EFFICIENT AND PREDICTION ENHANCEMENT SCHEMES IN CHAOTIC HYDROLOGICAL TIME SERIES ANALYSIS

DULAKSHI SANTHUSITHA KUMARI KARUNASINGHA

(B. Sc. Eng. (Hons), University of Peradeniya, Sri Lanka)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF CIVIL ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2005

ACKNOWLEDGEMENT

I would like to express my sincere and deep gratitude to my supervisor, Associate Prof. Liong Shie-Yui, who guided my research work. His constructive criticisms, valuable advices, suggestions and untiring guidance were very much helpful to me in completing my thesis successfully. I must add that his advices and encouragement were beyond the academic scope; they helped changing my attitudes and improve my personal life. I highly appreciate his support and encouragement at times when I was frustrated due to circumstances beyond our control. The freedom he gave made my research work truly an enjoyable experience. I could not have made this far without the help, encouragement and the freedom I received from him. I thank him from the bottom of my heart.

I must express my sincere thanks and deep gratitude to Prof. K. S. Walgama, who is behind all my academic endeavors since my graduation. He is the only one who educated me not to miss the fun out of this PhD process; he also taught me how to enjoy what I am doing. There was a time when it seemed that everything was going to fall apart; he was the one who showed me that I was getting some life experience. And taking his own experiences as examples he showed me how to face, appreciate and learn from the problems. Without him, I would not have been doing a PhD.

I must express my sincere thanks to Dr. Janaka Wijekulasooriya, who helped me with my leave matters, for placing trust in me and made my study in Singapore possible. His encouragement and friendly advices are highly appreciated.

Thanks are extended to A/Prof. Lin Pengzhi as well.

I would like to express my sincere thanks to Associate Prof. S. Sathiya Keerthi for his inspiring lectures on Neural Networks.

I would like to express my sincere thanks to Dr. Malitha Wijesundara for helping me with the computer related matters throughout my PhD study.

I must thank Prof. N.E. Wijesundara for his guidance and encouragement in tough times. I wish to thank Mr. OG Dayaratne Banda, Mr. Suranga Jayasena and Mr. Lesly Ekanayake for listening to my worries when I was in despair, and for their encouragement.

I wish to thank Dr. T. Vinayagam for helping me with the proofreading. I must also thank my friend Ms. Dinuka Wijethunge for helping me with the proofreading (we hadn't exchanged a word for 14 years till I asked her favour this time -- still the same friend whom I met in High School!) when she herself was busy with loads of work. I must thank my friend Ms. Rochana Meegaskumbura too for her help on this boring proofreading job.

I would also like to thank my friends and colleagues, Ms. Yu Xinying and Mr. Doan Chi Dung (who are now Dr. Yu Xinying and Dr. Doan Chi Dung, of course!), with whom I had a wonderful time, for their discussions on academic and non academic matters. Xinying, the *female* PhD student! , perfectly understood me all the time. I must thank my colleague Mr. M.F.K. Pasha for helping me in the initial stage of my study.

Many thanks to Mr. Krishna of Hydraulics lab who is always there to lend his assistance within his capacity. Thanks are also extended to the staff of Supercomputing and Visualization Unit, NUS, for their help. Thanks to two final-year project students, Andy and Afzal, for their help as well.

As the names of those who helped me cascade down my memory I am feeling happy and excited at the thought that there are so many helpful hands out there willing to reach me in need. There are simply too many to mention their names. My sincere thanks are extended to everyone who helped me in numerous ways.

My sincere thanks are extended to everyone at the Department of Engineering Mathematics and the University of Peradeniya for granting me a study leave.

I would like to thank the National University of Singapore for granting me the NUS research scholarship to pursue my Ph.D. study here.

Last, but not least, no words can express my deepest gratitude, love and admiration to my parents, Mrs. P. G. Somawathie and Mr. K. G. Gunapala. They kept all their sorrows secret so that their daughter is happy overseas with her study. Without their words of encouragement and tolerance I would not have been able to complete my study in Singapore. I must express my love and admiration for my sister, Lakshmi, and my brother, Waruna, too, who kept all the problems to themselves to allow their sister’s mind free from concerns during her study.

TABLE OF CONTENTS

SUMMARY

This study looked into means of improving prediction accuracy and facilitating efficient analysis of chaotic hydrological time series. The objectives were: (1) to investigate in detail the prediction performances of global prediction models (Artificial Neural Network (ANN) and Support Vector Machine (SVM)) compared to some widely used local prediction models (local averaging and local polynomial), and (2) to find means of incorporating noise reduction techniques in prediction improvement schemes, and (3) to investigate means of extracting system representative smaller sets of data from long data records.

(1) Global models in chaotic time series prediction

A chaotic noise-free Lorenz time series, a Lorenz series contaminated with some known noise levels, and two river flow time series were analyzed for 3 different prediction horizons. ANN outperformed local prediction models practically in all the cases. SVM, implemented with a decomposition technique to facilitate handling large data records, also performed better than local models with the exception of noise-free Lorenz series. On the average both global prediction techniques outperformed the local prediction models considered; however, at the expense of longer computational time. Comparison between performances obtained from ANN and from the relatively new SVM showed that both are equally good. For real time series, the prediction performance difference between them is insignificant.

(2) Noise reduction to improve predictions

Performance of both local and global models is unsatisfactory when data is noisy. This study identified some means to improve the predictions of noisy chaotic time series. It was shown that noise reduced inputs to a model can improve its prediction accuracy. A general perception that the models trained with noise reduced data may help in

improving prediction is found not necessarily true. The findings of this study show that the prediction performance is not necessarily improved by such models if they are not supported with inputs of equal or lesser noise levels. Hence, the study showed the necessity of real-time application of noise reduction to improve prediction. Nonlinear chaotic dynamics literature lacks established techniques capable of real-time noise reduction. It was shown that the Extended Kaman filter, originated from Controls literature, can be used as a reliable and robust technique for real-time noise reduction in chaotic time series. The study proposed a better approach, which eliminated the short-comings of the earlier approaches, to incorporate noise reduction to improve prediction accuracy. The effectiveness of the proposed scheme was demonstrated with EKF.

(3) Data extraction

Large data record demands significant computational resources in chaos analysis. This study proposed a procedure that couples a clustering method, a prediction method, and an optimization method (mGA) to extract a smaller set of system representative data from long data records. Demonstration with Subtractive Clustering Method, SCM (Chiu, 1994), on both synthetic and real time series, showed a considerable reduced data set (approximately 30% - 60% of the total data set) can still achieve the same prediction accuracy as that of the entire record. However, SCM, with four parameters to be optimized, required significant computational effort.

New simple clustering technique

A new clustering method is developed in this study that has only one single parameter. Method is shown to be as equally effective as SCM while it requires much less effort than SCM. The new method, though developed for data extraction in chaotic time series, was shown to be effective on some other multivariate data sets as well. Application of it, on proposed noise reduction scheme with EKF, showed the potential in data extraction procedure to yield efficient analysis of the normally time-consuming applications.

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Δt	=	Sampling interval
f_T	=	Prediction function for T lead-time (actual)
$\hat{f}_T$	=	Approximation of f_T
F_T	=	Global approximation function
F_T^i	=	Local approximation function
$c(m,k)$	=	Coefficients of polynomial models
$\phi_m(X)$	=	Polynomial basis functions
Z_0	=	New point
F_k	=	Local polynomial function corresponding to point X_k
$k(\boldsymbol{x}, \boldsymbol{x}')$	=	Kernel function
$\boldsymbol{x}_i$	=	Input vector
y_i	=	Output value
λ_k	=	Eigen value
$\phi_k(\boldsymbol{x})$	=	Nonlinear basis function of SVM
$\varphi(\boldsymbol{x})$	=	Feature space
$\boldsymbol{W}$	=	Weights of SVM
$R_{emp}[f]$	=	Training error
υ_n	=	Measurement noise
$s(\boldsymbol{x})$	=	Function that maps the points on the attractor into real numbers
$\boldsymbol{w}_n$	=	Dynamical noise

v_n'	=	Remaining discrepancy of from dynamical equations, of a noise reduced estimate
y_i	=	Observed value of a time series (with noise)
$\dot{x}$	=	First time derivative of variable *x* of Lorenz system
$\dot{y}$	=	First time derivative of variable *y* of Lorenz system
$\dot{z}$	=	First time derivative of variable *z* of Lorenz system
$\hat{x}_i$	=	Predicted value of x_i
$\overline{x}$	=	Average value of the time series
ε	=	ε-insensitive distance parameter
$\xi^{(*)}$	=	Slack variables in SVM optimization problem
$\alpha^{(*)}$	=	Lagrange multipliers in SVM optimization problem
$\eta^{(*)}$	=	Lagrange multipliers in SVM optimization problem
$\boldsymbol{e}_k^-$	=	*A priori error* estimate
$\boldsymbol{e}_k$	=	*A posteriori error* estimate
$\boldsymbol{P}_k^-$	=	*A priori* estimate error covariance
$\boldsymbol{P}_k$	=	*A posteriori* estimate error covariance
$\hat{\boldsymbol{x}}_k^-$	=	*A priori* state estimate
$\hat{\boldsymbol{x}}_k$	=	*A posteriori* state estimate
$\boldsymbol{P}_k$	=	*A posteriori* estimate error covariance
ε	=	Neighbourhood size of simple nonlinear noise reduction method
σ	=	Standard deviation of noise (in nonlinear noise reduction)
r_a	=	Influence range

P_1^*	=	Highest potential
r_b	=	Range in which the points will have considerable reduction in potential
k'	=	Modified nearest neighbours value
d	=	Parameter of the new clustering method
$\boldsymbol{A}_k$	=	Matrix relating the previous state to the current state
ANN	=	Artificial Neural Network
AR	=	Accept ratio
b	=	Scalar constant of SVM
$\boldsymbol{B}_k$	=	Matrix relating the control input, $\boldsymbol{u}$ to the state $\boldsymbol{x}$
C	=	Constant determining the trade off between the complexity and training error
EKF	=	Extended Kalman Filter
$\boldsymbol{H}_k$	=	Matrix relating state $\boldsymbol{x}$ to measurement $\boldsymbol{z}$
k	=	Number of nearest neighbours
$\boldsymbol{K}_k$	=	Kalman gain
m	=	Embedding dimension
M	=	Number of basis functions
MAE	=	Mean absolute error
mGA	=	Micro Genetic Algorithm
MLP	=	Multilayer perceptron
NB	=	Number of nearest neighbours
$NRMSE$	=	Normalized root mean square error
P_i	=	Potential of a data point (in clustering)
$\boldsymbol{Q}$	=	Process noise covariance
$\boldsymbol{R}$	=	Observation noise covariance

RR	=	Reject ratio
SCM	=	Subtractive Clustering Method
SF	=	Squash factor (in clustering)
SVM	=	Support Vector Machine
T	=	Lead time/ Prediction horizon
$\boldsymbol{u}$	=	Control input
UKF	=	Unscented Kalman filter
w	=	weight
X_i	=	Phase space vector
x_i	=	Value of a time series
$\boldsymbol{z}$	=	Measurement of a system
σ	=	Width parameter of the Gaussian kernel
τ	=	Time delay

CHAPTER 1

INTRODUCTION

Prediction of hydrological and meteorological time series is an important task in understanding the hydrological and meteorological systems. In the past, linear stochastic approaches such as ARMA were widely used in the prediction of hydrological time series. However, the inherent assumptions underlying such approaches such as linearity may not be applicable to complex and nonlinear hydrological systems (Jayawardena and Gurung, 2000). With the recent developments in chaos theory, it was revealed that most real world systems may be better understood using chaotic dynamical systems theory (e.g. Lorenz, 1963; Jayawardena and Lai, 1994; Rodriguez-Iturbe et al, 1989). This is a relatively new and developing field and yet it has shown promise in identification and prediction of nonlinear real world systems. Particularly due to its potential shown in short term prediction the approach is now gaining popularity in many diverse fields (e.g. physics, chemistry, biology, meteorology, etc) including the prediction of nonlinear hydrological time series.

Prediction of time series with this chaotic dynamical systems approach is generally referred to as phase space prediction. The development of phase space prediction models requires a large number of past records. Most of the current research focuses on methods to further improve the performance of phase space prediction. However, only the traditional local phase space prediction models, which have limited capacity, are widely used owing to their simplicity and ease in implementation with large number of data records. The presence of noise in data also considerably deteriorates the performance of phase space prediction (Kantz and Schreiber, 2004). Searching for and investigating more sophisticated prediction models and noise

reduction methods are thus essential. Also attempts towards extracting a small set of data from larger set of records is very important. This is particularly crucial when large data record poses problems computationally, e.g. memory size and long computational time.

The next section first briefly reviews the chaotic time series analysis: it provides basic understanding of dynamic systems approach and its applications. Thereafter the need for the present study becomes clearer and formulated.

1.1 CHAOTIC TIME SERIES ANALYSIS

1.1.1 Basics of chaos

A breakthrough finding by Takens (1981) served as the stepping-stone for the dynamical systems approach for analysis of chaotic systems. Takens showed that it is possible to reconstruct the dynamics of a chaotic system using only a single variable of the system. In dynamical systems approach, the dynamics of a real world system is modelled using a single observed variable of the system concerned (Takens, 1981). This approach is very useful in understanding dynamical systems where explicit governing equations are not available but one or few variable can be observed. There are two main stages in dynamical systems approach for chaotic time series analysis: (1) chaos identification and (2) prediction.

Identification of chaos in real world data is a difficult task given that the theory of chaos is still at developing stage. There are a few widely used chaos identification methods in the literature. Some of them are: (1) correlation dimension method (e.g. Grassberger and Proccacia, 1983a, b), (2) the Lyapunov exponent method (e.g. Wolf et al., 1985), and (3) Kolmogorov entropy method (e.g. Grassberger Proccacia, 1983c). None of the above methods are, however, without pitfalls. Nevertheless they are incorporated to verify chaos in real world data. Applications of these methods in real

world data have indicated the possibility of chaotic dynamics in various natural and physical systems including hydrological systems. Early studies put more emphasis in identifying chaos in real world data. However, some of these studies have been subjected to debate (e.g. Grassberger, 1986; Theiler, 1986, 1988) owing to the practical limitations in applying these identification methods. Many researchers are now more concerned with which approach is most 'useful' for a given experiment than resolving the question "is it chaos or is it noise?" (Kantz and Schreiber, 2004). Thus, the recent studies put more emphasis on applications of chaos-based techniques for prediction purposes, and the chaotic dynamical systems approach is gaining wide popularity due to the promise it has shown in phase space prediction.

Once a time series is identified as chaotic or can be better modelled by chaotic analysis, one may exploit the short-term predictability. Two types of prediction models: (1) local models which describe the dynamics in local neighbourhoods, and (2) global models which explain the dynamics globally (e.g. Artificial Neural Networks (ANN), Support Vector Machines (SVM)), are used in phase space prediction. Local models have been widely used due to their simplicity in calculation. However, with the advancement of computing resources, global models are now emerging as an alternative. These chaos identification and prediction methods have been increasingly popular in the analysis of hydrological systems as well. The next section discusses some such applications.

1.1.2 Chaos applications

A wide spectrum of chaos based applications has appeared in the past two decades. The major applications in hydrology have been reported in river flow analysis (e.g. Jayawardena and Lai, 1994; Porporato and Ridolphi, 1996, 1997) and in rainfall analysis (e.g. Rodriguez-Iturbe et al, 1989; Sharifi et al, 1990; Sivakumar et al., 1998;

1999). River flow time series analysis with chaotic dynamical systems approach has shown a lot of advancement than in rainfall analysis; the chaos-based techniques have shown good potential in short-term prediction (e.g. Jayawardena and Lai, 1994; Porporato and Ridolfi, 1996, 1997; Jayawardena and Gurung, 2000). This has triggered researchers to experiment more on chaos applications. As a result, current research shows lots of enthusiasm in finding the ways to improve the accuracy of phase space prediction. For example, several methods have been proposed from different data pre-processing methods (e.g. Porporato and Ridolfi, 1997; Jayawardena and Gurung, 2000; Yu et al., 2004) to radical approaches such as inverse approaches (e.g. Babovic et al., 2000; Phoon et al., 2002) to get better prediction performances from phase space prediction. However, all these improvements have been attempted with local prediction models only.

1.2 PRESSING ISSUES

1.2.1 Local or global models?

There is a general understanding that local approximation can give better predictions than global approximation in phase space prediction of chaotic time series. However, it is interesting to conduct a performance comparison between the widely used global models, such as Artificial Neural Network (ANN) and the widely used local models. The superiority of ANN to model the nonlinear dynamics in hydrological time series has been demonstrated in a number of studies (e.g. Karunanithi et al., 1994; Hsu et al., 1995; Elshorbagy et al., 2000; ASCE, 2000). However, only a very few studies (e.g. Elshorbagy et al., 2002a; Sivakumar et al., 2002) have used ANN as a global phase space prediction model in chaotic river flow time series prediction. Elshorbagy et al. (2002a) utilized an ANN model and the popularly used local model (averaging technique) while Sivakumar et al. (2002) used ANN and local polynomial

models. Both studies, however, did not give a conclusive performance comparison of the local and global models and, furthermore, they provided contradictory results. Therefore, further studies on the performance of global ANN models compared to local models in chaotic time series prediction are called for.

Also, very recently the Support Vector Machines (SVM), another machine learning technique, is gaining popularity. Applications have appeared in water resources related fields (e.g. Sivapragasam, 2003). Very recently an application of SVM was demonstrated on chaotic hydrological time series analysis as well (Yu et al, 2004). Yu et al. (2004) showed that SVM was superior to traditional local prediction models and the ARIMA models. However, no comparison between the two techniques, SVM and ANN, has been presented so far. It is thus timely to investigate the performance of the more novel technique compared to the more established competitor ANN.

1.2.2 Prediction with noisy data

Noise can considerably deteriorate the performance of even an impeccable prediction model. The chaotic systems, especially, are very sensitive to initial conditions, thus, the presence of even a small amount of noise can considerably hamper the prediction accuracy. Therefore, it is beneficial to reduce noise in real world data before chaos based techniques are attempted. However, when the true signals are unknown, any noise reduction attempt always leads to the same question “is the removed part indeed noise?”. Some established noise reduction methods have been shown to actually reduce noise and there are criteria used for partial verifications of noise removal. Hence, noise reduction is a very useful tool in noisy time series analysis, since, among other things, it is expected to improve the prediction performance.

Several noise reduction attempts have been made on chaotic hydrological time series analysis (e.g. Porporato and Ridolfi, 1997; Kawamura et al., 1998; Sivakumar et al., 1999b, c; Jawardena and Gurung, 2000). The popular simple nonlinear noise reduction techniques have been used in those studies. However, Elshorbagy et al. (2002b) raised serious doubts on the appropriateness of their noise reduction processes. In addition, the present study notices that the approaches followed by the above studies are not appropriate for real-time processing of noisy data. Most widely used nonlinear noise reduction techniques (together with those used in the above chaotic hydrological applications) in the literature are more appropriate for offline applications than for on-line processing. Kalman filtering and its variants have been successfully used in real-time applications mainly in Controls field where system states have to be estimated from noisy observations. Exploring the applicability of Kalman filtering and investigating the methodologies to further improve, by incorporating noise reduction, the real-time prediction performance is one of the main objectives of the present study.

A major difficulty in using such complex but promising techniques, including artificial intelligence (AI) based prediction techniques such as ANN and SVM, is the increased requirement of computational resources (e.g. memory and time). This is particularly critical in chaos based applications where large number of data records is considered necessary.

1.2.3 Handling of large data sets

Most of the chaos analysis methods are developed with the assumption that the time series are of infinite length. Several guidelines for calculating the minimum number of data record size necessary for estimating some of the system parameters have been formulated (Smith, 1988; Nerenberg and Essex, 1990; Sivakumar et al., 1998); but they are of limited practical applicability. No such guidelines are available

to determine the sufficient amount of data for prediction purpose. However, in chaotic dynamical systems approach, since future is predicted from past experience, it seems that a large data size is a necessity. Therefore, it is generally believed that the larger the data size considered the better is the prediction. Whether all these data contribute valuable information for prediction is an unanswered question. One of the major difficulties in using a large number of past records is the increased computational time and resources. This is emblematic of sophisticated tools such as Neural Networks, Support Vector Machines etc. The computational time and effort needed to train such models increase at least quadratically (Collobert et al., 2002) with the number of past records. Therefore, a method that can extract only a small set of representative data from raw data is desired.

Recently Liong and Doan (2002) applied a clustering technique, Subtractive Clustering Method (Chiu, 1994), with General Regression Neural Network to extract an effective and efficient data set from multivariate data. They were able to extract as small as 47 patterns, out of a total of 467 patterns of multivariate Bangladesh water level data, which yield similar performance as that when the entire set of patterns is considered. Their results indicate that the hydrological data may contain large amount of less informative data; thus, it may be possible to derive smaller and yet representative data sets from a very large data record. The applicability of such methods in extracting representative data sets from chaotic time series is a part of the present study.

1.3 OBJECTIVES OF THE STUDY

The chaotic dynamical systems approach has been gaining increased popularity in the analysis and prediction of chaotic hydrological and meteorological time series.

Many time series come with a very large past record which can impede, or even prohibit, the computational process. Therefore, the prediction applications have thus far been limited mainly to local models due to their simplicity in implementation. Developing methodologies for prediction improvement has been very much in the recent research interest. Thus, the present study focuses on the followings: (1) detailed performance comparison between the global prediction models and local prediction models, (2) to find means of incorporating noise reduction techniques in prediction improvement schemes, and (3) to investigate means of extracting system representative smaller sets of data from long data records.

The objectives of the study are as follows:

(1) To assess the performance of ANN over the widely used local prediction techniques.
(2) To compare the performance of SVM and that of ANN
(3) To investigate the appropriateness of Kalman filtering techniques in improving the prediction performance of noisy chaotic time series.
(4) To propose a noise reduction methodology for real-time predictions of chaotic time series.
(5) To investigate the possibilities of data extraction and develop methodologies to derive system representative data from long chaotic time series data.

This study considers two river flow time series in the analysis. However, all the techniques and the methodologies are first tested and applied to a known noise-free chaotic Lorenz series and then to the same series contaminated artificially with some known noise levels.

1.4 ORGANIZATION OF THE THESIS

Chapter 2 discusses basics of chaos and the chaotic time series analysis. It also reviews the chaos based applications in hydrology with respect to prediction, noise reduction and problems with large data records.

Chapter 3 first introduces the data considered in the study. It then presents a detailed comparison between ANN and local prediction models. A performance comparison between SVM and ANN is also included.

Chapter 4 first investigates some means of improving prediction accuracy of noisy chaotic time series. It then introduces the Extended Kalman filter for chaotic time series analysis. A noise reduction scheme for real-time prediction applications is then proposed and demonstrated.

Chapter 5 demonstrates the possibility of extracting smaller sets of system representative data from chaotic time series using Subtractive Clustering Method (Chiu, 1994). A new, much simpler clustering technique is then developed for the data extraction purpose. Application of the technique on the proposed noise reduction scheme is then presented.

Chapter 6 draws the conclusions resulting from the current study and gives a number of recommendations for further research.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

The revelation that disorganized and complex-looking behaviour can result from an elementary equation or a simple underlying cause was a real surprise to many scientists. With the recent developments in chaos theory, researchers have looked for the determinism in various random-looking fluctuations from different disciplines such as mathematics, physics, chemistry, biology, physiology, medicine, geology, economics, hydraulics, atmospheric sciences, meteorology etc. Most of such studies provided evidence regarding the existence of chaotic behaviour and the possibility of short term predictions. In hydrology too the processes (e.g. rainfall, runoff etc.), which were once analyzed using linear stochastic approaches, are now analyzed using deterministic chaotic approach. The chaotic dynamical systems approach is gaining popularity in many different fields due to the promise it has shown in short term prediction.

The first few sections of this chapter discuss basics of chaos and the chaotic time series analysis. More emphasis is, however, placed on prediction and related issues. The techniques used in this study such as Artificial Neural Networks, Support Vector Machines and Kalman Filtering are also introduced. Then it reviews the chaos based applications in hydrology with respect to prediction, noise reduction, and the problems related to large data record size and its remedy.

2.2 BASICS OF CHAOS

According to Williams (1997), chaos is sustained and disorderly-looking evolution that satisfies certain special mathematical criteria and that occurs in a

deterministic non-linear system. Chaos theory is the principles and mathematical operations underlying chaos. Real world systems that are deterministic, nonlinear and dynamic are susceptible to chaos. The main characteristic of chaos is its extreme sensitivity to even the slightest variations in initial conditions. This is also known as the "Butterfly effect". Edward Lorenz's famous quote 'the flap of a butterfly's wings in Brazil may set off a tornado in Texas' shows the extent of the sensitivity of the chaotic systems to its initial conditions. Chaos is a relatively young and rapidly developing field, and, at present, chaos is very difficult to identify in real-world data. However, due to the capacity of short-term prediction in chaotic systems, chaotic time series analysis is now becoming popular in various fields.

A breakthrough finding by Takens (1981) paved the way for the chaotic dynamical systems approach for analysis of chaotic time series. Takens (1981) showed that it is possible to reconstruct the dynamics of a chaotic system using only a single variable of the system. This approach has been shown to be very useful in understanding dynamical systems where explicit governing equations are not available but one or few variables can be observed. Before starting the discussion on chaotic dynamical systems approach for time series analysis some key terminologies related to chaos theory are first introduced.

Attractor. An attractor is a dynamical system's set of stable conditions (Williams, 1997). When reconstructed on a phase space, an attractor shows a system's long-term behaviour.

State Space Reconstruction. A state space is defined as the multi-dimensional space whose axes consist of variables of a dynamical system. When the state space is reconstructed from an observed time series data, it is called a phase space. There are many methods to reconstruct the state space; the time delay coordinate method is

currently the most popular choice. Packard et al. (1980) and Takens (1981) described the time delay coordinate method to approximate the state space from a scalar time series. According to the method, the phase space vector X_i can be expressed as

$$X_i = \left(x_i, x_{i-\tau}, ..., x_{i-(m-1)\tau}\right) \quad (2.1)$$

where x_i is the observed value at time $t_i = i\Delta t$, Δt is the sampling interval, m is the embedding dimension and τ is the time delay. Note that the actual time delay in physical units is $\tau(\Delta t)$.

Embedding Dimension (m)**.** The lowest dimension, which unfolds the attractor so that none of the self overlaps of the orbit remains, is called the embedding dimension.

Time Delay (τ). Time delay is a suitable multiple of the sampling time Δt.

2.3 ANALYSIS OF CHAOTIC TIME SERIES

Analysis of chaotic time series may be divided into three main phases: (1) System characterization; (2) Determining phase space parameters for prediction; and (3) Predicting the time series. Basically, system characterization investigates whether a time series is chaotic. System characterization may also include the determination of the number of degrees of freedom of the system and the extent of predictability of the system etc. In prediction, the future is predicted with the assistance of observed past patterns of the system. Both system characterization and prediction require the reconstruction of the phase space. This requires the phase space parameters, embedding dimension and the time delay. There are two main approaches to determine these parameters: (1) the standard approach and (2) the inverse approach. The identification, phase space parameter determination and prediction of chaotic time series are discussed in the following sections.

2.3.1 System characterization

If the mathematical formulation of a system is available, recognizing chaotic behaviour is relatively easy. Since the evolution is deterministic, broadband power spectra would be sufficient to identify chaos. However, for real world systems (e.g. runoff, rainfall etc.), whose governing equations and the total number of variables are not known exactly, Fourier analysis alone is not sufficient to indicate chaos since random series also have broadband power spectra. This has resulted in the emergence of wide variety of methods. Among these methods, the most popular ones are: (1) the correlation dimension method (e.g., Grassberger and Procaccia, 1983a, b; Termonia and Alexandrowicz, 1983; Theiler, 1987); (2) the Lyapunov exponent method (e.g., Wolf et al., 1985; Eckmann et al., 1986); (3) the Kolmogorov entropy method (e.g., Grassberger and Procaccia, 1983c); (4) the nonlinear prediction method (e.g., Farmer and Sidorowich, 1987; Casdagli, 1989; Sugihara and May, 1990; Tsonis and Elsner, 1992), including deterministic versus stochastic (DVS) diagram (e.g., Casdagli, 1991); (5) the surrogate data method (e.g., Theiler et al., 1992; Schreiber and Schmitz, 1996); and (6) the method of redundancy (e.g., Palus, 1995; Prichard and Theiler, 1995).

Out of the methods listed the correlation dimension method, also called the correlation integral analysis (CIA), is the prime and most widely used chaos identification method. Almost all the studies on investigating chaos in meteorological and hydrological time series have used the correlation dimension method. Dimension of an attractor is a measure of complexity of the system. Correlation dimension is an estimate to the dimension of an attractor. A finite value in correlation dimension is considered as an indication of the system being deterministic. If the value is low and non-integer it is taken as an indication of chaos. There is more than one algorithm for the computation of the correlation dimension of a time series (e.g. Grassberger and

Procaccia, 1983 a, b; Termonia and Alexandrowicz, 1983; Thieler, 1987). However, Grassberger-Procaccia algorithm (Grassberger and Procaccia, 1983 a, b) is the widely used algorithm. The Grassberger-Procaccia algorithm for correlation dimension calculation and for chaos identification is given in Appendix A.

There are, however, limitations and concerns about the application of correlation integral analysis for chaos identification (e.g. Grassberger et al., 1991). Such problems, however, are not limited only to the correlation dimension method. Such difficulties are reported in various studies in chaos identification. (e.g. Jayawardena and Lai, 1994). Due to these limitations in chaos identification methods, a series of debates were observed in 1980's and 1990's over the claims that certain phenomena were chaotic (e.g., Nicolis and Nicolis, 1984; Grassberger, 1986; Nicolis and Nicolis, 1987; Theiler, 1986; 1990; Osborn and Provenzale, 1989; Lorenz, 1991; Jayawardena and Lai, 1994; Pasternack, 1999; Liaw et al., 2001). More recent understanding is that it is more fruitful to identify the most suitable method for the analysis of a certain phenomena rather than establishing whether a system is chaotic or not. According to Kantz and Schreiber (2004) " … we think that nonlinear techniques (chaos based) can be very useful in situations where determinism could not be established. We feel that people have already spent too much of their time trying to find an answer to the question 'is it chaos or is it noise?' Often it is much more fruitful to ask which is the most *useful* approach for a given experiment. That a data set *is* stochastic to some degree does not mean that we have to use stochastic methods exclusively. In particular, we are not obliged to give up when no 100% appropriate method is available …". With this understanding, once a time series is identified as may be better modelled with nonlinear chaotic dynamics techniques the next step is to

determine the phase space parameters, embedding dimension and time delay, to reconstruct the time series in phase space.

2.3.2 Determination of phase space parameters

Reconstructing the phase space using the time delay coordinate method needs the estimation of the phase space parameters: embedding dimension (m) and time delay (τ). The two approaches for chaotic time series analysis, the standard approach and the inverse approach, differ in the way they determine these phase space parameters needed for prediction. The standard approach determines phase space parameters using the criteria with theoretical insight while the inverse approach finds the optimal phase space parameters with minimum prediction error as the target. The two approaches are described below.

2.3.2.1 Standard approach

In the standard approach, the phase space parameters are determined by incorporating the knowledge gained in the system characterization stage. In system characterization one determines the dimension of the attractor, which gives the degrees of freedom of the system. Once the dimension of the system is known several guidelines are available to determine an embedding dimension (m). According to the embedding theorem of Takens (1981), for a dynamic system with dimension d, an embedding dimension, m ($m>2d+1$) is adequate for phase space reconstruction. However, Farmer and Sidorowich (1987) and Abarbanel et al. (1990) suggested that an embedding dimension just greater than the attractor dimension ($m>d$) is sufficient.

From a mathematical point of view, time delay (τ) is arbitrary. Therefore, there exists no rigorous way of determining its optimal value, and it is even unclear what properties the optimal value should have (Kantz and Schreiber, 2004, p.148). However, for limited and noisy real data, an arbitrary selection of time delay may not

produce good phase space reconstruction. There is little information if time delay is too small; on the other hand, if it is too large all relevant information is lost since the neighboring trajectories will diverge. Therefore, selection of an optimum time delay is important. In the literature, at least a dozen different methods have been suggested as to how to estimate τ. All these methods yield optimal results for selected systems only, and perform just as average for others. Out of these methods the two widely used methods are: (1) Autocorrelation function method; (2) Average mutual information method. According to Holzfuss and Mayer-Kress (1986), time delay can be chosen as the value where the autocorrelation function first crosses the zero line. Other approaches consider the delay time at which the autocorrelation function attains a certain value; say 0.1 (Tsonis and Elsner, 1988), 0.5, or 1/e (Schuster, 1988). Since autocorrelation function measures the linear dependence, Frazer and Swinney (1986) suggested the use of the first local minimum of the mutual information for the choice of time delay.

As seen above, there is no single accepted criterion to select phase space parameters in standard approach. Also, the methods do not work well with all the time series. Therefore, inverse approaches have been proposed to determine these parameters.

2.3.2.2 Inverse approach

Since there is no single accepted criterion for determining phase space parameters in standard approach, various researchers have suggested the use of inverse approaches to determine phase space parameters. Casdagli (1989) was the first to propose an inverse approach to construct a robust predictive model directly from time series data. In his words, "The standard problem in dynamical systems is, given a nonlinear map, describe the asymptotic behaviour of iterates. The inverse problem is,

given a sequence of iterates, construct a nonlinear map that gives rise to them". Casdagli et al. (1991) and Gibson et al. (1992) have highlighted the advantage of using prediction accuracy as a useful criterion in practical state space reconstruction. The authors suggested that state space parameters are not constant and the embedded window width ((m-1)τ) should be adjusted to achieve an optimum balance between noise amplification and estimation error.

Babovic and Keijzer (1999) used an inverse approach to obtain phase space parameters, which produced the best predictions of discharge from river Luznice (Czech Republic), from a wide range of values of the embedding dimension, delay time and the number of nearest neighbors. In a more recent paper, Babovic et al., (2000) employed a genetic algorithm to evolve an embedding that would produce the most accurate forecast of water level at Punta Della Salute (Venice, Italy). Recently, a practical inverse approach was proposed by Phoon et al. (2002) to determine optimal phase space parameters. In this approach the phase space parameter determination and the prediction was achieved in a combined step. Phoon et al. (2002) determined the optimal values of the parameters embedding dimension (m), time delay (τ), and number of nearest neighbours (k) simultaneously which: (1) yields the lowest prediction error; and (2) hopefully carries the signature of chaos in the time series. They argued that since high prediction accuracy is, in general, the primary motivation for developing engineering models, the proposed approach was logical and they demonstrated that the proposed inverse approach yields a set of (m, τ, k) with higher prediction accuracy, as expected, than that resulting from its counterpart, the standard approach. Phoon et al. (2002) used an exhaustive (brute force) approach to select the optimal parameter set out of a range of possible combinations. In a more recent study Liong et al. (2005) showed that a micro-genetic algorithm search engine was more

robust and achieved global optimum with much less evaluations compared to that of the brute force search.

2.3.3 Prediction

Chaotic time series have a short-term predictability. Therefore, once a time series is identified as chaotic or a nonlinear chaotic approach is considered more appropriate for the analysis of the time series, one can make an attempt to forecast the time series (Casdagli, 1989; Sugihara and May, 1990; Tsonis and Elsner, 1992). In phase space prediction, the basic idea is to set a functional relationship between the current state X_t and future state X_{t+T} in the form

$$X_{t+T} = f_T(X_t) \tag{2.2}$$

where T is referred to as lead time. At time t, X_t and X_{t+T} are the current and the future phase space vectors as defined in Eq. 2.1. For a chaotic system, the predictor $\hat{f}_T$ which approximates f_T is necessarily nonlinear. There are two strategies to obtain $\hat{f}_T$: (1) local approximation, and (2) global approximation. In global approximation a function F_T, which is valid over the entire state space is approximated. Neural networks, polynomial and rational function etc. can be used as global approximators. On the other hand, the local approximation subdivides the domain of the attractor into many subsets each of which identifies some approximation F_T^i valid only in that subset. The set of all F_T^i functions constitute the F_T for the case of local approximation. Local averaging technique and local polynomial technique are the widely used local approximators. The local models (Farmer and Sidorowich, 1987; Casdagli, 1989; Sugihara and May, 1990) have been widely used due to their simplicity. With recent developments in computer field, global models such as Artificial Neural networks

(ANN) are also emerging as alternative techniques. These prediction techniques are discussed in the following sections.

2.3.3.1 Local Approximation: Averaging and polynomial models

In local approximation, only the states near the current state are used to make prediction. To predict a future state X_{i+T}, an Euclidean metric is imposed on the phase space to find the k nearest neighbours of the current state X_i. Once the nearest neighbours are found, one can project each of these states X_n to their respective future states X_{n+T}, and construct a local predictor using this group of future states. A local predictor can be constructed in several ways. Among them, the averaging technique (Farmer and Sidorowich, 1987; Casdagli, 1989; Sugihara and May, 1990) is the most popular way. Here, the estimate to future state $\hat{X}_{i+T}$ is calculated as

$$\hat{X}_{i+T} = \left(\sum_{n=1}^{k} X_{n+T} \right) / k \tag{2.3}$$

Another way of constructing a local predictor is to use local polynomials (Abarbanel, 1996). Here, local maps are formed for each state vector in the training set as

$$X_{k+1} = F_k(X_k) \tag{2.4}$$

where

$$F_k(X_k) = \sum_{m=1}^{M} \mathbf{c}(m,k)\phi_m(X_k) \tag{2.5}$$

and $\phi_m(X)$, $m = 1,2,\ldots,M$ are polynomial basis functions. The coefficients of the model ***c(m,k)*** can be determined by a least squares fit by minimizing $\sum_{r=1}^{NB} \left\| X_{k+1}^r - F_k(X_k^r) \right\|^2$ where $X_k^r, r = 1,2,\ldots,NB$ are the nearest neighbours of X_k and X_{k+1}^r are their corresponding one-step ahead states. When a new point Z_0 is given, its

nearest neighbour in training set X_j is found. Then the one-step evolution of Z_0 is found as

$$Z_1 = F_j(Z_0) \tag{2.6}$$

then the nearest neighbour to Z_1 in the training set is found and the procedure is repeated until the desired prediction horizon is reached.

2.3.3.2 Global Approximation: Artificial Neural Network (ANN)

Artificial Neural Network (ANN) is one of the major tools that form the core for developing intelligent systems. It is motivated by the recognition that the human brain computes in an entirely different way from the conventional digital computer. Artificial Neural Networks are now very popular in many different fields and are primarily used to solve two kinds of problems: (1) pattern recognition or classification problems and (2) regression or function approximation problems. The present study focuses on the use of ANN for function approximation (regression) purpose.

By definition, an artificial neural network is a massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experimental knowledge and making it available for use (Haykin, 1999). Artificial neural network does not need any priori knowledge of the actual physical processes and given that there is an exact relationship between input and output data, the ANN can be 'trained' to 'learn' that relationship. A neural network is characterized by its architecture. A typical ANN consists of a number of nodes that are organized according to a particular arrangement. Any non-cyclic arrangement of neurons (the basic building blocks of neural network), where the information flow begins with the inputs of the problem and ends at the outputs of the problem, is referred to as a Multi Layer Perceptron (MLP). However, for the ease in programming, a particular

arrangement where neurons are arranged in layers without any crisscross connection between non-successive layers is very popular and most widely used. These Multi Layer Perceptrons, also called as Sigmoidal Networks, Feed forward Neural Networks and Back-propagation Networks, have been found to provide excellent performance with regard to input-output function approximation and pattern recognition. Hence, more than 90% of the applications that use neural networks are based on MLPs. Although an MLP may have many layers, it has been proved that even MLPs with a single hidden layer have universal approximation capability (Haykin, 1999).

2.3.3.3 Global Approximation: Support Vector Machine (SVM)

Support vector machine and its related theory have been developed over the last 30 years. The algorithms in its present form, however, have been developed by V. Vapnik and his co-workers firstly for pattern recognition problem in 1992 and then for regression problem in 1997. Currently, the Support Vector Machines is the main competitor to Artificial Neural Networks in both pattern recognition and regression applications. Support Vector Machine algorithm is developed based on statistical learning theory. One of the main insights in statistical learning theory is that in order to obtain a small risk (error on unseen data) one needs to control both training error and model complexity by explaining the data with a simple model. This is achieved in SVM through the structural risk minimization inductive principle (Vapnik, 1999).

In Support vector machine, the analysis is performed in a high dimensional space called feature space rather than in the data space. The crucial ingredient facilitating this is the so-called Kernel-trick, which permits the computation of dot products in high dimensional feature spaces using simple functions defined on pairs of input patterns. Kernels that satisfy the conditions given by Mercer's theorem may be used in the Kernel-trick. Mercer's theorem lists the conditions required for a

symmetric continuous function $K(\mathbf{x}, \mathbf{x}')$ defined in a closed interval $\boldsymbol{a} \le \boldsymbol{x} \le \boldsymbol{b}$ and $\boldsymbol{a} \le \boldsymbol{x}' \le \boldsymbol{b}$ to have an expansion,

$$K(\boldsymbol{x}, \boldsymbol{x}') = \sum_{\boldsymbol{k}=1}^{\boldsymbol{N_F}} \lambda_{\boldsymbol{k}} \boldsymbol{\phi_k}(\boldsymbol{x}) \boldsymbol{\phi_k}(\boldsymbol{x}') \tag{2.8}$$

where $\lambda_{\boldsymbol{k}} \ge 0$ and $\boldsymbol{N_F} \le \infty$

Let us consider a training data set $(\boldsymbol{x}_i, y_i), i = 1, 2, \ldots N$ where $\boldsymbol{x}$ is the input vector with dimension n, and *y* is the corresponding output vector with dimension 1. In regression, one tries to estimate $f(\boldsymbol{x})$:

$$y = f(\boldsymbol{x}) + \upsilon \tag{2.9}$$

where $f(\boldsymbol{x})$ is the conditional expectation $E[y|\boldsymbol{x}]$ and υ is a random expectation error that represents the 'ignorance' about the dependence of *y* and $\boldsymbol{x}$. In SVM, the input space $\boldsymbol{x}$ is transformed to a higher dimensional space through $\varphi(\boldsymbol{x})$. $\{\phi_j(\boldsymbol{x})\}$ is called the feature space. The nonlinear basis functions $\{\phi_j(\boldsymbol{x})\}$ convert the original function $f(\boldsymbol{x})$ into linear functions in the feature space:

$$y = \sum_{i=0}^{m} w_i \phi_i(\boldsymbol{x}) = \boldsymbol{W_0}^T . \varphi_0(\boldsymbol{x}) = \boldsymbol{W}^T . \varphi(\boldsymbol{x}) + b \tag{2.10}$$

where *m* is the number of basis functions where *b* is a scalar constant. The beauty of SVM formulation is that one can determine $\boldsymbol{W}^T . \varphi(\boldsymbol{x})$ in Eq. 2.10 without using the functions $\phi_i(\boldsymbol{x})$explicitly. These calculations in feature space are dealt with kernel trick using only the input patterns. The SVM regression formulation given in Eq. 2.10 that uses a single expression to describe the total data space is a global approximation method. However, some local nature can also be introduced into this global approximation method through the use of kernels. Derivation of SVM using kernel functions will be explained in detail in Chapter 3.

The goal of the learning machine is to find a function f with a small risk (or test error). With the insight of statistical learning theory, SVM minimizes the regularized risk functional,

$$\frac{1}{2}\|\boldsymbol{W}\|^2 + C.R_{emp}[f] \qquad (2.11)$$

in order to obtain a simple model with low training error. $R_{emp}[f]$ represents the training error and C is a constant determining the trade off with the complexity penalizer $\|\boldsymbol{W}\|^2$. Different loss functions may be used to denote $R_{emp}[f]$, e.g. Hubbers loss function, Laplacian loss function etc. The ε-insensitive loss function devised by Vapnik (1999) is the most popularly used. It has a possibility to produce a sparse representation of data. Once the $R_{emp}[f]$ is determined with a suitable loss function the unknowns of Eq. 2.10 can be determined by minimizing Eq. 2.11 and the expression can be used to make future predictions.

Noise deteriorates the performance of any kind of prediction model regardless of whether local approximation or global approximation. Therefore, noise reduction is a very important aspect in any kind of analysis of data contaminated with noise. The next section discusses some noise reduction techniques used in time series analysis.

2.3.4 Noise Reduction

2.3.4.1 Introduction

Noise hampers identification of chaos. Also, noise is the most prominent limiting factors for the predictability of deterministic systems (Kantz and Schreiber, 2004). Therefore, noise removal is a subject of utmost importance.

By definition, *noise* is the unwanted part of the data (Kantz and Schreiber, 2004). All experimental data are contaminated by noise to certain extent. There are two

types of noise: (1) measurement noise, and (2) dynamical noise. Measurement noise refers to the corruption of observations by errors which are independent of the dynamics. Real errors of measurement (can be accidental or systematic) and other dynamics simultaneously present which superimpose themselves on the one under investigation are included in measurement error (Porporato and Ridolphi, 1997). When dynamics are given by

$$\boldsymbol{x}_{n+1} = F(\boldsymbol{x}_n) \tag{2.12}$$

and the scalar measurements are

$$y_n = s(\boldsymbol{x}_n) + v_n \tag{2.13}$$

where v_n are random numbers and $s(\boldsymbol{x})$ is a smooth function that maps points on the attractor to real numbers, the series $\{v_n\}$ is referred to as the *measurement noise*. The aim of the noise reduction techniques is to reduce measurement noise. *Dynamical noise* is a feedback process where the system is perturbed by small random fluctuations at each time step:

$$\boldsymbol{x}_{n+1} = F(\boldsymbol{x}_n + \boldsymbol{w}_n) \tag{2.14}$$

Noise reduction is about decomposition of every single time series value into two components, one of which supposedly contains the signal and the other one contains random fluctuations. Thus, it is assumed that the data can be thought of as an additive superposition of two different components which have to be distinguishable by some objective criterion. The basic problem with any noise reduction scheme is that we have to assume that a superposition into noise plus signal is meaningful. This is specially the case with real-world applications where true signal is unknown. Also, no clear-cut criterion for the amount of noise to be subtracted can be given in such cases and no performance measure is available. Therefore, the robustness of the method is considered as of utmost importance (Grassberger et al. 1993). A number of nonlinear

noise reduction techniques have been proposed in Chaos literature. Although not much experimented in chaos literature, the Kalman filtering and its variants are very successful in dealing with noisy data, in Controls theory. A discussion on the nonlinear noise reduction techniques and the Kalman Filtering techniques is given in the following sections.

2.3.4.2 Nonlinear Noise Reduction

Nonlinear noise reduction is closely related to forecasting. It is assumed that the data follows a deterministic evolution rule

$$x_{n+1} = F\left(x_{n-m+1}, \ldots, x_n\right) \tag{2.15}$$

but is measured with some uncertainty ν

$$y_n = x_n + \nu_n . \tag{2.16}$$

For convenience, delay is assumed to be unity here. The dynamical equations *F* have to be estimated. A forecast of a future value $\hat{y}_{n+1}$ can be made using estimated map $\hat{F}$. The aim of the noise reduction is to construct a cleaned data sequence $\hat{x}_n$, $n = 1,\ldots,N$ which is close to what we have measured but more consistent with the estimated dynamics, i.e.,

$$\hat{x}_{n+1} = F\left(\hat{x}_{n-m+1}, \ldots, \hat{x}_n\right) + \nu'_n \tag{2.17}$$

where $\left\langle \nu'^2 \right\rangle$ is the remaining discrepancy from the dynamical equations which should be smaller than the noise ν in Eq. 2.16. At a first glance, it may appear that all one needs it a good estimate $\hat{F}$ to use in Eq. 2.17 to obtain noise reduced data. However, it is not feasible due to two reasons: (1) Good estimates of $\hat{F}$ are not always available and virtually impossible for real data; (2) even if one has a good estimate $\hat{F}$, due to noise in the data, the estimate $\hat{x}_{n+1} = F\left(y_{n-m+1}, \ldots, y_n\right)$ can be even noisier than before.

This “noise amplification” is due to the sensitivity of chaotic dynamics to variations in initial conditions.

A number of different nonlinear noise reduction techniques have been proposed in chaos literature (e.g. Hammel, 1990; Kostelich and Yorke, 1990; Schreiber and Grassberger, 1991; Cawly and Hsu, 1992; Sauer, 1992; Schreiber, 1993 etc.). The different noise reduction methods differ in the way the dynamics are approximated, how the trajectory is adjusted and how the approximation and the adjustment steps are linked to each other. Some dangers in these nonlinear filtering methods (if carelessly applied) and the precautions to be taken to avoid such dangers are given in Mees and Judd (1993). Some nonlinear noise reduction algorithms cater for the uncommon case where the dynamical equations are known. Apart from these, most other methods reduce noise by a similar amount and their performance do not differ dramatically (Kantz and Schreiber, 2004). Therefore, the major criteria for the preferred algorithm is robustness, ease of implementation as well as the resources needed (time and memory). Kantz and Schreiber (2004) have found that the widely used algorithm, simple nonlinear noise reduction method, and an algorithm called locally projective scheme to be reliable and effective on a broad variety of data sets including artificial and real data. The simple nonlinear noise reduction method is used in this study.

2.3.4.3 Kalman Filtering

The Kalman filter (Kalman, R.E., 1960) is the most well-known and often used tool to estimate the system states from noisy measurements. It is the optimal linear filter in the sense of minimizing the variance of the estimation error. It uses observed measurements from a dynamical system to make more accurate estimates of the system’s states (Figure 2.1). Kalman filter is comprised of a set of equations that

describe the system and how the observations can be optimally blended to make better system estimates.

Since the time of its introduction in 1960 by Rudolf Kalman, the Kalman filter has been the subject of extensive research and applications especially in the area of autonomous or assisted navigation. Originally, the Kalman filter was proposed for linear processes. Most of the real world processes are, however, nonlinear. Therefore, a number of suboptimal nonlinear variations have appeared. The extended Kalman filter (EKF) (e.g. Maybeck, P. S., 1979) is the direct extension of linear Kalman filter for nonlinear systems through local linearization of system and observation models. For highly nonlinear systems EKF may be ineffective. Second order extended Kalman filters have also been proposed in the literature. Recently, a new variation called Unscented Kalman Filter (UKF) has been proposed to deal with nonlinear systems (e.g. Julier and Uhlmann, 1996; 1997). This method has been shown to perform better than EKF (Wan and van der Merwe, 2000). The more recent square root unscented Kalman filter is an improvement of UKF (van der Merwe and Wan 2001). Thus, the Kalman filter literature is rich with techniques to deal with nonlinear systems.

2.4 PREDICTION OF CHAOTIC HYDROLOGICAL TIME SERIES

Investigations on the presence of chaotic dynamics in hydrological time series and their prediction have covered a wide variety of systems and applications. In addition to the major applications: (1) river flow analysis (e.g. Jayawardena and Lai, 1994; Porporato and Ridolfi 1996; 1997; Babovic and Keijzer 1999; Liu et al., 1998; Jayawardena and Gurung 2000; Lisi and Villi, 2001; Islam and Sivakumar, 2002); and (2) rainfall analysis (e.g. Rodriguez-Iturbe et al., 1989; Sharifi et al., 1990; Tsonis et al., 1993; Islam et al., 1993; Sivakumar et al., 1998). Examples from other applications

are: biweekly volume time series of the Great Salt Lake (Sangoyomi et al. 1996 and Abarbanel et al. 1996); hourly water level data from Venice lagoon (Zaldivar et al. 2000); flood series in the Huaihe river basin in China (Zhou et al. 2002); and suspended sediment concentration in the Mississippi river (Sivakumar, 2002). Out of all such applications, chaotic analysis of river flow is the most popular since it has shown lots of potential in short term prediction.

Jayawardena and Lai, (1994) were the first to investigate the existence of chaos in river flow time series. Comparing the prediction accuracies of two daily stream flow time series in Hong Kong, they reported that there is convincing statistical evidence to believe that the stream flow data series could be better modelled by the chaotic dynamical systems approach than by the traditional linear ARMA (Auto Regressive Moving Average) approach. In a much more rigorous recent study, Jayawardena and Gurung (2000) also reported superior prediction performance with dynamical systems approach compared to linear stochastic approaches in several hydrological time series. Lisi and Villi, (2001) analysed daily discharge records of Adige river in Italy. By comparing prediction accuracy of a nonlinear model with a classical linear model they reported that chaotic modelling can be an effective method to improve prediction. Apart from these Porporato and Ridolfi (1996, 1997) and Islam and Sivakumar, (2002) among others have applied chaos in river flow time series and observed good quality predictions.

