Predictive Pattern Discovery in Dynamic Data Systems

Wenjing Zhang
Marquette University

PREDICTIVE PATTERN DISCOVERY IN DYNAMIC DATA SYSTEMS

by

Wenjing Zhang, B.S., M.S.

A Dissertation Submitted to the Faculty of the
Graduate School, Marquette University,
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

Milwaukee, Wisconsin

May 2013

ABSTRACT
PREDICTIVE PATTERN DISCOVERY IN DYNAMIC DATA SYSTEMS

Wenjing Zhang, B.S., M.S.

Marquette University, 2013

This dissertation presents novel methods for analyzing nonlinear time series in dynamic systems. The purpose of the newly developed methods is to address the event prediction problem through modeling of predictive patterns. Firstly, a novel categorization mechanism is introduced to characterize different underlying states in the system. A new hybrid method was developed utilizing both generative and discriminative models to address the event prediction problem through optimization in multivariate systems.

Secondly, in addition to modeling temporal dynamics, a Bayesian approach is employed to model the first-order Markov behavior in the multivariate data sequences. Experimental evaluations demonstrated superior performance over conventional methods, especially when the underlying system is chaotic and has heterogeneous patterns during state transitions.

Finally, the concept of adaptive parametric phase space is introduced. The equivalence between time-domain phase space and associated parametric space is theoretically analyzed.

ACKNOWLEDGEMENT

Wenjing Zhang, B.S., M.S.

I am deeply grateful to my academic advisor, Professor Xin Feng, for his guidance, support, and encouragement during my research study in the past few years. From Dr. Feng, I have learned not only his insightful vision of academic research, but also the wisdom of life through my study at Marquette University. Special thanks to Professor Edwin E. Yaz for his kind support, patience, and encouragement during the past few years.

I am grateful to EECE Department of Marquette University for its financial support of this research and to the members of ACT lab for many interesting discussions and friendships.

In addition, this research has benefited from Dr. Povinelli's pioneering work in this field. His kind help and inspiration is gratefully acknowledged. I would also like to thank the committee members, Dr. George Corliss and Dr. Naveen Bansal, for their valuable advice on my research and academic writing.

I want to express my deep gratitude to my wife, Wenting Zhou, for her consistent support, love, and sacrifice. I would also like to thank my parents for their support and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

This dissertation studies new computational methods with the goal of forecasting events and detecting temporal patterns in a dynamic data system (DDS) [1]. The focus of this research is to identify the temporal patterns predictive of future events of interest in the DDS. The methods introduced in this dissertation are major contributions in the field of machine learning and DDS analysis. The new methods extend the original univariate reconstructed phase space (RPS) framework [2, 3], based on the unsupervised clustering method, by incorporating a new mechanism of data categorization based on the definition of events. Furthermore, a Multivariate Reconstructed Phase Space (MRPS) is introduced to overcome the limitation of the univariate RPS approach by considering multivariate data sequences in the DDS. In addition to modeling temporal dynamics in a multivariate phase space, a Bayesian approach [4] is applied to model the first-order Markov behavior in multi-dimensional data sequences.

1.1 Background

A data sequence is a series of sequential observations representing the measurements of a DDS:

$$X = \{x_t,\ t = 1,2,...,N\}, \tag{1.1}$$

where *t* is the time index, and *N* is the total number of observations. This sequence contains the events of interest of the underlying dynamic system that are usually complex. These events are related closely to time-ordered structures, called temporal patterns in the sequence. In a multivariate dynamic system with *m* explanatory variables, multiple data sequences $\boldsymbol{X}(t)$ representing data measurements can be written as:

$$\boldsymbol{X}(t) = [x_{1t}, x_{2t}, ..., x_{mt}, x_{et}]^T,\ t = 1,2,\ldots,N, \tag{1.2}$$

where t is the time index; N is the total number of observations; $\{x_{e,t},\ t = 1,2,\ldots,N\}$ denotes the event sequence containing events of interest; and $\{x_{it},\ t = 1,2,\ldots,N\},\ i = 1,2,\ldots,m$ denotes multivariable sequences.

Discovering temporal patterns related closely to the events in a DDS is important for many applications. For example, in financial applications, significant interest has focused on determining the timings of positions of securities [5] and forecasting economic growth and outlook [6]. In the medical fields, medical anomaly detection [7, 8] has been used widely for monitoring the health conditions of patients. The interpretation of underlying system dynamics for preventing abnormal events is another area of interest [9, 10].

There are two major research directions: univariate and multivariate temporal pattern approaches.

Among the univariate methods, existing frequency domain approaches using a Discrete Fourier Transform (DFT) [11] or a Discrete Wavelet Transform (DWT) [12, 13] to classify or match time sequence data are based on spectral patterns to reduce the dimensions of the feature space. The frequency domain transformation chooses fewer but better coefficients to characterize the DDS. Since these approaches focus on the overall dynamic characteristics of the system, data sequences with different nonlinear dynamic patterns, but similar power spectra, may not be distinguished. The method using a piecewise linear representation in [14] is based on representing patterns as a set of simple templates and requires a priori knowledge of the internal structures of the DDS.

Multivariate approaches to temporal pattern identification include nonlinear classification using neural networks [15, 16], decision trees [17, 18], and clustering algorithms [19]. However, since the central focus of these methods is on the point-by-point "curve-fitting" strategy, not enough attention has been given to the exploration of dynamic relationships between the data sequence segments – temporal patterns – and the critical occurrences of the events of interest.

Studies in dynamic systems and chaos theory provide a new pattern identification approach based on the RPS [1, 20]. The RPS is capable of representing temporal patterns of nonlinear dynamic sequence data. The underlying theory discussed in [21, 22] guarantees that such an embedding in the RPS can describe the dynamics of a system given that the dimension of the phase space is greater than twice the box-counting dimension of the underlying system. Time Series Data Mining (TSDM) [2, 3, 7, 23] is an effective approach to detect the temporal pattern. The objective is to identify the hidden characteristics that lead to special events of interest in the DDS. The hidden characteristics under discussion are observed in terms of temporal patterns, and such patterns can be applied to forecast future events. Event functions were used to define and characterize the eventness at each time step.

The definition of events usually depends on the specific applications. However, there are several commonly used functions, called event characterization functions:

1. Next step thresholding:

$$g(\mathbf{x}_t) = x_{t+1} - c > 0 \ . \tag{1.3}$$

2. Multiple step thresholding:

$$g(\mathbf{x}_t) = \max\{x_{t+1}, \ldots, x_{t+k}\} - c > 0 \ . \tag{1.4}$$

3. Next step difference thresholding (typically used in stock price prediction):

$$g(\mathbf{x}_t) = \frac{x_{t+1} - x_t}{x_t} - c > 0 \ . \tag{1.5}$$

In Eqns. (1.3)-(1.5), $\mathbf{x} \in R^m$ represents a RPS embedding, *g* is a scalar event function, and $c > 0$ and $k > 0$ are given constants.

The algorithm design under the RPS frameworks typically involves a phase space embedding by selecting the time delay [24], embedding dimension [25], and the optimization algorithm [26, 27] that maximizes the objective function. Fig. 1.1 illustrates the general procedure in temporal pattern identification.

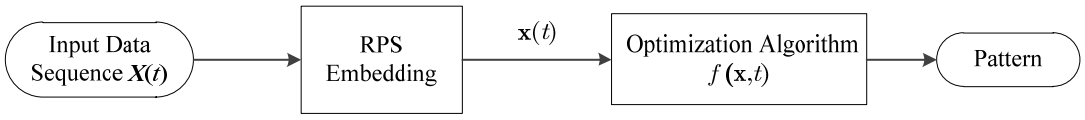

Figure 1.1: General procedure in a temporal pattern identification system

Although the existing RPS-based approach has proven effective for univariate DDS, much less research has been on multivariate case where multiple data sequences are present and correlated. Often in the area of data mining and pattern analysis, we are interested not only in detecting the events in the event data sequence, but also in exploring the causal relationship with the underlying factors and variables. This dissertation addresses some of the major challenges of univariate and multivariate temporal pattern detection and explores the relationships between causal variables and events of interest.

1.2 Research Objective

In this dissertation, we address the problem of detecting predictive temporal patterns in a complex DDS. Due to significant nonlinearity, techniques for the discovery of underlying hidden patterns need to be designed carefully. In this work, we present several fundamental contributions in designing robust algorithms for predictive pattern detection in DDS.

The major contributions of this dissertation include:

1) *A new RPS transformation and associated similarity measure definition*.
   Since the similarity measure defined by the Euclidean distance in existing methods was unable to identify similar patterns when data sequences have certain trends, the proposed transformation is capable of preserving the similarity between patterns by eliminating the effect of trending. Furthermore, the new transformation still gives a faithful representation of the underlying dynamic system.
2) *The development of data sequence categorization based on event function*.
   This approach introduces a new definition of multiple states in the DDS, including normal state, pattern state, and event state. This new definition of underlying states provides a new perspective so that the complex systems can be interpreted better in terms of multiple state transitions.
3) *The new MRPS framework for the temporal pattern detection in multivariate DDS*.
   By incorporating multiple data sequences in a DDS, we are able not only to explore internal dynamics within a single variable, but also the relationship

between the target event sequence and causing variable data sequences. This method considers a multivariate data system and addresses the temporal pattern identification problem by solving a regularized optimization problem. The problem of finding patterns is transformed into a pattern classification problem. A classifier is designed to take into account both the time-dependent and time-independent factors within the DDS. This generalization from the univariate case to the multivariate RPS opens up the applicability of the RPS-based method to a wide range of multivariate applications.

4) *The establishment of equivalency between the state space and the RPS representation*.

   A new relationship is established between the mean and covariances of temporal cluster in the RPS and that in the corresponding state space represented by the parameters of dynamic time domain model, for example, the autoregressive parameter of phase space.

1.3 Dissertation Outline

This dissertation is organized as follows:

Chapter 2 presents the background theory of traditional time series analysis, the RPS, and various recently developed approaches. We also present the basic optimization concepts that are essential tools in developing efficient and high performance algorithms.

Chapter 3 presents a kernel-based method for temporal pattern detection in a univariate data system. The new method takes advantage of a Support Vector Machine (SVM) and a Gaussian Mixture Model (GMM) to design a classifier for detecting temporal patterns. In addition, by applying a SVM to the RPS, the classifier finds an

optimal classifier to determine the decision boundary between the patterns of interest and other unrelated ones. Moreover, a Maximum A Posterior (MAP) classifier can be constructed via the GMM. A final classification decision can be made by combing outputs from both classifiers.

Chapter 4 presents the new MRPS method. Whereas existing methods under the RPS are applicable only for a univariate system, the MRPS method can be applied for more general cases with multivariate input data sequences. The new method applies a convex exponential loss function together with a quadratic penalty term placed on the parameters. This new formulation avoids the need to optimize a complex nonconvex problem, which is typically sensitive to initial conditions. Furthermore, we introduce a new classifier design that considers both the time-dependent similarity measure of patterns and time-independent factors, such as probabilistic distributions.

Chapter 5 presents a new formulation of the Parametric Reconstructed Phase Space (PRPS). It is shown that the new space preserves the statistical properties, such as mean and covariances. Compared with the existing univariate approach, this new method gives an estimation of local temporal dynamics.

Chapter 6 gives a conclusion of this work and discusses the potential research directions and work that could be performed.

CHAPTER 2 LITERATURE REVIEW

This chapter reviews several conventional research areas related to time series analysis and temporal pattern recognition in the DDS. In Section 2.1, the fundamentals of traditional time series analysis, such as the Autoregressive Moving Average (ARMA) model [29], are presented. In Section 2.2, a nonlinear approach to time series analysis based on the theory of dynamical systems is discussed. The theory of nonlinear dynamical systems provides not only a direct link between chaos theory [1] and the real world DDS in terms of nonlinear dynamics, but also new tools and a theoretical foundation for the RPS framework, specifically Takens' theorem [20] and Sauer's extension [21], to characterize complex time series data. In Section 2.3, we discuss the recent research developments in the field of temporal pattern identification. Finally, in Section 2.4, we review the fundamental concepts of kernel methods and optimization theory.

2.1 Review of Time Series Analysis

The primary objective of time series analysis is to develop mathematical models that reveal patterns in the underlying system and make forecasts. The time domain approach to analyze a data system is motivated generally by the presumption that the correlation between adjacent points in time can be explained in terms of a dependence of the value at a current time on the values in the past. This time domain approach focuses on modeling some future values or events in a data system as a function of the current and past values. Therefore, this approach can be used as a forecasting tool in various applications, such as in financial markets and in economic forecasting.

Box and Jenkins [30] developed a class of models called ARMA models, in which the observed time series data are assumed to result from the products of factors involving difference equation operators responding to a white noise input.

A time series $\{x_t, t = 1,2....\}$ is ARMA (p,q) if it is stationary and

$$x_t = \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q} + e_t, \tag{2.1}$$

with $\sigma_e^2 > 0$, and $\{e_t, t = 1,2....\}$ as Gaussian white noise. In particular, the ARMA (p, q) model in Eqn. (2.1) can then be rewritten in a compact form:

$$\phi(B)x_t = \theta(B)e_t \ , \tag{2.2}$$

where the autoregressive operator is

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p, \tag{2.3}$$

and the moving average operator is

$$\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^p. \tag{2.4}$$

Several steps fit an ARMA model to time series data:

1) Construct a time plot of the data and inspect the graph for any anomalies.
2) Transform the data; for example, if the variability in the data grows with time or the underlying process evolves as a small percentage change.
3) Identify the orders of dependence of the model, such as values of the autoregressive order p and the order of the moving average q.
4) Estimate model parameters using methods such as the Yule–Walker [31] method or a numerical optimization technique [26].
5) Test the model using statistical methods, such as the F-test.

The properties of an ARMA model are only well understood if the input noise sequence $\{e_t, t = 1,2....\}$ is Gaussian distributed, that is, it is Gaussian white noise. However, for many real world applications, the noises in the data are not Gaussian distributed. Furthermore, although it is possible to reproduce the data better if a higher order of the model is used, it is essential to limit the number of orders in the model to prevent over-fitting. Additionally, in ARMA models, it is assumed generally that the system generating the data is stationary, i.e., the mean and variance of the system do not change over time. However, in many applications, a data system is not stationary.

In existing statistical tools, such as the ARMA model, the central focus is on a point-by-point "curve-fitting" strategy. However, not enough attention has been given to the exploration of dynamic relationships between the time series segments in the data sequences and the critical occurrences of the events of interest. As a result, the applicability of ARMA models in such scenarios is limited due to the underlying assumptions of the system.

2.2 Nonlinear Time Series Analysis

A DDS can be described by a set of temporal states with underlying rules to govern how the system may switch from one state to another [1]. A vector $\mathbf{s} \in R^Q$ usually specifies a state, and a set of first-order ordinary differential equations acting on a *Q*-dimensional vector space defines the dynamic system. For the discrete case, the next state can be described by a function of current state:

$$\mathbf{s}_{t+1} = f(\mathbf{s}_t). \tag{2.5}$$

For the continuous case, the dynamics of the underlying process can be described by

$$\frac{d}{dt}\mathbf{s}(t) = f(\mathbf{s}(t)). \tag{2.6}$$

Studies in dynamic systems and chaos theory provide a new pattern identification approach based on the RPS. A RPS is a dimensional metric space into which a data sequence is unfolded [21]. Given a sequence of observations of state variables in a dynamic system, it was proven that the chaotic dynamics of the system could be reconstructed in a phase space in which hidden temporal patterns can be detected [20]. Specifically, Takens [20] showed how lagged variables of a single time series can be used as proxy variables to reconstruct patterns for an underlying dynamic system.

Takens Theorem [20]: Let M be the state space of a Q dimensional dynamic system. For pairs (φ, y), $\varphi: M \to M$ is a map that describes the dynamics of the system state, and $y: M \to R$ is a twice continuously differentiable function that represents the observation of a single variable, then the mapping

$$\Phi_{(\varphi,y)}(x) = (y(x), y(\varphi(x)), ..., y(\varphi^{Q}(x))), \tag{2.7}$$

is an embedding.

Takens showed that if the embedding dimension is large enough, the phase space is capable of capturing the intrinsic structure of the state space from which a data sequence is generated. Therefore, given a data sequence $\{x_t, t = 1, 2....\}$, a time-delay embedding of observations can reconstruct a state space. This provides the theoretical justification for reconstructing state spaces using a time-delay embedding.

2.3 Recent Research Developments

In [2,3], data mining and optimization techniques have been applied under the RPS framework to identify temporal patterns in dynamic systems, especially complex and chaotic systems. These approaches are able to overcome the limitations of existing time series methods. It was shown that the RPS-based methods have better performance than traditional neural network and decision tree methods in many applications, such as welding droplet and financial market predictions. This section gives a brief review of these RPS frameworks.

2.3.1 Definition of an Event and an Event Function

In a DDS, an event is an important occurrence or observation reflecting the internal state of the system. For example, in a water treatment plant, a spike in the reading of a chemical might indicate a malfunctioning of the plant. A consecutive negative reading of the quarterly growth of the GDP is an important event that indicates a recession of the economy. However, the "eventness" of a system needs to be predefined depending on specific applications so that a quantitative treatment to the underlying problem can be obtained.

The event function is introduced to characterize and measure the "eventness". The event characterization function represents the value of future "eventness" for the present time index. Therefore, an event function is defined a priori to address the specific goal of the time series being considered. For example, to formalize the concept of a one-step or *k*-step prediction problem, an event function $g(\bullet)$ can be defined as

$$g(\mathbf{x}_t) = x_{t+1}, \tag{2.8}$$

or a *k*-step forward event function,

$$g(\mathbf{x}_t) = \max\{x_{t+1}, x_{t+2}, ..., x_{t+k}\}, \tag{2.9}$$

respectively. Another event function can be defined to measure the actual occurrence of the event at the current time index,

$$g(\mathbf{x}_t) = x_t, \tag{2.10}$$

where vector $\mathbf{x}_t = (x_t, x_{t-\tau}, x_{t-(Q-1)\tau})$ is an embedding in phase space with time-delay τ and dimension Q.

For example, if the event function $g(\mathbf{x}_t)$ has a value higher than a predetermined threshold *c* at current time step *t*, we say the defined event occurs. To make a forecast, we can evaluate $g(\mathbf{x}_t)$ in one or multiple time steps ahead using Eqns. (2.8) or (2.9). In some applications, such as the prediction of future security movement in a financial application, the primary focus is on the percentage change rather than the actual values. In such a case, an event function also can be defined by the percentage change in the next step,

$$g(\mathbf{x}_t) = x_{t+1} / x_t - 1. \tag{2.11}$$

Thus, the definition of event function can provide a useful quantitative measurement of an event.

2.3.2 Phase Space Embedding and Parameter Estimation

Given a univariate data sequence, a phase space can be reconstructed by a process called time-delay embedding [1]. The phase space vectors in R^Q are represented by:

$$\begin{bmatrix} \mathbf{x}_{1+(Q-1)\tau} \\ \mathbf{x}_{2+(Q-1)\tau} \\ \vdots \\ \mathbf{x}_N \end{bmatrix} = \begin{bmatrix} x_{1+(Q-1)\tau} & \cdots & x_{1+\tau} & x_1 \\ x_{2+(Q-1)\tau} & \cdots & x_{2+\tau} & x_{1+\tau} \\ \vdots & & \ddots & \\ x_N & \cdots & x_{N-(Q-2)\tau} & x_{N-(Q-1)\tau} \end{bmatrix}_{(N-(Q-1)\tau \times 1}, \tag{2.12}$$

where τ is the time delay, and $Q$is the dimension of the embedding vectors.

The time delay τ can be calculated by using the first minimum of the mutual information function, which provides the quantitative characteristics of spatial patterns in the phase space [24]. The minimum of the mutual information function was found to be effective in estimating the time delay. Given a data sequence and time delay τ, the data can be summarized in a contingence table or a histogram, and the mutual information is computed by

$$M(x_t, x_{t-\tau}) = \sum_{i,j} p_{ij}(\tau) \ln\left(\frac{p_{ij}(\tau)}{p_i p_j}\right) \tag{2.13}$$

where p_i is the probability that x_t falls in the *i*th interval, and $p_{ij}(\tau)$ is the joint probability that x_t falls into the *n*th interval and $x_{t-\tau}$ falls into the *j*th interval.

The dimension Q of the RPS is determined using a false nearest-neighbors technique [25, 32]. The percentage of false nearest-neighbors changes with different choice of the embedding dimension Q. The smallest Q that gives the lowest percentage of false nearest-neighbors is selected as the optimal embedding dimension. Specifically, for each data point $\mathbf{x}_i^Q$ in R^Q, a difference measure is

$$r_i = \sqrt{\frac{\left\|\mathbf{x}_i^{Q+1} - \mathbf{x}_j^{Q+1}\right\|^2 - \left\|\mathbf{x}_i^Q - \mathbf{x}_j^Q\right\|^2}{\left\|\mathbf{x}_i^Q - \mathbf{x}_j^Q\right\|^2}}, \tag{2.14}$$

where $\left\|\mathbf{x}_i^Q - \mathbf{x}_j^Q\right\|$ is the Euclidean distance between $\mathbf{x}_i^Q$ and $\mathbf{x}_j^Q$

$$\mathbf{x}_j^Q = \underset{\mathbf{x}_j^Q, i \neq j}{\operatorname{argmin}} \left\| \mathbf{x}_i^Q - \mathbf{x}_j^Q \right\| . \tag{2.15}$$

$\mathbf{x}_i^Q$ is marked as having a false nearest neighbor if r_i exceeds a given threshold ρ. The criterion for an adequate embedding dimension Q is the number of data points for which $r_i > \rho$ is smallest in R^Q.

2.3.3 Formulation of Objective Function

Objective functions can have different formulations for temporal pattern identification. By optimizing an objective function with respect to parameters, such as the center and radius of clusters, a unique classifier can be determined to categorize patterns related to and not related to events, respectively. Under such a formulation, the problem of searching for predictive temporal patterns in the phase spaces can be transformed into an optimization problem maximizing or minimizing the objective function with respect to the underlying parameters.

The objective function can take different forms depending on the goal of the pattern identification. In [2], several objective functions are presented for different pattern identification tasks.

(1) The maximal event characterization function [3] was defined as

$$f(\mathbf{v}, \delta) = \begin{cases} \mu_M & \left\| \mathbf{x}_i - \mathbf{v} \right\| < \delta, \text{ if } M \geq \beta N \\ (\mu_M - g_0) \dfrac{M}{\beta N} + g_0 & \text{otherwise,} \end{cases} \tag{2.16}$$

where g_0 is the smallest event value in the cluster [3],

M is the number of data points in the cluster,

N denotes the number of total data points,

β is the rate of the minimum cluster size [3], and

μ_M is the mean of the event function within the cluster.

(2) The statistical significance of patterns is evaluated by a *t*-test, which represents the difference between two independent means,

$$f(\mathbf{v},\delta) = \frac{\mu_{Mc} - \mu_{\tilde{M}c}}{\sqrt{\frac{\sigma^2_{Mc}}{C(Mc)} + \frac{\sigma^2_{\tilde{M}c}}{C(\tilde{M}c)}}}\ , \tag{2.17}$$

where Mc represents the set of data points within cluster,

$\tilde{M}c$ is the set of data points outside of the cluster,

$C(Mc)$ and $C(\tilde{M}c)$ are the number of data points in these two sets,

μ_{Mc} and $\mu_{\tilde{M}c}$ are the mean event values of the two sets, and

σ_{Mc} and $\sigma_{\tilde{M}c}$ are the standard deviations of Mc and $\tilde{M}c$, respectively.

(3) The overall accuracy can be useful in problems where the accuracy of the predictions that an event occurs is of primary importance. The overall accuracy is defined as

$$f(\mathbf{v},\sigma) = \frac{tp + tn}{tp + tn + fp + fn}, \tag{2.18}$$

where *tp* represents the number of true positives,

tn represents the number of true negatives,

fp represents the number of false positives, and

fn represents the number of false negatives, respectively.

(4) Fuzzy-set objective function [3], which takes the density of patterns into consideration and can be robust to noisy data points:

$$f(\mathbf{v},\delta)\} = \sum_{t=1}^{N} \exp\left(-\frac{\|\mathbf{v}-x_t\|^2}{2\delta^2}\right) g(\mathbf{x}_t), \tag{2.19}$$

where $\mathbf{v}$ represents the center of the fuzzy cluster,

δ represents the radius of the fuzzy cluster,

N denotes the number of total data points,

β is the rate of the minimum cluster size [3], and

μ_M is the mean of the event function within the cluster.

2.4 Pattern Classification and Optimization Theory

In this section, we discuss three key components in developing the new methods: Bayesian decision theory, kernel methods, and optimization theory. The detailed applications will be discussed in Chapters 3 and 4.

2.4.1 Bayesian Decision Theory

Bayesian decision theory [4, 19, 33] is a statistical approach based on quantifying the tradeoffs between classification decisions to tackle the problem of pattern recognition. This theory uses probability and the costs to measure the tradeoffs that accompany classification decisions. For instance, given multiple possible hypotheses, we can apply Bayesian decision theory to find the optimum decision rule for deciding which hypothesis is correct.

Table 2.1: A list of decision theory notation

$\mathbf{x} \in R^d$	d-dimensional feature vector
$\{\omega_1, \omega_2, \ldots, \omega_c\}$	set of c classes/categories
$\{\alpha_1, \alpha_2, \ldots, \alpha_a\}$	set of a possible actions
$\lambda_{ij} = \lambda(\alpha_i \mid \omega_j)$	cost of action i if class j is true
$\alpha(\mathbf{x})$	decision rule/function
$P(\omega_j \mid \mathbf{x})$	posterior distribution of class j given $\mathbf{x}$

Given a classification task of *c* classes, $\omega_1, \omega_2, \ldots, \omega_c$ and an unknown pattern, which is represented by a feature vector $\mathbf{x}$, we can form *c* conditional probabilities $P(\omega_j \mid \mathbf{x}), j = 1, 2, ..., c$. They are also referred to as a posteriori probabilities, and each of them represents the probability that the unknown pattern belongs to the respective class ω_j.

Without loss of generality, we consider a two-class problem with each class denoted by ω_1 and ω_2. We assume that the priori probabilities of these two classes $P(\omega_1)$ and $P(\omega_2)$ are known. Even if they are not known under some circumstances, priori probabilities can easily be estimated from training datasets.

