UNIVERSITY of LIVERPOOL

Machine Learning Approaches To Complex Time Series

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor of Philosophy

in

Electrical Engineering and Electronics

by

D.I. Stamp , BSc, MSc

September 1999

Machine Learning Approaches
To Complex Time Series

by

D.I. Stamp

Acknowledgements

My supervisor Professor Q.H.Wu for his support and guidance

my wife Lynne for her unending support and patience

my daughter Stephanie for lighting up my life

This research was funded by an EPSRC studentship

Abstract

Machine Learning Approaches
To Complex Time Series

by

D.I. Stamp

It has been noted that there are numerous similarities between the behaviour of chaotic and stochastic systems. The theoretical links between chaotic and stochastic systems are investigated based on the evolution of the density of dynamics and an equivalency relationship based on the invariant measure of an ergodic system. It is shown that for simple chaotic systems an equivalent stochastic model can be analytically derived when the initial position in state space is only known to a limited precision.

Based on this a new methodology for the modelling of complex nonlinear time series displaying chaotic behaviour with stochastic models is proposed. This consists of using a stochastic model to learn the evolution of the density of the dynamics of the chaotic system by estimating initial and transitional density functions directly from a time series.

A number of models utilising this methodology are proposed, based on Markov chains and hidden Markov models. These are implemented and their performance and characteristics compared using computer simulation with several standard techniques.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

It has been noted that determinism does not imply either regular behaviour or predictability. Chaotically behaved deterministic dynamical systems appear to behave in a pseudo-random manner. This apparently random behaviour of chaotic systems stems from their inherent properties determined by sensitivity of system initial states to system non-linearities. Some chaotic systems behave in a similar manner to that of a certain class of stochastic systems whose randomness is caused by stochastic properties of system parameters, states and external disturbances as noted by both Wu & Cao [36] and Yang [26]. Therefore, the theory of statistical properties of dynamical systems and some concepts and techniques used in stochastic systems have been employed to study chaotic dynamics. A relationship between chaotic and stochastic systems has been identified by Wu & Cao [36] and this gives us the possibility of exploring the exact similarities of behaviour of the two systems and to establish a relevant methodology for the study. Identification of this relationship provides justification for the application of many established techniques of stochastic system analysis and control to the prediction and control of unknown complex chaotic systems.

1.1 Problem formulation

The problem to be examined is the modelling of nonlinear dynamic systems which display chaotic characteristics using only a time series generated by the system to learn the model.

Consider a nonlinear dynamical system:

$$x_{t+1} = f(x_t) \tag{1.1.1}$$

where x_t is a state vector of dimension k. Generally it is not possible to directly observe the system state. The only available data is a time series. The observed sequence:

$$z_t = \varphi(x_t) \tag{1.1.2}$$

where z_t is the observed vector at time t, will be known to a finite level of granularity and will be of limited length. The question as to whether it is possible to reconstruct the dynamics of the system using the time series is answered by Takens theorem which shows that generically the dynamics of a system

$$y_t = (z_{t-d}, z_{t-d+1}, \cdots, z_{t-1}) \tag{1.1.3}$$

where d is known as the embedding dimension, and $d \geq 2k + 1$, is completely equivalent to the dynamics of x_t apart from a smooth invertible coordinate transformation. Thus in theory a time series can be used to learn the underlying dynamics of the system that generated it.

1.1.1 History of problem

The latent tendency of non-linear dissipative systems to irregular motions remained unobserved for a long time. Poincare [15] was the first to note this characteristic, but it was not until the advent of the computer with its capacity for rapid numerical calculation and graphical display that the investigation of chaotic phenomenon was practical.

In 1963 Edward Lorenz [7] developed a highly simplified hydrodynamic model for meteorology which derives from the Navier-Stokes equation. These equations lead to an attractive structure in phase space which discloses an irregular, seemingly erratic behaviour of the trajectories. The cause of this erratic behaviour is the sensitivity of the dynamic system to small changes in initial system state over time, the so called butterfly effect.

Since then progress in the study of chaos has been rapid. Ruelle and Takens discovered in phase space a limited domain with unusual properties which they denoted a strange attractor. Once its existence had been noticed chaos was found in a multitude of different systems. Chaos only occurs in nonlinear systems, but almost all real systems display some degree of nonlinearity. Chaos thus appears in such diverse systems as meteorology, astronomy, biomechanics, coupled pendulums, chemical reactions, fluid mechanics, and numerical solutions to differential equations. Irrespective of the medium the dynamics of chaotic systems obey certain common rules.

The study of time series has progressed from the beginning of the century with the adoption of linear modelling techniques. Consequently linear modelling techniques have a large, well established body of theory. More recently it has been recognised that better results can be obtained in many systems by using nonlinear modelling techniques and many such techniques such as neural networks have been successfully developed and applied to the modelling of time series.

In the area of non-linear dynamics there has recently been study of the possible techniques for the control of chaotic systems, exploiting their unique characteristics. Techniques have been developed for the stabilization of unstable periodic orbits and the use of recurrence to allow stabilization to be applied locally by Vincent [44], Schuster et al [31], Flake [11] and Ott et al [34]. It has been demonstrated that physical systems respond well to both simple and sophisticated control strategies. Applications have been proposed

in communications, electronics, fluid mechanics and chemistry.

1.2 Motivation

1.2.1 Potential of the research

The research has the potential for establishing a new methodology for modelling the dynamics of chaotic systems from their times series using stochastic models. This has the benefit of looking at the problem of modelling chaotic systems in a new way, with the potential of new insights into the problem. Rather than concentrating on the evolution of a single trajectory through phase space the evolution of the density of dynamics is considered. That avoids tackling the initial condition problem of chaotic systems directly, as it is embedded in the density distribution function. Therefore the evolution of this density distribution is studied. This gives information about the system in a different form. Instead of point predictions a range of possible solutions are considered along with their probability of occurring. This is a more informative approach giving information on the bounds of future trajectories and an estimate of the accuracy of predictions while still allowing us to derive point predictions if required. It also opens up the possibility of applying a range of stochastic techniques to the modelling of chaotic systems for prediction and control purposes.

1.2.2 Current situation

Although many of the measures utilised in the characterisation of chaotic systems are statistical in nature there is no established methodology for the study of the relationship between chaotic and stochastic systems. Although stochastic models have been applied to time series analysis, this has not been in the field of chaotic dynamics where the systems are purely deterministic. There is thus a lack of stochastic models specifically designed for chaotic systems and

taking advantage of their unique characteristics. There is a corresponding lack of application of stochastic models to real applications where the systems are known to be chaotic in nature.

1.3 Overview of thesis

Chaotic systems are introduced in Chapter 2 and their characteristics identified. Methods for characterising chaos are presented. Stochastic processes are also introduced, concentrating on Markov processes, in particular Markov chains and their properties. Current methods of nonlinear time series modelling, and the various aspects of the modelling process are then detailed.

Chapter 3 compares the similarities between the chaotic systems and the stochastic systems introduced in Chapter 2. A theoretical link between the two via an equivalence relationship between chaotic and stochastic systems based on the study of density of dynamics and Brownian motion of stochastic systems is presented. The equivalency in invariant measure and ergodicity between the two systems is defined. The concept of the equivalency can be used to achieve system reconstruction of the dynamics learnt using the time series of chaotic systems. To demonstrate this an equivalent stochastic system model for a simple system (the Logistic Mapping) is derived.

Chapter 4 suggests that since motions of chaotic dynamical systems, after some transients, settle down to strange attractors and sustain for a long time. This provides a great opportunity in terms of time period for a computer to learn the properties of the chaotic systems through space-time patterns. As long as learning proceeds, prediction and control of future states of the chaotic systems can be achieved in theory. Based on the concept of the equivalence relation and the equivalent system model, system reconstruction of chaotic dynamics using learning techniques is investigated. Various stochastic models are suggested based on Markov models for modelling chaotic time series. Time se-

ries modelling is performed on analytically generated and real world time series and its performance is assessed. Results are compared with those generated using a linear autoregressive model, a feedforward neural network, and a time delay neural network.

To overcome some of the limitations of the Markov models, Chapter 5 investigates Hidden Markov models as equivalent stochastic models for chaotic time series. Modelling of time series is performed and comparisons with previous results made. A simulation study of the yearly sunspot series is performed.

The work is summarised in Chapter 6 with conclusions as to the effectiveness of the technique being drawn and suggestions for further work made.

1.4 Major contributions

A review of current techniques for modelling of chaotic systems has been undertaken. The analytical relationship proposed by Wu & Cao [36] has been studied and sufficient conditions for its validity investigated.

The validity of modelling chaotic dynamic systems using stochastic models has been demonstrated. It has been shown that it is possible to build a simple hidden Markov model of a simple chaotic system (the Logistic mapping) analytically, and that when the state of the system is observed to a specific limited precision the dynamics of the time series generated by this specific system are exactly modelled by the stochastic system.

It has been shown that Markov models and their extension hidden Markov models naturally emerge as models for the density evolution, and that they have several of the characteristics of the modelled chaotic system embedded in them.

It has been demonstrated by the use of computer simulations that hidden Markov models can learn the dynamics of chaotic time series through training in an iterative manner using only a time series as data and the characteristics

of the models along with their accuracy and their limits explored.

Chapter 2

Introduction To Chaotic Dynamics And Stochastic Systems

2.1 Dynamical systems

A dynamical system $\{S_t\}_{t\in\Re}$ on X is a family of transformations $S_t : X \to X, t \in \Re$, satisfying

- $S_0(x) = x$, $\forall x \in X$;
- $S_t(S_{t'}) = S_{t+t'}(x)$, $\forall x \in X$, with $t, t' \in \Re$; and
- The mapping $(t, x) \to S_t(x)$ from $X \times \Re$ into X is continuous.

Thus it consists of an abstract phase or state space whose coordinates describe the dynamical state at any instant and a dynamical rule which defines the future evolution of all the state variables. The phase space of a system is a mathematical space with orthogonal coordinate directions representing each of the variables needed to specify the instantaneous state of the system. A discrete phase space is often referred to as a state space.

A trajectory, path or orbit of a system is the path traced out in phase space by the solution of an initial value problem of the dynamical system. For a continuous time system this is a curve, while for a discrete time system this is a series of points.

A continuous time dynamical system is commonly represented as a series of first order differential equations

$$\begin{aligned} \frac{dx_1}{dt} &= f_1(x_1, \ldots, x_n) \\ &\vdots \\ \frac{dx_n}{dt} &= f_n(x_1, \ldots, x_n) \end{aligned} \tag{2.1.1}$$

or alternatively more compactly as

$$\dot{X} = F(X) \text{ where } X = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \tag{2.1.2}$$

and a discrete time dynamical system by a mapping rule

$$X_{t+1} = F(X_t) \tag{2.1.3}$$

Dynamical systems can be divided into two types, deterministic systems and stochastic systems. The future evolution of a deterministic system is uniquely determined by the systems history. Deterministic systems may be either invertible or non-invertible. The future evolution of a stochastic system has no single future evolution, rather it has a number of possible future trajectories, only one of which will be realised in a stochastic manner. By its nature a stochastic system is non-invertible.

The time series generated by a dynamic system under the following conditions

- its mean is independent of time
- its variance is independent of time
- its autocovariance is independent of time

is termed stationary in the wide sense.

Conservative dynamic systems maintain a constant phase volume under time evolution.

Dissipative nonlinear systems converge in the long term in phase space to attractors. An attractor A of a phase flow ϕ_t is a compact (closed and bounded) set with the following characteristics:

- for all t's the attractor A is invariant under the influence of the phase flow ϕ_t(thus $\phi_t A = A$).
- the attractor A has an open neighbourhood U which contracts under the influence of the phase flow ϕ_t onto A.
- the attractor A cannot be divided into two self contained, non-overlapping invariant sets.

Each attractor is surrounded in state space by a basin of attraction, and any system whose initial state lies within that basin will converge on that attractor.

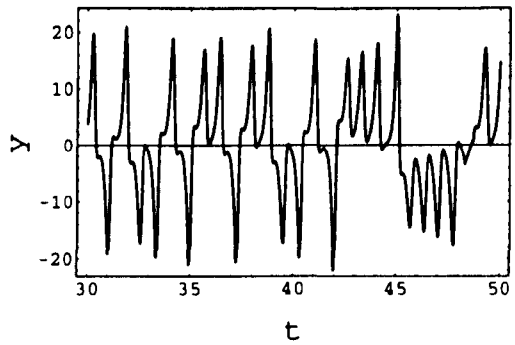

Figure 2.1: The Lorenz attractor in the time domain

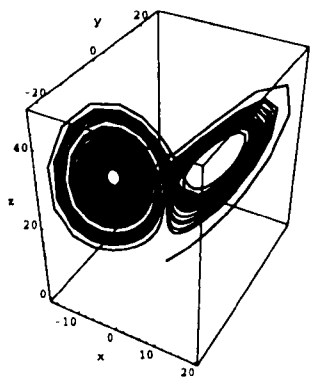

Figure 2.2: The Lorenz attractor

Attractors can consist of point attractors (sometimes at infinity), periodic limit cycles, or (in the case of chaotic systems) strange attractors which can be considered as an infinite length limit cycle.

It is known that many dissipative systems, for example the flow of fluids described by partial differential equations with an infinite number of degrees of freedom, display a long term behaviour which is determined by low dimensional attractors, thus allowing a dramatic reduction of the effective degrees of freedom, as described by Argyris et al[9].

2.2 Chaotic dynamics

Chaos is the irregular behaviour of the solutions of deterministic equations. To produce chaos the equations must be nonlinear, but can otherwise be quite simple. Chaotic time series are generated by nonlinear deterministic systems. They possess a number of characteristics. They display an apparent randomness in the time domain, for example the Lorenz attractor in Figure 2.1, but their deterministic structure is evident when viewed in phase space, given in Figure 2.2.

Chaos can only arise in systems of sufficient dimension. It requires a mini-

mum dimension of one for it to occur in a non-invertible discrete time system, two for an invertible discrete time system, and three for a continuous time system.

2.2.1 Strange attractors

When considered in the underlying state space solutions to a chaotic system relax onto a strange attractor. A strange attractor is a limited domain in phase space which attracts the trajectories globally and causes them to diverge locally from one another exponentially.

This attractor has a fractal structure, and typically a non-integral dimension. It is of a lower dimension than the system generating it. The attractor contains an infinite number of unstable periodic orbits.

Stretching and folding

Chaotic systems display an exponential divergence of adjacent trajectories and yet remain contained within a finite-sized phase space. The primary mechanism for this behaviour is that of stretching and folding. As the system evolves the phase space is stretched resulting in the exponential divergence of trajectories, and then folded back upon itself thus maintaining the finite size of the phase space.

This stretching and folding process causes mixing of the phase space such that initially adjacent trajectories will over a period of time be dispersed over the attractor. Trajectories adjacent at a point in time will have been widely dispersed in the past. Thus over a period of time the system loses its memory of its initial state.

Ergodicity

If μ is a Borel probability measure (Lasota and Mackey[1]) on M and μ is invariant under f, i.e. $\mu(f^{-1}(E)) = \mu(E)$ for every Borel set E, then a dynamical system $f : (M, \mu) \to (M, \mu)$ is said to be ergodic if $f^{-1}(E) = E \Rightarrow \mu(E) = 0$ or 1 (Yang[26]). That is, an ergodic system cannot be decomposed into two or more non-trivial subsystems which do not interact with each other. For a system to be ergodic it must be stationary (Lasota and Mackey[1]).

In an ergodic system a link between the temporal behaviour and the spatial behaviour is provided by the existence of an invariant natural measure μ.

A property is said to hold almost everywhere (a.e.) if it is enjoyed by every $x \in M$ except possibly on a set E with $\mu(E) = 0$.

If $(X, \mathcal{A}, \mu)$ is a measure space, $S : X \to X$ a non singular transformation, P the Frobenius-Perron operator associated with S, and S is ergodic, then there is at most one stationary density f^* of P. Also if there is a unique stationary density f^* of P and $f^*(x) > 0$ a.e., then S is ergodic. Since any dynamic system will have an associated Frobenius-Perron operator, and since a chaotic system converges to a strange attractor with a stationary density, chaotic systems are ergodic.

Birkhoff ergodic theorem

For any integrable function $\phi : (M, \mu) \to \Re$ the time average converges for a.e., x. If this is denoted $\phi^*(x)$

$$\phi^*(x) = \frac{1}{n} \sum_{i=0}^{n-1} \phi(f^i(x)) \tag{2.2.1}$$

then if the system is egodic then a.e.

$$\phi^*(x) = \int \phi(x) d\mu(x) \tag{2.2.2}$$

it is equal to the space average.

This means that if the dynamic system is ergodic the temporal mean value can be replaced by a mean value of the spatial distribution. Thus the total information of the dynamical system is contained in (almost) every arbitrarily selected trajectory.

Let $(X, \mathcal{A}, \mu)$ be a finite measure space and $S : X \rightarrow X$ be measure preserving and ergodic. Then for any set $A \in \mathcal{A}, \mu(A) > 0$, and almost all $x \in X$, the fraction of points $S^k(x)$ in A as $k \rightarrow \infty$ is given by $\frac{\mu(A)}{\mu(X)}$. Thus every set of nonzero measure is visited infinitely often by the iterates of almost every $x \in X$.

2.2.2 Characterising chaos

Power spectra

Chaotic systems exhibit a continuous power spectra, and thus appear as stochastic time series when analysed using linear techniques. The power spectra are a continuous wide band of frequency along with possible single characteristic peaks.

Lyapunov exponent

Chaotic systems are very sensitive to initial conditions and are only accurate for a length of time governed by the errors in initial conditions and by the Lyapunov exponents of the system (parameters that quantify the exponential divergence of state trajectories in chaotic systems). The Lyapunov exponent of a time series generated by a one dimensional system given by equation (1.1.1) is defined as

$$\sigma = \lim_{t \rightarrow \infty} \frac{1}{t} \sum_{i=0}^{t-1} log\,|f'(x_i)| > 0 \qquad (2.2.3)$$

where $f'(x_t)$ is the gradient of the function $f(x_t)$. A multi-dimensional system will have a Lyapunov exponent for each dimension. The Lyapunov exponents

measure the exponential rate of divergence of initially close trajectories over time. At least one positive Lyapunov exponent indicates a sensitivity to initial conditions which makes the time evolution of the system hard to predict for all but the short term. If a system has at least one positive Lyapunov exponent σ, then it is termed chaotic.

Exceptional trajectories, such as for example, those that end in an unstable fixed point for $t \to \infty$, may possess other Lyapunov exponents, but are not generally typical for the attractor and are of zero measure.

It is possible to make an estimate of the relaxation time t^* for a known error in initial conditions $\delta x(0)$

$$t^* \sim \frac{1}{\lambda_1} \ln \frac{L}{|\delta x(0)|} \tag{2.2.4}$$

where λ_1 is the largest Lyapunov exponent, and L is a characteristic length of the attractor. The relaxation time is the time after which there is no longer any correlation between the initial condition $x(0)$ and the current state $x(t)$.

Generalised dimension

Consider a strange attractor in an n-dimensional phase space and divide the phase space evenly into n-dimensional hypercubes of edge length ε. If $W(\varepsilon)$ denotes the number of hypercubes containing attractor points, N the total number of measuring points on the strange attractor, and N_I the number of measuring points in the i^{th} cell, a mean information of the q^{th} order can be defined, the Renyi information:

$$\bar{I}_q(\varepsilon) = \frac{1}{1-q} \ln \sum_{i=1}^{W(\varepsilon)} (p_i)^q \tag{2.2.5}$$

and using this define the generalised dimension of the q^{th} order

$$D_q = \lim_{\varepsilon \to 0} \frac{\bar{I}_q(\varepsilon)}{\ln 1/\varepsilon} \tag{2.2.6}$$

Capacity dimension

Setting $q = 0$ the Capacity dimension can be obtained

$$D_0 = D_c = \lim_{\varepsilon \to \infty} \frac{\ln W(\varepsilon)}{\ln(\frac{1}{\varepsilon})} \tag{2.2.7}$$

Information dimension

Setting $q = 1$ the Information dimension is obtained

$$D_1 = D_I = \lim_{\varepsilon \to \infty} \frac{\ln \bar{I}(\varepsilon)}{\ln(\frac{1}{\varepsilon})} \tag{2.2.8}$$

where the information gain per single measurement is

$$\bar{I}(\varepsilon) = \frac{I(\varepsilon)}{N} = -K \sum_{K=1}^{W(\varepsilon)} p_k \ln p_k \tag{2.2.9}$$

and setting $q = 2$ the correlation dimension D_K can obtained.

It can be proved that $D_K \leq D_I \leq D_C$ (Argyris[9]).

Kolmogorov-Sinai entropy

Kolmogorov-Sinai (KS) entropy provides the upper limit of the mean information production per unit time if both measurement precision and recording frequency of the data are varied. It quantifies the degree of our ignorance of the system, and allows us to state the fundamental limits to the predictability of a dynamical system. In the case of regular, predictable behaviour once a measurement has been made the state of the system is determined and additional measurements provide no new information. If the dynamics of the system are chaotic then each measurement provides further information as to the state of the system. The KS entropy is given by

$$h(\mu) = \sup_{Z, \Delta t} \left(\lim_{n \to \infty} \frac{\bar{I}(X^{(n)})}{n \Delta t} \right) \tag{2.2.10}$$

this is linked to the Lyapunov exponents via

$$h(\mu) \leq \sum_{i}^{+} \sigma_i \tag{2.2.11}$$

and gives an estimate of the relaxation time

$$t^* = \frac{D_I \log_2(1/\varepsilon)}{h(\mu)} \tag{2.2.12}$$

Autocorrelation

The mean autocorrelation of a time series $x(t)$ is defined as

$$R_X(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-T}^{+T} x(t+\tau)x(t)dt \tag{2.2.13}$$

and the autocovariance as

$$K_X(\tau) = R_X(\tau) - m_X^2 \tag{2.2.14}$$

where m_X is the mean of the series. These define the interrelation of one and the same signal at two distinct instants of time, t and $t+\tau$, and are functions of the time shift τ. For regular motion they are either periodic or quasiperiodic.

For chaotic motion $K_X(\tau) \to 0$ for $\tau \to \infty$. This decaying autocovariance, with the correlations between time lagged signals gradually vanishing with increasing lag, demonstrates the loss of memory of the initial system state over time.

For white noise $K_X(\tau)$ is a single impulse at $\tau = 0$.

For example for the Logistic mapping

$$\begin{aligned} K_X(\tau) &= \int_0^1 p(x)xf^\tau(x)dx \\ &= \int_0^1 \frac{xf^\tau(x)}{\pi\sqrt{(x(1-x)}}dx - \frac{1}{4} \\ &= \int_0^1 \frac{x\sin^2(2^\tau \arcsin(\sqrt{x}))}{\pi\sqrt{x(1-x)}}dx - \frac{1}{4} \end{aligned} \tag{2.2.15}$$

2.3 Stochastic processes

A stochastic process ξ_t is a family of random variables that depend on a parameter t, usually called time. If t assumes only integer values then the

stochastic process reduces to a sequence of random variables called a discrete time stochastic process. If t belongs to an interval of $\Re$, the stochastic process is called a continuous time stochastic process.

2.3.1 Markov processes

Definition

A Markov chain is a collection of random variables $\Phi : n \in T$ where T is a countable time set $T \in \Re_+$. The critical aspect of a Markov process is that it is forgetful of all but its immediate past. This is the Markov property

$$P(x_k|x_0, \ldots x_{k-1}) = P(x_k|x_{k-1}) \tag{2.3.1}$$

which implies

$$P(x_0, \ldots x_k) = p(x_0)p(x_1|x_0) \ldots p(x_k|x_{k-1}) \tag{2.3.2}$$

and thus to describe the process only the one step transition probabilities $P(x_k|x_{k-1})$ and the initial probabilities $P(x_0)$ are required.

If the transition probabilities are independent of time the Markov chain is said to be homogeneous.

$$P(x_k = j|x_{k-1} = i) = P(x_r = j|x_{r-1} = i) \tag{2.3.3}$$

i.e. $P_{ij} = P(x_k = j|x_{k-1} = i)$ does not depend on k

The transition probabilities p_{ij} satisfy

$$\sum_i p_{ij} = 1 \tag{2.3.4}$$

Defining the transition matrix P as

$$P = \begin{bmatrix} p_{11} & \cdots & p_{1j} \\ \vdots & \vdots & \vdots \\ p_{i1} & \cdots & p_{ij} \end{bmatrix} \tag{2.3.5}$$

P^n is the n-step transition matrix.

Markov chains which are fully parameterised (have $m^2 - m$) independently determined transition probabilities (where m is the number of states) are termed saturated.

Higher order chains

A model that is non-Markovian but is based only on a finite memory so that the system depends on the past only through the previous $k+1$ values in the probabilistic sense that

$$P(Y_{n+m} \in A | Y_j, j \leq n) = P(Y_{n+m} \in A | Y_j, j = n, n-1, \ldots, n-k) \quad (2.3.6)$$

can be reformulated into a Markov model through defining the vectors

$$\Phi_n = (Y_n, \ldots, Y_{n-k}) \quad (2.3.7)$$

and the motion of the first coordinate of Φ reflects that of Y.

Properties of transition matrix

Transition matrices for further than a single step into the future can be formed utilising the Chapman-Kolmgorov equation

$$p^{(n+m)} = p^{(n)} p^{(m)} \quad (2.3.8)$$

Communication between states

If $p_{jk}^{(n)} > 0$ then state k can be reached from state j. If $p_{jk}^{(n)} > 0$, $p_{kj}^{(m)} > 0$ states k and j communicate.

If C is a set of states such that no state outside C can be reached from any state in C, then C is closed. If each pair of states within C communicate, then C is a closed communicating class. If a closed set contains only one state it is

know as an absorbing state. Thus absorbing states are states one can make a transition into, but not out of.

Chains for which transitions from a state to any other state are eventually possible (i.e. all states communicate with one another) are called irreducible. Other chains (including those with absorbing states) are non-irreducible.

If a discrete Markov chain contains no closed sets with the exception of the set of all states then it is irreducible.

Long term behaviour

If some state has the property of periodicity, the state is said to be periodic.

The steady state probability $v_j = \lim_{j\to\infty} p_j^{(n)}$ satisfies $v = vP$. The following statements about the steady state probability can be made.