The earlier investigations of chaos placed greater emphasis on identification of chaotic dynamics in the underlying systems; the more recent studies, on the other hand, place more emphasis on applications of chaos based knowledge to yield high prediction accuracy (Porporato and Ridolfi, 1997; Sivakumar et al., 1999a; Phoon et

al., 2002), noise reduction (Schreiber and Grassberger, 1991; Sivakumar et al., 1999b), estimation of missing data (Elshorbagy et al., 2002a) etc. Almost all the chaos applications in hydrology have appreciated the potential of phase space based methods in short-term prediction (e.g. Porporato and Ridolfi, 1997; Sivakumar et al., 1999a; Phoon et al., 2002). Improving prediction accuracy of phase space models by various techniques (e.g. by pre-processing raw data, using inverse approaches for finding optimal embeddings for prediction) is the most recent trend in chaos applications. Several recent studies have used pre-processed data to enhance system identification and, more importantly the prediction accuracy. Some of the pre-processing techniques that have commonly been used are: (1) noise removal (e.g. Porporato and Ridolfi, 1997; Kawamura et al. 1998; Sivakumar et al. 1999b; Jayawrdena and Gurung, 2000); (2) filtering low/high frequency components (e.g., Porporato and Ridolfi, 1996); (3) differencing: First differencing and Seasonal differencing (e.g. Sugihara and May, 1990; Provenzale et al. 1992; Jayawardena and Lai, 1994; Porporato and Ridolfi, 1996, 1997; Yu et al., 2004); and (4) interpolation (e.g. Porporato and Ridolfi, 1997). Phoon et al (2002), Babovic et al (2000), among others, have used inverse approaches to determine optimal embeddings for phase space reconstructions so that the prediction accuracy is maximized.

However, most of the above refinements are applied mainly on the simple local phase space models due to its simplicity in implementations although it is not known whether these local models give best predictions. Use of local prediction models inevitably results in improvements which are biased by the number of nearest neighbours, a parameter, which is independent of the dynamics of the system. This constraint could be circumvented by global phase space models such as artificial neural network, which approximate dynamics over the entire attractor. There is a

general understanding, that local approximation can give better predictions than global approximation in phase space prediction of chaotic time series (e.g. Porporato and Ridolfi, 1996; 1997; Islam and Sivakumar, 2002). However, it is interesting to investigate how the widely used global models perform compared to the widely used local models. In hydrology, the number of studies that has made comparisons between local and global phase space prediction models for their prediction performance is very limited. Outside chaos applications, the superiority of the global model, artificial neural network (ANN), to model nonlinear dynamics in hydrological time series has been observed by a number of studies (e.g., Karunanithi et al., 1994; Hsu et al., 1995; Zealand et al., 1999; Elshorbagy et al., 2000; ASCE, 2000). However, only very few studies (e.g. Elshorbagy et al., 2002a) have used ANN as a global phase space prediction model in chaotic river flow time series prediction.

Elshorbagy et al. (2002a) utilized both ANN (multilayer perceptrons) and popularly used local model (averaging technique) for estimating missing stream flow data in English river, Canada. They reported superior performance of ANN over the local (averaging) nearest neighbour models. However, comparison between local and global models was not the main focus of their study. They considered a single set of phase space parameter values (embedding dimension and time delay), which suited their problem of estimating missing data, in both prediction models. These particular parameters may, however, not be optimal with respect to both prediction models. The authors also acknowledged that the superiority of ANN may be problem dependent. Therefore, a conclusive statement of local models and global ANN models in chaotic time series prediction could not be made by this study.

Sivakumar et al. (2002) made a comparative study on phase space reconstruction approach (with local polynomial models) and ANN (multilayer

perceptrons) considering them as two different black-box models. Their results on Chao Phraya river flow (known to exhibit chaotic dynamical behaviour) prediction (1 day and 7 days ahead predictions) showed much worse predictions with ANN. Their prediction errors of ANN models were more than 4-8 times the error of local models in terms of mean absolute error measures. They believed that the superiority of local phase space prediction approach was due to the capability of local approximation methods to better capture the chaotic dynamics of a system as opposed to the global approximation method. However, they acknowledged that one has to be careful in interpreting their results due to some concerns they had, among others, selection of data for training and test sets. Thus, the above studies are not conclusive and, furthermore, they showed contradictory results. It should be noted, however, that the local models used by the two studies were not exactly the same; Elshorbagy et al. (2002a) used local averaging of nearest neighbours while Sivakumar et al. (2002) used local polynomial models. Due to these reasons further studies on performance of global ANN models compared to local models in chaotic time series prediction is necessary and timely.

With any kind of prediction model, the noise in data deteriorates the prediction performance of any deterministic systems. In phase space prediction, where prediction models themselves are data-driven models, the noise in data can cause a considerable negative impact. Chaotic dynamics makes the effect of noise even worse due to its sensitive dependence on the initial conditions. Therefore, noise reduction is of utmost importance. The next section reviews the studies on noise reductions in chaotic hydrological time series.

2.5 NOISE REDUCTION IN CHAOTIC HYDROLOGICAL TIME SERIES

A few attempts have been made on noise reduction in chaotic hydrological time series (e.g. Porporato and Ridolfi, 1997; Kawamura et al., 1998; Sivakumar et. al. 1999b, c; Jayawardena and Gurung, 2000). However, all these applications have been severely criticized later by Elshorbagy et al. (2002b) due to certain shortcomings in the approaches.

Porporato and Ridolfi, (1997) used nonlinear noise reduction techniques to analyse mean daily discharges of Dora Baltea, left tributary of river Po in Italy. They used the algorithm proposed by Schreiber and Grassberger (1991) for noise reduction. They reported better forecasting performance with the noise-reduced series than analysing the original series. They have measured the prediction accuracy of noise-reduced case by comparing predictions against noise-reduced validation data. This is a questionable approach, as will be discussed in detail later, and thus their results are not reliable.

Sivakumar et al. (1999c) commented on the work of Porporato and Ridolfi, (1997). They suspected that over-correction might have occurred in Porporato and Ridolfi's analysis. To avoid the problem of over-correction, Sivakumar et al. (1999b, c) proposed an approach for systematic noise reduction. The approach included additional steps for determination of noise reduction parameters such as the neighbourhood size and the number of iterations. They suggested the use of an initial estimate of level of noise using some noise level estimation method. They used the method of Schouten et al. (1994) for noise level estimation. Method of Schreiber (1993) was used as the noise reduction method. Sivakumar et al. (1999b, c) used prediction accuracy as the diagnostic tool to identify noise reduction and thus,

determining when to stop the noise reduction. When prediction accuracy starts deteriorating noise removal is stopped and the last estimated noise level is taken as the noise level of the data. Like Porporato and Ridolfi (1997), Sivakumar et al. (1999b) also compared their predictions with noise reduced data, which is an inappropriate way of verifying performance of noise reduction as argued by Elshorbagy et al. (2002b).

Elshorbagy et al. (2002b) raised serious concerns about noise reduction applications in hydrology, especially the ones shown by Sivakumar et al. (1999b, c) and Porporato and Ridolfi (1997). Their doubts of general interest are summarized as follows:

(1) They repeatedly commented that the above studies have measured the prediction performance (in verification) by comparing the predicted values against 'noise-reduced' data rather than original data;

(2) They questioned the way the prediction accuracy had been used by the above studies for verifying the noise removal. Elshorbagy et al. (2002b) showed that such a diagnostic step implies that the prediction model perfectly models the underlying dynamics. However, the prediction model used by Sivakumar et al. (1999b) is merely the simple averaging technique suggested by Farmer and Sidorowich (1987). Elshorbagy et al. (2002b) suggested that in order to use prediction accuracy for verifying the noise reduction, either a variety of nonlinear models should be used or a single model that is known to perfectly model the whole phenomenon should be used. They concluded that what is removed by Sivakumar et al. (1999b) approach is the component, which is not modelled by the specific model employed. In other words, the removed part is considered noise to a single model, not the absolute noise.

The doubts raised by Elshorbagy et al. (2002b) above are justified concerns. They applied the approach proposed by Sivakumar et al. (1999b) on English River,

Ontario, Canada with the simple noise reduction proposed by Schreiber (1993). Using two prediction models, local linear model (LLM) and a global artificial neural network (ANN), they verified the doubts they raised. The investigations showed:

(1) The improvement in prediction performance with noise reduced data for the two models, LLM and ANN, were significantly different supporting the claim made that the perception of noise by different models.

(2) When prediction accuracy is measured by comparing the predicted values with noise reduced verification data, unrealistically high prediction accuracy was observed. This showed that comparing predictions with 'noise-reduced' data is improper.

They also made comments on using visual inspection of graphical representations to make implications of noise reduction (of Kawamura et al., 1998 and Jayawardena and Gurung, 2000); however, they were not of interest to prediction applications and are thus not discussed here.

The above investigations show that comparing the predicted values (using a prediction model that does not perfectly model the system) with 'noise-reduced' values to arrive at the prediction accuracy, and thus the effectiveness of noise reduction, is not a correct attempt. A fatal flaw in using 'noise-reduced' validation data, in making future prediction, is that such noise-reduced values are derived with the use of future data as well, which are not available for prediction application usage. Therefore, the approaches used in the above studies, which have used the off-line application of noise reduction (i.e. using both past and future records for noise reduction) in place of where real-time noise reduction (i.e. using only the past records up to the record of interest for noise reduction), are not appropriate for real-time prediction (or future forecasting). Comparing predictions with original data would have kept the noise reduction on a safe track provided again that the input validation data are not noise-reduced with the

use of future data. Also, use of more than one model, as later suggested by Elshorbagy et al. (2002b), could have helped detecting any flaws.

Lack of clear-cut ways to verify noise reduction is an inherent problem with noise reduction applications where the true signal is unknown. Thus, identifying better criteria for such applications is essential. With real world data, such as rainfall and runoff, using prediction accuracy as a criterion seems to be the most reasonable alternative. Prediction accuracy is employed as a criterion not only in noise reduction applications but also in model selection and determination of optimal parameters (e.g. Phoon et al., 2002). However, in real world noise reduction applications where true signal/ dynamics are unknown, prediction accuracy should be used so long as predicted values are compared against original values as noted by Elshorbagy et al. (2002b).

The procedures followed in the applications discussed above can only deal with off-line noise reduction. That is to reduce noise in recorded data. The noise reduction methods used in the discussed studies are meant for off-line applications and are not directly applicable in the real-time prediction. Although prediction is attempted for diagnostic purposes etc, due to the flaws that discussed earlier, the results from the above studies do not represent real-time forecasting. Therefore, proper investigations are necessary.

A practical approach for incorporating noise reduction for real-time forecasting is called for. There is a notion that using noise-reduced data for better configuration of prediction models is an obvious benefit of noise reduction. This may be true, but it is worthwhile investigating whether such better models would actually produce better predictions with noisy inputs. In chaotic systems the predictions could be bad even with a perfect model when the input data is noisy (Kantz and Schreiber, 2004). Thus, a better model may have to be supported with equally good input data for optimal

performance. To clean the input data, noise reduction should be considered on real-time basis.

Kalman Filtering is a promising state estimation technique that is widely popular in Control Theory for real-time state estimation and prediction applications where noisy observations of a system are available. Its nonlinear variants are very popular in both within and outside Controls literature. It is therefore of interest to explore such techniques in chaos analysis as well. Investigating such methods for noise reduction applications have been recommended by researchers such as Walker and Mees 1997 as well.

Walker and Mees (1997) investigated two techniques: (1) Kalman filtering (extended Kalman filter, EKF, with both forward and backward filtering and a variation of Kalman smoothing called non-causal filter); and (2) nonlinear noise reduction algorithm of Hammel (1990) based on the concept of shadowing, for noise reduction in chaotic time series. Like others, their application was also concerned with off-line noise reduction. They used perfect state space models either in the form of: (1) exact governing equations of Henon map and Ikeda map, or (2) a data driven model built from noise-free data. The so-called non-causal Kalman filtering performed best in their applications on chaotic Henon and Ikeda maps; they suggested considering the noise reduction methods favoured by control theorists by the dynamical systems community.

Judd (2003) compared the performance of what they called a gradient descent filter, a method close to nonlinear noise reduction techniques, with EKF for nonlinear state estimation of a chaotic Ikeda map. They reported superior performance of EKF at low levels of observation noise and poor performance at high noise levels compared to

the gradient descent filter. They too used exact governing equations as state space models in EKF.

The above studies show the potential of Kalman filtering techniques in chaos applications. However, none of those studies considered the case with real world data where perfect models are not available. Also no applications are reported in chaos literature incorporating Kalman filtering for real-time applications such as prediction.

Some applications of Kalman filtering have appeared in water resources literature using physically based models. Drecourt (2003) provides an overview of the uses of Kalman filter in hydrological modelling. Lee and Singh (1999) have successfully applied the Kalman filter for parameter estimation in a tank model. Bierkens et al (2001) used the Kalman filter for space –time modelling of water table depth. They concluded that Kalman filtering can be used for on-line forecasting, for instance, weather forecasts. Most recently, Doan and Liong (2004) applied a nonlinear variation called Unscented Kalman filter for river flow forecasting with a data driven ANN model as a state space model. The studies show the applicability of Kalman filter techniques in noise reduction applications of hydrological data. This study hopes to investigate the applicability of Kalman filtering techniques in real-time prediction of chaotic time series.

One of the major problems faced in chaos analysis, be it prediction, noise reduction or other applications, is the large data record size required in its analysis. In the next section, the problem of large data record size in chaos applications with respect to hydrological time series is reviewed, with the hope of investigating a possible scheme to circumvent the problem.

2.6 LARGE DATA RECORD SIZE IN CHAOS APPLICATIONS

With rapid developments in data measuring/recording/storing systems (e.g. remote sensing, GIS, improved measuring devices, automation etc.), data and information are becoming abundant in many fields. In hydrology too, the details and measurements frequency are increasing in the recent years. However, incorporating such resources in analysis such as better model development, forecasting etc. is not easy mainly due to the following reasons: (1) extracting relevant data from vast information is not quite straightforward, and (2) the computational resources requirement to analyse such large record size are very high. Although computer technology has significantly progressed, the actual complexity of the problem being addressed, with more and more data collected and made available, has also gone up. Therefore, devising a scheme to extract the best out of a bulk of information/data is an important task.

Chaos analysis is one of the fields where the use of large past data records is essential. Most of the chaos identification and prediction methods are developed with the assumption that the time series are noise-free and are of infinite length. Several guidelines for calculating the minimum size of data record necessary for estimating some system parameters such as correlation dimension were formulated (Smith, 1988; Nerenberg and Essex, 1990; Sivakumar et al., 1998), but they are of limited practical applicability. No such guidelines are available to determine the sufficient amount of data for prediction purpose. Since the future is predicted from the past experience, it appears that a large data set is a necessity and the norm is to use as much data as possible with the hope of improved performance. Also, in chaos hydrological literature, there is evidence to believe that more frequent data and thus more data records can improve the prediction performance (e.g. Porporato and Ridolfi).

However, the issue, whether each additional data record contributes distinct information has not been addressed in any study.

In chaos applications in river flow, researchers have used data of different record lengths varying from about 1800 records to 15000 records. Most of the studies have used maximum data available for both system characterization and prediction (Babovic and Keijzer, 1999; Liu et al., 1998; Jayawardena and Gurung, 2000; Lisi and Villi, 2001; Sivakumar, 2002). For most of the rivers, flow data is available for about 20-30 years or even longer periods.

One of the major difficulties in using a large data record size in chaos analysis is the heavy burden on the computational time and resources. In prediction case, the problem is particularly so when global models such as ANN and SVM are used, where the time and the computational effort needed to train the model increases more than linearly (at least quadratically in most cases) with the increasing number of past records (Collobert et al., 2002). This inevitably forces researchers, whenever possible, to use smaller data record size. For example, for Tryggevaelde catchment runoff, Phoon et al. (2002) used approximately 6900 data points in their local prediction model while Sivapragasam et al. (2001) used only 3000 data points in Support Vector Machines. Similarly, Sivakumar et al (2002) used less number of data for Chao Phraya river flow with Artificial Neural Network prediction than the number of data used in local prediction model by Jayawardena and Gurung (2002). Therefore, a method that can extract only a small set of representative data, from along data record, is highly desirable.

Clustering is a widely used technique in applications such as classification problems to group data with similar attributes together. When there is a group of data with similar attributes one point is selected to represent the whole group. This point is

called a cluster centre (Figure 2.2). Most clustering techniques are devised to group categorical data. Grouping numerical data for function approximation purposes is not very common. Recently this idea of clustering in function approximation problem is given a theoretical explanation (Kreinovich and Yam, 2000). A few fuzzy clustering methods (e.g. Filev and Yager, 1994; Chiu, 1994) are capable of extracting centres from high dimensional numerical data. They have been applied in function approximation problems in the form of fuzzy model identification.

Chiu (1994) introduced a clustering technique for fuzzy applications, and demonstrated it on: (1) a nonlinear function approximation problem, (2) prediction of chaotic Mackey-Glass time series, and (3) trip (automobile) generation modeling. Rantala and Koivisto (2002), in a study of Neuro-Fuzzy model identification, applied the same clustering technique on chaotic Mackey-Glass time series. On these applications both studies derived small number of fuzzy rules, with the cluster centers derived by the method, which yield comparable results with other methods. However, in those studies, the concern was mainly on fuzzy model identification; little concern was paid to aspects such as: representativeness of the selected data of the whole system; and the applicability of clustered data with other different models.

Liong and Doan (2002) applied the above mentioned subtractive clustering technique (Chiu, 1994) with General Regression Neural Network to extract an effective and efficient data set from multivariate data. There, they were able to extract as small as 47 patterns out of 467 total patterns of multivariate Bangladesh water level data which had similar prediction performance as that of the entire set of patterns. Their results indicate that the hydrological data may contain large amount of less

informative data and, hence, it may be possible to derive a data set of smaller record size with sufficient representativeness of the total.

2.7 SUMMARY

The potential of chaos based methods for short term prediction has received wide appreciation. Improving the accuracy of prediction, of such approaches, is one of the current interests. However, thus far the prediction applications have been mostly limited to local prediction tools mainly due to their simplicity in implementations. Also there is an understanding that local approximation is better than global approximation. However, a much more reliable comparison of widely used local prediction models and global prediction models must be conducted.

In improving the prediction accuracy, in addition to the prediction tool, noise is also one of the most important limiting factors. Several applications of noise reduction have appeared in chaos hydrological literature. However, all of them have been criticised for their inappropriate use of techniques. Incorporating noise reduction for real-time chaos based prediction applications is an area for further exploration. The Kalman filtering techniques from controls literature have been very successful in dealing with noisy data for real-time state estimation applications; and adopting such techniques in the chaos analysis can be an interesting study.

One of the major difficulties in incorporating novel sophisticated techniques in chaos applications is the large amount of data records required in computations. In chaos based predictions it is believed that the more the data used the better the predictions are. However, in general, the computational time and storage capacity of most prediction tools also increases drastically with the number of data patterns. Therefore, investigating methods to extract a small, representative set of data from large data records to circumvent the aforementioned problems is called for.

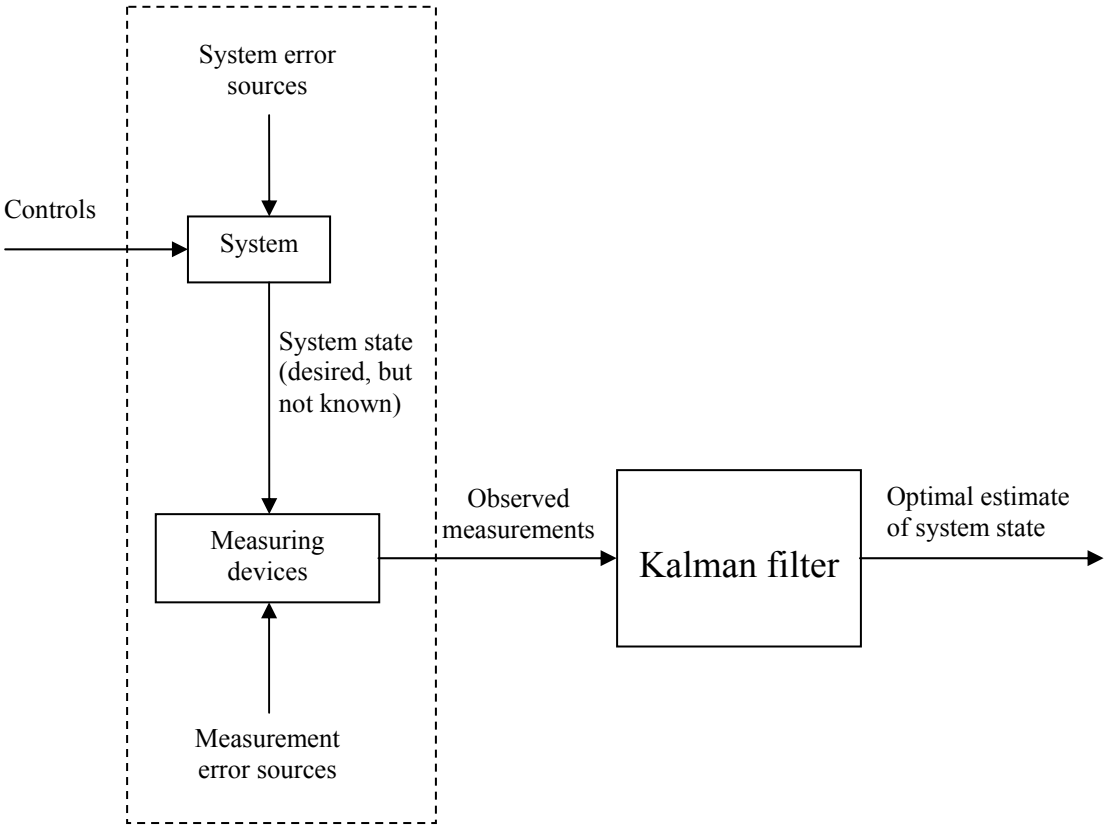


Figure 2.1 Kalman filter application (*Maybeck and Peter, 1979*)

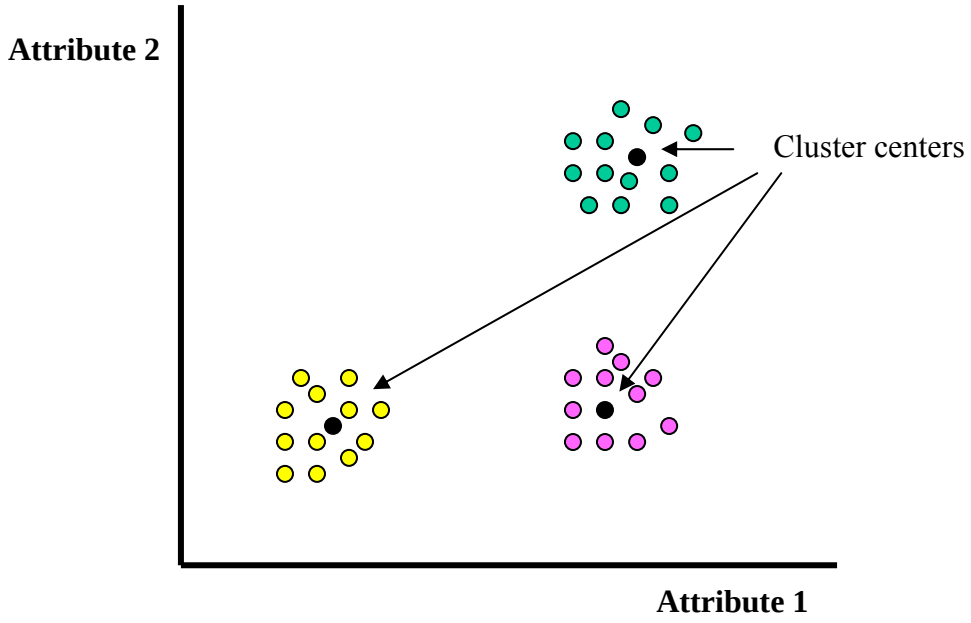


Figure 2.2 Clustering: grouping objects into classes of similar objects

CHAPTER 3

CHAOTIC TIME SERIES PREDICTION WITH GLOBAL MODELS: ARTIFICIAL NEURAL NETWORK AND SUPPORT VECTOR MACHINES

3.1 INTRODUCTION

The potential of short-term prediction in chaos based models has been widely appreciated. However, most of the applications have been confined to simple local phase space prediction models. There is a general understanding (e.g. Porporato and Ridolfi, 1996; 1997; Islam and Sivakumar, 2002), that local approximation can give better predictions than global approximation in phase space prediction of chaotic time series. However, it is of interest to thoroughly investigate how the widely used global models such as ANN perform compared to the widely used local models. This chapter assesses the performance of global models (ANN -multilayer perceptrons and SVM) as opposed to the widely used local models (both the averaging technique and the local polynomials) in phase space prediction. A performance comparison between ANN and SVM is also made. The analysis is first performed with ANN and the local models on a noise-free chaotic Lorenz series; this helps to reveal a more general and conclusive performance comparisons between methods considered. The analysis is then performed on the same Lorenz series now contaminated with some known noise levels; and then two river flow time series are analyzed. Finally, another global model, Support Vector Machines (SVM), is considered and its prediction performance is compared to ANN.

This chapter is organized as follows. The first section introduces the data used in the study. The second section presents a detailed performance comparison between ANN and the local prediction models. SVM as a global model is then introduced; this

is followed by a comparison of the prediction performance, on some time series, resulting from SVM and ANN. Finally, the computational times taken by different prediction models are presented.

3.2 DATA USED

A chaotic Lorenz time series and two mean daily river flow time series with very different flow characteristics are considered in this study. The two river flows are: (1) Mississippi River at Vicksburg, and (2) Wabash River at Mt. Carmel. The Mississippi river is characterized by large flow rates (mean flow of about 18,500 m^3/s) while the Wabash river is of moderate flow rates (mean flow of about 750 m^3/s). The data are downloaded from the US Geological Survey website.

3.2.1 Lorenz time series

Lorenz abstracted three ordinary differential equations from Galerkin approximation to the partial differential equations of thermal convection in the lower atmosphere derived by Salzman (Abarbanel, 1996). These have served as a set of benchmark equations for testing ideas in nonlinear dynamics. Lorenz model is given by the following three ordinary differential equations:

$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= -xz + \gamma x - y \\ \dot{z} &= xy - bz \end{aligned} \tag{3.1}$$

When standard parameter values $\sigma = 16$, $b = 4$ and $\gamma = 45.92$ are used, the orbits of the Lorenz system reside on a geometric object of dimension 2.06 (approximately) and exhibit non-periodic, chaotic motion (Abarbanel, 1996). The *x(t)* component is solved from the above equations by fourth order Runge-Kutta method with a time step of $\Delta t = 0.01$. Six thousand values of this *x(t)* time series is used in this study. It should be noted, however, that Lorenz series is fundamentally different from that of hydrological

time series in river flow. In the absence of any better model, Lorenz series is used in this study to test the techniques applied to river flow time series. Figure 3.1 shows the Lorenz time series used in this study. The figure shows that the series has a smooth variation. The statistics of the Lorenz series are; (1) mean = 0.84, (2) standard deviation = 12.68, (3) minimum = -30.52, and (4) maximum = 29.38.

3.2.2 Mississippi river flow time series

Mississippi river is one of the world's largest river systems (Figure 3.2). It is about 3,705 kilometers in length. The area of the Mississippi river basin is around 3.2 million square kilometers. The average amount of water discharged to the Gulf of Mexico is about 18,500 m^3/s. The spring floodwaters cause very costly flooding. Although billions have been spent to reduce flood damages, recent floods have cost billions of dollars and significant loss of life. Further understanding of the river flow behaviours and patterns is vital to the understanding of the complex ecosystem and development of protection strategy.

The daily river flow time series used in the analysis of this study is the Mississippi river flow measured at Vicksburg, Station No. 07289000 (Hydrologic Region 08 of USGS) for the period from January 01, 1975 to December 31, 1993. The time series data is downloaded from the USGS. The station is located close to the entrance to the sea. The time series used is shown in Figure 3.3. It shows that the Mississippi river flow has a somewhat smooth variation. The basic statistics of Mississippi river daily flow time series are: (1) mean flow = 18,458 m^3/s; (2) standard deviation = 9,728m^3/s; (3) minimum flow = 3,908 m^3/s; and (4) maximum flow = 52,108 m^3/s. This river flow time series has been shown to demonstrate chaotic behaviour by several studies (e.g. Liong et al., 2005; Yu et al., 2004). Also it has been

shown to produce better prediction performance with chaos approach than with conventional ARIMA models.

3.2.3 Wabash river flow time series

The Wabash River is a 475 miles (765 km) long river in the eastern United States that flows southwest from northwest Ohio. The basin area is approximately 33,100 square miles. The Wabash River has moderate flow rates, with the mean flow rate being about 750 m^3/s. The mean daily river flow measured at Mt. Carmel Station, station number 03377500 (hydrologic region 5 of USGS), are downloaded from USGS for this study. The basin is shown in Figure 3.4. The records used in the study covers daily data from January 01, 1960 to December 31, 1978. The time series is shown in Figure 3.5. Wabash river flow variation is not as smooth as in Mississippi flow. The basic statistics of Wabash river daily flow time series are: (1) mean flow = 756 m^3/s; (2) standard deviation = 792 m^3/s; (3) minimum flow = 48 m^3/s; and (4) maximum flow = 7023 m^3/s.

3.3 ANALYSIS: ARTIFICIAL NEURAL NETWORK AND LOCAL MODELS

3.3.1 Methodology

In this section, ANN prediction performance is compared with the two widely used local phase space prediction models: (1) the local averaging model and (2) the local polynomial model. In local polynomial models, the first and the second order polynomials are considered; the model which yields better prediction performance is then used for forecast of verification data. Predictions are performed for three different lead times: 1, 3 and 5. First, the investigation is performed on noise-free chaotic Lorenz time series to reveal a more general and conclusive comparison. Since real

world time series data are inevitably contaminated with different levels of noise, the same Lorenz series is then contaminated, with two different noise levels (5% and 30%), and analyzed. A Gaussian white random noise is used. The noise level is defined as the ratio of the standard deviation of the noise to the standard deviation of the noise-free signal as given in Eq. 3.2. Noisy data point y_i is obtained by adding noise υ_i to the noise-free value, x_i as in Eq. 3.3. Finally the results are demonstrated on the two river flow time series.

$$noise\ level = \left(\frac{standard\ deviation\ of\ noise}{standard\ deviation\ of\ noise\ free\ signal} \right) \times 100 \tag{3.2}$$

$$y_i = x_i + \upsilon_i \tag{3.3}$$

In the analysis, identifying chaotic signatures in the time series is essential. To do so Fourier analysis and the correlation integral analysis are used. For the phase space parameter determination and prediction purposes all the data sets are divided into 3 separate parts: training set, test set and validation set. Prediction performance on test set is used for the purposes of model selection and calibration of phase space parameters, wherever applicable. Validation set serves the purpose of verification of selected models/parameters on unseen data. For the Lorenz time series, the first 4800 points are used for training set, the next 600 for test set, and the last 600 for validation set. For the daily river flow time series, the first 15 years (approximately 5480 records) are used for training, the next 2 years (approximately 730 records) for test set, and the last two years (approx. 730 records) for validation. This prediction scheme is summarized in Appendix B.

Legates and McCabe (1999) recommend the use of at least one absolute error measure and one relative error measure to test the model performance in hydrologic

and hydro-climatic model validation. The relative error measure used in this study is the Normalized Root Mean Square Error (NRMSE) as given below:

$$NRMSE = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{N}(x_i - \bar{x})^2}} \tag{3.4}$$

where $\hat{x}_i$ is the predicted value of x_i and $\bar{x}$ is average value of the time series. A zero value in NRMSE indicates a perfect prediction while a value larger than 1 indicates that the predictions are no better than using the average value of the time series ($\bar{x}$). The absolute error measure, mean absolute error (MAE), is given as:

$$MAE = \frac{\sum_{i=1}^{N}\|x_i - \hat{x}_i\|}{N} \tag{3.5}$$

where x_i is the desired value and $\hat{x}_i$ is the predicted value. For the case of perfect prediction, the value of MAE is zero.

3.3.2 Analysis on Noise-free chaotic Lorenz time series

Application of Fourier analysis and the calculation of correlation integral analysis showed, as expected, the chaotic signatures in the noise-free Lorenz time series. Similar to the inverse approach (Phoon et al., 2002), in this study the parameter set (m, τ, (k)) is optimized simultaneously with the least prediction error as the objective function. For a certain combination of embedding dimension (m), time delay (τ) and nearest neighbours (k – only for local models), the phase space is reconstructed and the test set is used to check the performance of the trained model. For a certain prediction method, the combination of m, τ and k (k is needed only in local models), which gives the least prediction error (NRMSE) on the test set is selected as the optimal parameter set for prediction on the validation set. All possible combinations of the following ranges of parameters are considered. Following earlier studies (Phoon et

al., 2002; Yu et al., 2004; Liong et al. 2005), the considered range for the embedding dimension is 1 – 10. For the local models the number of nearest neighbours (k) is varied from 2 to 100 with an increasing step of 2. Due to the long computational time required, only four time delay values are considered based on earlier studies. Earlier studies (see Appendix C for details) have shown that very low time delays give best predictions for the noise-free Lorenz series. Hence, time delays of 1, 3, 6 and 9 are considered.

The optimal phase space parameter sets of the time series with different prediction techniques are shown in Table 3.1. It shows that the second order polynomial models perform better than the first order polynomial models. Therefore, tests are repeated with a third order polynomial model. Results, however, show that the second order performance is the best. The next section describes how ANN models have been trained.

3.3.2.1 Prediction with global Artificial Neural Network models

In the functional relationship for phase space prediction,

$$X_{i+T} = f_T(X_i) \tag{3.6}$$

where X_i and X_{i+T} are *m* dimensional vectors describing the state of the systems at times *i* and (*i*+*T*); the problem is to find a good approximation F_T to f_T. However, as it often happens, we are only interested in forecasting the last component x_{i+T} of X_{i+T}; the search is limited to a map $F_T : R^m \Rightarrow R$, which interpolates the pairs (X_i, x_{i+T}) instead of a function $F_T : R^m \Rightarrow R^m$ (Porporato and Ridolfi, 1996). A Multi Layer Perceptron can be used with m dimensional phase space vectors, X_i, as the inputs and the scalar x_{i+T} as the outputs to approximate a map $F_T : R^m \Rightarrow R$, valid over the entire phase space, i.e. a global fit in phase space prediction.

The MATLAB Neural Network Toolbox is used in this study. The study uses MLP with a single hidden layer since they are capable of universal approximation. The inputs are the elements of phase space vectors, X_i; hence, the number of input nodes are equal to the dimension of X_i. The output is the scalar x_{i+T} where T is the lead-time. Figure 3.6 shows the ANN architecture used in the study. Both input and output data are normalized into the range between 0 and 1. Logistic Sigmoid transfer function is used for all hidden neurons. Linear transfer function is used in the output neuron to allow for the unknown output range of the time series (Haykin, S. 1999). Nguyen-Widrow initialization algorithm is used for initialization of weights and biases. A training algorithm that uses Levenberg-Marquardt optimization method, known to give faster training for a smaller number of weights, is used for updating weights and biases. Training is continued until one of the following criteria is met: (1) a maximum number of epochs is reached; (2) the change of performance (error minimization in ANN) falls below a pre-specified value; or (3) the learning rate exceeds a maximum value specified (the study uses an adaptive learning rate). ANN has a series of parameters to be fine-tuned and appropriate methods to be selected before model training. This study fine-tuned the two critical parameters, the number of hidden neurons and the number of epochs, by trial and error.

To fine-tune the number of hidden neurons a lead-time 3 prediction horizon with m = 5 and τ = 1 is considered. The number of hidden neurons considered is varied from 10 – 300 (10-25 in steps of 5 and, 25-200 in steps of 25, 220-300 in steps of 40). The performance on the test set (Figure 3.7 (a)) shows that the prediction accuracy improves with the number of hidden neurons and levels off after the number of hidden neurons reaches 175. Since the computational time significantly increases with high number of neurons, the study uses only 100 hidden neurons (175 neurons (84 minutes)

requires more than twice the computational time of 100 neurons (31 minutes)); thus, less than the optimal size. After the number of neurons is fixed at 100, the most appropriate number of epochs (considered are 25, 50, 100 only) is searched. Results show 50 epochs are acceptable with regards to the computational time and performance accuracy (Figure 3.7 (b)).

The MLP converges to a solution depending on the initial weights. If a proper set of initial weights is not used, it is possible for the MLP to converge to an unsatisfactory local optimum. Therefore, for a certain combination of (m, τ), this study trains 5 MLPs (with 100 hidden neurons and 50 epochs) with 5 different sets of initial weights. The trained MLP with the lowest prediction error on the test set is selected as the optimally trained MLP for the (m, τ) combination considered. For a given lead time, from all possible (m, τ) combinations, the MLP that gives smallest error on test set is selected as the optimum network and the (m, τ) combination is taken as the optimal phase space parameters set. This network is then used to predict the validation set. The procedure is schematically shown in Figure 3.8 using the lead-time 1 as an example.

3.3.2.2 Results

Table 3.2 shows the prediction errors resulting from the applications of the local averaging, local polynomials and global artificial neural network on the validation set of the noise-free Lorenz time series at different lead-times. (The prediction performances of various models on test sets are given in Appendix D). The percentage improvement in prediction accuracy (or percentage reduction in error) of ANN models over the local averaging and local polynomial models indicated in the tables is calculated as $\frac{(a-b)}{a}\times 100$ where a is the error on local model and b is the error on ANN model. Positive percentage values indicate better prediction performance of

ANN models and negative values indicate better performance of local models. Percentage improvement is given for the linear error measure: MAE.

Table 3.2 shows that the prediction accuracy of ANN models is significantly higher than that of both local averaging and local polynomial models for all 1, 3 and 5 lead time predictions. The results are consistent in both error indicators, NRMSE and MAE. The ANN models yield remarkable improvement, about 98% over the local averaging models. The percentage improvement over local polynomial models is although not that impressive, over 9% for all lead-times. The prediction accuracies obtained by both ANN models and local polynomial models are remarkable. The performance of local averaging technique, however, is very poor compared to local polynomial and ANN models. With all prediction models, the prediction performance, as expected, has deteriorated with the increase of prediction horizon.

Figure 3.9 demonstrates the series of the validation data and corresponding prediction errors (error = (actual value – predicted value)) with local averaging model, local polynomial model and ANN model for lead–time 5. The prediction with local averaging technique (Figure 3.9 (b)) is extremely poor (note the difference in scale) compared to the other two models. The polynomial model and ANN model have errors of comparable magnitudes; however, the ANN prediction errors are smaller than those of local polynomial model. The particularly high prediction errors (between 200 – 300 time units) are seen to be corresponding to some phase space vectors in an unpopulated area in the attractor, and the reason for low accuracy may be that the models do not have sufficient past experience to model that region.

3.3.3 Analysis on Noise added Lorenz time series

All real data generally contain noise whose precise nature (e.g. white/ coloured; distribution; level of noise etc) is unknown. To gain an understanding of the

performance of the local models and global ANN models on noisy series, the analysis is first conducted on the Lorenz series added with a known noise level and then conducted on the real flow time series, as shown in a later section. Zero mean Gaussian noises with noise levels 5% and 30% (as defined in Eq. 3.2) are added to the clean signal (the noise-free series used above), as shown in Eq. 3.3, to obtain noisy Lorenz series. In the analysis, the procedures and the parameters for the prediction tools are exactly the same as those in noise-free Lorenz series. For ANN models, 25 hidden neurons and 50 epochs are chosen from a trial-and-error procedure on 5% noisy series similar to that of the noise-free Lorenz series. The same values are used with 30% noisy series as well.

The optimal parameter sets obtained, using the ranges in noise-free series, contained sets with m values equal to 10 (the upper bound). Therefore, only for local models, the tests are repeated with m expanded to 16. The final optimal parameter sets with each prediction method are shown in Tables 3.3 and 3.4 for 5% and 30% noisy series respectively. Unlike in noise-free time series, in noise added Lorenz series the first order polynomial models perform better than the second order polynomial models. This shows that for the noisy time series, increasing the order of polynomials does not necessarily yield a significant prediction improvement. In fact, the higher order polynomials are more likely to yield worse predictions. The reason is perhaps that in noisy time series the coefficients of the polynomial models, $\boldsymbol{c}(m,k)$ (Eq. 2.5), cannot be determined accurately by matrix inversion procedure. These errors may propagate large prediction errors when iterated with higher order polynomial models.

The prediction results obtained are shown in Table 3.5 and 3.6 for 5% and 30% noisy series respectively. (The prediction performances of various models on test sets are given in Appendix D). It shows that the prediction performance of both the

polynomial models and ANN models drastically reduces with the introduction of noise. The performance of the local averaging models is also considerably reduced. The ANN models, however, still outperform, the local models in prediction accuracy at 5% noise level although the improvement is not as pronounced as in the noise-free Lorenz series. At a very high noise level of 30%, however, performance of all the methods are of the same level. It can be noted in both 5% and 30% noisy series, the polynomial models' relative performance deteriorates with the increase in lead-time. The reason is perhaps, as noted earlier, the inaccurately determined coefficients, ***c****(m,k)* (E.q. 2.5), may propagate large prediction errors when iterated over longer lead-times. Figures 3.10 and 3.11 show the validation data and the prediction errors (error = (noisy series value – predicted value)) with various models on validation set for a lead time 5 on 5% and 30% noisy series respectively. They show that the errors are relatively very large compared to those in the noise-free Lorenz series (note the differences in scales of figures in the two cases, noise-free and noisy). The errors resulted from the three different models are of same order of magnitudes for the noisy series; and the differences in prediction performance are not reflected in the figures.

3.3.4 Analysis on river flow time series

The performances of the local and the global models are then verified on two daily river flow time series. Mississippi river flow has been reported to show chaotic behaviour (e.g. Liong et al, 2005; Yu et al., 2004). Fourier analysis on Wabash river flow shows a broad band power spectrum (Figure 3.12 (a)); and the correlation integral analysis (Figure 3.12 (b)) shows low, non-integer correlation dimension. These indicate low dimensional chaotic behavior in the Wabash river flow.

Analysis is performed similarly as the case for noise-free and noisy Lorenz series. Earlier studies (e.g. Liong et al., 2005; Yu et al., 2004; Appendix C) on the flow

time series have shown that a delay time of one day gives best predictions. Therefore, a time delay of 1 day is considered in this study. For predictions with ANN, 25 hidden neurons and 50 epochs are selected from a limited number of trial-and-error runs on Mississippi river flow time series. The same values are used for Wabash River flow as well. The other parameters and methods follow those used in Lorenz series analysis.

The optimal parameter sets for each prediction method for Mississippi flow time series and for Wabash flow time series are shown in Table 3.7 and Table 3.8 respectively. In this study, for Mississippi river the second order polynomial models give better predictions, although not markedly better, than the first order polynomial models (see Appendix E). For the Wabash series, however, the first order polynomial models give better prediction accuracy over its second order polynomial counterpart. A performance comparison between ANN, the local averaging technique, and the local polynomial model resulting from the Mississippi river flow data and Wabash river flow data are given in Table 3.9 and Table 3.10 respectively. (The prediction performances of various models on test sets are given in Appendix D). Table 3.9 shows that for Mississippi the prediction accuracy of ANN models is higher than that of the local averaging technique. However, the performance of ANN is only slightly higher than that of local polynomial models. In Wabash river flow time series (Table 3.10) too, the ANN performs better than the local models. Figures 3.13 and 3.14 give a graphical performance of the trained models on the validation data and corresponding prediction errors (error = (observed value – predicted value)) for lead-time 5 for Mississippi and Wabash river flow time series. Similar to noise added Lorenz series, errors resulting from river flow analysis with all three models are of the same order of magnitude and the differences in performance are not visible in graphical representations.

3.3.5. Discussion

In all time series considered, ANN models show better prediction performance than the local prediction models; this is with the exception of Lorenz time series with 30% noise level where the performance of the ANN and the local averaging models are almost the same. The prediction accuracy of ANN on the noise-free chaotic Lorenz series is clearly significantly better than those of the local averaging and the local polynomial models. Although the performance of polynomial models is not as good as ANN models, its prediction accuracies on noise-free Lorenz series are nevertheless commendable. In Lorenz series with known noise levels (except at 30% noise level) and in two real river flow time series, ANN still outperforms the other methods although the improvement is not as pronounced as that in the noise-free Lorenz series. It appears, however, that in time series of very high noise levels the performance of ANN is no better than local averaging techniques.

It should be noted that in this study the parameters of ANN (e.g. the number of hidden neurons, epochs, etc) are selected only after a few trial-and-error tests on a limited number of parameter values. The fact that the performance of the model with non-optimal parameter values is better than its counterparts (the local averaging and the local polynomials) implies that an ANN with optimal parameters will surely lend the trained models as equally good as or even better performance than those of limitedly trained ANN models conducted in this study.

In relation to the performance of the local averaging models, the results obtained in this study agree with those of Elshorbagy et al. (2002) where it is reported that ANN models outperformed the local averaging models. However, it is interesting to note that Sivakumar et al. (2002) observed that ANN (with MLP) performed much worse than the local phase space prediction (polynomial models). The reported

absolute errors of ANN models were 4-8 times larger than those of the local polynomial models. They believed that better performance of local approximation method was due to the representation of dynamics in the phase space step by step in local neighbourhoods, and such local approximation was capable of better capturing the dynamics than a global method when the system under investigation exhibits low dimensional chaotic dynamical behaviour. Furthermore, they believed that MLPs might not be the best type of ANN for longer prediction horizons and suggested to opt for ANN of other types.

Like any model with calibration parameters, ANN has a series of parameters to be calibrated before it can be used to its optimum. As noted earlier, in this study the values used for parameters of ANN (the MLP models) are derived through only a limited number of combinations. However, unlike in other studies, the parameters for local prediction models are chosen from a wide range of values to yield the most optimal parameter sets. In addition, it is worth to note that the analysis is first performed on a noise-free synthetic chaotic time series and then on a noise added synthetic series to gain a more decisive performance comparison among models tested. It can be safely extrapolated that should the parameters under consideration were rigorously optimized, the results would show that ANN would yield better performance than, if not equal to, those shown here.

3.3.6. Conclusion

This study investigated the performance of ANN as a global model in chaotic time series predictions compared to the widely used local prediction models (the local averaging models and the local polynomial models). To gain more general and robust conclusions, the analysis was first conducted on a noise-free chaotic Lorenz series.

The analyses were then continued and performed on Lorenz series contaminated with some known noise levels and two daily river flow time series (for 3 different prediction horizons, 1, 3 and 5). A limited number of trial-and-errors lead to the ANN (MLP) parameter choice. In all time series (with the exception of 30% noise level), ANN models showed better prediction performance than local prediction models. At a very high noise of 30%, however, the performance of ANN was similar to that of local averaging models. For the noise-free Lorenz series, the improvement of ANN models over local averaging models was highly significant. Both ANN models and local polynomial models gave remarkably high prediction accuracy in noise-free Lorenz series. However, the performance of ANN models was still better than that of the polynomial models. The prediction accuracies of all the models dropped considerably when noise was added to Lorenz series. The ANN models, however, still outperformed the local models. For the river flow time series too the performance of ANN was better than that of local prediction methods. It can be safely concluded that global ANN models can yield equally good prediction as, if not better than, the widely used local models in phase space prediction of chaotic time series.

3.4 SUPPORT VECTOR MACHINES AS A GLOBAL MODEL

3.4.1 Introduction

SVM with ε-insensitive loss function (to be explained in the next section) is used in this study. As it will be shown later a Kernel function is employed to facilitate computations in higher dimensional feature space. Many kernels are used in SVM. Of those, the Gaussian kernel (Eq. 3.7) is capable of mapping data into infinite dimensional feature spaces.

$$K(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{\sigma^2}\right) \tag{3.7}$$

The width parameter σ is used to control the power of the feature space. Small values of σ lead to very powerful feature spaces. Several studies (e.g. Babovic et al., 2000; Dibike et al., 2001; Liong and Sivapragasam, 2002) have shown that Gaussian kernel produces good performance in hydrology. Hence, the Gaussian Kernel is the choice in this study.

The SVM used in this study therefore, has 3 parameters (C, σ, ε) whose optimal values have to be determined for optimal prediction performance. With the two parameters (m, τ) for reconstruction of phase space, there are hence altogether 5 parameters to be determined. In this study all the 5 parameters (m, τ, C, σ, ε) are optimized simultaneously with the least prediction error as the objective function. The approach used is schematically shown in Figure 3.15.

Optimizing five parameters using exhaustive search approaches will be quite a time consuming task. Instead, Genetic Algorithm (GA), inspired by the process of natural selection in nature (Holland and Nafpliotis, 1975) is used. GAs are much more efficient in optimal search problems where large number of parameters and wide parameter search ranges are involved. GA differs from other classical search and optimization methods in a number of ways. The desirable features of GA over other optimization methods are: (1) it does not need an explicit objective function in terms of the free parameters, (2) it does not use the gradient information in search space, and (3) it works with a set of solutions instead of one solution in each iteration and thus chances of being trapped in a local optima are less. These features ideally fit the present problem of searching optimal parameters (m, τ, C, σ, ε) where an explicit objective function in terms of parameters is not available and the search space is

possibly crowded with local optima. Therefore, an evolutionary search technique, micro Genetic Algorithm, is employed to determine the optimal parameters (m, τ, C, σ, ε) simultaneously.

This section is organized as follows. First, the SVM formulation with ε-insensitive loss function is presented. This original formulation has been found to be inefficient for problems with large numbers of data records. Therefore, a decomposition technique is employed to make the computations efficient (e.g. Joachims, 1999; Collobert and Bengio, 2001; Yu et al, 2004). This decomposition method for large scale SVM regression is then explained. This is followed by a description of the micro genetic algorithm, mGA. The implementation of the SVM – mGA coupled procedure is then explained. Finally the application results are presented.

3.4.2 Support Vector Machine formulation with ε-insensitive loss function

The SVM formulation with ε-insensitive loss function (Scholkopf and Smola, 2002) is presented in this section. A training data set ($\boldsymbol{x}_i, y_i$), i = 1, 2, … , N where $\boldsymbol{x}$ is m dimensional input vector and the y is one dimensional output is considered. ε-insensitive loss function can be expressed as,

$$\left|y - f(\boldsymbol{x})\right|_\varepsilon = \max\left\{0, \left|y - f(\boldsymbol{x})\right| - \varepsilon\right\} \tag{3.8}$$

This is illustrated in Figure 3.16. Using the above loss function the empirical risk function $R_{emp}[f]$ of (Eq. 2.15) can be expressed as,

$$R_{emp}[f] := \frac{1}{N}\sum_{i=1}^{N}\left|y_i - f(\boldsymbol{x}_i)\right|_\varepsilon \tag{3.9}$$

Now minimizing the regularized risk functional $\frac{1}{2}\|\boldsymbol{W}\|^2 + C.R_{emp}[f]$ is equivalent to the following constrained optimization problem,

$$
\begin{aligned}
&\textit{min } E\left(\boldsymbol{W}, \xi^{(*)}\right) = \frac{1}{2}\|\boldsymbol{W}\|^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right) \\
&\textit{such that } \left(\boldsymbol{W}^T.\varphi(\boldsymbol{x}_i) + b\right) - y_i \le \varepsilon + \xi_i \\
&\qquad\qquad y_i - \left(\boldsymbol{W}^T.\varphi(\boldsymbol{x}_i) + b\right) \le \varepsilon + \xi_i^* \\
&\qquad\qquad \xi^{(*)} \ge 0
\end{aligned}
\tag{3.10}
$$

In the above equation, $^{(*)}$ is shorthand implying both the variables with and without asterisks. To solve Eq. 3.10, a Lagrangian is formed by introducing a dual set of variables. The Lagrangian formulation is,

$$
\begin{aligned}
L := \frac{1}{2}\|\boldsymbol{W}\|^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right) - \sum_{i=1}^{N}\left(\eta_i\xi_i + \eta_i^*\xi_i^*\right) \\
- \sum_{i=1}^{N}\alpha_i\left(\varepsilon + \xi_i + y_i - \boldsymbol{W}^T.\varphi(\boldsymbol{x}_i) - b\right) \\
- \sum_{i=1}^{N}\alpha_i^*\left(\varepsilon + \xi_i^* - y_i + \boldsymbol{W}^T.\varphi(\boldsymbol{x}_i) + b\right)
\end{aligned}
\tag{3.11}
$$

The dual variables (or Lagrange multipliers) in Eq. 3.11 have to satisfy possitivity constraints,

$$
\alpha_i^{(*)}, \eta_i^{(*)} \ge 0 \tag{3.12}
$$

At the optimal solution all partial derivatives vanish.

$$
\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{W}} = 0, \quad \frac{\partial \boldsymbol{L}}{\partial \xi_i^{(*)}} = 0, \quad \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{b}} = 0 \tag{3.13}
$$

These yield,

$$
\boldsymbol{W} = \sum_{i=1}^{N}\left(\alpha_i - \alpha_i^*\right)\varphi(\boldsymbol{x}_i), \qquad \eta_i^{(*)} = \boldsymbol{C} - \alpha_i^{(*)} \tag{3.14}
$$

Substituting Eq. 3.13 into Eq. 3.14 and Eq. 3.11, the optimization problem now reads,

$$
\begin{aligned}
&\textit{Maximize} \quad E2\left(\alpha, \alpha^*\right) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\alpha_i - \alpha_i^*\right)\left(\alpha_j - \alpha_j^*\right)K\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) + y_i\left(\alpha_j - \alpha_j^*\right) - \varepsilon\left(\alpha_j + \alpha_j^*\right) \\
&\textit{such that} \quad \sum_{i=1}^{N}\left(\alpha_i - \alpha_i^*\right) = 0 \\
&\qquad\qquad 0 \le \alpha^{(*)} \le C, \quad i = 1, 2, ..., N
\end{aligned}
\tag{3.15}
$$

where $\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \varphi(\boldsymbol{x}_i)^T . \varphi(\boldsymbol{x}_j)$ is the inner product kernel. The dual problem of maximizing $E2(\alpha, \alpha^*)$ is a quadratic function subjected to a linear constraint. Solving this quadratic programming problem, the weights can be determined through Eq. 3.14. Practically, it is not needed to solve $\boldsymbol{W}$ explicitly. With Eq. 3.14 in regression function $f(x) = \boldsymbol{W}^T . \boldsymbol{\varphi(x)} + b$, it now reads,

$$f(\boldsymbol{x}) = \boldsymbol{W}^T . \varphi(\boldsymbol{x}) + b = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) K(\boldsymbol{x}_i, \boldsymbol{x}) + b \tag{3.16}$$

Here, the kernel trick has eliminated the need to determine $\boldsymbol{W}$ explicitly. The KKT (Karush - Khun – Tucker) conditions can be used to compute b. These state that at the point of solution, the product between dual variables and constraints have to vanish.

$$\begin{aligned} \alpha_i (\varepsilon + \xi_i - y_i + \boldsymbol{W}^T . \boldsymbol{\varphi(x_i)} + b) = 0 \\ \alpha_i^* (\varepsilon + \xi_i^* + y_i + \boldsymbol{W}^T . \boldsymbol{\varphi(x_i)} - b) = 0 \end{aligned} \tag{3.17}$$

$$\begin{aligned} (C - \alpha_i) \xi_i = 0 \\ (C - \alpha_i^*) \xi_i^* = 0 \end{aligned} \tag{3.18}$$

The above equations allow one to draw several conclusions (Scholkopf and Smola, 2002):

(1) Only examples $(\boldsymbol{x_i}, y_i)$ with corresponding $\alpha_i^{(*)} = C$ can lie outside the ε-insensitive tube (i.e., $\zeta_i^{(*)} > 0$) around f.