The Bayes' formula is

$$P(\omega_j \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid \omega_j) P(\omega_j)}{P(\mathbf{x})}, \tag{2.20}$$

where in the case of two categories, $j = 1, 2,$ and

$$P(\mathbf{x}) = \sum_{j=1}^{2} P(\mathbf{x} \mid \omega_j) P(\omega_j). \tag{2.21}$$

The Bayesian classification rule can be stated as

$\mathbf{x}$ is assigned to class 1, if $P(\omega_1 \mid \mathbf{x}) > P(\omega_2 \mid \mathbf{x})$

$\mathbf{x}$ is assigned to class 2, if $P(\omega_1 \mid \mathbf{x}) < P(\omega_2 \mid \mathbf{x})$.

For a two-class classification problem, as illustrated in Fig. 2.2, each class can be represented by a probability density function on the domain of x, and x_0 is the decision boundary between these two classes. The risk associated with such a decision rule is equal to the total shaded area under the curves in Fig. 2.2.

Hence, a decision rule, by definition, is a function that maps feature vectors in the feature space to each class. Regions in the feature space associated with the various classes are called decision regions.

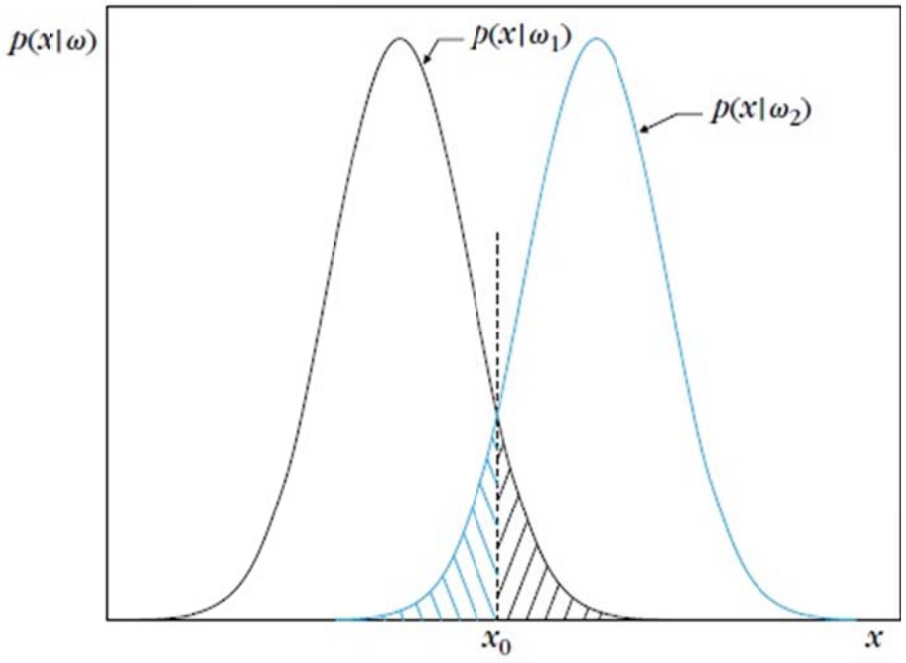

Figure 2.1: An illustration of the two-class Bayesian decision rule.

Furthermore, the Bayesian classifier also is optimal with respect to minimizing the classification error probability. If we have the decision rule to pick the class with a maximum a posteriori (MAP) probability $P(\omega_j \mid \mathbf{x})$, the posterior probability of error is

$$P(error \mid \mathbf{x}) = \sum_{i=1}^{c} \big(1 - P(\omega_i \mid \mathbf{x})\big) P(\omega_i), \tag{2.22}$$

and the total overall probability of error is

$$P(error) = \int_{-\infty}^{\infty} P(error \mid \mathbf{x})\, P(\mathbf{x})\, dx. \tag{2.23}$$

The idea of the posterior probability of error can be generalized to include the concept of variable costs for taking different actions. If α_i is an action, and $\lambda_{ij} = \lambda(\alpha_i \mid \omega_j)$ is the cost of taking action α_i in situation ω_j, then the resulting expected value of the cost, called the conditional risk, is given by

$$R(\alpha_i \mid \mathbf{x}) = \sum_{j=1}^{c} \lambda_{ij} P(\omega_j \mid \mathbf{x}), \tag{2.24}$$

and the expected value of the risk over all data is then

$$R = \int R\big(\alpha(\mathbf{x})\big)\, P(\mathbf{x})\, dx. \tag{2.25}$$

The minimum risk (i.e., the minimum expected cost) decision rule chooses the lowest-risk action for each possible $\mathbf{x}$,

$$\text{Select action } i = \underset{i=1\ldots a}{\arg\min} \left\{ R(\alpha_i \mid \mathbf{x}) = \sum_{j=1}^{c} \lambda_{ij} P(\omega_j \mid \mathbf{x}) \right\}. \tag{2.26}$$

This is called the Bayes' decision rule, and the resulting overall risk is called the Bayes' risk.

2.4.2 Kernel Methods

In Section 2.4.1, Bayesian decision theory is discussed in designing classifiers based on probability density functions. However, not all problems are well suited to this approach. For example, in some cases, the probability density functions in the problem are so complicated that direct estimation is not an easy task.

In this section, designing a classifier using kernel method [4, 34-41] is discussed. One of the major advantages of kernel-based classifiers is their simplicity and computational attractiveness. In addition, these classifiers do not involve the estimation of distribution functions in the data. For example, one kernel-based model is the Parzen probability density model [42] comprised of a linear combination of kernel functions, each one centered on one of the training data points.

Given two vectors $\mathbf{x}$ and $\mathbf{x}'$ in a high dimensional space and a fixed nonlinear basis function $\phi(\mathbf{x})$, the kernel function $k(\mathbf{x},\mathbf{x}')$ is given by the relation,

$$k(\mathbf{x},\mathbf{x}') = \phi(\mathbf{x})\phi(\mathbf{x}') . \tag{2.27}$$

With this definition, we can also see that $k(\mathbf{x},\mathbf{x}')$ is symmetric, for example, $k(\mathbf{x},\mathbf{x}') = k(\mathbf{x}',\mathbf{x})$. There are several forms of kernel functions:

1) A polynomial kernel of order p,

$$k(\mathbf{x},\mathbf{x}') = (1+\mathbf{x}\mathbf{x}')^{p} . \tag{2.28}$$

2) A Gaussian radial-basis function kernel,

$$k(\mathbf{x},\mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right). \tag{2.29}$$

3) A sigmoid kernel,

$$k(\mathbf{x},\mathbf{x}') = \tanh(\beta_1 \mathbf{x}\mathbf{x}' + \beta_2)\,. \tag{2.30}$$

In the context of machine learning and pattern recognition, the Representer Theorem [4] shows how the kernel functions can be applied in the design of classifiers.

Theorem 2.1 : (Representer Theorem) [4]

Denote by $\Omega : [0,\infty) \to R$*, a strictly monotonic increasing function, by a set* X*, and by* $L : (X \times \mathbb{R}^2)^m \to \mathbb{R} \cup \{\infty\}$ *an arbitrary loss function,* $\mathcal{H}$ *is the reproducing kernel Hilbert space associated with kernel* K*. Then each minimizer* $f \in \mathcal{H}$ *of the regularized risk,*

$$L((x_1, y_1, f(x_1)), \ldots, (x_m, y_m, f(x_m)) + \Omega(\|f\|_{\mathcal{H}})\,,$$

admits a representation of the form

$$f(x) = \sum_{i=1}^{m} \alpha_i K(x_i, x)\,.$$

This theorem states that although working in a high-dimensional space, the minimizer of the loss function, for example, the optimal solution, can be expressed as a linear combination of only finite kernels placed at the training points.

One of the popular kernel methods is the SVM classification method [33-35], which has received considerable interest, both in terms of theory and applications. See Vapnik [34] and Burges [38] for details.

In a two-class classification problem, our training dataset includes vectors $\mathbf{x}_i, i = 1,2,\ldots,l$ with an indicator vector $y_i \in \{+1,-1\}$ and slack variables ξ_i. We can define a hyperplane by

$$y(\mathbf{x}) = \mathbf{w}\phi(\mathbf{x}) + b\,. \tag{2.31}$$

A SVM approaches this problem through the concept of the margin, which is defined to be the smallest distance between the decision boundary and any of the samples, as illustrated in Figure 2.3. The goal of this approach is to find the hyperplane that creates the biggest margin between the training points of two classes.

The support vector classifier solves the optimization problem

$$\begin{aligned} \min_{\mathbf{w},b,\mathbf{x}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_i)+b) \geq 1-\xi_i \\ & \xi_i \geq 0,\ i=1,\ldots,l. \end{aligned} \tag{2.32}$$

By using the Karush–Kuhn–Tucker (KKT) condition, we can reformulate the problem as a dual problem:

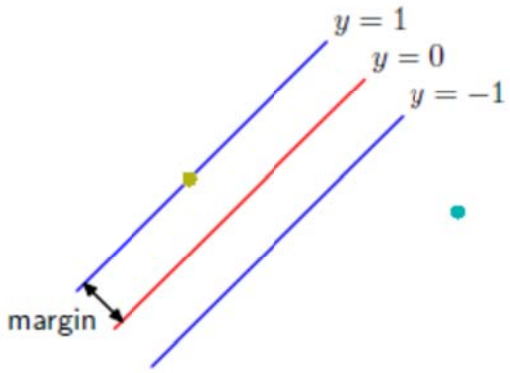

Figure 2.2: Definition of margin.

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j k(\mathbf{x}_i,\mathbf{x}_j), \\ \text{subject to} \quad & \sum_{i=1}^{l}\alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C, i=1,\ldots,l, \end{aligned} \tag{2.33}$$

where $k(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ is the kernel function.

After the problem in Eqn. (2.33) is solved, we can use the primal-dual relationship to obtain the optimal coefficients $\mathbf{w}$ that satisfy

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \phi(\mathbf{x}_i), \tag{2.34}$$

and the resulting decision rule is

$$\text{sgn}(\mathbf{w}^{\mathrm{T}} \phi(\mathbf{x}) + b), \tag{2.35}$$

which can also be written in the form of kernel functions as

$$\text{sgn}\left(\sum_{i=1}^{l} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b \right). \tag{2.36}$$

Therefore, a classifier can be constructed using kernel functions according to Eqn. (2.36).

2.4.3 Optimization Theory

In the previous section, we reviewed the kernel methods applied widely in the machine learning community. The method of optimization [26, 27], in general, is also a powerful tool that can deal with various kinds of pattern recognition tasks. Optimization is well rooted as a principle underlying the analysis of many complex decision-making or resource allocation problems. From a mathematical perspective, optimization is to maximize or minimize a real-valued objective function with or without constraints on its variables. When no constraints are placed on its variables, the problem is called unconstrained optimization; otherwise, it is called constrained optimization.

The general mathematical programming problem is

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) \\ &\text{subject to} && h_i(\mathbf{x}) = 0, \qquad i = 1,2,...,m \\ &&& g_j(\mathbf{x}) \le 0, \qquad j = 1,2,...,r \\ &&& \mathbf{x} \in S. \end{aligned}$$

In this formulation, $\mathbf{x}$ is an *n*-dimensional vector of unknowns, $\mathbf{x} = (x_1, x_2, ..., x_n)$, and f, h_i, $i = 1,2,...,m$, and $g_j(\mathbf{x}) \le 0$, $j = 1,2,...,r$, are real-valued functions of the variables $x_1, x_2, ..., x_n$. The set *S* is a subset of an *n*-dimensional space. The function f is the objective function of the problem, and the equations, inequalities, and set restrictions are constraints.

Now that basic concepts of decision theory, kernel methods, and optimization methods for pattern identification are reviewed, in the following Chapters 3 and 4 the event prediction problem will be addressed based on event characterization and temporal pattern classification.

CHAPTER 3 ENHANCED TEMPORAL PATTERNS IDENTIFICATION USING A GAUSSIAN MIXTURE MODEL AND A SUPPORT VECTOR MACHINE

In this chapter, a new approach is presented for identifying temporal patterns that are predictive of events, i.e., temporal predictive patterns, in univariate data sequences. This approach employs an event function to define quantitatively the problem and events based on specific application. A hybrid model using a Gaussian Mixture Model (GMM) and a Support Vector Machine (SVM) is applied to predict events based on identification of temporal patterns in the RPS. Since temporal patterns typically are hidden in noise or unrelated signals, directly embedding time series data into the phase space may produce unsatisfactory results since the patterns of interests cannot be separated from unrelated embeddings. Therefore, it is desirable to apply a filtering method to preprocess the original data sequence based on the statistical properties of the underlying data sequence and events.

Most previous work considered temporal patterns forming spherical clusters in the phase space. This assumption may not be appropriate if patterns in the phase space form irregular non-spherical clusters. Since the goal is to predict events using temporal patterns and not to make point-by-point predictions, it is advantageous to separate the event patterns from unrelated nonevent patterns or noise.

3.1 New Event Function for Temporal Pattern Classification

A dynamic data system can be considered to have three different states, e.g., a normal state, a pattern state, and an event state. Consequently, data points in the system can be clustered into three categories of signals, each of them belong to a normal state, a

pattern state, and an event state, respectively. In the following, we denote ω_n, ω_p, and ω_e as three class labels for the normal state, the pattern state, and the event state, respectively.

Consider a univariate data sequence defined as $X = \{x_t,\ t = 1, 2, \ldots N\}$. A general form of the event function is

$$g(\mathbf{x}_t) = \begin{cases} +1 & \text{if } x_t <= c \text{ and } \max\{x_{t+1}, \ldots, x_{t+k}\} > c \\ -1 & \text{if } x_t <= c \text{ and } \max\{x_{t+1}, \ldots, x_{t+k}\} <= c \\ 0 & \text{if } x_t > c, \end{cases} \tag{3.1}$$

where *k* is the time-step ahead, and *c* is the defined threshold of the event. A constant *k* is predetermined to specify the maximum forecasting time horizon.

In the training stage, based on the event function in (3.1), each observation $\mathbf{x}_t$ can be associated with a label, which takes a value in $\{+1, -1, 0\}$, representing the true occurrence of the event within a *k* step time horizon. Each observation vector $\mathbf{x}_t$ is assigned a label and a category:

1. Predictive temporal patterns are data points labeled +1 and categorized as class ω_p,
2. Non-predictive points are data points labeled –1 and categorized as class ω_n.
3. Event points are data points labeled 0 and categorized as class ω_e.

In summary, based on the definitions above, a vector $\mathbf{x}_t$ can be classified as ω_p, ω_e, or ω_e according to Eqn. (3.1). A data sequence can be considered a mixture of three classes of signals representing three recurring states: ω_n, ω_p, and ω_e.

For most applications, a primary focus is to predict events based on temporal patterns when the underlying system is not in an event state. Thus, for identification of predictive patterns in the training stage, our focus is on the classification of two

categories of data that are ω_p and ω_n. Then, in the testing stage, if a temporal pattern is classified as ω_p, a forecast that an event will occur is made.

3.2 Algorithm Design

Previous work under the RPS was able to characterize the temporal structures of the patterns by an unsupervised clustering approach. However, there is a lack of literature applying a supervised classification approach under a RPS framework to predict events. In this section, a new hybrid classification method is presented by employing a SVM [33-39] and a GMM [4]. Given the assumption that a data sequence can be considered a mixture of three classes of variables – ω_n, ω_p, and ω_e, a GMM is well suited to make statistical inferences of underlying probability distributions of three underlying classes. A Maximum A Posterior (MAP) classifier then can be used based on the estimations of underlying distributions from the GMM to separate event patterns from nonevent patterns or noise. SVM [37-39] is a popular nonlinear model for two class classification problems based on pattern similarities, which is a central focus of the RPS-based methods. Fig. 3.1 shows the overall diagram of GMM-SVM method.

The GMM-SVM method can be summarized as follows:

1. Determine the dimension *Q* of the phase space and the lengths of the temporal patterns τ.
2. Using the Expectation Maximization (EM) algorithm [52] to estimate the three mixtures of normal, pattern, and event, a GMM is learned from the RPS training dataset.

3. Apply a MAP classifier to determine the decision threshold for classifying the three mixtures. Then for a data point x_t classified as a pattern point, the time sequence $(x_{t\text{-}(Q\text{-}1)\tau\text{-}1}, x_{t\text{-}(Q\text{-}2)\tau\text{-}1}, x_{t\text{-}1})$ is embedded into the phase space as vector $\mathbf{x}_{t-1}^{p}$.
4. Apply a SVM to classify the temporal pattern structure in the phase space based on the true event occurrence defined by the event function. This second stage classification finds the decision function that separates the "false" and "true" patterns predictive of events with high confidence.

We will discuss each of these steps in turn in the following subsections. The overall procedure of this method is illustrated in Figs. 3.1 and 3.2.

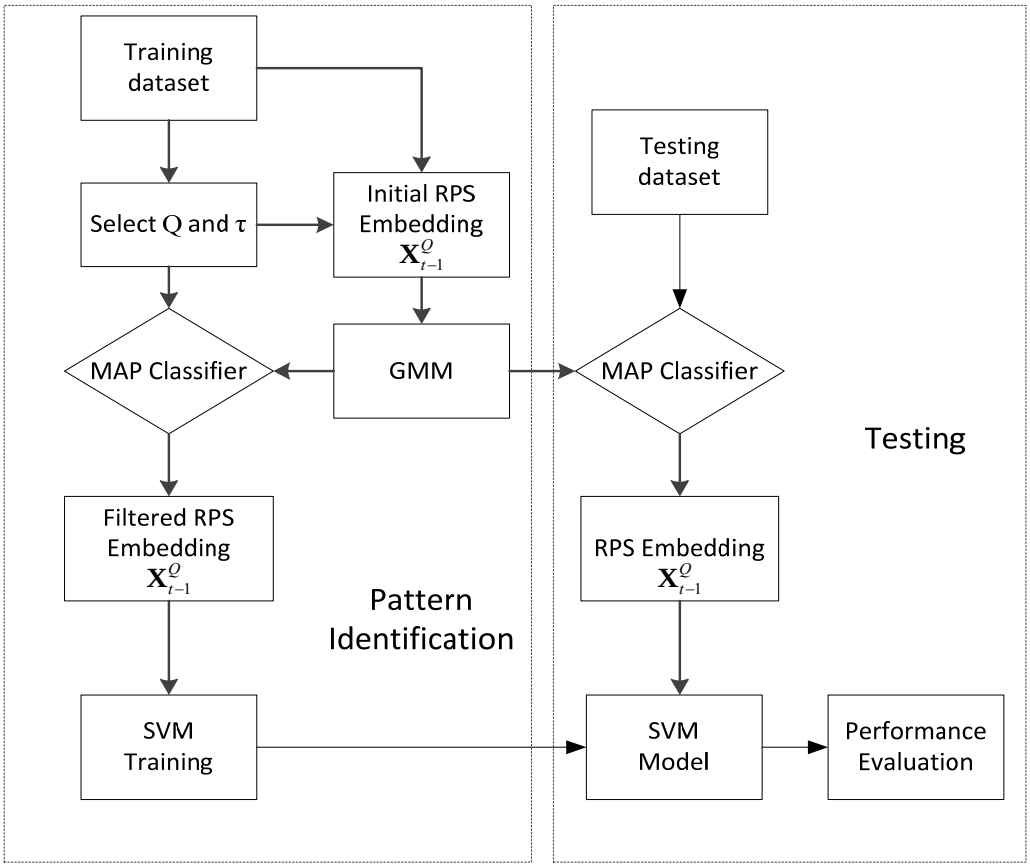

Figure 3.1: A detailed block diagram of the GMM-SVM.

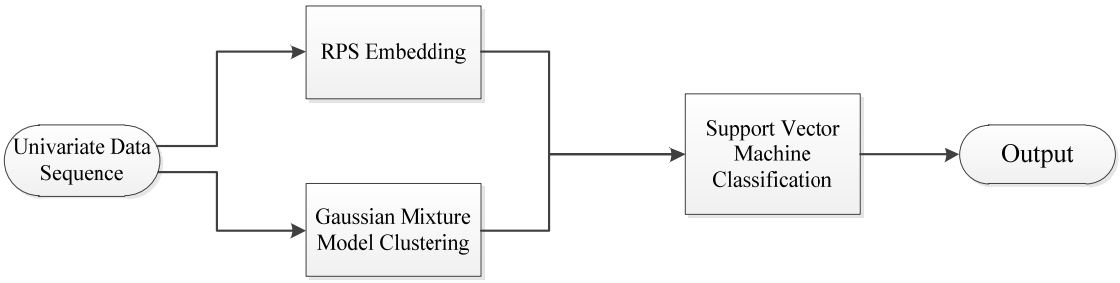

Figure 3.2: An overview diagram of the GMM-SVM.

3.3 Initial Parameter Estimation for Embedding

The time delay τ can be calculated by using the first minimum of a mutual information function that provides the quantitative characteristics of spatial patterns in phase space [24]. The minimum of the mutual information function was found to be effective in estimating the time delay. Given a data sequence and time delay τ, the mutual information is computed by:

$$M(x_t, x_{t-\tau}) = \sum_{i,j} p_{ij}(\tau) \ln\left(\frac{p_{ij}(\tau)}{p_i p_j}\right), \tag{3.2}$$

where p_i is the probability that x_t falls in the *i*th interval, and $p_{ij}(\tau)$ is the joint probability that x_t falls into the *n*th interval and $x_{t-\tau}$ falls into the *j*th interval.

The dimension Q of the RPS is determined using a false nearest-neighbors technique [25, 32]. The percentage of false nearest-neighbors changes with different choice of the embedding dimension Q. The smallest Q that gives the lowest percentage of false nearest-neighbors is selected as the optimal embedding dimension.

3.4 Probability Density Estimation Based on GMM

In this approach, DDS is considered to have three different recurring states: ω_n, ω_p, and ω_e. Data points in the system can be clustered into three categories of signals, each of them belonging to ω_n, ω_p, or ω_e, respectively. To predict events by temporal patterns, we apply a MAP algorithm [4] to preprocess the data sequence so that nonevent related data of ω_n can be filtered out. To construct a MAP algorithm, the distribution of the three categories of data in the sequence needs to be estimated. A GMM can be well suited for this goal of probability density estimation of three mixtures: ω_n, ω_p, and ω_e.

The EM algorithm [52] is used to estimate the parameters of the GMM:

$$\hat{p}(\omega_i) = \frac{1}{n}\sum_{k=1}^{n} \hat{p}\left(\omega_i \middle| x_k, \hat{\theta}\right), \tag{3.6}$$

$$\hat{\mu}_i = \frac{\sum_{k=1}^{n} p\left(\omega_i \middle| x_k, \hat{\theta}\right) x_k}{\sum_{k=1}^{n} p\left(\omega_i \middle| x_k, \hat{\theta}\right)}, \tag{3.7}$$

$$\hat{\Sigma}_i = \frac{\sum_{k=1}^{n} P\left(\omega_i \middle| x_k\right)\left(x_k - \hat{\mu}_i\right)\left(x_k - \hat{\mu}_i\right)^T}{\sum_{k=1}^{n} P\left(\omega_i \middle| x_k\right)}, \tag{3.8}$$

where

$$\hat{p}\left(\omega_i \middle| x_k, \hat{\theta}\right) = \frac{p(x_k \mid \hat{\mu}_i, \hat{\Sigma}_i)\hat{p}(\omega_i)}{\sum_{j=1}^{c} p(x_k \mid \hat{\mu}_i, \hat{\Sigma}_i)\hat{p}(\omega_j)}, \tag{3.9}$$

with $p(x_k \mid \hat{\mu}_i, \hat{\Sigma}_i) \sim \mathcal{N}(\hat{\mu}_i, \hat{\Sigma}_i)$. One advantage of the GMM is that it can represent many distributions. For many cases, in the limit as $c \to \infty$, it can represent any possible distribution [4].

Given the distribution of the three states using a GMM, we can construct a MAP algorithm [4] to classify the data points as ω_n or ω_p in a given data sequence. A data point x_k is classified into one of the three categories that give the highest posterior likelihood:

$$\hat{\omega} = \arg\max_{i \in \{n,p,e\}} p(x_k \mid \hat{\mu}_i, \hat{\Sigma}_i) \hat{p}(\omega_i) \ , \tag{3.5}$$

where each component distributed as $\mathcal{N}(\mu_i, \Sigma_i)$ with a mean μ_i and covariance matrix Σ_i, and $p(\omega_i)$ is the marginal distribution for the i^{th} component of the mixtures, with constraint $\Sigma_i p(\omega_i) = 1$. By using a MAP classifier, we can determine the Bayesian optimal thresholds that separate the three states. The classification applied here provides a filtering of data points before phase space embedding is applied.

3.5 Pattern Classification in the RPS using a SVM

Previous RPS-based methods typically use a clustering method together with an event function to separate the event and nonevent patterns. These methods are based on an unsupervised method by grouping phase space patterns into a number of clusters. This approach performs well for data sets in which temporal patterns share close similarity, that is, those that are homogenous, but it may not work well when the underlying system has multiple time-varying dynamics that can generate volatile temporal patterns with low degree of similarities, that is, heterogeneous data sets. Under such circumstances, well-

defined spherical clusters may not exist in a phase space embedding. In the SVM-GMM approach, a first stage MAP classifier filtering based on the GMM reduces the number of data points embedded in the phase space effectively. As a result, this approach typically results in sparse and separable embeddings, well suited for a SVM classification task.

Given phase space embedding $\mathbf{x}_i$, $i=1,2\ldots,l$, with an event function $g(\mathbf{x}_i)\in\{+1,-1\}$ and slack variables ξ_i, we can apply a support vector classifier [33] to the phase space. The support vector classifier solves the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w},b,\mathbf{x}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \quad & g(\mathbf{x}_i)(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_i)+b) \geq 1-\xi_i \\ & \xi_i \geq 0, \ \ i=1,\ldots l \end{aligned} \tag{3.10}$$

By using the Karush–Kuhn–Tucker (KKT) condition, we can reformulate problem (3.10) as a dual problem:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j g(\mathbf{x}_i)g(\mathbf{x}_j)K(\mathbf{x}_i,\mathbf{x}_j), \\ \text{subject to} \quad & \sum_{i=1}^{l}\alpha_i g(\mathbf{x}_i) = 0 \\ & 0 \leq \alpha_i \leq C, \ \ i=1,\ldots l, \end{aligned} \tag{3.11}$$

where $K(\mathbf{x}_i,\mathbf{x}_j)\equiv\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$ is the kernel function. Therefore, according to this formulation, the phase space vector $\mathbf{x}^p$ is classified as ω_p if

$$\sum_{i=1}^{l}\alpha_i g(\mathbf{x}_i)K(\mathbf{x}_i^p,\mathbf{x}^p)+b>0, \tag{3.12}$$

where α_i is the solution of the dual problem.

For a general application of classification problem, a support vector classifier has more generalizability than a neural network method. This is because the optimization

method used in a SVM results in a sparse solution, which avoid the overfitting problem neural network models typically have. The following experiments can illustrate the effectiveness of using a SVM and a GMM.