- In any aperiodic Markov chain the limits $v_j = \lim_{j\to\infty} p_j^{(n)}$ exist.
- In any irreducible aperiodic Markov chain the limits v_j do not depend on the initial distribution.
- In any finite, irreducible, aperiodic Markov chain the limit vector v is the unique stationary probability vector of the process.

A full treatment of the long term behaviour of Markov chains is given in Tweedie & Meyn [42].

2.4 Time series

2.4.1 Methodologies for time series analysis

One of the central problems of science is forecasting: given the past, how can the future be predicted? The classical approach to this problem is to build an explanatory model from first principles, measure initial data, and use the

model to make predictions. Unfortunately this is often not possible. In some fields the first principles upon which to construct a model are lacking, in others although the models are good there is difficulty in obtaining the required initial data.

An alternative approach is to build a predictive model based directly on the data. This is usually done by selecting a model based on the observed characteristics of the data and any a priori knowledge of the problem. This selecting of the correct model in the case of nonlinear systems has been described more as an art than a science.

The theoretical basis for this data driven approach to modelling was derived by Takens [8]. Take the system

$$x_t = F(x_{t-1}) \tag{2.4.1}$$

where $F(\bullet)$ is an unknown function (possibly nonlinear), and x_t is a state vector of dimension k. In practice usually direct access to the state vector is not available and it is merely possible to observe some function $\varphi(x)$ of the state vector, where φ is called the measurement function. At first sight the observed sequence

$$z_t = \varphi(x_t) \tag{2.4.2}$$

contains little information about the behaviour of the state vector. However Takens [8] proved that subject to genericity assumptions the dynamics of a system

$$y_t = (z_{t-d}, z_{t-d+1}, \cdots, z_{t-1}) \tag{2.4.3}$$

where d is known as the embedding dimension, and $d \geq 2k + 1$, is completely equivalent to the dynamics of x_t apart from a smooth invertible coordinate transformation. It is possible therefore to take the system defined by equation (2.4.3), model it, and that model will reproduce the dynamics of the original system, equation (2.4.1).

When it comes to the behaviour of a nonlinear system, there are three main sources of complexity to be considered using a dynamical systems point of view. These are:

- observation or measurement errors.
- the dimensionality of the system. This is due to the possible action of many variables (which may give rise to process noise).
- the multiplicative or nonlinear interaction of these variables (possibly giving rise to chaos).

A real time series will consist of a mixture of these effects.

The time series, generated by a noise-free process, defined by equation (2.4.1), is actually observed in the presence of observational noise as:

$$Y_t = \varphi(X_t) + \varepsilon_{obs} \tag{2.4.4}$$

where ε_{obs} is usually assumed to be white Gaussian noise. The effect of observational noise on a model is usually reduced by using as many observations as possible from the time series in the construction of the model.

Process noise is that part of the system dynamics that remain unexplained (and unmodelled). Given a real m-dimensional system, and using a k-dimensional system to model it, the remaining $m-k$ dimensions are approximated by noise:

$$Y_t = F(X_t + \varepsilon_{process}) \tag{2.4.5}$$

where $\varepsilon_{process}$ is the process noise, and $F(X_t)$ is the modelled noise free skeleton. Process noise can cause a stable deterministic skeleton to become unstable leading to stochastic chaos.

2.5 Conventional techniques for non-linear system identification and prediction

Current models can be divided into two broad categories, global models and local models, and into two groups, parametric models and non-parametric models.

Global models

These are models where a single model is used to represent the entire state space of the nonlinear system. An example would be a standard multilayer perceptron model, or a NARMA (nonlinear autoregressive/moving average) model.

Local models

These are models where the state space of the nonlinear system is split into several subregions each of which is modelled by a submodel. Examples of this include local linear models and Threshold ARMA (autoregressive/moving average) models.

Often there is not a hard boundary between the individual sub models with a weighted result based on the position in state space being used.

Parametric models

These are models where the structure of the underlying model is assumed a priori and then the parameters of the model are fitted to the time series. An example is nonlinear ARMA models. The difficulty is that it relies on a good a priori model choice. If the underlying dynamics of the time series do not match that of the assumed model then the results are poor. Stochastic optimisation methods such as genetic algorithms have been used to try and learn the model

structures with a certain amount of success with simple applications, but the potential search space is very large.

Non-parametric models

These are models where few assumptions are made about the underlying dynamical structure of the time series. They include models such as feedforward multilayer perceptron networks and recursive networks, as well as local linear models.

They are usually function approximation techniques approximating the underlying transition function of the system.

The models can be further divided into those which perform prediction based purely on the time series and those which perform prediction based on the current internal state of the model (which is based on past values on the time series) combined with the current value of the time series. An example of the former are standard feedforward networks and of the latter are recurrent networks.

2.5.1 Extension of linear methods

Linear Gaussian models have dominated time series modelling for the past 50 years. They are supported by a large body of well developed theory and computation time required for the models reasonable.

Linear approximation

The simplest method for nonlinear system identification is to make the approximation that the system is linear and use one of the numerous linear models such as an ARMA model:

$$X_t = a_0 + \sum_{j=1}^{k} a_j X_{t-j} + \sum_{j=0}^{l} b_j \epsilon_{t-j} \tag{2.5.1}$$

where a_j and b_j are real constants, and the ϵ_t are zero-mean uncorrelated random variables. If the system being modelled is close to being linear (almost all actual systems display some degree of linearity) then the linear model works well. However, if the system displays a high degree of nonlinearity then the performance of a linear model can be extremely poor and truly nonlinear models are required.

NARMAX

This is a natural extension of the linear ARMAX model, but includes nonlinear terms:

$$X_t = \sum_{j=1}^{k} a_j p_j(t) + \sum_{j=0}^{l} b_j \epsilon_{t-j} \tag{2.5.2}$$

where $p_j(t)$ is a nonlinear term such as X_{t-1}^2. The model has available a number of possible nonlinear terms which are selected along with the coefficients using global selection and optimisation techniques such as genetic algorithms.

2.5.2 Local models

Local models are constructed by dividing the phase space of the process into regimes, and having a separate submodel associated with each regime. The distinction between regimes can be hard (each point is in only one regime), or soft (overlapping regimes are allowed). Various different submodels have been used, including neural networks.

Threshold models

Tong [16] covers a number of threshold models such as the SETAR (self-exciting threshold autoregressive model) which takes the form:

$$X_t = a_0^{(k)} + \sum_{j=1}^{k} a_j^{(k)} X_{t-j} + h_1^{(k)} \epsilon_t X_{t-d} \in R_j \tag{2.5.3}$$

where R_j is the regime.

Local linear maps

A similar approach is taken by Casgali [28] who proposes a prediction algorithm that constructs piecewise linear approximations to the unknown function by using a variable number k neighbours. A small value of k corresponds to a deterministic approach to behaviour, while a large value of k corresponds to a stochastic linear autoregressive model. An intermediate value corresponds to fitting nonlinear stochastic models.

The algorithm consists of taking a test vector from which the next values will be predicted, finding the k nearest neighbours belonging to a fitting set and fitting a linear model to these vectors. This model is then used to make the prediction.

Sequential locally weighted linear maps

Sugihara [12] proposes a variant on this method which weights the effect of the vectors in the fitting set according to a weighting factor:

$$\omega(d) = e^{-\theta d/D} \tag{2.5.4}$$

where d is the distance between the test vector and the respective fitting vector, θ is a constant that determines the degree of local weighting and D is the average distance between vectors. If θ=0 the result is a global linear solution. As θ is increased the solutions become more local and hence nonlinear. A weighting factor of the form $\omega(d) = 1$ for $d < k$, and $\omega(d) = 0$ elsewhere gives you the local linear maps described by Casgali [28]. However, when there is a mixture of linear and nonlinear signals in the data then this weighting function can penalise distant points too severely.

Local linear approximation

Farmer and Sidorowich [18] use the k nearest neighbours approach of Casgali [28] to construct a local estimator by fitting a linear polynomial using least

squares by single value decomposition.

Fuzzy logic models

Kim [4] details a genetic fuzzy predictor ensemble for the prediction of chaotic time series. It consists of an ensemble of fuzzy predictors combining their predictions by an equal error prediction weighting method. Each individual fuzzy predictor consists of two stages. The first stage generates a fuzzy rule base and the second fine tunes the membership functions. Both stages are trained by separate genetic algorithms. Results on the Mackey-Glass time series give errors slightly worse than a backpropagation neural network.

2.5.3 Neural networks

Neural networks with their capability to generalise, fault tolerance, lack of requirement for a priori models and nonlinear operation are an obvious method of modelling the behaviour of complex time series and several different architectures have been utilised.

Feedforward networks

Multilayer feedforward perceptron networks trained using the backpropagation algorithm are the type of neural network in most frequent use in a wide range of applications. They provide a practical method of modelling the behaviour of a nonlinear input-output mapping of a general nature in a global manner and function as universal approximators.

The networks consist of a series of layers of neurons which perform a nonlinear mapping on the weighted outputs from previous layers. They are trained by being presented with known pairs of input and output training vectors, with the errors in prediction being propagated back through the network and being used to adjust the weights down towards a minimum in the error surface. This

process is repeated many times until a required error level is reached or training is otherwise halted.

Backpropagation networks have several disadvantages including long training times and the potential to get trapped in a local minima in the error surface rather than finding the required global minima. There are many variations on the backpropagation algorithm.

Recurrent networks

Deco and Schurmann [10] have used networks with feedback as well as feedforward connections using a modified version of the backpropagation through time algorithm for training. They analyse three different chaotic series - the Henon map, Belousov-Zhabotinskii reaction and the logistic mapping. They investigated performance not only in terms of the prediction error, but also in terms of the dynamical invariants such as the largest Lyapunov exponent and the fractal dimension as well as the power spectrum. They concluded that recurrent networks performed better than feedforward networks when it came to reproducing these qualities.

McDonnell and Waagen [19] used evolutionary programming to train a network utilising recurrent perceptrons that were nonlinear IIR filters. They analysed Wolf's sunspot numbers for the years 1700-1988 and the logistic mapping, producing results slightly inferior to backpropagation networks.

Radial basis function networks

RBF (Radial Basis Function networks) are two layered feedforward networks with a single hidden layer. The neurons in the hidden layer have a radial basis activation function and a linear output layer. The radial basis function is of the form $\varphi(|v_i c_j|)$ where the φ-function is actually a thin-plate spline, v is the input vector and c_j is the RBF centre.

RBF networks are good function interpolators and so are a natural choice to model a function of the form $x_{n+1} = F(x_n, ..., x_{n-d})$. Haykin [39] describes how Broomhead and Lowe have used RBF networks to predict the logistic mapping, as has Kadirkamanathan [30]. RBF networks have been very successful, achieving low prediction errors for example Chaudhury et al [27] and Stark [20].

CMAC networks

Berger [3] describes using both Cerebellar Motor Articulator Controller (CMAC) models and CErebellar model with INTerpolation (CEINT) models (a variant of CMAC which uses linear interpolation) to predict nonlinear systems. CMAC networks quantize inputs into elements each of which address m association cells which contain weights. The output is the sum of the active weights. Berger predicted both linear and nonlinear time series to a fair degree of accuracy.

Runge-Kutta networks

Wang [47] proposes learning the ODEs describing the dynamical system. This is implemented with a combination of the Runge-Kutta method for solving initial value ODE problems and neural networks to learn the ODEs. This gives superior results to neural networks learning the one step ahead transition function, particularly for iterated multistep ahead prediction.

2.6 Stochastic aspects of chaotic dynamics

There are many similarities between stochastic and chaotic systems, notably in terms of their asymptotic behaviour. A basin of attraction is analogous to an closed class in a Markov model in that it is not possible for the system to leave either. A strange attractor is analogous to an aperiodic irreducible Markov

process in that they both possess a limiting distribution which is independent of the initial conditions.

Markov models are memoryless while chaotic systems display a loss of memory of their initial state over time and whenever their initial state is known with limited precision (as is almost always the case in practice) the memory is of a limited duration however it has already been shown, by equation (2.3.7), that any stochastic process with a limited memory can be transformed into a Markov process.

Just as chaotic systems converge in the long term to attractors Markov processes converge to an irreducible class. Neither attractors nor irreducible classes can be divided into two self contained, non-overlapping invariant sets.

The link between chaotic and stochastic systems is provided by the study of the density of dynamics, which is examined in the next chapter.

Chapter 3

Relationship Between Chaotic And Stochastic Systems

3.1 Introduction

One of the reasons that chaos has only been recognised as a phenomenon relatively recently is that it was mistaken for noise. The behavior of chaotic dynamic systems has many similarities to stochastic systems (a characteristic that has been exploited in the creation of pseudo random number generators see Fishman [14]). This apparent similarity between chaotic systems and stochastic systems would suggest that techniques and concepts used in the analysis of stochastic systems could be usefully employed in the study of chaotic dynamical systems. In fact it should be noted that many of the measures used to characterise chaos are in fact statistical in nature.

The previous chapter showed that chaotic systems and certain stochastic systems displayed similar properties. The link between chaotic and stochastic systems is formed when, as an alternative to the more common approach of studying the evolution of a single trajectory through time, the evolution of the density of dynamics over time is considered. Instead of assuming an exact

initial condition for a system, and then modelling the evolution of the trajectory through time to find a single possible outcome for that initial condition, the acknowledgement that the initial conditions are known only to a finite degree of accuracy can be made. The behaviour of the whole set of initial conditions that are possible given our limited knowledge of the state of system can then be studied.

When modelling uncertainty, probability is a natural tool. A probability density can be assigned to each possible point in state space, based on our knowledge of the initial conditions, and then the evolution in time of this density of the dynamics in phase space can be studied. Thus moving from the view point of a single trajectory to the density of an ensemble of trajectories there is a natural move from a deterministic to a stochastic model.

3.2 Chaotic dynamics in state space

3.2.1 Symbolic dynamics

Information about the periodic orbits of a dynamical system can often be captured in terms of sequences of symbols. These symbols code the movement of the orbit around the attractor.

The invariant interval can be divided into two subintervals denoted L and R. It is then possible to assign to a typical orbit of the attractor a sequence of symbols called the itinerary of the orbit. The information about possible

itineraries can be expressed in the form of a transition matrix.

$$\mathbf{M} = \begin{array}{c} \\ ABA \\ ABB \\ BAB \\ BBA \\ BBB \end{array} \begin{array}{c} \begin{array}{ccccc} ABA & ABB & BAB & BBA & BBB \end{array} \\ \left[\begin{array}{ccccc} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \end{array}\right] \end{array} \tag{3.2.1}$$

The above is the transition matrix for the three symbol itinerary of an attractor from the Belousov-Zhabotinskii reaction as given by [6]. The ones represent allowed transitions, while the zeroes represent forbidden transitions. The determination of M does depend on the choice of the symbols used. As can been seen not all sequences of symbols are possible. This exclusion is known as pruning.

The topological entropy h_t of the attractor can be given by

$$h_t = \lim_{p\to\infty} \frac{1}{p} \log_2 N_p \tag{3.2.2}$$

where N_p is the number of periodic orbits of period p in the attractor. It can be shown that this can be determined using the transition matrix M giving us

$$h_t = \lim_{p\to\infty} \frac{1}{p} \log_2 tr M^p \tag{3.2.3}$$

assuming the dynamics are described by M. In this case $h_t \to \log_2 \lambda_{max}$ where λ_{max} is the largest eigenvalue of M.

Symbolic dynamics enables us to identify allowed symbol sequences, or valid trajectories. The dynamics of any model of the underlying system should be consistent with the symbolic dynamics of the underlying system.

It is also clear that if the model shares the symbolic dynamics of the system then some of the invariant properties of the strange attractor such as the topological entropy can be calculated directly from the model.

3.3 Chaotic dynamics in time evolution

If a volume element in phase space is observed, then for $t \to \infty$ it will contract to a subset of an attractor, the dimensionality of which is lower than that of the phase space.

If the initial state of a trajectory lies within a basin of attraction then the trajectory will experience a transient period while it relaxes to its strange attractor followed by a steady state of orbiting upon the attractor.

3.4 Density of dynamics

In general the systems initial state is known imprecisely. If the data is quantized then it is known that the initial condition of the system lies within an interval, rather than the exact value. Using our knowledge of the system a distribution of dynamics within that interval can be implied.

The evolution of the density of a dynamic system is described by the Frobenius-Perron operator.

$$\int_A Pf(x)\mu(dx) = \int_{S^{-1}(A)} f(x)\mu(dx) \tag{3.4.1}$$

Let $(X, \mathcal{A}, \mu)$ be a measure space, $f(x)$ be the state density, and $Pf(x)$ the state density after transformation S. If $S : X \to X$ is a nonsingular transform the unique operator $P : L^1 \to L^1$ defined by equation (3.4.1) is called the Frobenius-Perron operator corresponding to S. The Frobenius-Perron operator is a Markov operator.

In the special case that the transformation S is differentiable and invertible an explicit form for Pf is available.

$$Pf(x) = \frac{d}{dx}\int_{S^{-1}([a,x])} f(s)ds \tag{3.4.2}$$

The Frobenius-Perron operator has the following properties:

- $P(\lambda_1 f_1 + \lambda_2 f_2) = \lambda_1 P f_1 + \lambda_2 P f_2$
- $Pf \geq 0$ if $f \geq 0$
- $\int_X Pf(x)\mu(dx) = \int_X f(x)\mu(dx)$

In the two dimensional case

$$\int_a^x ds \int_b^y Pf(s,t)dt = \int_{S^{-1}([a,x]*[c,y])} \int f(s,t)ds\, dt \tag{3.4.3}$$

and if an explicit form for Pf is available

$$Pf(x) = \frac{\partial^2}{\partial y \partial x} \int_{S^{-1}([a,x]*[c,y])} f(s,t)ds\, dt \tag{3.4.4}$$

As $n \to \infty$ the density $P^n f$ either converges to a unique density (asymptotic stability), approaches a set spanned by a finite number of densities (asymptotic periodicity) or it is sweeping. P^n is said to be sweeping with respect to $\mathcal{A}_*$ if

$$\lim_{n\to\infty} \int_A P^n f(x)\mu(dx) = 0 \text{ for every } f \in D \text{ and } A \in \mathcal{A}_* \tag{3.4.5}$$

A chaotic system relaxes onto a strange attractor over time. It is known from ergodic theory that the space average of the system is the same as the time average. Since the time average is stationary in the form of the strange attractor then the space average (as shown in equation (2.2.2)) and thus the long term density distribution should also tend to a stationary distribution. This limiting density must satisfy

$$Pf_*(x) = f_*(x) \tag{3.4.6}$$

Since the system is ergodic this limiting density is unique (see Lasota & Mackey [1]).

Mixing

Let $(X, \mathcal{A}, \mu)$ be a normalised measure space, and $S : X \to X$ a measure preserving transformation. S is called mixing if

$$\lim_{n\to\infty} \mu(A \bigcap S^{-n}(B)) = \mu(A)\mu(B) \tag{3.4.7}$$

This can be interpreted as meaning that the fraction of points starting in A that ended up in B after n iterations (n must be a large number) is just given by the product of the measures of A and B in X. This indicates that as time progresses the system loses its memory of its initial conditions. Any mixing transformation must be ergodic.

Exactness

Let $(X, \mathcal{A}, \mu)$ be a normalised measure space, and $S : X \to X$ a measure preserving transformation such that $S(A) \in \mathcal{A}$ for each $A \in \mathcal{A}$, if

$$\lim_{n\to\infty} \mu(S^n(A)) = 1 \text{ for every } A \in \mathcal{A}, \mu(A) > 0 \tag{3.4.8}$$

then S is exact. Exactness of S implies that S is mixing. Invertible transforms cannot be exact (Lasota and Mackey [1]).

3.5 Fokker-Plank equation

The Fokker-Plank equation describes the evolution of the density of dynamics of a stochastically perturbed continuous system.

3.5.1 Stochastic process

A stochastic process can be described by the Itô equation, which is driven by the white noise process:

$$\frac{d}{dt}X_t = b(X_t, t) + \sigma(X_t, t)\zeta_t \tag{3.5.1}$$

where ζ_t is a Gaussian white noise. It is known that $\int_0^t \zeta_s ds$ has all the attributes of a Brownian motion W_t. Hence, equation (3.5.1) appears to be equivalent to

$$X_t = X_a + \int_a^t b(X_s, s)ds + \int_a^t \sigma(X_s, s)dW_s \tag{3.5.2}$$

with stochastic integrals having been well defined. Under some conditions given by Itô, the stochastic integral equation (3.5.2) has a unique sample-continuous solution which is a Markov process. The precise interpretation of equation (3.5.1) can be described as follows: Take a sequence of Gaussian processes $\{\zeta_n(t)\}$ which converges to a white Gaussian noise, and yet for each n, $\zeta_n(t)$ has well-behaved sample functions. Now, for each n, the equation

$$\frac{d}{dt}X_n(t) = b(X_n(t),t) + \sigma(X_n(t),t)\zeta_n(t) \ , a \leq t \leq T \tag{3.5.3}$$

together with the initial condition $X_n(a) = X_a$ can be solved, if the functions b (the function describing the deterministic skeleton) and σ (the function describing the magnitude of the stochastic component) are such that the solution exists and is unique for almost all sample functions. For a sequence of processes $\{X_n(t), a \leq t \leq T\}$, suppose that as $n \to \infty$, $\{\zeta_n(t)\}$ converges to a white Gaussian noise, and the sequence processes $\{X_n(t), a \leq t \leq T\}$ converges almost surely, or in quadratic mean, or even merely in probability, to a process $\{X(t), a \leq t \leq T\}$. Then it is natural to say that $X(t)$ is the solution of equation (3.5.2). In this sense, the white-noise-driven differential equation is equivalent to a stochastic differential equation given by

$$dX_t = b(X_t,t)dt + \frac{1}{2}\sigma(X_t,t)\sigma'(X_t,t)dt + \sigma(X_t,t)dW_t$$

The extra term $\frac{1}{2}\sigma\sigma'$ is referred to as the correction term.

Let $\{X_t, a \leq t \leq b\}$ be a Markov process, and denote

$$P(x,t\,|x_0,t_0) = Pr(X_t < x | X_{t_0} = x_0)$$

where $P(x,t\,|x_0,t_0)$ is the transition function of the process. If there is a function $p(x,t\,|x_0,t_0)$ such that

$$P(x,t\,|x_0,t_0) = \int_{-\infty}^{x} p(u,t\,|x_0,t_0)du$$

then $p(x,t\,|x_0,t_0)$ is said to be the transition density function. Define, for a positive ϵ,

$$\begin{aligned} M_k(x,t;\epsilon,\Delta) &= \int_{\|y-x\|\leq\epsilon}(y-x)^k dP(y,t+\Delta|x,t) \\ & \qquad\qquad k=0,1,2 \\ M_3(x,t;\epsilon,\Delta) &= \int_{\|y-x\|\leq\epsilon}\|y-x\|^3 dP(y,t+\Delta|x,t) \end{aligned}$$

It is assumed that the Markov process $\{X_t, a\leq t\leq b\}$ satisfies the following conditions:

$\frac{1}{\Delta}[1-M_0(x,t;\epsilon,\Delta)] \xrightarrow{\Delta\downarrow 0} 0$

$\frac{1}{\Delta}[M_1(x,t;\epsilon,\Delta)] \xrightarrow{\Delta\downarrow 0} m(x,t)$

$\frac{1}{\Delta}[M_2(x,t;\epsilon,\Delta)] \xrightarrow{\Delta\downarrow 0} \sigma^2(x,t)$

$\frac{1}{\Delta}[M_3(x,t;\epsilon,\Delta)] \xrightarrow{\Delta\downarrow 0} 0$

If $P(x,t|x_0,t_0)$, $m(x,t)$ and $\sigma^2(x,t)$ are sufficiently smooth, then the Fokker-Planck equation of this stochastic process is as follows:

$$\begin{aligned} \frac{\partial}{\partial t}p(x,t\,|x_0,t_0) &= \frac{1}{2}\frac{\partial^2}{\partial x^2}[\sigma^2(x,t)p(x,t|x_0,t_0)] \\ &- \frac{\partial}{\partial x}[m(x,t)p(x,t|x_0,t_0)]; \qquad (3.5.4) \\ & \qquad\qquad b>t>t_0>a. \end{aligned}$$

This gives us the evolution of the density distribution of the dynamics in a continuous time stochastic system.

3.6 Relationship between chaotic and stochastic processes

Consider a chaotic system in a general mapping form:

$$x_{t+1} = f(x_t) \qquad (3.6.1)$$

which is referred to equation (2.1.3). The relationship between chaotic and stochastic processes is discussed as follows.

It is proposed to define that the two systems are equivalent if they have a same ergodic invariant measure as described by Wu & Cao [36] and Stamp & Wu [5]. This implies that the chaotic dynamics side will be analysed from the viewpoint of space average and the stochastic system side will be studied from the viewpoint of time average. If the space average of a function is equal to its time average, the function is the probability distribution which will be employed as an invariant measure to bridge the two systems. Based on the concept of the equivalency in invariant measure and ergodicity, the relation between the two systems will be described mathematically through an equivalent stochastic system model derived under certain conditions.

Based on the above existing theory and the ideas mentioned in the introduction to this chapter, the equivalence relation between chaotic and stochastic systems is studied. If the ω-limit set of system defined by equation (3.6.1) is a strange attractor with an ergodic invariant measure, then from the viewpoint of probability and density of dynamics, there exists a stochastic system, with a certain probability measure, equivalent to the chaotic system. The two measures will be identical.

Definition If the ω-limit set of the system defined by equation (3.6.1), $\omega(x)$, has an ergodic invariant measure $\mu(\omega)$ and

$$\mu(\omega) = \lim_{t\to\infty} \pi(x,t), \qquad x \in M \tag{3.6.2}$$

where $\pi(x,t)$ is the probability distribution function of a stochastic process $x(t)$, then the system (equation (3.6.1)) is said being equivalent to the stochastic process. Based on the above definition given by Wu & Cao [36], the same characteristics of the two different kinds of dynamics, chaotic and stochastic, can be found.

3.6.1 The relationship between the two systems

It can be seen that the Fokker-Planck equation provides a link to density of dynamics. Thus the relationship between chaotic dynamics and stochastic processes can be obtained in the following theorem due to Wu & Cao [36].