(2) We have $\alpha_i \alpha_i^* = 0$. In other words, there can never be a set of dual variables α_i, α_i^*, which are both simultaneously nonzero.

(3) For $\alpha_i^{(*)} \in (0, C)$, we have $\zeta_i^{(*)} = 0$ and furthermore the second factor in Eq. 3.17 must vanish. Hence *b* can be computed as follows.

$$\begin{aligned} b = y_i - \boldsymbol{W}^T \varphi(\boldsymbol{x}_i) - \varepsilon \ \ for \ \alpha_i \in (0, C), \\ b = y_i - \boldsymbol{W}^T \varphi(\boldsymbol{x}_i) + \varepsilon \ \ for \ \alpha_i^* \in (0, C) \end{aligned} \tag{3.19}$$

Theoretically it is sufficient to use any Lagrange multiplier in $(0, C)$. Given the choice between several such multipliers it is safe to use one that is not too close to 0 or *C*.

(4) Sparcity of support vector expansion: from Eq 3.17 it follows that the Lagrange multipliers may be nonzero only for $|f(\boldsymbol{x}_i) - y_i| \geq \varepsilon$; in other words for all points inside the $\varepsilon -$ tube, their α_i, α_i^* vanish. This is because when $|f(\boldsymbol{x}_i) - y_i| < \varepsilon$ the second factor in Eq. 3.17 is nonzero, hence α_i, α_i^* must be zero for the KKT conditions to be satisfied. Therefore, one does not need all $\boldsymbol{x}_i$ to describe $\boldsymbol{W}$. The examples that come with nonvanishing coefficients are called *Support Vectors*. It is geometrically plausible that the points inside the tube do not contribute to the solution: one could remove any of them, and still obtain the same solution. Therefore, they do not carry any relevant information.

Once *b* is found, the regression function (Eq. 3.16) can be used to provide future predictions (Figure 3.17).

3.4.3 Decomposition algorithm for large scale SVM regression

The original formulation of SVM (Eq. 3.15) with ε-insensitive loss function deals with a standard quadratic programming problem of the form: $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{x}'\boldsymbol{H}\boldsymbol{x} + \boldsymbol{c}'\boldsymbol{x}$ where $\boldsymbol{H}$ is the Hessian Matrix. Denoting,

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \dots \\ \alpha_N \end{bmatrix} \quad \boldsymbol{\alpha}^* = \begin{bmatrix} \alpha_1^* \\ \alpha_2^* \\ \dots \\ \alpha_N^* \end{bmatrix} \quad \boldsymbol{K} = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_1, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_2, \boldsymbol{x}_N) \\ \dots & \dots & \dots & \dots \\ k(\boldsymbol{x}_N, \boldsymbol{y}_1) & k(\boldsymbol{x}_N, \boldsymbol{x}_2) & \dots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \quad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_N \end{bmatrix}$$

(3.20)

and $\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\alpha}^* \end{pmatrix}$ $\widetilde{\boldsymbol{K}} = \begin{pmatrix} \boldsymbol{K} & \boldsymbol{K} \\ \boldsymbol{K} & \boldsymbol{K} \end{pmatrix}$ $\boldsymbol{b} = \begin{pmatrix} -\boldsymbol{y} - & \varepsilon \\ -\boldsymbol{y} + & \varepsilon \end{pmatrix}$ (3.21)

now the Eq. 3.15 is equivalent to

Minimize: $\widetilde{\boldsymbol{E}}(\boldsymbol{\beta}) = \frac{1}{2}\boldsymbol{\beta}\widetilde{\boldsymbol{K}}\boldsymbol{\beta} - \boldsymbol{\beta}\boldsymbol{b}$

subject to : $\boldsymbol{\beta}^T \boldsymbol{1} = 0$ (3.22)

$$0 \le \delta_i \beta_i \le C, \quad i = 1, 2 \ldots 2N$$

where $\delta_i = 1$ for $1 \le i \le N$ and $\delta_i = -1$ for $N+1 \le i \le 2N$.

This formulation becomes intractable in terms of computer memory and time when a large number of training patterns has to be used. This is because the Hessian matrix in Eq. 3.22 has the size of the square of twice the sample size. For example, as shown by Yu et al. (2004), for a 20 year daily flow time series where the total number of records is about 7300, the size of the Hessian matrix is 213.16 million. If one stores this matrix in double precision (64 bit), it will take 1705 MB of computer memory. Common PCs have RAM sizes of 256/ 512 MB only. Therefore, to solve problems of large data sizes such as chaotic time series the commonly used SVM formulation is not feasible.

Recently, two techniques have been developed to overcome the above problem: (1) decomposition methods and (2) the Least Squares Support Vector Machines, LS-SVM (Suykens et al., 2002). The decomposition technique is used in this study. Here, the idea is to decompose the large quadratic program into manageable sub-problems. Platt (1999) developed sequential minimal optimization (SMO) algorithm for classification problems. Joachims (1999) developed SVM^{light} for classification. More recently, Collobet and Bengio (2001) successfully implemented a decomposition method similar to Joachims (1999) for regression problems. They showed that there is a convergence proof for their algorithm. The decomposition algorithm for SVM is summarized as follows.

(1) Set an initial value $\boldsymbol{\beta}^0$ to $\boldsymbol{\beta}$

(2) Select 2 working variables (selecting q variables $2 \leq q \leq 2N$ is the most general case, but 2 variables has shown to be the most efficient) i.e. β_1, β_2 among $2N$ variables $\boldsymbol{\beta}$.

(3) Solve the quadratic program having 2 variables analytically.

(4) Check whether the optimal conditions are met. If KKT conditions are satisfied then the optimum is reached. Otherwise go to step 2 and repeat the steps.

To make the above algorithm more efficient, Collobet and Bengio (2001) use an additional step after step 3 called *shrinking*. In shrinking a search is made for variables whose values have been at 0 or C for long time and will probably not change anymore. These variables are removed from the problem.

Selection of two working variables. There are $\frac{1}{2}C_2^{2N}$ ways of selecting 2 variables from 2N number of variables. Selection of a good working set is crucial for faster convergence. This selection is made based on Zoutendijk's feasible direction method (1970).

Solving the quadratic program with 2 variables. Let the set of working variables be denoted by S and the fixed set be F. Denoting,

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_S \\ \boldsymbol{\beta}_F \end{bmatrix} \quad \widetilde{\boldsymbol{K}} = \begin{bmatrix} \boldsymbol{K}_{SS} & \boldsymbol{K}_{SF} \\ \boldsymbol{K}_{FS} & \boldsymbol{K}_{FF} \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{b}_S \\ \boldsymbol{b}_F \end{bmatrix} \tag{3.23}$$

$\boldsymbol{\beta}_S$ contains the working variables. The objective function (Eq. 3.22) now becomes

$$\widetilde{\boldsymbol{E}}(\boldsymbol{\beta}) = \frac{1}{2}\boldsymbol{\beta}\widetilde{\boldsymbol{K}}\boldsymbol{\beta} - \boldsymbol{\beta}\boldsymbol{b} = \frac{1}{2}\boldsymbol{\beta}_S^T\widetilde{\boldsymbol{K}}_{SS}\boldsymbol{\beta}_S - \boldsymbol{\beta}_S^T\left(\boldsymbol{b}_S - \widetilde{\boldsymbol{K}}_{SF}\boldsymbol{\beta}_F\right) + \frac{1}{2}\boldsymbol{\beta}_F^T\widetilde{\boldsymbol{K}}_{FF}\boldsymbol{\beta}_F - \boldsymbol{\beta}_F^T\boldsymbol{b}_F \tag{3.24}$$

Denoting $\boldsymbol{h} = \left(\boldsymbol{b}_S - \widetilde{\boldsymbol{K}}_{SF}\boldsymbol{\beta}_F\right)$, it is now equivalent to the standard quadratic program,

Minimize : $\widetilde{\boldsymbol{E}}(\boldsymbol{\beta}_s) = \frac{1}{2}\boldsymbol{\beta}_s^T \widetilde{\boldsymbol{K}}_{ss}\boldsymbol{\beta}_s - \boldsymbol{\beta}_s^T \boldsymbol{h}$

Subject to: $\boldsymbol{\beta}_s^T \boldsymbol{1} = -\boldsymbol{\beta}_F^T \boldsymbol{1}$ (3.25)

$$0 \le \delta_i \beta_i \le C, \quad i = 1, 2 \ldots 2N$$

With this quadratic program only the $\widetilde{\boldsymbol{K}}_{ss}$ and the $\widetilde{\boldsymbol{K}}_{sF}$ have to be stored in the memory. Those are the 2 rows of the Hessian matrix corresponding to 2 working variables. Now the memory requirement has decreased from size $4N^2$ to $4N$. Another advantage of having only two variables is that the above optimization problem can be solved analytically (Collobet and Bengio 2001).

Checking the KKT conditions. SVM solves a quadratic programming problem which has a unique optimal. Karush-Khun-Tucker (KKT) conditions are necessary and sufficient conditions for an optimal solution. Checking the KKT conditions for the problem stated in Eq. 3.15 it can be determined whether the optimal solution has been reached.

This study uses the software SVMTorch II (Collobet and Bengio 2001) written in C language and running in Unix/Linux platforms. The SVM explained here has 3 parameters of which the optimal values are to be determined. To optimize these parameters together with the phase space parameters, (m, τ), a Micro Genetic Algorithm (mGA) is used. The mGA search technique is explained in the next section.

3.4.4 Micro Genetic Algorithm for SVM parameter optimization

Genetic algorithm (GA) is inspired by the process of natural selection in nature (Holland and Nafpliotis, 1975). In GA the solutions to the problem are evolved rather than the problem being solved directly. In GA, each parameter set, generally coded in

binary, is called a chromosome. A fitness value is assigned to each chromosome depending on its performance on the objective function. The total number of chromosomes in each iteration is known as the population size and each iteration is known as a generation. The chromosomes in the first generation are generally generated randomly and the chromosomes of the subsequent generations are generated through the basic mechanisms, selection, crossover and mutation. The chromosomes associated with higher fitness values are selected more often than the less fit chromosomes, following the Darwinian principle of 'survival of the fittest'. In crossover an offspring is generated from last generation through transfer of genes between chromosomes. In mutation, one or more individuals of the population is mutated to yield new individual(s). This maintains the diversity within the population and inhibits premature convergence. As the population evolves, the overall performance of the population is expected to improve. This process is repeated until a predetermined stopping criterion is met. Good introductions and explanations of GA techniques can be found in, among others, Goldberg (1989a) and Michalewics (1996).

The commonly used GAs, which typically use population size ranging from 30 to 200, have been proven to be useful tools for many optimization problems. However, they have several limitations. A serious limitation of these GAs is the time penalty involved in evaluating the fitness function for large populations particularly in complex problems (Abu-Lebdeh and Benekohal, 1999). Goldberg (1989b) suggested that small populations could be successfully used with GAs if the population is restarted sufficient number of times. This is possible since smaller populations converge in fewer generations than do large populations. These small-population GAs are called micro-GAs. Faster convergence provides the opportunity to restart the

micro-GA more often than the regular GA. For problems where function evaluations are expensive, many researchers have resorted to micro-GAs (Deb, 2001).

Micro-GA (mGA) uses almost the same basic operations as regular GAs. However, it differs from regular GAs in two important aspects: (1) small population size, and (2) no implementation of conventional mutation. Also the mGA uses the elitism strategy where the best individual in current population is transferred to the next generation. Krishnakumar (1989) was the first to report the implementation of micro-GA. The flow chart of mGA algorithm used in this study is shown in Figure 3.18. A fixed number of generations (100 generations) is used as the stopping criteria. The population size is 10. Two convergence criteria used are: (1) when more than 7 individuals are not significantly different from the performance of the best individual the population is considered converged, or (2) when the percentage of bits different from the best individual is less than 10%, the population is converged. The prediction accuracy on test set is taken as the fitness function.

3.4.5 Implementation and Results

The mGA is implemented using the Genetic and Evolutionary Algorithm Toolbox (GEATbx) of Pohlheim (2000) in Matlab. The SVM with decomposition algorithm is coupled with mGA to determine both optimal phase space parameters (m, τ) and SVM parameters (C, σ, ε) simultaneously. The module shown in Figure 3.15 is now an evaluation of a single chromosome (or individual in the population) in mGA (Figure 3.18). The prediction performance on test set is taken as the fitness value of that individual. This procedure is repeated until the stopping criteria are met. The range of values chosen for phase space parameters are the same as in the analysis of ANN discussed before. The SVM literature provides some guidelines for the selection of

($\boldsymbol{C}$, ε). According to Matterra and Haykin (1999), C can be taken as the range of output values (i.e. $(y_{max} - y_{min})$). Cherkasky and Ma (2004) recommend $C = \max(|\bar{y} + 3\sigma'|, |\bar{y} - 3\sigma'|)$ where $\bar{y}$ is the average of the time series and σ' is the standard deviation of the series. They propose the ε value to be set at $\varepsilon = 3\sigma_0\sqrt{\frac{\ln n}{n}}$ where σ_0 is the standard deviation of the noise present in the series and n is the number of training samples. However, these recommendations do not necessarily provide the optimal prediction performance. Therefore, with mGA, this study considers a range of values for (C, σ, ε) in the vicinity of the recommended values. However, a slightly larger range is considered for C since high C values have shown good predictions on hydrological time series (Yu, 2004). The range considered for C is (0 to 2.5)* $(y_{max} - y_{min})$. The one-step prediction error is taken as an approximation for the σ_0 (Cherkasky and Ma, 2004) value and the ε was varied from 0 – $2\sigma_0$ so that the recommended value is well within the considered range. For the special case of noise-free data, ε is varied from 0 – 10^{-4} to minimize the algorithm being trapped in numerical problems. There is no definite criterion for the range of σ (kernel width) value. Following an application by (Cherkasky and Ma, 2004) where they used (0.1 to 0.5)* $(y_{max} - y_{min})$, this study used (0.0 to 1.25)* $(y_{max} - y_{min})$. The performance of SVM is directly compared with that of ANN, which was shown to be superior to local models in the earlier section.

The optimal parameter sets with SVM predictions are shown in Table 3.11. The prediction performance of SVM is compared with that of ANN in Tables 3.12, 3.13, 3.14, 3.15 and 3.16 for noise-free Lorenz, 5% noisy Lorenz, 30% noisy Lorenz, Mississippi River flow and Wabash River flow respectively. The percentage

improvement of SVM prediction performance, compared to ANN, is also presented. The positive percentage values represent better prediction performance of SVM compared to ANN and vice versa.

The prediction performance of SVM on noise free Lorenz series (Tables 3.12) is very poor compared to ANN. This could be due to two reasons: (1) the SVM decomposition algorithm with shrinking technique is an approximate method and it may not be able to deal with delicate noise-free data which can model up to very high accuracies, (2) the SVM selects points lying outside the ε tube as its support vectors. For noise-free time series, however, ideally all points should fall on the regression function and all of them should be considered as support vectors. This is possible only when $\varepsilon = 0$ and the Hessian is well conditioned. This, however, cannot be expected in numerical solution of Quadratic Programming problem. Similar to the behaviour of other prediction models, the introduction of noise has caused a considerable drop in SVM prediction accuracy too. The differences in prediction errors between ANN and SVM**,** applied on Lorenz series with 5% and 30% noise levels, and on Mississippi and Wabash flow time series (Tables 3.13, 3.14, 3.15 and 3.16), is insignificant.

3.5 COMPUTATIONAL TIME IN LOCAL/ GLOBAL PREDICTION TECHNIQUES

This section presents the computational time required by different prediction methods: (1) local averaging model; (2) local polynomial model; (3) ANN; and (4) SVM. The different methods are implemented in different computer languages and run on different platforms.

The local averaging technique is coded in FORTRAN language and was run in Pentium IV, 2.4 GHz, 512 MB RAM machine running Windows XP.

The local polynomial technique is coded in MATLAB and was run in Pentium IV, 2.4 GHz, 512 MB RAM machine running Windows XP.

The ANN is implemented using MATLAB Neural Networks toolbox and was run in Pentium IV, 2.4 GHz, 512 MB RAM machine running Windows XP.

The SVM used in the study has a combination of codes. The mGA is coded in MATLAB (using the Genetic and Evolutionary Algorithm Toolbox (GEATbx) of Pohlheim, 2000) and the SVM decomposition algorithm is coded in C++ language. The C++ module is called within MATLAB (in mGA) in evaluations of SVM (Figure 3.19). The SVM with mGA is run in HP workstation, 3.06GHz, 2GB Memory running on LINUX platform.

The programs coded in FORTRAN and C++ languages are generally faster than when they are coded in MATLAB. However, MATLAB is user-friendly and many detailed toolboxes are available facilitating easy coding. The novel techniques are easily adopted and made available in MATLAB toolboxes. Therefore, MATLAB is preferred to toolboxes of low level languages, which are not common and also not updated as frequently as MATLAB, in applications that demand lot of computational details.

The comparison that will be given in this section is made disregarding the differences in languages used. However, the differences in computer resources used have to be considered. A rough estimate is made between the performance of Pentium IV, 2.4 GHz, 512 MB RAM machine running Windows XP and the HP workstation, 3.06GHz, 2GB Memory running on LINUX platform by running a few programs which are compatible with both Windows XP and LINUX. It shows that the HP machine is approximately two times faster than Pentium IV.

The approximate time taken with different prediction methods for different time series is given in Table 3.17. It should be noted that the times reported are the total times taken to arrive at a certain optimal solution, i.e. considering the time taken to evaluate all possible parameter combinations. Table 3.17 shows that the local averaging technique is clearly the most efficient. Both global models, ANN and SVM, and the local polynomial model are computationally more time consuming. The time required for local polynomial models increases with the higher order due to the increased number of coefficients to be determined. The increased time taken in SVM with noise-free time series is notable. This is because the Quadratic Programming problem does not converge easily with noise-free data since it can go to higher accuracies by minimizing the error. When the data are noise free, they should ideally fit into the model and the $\xi^{(*)}$ in Eq. 3.10 should ideally be zero. Therefore, the iterations taken to minimize the objective function in Eq. 3.10 can be higher and hence cause longer computational time. ANN is comparatively more efficient than SVM in noise-free time series analysis. The increased time taken by ANN for noise-free Lorenz series, compared to other time series, is due to the large number hidden neurons used.

3.6 CONCLUSION

ANN models are shown to be robust in chaotic time series prediction, i.e., that a set of parameters chosen from a trial-and-error approach can yield good predictions on a wide variety of time series. The global prediction tools, ANN and SVM, showed superior performance over that of the local prediction tools in the Lorenz series and river flow time series. The ANN yielded more accurate predictions and was more efficient in noise-free series than SVM. Otherwise, ANN and SVM have similar prediction capabilities. The prediction performance of the global models as well as the local models deteriorated considerably when noise is present. Therefore, it is

interesting to explore the possibilities of improving their predictions on noisy data. The better predictions of global models (ANN and SVM) were achieved at a cost of increased computational time. The next chapter deals with how to improve prediction when noise is present. A subsequent chapter will look into the possibility of extracting a smaller data set in order to make the computations more efficient.

Table 3.1 Optimal phase space parameter sets with various models: Noise-free Lorenz series

Method	Lead time		
	1	3	5
	Optimal (m, τ, k)		
Local averaging	(2,6,6)	(4,3,6)	(6,3,4)
Local polynomial	(9,3,64) [2]	(9,3,56) [2]	(9,3,40) [2]
ANN	(7, 6, -)	(7, 6, -)	(7, 6, -)

Note: Values in brackets [ ] indicate the order of polynomials

Table 3.2 Prediction errors with various models on validation set: Noise-free Lorenz series

Lead time	Local model (Averaging)		Local model (Polynomial)		ANN	
	NRMSE	MAE	NRMSE	MAE	NRMSE	MAE
1	0.01827	0.1448	0.00034	0.0035	0.00031	0.0032 (98%)* (9%)**
3	0.02506	0.2136	0.00055	0.0049	0.00037	0.0036 (98%)* (27%)**
5	0.02755	0.2469	0.00095	0.0065	0.00047	0.0042 (98%)* (35%)**

* The percentage prediction improvement over local averaging model
** The percentage prediction improvement over local polynomial model

Table 3.3 Optimal phase space parameter sets with various models: 5% Noisy Lorenz time series

Method	Lead time		
	1	3	5
	Optimal (m, τ, k)		
Local averaging	(10,3,8)	(12,3,10)	(12,3,8)
Local polynomial	(10,3,96) [1]	(11,3,60) [1]	(10,3,68) [1]
ANN	(10, 3,-)	(10, 3,-)	(10, 3,-)

Note: Values in brackets [ ] indicate the order of polynomials

Table 3.4 Optimal phase space parameter sets with various models: 30% Noisy Lorenz time series

Method	Lead time		
	1	3	5
	Optimal (m, τ, k)		
Local averaging	(10,3,22)	(9,3,34)	(10,9,18)
Local polynomial	(8,3,96) [1]	(9,9,76) [1]	(9,9,92) [1]
ANN	(10, 3,-)	(9, 3,-)	(10, 3,-)

Note: Values in brackets [ ] indicate the order of polynomials

Table 3.5 Prediction errors with various models on validation set: 5% Noisy Lorenz series

Lead time	Local model (Averaging)		Local model (Polynomial)			ANN	
	NRMSE	MAE	Polynomial Order	NRMSE	MAE	NRMSE	MAE
1	0.0710	0.7264	1	0.0658	0.6723	0.0634	0.6395 (12%)* (5%)**
3	0.0743	0.7516	1	0.0719	0.7163	0.0667	0.6761 (10%)* (6%)**
5	0.0797	0.7909	1	0.0811	0. 8041	0.0719	0.7167 (9%)* (11%)**

* The percentage prediction improvement over local averaging models
** The percentage prediction improvement over local polynomial model

Table 3.6 Prediction errors with various models on validation set: 30% Noisy Lorenz series

Lead time	Local averaging		Local Polynomial			ANN	
	NRMSE	MAE	Polynomial Order	NRMSE	MAE	NRMSE	MAE
1	0.3792	4.0991	1	0.3888	4.1527	0.3793	4.0604 (1%)* (2%)**
3	0.3883	4.0600	1	0.4463	4.5391	0.3999	4.2094 (-4%)* (7%)**
5	0.4404	4.5283	1	0.4943	5.0603	0.4268	4.4450 (2%)* (12%)**

* The percentage prediction improvement over local averaging models
** The percentage prediction improvement over local polynomial model

Table 3.7 Optimal phase space parameter sets with various models: Mississippi river flow

Method	Lead time		
	1	3	5
	Optimal (m, τ, k)		
Local averaging	(2,1,6)	(2,1,8)	(2,1,8)
Local polynomial	(2,1,48) [2]	(2,1,36) [2]	(2,1,48) [2]
ANN	(3, 1, -)	(3, 1, -)	(3, 1, -)

Note: Values in brackets [ ] indicate the order of polynomials

Table 3.8 The optimal phase space parameter sets with various models: Wabash river flow

Method	Lead time		
	1	3	5
	Optimal (m, τ, k)		
Local averaging	(2,1,10)	(2,1,14)	(2,1,20)
Local polynomial	(3,1,50) [1]	(4,1,50) [1]	(4,1,44) [1]
ANN	(5, 1, -)	(5, 1, -)	(7, 1, -)

Note: Values in brackets [ ] indicate the order of polynomials

Table 3.9 Prediction errors with various models on validation set: Mississippi river flow

Lead time	Local model (Averaging)		Local model (Polynomial)		ANN	
	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)
1	0.0437	251.66	0.0412	225.13	0.0388	207.31 (18%)* (8%)**
3	0.1453	845.07	0.1371	810.59	0.1330	767.93 (9%)* (5%)**
5	0.2644	1586.18	0.2476	1512.50	0.2435	1465.24 (8%)* (3%)**

* The percentage prediction improvement over local averaging model
** The percentage prediction improvement over local polynomial model

Table 3.10 Prediction errors with various models on validation set: Wabash river flow

Lead time	Local model (Averaging)		Local model (Polynomial)		ANN	
	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)
1	0.0849	34.04	0.0641	27.24	0.0606	25.66 (25%)* (6%)**
3	0.2671	120.59	0.2522	118.91	0.2312	105.96 (12%)* (11%)**
5	0.4331	199.17	0.4317	206.63	0.4084	189.32 (5%)* (8%)**

* The percentage prediction improvement over local averaging model
** The percentage prediction improvement over local polynomial model

Table 3.11 Optimal phase space parameter sets with SVM for different time series

Time series	Lead time		
	1	3	5
	Optimal (m, τ)		
Noise-free Loenz	(9, 3)	(5, 6)	(6, 6)
5% noisy Lorenz	(10, 3)	(9, 3)	(10, 3)
30% noisy Lorenz	(10, 3)	(9, 3)	(10, 3)
Mississippi River	(5, 1)	(3, 1)	(4, 1)
Wabash River	(4, 1)	(5, 1)	(6, 1)

Table 3.12 Prediction errors with ANN and SVM on validation set: Noise-free Lorenz series

Lead time	ANN		SVM	
	NRMSE	MAE	NRMSE	MAE
1	0.00031	0.0032	0.00053	0.0044 (-38%)
3	0.00037	0.0036	0.00100	0.0064 (-78%)
5	0.00047	0.0042	0.00158	0.0088 (-110%)

Values in parenthesis ( ) are the percentage prediction improvement of SVM over ANN model

Table 3.13 Prediction errors with ANN and SVM on validation set: 5% Noisy Lorenz series

Lead time	ANN		SVM	
	NRMSE	MAE	NRMSE	MAE
1	0.0634	0.6395	0.0630	0.6392 (0%)
3	0.0667	0.6761	0.0675	0.6847 (-1%)
5	0.0719	0.7167	0.0732	0.7231 (-1%)

Values in parenthesis ( ) are the percentage prediction improvement of SVM over ANN model

Table 3.14 Prediction errors with ANN and SVM on validation set: 30% Noisy Lorenz series

Lead time	ANN		SVM	
	NRMSE	MAE	NRMSE	MAE
1	0.3793	4.0604	0.3707	3.9613 (2%)
3	0.3999	4.2094	0.3813	4.0228 (4%)
5	0.4268	4.4450	0.4167	4.3339 (2%)

Values in parenthesis ( ) are the percentage prediction improvement of SVM over ANN model

Table 3.15 Prediction errors with ANN and SVM on validation set: Mississippi time series

Lead time	ANN		SVM	
	NRMSE	MAE (m^3/s)	NRMSE	MAE (m^3/s)
1	0.0388	207.31	0.0395	206.99 (0%)
3	0.1330	767.93	0.1373	792.65 (-3%)
5	0.2435	1465.24	0.2511	1483.97 (-1%)

Values in parenthesis ( ) are the percentage prediction improvement of SVM over ANN model

Table 3.16 Prediction errors with ANN and SVM on validation set: Wabash time series

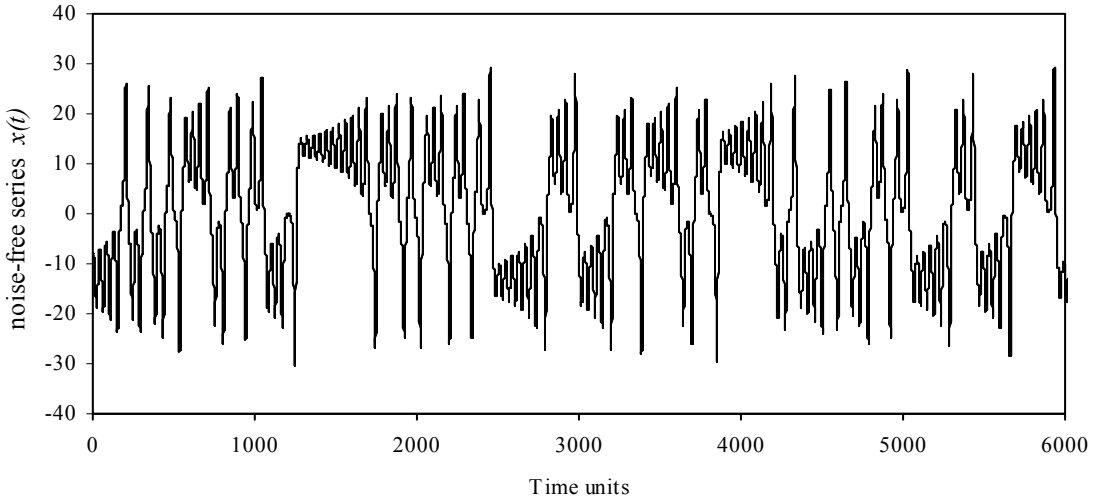
Lead time	ANN		SVM	
	NRMSE	MAE (m^3/s)	NRMSE	MAE (m^3/s)
1	0.0606	25.66	0.0638	25.17 (2%)
3	0.2312	105.96	0.2364	105.24 (1%)
5	0.4084	189.32	0.3982	173.46 (8%)

Values in parenthesis ( ) are the percentage prediction improvement of SVM over ANN model

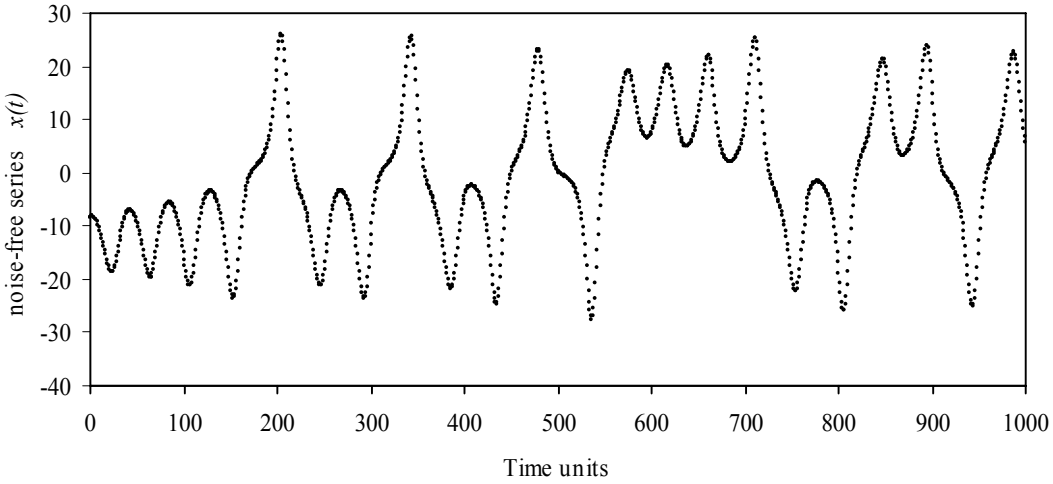
Table 3.17 Approximate computational time for different prediction methods with different time series

Time series	Local Averaging	Local Polynomial		ANN	SVM
		Order 1	Order 2		
Noise-free Lorenz	1 hr	97 hrs	267 hrs	26 hrs*	213 hrs
Noisy Lorenz	1 hr	97 hrs	267 hrs	4.3 hrs*	5 hrs
River flow	0.4 hrs	32 hrs	80 hrs	0.8 hrs*	10 hrs

* Times taken to derive the optimal parameters (the number of neurons and epochs etc) are not included



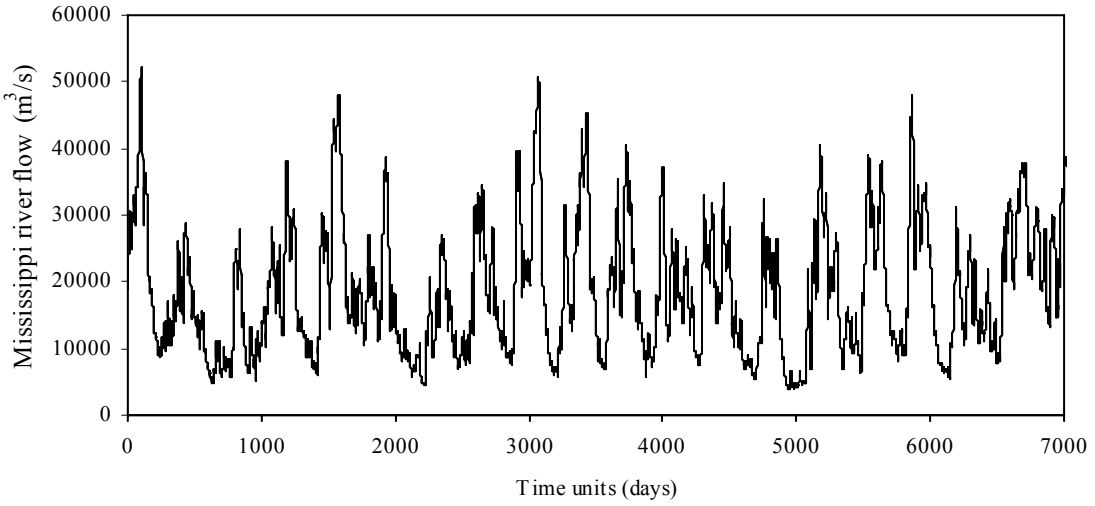
(a) All Points



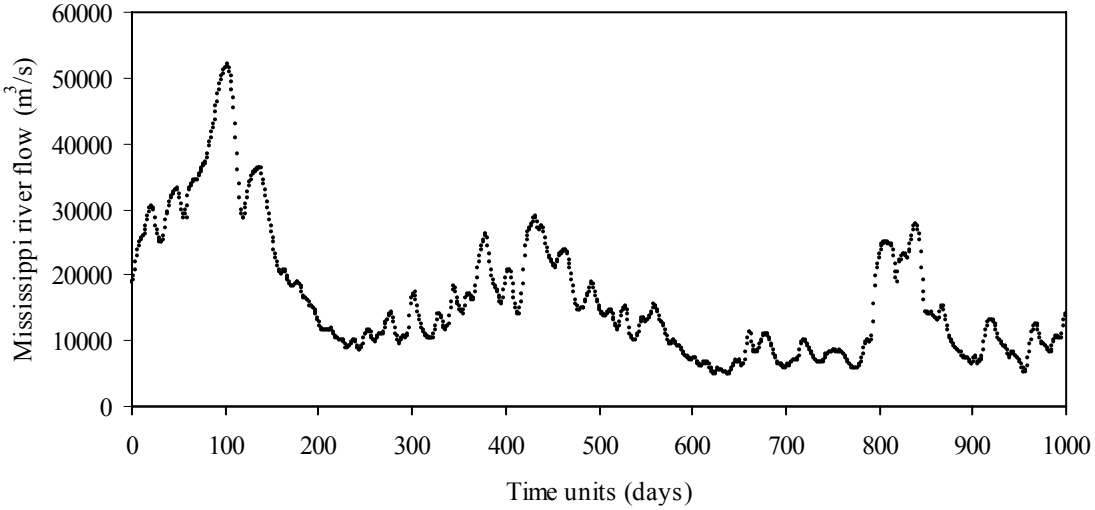
(b) First 1000 Points: close-up

Figure 3.1 x*(t)* component of Lorenz time series

Figure 3.2 Mississippi river catchment



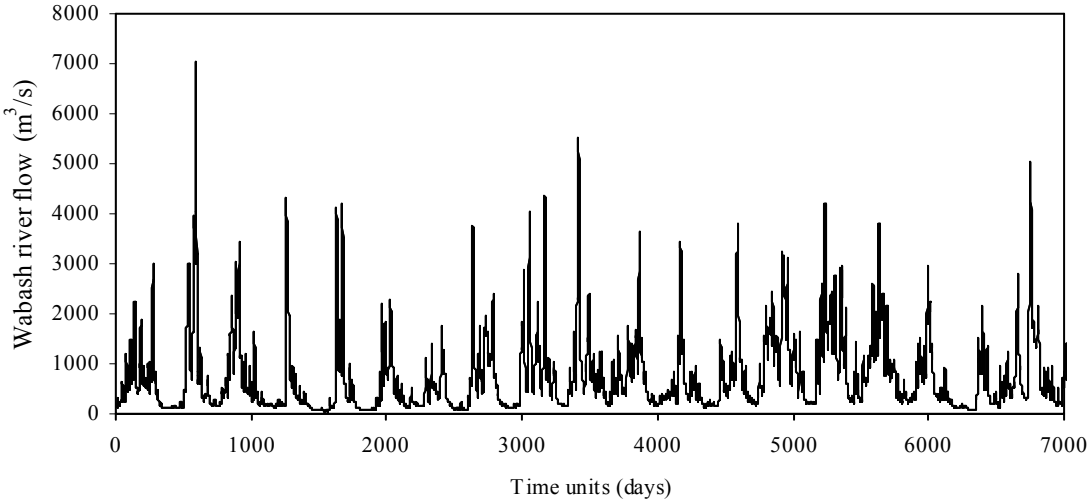
(a) All Points



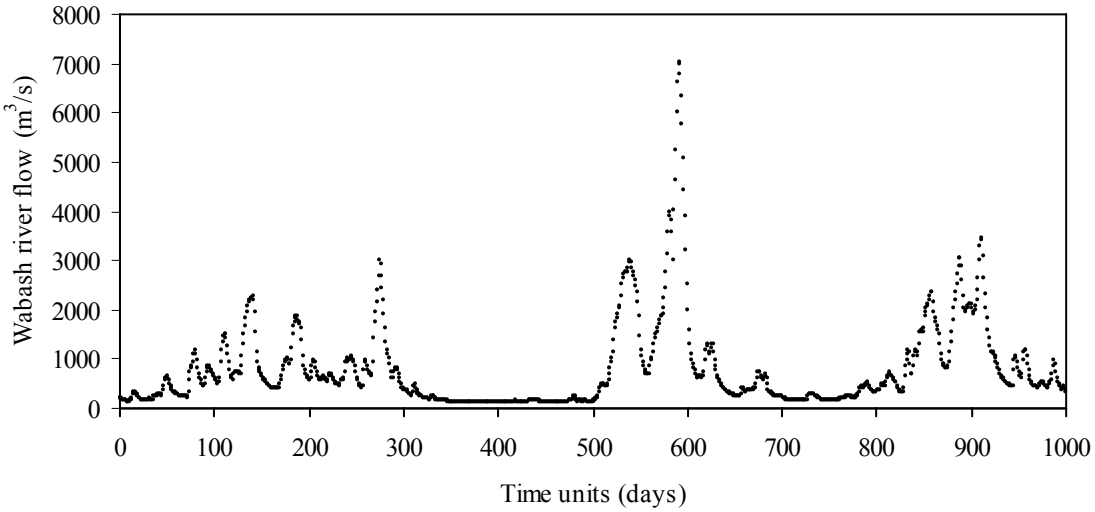
(b) First 1,000 Points: close-up

Figure 3.3 Mississippi river daily flow time series

Figure 3.4 Wabash river catchment



(a) All Points



(b) First 1,000 Points: close-up

Figure 3.5 Wabash river daily flow time series

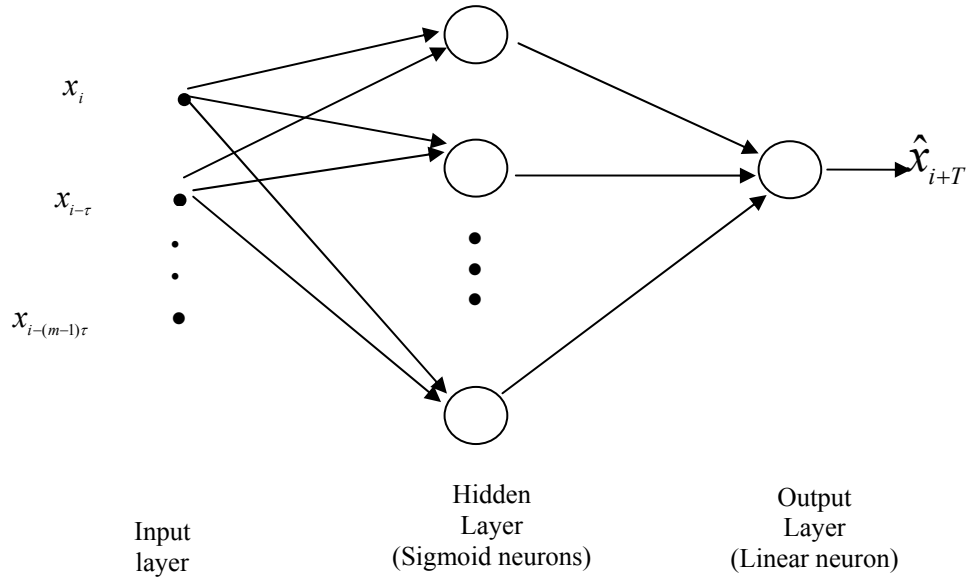


Figure 3.6 Architecture of Multi Layer Perceptron used in the study

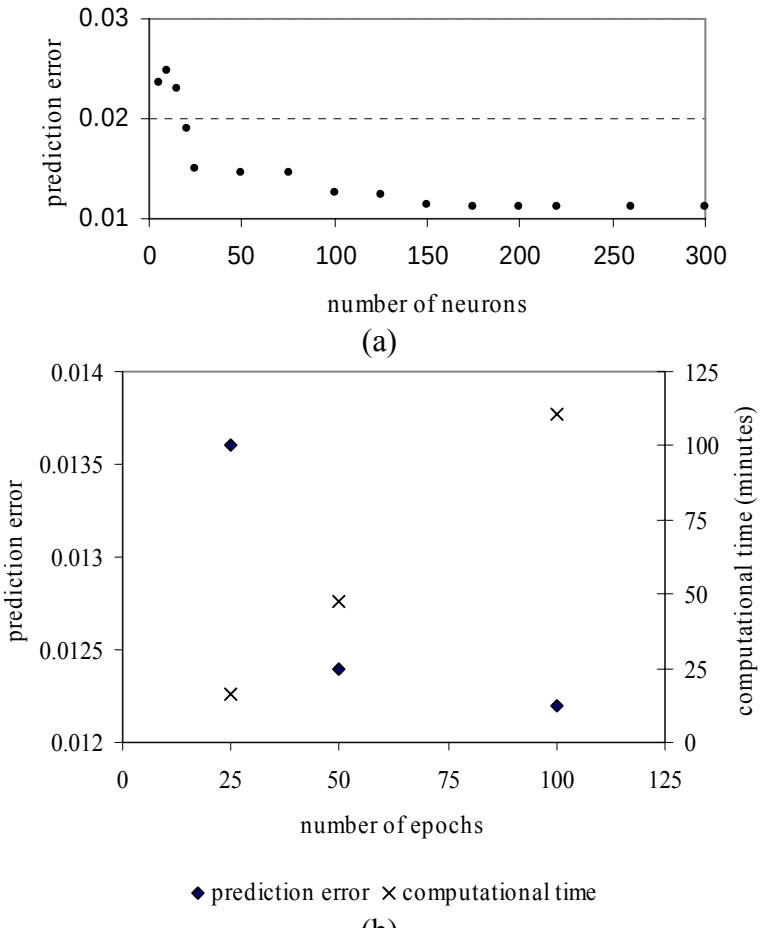


Figure 3.7 Variation of prediction errors and computational times with (a) number of hidden neurons and (b) number of epochs: Lorenz series (m = 5, τ =1, T=3 prediction)

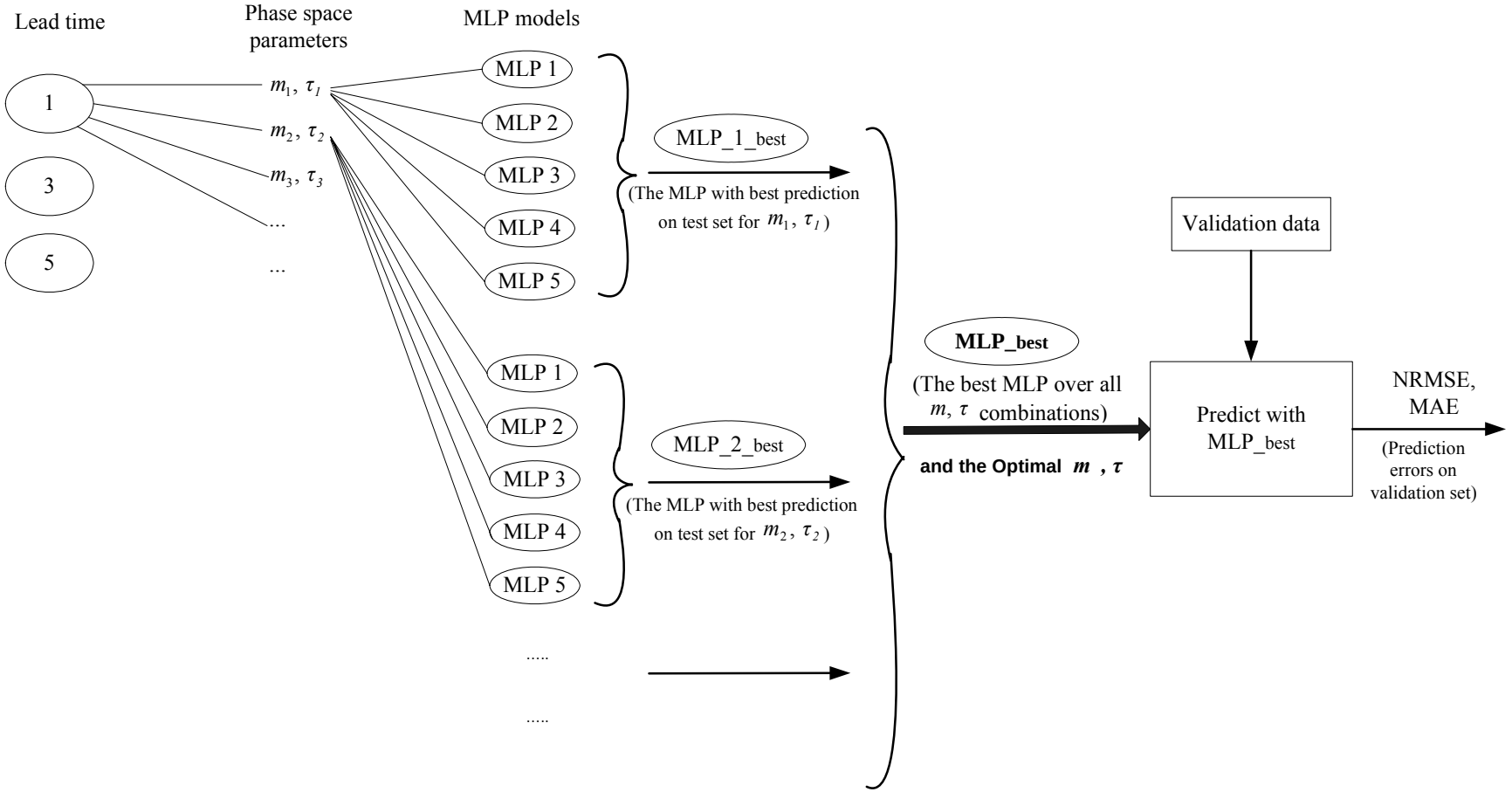
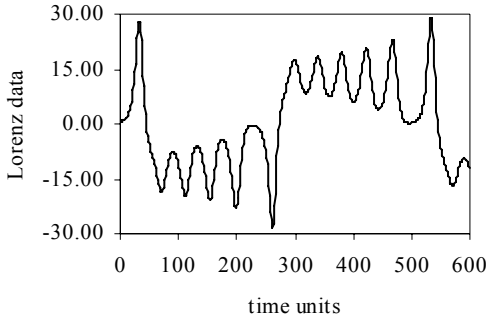
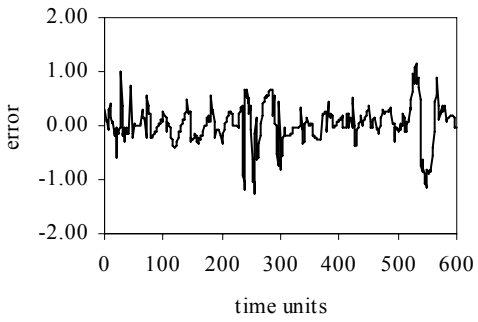


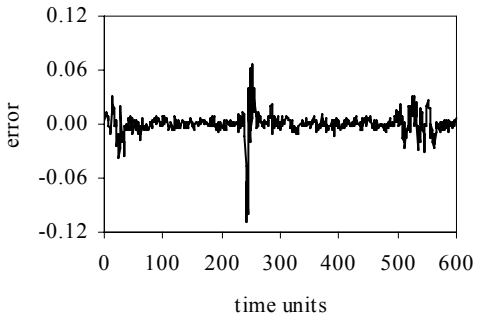
Figure 3.8 Schematic diagram of the selection procedure of optimally trained MLP



