TIME SERIES DATA MINING OF STRUCTURE CHANGES USING DYNAMIC SYSTEMS

by

JUNFENG QU

(Under the Direction of Hamid R. Arabnia)

ABSTRACT

This research emphasizes discovery of important changes in structure of time series data. These structural changes imply the loss of customary patterns and the appearance of a novel pattern that has not been experienced previously. Most previous work in this area has been concentrated on identifying previously known or predefined patterns. The major distinction of my research is that the framework offers the ability to discover internal structural changes of time series online dynamically without statistical assumptions about data. The internal structure within time series can be used to improve solutions and provide important insights into the problem domains. Analysis of structured time series data is widely used for many applications, such as economic forecasting, stock market analysis, and networks, etc. This dissertation introduces our research on high-precision modeling, prediction, similarity matching of time series data with consideration of internal structures of data.

Our objectives include: (i) formulating a framework for online dynamic gray modeling of time series data streams, (ii) analyzing and characterize the structure of time series stream data using reference and test models, (iii) developing real-time prediction methods based on the

online modeling results of corresponding internal structure, and (iv) developing a real-time online similarity matching method that considers the identified internal structures of the time series.

We have developed an integrated online structural changes mining (SCM) framework to achieve these objectives. The framework is composed of (a) a dynamic gray model (DGM) that captures the internal structure of time series data online, (b) an algorithm that whitens the incoming data into structures based on a reference model and a test model according to the underlying DGM, (c) a set of analytical methods for data analysis, including online subsequence matching, which generates dynamic query subsequences, defines new subsequence similarity measures, and performs similarity matching with consideration of the internal structures of time series data.

This framework is very useful for real-time systems where response time is critical. We have applied the framework to multiple problem domains, such as financial data analysis and network traffic analysis.

INDEX WORDS: time series, data mining, dynamic system, gray model, similarity matching

TIME SERIES DATA MINING OF STRUCTURE CHANGES USING DYNAMIC SYSTEMS

by

JUNFENG QU

B. Eng. East China University, China 1990

M. Eng. East China University, China 1994

M. Sc. University of Georgia, 2000

A Dissertation Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

ATHENS, GEORGIA

2006

TIME SERIES DATA MINING OF STRUCTURE CHANGES USING DYNAMIC SYSTEMS

by

JUNFENG QU

Major Professor:	Hamid R. Arabnia
Committee:	Khaled Rasheed Jack Houston

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
May 2006

ACKNOWLEDGEMENTS

I would like to thank Dr. Hamid R. Arabnia for the encouragement, support, and direction he has provided during the past three years. His insightful suggestions, enthusiastic endorsement, and proverbs have made the completion of this research possible. I also owe a debt of gratitude to my committee members, Dr. Khaled Rasheed and Dr. Jack Houston, who have helped me to expand the breadth of my research by providing me insights into their area of expertise.

I thank Dr. Robert Robinson for discussing mathematical modeling in the beginning of the dissertation, and I am grateful to the University of Georgia for its financial support during my studies.

I am deeply grateful to my wife, Li, for her generous ongoing moral support and acceptance of my taking time away from our family. I am also grateful to my parents-in-law for their understanding and support for my family and taking care of my son, David, during the elaboration of this dissertation.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

Time series data mining is the combination of time series analysis and data mining fields. The time series data mining does not have traditional time series constraints such as stationarity, linearity, and data distribution, etc., by incorporating data mining concepts and techniques. In this chapter, the definition of time series is reviewed, the general goal of the data mining is discussed, and the problem statement is finalized.

A time series is a sequence of data observed and ordered in time.

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

where *t* is a time index, and N is the number of observations. The components of X can be any observable variable. Theoretically, *X* can be seen as a continuous function of time variable t. Generally, time is viewed in terms of discrete time steps. The time interval can be constant or nonuniform. If *X* contains only one component, it is called a univariate time series. When several time series variables interact with each other, a multivariate time series is observed.

Researchers study systems that generate time series and evolve over time, and hope to discover underlying principles, patterns, and generating mechanisms. Models are often built by researchers and used for predicting and controlling the systems. The better the understanding of the operating mechanism of systems that generate the time series, the more accurately predictability and controllability can be achieved.