Theorem If the system defined by equation (3.6.1) is chaotic, and is equivalent in the sense specified by definition (3.6.2) to the Itô stochastic differential equation:

$$\frac{dx}{dt} = g(x) + N(x,t), \qquad x \in M \tag{3.6.3}$$

where $N(x,t)$ is the stationary distribution with a zero mean value and a covariance function $\Omega(x)$ given as:

$$\Omega(x) = |f(x)| \tag{3.6.4}$$

then:

$$g(x) = \frac{1}{2p(x)}\{c + \frac{\partial}{\partial x}[|f(x)|p(x)]\} \tag{3.6.5}$$

where $p(x)$ is the probability density function, c is an arbitrary constant.

Proof: Consider the stochastic system

$$\frac{dx}{dt} = b(x) + \sigma(x)\xi, \qquad x \in M \tag{3.6.6}$$

where $\xi = d\omega/dt$ is known as a white noise, that may be considered the time derivative of a Wiener process. This is the stationary form of equation (3.5.1)

A one-dimensional normalised Wiener process $\{\omega(t)\}_{t\geq 0}$ is a continuous stochastic process with independent increments such that

$$\omega(0) = 0 \tag{3.6.7}$$

and for every $s, t, \quad 0 \leq s < t$, the random variable $\omega(t) - \omega(s)$ has the Gaussian density

$$g(t-s,x) = \frac{1}{\sqrt{2\pi(t-s)}} \exp(\frac{-x^2}{2(t-s)}) \tag{3.6.8}$$

To solve equation (3.6.6) the system can be formally integrated to get

$$x(t) = \int_0^t b(x(s))ds + \int_0^t \sigma(x(s))d\omega(s) + x^0 \tag{3.6.9}$$

An approximate solution can be obtained by using the Euler-Bernstein equation

$$x(t_0 + \Delta t) = x(t_0) + b(x(t_0))\Delta t + \sigma(x(t_0))\Delta\omega \tag{3.6.10}$$

where

$$\Delta\omega = \omega(t_0 + \Delta t) - \omega(t_0) \tag{3.6.11}$$

Existence of solution

The existence and uniqueness of the solution $\{x(t)\}_{t\geq 0}$ to equation (3.6.6) is guaranteed if the Lipschitz conditions guaranteeing the continuity of $b(x)$ and $\omega(x)$

$$|b(x) - b(y)| \leq L|x - y| \tag{3.6.12}$$

$$|\sigma(x) - \sigma(y)| \leq L|x - y| \tag{3.6.13}$$

are satisfied, with L some constant.

Existence of stationary density

It can be shown that limiting density u_* exists if $\sigma(x)$ and $b(x)$ are regular for the Cauchy problem and there is a Lyapunov function V satisfying

$$\sigma^2\frac{\partial^2 V}{\partial x^2} + b(x)\frac{\partial V}{\partial x} \leq -\alpha V(x) + \beta \tag{3.6.14}$$

with positive constants α and β.

Given

$$\bar{b}(x) = -b(x) + \frac{\partial\sigma^2(x)}{\partial x} \tag{3.6.15}$$

and

$$\bar{c}(x) = \frac{1}{2}\frac{\partial^2\sigma^2(x)}{\partial x^2} - \frac{\partial b}{\partial x} \tag{3.6.16}$$

$\sigma(x)$ and $b(x)$ are regular for the Cauchy problem if they are C^4 functions such that the corresponding coefficients σ, $\bar{b}(x)$, $\bar{c}(x)$ satisfy the uniform parabolicity condition

$$\sigma^2 \geq \rho \tag{3.6.17}$$

where ρ is a positive constant, and the growth conditions

$$|\sigma^2(x)| \leq M \tag{3.6.18}$$

$$|\bar{b}(x)| \leq M(1 + |x|) \tag{3.6.19}$$

$$|\bar{c}(x)| \leq M(1 + |x|^2) \tag{3.6.20}$$

are satisfied.

A Lyapunov function is any function $V : \Re \rightarrow \Re$ that satisfies the following four properties:

$$V(x) \geq 0 \ \forall x \tag{3.6.21}$$

$$\lim_{|x| \rightarrow \infty} V(x) = \infty \tag{3.6.22}$$

where V has continuous derivatives $\dfrac{\partial V}{\partial x}, \dfrac{\partial^2 V}{\partial x^2}$ and

$$V(x) \leq \rho e^{\delta|x|}, \quad |\frac{\partial V(x)}{\partial x}| \leq \rho e^{\delta|x|}, \quad |\frac{\partial^2 V(x)}{\partial x^2}| \leq \rho e^{\delta|x|} \tag{3.6.23}$$

for some constants ρ, δ.

The stationary distribution

If the density function $u(x,t)$ of the process $x(t)$ is considered the Fokker-Planck equation can be used

$$\frac{\partial u(x,t)}{\partial t} = -\frac{\partial}{\partial x}[b(x)u(x,t)] + \frac{1}{2}\frac{\partial^2}{\partial x^2}[\sigma^2(x)u(x,t)]. \tag{3.6.24}$$

to describe its time evolution. In order to ensure the existence and differentiability of u it is sufficient that

$$\sigma(x), \frac{\partial\sigma}{\partial x}, \frac{\partial^2\sigma}{\partial x^2}, b(x), \frac{\partial b}{\partial x}, \frac{\partial u}{\partial t}, \frac{\partial u}{\partial x}, \text{and} \quad \frac{\partial^2 u}{\partial x^2}$$

are continuous for $t > 0$.

Lasota & Mackey [1] state the theorem that assuming the coefficients are regular for the Cauchy problem and the inequality

$$\int_0^\infty e^{-B(x)}dx = \int_\infty^0 e^{-B(x)}dx = \infty \tag{3.6.25}$$

where

$$B(x) = \int_0^x \frac{2b(y)}{\sigma^2(y)}dy \tag{3.6.26}$$

is satisfied then the system is asymptotically stable if

$$\int_{-\infty}^\infty \frac{1}{\sigma^2(x)}e^{B(x)}dx < \infty \tag{3.6.27}$$

They also note that if

$$xb(x) \leq 0 \text{ for } |x| \geq r \tag{3.6.28}$$

where r is a positive constant, then equation (3.6.25) is satisfied. This condition simply means that the interval $[-r, r]$ is attracting (or at least not repelling) for trajectories of the unperturbed equation

$$\frac{dx}{dt} = b(x), \qquad x \in M \tag{3.6.29}$$

As the steady state probability density function is independent of time, it can be seen that:

$$\begin{aligned} \lim_{t\to\infty} u(x,t) &= u(x), \\ \lim_{t\to\infty} \frac{\partial u(x,t)}{\partial t} &= 0. \end{aligned} \tag{3.6.30}$$

Applying these to equation (3.6.24) the limiting function $u(x)$ is the unique density satisfying the elliptic equation

$$\frac{1}{2}\frac{\partial^2}{\partial x^2}\{[\sigma^2(x)u(x)] - \frac{\partial}{\partial x}[b(x)u(x)]\} = 0. \tag{3.6.31}$$

making the substitution $z = \sigma^2 u$ into equation (3.6.31)

$$\frac{dz}{dx} = \frac{2b(x)}{\sigma^2(x)}z + c_1 \tag{3.6.32}$$

is obtained, this has the solution

$$z(x) = e^{B(x)}\{c_2 + c_1 \int_0^x e^{-B(y)}dy\} \tag{3.6.33}$$

where c_1 and c_2 are constant and $B(x)$ is given by equation (3.6.26).

From equation (3.6.28) if follows that

$$\int_0^x e^{-B(y)}dy \tag{3.6.34}$$

converges to ∞ as $x \to \infty$ and $-\infty$ as $x \to -\infty$, therefore to satisfy equation (3.6.27), $c_1 = 0$. This gives us the solution

$$u_*(x) = \frac{c_2}{\sigma^2(x)}e^{B(x)} \tag{3.6.35}$$

with $c_2 > 0$. This can be rearranged to give $b(x)$ in terms of $u_*(x)$ an $\sigma(x)$. This gives a result equivalent to that given by Wu & Cao [36].

$$b(x) = \frac{1}{2u(x)}\frac{\partial}{\partial x}[\sigma^2(x)u(x)] \tag{3.6.36}$$

Example

Consider the example given in Wu & Cao [36], $b(x) = 1 - 2x$ and $\sigma(x) = 2\sqrt{x(1-x)}$, $x \in [0, 1]$. This gives us

$$B(x) = \int_0^x \frac{2b(y)}{\sigma^2(y)} = \int_0^x \frac{2-4y}{4y(1-y)}dy = \frac{1}{2}\ln(4x(1-x)) \tag{3.6.37}$$

thus

$$u_*(x) = \frac{c_2}{4x(1-x)}\sqrt{4x(1-x)} = \frac{c_2}{2\sqrt{x(1-x)}} \tag{3.6.38}$$

which using

$$\int_0^1 u_*(x)dx = 1 \tag{3.6.39}$$

gives

$$u_*(x) = \frac{1}{\pi\sqrt{x(1-x)}} \tag{3.6.40}$$

which is the limiting density function of the logistic mapping as required.

However in order to guarantee the existence of the limiting density certain conditions detailed above must be met. It should be noted that these conditions are only required to be met over the region of the strange attractor. The behaviour outside this region is irrelevant to the model since the density is zero, and only the behaviour of the system once it is no longer transient is of interest. For the existence of a solution to equation (3.6.6) it is required that the Lipschitz conditions are satisfied. Using our examples $b(x)$ and $\sigma(x)$, the attractor is in the range $0 < x < 1$. This gives

$$|b(x) - b(y)| = 2|x - y| \tag{3.6.41}$$

and

$$|\sigma(x) - \sigma(y)| = 2|\sqrt{x(1-x)} - \sqrt{y(1-y)}| \tag{3.6.42}$$

Equation (3.6.41) clearly satisfies equation (3.6.12). Equation (3.6.42) satisfies equation (3.6.13) for $0 \leq x \leq 1$, and the solution is not real outside this range. However it is sufficient that equation (3.6.13) is satisfied for $0 < x < 1$ so the Lipschitz conditions are satisfied.

For this system

$$\sigma(x) = 2\sqrt{x(1-x)} \tag{3.6.43}$$

$$\frac{\partial\sigma}{\partial x} = \frac{1-2x}{\sqrt{x(1-x)}} \tag{3.6.44}$$

$$\frac{\partial^2\sigma}{\partial x^2} = \sqrt{\frac{1-x}{x^3}} - \sqrt{\frac{x}{(1-x)^3}} \tag{3.6.45}$$

$$b(x) = 1 - 2x \tag{3.6.46}$$

$$\frac{\partial b}{\partial x} = -2 \tag{3.6.47}$$

$$u(x) = \frac{1}{\pi\sqrt{x(1-x)}} \tag{3.6.48}$$

$$\frac{\partial u}{\partial x} = \frac{2x-1}{2\pi\sqrt{x^3(1-x)^3}} \tag{3.6.49}$$

$$\frac{\partial^2 u}{\partial x^2} = \frac{(2x-1)(-2x^2+5x-\frac{3}{2})}{2\pi\sqrt{x^3(1-x)}} \tag{3.6.50}$$

and thus the conditions that

$$\sigma(x), \frac{\partial\sigma}{\partial x}, \frac{\partial^2\sigma}{\partial x^2}, b(x), \frac{\partial b}{\partial x}, \frac{\partial u}{\partial x}, \text{and} \quad \frac{\partial^2 u}{\partial x^2}$$

are continuous are satisfied for $x \in [0\ 1]$.

$$|\sigma^2(x)| = |4x(1-x)| \leq 1 \text{ for } 0 < x < 1 \tag{3.6.51}$$

$$\bar{b}(x) = -6x + 3 \leq 6(1 + |x|) \tag{3.6.52}$$

$$\bar{c}(x) = 8 \leq (1 + |x^2|) \tag{3.6.53}$$

thus with $M = 8$, $b(x)$ and $\sigma(x)$ are regular.

If the Lyapunov function $V(x) = x^2$ is considered

$$\sigma^2\frac{\partial^2 V}{\partial x^2} + b(x)\frac{\partial V}{\partial x} = 10x - 12x^2 \leq -\alpha x^2 + \beta \tag{3.6.54}$$

for $\alpha = 2$ and $\beta = 10$. This ensures that the limiting distribution u_* exists. This demonstrates that the example given in Wu & Cao [36] is valid.

Further considerations

In Wu & Cao [36] $\sigma^2(x)$ is defined as $|f(x)|$, however equation (3.6.35) actually gives a family of solutions as it specifies the relationship between $b(x)$ and $\sigma(x)$. In order to guarantee the existence of the asymptotically stable solution it is sufficient that the conditions noted previously are met.

It should be noted that in general it will usually not be possible to find an analytical expression for the limiting density of a chaotic process, particularly in higher dimensions. Only in special cases is an analytical solution available. In such cases it is not always the case that the limiting density will fit the conditions required for the Fokker-Plank equation to be valid. The condition that

$$u(x) \ , \ \frac{\partial u}{\partial x} \text{ and } \frac{\partial^2 u}{\partial x^2} \tag{3.6.55}$$

are continuous may not be true. However it should be possible to approximate u_* with a sum of functions such as Gaussians (which do meet the conditions) to an arbitrary degree of accuracy and this approximation would have a valid Fokker-Plank equation and thus an equivalent stochastic system.

3.6.2 Two dimensional extension

Consider the two dimensional stochastic system

$$\frac{dX}{dt} = b(X) + \sigma(X)\xi, \qquad X \in M^2 \tag{3.6.56}$$

$$b(x) = \begin{bmatrix} b_1(x) \\ b_2(x) \end{bmatrix} \tag{3.6.57}$$

$$\sigma(x) = \begin{bmatrix} \sigma_{11}(x) & \sigma_{12}(x) \\ \sigma_{21}(x) & \sigma_{22}(x) \end{bmatrix} \tag{3.6.58}$$

$$\xi = \begin{bmatrix} \frac{d\omega_1}{dt} \\ \frac{d\omega_2}{dt} \end{bmatrix} \tag{3.6.59}$$

where $\omega(t)$ is a two-dimensional vector process

$$\omega(t) = \{\omega_1(t), \omega_2(t)\} \tag{3.6.60}$$

with its components $\{\omega_1(t)\}_{t\geq 0}, \{\omega_2(t)\}_{t\geq 0}$ are one-dimensional independent Wiener processes. Thus the process has the joint density

$$g(t, x_1, x_2) = \frac{1}{(2\pi t)} \exp(\frac{-(x_1^2 + x_2^2)}{2t}) \tag{3.6.61}$$

To solve equation (3.6.56) the system can be formally integrated as with the one dimensional case to get equation (3.6.9).

An approximate solution can be obtained by using the Euler-Bernstein equations (3.6.10) and (3.6.56) giving

$$\begin{aligned} x_1(t_0 + \Delta t) &= x_1(t_0) + b_1(x_1(t_0), x_2(t_0))\Delta t + \sigma_{11}(x(t_0))\Delta\omega_1 + \sigma_{12}(x(t_0))\Delta\omega_2 \\ x_2(t_0 + \Delta t) &= x_2(t_0) + b_2(x_1(t_0), x_2(t_0))\Delta t + \sigma_{21}(x(t_0))\Delta\omega_1 + \sigma_{22}(x(t_0))\Delta\omega_2 \end{aligned} \tag{3.6.62}$$

where

$$\Delta\omega_d = \omega_d(t_0 + \Delta t) - \omega_d(t_0) \tag{3.6.63}$$

If the density function $u(x,t)$ of the process $x(t)$ is considered, the two dimensional Fokker-Planck equation

$$\begin{aligned} \frac{\partial u(x,t)}{\partial t} &= -\frac{\partial}{\partial x_1}[b_1(x)u(x,t)] - \frac{\partial}{\partial x_2}[b_2(x)u(x,t)] \\ &+\frac{1}{2}\frac{\partial^2}{\partial x_1^2}[(\sigma_{11}^2(x) + \sigma_{12}^2(x))u(x,t)] + \frac{\partial^2}{\partial x_1 \partial x_2}[(\sigma_{11}(x)\sigma_{21}(x) \\ &+\sigma_{12}(x)\sigma_{22}(x))u(x,t)] + \frac{1}{2}\frac{\partial^2}{\partial x_2^2}[(\sigma_{21}^2(x) + \sigma_{22}^2(x))u(x,t)] \end{aligned} \tag{3.6.64}$$

can be utilised to describe its time evolution. The equation is valid under the conditions that

$$\sigma_{ij}(x), \frac{\partial \sigma_{ij}}{\partial x_k}, \frac{\partial^2 \sigma_{ij}}{\partial x_k \partial x_l}, b_i(x), \frac{\partial b_i}{\partial x_k}, \frac{\partial u}{\partial t}, \frac{\partial u}{\partial x_k}, \text{and} \quad \frac{\partial^2 u}{\partial x_k \partial x_l}$$

are continuous for $t > 0$ and $x \in \Re^2$ and $b_i(x), \sigma_{ij}(x)$ and their first derivatives are bounded.

The limiting density u_* exists if the $\sigma_{ij}(x)$ and $b_i(x)$ are regular for the Cauchy problem and there is a Lyapunov function V satisfying

$$\sum_{i,j=1}^{2} \left[\sum_{k=1}^{2} \sigma_{ik}(x)\sigma_{jk}(x)\right] \frac{\partial^2 V}{\partial x_i \partial x_j} + \sum_{i=1}^{2} b_i(x)\frac{\partial V}{\partial x_i} \leq -\alpha V(x) + \beta \tag{3.6.65}$$

with positive constants α and β.

As the steady state probability density function is independent of time equation (3.6.30) applies and together with equation (3.6.64) it is demonstrated that the limiting function $u(x)$ is the unique density satisfying the elliptic equation

$$\begin{aligned}\frac{\partial}{\partial x_1}[b_1(x)u(x)] + \frac{\partial}{\partial x_2}[b_2(x)u(x)] &= \frac{1}{2}\frac{\partial^2}{\partial x_1^2}[(\sigma_{11}^2(x) + \sigma_{12}^2(x))u(x)] \\ &+ \frac{\partial^2}{\partial x_1 \partial x_2}[(\sigma_{11}(x)\sigma_{21}(x) + \sigma_{12}(x)\sigma_{22}(x))u(x)] \\ &+ \frac{1}{2}\frac{\partial^2}{\partial x_2^2}[(\sigma_{21}^2(x) + \sigma_{22}^2(x))u(x)]\end{aligned} \tag{3.6.66}$$

This demonstrates that equivalent stochastic systems exist for dimensions higher than one.

3.6.3 Examples

It has been shown that it is possible to consider a stochastic system equivalent to a chaotic system when they share certain characteristics. Now the construction of a stochastic model of a chaotic system is considered.

The Logistic Mapping is a deterministic chaotic mapping

$$x_{n+1} = Kx_n(1 - x_n) \tag{3.6.67}$$

and for $K = 4$ it has the analytical solution

$$x_n = \frac{1}{2} + \frac{1}{2}\cos(2^n \pi \theta_0) \qquad 0 \leq x \leq 1 \tag{3.6.68}$$

$$x_{n+1} = \frac{1}{2} + \frac{1}{2}\cos(2^{n+1} \pi \theta_0) \qquad 0 \leq x \leq 1 \tag{3.6.69}$$

From the above it can be seen that there is a one to one mapping between x_n and θ_n and that if how θ evolves is considered it is found that

$$\theta_{n+1} = 2\theta_n \qquad 0 \leq \theta \leq 1 \tag{3.6.70}$$

Given an exact initial value of θ_0 and the above equations the evolution of the system can be determined for however far into the future is required. However

when the value of θ_0 can only be obtained to a finite precision as is generally the case then the situation alters. If the value of θ is determined to a precision of 8 bits, writing θ in binary form gives

$$\theta_0 = 0.b_1b_2b_3b_4b_5b_6b_7b_8 \tag{3.6.71}$$

$$\theta_1 = 0.b_2b_3b_4b_5b_6b_7b_8? \tag{3.6.72}$$

$$\theta_8 = 0.???????? \tag{3.6.73}$$

where ? represents a bit whose value cannot be determined. The evolution of the system with time corresponds to a binary shift operator. After one time step although the values first 7 bits are known with certainty, the value of the 8th bit is dependent on the 9th bit of θ_0. However the value of the 9th bit is not known. One method for dealing with this uncertainty is to use probability. The are two possible values for the 9th bit, 0 or 1. There is no reason to prefer either of these values and so they can be treated as equally probable. It is to be noted that after 8 iterations the value of θ_8 and thus x_8 is independent of the eight bit value of θ_0. This demonstrates the sensitivity of chaotic systems to initial conditions and the consequent problems with long term prediction. It is now possible to form a new solution to the Logistic mapping, valid when the value of θ_n is known only to a limited precision.

$$x_{n+1} = \frac{1}{2} + \frac{1}{2}\cos(2^{n+1}\pi\theta_0) + F(\theta_n) \tag{3.6.74}$$

It can be seen that this solution consists of a deterministic term, and a stochastic term, both of which are solely dependent on θ_n. This can easily be adjusted to give an equation with a zero mean stochastic term and thus can be considered a discrete time version of equation (3.6.6).

The question remains, is it feasible to construct a stochastic model of the dynamical system. Using again the one dimensional chaotic function, the logistic mapping which for $K = 4$ has the solution

$$x_n = \sin^2(2^n\pi I) \qquad 0 \leq x \leq 1 \tag{3.6.75}$$

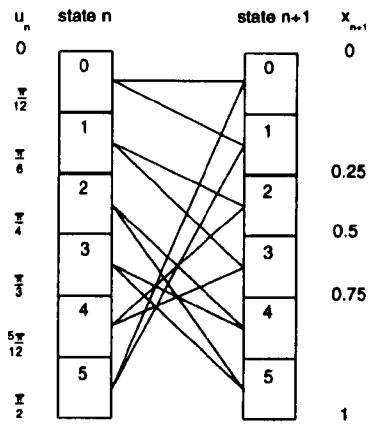

Figure 3.1: State transition with a state transformation

where

$$I = \frac{\sin^{-1}\sqrt{x_0}}{\pi} \tag{3.6.76}$$

with the long term density distribution

$$f(x) = \frac{1}{\pi\sqrt{1-x}} \tag{3.6.77}$$

If a transformation is defined

$$u = \sin^{-1}(\sqrt{x}) \tag{3.6.78}$$

then

$$u_{n+1} = 2u_n, \qquad 0 \leq u \leq \pi \tag{3.6.79}$$

and it can be found using the Peron-Frobentius function that the long term density distribution in the u domain is

$$g(u) = \frac{1}{\pi} \tag{3.6.80}$$

Thus the dynamics are uniformly distributed over the interval $0 \leq u \leq \pi$ after performing this transformation.

Given that only whether the value of the function falls within an interval can be measured, a stochastic model of the process in the limited information

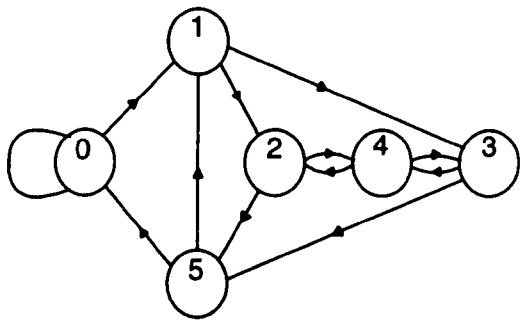

Figure 3.2: State transition diagram of 6 state model of the Logistic mapping

situation can be built. If the value of the logistic mapping can only be observed to fall within one of four equally sized observation intervals then a six state Markov model can be constructed by transforming to the u domain and partitioning into 6 equal states Figure 3.1. When in each state the output of the system corresponds to one of the 4 specified observation intervals. From this a state transition diagram can be formed, Figure 3.2. For any sequence of output intervals generated by the model, initial conditions can be found for the logistic mapping that will generate the same quantized sequence. Conversely any quantized sequence generated by the mapping may be generated by the model. This model can be considered optimal in the sense that no other model of the 4 quantisation level logistic mapping will give improved performance.

As with the logistic mapping the dimensionality of the process is one. The long term probability density function of the model is also the same as that of the quantized logistic mapping. As with the logistic mapping the future output of the model can only be predicted in the short term.

That this model can be constructed demonstrates that a stochastic model can be used to effectively model a chaotic function when the function value is subject to a degree of uncertainty. In almost all practical problems this degree of uncertainty exists, due to such factors as observation noise. Thus the methodology of using an stochastic model should be valid for modelling real

world chaotic time series.

3.7 Conclusion

The relationship between chaotic and stochastic systems has been investigated. Based on the dynamic density of chaotic systems and Brownian motion in stochastic systems, the equivalency in invariant measure and ergodicity between the two systems has been defined. Based on the definition, the equivalence relationship is studied and the equivalent stochastic system model of the chaotic system is derived analytically, which leads to an equivalent stochastic system model. This provides a potential to deal with the problems of high-dimensional chaotic systems using low-dimensional equivalent models.

It has been demonstrated that the behaviour of chaotic system defined by equation (3.5.1) can be described as a stochastic process. Thus its dynamics are characterized by the initial distribution $p(x_0)$ and the transition probability density function $p(x_{k+1}|x_k), k = 1, 2, ..., n$. $p(x_k)$ and $p(x_{k+1}|x_k)$ can conveniently be obtained using the time series of the chaotic dynamics. In other words, if the probability density function of a chaotic system has been obtained, the chaotic dynamics can be represented by an equivalent stochastic system model which is constructed based on the probability density function.

Inversely, an equivalent chaotic system can be constructed if a steady state probability density function of a stochastic system is given.

It can be seen that the probability measure $\mu(\omega)$ mentioned is actually the probability distribution function. This implies $\mu(\omega) = \int_\omega p(x)dx$. Thus it can be stated that a chaotic system and a stochastic system are equivalent if they have a same ergodic invariant measure.

Chapter 4

Equivalent Model

4.1 Introduction

Based on the links between stochastic and chaotic systems a new methodology for the study of chaotic time series is obtained. It consists of describing the chaotic dynamic system as an equivalent stochastic system. The dynamics of this stochastic system are described in terms of its initial and transitional probability density functions. These density functions are determined directly from the time series. Since motions of chaotic dynamical systems, after some transients, settle down to strange attractors and sustain for a long time, this provides a great opportunity in terms of time period for a computer to learn the properties of the chaotic systems through space-time patterns so as to build an equivalent stochastic model. This stochastic model is then used to predict the dynamics of the chaotic system. As long as learning proceeds, theoretically prediction of future states of the chaotic systems can be achieved. Based on the concept of the equivalence relation and the equivalent system model, system reconstruction of chaotic dynamics using learning techniques can be investigated.