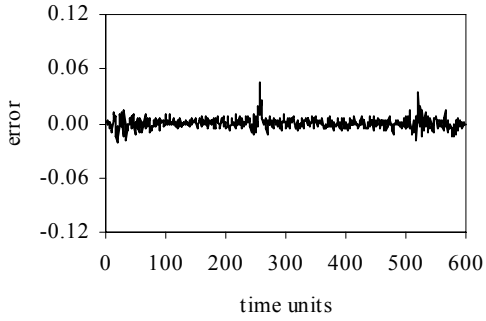
(a) Validation data



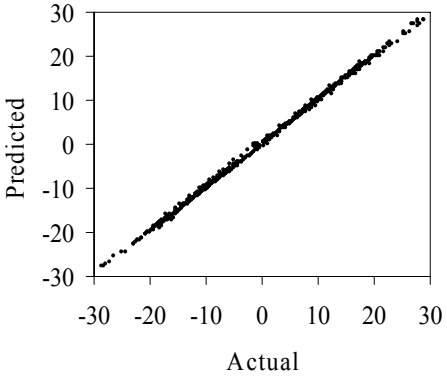
(b) Error on local averaging model



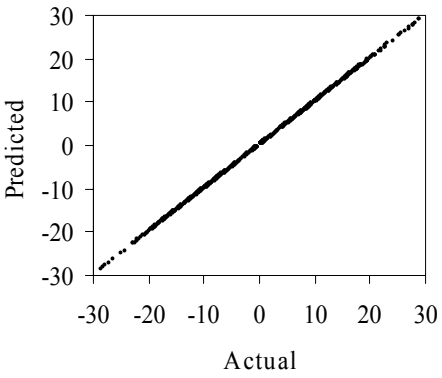
(c) Error on local polynomial model



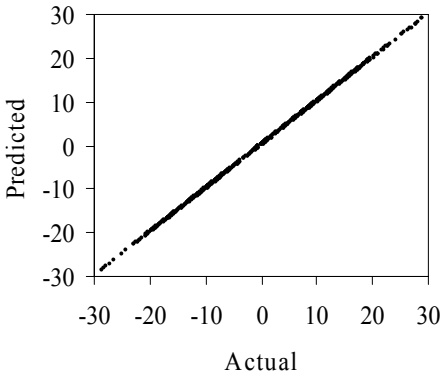
(d) Error on ANN model



(e) Scatter plot: Local averaging

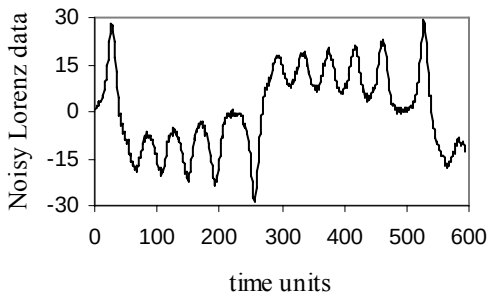


(f) Scatter plot: Local polynomial

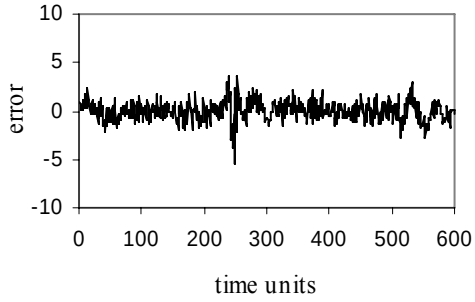


(f) Scatter plot: ANN

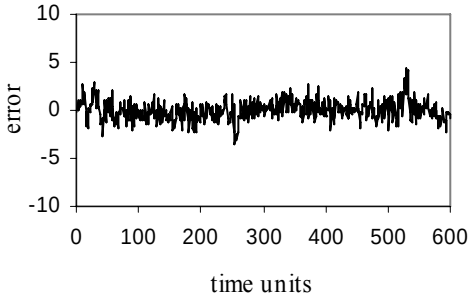
Figure 3.9 Validation data and prediction errors in lead-time 5 predictions of various models: noise-free Lorenz series



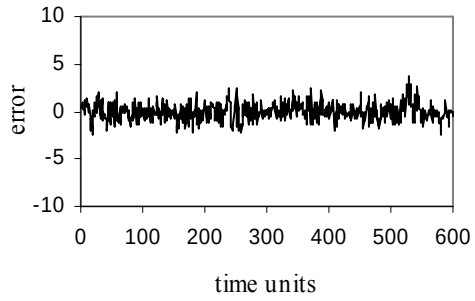
(a) Validation data



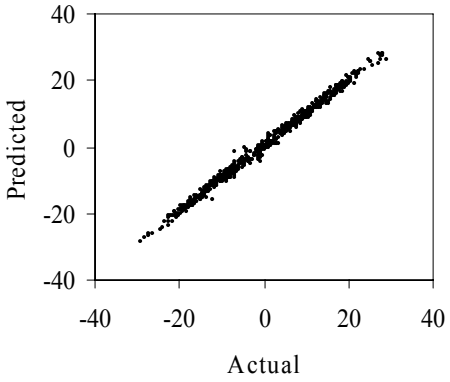
(b) Error on local averaging model



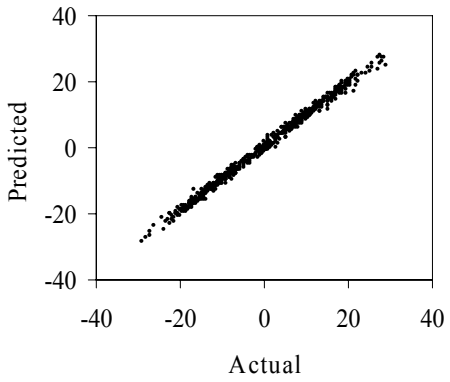
(c) Error on local polynomial model



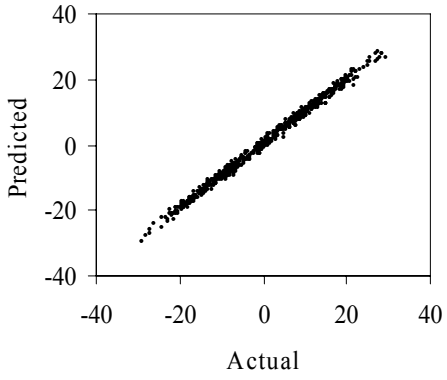
(d) Error on ANN model



(e) Scatter plot: Local averaging

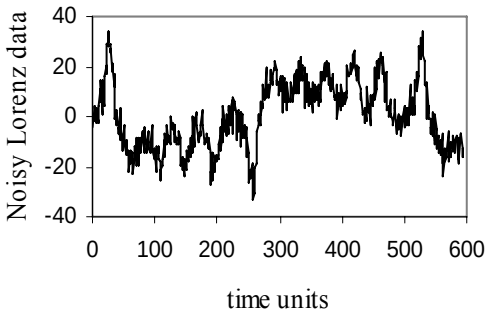


(f) Scatter plot: Local polynomial

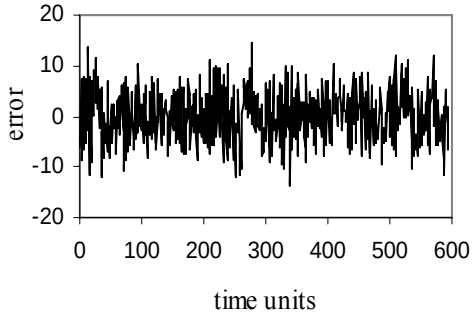


(f) Scatter plot: ANN

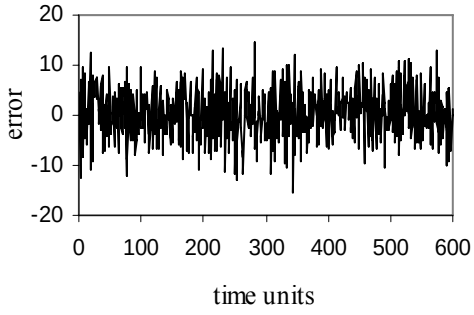
Figure 3.10 Validation data and prediction errors in lead-time 5 predictions of various models: 5% Noisy Lorenz series



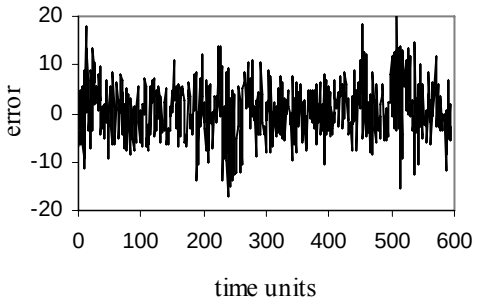
(a) Validation data



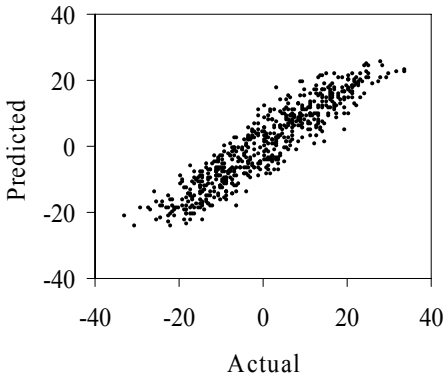
(b) Error on local averaging model



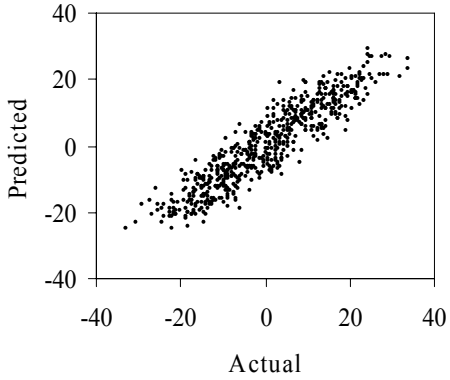
(c) Error on local polynomial model



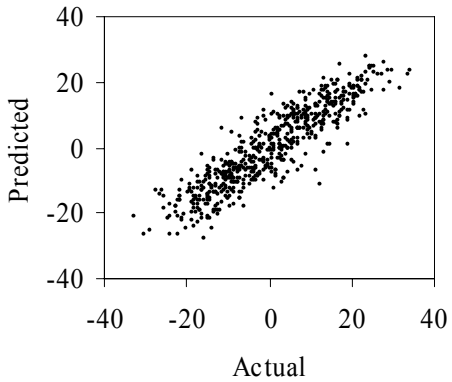
(d) Error on ANN model



(e) Scatter plot: Local averaging

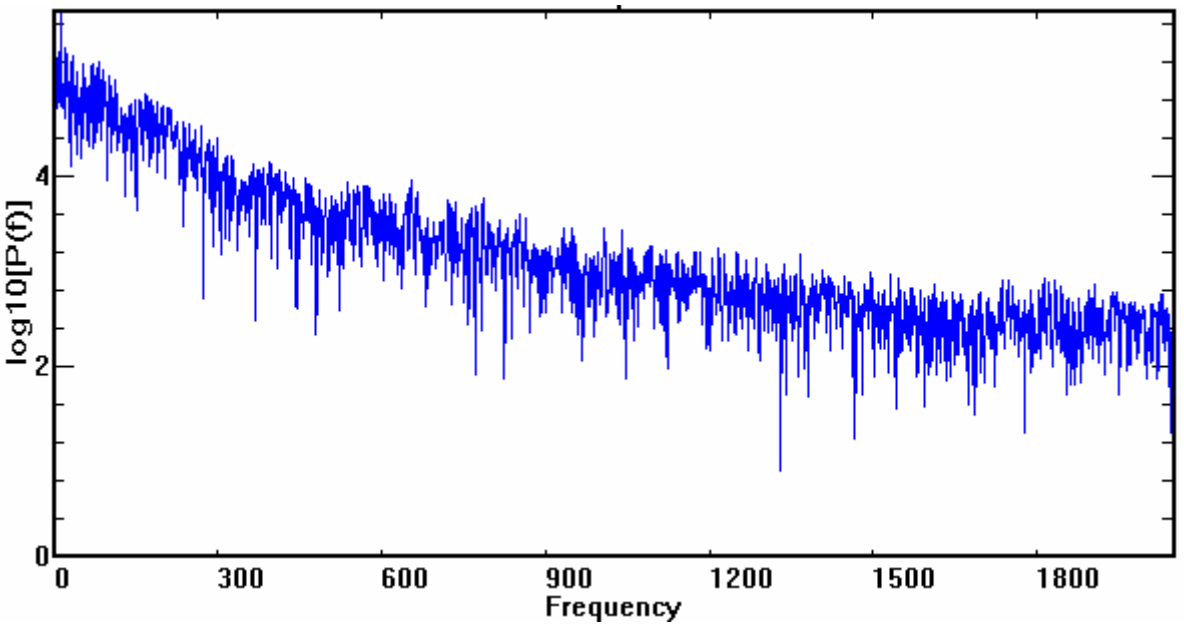


(f) Scatter plot: Local polynomial

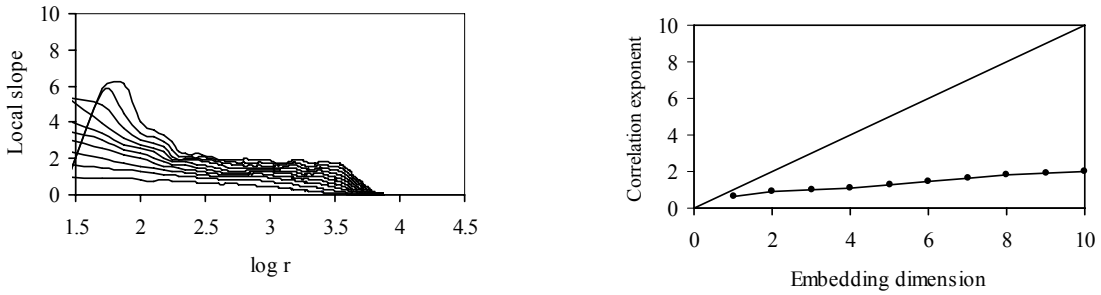


(f) Scatter plot: ANN

Figure 3.11 Validation data and prediction errors in lead-time 5 predictions of various models: 30% noisy Lorenz series

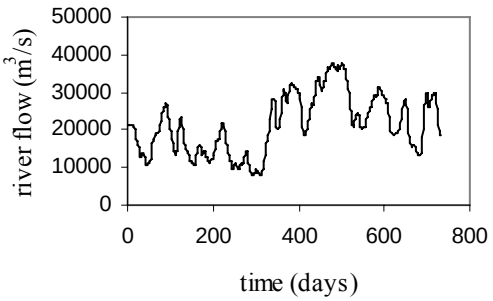


(a) Fourier power spectrum

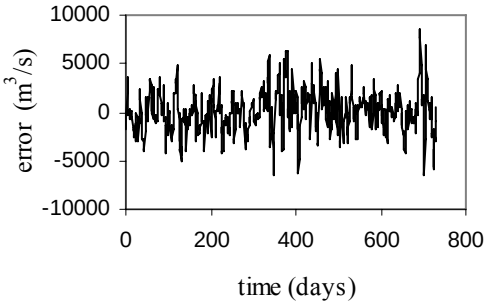


(b) Correlation integral analysis

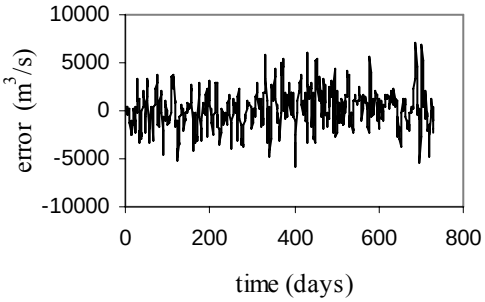
Figure 3.12 Correlation integral analysis and Fourier power spectrum on Wabash river flow time series



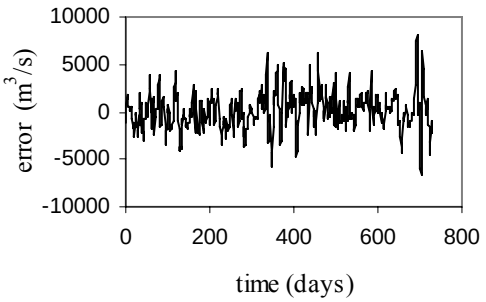
(a) Validation data



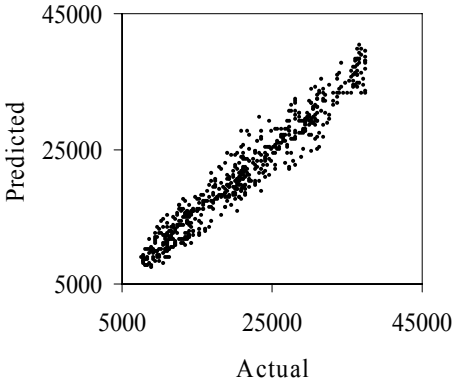
(b) Error on local averaging model



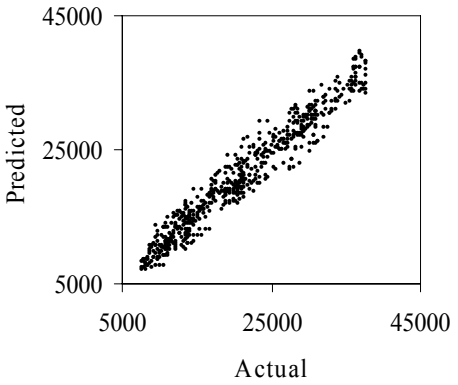
(c) Error on local polynomial model



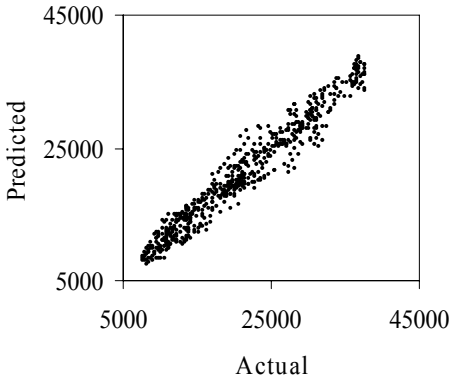
(d) Error on ANN model



(e) Scatter plot: Local averaging

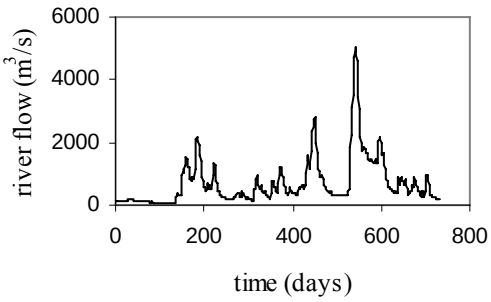


(f) Scatter plot: Local polynomial

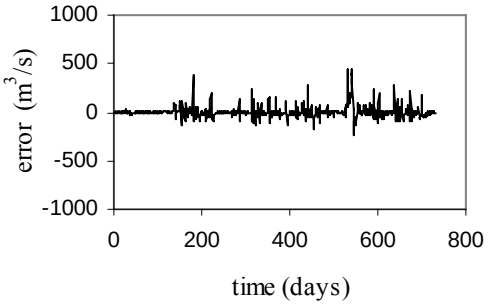


(f) Scatter plot: ANN

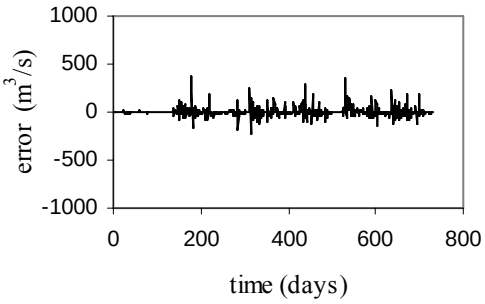
Figure 3.13 Validation data and prediction errors in lead-time 5 predictions of various models: Mississippi river flow time series



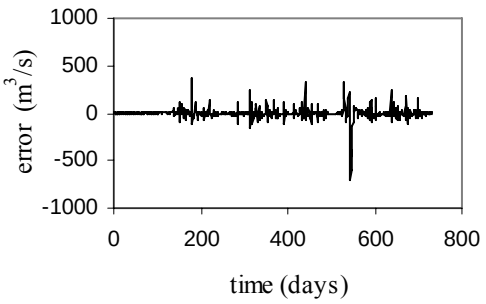
(a) Validation data



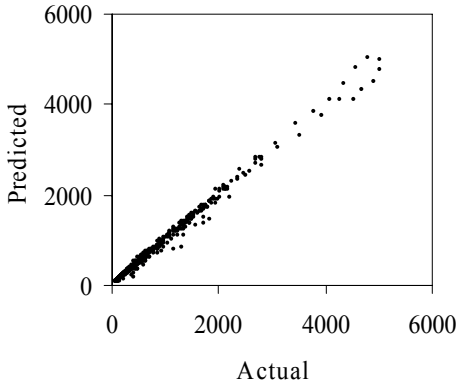
(b) Error on local averaging model



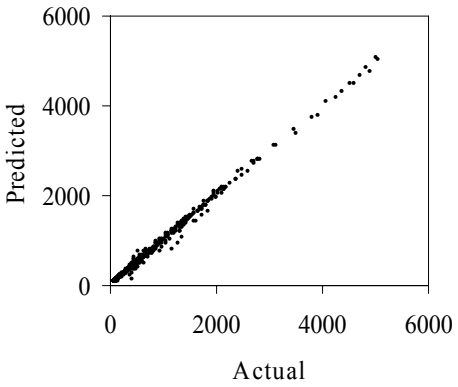
(c) Error on local polynomial model



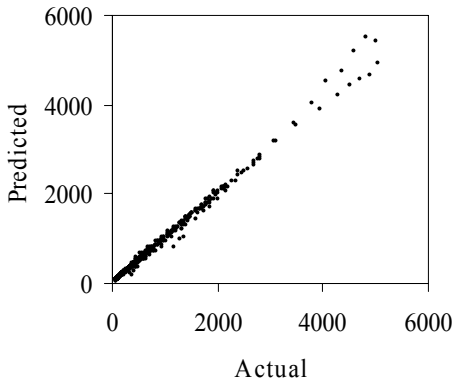
(d) Error on ANN model



(e) Scatter plot: Local averaging



(f) Scatter plot: Local polynomial



(f) Scatter plot: ANN

Figure 3.14 Validation data and prediction errors in lead-time 5 predictions of various models: Wabash river flow time series

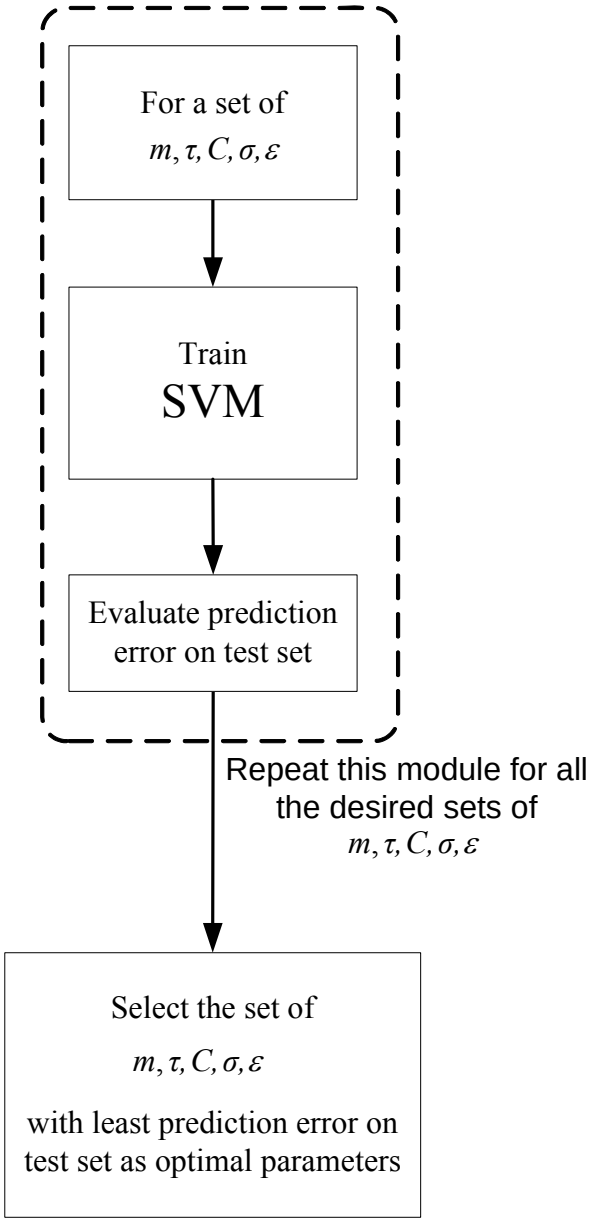


Figure 3.15 Schematic diagram of (m, t, c, std, eps) selection with SVM

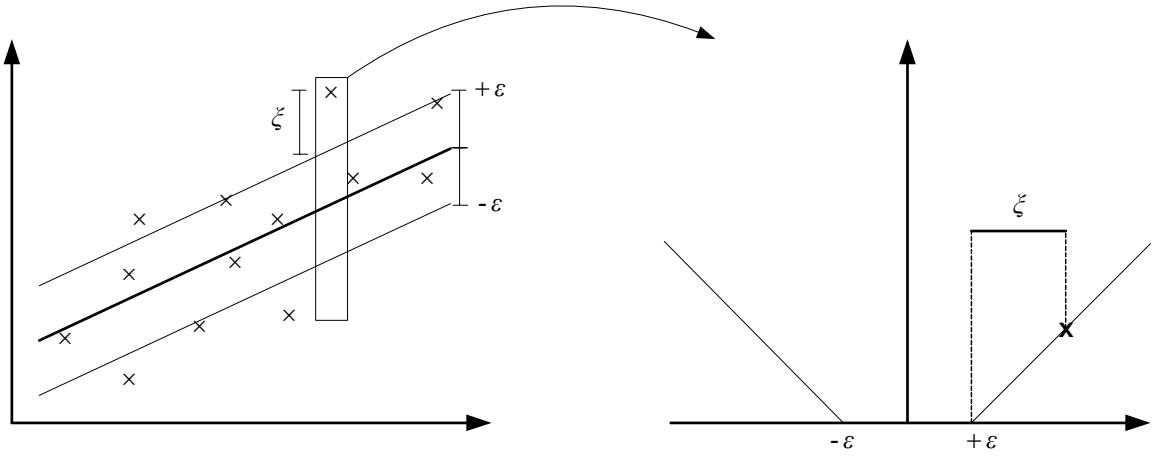


Figure 3.16 ε - insensitive loss function

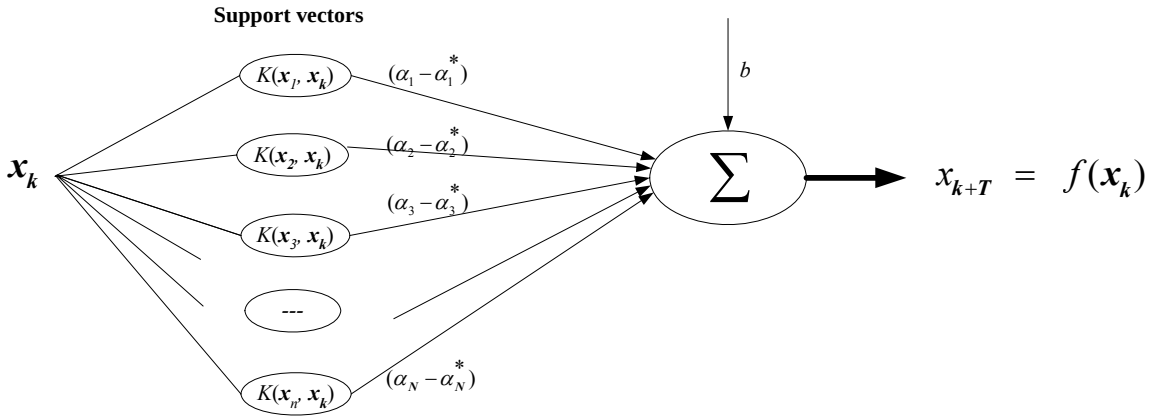


Figure 3.17 Prediction with support vector machine

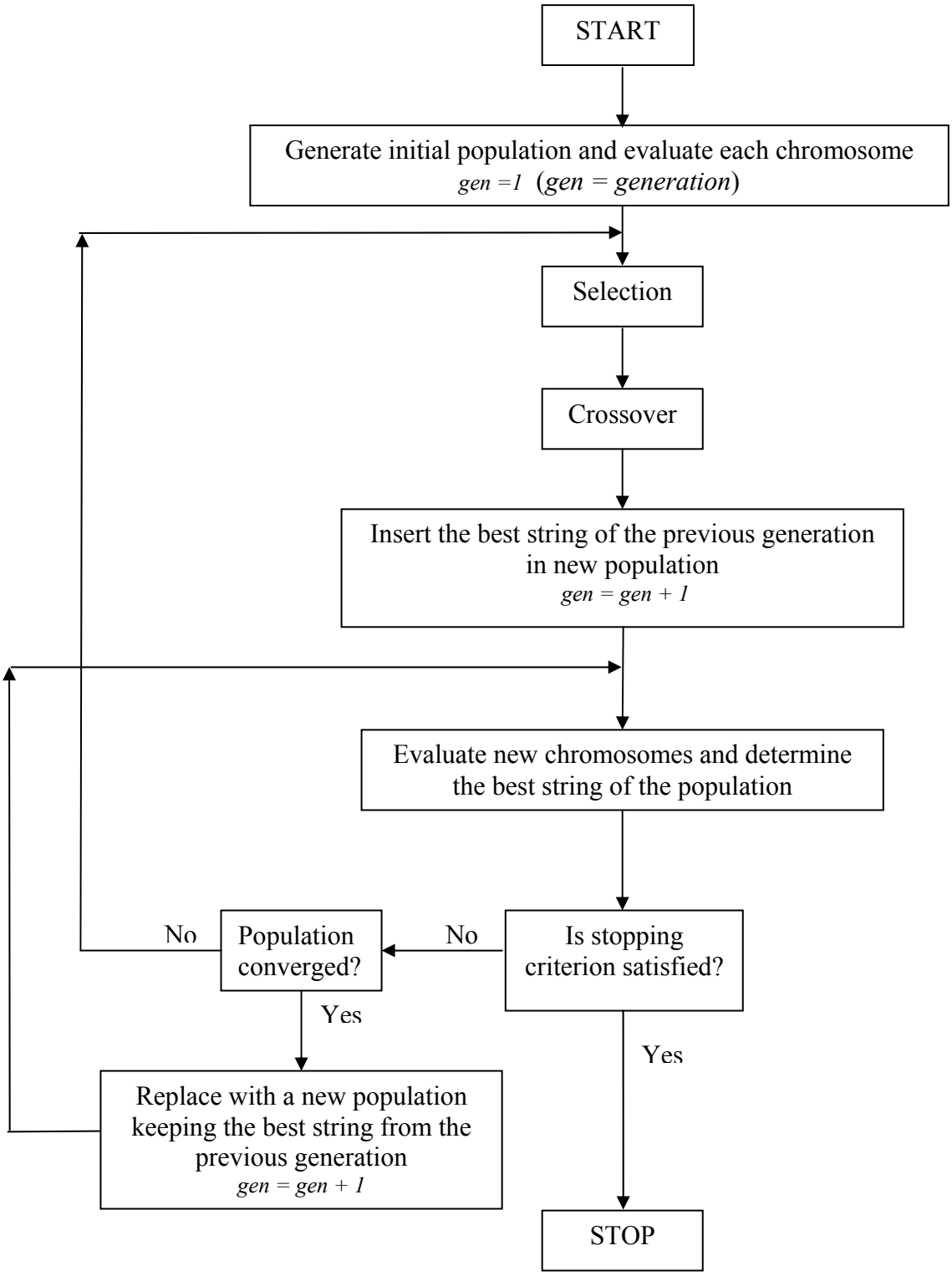


Figure 3.18 Schematic diagram of mGA

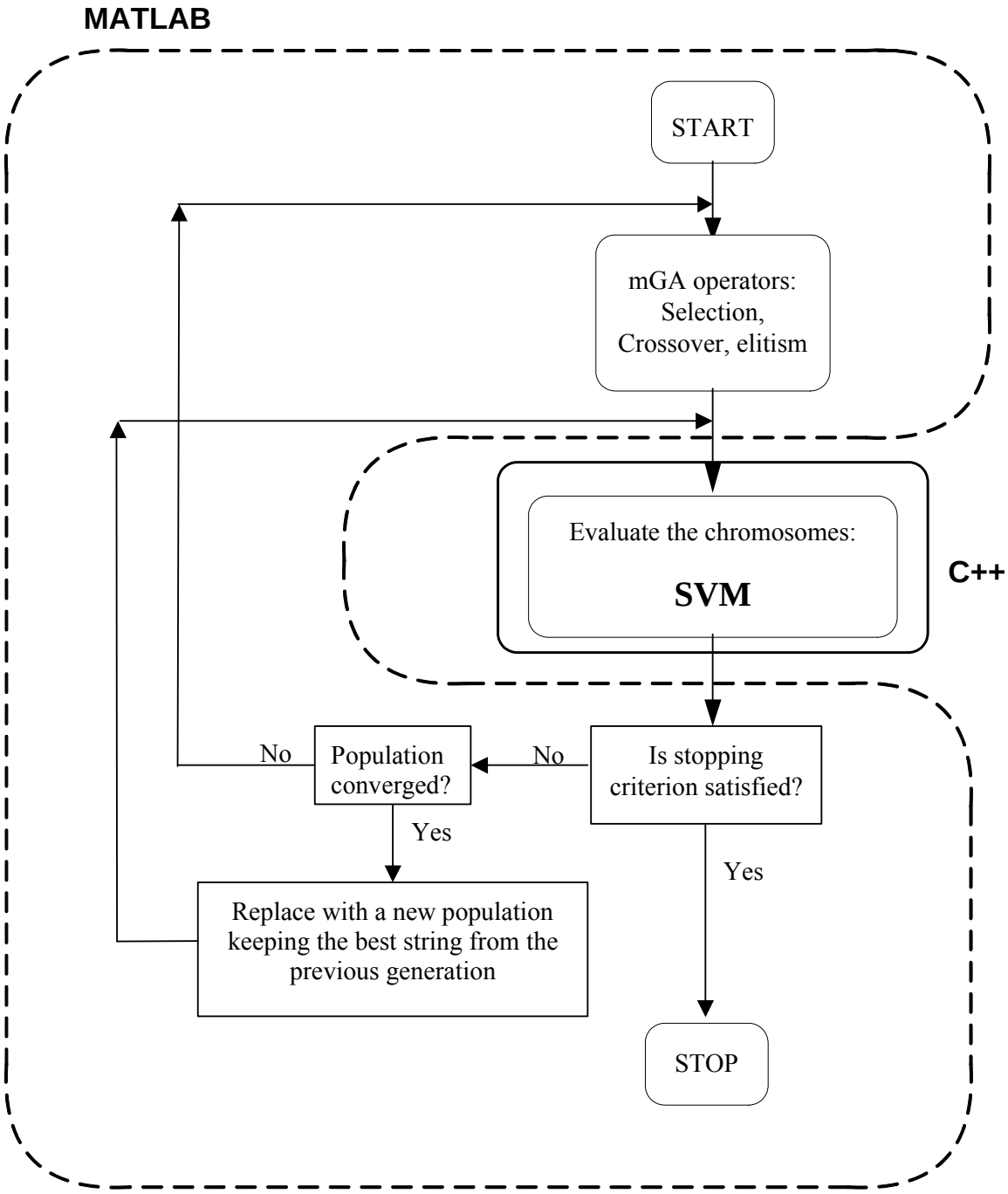


Figure 3.19 Implementation of SVM/ Matlab

CHAPTER 4

REAL-TIME NOISE REDUCTION AND PREDICTION OF CHAOTIC TIME SERIES WITH EXTENDED KALMAN FILTERING

4.1 INTRODUCTION

The need for investigating methods to improve predictions of noisy chaotic time series was highlighted in chapter 3. Investigating the use of noise reduction techniques to improve the quality of data and the prediction accuracy in real-time prediction applications is the main aim of this chapter. In addition, investigating the applicability of the popular state estimation technique in controls theory, the Kalman Filtering (nonlinear version – Extended Kalman filter (EKF)), in chaotic time series analysis is also conducted in this chapter.

As discussed in chapter 2, the procedures followed in the studies of noise reduction in chaotic hydrological time series (e.g. Porporato and Ridolfi, 1997; Kawamura et al., 1998; Sivakumar et. al. 1999b; c; Jayawardena and Gurung, 2000) are not suitable for real-time prediction applications. This study identifies the possible ways to improve real-time predictions of noisy chaotic time series, identifies appropriate techniques, and proposes a robust scheme. The Extended Kalman Filter (EKF) and simple nonlinear noise reduction is demonstrated on the proposed procedure. The validity of the scheme is assessed using two different prediction models: ANN and SVM.

In noise reduction of real data, totally reliable ways of verifying the noise removal are not available (Grassberger et al., 1993). It is, therefore, very important to test any noise reduction procedure on a data set where the true signal is known (Kantz and

Schreiber, 2004). In this study, all the methods and procedures are first tested on a chaotic Lorenz series contaminated with known noise levels. Gaussian random noise is added to noise-free chaotic time series (as described in Section 3.3.1) to obtain noisy series. Four different noise levels (noise levels are defined as in Section 3.3.1), from very mild noise to very high noise, 1%, 10%, 20%, 30%, are considered. Noises are generated with different seed numbers. Four sets of noisy series are generated for each noise level to allow a more thorough investigation. Analysis is first performed on known noisy Lorenz time series where comparison against noise-free series is possible; it is only then applied on real flow time series. The next section investigates some possible ways to improve prediction accuracy of noisy chaotic time series.

4.2 IMPROVING PREDICTION PERFORMANCE OF NOISY TIME SERIES

4.2.1 Introduction

Let us consider the evolution of a dynamical system that relates the current state to a future state,

$$\boldsymbol{x}_{n+1} = f(\boldsymbol{x}_n) \tag{4.1}$$

There are two main ingredients: (1) the function *f*, and (2) the current input $\boldsymbol{x}_n$. In the case of chaotic time series analysis, an approximation to the function *f*, $\hat{f}$, is derived out of past records. Real world data contains noise and the recorded data are not exactly the series $\{x_i\}$ but some measured series $\{y_i\}$ expressed as

$$y_i = x_i + \upsilon_i \tag{4.2}$$

where υ_i is measurement noise. If the data contains noise, the function *f* cannot be accurately approximated and that inevitably affects the prediction accuracy. When the

input data ($\boldsymbol{x}_n$) contains noise (i.e. $\boldsymbol{y}_n$), again, the prediction performance is going to suffer. Therefore, the possible ways of improving prediction performance are,

(1) configuring better prediction models f_b using noise reduced data.

(2) Using noise reduced inputs, $\boldsymbol{x}_{nred}$, instead of $\boldsymbol{y}_n$.

(3) Using a combination of (1) and (2).

To configure a better prediction model, a noise reduced data set can be used. For this purpose, noise reduction techniques can be applied off-line. In off-line applications, to reduce noise in a particular record, both past and future data records of that data record can be used (Figure 4.1(a)). To reduce noise in input data (i.e. the current record) noise reduction has to be incorporated in real-time (or on-line). In this case, only the past records of the current record can be used to yield a noise reduced estimate of the current record (Figure 4.1(b)). Earlier studies on noise reduction, criticized by Elshorbagy et al (2002b), have used off-line noise reduction to reduce noise in validation data as well. However, as noted earlier, such offline noise reduction techniques can not reduce noise in validation inputs in real-time prediction.

A review on popular nonlinear noise reduction techniques (Schreiber and Grassberger, 1991; Schreiber, 1993) shows that they cater off-line applications. Out of the possibilities listed above, the only way to improve predictions of data driven models by incorporating off-line noise reduction is to configure models using noise reduced data. The earlier noise reduction attempts also implicitly assumed that models configured with noise reduced data can improve predictions (e.g. Porporato and Ridolfi, 1997; Sivakumar et al., 1999b). Whether such a model derived from noise-reduced data will be more effective on chaotic time series prediction applications than a model derived of noisy data is, however, uncertain. It is possible that even a perfect model might not produce better prediction with noisy inputs due to its high sensitivity

to initial conditions. Therefore, the following section verifies, through simulations, whether such models derived with noise-reduced data would yield better predictions. A later section will verify whether noise-reduced inputs for models would improve the prediction accuracy.

4.2.2 Do models trained with less noisy data produce better predictions?

As noted earlier, it seems natural to expect that prediction models derived with noise-reduced data will perform better. This section explores whether such models indeed give better prediction performance with chaotic time series when noisy inputs (data whose quality is less than that of the data used to derive prediction model) are used in validation. Noise induced chaotic Lorenz time series are used in the simulations. The prediction performance of models derived out of noise free data compared with that of performance of models derived from noisy data, when noisy validation data is given as inputs, is used to evaluate the models' performance. ANN is used as the prediction model. In this chapter, ANN models are trained in the same way as in the last chapter except that hyperbolic tangent transfer function is used for the neurons. The Chapter 3 showed that the ANN models trained with noise-free data produces almost perfect predictions. Therefore, in this chapter, these nearly perfect models will be used to denote the case of perfect models.

A prediction performance comparison is made between ANN trained with noise free data (the best possible model) and ANN trained with noisy data, using noisy validation data as inputs. Two different noise levels are used: 1% and 30%. Prediction errors are measured against both noise-free and noisy data. The procedure is shown in Figure 4.2 (a) and (b) respectively for the models trained with noisy data and the model trained with noise-free data. In the goodness-of-fit measure expressions, Eq. 3.4 and Eq. 3.5 where $\hat{x}_i$ is the predicted value, x_i is the noisy series value when

comparing against noisy validation data and x_i is the noise-free series value when comparing against noise-free validation data.

Table 4.1 shows the results for 1% noisy data and 30% noisy data as inputs to model trained with noise free data and the models trained with noisy data of levels 1% and 30% respectively. Since the prediction performances are consistent with both error measures, NRMSE and MAE (as noted in Chapter 3 as well), only the MAE values are shown from here onwards. The prediction performances on validation set measured against both noisy data and noise-free data, marked as A, B, C and D on Table 4.1, correspond to the A, B, C and D shown on the Figure 4.2. The results show that the model trained with noisy data (e.g. model trained with 1% noisy data) performs even slightly better than the model trained with noise-free data when the models are subjected to data inputs of similar noise level (e.g. 1% noisy validation inputs). The reason why the ANN model trained with noisy data performs better may be because it is more robust and consistent on the noisy data of similar amount of noise. The sophisticated model trained with noise-free data on the other hand, may produce worse predictions with noisy inputs due to divergence of trajectories due to sensitivity to initial conditions.

In addition to the above nearly ideal case of model trained with noise-free data, several other models are also trained with data of 1%, 10% and 20% noise levels. These models can be regarded as models trained with less noisy data to input data of noise levels higher than, 1%, 10%, and 20% respectively. In the simulations performed using the above models too, the prediction accuracy of the model trained with the data, which have the same level of noise as the input data, yields better or equally good prediction accuracy as that of the models trained with less noisy data (See Appendix F).

The above results show important implications: (1) a model trained with data of same level of noise as its inputs is more robust than a model trained with less noisy data, and (2) training models with noise-reduced data may not necessarily yield better prediction accuracy when the input data is not noise-reduced. It is, therefore, interesting to verify whether noise-reduced data inputs may improve the prediction performance of the models.

4.2.3 Do noise-reduced data inputs cause models to predict better?

This section assesses how ANN models trained with noisy data perform with noisy input data whose level of noise is less than those data used to train the ANN model. Simulations are performed using an ANN model trained with 30% noisy Lorenz series, with noise-free, 1%, 10%, 20% and 30% noisy Lorenz data as inputs. As before, the prediction performance is measured against both noisy and noise-free data as shown in Figure 4.3. The results are shown in Table 4.2. The results show that the ANN model trained with 30% noise provides better prediction performance when the input data are less noisy (compare columns 1 – 4 with column 5). It is interesting to note that the prediction improvement is reflected in prediction error against noisy data too. This is advantageous since there is hope in using prediction error measured against noisy data as a criterion for identifying noise reduction in real world data where the noise free signal is unknown. As before, simulations are performed with models trained with other noise levels, 1%, 10% and 20%, where input data of noise levels less than 1%, 10% and 20% respectively are used as inputs. The results confirm the above observation; a model trained with noisy data performs better (as expected) with less noisy validation inputs (See Appendix F).

Findings from Sections 4.2.2 and 4.2.3 can be summarized as in Table 4.3. Let the noise level of a series be x %, and let it be y % ($y < x$) after noise reduction. The

possible ways of model training, validation and their outcomes are shown in Table 4.3. Row (1) of Table 4.3 shows the general case where no noise reduction is involved. Row (2) shows that a model trained with noisy data can provide better predictions when the noise level of input data is reduced. However, a model trained with noise-reduced data may not necessarily yield better predictions than a model trained with noisy data when noisy input data are used (Row (3)). Section 4.2.2 noticed that a model trained with data of same level of noise as its inputs is more robust. Therefore, it can be deduced that a model trained with low noise level should improve the prediction accuracy if it is supported with input data of equally low noise or even lower (Row (4)). Therefore, the key factor in improving the prediction accuracy is the noise-reduced input data.

The above results indicate the necessity of adopting noise reduction in real-time, i.e. noise reduction of the current input (current record), to yield a better prediction to the future. For this, one has to look for noise reduction techniques capable of real-time applications. In controls literature, the Kalman filter and its variants are very popular in real-time state estimation of dynamical systems. The possibility of using this technique for noise reduction in chaotic time series is investigated in the next section.

4.3 EXTENDED KALMAN FILTER IN PREDICTION OF NOISY CHAOTIC TIME SERIES

In this section, the Extended Kalman filter is first introduced. This is followed by an investigation on the appropriateness of EKF for real-time noise reduction. A noisy data-driven state-space model of EKF for chaotic phase space predictions is then derived. Finally, the application of the EKF with ANN trained with noisy data is demonstrated on noise-induced Lorenz time series.

4.3.1 Extended Kalman Filter

Kalman filter is originally derived for linear systems. But most of the real world systems are nonlinear and variations of KF have been proposed for nonlinear systems. A direct extension of linear KF for nonlinear systems is the Extended Kalman filter (EKF), which is used in this study. Improved variations and other forms of KF, which are claimed to be better than EKF, are also available in the literature to deal with nonlinearities. To understand the Kalman filter technique and the EKF formulation, it is essential to understand the linear Kalman filter. Linear Kalman filter is first explained in this section and then the EKF in its most basic form is presented. Out of the many references the derivation of these filters has been documented, this chapter follows Welch and Bishop (2004).

The Kalman filter addresses the problem of trying to estimate the state $\boldsymbol{x} \in R^n$ of a discrete-time controlled process governed by the linear stochastic difference equation

$$\boldsymbol{x}_k = \boldsymbol{A}_k \boldsymbol{x}_{k-1} + \boldsymbol{B}_k \boldsymbol{u}_k + \boldsymbol{w}_{k-1} \tag{4.3}$$

with a measurement $\boldsymbol{z} \in R^m$ that is

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \upsilon_k \tag{4.4}$$

The random variables $\boldsymbol{w}_k$ and υ_k represent the process and measurement noise respectively. They are assumed to be independent (of each other), white, and of normal probability distributions,

$$\begin{aligned} p(\boldsymbol{w}) &\sim N(0, \boldsymbol{Q}) \\ p(\upsilon) &\sim N(0, \boldsymbol{R}) \end{aligned} \tag{4.5}$$

In practice, the process noise covariance $\boldsymbol{Q}$ and measurement noise covariance $\boldsymbol{R}$ matrices might change with each time step or measurement; however, here it is

assumed that they are constant. The set of relations given in Eq. 4.3 and Eq. 4.4 is called the state-space model.

The $n \times n$ matrix $\boldsymbol{A_k}$ in Eq. 4.3 relates the state at the previous time step $k-1$ to the state at the current step k in the absence of either a driving function or process noise. The $n \times l$ matrix $\boldsymbol{B_k}$ relates the optional control input $\boldsymbol{u} \in R^l$ to the state $\boldsymbol{x}$. The $m \times n$ matrix $\boldsymbol{H_k}$ in the measurement equation relates the state to the measurement $\boldsymbol{z_k}$.

Define $\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-} \in \boldsymbol{R}^{\boldsymbol{n}}$ to be *a priori* state estimate at step k given the knowledge of the process prior to step k, and $\hat{\boldsymbol{x}}_{\boldsymbol{k}} \in \Re^{\boldsymbol{n}}$ to be our *a posteriori* state estimate at step k given measurement $\boldsymbol{z_k}$. Then the *a priori* and *a posteriori* estimate errors are,

$$\begin{aligned} \boldsymbol{e}_{\boldsymbol{k}}^{-} &= \boldsymbol{x}_{\boldsymbol{k}} - \hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-} \\ \boldsymbol{e}_{\boldsymbol{k}} &= \boldsymbol{x}_{\boldsymbol{k}} - \hat{\boldsymbol{x}}_{\boldsymbol{k}} \end{aligned} \tag{4.6}$$

The *a priori* estimate error covariance is then

$$\boldsymbol{P}_{\boldsymbol{k}}^{-} = \boldsymbol{E}\left[\boldsymbol{e}_{\boldsymbol{k}}^{-}.\boldsymbol{e}_{\boldsymbol{k}}^{-T}\right] \tag{4.7}$$

and the *a posteriori* estimate error covariance is

$$\boldsymbol{P}_{\boldsymbol{k}} = \boldsymbol{E}\left[\boldsymbol{e}_{\boldsymbol{k}}.\boldsymbol{e}_{\boldsymbol{k}}^{T}\right] \tag{4.8}$$

The main goal is to find a posteriori state estimate $\hat{\boldsymbol{x}}_{\boldsymbol{k}}$ as a linear combination of an *a priori* estimate $\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-}$ and the new measurement $\boldsymbol{z_k}$. This is equivalent to finding a linear combination of an *a priori* estimate $\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-}$ and a weighted difference between an actual measurement $\boldsymbol{z_k}$ and a measurement prediction $\boldsymbol{H}\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-}$ as shown below (see Appendix G for details),

$$\hat{\boldsymbol{x}}_{\boldsymbol{k}} = \hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-} + \boldsymbol{K}_{\boldsymbol{k}}\left(\boldsymbol{z}_{\boldsymbol{k}} - \boldsymbol{H}\hat{\boldsymbol{x}}_{\boldsymbol{k}}^{-}\right) \tag{4.9}$$

the difference $\left(\boldsymbol{z}_k - \boldsymbol{H}\hat{\boldsymbol{x}}_k^-\right)$ in Eq. 4.9 is called the measurement innovation. The $n \times m$ matrix $\boldsymbol{K_k}$ (gain or blending factor) is chosen so that the *a posteriori* error covariance (Eq. 4.8) is minimized.

Substituting Eq. 4.9 and Eq. 4.4 in Eq. 4.6

$$\boldsymbol{e}_k = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right)\boldsymbol{e}_k^- + \boldsymbol{K}_k \upsilon_k \tag{4.10}$$

substitute Eq. 4.10 in Eq. 4.8

$$\boldsymbol{P}_k = \boldsymbol{E}\left\{\left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right)\boldsymbol{e}_k^-\left[\boldsymbol{e}_k^{-T}\left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right)^T + \upsilon_k{}^T \boldsymbol{K}_k{}^T\right] + \boldsymbol{K}_k \upsilon_k\left[\boldsymbol{e}_k^{-T}\left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right)^T + \upsilon_k{}^T \boldsymbol{K}_k{}^T\right]\right\}$$

By definition

$$\boldsymbol{E}\left[\upsilon_k \upsilon_k{}^T\right] = \boldsymbol{R}_k \tag{4.11}$$

and as a result of measurement errors being uncorrelated,

$$\boldsymbol{E}\left[\boldsymbol{e}_k^- . \upsilon_k{}^T\right] = \boldsymbol{E}\left[\upsilon_k \ . \boldsymbol{e}_k^{-T}\right] = 0 \tag{4.12}$$

thus,

$$\boldsymbol{P}_k = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right)\boldsymbol{P}_k^-\left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right)^T + \boldsymbol{K}_k \boldsymbol{R}_k \boldsymbol{K}_k{}^T \tag{4.13}$$

Optimum choice of $\boldsymbol{K}_k$. The criterion for choosing the optimal $\boldsymbol{K}_k$ is to minimize a weighted scalar sum of the diagonal elements of the error covariance matrix $\boldsymbol{P}_k$. Thus a cost function can be chosen as

$$\boldsymbol{J}_k = \boldsymbol{E}\left[\boldsymbol{e}^T . \boldsymbol{S} . \boldsymbol{e}\right] \tag{4.14}$$

where $\boldsymbol{S}$ is any positive semidefinite matrix. It can be shown that the optimal estimate is independent of $\boldsymbol{S}$. Choosing $\boldsymbol{S} = \boldsymbol{I}$ yields,

$$\boldsymbol{J}_k = \boldsymbol{trace}\left[\boldsymbol{P}_k\right] \tag{4.15}$$

To find the $\boldsymbol{K}_k$ which provides a minimum, the partial derivative of $\boldsymbol{J}_k$ with respect to $\boldsymbol{K}_k$ is equated to zero. The following relation for partial derivative of the trace of the product of two matrices $\boldsymbol{A_o}$ and $\boldsymbol{B_o}$ (with $\boldsymbol{B_o}$ symmetric) is used:

$$\frac{\partial}{\partial \boldsymbol{A}}\left[\boldsymbol{trace}\left(\boldsymbol{A}_o \boldsymbol{B}_o \boldsymbol{A}_o{}^T\right)\right] = 2\boldsymbol{A}_o \boldsymbol{B}_o \tag{4.16}$$

From Eqs. 4.13 and 4.15 the following equation can be formed

$$-2\left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right)\boldsymbol{P}_k^- \boldsymbol{H}_k{}^T + 2\boldsymbol{K}_k \boldsymbol{R}_k = 0 \tag{4.17}$$

Solving for Eq. 4.17 for $\boldsymbol{K}_k$ yields

$$\boldsymbol{K}_k = \boldsymbol{P}_k^- \boldsymbol{H}_k{}^T \left[\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k{}^T + \boldsymbol{R}_k\right]^{-1} \tag{4.18}$$

which is referred to as the Kalman gain matrix. Substituting Eq. 4.18 into Eq. 4.13 and after some manipulations,

$$\boldsymbol{P}_k = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right)\boldsymbol{P}_k^- \tag{4.19}$$

which is the optimized value of the updated estimation error covariance matrix.