The most popular traditional time series analysis methods include Autoregressive Moving Average (ARMA), Autoregressive Integrated Moving Average (ARIMA), Autoregressive Conditional Heteroskedasticity (ARCH), and Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model[1-3]. However, most statistical time series analysis methods such as ARIMA are limited by the requirement of stationarity of the time series data, Gaussian distribution, and independence of residuals. The stationarity in statistics means that the statistical characteristics of time series data remain constant over time. Residuals are the errors between the observed values of time series and the values generated by the model. Residuals are often required to be uncorrelated, independent, and normally distributed.

Data mining is a set of techniques that analyze data to uncover hidden patterns[4, 5]. Like gem mining is a search for gems from sands, data mining is the search for gems of information from massive amounts of information. In time series data mining, these gems are often referred to as events or changes. As a gem is hidden underground, the gemstone of information is hidden in data. As goals of prospectors are different, such as seeking silver, gold or oil, the mining processes should be variant accordingly. The mining of gemstones of information that is desired should also be defined clearly. Therefore, in Chapter 3, we defined our gemstone of information to be mined after review of techniques of time series analysis and data mining in Chapter 2. Without clear definition of what is to be discovered, there is no way to know which and when the gemstone of information is going to be discovered under what conditions.

Our time series data mining of structural changes framework has its foundations in several fields. It builds upon concepts from data mining [4, 5], time series analysis [1, 3, 6], non-linear dynamics and dynamic systems [7-10], and gray systems [11-13]. From time series analysis comes the theory for analyzing linear, stationary time series. From time series analysis limitations such as statistical assumptions suggest the possibility of a new approach to analyze time series data. From data mining comes the discovery of hidden temporal pattern and events. From the dynamic systems and nonlinear dynamics comes the new approach to mine the internal structure of time series, which defines the meaning of time series and allows more accurate forecasting. From gray systems come the limitations of the accurate description and building of a model in neural network or machine learning methods when the data are rare. Furthermore, the motivation to build a new distance measure in similarity matching arises from the fact that general dimensionality reduction techniques do not consider the internal structure of time series and the relative position of each end point being compared. The gray systems also provide the power of modeling time series data that contain a lot of noise. Introduction of dynamic adaption to the gray systems would give the cognitive system the power to understand the changing structure of drifting parameters characteristic of time series evolution, especially when the systems generating the time series are not necessarily linear or stationary.

A number of techniques have been suggested in the literature for detecting novelties, anomalies, and faults in monitored systems [14-18]. These include control charts, model based methods, knowledge-based expert systems, pattern recognition and cluster analysis, hidden Markov models, and neural networks. Most existing methods require either prior knowledge about various novelty conditions or precise theoretical model of the monitored system. A robust

method, however, should detect any unacceptable (unseen) change rather than looking for specific (known) abnormal activity pattern. The main contributions of our researches include:

- Statistical assumptions/independent dynamic time series modeling.
- Dynamic adaptive structural change detection with a threshold that is easy to set up.
- Structural-driven dimension reduction with consideration of internal structure of time series.
- Two-dimensional comparison measures to evaluate and compare structural mining results
- Online event-driven time series clustering and similarity search based on our newly defined metric function.

Problem Statement

Given a time series, suppose there is an unknown function O that describes operating mechanism of system activity of time series denoted as $O(X_t)$. There exits a cognitive function G, such that $G(O(X_t))$ continuously whitening the operating mechanism of the system. The structural change of a system that generates the time series at time *t* if and only if $D(G(X_t), O(X_t)) > \delta$, where $\delta > 0$, δ is the tolerance and $D(\bullet)$ is the distance measure.

Therefore, the goal of the research is to discover a series of structural change points (a sequence of events, as defined in chapter 3), given a time series, in an online fashion. Once a sequence of events is defined, such as $E = \{e_{t_1}, e_{t_2}, ..., e_{t_m}\}$, where each e_{t_i} is an event and $t_i < t_{i+1}$,

the clustering and similarity matching of the time series can be performed after their historical events are discovered using our newly defined distance measure function discussed in Chapter 5.