It is clear that the simple Markov chain models developed in this chapter

are an extension of the symbolic dynamics approach, utilising more symbols, and replacing the 1s in the transition matrix by probabilities. Thus the Markov chain model fulfills the criteria of allowing only valid symbol sequences since it has the same symbolic dynamics as the underlying system.

It is also clear that some of the invariant properties of the strange attractor such as the topological entropy can be calculated directly from the model.

4.2 Construction of the equivalent models by learning

An equivalent stochastic model can be used to reconstruct a unknown chaotic system as mentioned in the previous section. The reconstruction can be achieved based on the probability density function which may be obtained using learning approaches to the time series of the chaotic system.

Since the state of the chaotic system can only be estimated to a limited precision, instead of trying to model an individual trajectory, the evolution of the density of the dynamics can be modelled.

Given a probability density function of the density of dynamics at time t, the Frobenius-Perron operator for the system gives a link to the density of the dynamics at time $t+1$. It is not possible to analytically derive the Frobenius-Perron operator without the system equations. However it should be possible to create a model which can approximate the operator via learning.

There are two possible approaches to modelling the evolution of the density of dynamics. The first is to model the transition function and use that model to derive an estimate of the Frobenius-Perron operator. A method for approximating the Frobenius-Perron operator is given by Dellnitz & Junge [29]. This can then be applied to an initial density estimate to determine the evolution of the density.

The second approach is to model the evolution of the density directly. This is the approach that has been taken.

The first question is how to represent the probability density function of the dynamics. A common approach to modelling density functions is to approximate them as a sum of basis functions.

$$f(x) = \sum_{i}^{N} a_i G_i(x) \tag{4.2.1}$$

This is a universal approximator in that it can approximate arbitarily well any multivariate continuous function on a compact subset of $\Re^k$, given a sufficient number of suitable functions ,N, (Haykin[39]). If a set of suitable basis functions is defined to cover the state space of the system, the link between the set of coefficients at time t, $A(t) = (a_1(t), ..., a_N(t))$, and the set of coefficients at time $t+1$, $A(t+1)$ needs to be modelled.

It is easy to see that if the density function at time t is equal to one of the set of approximating functions G_i, then $a_i = 1$, and $a_j = 0$ where $j \neq i$. At time $t = 1$ the density function will be represented by a new set of coefficients $A(t+1)$. Thus there are a set of coefficients (independent of time) which link $f_t = G_i$ to $f_{t+1} = \sum_j m_{ij} G_j$.

However it is a property of the Perron-Frobenius function that it is linearly additive i.e.

$$P(\lambda_1 f_1 + \lambda_2 f_2) = \lambda_1 P f_1 + \lambda_2 P f_2 \tag{4.2.2}$$

thus if the coefficients for each of the N functions $G_i(x)$ are found then if

$$f(x)_t = \sum_{i=0}^{N} a_i G_i(x) \tag{4.2.3}$$

then

$$f(x)_{t+1} = \sum_{i=0}^{N} \sum_{j=0}^{N} m_{ij} a_i G_j \tag{4.2.4}$$

or looking at the evolution of the coefficient vector A

$$A_{t+1} = M A_t \tag{4.2.5}$$

The simplest functional form for $G_i(x)$ is the characteristic function for a set $\triangle$ which is defined as

$$1_{\triangle} = \begin{cases} 1 \text{ if } x \in \triangle \\ 0 \text{ if } x \notin \triangle \end{cases} \tag{4.2.6}$$

If the probability of occupying a certain state is considered as proportional to the density of dynamics of that state then this gives a discrete state space Markov chain model.

4.3 1D, 2D, nD, Markov chain models

4.3.1 A simple learning methodology

The above shows that a chaotic system can be described by an equivalent stochastic process. It is to be noted that no knowledge of the chaotic system is required beyond its dimension and that it converges to a strange attractor. Since that equivalent system may be described in terms of the probability density function and initial probability distribution, the problem of modelling the equivalent stochastic system is one of extracting these quantities from the time series generated by the chaotic system.

In order to find an equivalent stochastic model for a chaotic time series a new time series y_t must first be constructed by embedding the time series as described previously in equation (2.4.3). The initial and conditional probability density functions must then be estimated.

The simplest method for estimating these values is to quantize the invariant interval of the chaotic dynamics into n subintervals $I^i, i = 1, 2, ..., n$ of equal length, or in the case of an embedding dimension higher than 1, into d-dimensional hypercubes. The points which are allocated to the subinterval I^i are all denoted as the same value y^i, i.e. they are quantized. It is to be noted that it is not compulsory to have intervals of equal length, merely convenient.

Then using the time average to replace the set average the probability density function can be estimated through a learning process as:

$$P_k(x^i) = N_k(x^i)/k; \quad x^i \in I^i, \quad i = 1, 2, ..., n \tag{4.3.1}$$

where P, N and k are the probability, frequency and number of samples respectively. This can also be realised using the following recurrence formula:

$$P_{k+1}(x^i) = \begin{cases} \frac{k}{k+1} P_k(x^i) + \frac{1}{k+1}, & x \in x^i \\ \frac{k}{k+1} P_k(x^i), & x \notin x^i \end{cases} \tag{4.3.2}$$

where $0 \leq P_k(x^i) \leq 1$, $i = 1, 2, ..., n$ and $\sum_{i=1}^{n} P_k(x^i) = 1$. The transitional probability can be obtained in an analogous manner. If the minimum variance method is used as the performance index:

$$\min_{x^i} E\{(x^i - \hat{x}_{k+1})^2\}. \tag{4.3.3}$$

where $\tilde{x}$ is the estimate error and $\tilde{x}^i = x^i - \hat{x}_{k+1}$. $\hat{x}_{k+1}$ can be obtained as a conditional mean:

$$\hat{x}_{k+1} = \sum_{i=1}^{n} x^i P(x^i | x_k). \tag{4.3.4}$$

This can be regarded as a weighted mean of states. In its special case, x_{k+1} can be predicted using the maximum transition probability:

$$\hat{x}_{k+1} = \{x^i | \max_{x^i} P(x^i | x_k)\} \tag{4.3.5}$$

Problems

This simple model suffers from a number of problems. If the embedding dimension d is too low then process noise (errors due to that part of the system not modelled) appears. This can be solved by increasing the embedding dimension d.

Intrinsic to the partitioning of the state space into hypercubes is the appearance of quantisation noise. This can be reduced by reducing the size of the hypercubes.

However both reducing the size of the hypercubes and increasing the embedding dimension will lead to a rapid increase in the number of hypercubes required.

If the state space is filled by hypercubes of a fixed side length ε then the number of hypercubes required to fill the state space is N where

$$N \propto (\frac{1}{\varepsilon})^d \tag{4.3.6}$$

thus there is an exponential relationship between the embedding dimension and the number of hypercubes required for a fixed side length. However, since a chaotic time series is being dealt with, it is not required to fill the entire state space. In state space the series will relax onto a strange attractor so only those hypercubes which contain elements of the attractor are needed. This in general means that the required N is considerably less in most cases than that given in equation (4.3.6). It will in fact be estimated from the capacity dimension D_c equation (2.2.7) giving

$$N \propto (\frac{1}{\varepsilon})^{D_c} \tag{4.3.7}$$

However there is still an exponential increase in the number, just at a lower rate.

A larger number of hypercubes, and hence states, will naturally require an increased number of model parameters, especially since the size of the transition table is dependent on the number of states squared.

An increased number of states will also require a correspondingly longer data series in order for reasonably representative probability density functions to be estimated.

In order to reduce the effects of noise without dramatically increasing the number of states along with the associated problems the simple stochastic model may be modified in a number of ways. The main idea being to combine the predictions of a number of submodels as in Warwick & Karny [22].

4.3.2 Weighted sum of models

A straight forward method is to construct N 1D models, based on $P(x^i|x_k)$ to $P(x^i|x_{k-N})$ and to take the mean of their predictions, thus reducing the effects of noise. The effectiveness of this is limited however, since, as a chaotic system is being dealt with, the Lyapunov exponent σ is greater than 0. This indicates that on average there is an exponential divergence of trajectories in phase space leading to increasing inaccuracies in prediction as prediction further ahead in time is attempted. Thus if the number of models used, N, is too large the accuracy of the mean prediction decreases rather than improves. The optimum value of N depends on the level of noise present and the rate of divergence of trajectories. This model can be improved by weighting the values of each individual prediction giving as the prediction:

$$\hat{x}_{k+1} = \sum_{j=0}^{N} w_j \sum_{i=1}^{n} x^i P(x^i|x_{k-j}). \tag{4.3.8}$$

This can also be viewed as a weighted sum of the conditional probability density functions. A simple weighting scheme would be to weight the predictions based on more recent values more heavily. It should be noted however that although on average the more recent predictions are more accurate on an individual basis this is not always so.

Several different weighting schemes were considered. The simplest was equally weighted as described initially. As discussed above, this weights the further ahead predicting models too heavily. In order to weight the more recent models more heavily another model was proposed where the weights were assigned proportionally to the prediction delay (i.e the $P(x^i|x_{k-j})$ term was given a weighting of $N - j$).

The state trajectories of a chaotic system diverge at an exponential rate. This implies than on average the prediction errors due to predicting further ahead will also increase in an exponential manner as would the uncertainty

in prediction (the width of the associated density function). An appropriate model would be to adopt a weighting of $\frac{1}{n^j}$ where n is an arbitrary constant.

As has been stated earlier it is not necessarily true that a j step ahead prediction model will give a more certain answer than a $j+1$ step ahead model in a specific instance, although it will do so on average. It would therefore seem desirable to adopt a weighting scheme based on how certain the model is about its answer. Two weighting methods were used to implement this approach. The first method was to use the variance of each models probability density function for each prediction as a measure of certainty and to weight the model accordingly. However the variance has disadvantages for this purpose. This is because if the density function consists of a few relatively probable values spread far apart the variance is high despite a high degree of certainty as to the answer. In order to overcome this a new measure was used - the entropy of the distribution.

Entropy is a quantity utilised in information theory in order to measure the amount of information being delivered and is defined as follows:-

$$H(p) = -\sum_{i=0}^{R} p_i \log(p_i) \tag{4.3.9}$$

where p_i is the probability of event i and R is the number of possible events. This tells us that an entropy gives a value of 0 when an event is certain, and a value of $\log(R)$ when all events are equally likely. The entropy can be considered as measuring the roughness of the density function. The weighting given to each density function is a function of the entropy. The function utilised was $\frac{1}{H(p)}$.

It is to be noted that in all the weighting schemes the weights are normalised.

4.3.3 Probability density function combinatorial models

As was noted previously the weighted sum of models is actually also a weighted sum of density functions. This can be viewed as similar in philosophy to a Bayesian approach (although different in execution) in that it is using prior information as to the shape of the density function (from the j-step ahead models to alter the shape of the 1-step ahead model). It can be viewed as trying to form an estimate of the conditional probability density function $P(x^i|x_k,...,x_{k-N})$ from a number of conditional probability density functions $P(x^i|x_{k-j}), j = 0 \rightarrow N$. Now it is possible to evaluate $P(x^i|x_k,...,x_{k-N})$ directly, however this is simply equivalent to increasing the embedding dimension with all the problems that entails. The other approach is to find methods of combining $P(x^i|x_{k-j}), j = 0 \rightarrow N$ to produce a modified density function $\hat{P}(x^i|x_k,...,x_{k-N})$, an estimate of the density function of a higher dimensional model.

What is needed is a method of combining density functions. In the case that $P(x^i|x_k)$ and $P(x^i|x_{k-j})$ are independent then an appropriate model would be:

$$\hat{P}(x^i|x_k,...,x_{k-N}) = \prod_{j=0}^{N} P(x^i|x_{k-j}) \tag{4.3.10}$$

However, it is known that the density functions are not independent. It is not known what that dependence is however, so in the absence of that knowledge this may prove a reasonable model as discussed by Justice [21].

Looking at the problem of combining density functions it can be seen that there are a few desirable features for a method to have:

- where there is a probability of 0 (no chance of event occurring) at a point in one of the density functions then there should be a 0 at that point in the combined density function.
- where there is a probability of 1 (event is certain to occur) at a point in one of the density functions then there should be a 1 at that point in the

combined density function. This is true if the above feature is true.

- when the density functions are identical then the combined density function should be the same as our knowledge has not been increased in any way.

The sum of weighted models method has only the last of these properties. It can be seen that although the assumption of independence method has the first two features it does not have the last. It warps the density function in favour of the more likely results at the expense of those with only a small chance of occurring. In effect it moves towards choosing the most probable result so it is still likely to give fairly reasonable results.

4.3.4 Model learning with information entropy

If information theory is turned to again, the entropy of a distribution relative to another distribution is defined as:

$$H(p:q) = -\sum_{i=0}^{R} p_i \log(\frac{p_i}{q_i}) \tag{4.3.11}$$

Remembering that entropy is a measure of certainty, a combined density function is desired that has a maximum entropy relative to the initial density functions, i.e. it is desired to maximise the sum $H(r:p) + H(r:q)$ where r is the combined conditional density function.

$$H(r:p) + H(r:q) = -\sum_{i=0}^{R} 2r_i \log(\frac{r_i}{p_i^{1/2} q_i^{1/2}}) \tag{4.3.12}$$

The relative entropy $H(r:p)$ is maximised when $p = r$ so the new combined probability distribution estimate is:

$$\hat{P}(x^i|x_k, ..., x_{k-N}) = \prod_{j=0}^{N} P(x^i|x_{k-j})^{1/N} \tag{4.3.13}$$

This new model meets all of the criteria established earlier. With both these models the combined conditional density function needs to be normalised.

4.3.5 Variable size quantisation layers

All the stochastic models developed have used a fixed size of quantisation interval. However since as can be seen from the conditional density functions, some states lead to larger prediction errors than others. It would possibly make sense to vary the size of the quantisation layers in order to improve the accuracy of prediction. An algorithm for variable quantisation was not developed, but since the logistic function is relatively easy to analyse it was decide to try and determine the effect analytically.

This gives us for the logistic function, for an interval i $[a, b]$

$$p_i = \frac{2}{\pi}[u]_\alpha^\beta \tag{4.3.14}$$

$$\mu_i = \frac{1}{\pi p_i}[u - \frac{1}{2}\text{sin2u}]_\alpha^\beta \tag{4.3.15}$$

$$\sigma_i^2 = \frac{1}{2\pi p_i}[\frac{3}{2}u - \text{sin2u} + \frac{1}{8}\text{sin4u}]_\alpha^\beta - \mu_i^2 \tag{4.3.16}$$

$$\alpha = \sin^{-1}\sqrt{\text{a}}, \beta = \sin^{-1}\sqrt{\text{b}} \tag{4.3.17}$$

where p_i is the long term probability of being in that interval or state, and μ_i and σ_i^2 are the mean and variance of the one step ahead prediction probability density function for that state. The root mean squared error is defined by

$$e_{\text{rms}}^2 = \frac{1}{N}\sum_{t=0}^{N}(x_t - \hat{x}_t)^2 \tag{4.3.18}$$

The contribution of state i to the overall squared error is

$$p_i\sigma_i^2 \tag{4.3.19}$$

and the total squared error is

$$\sum_i p_i\sigma_i^2 \tag{4.3.20}$$

which is the function that is wished to be minimised. There is no clear approach as to how to do this so the state boundaries were adjusted so that each state contributed the same amount to the total error. The error when the states

were equally quantized was also calculated and was found to be equal to that found though the simulation studies. The theoretical variable quantisation was found to improve the root mean square error performance for ten quantisation levels by 31 percent. Although this is a significant amount it is probably not

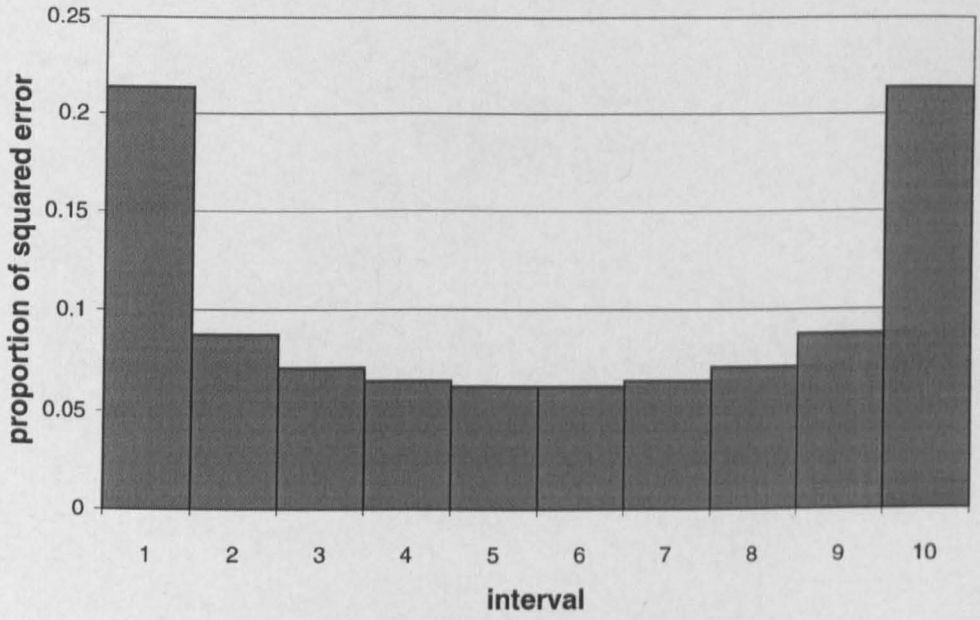

Figure 4.1: Contribution of each state to overall error

A similar evaluation for more than one step ahead prediction has not been performed and the results of this might make variable quantisation more attractive.

4.3.6 Neural network equivalent to stochastic model

This leads us to possible structures of a neural network to implement a stochastic transition approach. The model naturally follows a multilayer feed forward structure. The input layer would consist of a window of previous

values of the time series. The second layer would be a classification layer, classifying the input vector into a state, or if points were allowed to partially belong to several states, into a weighted set of states. The third layer consists of a set of conditional probability density functions, each of which is associated with a state and is selected when that associated state is activated in the second layer. The final layer simply either selects the most likely state from the probability density function or calculates the mean value of the probability density function.

This gives the structure of a network for one step ahead prediction. It is to be noted that if radial basis functions are used for the classification layer and the density function and output layer are combined (as is possible in the simple case) then a radial basis function network is obtained.

This basic structure can be altered to accommodate the inclusion of the two and three step ahead density functions.

Tino [35] describes transforming recurrent neural networks to stochastic automata, thus recurrent networks may be a viable extension of the network described.

4.4 Comparison study

4.4.1 Overview

Various aspects of the equivalent stochastic system models were investigated including

- the effect of the number of quantisation levels
- the effect of the number of samples in the time series modelled
- the effect of predicting multiple steps ahead
- the effect of increasing the embedding dimension on prediction accuracy

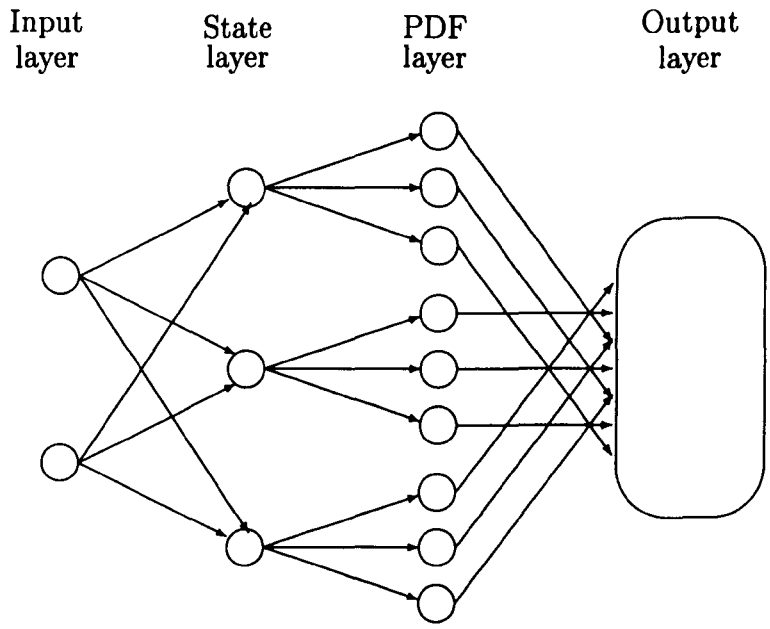

Figure 4.2: Neural network structure of equivalent stochastic model

- the effect of increasing the embedding dimension on the number of hypercubes required to contain the attractor
- the models performance using different data sets
- the characteristics of the different models

In order to compare the performance of the stochastic models to more conventional techniques the various data sets were also predicted using three common techniques - a linear autoregressive model, a feed forward perceptron neural network, and a time-delay neural network.

The computation was performed on a Pentium 90 PC using programs written in C++. For the linear autoregressive model Matlab was used. Graphs of the resulting predictions are shown with the actual time series as a solid line and the time series estimated by the model as a dotted line.

When fitting a model to a time series there are a number of considerations to be taken into account.

Generalisation

A model is said to generalise well when the input-output relationship computed by the model is correct (or nearly so) for data never used in creating or training the model.

The fewer adjustable parameters a model has the more likely it is to generalise well, however it must have sufficient adjustable parameters to adequately model the system.

Overtraining

When a model learns too many specific input-output relationships (i.e. it is overtrained), the model may memorise the training data and therefore be less able to generalise between similar input-output patterns. Deco & Schurmann [10] identify a specific type of overtraining which they call dynamic overtraining where the model overtrains in such a way as to cease to model the dynamic invariants of the system.

4.4.2 Validation criteria

It is required to assess the ability of a particular model to generalise a system. A standard procedure for model generation is to partition the data into two sets. The first larger set is the training set. This set is used to generate the model. The performance of the model is then checked using the second

set, the validation set. The quality of the model will be judged based on its performance with this data set, rather than with the training data.

An important question is how the performance of the model is measured. This is of particular important when the performance measure is used in the model generation as is generally the case.

A common measure is the square root of the mean square (rms) error, or equivalently when comparing performance on different time series the average relative variance (arv). However, the one step ahead prediction performance is not always the best measure for a dynamic system.

The root mean square error is given by equation (4.3.18) and the average relative variance is given by:

$$e_{\text{arv}} = \frac{1}{\sigma^2 N} \sum_{t=0}^{N} (x_t - \hat{x}_t)^2 \tag{4.4.1}$$

It should be noted that a trivial predictor which simply chose the mean value of the time series would give an arv of 1. Thus any model with arv close to 1 is performing extremely poorly.

In order to assess model performance for recursive prediction some form of performance measure is required. One possible measure is the 'long term error' ELT as defined by [10]

$$ELT = \frac{1}{N} \sum_{t=0}^{N} \frac{1}{5} \sum_{i}^{5} [\hat{x}(t+i) - x(n+i)]^2 \tag{4.4.2}$$

which is an extension of the root mean square error frequently used for one step ahead prediction. Due to the difficulties of long term forecasting of chaotic time series reconstruction in the work done N was limited to 5 steps ahead.

Multiple step ahead prediction

When modelling dynamic systems predicting further ahead than a single time step is often of interest. There are several approaches to multi-step prediction. The simplest is to build a one step ahead predictor and then iteratively

predict one step ahead multiple times. A second approach is to build a model to predict the number of steps ahead as required. A third approach is to produce a model that will simultaneously predict all the required future steps.

However, one step ahead prediction is not necessarily the best indicator of model performance. The performance of good and bad models of slowly varying time series with significant observation noise can often appear similar when measured using purely a one step ahead prediction criteria. A tougher test of dependent ability is to perform system reconstruction, predicting a time series several steps ahead using the one step ahead model recursively. This is a particularly difficult task for chaotic time series as the exponential rate of trajectory divergence in state space makes the system extremely sensitive to initial conditions. If a model can predict several steps ahead in this manner then it is probably a good model. However, if a model cannot predict a certain distance ahead this does not necessarily mean that it is a poor model. This is only the case if another model can produce better results. The predictability of a time series is dependent upon the characteristics of that series such as the Lyapunov exponent. It was shown in an earlier example that limited knowledge of initial conditions leads to a prediction horizon beyond which one cannot make accurate predictions regardless of how accurate the model of the system may be.

Given the one-step ahead predictor

$$\hat{x}(n) = F(x(n-1), x(n-2), ..., x(n-d)) \tag{4.4.3}$$

the equivalent recursive predictor is

$$\hat{x}(n) = F(\hat{x}(n-1), \hat{x}(n-2), ..., \hat{x}(n-d)) \tag{4.4.4}$$

The question is which method is best in practice. It has been found that for local linear models that an iterative approach gives the best results. Work conducted with neural networks indicated that multilayer perceptrons had problems learning when used as multistep ahead predictors. The reason for this is

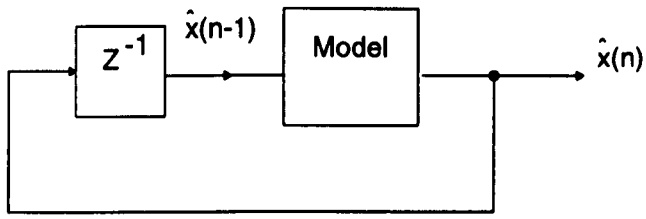

Figure 4.3: System reconstruction model

that the n-step ahead function (the one step ahead function performed n times) is more complex than the one step ahead function and correspondingly harder to learn. However, this objection does not hold for the equivalent stochastic model approach as the model is based on the density of states rather than attempting to model a function. It thus remains to be seen which is the better approach for this type of model.

This method has the advantage over just measuring the one-step ahead prediction accuracy that it shows whether the system is really being modelled properly. For example a trivial one-step ahead prediction algorithm is to predict the value of the previous sample as the next value. With many time series (particularly if the series is slowly varying) this may give a reasonable one-step ahead prediction error, however when used for recursive modelling the modelled output would rapidly diverge from the real time series as the true dynamics of the system are not being modelled.

The divergence for linear time series can be relatively slow provided that the linear model used is reasonably accurate. Chaotic systems however are highly sensitive to initial conditions, with nearby trajectories displaying exponential divergence, this rate of divergence being measured by the systems Lyapunov values. This means that highly chaotic systems which are recursively modelled diverge extremely rapidly from the real time series with even small errors in the model used.