The *a priori* estimates are obtained in the "prediction step" or the Time Update step. The projection of the quantities through time is called the Time Update step or "prediction step". The *a priori* estimates are obtained as,

$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{A}\hat{\boldsymbol{x}}_{k-1} + \boldsymbol{B}\boldsymbol{u}_k \tag{4.20}$$

$$\boldsymbol{P}_k^- = \boldsymbol{A}\boldsymbol{P}_{k-1}\boldsymbol{A}^T + \boldsymbol{Q} \tag{4.21}$$

Once $\boldsymbol{K}_k$ and $\boldsymbol{P}_k$ is found the *a posteriori* estimates can be obtained in the "correction step" or the Measurement Update step as follows,

$$\boldsymbol{K}_k = \boldsymbol{P}_k^- \boldsymbol{H}_k{}^T \left[\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k{}^T + \boldsymbol{R}_k\right]^{-1} \tag{4.22}$$

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k\left(\boldsymbol{z}_k - \boldsymbol{H}\hat{\boldsymbol{x}}_k^-\right) \tag{4.23}$$

$$\boldsymbol{P}_k = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right)\boldsymbol{P}_k^- \tag{4.24}$$

The initial conditions for the filter can be given as

$$\boldsymbol{E}\left[\boldsymbol{x}_0\right] = \hat{\boldsymbol{x}}_0 \tag{4.25}$$

$$\boldsymbol{E}\left[\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)\left(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0\right)^T\right] = \boldsymbol{P}_o$$

When no other information is available, an arbitrary value may be chosen as the initial state estimate.

Extended Kalman Filter

The above formulations are valid for linear systems. For nonlinear systems the Extended Kalman filter can be derived as follows Welch and Bishop (2004).

Assuming the system has a state vector $\boldsymbol{x} \in \boldsymbol{R}^n$ and is governed by the nonlinear stochastic difference equation

$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}, \boldsymbol{w}_{k-1}) \tag{4.26}$$

with a measurement $\boldsymbol{z} \in \boldsymbol{R}^m$ where

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k, \upsilon_k) \tag{4.27}$$

In Eqs. 4.26 and 4.27, $\boldsymbol{w}_k$ and υ_k represent, as earlier, process and measurement noises. In practice, the individual values of the noise $\boldsymbol{w}_k$ and υ_k are unknown and one may approximate the state and measurement vector without them as

$$\tilde{\boldsymbol{x}}_k = \boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_{k-1}, 0) \tag{4.28}$$

$$\tilde{\boldsymbol{z}}_k = \boldsymbol{h}(\tilde{\boldsymbol{x}}_k, 0) \tag{4.29}$$

where $\hat{\boldsymbol{x}}_k$ is some a posteriori estimate of the state. It should be noted that a fundamental flaw of the EKF is that the distributions of the various random variables are no longer normal after undergoing their respective nonlinear transformations. The EKF is simply an *ad hoc* state estimator that only approximates the optimality of Bayes' rule by linearization.

Taylor approximation to linearize the estimates around Eq. 4.28 and Eq. 4.29 yields

$$\boldsymbol{x}_k \approx \tilde{\boldsymbol{x}}_k + \boldsymbol{A}(\boldsymbol{x}_{k-1} - \hat{\boldsymbol{x}}_{k-1}) + \boldsymbol{W}\boldsymbol{w}_{k-1} \tag{4.30}$$

$$\boldsymbol{z}_k \approx \tilde{\boldsymbol{z}}_k + \boldsymbol{H}(\boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k) + \boldsymbol{V}\upsilon_k \tag{4.31}$$

where

- $\boldsymbol{x}_k$ and z_k are the actual state and measurement vectors,
- $\widetilde{\boldsymbol{x}}_k$ and $\widetilde{\boldsymbol{z}}_k$ are the approximate state and measurement vectors from Eqs. 4.28 and 4.29,
- $\hat{\boldsymbol{x}}_k$ is an *a posteriori* estimate of the state at step k,
- the random variables $\boldsymbol{w}_k$ and υ_k represent the process and measurement noise as in Eqs. 4.3 and 4.4,
- $\boldsymbol{A}$ is the Jacobian matrix of partial derivatives of $\boldsymbol{f}$ with respect to $\boldsymbol{x}$,

$$\boldsymbol{A}_{[i,j]} = \frac{\partial \boldsymbol{f}_{[i]}}{\partial \boldsymbol{x}_{[j]}}\left(\hat{\boldsymbol{x}}_{\boldsymbol{k}-1}, \boldsymbol{u}_k, 0\right) \tag{4.32}$$

- $\boldsymbol{W}$ is the Jacobian matrix of partial derivatives of $\boldsymbol{f}$ with respect to $\boldsymbol{w}$,

$$\boldsymbol{W}_{[i,j]} = \frac{\partial \boldsymbol{f}_{[i]}}{\partial \boldsymbol{w}_{[j]}}\left(\hat{\boldsymbol{x}}_{\boldsymbol{k}-1}, \boldsymbol{u}_k, 0\right) \tag{4.33}$$

- $\boldsymbol{H}$ is the Jacobian matrix of partial derivatives of $\boldsymbol{h}$ with respect to $\boldsymbol{x}$,

$$\boldsymbol{H}_{[i,j]} = \frac{\partial \boldsymbol{h}_{[i]}}{\partial \boldsymbol{x}_{[j]}}\left(\widetilde{\boldsymbol{x}}_k, 0\right) \tag{4.34}$$

- $\boldsymbol{H}$ is the Jacobian matrix of partial derivatives of $\boldsymbol{h}$ with respect to υ,

$$\boldsymbol{V}_{[i,j]} = \frac{\partial \boldsymbol{h}_{[i]}}{\partial \upsilon_{[j]}}\left(\widetilde{\boldsymbol{x}}_k, 0\right) \tag{4.35}$$

Note that for simplicity of notation the subscript k of Jacobians ***A, W, H,*** and ***V*** are not used although they are different at each time step.

The prediction error can now be defined as

$$\widetilde{\boldsymbol{e}}_{x_k} = \boldsymbol{x}_k - \widetilde{\boldsymbol{x}}_k \tag{4.36}$$

while the measurement residual is defined as

$$\tilde{\boldsymbol{e}}_{z_k} = \boldsymbol{z}_k - \tilde{\boldsymbol{z}}_k \tag{4.37}$$

From Eqs. 4.30 , 4.31, 4.36 and 4.37, the following expressions can be obtained

$$\tilde{\boldsymbol{e}}_{x_k} = \boldsymbol{A}\left(\boldsymbol{x}_{k-1} - \hat{\boldsymbol{x}}_{k-1}\right) + \varepsilon_k \tag{4.38}$$

$$\tilde{\boldsymbol{e}}_{z_k} = \boldsymbol{H}\tilde{\boldsymbol{e}}_{x_k} + \eta_k \tag{4.39}$$

where ε_k and η_k are independent random variables having zero mean and covariance matrices $\boldsymbol{WQW}^T$ and $\boldsymbol{VRV}^T$ with $\boldsymbol{Q}$ and $\boldsymbol{R}$ as defines in Eq. 4.5.

It should be noted that Eqs. 4.38 and 4.39 are linear and closely resemble Eqs. 4.3 and 4.4 of the linear Kalman filter. This motivates the use of a second (hypothetical) Kalman filter to estimate the prediction error $\tilde{\boldsymbol{e}}_{x_k}$. This estimate, $\hat{\boldsymbol{e}}_k$, could then be used together with Eq. 4.36 to obtain the a posteriori state estimates for the original nonlinear process as

$$\hat{\boldsymbol{x}}_k = \tilde{\boldsymbol{x}}_k + \hat{\boldsymbol{e}}_k \tag{4.40}$$

The random variables in Eqs. 4.38 and 4.39 have approximately the following probability distributions:

$$\begin{aligned} p\left(\tilde{\boldsymbol{e}}_{x_k}\right) &\sim \boldsymbol{N}\left(0,\ \boldsymbol{E}\left[\tilde{\boldsymbol{e}}_{x_k}.\tilde{\boldsymbol{e}}_{x_k}{}^T\right]\right) \\ p\left(\varepsilon_k\right) &\sim \boldsymbol{N}\left(0,\ \boldsymbol{WQ}_k\boldsymbol{W}^T\right) \\ p\left(\eta_k\right) &\sim \boldsymbol{N}\left(0,\ \boldsymbol{VR}_k\boldsymbol{V}^T\right) \end{aligned} \tag{4.41}$$

With those approximations and letting the predicted value of $\hat{\boldsymbol{e}}_k$ be zero, the Kalman filter equation used to estimate $\hat{\boldsymbol{e}}_k$ is

$$\hat{\boldsymbol{e}}_k = \boldsymbol{K}_k\tilde{\boldsymbol{e}}_{z_k} \tag{4.42}$$

Substituting Eq. 4.42 into Eq. 4.40 and using Eq. 4.37 lead to

$$\begin{aligned} \hat{\boldsymbol{x}}_k &= \tilde{\boldsymbol{x}}_k + \boldsymbol{K}_k\tilde{\boldsymbol{e}}_{z_k} \\ &= \tilde{\boldsymbol{x}}_k + \boldsymbol{K}_k\left(\boldsymbol{z}_k - \tilde{\boldsymbol{z}}_k\right) \end{aligned} \tag{4.43}$$

Eq. 4.43 can be used for the measurement update in the Extended Kalman filter with $\widetilde{\boldsymbol{x}}_k$ and $\widetilde{\boldsymbol{z}}_k$ obtained from Eqs. 4.28, 4.29 and the Kalman gain obtained from Eq. 4.20 with appropriate substitution for the measurement error covariance.

The final EKF Time Update and Measurement Update equations can be given as follows.

Time Update:

$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_{k-1}, 0) \tag{4.44}$$

$$\boldsymbol{P}_k^- = \boldsymbol{A}_k \boldsymbol{P}_{k-1} \boldsymbol{A}_k^T + \boldsymbol{W}_k \boldsymbol{Q}_{k-1} \boldsymbol{W}_k^T \tag{4.45}$$

Measurement Update:

$$\boldsymbol{K}_k = \boldsymbol{P}_k^- \boldsymbol{H}_k^T (\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{V}_k \boldsymbol{R}_k \boldsymbol{V}_k^T)^{-1} \tag{4.46}$$

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k (\boldsymbol{z}_k - \boldsymbol{h}(\hat{\boldsymbol{x}}_k^-, 0)) \tag{4.47}$$

$$\boldsymbol{P}_k = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \boldsymbol{P}_k^- \tag{4.48}$$

The discrete Kalman filter cycle is illustrated in Figure 4.4

4.3.2 Appropriateness of EKF in real-time noise reduction of chaotic time series

Ideally EKF assumes a perfect state-space model. As shown in Chapter 2, the few studies that have applied EKF in chaotic time series prediction have used the exact governing equations or a nearly 'perfect' model derived from noise-free data. However, this kind of 'perfect' models are not in existence for real world systems such as chaotic systems where explicit governing equations are not known. In chaotic time series analysis all one can obtain is a data-driven model trained from, most probably, noisy data. Justification of the use of less than perfect model in EKF can be given as follows.

In order for EKF to make an improved 'prediction' with the more 'correct' estimate of the state (see Figure 4.4), the prediction model should be able to produce better predictions with less noisy inputs. This is because the EKF uses measurement update value (which is supposed to be less noisy than the measurement) to predict in time update step. If the prediction model is able to give a better 'prediction' with the less noisy measurement update ('correction'), the next measurement update step ('correction'), which uses the predicted value will, in turn, yield a better estimate ('correction'); thus the whole algorithm will lead to improved estimates and predictions. Section 4.2.3 showed that for chaotic time series, input data with lower noise level leads models trained with noisy data to better predictions. This shows the use of models trained with noisy data in EKF, for chaotic time series prediction, to yield better prediction, is feasible.

EKF uses only the past records of a certain point to come up with a better estimate for that point. Therefore, it has the potential to be used as a real-time noise reduction technique. It can be noticed that Kalman filter estimates can be considered as noise reduced data for systems where the system state is directly observed, i.e. in systems where $\mathbf{H} = \mathbf{I}$ (where $\mathbf{I}$ is the identity matrix) in the measurement model (Eq. 4.4). The chaos application is a special case where the observations are directly used to determine the states and, therefore, the KF estimates can be considered as noise reduced data without any conversion errors due to transformations through $\mathbf{H}$.

The above discussion shows that it is possible to use a model trained with noisy data in EKF and it is also possible to use EKF as a real-time noise reduction technique in chaotic time series analysis. However, it should also be noted that there is a limit in the quality of predictions that can be achieved by these noisy data trained models in EKF. For example, with the model trained with 30% noisy data (Table 4.2) even if the

EKF were able to yield perfect estimates (i.e. noise-free data), the prediction error measure, MAE, will never be less than 0.7356 (column 1 of Table 4.2) due to the imperfectness of the model. These results indicate that an iterative approach (i.e. training a model with noise-reduced data and using it then in EKF to make better estimations; and continuously repeating the procedure) may provide better predictions. However, this is beyond the scope of the current study.

The next section explains how the noisy data trained ANN model is incorporated in the state-space model of the Extended Kalman filter.

4.3.3 Noisy data trained ANN model in EKF

In chaotic time series prediction, one is interested in deriving an approximation for the dynamical rule,

$$\boldsymbol{x}_{t+T} = \boldsymbol{f}_T(\boldsymbol{x}_t, \boldsymbol{x}_{t-\tau}, \ldots \boldsymbol{x}_{t-(\boldsymbol{m}-1)\tau}) \tag{4.49}$$

relating a future coordinate to past coordinates. The approximation can be expressed as

$$x_{t+T} = F_T(x_t, x_{t-\tau}, \ldots x_{t-(m-1)\tau}) + w_t \tag{4.50}$$

where F_T is the approximate model and w_t is the residual error. Assuming both the time delay (τ) and the prediction horizon (T) to be unity, the relation can be expressed as

$$x_k = f(x_{k-1}, x_{k-2}, \ldots x_{k-m}) + w_k \tag{4.51}$$

Since the observations inevitably contain noise, the observations can be related to actual signal as,

$$y_t = x_t + \upsilon_t \tag{4.52}$$

Eqs. 4.51 and 4.52 closely resemble the state-space model of Extended Kalman filter given in Eqs. 4.26 and 4.27 with the exception of scalars at the places of vectors in

EKF state-space model. Following Haykin (2001) the state-space representation can be formulated as

$$\boldsymbol{x}_k = \boldsymbol{F}\left(\boldsymbol{x}_{k-1}\right) + \boldsymbol{B}\boldsymbol{w}_k \tag{4.53}$$

$$\begin{bmatrix} x_k \\ x_{k-1} \\ : \\ : \\ x_{k-(m-1)} \end{bmatrix} = \begin{bmatrix} f(x_{k-1}, x_{k-2}, \ldots x_{k-m}) \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ : & : & : & : \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ x_{k-2} \\ : \\ x_{k-m} \end{bmatrix} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ : \\ : \\ 0 \end{bmatrix} w_k$$

$$\begin{aligned} y_k &= \boldsymbol{C}\boldsymbol{x}_k + \upsilon_k \\ &= \begin{bmatrix} 1 & 0 & . & . & 0 \end{bmatrix} \boldsymbol{x}_k + \upsilon_k \end{aligned} \tag{4.54}$$

where the state $\boldsymbol{x}_k$ is chosen to be the phase space vector and the state transition function $\boldsymbol{F}(.)$ has its first element given by $f(.)$ with the remaining values corresponding to the shifted values of the previous state. Here, the residual error is considered as process noise (Haykin, 2001).

If the time delay for phase space reconstruction is equal to unity, adopting the data driven model (Eq. 4.53) for EKF state-space formulation is straight-forward as shown above. However, in chaotic time series prediction, the optimal time delay may not necessarily be unity. The following procedure is followed in this study. For delay times different from unity the state space model is taken as

$$\boldsymbol{x}_k = \boldsymbol{F}\left(\boldsymbol{x}_{k-1}\right) + \boldsymbol{B}\boldsymbol{w}_k \tag{4.55}$$

$$\begin{bmatrix} x_k \\ x_{k-\tau} \\ : \\ : \\ x_{k-(m-1)\tau} \end{bmatrix} = \begin{bmatrix} f(x_{k-1}, x_{k-1-\tau}, \ldots x_{k-1-(m-1)\tau}) \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ : & : & : & : \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{k-\tau} \\ x_{k-2\tau} \\ : \\ x_{k-m\tau} \end{bmatrix} \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ : \\ : \\ 0 \end{bmatrix} w_k$$

As before, the first element of the state vector is given by $f(.)$ with the remaining values now given by the corresponding lag elements not necessarily from previous state. It should be noted that although the updated previous state $\boldsymbol{x}_{k-1}$ is readily available for use in $f(.)$, the updated values of other lag elements are not. Therefore, a pool of states is maintained in a temporary file where their elements are updated once a state estimate is made. The idea is, when a "corrected estimate" is obtained in the Measurement Update step, the corresponding elements in the pool are updated with the elements of state $\boldsymbol{x}_k$. The corresponding lag elements for the use in Eq. 4.55 are chosen from this updated file.

The temporary states file looks like Eq. 4.56. When the state estimate of $\boldsymbol{x}_k$ is made, the elements of the future states, which have similar elements are replaced by those updated elements of $\boldsymbol{x}_k$. For example, in Eq. 4.56, when updated $\boldsymbol{x}_k$ is available, the last element of state $\boldsymbol{x}_{k+(m-1)\tau}$ can be replaced by the first element of $\boldsymbol{x}_k$. It should also be noted that some of the values of this pool of states are empty. For example, the elements starting from x_{k+1} upwards are unknown. This updating process continues when estimates are made and the earlier estimate is removed from the temporary file and a new column is added. In this process an element is generally replaced *m* times by better estimates.

$$\begin{bmatrix} x_k & x_{k+1} & \cdots & x_{k+(m-1)\tau} \\ x_{k-\tau} & x_{k+1-\tau} & \cdots & x_{k+(m-2)\tau} \\ \vdots & \vdots & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ x_{k-(m-1)\tau} & x_{k+1-(m-1)\tau} & \cdots & x_k \end{bmatrix}_{m\times\left((m-1)\tau+1\right)} \tag{4.56}$$

The next section explores the possibility of Extended Kalman Filtering predictor to improve prediction accuracy of noisy chaotic data with the ANN state-space model trained with noisy data.

4.3.4 Application of EKF with noisy data trained ANN: Lorenz time series

This section tests the EKF on noisy chaotic Lorenz time series prediction with noisy data trained ANN model used in EKF state space model. The Kalman filtering toolbox ReBEL-0.2.6 (the code, in MATLAB, is developed by Rudolph van der Merwe - http://choosh.ece.ogi.edu/rebel/) is used in this study. EKF is tested on four noisy chaotic Lorenz time series of different observation noise levels: 1%, 10%, 20%, and 30%. For the EKF, the model error or residual error (Eq. 4.50) is considered as process noise (Haykin, 2001). EKF has two parameters, the observation noise covariance and the process noise covariance, of which the optimal values have to be determined. An exhaustive search is conducted on a predetermined range of observation noise covariance and process noise covariance to find the optimal values. The time series data are normalized between 0 – 1; and the values considered for observation noise covariance and process noise covariance are 0.1 – 1 in steps of 0.1. For each set of observation noise covariance and process noise covariance, the EKF is run and the prediction is conducted on the test set (Figure 4.5) and the prediction performance on the test set is evaluated. This study uses the prediction error, evaluated comparing the predicted values against the noisy test data, as the criterion for selection of optimal observation noise covariance and process noise covariance. The EKF (e.g. P_k, K_k) tuned using the optimal parameters is then applied to estimate and predict validation data in real-time (Figure 4.6). It should be noted that EKF is continuously updated in this validation stage too.

Justification for the use of prediction error with respect to noisy data as the selection criteria may be offered as follows. Since noise-free signal is not available in real world systems, verifying the performance of noise removal attempts can only be done against noisy data. Even when choosing an optimally trained data driven model, what one tries is to obtain a model that gives lowest prediction error compared against noisy data. Therefore, using prediction error with respect to noisy data to assess noise removal does not do any harm than when using it for model selection. On the other hand, earlier results, discussed in Section 4.2.3, indicated that better prediction performance is reflected in both the error measured with respect to noise free data as well as the error measured against noisy data. This yields feasibility to use prediction error against noisy data as a criterion to identify noise reduction endeavors.

For comparison purposes, the performance ANN models, trained with Lorenz series data of noise levels: 1%, 10%, 20% and 30%, on validation data of noise levels as same as those used to train the models, are given in Table 4.4 (a). The optimal phase space parameters (m, τ) used to train those ANN models are shown in column (1). The prediction error with respect to noisy validation data is given in column (2) while that with respect to noise-free validation data is given in column (3). Results obtained from EKF application are shown in Table 4.4 (b). The percentage prediction improvement over ANN models (measured in MAE) is shown in the last columns for prediction errors measured against both noisy and noise-free data. The percentage improvement is calculated as

$$\%\ improvement = \frac{prediction\ error\ with\ ANN - \ prediction\ error\ with\ EKF}{prediction\ error\ with\ ANN} \times 100 \qquad (4.57)$$

The results show that EKF yields significantly higher prediction performance at all noise levels. Equally good performance on all different noise levels from 1% to 30% shows the robustness of EKF. The predicted values are approximately 20% more

accurate than those predicted with ANN only. The prediction performance measured against noisy data also shows, although not very significant, a considerable improvement. Similar performance is observed in the other three sets of simulations with noises generated from different seed numbers (Appendix H).

The results also show that the use of prediction error with respect to noisy data as the criterion for identifying noise removal has been successful since the solutions, which have shown prediction improvements with respect to noisy data, at the same time, have shown improvement with respect to noise-free data as well. This complies with what was expected from observations in Section 4.2.3.

This section explored the feasibility of using EKF predictor, with data driven ANN model trained with noisy data, in real-time prediction applications of chaotic time series. In conclusion, the EKF has been very successful in improving the prediction performance of chaotic time series with various noise levels. Use of prediction error with respect to noisy data to identify the optimal noise removal has also shown to be a reliable criterion.

EKF is a borrowed technique and it may have its own limitations on chaotic time series. The next section proposes a scheme for real-time noise reduction and prediction that can be applied with any noise reduction method capable of real-time application. The scheme incorporates two pronged approaches: (1) train model with noise reduced data; and (2) provide noise reduced input data to the model trained with noise reduced data.

4.4 SCHEME FOR REAL-TIME NOISE REDUCTION AND PREDICTION

As discussed in Chapter 2 and Section 4.2.1, the procedures proposed for noise reduction in chaotic hydrological time series literature are not applicable in real-time applications. Prediction is essentially a real-time application and should noise

reduction be applied to improve prediction, it should be incorporated in real-time. Following the deduction, made in section 4.2.3, that a model trained with low noise level should improve the prediction accuracy if it is supported with input data of equally low noise or even lower, this section investigates the possibility of using noise reduced inputs and less-noisy-data trained models to further improve the prediction accuracy. This study proposes a procedure coupling noise reduction and prediction to address real-time applications.

The procedure couples a noise reduction method with a prediction model as shown in Figure 4.7. First, some historical data (training and test data sets) are fed into a noise reduction method to yield the noise reduced data sets. The noise-reduced training data set is then used to train a prediction model. An optimal model is chosen depending on the performance on the test set. The procedure thus far is an off-line process where available historical data is applied. The next step, the validation, is operated in real-time. Whenever a new data point y_k comes in, the noise-reduced value of it is obtained with the noise reduction method and the corresponding input vector is then prepared. Inserting this input vector in the optimal prediction model previously obtained, the prediction $\hat{x}_{k+T}$ for the future value x_{k+T} is obtained. It should be noted that unlike in EKF where prediction is inevitably confined to 1-step predictions, in the proposed procedure any prediction horizon (T) may be employed by training the prediction model to the desired prediction step.

Often it is required to tune parameters in noise reduction schemes as well. In EKF, for example, to be determined are the process noise covariance and the observation noise covariance. The off-line process of deriving a prediction model with less-noisy data set may also include tuning the filter/ noise reduction parameters as well. This study uses an exhaustive search approach to select the optimal parameters

from a pre-determined set of feasible filter parameters. The prediction performance resulting from the test set is used as the criteria for selection of both the optimal prediction model and the optimal parameters. These parameters, prediction model and the tuned noise reduction method (e.g. EKF), where applicable, is transferred to the real-time step of the procedure. The tuned noise reduction method is then used to reduce noise in input data (validation data) and the prediction model is used to predict future values with those noise-reduced inputs. The detailed procedure is illustrated in Figure 4.8. The next section demonstrates the EKF on the proposed procedure.

4.5 THE PROPOSED SCHEME WITH EKF NOISE-REDUCED DATA: LORENZ SERIES

This section demonstrates the effectiveness of the proposed scheme for real-time noise reduction and prediction using EKF as a noise reduction method. Another objective is to verify if this procedure can yield higher prediction accuracy over the EKF predictor.

It was shown (section 4.3.2) that the KF estimates can be treated as noise reduced data in chaotic time series analysis. Kalman Filter Smoothing, which includes both forward and backward filtering, is the generally used approach for noise reduction as it can provide better estimates than forward / backward filtering alone. In KF smoothing the smoothed estimate is controlled from both past and future records. The two end data points, i.e. the first and the last records are determined only by backward filtering and forward filtering respectively. The variation of mean square estimation error of forward filtering, backward filtering and smoothing are shown in Figure 4.9 (Gelb, 1974). This shows that the estimates are more accurate in the middle part and not that accurate at both ends. However, the last estimate is important as it is used as the input to make the future prediction. Section 4.2.3 showed that models trained with

less-noisy data may not necessarily produce good predictions if the input data are noisy. Therefore, this study chose to use forward filtering for noise reduction since its last estimate is also of the same level of noise of the rest, which is used to train data driven model.

An exhaustive search is conducted over the observation noise covariance and process noise covariance values (considered values are same as those for EKF predictor in section 4.3.4) to choose the optimal filter and the optimal trained ANN model (see the off-line part of Figure 4.8). Prediction error, with respect to noisy data, on the test set is taken as the calibration criterion in the selection of optimal model. The optimal filter and the optimal model are then used to predict the validation data. It should be noted, however, that the state-space model of EKF is not replaced by the optimal prediction model. This phase is similar to that of EKF predictor except the model trained with noise-reduced data is now used for prediction. The prediction performance for the data sets of 1%, 10%, 20% and 30% noise levels is shown in Table 4.5. (The statistics of noise reduced data; and the plots of noise-free data, noisy data and noise reduced data; the attractor in these cases; and the plots of actual and predicted values are shown in Appendix I). Columns 3 and 4 of Table 4.5 show that EKF with the proposed procedure gives significant prediction improvement (as high as 25% – 30% in prediction error measured against noise-free data) over ANN models. Similar to the EKF predictor, prediction error with respect to noisy data also shows some improvement although not as remarkable as prediction error improvement with respect to noise-free data. Comparison of these prediction performances with columns 3 and 4 of Table 4.4 (b) (where the prediction performance of EKF predictor is given) shows that the proposed procedure yields higher prediction accuracy than the EKF predictor. To ensure that the results are not biased by the prediction tool, the ANN

models used within EKF and in the test and validation processes, the experiments were repeated with Support Vector Machines (SVM) as the prediction tool in the test and the validation processes. The results showed similar trends as obtained from ANN (Appendix J).

Results show that the use of EKF estimates as noise reduced data on the proposed noise reduction scheme significantly improves the prediction accuracy. The robustness of EKF on the various noise levels is also noteworthy.

The applicability of simple nonlinear noise reduction method on the proposed procedure will be explored in the following section.

4.6 THE PROPOSED SCHEME WITH SIMPLE NONLINEAR NOISE REDUCTION: LORENZ SERIES

This section investigates the possibility of adopting a popular noise reduction technique in nonlinear chaotic dynamic literature for real-time noise reduction. Kantz and Schreiber (2004) found that the simple nonlinear noise reduction algorithm to be reliable and effective on a broad variety of data sets including artificial and real data. However, as other nonlinear noise reduction methods, it is basically meant for off-line noise reduction applications. This section explores the possibility of incorporating it for real-time application and it is tested on the proposed scheme. Its prediction performance is then compared to that of Extended Kalman filter introduced in the last section.

In this section simple nonlinear noise reduction method will first be explained. This is followed by its application on real-time noise reduction, together with the proposed scheme, on Lorenz time series. Finally, the performance is compared with that of EKF.

4.6.1 Simple nonlinear noise reduction method

For an observed time series, y_n, $n = 1, \ldots N$, where the dynamical rule and the measurements are expressed as Eq. 2.15 and Eq. 2.16, simple nonlinear noise reduction solves the following implicit equation

$$x_n - f\left(x_{n-m}, \ldots, x_{n-1}\right) = 0 \tag{4.58}$$

for one of the coordinates. The function f is unknown; even if it is known it is generally impossible to solve it for one of its arguments. Therefore, a locally constant function is used to approximate f. This is the basic principle of the method proposed by Schreiber (1993), which is the simplest and most widely used nonlinear noise reduction method. The main idea of the method is to replace each measurement y_i by the average value of this coordinate in a suitably chosen neighborhood. The neighborhoods are defined in a phase space reconstructed by *k* past coordinates and *l* future coordinates given by

$$\boldsymbol{y}_i = \left(y_{i-k}, \ldots, y_{i+l}\right) \tag{4.59}$$

In a neighborhood of ε if the set of all neighbors $\boldsymbol{y}_j$ satisfying $\left\|\boldsymbol{y}_i - \boldsymbol{y}_j\right\| < \varepsilon$ is $\Im_i^{\varepsilon}$, the 'present' coordinate y_i is replaced by y_i^{clean} given by

$$y_i^{clean} = \frac{1}{\left\|\Im_i^{\varepsilon}\right\|} \sum_{\Im_i^{\varepsilon}} y_j \tag{4.60}$$

Only the central coordinate in the delay window is corrected since only this coordinate is optimally controlled by the past and future. In the applications *k* is generally taken to be equal to *l* or else when the total number of delay elements, m (Eq. 2.1) is even, the correction is made to $y_{n-m/2}$. In practice, these corrections are performed for several iterations. That is, once a cleaned series is obtained, the replacement in Eq. 4.59 is performed on the cleaned series and the procedure is repeated. The errors induced by

these replacements are of both statistical and geometrical in nature. If the points in $\Im_i^\varepsilon$ are regarded as a random sample distributed according to the natural measure, the statistical uncertainty of the center of mass is damped out like $\left|\Im_i^\varepsilon\right|^{-\frac{1}{2}}$ whereas the error introduced by replacing the geometrical center of the neighbourhood by the center of mass depends on the non-uniformity of the distribution within $\Im_i^\varepsilon$ and will generally grow with the size of the neighborhood. The method is expected to work when these errors are smaller than the individual errors of the coordinates.

4.6.2 Application of simple nonlinear noise reduction on proposed scheme

The simple nonlinear noise reduction method uses both past and future coordinates to estimate the noise-reduced value of a certain point, this is the optimal approach for off-line noise reduction applications. Since no correction is made to the end values, a better estimate of the current observation is not possible, i.e. it is impossible to provide noise reduced inputs to the prediction model. The earlier investigations of this study showed that it is not only the model but also the input values should be of lower noise levels in order to produce better predictions. It is, therefore, important to seek an implementation, which corrects the end values as well.

Hegger et al. (1999) implemented the simple nonlinear noise reduction method in TISEAN software package under program named *lazy*. They also implemented the same nonlinear noise reduction algorithm but with correction made to all the coordinates in a program called *nrlazy*, where the end values also get an opportunity to be noise-reduced. Since every single time series element is an element of *m* different phase space vectors, this gives *m* typically conflicting corrections. *nrlazy* takes the arithmetic mean of all these; the set of corrected phase space vectors is converted back into a corrected time series. It should be noted, however, the end values are corrected only once since they appear only in one phase space vector. *nrlazy* is found to be

superior to *lazy* for flow like data (see http://www.mpipks-dresden.mpg.de/~tisean/TISEAN_2.1/docs/docs_c/nrlazy.html). Since this algorithm performs a correction to end values as well, this algorithm is incorporated in this study.

Being a smoothing technique, *nrlazy* reduces more noise in the central points of a time series than it does on the end values (similar to EKF smoothing explained in section 4.5), which are used as validation inputs. Using such smoothed data in the training and test sets in calibration of the noise reduction parameters and a prediction model (see off-line part of Figure 4.8) may lead to 'too good' models that do not necessarily perform well with more noisier validation input data. Therefore, in this study the nonlinear noise reduction is performed as follows. To reduce noise of a certain point y_i, only the points up to time *i* (i. e. from 1, 2, …, *i*) are used. This means that no future coordinates are used. A correction is thus made to any observation only once instead of *m* times as explained above. In this way, the amount of noise reduction in validation inputs is approximately same as the rest, which are used to train a prediction model. This is a non-optimal noise reduction approach in the case of nonlinear noise reduction techniques. However, for real-time applications this is inevitable.

The simple nonlinear noise reduction requires two parameters to be specified: (1) the neighbourhood size ε, and (2) the number of iterations. Kantz and Schreiber (2004) found that a good choice for the size of neighbourhoods is about 2 – 3 times the noise standard deviation (σ). Only a few numbers of iterations are recommended since the signal may be distorted otherwise (e.g. Schreiber, 1993; Mees and Judd, 1993). Similar to the EKF application, this study conducts an exhaustive search to determine the optimal values for ε and the number of iterations. The ranges

considered are $0.1*\sigma$ to $4*\sigma$ in steps of $0.1*\sigma$ for neighborhood size ε: the number of iterations is 1 – 3.

The results are shown in Table 4.6. The nonlinear noise reduction does not yield any prediction improvement at very low noise levels (e.g. 1% noisy data). However, at high noise levels, some prediction improvement is observed. The nonlinear noise reduction technique is known to be more effective on high noise levels. This explains the poorer performance on low noise levels. Although no improvement is evident in the 20% noisy data set, in which the results are shown in Table 4.6, the noisy data sets generated with different seeds (Appendix H) shows that some prediction improvement is observable at high noise levels. Comparison with the columns 3 and 4 of Tables 4.4 (b) and 4.5 shows, however, that the prediction improvement with nonlinear noise reduction technique is very much poorer than that with EKF predictor and EKF estimates as noise reduced data in the proposed scheme. The nonlinear noise reduction, primarily meant for off-line noise reduction applications, has not been much successful in the real-time applications. Therefore, it is beneficial to look for techniques designed for real-time noise reduction/filtering applications.

4.7 APPLICATION OF EKF AND THE NOISE-REDUCTION SCHEME ON RIVER FLOW TIME SERIES

All the techniques discussed above: (1) EKF, (2) EKF together with the proposed scheme and (3) nonlinear noise reduction together with the proposed scheme are now applied on river flow time series. For the EKF, the same ranges of observation noise covariance and process covariance values considered earlier in the Lorenz series are considered initially with time series values normalized into an interval between 0 – 1. The results indicated that the optimal observation noise covariances are very low.

Therefore, the tests were repeated with observation covariance varying from 0.001 to 0.1 in steps of 0.005: this is in addition to the initial range. For the nonlinear noise reduction, the same ranges used in Lorenz series are considered with the one-step prediction error of ANN on test set (in terms on MAE) used to approximate σ for ε-range.

Results are shown in Table 4.7. Since the true signal is unknown, only prediction errors with respect to noisy data can be computed. Results show that no method has given any significant prediction improvement on river flow time series. This can be due to the followings reasons. First, the EKF assumed the noise to be white and Gaussian distributed. This may not be true in the real river flow time series and it is also possible that the noise is correlated. These may lead to unsatisfactory performance. It was noted that, in the case of EKF, the optimal observation noise covariance is much smaller than the process noise covariance. This is perhaps due to the dynamical noise being more prominent than the observation noise. If that is the case, the dynamic noise can largely contribute to the prediction error and the removal of less prominent observation noise may not be reflected in the error measures.

4.8 SUMMARY AND DISCUSSION OF RESULTS

The models trained with less noisy data did not provide higher prediction accuracy than models trained with more noisy data when the input data were also equally noisy. Noise-reduced data inputs, however, help the noisy data trained model to yield higher prediction accuracy.

The EKF from controls literature was adopted for prediction and noise reduction of noisy chaotic time series. A data driven model (ANN) trained with noisy data is incorporated in EKF. Results showed the effectiveness of EKF to improve prediction accuracy in noisy chaotic time series. The EKF resulted in significant

prediction improvement (as high as 15% – 25%) over ANN models in Lorenz time series prediction. The robustness of EKF, on series with noise levels ranging from mild to very high, is also commendable. It should be noted that the robustness of the algorithm is the most important factor for noise reduction applications where true signal is unknown (Grassberger et al., 1993).

A noise reduction procedure for real-time prediction applications was proposed and it was shown to be effective. The EKF state estimates were incorporated as noise reduced data and was applied on the proposed procedure. This gave even better prediction improvement (as high as 25% – 35% over ANN alone with noisy Lorenz series) compared to EKF predictor. Results imply that in the Kalman Filtering applications, which use data driven models, higher prediction accuracy can be obtained by modifying the model with noise-reduced estimates rather than applying the KF alone. This implies that dual KF approach may also improve the prediction performance over EKF in chaotic time series prediction. An advantage of the proposed scheme over dual KF approach is that the proposed procedure can be readily used for lead times different from 1 by simply training prediction model of the desired lead-time. This is in contrast to both Kalman Filtering and dual Kalman filtering approaches which are limited to 1-step prediction. However, since various prediction horizons are desirable in practical applications, the proposed scheme is more advantageous over KF and dual KF approaches. The scheme is expected to perform well with any noise reduction technique capable of real-time application.

The Kalman filtering is shown to be much more superior to simple nonlinear noise reduction when it is incorporated with real-time prediction applications. The poor performance of nonlinear noise reduction is due to the fact that the nonlinear noise reduction techniques are designed primarily for off-line noise reduction

applications. This implies that it is more advantageous for the dynamical systems community to consider methods developed in controls literature when dealing with real-time applications, such as prediction, of noisy data. Similar recommendations were made for noise reduction by Walker and Mees (1997).

The applications of any noise reduction technique on river flow time series, however, did not show any prediction improvement. This could be due to at least two reasons:

(1) the precise nature of the noise present (e.g. white/ coloured; distribution; level of noise) in the real time series are unknown. The EKF assumed the noise to be white and Gaussian distributed. However, the noise in real world data may not be so;

(2) the observation noise levels in both Mississippi and Wabash river flow time series may be very low compared to dynamical noise; when the dynamical noise is the prominent contributing factor in the prediction error, the removal of part of measurement noise may not be reflected in error measures.

4.9 CONCLUSION

This study identified several means to improve the prediction accuracy of noisy chaotic time series. It was shown that noise reduced inputs can enhance prediction accuracy. To the contrary of the general anticipation that the use of noise reduced data to train prediction model may help in improving prediction, the findings of this study show that the prediction accuracy may not necessarily be enhanced with noise-reduced data trained models if it is not supported with noise-reduced input data as well. Due to the above reasons, the study identified the necessity for real-time application of noise reduction.

It was shown that the Kalman filtering technique, specifically the Extended Kaman filter, together with a data driven model trained with noisy data as a state-space

model, can be used as a reliable and robust technique for real-time noise reduction in chaotic time series. It improved the prediction performance of chaotic time series over the ANN model alone. It was also shown that incorporating the popular nonlinear noise reduction techniques for real-time applications is very unsatisfactory and there is a need to identify better techniques capable of real-time application.

The study proposed a scheme, which incorporates noise reduction to improve prediction of chaotic time series. This scheme has circumvented the short-comings of the earlier approaches. The scheme couples the use of noise-reduced data inputs and noise-reduced data trained models to arrive at higher prediction accuracy. The effectiveness of the proposed scheme was demonstrated with EKF. The proposed scheme produced predictions considerably better than when EKF was applied alone.

More studies should be conducted to identify the levels and the effects of measurement and dynamic noises in real world data on prediction. Identifying the characteristics of measurement noise in real world data and then applying the appropriate noise reduction methods accordingly can, hopefully, improve the prediction performance.

Although the prediction performance is significantly improved with the proposed approach, the required computational time is also very high. This is due to the use of time consuming ANN prediction model in the calibration of parameters. The next chapter investigates the possibility of extracting a smaller set of system representative data from a large data record to make the analysis efficient in time consuming applications.

Table 4.1 Prediction performances of ANN models, trained with noise-free and noisy data sets, with noisy validation input data sets

Prediction error (MAE)	validation input data set: 1% noise level		validation input data set: 30% noise level	
	ANN trained with data of 1% noise level	ANN trained with noise-free data	ANN trained with data of 30% noise level	ANN trained with noise-free data
Against noisy data	0.1279 (A)	0.1328 (C)	4.0831	4.4598
Against noise-free data	0.0725 (B)	0.0834 (D)	2.2602	2.9709

Table 4.2 Prediction performance of ANN model trained with 30% noisy data when noise-free, 1%, 10%, 20% and 30% noisy validation data are used as inputs

Prediction error (MAE)	Validation data of different noise levels				
	Noise-free (1)	1% noise (2)	10% noise (3)	20% noise (4)	30% noise (5)
Against 30% noisy data	3.4794	3.4791	3.5223	3.7759	4.0831
Against noise-free data	0.7356	0.7390	0.9732	1.6495	2.2602

Table 4.3 Summary of findings on means of improving prediction performance

Row	Level of noise of the data used		Prediction error
	For model training	As validation inputs	
(1)	x^*	x	E
(2)	x	y^{**}	$< E$
(3)	y	x	Not necessarily less than E
(4)	y	y	$< E$

* x is the noise level of a time series before noise reduction
** y is the noise level of a time series after noise reduction

Table 4.4 (a) Prediction performance of ANN models trained with noisy data with equally noisy validation inputs: Lorenz time series

Noise Level (%)	Optimal Phase space parameters (m, τ) (1)	prediction error (*MAE*)	
		Against noisy data (2)	Against noise-free data (3)
1	(10, 3)	0.1279	0.0725
10	(10, 3)	1.2015	0.6906
20	(9, 1)	2.7378	1.5433
30	(9, 3)	4.0831	2.2602

Table 4.4 (b) Prediction performance of EKF predictor on Noise-induced chaotic Lorenz time series

Noise Level (%)	prediction error (*MAE*)		Prediction improvement over ANN alone (%)	
	Against noisy data (1)	Against noise-free data (2)	Against noisy data (3)	Against noise-free data (4)
1	0.1196	0.0525	7	28
10	1.1544	0.5421	4	22
20	2.6475	1.3077	3	15
30	3.9173	1.8527	4	18

Table 4.5 Prediction performance of EKF estimates on the proposed scheme: noise-induced chaotic Lorenz time series with ANN

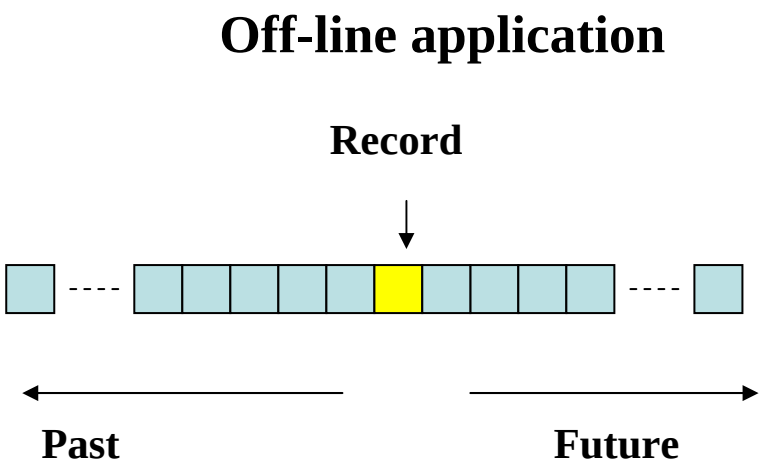
Noise Level (%)	prediction error (*MAE*)		Prediction improvement over ANN alone (%)	
	Against noisy data (1)	Against noise-free data (2)	Against noisy data (3)	Against noise-free data (4)
1	0.1177	0.0475	8	35
10	1.1105	0.4672	8	32
20	2.5901	1.2464	5	19
30	3.7888	1.6005	7	29

Table 4.6 Prediction performance of nonlinear noise reduction on the proposed scheme: noise-induced chaotic Lorenz time series with ANN

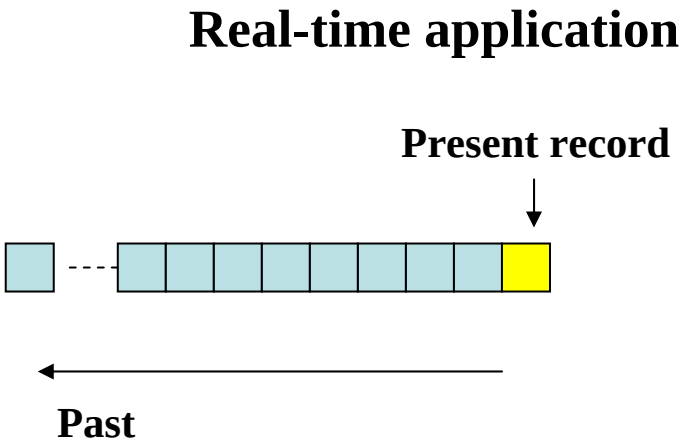
Noise Level (%)	prediction error (*MAE*)		Prediction improvement over ANN alone (%)	
	Against noisy data (1)	Against noise-free data (2)	Against noisy data (3)	Against noise-free data (4)
1	0.1279	0.0725	0	0
10	1.1768	0.6585	2	5
20	2.7893	1.5408	-2	0
30	4.0313	2.0430	1	10

Table 4.7 Prediction performance of ANN/ EKF predictor/ EKF estimates and Nonlinear noise reduction on the proposed scheme: River flow time series

Time series	Parameter	ANN	EKF	Proposed Scheme	
				EKF Estimates	Nonlinear Noise Red
Mississippi River flow	Prediction error against noisy data (*MAE*) (m^3/s)	205.15	205.55	204.11	205.01
	Prediction improvement over ANN alone	N/A	-0.2 %	0.5 %	0%
Wabash River flow	Prediction error against noisy data (*MAE*) (m^3/s)	25.76	25.98	25.75	25.77
	Prediction improvement over ANN alone	N/A	-0.8 %	0.1 %	0%

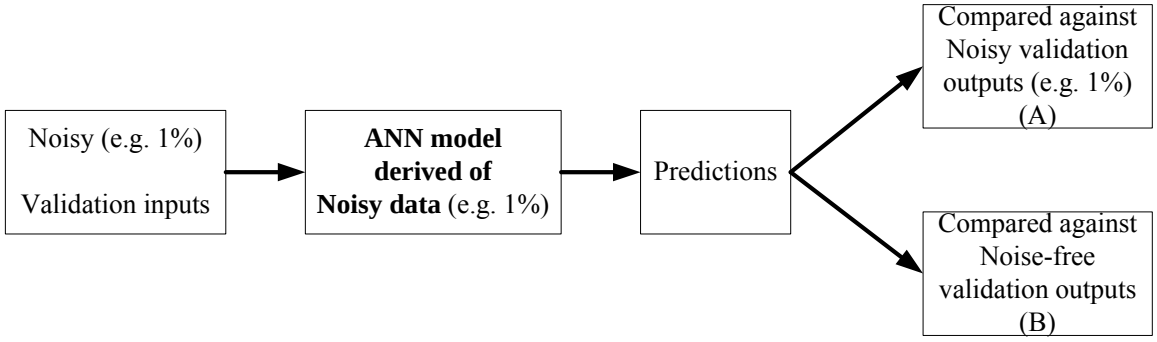


(a) Off-line

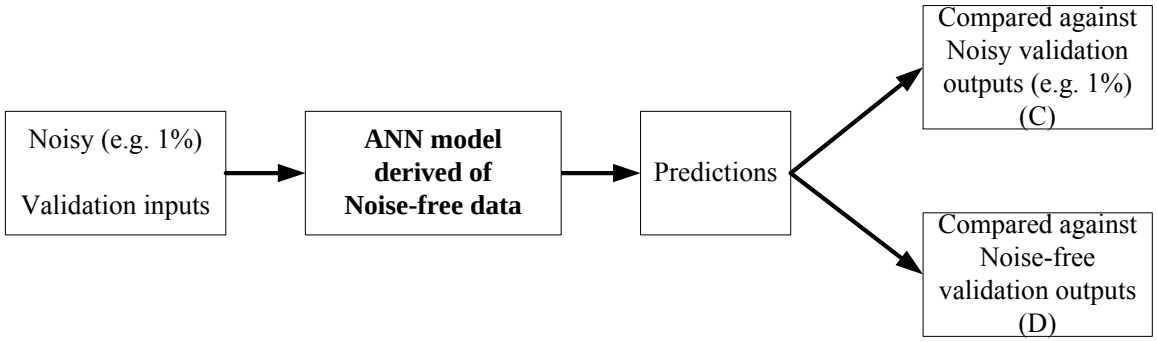


(b) Real-time

Figure 4.1 Off-line and Real-time application of noise reduction



(a) Model trained with noisy data



(b) Model trained with noise-free data

Figure 4.2 Performance evaluation of models derived of noisy and noise-free data

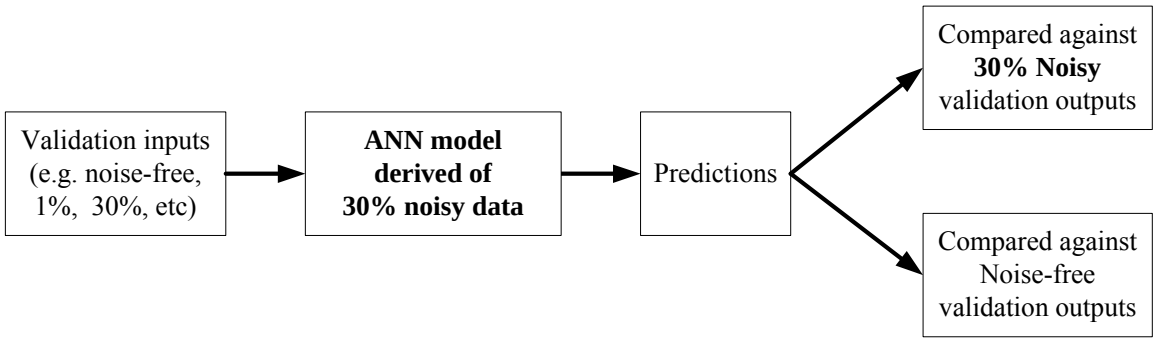


Figure 4.3 Performance evaluation of model derived of 30% noisy data with inputs of different quality