Objectives & Methodologies:

Like the goal of mining gold from sand is to find nuggets, the objective of this study is to discover changes of structures in the data stream. These structures help researchers to gain insight view of the data generating mechanism and understanding the changes. Once the structures are discovered, our second goal is to cluster and match the similar time series.

In order to achieve these goals, a dual-model approach is developed based on dynamic gray systems. The algorithms fully utilize the high-precision property of the dynamic gray models. The difference between the reference model and test model monitors the structural changes of the data stream based on the distance measure we created. The data are then converted into a smaller dimension by representing the data by these structural change points. Therefore, we could perform the similarity search and clustering on this much smaller dimension space obtained. A new distance metric is also developed to take the structures into consideration during similarity matching and clustering.

Finally these algorithms and parameter sensitivity are evaluated against synthetic time series and real world time series data. In order to compare the results, a two-dimension evaluation space is created to compare different results.

Dissertation Structure

The dissertation is composed of eight chapters. Chapter 2 overviews the general techniques used in time series analysis. After the reviews of the stationary and nonstationary, time domain, and spectral domain analysis of time series analysis, the data mining technologies are discussed; the time series data mining is further reviewed later.

Chapter 3 defines key Time Series Data Mining concepts, such as events, temporal pattern, structural change, similarity matching, and clustering. After definitions of each concept, the latest research studies are summarized and discussed in detail.

Chapter 4 elaborates the dynamic system and dynamic gray system that are used in the dissertation as a cognitive system to whiten the underlying generating mechanism of the time series. The general methodologies and algorithms to build a static and a dynamic cognitive system are discussed in detail.

Chapter 5 establishes the fundamental time series data mining of structural changes and clustering methodology. The fundamentals of defined distance measure, both for the whitening function and the similarity clustering, are introduced. The theory behind the measures is explained.

Chapter 6 extends the time series structural change mining and similarity matching algorithms with their experimental results on synthetic time series data. The experimental results

on both structural change identification in time domain and phase space are discussed. The results of similarity measure with synthetic examples are also presented.

Chapter 7 presents the results from examples of real world data, such as financial time series structural change mining and network traffic flow time series structural change mining. The financial time series clustering and similarity matching with event-driven measure are also discussed in experimental analysis.

In chapter 8, a summary of the dissertation and a discussion of future work are included.

CHAPTER 2

REVIEW OF TECHNIQUES

Time Series Preliminaries

A time series is a sequence of values that are recorded at certain intervals. These values are usually real numbers, and the intervals are often regular, occurring in forms such as yearly, monthly, weekly, daily, hourly, etc. Irregularly recorded data also occur, at times, in the real world; however, these data are often interpolated from sequences of values to regular intervals before a time series analysis is performed.

Time series data are perhaps the most frequently encountered type of data. The time series data touch almost every aspect of human life, including computer science, economics, finance, aerospace, social sciences, and other disciplines. The most commonly seen time series is perhaps financial time series data observed in various time intervals, such as hourly, daily, monthly, or even yearly. Figure 2.1 plots the daily stock price and transaction volume of MSFT from Nov. 2004 to June 2005. Figure 2.2 shows the monthly high, low, and average temperatures at the Athens, Georgia Airport in 2004. These kinds of climatological time series data are often mined for predictive purposes. Figure 2.3 plots the hourly Kbytes transferred on the web server of http://www.fungen.org. Time series data often contain valuable information that can be used for predicting, resource reallocating, business planning and customer-based marketing, web mining and engineering, user pattern analysis, etc.