4.4.3 Time series

Four different time series were utilised to compare the predictive properties of the various models. Two of the time series were generated by computer using difference equations and two were taken from experimental data. There are limitations in using computer generated data for the simulation of chaotic systems. Since computer data is stored to a finite precision any computer generated data must eventually repeat itself for a sufficiently long time series. It was found that 16 bit precision was inadequate for the simulation of the Logistic mapping for lengths greater than 3000 samples since it would reach the unstable fixed point of 0 and stay there. 32 bit precision was found to be adequate for the lengths utilised.

The Logistic mapping

The logistic mapping is a discrete time equation that takes its name from the corresponding differential equation

$$\frac{dx}{dt} = \mu x(1-x) \tag{4.4.5}$$

originally used by P.F. Verhulst in 1845 to model population growth in a limited environment (May [38]). It is a one dimensional time series generated in an iterative manner from the algorithm:

$$X_t = aX_{t-1}(1-X_{t-1}) \tag{4.4.6}$$

where X_t is the t^{th} member of the series and a is a constant. For $0 \leq a \leq 4$ this defines a map from the unit interval onto itself. As the value of a is increased from zero the function produces a series of limit cycles with a series of period doublings as a is increased until the period becomes infinite and the limit cycle is a strange attractor. The value of a used for the time series utilised was $a = 4$.

This mapping has an invariant measure ρ with density:

$$g(x) = \frac{1}{\pi\sqrt{x(1-x)}} \tag{4.4.7}$$

and the Lyapunov exponent of equation (4.4.6) is $\lambda = \ln 2$. It has the solution

$$x_n = \sin^2(2^n \pi I) \qquad 0 \leq x \leq 1 \tag{4.4.8}$$

with I given by equation (3.6.76). It has unstable fixed points at 0 and 0.75 as solutions. If I is rational then the solution is an unstable periodic orbit. If it is irrational then the orbits are aperiodic. Since there are infinitely more irrational numbers than rational numbers in general the solution will be aperiodic.

A simple one to one transformation will convert the Logistic mapping to the tent mapping

$$x_{n+1} = 2x_n \qquad 0 < x < \frac{1}{2} \tag{4.4.9}$$

$$x_{n+1} = 2 - 2x_n \qquad \frac{1}{2} < x < 1 \tag{4.4.10}$$

which thus has equivalent dynamic behaviour and is easy to analyse.

The logistic mapping was chosen as it a simple, easily understood one dimensional mapping which has been used by numerous researchers for the study of chaotic time series making comparisons easy.

The Henon map

The Henon map is a two dimensional time series generated in an iterative manner using the algorithm:

$$X_t = 1 + Y_{t-1} - aX_{t-1}^2 \tag{4.4.11}$$

$$Y_t = bX_{t-1} \tag{4.4.12}$$

or equivalently:

$$X_t = 1 + bX_{t-2} - aX_{t-1}^2 \tag{4.4.13}$$

where a and b are constants. The values used were $a = 1.4$ and $b = 0.3$. This produces a chaotic time series with an invariant interval of approximately [-1.3,1.3].

It is diffeomorphic (a uniquely reversible, continuously differentiable map). It has Lyapunov exponents of $\sigma_1 = 0.42$ and $\sigma_2 = -1.62$

Similarly to the Logistic mapping, it has been used by several researchers into chaotic time series and so allows comparison of results. Also it is two dimensional allowing the effect of increasing dimension to be investigated.

Wolf's sunspot series

Data was available for Wolf's sunspot series from 1944 to 1988 in the form of a monthly mean sunspot number (a total of 300 data points). This is another popular series for study, but this time a naturally occurring rather than an artificially generated series. The natural mechanism for the generation of the sunspots is not well understood and the series is thought to contain a fair amount of noise. Although linear methods have been used in an attempt to predict it nonlinear methods have proved to be more successful in practice implying a degree of nonlinearity in the series. This series was included for an appreciation of the problems involved with natural time series, such as observation and process noise, unknown dimensionality, limited data, etc, as opposed to the idealised artificially generated series.

NH_3 laser data

One model of the laser is the set of differential equations known as the Maxwell-Bloch equations (Baker & Gollub [13]). These describe the time dependence of the electric field, mean polarization and population inversion. The form of these equations is similar to the Lorenz model for chaotic convection. Many practical lasers do not operate within a parameter range where chaos

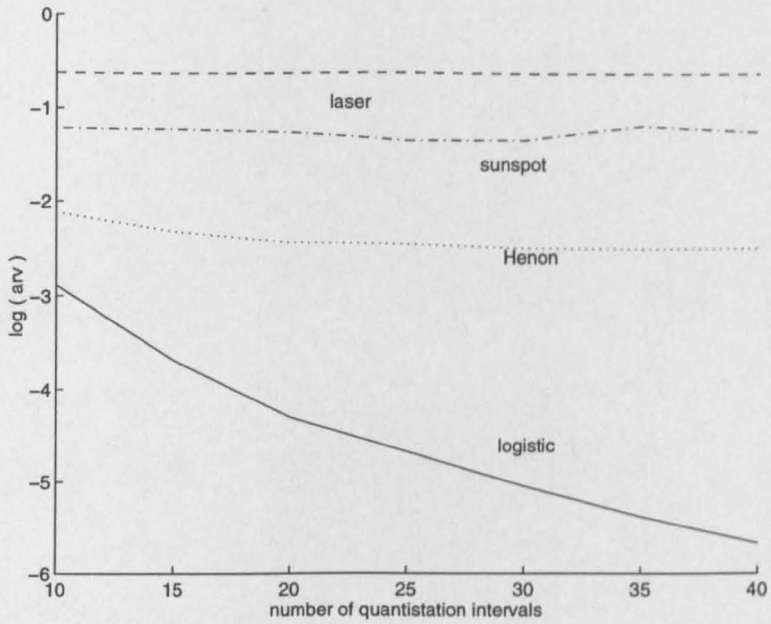

Figure 4.4: Effect of increasing quantisation intervals on prediction of logistic mapping

occurs, however chaotic behavior may be realised when the laser configuration is modified by tuning the cavity length, varying the laser gain or tilting one of the mirrors.

In 1992 the Santa Fe time series competition was held. One of the time series used was that of a experimentally derived series of the chaotic intensity pulsations of a NH_3 laser. The data was given as an 8 bit number and thus quantized into 256 intervals. The fractal dimension of the series was found by Huber & Weiess [45] to be just over 3.

4.4.4 One dimensional Markov chain model

Number of quantisation intervals

The one dimensional Markov chain model was used to predict the different time series, for example Figure 4.5. The number of quantisation intervals used

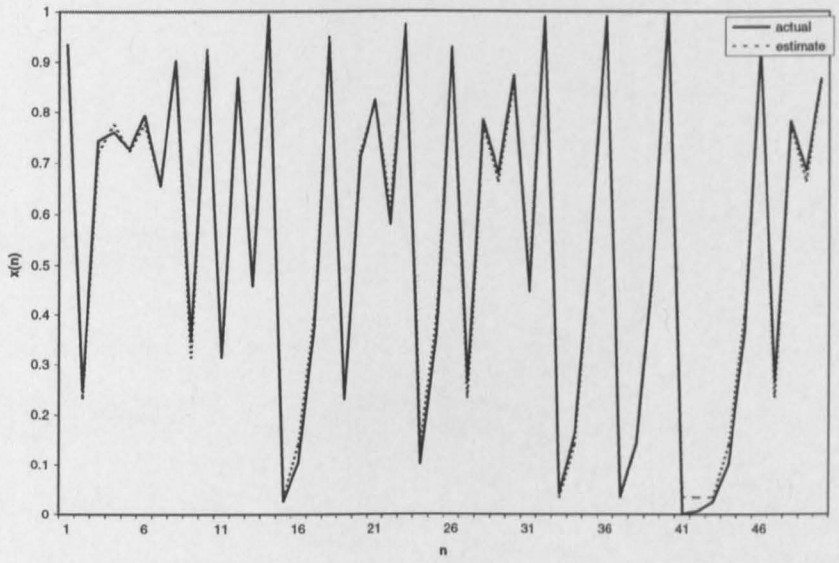

Figure 4.5: Logistic map predicted with 1D Markov chain

in each model was varied. The results are shown in Figure 4.4 giving the log of the average relative variance of one step ahead prediction. It is known that the average divergence $\delta_{n+1} = \delta_n e^{-\lambda}$ where λ is the largest Lyapunov exponent. Thus the divergence is proportional to the initial deviation, which for the Markov chain is the size of the quantisation interval. It is to be expected from this that the average prediction error should be inversely proportional to the number of intervals leading to a value of the variance of the error that is inversely proportional to the square of the number of quantisation intervals. This is found to be the case for the Logistic mapping, but not for the Henon, laser, or sunspot series. After an initial decrease the prediction error for the model of the Henon map reaches a plateau. This is due to modelling the two dimensional Henon map as a one dimensional process. This introduces process

j	1	2	3	4
Logistic mapping	0.0122	0.0808	0.3063	0.4682
Henon map	0.0861	0.3008	0.3712	0.5892
Sunspot series	0.0834	0.0999	0.1096	0.1408

Table 4.1: Effect of prediction interval on accuracy

noise representing that part of the system inadequately modelled and this noise forms a limit on the accuracy of prediction. The effect of modelling the laser series as one dimensional has the same effect.

The sunspot series actually shows an increase in error as the number of quantisation intervals increases. This is due to insufficient data. As the number of states increases there are on average less samples per state. If there are insufficient samples to form a reasonable estimate of the density function the performance of the model will decline.

Probability density distributions

In general the shape of the density distributions become wider as the step interval increases, as would be expected since position in state space is becoming less certain as time progresses. In a few specific cases however the distributions become sharper and the position in state space better defined. It was noted that if following a time series the j step ahead and the $j+1$ step ahead distributions sometimes overlapped, rather than the $j+1$ distribution completely containing the j distribution. This implies that combining the two would lead to a sharper and thus better defined distribution, providing a justification for the combined density function models.

N	1	2	3	4
Logistic mapping	0.0122	0.0247	0.0524	0.0870
Henon map	0.0861	0.0963	0.1038	0.1433
Sunspot series	0.0834	0.0689	0.0680	0.0745

Table 4.2: Equal weight model

Multiple step ahead prediction

It is known that the average divergence is exponential with increasing prediction period. This rate of divergence is only valid for a small initial difference δ and a number of time steps into the future j. The divergence of trajectories is bounded by the invariant interval of the attractor and this provides a limit for the error at large j.

Table 4.1 demonstrates this effect giving the error in terms of e_{arv} for 1D Markov chain models with 20 quantisation intervals predicting j time steps into the future. The two chaotic time series follow this expected pattern of rapid increase in error followed by tending towards a limit. The sunspot series however displays a more linear increase implying that it may have a more linear nature.

4.4.5 Equal weight model

The equal weight model consisting of summing the density functions of N one dimensional Markov chain models of varying prediction length and 20 quantisation intervals was used to predict three different time series. The results are shown in Table 4.2 giving the average relative variance of one step ahead prediction. The idea behind the model is to reduce the effects of noise

N	1	2	3	4
Logistic mapping	0.0122	0.0157	0.0207	0.0256
Henon map	0.0861	0.0713	0.0697	0.0738
Sunspot series	0.0834	0.0687	0.0661	0.0667

Table 4.3: Exponential weight model, n=0.5

(quantisation, process and observation).

$$P(x_{n+1} \in Q_i | x_n, x_{n-1}, ..., x_{n-N}) = \frac{1}{N} \sum_{1}^{N} P(x_{n+1} \in Q_i | x_{n+1-N}) \qquad (4.4.14)$$

where Q_i is the i^{th} quantisation interval. The results for the two computer generated series show a decrease in performance for increasing N. This is most marked for the logistic mapping, due to the exponential increase in prediction error for increasing prediction interval. This leads to the inaccurate predictions from the higher j-step ahead models drowning out the more accurate prediction from the one step ahead model. The effect is less marked for the Henon map, possibly due to the presence of process noise. The model actually gives an improvement for the sunspot series for medium values of N. It should be noted that previous results show that the sunspot series does not display a rapid increase in prediction error for increasing time interval so the effects of noise reduction give an improvement.

In general the performance of the equal weight model is poor giving benefit only for high noise processes.

Exponentially weighted model

The exponentially weighted model uses a weighting factor of $\frac{1}{n^j}$ where n is a constant and j the prediction time interval of the density function being weighted. The model was compared using two different values of n.

N	1	2	3	4
Logistic mapping	0.0122	0.0112	0.0111	0.0111
Henon map	0.0861	0.0735	0.0728	0.0727
Sunspot series	0.0834	0.0774	0.0770	0.0769

Table 4.4: Exponential weight model, n=0.1

N	1	2	3	4
Logistic mapping	0.0173	0.0133	0.0092	0.0098
Henon map	0.144	0.069	0.070	0.070
Sunspot series	0.143	0.137	0.149	0.149

Table 4.5: Independence model

The performance modelling the logistic mapping as shown in Tables 4.3 and 4.4 declined with increasing N for $n = 0.5$ and improved slightly for $n = 0.1$. With the higher value of n the exponential divergence of trajectories outweighs any noise reduction effect. The lower value of n leads to a slight improvement in prediction due to reduction in quantisation noise.

Although the exponential weight models lead to a slight improvement in performance overall the effect was marginal. Since the Henon map suffers from process noise both values of n give improved results for increasing N, while the sunspot series shows a slight improvement.

Distribution combinatorial models

Both the assumed independence model based on equation (4.3.10) and the $\frac{1}{N}$ model based on equation (4.3.13) with 20 quantisation intervals give similar results. Both models give initial improvement with increasing N before reach-

N	1	2	3	4
Logistic mapping	0.0173	0.0128	0.0097	0.0098
Henon map	0.144	0.071	0.070	0.070
Sunspot series	0.152	0.131	0.126	0.130

Table 4.6: $\frac{1}{N}$ model

ing a performance limit. This limit is due to density functions for the larger prediction intervals j in equations (4.3.10) and (4.3.13) being in general less well defined (spread over a larger range). They therefore have less influence than the density functions for the shorter prediction intervals. Increasing N thus has an increasing small influence on the model results. The two models give better results than any of the weighted sum models for the computer generated series demonstrating an ability to reduce process and quantisation noise. The $\frac{1}{N}$ model gives slightly better results than the assumed independence model.

Overall, although it is possible to gain a small improvement in the performance of the one dimensional Markov chain model by combining one dimensional models of increasing prediction interval, in order to improve results significantly another approach is required.

4.4.6 Multiple dimensional models

Embedding dimension

The number of states required to model the three time series for increasing embedding dimension when the invariant interval is equally partitioned into 20 intervals is shown in Table 4.7. It can be seen that for the logistic and Henon mappings the number of states required approximately doubles each time the dimension is increased demonstrating an exponential increase with dimension as expected. The sunspot series follows a similar pattern, but only contains

N	1	2	3	4
Logistic mapping	20	56	128	272
Henon map	20	97	203	367
Sunspot series	19	94	189	239

Table 4.7: Effect of dimension on the required number of states

j	1	2	3	4
Logistic mapping	0.01219	0.00846	0.00736	0.00722
Henon map	0.08607	0.00780	0.00550	0.00488
Sunspot series	0.08344	0.18014	0.85997	1.30232

Table 4.8: Multi-dimensional model

300 samples and so as the dimension increases the true number required is probably larger than that given in Table 4.7. It can be seen that any sunspot model with an embedding dimension of 4 will have less than two samples per state and thus the transitional density functions learned will be a very poor estimate.

Prediction

The general effect of increasing the embedding dimension j on the computer generated time series is an increase in accuracy as shown in Table 4.8. In the case of the logistic mapping this is because the larger number of states is equivalent to using more quantisation intervals. The effect however tails off for increasing dimension and it should be noted that greater improvement in accuracy would be given by using the same number of states in a one dimensional model. The prediction accuracy of the Henon map improves by an order

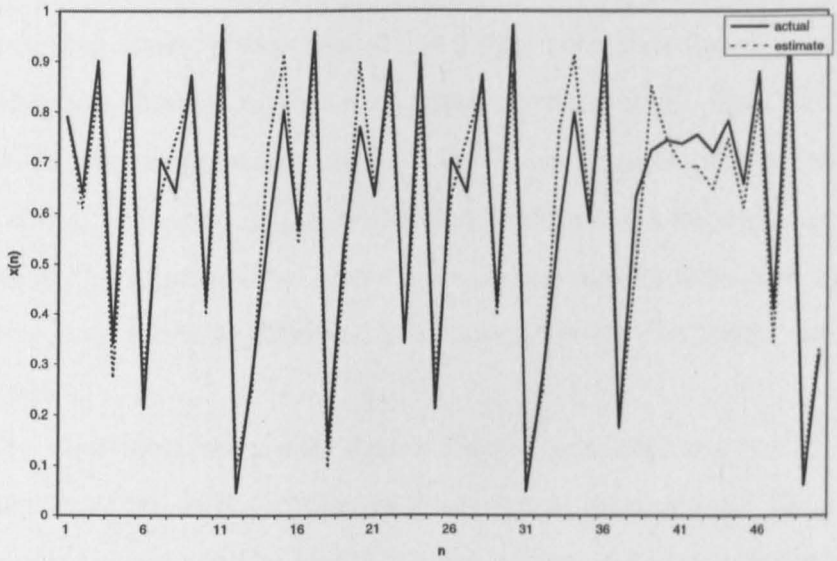

Figure 4.6: Henon map predicted with 1D Markov chain

of magnitude from when embedded to a system dimension of 1 (as shown in Figure 4.6) to when it is embedded to the system dimension of 2 (as shown in Figure 4.7). This is due to the removal of the process noise due to the Henon map being a two dimensional process which was present in the one dimensional model.

The sunspot series prediction accuracy actually decreases for larger dimension. The results are distorted by the short length of the data series which means that at higher dimensions there are few samples per state from which to estimate density functions. Indeed the series does not cover the entire state space and during prediction states are entered which are not modelled. In this case the prediction program assigns the state a default density function. This has the consequence that if many new states are encountered the accuracy of

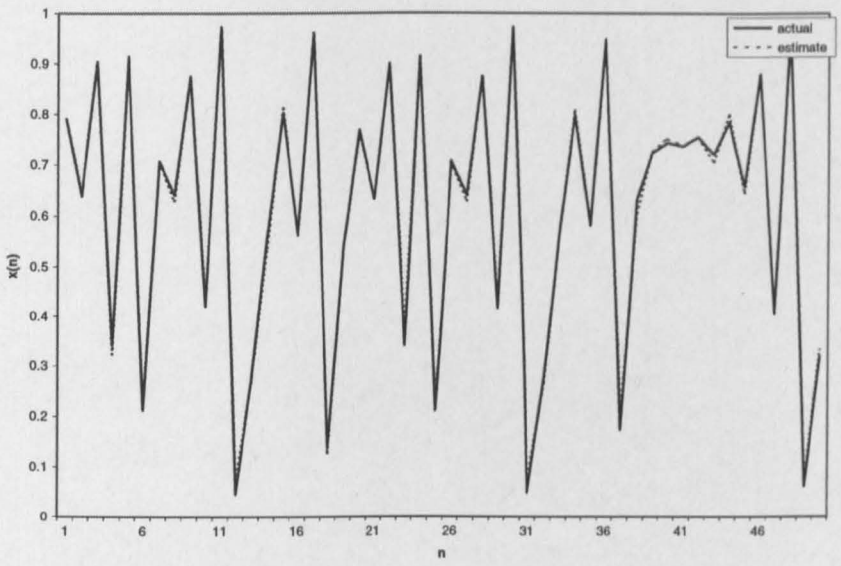

Figure 4.7: Henon map predicted with 2D Markov chain

the model is poor and the situation deteriorates for larger dimensions. The results for the four dimensional model are exceedingly poor, to the extent that a model which simply assigned the mean value of the time series as its prediction would give better results.

4.5 Multiple step ahead and recursive prediction

Once a model of a system has been constructed and parameter values determined the model needs to be tested in order to determine how well it models the target system. In the work undertaken so far the root mean square value of the one-step ahead prediction error has been used in this performance evaluation

process. An alternative model evaluation process is by performing recursive prediction. A prediction of the time series is computed iteratively by feeding the one-step ahead predictions of the model back into the input so that after initialisation there is no external input to the model. Thus the recursively modelled time series starts the from the same initial conditions as the actual time series, but errors in the model lead to differences between the recursively modelled (or reconstructed) time series which accumulate over time causing the two time series to diverge. The more accurate the model the slower the divergence.

The equivalent stochastic system model was implemented in a recursive manner in order to determine its effectiveness as a model. Its performance was poor, but due to the highly chaotic nature of the systems modelled it is difficult to determine how much was due to the nature of the systems and how much due to the inadequacies of the model. In order to afford a comparison the capabilities of temporal processing neural networks to model the same time series were investigated since such neural networks have been used to effectively model certain chaotic time series in the past.

4.5.1 Equivalent stochastic system model

The equivalent stochastic system model was used to perform recursive prediction (system reconstruction) on the logistic (Figure 4.8) and Henon mappings (Figure 4.9). Although long term prediction was not expected due to the chaotic nature of the time series, it was hoped that the reconstructed system would display similar properties to the original time series. In both cases the predictions rapidly diverge from the actual time series and the predictions cease to be meaningful after 4 or 5 iterations. This is due to the sensitivity to initial conditions characteristic of chaotic series. This is possibly made worse by the effect of the quantisation noise inherent in the model utilised.

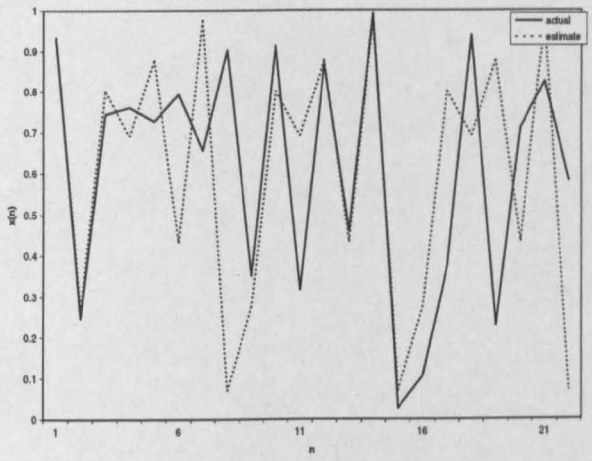

Figure 4.8: System reconstruction of Logistic mapping using stochastic model, d=1

The main feature of the reconstructed systems is periodicity. This strongly contrasts with the real time series which, since they are chaotic, are aperiodic. The form of equivalent stochastic system model utilised can be described as an n state Markov chain with an $n \times n$ state transition matrix:

$$\mathbf{P} = \begin{bmatrix} p_{00} & \cdots & p_{0n} \\ \vdots & & \vdots \\ p_{n0} & \cdots & p_{nn} \end{bmatrix} \tag{4.5.1}$$

However when the one step ahead prediction is made only one state is selected. If the initial state is state q, and the state predicted is state m then the Markov chain model is altered so that

$$p_{qr} = \begin{cases} 1 & r = m \\ 0 & r \neq m \end{cases} \tag{4.5.2}$$

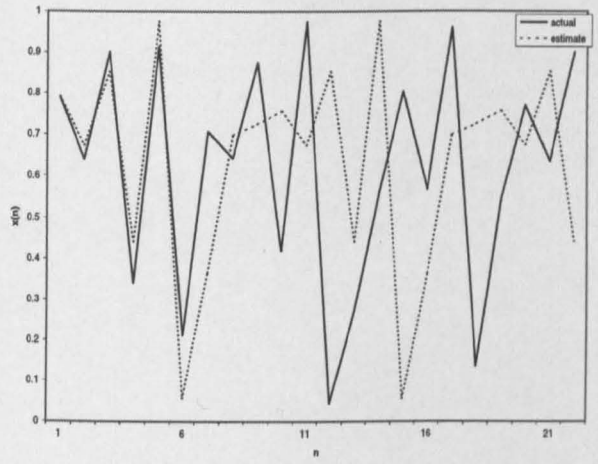

Figure 4.9: System reconstruction of Henon mapping using stochastic model, d=2

This gives a state transition matrix consisting of only 0s and 1s and which is thus deterministic. The states can be divided into one or more irreducible classes, plus possibly some transient classes. Depending on the initial conditions the system will enter (possibly via a transient class) an irreducible class and then remain in that class following a periodic path equal in length to the number of states in that class. The periodicity evident in the reconstructed systems is thus due to the change from a stochastic model to a deterministic model when the weighted mean of states, or most probable state is chosen as the predicted state.

The stochastic model uses the conditional probability density function $P(x_n|x_{n-1}, ..., x_{n-d})$, where d is the embedding dimension, as the basis of its prediction. The initial system reconstruction was performed using the dimension of the process as the embedding dimension, $d = 1$ for the logistic mapping, $d = 2$ for the Henon mapping. The accuracy of the stochastic model used for one step ahead prediction improved slightly for higher values of d although with

problems such as higher computational, memory requirements and requiring more training data. System reconstruction for higher values of d were implemented for both Logistic and Henon mappings, but the results were on average only marginally better than those with low embedding dimensions. Periodicity was still evident in the reconstructed systems but tended to be of greater length. Occasionally after the real and reconstructed time series had diverged they would after an amount of time overlap once more and the reconstructed system would follow the real system for a few iterations before diverging once more. A good example of this is in the graph of the reconstructed Henon map with an embedding dimension of 6, Figure 4.10. This demonstrates that the stochastic model does at least partially reflect the modelled time series.

Increasing the number of quantisation levels which in one step ahead prediction led to reduced prediction errors was also investigated, but it gave no significant improvement in the results.