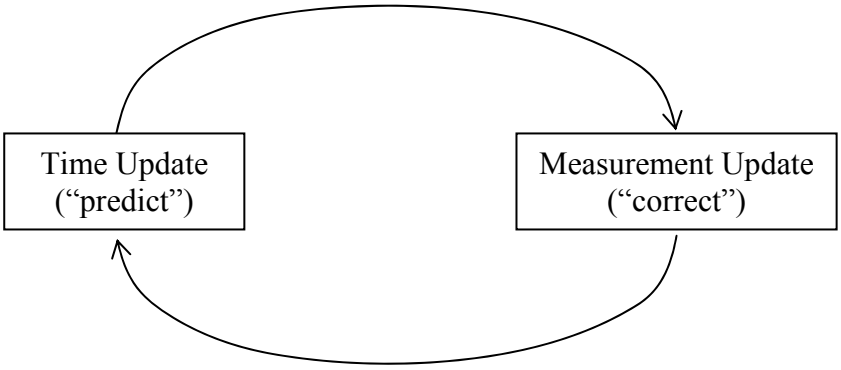


Figure 4.4 Discrete Kalman filter cycle

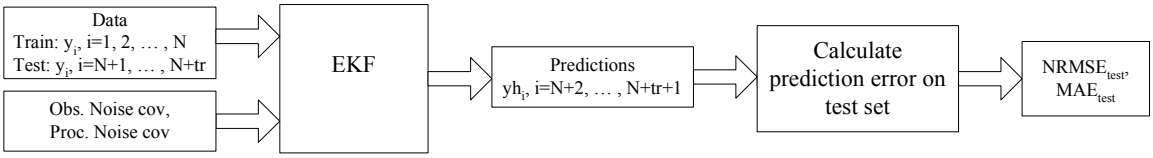


Figure 4.5 Tuning observation and process noise covariance in EKF

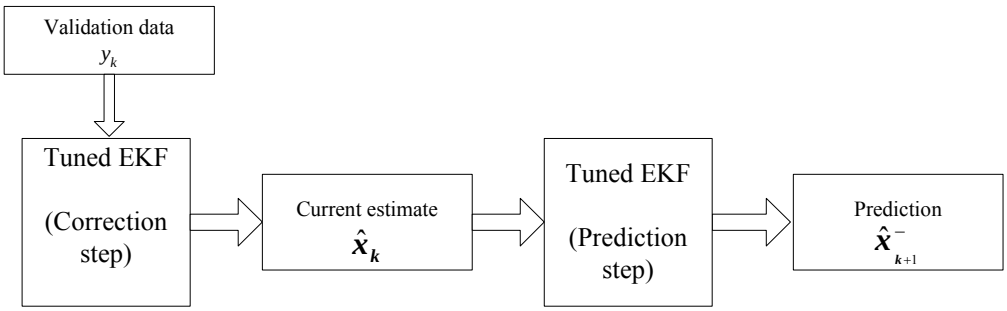


Figure 4.6 Prediction of validation data with EKF

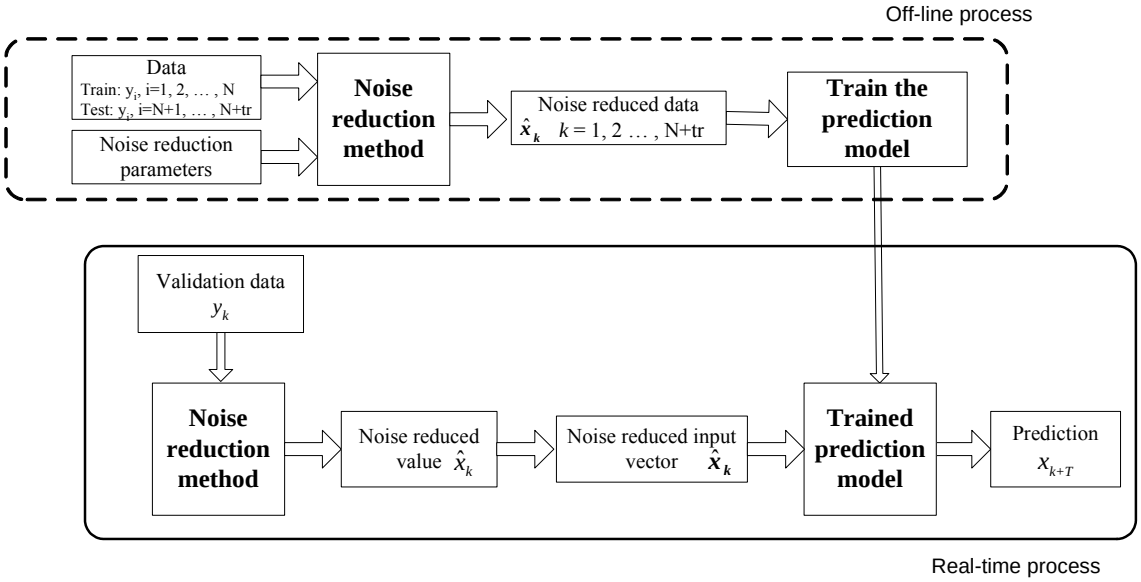


Figure 4.7 Proposed scheme for real-time noise reduction and prediction

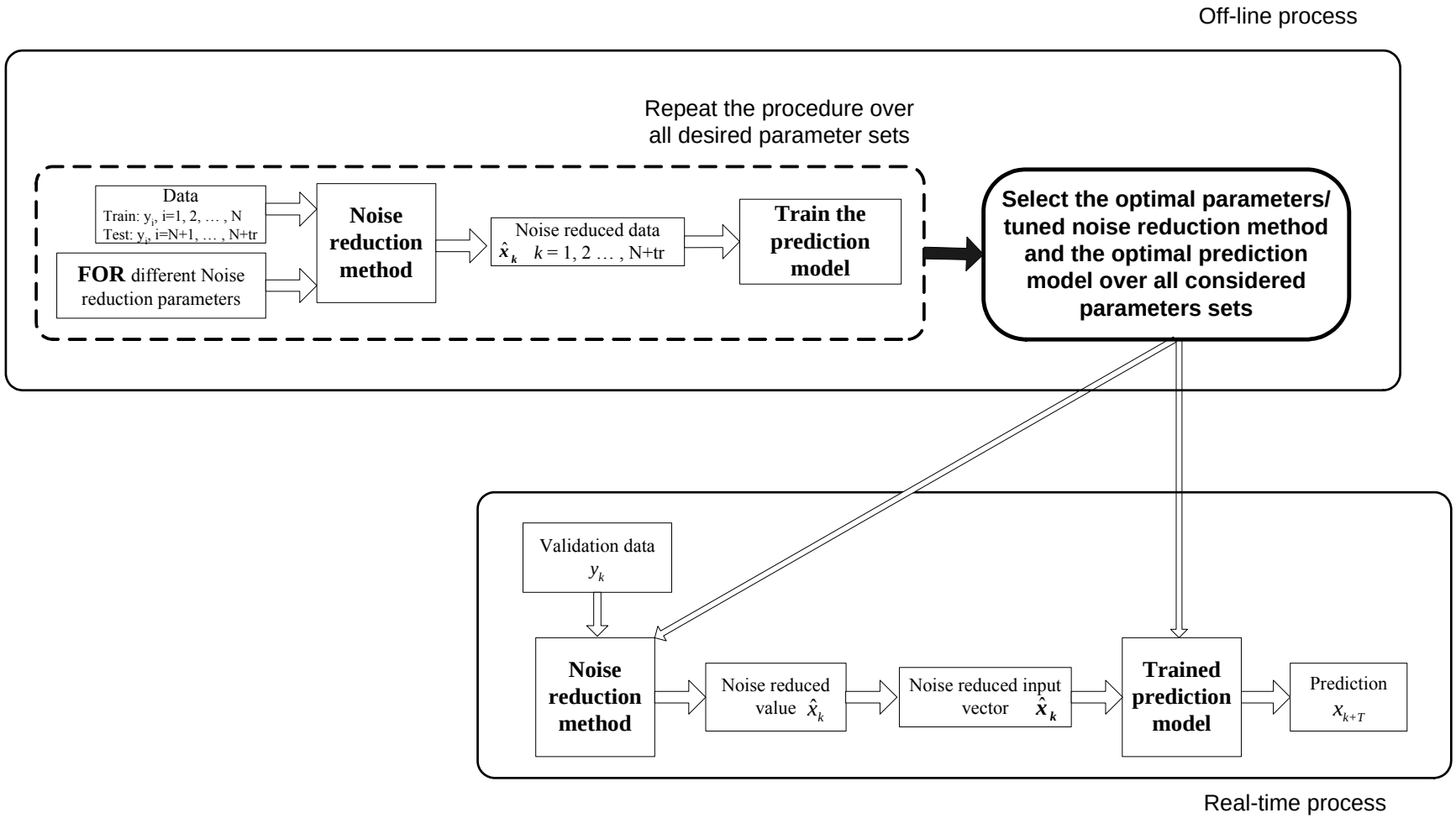


Figure 4.8 Proposed scheme for real-time noise reduction and prediction (in detail)

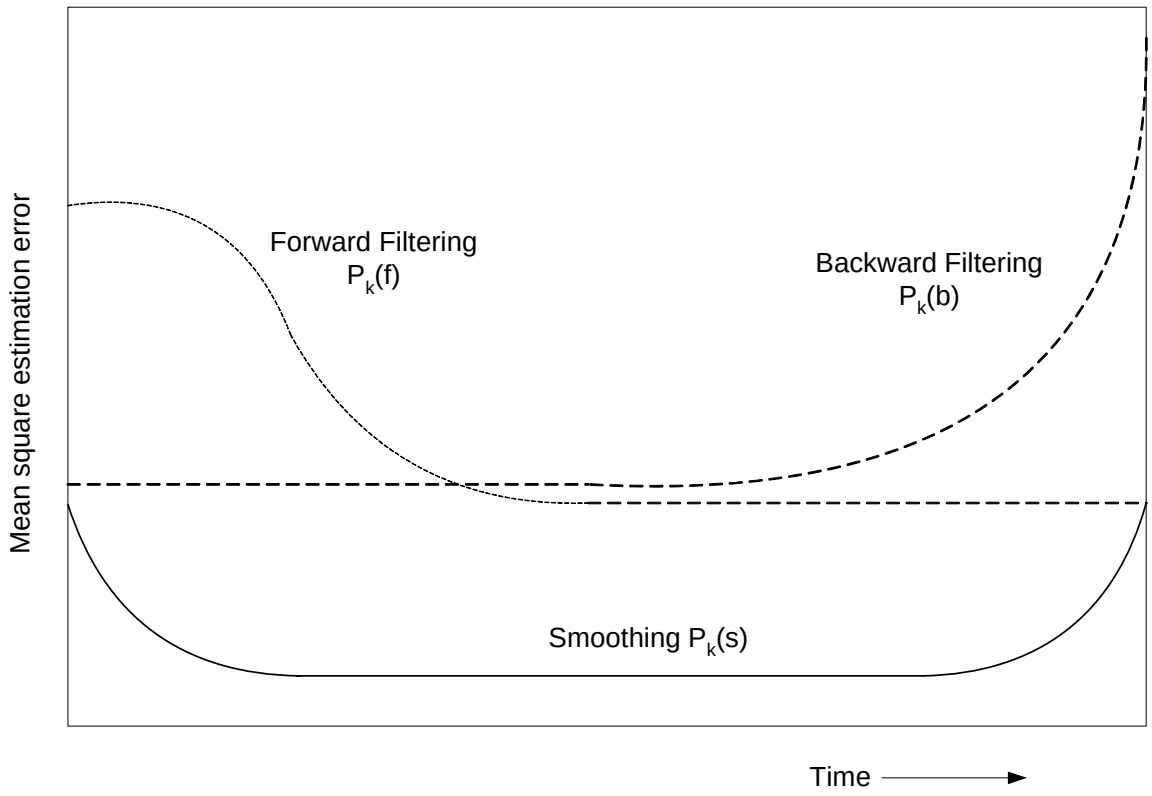


Figure 4.9 Mean square estimation errors of Forward filtering/ Backward filtering and Smoothing

CHAPTER 5

DERIVING AN EFFECTIVE AND EFFICIENT DATA SET FOR PHASE SPACE PREDICTION

5.1 INTRODUCTION

The last two chapters explored how the prediction accuracy of chaotic time series can be improved. It was noticed that the computational burden (time and storage capacity) can be overwhelming with chaos analysis due to the large data record size required especially in prediction applications. The large data sets require lots of computational resources such as memory and time. For example, the time complexity of ANN and SVM are of the order of N^2 where N is the number of training patterns. With new records continuously coming, the data sets get increasing larger. Therefore, a methodology to extract most representative data from large data records is highly desirable.

In chaotic time series analysis, a long past data record is used in both system characterization and prediction. The phase space prediction models generally assume that the larger the number of past records the better the predictions would yield. It is, however, questionable as to whether all such data contribute valuable information for phase space prediction. Redundancy of data can occur due to two reasons: (1) it is possible that not all the points are necessary to represent a certain relationship (e.g. two points suffice to represent a linear relationship of a single input/ single output system), (2) also, there can be points that are repeated and/ or that are closer than noise level, which do not contain any distinct information. This chapter explores the possibility of extracting a system representative data set from a large raw data set for phase space prediction by hopefully filtering out the redundant data. Clustering, a process of

grouping the data into classes or clusters (Figure 2.2), widely used in data mining, statistics, biology, machine learning etc., is applied in this study to derive a compact effective data set from a long original data set.

Since the objective is to extract a compact set of data representative of a system, it is better achieved when performed on the data reconstructed in the organized space, the phase space. This study applies clustering on the reconstructed phase space. Thus, the aim is to select a system representative set of phase space vectors out of all phase space vectors. Most clustering techniques produce artificial points as cluster centers. For example, the K-means clustering selects centroids of sets of data points as cluster centers. For chaotic data, introducing such artificial points is not recommended since they can alter the true dynamics. Subtractive clustering method (SCM - Chiu, 1994) is one technique that selects a subset of original data as cluster centers. Therefore, this method is employed in this study.

In this chapter, the possibility of extracting a compact set of data is first investigated with SCM. Then a new simple clustering technique is proposed to overcome some of the difficulties faced with SCM. All the techniques will be demonstrated on noise-free and noisy (5% and 30% noise levels) Lorenz time series and river flow time series. Finally, the application of clustering to improve the time taken in Extended Kalman Filtering noise reduction application is demonstrated.

5.2 DATA EXTRACTION WITH SUBTRACTIVE CLUSTERING METHOD

5.2.1 Subtractive clustering method

The Subtractive clustering method (Chiu, 1994) works as follows. Let $\boldsymbol{x}_1, \boldsymbol{x}_2, ...\boldsymbol{x}_n$ be n data points from m dimensional space. Then these data points are

normalized in each dimension so that they are bounded by a unit hypercube. If the normalized points are $\boldsymbol{y}_1, \boldsymbol{y}_2, ...\boldsymbol{y}_n$, the potential, P_i, of a data point y_i is defined as,

$$P_i = \sum_{j=1}^{n} e^{-\alpha \|y_i - y_j\|^2} \tag{5.1}$$

where $\alpha = \frac{4}{r_a^2}$

and r_a, called influence range, is a positive constant defining a neighbourhood. The data points outside this radius have little influence on the potential of the point.

Once the potential of each data point is computed, determine the point $\boldsymbol{y}_1$ with the highest potential P_1^*. Select this point as a cluster center and then set its potential to zero. Then the following procedure is followed for selecting the other cluster centers using subtractive clustering method.

1. Select the data point ($\boldsymbol{y}_k$) with the highest potential (P_k^*) as a candidate for a cluster center.
2. Accept or reject the data point as a cluster center depending on the selection criteria listed in Table 5.1.
3. If the data point is accepted as a cluster center, revise the potential of each data point using the formula,

   $$P_i = P_i - P_k^* e^{-\beta \|y_i - y_k\|^2} \tag{5.2}$$

   where $\beta = \frac{4}{r_b^2}$

   r_b is a positive constant defining the neighbourhood that will have measurable reductions in potential. (from here onwards the ratio r_b/r_a is called as Squash Factor (SF))

4. Repeat steps 1 to 3 until $\frac{P_k^*}{P_1^*} < RR$ where *RR* is called the reject ratio.

AR is the accept ratio, a parameter which determines whether a data point should be accepted as a cluster center and *RR* is the reject ratio, a parameter which determines whether a point should be rejected as a cluster center. The values of AR and RR may vary between zero and one. Altogether, SCM has four parameters (influence range (r_a), squash factor (SF= r_a/r_b), accept ratio (AR) and reject ratio (RR)) governing the determination of cluster centers.

5.2.2 Procedure for data extraction

The first step in the procedure is to reconstruct the phase space with appropriate phase space parameters. Then SCM is applied on the reconstructed phase space assuming the normalized phase space vectors as patterns $\boldsymbol{y}_i$ of SCM. Once a set of cluster centers corresponding to a set of SCM parameters is derived, the corresponding output values are also selected (see Figure 5.1). These cluster centers and outputs can serve as a smaller set of input/output patterns to train a prediction model. Then the prediction model can be trained with the smaller data set instead of the entire training data set. The model can then be used for prediction. The problem lies in the selection of SCM parameters which give smaller number of patterns and yet sufficient to represent the entire training data set. Subtractive clustering method (SCM) has four parameters to be optimized. Therefore, the selection of optimal values for the parameters needs a cost effective optimization technique. Micro Genetic Algorithm (that was used in finding optimal parameters in SVM in section 3.4.4) is chosen to optimize the SCM parameters. Thus, the procedure couples a clustering method and a prediction tool with mGA to extract system representative data from long data records.

The data extraction procedure has two phases: (1) calibration of SCM parameters; and (2) validation of the optimal solutions. In the calibration stage mGA is coupled with a local prediction technique and SCM to determine the optimal SCM parameters. This study uses the prediction error on test set as the criterion to determine the effectiveness of the selected data sets and hence the optimal parameters. In the validation stage, the optimal SCM parameters are used to derive representative data sets and their prediction performance on unseen data sets is measured with local averaging and global ANN prediction models.

Calibration: First, the phase space is reconstructed with the optimal phase space parameters (m, τ) of ANN models obtained using the exhaustive search in Chapter 3. Then SCM parameters generated with Micro-Genetic algorithm (mGA) are used with SCM to extract a reduced set of phase space vectors from the phase space reconstructed from the training set. The reduced phase space vectors are then used for prediction of the test set using the local averaging prediction method. Local averaging technique is employed in this application to facilitate the evaluation of data extraction procedure with respect to both local and global prediction techniques. For the local model, the optimum number of nearest neighbors (k), corresponding to the optimal (m, τ) of ANN, is chosen from the exhaustive search (Chapter 3). It should be noted that in prediction with the reduced number of patterns, it is necessary that a modified value of nearest neighbors, k', is chosen. k' is defined as

$$k' = \frac{k}{\textit{Total number of patterns used to determine } k} \times \textit{Reduced number of patterns} \qquad (5.3)$$

The procedure is repeated until a predetermined stopping criterion is reached. The optimal SCM parameters are selected considering the prediction error on the test set. The calibration procedure is illustrated in the schematic diagram given in Figure. 5.2.

Validation: Once the optimal SCM parameters are found, the reduced number of phase space vectors obtained from those parameters is used to train an ANN prediction model and to predict the validation data. The validation procedure is shown in Figure 5.3.

The ranges of SCM parameters used are: (1) $0.001 \leq r_a \leq 0.5$; (2) $1.0 \leq SF \leq 2.0$; (3) $0 \leq AR \leq 1.0$; and (4) $0 \leq RR \leq 0.5$. The mGA parameters and algorithm is the same as the one used in Chapter 3 with SVM. Prediction error on test set is used as the fitness criteria.

5.2.3 Results

The data extraction procedure is applied to the noise-free Lorenz series, two Lorenz series one contaminated with a moderate noise level of 5% while the other with a very high noise level of 30%, Mississippi river flow time series and Wabash river flow time series. Analysis is performed for lead-time 1. The training, test and validation sets are same as defined in Chapter 3. The mGA solutions, which give prediction errors on the test set less than 120% of the prediction error resulting from the use of the entire training data sets, are selected as optimal solutions. The selection of solutions with errors up to 120% is for the purpose of examining the reduction of prediction accuracy with the reduction of data set. The ANN models trained with extracted training data sets corresponding to the optimal solutions are used to predict the validation sets. The reason for selecting a set of solutions instead of one optimal solution is to examine the deterioration of prediction error with the reduction of patterns. The prediction errors resulting from the entire training data set are shown in Tables 5.2 and 5.3 for Lorenz series and river flow time series respectively. Results of data extraction are shown in Figures 5.4, 5.5, 5.6, 5.7 and 5.8 respectively for the

noise-free Lorenz series, 5% and 30% noisy Lorenz series, and Mississippi and Wabash river flow time series respectively.

In the Figures, the prediction error (in MAE) is expressed as a percentage of the prediction error resulting from the entire training data patterns (Eqs. 5.4). Similarly, the reduced number of patterns is expressed as a percentage of the total number of patterns used (Eq. 5.5).

$$MAE\ (\%) = \frac{(MAE)_{reduced\ \ number\ of\ \ patterns}}{(MAE)_{Total\ \ number\ of\ \ patterns}} \times 100 \qquad (5.4)$$

$$Number\ of\ \ patterns\ (\%) = \frac{Reduced\ \ number\ of\ \ patterns}{Total\ \ number\ of\ \ patterns} \times 100 \qquad (5.5)$$

In the noise free time series (Figures 5.4), models trained with smaller data sets of about 60% of the entire training data set have provided the same prediction performance as the model trained with the entire training data set. In this noise free case, the possibility of reduction of the data may be due to the fact that not all the data are necessary to represent a certain relationship.

In Lorenz series with 5% and 30% noise levels (Figures 5.5, 5.6), reduced data sets of about 30% - 40% of the entire training data set produce equally good predictions as that using the entire training data sets. Results are consistent with both prediction models, local averaging model and ANN. In the Mississippi river flow time series (Figure 5.7), reduction of data up to about 40% of the entire training set does not seem to affect the prediction error considerably. Similarly, in Wabash flow series (Figure 5.8), the data sets of about 30%-40% of the entire training data set produce equally good predictions as that using the entire training data set, with ANN. However,

the performance with local averaging model is very poor. In Lorenz series contaminated with some noise levels and in real flow time series the possible amount of data reduction is higher than that of noise-free time series. This could be due to effects of noise; when points are closer than the effective noise level, they may not contain distinct information.

On the average, the data sets that produce good predictions on local averaging produce good predictions on ANN too. This may be an indication that the reduced data sets are truly representative of the entire training data set. The somewhat inferior performance of the reduced data sets on local averaging models compared to ANN models, especially on Lorenz series with 5% noise level, Mississippi and Wabash river flow time series, could be due to the nearest neighbours (k) being not optimally chosen for each individual case (note that these series have very low k values; for example, optimal k in Wabash series is only 3).

Results show that there are considerable amount of redundant data (for prediction purposes) in real as well as in synthetic time series when reconstructed in a phase space. It is possible to extract only a smaller set of representative data from a long data set by filtering out these redundant data. The proposed procedure with SCM is shown to be effective in extracting representative data sets from long data records. However, since SCM has 4 parameters to be fine-tuned, use of SCM is very costly in terms of computational time. The next section proposes a new, simple method that has only one single parameter and yet has the same effectiveness as SCM for data extraction in chaotic time series.

5.2 SIMPLE CLUSTERING METHOD

This section proposes a new clustering algorithm similar to SCM but has only a single parameter. The algorithm of the present clustering method is based on the

following observation. Consider a few trajectories of an attractor lying close to each other (Figure 5.9(a)). If the time series is noisy, the points may take positions deviated from their true states as shown in Figure 5.9(b). What this means is that for a noisy time series one cannot distinguish the trajectories closer than an effective noise level separately. One may only get a rough indication of the regions the phase space trajectories evolve. Not all the points are essential for this purpose. A few representative points can indicate the directions and the locations of the evolving trajectories. Since points closer than an effective noise level do not provide any distinct information, one may choose one point to represent a neighbourhood roughly of the order of the noise level. This is the basic idea underlying the present clustering algorithm. As noted earlier, two possible ways of data being redundant are: (1) not all points are necessary to represent a certain relationship, and (2) when points are closer than the effective noise level, each of them may not contain distinct information. The algorithm that is proposed is based on the second reason. An overview of the method is given below.

The algorithm uses both a density measure and a distance measure to select the cluster centers. The density measure similar to SCM is such that the points that are closely surrounded by other points have a higher density and are more likely to be chosen as cluster centers. The distance measure defines the neighborhood size or the minimum distance between two cluster centers. Overlapping neighborhoods are allowed. The present algorithm ensures that every point in the original data set is either a cluster center or close to a center by a distance smaller or equal to d after clustering; where d is the distance measure of the method.

Practically all clustering techniques, based on classification point of view, treat isolated points (points that are far from other points) as outliers. However, in analysis

of time series such as river flow time series, such points often represent extreme events (e.g. very high flows), which are important. The points lying in the less dense areas of data space are not considered as outliers in the proposed clustering method. This is how the present method is radically different from other clustering methods. The selection of points lying far from other points is achieved in the algorithm in a way that an additional parameter determining stopping criteria is eliminated.

5.2.1 Simple clustering algorithm

The present clustering algorithm can be given as follows. Consider N points, $X_i,\ i = 1, 2, \ldots N$, of dimension m. Assume that these points have been normalized so that they lie in a unit hypercube. This makes it possible for the only parameter of this method, d, which defines the neighbourhood, to be specified without using the domain specific knowledge.

Step 1: Calculate a density measure for each point X_i. Similar to SCM, a Gaussian 'influence function', which indicates the influence of each data point on a certain data point, is used as the density measure.

$$P_i = \sum_{j=1}^{N} e^{-\left(\|X_i - X_j\|^2 / d^2 \right)} \qquad (5.6)$$

where P_i = density measure of point i, and d = radius of neighbourhood. The density P is higher for closely surrounded points and lower for less surrounded points. d may take small positive values less than 1 (a guide to tune d is given later).

Step 2: Select the point with the highest density as the first cluster center.

Step 3: Set the density measure of the selected cluster center and the density of points closer than d from the selected cluster center to zero.

Step 4: Select the point with the next highest density measure. If its density measure is greater than 0 select the point as a cluster center and go to step 3. Else stop.

Note that no additional parameter is required as stopping criteria.

The effective radius for calculation of P_i (Eq. 5.6) is approximately $2d$ while the neighbourhood size (or the minimum distance between two cluster centers) is d. The selection of this ratio is arbitrary; however, our experience shows that this combination provides sufficiently good performance. Changing this ratio may not improve the clustering performance significantly.

5.2.2 Application and results

The data extraction procedure discussed in the section 5.2.2 and illustrated in Figure 5.1 is now applied with the proposed clustering method. Instead of mGA, an exhaustive search on the single parameter is performed (Figure 5.10). Higher values of d give smaller number of cluster centers and vise- versa. When the number of centers is too small the resulting prediction performance is anticipated to be poor. An optimal d value which gives a balance between the number of cluster centers and prediction performance is preferred. Noting that d may be related to the noise level, and for moderate noise levels the effective values of d may take values close to zero, this study started off with 3 trial values for d, 0.001, 0.1 and 0.5, and used interval bisection strategy to identify a suitable range for d (values which give low numbers of patterns and satisfactory predictions) to be explored. Practically, it may suffice to identify a single solution with satisfactory low number of patterns and prediction performance. However, for exploratory and illustration purposes, the study considered a range of d values. Once the suitable range for d is determined, the range is evenly subdivided into approximately 50 points. The reduced data sets obtained resulting from these d values are used for validation. The d values and the corresponding extracted number of

patterns and the prediction errors resulting from the reduced data sets for the time series analyzed are given in Appendix K.

Similar to the case of SCM, results from noise-free Lorenz series, and Lorenz series with 5% and 30% noise levels, and the two river flow time series (Mississippi and Wabash) are shown in Figures 5.11, 5.12, 5.13, 5.14 and 5.15 respectively. The results show that the performance of simple clustering method is very similar to that of SCM. On all the time series considered, both the methods have produced similar amounts of percentage reductions of data (without considerably affecting the prediction accuracy). For example, on noise-free time series, 60% of the total data have been derived, by both methods, as representative data, which have produced equally good predictions as that when the entire training data set is used.

On Mississippi river flow time series, only a few solutions with good predictions on local model appear (Figure 5.14) because solutions with smaller number of patterns provided lower prediction accuracy (prediction errors of worse than 140% that of the entire training data set, which are not shown in the figure). The inferior performance of reduced data sets on local models compared to ANN, as noted before, could possibly be due to the *k* values used being not optimal.

5.2.3 Similarities/differences and advantages/disadvantages of the simple clustering method over SCM

The SCM and the new simple clustering method share several similarities. Both methods select a subset of original data as cluster centers. They use a similar density measure to evaluate the potential of a data point as a cluster center. Thus, both methods give priority to points closely surrounded by other points as cluster centers. The SCM discourages closely spaced cluster centers whereas the new clustering method

eliminates the selection of cluster centers which are closer than a certain distance. The SCM discourages points, lying far away from other points, being selected as cluster centers, whereas the new clustering method ensures the selection of such points. This is how the new clustering method is completely different from practically all other clustering techniques. The advantages and disadvantages of the new clustering method over SCM, with data extraction application in mind, are explained below.

The proposed simple clustering technique achieves the same performance as that from SCM, with much less effort. In this study, the SCM solutions are derived from about 1000 evaluations in each time series whereas the new method uses only 40-50 evaluations on each time series. It should also be noted that since there is only a single parameter, it is possible to reach an optimal solution with the new method using the interval bisection strategy with much less effort than 50 evaluations.

With the new method, there is a gradual variation between the parameter d and the number of patterns selected (Figure 5.16). This is beneficial since one may adopt a trial and error approach to arrive at data sets of desired sizes. This is not the case with SCM; since the number of patterns selected depends on more than one parameter, it makes the manipulation of the parameters by trial and error difficult. Furthermore, it is noticed that even with large number of evaluations with micro-genetic algorithms, SCM does not produce solutions that cover the whole range of number of patterns (e.g. see Figures 5.6 and 5.8: no solutions representing more than 80% of the total data and between 25% – 35%) if special care is not taken to ensure the diversity of the solutions.

The selection of points lying far away from other points may be disadvantageous when data sets with outliers (points that do not represent system dynamics) are analyzed.

5.2.4 Simple clustering method applied on a multivariate data set: Bangladesh data water level data

Although the proposed clustering method is developed with noisy chaotic time series in mind, the method is shown to be effective on other multivariate data as well. This section shows the performance of the method on a multivariate data set, Bangladesh water level data.

Bangladesh, a land area of approximately 145,000 km^2 is located on the world's largest delta comprising three large rivers, the Ganges, the Brahmaputra and the Meghna. All the rivers carry heavy runoff during the monsoon period (May to September) when their catchment receive intense rainfalls as high as 11,000 mm. The major rivers have their origins outside Bangladesh, and only 7.5% of the total catchment area of 1,500,000 km^2 lies within Bangladesh. Liong et al. (1999) suggested a data driven approach for predicting water level in Dhaka using minimum information, the historical water level data available within the country. Liong et al. (1999) identified, based on a sensitivity analysis, 5 out of 8 stations as significant contributors to the flood stage at Dhaka.

This study uses the water level data from those 5 most significant stations. A schematic diagram showing the stations is given in Figure 5.17. The daily data from the 5 stations during monsoon seasons from 1991 – 1996 (841 records) are used in this study. Similar to the study by Liong et al. (1999), the data set is divided into two sets: a training set and a validation set; and 467 patterns recorded in 1992, 1993, and 1995 are used for training and 374 patterns recorded in 1991, 1994, and 1996 are used for

validation. One day ahead prediction of water level at Dhaka is considered. The input/ output relation desired is expressed as follows,

$$ST12_{i+1} = \mathbf{F}\ (ST33_i, ST11_i, ST14_i, ST18_i, ST12_i) \qquad (5.7)$$

where, $ST33_i$, $ST11_i$, $ST14_i$, $ST18_i$, $ST12_i$ are inputs and $ST12_{i+1}$ is the output.

The prediction performance on training and validation sets using the ANN model trained with all training patterns is shown in Table 5.4. The new clustering technique is applied on the data as before except now only ANN models are used and no calibration is performed. The prediction performance of the reduced data sets on validation set is shown in Figure 5.18. The results show that it is possible to derive a small set of data of about 20% of the entire training data set and yet still maintain the same prediction accuracy as that of the entire training data set. The results show that the technique is as equally effective on multivariate Bangladesh water level data as on chaotic time series.

5.2.5 Tuning the parameter *d*

Higher values of *d* give smaller number of cluster centers and vise – versa. In the extreme cases, very high values will result in only one cluster center selected while very low values will result in all the points selected as centers. Identifying the 'effective range' of *d*, the range between the two extremes (see Figure 5.19) may allow the tuning of the parameter even easier. The lower bound (d_1 in Figure 5.19) corresponds to the shortest distance between two points in the data set (with the exception of zero) and the upper bound (d_2 in Figure 5.19) corresponds to the largest distance between two points in the data set. It is noticed, generally, that the effective range of *d* values is concentrated close to zero. The range of *d* is data dependent. Our experience shows that the starting values of a lower bound and an upper bound: δ and 0.5 (where δ is a very small value, e.g. 0.001) are sufficient to indicate a suitable

range for *d* for most of the data sets. Once a range is chosen, one may use interval bisection to arrive at an optimal *d* value for the data extraction.

5.3 DATA EXTRACTION WITH SIMPLE CLUSTERING METHOD DEMONSTRATED ON EKF NOISE REDUCTION APPLICATION

This section, as an example, demonstrates the advantage of using data extraction technique to circumvent the time consuming applications. A practical problem, the time consuming application of EKF estimates as noise-reduced data with the proposed noise reduction scheme, discussed in Chapter 4 is used. A Lorenz series with 10 % noise level is chosen for the demonstration. This series took approximately 13 hours (on Pentium IV, 2.4 GHz, 512 MB RAM machine running Windows XP) to derive the optimal parameters through the proposed procedure discussed in Section 4.5. The reason for the long time required is because of the need to train ANNs for each evaluation. This section investigates the application of the data extraction with the new clustering technique to overcome the aforementioned problem. The experiment is designed as follows.

Earlier results of this Chapter showed that up to about 30% - 40% reduction of data is possible on noisy chaotic time series. This experiment selected a representative data set of approximately 50% of the training data of 10% noisy Lorenz series by applying the proposed new clustering technique. Instead of using the entire training data, the noise-reduced values of these extracted data are then used to train the ANN prediction model in the procedure shown in Chapter 4 (see Figure 4.8). Results obtained are illustrated in Table 5.5. Figure 5.20 shows the prediction errors corresponding to certain parameter sets (observation noise covariance and process noise covariance) when total noise-reduced data is used in model training compared to when smaller set of data (50% of the total) is used for model training. Figure 5.20

shows that although there is a slight deterioration of accuracy with the derived data set, the variation is linear, i.e. the optimal parameters identified using the entire training data set are identified as optimal with smaller set of data as well. Table 5.5 shows that by using the derived data set, a considerable reduction in computational time (less than half the time taken with the total data set) is achieved at the expense of a negligible reduction in prediction accuracy. This example application shows that application of data extraction technique can be useful in many such time consuming applications.

5.4 CONCLUSION

A method coupling SCM (Chiu, 1994), a prediction model, and an optimization method (mGA) is proposed for extracting representative data sets from long data records reconstructed on phase space. It was demonstrated on noise-free chaotic Lorenz series, Lorenz series contaminated with some known noise levels, and Mississippi and Wabash river flow time series. Considerable reduction in data sets was observed on all time series without affecting the prediction accuracy. Results showed that river flow time series contain considerable amounts of redundant data when reconstructed on phase space. Some advantages of having a small set of data are namely the reduction of required storage capacity for training data and the reduction in computational time taken for training forecasting models. The success of the proposed method shows the ability of the clustering techniques to extract representative data sets from long numerical data patterns. SCM clustering technique has, however, 4 parameters to be fine-tuned and, therefore, requires considerable computational effort.

A new, simple and yet effective clustering method, which has only one single parameter to be fine-tuned, is developed in this study to extract representative data sets from chaotic time series. Application of the method on Bangladesh data showed that the proposed clustering method is effective on other multivariate data sets as well. The

new method is shown to be equally effective as SCM in deriving representative data sets. The method has several advantages over SCM. Having only a single parameter, it requires much less effort to find the optimal values of the parameter compared to SCM. In the new method, the number of extracted patterns has a gradual variation with the parameter; it can therefore be easily manipulated to obtain solutions of desired number of patterns. Guidelines to tune the parameter are also proposed and the tuning can be easily achieved through interval bisection strategy.

The uniqueness of the proposed new clustering method over practically all other clustering techniques is that the method does not treat data points lying away from the main bulk as outliers. Therefore, it is hoped to capture the information should such points correspond to infrequent events of systems (e.g. flood flows in river flow systems). However, the selection of data lying away from the other points can be detrimental when such points are actually outliers, i.e. points that do not represent system events.

The data extraction is demonstrated on a practical problem, the EKF applied on the proposed noise reduction scheme (in Chapter 4). Lorenz series with 10% noise level was considered in the analysis and it was shown that the time taken for the computations was reduced by more than half with only a negligible reduction in the prediction accuracy as a result. This indicates the potential of using data extraction proposed in this study in the practical applications for efficient analysis.

Table 5.1 Criteria for selection of cluster centers

Criterion	Detail	Note
1st	$\frac{P_k^*}{P_1^*} > AR$	Accept y_k as a cluster center
2nd	$AR \geq \frac{P_k^*}{P_1^*} \geq RR$ and $\frac{d_{\min}}{r_a} + \frac{P_k^*}{P_1^*} \geq 1$	
3rd	$AR \geq \frac{P_k^*}{P_1^*} \geq RR$ and $\frac{d_{\min}}{r_a} + \frac{P_k^*}{P_1^*} < 1$	Reject y_k as a cluster center
4th	$\frac{P_k^*}{P_1^*} < RR$	
d_{min} is the shortest of the distance between y_k^* and all previously found cluster centers		

Table 5.2 Prediction errors of ANN and local averaging models trained with the entire training data set: Lorenz time series

Noise level	ANN		Local Averaging method	
	(m, τ)	Prediction error on Validation set MAE	(m, τ, k)	Prediction error on Validation set MAE
Noise-free	(7, 6)	0.0032	(7, 6, 8)	0.3115
5%	(10, 3)	0.6395	(10, 3, 8)	0.7418
30%	(10, 3)	4.0604	(10, 3, 22)	4.0991

Table 5.3 Prediction errors of ANN and local averaging models trained with the entire training data set: River flow time series

Noise level	ANN		Local Averaging method	
	(m, τ)	Prediction error on Validation set MAE (m^3/s)	(m, τ, k)	Prediction error on Validation set MAE (m^3/s)
Mississippi river	(3, 1)	207.31	(3, 1, 5)	290.76
Wabash river	(5, 1)	25.66	(5, 1, 3)	47.56

Table 5.4 Prediction errors of ANN trained using total training data applied on validation set: Bangladesh water levels

Prediction error (MAE) (meters)	
Test set	Validation set
0.0451	0.0525

Table 5.5 Prediction errors of EKF noise reduction application on 10% noisy Lorenz series with total data in model training and reduced data (with new clustering method) in model training

Amount of data set used	Prediction errors (MAE)		Computational time (hrs)
	Against noisy data	Against noise-free data	
100% of training data set	1.1105	0.4672	13
50% of training data	1.1061	0.4776	5.5

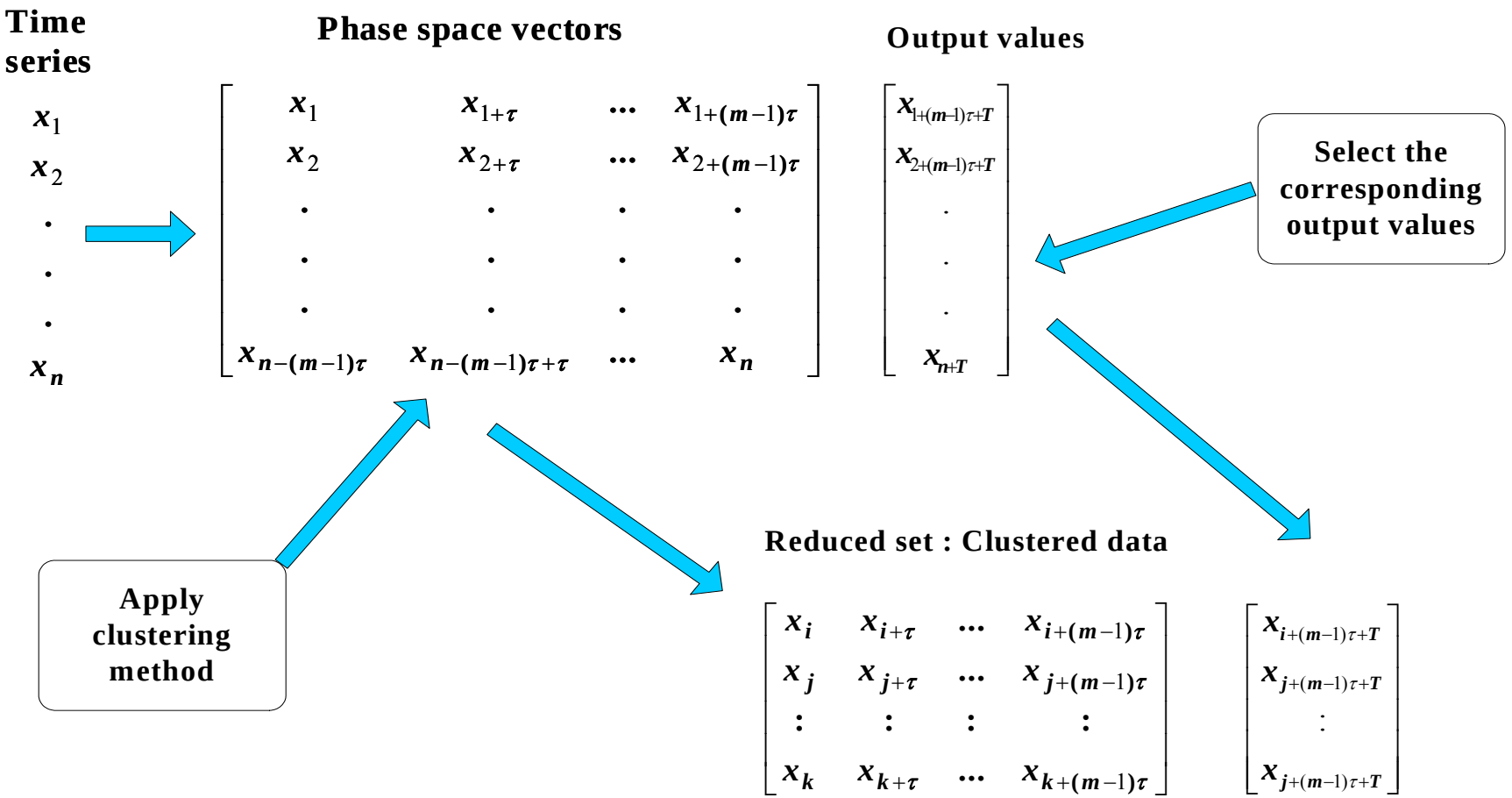


Figure 5.1 Overview of the data extraction procedure

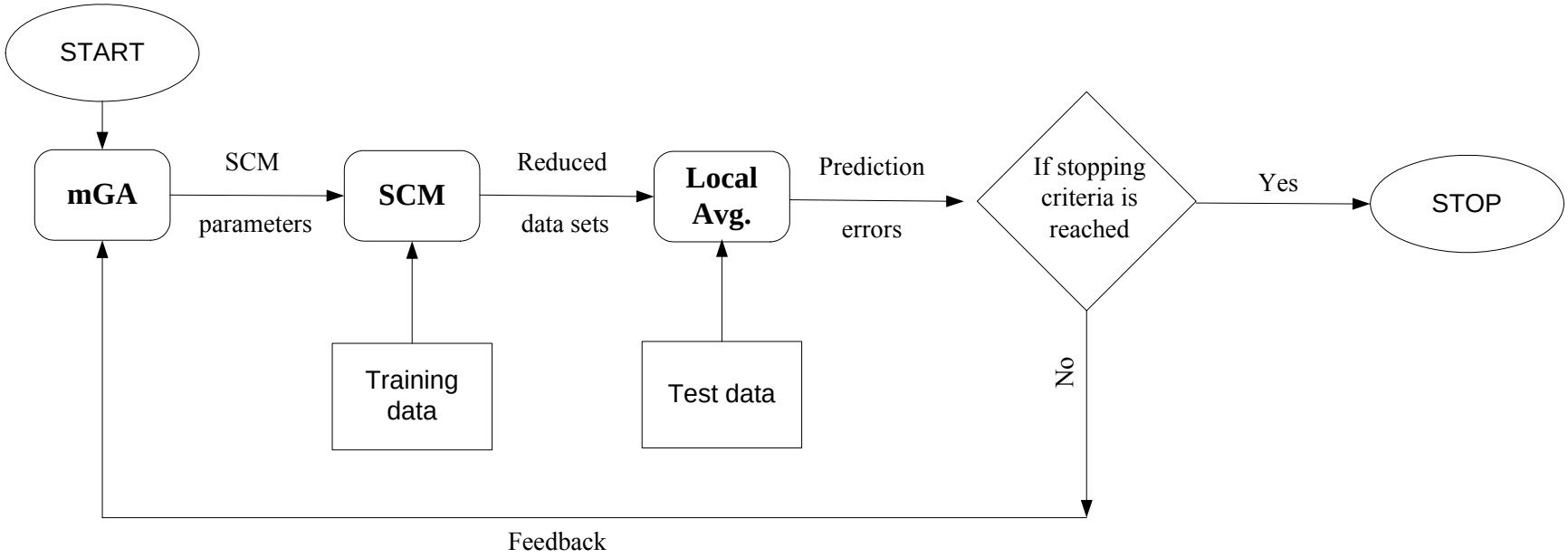


Figure 5.2 Schematic diagram of calibration process of SCM parameters

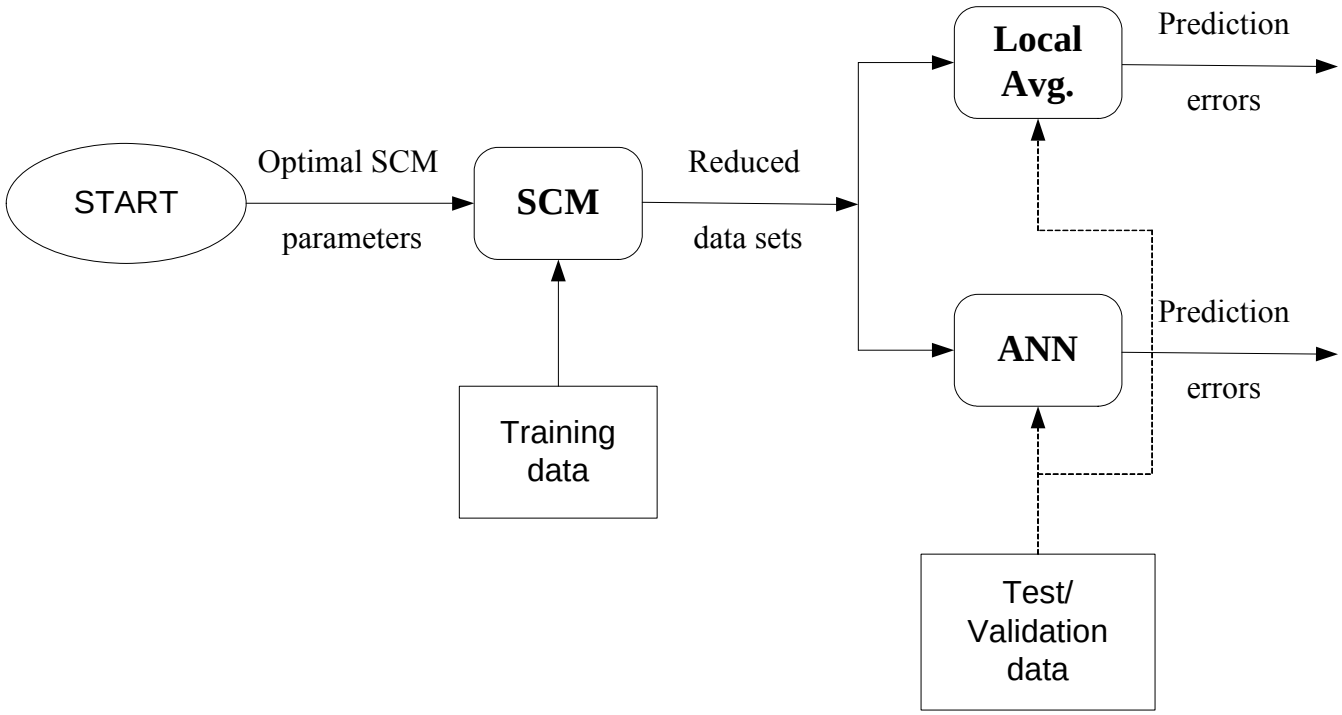
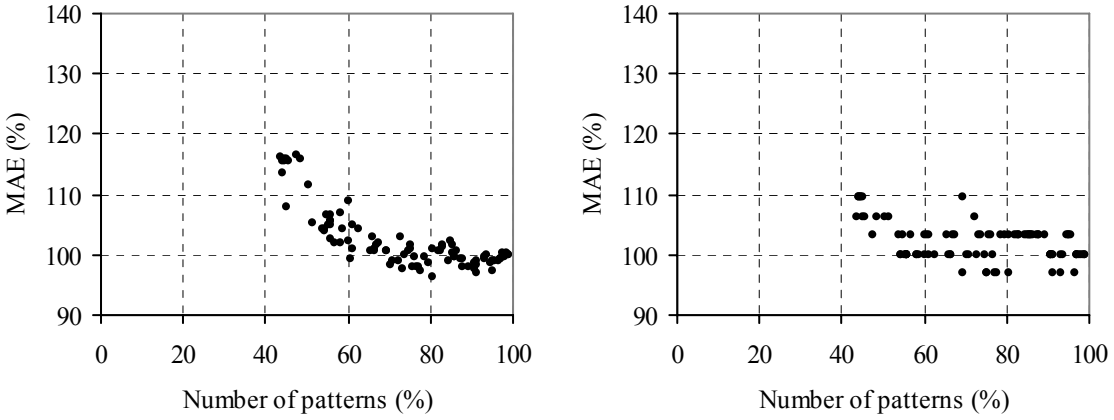
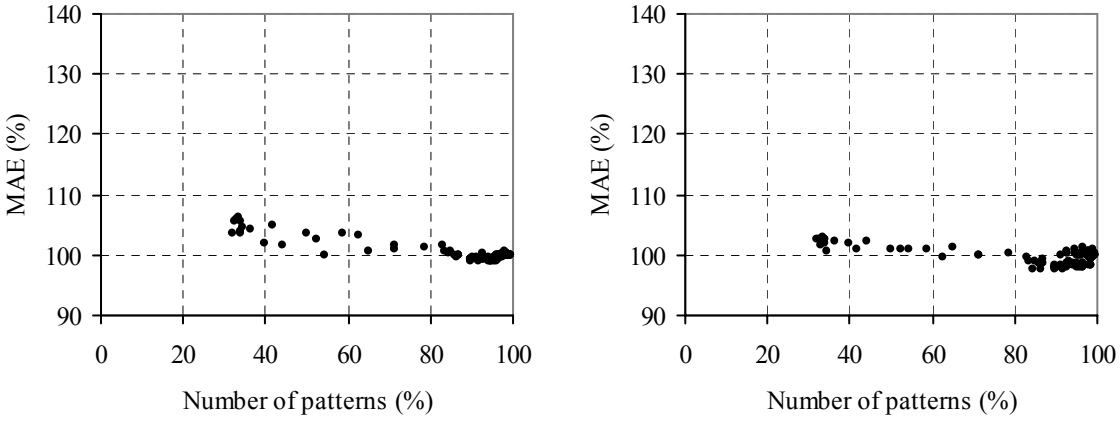