Figure 2.1 Daily stock price movement

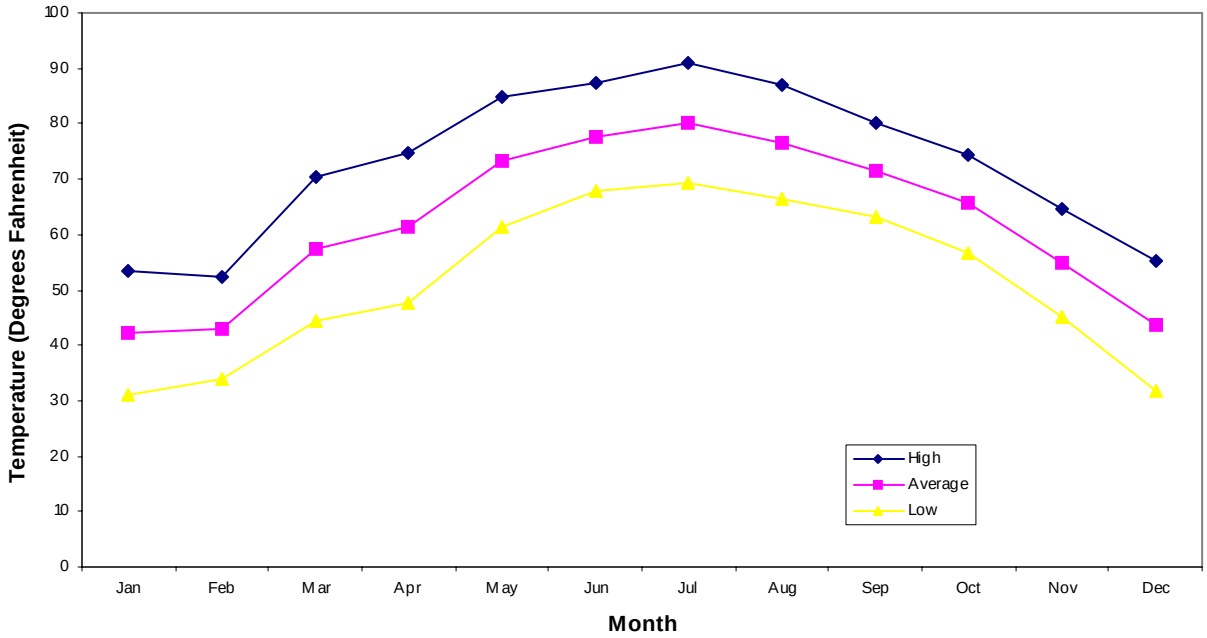

Figure 2.2. Monthly high, low, and average temperatures in Athens, Georgia, in 2004