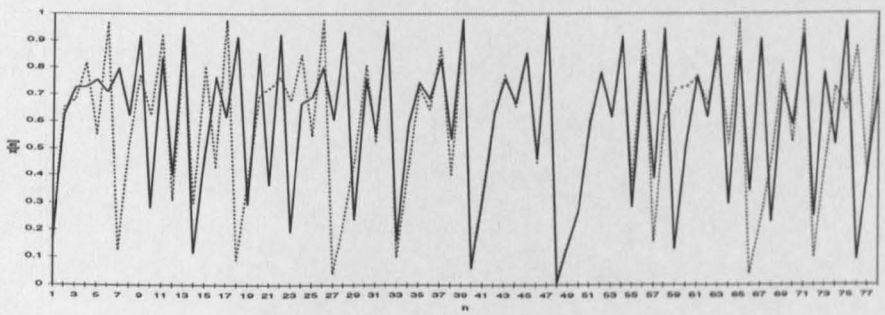

Figure 4.10: System reconstruction of Henon map using stochastic model, d=6

Since the periodicity of the stochastic model is due to the choice of the most likely state, the model was implemented with random state transitions, with the state transition probabilities taken from the density function. This unsurprisingly gives worse results for one step ahead prediction. When it is used for system reconstruction rapid divergence occurs as with the other models, however the system is no longer periodic, and the system looks similar in form to the original series once quantisation effects have been taken into account.

In the stochastic model if the system is in one state then the next state to be predicted will always be the same leading to the periodic pattern on reconstruction. In an attempt to add information to the prediction process it was thought that using the gradient $\dot{x}(n)$ might improve matters. Since the gradient itself forms a chaotic time series the stochastic model can also be used to predict future values of $\dot{x}(n)$ based on previous values. This was implemented for the Logistic mapping shown in Figure 4.11. The gradient was approximated as

$$\dot{x}(n) = x(n) - x(n-1) \tag{4.5.3}$$

The root mean squared one step ahead prediction error for the gradient was just under twice the error obtained using the stochastic model on the normal time series, although the average relative variance was slightly better. The reason for this is that although the Logistic mapping varies from 0 to 1, its gradient varies from -1 to 1, a range twice as large.

Predicting the gradient based on the original time series was also tried. This gave an root mean square error similar to that for the original series as might be expected.

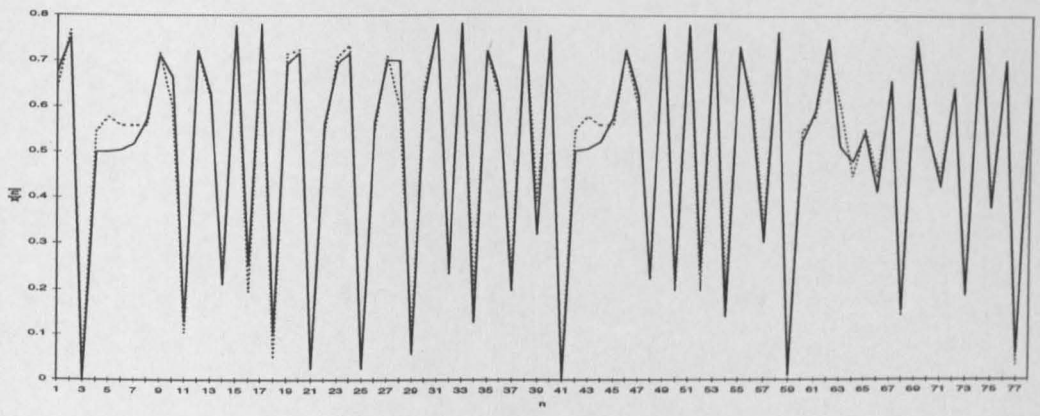

Figure 4.11: One step ahead gradient of Logistic mapping using stochastic model, d=2

One step ahead prediction of the series using both the time series and the gradient of the time series was then implemented (Figure 4.12) using the probability density function

$$P(x_n, \dot{x}_n | x_{n-1}, ..., x_{n-d}, \dot{x}_{n-1}, ..., \dot{x}_{n-d}) \tag{4.5.4}$$

This gave similar results to an transition stochastic model not utilising the gradient. The model was then used for system reconstruction, but gave similar results to previous models, diverging after a few iterations and then behaving periodically.

4.5.2 Linear autoregressive model

The system identification toolbox addition to the Matlab programming language contains routines for parameter estimation and time series prediction of linear models [23]. These were used to create linear autoregressive models

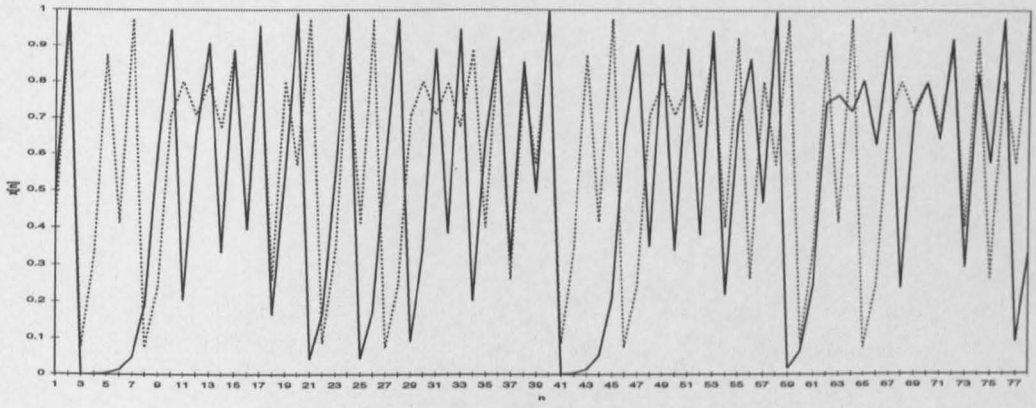

Figure 4.12: System reconstruction of Logistic mapping using stochastic model, $\dot{x}(n)$ and $\dot{x}(n)$, d=2

of the form:

$$x(n) = \alpha_0 + \sum_{k=1}^{m} \alpha_k x(n-k) \tag{4.5.5}$$

The predictive performance of these global linear models of both the Logistic and Henon mappings were poor with mean squared errors of about 0.3, for an arv of approximately 0.72 for the Logistic mapping and worse for the Henon map. Due to this poor performance they were not tried with the sunspot series, although since that series displays a fair degree of linearity it would be expected that the model would be more successful. As expected the linear model does not perform well when used to model a chaotic system when compared to the equivalent stochastic model or other nonlinear approaches and therefore this model is not suitable for this study.

4.5.3 Feedforward perceptron network

A multilayer feedforward perceptron neural network consists of a number of local computational units known as neurons. These are arranged in a series

of layers with the neurons in a layer being connected to the neurons in the next layer.

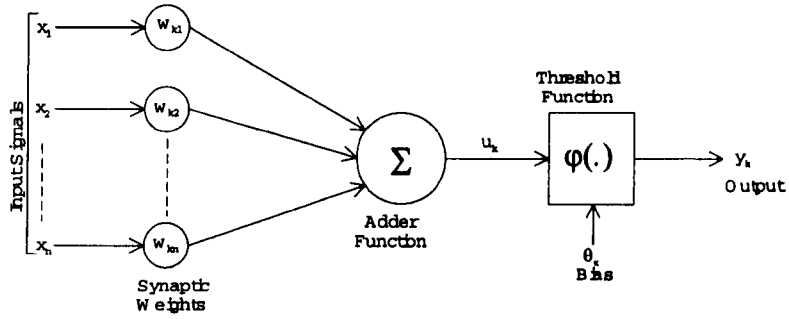

Figure 4.13: An artificial neuron

A neuron consists of a number of inputs known as synapses which are individually weighted and then summed together as shown in Figure 4.13. They are then transformed by a squashing function such the hyperbolic tangent function, thus creating the output of the neuron which is used as the input to the synapses of the neurons in succeeding layers.

The network consists of an input layer, one or more hidden layers followed by an output layer as shown in Figure 4.14. The network is trained by means of the backpropagation algorithm. This is a form of supervised learning. A known input vector is presented to the network and propagated through all the neurons until an output vector is produced at the output layer. This output vector is then compared with a known desired output for that input generating an error signal, the difference between the output and desired output. This error is then propagated back through the network and the synaptic weights are adjusted in accordance to an error correction rule.

Input-Output training pairs of vectors are presented to the network successively, and the weights updated. When all the training data has been presented to the network the process is repeated a second time starting with the first training pair. This continues until the error performance of the network

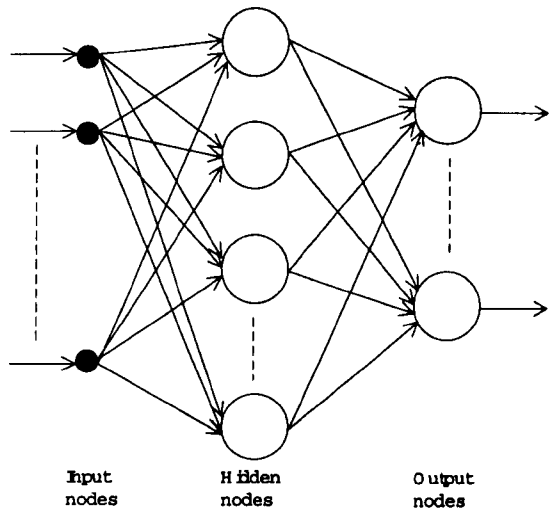

Figure 4.14: A feedforward neural network with one hidden Layer

reaches a satisfactory level, or the error ceases to improve significantly.

Before training starts the weights in the network are initialised to small random values. These values are small in order to avoid saturating the neurons which leads to long training times. This randomised initialisation means the backpropagation algorithm is a stochastic gradient descent method. Each time the network is trained on the same data, the weights start at different values, so the time taken to train may vary. Also since the algorithm follows the gradient of the error surface in weight space it is possible to get stuck in a local minima of the surface, rather than find the global minima. Since the route taken depends on the random starting point in weight space then, in the presence of local minima in the error surface, the network may give different results each time it is trained.

When the backpropagation network is utilised for time series prediction it is being trained to act as a function estimator estimating the function in the equation

$$x(n) = F(x(n-1), x(n-2), ...x(n-M)) \tag{4.5.6}$$

It is wished for the network to generalise the function well from the data that it is trained on. This is in order that it will give a good estimate of the function when used with data on which it has not been trained. In other words the network is required to perform a good nonlinear interpolation of the function. Just because a network can accurately model a function for the data on which it has been trained does not mean that it will generalise well. This can be due to overtraining. A large neural network may be merely memorising the input-output mappings rather than smoothly interpolating between them.

One of the most important factors which determines a networks performance is its architecture. There exists a universal approximation theorem for nonlinear multilayer feedforward networks which states that a network with a single hidden layer can approximate any continuous bounded function providing it has sufficient neurons in the hidden layer. This suggests that a simple network with a single large hidden layer is all that is required. However, although it is known that this architecture is capable of approximating the function, the difficulty is in training it to do so. The problem with a single hidden layer is that the neurons tend to interact with each other globally so it is difficult to improve on an approximation at one point without worsening it at another. Using two hidden layers the interactions are less global, and training can be easier. Another problem is the complexity of the task. As a task becomes more complex then the time required to train a network tends to increase exponentially.

The number of neurons used in each layer is also important. Too few neurons and the network will not be able to approximate a complex function well. Too many neurons and there is a danger of overtraining.

4.5.4 FIR multilayer perceptron network

In 1992 the Santa Fe time series competition was held. One of the time series used was that of the chaotic intensity pulsations of a NH_3 laser. One thousand samples were supplied and the next 100 had to be predicted. The winner was a Finite Impulse Response (FIR) multilayer perceptron network. This is an extension of the standard backpropagation network using finite FIR filters as the neuron synapses. In view of the proven ability of this type of network to model a chaotic time series it was implemented in C++ in order to provide a comparison with the equivalent stochastic system model.

FIR multilayer perceptron networks are feedforward networks similar to standard backpropagation networks except that the synapses of the artificial neurons are modelling by FIR filters. This allows them to model temporal as well as spatial patterns.

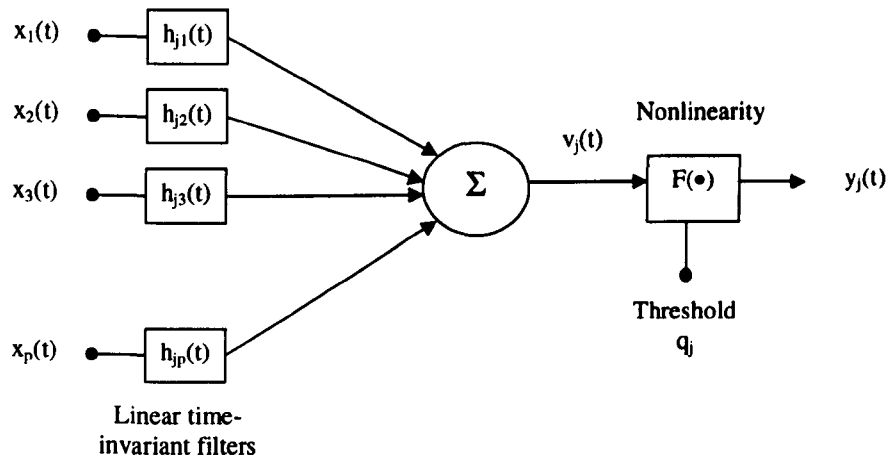

Figure 4.15: Dynamic model of a neuron using FIR filters as synapses

The network can be represented as a number of vectors.

Defining:

$$\mathbf{x}_i(n) = [x_i(n), x_i(n-1), ..., x_i(n-M)]^T \tag{4.5.7}$$

$$\mathbf{w}_{ji} = [w_{ji}(0), w_{ji}(1), ..., w_{ji}(M)]^T \tag{4.5.8}$$

where M is the order of the FIR filters, then the output of neuron j is given by $y_j(n)$:

$$v_j(n) = \sum_{i=0}^{p} \mathbf{w}_{ji}^T \mathbf{x}_i(n) \tag{4.5.9}$$

$$y_j(n) = \varphi(v_j(n)) \tag{4.5.10}$$

In order to train the network a variant on the standard back propagation algorithm is used. After a forward pass, the time delayed errors are propagated back through the network in a manner analogous to standard backpropagation. Defining:

$$\Delta_m(n-M) = [\delta_m(n-M), \delta_m(n+1-M), ..., \delta_m(n)]^T \tag{4.5.11}$$

then for neuron j in the output layer the weight updating equations are

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \eta\delta_j(n)\mathbf{x}_i(n) \tag{4.5.12}$$

$$\delta_j(n) = e_j(n)\varphi_j'(n) \tag{4.5.13}$$

where e_j is the difference between actual and target outputs of output neuron j, and for neuron j in a hidden layer

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \eta\delta_j(n-lM)\mathbf{x}_i(n-lM) \tag{4.5.14}$$

$$\delta_j(n-lM) = \varphi_j'(v_j(n-lM)) \sum_{m \in A} \Delta_m^T(n-lM)\mathbf{w}_{mj} \tag{4.5.15}$$

As can be seen these update equations are analogous for to those for the standard backpropagation algorithm. Indeed if the memory $M = 0$ then the network is a standard multilayer feedforward network trained by the backpropagation algorithm. It is actually possible to form an feedforward backpropagation network by unfolding the temporal network in time. This leads to a larger network with some of the weights constrained to the same value as other weights.

The FIR neural network program was written in C++ and was run on a Pentium 90 PC. The output was then sent to Excel for display. On all the graphs of system reconstruction, the recursive prediction starts at time step 50. The network used hyperbolic tangent activation functions and the algorithm included a momentum term. There are a number of parameters to be varied. The architecture, η, α, the size of the training set, the size of the test set, the value of the memory M, and the number of epochs trained for. These were all varied to improve performance. The network architectures tried were initially small, single hidden layer structures, and then larger structures and an additional hidden layer were tried. It made sense to start with smaller structures as they tend to converge faster since they are simpler, and each epoch computes faster as there are fewer neurons. If, when the training records were examined, the network was still improving its error performance to a significant degree when it stopped, then it was rerun with a larger number of epochs. If there was evidence of overfitting then the size of the training sample was increased. Sometimes the network failed to train to any significant degree. It would then be rerun several times using randomised initial weights to see if this happened consistently. The errors were given in terms of root mean square one step ahead prediction error and the error for the training set and the test set were compared.

The first data series tried was the logistic mapping as this was the simplest. Initially the values of α and η were varied as shown in Table 4.9. This was

M	α	η	training error	test error
0	0.2	0.1	fail	fail
0	0.01	0.01	0.02026	0.02259
0	0.5	0.01	0.01532	0.01394
0	0.9	0.01	0.01394	0.012904

Table 4.9: 1-4-1 network

with 1600 training samples and 500 epochs. Similar results were obtained with different architectures and memory, so $\alpha = 0.9$ and $\eta = 0.01$ were generally used.

The effect of different architectures was then investigated and the results are given in Table 4.10. The time taken to train varied from 5 minutes for the simplest networks, to over an hour for the larger networks with more training samples. A greater number of training samples was required for the larger networks to avoid over training.

The first thing that is noticeable is that the FIR filter memory M had a neglible effect on the networks ability to model the logistic mapping for the single hidden layer networks. Standard backpropagation networks ($M = 0$) gave similar results, any improvement due to $M > 0$ was small, or in the case of the larger values of M could even lead to a failure of the network to train effectively. This is to be expected as the logistic mapping is a one dimensional function with its next value solely dependent on its current value, thus the extension of the network to take into account previous values would be expected to be of little utility. It is artificial that the smallest structure performs worse than most of the other structures. This is probably due to it having insufficient neurons to properly approximate the function. However the worst performing network is that with the most neurons. This is not due

Structure	M	epochs	samples	training error	test error	reconstruction
1-2-1	0	500	500	0.0196	0.02164	
1-3-1	0	500	500	0.0139	0.01337	
1-3-1	1	500	500	0.0132	0.01268	
1-3-1	2	500	500	0.0137	0.01328	
1-4-1	0	500	500	0.01394	0.01290	
1-4-1	1	500	500	0.01324	0.01280	
1-4-1	2	500	500	0.01509	0.01281	
1-4-1	3	2000	1500	0.01270	0.01248	
1-5-1	0	500	1600	0.01324	0.01311	
1-5-1	1	500	1600	0.01306	0.01299	
1-5-1	2	500	500	0.01462	0.01488	
1-5-1	3	500	1500	0.01306	0.01299	
1-5-1	4	500	1500	fail	fail	
1-8-1	0	500	1500	0.01366	0.01475	
1-10-1	0	2000	1500	0.01219	0.01191	
1-15-1	0	1000	1500	0.02309	0.02640	
1-15-1	1	500	1500	fail	fail	
1-3-3-1	0	100	2500	0.0150	0.0130	0.511
1-3-3-1	1	500	2500	0.00171	0.00165	0.812
1-3-3-1	2	500	2500	0.00206	0.00193	0.824
1-3-3-1	3	100	2500	0.00531	0.00406	0.739

Table 4.10: One step ahead prediction of the logistic mapping

to the networks inability to approximate the function, but of the difficulty in training it to do so. The larger the network the greater its capacity to accurately approximate the function, but the harder it is to train for reasons mentioned earlier. When a second hidden layer is added things change however. Although the network with no memory performs similarly to the single hidden layer networks, when the memory is increased to $M = 1$ then the performance improves by an order of magnitude. Further increasing M however has no positive effect. This does however demonstrate the FIR networks improved capacity to model time series.

The column labeled reconstruction is the root mean square error for the time series predicting up to 100 steps ahead using its previous estimates for its predictions as described earlier. As can be seen the error is large indicating the networks inability to predict that far into the future.

The Henon mapping is two dimensional so it comes as no surprise to note that the networks with no memory are significantly poorer at representing it than those with more memory (Table 4.11) since the $M = 0$ networks are attempting to model a two dimensional process as a one dimensional process, since the extra unmodelled dimension appears as process noise. Increasing the memory so $M > 1$ however leads to little or no improvement as was the case with logistic mapping when the dimension of the mapping was exceeded using single hidden layer networks. As was the case with the logistic function two hidden layers leads to an increase in performance in general but it is not so artificial as previously. Indeed the best single hidden layer network is better than many of the two hidden layer networks. Increasing the number of neurons used improved performance and the best configuration consisted of a large initial hidden layer followed by a second smaller hidden layer.

One of the problems that has been noted with networks with more than one hidden layer is that the network does not tend to train evenly. The layers nearer the output layer tend to train at a higher rate. For better training all

Structure	M	epochs	samples	training error	test error	reconstruction
1-5-1	0	100	2500	0.0811	0.0797	0.3563
1-5-1	1	500	2500	0.0089	0.0086	0.3640
1-5-1	2	500	2500	0.0090	0.0095	0.410
1-5-1	3	200	2500	0.0086	0.0084	0.3333
1-5-1	4	200	2500	0.0095	0.0089	0.3908
1-10-1	0	100	2500	0.0812	0.0802	0.3436
1-10-1	1	500	2500	0.0066	0.0058	0.2624
1-10-1	2	500	2500	0.0096	0.0083	0.3146
1-3-3-1	0	100	2500	0.0817	0.0800	0.3470
1-3-3-1	1	1500	2500	0.00356	0.00281	0.2990
1-3-3-1	2	100	2500	0.00902	0.00942	0.3991
1-3-3-1	3	200	2500	0.00774	0.00692	0.3167
1-3-1-1	1	200	2500	0.01522	0.01221	0.3420
1-3-2-1	1	200	2500	0.00798	0.00745	0.3591
1-3-4-1	1	200	2500	0.00978	0.00928	0.3876
1-5-3-1	1	200	2500	0.00615	0.00607	0.3409
1-5-2-1	1	200	2500	0.00671	0.00623	0.3215
1-10-2-1	1	200	2500	0.00508	0.00521	0.3720
1-10-3-1	1	200	2500	0.00457	0.00421	0.2839
1-10-5-1	1	200	2500	0.00430	0.00398	0.3008

Table 4.11: One step ahead prediction of the Henon mapping

Structure	M	epochs	samples	training error	test error	reconstruction
1-2-1	0	100	2500	0.06346	0.07206	0.3136
1-2-1	1	100	2500	0.06157	0.07731	0.3444
1-2-2-1	0	200	2500	0.06348	0.08199	0.2741
1-5-1	0	100	2500	0.09907	0.09926	0.2610
1-5-1	1	100	2500	0.06711	0.0795	0.3088
1-3-3-1	0	200	2500	0.06061	0.0775	0.3298
1-3-3-1	1	200	2500	0.06057	0.0729	0.3470
1-3-3-1	2	200	2500	0.06140	0.0708	0.3470
1-3-3-1	3	200	2500	0.06247	0.0759	0.4010
1-3-3-1	4	200	2500	0.06361	0.08595	0.3722
1-3-3-1	6	200	2500	0.06365	0.07624	0.3235
1-3-3-1	7	200	2500	0.06359	0.07131	0.2548
1-3-3-1	8	200	2500	0.07349	0.2923	0.3056
1-5-5-1	1	200	2500	0.06035	0.0707	0.3277
1-5-5-1	7	800	2500	0.05874	0.0714	0.3051

Table 4.12: One step ahead prediction of sunspot series

the neurons should be converging on the solution at the same rate. Ideally each neuron should have its own individual learning rate and algorithms such as the Delta-Bar-Delta learning rule which implement this do exist, however they are computationally expensive. To improve training a simpler method was used. The learning rate constant η was multiplied by a constant γ for each layer from the input layer i.e. the learning rate constant for a layer was taken to be $\eta\gamma^l$ where l is the number of layers from the input layer. This means that the layers nearer the output layer are adjusted more slowly, making learning more even. This was implemented in the program and was found to give slightly improved results for the networks with two hidden layers, with a value of $\gamma = 0.5$ working well.

System reconstruction for the Henon mapping is poor. The problem is that the reconstruction length of 100 steps is too far beyond the capabilities of any of the networks to predict, so any comparison is of dubious value. For meaningful comparison a shorter reconstruction length is required.

The sunspot series was the first of the series taken from the real world. The error performance for all the networks was fairly similar as shown in Table 4.12. The amount of memory M did not make a significant difference to performance. Although the test error was not poor in overall terms there are indications that the models produced were poor. The sunspot series is thought to be noisy, and visual inspection of the data would seem to confirm that. The data was of monthly sunspot series. The series has been predicted in the past with a degree of success using yearly data with linear autoregressive models or neural networks using data from up to fifteen previous years. The series appears to have an approximately periodic cycle of eleven years. The largest of the networks tried only takes values from the previous 24 months and so does not take data from nearly as far back as the other models. Looking at the data there appears to a high level of high frequency noise overlaying the low frequency eleven year cycle. It seems likely that all the networks that were tried were doing was learning the noise, rather than the underlying dynamics of the system. This view is reinforced by the results of recursive prediction where the network fails to represent the dynamics of the system.

One of the problems found was that when a large value of M was utilised so that many of the previous values were taken into account the number of parameters became very large, in the largest 280. As well as slowing down the training it caused the networks with two hidden layers to overtrain. Since the series is thought to be noisy it would be useful if the neurons in the first hidden layer had a large M to enable it to filter out the noise, but that this should not be needed in later classification layers. It would seem preferable therefore to be able to vary the amount of memory in each layer. The network that won the Santa Fe competition had this type of configuration.

The reconstruction error was uniformly large. This is interesting for although the inability to perform system reconstruction for the logistic and Henon mappings can be attributed to their highly chaotic natures (large Lya-

punov exponents), the sunspot series can be predicted, although inaccurately at least a year ahead. This implies that the networks are failing to successfully model the system, possibly due to the observation noise present.

Structure	M	epochs	samples	training error	test error
1-3-3-1	0	100	2500	0.1451	0.1258
1-3-3-1	1	100	2500	0.02731	0.01353
1-3-3-1	2	100	2500	0.02285	0.00631
1-3-3-1	3	100	2500	0.02161	0.00672
1-3-3-1	4	100	2500	0.02128	0.00487
1-3-3-1	5	100	2500	0.02064	0.00818

Table 4.13: One step ahead prediction of laser series

It can be seen from Table 4.13 that the errors in the prediction of the test series are significantly smaller than that in the training series. This is the opposite to what would be expected. The explanation is that the test series is only 100 samples long as compared to the training series of 2500 samples. It seems that the test series is in a part of the series that is easy to predict compared to the series as a whole. If this were so then if the test series were to be extended it seems likely that it would become more representative of the dynamics of the series as a whole and the test error would increase accordingly. The test series length was thus increased in length to 200 samples leading to an increase in the test error as expected. This is in Table 4.14.