Figure 5.3 Schematic diagram of validation process of optimal solutions



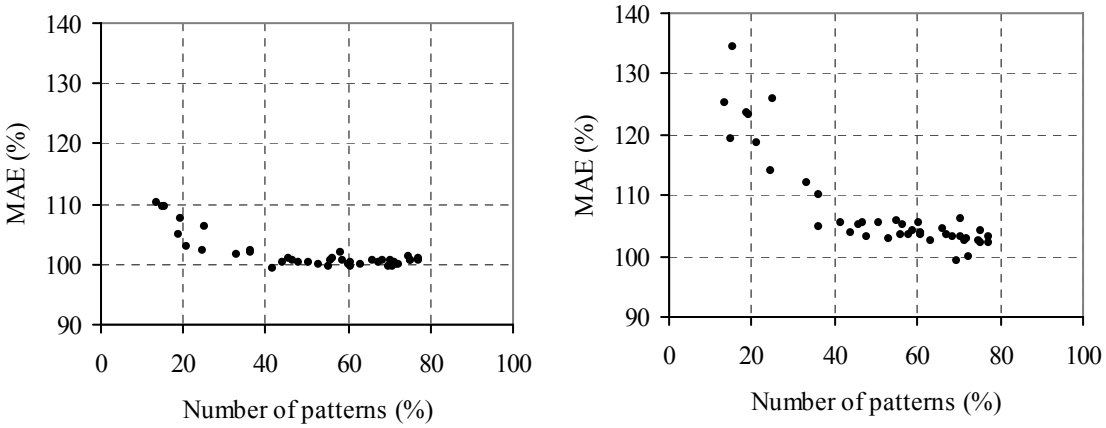
(a) local averaging model (b) ANN

Figure 5.4 Performance of SCM on validation set: Noise-free Lorenz series



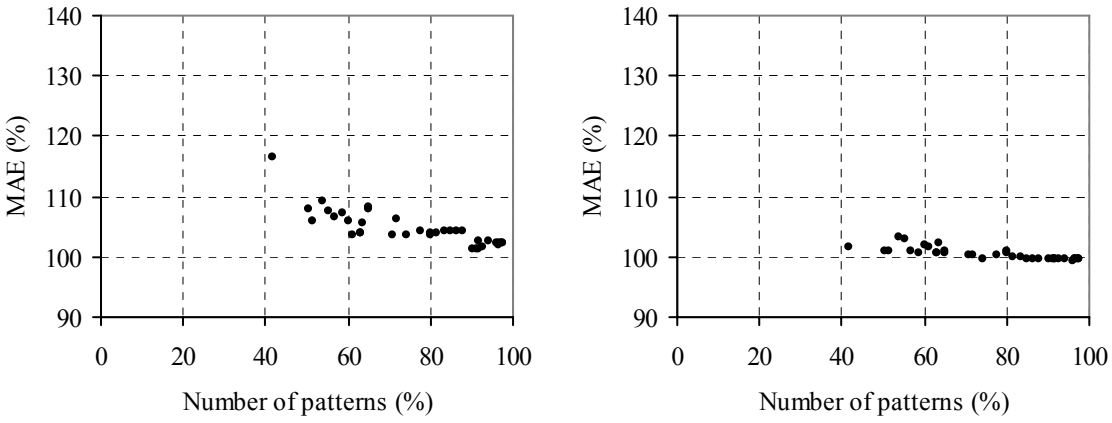
(a) local averaging model (b) ANN

Figure 5.5 Performance of SCM on validation set: 5% noisy Lorenz series



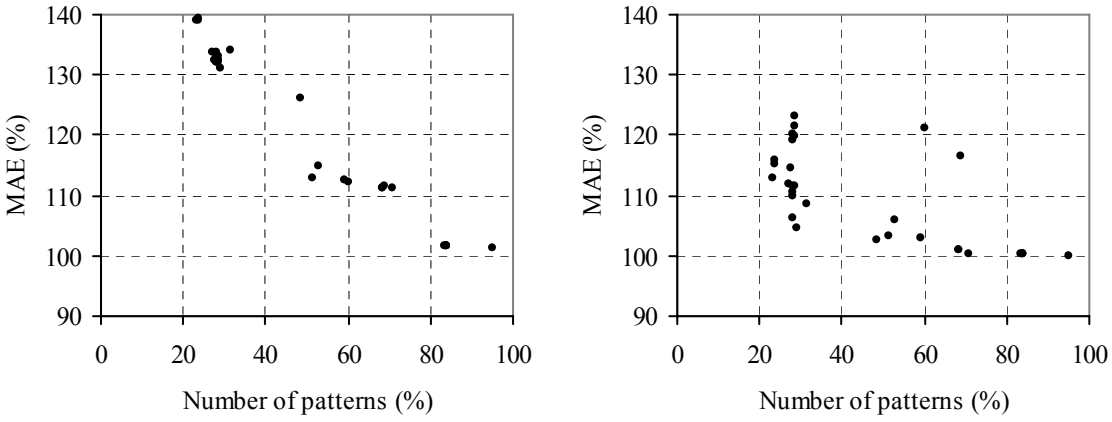
(a) local averaging model (b) ANN

Figure 5.6 Performance of SCM on validation set: 30% noisy Lorenz series



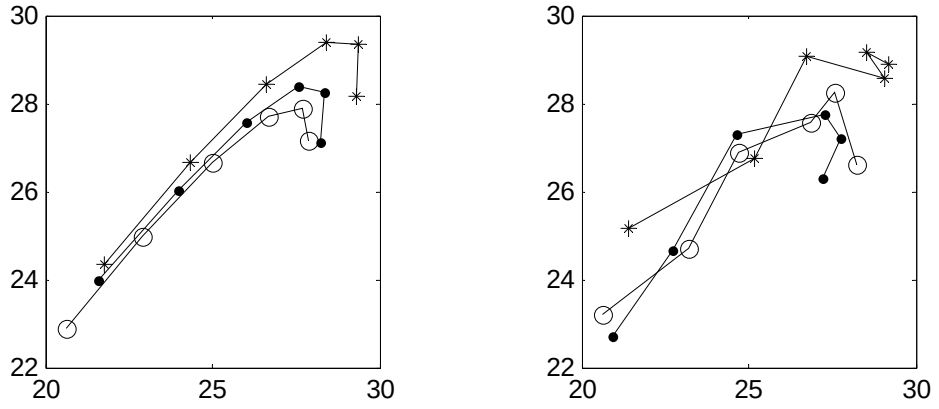
(a) local averaging model (b) ANN

Figure 5.7 Performance of SCM on validation set: Mississippi flow time series



(a) local averaging model (b) ANN

Figure 5.8 Performance of SCM on validation set: Wabash flow time series



(a) noise-free data (b) noisy data

Figure 5.9 Trajectories of an attractor

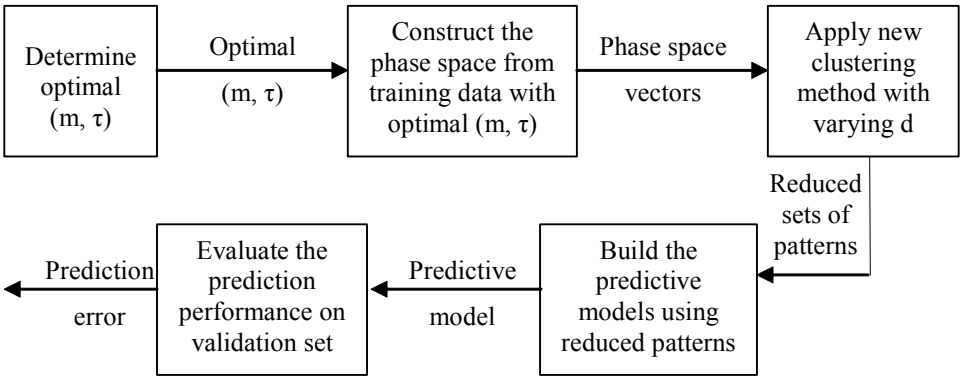
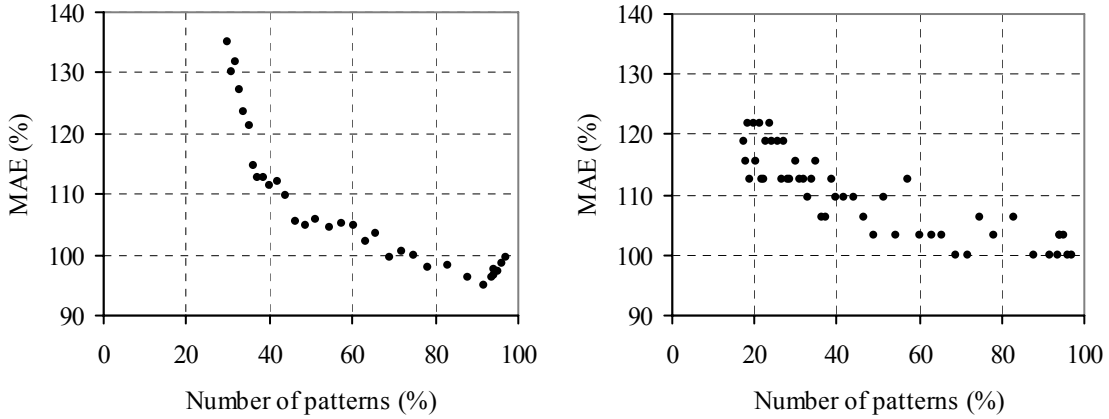
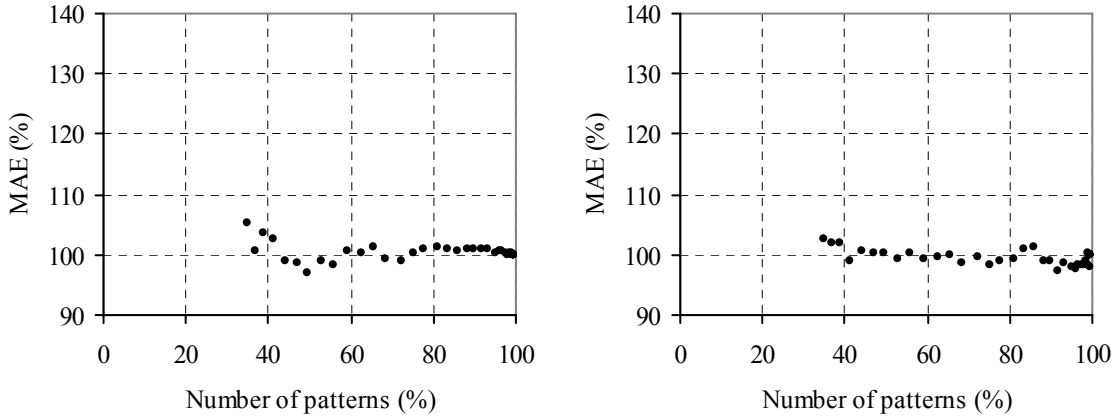


Figure 5.10 Schematic diagram of the procedure followed with new clustering method



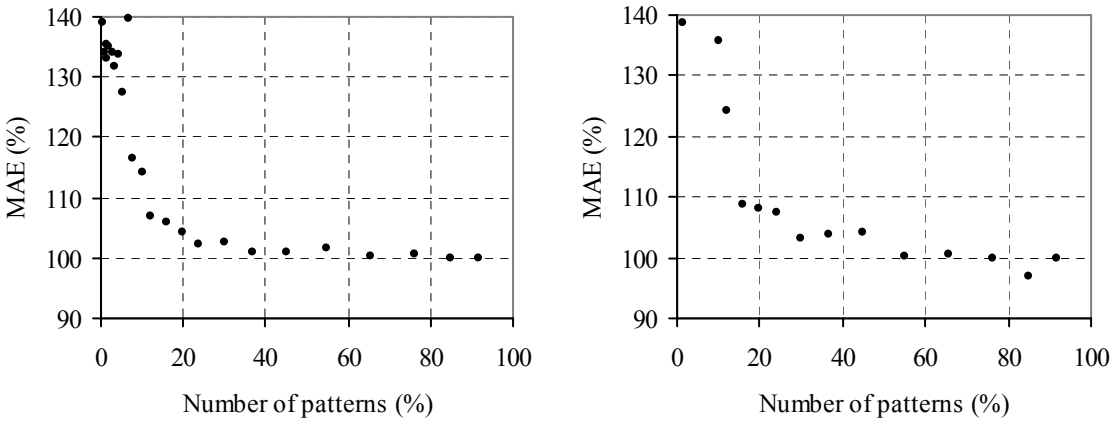
(a) local averaging model (b) ANN

Figure 5.11 Performance of Simple clustering method on validation set: Noise-free Lorenz series



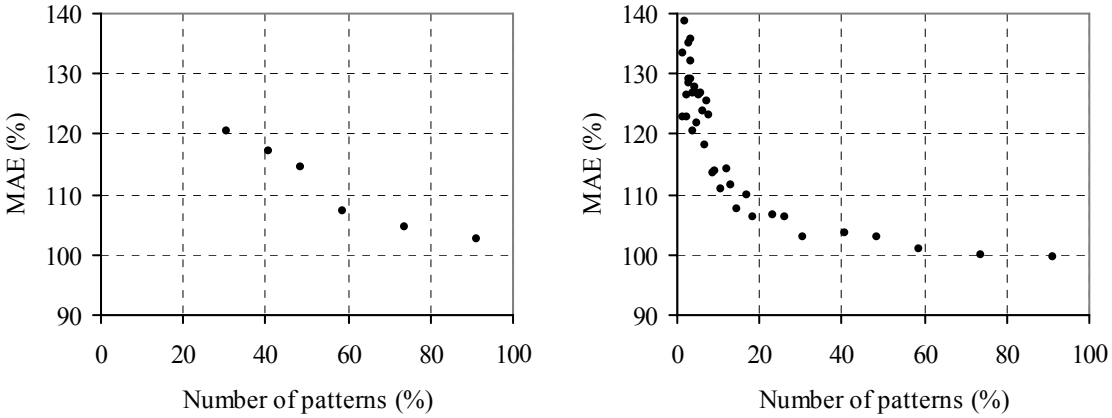
(a) local averaging model (b) ANN

Figure 5.12 Performance of Simple clustering method on validation set: 5% noisy Lorenz series



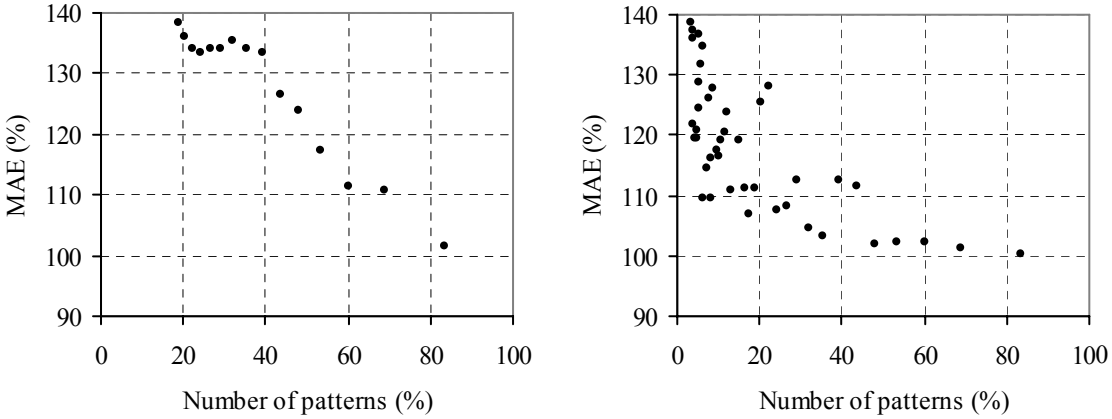
(a) local averaging model (b) ANN

Figure 5.13 Performance of Simple clustering method on validation set: 30% noisy Lorenz series



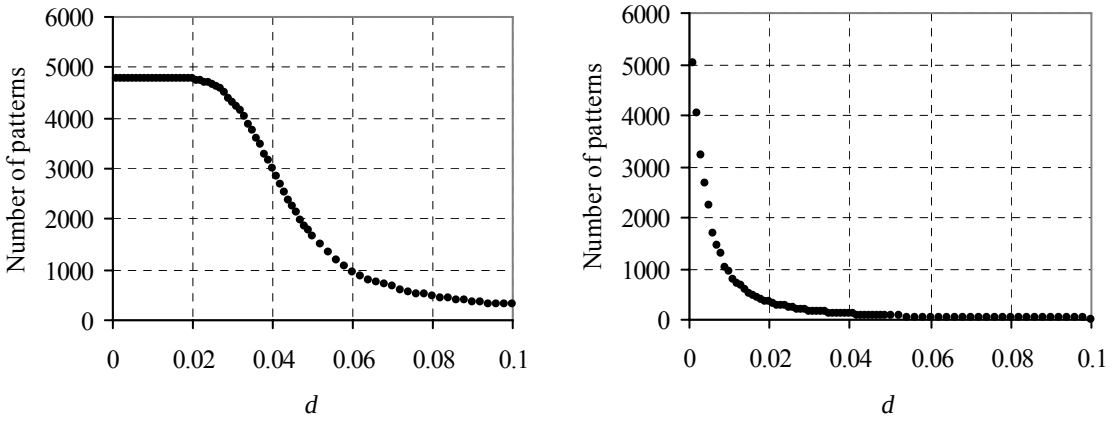
(a) local averaging model (b) ANN

Figure 5.14 Performance of Simple clustering method on validation set: Mississippi flow time series



(a) local averaging model (b) ANN

Figure 5.15 Performance of Simple clustering method on validation set: Wabash flow time series



(a) Lorenz series with 5% noise level (b) Mississippi flow time series

Figure 5.16 Variation of number of patterns with neighborhood size (d)

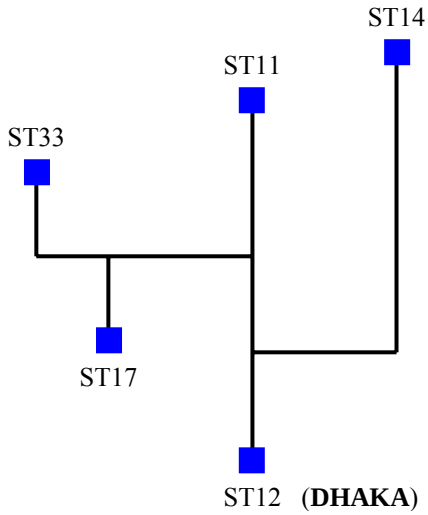


Figure 5.17 Schematic diagram of river system showing the stations (ST)

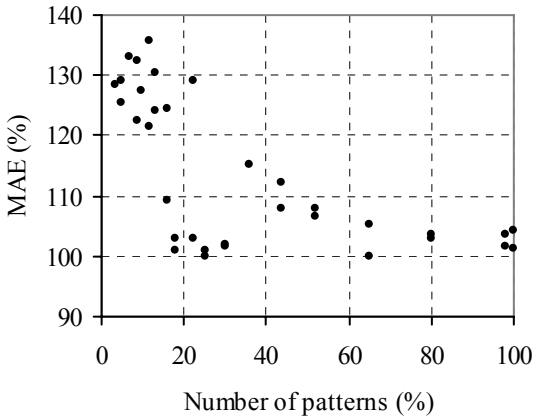


Figure 5.18 Performance of Simple clustering method on validation set: Bangladesh water levels with ANN model*

Note: * Because ANN models converge to different solutions with different sets of initial weights the MAE is different from 100% in this case

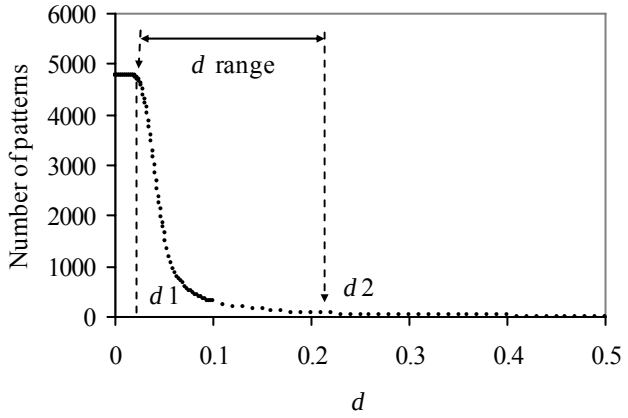


Figure 5.19 The effective range for d : from $d_1 - d_2$

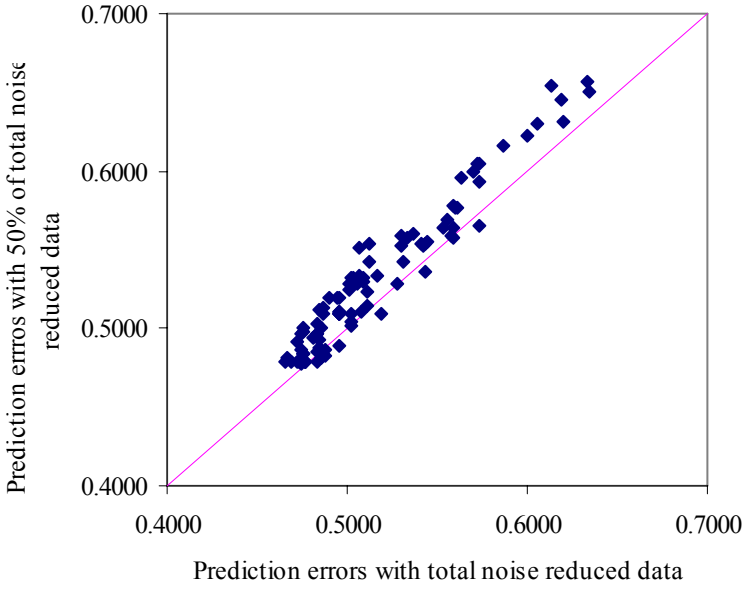


Figure 5.20 Comparison between prediction performances of smaller data sets (50% of the total) and total data set used to train model: EKF noise reduction application

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

6.1 SUMMARY

The potential in short term prediction, of chaotic dynamical systems approach for time series prediction, is widely recognized. The approach is finding its applications more and more in hydrological time series too, especially in the analysis of river flow data. The prediction accuracies are, however, still not at a very satisfactory level with the conventional chaos prediction techniques. One difficulty with the chaotic approach for time series analysis is the large number of past data records required to yield satisfactory prediction. The problem is compounded with the new data records streaming in daily. This demands lots of computational resources. This study looked into means of improving prediction accuracy and facilitating efficient analysis. The primary objectives of this study were thus (1) to investigate in detail the prediction performance of global prediction models (Artificial Neural Network and Support Vector Machine) compared to some widely used local prediction models, and (2) to investigate means of incorporating noise reduction techniques to improve prediction, and (3) to investigate means of extracting system representative smaller sets of data from long data records.

The study showed the superiority of the global prediction models over the widely used local prediction models. Comparison between the two competitive machine learning techniques, ANN and SVM, showed that they are equally good predictors. Kalman filtering technique was introduced to further improve the prediction accuracy of noisy chaotic time series. A methodology incorporating noise reduction for real-time prediction was proposed. To facilitate an efficient analysis, a methodology to

extract system representative smaller data sets from long chaotic data records was proposed. To the best of the author's knowledge, this study is the first to propose such a data extraction technique for chaotic time series analysis. A simple and much more efficient clustering technique was also developed for data extraction purposes. All the above techniques/ methodologies were demonstrated on a benchmark chaotic Lorenz series and two river flow time series. Applications showed how the proposed data extraction scheme can make time-consuming chaos analysis of long data records more efficient.

6.2 GLOBAL MODELS IN CHAOTIC TIME SERIES PREDICTION

The chaotic time series prediction has been thus so far very much confined to local phase space prediction models due to their simplicity. Also there is a general understanding, that local approximation can give better predictions than global approximation. This study assessed the performance, in detail, of the two promising machine learning techniques, ANN and SVM, compared to the two widely used local models, local averaging technique and local polynomial models. The investigations were first performed on a synthetic noise-free chaotic Lorenz series. Since the real world data contain noise, the analysis was then performed on the same Lorenz series, however, contaminated with some known noise levels (5% and 30%). Finally the results were demonstrated on two river flow time series, Mississippi and Wabash Rivers. Three prediction horizons, 1, 3 and 5, were considered. ANN outperformed local prediction models in practically all the cases. SVM, implemented with a decomposition technique to facilitate handling large data records, also performed better than local models with the exceptions of noise-free Lorenz series where its performance was poor compared to local polynomial models. On the average both the global prediction techniques (ANN and SVM) outperformed the local prediction

models considered. It can be concluded that global models (ANN, SVM) can yield better prediction performance than, if not equal to, that of the widely used local models.

Comparison of ANN with the relatively new SVM prediction tool showed that both techniques are equally good. However, on noise-free Lorenz time series, SVM required much longer computational times and produced very poor prediction performance compared to that of ANN. For real world chaotic time series, the difference between prediction accuracies of ANN and SVM is insignificant. SVM is mathematically well founded compared to ANN, which is founded more on heuristics. The perception is, therefore, that SVM may outperform ANN, although no direct comparisons have been conducted (e.g. Sivapragasam, 2002; Yu, 2004). However, as shown in this study, SVM in its present form does not outperform, in terms of prediction accuracy and computational effort, the ANN at least in chaotic time series prediction.

6.3 NOISE REDUCTION

Although global models produce better predictions than local prediction models, their performance is still unsatisfactory when data is noisy. This study identified some means, with the use of data driven prediction models, to improve the predictions of noisy chaotic time series. It was shown that the key factor to improve prediction accuracy is noise reduced input data. Contrary to the general anticipation that the use of noise reduced data to train models may help in improving prediction, the findings of this study shows that the prediction performance is not necessarily improved if those trained models are not supported with input data of equal or lower noise levels. Due to the above reasons, the study showed the necessity of real-time application of noise reduction.

It was noticed that there is a need to identify nonlinear noise reduction techniques capable of real-time application. It was shown that the Extended Kalman filter, from Controls literature, can be used as a reliable and robust technique for real-time noise reduction in chaotic time series. Use of less than perfect models, i.e. models trained with noisy data, as the state space model in EKF was shown to be feasible. Application of the EKF predictor, with the models trained with noisy data, improved the prediction performance of chaotic time series over the ANN model.

The study proposed a better scheme incorporating noise reduction to improve prediction of chaotic time series. The scheme couples the use of model trained with noise reduced data and the use of noise reduced input data. The scheme consists of two phases: an off-line phase where a model is trained with noise reduced data and a real-time phase where the inputs are noise reduced and the predictions are made with the model. This scheme has eliminated the short-comings of the earlier approaches. The effectiveness of the proposed scheme was demonstrated with EKF incorporated as a noise reduction method. The proposed scheme was shown to be more effective than the EKF alone.

Identifying the characteristics of measurement noise and then applying the appropriate noise reduction methods accordingly are hoped to improve the prediction performance in real world data.

6.4 DATA EXTRACTION

A major difficulty in chaotic time series analysis is the large number of data records required to yield satisfactory predictions. This demands significant computational resources (memory space and computational time). This study proposed a data extraction procedure that couples a clustering method, a phase space prediction method, and a parameter optimization method (mGA). Demonstration of the procedure

is made with Subtractive Clustering Method, SCM (Chiu, 1994), local averaging prediction model, and ANN on noise-free chaotic Lorenz series, Lorenz series contaminated with 5% and 30% noise level, and Mississippi and Wabash River flow time series. Considerable reductions (approximately 30% - 60%) of the total data sets were obtained on the time series considered. In river flow time series, reductions were possible up to about 30% - 40% of the total data sets. The results indicate that not all the points of long data record contribute distinct information for phase space prediction; this observation is irrespective of whether the series is synthetic or real. Although the SCM with the proposed procedure was successful in extracting representative smaller data sets it still required significant computational effort.

6.5 NEW SIMPLE CLUSTERING TECHNIQUE

The SCM has 4 parameters to be optimized and it requires lots of computational effort. A new clustering method is developed in this study. The new clustering method has only one single parameter. This method is shown to be as equally effective as SCM while it requires much less effort than SCM to tune its only parameter. Guidelines to tune the parameter are also proposed. Another advantage of the new method is that its parameter can be easily manipulated to derive data sets of desired sizes. These tuning operations can easily be achieved through interval bisection technique. The new method, though developed for data extraction in chaotic time series, was shown to be effective on other multivariate time series as well.

The specialty of the proposed clustering technique over practically all the other clustering techniques is that the new method does not treat data points lying away from the main bulk as outliers. Thus, it incorporates two aspects: (1) selecting the most crowded points from their respective crowded neighbourhoods, and (2) accepting most unpopulated points. Selection of unpopulated points is hoped to preserve the

information of systems, which may be of rare events and yet critical (e.g. flood flows in river flow systems).

Application of the new clustering technique on a practical problem (EKF on noise reduction scheme) showed the promising approach in extracting representative data from large data record to yield efficient analysis of the normally time-consuming applications.

6.6 RECOMMENDATIONS FOR FUTURE STUDY

The following are recommended for future research and practical applications:

(1) Global prediction tools (ANN and SVM) are recommended for chaos prediction applications in place of the widely used local prediction tools to yield high prediction accuracies. For real world chaotic time series data, similar prediction performance can be expected from both ANN and SVM;

(2) Nonlinear chaotic dynamics literature lacks established noise reduction techniques for real-time noise reduction applications. Therefore, developing noise reduction techniques and investigating the means to incorporate the existing techniques for real-time noise reduction applications are important. Nonlinear dynamical system community is suggested to consider Kalman filtering techniques developed in the Controls literature for real-time prediction and noise reduction applications. Also better filtering techniques (e.g. UKF), claimed to be better than the EKF used in this study, are recommended for further investigation;

(3) It is important to identify the nature of noise present (e.g. white/ coloured; distribution; level of noise) in the real time series and to incorporate the appropriate noise reduction methods. It is also possible that the levels of noise in real world data can change over the time. Therefore, ways to calibrate/ adapt the models should also be considered. Testing the proposed scheme for noise reduction and prediction on real

time series with considerable measurement noise levels whose nature is identified will render the generality of the proposed scheme for real data;

(4) In the proposed noise-reduction scheme, the noise reduction, model training, and noise reduction in validation input data, are performed one full cycle only. Iterative approach (noise reduction – model training – noise reduction of validation input data) may further improve the prediction performance. An investigation is suggested;

(5) This study tested noise reductions on time series contaminated only with the observation noise. Effect of dynamical noise on phase space prediction in general and on noise reduction procedures in particular should be investigated as well;

(6) The proposed new clustering method is recommended for extracting most representative data from long multivariate time series over the Subtractive Clustering Method, SCM (Chiu, 1994). The method is expected to preserve information on less frequently occurred events, with selection of these points, which are distant away from the main bulk, as cluster centers. This, however, may cause selections of undesirable outliers. Removal of outliers before application of the clustering method is recommended. The performance of the method in the presence of outliers has to be tested.

REFERENCES

1 Abarbanel, H. D. I., 1996. Analysis of observed chaotic data. Springer, New York, New York.

2 Abarbanel, H.D.I., 1990. Prediction in chaotic nonlinear systems: methods for time series with broadband spectra. Phys. Rev. Lett. 41(4), 1782-1807.

3 Abarbanel, H.D.I., Lall, U., Moon, Y., Mann, M.E. and Sangoyomi, T., 1996. Nonlinear dynamics and the great salt lake: A predictable indicator of regional climate. Energy 21(7/8), 655-665.

4 Abu-Lebdeh, G., and Benekohal, R.F., 1999. Convergence variability and population sizing in Micro-Genetic algorithms. Computer-Aided Civil and Infrastructure Engineering 14, 321-334.

5 ASCE Task Committee on Application of Artificial Neural Networks in Hydrology, (2000) Artificial Neural Networks in Hydrology II: Hydrologic applications. Journal of Hydrologic Engineering, 5(2), 124-137.

6 Babovic, V., and Keijzer, M., 1999. Forecasting of river discharges in the presence of chaos and noise. Coping with floods: Lessons learned from Recent Experiences, Marsalek, J. (ed), Kluwer, Dordrecht, 1999

7 Babovic, V., Keijzer, M., and Stefansson, M., 2000. Optimal embedding using evolutionary algorithms. Proceedings of the 4th international conference on Hydroinformatics, Iowa city, July 2000.

8 Bierkens, M. F. P. Knotters, M. and Hoogland, T., 2001. Space-time modeling of water table depth using a regionalized time series model and the Kalman filter. Water Resource. Res. 37 (5), 1277 – 1290.

9 Casdagli M., 1991. Chaos and deterministic versus stochastic non-linear modeling. J of the royal statistical society. Series B (Methodological) 54(2), 303-328.

10 Casdagli, M. 1989. Nonlinear prediction of chaotic time series. Physica D 35, 335-356.

11 Casdagli, M., Eubank, S., Farmer, J.D., and Gibson, J., 1991. State space reconstruction in the presence of noise. Physica D, 51, 52-98.

12 Cawly, R. and Hsu, G. H., 1992. Local-geometric-projection method for noise reduction in chaotic mps and flows. Physical Review A, 46(6), 3057 – 3082.

13 Cherkassky, V. and Ma, Y., 2004. Practical selection of SVM parameters and noise estimation for SVM regression. Neural Networks, 17(1), 113 – 126.

14 Chiu, S.L., 1994. Fuzzy model identification based on cluster estimation. J of intelligent and fuzzy systems 2, 267-278.

15 Collobert, R. and Bengio, S., 2001. SVMtorch: Support Vector Machine for large scale regression problems. Journal of machine learning research, 1, 143 – 160.

16 Collobert, R., Bengio, S. and Bengio, Y., 2002 A parallel mixture of SVMs for very large scale problems, Neural Computation, 14, 1105-1114.

17 Deb. K., 2001. Genetic Algorithms for Optimization. KanGAL Report Number 2001002

18 Dibike, Y. B., Velickov, S., Solomatine, D. P. and Abbott, M. B. (2001). Model induction with support vector machines: Introduction and applications. Journal of Computing in Civil Engineering (ASCE), 15(3), 208 – 216.

19 Doan, C. D. and Liong, S. Y., 2004. Derivative-free Kalman filter with state space Neural Network for prediction in water resources. Proceedings of Sixth International Conference on Hydroinformatics, Singapore, 20-25 Jun 2004.

20 Drécourt, J. P., 2003. Kalman filtering in hydrological modeling. DAIHM Technical report 2003-1, DHI water and environment.

21 Eckmann, J. P., Kamphorst, S. O., Ruelle, D., and Ciliberto, S., 1986. Liapunov exponents from time series, Physical Review A, 34(6), 4971-4979.

22 Elshobagy, A., Simonovic, S. P., and Panu, U. S., 2002a. Estimation of missing streamflow data using principles of chaos theory. J. of Hydrol. 255, 123-133.

23 Elshobagy, A., Simonovic, S. P., and Panu, U. S., 2002b. Noise reduction in chaotic hydrologic time series: facts and doubts. J. of. Hydrol. 256, 147-165.

24 Elshorbagy, A., Simonovic, S. P. and Panu, U. S., 2000. Performance evaluation of artificial neural networks for runoff prediction. Journal of Hydrologic Engineering, 5(4), 424-427.

25 Farmer, J. D., and Sidorowich, J. J., 1987. Predicting chaotic time series. Phys. Rev. Lett. 59(8), 845-848.

26 Fraser, A. M., and Swinney, H. L., 1986. Independent coordinates for strange attractors from mutual information. Phys. Rev. Lett. A 33(2), 1134-1140.

27 Gelb, A., 1974. Applied optimal estimation. Written by: Technical Staff, Analytic Sciences Corporation. Edited by Arthur Gelb. Cambridge, Mass. , M.I.T. Press.

28 Gibson J. F., Farmer J.D., Casdagli M. and Eubank S., 1992. An analytical approach to practical state space reconstruction. Physica D, 57, 1-30.

29 Goldberg, D. E., 1989a. Genetic algorithms in search, optimization, and machine learning, Addison-Wesley Pub. Co.

30 Goldberg, D. E., 1989b. Sizing populations for serial and parallel genetic algorithms. Proceedings of the Third International Conference on Genetic Algorithms, held at George Mason University, June 4-7, 1989, pp. 70-79.

31 Grassberger, P., 1986. Do climatic attractors exist? Nature 323, 609-612.

32 Grassberger, P., and Procaccia, I., 1983a. Characterization of strange attractors. Phys. Rev. Lett. 50(5),346-349.

33 Grassberger, P., and Procaccia, I., 1983b. Measuring the strangeness of strange attractors. Physica D, 9, 189 – 208.

34 Grassberger, P., and Procaccia, I., 1983c. Estimation of the Kolmogorov entropy from a chaotic signal. Physical Review A, 28(4), 2591-2593.

35 Grassberger, P., Hegger, R., Kantz, H. and Schaffrath, C., 1993. On noise reduction methods for chaotic data. Chaos, 3(2), 127 – 141.

36 Grassberger, P., Schreiber, T., and Schaffrath, C., 1991. Nonlinear time sequence analysis. International journal of bifurcation and chaos 1(3) 521-547.

37 Hammel, S., 1999. A noise reduction method for chaotic systems. Physics letters A, 148 (8,9), 421 – 428.

38 Haykin, S., 1999. Neural Networks: A comprehensive foundation. Prentice Hall, Upper Saddle River, New Jersey.

39 Haykin, S., 2001. Kalman filtering and Neural Networks. Wiley & Sons Inc. New York.

40 Hegger, Kantz, R H. and Schreiber, T., 1999. Practical implementation of nonlinear time series methods: The TISEAN package, chaos, 9, 413.

41 Holland, J. and Nafpliotis, N., 1975. Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor, 1975

42 Holzfuss, J. and Mayer-Kress, G., 1986. An approach to error-estimation in the application of dimension algorithms. In Dimensions and Entropies in Chaotic Systems. (Ed). G. Mayer-Kress, Springer-Verlag, New York, 114-122.

43 Hsu, K. L., Gupta, H. V. and Sorooshian, S., 1995. Artificial neural network modeling of the rainfall-runoff process. Water resources research, 31(10), 2517-2530.

44 Islam, M.N., and Sivakumar, B., 2002. Characterization and prediction of runoff dynamics: a nonlinear dynamical view. Advances of water resources 25, 179-190.

45 Islam, S., Bras, R.L., and Rodriguez-Iturbe, I., 1993. A possible explanation for low correlation dimension estimates for the atmosphere. Journal of applied meteorology 32, 203-208.

46 Jayawardena, A. W., and Gurung, A. B., 2000. Noise reduction and prediction of hydro-meteorological time series: dynamical systems approach vs. stochastic approach. J. of Hydrol. 228, 242-264.

47 Jayawardena, A. W., and Lai, F., 1994. Analysis and prediction of chaos in rainfall and streamflow time series. Journal of Hydrology, 153, 23-52.

48 Joachims, T. (1999). Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, Schölkopf, B., and Burges, C., and Smola, A. (ed.), MIT Press, 1999.

49 Judd, K. (2003). Nonlinear state estimation, indistinguishable states, and the extended Kalman filter. Physica D, 183, 273 – 281.

50 Julier, S. J. and Uhlmann, J. K. (1996). A general method for approximating nonlinear transformations of probability distributions. Technical Report, RRG, Department of Engineering Science, University of Oxford, November, 1996.

51 Julier, S. J. and Uhlmann, J. K. (1997). A new extension of the Kalman filter to nonlinear systems. Proceedings of AeroSense: The 11th International Symposium on Aerospace/defense Sensing, Simulation and Controls, Multi Sensor Fusion, Tracking and Resource Management II, SPIE, 1997.

52 Kalman, R.E., 1960. A new approach to linear filtering and prediction problems. Transactions of the ASME, Ser. D, Journal of Basic Engineering, Vol. 82, 34 – 45.

53 Kantz, H. and Schreiber, T., 2004. Nonlinear time series analysis (2nd Ed.), University Press, Cambridge.

54 Karunanithi, N., Grenney, W. J., Whitley, D. and Bovee, K., 1994. Neural Networks for river flow prediction. Journal of Computing in Civil Engineering 8(2), 201-220.

55 Karunasinghe, D.S.K., 2003. Effective and efficient data for phase space prediction (PhD Qualifying Exam Report), Dept of Civil Engineering, National University of Singapore.

56 Kawamura, A., McKerchar, A.I., Spigel, R.H., and Jinno, K., 1998. Chaotic characteristics of the Southern Oscillation Index time series. J of Hydrol. 204, 168-181.

57 Kostelich, E. J. and Yorke, J. A., 1990. Noise reduction: Finding the simplest dynamical system consistent with the data. Physica D, 41, 183 – 196.

58 Kreinovich, V. and Yam, Y., 2000. Why clustering in function approximation? Theoretical explanation. Int. journal of Intelligent Systems, vol. 15, 959-966.

59 Krishnakumar, K., 1989. Micro-Genetic algorithms for stationary and non-stationary function optimization. SPIE conference on intelligent control and adaptive systems Vol. 1196, Philadelphia, PA

60 Lee, Y.H. and Singh, V.P.1999. Tank model using Kalman filter. Journal of hydrologic engineering, 4(4), 344-349.

61 Legates, D.R., and McCabe, G. J., Jr., 1999. Evaluating the use of "goodness-of-fit" measures in hydrologic and hydro-climatic model validation. Wat. Resourc. Res. 35(1), 233-241.

62 Liaw, C. Y., Islam, N., Phoon, K. K. and Liong, S. Y., 2001. Comment on "Does the river run wild? Assessing chaos in hydrological systems" by G. B. Pasternak. Advances in water resources 24, 575-580.

63 Liong, S. Y. and Sivapragasam, C., 2002. Flood stage forecasting with SVM. J. Am. Water Res. Assoc., 38(1), 173 – 186.

64 Liong, S. Y., and Doan, C.D., 2002. Derivation of effective and efficient data set for training forecasting model, Proceedings of the 13th IAHR-APD Congress, Singapore, 681-686, Aug. 6-8 2002

65 Liong, S. Y., Lim, W. H., and Paudyal, G., 1999. Real time river stage forecasting for flood stricken Bangladesh: Neural Network Approach. Journal of Computing in Civil Engineering, ASCE, 4(1), 38 – 48.

66 Liong, S.Y., Phoon, K. K., Pasha, M. F. K. and Doan, C. D., 2005. Efficient implementation of inverse approach for forecasting hydrological time series using micro GA. Journal of Hydroinformatics, 7, 151 – 163.

67 Lisi, F., and Villi, V., 2001. Chaotic forecasting of discharge time series: A case study. J. of the American water resources association 37(2), 271-279.

68 Liu, Q., Islam, S., Rodriguez-Iturbe, I., and Le, Y., 1998. Phase-space analysis of daily streamflow: characterization and prediction. Advances in water resources 21, 463-475.

69 Lorenz, E. N., 1963. Deterministic Non-periodic Flow. Journal of Atmos. Sci., 20, 130 – 141.

70 Lorenz, E. N., 1991. Dimension of weather and climate attractors. Nature 353, 241-244.

71 Matterra, D. and Haykin, S., 1999. Support Vector Machines for Dynamic Reconstruction of a chaotic system. In: Advances in Kernel Methods, Ed. by Cholkopf, B., Burges, C. J., and Smola, A. J., pp 211 – 241. MIT Press.

72 Maybeck, P. S., 1979. Stochastic Models, Estimation and Control. Academic Press.

73 Mees, A. I., and Judd, K., 1993. Dangers of geometric filtering. Physica D, 68, 427 – 436.

74 Merwe, R. van der and Wan, E. A., 2001. Efficient derivative-free Kalman filters for online learning. Proceedings of the European Symposium on Artificial Neural Networks (ESANN), Burges, Belgium, April 2001.

75 Michalewics, Z., 1996. Genetic Algorithms + Data Structures + Evolution Programs 3rd Ed. Springer-VerlagBerlin Heidelberg.

76 Mitchell, T. M., 1997. Machine Learning. New York : McGraw-Hill

77 Nerenberg, M.A.H. and Essex, C., 1990. Correlation dimension and systematic geometric effects. Physical Review A, 42(12), 7065-7074.

78 Nicolis, C., and Nicolis, G., 1984. Is there a climatic attractor? Nature 311, 529-532

79 Nicolis, C., and Nicolis, G., 1987. Evidence for climatic attractors. Nature 326, 523.

80 Osborn, A. R., and Provenzale, A., 1989. Finite correlation dimension for stochastic systems with power-law spectra. Physica D 35, 357-381.

81 Packard, N. H., Crutchfield, J.P., Farmer, J.D., and Shaw, R. S., 1980. Geometry from a time series. Phys. Rev. Lett. 45(9), 712-716.

82 Palus, M., 1995. Testing for nonlinearity using redundancies: quantitative and qualitative aspects. *Physica D*, 80, 186-205.

83 Pasternack, G. B., 1999. Does the river run wild? Assessing chaos in hydrological systems. Adv Wat Resourc. 23, 253-260.

84 Phoon, K. K., Islam, M. N., Liaw, C. Y., and Liong, S. Y., 2002. Practical inverse approach for forecasting nonlinear hydrological time series. J of Hydrologic. Eng. (ASCE) 7(2), 116-128.

85 Pohlheim, H. Geatbx: Genetic and evolutionary algorithm toolbox for use with Matlab, http://www.geatbx.com/links/ea_matlab.html.

86 Porporato, A., and Ridolfi, L., 1996. Clues to the existence of deterministic chaos in river flow. International Journal of modern physics B 10(15), 1821-1862.

87 Porporato, A., and Ridolfi, L., 1997. Nonlinear analysis of river flow time sequences. Wat. Resourc. Res. 33(6), 1353-1367.

88 Prichard, D. and Theiler, J., 1995. Generalized redundancies for time series analysis. Physica D, 84, 476-493.

89 Provenzale, A., Smith, L. A., Vio, R., and Murante, G., (1992). Distinguishing between low-dimensional dynamics and randomness in measured time series, Physica D, 58, 31-49.

90 Rantala, J. and Koivisto, H., 2002. Optimized Subtractive Clustering for Neuro-Fuzzy models, 3rd WSES International Conference on Fuzzy Sets and Fuzzy Systems (FSFS' 02), Interlaken, Switzerland, February 11-15, 2002.

91 Ristic, B., Arulampalm, S., Gordon, N., 2004. Beyond the Kalman filter: particle filters for tracking applications. Boston, MA : Artech House

92 Rodriguez-Iturbe, I., De Power, B. F., Sharifi, M. B. and Georgakakos, K. P., 1989. Chaos in rainfall. Water Resources Research, 25(7), 1667-1775.

93 Sangoyomi, T. B., Lall, U., and Abarbanel, H. D. I., 1996. Nonlinear dynamics of the great salt lake: Dimension estimation. Water resources research 32(1), 149-159.

94 Sauer, T., 1992. A noise reduction method for signals from nonlinear systems. Physica D, 58, 193 – 201.

95 Scholkopf, B. and Smola, A., 2002. Learning with Kernels. MIT Press.

96 Schreiber, T. and Schmitz, A., 1996. Improved surrogate data for nonlinearity tests. *Physical Review Letters*, 77(4), 635-638.

97 Schreiber, T., 1993. Extremely simple nonlinear noise-reduction method. Physical Review E, 47 (4), 2401 – 2404.

98 Schreiber, T., and Grassberger, P., 1991. A simple noise-reduction method for real data, Physics letters A, 160(5), 411-418.

99 Schuster, H. G., 1988. Deterministic Chaos. VCH Weinheim, Germany.

100 Sharifi, M. B., Georgakakos, K. P. and Rodriguez-Iturbe, I., 1990. Evidence of deterministic chaos in the pulse of storm rainfall. Journal of the Atmospheric Sciences, 47(7), 888-893.

101 Sivakumar, B., 2002. A phase-space reconstruction approach to prediction of suspended sediment concentration in rivers. J. of. Hydrol. 258, 149-162.

102 Sivakumar, B., Jayawardena, A. W. and Fernando, T. M. K. G., 2002. River flow forecasting: use of phase-space reconstruction and artificial neural networks approaches. Journal of Hydrology, 265, 225-245.

103 Sivakumar, B., Liong, S. Y., and Liaw, C. Y., 1998. Evidence of chaotic behaviour in Singapore rainfall. Journal of American water resources association 34(2), 301-310.

104 Sivakumar, B., Liong, S. Y., Liaw, C. Y., and Phoon, K.K., 1999a. Singapore rainfall behaviour: Chaotic? J. of Hydrol. Eng. 4(1), 38-48.

105 Sivakumar, B., Phoon, K. K., Liong, S. Y., and Liaw, C. Y., 1999b. A systematic approach to noise reduction in chaotic hydrological time series. J. of Hydrol. 219, 103-135.

106 Sivakumar, B., Phoon, K. K., Liong, S. Y., and Liaw, C., 1999c. Comment on nonlinear analysis of river flow time series sequences by Amilcare Porporato and Luca Ridolfi. Water Resour. Res. 35 (3). 895 – 897.

107 Sivapragasam, C., 2003. Multi-objective evolutionary techniques in defining optimal policies for real time operation of reservoir systems. PhD thesis, Naitonal University of Singapore.

108 Sivapragasam, C., Liong, S. Y. and Pasha, M. F. K., 2001. Rainfall and runoff forecasting with SSA-SVM approach. Journal of Hydroinformatics, 3(2), 141 – 152.

109 Smith, L.A., 1998. Intrinsic limits on dimension calculations. Physics Letters A, 133(6), 283-288.

110 Sugihara, G., and May, R. M., 1990. Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series. Nature 344, 734-741.

111 Suykens, J. A. K., Van, G. T., De, B. J., De, M. B., Vandewalle, J., 2002. Least squares support vector machines. Singapore; River Edge, N.J.: World Scientific.

112 Takens, F., 1981. Detecting strange attractors in turbulence. In *Dynamical Systems and Turbulence, Lecture Notes in Mathematics*, 898, (Ed). D. A. Rand and L. S. Young, Springer-Verlag, Berlin, 366-381.

113 Termonia, Y. and Alexandrowicz, Z., 1983. Fractal dimension of strange attractors from radius versus size of arbitrary clusters. Physical Review Lettes 51(14), 1265-1268.

114 Theiler J., 1986. Spurious dimension from correlation algorithms applied to limited time-series data. Phys. Rev. A, 34(3), 2427-2432.

115 Theiler, J., 1987. Efficient algorithm for estimating the correlation dimension from a set of discrete points. Phys. Rev. A 36(9), 4456-4462.

116 Theiler, J., 1990. Statistical precision of dimension estimators. Phys. Rev. A 41(6), 3038-3050.

117 Tsonis, A. A. and Elsner, J. B., 1988. The weather attractor over very short timescales. Nature, 333, 545-547.

118 Tsonis, A. A., Elsner, J. B., and Georgakakos, K. P., 1993. Estimating the dimension of weather and climate attractors: Important issues about the procedure and interpretation. Journal of atmospheric sciences 50(15), 2549-2555.

119 Tsonis, A., A., and Elsner, J., B., 1992. Nonlinear prediction as a way of distinguishing chaos from random fractal sequences. Nature 358, 217-220

120 Vapnik, V. N., 1999. The nature of statistical learning theory. 2nd Ed. New York : Springer.

121 Walker, D. M. and Mees, A. I., 1997. Noise reduction of chaotic systems by Kalman Filtering and by Shadowing. International Journal of Bifurcation and Chaos, Vol. 7, No, 3, pp 769-779.

122 Wan, E. A and Merwe, R. van der, 2000. The unscented Kalman filter for nonlinear estimation. In proceedings of symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC), IEEE Press.

123 Welch, G. and Bishop, G. (2004). An introduction to the Kalman filter. University of North Carolina at Chapel Hill, Department of Computer Science, TR 95-041.

124 Williams, G. P., 1997. Chaos theory tamed. Taylor & Francis, London.

125 Wolf, A., Swift, J. B., Swinney, H. L., and Vastano, J. A., 1985. Determining Lyapunov exponents from a time series. Physica D, 16, 285-317.

126 Yager, R.R. and Filev, D. P., 1994. Generation of Fuzzy Rules by Mountain Clustering. Journal of Intelligence and Fuzzy Systems. Vol. 2, 209-219

127 Yager, R.R. and Filev, D. P., 1994. Approximate clustering via the mountain method. IEEE Transactions on Systems Man. and Cybernetics, Vol 24 (8), pp 1279-1284.

128 Yager, R.R. and Filev, D. P., 1994. Generation of fuzzy rules by mountain clustering. Journal of intelligent fuzzy systems, Vol. 2, pp 209-219.

129 Yu, X. Y., 2004. Support Vector Machie in chaotic hydrological time series forecasting. PhD Thesis, The National University of Singapore.

130 Yu, X. Y., Liong, S. Y. and Babovic, V., 2004. EC-SVM approach for real time hydrologic forecasting, Journal of Hydroinformatics, 6(3), 209 – 223.

131 Zaldivar, J.M., Gutierrez, E., Galvan, I.M., Stozzi, F., and Tomasin, A., 2000. Forecasting high waters at Venice lagoon using chaotic time series analysis and nonlinear neural networks. Journal of hydroinformatics 2, 61-84.

132 Zealand, C. M., Burn, D. H. and Simonovic, S. P., 1999. Short term streamflow forecasting using artificial neural networks. Journal of Hydrology, 214, 32-48.

133 Zhou, Y., Ma, Z., and Wang, L., 2002. Chaotic dynamics of the flood series in the Huaihe river basin for the last 500 years. J. of Hydrol. 258, 100-110.

134 Zoutendijk, G., 1970. Methods of feasible directions: a study in linear and nonlinear programming. Elsevier.

APPENDIX A

GRASSBERGER-PROCACCIA ALGORITHM FOR CORRELATION DIMENSION CALCULATION

This algorithm requires phase space reconstruction of a time series. As mentioned earlier, the method of delays (e.g. Takens, 1981) can be used to reconstruct the phase space. When the phase space vector is given, as shown in equation 2.1, the correlation integral C(r) for an *m*-dimensional state space is expressed by

$$C_r = \frac{2}{N(N-1)} \sum_{i,j} H\left(r - |X_i - X_j|\right) \qquad 1 \le i < j \le N \qquad \text{(A.1)}$$

where H is the Heaviside step function with $H(u)=1$ for $u>0$ and $H(u)=0$ for $u \le 0$ and r is the radius of a sphere centered on X_i or X_j; $|X_i-X_j|$ is the distance between the m dimensional delay vectors; and N is the number of data points. For a time series which is characterized by an attractor, for positive values of r

$$C_r \cong \alpha r^{\nu} \qquad \text{when } r \to 0 \;\; N \to \infty \qquad \text{(A.2)}$$

where α is a constant and ν is the correlation exponent or the slope of the Log C_r versus Log r plot given by

$$\nu = \lim_{\substack{r \to 0 \\ N \to \infty}} \frac{Log\, C(r)}{Log\, r} \qquad \text{(A.3)}$$

The correlation exponent values are plotted against the corresponding embedding dimensions. In the scaling region, the saturated correlation exponent ν is considered as the correlation dimension d of the attractor represented by the time series. A procedure to identify the scaling region is given by (Caputo et al., 1986). If the correlation

exponent leads to a finite value, then the system is thought to be governed by deterministic dynamics. In addition, if the value is small and non-integer, the system is considered to be governed by low dimensional chaos. If the correlation exponent increases, without bound, with the increase of embedding dimension, then the system is considered to be stochastic (e.g. Osborne and Provenzale, 1989).

APPENDIX B

SUMMARY OF CHAOS ANALYSIS PREDICTION SCHEME USED IN THE STUDY

As explained in chapters 2 and 3, the major steps in chaos analysis are as follows.

1. Identification
2. Phase space parameter (m, τ) determination
3. Prediction

(1) Identification

Fourier analysis and correlation dimension method are used for the identification of chaos in the time series data.