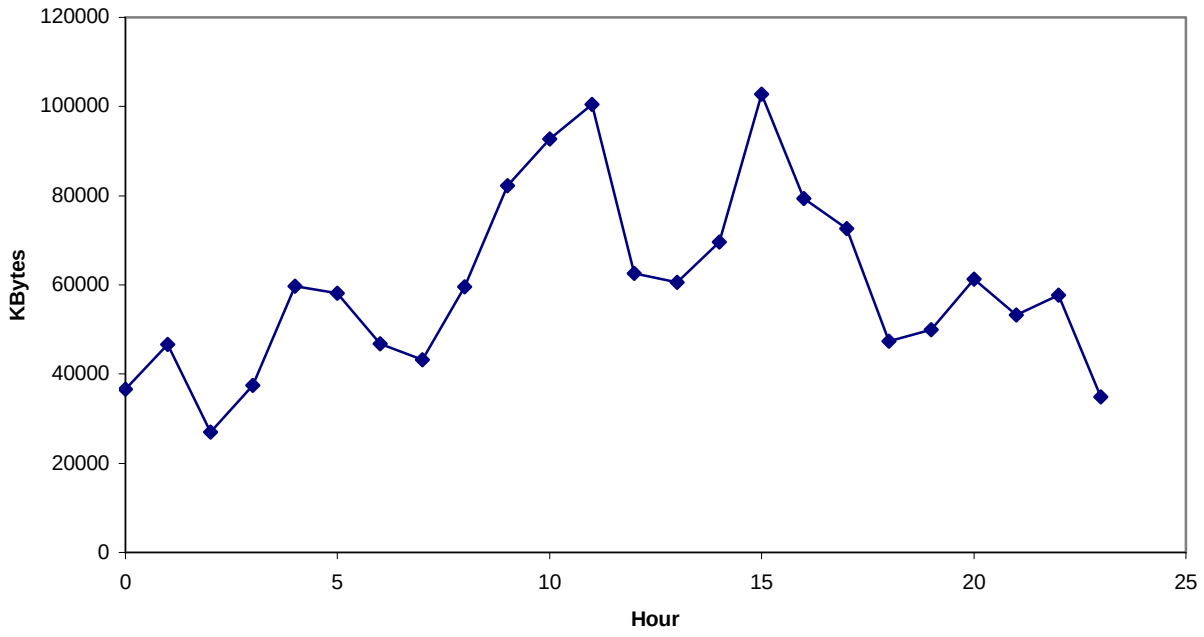

Figure 2.3. Hourly Kbytes transferred by Web server at http://www.fungen.org

Time series analysis is fundamental to engineering, science, and business. Researchers study the systems as they evolve over time. The researchers hope to discover these underlying rules and mechanisms during the evolutionary process of time series and attempt to develop models that are useful for predicting or controlling time-series generating systems. Most of the work in time series analysis, both in statistical and knowledge-discovery aspects, focuses on the following purposes:

- Predicting the future values of the time series;
- Classifying the time series into predefined classes;
- Modeling the time series in terms of parameters;
- Matching time series to one another.

Predicting future values of a time series is a problem of function approximations that estimate the future value(s) based on historical data and some time-independent variables. Usually, the evaluation of forecasting performance is measured by computing an error, *E*, over a number of time series elements among the estimated elements of the model and the actual sequence elements. The typical distance measure – Euclidean distance – can be use here, also. The most common approach is to find the least squared error between the estimated and the actual values.

Classification of time series can be expressed as the problem of finding a function that assigns one of several classes to a time series. Classification can be viewed as a special case of function approximation, where the function to be approximated maps continuous vectors onto single-valued ones. Trend analysis and discovery of seasonality patterns are both applications of classification.

Modeling time series enables us to describe the series in terms of parameters. Essentially, it is a generalization of forecasting. The time series model is capable of generating the series by successively substituting input by estimates.

Matching one time series to another time series can be viewed as similar time series pattern matching, where separate models describe the time series and then employ a mapping function among the series.

Practically, an exact model of the time series is not a realistic goal. The residual error can't be removed even in the most optimized model. Many forecasting models estimate both the forecast value and the expected disturbance from noise. The general methods in time series analysis include Autoregressive Integrated Moving Average (ARIMA) method, the structural time series state method, non-linear methods such as neural network, generic algorithm, and spectral decomposition analysis.

Traditionally, there are two ways to analyze time series data: time domain analysis and frequency domain analysis. Time domain analysis examines how a time series evolves over time using methods such as linear regression and autocorrelation analysis. Frequency domain analysis, which is often known as spectral analysis, focuses on how periodic components at different frequencies describe the evolution of a time series. In the following sections, most common time series analysis methods in time domain and frequency domain will be examined.

Traditional time domain analysis includes statistical time series analysis and modeling. Statistical time series analysis includes linear/non-linear regression, ARIMA, etc., that analyze stationary time series. The ARIMA method was proposed by Box-Jenkins[1] and had been widely used in the area of time series modeling. The ARIMA method assumes that a series can be reduced to a stationary time series by differencing, logarithm, or other detrending methods. In this section, only the ARIMA method (most commonly used) is summarized.

The ARIMA method involves finding solutions to the differential equation of the general ARIMA model of order (p, P, q, Q), which is

$$\phi_p(B)\phi_p(B^L)x_t = \delta + \theta_q Q(B^L)a_t$$

Where:

$\phi_p(B) = (1 - \phi_1 B - \phi_2 B^2 - ... - \phi_p B^p)$ is the nonseasonal autoregressive operation of order p models low-order feedback responses:

$\phi_p(B^L) = (1 - \phi_{1,L} B^L - \phi_{2,L} B^{2L} - ... - \phi_{p,L} B^{pL})$ is the seasonal autoregressive operator of order *P* models feedback response that occur periodically at seasonal intervals,

$\theta_q(B) = (1 - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q)$ is the nonseasonal moving average operator of order *Q* that models low-order weighted average responses,

$\theta_Q(B^L) = (1 - \theta_{1,L} B^L - \theta_{2,L} B^{2L} - ... - \theta_{Q,L} B^{QL})$ is the seasonal moving average operator of order *Q* that models the seasonal weighted average responses, and