3.6 Experimental Results

In this section, to show its effectiveness, the GMM-SVM method is applied to several benchmark applications: chaotic series predictions [1] in example (a)-(c) and Sludge Volume Index (SVI) prediction [10] in example (d). Examples (a) and (b) are used for illustrative and explanatory purposes and present prediction performances of our GMM-SVM method. In examples (c) and (d), a comparative study is conducted and the performance of the GMM-SVM method is evaluated compared with the baseline TSDM method [2].

Chaotic time series are defined by the state equations [1, 3] with adjustable parameters. Since chaotic data systems are defined strictly according to state equations, the underlying systems are therefore deterministic instead of completely random. Thus, chaotic series are predictable for a limited number of iterations or time horizon. By identifying temporal patterns associated with events of interest, the underlying deterministic relationship between temporal patterns and events can be revealed and then used for future predictions. For each example, three thousand data points are simulated, with the first 2000 used as a training set. The remaining 1000 data points are used as a testing set for validation.

Example (a): Consider the Henon map defined by (3.13) and illustrated in Figs. 3.3 and 3.4. Denoting by σ_x^2 the variance of the x component of Henon map, in Fig. 3.4, 10%

Gaussian white noise $\varepsilon \sim \mathcal{N}(0, \sigma_x^2 / 10)$ corrupts the Henon map. The Henon map is defined by

$$\begin{cases} \dfrac{dx}{dt} = -x^2 + by + a \\ \dfrac{dy}{dt} = x. \end{cases} \tag{3.13}$$

For example, we take $a = 1.4$ and $b = 0.3$.

In this explanatory example, the x component of Henon map is chosen as the target series, and the goal is to predict that in the next time step, x exceeds 1.0. The event characterization function therefore is defined as

$$g(\mathbf{x}_t) = \begin{cases} +1 & x_{t+1} > 1.0 \\ -1 & x_{t+1} <= 1.0. \end{cases} \tag{3.14}$$

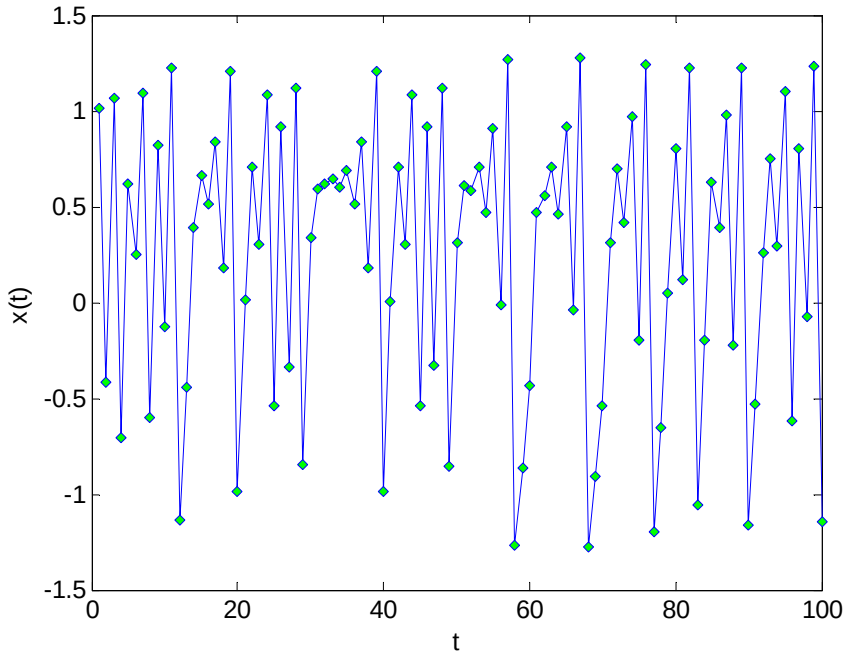

Figure 3.3: $x(t)$ plot of a Henon map without noise.

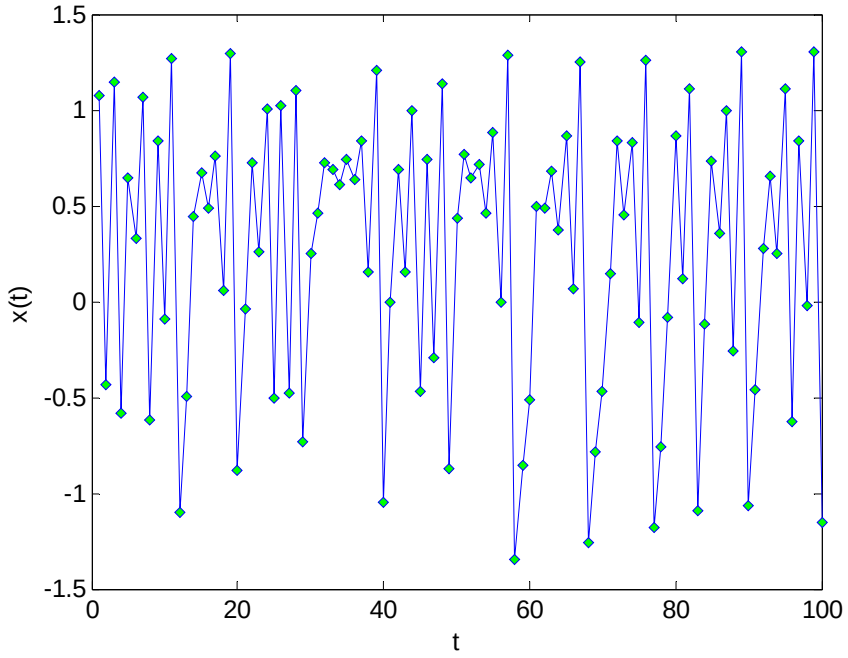

Figure 3.4: $x(t)$ plot of a Henon map with 10% Gaussian white noise added.

To determine the time delay for the embedding, the minimum mutual information method [24] is applied to the Henon map. By using the mutual information method, we can explore the dependence of $x_{t+\tau}$ on the value of x_t. A high mutual information value indicates a strong dependence, whereas a low mutual information value suggests a low dependence. Figure 3.5 presents the value of the mutual information between delayed x values under different values of the time delay τ. The mutual information fluctuates and decreases as the time delay increases from 1 to 10. This means that the correlation between delayed embeddings vanishes as the time delay increases. Several local minima are located at $\tau = 2, 4, 7, 8$. As Fraser and Swinney [24] suggest, the first local minima at $\tau = 2$ is preferred to later local minima to avoid over-estimation.

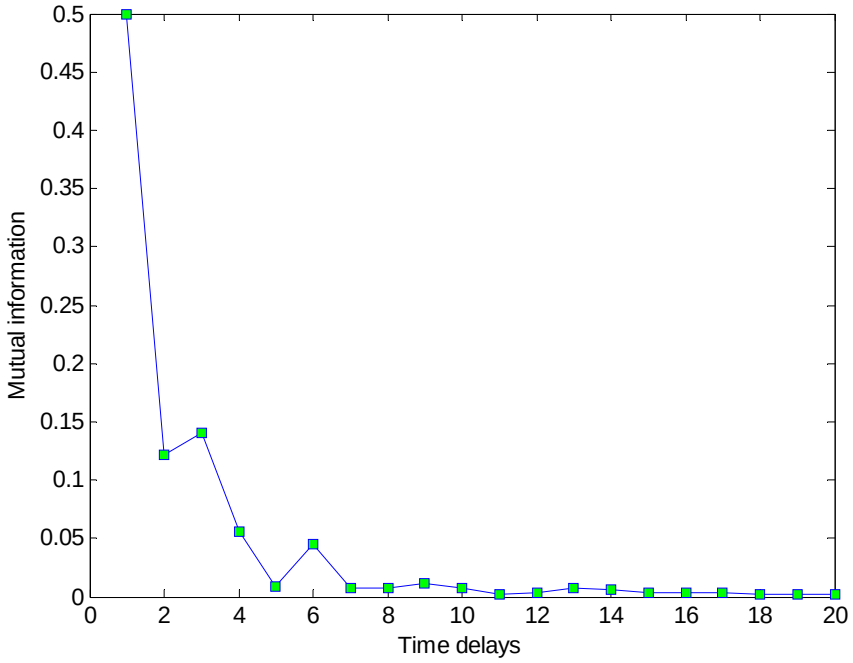

Figure 3.5: Mutual information of the Henon map with different time delays τ.

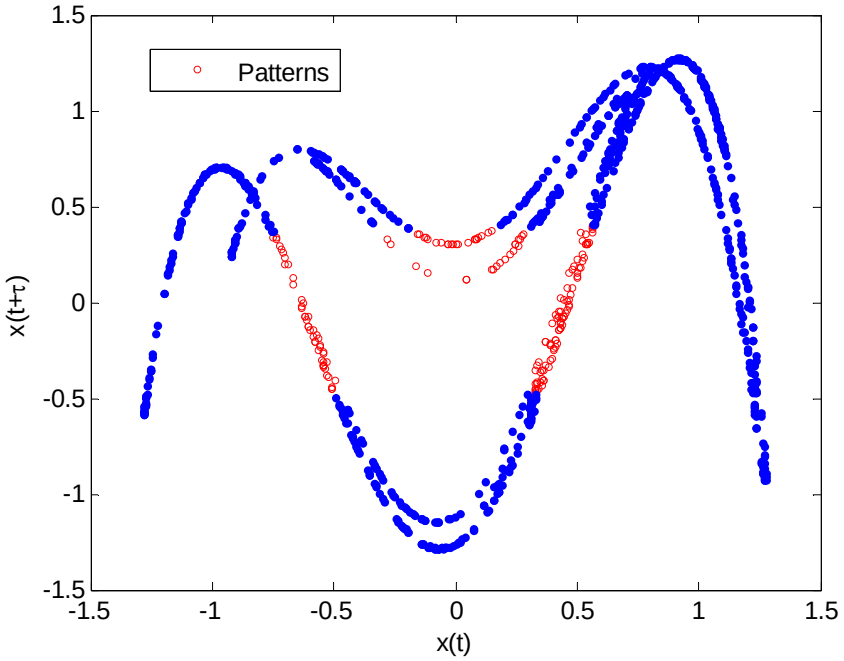

Figure 3.6: The trajectory of $x(t)$ in the Henon map and patterns (with a time delay $\tau = 2$)

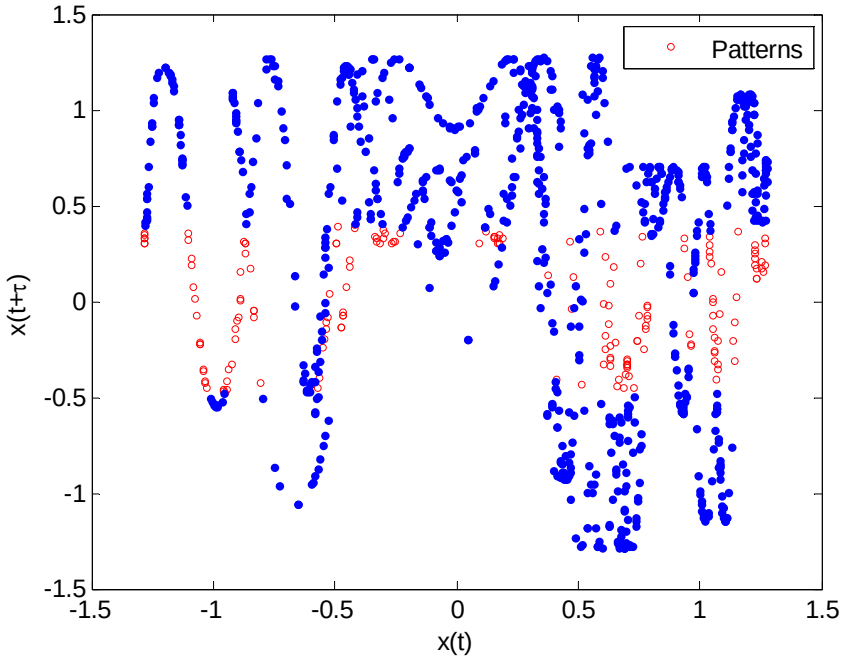

Figure 3.7: The trajectory of $x(t)$ in the Henon map and patterns (with a time delay $\tau = 5$)

Fig. 3.6 highlights the predictive temporal patterns, i.e., embeddings with event function values +1 in Eqn. (3.14) and displays the trajectory of the Henon map $x(t)$ without noise in the phase space with a time delay $\tau = 2$. Fig. 3.8 shows the phase space embedding under a time delay $\tau = 5$. It shows that a higher choice of time delay results in overlap between patterns of interest and unrelated data points. Therefore, for the purpose of pattern classification and events predictions based on predictive patterns, the embedding time series with a larger τ value does not lead necessarily to better separability between event-related and nonevent-related phase space points. Thus, a large time delay is not always suitable for the goal of finding predictive patterns.

We have discussed the case when we assume no noise is added into the chaotic signal. In the following, we consider the case illustrated in Fig. 3.4 when 10% Gaussian

white noise is added into the signal. Under this setting, we simulate the Henon map with added noise. In Fig. 3.8, we present the phase space embeddings of the Henon map with 10% Gaussian white noise. As expected, in some areas in the phase space with added noise, the event-related patterns now overlap with low eventness points. Event-related patterns now have a range of (–0.6, 0.3) for the x_t dimension and (–0.5, 0.2) for the x_{t+1} dimension.

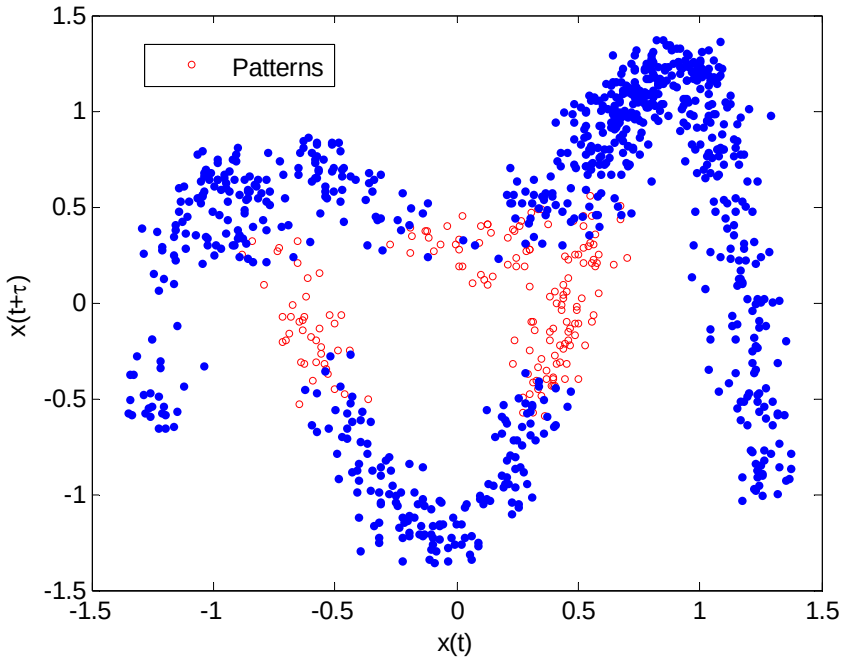

Figure 3.8: The trajectory of the x dimension in the Henon map with 10% Gaussian white noise added (time delay $\tau = 2$).

The underlying theory discussed in [1, 20, 21] guarantees that a sufficiently high dimensional embedding in the RPS can describe the dynamics of a system. However, in practice, the false nearest-neighbor method or cross-validation method is more popular for choosing the embedding dimension. One reason is that although twice the box-

counting dimension is an obvious lower bound of the dimension, the theory in [20, 21] does not state explicitly what an appropriate upper bound for an embedding dimension should be. For our goal of pattern identification and event prediction, today's computing power enables us to use a cross-validation method running multiple scenarios to estimate the best choice for the embedding dimension. Tables 3.1 and 3.2 and Fig. 3.9 display the prediction accuracy with respect to the value of the embedding dimension Q.

Q	True Positive	True Negative	False Positive	False Negative	Acc (%)
1	104	755	70	68	86.16
2	140	819	3	31	96.58
3	142	812	6	29	96.46
4	142	806	9	28	96.24
5	143	801	10	27	96.23
6	141	794	13	29	95.70
7	141	784	20	28	95.07
8	137	781	19	32	94.74
9	136	774	23	32	94.30
10	131	771	22	37	93.86

Table 3.1: Event prediction accuracy of the Henon map (no noise) with different values of the embedding dimension Q.

Q	True Positive	True Negative	False Positive	False Negative	Acc (%)
1	112	747	75	63	86.16
2	135	797	22	39	93.86
3	136	798	17	38	94.44
4	137	792	20	36	94.31
5	133	791	17	40	94.19
6	128	788	17	44	93.76
7	125	787	15	46	93.73
8	123	781	17	48	93.29
9	123	778	17	47	93.37
10	125	771	20	45	93.24

Table 3.2: Event prediction accuracy of the Henon map (10% noise) with different values of the embedding dimension Q.

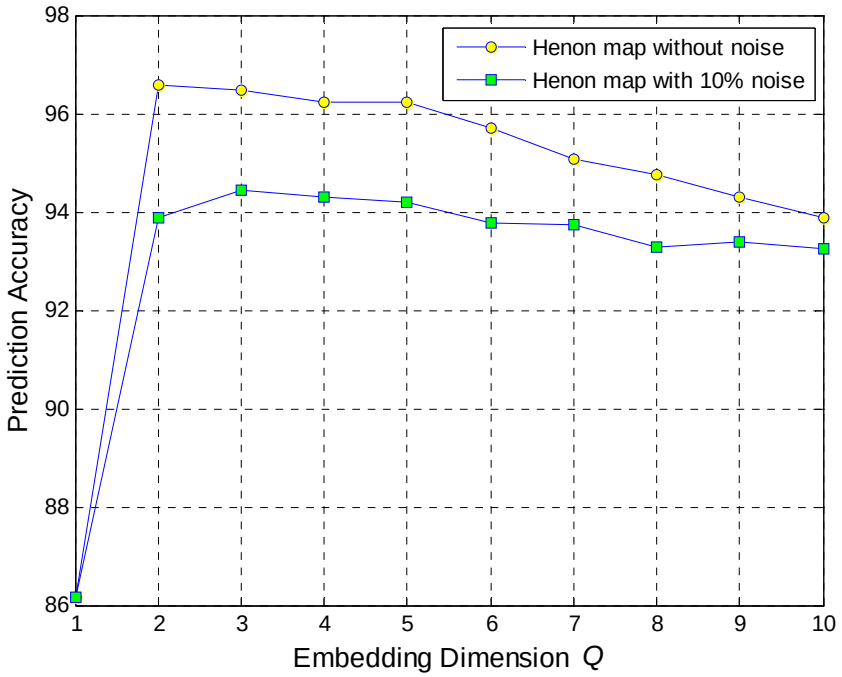

Figure 3.9: Prediction accuracy of events in the Henon map with different values of the embedding dimension Q.

In Fig. 3.9, prediction accuracy increases when embedding dimension Q increases from 1 to 2 and decrease as Q increases from 2 to 10. This observation suggests that the choice of embedding dimension can affect the prediction performance. A high embedding dimension does not necessarily result in higher prediction accuracy. Comparing the performance of the GMM-SVM method in the noise-free and in the 10% additive Gaussian noise cases, we can observe from Tables 3.1 and 3.2 that prediction accuracy is consistently higher in the noise-free case than that in the additive noise case. However, the difference between these two situations is within 0 to 2.5%, and the GMM-SVM method still can achieve prediction accuracies above 92% in the noisy setting.

Example (b): The second example is the Rossler map as illustrated in Figs. 3.10 and 3.11. In Fig. 3.10, the Rossler map is not corrupted by noise, and in Fig. 3.11, the Rossler map is corrupted by 10% Gaussian white noise. The Rossler map is defined by

$$\begin{cases} \dfrac{dx}{dt} = -y - z \\ \dfrac{dy}{dt} = x + ay \\ \dfrac{dz}{dt} = z(x - c) + b. \end{cases} \tag{3.15}$$

For example, we take $a = 0.3, b = 0.5,$ and $c = 5$.

For the Rossler map, the z component in the system state variables is chosen as the target series. In this simulation experiment, the goal is to predict that in the next time step, z exceeds 10. The event characterization function therefore is defined as:

$$g(\mathbf{x}_t) = \begin{cases} +1 & z_{t+1} > 10 \\ -1 & z_{t+1} <= 10. \end{cases} \tag{3.16}$$

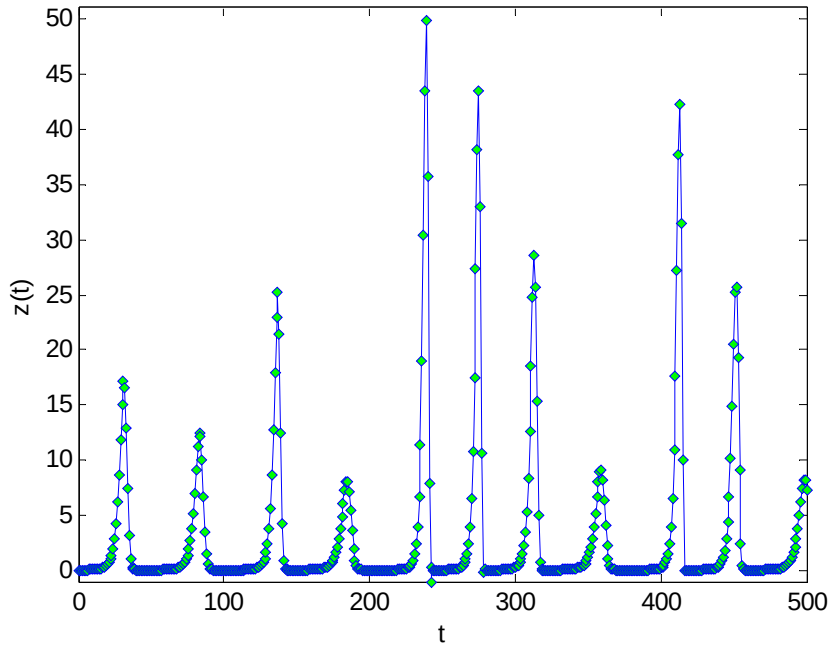

Figure 3.10: The z component of a Rossler map without noise.

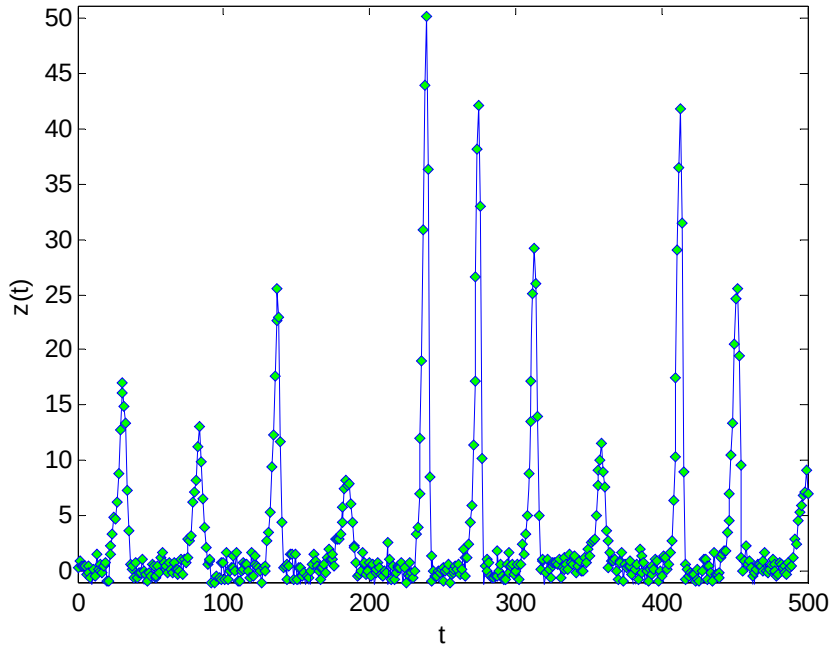

Figure 3.11: The z component of a Rossler map with 10% Gaussian white noise.

From Figs. 3.10 and 3.11, the Rossler map takes several time steps in an upward direction to reach an event, whereas in the case of Henon map shown in Figs. 3.3 and 3.4, the data sequence has a relatively higher level of fluctuation or volatility with respect to time. Similarly, we apply the minimum mutual information method to the Rossler map. Fig. 3.12 displays the value of the mutual information between the delayed x values under different values of time delay τ. The mutual information decreases consistently as the time delay increases from 1 to 5 and increases after reaching local minima at $\tau = 5$. This indicates that the correlation between delayed embeddings is at its smallest at a time delay of 5. Therefore, the time delay for the Rossler map is chosen as 5.

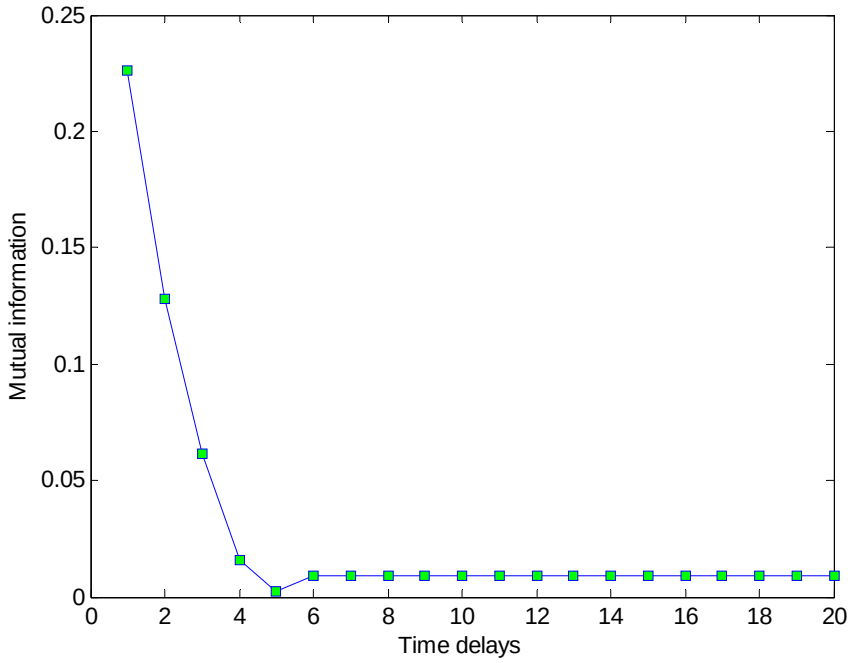

Figure 3.12: The mutual information for the Rossler map with different time delays.