(2) Phase space parameter (m, τ) determination

Determination of appropriate (m, τ) values is important to obtain good prediction accuracy. The steps are;

1. Divide data in to 3 separate sets: training, test and validation sets (see Figure B.1)
2. For a certain set of parameter values, and a set of prediction parameters, train a prediction model using the training set and use that model to predict the test set and note the prediction accuracy.
3. Select the trained model and the corresponding parameters that give the best prediction accuracy on test set as the optimal parameters and the model.

(3) Use the optimal model and the parameters determined in the previous step to predict the unseen data, the validation set.

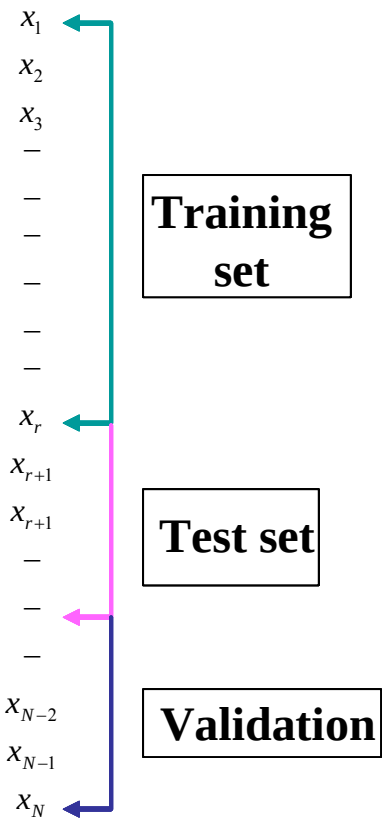


Figure B.1 Division of data sets into training, test and validation sets

APPENDIX C

OPTIMAL PHASE SPACE PARAMETERS FOR NOISE-FREE CHAOTIC LORENZ SERIES, MISSISSIPPI RIVER AND WABASH RIVER FLOW TIME SERIES

The following optimal parameters have been observed in a study (Karunasinghe, 2003) that used an inverse approach similar to the one used in Liong et al. (2005) where micro Genetic Algorithm was used to find the optimal phase space parameters. The range for *m* values was from 1 – 10 while the ranges for both τ and k were 1 – 100. A local averaging technique was used as the prediction tool.

Table C.1 Optimal phase space parameter sets for Lorenz series, Mississippi river and Wabash river flow time series

Series	Parameter set (m, τ, k)		
	Lead Time 1	Lead Time 3	Lead Time 5
Lorenz	(2, 4, 9)	(3, 6, 9)	(6, 3, 6)
Mississippi River	(2, 1, 5)	(2, 1, 9)	(2, 1, 8)
Wabash River	(2, 6, 16)*	(3, 1, 21)	(2, 1, 25)

* It was noticed in this particular case the search has failed to reach a satisfactory solution. Investigations with different seeds revealed optimal parameters 2, 1 (for m, τ) with much better prediction accuracies. An example prediction performance on a validation set with the two sets of parameters are given in Table C.2

Table C.2 Prediction errors on validation set for different (m, τ): Wabash River flow with lead time 1 prediction

Parameter set (m, τ, k)	(2, 6, 16)	(2, 1, 10)
Normalized root mean square error	0.1163	0.0850
Mean absolute error (m^3/s)	51.11	34.08

Conclusion: Low values of τ (<10) produced best prediction accuracies for Lorenz series and a time delay of 1 day yielded best predictions on the two river flow time series (Mississippi and Wabash Rivers).

APPENDIX D

PREDICTION PERFORMANCE OF VARIOUS PREDICTION MODELS ON TEST SETS

Table D.1 Prediction errors with various models on test set: Noise-free Lorenz series

Lead time	Local model (Averaging)		Local model (Polynomial)		ANN	
	NRMSE	MAE	NRMSE	MAE	NRMSE	MAE
1	0.01576	0.1295	0.00032	0.0033	0.00030	0.0031
3	0.02017	0.1790	0.00041	0.004	0.00035	0.0035
5	0.02449	0.2306	0.00053	0.0051	0.00041	0.0039

Table D.2 Prediction errors with various models on test set: 5% Noisy Lorenz series

Lead time	Local model (Averaging)		Local model (Polynomial)			ANN	
	NRMSE	MAE	Polynomial Order	NRMSE	MAE	NRMSE	MAE
1	0.0632	0.6273	1	0.0605	0.5956	0.0581	0.5841
3	0.0653	0.6500	1	0.0623	0.6237	0.0606	0.6022
5	0.0701	0.6881	1	0.0676	0.6506	0.0636	0.6235

Table D.3 Prediction errors with various models on test set: 30% Noisy Lorenz series

Lead time	Local averaging		Local Polynomial			ANN	
	NRMSE	MAE	Polynomial Order	NRMSE	MAE	NRMSE	MAE
1	0.3447	3.5834	1	0.3524	3.5760	0.3452	3.5358
3	0.3664	3.8657	1	0.3726	3.8754	0.3663	3.8675
5	0.3880	4.0279	1	0.4210	4.3013	0.3822	3.9992

Table D.4 Prediction errors with various models on test set: Mississippi river flow

Lead time	Local model (Averaging)		Local model (Polynomial)		ANN	
	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)
1	0.0357	246.46	0.0324	209.73	0.0298	202.97
3	0.1099	835.24	0.1042	795.51	0.1003	758.32
5	0.1947	1515.58	0.1901	1477.52	0.1801	1408.77

Table D.5 Prediction errors with various models on test set: Wabash river flow

Lead time	Local model (Averaging)		Local model (Polynomial)		ANN	
	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)
1	0.0842	35.72	0.0686	28.86	0.0649	26.66
3	0.2835	129.08	0.2724	126.32	0.2508	112.97
5	0.4485	212.79	0.4493	213.19	0.4153	197.66

Table D.6 Prediction errors of SVM on test sets: Noise-free, 5% noisy, and 30% noisy Lorenz series

Lead time	Noise-free		5% noisy		30% noisy	
	NRMSE	MAE	NRMSE	MAE	NRMSE	MAE
1	0.00042	0.0038	0.0580	0.5835	0.3385	3.4678
3	0.00065	0.0049	0.0631	0.6278	0.3637	3.8324
5	0.00093	0.0070	0.0648	0.6328	0.3774	3.9141

Table D.7 Prediction errors of SVM on test sets: Mississippi and Wabash flow time series

Lead time	Mississippi		Wabash	
	NRMSE	MAE (m^3/s)	NRMSE	MAE (m^3/s)
1	0.0305	195.42	0.0653	25.78
3	0.0995	743.90	0.2552	110.62
5	0.1841	1388.84	0.4267	187.15

APPENDIX E

PREDICTION PERFORMANCE OF FIRST AND THIRD ORDER POLYNOMIAL MODELS

The Table E.1 shows the prediction performance of first, second and third order polynomial models for the Mississippi river flow series.

Table E.1 Prediction errors with first, second and third order polynomial models on validation set: Mississippi river flow

Lead time	1st order polynomial		2nd order polynomial		3rd order polynomial	
	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)	NRMSE (/)	MAE (m^3/s)
1	0.0422	224.25	0.0412	225.13	0.0411	224.64
3	0.1373	812.82	0.1371	810.59	0.1360	807.51
5	0.2513	1532.18	0.2476	1512.50	0.2473	1511.08

APPENDIX F

PERFORMANCE OF PREDICTION MODELS TRAINED WITH DATA OF NOISE LEVELS DIFFERENT FROM THAT OF VALIDATION INPUT DATA

F.1 PERFORMANCE OF MODELS TRAINED WITH LESS-NOISY DATA WITH NOISY VALIDATION INPUTS

Table F.1 shows the prediction performance of models, trained with Lorenz series data of 1%, 10%, 20% and 30% noise levels, validated on input data of equal noise levels. Tables F.2, F.3, and F.4 show the prediction performance of models, trained respectively with data of 1%, 10% and 20% noise levels, validated however on input data of higher noise levels. Comparisons of prediction accuracies listed in Tables F.2, F.3, and F.4 with corresponding entries in Table F.1 show that models trained with lower noise level (e.g. 1%, Table F.2) do not necessarily yield higher prediction accuracies (e.g. MAE= 4.2926, Column D, Table F.2), when validated on input data of higher noise level (e.g. 30%, Column D, Table F.2), than models trained with higher noise levels (e.g. 30%, Table F.1) and, at the same time, validated on input data of the same noise level (MAE=4.0831, 30% noise level, Column D, Table F.1).

Table F.1 Prediction performance of ANN models trained with data of known noise levels and validated on input data of the same noise levels: Lorenz series

Prediction error (MAE)	Noise level for model's training data and validation input data			
	1% (A)	10% (B)	20% (C)	30% (D)
Against noisy data	0.1279	1.2015	2.7378	4.0831
Against noise-free data	0.0725	0.6906	1.5433	2.2602

Table F.2 Prediction performance of ANN model trained with 1% noise level data and validated with input data of other noise levels

Prediction error (MAE)	Noise level of validation input data		
	10% (B)	20% (C)	30% (D)
Against noisy data	1.2348	2.7223	4.2926
Against noise-free data	0.7484	1.5888	2.5826

Table F.3 Prediction performance of ANN model trained with 10% noise level data and validated with input data of other noise levels

Prediction error (MAE)	Noise level of Validation data	
	20% (C)	30% (D)
Against noisy data	2.7183	4.0035
Against noise-free data	1.4860	2.2679

Table F.4 Prediction performance of ANN model trained with 20% noisy data when 30% noisy validation data are used as inputs

Prediction error (MAE)	Noise level of validation input data
	30% (D)
Against noisy data	4.0915
Against noise-free data	2.3044

F.2 PERFORMANCE OF MODELS TRAINED WITH NOISY DATA WITH LESS-NOISY VALIDATION INPUTS

Tables F.5, F.6, and F.7 show the prediction performance of models, trained respectively with data of 20%, 10% and 1% noise levels, validated however on input data of lower noise levels. In Tables F.5, F.6, and F.7, the last column (e.g. column (4) of Table F.5) gives the prediction performance of model trained with data of same noise level as that of input data (e.g. 20%). Comparing this prediction performance with those of earlier columns (e.g. columns (1), (2) and (3) of Table F.5), i.e. the performance of the model with less noisy validation inputs, show that the smaller the level of noise in validation inputs the better are the predictions. In other words, the less noisy input data improves the prediction performance of the models trained with more noisy data. It should be noted that prediction errors are determined by comparing the predictions against noisy validation outputs whose level of noise is same as that of the data used to train the model. This is because, in real applications, one has to use the original data to compare the predictions instead of noise-reduced data as explained in Chapter 2.

Table F.5 Prediction performance of ANN model trained with 20% noise level data and validated with input data of less noise levels

Prediction error (MAE)	Noise level of validation input data			
	Noise-free (1)	1% (2)	10% (3)	20% (4)
Against 20% noisy data	2.2915	2.2950	2.3684	2.7378
Against noise-free data	0.5173	0.5313	0.8866	1.5433

Table F.6 Prediction performance of ANN model trained with 10% noise level data and validated with input data of less noise levels

Prediction error (MAE)	Noise level of validation input data		
	Noise-free (1)	1% (2)	10% (3)
Against 10% noisy data	1.0364	1.0350	1.2015
Against noise-free data	0.2182	0.2273	0.6906

Table F.7 Prediction performance of ANN model trained with 1% noise level data and validated with input data of less noise levels

Prediction error (MAE)	Noise level of validation input data	
	Noise-free (1)	1% (2)
Against 1% noisy data	0.1082	0.1279
Against noise-free data	0.0233	0.0725

APPENDIX G

FINDING *A POSTERIORI* STATE ESTIMATE $\hat{x}_k$ AS A LINEAR COMBINATION OF AN *A PRIORI* ESTIMATE $\hat{x}_k^-$ AND NEW MEASUREMENT z_k

Following Haykin (2001), the derivation of *a posteriori* estimate $\hat{\boldsymbol{x}}_k$ as a linear combination of an *a priori* estimate $\hat{\boldsymbol{x}}_k^-$ and new measurement $\boldsymbol{z}_k$ may be given as follows. The following theorem will be used in the derivation.

Theorem G.1: Principle of orthogonality

Let the stochastic processes $\{x_k\}$ and $\{y_k\}$ be of zero means, that is

$E[x_k] = E[y_k] = 0$ for all k.

Then:

(i) the stochastic process $\{x_k\}$ and $\{y_k\}$ are jointly Gaussian; or

(ii) if the optimal estimate $\hat{x}_k$ is restricted to be a linear function of the observables and the cost function is the mean square error;

(iii) then the optimum estimate $\hat{x}_k$, given the observables $y_1, y_2, \ldots y_k$ is the orthogonal projection of x_k on the space spanned by these observables.

When a measurement on a linear dynamical system, described by Eqs. G.1 and G.2, has been made at time k, the requirement is to use the information contained in the new measurement $\boldsymbol{z}_k$ to update the estimate of the unknown state $\boldsymbol{x}_k$.

$$\boldsymbol{x}_k = \boldsymbol{A}_k \boldsymbol{x}_{k-1} + \boldsymbol{B}_k \boldsymbol{u}_k + \boldsymbol{w}_{k-1} \tag{G.1}$$

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \upsilon_k \tag{G.2}$$

Let $\hat{\boldsymbol{x}}_k^-$ denote *a priori* estimate of the state, which is already available at time *k*. With a linear estimator as the objective, one may express the *a posteriori* estimate $\hat{\boldsymbol{x}}_k$ as a linear combination of the *a priori* estimate and the new measurement, as shown by,

$$\hat{\boldsymbol{x}}_k = \boldsymbol{G}_k^1 \boldsymbol{x}_k + \boldsymbol{G}_k \boldsymbol{z}_k \tag{G.3}$$

where factors $\boldsymbol{G}_k^1$ and $\boldsymbol{G}_k$ are to be determined. Let the *state-error vector* be,

$$\tilde{\boldsymbol{x}}_k = \boldsymbol{x}_k - \hat{\boldsymbol{x}}_k \,. \tag{G.4}$$

Applying the principle of orthogonality (Theorem G.1),

$$E\left[\tilde{\boldsymbol{x}}_k \boldsymbol{z}_i^T\right] = \boldsymbol{0} \quad \text{for } i = 1, 2, \ldots, k-1 \,. \tag{G.5}$$

using Eqs. G.2, G.3, and G.4 in G.5, we get

$$E\left[(\boldsymbol{x}_k - \boldsymbol{G}_k^1 \hat{\boldsymbol{x}}_k^- - \boldsymbol{G}_k \boldsymbol{H}_k \boldsymbol{x}_k - \boldsymbol{G}_k \boldsymbol{w}_k)\boldsymbol{z}_i^T\right] = \boldsymbol{0} \text{ for } i = 1, 2, \ldots, k-1 \,. \tag{G.6}$$

Since the process noise $\boldsymbol{w}_k$ and the measurement noise υ_k are uncorrelated, it follows that

$$E\left[\boldsymbol{w}_k \boldsymbol{z}_i^T\right] = \boldsymbol{0} \,. \tag{G.7}$$

Using Eq. G.7 in Eq. G.6 and rearranging the terms give

$$E\left[(\boldsymbol{I} - \boldsymbol{G}_k \boldsymbol{H}_k - \boldsymbol{G}_k^1)\boldsymbol{x}_k \boldsymbol{z}_i^T + \boldsymbol{G}_k^1 (\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-)\boldsymbol{z}_i^T\right] = \boldsymbol{0} \tag{G.8}$$

where $\boldsymbol{I}$ is the identity matrix. From the principle of orthogonality, we now note that

$$E\left[(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-)\boldsymbol{z}_i^T\right] = \boldsymbol{0} \,. \tag{G.9}$$

With Eq. G.9 in Eq. G.8 it yields

$$(\boldsymbol{I} - \boldsymbol{G}_k \boldsymbol{H}_k - \boldsymbol{G}_k^1) E\left[\boldsymbol{x}_k \boldsymbol{z}_i^T\right] = \boldsymbol{0} \quad \text{for } i = 1, 2, \ldots, k-1 \,. \tag{G.10}$$

For arbitrary values of the state $\boldsymbol{x}_k$ and the observable $\boldsymbol{z}_i$, Eq. G.10 can only be satisfied if the scaling factors $\boldsymbol{G}_k^1$ and $\boldsymbol{G}_k$ are related as follows:

$(\boldsymbol{I} - \boldsymbol{G}_k \boldsymbol{H}_k - \boldsymbol{G}_k^1) = \boldsymbol{0}$ or, $\boldsymbol{G}_k^1$ is defined in terms of $\boldsymbol{G}_k$ as

$$\boldsymbol{G}_k^1 = \boldsymbol{I} - \boldsymbol{G}_k \boldsymbol{H}_k \tag{G.11}$$

Substituting Eq. G.11 in Eq. G.3, we may express the *a posteriori* estimate of the state at time *k* as

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{G}_k\left(\boldsymbol{z}_k - \boldsymbol{H}_k \hat{\boldsymbol{x}}_k^-\right) \tag{G.12}$$

where the matrix $\boldsymbol{G}_k$ is called the *Kalman gain*.

APPENDIX H

PREDICTION PERFORMANCE OF NOISE REDUCTION APPLICATIONS ON NOISES GENERATED FROM DIFFERENT SEEDS

Table H.1 Prediction performance of ANN on noisy chaotic Lorenz time series: with noises generated from different seeds

Data set generated from different seeds	Noise Level (%)	Optimal Phase space parameters (m, τ) (1)	Prediction error (*MAE*)	
			Against noisy data (2)	Against noise-free data (3)
Seed No.1	1	(10, 6)	0.1364	0.0845
	10	(10, 1)	1.2156	0.7012
	20	(10, 3)	2.4805	1.3787
	30	(10, 1)	3.6261	1.8633
Seed No.2	1	(10, 3)	0.1197	0.0727
	10	(10, 3)	1.2127	0.7008
	20	(10, 1)	2.3473	1.2222
	30	(10, 3)	3.6854	2.0093
Seed No.3	1	(10, 3)	0.1203	0.0729
	10	(10, 3)	1.1592	0.6856
	20	(10, 6)	2.7757	1.6174
	30	(9, 1)	3.6462	2.0382

Table H.2 Prediction performance of EKF predictor on noisy chaotic Lorenz time series: with noises generated from different seeds

Data set generated from different seeds	Noise Level (%)	prediction error against noisy data (*MAE*) (1)	prediction error against noise-free data (*MAE*) (2)	Percentage improvement in prediction accuracy compared to ANN alone	
				With respect to noisy data (3)	With respect to noise-free data (4)
Seed No.1	1	0.1226	0.0648	10.1	23.3
	10	1.1648	0.6602	4.2	5.8
	20	2.3451	1.1789	5.5	14.5
	30	3.4700	1.5707	4.3	15.7
Seed No.2	1	0.1121	0.0579	6.4	20.4
	10	1.1456	0.5600	5.5	20.1
	20	2.2506	1.0314	4.1	15.6
	30	3.4174	1.6583	7.3	17.5
Seed No.3	1	0.1121	0.0568	6.9	22.1
	10	1.1279	0.6123	2.7	10.7
	20	2.5130	1.1649	9.5	28.0
	30	3.4531	1.6970	5.3	16.7

Table H.3 Prediction performance of EKF estimates on proposed procedure: noisy chaotic Lorenz time series with ANN: with noises generated from different seeds

Data set generated from different seeds	Noise Level (%)	prediction error against noisy data (*MAE*) (1)	prediction error against noise-free data (*MAE*) (2)	Percentage improvement in prediction accuracy compared to ANN alone	
				With respect to noisy data (3)	With respect to noise-free data (4)
Seed No.1	1	0.1202	0.0596	11.9	29.5
	10	1.1472	0.6411	5.6	8.6
	20	2.2201	0.9580	10.5	30.5
	30	3.4207	1.5181	5.7	18.5
Seed No.2	1	0.1103	0.0501	7.9	31.1
	10	1.1225	0.4978	7.4	29.0
	20	2.2416	0.9875	4.5	19.2
	30	3.3148	1.4086	10.1	29.9
Seed No.3	1	0.1095	0.0515	9.0	29.4
	10	1.1008	0.5405	5.0	21.2
	20	2.4267	1.0340	12.6	36.1
	30	3.4362	1.6615	5.8	18.5

Table H.4 Prediction performance of nonlinear noise reduction on the proposed procedure: noisy chaotic Lorenz time series with ANN: with noises generated from different seeds

Data set generated from different seeds	Noise Level (%)	prediction error against noisy data (*MAE*) (1)	prediction error against noise-free data (*MAE*) (2)	Percentage improvement in prediction accuracy compared to ANN alone	
				With respect to noisy data (3)	With respect to noise-free data (4)
Seed No.1	1	0.1364	0.0845	0	0
	10	1.1751	0.6842	3	2
	20	2.3628	1.2981	5	6
	30	3.5534	1.8414	2	1
Seed No.2	1	0.1208	0.0738	-1	-2
	10	1.2073	0.6954	0	1
	20	2.3178	1.1698	1	4
	30	3.6630	1.9655	1	2
Seed No.3	1	0.1203	0.0729	0	0
	10	1.2085	0.6985	-4	-2
	20	2.7461	1.6026	1	1
	30	0.1899	1.9105	2	6

APPENDIX I

LORENZ SERIES IN THE APLICATION OF NOISE REDUCTION

It is interesting to notice how the data changes and the attractor change in the process of noise removal. In this Appendix, the statistics of noise reduction and some graphical representations showing (1) the plots of noise-free, noisy and EKF noise reduced data (section 4.5); (2) the attractor in noise reduction; and (3) the plots of actual and predicted data with and without noise reduction, are given. The illustrations are provided for the case of 10% noisy data used in Chapter 4.

I.1 Statistics and plots showing noise reduction

Table I.1 shows the standard deviations of noisy series and the EKF noise reduced series and the corresponding prediction improvement gained using the proposed noise reduction scheme. Figure I.1 shows the plots of Lorenz validation data for the noise-free, 10% noisy data used in the study and the EKF noise reduced data using the procedure proposed in the study (Section 4.5). The effectiveness in noise reduction is evident from Table I.1 as well as Figure I.1.

Table I.1 Noise reduction – statistics

Noisy Series	Standard deviation		% Performance	
	Noise added	Remaining noise after EKF noise reduction	% noise reduction	% prediction improvement
1 %	0.1268	0.0601	53	28
10 %	1.2684	0.6424	49	22
20 %	2.5368	1.5491	39	15
30 %	3.8052	2.3863	37	18

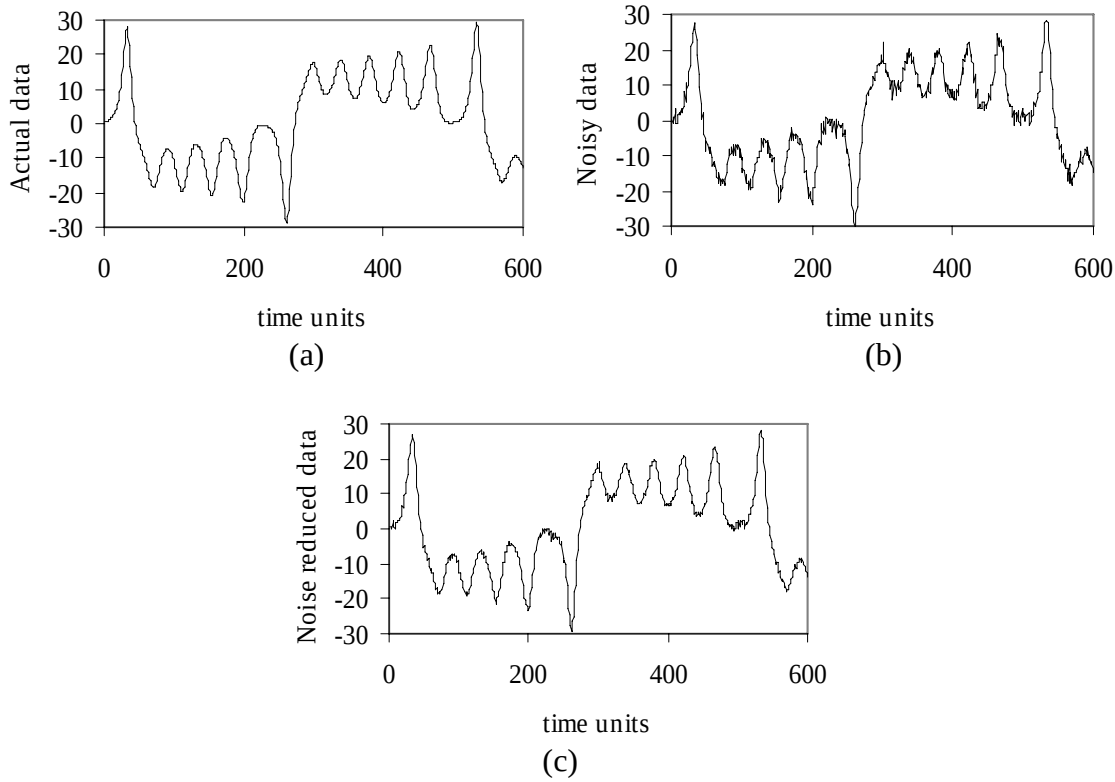
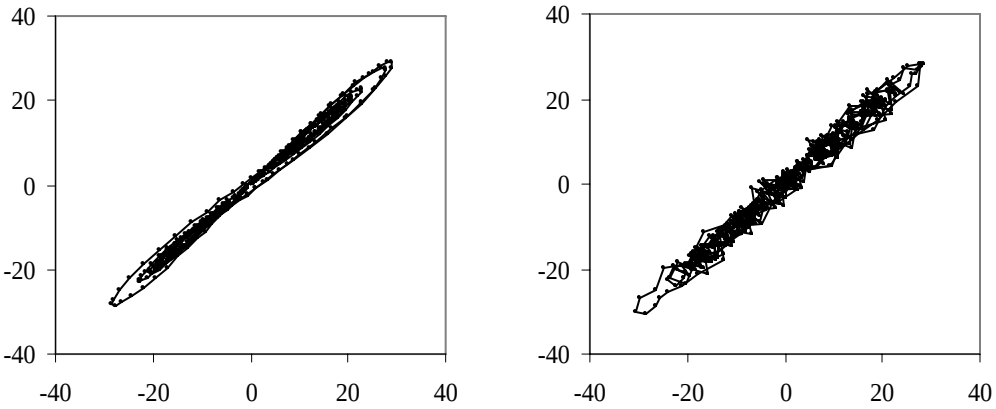


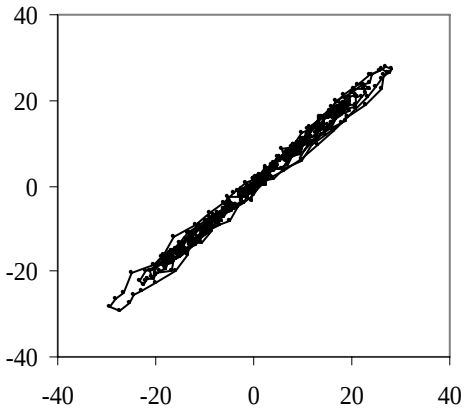
Figure I.1 10% noisy Lorenz series validation data (a) Noise free data (b) noisy data and (c) EKF noise reduced data

I.2 Lorenz attractor in noise reduction

The Lorenz attractor is shown here using noise-free, 10% noisy and EKF noise reduced data. The validation data is used. The attractor is first shown with time delay 1 (Figure I.2) and then for better clarity is shown with a delay time of 6 (Figure I.3). Figures show that the attractor gets closer to the actual one when noise is reduced.



(a) (b)



(c)

Figure I.2 The Lorenz attractor for (a) noise-free, (b) 10% noisy data and (c) EKF noise-reduced data with delay time of 1

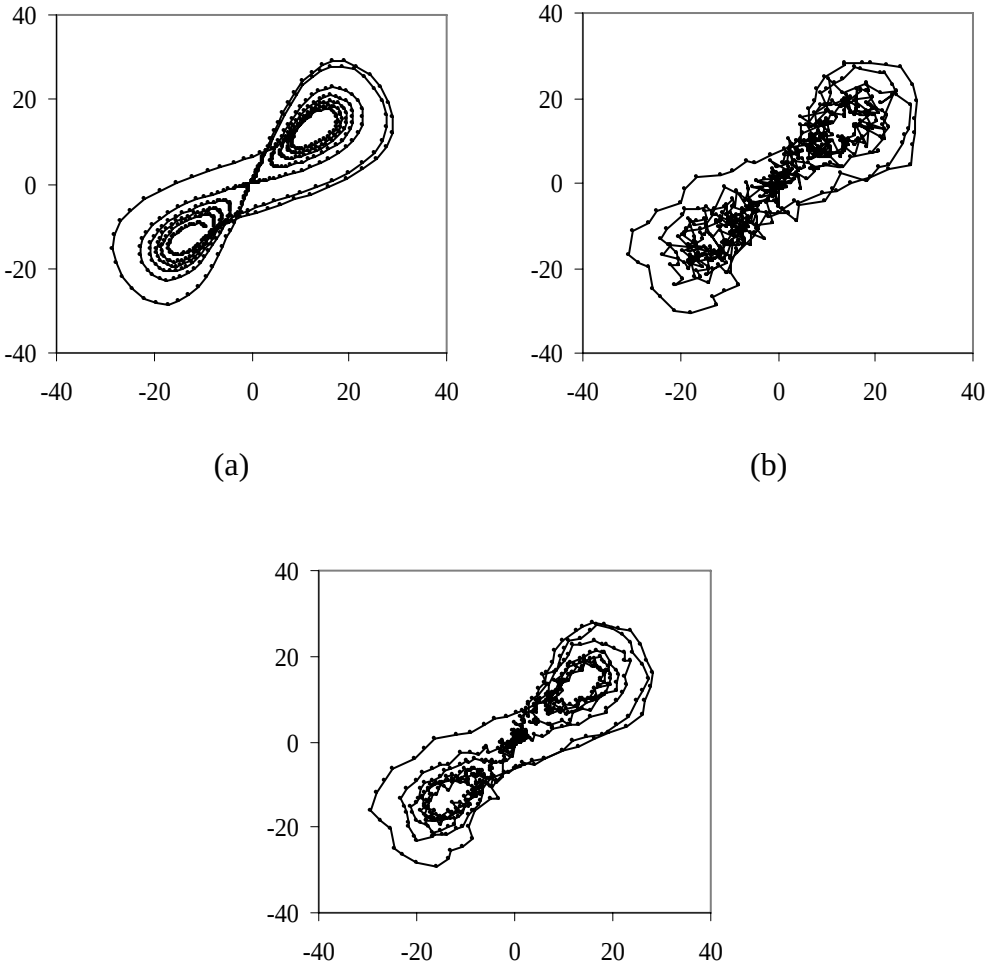
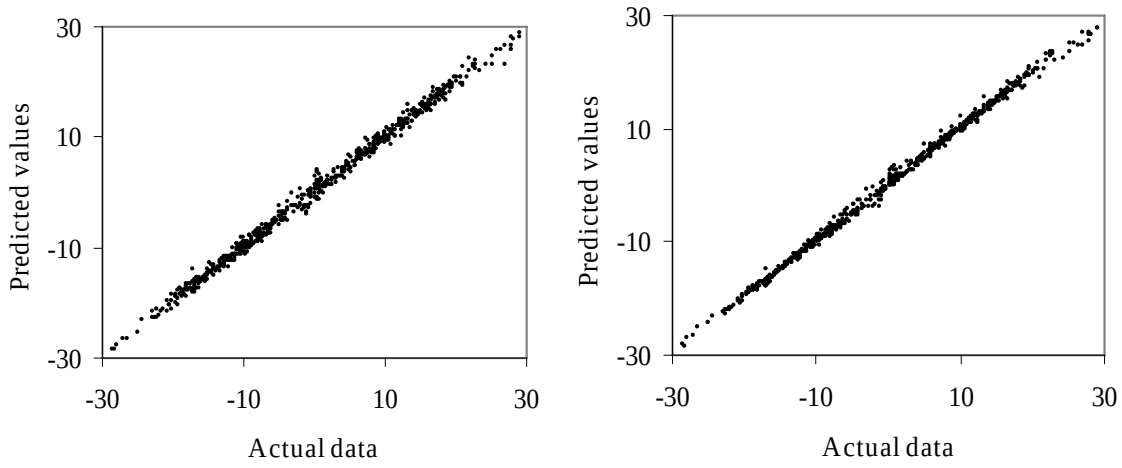


Figure I.3 The Lorenz attractor for (a) noise-free, (b) 10% noisy data and (c) EKF noise-reduced data with delay time of 6

I.2 Plots of actual and predicted data with and without noise reduction

Figure I.4 shows the scatter plots of actual data and the predicted data for the cases of using noisy data for prediction and prediction using noise reduction scheme proposed in Section 4.5. The more closer fit in the case of noise reduction clearly shows the proposed noise reduction procedure has improved the prediction performance.



(a) Prediction without noise reduction (b) With proposed noise reduction

Figure I.4 Prediction performance with and without noise reduction

APPENDIX J

PERFORMANCE OF PROPOSED NOISE REDUCTION SCHEME WITH SVM AS THE PREDICTION TOOL

A risk in using the prediction accuracy of a certain model as a criterion to determine noise reduction is that the removed noise may be biased by the prediction model. In other words, what has been identified as noise by that particular model may not be noise to other models. Such doubts were raised by Elshorbagy et al (2002). In the present study, the optimal noise reduction is identified by the prediction error of ANN prediction models on the test set. In addition, the state space model in EKF also consists of an ANN model. To verify the performance of these optimally noise reduced data on a different model, the prediction models (Fig. 4.8) were trained with SVM using those noise reduced data.

Table J.1 shows the prediction performance when SVM prediction models are trained with a noise reduced data that have been identified as optimal solutions with ANN prediction model (in Figure 4.8). Comparison of columns 3 and 4 of Tables 4.4 and Table J.1 shows that the amounts of percentage improvements are approximately of the same order of magnitude. Similar results are observed on river flow time series (Table J.2) as well. This implies that two different prediction models (ANN and SVM) have recognized the amount of noise reduction with equal effectiveness. Therefore, the noise reduced using ANN in the EKF state space model can be considered not biased by the ANN model.

Table J.1 Prediction performance of EKF estimates on the proposed procedure: noisy chaotic Lorenz series with SVM

Noise Level (%)	prediction error against noisy data (*MAE*)	prediction error against noise-free data (*MAE*)	Percentage improvement in prediction accuracy compared to ANN alone	
			With respect to noisy data	With respect to noise-free data
	(1)	(2)	(3)	(4)
1	0.1168	0.0470	9	35
10	1.1264	0.5095	6	26
20	2.5143	1.0517	8	32
30	3.7587	1.6745	8	26

Table J.2 Prediction performance of EKF estimates on proposed procedure: river flow time series with SVM

Time series	prediction error with respect to noisy data (*MAE*) (m^3/s)	Percentage improvement in prediction accuracy compared to SVM alone (With respect noisy data)
Mississippi River	204.88	1.0
Wabash River	24.95	0.9

APPENDIX K

NUMBER OF PATTERNS EXTRACTED AND THE CORRESPONDING PREDICTION ERRORS WITH DIFFERENT *d* VALUES

Table K.1 The *d* values and the corresponding number of patterns selected and the prediction errors on validation set using for local model and ANN: Noise free Lorenz series

d	number of patterns	Local averaging		ANN	
		NRMSE	MAE	NRMSE	MAE
0.001	4666	0.03973	0.3096	0.0003	0.0032
0.002	4604	0.03968	0.3067	0.0003	0.0032
0.003	4561	0.03950	0.3032	0.0003	0.0033
0.004	4530	0.03949	0.3035	0.0003	0.0033
0.005	4517	0.03911	0.3005	0.0003	0.0033
0.006	4497	0.03909	0.3000	0.0003	0.0032
0.007	4412	0.03856	0.2956	0.0003	0.0032
0.008	4211	0.03871	0.3002	0.0003	0.0032
0.009	3985	0.03938	0.3064	0.0003	0.0034
0.01	3763	0.04202	0.3053	0.0003	0.0033
0.011	3586	0.04246	0.3116	0.0003	0.0034
0.012	3454	0.04205	0.3127	0.0003	0.0032
0.013	3317	0.04071	0.3105	0.0003	0.0032
0.014	3144	0.04390	0.3225	0.0003	0.0033
0.015	3027	0.04372	0.3179	0.0003	0.0033
0.016	2895	0.04436	0.3269	0.0003	0.0033
0.017	2761	0.04418	0.3276	0.0003	0.0036
0.018	2616	0.04495	0.3251	0.0003	0.0033
0.019	2460	0.04515	0.3300	0.0003	0.0035
0.02	2342	0.04474	0.3264	0.0003	0.0033
0.021	2228	0.04339	0.3290	0.0003	0.0034
0.022	2112	0.04373	0.3414	0.0003	0.0035
0.023	2009	0.04457	0.3493	0.0003	0.0035
0.024	1917	0.04272	0.3471	0.0003	0.0035
0.025	1855	0.04314	0.3509	0.0003	0.0036
0.026	1794	0.04291	0.3512	0.0003	0.0034
0.027	1747	0.04375	0.3575	0.0003	0.0034
0.028	1688	0.04746	0.3778	0.0004	0.0037
0.029	1625	0.04832	0.3853	0.0004	0.0036
0.03	1581	0.04735	0.3966	0.0003	0.0035
0.031	1528	0.04944	0.4101	0.0004	0.0036
0.032	1485	0.04930	0.4053	0.0004	0.0036
0.033	1434	0.05076	0.4209	0.0004	0.0037
0.034	1383	0.05177	0.4414	0.0004	0.0036
0.035	1347	0.05236	0.4433	0.0004	0.0036

Table K.1 (Continued)

0.036	1308	0.05364	0.4615	0.0004	0.0038
0.037	1277	0.05279	0.4487	0.0004	0.0036
0.038	1237	0.05296	0.4599	0.0004	0.0038
0.039	1176	0.05283	0.4610	0.0004	0.0038
0.04	1139	0.05156	0.4541	0.0004	0.0039
0.041	1104	0.05262	0.4648	0.0004	0.0038
0.042	1074	0.05426	0.4763	0.0004	0.0036
0.043	1050	0.05467	0.4760	0.0004	0.0036
0.044	1015	0.05495	0.4844	0.0004	0.0039
0.045	987	0.05572	0.4853	0.0004	0.0037
0.046	948	0.05590	0.4895	0.0004	0.0039
0.047	918	0.05558	0.4990	0.0004	0.0036
0.048	890	0.06910	0.6482	0.0004	0.0039
0.049	869	0.07136	0.6713	0.0004	0.0037
0.05	839	0.06970	0.6492	0.0004	0.0038

Table K.2 The d values and the corresponding number of patterns selected and the prediction errors on validation set using for local model and ANN: 5% noisy Lorenz series

d	number of patterns	Local averaging		ANN	
		NRMSE	MAE	NRMSE	MAE
0.001	4772	0.0728	0.7418	0.0634	0.6395
0.002	4772	0.0728	0.7418	0.0634	0.6395
0.003	4772	0.0728	0.7418	0.0634	0.6395
0.004	4772	0.0728	0.7418	0.0634	0.6395
0.005	4772	0.0728	0.7418	0.0634	0.6395
0.006	4772	0.0728	0.7418	0.0634	0.6395
0.007	4772	0.0728	0.7418	0.0634	0.6395
0.008	4772	0.0728	0.7418	0.0634	0.6395
0.009	4772	0.0728	0.7418	0.0634	0.6395
0.01	4772	0.0728	0.7418	0.0634	0.6395
0.011	4772	0.0728	0.7418	0.0634	0.6395
0.012	4772	0.0728	0.7418	0.0634	0.6395
0.013	4772	0.0728	0.7418	0.0634	0.6395
0.014	4772	0.0728	0.7418	0.0634	0.6395
0.015	4772	0.0728	0.7418	0.0634	0.6395
0.016	4772	0.0728	0.7418	0.0634	0.6395
0.017	4770	0.0728	0.7418	0.0620	0.6258
0.018	4768	0.0728	0.7418	0.0632	0.6400
0.019	4766	0.0728	0.7418	0.0632	0.6399
0.02	4760	0.0728	0.7420	0.0633	0.6404

Table K.2 (Continued)

0.021	4746	0.0729	0.7431	0.0624	0.6289
0.022	4725	0.0729	0.7432	0.0626	0.6334
0.023	4711	0.0728	0.7421	0.0624	0.6291
0.024	4679	0.0728	0.7446	0.0624	0.6289
0.025	4641	0.0729	0.7452	0.0624	0.6291
0.026	4602	0.0730	0.7471	0.0621	0.6243
0.027	4560	0.0728	0.7449	0.0621	0.6265
0.028	4481	0.0731	0.7476	0.0625	0.6308
0.029	4395	0.0736	0.7482	0.0619	0.6226
0.03	4308	0.0737	0.7476	0.0631	0.6335
0.031	4233	0.0737	0.7478	0.0629	0.6337
0.032	4128	0.0736	0.7471	0.0644	0.6483
0.033	4012	0.0735	0.7480	0.0637	0.6445
0.034	3882	0.0738	0.7518	0.0629	0.6347
0.035	3737	0.0733	0.7499	0.0625	0.6326
0.036	3605	0.0728	0.7442	0.0625	0.6288
0.037	3463	0.0723	0.7351	0.0633	0.6371
0.038	3294	0.0724	0.7352	0.0628	0.6305
0.039	3145	0.0732	0.7500	0.0638	0.6389
0.04	2997	0.0728	0.7428	0.0631	0.6372
0.041	2840	0.0728	0.7458	0.0631	0.6348
0.042	2691	0.0721	0.7289	0.0637	0.6402
0.043	2542	0.0729	0.7331	0.0631	0.6347
0.044	2384	0.0720	0.7196	0.0632	0.6415
0.045	2269	0.0729	0.7325	0.0631	0.6407
0.046	2122	0.0732	0.7343	0.0636	0.6427
0.047	1990	0.0756	0.7601	0.0627	0.633
0.048	1864	0.0759	0.7679	0.0645	0.6515
0.049	1773	0.0744	0.7468	0.0645	0.6525
0.05	1668	0.0781	0.7801	0.0648	0.6553

Table K.3 The d values and the corresponding number of patterns selected and the prediction errors on validation set using for local model and ANN: 30% noisy Lorenz series

d	number of patterns	Local averaging		ANN	
		NRMSE	MAE	NRMSE	MAE
0.1	3901	0.3893	4.1873	0.3917	4.2137
0.105	3693	0.3877	4.1745	0.3796	4.0845
0.11	3435	0.3889	4.1856	0.3903	4.2024
0.115	3176	0.3874	4.1539	0.3813	4.0969
0.12	2912	0.3892	4.1821	0.3889	4.2285
0.125	2626	0.3921	4.2227	0.4024	4.3361
0.13	2375	0.3950	4.2352	0.3932	4.2655
0.135	2152	0.3955	4.2454	0.3972	4.2195
0.14	1901	0.3948	4.2029	0.4048	4.3364
0.145	1730	0.4013	4.285	0.4047	4.3545
0.15	1531	0.4008	4.2458	0.4056	4.3228
0.155	1368	0.4061	4.262	0.4195	4.4964
0.16	1222	0.3939	4.1772	0.4050	4.3281
0.165	1128	0.4049	4.3085	0.4237	4.4793
0.17	1000	0.4112	4.3657	0.4239	4.5620
0.175	892	0.4141	4.3832	0.4549	4.7859
0.18	806	0.4186	4.4374	0.4662	4.8889
0.185	739	0.4159	4.4226	0.4623	4.9317
0.19	632	0.4375	4.6860	0.4947	5.1245
0.195	569	0.4390	4.7181	0.4746	4.9304
0.2	521	0.4532	4.8985	0.5187	5.4150
0.205	480	0.4345	4.6806	0.4770	4.9968
0.21	412	0.4527	4.7791	0.5222	5.3569
0.215	390	0.4493	4.7176	0.5709	5.8467
0.22	329	0.4596	4.8248	0.6079	6.2022
0.225	292	0.4490	4.7564	0.7012	6.8881
0.23	272	0.5580	5.7964	0.6446	6.5516
0.235	231	0.5469	5.6923	0.7460	7.4022
0.24	210	0.5075	5.4052	0.7163	7.2265
0.245	200	0.5343	5.6399	0.6630	6.8653
0.25	182	0.5321	5.6567	0.7021	7.0353
0.255	157	0.5696	5.8836	0.7021	7.3372
0.26	146	0.5513	5.6169	0.6097	6.0901
0.265	128	0.5642	6.0273	0.6739	7.1183
0.27	125	0.5072	5.2646	0.5343	5.6188
0.275	105	0.5563	5.8843	0.6741	7.1467
0.28	102	0.5365	5.6609	0.5564	5.8154
0.285	89	0.5257	5.5704	0.5860	6.1477
0.29	85	0.5216	5.5387	0.5514	5.6884
0.295	74	0.5351	5.7071	0.5819	5.9236
0.3	69	0.5626	5.7714	0.6571	6.8498

Table K.4 The *d* values and the corresponding number of patterns selected and the prediction errors on validation set using for local model and ANN: Mississippi river flow time series

d	number of patterns	Local averaging		ANN	
		NRMSE	MAE (m^3/s)	NRMSE	MAE (m^3/s)
0.001	5011	0.0517	297.81	0.0385	206.29
0.002	4040	0.0527	303.97	0.0388	207.48
0.003	3225	0.0541	312.11	0.0389	209.33
0.004	2659	0.0583	332.89	0.0391	213.54
0.005	2224	0.0590	341.08	0.0397	214.92
0.006	1679	0.0594	350.42	0.0395	213.38
0.007	1449	0.0725	418.76	0.0405	219.91
0.008	1277	0.0758	437.82	0.0407	220.82
0.009	1023	0.0757	454.67	0.0403	220.19
0.01	923	0.0724	437.66	0.0418	227.62
0.011	788	0.0683	413.57	0.0410	223.21
0.012	718	0.0736	436.27	0.0423	230.94
0.013	658	0.0738	436.31	0.0428	236.56
0.014	588	0.0825	491.63	0.0423	230.09
0.015	511	0.0886	524.99	0.0432	235.99
0.016	480	0.0850	522.54	0.0430	235.64
0.017	438	0.0857	512.20	0.0463	255.16
0.018	394	0.0924	549.82	0.0465	260.02
0.019	372	0.0846	516.19	0.0449	244.79
0.02	342	0.0866	529.91	0.0461	256.91
0.021	323	0.0917	555.05	0.0467	263.04
0.022	285	0.0962	591.16	0.0464	262.12
0.023	266	0.0956	592.09	0.0453	252.21
0.024	256	0.0943	598.32	0.0457	262.80
0.025	241	0.0927	580.77	0.0471	264.51
0.026	225	0.0953	603.44	0.0441	249.67
0.027	211	0.1057	649.04	0.0467	262.99
0.028	195	0.1037	649.04	0.0503	273.86
0.029	185	0.1031	642.99	0.0479	281.13
0.03	176	0.1117	700.48	0.0483	267.77
0.031	170	0.1084	664.14	0.0463	267.78
0.032	162	0.1028	651.44	0.0453	266.53
0.033	153	0.1079	682.78	0.0470	280.24
0.034	149	0.1181	738.49	0.0497	304.14
0.035	133	0.1156	735.62	0.0455	262.25
0.036	128	0.1177	748.64	0.0465	254.63
0.037	125	0.1186	745.42	0.0502	301.91
0.038	120	0.1188	740.78	0.0481	298.65
0.039	110	0.1241	790.64	0.0651	386.57
0.04	106	0.1206	760.18	0.0504	303.26

Table K.5 The d values and the corresponding number of patterns selected and the prediction errors on validation set using for local model and ANN: Wabash river flow time series

d	number of patterns	Local averaging		ANN	
		NRMSE	MAE (m^3/s)	NRMSE	MAE (m^3/s)
0.001	4581	0.1195	48.23	0.0605	25.74
0.002	3767	0.1285	52.66	0.0606	26.00
0.003	3294	0.1287	52.92	0.0617	26.25
0.004	2928	0.1305	55.77	0.0615	26.27
0.005	2642	0.1490	58.93	0.0603	26.16
0.006	2385	0.1482	60.10	0.0674	28.62
0.007	2163	0.1521	63.49	0.0694	28.83
0.008	1944	0.1536	63.71	0.0599	26.51
0.009	1761	0.1545	64.44	0.0639	26.79
0.01	1586	0.1516	63.78	0.0733	28.83
0.011	1466	0.1530	63.71	0.0729	27.80
0.012	1325	0.1525	63.50	0.0732	27.59
0.013	1225	0.1526	63.73	0.0676	32.87
0.014	1120	0.1533	64.78	0.0717	32.22
0.015	1033	0.1543	65.88	0.0820	28.50
0.016	968	0.1563	67.04	0.0652	27.45
0.017	892	0.1549	66.91	0.0750	28.53
0.018	817	0.1626	73.74	0.0656	30.58
0.019	769	0.1764	75.83	0.0797	41.14
0.02	713	0.1623	72.46	0.0729	28.46
0.021	673	0.1634	73.08	0.0664	31.78
0.022	627	0.1623	72.57	0.0903	30.88
0.023	585	0.1679	74.76	0.0944	30.57
0.024	559	0.1669	74.39	0.0766	29.87
0.025	520	0.1689	76.66	0.0698	30.14
0.026	490	0.1700	76.71	0.0726	32.78
0.027	459	0.1741	77.21	0.0657	29.82
0.028	444	0.1718	80.50	0.0670	28.08
0.029	418	0.1744	84.98	0.0678	32.33
0.03	399	0.1716	80.76	0.0656	29.39
0.031	380	0.1809	85.18	0.0716	37.78
0.032	358	0.1740	84.82	0.0655	28.14
0.033	345	0.1774	90.02	0.0662	34.60
0.034	324	0.1823	92.32	0.1050	40.59
0.035	317	0.1837	88.47	0.1122	33.81
0.036	300	0.1816	86.53	0.0971	33.04
0.037	296	0.1781	87.44	0.0800	35.07
0.038	280	0.1812	91.80	0.0709	31.91
0.039	269	0.1803	91.33	0.0862	30.64
0.04	255	0.1784	89.07	0.0666	30.98
0.041	242	0.1738	88.90	0.0715	30.63
0.042	231	0.2284	123.23	0.0769	46.57

Table K.5 (Continued)

0.043	234	0.1846	97.15	0.0767	40.03
0.044	222	0.1911	100.53	0.0685	31.23
0.045	210	0.1994	106.59	0.0746	35.25
0.046	206	0.1834	94.76	0.0752	34.87
0.047	201	0.1846	101.20	0.0775	36.53
0.048	193	0.1940	104.06	0.0698	35.62
0.049	177	0.2118	123.61	0.0748	39.44
0.05	178	0.2113	112.29	0.0730	38.19

LIST OF PUBLICATIONS

Parts of this thesis have been published in or in the preparation for submission to the following international Journals or Conferences:

INTERNATIONAL JOURNALS

(1) Karunasinghe, D.S.K and Liong, S.Y. (2005). Chaotic time series prediction with a global model: Artificial Neural Network. Journal of Hydrology (in press).

(2) Doan, C.D., Liong, S.Y., and Karunasinghe, D.S.K. (2005). Derivation of effective and efficient data set with subtractive clustering method and genetic algorithm. Journal of Hydroinformatics, 7(4), pp 219-233.

(3) Karunasinghe, D.S.K., and Liong, S.Y. A simple clustering technique to extract most representative data from noisy chaotic time series. Submitted for the possible publication in the Journal of Hydrologic Engineering (ASCE).

(4) Karunasinghe, D.S.K., and Liong, S.Y. Real-time noise reduction and prediction of chaotic hydrological time series with Extended Kalman Filtering (EKF) (in preparation).

KEYNOTE PAPER

Liong, S.Y., Doan, C.D., and Karunasinghe, D.S.K. (2005). Role of Hydroinformatics in Integrated water resources management. MTERM International Conference, 06 – 10 June 2005, AIT, Thailand.

INTERNATIONAL CONFERENCES

(1) Karunasingha, D.S.K., and Liong, S.Y. (2003). Extracting effective phase space vectors for prediction in dynamical systems approach. First international conference on Hydrology and Water Resources in Asia Pacific Region (APHW2003), 13-15 March 2003, Japan, pp. 576 – 581.

(2) Liong, S.Y, Doan, C.D., Karunasingha, D.S.K., Ong, C.H. (2003). Deriving effective and efficient data set with subtractive clustering algorithm and genetic algorithm. XXX IAHR Congress 24-29 August 2003 in Greece, Theme D, pp.23 – 30.

(3) Karunasingha, D.S.K., and Liong, S.Y. (2004). A simple clustering technique to extract most representative data from noisy chaotic time series. Proceedings of the 6th International Conference on Hydroinformatics, 21-24 June, Singapore, pp. 1613 – 1620.

(4) Karunasingha, D.S.K., and Liong, S.Y. (2005). Real-time noise reduction and prediction of chaotic time series with Extended Kalman Filtering (EKF). 2nd Conference of Asia Oceania Geosciences Association, 20-24 June, Singapore, CD Rom.