$\delta = \mu\phi_p(B)\phi_p(B^L)$ is a constant term, where μ is the true mean of the stationary time series being modeled,

$\phi_1, \phi_2, ..., \phi_p : \phi_{1,L}, \phi_{2,L}, ..., \phi_{p,L} : \theta_1, \theta_2, ..., \theta_q : \theta_{1,L}.\theta_{2,L}, ..., \theta_{Q,L}$ and δ are unknown parameters that must be estimated from sample data, and $a_t, a_{t-1}, ...$ are random shocks that are assumed to be statistically independent of each other; each is assumed to have been randomly selected from a normal distribution that has mean zero and a variance that is the same for each and every time period *t*.

The *B* is called the *backshift operator*. It shifts the subscript of a time series observation backward in time. For example, $By_t = y_{t-1}$ and $B^k y_t = y_{t-k}$.

In ARIMA, the order of the operators is selected specifically, and the parameters are computed from the time series data using optimization methods, such as maximum likelihood and least squares methods.

The ARIMA techniques provided a comprehensive approach for analyzing stationary time series whose residuals are normal and independent. However, the ARIMA method is limited by the requirement of invertibility of the time series; i.e. the system generating the time series must be time-invariant and stable. In addition, the residuals must be independent and distributed normally.

Autocorrelation for time series analysis is a method that studies the sequential progress of events. Predictions for future values of a variable are expressed based on the dependence or regression on the variable's historical values. More details on ARIMA model can be found in [1].

Structural time series models are linear models that study the variables of interest in the time series. Each time series is thought of as a system consisting of four components:

Observed time series = trend + cycle + seasonal variation + irregular fluctuation

(where irregular component reflect non-systematic movement in the time series[19]. In general, in a given time series, all these four components can occur together or in any combinations[20, 21]. By understanding the trend, seasonality, cycles, and irregular components of historical series data, forecasting the current or future values is accomplished.

Structural time series models are estimated by converting the time series into a state space form[19]. In general, the value of a time series at time *t* is plotted against the value of the time series at time *t-1*, using first-order Markov process. After a model has been put into a state space form, the *Kalman* filter can be applied[19]. The *Kalman* filter is a recursive process that computes the optimal estimator of the state vector at time *t* based on the information at time *t*. The *Kalman* filter is an optimal estimator if the disturbances, and the initial state vector are normally distributed. The structural models overcome many of the limitations of the ARIMA models, such as trending. However, as linear models, their prediction capability is limited to a short window of prediction.

Non-linear analysis

Non-linear methods such as neural networks and genetic algorithms have been used to solve these types of problems. Neural network (NN) is a representation of a nonlinear nonparametric model. The procedure allows the data to determine the structure and parameters of the model without any restrictive assumptions. In other words, it is a technique that allows the data to speak for themselves rather than introducing parameter restrictions based on preconceived ideas and hypotheses. Neural networks can be viewed as being a special class of nonlinear parametric models where the process of 'learning' or 'training' is equivalent to the estimation of parameters. Neural networks have attracted the attention of scholars from a variety of fields because of their power in recognizing patterns of behavior.

Dorffner [22] presents an overview of the most common neural network types for time series processing such as pattern recognition and forecasting in spatial-temporal patterns. Neural

networks have been used extensively besides the ARIMA and the state-space linear models mentioned above. Non-linear models (such as neural networks) are more powerful than linear ones, because they are tolerant of problem complexity and are independent from experts, do not assume, theoretically, stationarity, have self-learning and self-tuning capabilities, and do not require knowledge of data relationships. However, as in static pattern recognition, neural networks require much more care and caution than linear methods in that they[22] require a large amount of sample data to solve the parameters of each node. Neural networking can also encounter a variety of problems, such as overfitting and sub-optimal minima as a result of learning (local minimum vs. global minimum) due to many degrees of freedom, and they rely on the trial-and-error method to determine hidden layers and nodes.

Time Series Spectral Analysis

The Fourier transform and wavelet transform are two main tools for spectral analysis of time series. In the following sections, Fourier transform and wavelet transform are briefly discussed.