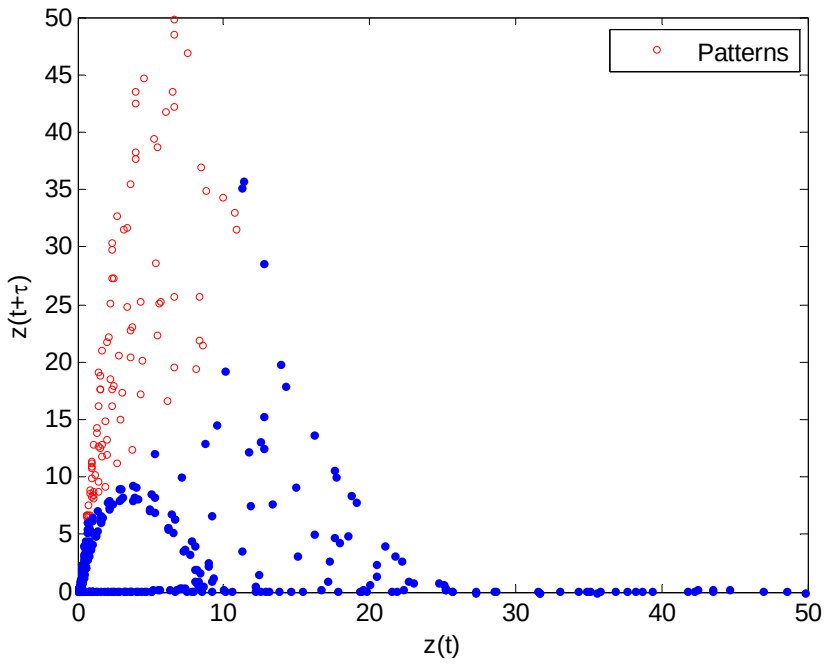

Figure 3.13: The trajectory of the z dimension of the Rossler map and its associated patterns (time delay $\tau = 5$).

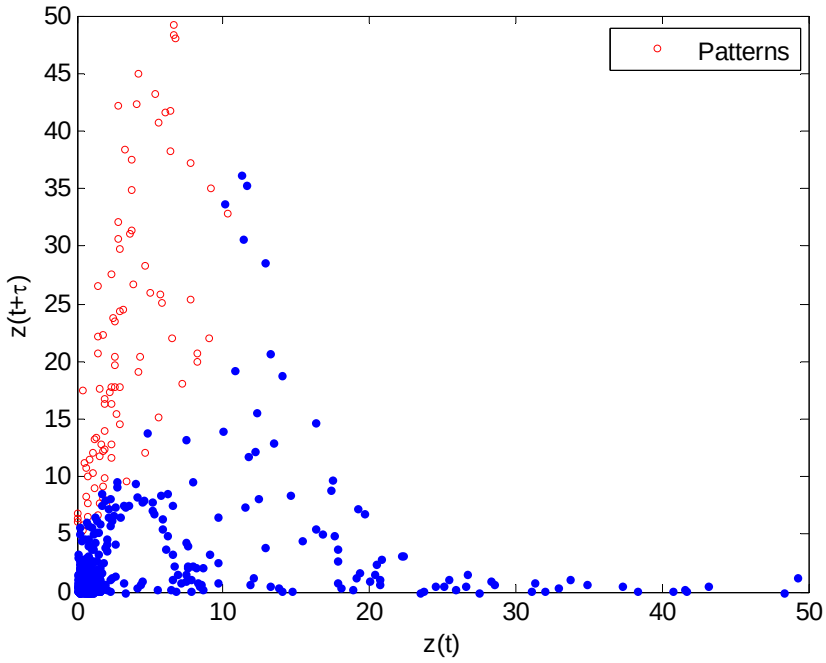

Figure 3.14: The trajectory of the z dimension of the Rossler map with 10% Gaussian white noise and its associated patterns (time delay $\tau = 5$)

Fig. 3.13 highlights the predictive temporal patterns, i.e., embeddings with event function values +1 in Eqn. (3.16) and displays the trajectory of the Rossler map z series without noise in the phase space with a time delay of 5. The predictive patterns generally have high value of x_{t+1}, ranging from 0.1 to 1.5. For pattern classification purpose, this means that there is good class separability between the predictive temporal patterns and nonevent-related embeddings in the phase space.

Fig. 3.14 shows the phase space embeddings of the z dimension in the Rossler map with added noise. In this case, there are small overlapped regions between the event related patterns and nonevent related points in phase space.

In this experiment, we simulated 3000 data points and used the first 2000 data as a training set. The remaining 1000 data points were used as a testing set for validation. Again, we apply a cross-validation method running multiple scenarios to estimate the best choice of embedding dimension. Tables 3.4 and 3.5 and Fig. 3.15 display the prediction accuracy with respect to the value of the embedding dimension *Q*.

Q	True Positive	True Negative	False Positive	False Negative	Acc (%)
1	74	418	15	16	94.07
2	89	432	0	1	99.81
3	89	431	0	1	99.81
4	89	421	0	1	99.80
5	85	415	0	1	99.80
6	85	409	0	0	100.00
7	83	408	1	2	99.39
8	79	408	1	6	98.58
9	80	400	2	5	98.56
10	81	391	2	3	98.95

Table 3.3: The event prediction accuracy of the Rossler map (without noise) with different values of the embedding dimension Q.

Q	True Positive	True Negative	False Positive	False Negative	Acc (%)
1	71	527	16	18	94.62
2	84	535	4	5	98.57
3	82	531	3	7	98.39
4	82	527	2	7	98.54
5	79	522	2	6	98.69
6	77	513	2	8	98.33
7	76	509	2	9	98.15
8	77	502	3	8	98.14
9	76	497	1	9	98.28
10	76	489	1	8	98.43

Table 3.4: The event prediction accuracy of the Rossler map (with 10% Gaussian white noise) with different values of embedding dimension Q.

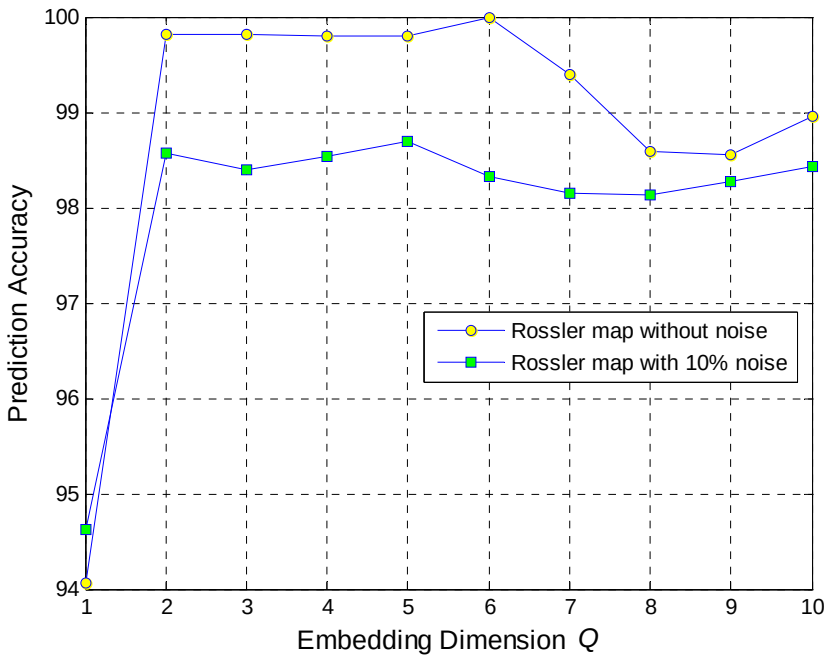

Figure 3.15: The prediction accuracy of events in a Rossler map with different Q.

From Tables 3.3 and 3.4 and Figure 3.15, prediction accuracy increases when embedding dimension Q increases from 1 to 5 and decreases as Q increases from 5 to 10. This observation again suggests that the choice of embedding dimension can affect the prediction performance, and a high embedding dimension does not necessarily result in higher prediction accuracy. Comparing the performance of the GMM-SVM method in the noise-free and in the 10% additive Gaussian noise cases for Rossler map, we can observe that the difference of prediction accuracy between these two cases is within 0 to 2.5%, and our GMM-SVM method still achieves prediction accuracies above 98% in the noisy setting.

Now that we have illustrated how the GMM-SVM method can be applied to two basic chaotic time series, in the following two examples (c) and (d) we compare the

GMM-SVM method in two more complex datasets, Lorenz map [3], and Sludge Volume Index (SVI) [10].

Example (c): The third example is the Lorenz map as illustrated in Fig. 3.16. The Lorenz map is defined by

$$\begin{cases} \dfrac{dx}{dt} = \sigma(y - x) \\ \dfrac{dy}{dt} = x(\rho - z) - y \\ \dfrac{dz}{dt} = xy - \beta z. \end{cases} \tag{3.17}$$

In the simulation, the Lorenz time series is generated by setting the initial values of $x_0 = 0,\ y_0 = -0.01,$ and $z_0 = 0.01,$ and the parameters of $\sigma = 9,\ \rho = 25,$ and $\beta = 3.3$. For the Lorenz map, the x component in the system state variables is chosen as the target series. In this simulation experiment, the goal is to predict that in the next time step, x exceeds 11. The event characterization function therefore is defined as:

$$g(\mathbf{x}_t) = \begin{cases} +1 & \text{if } x_{t+1} > 11.0 \\ -1 & \text{if } x_{t+1} <= 11.0. \end{cases} \tag{3.18}$$

Similar to examples (a) and (b), the time delay was estimated as $\tau = 6$, and the embedding dimension as $Q = 3$. Fig. 3.17 illustrates the temporal patterns of the Lorenz map.

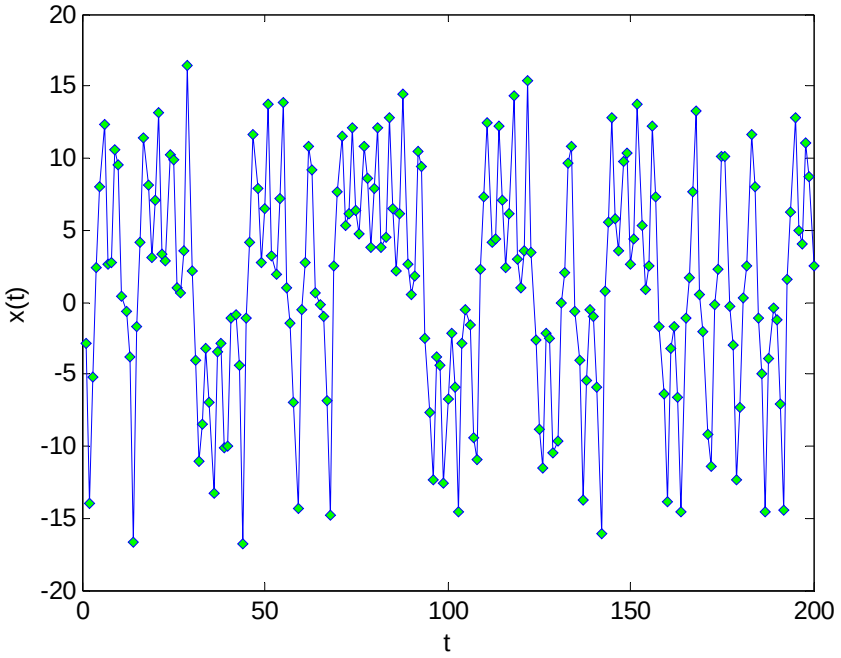

Figure 3.16: Time series generated by a Lorenz map.

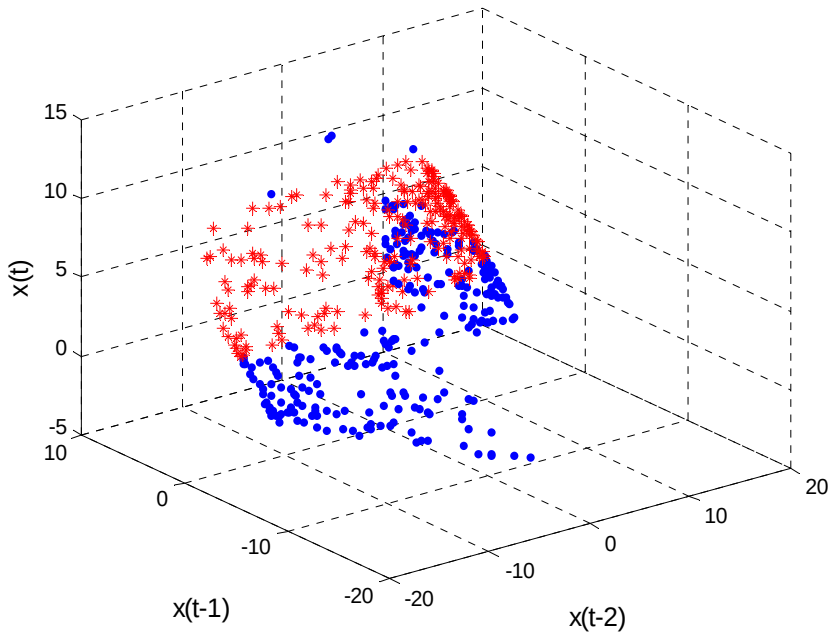

Figure 3.17: Temporal patterns of a Lorenz map in a 3D phase space.

	Predicted as events	Predicted as nonevents
Actual events	True Positive = 49	False Negative = 5
Actual nonevents	False Positive = 3	True Negative = 436

Table 3.5: The test results of the GMM-SVM method for the Lorenz map.

	Training Set		Test Set	
	GMM-SVM	TSDM	GMM-SVM	TSDM
Prediction Accuracy	99.56%	99.86%	89.09%	77.35%
True Positive Rate	87.45%	62.35%	84.48%	58.52%

Table 3.6: Results of the prediction performance comparison.

The results of the GMM-SVM method and the TSDM method proposed by Povinelli and Feng [2] are presented in Tables 3.5 and 3.6. The prediction accuracy measure and true positive rate measure are defined as:

Prediction Accuracy = (True Positives + True Negatives) / (True Positives + True Negatives + False Positive + False Negative),

True Positive Rate = True Positives / (True Positives + False Negative).

Table 3.6 presents the results of the GMM-SVM method and the previous TSDM method. By comparing two methods, it can be observed that for event prediction in the Lorenz map, the GMM-SVM method outperforms the TSDM method by 25.96% in true positive rate and 11.74% in overall prediction accuracy in the testing phase. In addition, the testing prediction performance of the GMM-SVM is consistent with the training results, whereas the TSDM method shows significantly lower accuracy predicting events in the testing dataset due to overfitting.

Example (d): The fourth example is the Sludge Volume Index (SVI) series illustrated in Fig. 3.18. SVI is an empirical measurement for the sludge-bulking problem. A Sludge-bulking anomaly is one of the primary causes of water treatment plant failure, as the abnormal bulking conditions can result in exceeding discharge permit limits. If sludge bulking occurs, it can generate a high SVI value. Efforts have been made to design a monitoring system for the sludge-bulking conditions of water treatment plants to provide early alerts. The cause of this problem has been studied from a biological point of view, but due to its complexity, a deterministic causal relationship has not been formulated. The modeling approaches applied include stochastic models and artificial neural systems [10]. With data from 2003 to 2008 provided by a Chicago water treatment company, the first three years data are used as a training data set, and the remaining three years data are used as a testing set for validation.

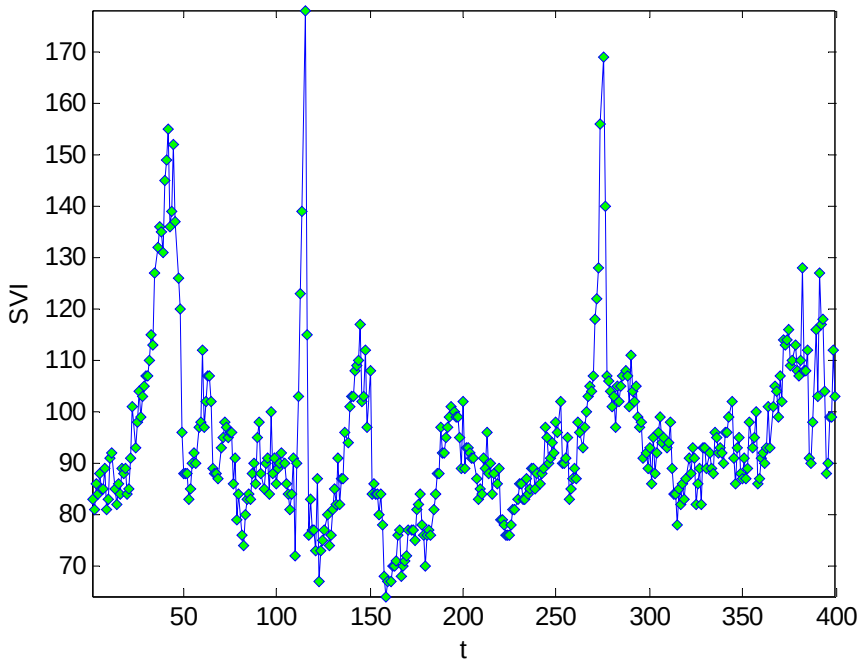

Figure 3.18: An example of a SVI time series.

Denoting the SVI daily sequence as S_t, the goal is to predict that within the next three time steps SVI, exceeds 150. The event function is defined as

$$g(\mathbf{x}_t) = \begin{cases} +1 & \max\{S_{t+1},\ldots,S_{t+3}\} > 150.0 \\ -1 & \max\{S_{t+1},\ldots,S_{t+3}\} <= 150.0 \end{cases}. \tag{3.19}$$

The time delay was estimated as $\tau = 5$ and the embedding dimension as $Q = 4$. Fig. 3.19 shows the SVI daily sequence S_t from 2002 to 2008 with its identified temporal patterns. By examining Fig. 3.19, it is tempting to consider applying a simple threshold rule, for example $S_t >= 120$, to forecast the sludge bulking events. However, this simple rule can result in significantly more false alarms, which can trigger substantial costs associated with shutting down the water treatment plant and other related costs.

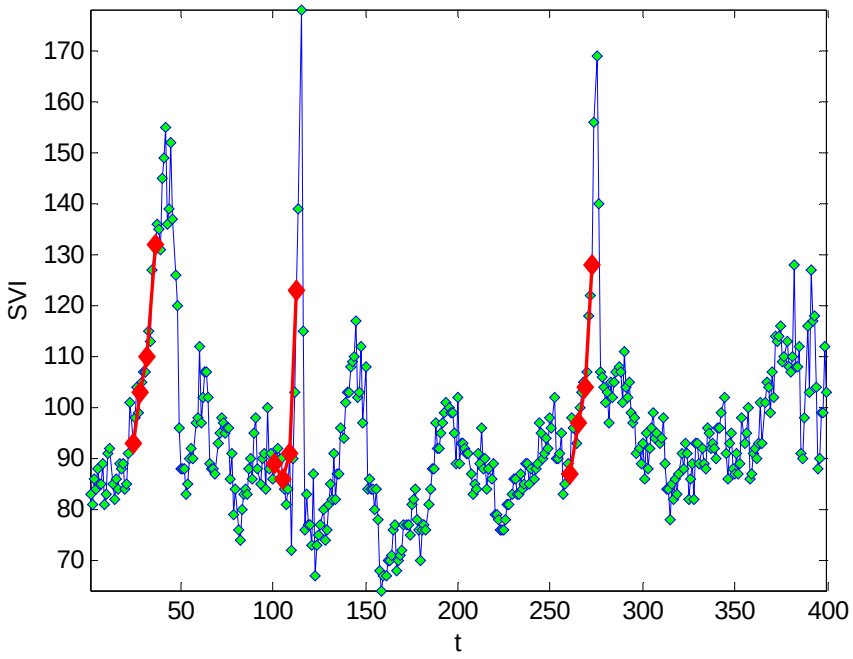

Figure 3.19: Identified temporal patterns of SVI.

	Training Set		Test Set	
	GMM-SVM	TSDM	GMM-SVM	TSDM
Prediction Accuracy	90.25%	75.32%	81.25%	65.73%
True Positive Rate	75.35%	63.58%	70.56%	51.28%

Table 3.7: Prediction performance comparison between the GMM-SVM and TSDM.

In Fig. 3.19, SVI testing series are plotted with the diamond boxes marking the temporal patterns. The last point of the pattern is the predicting point, which indicates a high probability of the sludge bulking. Table 3.7 presents the results of the GMM-SVM method and the TSDM method. By comparing the two methods, it can be observed that for SVI prediction, the GMM-SVM method outperforms the TSDM method by 15.52% in overall prediction accuracy and 19.25% in true positive rate in the testing phase.

In addition, by examining the temporal patterns plotted in Fig. 3.19, we see that the patterns that relate to the events are not consistent in their structures or shapes. Instead, the temporal patterns are time-evolving and therefore not obvious to capture by traditional approaches. The results demonstrate that the GMM-SVM approach can be applied in a monitoring system to provide early alerts for the potential sludge-bulking problems in water treatment plants.

In summary, the GMM-SVM method provides a discriminative approach that uses both by GMM and SVM techniques to classify temporal patterns that are predictive of events in a dynamic data system. Experiments compared with the baseline TSDM method show that the new method yields significant improvements in the prediction accuracy of future events.

CHAPTER 4 IDENTIFICATION OF TEMPORAL PATTERNS IN MULTIVARIATE DATA SEQUENCES

In this chapter, a new approach is presented to identify multivariate temporal patterns predictive of future events of interest in a multivariate dynamic data system. The new Multivariate Reconstructed Phase Space (MRPS) method is based on the multivariate RPS transformation, data categorization, and the nonlinear optimization.

One major limitation of traditional RPS approaches is that temporal patterns typically are assumed to exist only in the event sequence. Although this assumption may be true for some problems, there are applications where multivariate data sequences can result in a higher event identification rate or better performance. For example, in monitoring a patient with severe cardiovascular conditions, besides measuring the electrocardiography signals, other measurements, such as blood pressure and body temperature, are also monitored constantly to track the patient's overall conditions.

Theoretically, if a Dynamic Data System (DDS) has two partially correlated components, e.g., a scalar output component $y(t)$ and a control component $u(t)$, the temporal dynamics in $y(t)$ can only describe the overall system partially unless dynamics from both $y(t)$ and $u(t)$ are included. Therefore, compared with a univariate modeling approach, modeling the system dynamics using multivariate data sequences can provide more insights and a better understanding of the overall system.

4.1 Event Functions for Multivariate Data Systems

As discussed in Chapter 3, a dynamic data system can be considered to have three different states, e.g., a normal state, a pattern state, and an event state. Consequently, data

points in the system can be clustered into three categories of signals, a normal state, a pattern state, and an event state, respectively. In the following, we denote ω_n, ω_p, and ω_e as three class labels for the normal state, the pattern state, and the event state, respectively.

Consider *p*-variable data sequences with the *j*th variable sequence defined as:

$$X_j = \{x_{tj},\ t = 1,2,\ldots N,\ j = 1,2,\ldots, p\}, \tag{4.1}$$

where *t* is the time index, and *N* is the total number of observations. For each time instance *t*, a multidimensional observation vector $\mathbf{x}_t = (x_{t1}, x_{t2},\ldots,\ x_{tp})$ is measured. In this multivariate system, a target event sequence needs to be specified. This target event sequence contains the events or critical points that we want to predict by detecting the hidden patterns. The target sequence Y_e is denoted by

$$Y_e = \{y_t,\ t = 1,2,\ldots, N\}, \tag{4.2}$$

where the subscript *e* indicates the sequence containing events of interest. Based on the target sequence, we can define an event function using a multiple step forward threshold function:

$$g(\mathbf{x}_t) = \begin{cases} +1 & \text{if } y_t <= c \text{ and } \max\{y_{t+1},\ldots, y_{t+k}\} > c \\ -1 & \text{if } y_t <= c \text{ and } \max\{y_{t+1},\ldots, y_{t+k}\} <= c \\ 0 & \text{if } y_t > c, \end{cases} \tag{4.3}$$

where *k* is the time horizon, and c is the predefined threshold of the event.

In the training stage, with the event function in Eqn. (4.3), each multidimensional observation $\mathbf{x}_t$ is associated with a label, which takes values in $\{+1, -1, 0\}$, representing the true occurrence of the event in the *k* step horizon. Each observation vector $\mathbf{x}_t$ is assigned a label and a category:

1. Predictive multivariate temporal patterns are data points labeled +1 and categorized as class ω_p,
2. Non-predictive points are data points labeled –1 are and categorized as class ω_n.
3. Event points are data points labeled 0 and categorized as class ω_e.

Under this formulation, a multidimensional vector $\mathbf{x}_t$ can be classified as ω_p, ω_e, or ω_e according to Eqn. (4.3). A data sequence can be considered as a mixture of the three classes of variables representing the three recurring states: ω_n, ω_p, and ω_e.

For most applications, a primary focus is to predict events based on temporal patterns when the underlying system is not in an event state. Thus, for identification of predictive patterns in the training stage, our focus is on the classification of two categories of data that are ω_p and ω_n. Then, in the testing stage, predictions can be made based on the classifier trained from the training data set.

4.2 The Multivariate Reconstructed Phase Space

In Chapter 3, we considered the univariate phase space embedding and the method for parameter estimation. In this chapter, univariate phase space embedding is extended into the multivariate case, which is more suitable for applications with more than one variable in the system. The idea of mining temporal patterns using RPS approach in a multi-dimensional system was first proposed in [56] by embedding multiple data sequences using the same time-delay and embedding dimension. In this research, the MRPS method employs a different formulation by estimating the time-delay and the embedding dimension individually for each variable sequence. In other words, the time-delay and the embedding dimension can be different for different data sequences.

Given a multivariate data system as in Eqn. (4.1), with *p* explanatory variables and one target variable, all $p \times N$ observations from $t = 1$ to N can be represented by an observation matrix **X**:

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1N} \\ x_{21} & x_{22} & \ddots & x_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{p1} & x_{p2} & \cdots & x_{pN} \end{bmatrix}_{p \times N} . \tag{4.4}$$

In addition to the explanatory variable sequences in the observation matrix, it is advantageous to include the event sequence in the observation matrix so that the causal relationship between events and temporal patterns within an event sequence can be identified. Hence, the new augmented observation matrix, including the event sequence, can be expressed as:

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_p \\ Y_e \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1N} \\ x_{21} & x_{22} & \ddots & x_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{p1} & x_{p2} & \cdots & x_{pN} \\ y_1 & y_2 & \cdots & y_N \end{bmatrix}_{(p+1) \times N} . \tag{4.5}$$

To construct the MRPS, the time delay τ_j and the embedding dimension Q_j need to be estimated for each variable sequence $X_j = \{x_{jt}, t = 1, ..., N\}$. Similar to the univariate case in Section 3.6, the minimum mutual information method is applied to estimate the multiple time-delays. Recalling the univariate case, the cross-validation method was used to select the embedding dimension. However, in the multivariate case, the estimation of multiple embedding dimensions by cross validation is not efficient for a large dataset with multiple input variables. Thus, we apply the false nearest-neighbor method [25, 42]

to estimate the embedding dimension for each variable sequence assuming no correlation between each embedding dimension.