The laser data is fairly noise free and all the networks with $M > 0$ trained well. The networks with two hidden layers seemed to perform best. It was noticed that as M increased to about 8 the training became unstable with the error increasing and decreasing by a significant degree as the training progressed, rather than the usual steady decline. This is possibly due to the large

Structure	M	epochs	samples	training error	test error	reconstruction
1-2-1	0	100	2500	0.1449	0.1681	0.2249
1-2-1	1	100	2500	0.0307	0.0339	0.5635
1-2-2-1	0	200	2500	0.1452	0.1718	0.2184
1-3-3-1	2	200	2500	0.02286	0.01022	0.1972
1-3-3-1	6	200	2500	0.01646	0.01589	0.2965
1-3-3-1	7	200	2500	0.01244	0.00801	0.2282
1-3-3-1	8	200	2500	0.01288	0.01326	0.2341
1-3-3-1	9	200	2500	0.02617	0.08719	0.3522
1-3-3-1	15	200	2500	0.2080	0.1867	0.2122
1-5-5-1	0	200	2500	0.14257	0.14852	0.1981
1-5-5-1	1	200	2500	0.02536	0.01483	0.4315
1-5-5-1	2	200	2500	0.02290	0.00834	0.1908
1-5-5-1	4	200	2500	0.01510	0.00797	0.3071
1-5-5-1	6	200	2500	0.01259	0.00940	0.5221
1-5-5-1	8	200	2500	0.02333	0.01479	0.2090
1-5-5-1	9	200	2500	0.01257	0.04832	0.3358
1-5-5-1	10	200	2500	fail	fail	fail
1-3-3-3-1	0	200	2500	0.1456	0.1473	0.2024
1-3-3-3-1	1	200	2500	0.02509	0.01657	0.2242
1-3-3-3-1	2	200	2500	0.01806	0.01681	0.3210
1-3-3-3-1	4	200	2500	0.04262	0.05391	0.2407
1-3-3-3-1	6	200	2500	0.04487	0.05675	0.2592

Table 4.14: One step ahead prediction of laser series

number of available parameters, many of which may have been redundant.

The system reconstruction errors were still large, although there was variation in values, even when one step ahead prediction values were similar. They were however better than for other series, however they still could not adequately predict 100 steps ahead. They did however in some cases perform well for predicting shorter sequences, but performance was not consistent. Sometimes it performed system reconstruction well, other times poorly.

The ability to predict the laser series further into the future than the logistic and Henon mappings is due to the laser series having a smaller Lyapunov exponent.

4.5.5 Network training to improve recursive prediction performance

It is desired to train a network which performs well when used to predict recursively. So far the networks have been trained to perform one step ahead prediction and then used to predict in a recursive manner. The problem with this is that with high noise levels the noise could be learned rather than the dynamics of the system. Also areas of the state space that are unimportant for one step ahead prediction may prove important when predicting further ahead, and thus be under represented in the model for one step ahead prediction. It would seem sensible instead to train them from the start to perform recursive prediction effectively.

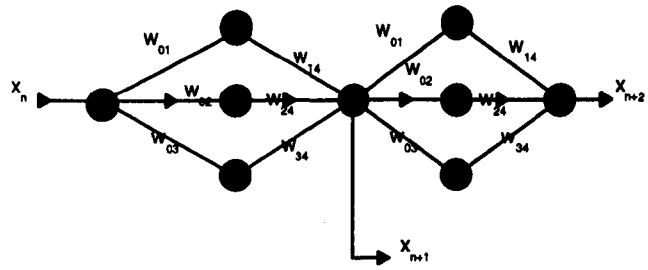

Figure 4.16: System reconstruction model

A possible method to train a one step ahead prediction network to perform well for two step ahead prediction is to concatenate two identical one step ahead networks in series as shown above. This combined network can then be trained for two step ahead prediction using a standard backpropagation algorithm backpropagating the two step ahead error signal back through the network and constraining the transition weights in the two halves of the network to change by the same amount. In order to ensure good one step ahead performance is retained the error signal for one step ahead prediction is backpropagated through the first half of the network. After the network has been trained it

can be divided again into the two one step ahead networks. This gives a one step ahead network that has been trained to perform two step ahead recursive prediction two steps ahead. This procedure could be used to train networks for n step ahead prediction by adding more identical networks to the series prior to training. A potential problem with this approach is that networks with a large number of hidden layers can have difficulties training.

The FIR neural network program was altered to allow training of this type. Normal backpropagation was used and when the weights were updated they were changed by the average weight change of the constrained weights.

$$\Delta_{ij} = \Delta_{i'j'} = \frac{\Delta_{ij} + \Delta_{i'j'}}{2} \tag{4.5.16}$$

At the junction of the two concatenated networks the one step ahead error was injected into the backpropagation. Three different options were experimented with. The first was using the average of the one step ahead error and the two step ahead error already backpropagated through the second half of the network, the second was to just use the two step ahead error so that the one step ahead error was not introduced, the third was to replace the two step ahead error with only the one step ahead error. In the second case the junction of the networks was left floating and not explicitly trained to produce a good one step ahead prediction, only for two step ahead prediction. In the third case the first half of the network trained for one step ahead prediction and the second half for two step ahead prediction. The training method was evaluated using the Logistic function with $M = 0$.

4.5.6 Non-random weight initialisation

Before the training of a backpropagation neural network begins its weights are set to small random values and the network slowly trains towards an error minimum. If however the weights could be estimated utilising knowledge of the

Structure	Option	epochs	γ	OSA train	OSA test	TSA train	TSA test
1-3-1-3-1	1	500	1	fail	fail	fail	fail
1-3-1-3-1	2	500	1	fail	fail	fail	fail
1-3-1-3-1	2	500	0.5	fail	fail	fail	fail
1-3-1-3-1	3	500	1	0.02665	0.02443	0.04767	
1-3-1-3-1	3	500	0.5	0.01661	0.01591	0.03670	
1-5-1-5-1	1	500	0.5	0.01508	0.01354	0.03748	0.03361
1-5-1-5-1	3	500	0.5	0.01571	0.01510	0.03481	0.03216
1-8-1-8-1	3	500	0.5	0.01553	0.01487	0.03446	0.03241
1-3-3-1-3-3-1	3	500	0.5	0.01553	0.01470	0.03791	0.03704
1-3-3-1-3-3-1	2	500	0.5	fail	fail	fail	fail
1-3-3-1-3-3-1	1	500	0.5	0.00211	0.00209	0.00470	0.00417
1-3-3-1-3-3-1	1	800	0.5	0.00195	0.00184	0.00420	0.00371
1-3-3-1-3-3-1	1	500	0.7	0.00310	0.00322	0.00644	0.00593
1-3-3-1-3-3-1	1	500	0.4	0.00230	0.00228	0.00531	0.00480

Table 4.15: Logistic mapping

system then the network could be started closer to a minima, making training faster and hopefully making better use of its processing resources. Inspection of the way in which a single hidden layer network models the logistic function shows a similarity to the stochastic model if the neurons are assumed to be hard limiting. Using this approximation it should be possible to calculate weights which will be close to their final values and then train the network using backpropagation. This approach may allow the use of larger numbers of neurons efficiently resulting in a lesser number of inefficient neurons, and subsequent reduction in disruption to training efficiency.

4.6 Remarks

It has been shown that the Markov chain models have the capability to effectively predict chaotic time series. For sufficient data and memory they can learn the dynamics of a time series to an arbitrary degree of accuracy. How ever in practice these conditions may not apply. Combining the conditional

density functions can be utilised to reduce the quantisation and observation noise, but does not gain significant performance for the added complexity.

The best approach is to utilise a multidimensional model with an embedding dimension equal or greater than that of the modelled series. Unfortunately this can lead to problems due to the rapid increase in number of states required and consequent data requirements.

It was found that linear predictors gave poor results for chaotic time series and were not suitable for this role. Feedforward perceptron networks were found to give better results in general than the stochastic models, due to the in-built quantisation noise of the stochastic models. If it is practical to increase the number of quantisation intervals the stochastic models have the capability to out perform the networks. Also the stochastic models do not suffer from any of the training problems that the neural networks are subject to, i.e. becoming trapped in local minima and long convergence times.

All the approaches examined so far have the disadvantage that they require the dimension of the system to be explicitly built into the model.

It should be noted that due to the use of quantisation intervals the stochastic models were effectively operating on data of much lower precision than the conventional techniques. It is to be expected that if the conventional techniques were to use data of equivalent precision then their performance would suffer, and in such situations use of the stochastic models would prove superior since, as has been shown, the behaviour of chaotic systems known to a limited precision is equivalent to that of stochastic systems.

Chapter 5

Hidden Markov Models For Time Series

5.1 HMM process

The simple Markov chain models developed have a number of disadvantages. Primarily they require the approximate dimension of the dynamic system to be known so a model of an appropriate order can be built, also it needs to be determined which time delayed variables are important to the system dynamics. They are not as efficient in representing the system as they might be and do not take into the account any effects of noise, which is a problem as quantisation noise is inherent in the model. A more sophisticated type of stochastic model is the hidden Markov model.

Hidden Markov models (HMMs) are doubly stochastic models which are commonly used in such applications as speech recognition and image processing [18] [17] [37] [25] [43]. HMMs are an extension of Markov models to the case where the observation is a probabilistic function of the state. They consist of a Markov chain in which each state has a probability of outputting various different output symbols. To an observer only the output string of symbols is

accessible, thus the Markov chain is said to be hidden.

They have the advantage that they do not require a specific model dimension to be known a priori and their structure takes into account the possibility of there being noise in the system.

The probability that a particular sequence of observation signals was produced by a specific hidden Markov model can be assessed. Finding the specific hidden Markov model that has the highest probability of producing the observed data can then be attempted.

5.2 Architecture of hidden Markov model network

A discrete observation signal hidden Markov model is used to model a time series where there are M observation symbols $V = \{v_1, v_2, ...v_M\}$ and O_t is the observed symbol at time t. Thus the time series can be denoted as $O = O_1O_2...O_T$ where each O_t is one of the symbols from V, and T is the length of the time series.

A hidden Markov model is specified by its parameters $\lambda = (A, B, \pi)$. The model has N states, $S = \{S_1, S_2, .., S_N\}$. The state at time t is denoted as q_t. A is the state transition matrix where the element a_{ij} is the transition probability from state S_i to state S_j. B is the observation symbol probability distribution, where the element $b_j(k)$ is the probability of observing symbol v_k when in state S_j. $\pi = \{\pi_i\}$ is the initial state distribution at time $t = 0$.

5.3 Learning algorithm for the hidden Markov model

In order to find the hidden Markov model most likely to have generated the symbol sequence an iterative training algorithm known as the Baum Welsh algorithm can be employed. This is an iterative routine which takes an initial hidden Markov model and then updates its parameters in such a way that the new hidden Markov model is more likely to have generated the time series, or at worst as probable. This new model can then form the basis for another iteration leading to another improved model. This may be repeated until there is no longer any improvement.

It should be noted that the algorithm leads to a local rather than global maxima and that in most complex problems the optimisation surface has several local maxima. The algorithm is related to the standard Lagrange method of constrained optimisation.

The following description follows that given in Rabiner[25]. It is useful to first define two intermediate variables α and β as

$$\alpha_t(i) = P(O_1, O_2 ... O_t, q_t = S_i|\lambda) \tag{5.3.1}$$

$$\alpha_1(i) = \pi_i b_i(O_1), \qquad 1 \leq i \leq N \tag{5.3.2}$$

$$\alpha_{t+1} = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \tag{5.3.3}$$

and

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, ... O_T|q_t = S_i, \lambda) \tag{5.3.4}$$

$$\beta_T(i) = 1, \qquad 1 \leq i \leq N \tag{5.3.5}$$

$$\beta_t = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \tag{5.3.6}$$

Once these have been calculated for a particular model and time series combination, the probability of being in state S_i at time t and state j at time $t+1$,

$\xi_t(i,j)$ can be calculated using equation (5.3.7) and the probability that the state at time t is S_i, $\gamma_t(i)$ using equation (5.3.8).

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)} \tag{5.3.7}$$

$$\gamma_t(i) = \sum_{j=1}^{N}\xi_t(i,j) \tag{5.3.8}$$

Using these variables the Baum-Welch re-estimation equations can then be utilised to find a better estimate for the system parameters.

$$\bar{\pi}_i = \gamma_1(i) \tag{5.3.9}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \tag{5.3.10}$$

$$\bar{b}_j(k) = \frac{\sum_{t=1}^{T-1}\{\gamma_t(j)|O_t = v_k\}}{\sum_{t=1}^{T-1}\gamma_t(j)} \tag{5.3.11}$$

Use of these re-estimation equations gives a new model $\bar{\lambda}$ with the property $P(O|\bar{\lambda}) \geq P(O|\lambda)$.

5.4 Estimation of algorithm parameters

The Baum-Welsh algorithm improves the measure $P(O|\lambda)$ of an initial model until it reaches a local minima. Obviously the minima it reaches is dependent on the initial model parameters λ chosen. The question is how to chose the initial model parameters prior to training in order to finish at the global minima, or at least at an acceptable local minima.

In the simulation study two different methods were used. The first method was to randomly assign values to the parameters subject to probabilistic constraints. The second method was employed only with the Logistic mapping and consisted of a simple estimate of the parameters based on the same techniques used to form the stochastic model of the logistic mapping earlier.

In practice it was found that although the a priori estimate method would converge to a local minima within a few iterations the results in terms of $P(O|\lambda)$ were frequently worse than using the purely random initialisation procedure, although never particularly bad. Unsurprisingly the performance of the purely random procedure varied considerably from poor to quite good.

One question about hidden Markov models is how much data is required to construct an accurate model. The quantity $P(O|\lambda)$ gives the performance measure on which training is based. Longer time series give smaller values so for comparison purposes it is useful to divide this by the number of samples in the time series. This gives the average probability of correctly predicting the next symbol. Figure 5.1 shows how this varies for an 8 state hidden Markov model of the Logistic mapping. Since $P(O|\lambda)$ is calculated from the time series that is used to train the model it is not surprising that for low numbers of samples the probability is near to 1 since each state represents only a few samples. This leads to a unsatisfactory model since its generalisation abilities are poor. When a large number of samples are used then the probability tends towards a limiting value of 0.5. It was observed when training the 3 state hidden Markov model that the average of this parameter per symbol, $(\frac{1}{T}P(O|\theta))$ was 0.5. It is to be noted that for the logistic mapping $e^{-\lambda} = 0.5$ where $\lambda = \ln 2$ is the Lyapunov exponent of the system. This is to be expected since if a difference in trajectories in state space of δ is considered, after a single iteration the difference on average becomes $\delta e^{\lambda} = 2\delta$. If the current area of state space containing the system state variable is known, after one iteration that area will on average have doubled, so the chance of it being in either half is $0.5 = e^{-\lambda}$. It would seem from Figure 5.1 that at least 40 samples per state are required for good generalisation of the logistic mapping. In practice for the Logistic and Henon maps 5000 samples were used to ensure adequate generalisation.

Two of the parameters of the hidden Markov model that have to be fixed a priori are the number of observation symbols, and the number of hidden

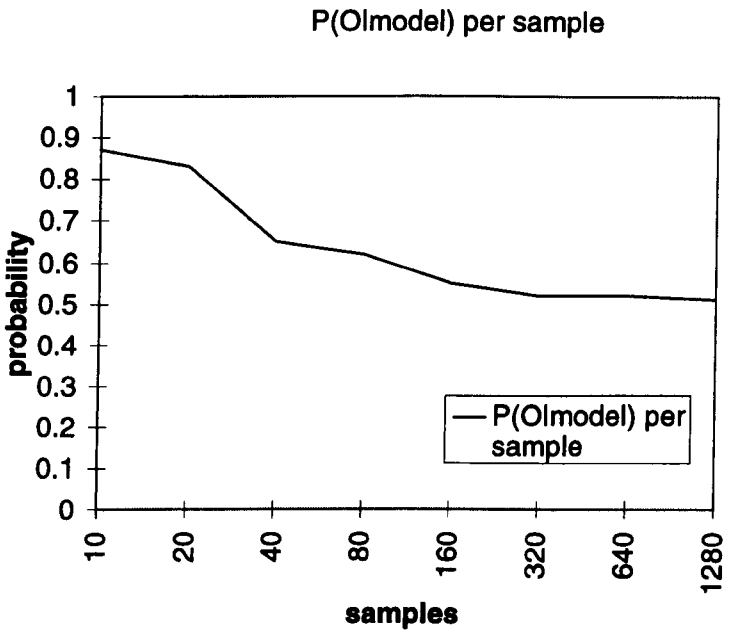

Figure 5.1: The effect of training data quantity

states. It was found that for a fixed number of observation symbols increasing the number of hidden states improved the values of $P(O|\lambda)$ obtained. However after a certain point adding more states ceased to improve the model. The ratio of minimum number of states to reach this point to the number of observation symbols seemed to be approximately constant for each particular time series. Thus to model the time series more accurately both hidden states and number of observation symbols need to be increased.

The time taken to train for one epoch was approximately proportional to the square of the number of hidden states. This is due to the need to calculate all the values of $\bar{a}_{ij}$. However, for models with a larger number of hidden states a larger number of epochs was usually required to achieve convergence. In practice, on a P90 PC, small models took a few minutes to train while larger models with up to 100 states would take a number of hours.

5.5 Application of hidden Markov model for time series prediction

A hidden Markov model routine using the Baum-Welsh re-estimation procedure was implemented on a PC in C++. The actual algorithm utilised differed from that given above to deal with the problem of scaling for long observation sequences. Since α_t and β_t are calculated by the multiplication of a number of terms all of which are less than 1 if t is large then the values of α_t and β_t head exponentially to zero. Similarly for long sequences $P(O|\lambda)$ tends to zero. This leads the precision range of the computer performing the calculations to be exceeded. To counteract this problem a scaling coefficient can be used.

$$c_t = \frac{1}{\sum_{i=1}^{N} \alpha_t(i)} \tag{5.5.1}$$

This is used to scale α and β to form a scaled coefficient set

$$\hat{\alpha}_t(j) = (\prod_{r=1}^{t} c_r)\alpha_t(j) \tag{5.5.2}$$

$$\hat{\beta}_t(i) = c_t\beta_t(i) \tag{5.5.3}$$

which gives the same results as the unscaled set when used in the re-estimation equations, but avoids the problem of values tending to zero.

5.5.1 One step ahead prediction

Once a model has been trained its performance as a one step ahead predictor (e.g. Figure 5.4) can be examined. The two performance indicators used were root mean square error and the proportion of the predicted symbols which were correct. When the models trained well then the percentage correctly predicted symbols was approximately constant regardless of the actual number of observation symbols used. This is not surprising since the precision of the prediction is dependent on the precision to which the state of the system can

be determined. If the model trained well, then as the number of symbols was increased the root mean square error decreased in proportion to the inverse of the number of symbols.

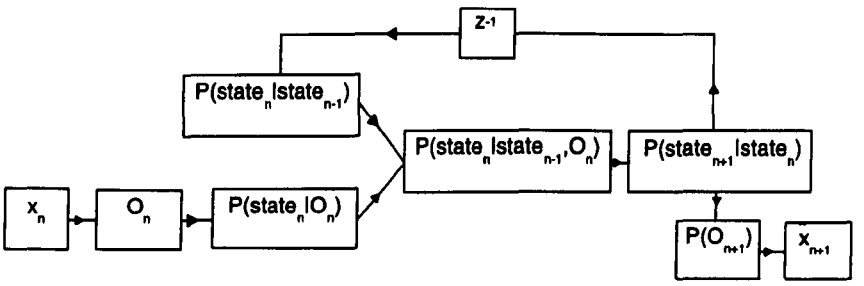

Figure 5.2: Prediction using a hidden Markov model

When the hidden Markov model is utilised for prediction, as shown in Figure 5.2, the initial information is the estimate of the current state of the model and the current observation symbol. These two pieces of information are combined using Bayes law to form a new estimate of the state of the model. The transition matrix is then applied to this state estimate to provide an estimate of the state of the model one time step into the future. This estimate is then fed back to be used as the base state estimate for the next prediction. It is also used to calculate an estimate of the probability of each of the possible symbols being displayed. Thus a quantized probability density function for the output variable is produced. If a single symbol prediction is required then either the most probable symbol can be chosen, which will give the best value for number of correct symbols, or a weighted mean of states giving the best value for root mean square error.

The first model tested was a 3 state 4 symbol hidden Markov model of the logistic mapping which is a more compact version of that in Figure 5.3 but with the same statistical properties. This model was used as it is small enough to be easily understood and analysed. It is also an exact analytical model in the sense that the state space boundaries dividing the states coincide with

the observation space boundaries dividing the symbols. This means each state corresponds exactly to one or more observation symbols.

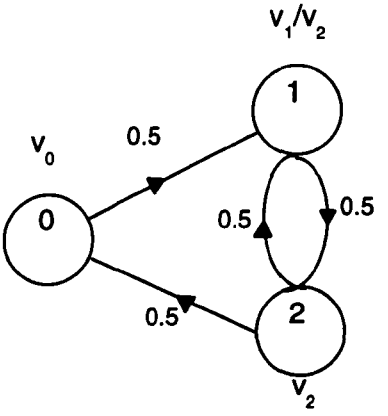

Figure 5.3: The three state hidden Markov model of the Logistic mapping

The hidden Markov model can be used to predict its own performance as a one step ahead predictor assuming that it is an accurate model. Since all the states are equiprobable $P(q_t = S_0) = P(q_t = S_1) = P(q_t = S_2) = \frac{1}{3}$.

If $q_t = S_0$

$$P(O_{t+1} = v_0) = 0.5 \tag{5.5.4}$$

$$P(O_{t+1} = v_1) = 0.25 \tag{5.5.5}$$

$$P(O_{t+1} = v_2) = 0.25 \tag{5.5.6}$$

thus

$$\hat{O}_{t+1} = v_0, \text{ and so } \mathrm{E}(\hat{\mathrm{O}}_{t+1} = \mathrm{O}_{t+1} | \mathrm{q}_t = \mathrm{S}_0) = 0.5 \tag{5.5.7}$$

if $q_t = S_1$

$$P(O_{t+1} = v_3) = 1 \tag{5.5.8}$$

thus

$$\hat{O}_{t+1} = v_3, \text{ and so } \mathrm{E}(\hat{\mathrm{O}}_{t+1} = \mathrm{O}_{t+1} | \mathrm{q}_t = \mathrm{S}_1) = 1 \tag{5.5.9}$$

if $q_t = S_2$

$$P(O_{t+1} = v_0) = 0.5 \tag{5.5.10}$$

$$P(O_{t+1} = v_1) = 0.25 \tag{5.5.11}$$

$$P(O_{t+1} = v_2) = 0.25 \tag{5.5.12}$$

thus

$$\hat{O}_{t+1} = v_0, \text{ and so } \mathrm{E}(\hat{\mathrm{O}}_{t+1} = \mathrm{O}_{t+1}|\mathrm{q}_t = \mathrm{S}_2) = 0.5 \tag{5.5.13}$$

This gives

$$E(\hat{O_{t+1}} = O_{t+1}) = (0.33 * 0.5) + (0.33 * 1) + (0.33 * 0.5) = 0.67 \tag{5.5.14}$$

Repeating one step ahead prediction 300 times gave a one step ahead correct symbol prediction rate of 67%, the same as that predicted implying that the model is a good representation of the logistic mapping quantized into 4 symbols.

Instead of doing the above calculation by hand the predictor can make an estimate of the number of correct predictions it expects to make. It is to be noted that a 100% prediction rate is not to be expected unless the underlying system was purely deterministic and the data known to infinite precision. Since the initial condition in state space is only known to a limited precision due to quantisation noise the underlying system is effectively stochastic as has been shown previously.

Table 5.1: Correct prediction of hidden Markov model symbols

HMM	Symbols Correct	Expected Correct
Logistic	61%	62%
Henon	65%	67%

The one step ahead performance of the hidden Markov model was compared to that of the simple Markov chain model used previously and a feed forward multilayer perceptron network trained using backpropagation (Table 5.2). The Markov chain model and the hidden Markov model both had 20 observation symbols while the MLP network had 10 hidden neurons. The hidden Markov

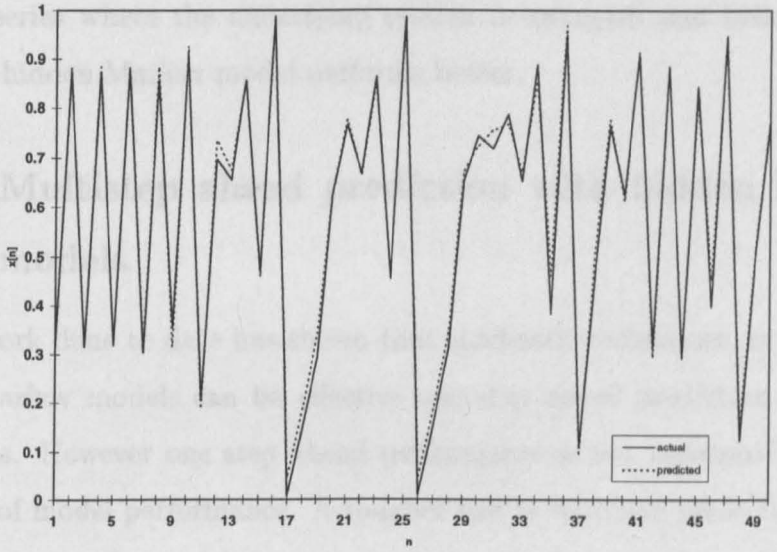

Figure 5.4: One step ahead prediction of the Henon mapping using a 100 state hidden Markov model

Table 5.2: RMS errors of three models

Model	Logistic	Henon	Sunspot
Markov Model	0.0411	0.0259	
HMM	0.0363	0.0216	0.0566
MLP network	0.0119	0.0252	0.0697

model gave improved performance over the Markov chain model and thus is clearly an improvement, although at the expense of added complexity. The hidden Markov model was not as effective at modelling the logistic mapping as the MLP network. This is not surprising since the hidden Markov model is limited to predicting a symbol representing a level 0.05 wide and thus has a maximum precision available to it even if it predicts the correct symbol each time. Since the Logistic mapping is a simple system the MLP network can approximate it easily. The Henon map is a more complex system and the MLP

is slightly worse at predicting it than the hidden Markov model, while for the Sunspot series where the underlying system is unknown and believed to be noisy the hidden Markov model performs better.

5.5.2 Multistep ahead prediction with hidden Markov models

The work done to date has shown that stochastic techniques, in particular hidden Markov models can be effective one step ahead predictors of chaotic time series. However one step ahead performance is not necessarily the best indicator of model performance. A tougher test is recursive prediction.

