleform.cpp	557	Scale graph dialog
scaleform.dfm	1,886	Form definition of graph dialog
scaleform.h	1,038	Header of scaleform.cpp
seriesform.cpp	548	Select series for 3D dialog
seriesform.dfm	1,880	Form definition of 3D dialog
seriesform.h	1,051	Header of seriesform.cpp
server.cpp	99,381	Service module to handle objects
server.h	9,840	Header of server.cpp
Settings.cpp	6,698	Read network settings module
Settings.h	1,436	Header of settings.cpp
spectrum.cpp	11,629	Spectrum window
spectrum.dfm	2,585	Form definition of spectrum window
spectrum.h	2,617	Header of spectrum.cpp
splash.cpp	2,024	Opening splash screen
splash.dfm	669	Form definition of splash screen
splash.h	1,073	Header of splash.cpp
stack.h	6,672	Modified STL template of stack class
stats.cpp	2,526	Statistics window
stats.dfm	1,797	Form definition of statistics window
stats.h	1,602	Header of stats.cpp
unitclass.cpp	36,800	Unit implementation module
unitclass.h	4,527	Header of unitclass.cpp
wait.cpp	1,684	Wait dialog

Filename	File size	Description
wait.dfm	452	Form definition of wait dialog
wait.h	1,043	Header of wait.cpp
Total size	715,054	(Size in bytes)

A.3 Evaluation function

This section describes the reverse polish notation evaluator of the equation parser. This function is optimized for speed and is part of the equation parser class “formuClass”. It accepts as input an array of unsigned integers, that contains the encoded function and the size of that array. The function returns the calculated value. It uses a predefined type from the Standard Template Library (STL) that implements a variable size stack of double values.

```
typedef std::stack<double, std::vector<double> > Db1Stack;
```

Note that exception handling of possible errors is performed globally by the formuClass class.

```

1  double __fastcall formuClass::value(unsigned int *function, unsigned int len)
2  {
3      double x,y,z;
4      double result;
5      unsigned int index;
6      unsigned int stksize=dblstack.size();
7      //get the current stack size, at the end of
8      //this function this must be the same.
9
10     if(len<1)
11     {
12         fset_error("empty coded function");
13         return 0; // empty function; result of an unsuccessful call to translate
14     }
15
16     for(index=0;index<len;index++)
17     {
18         switch(function[index])
19         {
20             case 0: break; //end of function
21             case CHARDOUBLE:
22                 { //push the next value on the stack:
23                     index++;

```

```

24         dblstack.push(ctable[function[index]]);
25         break;
26     }
27     case CHARVAR:
28     {
29         index++;
30         dblstack.push(aVars[function[index]].value);
31         break;
32     }
33     case CHAREXT:
34     {
35         index++;
36         result=aExtVars[function[index]].aVars[function[index+1]].value;
37         dblstack.push(result);
38         index++;
39         break;
40     }
41     case CHARDEL:
42     {
43         index++;
44         aDelays[function[index]].length=dblstack.top_pop();
45         result=FindDelay(function[index]);
46         dblstack.push(result);
47         break;
48     }
49     case CHARNEG:
50     {
51         //negate value on stack, and push new value
52         result = -dblstack.top_pop();
53         dblstack.push(result);
54         break;
55     }
56     case '+':
57     {
58         y = dblstack.top_pop();
59         result = y + dblstack.top_pop();
60         dblstack.push(result);
61         break;
62     }
63     case '-':
64     {
65         y = dblstack.top_pop();
66         result= dblstack.top_pop() - y;
67         dblstack.push(result);
68         break;
69     }
70     case '*':
71     {
72         y = dblstack.top_pop();
73         result= dblstack.top_pop() * y;
74         dblstack.push(result);
75         break;
76     }
77     case '/':
78     {
79         y = dblstack.top_pop();
80         result= dblstack.top_pop() / y;
81         dblstack.push(result);
82         break;
83     }
84     case '^':
85     {
86         y = dblstack.top_pop();
87         result= pow(dblstack.top_pop(),y);
88         dblstack.push(result);
89         break;
90     }
91     case CHARFUNC:
92     {
93         index++;
94         switch(ftable[function[index]].n_pars)
95         {
96             case 0:
97                 {

```

```

98         dblstack.push(((Func0)fable[function[index]].f)());
99         break;
100     }
101     case 1:
102     {
103         x = dblstack.top_pop();
104         dblstack.push(fable[function[index]].f(x));
105         break;
106     }
107     case 2:
108     {
109         y = dblstack.top_pop();
110         x = dblstack.top_pop();
111         dblstack.push(((Func2)fable[function[index]].f)(x,y));
112         break;
113     }
114     case 3:
115     {
116         z = dblstack.top_pop();
117         y = dblstack.top_pop();
118         x = dblstack.top_pop();
119         dblstack.push(((Func3)fable[function[index]].f)(x,y,z));
120         break;
121     }
122     default:
123     {
124         fset_error("I2: too many parameters\n");
125         return 0;
126     }
127 }
128 break;
129 }
130 default:
131 {
132     fset_error("I1: unrecognized operator");
133     return 0;
134 }
135 }
136 }
137 //check if we're out of stack, or not
138 result=dblstack.size();
139 if(result!=stksize+1)
140 {
141     fset_error("I3: corrupted buffer");
142 }
143 return dblstack.top_pop();
144 } /* end of value */

```