After selecting the time delay τ_j and the embedding dimension Q_j for each sequence defined in Eqn. (4.1), the resulting embedding for each sequence becomes

$$\mathbf{x}_{jt} = [x_{jt} \quad x_{j,t-\tau_j} \quad \cdots \quad x_{j,t-(Q_j-1)\tau_j}] \,, \tag{4.6}$$

where $t = 1,2,\ldots N$, $j = 1,2,\ldots, p+1$. The multivariate phase space embedding then can be constructed as:

$$\boldsymbol{X}_t = (\mathbf{x}_{1t}, \mathbf{x}_{2t}, ..., \mathbf{x}_{jt,} \ldots, \mathbf{x}_{pt}, \boldsymbol{y}_t) \,, \tag{4.7}$$

at each time *t*, where $\mathbf{x}_{jt}$ represents the phase space embedding for the *j*th variable x_j given in Eqn. (4.1) with the time delay τ_j and dimension Q_j at time *i*. The dimension Q of the multivariate embedding is the sum of each embedding dimension Q_j, $Q = \sum_j Q_j$.

4.3 Similarity Measure of Temporal Patterns

One of the challenges when embedding a data sequence into the RPS is that similar temporal patterns may fall into different regions, whereas they are supposed to be categorized as the same patterns. There are many potential reasons for this problem depending on the specific application. One explanation is that the difference in the starting values between temporal patterns is large, which could be because there is a trend in the data sequence. As a result, this means that the Euclidean distance between those patterns in the RPS also can be large even if they have the same temporal structure. However, these patterns representing the same dynamics should be considered as the same category of patterns. For example, in the existing embedding approaches discussed

in [2,3], similar temporal patterns can fall in different regions in the RPS if the difference in the starting values of these patterns is sufficiently large. To address this problem, we consider a phase space embedding on differenced data series such a transform usually results in a detrended representation of the data sequence, removing any trend in the data sequence. By the theorem of filtered delay embedding prevalence in Sauer et al. [21], given a linear constant transformation, the resulting filtered delay mapping also gives a valid representation of the underlying dynamic system. Moreover, the linearly transformed embedding can preserve the same local dynamics as the regular embedding of the original dataset. Therefore, by eliminating the effects of a trend, a linearly transformed embedding can provide a better representation of the data system since the similarity of transformed embeddings does not depend on the initial starting values. As a result, the Euclidean distance in the newly transformed RPS is capable of measuring the similarity between temporal patterns correctly.

Definition: Consider two temporal pattern embeddings in a RPS with embedding dimension Q and time delay τ, and starting values x_{t0} and x_{t1}, respectively,

$$\mathbf{x}_1 = \{x_{t0}, x_{t0+\tau}, \cdots x_{t0+(Q-1)\tau}\}, \mathbf{x}_2 = \{x_{t1}, x_{t1+\tau}, \cdots x_{t1+(Q-1)\tau}\}.$$

The similarity measure of two temporal patterns is defined by

$$d(\mathbf{x}_1, \mathbf{x}_2) = \sum_{i=1}^{Q} (x_{t0+(Q-i)\tau} - x_{t1+(Q-i)\tau} - d_0)^2, \tag{4.8}$$

where $d_0 = (x_{t0} - x_{t1})$ is the initial difference of the two embeddings.

This new similarity measure captures the similarity of the temporal structure independent of the initial starting values. Based on this new similarity measure, we introduce a new phase space constructed by applying a linear transformation on the

original RPS embedding. The resulting phase space has the property that the Euclidean distance in the new phase space is equivalent to the distance defined by the similarity measure in Eqn. (4.8). Furthermore, since the transformation is linear, according to the Filtered Delay Embedding Prevalence Theorem [21], the resulting embedding in this new space gives a faithful representation of the underlying dynamic system.

In the following, we will show that by applying a transformation on a regular embedding $\{x_t, x_{t+\tau}, \cdots x_{t+(Q-1)\tau}\}$, the Euclidean distance in the new space is equivalent to the similarity measure defined in Eqn. (4.8) between the two sampled data sequences.

Lemma 4.1: *Given the dimension of the embedding of Q, there exists a transformation* $\phi(\mathbf{x}): R^Q \to R^{Q-1}, \phi(\mathbf{x}) = \mathbf{A}^T\mathbf{x}$, *such that the Euclidean distance between any two embeddings* $\phi(\mathbf{x}_1)$ and $\phi(\mathbf{x}_2)$ *in the transformed phase space is equivalent to the similarity measure* $d(\mathbf{x}_1, \mathbf{x}_2)$ *defined in Eqn. (4.8).*

Proof: The similarity measure in Eqn. (4.8) can be rewritten as

$$\sum_{i=1}^{Q} \{(x_{t0+(Q-i)\tau} - x_{t0}) - (x_{t1+(Q-i)\tau} - x_{t1})\}^2 . \tag{4.9}$$

For each *i*, the first term in Eqn. (4.9) can be decomposed into a summation of differences:

$$x_{t0+(Q-i)\tau} - x_{t0} = (x_{t0+(Q-i)\tau} - x_{t0+(Q-i-1)\tau}) + (x_{t0+(Q-i-1)\tau} - x_{t0+(Q-i-2)\tau}) + ... + (x_{t0+\tau} - x_{t0}) .$$

Therefore, Eqn. (4.9) can be rewritten as

$$\sum_{i=1}^{Q-1} \{\sum_{j=1}^{Q-i} [(x_{t0+j\tau} - x_{t0+(j-1)\tau}) - (x_{t1+j\tau} - x_{t1+(j-1)\tau})]\}^2 . \tag{4.10}$$

Denoting differenced embeddings as $\nabla\mathbf{x}_{t+(Q-1)\tau} = (x_{t+(Q-1)\tau} - x_{t+(Q-2)\tau}, \ldots, x_{t+\tau} - x_t)$, by some algebraic operations, Eqn. (4.10) can be written in a quadratic form,

$$d(\mathbf{x}_1, \mathbf{x}_2) = \nabla(\mathbf{x}_1 - \mathbf{x}_2)^T \mathbf{P} (\nabla(\mathbf{x}_1 - \mathbf{x}_2)), \tag{4.11}$$

where

$$\mathbf{P} = \begin{bmatrix} Q-1 & Q-2 & \cdots & 1 \\ Q-2 & Q-2 & \cdots & 1 \\ \vdots & \vdots & \ddots & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix},$$

Since **P** is a symmetric positive definite matrix, we can find a Cholesky decomposition of matrix **P**, such that $\mathbf{P} = L^T L$. Eqn. (4.11) then becomes

$$d(\mathbf{x}_1, \mathbf{x}_2) = (L\nabla(\mathbf{x}_1 - \mathbf{x}_2))^T L\nabla(\mathbf{x}_1 - \mathbf{x}_2). \tag{4.12}$$

Defining a linear transformation:

$$\mathbf{A} = L\nabla, \tag{4.13}$$

we can obtain a new embedding vector,

$$\phi(\mathbf{x}) = \mathbf{A}^T \mathbf{x}. \tag{4.14}$$

As a result, the Euclidean distance in the new space $\phi(\mathbf{x})$ is equal to the similarity measure in Eqn. (4.8).

Q.E.D.

Since the operations L and ∇ are linear matrices for a given Q, the transformation $\mathbf{A} = L\nabla$ is also a matrix. This means that according to the Filtered Delay Embedding Prevalence Theorem [14], the new embedding can also preserve the same dynamics as a regular embedding does.

4.4 Optimization Algorithm

Another important component in the new MRPS method is the optimization of the classifier. The existing optimization methods in [2,3] showed effectiveness in some applications. However, these methods typically employ a nonconvex objective function and result in multiple local minima. To overcome this difficulty, heuristic rules have been applied based on application-specific domain knowledge to limit the solution space in a certain range.

4.4.1 Objective Function and Classifier Design

In the new MRPS method, to achieve robustness and stability in the optimization procedure, the following convex exponential loss function is proposed:

$$\min_{\boldsymbol{\beta}} \{L(g(\mathbf{x}), f(\mathbf{x}))\} = \min_{\boldsymbol{\beta}} \sum_{t=1}^{N} \exp(-g(\mathbf{x}_t) f(\mathbf{x}_t)) + \frac{\eta}{2} \|\boldsymbol{\beta}\|^2 \ , \tag{4.15}$$

where $L(g, f(x))$ is the objective function defined as the weighted exponential sum of an event function and a classification result for each $\mathbf{x}_t$. The classifier $f(\mathbf{x}_t)$ in Eqn. (4.15) is defined as:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \beta_i \exp\left(-\frac{\|\phi(\mathbf{x}) - \phi(\mathbf{x}_i)\|^2}{\sigma^2} \right) + \alpha_1 \varphi(\mathbf{x}) + \alpha_0 \ , \tag{4.16}$$

where $\varphi(\mathbf{x}) = \log(p(\omega_p \mid \mathbf{x}) / p(\omega_n \mid \mathbf{x}))$, $\boldsymbol{\beta} = (\beta_1 \ldots \beta_n, \alpha_1, \alpha_0)$. The first term in the classifier denotes the similarity of the differenced data series represented by the kernel estimation in the phase space. The second term denotes the Gaussian mixture log-likelihood score. This formulation considers both the local temporal dynamics of the data sequence and the statistical interpretation given by a Gaussian Mixture Model (GMM).

The objective function in Eqn. (4.15) takes the form of a regularized objective function with a penalty term placed on the coefficients of the classifier $f(\mathbf{x}_t)$. This objective function belongs to a more general class of regularization problems [33] that take the form:

$$\min_{f\in\mathcal{H}}\left[\sum_{i=1}^{N} L(y_i, f(x_i)) + \eta J(f)\right], \tag{4.17}$$

where $L(y_i, f(x_i))$ is a loss function, $J(f)$ is the penalty function, and $\mathcal{H}$ is a space of functions.

In fact, many well-known existing classifications and regression methods can be viewed as a particular optimization method with a specially designed of loss function.

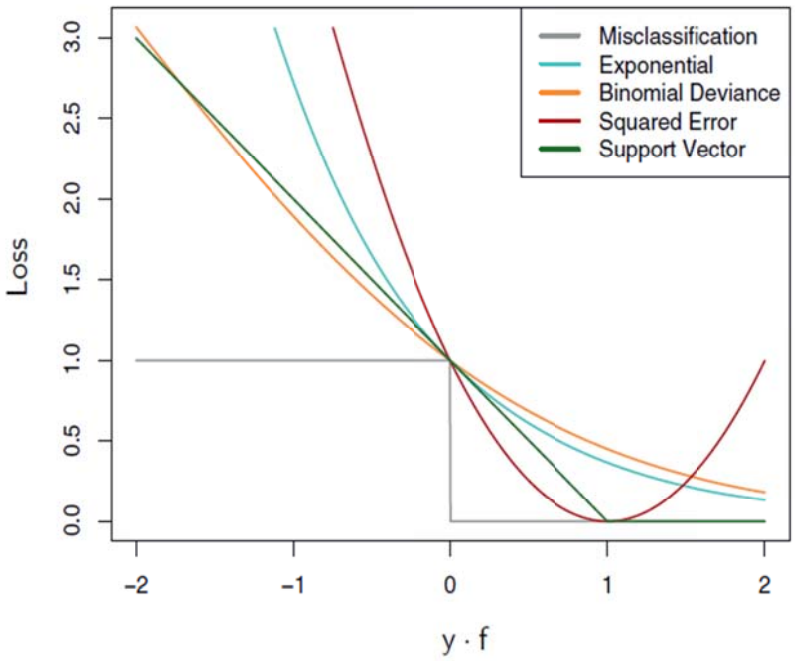

Figure 4.1: A comparison of different five loss functions.

The following is a short list of examples:

1. The squared error loss function: $(y-f(x))^2$,
2. Support Vector Machine (SVM) loss function [34]: $L(y,f(x))=[1-yf(x)]_+$, and
3. Logistic regression loss function [4]: $L(y,f(x))=\log[1+e^{-yf(x)}]$.

Fig. 4.1 shows a comparison of five different categories of loss functions.

The parameters β_i and α_1 in Eqn. (4.16) determine the constraints of the temporal dynamics and the statistical correlations between features, respectively. In general, large weights in β_i indicate the events are more likely relevant to the local dynamics of the system, whereas a large weight in α_1 suggests the cause of events is correlated more with the feature variables than with the temporal dynamics.

4.4.2 Classifier Optimization

The parameters β_i, α_1, and α_0 can be determined by minimizing the objective function $L(g(\mathbf{x}),f(\mathbf{x}))$ defined in Eqn. (4.15). The gradient of the objective function $\nabla L(g(\mathbf{x}),f(\mathbf{x}))$ with respect to α_i is given by:

$$\frac{\partial L(g,f(\mathbf{x}))}{\partial \beta_i}=\sum_{j=1}^{N}-g(\mathbf{x}_j)\exp\left(-\frac{\left\|\phi(\mathbf{x})-\phi(\mathbf{x}_i)\right\|^2}{\sigma^2}\right)\exp(-g(\mathbf{x}_j)f(\mathbf{x}_j))+\eta\beta_i \ . \quad (4.18)$$

The gradient of the objective function with respect to β_1 is expressed by:

$$\frac{\partial L(g,f(\mathbf{x}))}{\partial \alpha_1}=\sum_{i=1}^{N}-g(\mathbf{x}_i)\varphi(\mathbf{x}_i)\exp(-g(\mathbf{x}_i)f(\mathbf{x}_i))+\eta\alpha_1 \ . \quad (4.19)$$

The gradient of the objective function with respect to β_0 is:

$$\frac{\partial L(g,f(\mathbf{x}))}{\partial \alpha_0}=\sum_{i=1}^{N}-g(\mathbf{x}_i)\exp(-g(\mathbf{x}_i)f(\mathbf{x}_i))+\eta\alpha_0 \ . \quad (4.20)$$

Given the gradient of the objective function, a second-order quasi-Newton method is used to search for the optimal coefficients $\hat{\boldsymbol{\beta}} = \left(\hat{\beta}_1 \ldots \hat{\beta}_n, \hat{\alpha}_1, \hat{\alpha}_0\right)$ to minimize the cost function in Eqn. (4.15). Using a Taylor expansion, the function $L(\boldsymbol{\beta}_{k+1})$ can be approximately by

$$L(\boldsymbol{\beta}_{k+1}) = L(\boldsymbol{\beta}_k) + \nabla L(\boldsymbol{\beta}_k) s_k + \frac{1}{2} s_k^T H(\boldsymbol{\beta}_k) s_k \ , \tag{4.21}$$

where $s_k = \boldsymbol{\beta}_{k+1} - \boldsymbol{\beta}_k$, and $H(\boldsymbol{\beta}_k)$ is the Hessian matrix at iteration *k*.

The iterative algorithm to estimate $\boldsymbol{\beta}$ can be represented as

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k - H^{-1}(\boldsymbol{\beta}_k) s_k \, . \tag{4.22}$$

The Broyden–Fletcher–Goldfarb–Shanno (BFGS) formula [26, 27] was used to obtain $H^{-1}(\boldsymbol{\beta}_k)$, an approximation to the inverse of the Hessian matrix

$$H^{-1}(\boldsymbol{\beta}_{k+1}) = H^{-1}(\boldsymbol{\beta}_k) + \frac{l_k l_k^T}{l_k^T s_k} - \frac{H_k^{-1} s_k s_k H_k^{-1}}{s_k^T H_k^{-1} s_k} , \tag{4.23}$$

where $l_k = \nabla L(\boldsymbol{\beta}_{k+1}) - \nabla L(\boldsymbol{\beta}_k)$.

4.5 The Algorithm Design

The MRPS framework can be summarized and is illustrated in Fig. 4.2.

4.5.1 Pre-Processing Stage

1. Divide the multivariate data sequences into training and testing datasets.
2. Define the event function in Eqn. (4.3) based on the domain knowledge of a specific application.
3. Partition the training data set into three categories of data: ω_p, ω_e, and ω_e according

to the defined event function Eqn. (4.3)

4. For each variable sequence $X_j = \{x_{ji}, i = 1,..., N\}$, determine the dimension Q_j of the phase spaces and the delay τ of the temporal patterns.

4.5.2 Training Stage

5. Construct the multivariate embedding by combining all individual embeddings for each sequence into an MRPS vector $\boldsymbol{X}_i = (\mathbf{x}_{1i}, \mathbf{x}_{2i}, ..., \mathbf{x}_{ji}, \ldots, \mathbf{x}_{mi}, \mathbf{x}_{e,i})$ as in Eqn. (4.7).
6. Transform the multivariate embedding $\boldsymbol{X}_i$ by applying the operator $\mathbf{A} = L\nabla$ defined in Eqn. (4.13) to obtain the trend invariant embedding $\phi(\boldsymbol{X}_t)$.
7. Given the data partition in Step 3, construct the GMM from the training dataset, using the EM algorithm [52] to estimate mixtures of the three categories of data: normal, pattern, and event points.
8. Perform the optimization according to Eqn. (4.15) to obtain the minimizer and the corresponding classification function that can identify the predictive temporal patterns.

4.5.3 Test Stage

In the testing stage, we apply the predictive pattern classifier obtained in the training stage to predict the events in the target sequence. At each time *t*, based on the classification decision, a forecast will be made whether an event will occur.

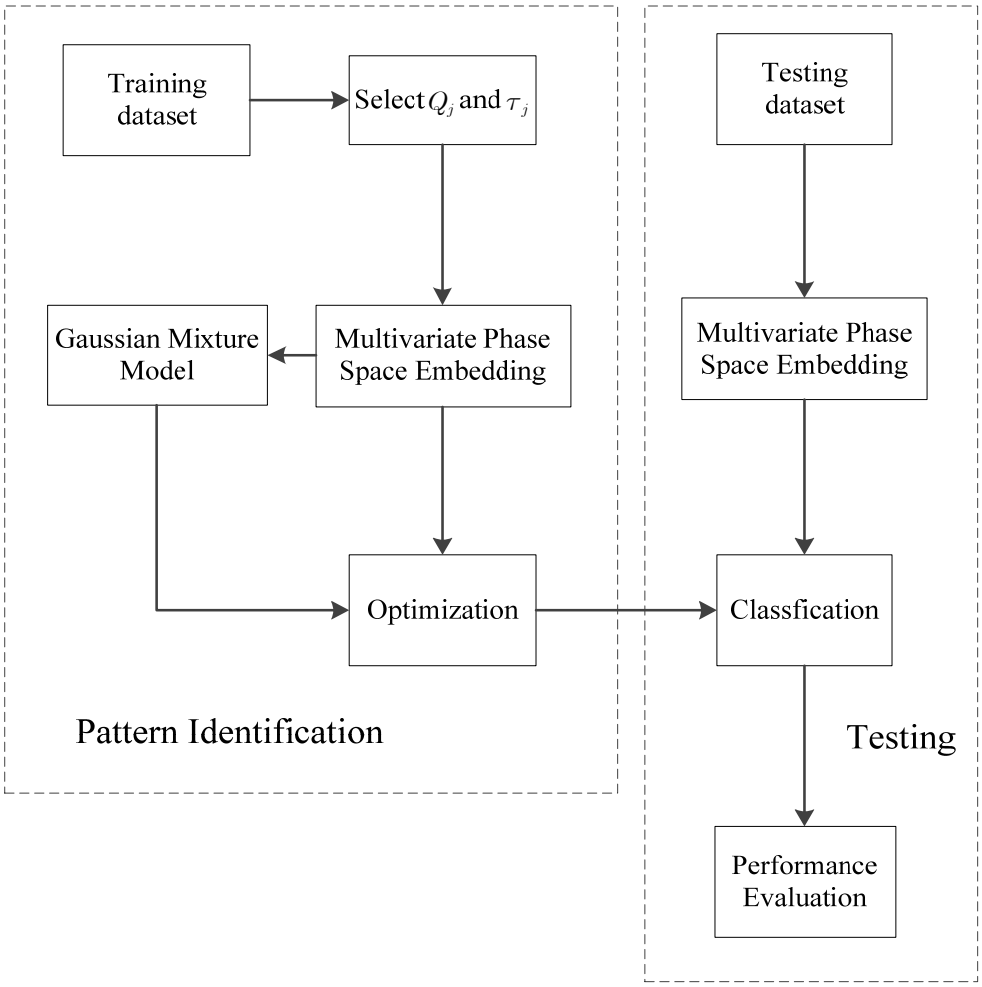

Figure 4.2: Overview of the MRPS method.

4.6 Experimental Results

In this section, similar to Section 3.6, several benchmark applications – chaotic series predictions [1] in examples (a)-(c) and Sludge Volume Index (SVI) prediction [10] in example (d) – are used as examples to demonstrate the effectiveness of the MPRS method. Examples (a) and (b) are used as for illustrative and explanatory purposes and to present prediction performances of the MRPS method. In examples (c) and (d), we

evaluate the performance of the MRPS method by comparing it to two baseline methods, one based on Artificial Neural Networks (ANN) [4] and one based on Time Series Data Mining (TSDM) [2].

Similar to Section 3.6, for each example (a)-(c), three thousand data points are simulated, with the first 2000 used as a training set. The remaining 1000 data points are used as a testing set for validation. For example (d), the Sludge Volume Index (SVI) data from 2003 to 2008, the first three years data are used as a training data set, and the remaining three years data used as a testing set for validation.

For these two types of problems, the majority of the existing literature has focused on the one-dimensional case, in which we simply use the original sequence to predict or characterize the event. In the following paragraphs, it will be shown that by using a multi-dimensional vector sequence, we can describe and characterize the DDS better and hence achieve higher event prediction accuracy.

In addition, as discussed in Section 4.2, to construct a multivariate phase space for pattern detection in a multivariate dynamic system, we estimate the time-delay and embedding dimension for each individual data sequence. We will apply the false nearest-neighbor method [25, 42] to estimate the embedding dimension of each sequence.

Example (a): The first example is the Henon map as illustrated in Figs. 4.3 and 4.4. In Fig. 4.3, noise does not corrupt the Henon map. Denoting σ_x^2 the variance of the x component of Henon map, in Fig. 4.4, 10% Gaussian white noise $\varepsilon \sim \mathcal{N}(0, \sigma_x^2/10)$ corrupts the Henon map. The Henon map is defined by:

$$\begin{cases} \dfrac{dx}{dt} = -x^2 + by + a \\ \dfrac{dy}{dt} = x. \end{cases} \tag{4.25}$$

For example, we take $a = 1.4$ and $b = 0.3$.

In this explanatory example, the x component of Henon map is chosen as the target series, and the goal is to predict that in the next time step, x exceeds 1.0. Both the x and y components are used to predict the events in a x time series. The x and y components are shown in Fig. 3.4-3.5 and Fig. 4.4-4.5, respectively. Denoting the combined embedding $\mathbf{x}_t = (x_t, x_{t-\tau_x}, ..., x_{t-(Q_x-1)\tau_x}, y_t, y_{t-\tau_y}, ..., y_{t-(Q_y-1)\tau_y})$, the event characterization function therefore is

$$g(\mathbf{x}_t) = \begin{cases} +1 & x_{t+1} > 1.0 \\ -1 & x_{t+1} <= 1.0. \end{cases} \tag{4.24}$$

By using the mutual information method, we can explore the dependence of $y_{t+\tau}$ on the value of y_t. Fig. 4.5 presents the value of the mutual information between delayed y values under different values of the time delay τ. The mutual information fluctuates and decreases as the time delay increases from 1 to 20. As suggested by Fraser and Swinney in [24], the first local minima at $\tau = 2$ is preferred to later local minima.

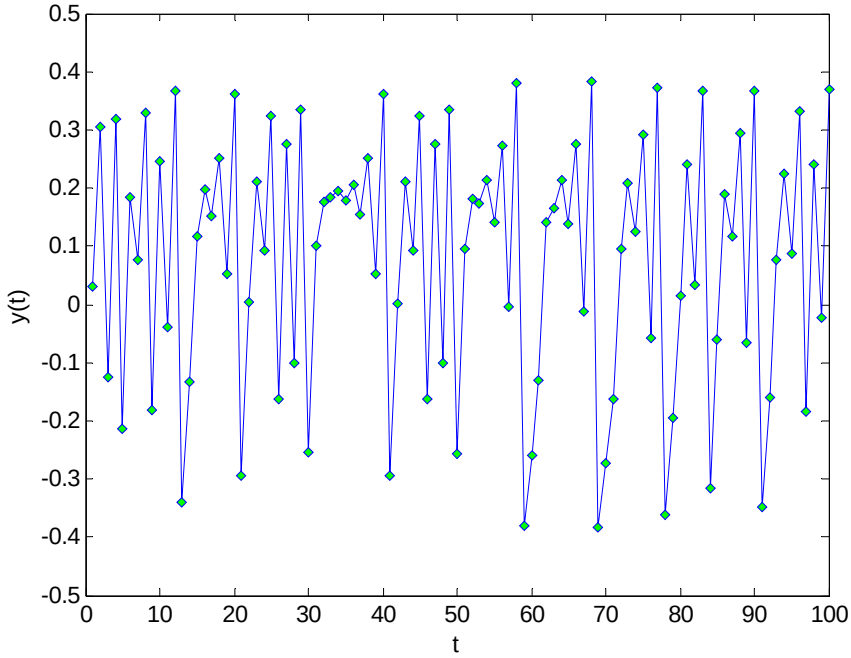

Figure 4.3: The y component of a Henon map without added Gaussian noise.