In time series analysis, a finite sequence of values is observed in discrete time, so Fourier transform cannot be applied directly to a time series. The discrete Fourier transform (DFT) will map a sequence in the time domain into another one in the frequency domain. DFT analysis decomposes a time series into its component parts. Fourier spectral analysis provides information on which waveforms are present in the series, which frequencies are present, and how strong each of the component parts is.

The Discrete Fourier transform (DFT) is defined as follows:

(DFT) Given a time sequence x={x_0, x_1, …, x_{n-1}}, its Discrete Fourier Transform (DFT) is

$$X = DFT(x) = \{X_0, X_1, ..., X_{n-1}\}$$

where

$$X_F = \frac{1}{\sqrt{n}} \sum_{i=0}^{n-1} x_i e^{-j2\pi Fi/n}, F = 0,1,...,n-1$$

The inverse Discrete Fourier Transform (IDFT) of X, x = IDFT(X) is given by:

$$x_i = \frac{1}{\sqrt{n}} \sum_{i=0}^{n-1} X_F e^{j2\pi Fi/n}, i = 0,1,...,n-1$$

We note these relations as:

X = DFT (x) and

x = IDFT(X)

The time series $x = \{x_0, x_1,\ldots, x_{n-1}\}$ can be thought as samples from a function *f(x)* on the interval (0, T), s.t. $x_i = f(iT/n)$.

The DFT preserves he Euclidean distance between time series. For most real time series, the first few coefficients contain most of the information. It is reasonable to expect those coefficients to capture the raw shape of the time series.

The symmetry of the DFT and the IDFT makes it possible to compute the DFT efficiently. Cooley and Tukey [23] published a fast algorithm for Discrete Fourier Transform in 1965. It is known as *Fast Fourier Transform (FFT)*. The FFT method starts by taking this signal and breaking it down into two equal parts that consisted of the odd and even numbered samples, dividing each sample into two *N/2* points DFT. Then the *Recombine Algebra* is used to

recombine the samples in the correct order. The decimation process provides another stage by breaking down the N/2 points DFTs into N/4 points DFTs. This is repeated again and again until we reach a series of two-point DFTs. Since the recombination algebra requires N complex multiplications and there are $log_2 (N)$ stages, the approximate number of complex multiplications is $N \log_2 (N)$. FFT is one of the most important inventions in computational techniques during the last century, since it significantly reduced the computation of the DFT. The time complexity of the DFT for a time series of length n is $O(n^2)$ is reduced to $O(n \log n)$ using the FFT.

Because the Euclidean distance is preserved in the Fourier transform in time or frequency domain, DFT has been used to perform similarity matching, discretizing, dimension reduction and clustering in time series. Agrawa, et.al. [24] used the DFT to map series into the frequency domain and found that most sequences display only a few strong frequencies. They mapped sequences to a lower dimension space by using only the first few Fourier coefficients; then similarity queries could be answered using an R*-tree index to the sequences. Further information on the Fourier transform can be found in any digital signal processing book, such as Robinson [25].

The theory of Wavelet transform was developed based on the Fourier transform and has gained popularity in time series analysis where the series varies significantly over time because the FT is ideal only for periodic time series, and thus is not real as applied to most financial time series.

Wavelets allow a time series to be viewed in multiple resolutions, each of which reflects a different frequency. The wavelet technique assesses averages and differences of a signal, breaking the signal down into a spectrum. In this process, each step of the wavelet transformation produces two sets of values: a set of averages and a set of differences (differences are referred to as wavelet coefficients). Each step produces a set of averages and coefficients that is half the size of the input data. For example, if the time series contains 256 elements, the first step will produce 128 averages and 128 coefficients. The averages then become the input for the next step (e.g., 128 averages resulting in a new set of 64 averages and 64 coefficients). This continues until one average and one coefficient (e.g., 2^0) are calculated.

The configuration of average and difference of the time series is made across a window of values, with most wavelet algorithms calculating each new average and difference by shifting this window over the input data. For example, if the input time series contains 256 values, the window will be shifted by two elements, 128 times, in calculating the averages and differences. The next step of the calculation uses the previous set of averages, also shifting the window by two elements. This process has the effect of averaging across a four-element window. Logically, the window increases by a factor of two each time. In the wavelet literature, this tree structured recursive algorithm is referred to as a pyramidal algorithm.