One measure for multiple step prediction is the error long term (ELT) given by equation (4.4.2). Another possible measure when the time series is treated as a series of observation symbols is the number of length 5 observation symbol sequences correct per 1000 reconstructions. Both these measures were used to assess the performance of hidden Markov models for multistep ahead prediction.

Our task is to provide an optimum prediction of the next five steps of the time series. Using a hidden Markov model there are several choices as to what to consider optimum. The first possible criteria is to just use the ELT. This would be the mean state symbol at each step (not the most probable symbol). This has the disadvantage that the sequence generated might not be a legal sequence of symbols according to the model.

The first reconstruction technique used was to select the most probable next state at each state and display the most probable symbol given that state. The next state was chosen purely on the basis of the last state being that previously selected. This has the advantage that it is easy to calculate.

The first model used was the 3 state model of the logistic mapping given earlier (Figure 5.3). Out of 15000 5-step ahead reconstructions 4.16% resulted

in a sequence of 5 symbols that exactly matched the actual sequence. The ELT for 5000 reconstructions was 0.225. The ELT is fairly poor, but considering that only 4 observation symbols were used this is to be expected. Since the model is fairly simple the expectation of the percentage of correct reconstruction sequences can be calculated.

If the system is currently in state S_0 the most probable next state is S_0 , if in S_1 it is S_2, and if S_2 it is S_0. This gives us two different reconstructed sequences. If the initial state is S_0 or S_2 then the predicted sequence is 00000, while if it is S_1 then the sequence is 30000. Since the probability of 00000 occurring from state S_0 of S_2 is $\frac{1}{32}$ and 30000 from S_2 is $\frac{1}{16}$ and all initial states are equally likely then the overall probability of the sequence being correct is $\frac{1}{24}$ giving an expectation of 4.17% of predictions being correct giving good agreement with the simulation results.

HMM	$logP(O\|\lambda)$	OSA error	OSA correct	R(5) correct	R(5) exptd	R(5) error
3/4	-1505	0.195	66%	4.2%	4.2%	0.474
8/8	-1744	0.094	63%	7.0%	5.5%	0.419
12/8	-1597	0.082	64%	7.4%	8.4%	0.400
12/8*	-1617	0.082	64%	5.9%	6.6%	0.369
16/8	-1546	0.079	64%	4.6%	4.3%	0.380
16/8	-1540	0.079	64%	3.1%	3.4%	0.338
16/8*	-1938	0.089	59%	2.0%	3.5%	0.373
20/20	-2216	0.052	55%	5.9%	3.9%	0.332
20/20*	-2347	0.038	50%	0.3%	0.8%	0.295
30/20	-1765	0.034	61%	7.0%	5.4%	0.309
30/20	-1755	0.035	62%	7.3%	5.5%	0.309
40/20	-1752	0.035	60%			
60/20*	-1762	0.034	60%	3.0%	4.3%	0.263

Table 5.3: HMM System reconstruction method 1 results

Those hidden Markov models marked * in Table 5.3 were trained by setting the initial state transition and observation symbol probabilities based on the analytical technique used to build the 3/4 hidden Markov model, and then

trained normally. The other hidden Markov models were trained using random initial probabilities.

The 3/4 hidden Markov model performed as predicted by theory, with the actual one step ahead correct symbol prediction and 5 step ahead correct symbol sequence percentages equal to their expected values.

If the long term behaviour of systems reconstructed using this method is examined then it is found that they display periodicity. This strongly contrasts with the real time series which are aperiodic. The reasons for this are the same as the periodicity in the iterated Markov chain model.

The second technique used was to take the initial state probability distribution and use that to calculate the state probability distribution for the next state. That state distribution was used to calculate a symbol probability distribution and the most probable symbol in that distribution chosen. The state distribution was then used to calculate the state distribution for the next step.

The results for this method, shown in Table 5.4, are very poor when it comes to correct symbol sequence prediction, however the ELT is slightly better than the previous method. The reason for the poor symbol sequence results is that because the state distribution is being used, rather than choosing individual states at each step. This means that sequences that are illegal, in that they cannot be formed by the model, can be generated. Thus this method can be considered a poor method of reconstruction. However it is to be noted that the ELT does not consider this a poor method, showing that there are problems with using ELT as a performance measure for system reconstruction.

The third technique was identical to the second technique except that the weighted mean symbol was chosen.

The correct symbol sequence prediction, as shown in Table 5.5, is even worse than the previous method, which is unsurprising since the most probable symbol is no longer predicted at each stage. However the ELT is significantly better than either of the previous methods. This is not surprising since it is

HMM	$logP(O\|\lambda)$	R(5) correct	R(5) exptd	R(5) error
3/4	-1505	0.0%	0.0%	0.432
8/8	-1744	0.0%	0.0%	0.45
12/8	-1597	0.4%	0.4%	0.382
12/8*	-1617	0.0%	0.0%	0.376
16/8	-1546	0.4%	0.5%	0.367
16/8	-1540	0.9%	0.9%	0.361
16/8*	-1938	1.4%	1.5%	0.425
20/20	-2216	0.0%	0.1%	0.335
20/20*	-2347	0.3%	0.2%	0.348
30/20	-1765	1.3%	1.2%	0.311
30/20	-1755	1.1%	1.1%	0.314
60/20*	-1762	1.1%	1.7%	0.31

Table 5.4: HMM System reconstruction method 2 results

effectively a weighted mean symbol sequence and thus should give the best ELT it is possible to generate using the hidden Markov model. Thus the method that generates the best ELT tends to generate most illegal symbol sequences.

Both this and the previous method deal with the evolution of the density of states. In the long term this density converges to an invariant distribution, leading to an invariant symbol distribution corresponding to the symbol distribution of the strange attractor. This means that the predicted symbol sequence converges to a single symbol, either the most probable symbol in the attractors symbol distribution, or the mean symbol value. In this case the series converges to the mean value of the time series, giving the long term prediction with the smallest ELT, but generally an invalid symbol sequence.

The fourth technique was to, instead of predicting each step individually,

HMM	$logP(O\|\lambda)$	R(5) correct	R(5) exptd	R(5) error
3/4	-1505	0.0%	0.0%	0.322
8/8	-1744	0.0%	0.0%	0.294
12/8	-1597	0.0%	0.0%	0.258
12/8*	-1617	0.0%	0.0%	0.259
16/8	-1546	0.0%	0.0%	0.254
16/8	-1540	0.0%	0.0%	0.253
16/8*	-1938	0.0%	0.0%	0.272
20/20	-2216	0.0%	0.0%	0.222
20/20*	-2347	0.0%	0.0%	0.229
30/20	-1765	0.2%	0.2%	0.204
30/20	-1755	0.2%	0.2%	0.207
60/20*	-1762	0.6%	1.4%	0.209

Table 5.5: HMM System reconstruction method 3 results

calculate the probability of all the possible sequences of five states, and then choose the most probable symbol sequence given that state sequence.

This method can be considered an improvement over the first method since the whole state sequence is considered at once rather than a step by step approach. This however leads to more complex implementation, taking longer to calculate. It does give better results than the first method, if not greatly so. It is to be noted that both this and the first method only allow legal symbol sequences.

Both these methods however have the disadvantage that they are generating single state sequences. The problem is that the states are hidden, not having any real physical meaning, what is required is not a state sequence, but a symbol sequence. Thus and improved method would be to find the most

HMM	$logP(O\|\lambda)$	R(5) correct	R(5) exptd	R(5) error
3/4	-1505	4.4%	4.3%	0.422
8/8	-1744	3.1%	1.8%	0.416
12/8	-1597	6.6%	6.6%	0.371
12/8*	-1617	5.1%	4.9%	0.348
16/8	-1546	6.6%	6.7%	0.376
16/8	-1540	4.5%	5.1%	0.356
16/8*	-1938	3.0%	5.1%	0.405
20/20	-2216	6.4%	4.1%	0.343
20/20*	-2347	1.1%	1.3%	0.311
30/20	-1765	5.8%	5.1%	0.303
30/20	-1755	6.6%	5.3%	0.307
60/20*	-1762	2.8%	4.0%	0.275

Table 5.6: HMM System reconstruction method 4 results

probable symbol sequence. Unfortunately this is complex to implement and requires much computation, especially with long sequences.

5.6 Simulation study of the sunspot series

A simulation study of the ability of a HMM to model the sunspot series was performed. The series consisted of yearly averages (rather than the monthly averages that had been used previously), this meant that the data series was only 225 samples.

A series of HMMs were generated, starting with a small number of quantisation intervals (symbols) and states. The number of both symbols and states were then increased. The smaller models not only trained faster, taking sec-

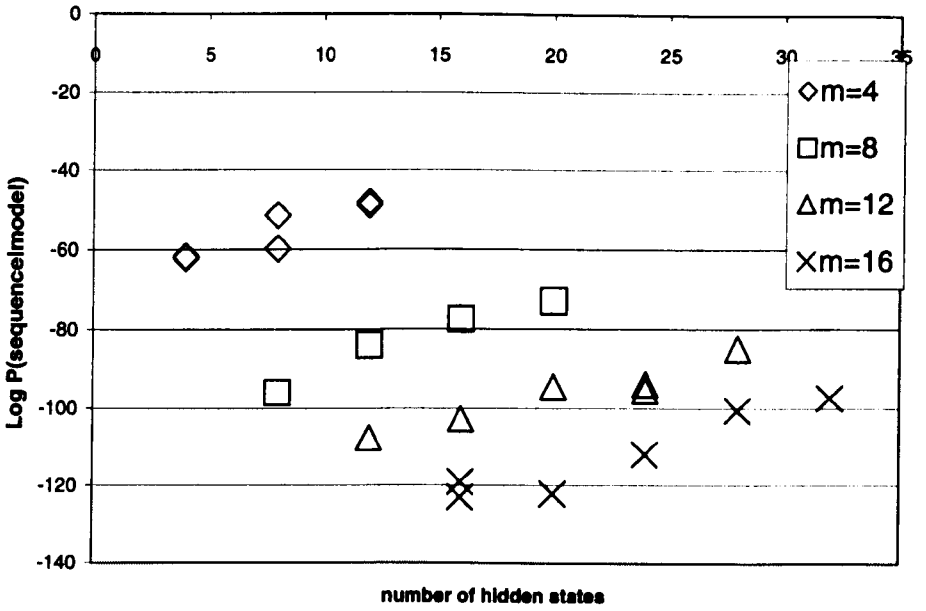

Figure 5.5: The effect of the number of symbols and states on the training measure

onds to train, but required only a small number of epochs to train. As the models increased in size the number of epochs required before the ceased to improve also increased. Thus the largest models took up to half an hour to train.

Each time the number of symbols m increased the quality of the model, as given by the probability of the actual sequence being generated given the trained model, decreased. To maintain the quality of the model the number of states was also required to increase. This is shown in Figure 5.5. Although increasing the number of symbols reduced the measure $P(observedsequence|model)$ it improved the performance in terms of root mean square error. This is because although the model predicts the correct symbol slightly less often the interval represented by the symbol is smaller leading to a more accurate result.

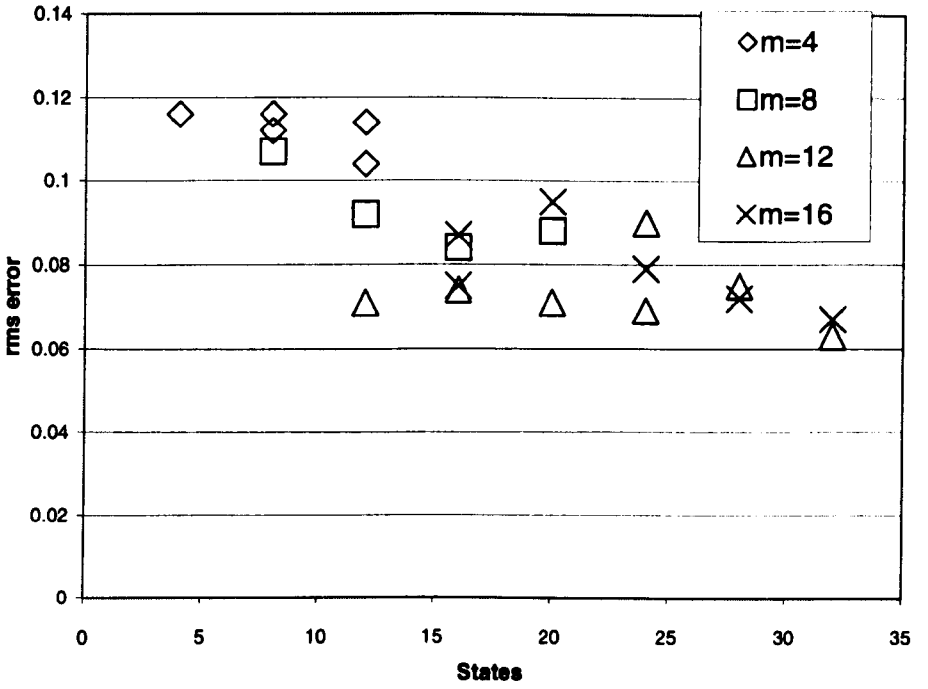

Figure 5.6: The effect of the number of symbols and states on the one step ahead prediction error

It can be seen from Figure 5.6 that increasing the model size ceases to improve the prediction error after a certain size of model is reached. This is due to larger models being difficult to train, and requiring more data. Since there are only 225 data points there are less than 8 data points per state on average leading to poor estimates of the symbol density functions for each state and thus poor generalisation.

The results are given in Table 5.7. The number of symbols correctly predicted out of 100 predictions is given along with the expected number of correct predictions if the model were a true representation of the underlying system.

5.7 Remarks

The hidden Markov models were shown to be more effective than the Markov chain models and are a more efficient representation of the modelled system. They have the advantage that they do not require the dimensionality of the modelled system to *be explicitly* specified. They do however have to be trained and like neural networks are subject to the possibility of becoming trapped in local minima and may require long training times. Training does become more difficult as the number of states increases.

It should be noted however that the hidden Markov models implemented were relatively unsophisticated. There is great potential for optimising the model architecture and training algorithms to take advantage of the chaotic structure of the system. This should lead to a lower number of parameters, better generalisation and faster training.

As with the Markov chain models, the hidden Markov models utilise data to a lower level of precision that available to the conventional techniques due to the use of discrete symbols.

States	Symbols	Iterations	Correct	Exptd Correct	Error	$logP(O\|\lambda)$
4	4	20	76	72.3	0.116	-62.037
8	4	20	76	71.2	0.116	-59.85
4	8	100	81	74.6	0.112	-51.49
4	4	100	76	72.2	0.116	-61.82
12	4	100	77	77.4	0.114	-48.05
12	4	100	79	78.2	0.104	-48.9
8	8	100	54	54	0.107	-95.89
12	8	100	54	58.1	0.092	-83.75
16	8	100	64	62.8	0.084	-77.34
20	8	100	68	66.4	0.088	-72.76
12	12	100	56	55.3	0.071	-107.62
16	12	100	59	56.7	0.074	-102.65
20	12	100	55	60.5	0.071	-94.43
24	12	100	64	63.2	0.09	-95.42
24	12	100	58	57.9	0.069	-93.78
28	12	150	66	62.3	0.075	-84.68
32	12	150	60	62.5	0.063	
16	16	100	51	50.8	0.087	-119
16	16	150	46	46.7	0.075	-123.02
20	16	150	47	49.7	0.095	-122.16
24	16	150	53	53.2	0.079	-111.876
28	16	150	52	55.5	0.072	-100.43
32	16	150	60	59.4	0.067	-97.26

Table 5.7: HMM one step ahead prediction of yearly sunspot series

Chapter 6

Conclusion

6.1 Summary

The validity of modelling chaotic dynamic systems using equivalent stochastic models has been demonstrated. This has been achieved in a number of steps.

1. It has been shown that if the initial conditions of a chaotic system (the Logistic mapping) are known only to a finite precision then the system appears identical to a stochastic system.

2. The theoretical links between chaotic and stochastic dynamic systems has been investigated via an equivalence relationship and the evolution of the density of dynamics described by the Frobenius-Perron operator.

3. It has been shown that it is possible to build a simple hidden Markov model of the Logistic mapping analytically.

4. It has been shown using computer simulations that hidden Markov models can learn the dynamics of the Logistic mapping through training in an iterative manner using only a time series generated by the Logistic mapping as data.

5. It has been demonstrated that they can also learn the dynamics of other chaotic systems of more than one dimension (the Henon mapping) as well as real world time series such as the Sunspot series.

6.2 General remarks

Some of the characteristics of the models have been explored. Their accuracy is limited by both the number of observation symbols and the number of hidden states. It is also limited by the length of the training data. If sufficient data, computing power and noise free data of high precision is available accuracy can be increased by increasing both the number of symbols and states. However with noisy or low precision data there is a limit to the accuracy that can be obtained. It is under these conditions however that hidden Markov models are most likely to be useful as they take the uncertainty in the current state into directly account in contrast to function approximation techniques which rely on assumptions of Gaussian noise.

Hidden Markov models have the advantage that several of the characteristics of the modelled chaotic system are embedded in the model. The Lyapunov exponent is related to the quantity $P(O|\lambda)$, the topological entropy is related to the state transition matrix. The hidden Markov model has a long term symbol distribution that should match the invariant distribution of the strange attractor and the hidden Markov model, if properly trained, should have the same symbolic dynamics as the system.

If the stochastic equivalent model of a chaotic system when there is limited precision (equation (3.5.1)) is examined it can be seen that it consists of two terms. The first is the deterministic mean of the systems evolution, the second is a stochastic term taking into account the uncertainty due to the imprecision in knowledge of the current state. Most conventional function approximation techniques (such as ARMAX system identification techniques) concentrate on

finding the deterministic term to the highest possible accuracy, while ignoring the stochastic term. This means that they begin to struggle for all but the shortest term predictions. Discrete symbol hidden Markov models take into account the stochastic term, but do not model the deterministic term as well as the function approximation techniques. In order to model chaotic time series well it is necessary to model both terms to the limit of the accuracy of the available data.

6.3 Limitations of approach

The approach is only valid for a fairly narrow class of dynamic systems, those nonlinear dynamic systems which display low dimensional chaos and have settled to a steady state condition. In order for the approach to be effective it requires a sufficient quantity of data to provide a good estimate of the transitional probability density functions.

For time series of limited length this means that it is not as effective as standard point prediction techniques for point prediction of low noise systems. It does however provide a more informative output in the form of a probability distribution.

The number of states required to represent higher dimensional attractors grows exponentially with increasing dimension exacerbating the problem of data quantity and requiring significant amounts of computer capacity and increasing training times.

6.4 Recommendations for further study

The hidden Markov models studied have been fairly simple examples. More sophisticated models such as continuous distribution hidden Markov models described by Juang [2] and Liporace [24] are a further avenue of enquiry, as

are partially parameterised models for improved training (Gales [33] [32]), and recursive updating for online model estimation.

Alternatively a neural network or hybrid hidden Markov model/neural network such as that proposed by Bengio & Frasconi [46] or Bourlard et al [40] could be used to model density evolution. Tino [35] describes modelling stochastic automata with recurrent neural networks suggesting these may be suitable, while Valtchev [41] proposes recurrent input transformations for hidden Markov models.

Another approach would be to model the transition function using a standard technique, rather than model the evolution of the density directly, and then use the modelled transition function to find the Perron-Frobenius operator analytically. The Perron-Frobenius operator could then be utilised to model the density evolution.

It is suggested that techniques based on estimating the evolution of the density of the dynamics may prove more effective for the prediction of chaotic systems in the presence of significant uncertainty in the initial conditions due to noise. Further investigation is required.

Since it has been shown that chaotic dynamic systems can be effectively modelled by stochastic systems, the control of chaotic systems via stochastic control techniques is a logical extension.

Bibliography

[1] LASOTA A. and MACKEY M.C. *Chaos, fractals and noise: Stochastic aspects of dynamics.* Springer Verlag, 1994.

[2] JUANG B.H. Maximum-likelihood estimation for mixture multivariate stochastic observations of markov chains. *AT&T Technical Journal*, 64(6):1235–1249, July 1985.

[3] BERGER C.S. Linear splines with adaptive mesh sizes for modelling nonlinear dynamic systems. *IEE Proc-Control Theory Appl*, 141(5), 1994.

[4] KIM D. Forecasting time series with genetic fuzzy predictor ensemble. *IEEE Trans. on Fuzzy Systems*, 5(4):523–535, November 1997.

[5] STAMP D.I. and WU Q.H. Prediction of chaotic time series using hidden markov models. *UKACC international conference on control '98*, 2:1420–1425, 1998.

[6] LATHROP D.P. and KOSTELICH E.J. Characterisation of an experimental strange attactor by periodic orbits. *Phys.Rev.A*, (40), 1989.

[7] LORENZ E.N. Deterministic non-periodic flow. *J.Atmos.Sci*, 20:130–141, 1963.

[8] TAKENS F. Detecting strange attractors in turbulance. *Dynamical Systems and Turbulence, Lecture Notes in Mathematics*, (898):366–381, 1980.

[9] ARGYRIS J. FAUST G. and HAASE M. *An exploration of chaos*, volume VII of *Texts on Computational Mechanics.* North-Holland, 1994.

[10] DECO G. and SCHURMANN B. Neural learning of chaotic dynamics. *Neural Processing Letters*, 2, 1995.

[11] FLAKE G. A poor man's approach to controlling chaos with neural networks.

[12] SUGIHARA G. Nonlinear forecasting for the classification of time series. *Phil.Trans.R.Soc. Lond. A*, (384), 1994.

[13] BAKER G.L. and GOLLUB J.P. *Chaotic dynamics an introduction.* Cambridge University Press, 1990.

[14] FISHMAN G.S. *Monte Carlo Concepts, Algorithms, and Applications.* Springer, 1996.

[15] POINCARE H. *Les methodes nouvelles de la mechanique celeste.* Springfield, 1967.

[16] TONG H. *Nonlinear time series.* Clarendon Press, 1990.

[17] MACDONALD I.L. and ZUCCHINI W. *Hidden Markov and other models for discrete-valued time series.* Chapman and Hall, 1997.

[18] FARMER J. and SIDOROWICH J. Predicting chaotic time series. *Physical Review Letters*, 59(8), 1987.

[19] MCDONNELL J. and WAAGEN D. Evolving recurrent perceptrons for time series modelling. *IEEE Transactions on Neural Networks*, 5(1), 1994.

[20] STARK J. Recursive prediction of chaotic time series. 1992.

[21] JUSTICE J.H.(Ed). *Maximum Entropy and Bayesian Methods in Applied Statisics.* Cambridge University Press, 1984.

[22] WARWICK K. and KARNY M. System identification using partioned least squares. *IEE Proc-Control Theory Appl.*, 142(3), 1995.

[23] LJUNG L. *System Identification Toolbox.* The Mathworks Inc., 1991.

[24] LIPORACE L.A. Maximum likelihood estimation for mutivariate observations of markov sources. *IEE Tran. Information Theory*, 28(5), 1982.

[25] RABINER L.R. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77(2):257–284, Feb 1989.

[26] YANG L.S. Entropy, lyapunov exponents, and hausdorff dimension in differentiable dynamical systems. *IEEE Trans. on Circuits and Systems*, 8(30):599–607, 1983.

[27] BEHERA L. GOPAL M. and CHAUDHURY S. Inversion of rbf networks and applications to adaptive control of nonlinear systems. *IEE Proc-Control Theory Appl.*, 142(6), 1994.

[28] CASDAGLI M. Chaos and deterministic versus stochastic nonlinear modelling. *J.Statist.Soc. B*, (2), 1991.

[29] DELLNITZ M. and JUNGE O. On approximation of complicated dynamical behavior. Sept 1997.

[30] KADIRKMANATHAN V. NIANJAN M. and FALLSIDE F. Sequential adaption of radial basis function neural networks. *Advances in Neural Information Processing Sysyems*, 3:721–727, 1991.

[31] GRIGORIEV R.O. CROSS M.C. and SCHUSTER H.G. Pinning control of spatiotemporal chaos. April 1997.

[32] GALES M.J.F. Semi-tied full-covariance matrices for hidden markov models. April 1997.

[33] GALES M.J.F. and YOUNG S.J. The theory of segmental hidden markov models. June 1993.

[34] YORKE J.A. OTT E., GREBOGI C. Controlling chaos. *Physical Review Letters*, (64):1196–1199, March 1990.

[35] TINO P. and KOTELES M. Extracting finite-state representations from recurrent neural networks trained on chaotic symbol sequences. *IEEE Trans. on Neural Networks*, 10(2):284–302, March 1999.

[36] WU Q.H. and CAO Y.J. An equivalent stochastic system model for control of chaotic dynamics. *The* 34^{th} *IEEE Conference on Decision and Control*, 3:2898–2903, 1995.

[37] ELLIOT R.J. and AGGOUN L. *Hidden Markov models, estimation and control.* Springer, 1991.

[38] MAY R.M. Simple mathematical models with very complicated dynamics. *Nature*, (26):459–67, 1976.

[39] HAYKIN S. *Neural Networks: A Comprehensive Foundation.* MacMillan College Publishing, 1994.

[40] HENNEBERT J. RIS C. BOURLARD H. RENALS S. and MORGAN N. Estimation of global posteriors and forward-backward training of hybrid hmm/ann systems.

[41] VALTCHEV V. KAPADIA S. and YOUNG S.J. Recurrent input transformations for hidden markov models.

[42] MEYN S.P. and TWEEDIE R.L. *Markov chains and stochastic stability.* Springer-Verlag, 1993.

[43] BRANTS T. Parameter optimization for hidden markov models. Master's thesis, Universitat des Saarlandes.

[44] VINCENT T.L. Control using chaos. Dec 1997.

[45] HUBER U. and WEIESS C. Lorenz-like chaos in $nh_3 - fir$ lasers. *Time Series Prediction, WEIGEND, A., GERSHENFELD, N.(Ed)*, 1994.

[46] BENGIO Y. and FRASCONI P. Input-output hmm's for sequence processing. *IEEE Trans. on Neural Networks*, 7(5):1231–1248, Sept 1996.

[47] WANG Y. and LIN C. Runge-kutta neural network for identification of dynamical systems in high accuracy. *IEEE Transactions on Neural Networks*, 9(2):294–307, March 1998.