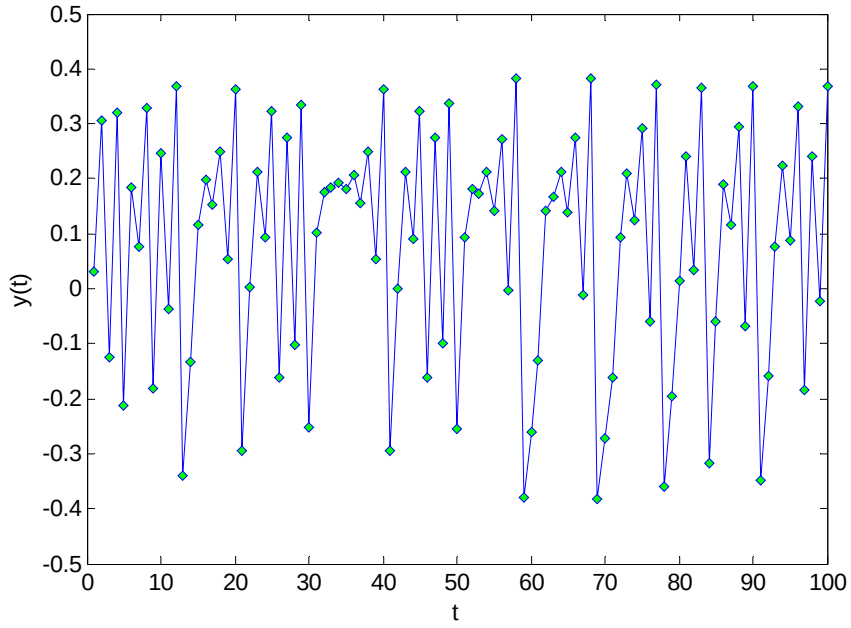

Figure 4.4: The y component of a Henon map with 10% Gaussian white noise added.

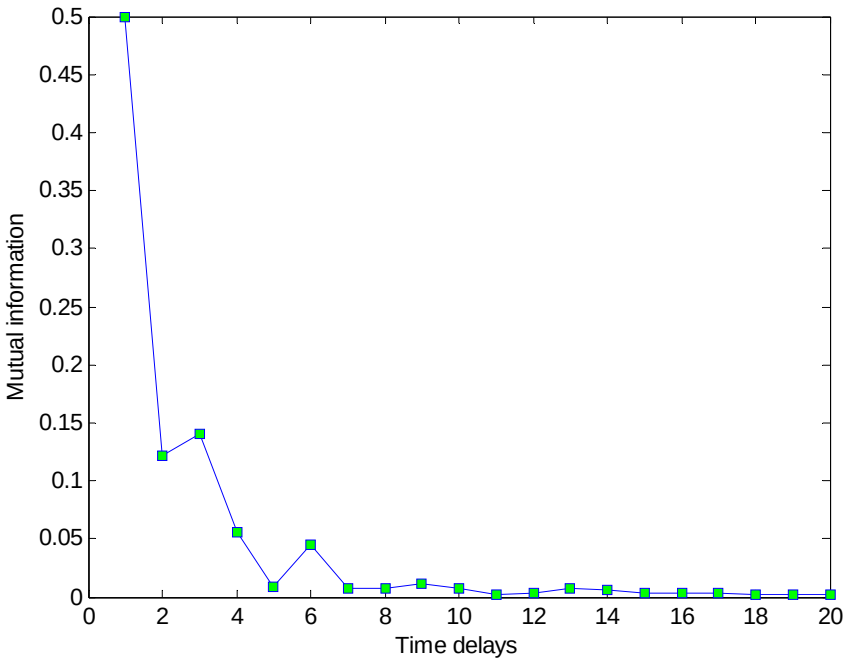

Figure 4.5: Mutual information of the y component of a Henon map with different time delays.

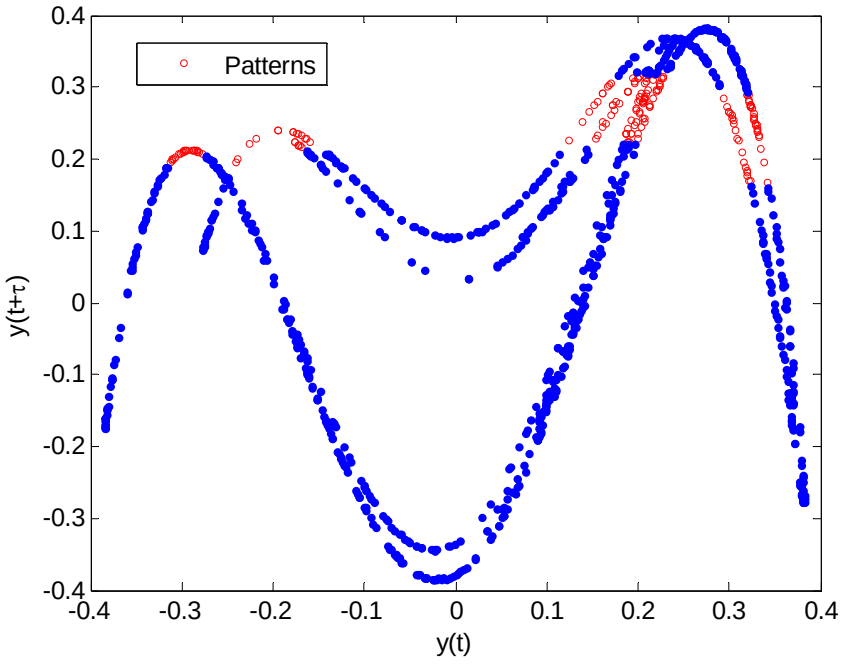

Figure 4.6: The trajectory of the y dimension in the Henon map and patterns (with time delay $\tau = 2$).

Fig. 4.6 highlights the predictive temporal patterns, i.e., embeddings with event function values +1 in Eqn. (4.24) and displays the trajectory of the Henon map without noise in a phase space with a time delay $\tau = 2$. Comparing with the patterns in the x time series in Fig. 3.5, the patterns in the y component are located in different regions in the phase space but are still clearly separable from low eventness phase space embeddings. Hence, from the y component, we can identify patterns related to events in the x sequence. By using this added information from the y component, we can achieve better accuracy compared with univariate case, as we will see in the testing stage.

The Henon map we simulated in this example is under the assumption that no noise is added into the chaotic signal. In many real systems, we do not have such an ideal situation without measurement errors. Therefore, we consider a case when 10% white noise is added into the signal. Fig. 4.7 illustrates different values of mutual information between the delayed time series in the y dimension under different values of the time delay τ.

The simulated Henon map with 10% Gaussian white noise added and a time delay $\tau = 2$ is displayed in Fig. 4.8. As expected with the added noise, the event related patterns have overlapping areas with low eventness points in the phase space. High eventness points, that is, event related patterns, are separated into several regions in a range of (–0.35, 0.35) in the y_t dimension and (0.15, 0.4) in the $y_{t+\tau}$ dimension.

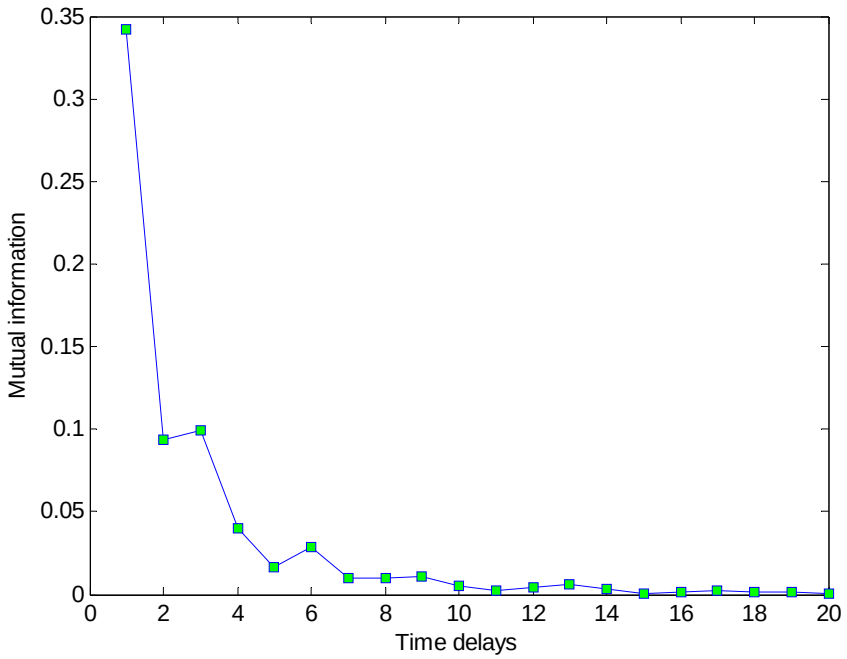

Figure 4.7: The mutual information of the y component of the Henon map (10% Gaussian noise added) with different time delays.

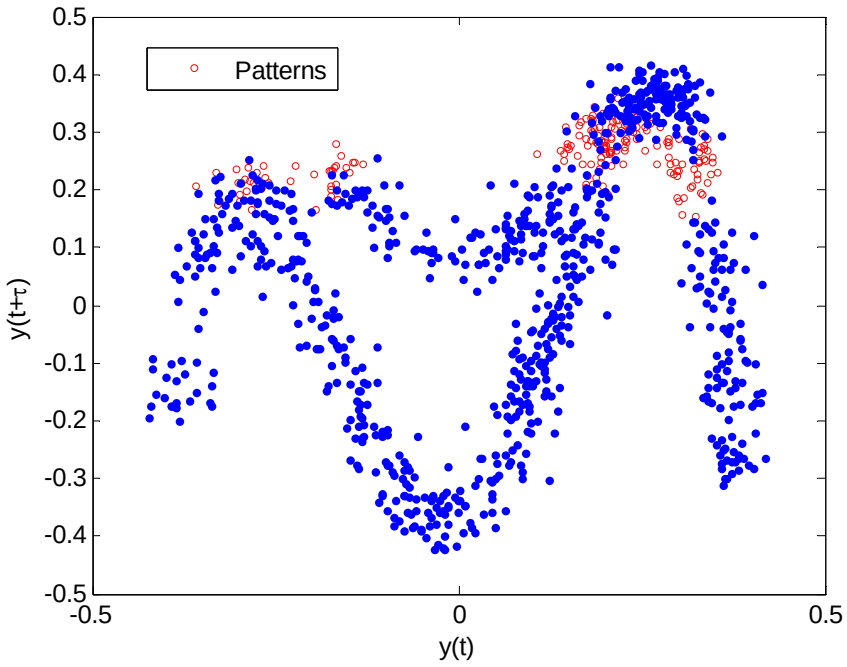

Figure 4.8: The trajectory of the y dimension in the Henon map with 10% white noise added (time delay $\tau = 2$).

In the next step, we need to determine the embedding dimension of the Henon map both in x and y dimensions. In Chapter 3, we considered the univariate case in which we applied a cross validation method to choose the best dimension Q that could generate the highest prediction accuracy. However, in the multivariate case, it is not feasible to estimate the embedding dimensions of each variable by running hundreds or thousands times of cross validations for even a medium dataset with 50–100 input variables. Instead, we will apply the false nearest-neighbor method described in [25] to estimate the embedding dimension of each sequence. Figs. 4.9 and 4.10 present the results of the false nearest-neighbor algorithm of the Henon map in the x and y dimensions, respectively.

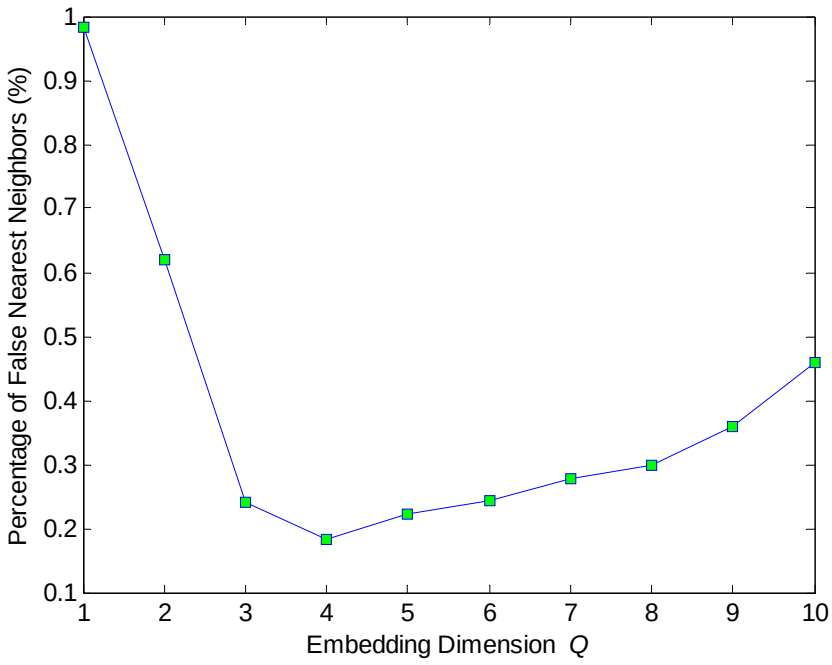

Figure 4.9: False nearest neighbors of the x component of the Henon map (10% Gaussian noise added) with different embedding dimensions.

In Fig. 4.9, the percentage of false nearest neighbors decreases from 1 to less than 0.1, as the embedding dimension increases from 1 to 4. After that, the percentage slowly increases as the dimension increases from 4 to 10. This means an embedding dimension of $Q = 4$ is a sufficient choice to embed the time series in the x dimension.

Similarly in Fig. 4.10, the percentage of false nearest neighbors decreases as the embedding dimension increases in the range of 1 to 4 and achieves a minimum at $Q = 4$. Hence, the dimension $Q = 4$ is a sufficient choice to embed the time series in the y dimension. Although for the Henon map, the embedding dimension is the same for both the x and y dimensions, this is not always the case, as we will see in the Rossler map example.

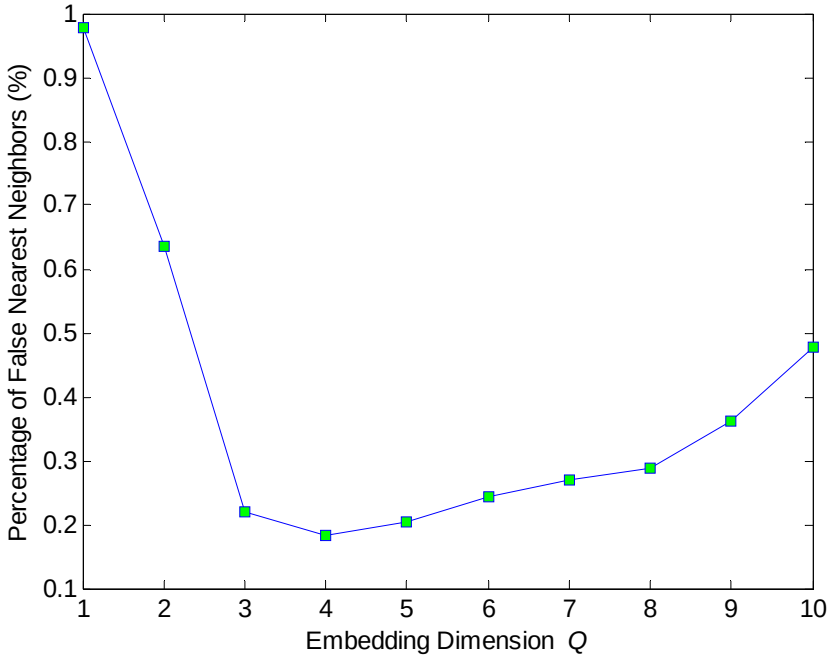

Figure 4.10: False nearest neighbors of the y component of the Henon map (10% Gaussian noise added) with different embedding dimensions.

In this multivariate case experiment, we included 3000 data points with both x and y components of the Henon map. The first 2000 data are used as a training set, and the remaining 1000 data are used as a testing set for validation. As discussed in Section 4.5, in the optimization step, a radial basis kernel function is used to construct the classifier to predict events. Choice of parameter σ in the definition of radial basis kernel function is problem specific. In practice, typically cross validation is applied to determine this tuning parameter. At the training step, multiple scenarios are tested with different values of σ. Recall that in the definition of the classifier, one component is a kernel function of the transformed multivariate phase space, and the other component is the GMM log-odds function. Simulation results with the GMM component are presented in Tables 4.1 and 4.2, whereas Tables 4.3 and 4.4 present the classification results when the GMM component is not included. The results show that the event prediction accuracy is higher when both components are included in the model than in the case without the GMM component. Thus, the results provide a good justification of including a GMM component in the design of a classifier.

σ	True positive	True negative	False Positive	False Negative	Acc (%)
0.2	167	814	4	4	99.19
0.25	167	813	5	4	99.09
0.3	167	813	5	4	99.09
0.35	167	812	6	4	98.99
0.4	166	810	8	5	98.69
0.45	166	809	9	5	98.58
0.5	165	805	13	6	98.08
0.55	167	802	16	4	97.98

Table 4.1: Event prediction accuracy of the Henon map (no noise) with different σ values.

σ	True positive	True negative	False Positive	False Negative	Acc (%)
0.2	150	785	30	24	94.54
0.25	151	786	29	23	94.74
0.3	151	787	28	23	94.84
0.35	151	788	27	23	94.94
0.4	149	786	29	25	94.54
0.45	148	785	30	26	94.34
0.5	149	784	31	25	94.34
0.55	146	780	35	28	93.63

Table 4.2: Event prediction accuracy of the Henon map (10% noise) with different σ values.

σ	True positive	True negative	False Positive	False Negative	Acc (%)
0.2	167	813	5	4	99.09
0.25	167	813	5	4	99.09
0.3	167	812	6	4	98.99
0.35	165	810	8	6	98.58
0.4	163	808	10	8	98.18
0.45	163	808	10	8	98.18
0.5	161	806	12	10	97.78
0.55	159	802	16	12	97.17

Table 4.3: Event prediction accuracy of the Henon map (no noise) without a GMM component with respect to different σ values.

σ	True positive	True negative	False Positive	False Negative	Acc (%)
0.2	147	786	29	27	94.34
0.25	146	787	28	28	94.34
0.3	144	784	31	30	93.83
0.35	140	784	31	34	93.43
0.4	139	783	32	35	93.23
0.45	135	785	30	39	93.02
0.5	136	783	32	38	92.92
0.55	132	781	34	42	92.32

Table 4.4: Event accuracy of the Henon map (10% noise) without a GMM component with respect to different σ values.

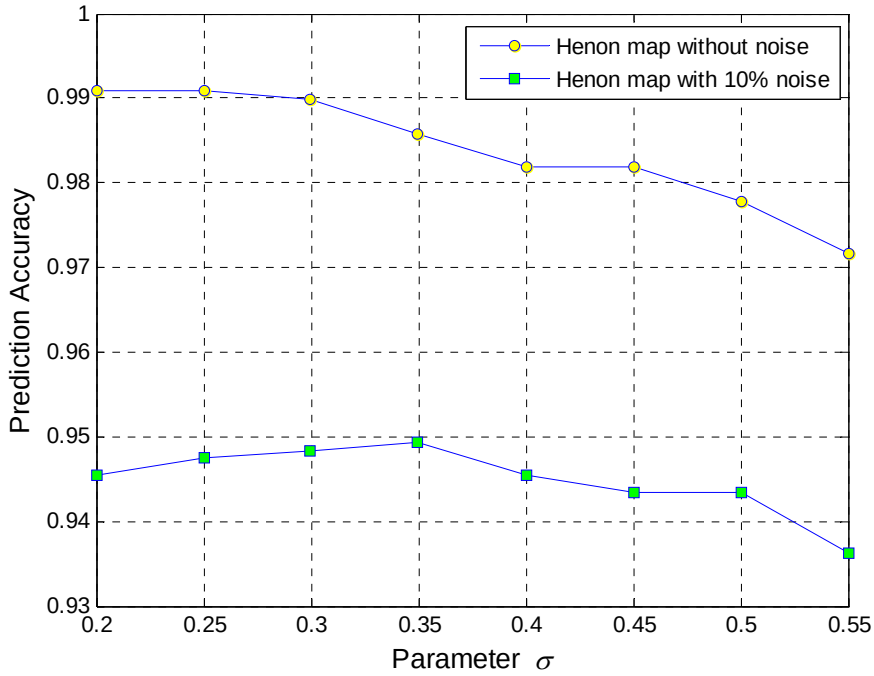

Figure 4.11: Event prediction accuracy of events in a Henon map with different σ values.

The MRPS method achieved a high accuracy (99.1%) for the no noise case and 94.94% for the 10% noise case. This indicates a good performance of this new algorithm in predicting events in a complex chaotic series that is representative of a range of multivariate nonlinear system identification problems. Comparing with the univariate results in Chapter 3, the accuracy is also improved with added information in the y component of the Henon map. In Chapter 3, by applying univariate RPS approach to the no additive noise Henon map, the highest prediction accuracy and true positive rates are 96.58% and 81.87%, respectively, whereas for 10% additive noise Henon map, the highest prediction accuracy and true positive rates are 94.44% and 78.16%, respectively. This means that including multivariate data sequence information impacts on the overall event prediction performance.

Example (b): The second example is the Rossler map as illustrated in Fig. 4.12. The Rossler map is defined by:

$$\begin{cases} \dfrac{dx}{dt} = -y - z \\ \dfrac{dy}{dt} = x + ay \\ \dfrac{dz}{dt} = z(x - c) + b. \end{cases} \tag{4.26}$$

For example, we use $a = 0.3, b = 0.5,$ and $c = 5$.

For the Rossler map, the z component in the system state variables is chosen as the target series. In this simulation experiment, the goal is to predict when in the next time step the z time series value exceeds 10. As an illustrative example, we choose the x and z component to predict the events in z time series. The event characterization function is defined as:

$$g(\mathbf{x}_t) = \begin{cases} +1 & z_{t+1} > 10 \\ -1 & z_{t+1} <= 10. \end{cases} \tag{4.27}$$

Similarly, we apply the minimum mutual information method to the Rossler map and apply the false nearest-neighbor method to estimate the embedding dimension of each sequence. Fig. 4.13 displays the value of the mutual information between delayed x values under different values of time delay τ. The mutual information decreases consistently as the time delay increases from 1 to 7 and increases after reaching a local minima at $\tau = 7$. This indicates that the correlation between delayed embeddings is smallest at a time delay of 7.

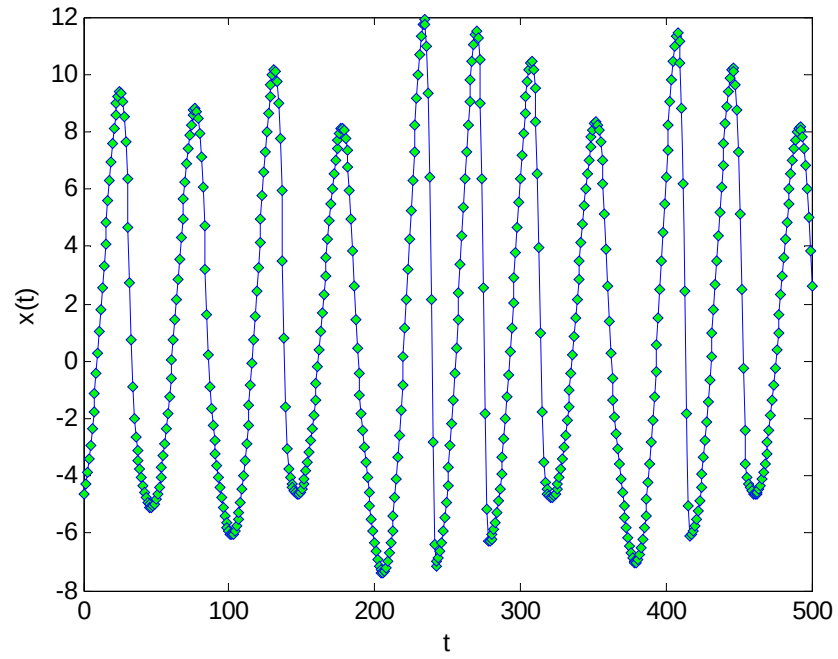

Figure 4.12: The $x(t)$ component of a Rossler map.

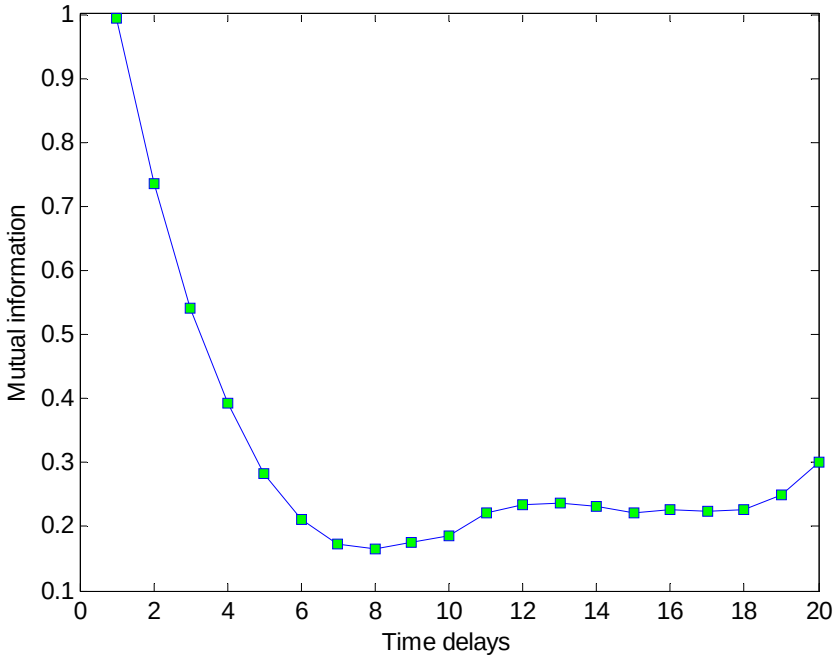

Figure 4.13: Mutual information of the $x(t)$ component of a Rossler map with time delays.

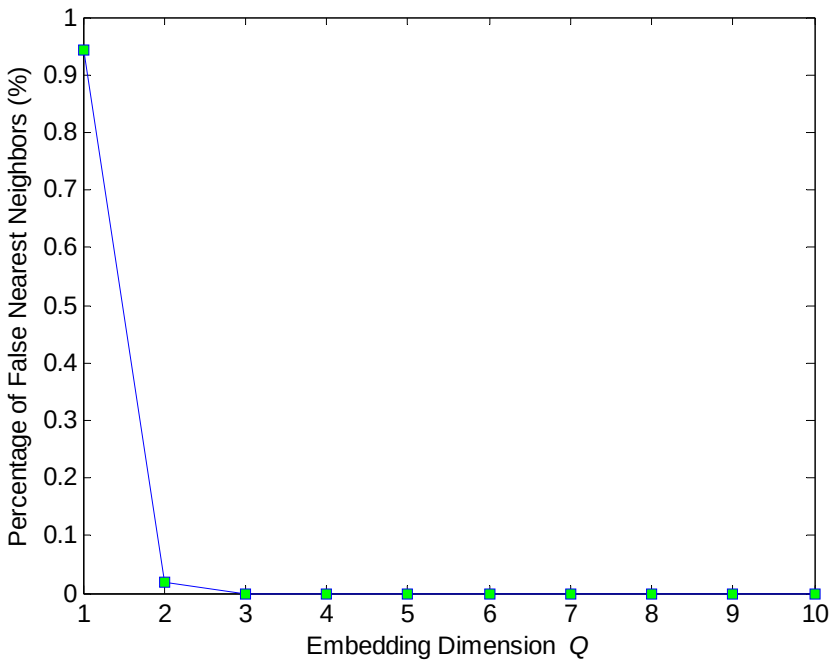

Figure 4.14: False nearest neighbors of the $x(t)$ component of the Rossler map with different embedding dimensions.

In Fig. 4.14, the percentage of false nearest neighbors decreases from 10 as the embedding dimension increases from 1 to 3. This indicates that an embedding dimension Q of 3 is a sufficient choice to the embed time series in the x dimension.

In this multivariate case experiment, we included 3000 data points with both x and z components of a Rossler map. The first 2000 data are used as a training set, and the remaining 1000 data are used as a testing set for validation. At the training step, multiple scenarios are tested with different values of σ. Table 4.5 presents the prediction accuracy results with respect to σ for the Rossler map.

σ	True positive	True negative	False Positive	False Negative	Acc (%)
0.2	96	399	1	0	99.80
0.25	96	399	1	0	99.80
0.3	96	399	1	0	99.80
0.35	96	399	1	0	99.80
0.4	96	399	1	0	99.80
0.45	96	400	0	0	100.00
0.5	96	400	0	0	100.00

Table 4.5: The event prediction accuracy of a Rossler map with different σ values.

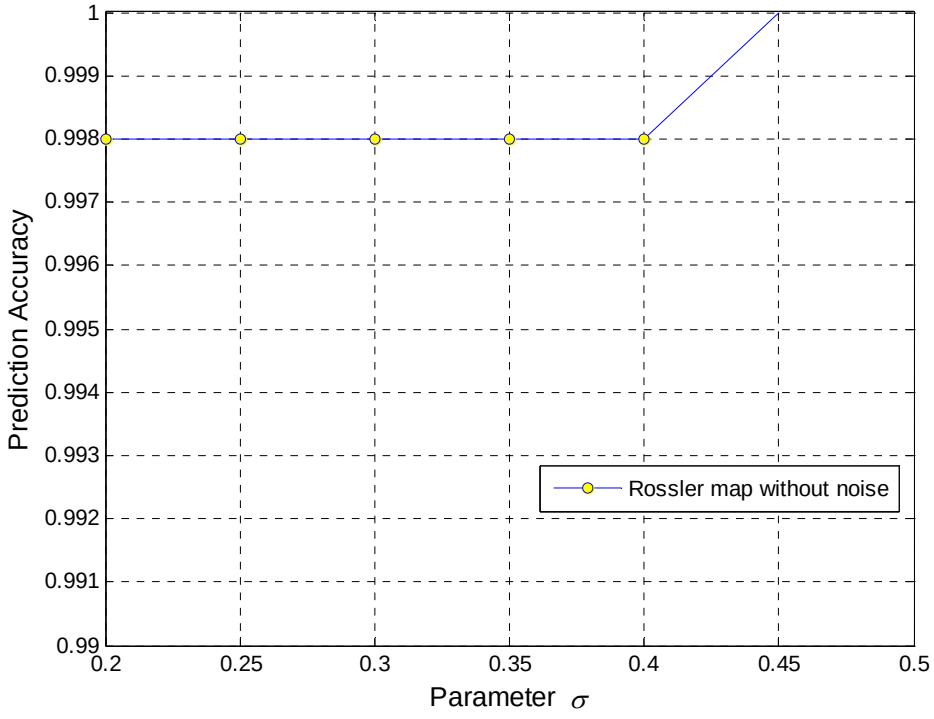

Figure 4.15: Event prediction accuracy of events in a Rossler map with different σ values.

The MRPS method achieved a high accuracy (99.8%) for Rossler map. The results presented in Table 4.5 and Fig. 4.15 demonstrate a good performance of MRPS. Comparing with univariate case in Chapter 3, both methods achieved 100% accuracy for Rossler map.

Now that we have illustrated how the MRPS method can be applied to two basic chaotic time series, in the following two examples (c) and (d) we compare the MRPS method in two more complex datasets, Lorenz map [3], and Sludge Volume Index (SVI) [10].

Example (c): The third example is the Lorenz map as illustrated in Fig. 3.17. The Lorenz map is defined by:

$$\begin{cases} \dfrac{dx}{dt} = \gamma(y - x) \\ \dfrac{dy}{dt} = x(\rho - z) - y \\ \dfrac{dz}{dt} = xy - \beta z. \end{cases} \tag{4.28}$$

In the simulation, the Lorenz time series is generated by setting the initial values, $x_0 = 0$, $y_0 = -0.01$, and $z_0 = 0.01$, and parameters, $\gamma = 9$, $\rho = 25$, and $\beta = 3.3$. For the Lorenz map, the overall strategy was to detect temporal patterns in the multivariate sequence $\mathbf{x}_t = (x_t, y_t)$ to predict the events in the x_t sequence. Multivariate sequences $\mathbf{x}_t = (x_t, y_t)$ are embedded into the MRPS, and the optimization method applied to classify the patterns. In this simulation experiment, the goal is to predict that in the next time step the x time series value exceeds 11. The event characterization function is defined as

$$g(\mathbf{x}_t) = \begin{cases} +1 & \text{if } x_{t+1} > 11.0 \\ -1 & \text{if } x_{t+1} <= 11.0. \end{cases} \tag{4.29}$$

Similar to examples (a) and (b), the time delay was estimated as $\tau = 6$, and the embedding dimension $Q = 3$ for both x and y dimensions.

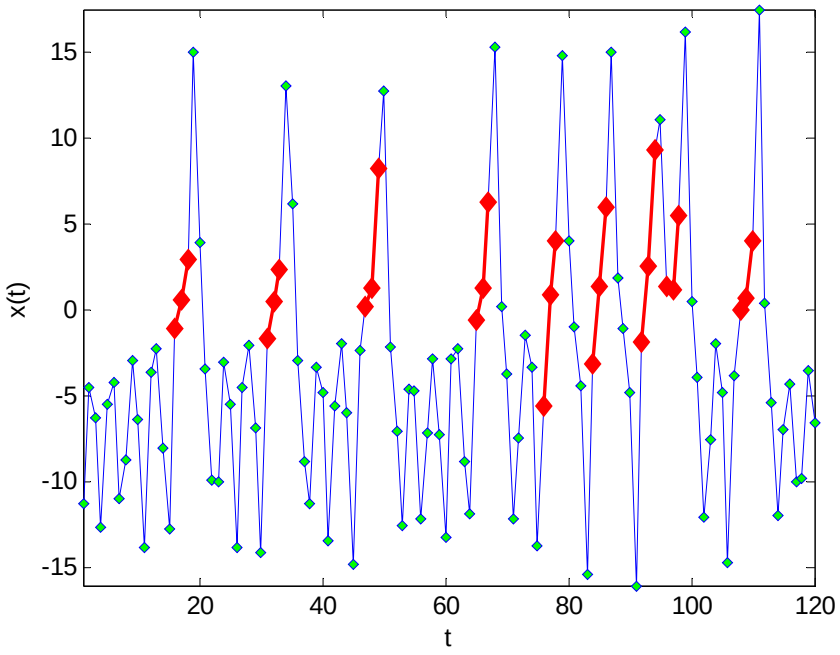

Figure 4.16: Time series and temporal patterns of a Lorenz map.

The resulting temporal patterns associated with a defined event function are plotted in Fig.4.16. It can be observed that the temporal patterns are different in structure and initial starting point compared with similar patterns found in [3]. This result indicates that our new approach is not only capable of identifying similar patterns but also of identifying patterns rather different in structure and initial starting values.

This is because the new approach is not constrained by searching individual clusters, but rather finds a decision boundary that includes all the predictive patterns that

are significantly correlated with events. To evaluate the performance of the new MRPS approach, we compared the results of our MRPS method with the TSDM method proposed by Povinelli and Feng [2]. Also included in the comparison was the Artificial Neural Network (ANN) with three layers with 6 neurons in the input layer and 12 neurons in the hidden layer. In the input and hidden layers, a sigmoid activation function was used. A threshold function was applied to convert the output to a binary decision output. Tables 4.6 and 4.7 present the comparative results of the MRPS method and previous methods.

	Predicted as events	Predicted as nonevents
Actual events	TP = 55	FN = 6
Actual nonevents	FP = 5	TN = 435

Table 4.6: Test results of the Lorenz map.

	Training Set		Test Set	
Method	True Positive Rate (%)	Accuracy (%)	True Positive Rate (%)	Accuracy (%)
MRPS	94.72	98.36	90.17	97.21
TSDM	72.45	96.32	65.65	93.52
ANN	87.00	94.35	78.30	92.59

Table 4.7: Comparison of the prediction performance

The accuracy measure and true positive rate measure are defined as True Positive rate = TP/(TP + FN) and Accuracy = (TP + TN)/(TP + FN + TN + FP), where TP, TN, FP, and FN represent the number of true positive, true negative, false positive, and false negative predictions, respectively. The results demonstrate that the MRPS method has a better and more consistent performance over the other two methods in both training and testing, whereas the other two methods had lower rates of accuracy. Another observation is that our new MRPS framework outperforms the other two methods by a large margin in predicting the events correctly, as shown by the significantly higher TP rate in the testing phase. Since new MRPS framework is based on modeling both the temporal dynamics and time-independent discriminative information in multivariate data sequences, from the experimental results we conclude that the MRPS method is superior when the underlying data system is complex and includes heterogeneous patterns.

σ	β_{max}	Center $\phi(\mathbf{x})$ associated with β_{max}	α_1	α_0	$L(\sigma, \beta)$
0.15	4.69	(0.80 0.58 4.74)	6.12	4.16	468.99
0.20	4.31	(0.80 0.58 4.74)	6.93	3.71	456.36
0.25	3.97	(0.80 0.58 4.74)	7.58	3.32	454.00
0.30	3.62	(0.80 0.58 4.74)	8.06	2.90	465.12
0.35	3.15	(0.80 0.58 4.74)	8.27	2.29	465.12
0.40	2.63	(0.80 0.58 4.74)	8.12	1.59	462.77
0.45	2.11	(0.80 0.58 4.74)	7.79	0.84	467.47
0.50	1.70	(0.80 0.58 4.74)	7.48	0.20	469.38
0.55	1.55	(-0.21 -0.19 -1.17)	7.21	-0.17	475.99

Table 4.8: Objective loss function values for Lorenz dataset with different σ values.

β	Center $\phi(\mathbf{x})$	$\varphi(\mathbf{x})$
3.97	(0.80 0.58 4.74)	0.24
1.36	(0.38 0.29 0.59)	0.15
1.29	(–0.62 –0.06 –1.13)	–0.19
1.12	(–0.71 -0.02 –1.07)	–0.39
1.12	(0.06 0.22 0.27)	0.24
1.07	(1.35 0.17 –0.82)	0.05
1.07	(–0.71 –0.01 –1.06)	–0.15
1.06	(1.06 0.31 –1.33)	–0.19

Table 4.9: Phase space vector with large weights for Lorenz dataset with different β values.

It can also be concluded that the performance of RPS-based approaches is superior to the neural network method in identifying temporal patterns predictive of events in general. This is because the RPS-based methods are capable of modeling the temporal dynamic structures that neural network methods usually cannot capture fully.

In our approach, the parameter σ in the classifier has to be predetermined. To see the effect of choosing the correct value of σ, we performed experiments by varying the value of σ and reporting the value of the objective loss function in Table 4.8. We can see that the value of objective function decreases and then increases as σ increases from 0.15 to 0.55. The objective loss function has the smallest value when σ = 0.25. This result indicates that a proper choice of σ gives a better performance. Table 4.9 presents the estimated results of the eight largest weight coefficients β with their associated phase space center $\phi(\mathbf{x})$ and ratio $\varphi(\mathbf{x})$. The center $\phi(\mathbf{x})$ in the phase space associated with the largest weight $\beta_{\max}$ remains the same with regards to the varying of parameter σ.

This indicates that our algorithm is robust in identifying the multivariate temporal patterns in the transformed phase space.

Example (d): The fourth example is the SVI series, and the MRPS approach is applied to tackle the sludge-bulking problem as discussed in Section 3.6. There are many potential causal factors to the sludge bulking, and one causal factor is the level of Dissolved Oxygen (DO). In this experiment, among 25 potential variables, we consider two data sequences: the SVI and the DO indices as the probable factors related to the sludge-bulking problem. With data from 2003 to 2008 provided by a Chicago water treatment company, the first three years data are used as a training data set, and the remaining three years data are used as a testing set for validation. We denote the two-dimensional sequence, consisting of SVI and DO, as $\mathbf{x}_t = (S_t, D_t)$. The events are defined as

$$g(\mathbf{x}_t) = \begin{cases} +1 & \max\{S_{t+1}, S_{t+2}, S_{t+3}\} \geq 150.0 \\ -1 & \max\{S_{t+1}, S_{t+2}, S_{t+3}\} < 150.0 \end{cases}. \tag{4.29}$$

The time delay was estimated as $\tau = 5$, and the embedding dimension as $Q = 4$. In Fig. 4.17, the two-dimensional patterns are plotted, with each point in the figure representing a two-dimensional data $\mathbf{x}_t = (S_t, D_t)$. Fig. 4.17 shows a decision boundary separating the event-related patterns and normal points based on the estimation of the GMM. Instead of using a hard decision boundary, our framework includes a log-odds ratio estimated from the GMM to exploit the time-independent discriminative structure hidden in the multivariate data sequences.

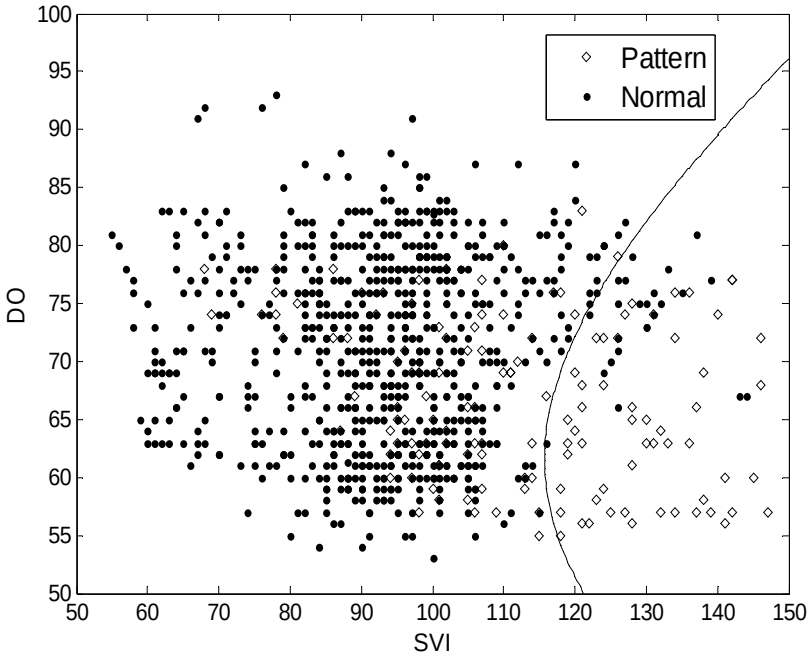

Figure 4.17: Two-dimensional temporal patterns of the SVI index.

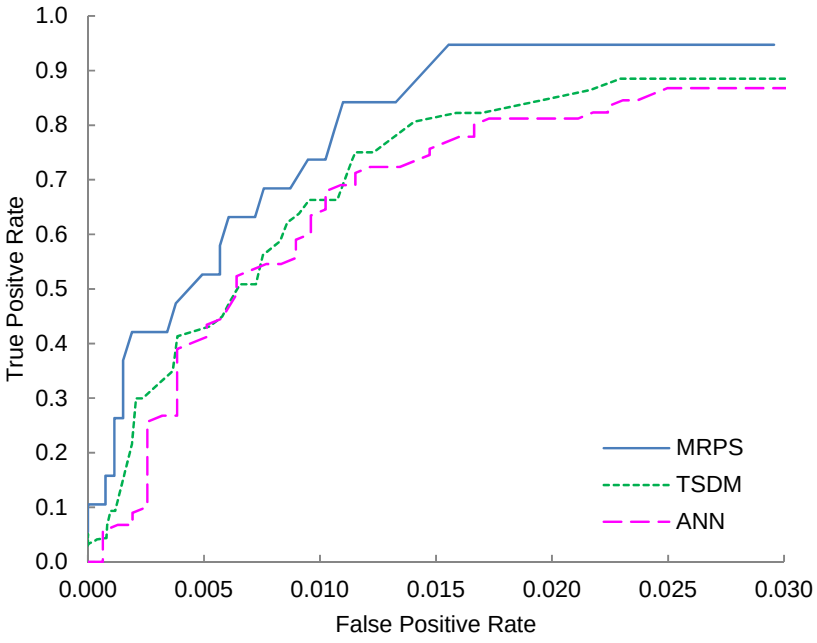

Figure 4.18: The Receiver Operating Characteristic (ROC) performance analysis.

We plot the Receiver Operating Characteristic (ROC) curves of the MRPS together with the TSDM and ANN in Fig. 4.19. The areas under the ROC curves are 0.968, 0.935, and 0.925 for MRPS, TSDM, and ANN, respectively. These ROC curves show TP rates with respect to the FP rates in the SVI series test dataset. Since the event/samples ratio is relatively small, the ROC curves are displayed in a region with a low FP ratio. Fig. 4.19 shows that the performance in the ROC curves of the TSDM and ANN methods are below that of the MRPS in general. The TSDM method performs fairly close to the ANN method for lower FP rates and slightly better for higher FP rates. Compared with the ANN ROC curve, the relative FP rate reduction for the MRPS approach is about 40%, and the increase of TP rates is about 15%.

Table 4.10 presents the confusion matrix results of MRPS method. Similar to example (c), we evaluated the performance of the new MRPS method by comparing it with the TSDM method and the neural network results, as illustrated in Table 4.11. The results show that the MRPS method outperforms the other two methods in both accuracy and rates of TPs. The MRPS method generates a higher TP rate by a large margin in both training and testing, whereas the other two methods had lower rates.

	Predicted as events	Predicted as nonevents
Actual events	TP=29	FN=5
Actual nonevents	FP =6	TN=1605

Table 4.10: The confusion matrix of the SVI dataset.

Method	TP Rate (%)	TN Rate (%)	Accuracy (%)
MRPS	83.86	99.63	99.09
TSDM	65.53	98.85	97.57
ANN	61.37	98.45	97.26

Table 4.11: A comparison of the testing set results of the SVI dataset.

σ	β_{max}	Center $\phi(\mathbf{x})$ associated with β_{max}	α_1	α_0	$L(\sigma, \beta)$
0.01	7.60	(–2.17 –0.27 0.14 –0.44 0.24)	1.74	–2.16	23.18
0.04	7.59	(–2.17 –0.27 0.14 –0.44 0.24)	1.74	–2.14	23.03
0.07	7.48	(–2.17 –0.27 0.14 –0.44 0.24)	1.71	–2.10	22.01
0.1	7.10	(–2.17 –0.27 0.14 –0.44 0.24)	1.64	–1.86	21.39
0.13	6.70	(–2.17 -0.27 0.14 –0.44 0.24)	1.60	–1.56	22.28
0.16	6.36	(–2.17 –0.27 0.14 –0.44 0.24)	1.56	–1.27	25.40
0.19	6.02	(–2.17 –0.27 0.14 –0.44 0.24)	1.52	–0.99	30.52
0.22	5.71	(–2.17 –0.27 0.14 –0.44 0.24)	1.49	–0.73	36.84
0.25	5.45	(–2.17 –0.27 0.14 –0.44 0.24)	1.45	–0.52	43.56
0.28	5.24	(–2.17 –0.27 0.14 –0.44 0.24)	1.42	–0.37	50.02

Table 4.12: Objective loss function values for SVI dataset with different σ values.

β	Center $\phi(\mathbf{x})$ associated with β	$\varphi(\mathbf{x})$ associated with β
7.48	(–2.17 –0.27 0.14 –0.44 0.24)	0.24
6.68	(–0.95 –0.14 0.27 –0.34 0.15)	0.15
5.76	(–0.55 0.17 0.29 –0.68 –0.19)	–0.19
4.83	(0.17 0.34 0.39 0.00 –0.39)	–0.39
4.68	(0.35 0.32 0.34 –0.34 0.24)	0.24
4.19	(0.11 0.18 –0.08 –0.68 0.00)	0.00
4.12	(0.70 0.25 0.35 –0.15 0.05)	0.05
3.93	(0.37 0.04 0.18 0.73 –0.15)	–0.15
3.89	(0.72 0.27 0.27 –0.10 0.00)	0.00
3.23	(–0.43 0.11 0.13 –0.78 –0.19)	–0.19

Table 4.13: Phase space vector with large weights for SVI dataset with different β values.

To see the effect of choosing the correct value of σ, in Table 4.12, we show the value of the objective loss function with respect to different values of σ. We can see that the value of the objective function decreases and then increases as σ increases from 0.01 to 0.28. The objective loss function has the smallest value when $\sigma = 0.1$; therefore, a proper choice of σ gives a better performance. To see the robustness of MRPS method, it can be observed from Table 4.12 that the center $\phi(\mathbf{x})$ in the phase space associated with the largest weight $\beta_{\max}$ remains the same with varying values of parameter σ.

Table 4.13 shows the results of the ten largest weight coefficients β and their associated phase space center $\phi(\mathbf{x})$ and ratio $\varphi(\mathbf{x})$. We observe that while some patterns have large phase space weights, indicating high importance as temporal patterns, they do not necessarily have large weights in the Bayesian log-odds ratio. In other words, applying a discriminative method alone would not identify these temporal patterns. This demonstrates that by modeling temporal dynamics through RPS, the MRPS approach can

provide additional information that a discriminative method does not capture.

In this chapter, the MRPS method is presented for identification of predictive temporal patterns in a multivariate dynamic data system. The new MRPS method extends the original univariate reconstructed phase space framework, which is based on a fuzzy unsupervised clustering method, by incorporating a new mechanism of data categorization based on the definition of events. The new method uses an exponential loss objective function to optimize a classifier which consists of a radial basis kernel function and a log-odds ratio component. Experimental results demonstrated the effectiveness of this new approach. The results in this chapter have been published in [89].

CHAPTER 5 EQUIVALENCE ANALYSIS OF PHASE SPACE AND THE ASSOCIATED PARAMETRIC SPACE

Although pattern detection in a traditional RPS through optimization has been a proven effective approach, theoretical validation or the exploration of the underlying dynamics in terms of traditional time series analysis methods, such as parametric AR models, is still an open issue. To explore such a relationship, we introduce a parametric space constructed using adaptively estimated AR type parameters.

There are several possible approaches to adaptive parameter estimation, such as least squares (LS) [4], Kalman filters [30], and Recursive Least Squares (RLS) methods [30]. Since the underlying system is unknown, a Kalman filter approach is not applicable in the absence of information regarding the transition matrix and the variances of noises. Moreover, in most cases, the length of the temporal patterns is typically short; as a result, the number of data points is not large enough to make a sufficiently reliable or accurate estimation using a simple sliding window LS method. In contrast, the RLS method does not require detailed structural information of the system and can solve the estimation problem adaptively by keeping historical estimates.

5.1 Equivalence Analysis

To solve a pattern detection problem, the RPS approach assumes that temporal patterns, with dimension Q and time delay τ, in the form of $\mathbf{x}_t = (x_t, x_{t-\tau}, ..., x_{t-(Q-1)\tau})$ are statistically predictive of the magnitude of a *k*-step future value x_{t+k} in terms of an event function, $g(x_{t+k})$, typically against a certain threshold. In other words, such temporal patterns represent hidden underlying dynamics that at least are correlated, if not

causal, to the future values in the data sequence. In other words, there is a linear or nonlinear relationship between $\mathbf{x}_t = (x_t, x_{t-\tau}, ..., x_{t-(Q-1)\tau})$ and x_{t+k}.

Let us consider a data sequence, x_t, $t = 1, ..., N$, with embedding dimension Q and time delay τ, and assume every *k*-step ahead future value x_{t+k} can be represented as a linear, or linearized for a nonlinear system, summation of temporal pattern vectors, $\mathbf{x}_t = (x_t, x_{t-\tau}, ..., x_{t-(Q-1)\tau})$ by

$$x_{t+k} = \beta_{1,t} x_t + \beta_{2,t} x_{t-\tau} ... + \beta_{Q,t} x_{t-(Q-1)\tau} + \beta_{0,t} + \varepsilon_t, \tag{5.1}$$

where $\boldsymbol{\beta}_t = (\beta_{1,t}, \beta_{2,t}, ..., \beta_{Q,t})$, $t = 1, ..., N$, are the linear coefficients. These linear coefficients are assumed stochastic, as they are representative of constantly changing underlying system states over time, but when a system is in a certain state, the change is assumed to be sufficiently small. Without loss of generosity, it is assumed that k =1 and that the system occupies a certain state from time $t - 2Q\tau$ to t. Combining Q equations, and each is τ step delayed, we have

$$\begin{bmatrix} x_{t+k} \\ x_{t+k-\tau} \\ \vdots \\ x_{t+k-(Q-1)\tau} \end{bmatrix} = \begin{bmatrix} x_t & x_{t-\tau} & \cdots & x_{t-(Q-1)\tau} & 1 \\ x_{t-\tau} & x_{t-2\tau} & \cdots & x_{t-(Q-1)\tau} & 1 \\ \vdots & \vdots & \ddots & \vdots & 1 \\ x_{t-(Q-1)\tau} & x_{t-Q\tau} & \cdots & x_{t-2Q\tau} & 1 \end{bmatrix} \begin{bmatrix} \beta_{1,t} \\ \beta_{2,t} \\ \vdots \\ \beta_{Q,t} \\ \beta_{0,t} \end{bmatrix} + \varepsilon_t \ . \tag{5.2}$$

Eqn (5.2) can then be rewritten in a vector form,

$$\mathbf{x}_{t+k} = \begin{bmatrix} \mathbf{x}_t & \mathbf{x}_{t-\tau} & \cdots & \mathbf{x}_{t-(Q-1)\tau} & 1 \end{bmatrix} \boldsymbol{\beta}_t + \varepsilon_t \ , \tag{5.3}$$

where $\boldsymbol{\beta}_t = \begin{bmatrix} \beta_{1,t} & \beta_{2,t} & \cdots & \beta_{Q,t} & \beta_{0,t} \end{bmatrix}^T$, and $\mathbf{x}_t = \begin{bmatrix} x_t & x_{t-\tau} & \cdots & x_{t-(Q-1)\tau} \end{bmatrix}$.

Assuming the model in Eqn. (5.3), and taking the expected value of both sides, we have

$$\boldsymbol{\mu} = \overline{\beta}_{1,t}\boldsymbol{\mu} + \overline{\beta}_{2,t}\boldsymbol{\mu} + ... + \overline{\beta}_{Q,t}\boldsymbol{\mu} + \overline{\boldsymbol{\beta}}_{0,t}\,, \tag{5.4}$$

where $\boldsymbol{\mu} = E\{\mathbf{x}_t\}$. Rewriting both sides, we get

$$\boldsymbol{\mu} = \frac{\overline{\boldsymbol{\beta}}_{0,t}}{(1 - \overline{\beta}_{1,t} - \overline{\beta}_{2,t} - ... - \overline{\beta}_{Q,t})}\ . \tag{5.5}$$

Thus, the mean or center of the RPS embedding cluster can be calculated using Eqn. (5.5). This relationship in Eqn. (5.5) shows a one-to-one and onto mapping $\overline{\boldsymbol{\beta}}_t \rightarrow \boldsymbol{\mu}$ from the mean of a parameter space cluster to the mean of a RPS embedding cluster. Therefore, the new parameter space preserves the important first-order statistic, that is, the mean of the original RPS cluster.

Given the results in Eqn. (5.5), we can rewrite Eqn. (5.4) as

$$\mathbf{x}_t - \boldsymbol{\mu} = \beta_{1,t}(\mathbf{x}_{t-\tau} - \boldsymbol{\mu}) + \beta_{2,t}(\mathbf{x}_{t-2\tau} - \boldsymbol{\mu}) + ... + \beta_{Q,t}(\mathbf{x}_{t-Q\tau} - \boldsymbol{\mu}) + \boldsymbol{\nu}_t\ . \tag{5.6}$$

By multiplying $\mathbf{x}_{t-k} - \boldsymbol{\mu}$ and taking expectations of both sides, we have

$$\begin{aligned} E\{(\mathbf{x}_{t-k} - \boldsymbol{\mu})(\mathbf{x}_t - \boldsymbol{\mu})\} = {} & \beta_{1,t}E\{(\mathbf{x}_{t-k} - \boldsymbol{\mu})(\mathbf{x}_{t-\tau} - \boldsymbol{\mu})\} + ... \\ & + \beta_{Q,t}E\{(\mathbf{x}_{t-k} - \boldsymbol{\mu})(\mathbf{x}_{t-Q\tau} - \boldsymbol{\mu})\} + E\{(\mathbf{x}_{t-k} - \boldsymbol{\mu})\boldsymbol{\nu}_t\}. \end{aligned}$$

Denoting $E\{(\mathbf{x}_t - \boldsymbol{\mu})(\mathbf{x}_{t-k} - \boldsymbol{\mu})\} = \boldsymbol{\gamma}_k$, we obtain

$$\boldsymbol{\gamma}_k = \overline{\beta}_{1,t}\boldsymbol{\gamma}_{k-\tau} + \overline{\beta}_{2,t}\boldsymbol{\gamma}_{k-2\tau} + ... + \overline{\beta}_{Q,t}\boldsymbol{\gamma}_{k-Q\tau}\ . \tag{5.7}$$

Since the second-order moment of the sequence can be obtained from Eqn. (5.7), this shows a direct mapping $\overline{\boldsymbol{\beta}}_t \rightarrow \boldsymbol{\gamma}_k$ between the center of parameter space cluster to the covariance of corresponding RPS cluster. Hence, the new parameter space can also preserve the important second-order statistics of the original RPS.

5.2 Simulation Example

The following experimental example illustrates how the mean value of traditional phase space can be estimated using the mean value of parametric space. Consider the following third-order autoregressive series:

$$x_t = 0.2x_{t-1} + 0.5x_{t-3} + 0.2 + \varepsilon_{t+1}. \tag{5.8}$$

For this system, from Eqn. (5.8), we can see two lagging components x_{t-1} and x_{t-3} with a two-step delay between them. Thus, we choose dimension $Q = 2$ and time delay $\tau = 2$ to embed this series.

Fig. 5.1 illustrates the parametric space composed of estimated coefficients associated with Eqn. (5.8). The mean in each dimension is estimated as $\overline{\beta}_1 = 0.1426$, $\overline{\beta}_2 = 0.3537$, and $\overline{\beta}_0 = 0.3615$. By the relationship in Eqn. (5.5), the center in the original phase space is estimated as $E[x_t] = \hat{\mu}_x = \overline{\beta}_0 / (1 - \overline{\beta}_1 - \overline{\beta}_2) = 0.7177$.

Fig. 5.2 displays the traditional phase space embedding in a two-dimensional space. The means of the embeddings in both dimensions are $\overline{x}_t = 0.7227$ and $\overline{x}_{t+\tau} = 0.7254$. The results suggest the experimental results in both dimensions match closely with the theoretical estimates.

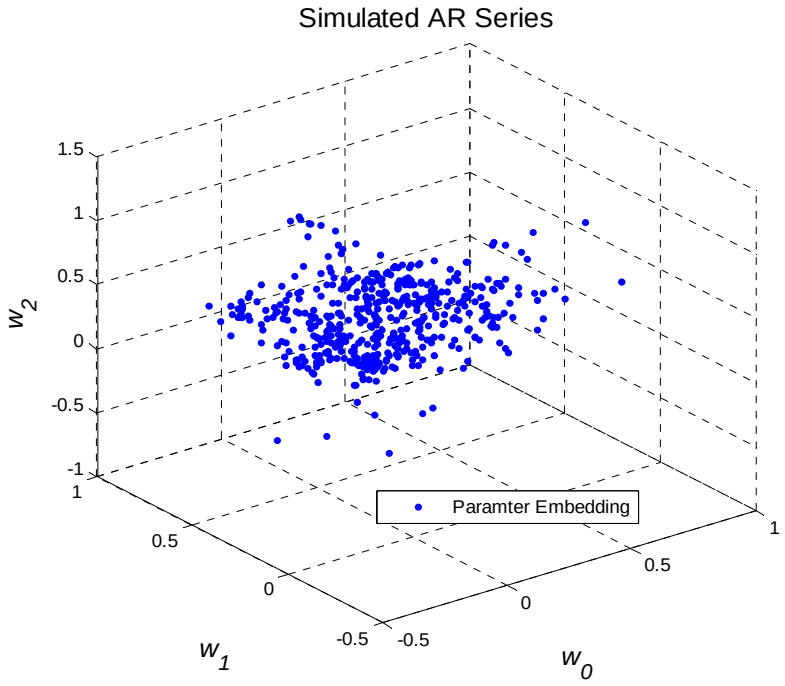

Figure 5.1: Parametric space of a third-order autoregressive series.

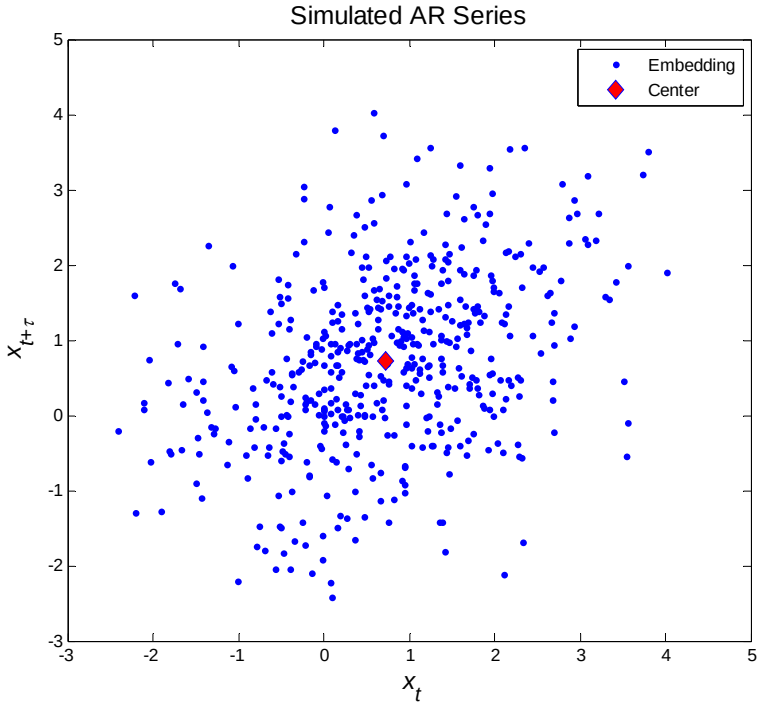

Figure 5.2: RPS embedding of a third-order autoregressive series.

The simulation results in Fig. 5.2 show that the center of the parametric space has a sufficiently close mapping relationship with the center of the RPS. Thus, the relationship defined in Eqns. (5.5) and (5.7) establishes time-domain equivalence mapping between the RPS and its associated parametric space. The formulation in Eqn. (5.4) shares a similar form as a vector autoregressive model, which partly explains why the phase space approach has been successful in identifying temporal patterns.

CHAPTER 6 CONCLUSIONS AND FUTURE WORK

In this dissertation, several novel methods were presented for identifying temporal patterns predictive of events in a dynamic data system. These new methods are original contributions to the field of nonlinear time series analysis and machine learning. The key components in these methods include the categorization of event functions, a phase space transformation, supervised classification of temporal patterns, kernel radial basis functions in phase space, as well as classifier design through optimization of an exponential loss function.

In Chapter 3, we developed a new GMM-SVM method addressing event prediction and pattern detection problem in univariate data systems. This new method uses both generative and discriminative models to provide a two-stage pattern classification for event prediction in a nonlinear system. We demonstrated that the new method has improved performance compared with conventional methods.

In Chapter 4, we introduced a new multivariate reconstructed phase space (MRPS) method. The new MRPS method extends the original univariate reconstructed phase space framework, which is based on a fuzzy unsupervised clustering method, by incorporating a new mechanism of data segmentation based on the categorical definition of event functions. In addition to modeling temporal dynamics in a multivariate phase space, a Bayesian approach is applied to model the first-order Markov behavior in the multi-dimensional data sequences. The method uses an exponential loss objective function to optimize a hybrid classifier which consists of a radial basis kernel function and a log-odds ratio component. Compared with a univariate modeling approach,

modeling the system dynamics using multivariate data sequences provides more insights and a better understanding of the overall system. This dissertation has demonstrated the effectiveness of the MRPS method by applying it to several experiments and showed significant improvements in event prediction and predictive pattern identification compared with baseline methods.

In Chapter 5, an equivalence analysis of phase space and the associated parametric space was presented. An equivalence mapping was established between the time-domain RPS and its associated parametric space. It was shown that the new parametric space can preserve the first-order and the second-order statistics of the original RPS.

Although this dissertation has demonstrated the effectiveness of several novel methods, further work still can be done to enhance the performance of prediction and pattern detection. Future work will include:

- The MRPS method can be extended to be applicable in a wider range of multivariate event prediction problems. Although a thresholding event function was used, a relaxation of the assumption of thresholding event function can be done by applying more complex event functions to improve the applicability of RPS-based methods in other multivariate applications.
- Although we have employed a Gaussian Mixture Model as the generative model component in the methods, in further study, the Gaussian distribution assumption can be extended to other distributions.

- Since the parametric phase space preserves the same first and second order statistics of the time domain RPS, in future work, parametric pattern detection through adaptive parameter estimation and prediction can be investigated.
- Since computational complexity of the optimization in MRPS method increases as the size of datasets increases, alternative optimization methods can be applied to ensure a fast convergence and a better computational performance.

BIBLIOGRAPHY

[1] H. Kantz and T. Schreiber, Nonlinear *Time Series Analysis*, Cambridge University Press, Cambridge, England, 1997.

[2] R.J. Povinelli and X. Feng, "A new temporal pattern identification method for characterization and prediction of complex time series events," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 2, pp. 339–352, 2003.

[3] X. Feng and H. Huang, "A fuzzy-set-based reconstruction phase space method for identification of temporal patterns in complex time series," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 5, pp. 601–613, 2005.

[4] C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2007.

[5] C.-H. Lee, A. Liu, W.-S. Chen, "Pattern discovery of fuzzy time series for financial prediction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 5, 2006.

[6] L. Pritchett, "Understanding patterns of economic growth: Searching for hills among plateaus, mountains, and plains," *World Bank Economic Review*, vol. 14, issue 2, 2000.

[7] J. V. Hansen and R.D. Nelson, "Neural networks and traditional time series methods: A synergistic combination in state economic forecasts," *IEEE Transactions on Neural Networks*, vol. 8, no. 4, July 1997.

[8] R. J. Povinelli, M. T. Johnson, A. C. Lindgren, and J. Ye, "Time series classification using Gaussian mixture models of reconstructed phase spaces", *IEEE Transactions on Knowledge and Data Engineering,* vol. 16, pp. 779–783, 2004.

[9] K. Sternickel, "Automatic pattern recognition in ECG time series," *Computer Methods and Programs in Biomedicine*, vol. 68, issue 2, pp. 109–115, 2002.

[10] A.G. Capodaglio, H. Jones, V. Novotny, X. Feng, "Sludge bulking analysis and forecasting: Application of system identification and artificial neural computing technologies", *Water Research*, vol. 25(10), pp. 1217-1224, 1991.

[11] C. Faloutsos, M. Ranganathan and Y. Manolopoulos, "Fast subsequence matching in time-series databases," *Proceedings ACM SIGMOD Int'l Conf. Management of Data*, pp. 419–429, 1994.

[12] K. Chan, A. Fu, "Efficient time series matching by wavelets", *Proceedings IEEE Int'l Conf. Data Eng.*, pp. 126–133, 1999.

[13] D.B. Percival and A.T. Walden, *Wavelet Methods for Time Series Analysis*, Cambridge University Press, Cambridge, England, 2000.

[14] E. Keogh and P. Smyth, “A Probabilistic Approach to Fast Pattern Matching in Time Series Databases,” *Proceedings Int’l. Conf. Knowledge Discovery and Data Mining*, 1997.

[15] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed., Prentice Hall, Upper Saddle River, 1999.

[16] A. Nanopoulos, R. Alcock, and Y. Manolopoulos, “Feature-based classification of time-series data,” *International Journal of Computer Research*, pp. 49-61, 2001.

[17] J.J. Rodriguez, and C.J. Alonso, "Interval and dynamic time warping-based decision trees," *Proceedings of ACM symposium on Applied Computing*, pp. 548-552, 2004.

[18] A. Suarez, J.F. Lutsko, “Globally optimal fuzzy decision trees for classification and regression,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21(12), pp. 1297–1311, 1999.

[19] S. Theodoridis, K. Koutroumbas, *Pattern Recognition*, Academic Press, Waltham, 2008.

[20] F. Takens, “Detecting strange attractors in turbulence”, *Lecture Notes in Math.*, vol. 898, Springer, New York, 1981.

[21] T. Sauer, J.A. Yorke, and M. Casdagli, “Embedology,” *J. Statistical Physics*, vol. 65, pp. 579–616, 1991.

[22] J. Iwanski and E. Bradley, “Recurrence plot analysis: To embed or not to embed,” *Chaos,* vol. 8, pp. 861–871, 1998.

[23] X. Feng, O. Senyana, “Mining multiple temporal patterns of complex dynamic data systems”, *Proceedings IEEE Symposium on Computational Intelligence and Data Mining*, pp. 411-417, 2009.

[24] A. M. Fraser and H. L. Swinney, “Independent coordinates for strange attractors from mutual information,” *Phys. Rev.*, A 33, pp. 1134-1146, 1986.

[25] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, “Determining embedding dimension for phase-space reconstruction using a geometrical construction,” *Phys. Rev.,* A 45, pp. 3403, 1992.

[26] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer, New York, 2006.

[27] R. Fletcher, *Practical Methods of Optimization*, 2nd ed., John Wiley & Sons, New York, 1987.

[28] S.M. Pandit and S.-M. Wu, *Time Series and System Analysis with Applications*, John Wiley & Sons, New York, 1983.

[29] G. Box and G. Jenkins, *Time Series Analysis: Forecasting and Control*, Holden-Day, San Francisco, 1976.

[30] S. Haykin, *Adaptive Filter Theory*, 3rd ed., Prentice Hall, Upper Saddle River, 1996.

[31] R. Paredes, E. Vidal, "Learning weighted metrics to minimize nearest neighbor classification error," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28(7), pp. 1100–1111, 2006.

[32] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*, Morgan Kaufmann, San Francisco, 1988.

[33] V.N. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, New York, 1998.

[34] V.N. Vapnik, *The Nature of Statistical Learning Theory*, Springer Verlag, New York, 2000.

[35] C.J. Burges, B. Scholkoff, "Improving the accuracy and speed of support vectors learning machines," *Advances in Neural Information Processing Systems 9*, pp. 375–381, MIT Press, Cambridge, 1997.

[36] C.J. Burges, "Geometry and invariance in kernel based methods," *Advances in Kernel Methods: Support Vector Learning*, MIT Press, Cambridge, 1999.

[37] C.J. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Disc.*, vol. 2, no. 2, pp. 1–47, 1998.

[38] B. Scholkoph, A.J. Smola, R.C. Williamson, P.L. Bartlett , "New support vector algorithms," *Neural Computation*, vol. 12, pp. 1207–1245, 2000.

[39] C. Cortes and V. Vapnik, "Support vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[40] A. J. Smola, T. Friess, and B. Scholkopf, "Semiparametric support vector and linear programming machines," *NeuroCOLT Technical Report NC-TR-98-021*, Royal Holloway College, University of London, UK, 1998.

[41] E. Parzen, "On the estimation of a probability density function and mode," *Ann. Math. Stat.,* vol. 33, pp. 1065–1076, 1962.

[42] Y.-S. Huang, C.-C. Chiang, J.W. Shieh, E. Grimson, "Prototype optimization for nearest neighbor classification," *Pattern Recognition*, vol. 35, pp. 1237–1245, 2002.

[43] C.-L. Liu, H. Sako, H. Fusisawa "Discriminative learning quadratic discriminant function for handwriting recognition," *IEEE Transactions on Neural Networks*, vol. 15(2), pp. 430–444, 2004.

[44] J. McNames, "A fast nearest neighbor algorithm based on principal axis search tree,"

IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23(9), pp. 964–976, 2001.

[45] A. Papoulis, *Probability Random Variables and Stochastic Processes*, 3rd ed., McGraw-Hill, New York, 1991.

[46] C. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, UK, 1995.

[47] J. Lin, E.J. Keogh, L. Wei, and S. Lonardi, "Experiencing SAX: A novel symbolic representation of time series," *Data Min. Knowl. Discov.*, vol. 15, Issue 2, pp. 107–144, 2007.

[48] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "An online algorithm for segmenting time series," *Proceedings IEEE Int'l Conf. Data Mining*, pp. 289-296, 2001.

[49] H. Wang, W. Fan, P.S. Yu and J. Han, "Mining concept-drifting data streams using ensemble classifiers," *Proceedings ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Ming*, pp. 226-235, 2003.

[50] H. Wang, J. Yin, J. Pei, P.S. Yu, and J.X. Yu, "Suppressing model overfitting in mining concept-drifting data streams," *Proceedings ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, pp. 736–741, 2006.

[51] X. Gu and H. Wang, "Online anomaly prediction for robust cluster systems," *Proceedings IEEE Int'l Conf. Data Eng.*, pp. 1000-1011, 2009.

[52] T.K. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Mag*., vol. 13, pp. 47–59, 1996.

[53] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Statistical Society*, Series B, vol. 39, no. 1, 1977.

[54] J. Friedman, T. Hastie, R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *The Annals of Statistics*, vol. 28(2), pp. 337–407, 2000.

[55] D. Sciamarella and G. Mindlin, "Unveiling the topological structure of chaotic flows from data," *Phys. Rev.* E, vol. 64, pp. 036 209:1–7, 2001.

[56] R.J. Povinelli, "Time series data mining: Identifying temporal patterns for characterization and prediction of time series events," *Ph.D. Dissertation*, Marquette University, Milwaukee, WI, 1999.

[57] S.M. Weiss and N. Indurkhya*, Predictive Data Mining: A Practical Guide*, Morgan Kaufmann, San Francisco, 1998.

[58] C. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2006.

[59] T. Hastie, R. Tibshirani, J. Friedman*, The Elements of Statistical Learning: Data Mining, Inference and Prediction*, Springer, New York, 2001.

[60] C.-C. Chang, C.-J. Lin, "Training support vector classifiers: Theory and algorithms," *Neural Computation,* vol. 13(9), pp. 2119–2147, 2001.

[61] P.-H. Chen, R.-E. Fan, C.-J. Lin, "A study on SMO-type decomposition for support vector machines," *IEEE Transactions on Neural Networks*, vol. 17(4), pp. 893–908, 2006.

[62] J. Cid-Sueiro, J.I. Arribas, S. Urban-Munoz, A.R. Figueras-Vidal, "Cost functions to estimate a-posteriori probabilities in multi-class problems," *IEEE Transactions on Neural Networks*, vol. 10(3), pp. 645–656, 1999.

[63] D.J. Crisp, C.J. Burges, "A geometric interpretation of ν-SVM classifiers," *Proceedings of Neural Information Processing*, vol. 12, MIT Press, Cambridge, 1999.

[64] J.-X. Dong, A. Krzyzak, C.-Y. Suen, "Fast SVM training algorithm with decomposition on very large data sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27(4), pp. 603–618, 2005.

[65] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd ed., Academic Press, Waltham, 1990.

[66] S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, K.R.K. Murthy, "A fast iterative nearest point algorithm for support vector machine classifier design," *IEEE Transactions on Neural Networks*, vol. 11(1), pp. 124–136, 2000.

[67] D.G. Luenberger, *Optimization by Vector Space Methods*, John Wiley & Sons, New York, 1969.

[68] D.J.C. MacKay, *Information Theory, Inference and Learning Algorithms,* Cambridge University Press, Cambridge, England, 2003.

[69] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," *Technical Report, Microsoft Research*, MSR-TR-98-14, April 21, 1998.

[70] P. Bartlett, S. Boucherou, G. Lugosi, "Model selection and error estimation," *Machine Learning*, vol. 48, pp. 85–113, 2002.

[71] Y. Quilfen, A. Bentamy, P. Delecluse, K. Katsaros, and N. Grima, "Prediction of sea level anomalies using ocean circulation model forced by scatterometer wind and validation using topex/poseidon data," *IEEE Transactions on Geoscience and Remote Sensing,* vol. 38, no. 4, July 2000.

[72] E. Bauer, R. Kohavi, “An empirical comparison of voting classification algorithms: bagging, boosting, and variants,” *Machine Learning*, vol. 36, pp. 105–139, 1999.

[73] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, pp. 5–32, 2001.

[74] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, “Choosing multiple parameters for support vector machines,” *Machine Learning*, vol. 46, pp. 131–159, 2002.

[75] K. Duan, S.S Keerthi, A.N. Poo, “Evaluation of simple performance measures for tuning SVM hyper-parameters,” *Neurocomputing*, vol. 51, pp. 41–59, 2003.

[76] J.B. Gomm, D.-L. Yu, “Selecting radial basis function network centers with recursive orthogonal least squares training,” *IEEE Transactions on Neural Networks*, vol. 11(2), pp. 306–314, 2000.

[77] A.K. Jain, P.W. Duin, J. Mao, “Statistical pattern recognition: A review,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22(1), pp. 4–37, 2000.

[78] H. Kang, S. Lee, “An information-theoretic strategy for constructing multiple classifier systems,” *Proceedings of the 15th International Conference on Pattern Recognition (ICPR)*, vol. 2, pp. 483–486, 2000.

[79] S.S. Keerthi, K.-B. Duan, S.K. Shevade, A.N. Poo,“A fast dual algorithm for kernel logistic regression,” *Machine Learning*, vol. 61, pp. 151–165, 2005.

[80] L.I. Kuncheva, C.J. Whitaker, “Measures of diversity in classifier ensembles,” *Machine Learning*, vol. 51, pp. 181–207, 2003.

[81] L. Mason, J. Baxter, P. Bartleet, M. Frean, “Boosting algorithms as gradient descent,” *Neural Information Processing Systems*, vol. 12, 2000.

[82] D. Miller, A. Rao, K. Rose, A. Gersho, “A global optimization technique for statistical classifier design,” *IEEE Transactions on Signal Processing*, vol. 44(12), pp. 3108–3122, 1996.

[83] Z.-H. Zhou, X.-Y. Liu, “Training cost sensitive neural networks with methods addressing the class imbalance problem,” *IEEE Transactions on Knowledge Data Engineering*, vol. 18(1), pp. 63–77, 2006.

[84] W. Zhang, X. Feng and N. Bansal, “Detecting temporal patterns using RPS and SVM in the dynamic data systems,” *Proceedings of IEEE International Conference on Information and Automation*, pp. 209–214, 2011.

[85] N.K. Bansal, X. Feng, W. Zhang, W. Wei, Y. Zhao, "Modeling temporal pattern and event detection using hidden Markov model with application to a sludge bulking data," *Procedia Computer Science*, vol. 12, pp. 218–223, 2012.

[86] W. Zhang and X. Feng, "Predictive temporal patterns detection in multivariate dynamic data system," *Proceedings of 10th World Congress on Intelligent Control and Automation (WCICA),* pp. 803–808, 2012.

[87] W. Zhang and X. Feng, "Pattern identification using reconstructed phase space and hidden Markov model," *Proceedings of 11th International Conference on Machine Learning and Applications (ICMLA),* pp. 374–379, 2012.

[88] W. Zhang and X. Feng, "Event Characterization and Prediction by Detecting Temporal Patterns Using Reconstructed Phase Space and Gaussian Mixture Model in Dynamic Data System," *IEEE Transactions on Knowledge and Data Engineering,* to appear.

APPENDICES

Publications from this Dissertation

[1] W. Zhang and X. Feng, "Pattern identification using reconstructed phase space and hidden Markov model," *Proceedings of 11th International Conference on Machine Learning and Applications (ICMLA),* pp. 374–379, 2012.

[2] W. Zhang and X. Feng, "Event Characterization and Prediction by Detecting Temporal Patterns Using Reconstructed Phase Space and Gaussian Mixture Model in Dynamic Data System," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, to appear.

[3] W. Zhang and X. Feng, "Predictive temporal patterns detection in multivariate dynamic data system," *Proceedings of 10th World Congress on Intelligent Control and Automation (WCICA),* pp. 803–808, 2012.

[4] W. Zhang, X. Feng and N. Bansal, "Detecting temporal patterns using RPS and SVM in the dynamic data systems," *Proceedings of IEEE International Conference on Information and Automation*, pp. 209–214, 2011.

[5] N.K. Bansal, X. Feng, W. Zhang, W. Wei, Y. Zhao, "Modeling temporal pattern and event detection using hidden Markov model with application to a sludge bulking data," *Procedia Computer Science*, pp. 218–223, 2012 .