The power of two coefficients (difference) spectrum generated by a wavelet calculation reflects change in the time series at various resolutions. The first coefficient band generated demonstrates the highest frequency changes, while each later band reflects changes at lower and lower frequencies.

There are an infinite number of wavelet basis functions. The more complex functions (like the Daubechies wavelets) produce overlapping averages and differences that provide a better average than the Haar wavelet at lower resolutions. However, these algorithms are more complicated.

Haar wavelet [26]is the simplest type of wavelet. In discrete form, Haar wavelets are related to a mathematical operation called the Haar transform, which serves as a prototype for all other wavelet transforms. Besides the Haar wavelet, there are a wide variety of popular wavelet algorithms, such as Daubechies wavelets, Mexican Hat wavelets and Morlet wavelets [27, 28]. These wavelet algorithms have the advantage of better resolution for smooth changes occurring in time series, but they have the disadvantage of being more expensive in computation than that of the Haar wavelet. I will briefly summarize the Haar wavelet's algorithm below, followed by an example.

The Haar wavelet algorithms summarized here are applied to time series where the number of samples is a power of two (e.g., 2, 4, 8, 16, 32, 64...). The Haar wavelet uses a rectangular window to sample the time series, with the first pass over the time series using a window width of two. The window width is doubled at each step until the window encompasses the entire time series.

Each pass over the time series generates a *new* time series and a set of coefficients. This new time series results from the average of the previous time series over the sampling window,

with the coefficients representing the average change in the sample window. For example, if we have a time series consisting of the values $v_0, v_1, \ldots v_n$, a new time series, containing half as many points, is calculated by averaging the points in the window. If it is the first pass over the time series, the window width will be two, so two points will be averaged:

for (i = 0; i < n; i = i + 2)

$s_i = (v_i + v_{i+1})/2;$

The wavelet coefficients are computed along with the new average time series values, with the coefficients representing the average change over the window. If the width of window is two, this would be:

for (i = 0; i < n; i = i + 2)

$c_i = (v_i - v_{i+1})/2;$

Let's use an example to illustrate the above algorithm:

Suppose the original time series is defined by eight values, e.g.:

S = (56, 40, 8, 24, 48, 48, 40, 16)

The mean and difference computation is shown in the table below, with differences emphasized in bold and italic type:

56	40	8	24	48	48	40	16
48	16	48	28	***8***	***-8***	***0***	***12***
32	38	***16***	***10***	***8***	***-8***	***0***	***12***
35	***-3***	***16***	***10***	***8***	***-8***	***0***	***12***

It is important to observe that no information has been lost in this transformation of the first row into the fourth row. This means that we can reverse the computation. Beginning with the last row, we compute the first two entries in the third row as 32 = 35+(-3) and 38 = 35-(-3); analogously, the first four entries in the second row are computed as 48 = 32+(16), 16=32 – (16), 48 = 38 + (10), and finally 28 = 38 – (10). Repeating this procedure, we get the first row in the table.

The Haar wavelet transform has a number of advantages, since it is::

- Simple and fast;
- Memory efficient because computation is performed without a temporary array; and
- Exactly reversible without the edge effects that pose a problem in other wavelet transforms.

However, the limitations of Haar transform can pose a problem for some applications. For example, the high frequency coefficient spectrum should reflect all high-frequency changes, but the Haar window is only two elements wide. If a large change takes place from an even value to an odd value, the change will not be reflected in the high-frequency coefficients. The Daubechies D4 algorithm has a slightly higher computational overhead and is conceptually more complex. As the matrix forms of the Daubechies D4 algorithm demonstrate, there is an overlap between iterations in the Daubechies D4 transform step. This overlap allows the Daubechies D4 algorithm to pick up details that are missed by the Haar wavelet algorithm shown in the following figure: 2.4[29].

The red line in the plot below reveals a signal with large changes between even and odd elements. The pink line plots the largest band of Haar wavelet coefficients (e.g., the result of the Haar wavelet function). The green line plots the largest band of Daubechies wavelet coefficients, and the coefficient bands contain information about the change in the signal at a particular resolution.

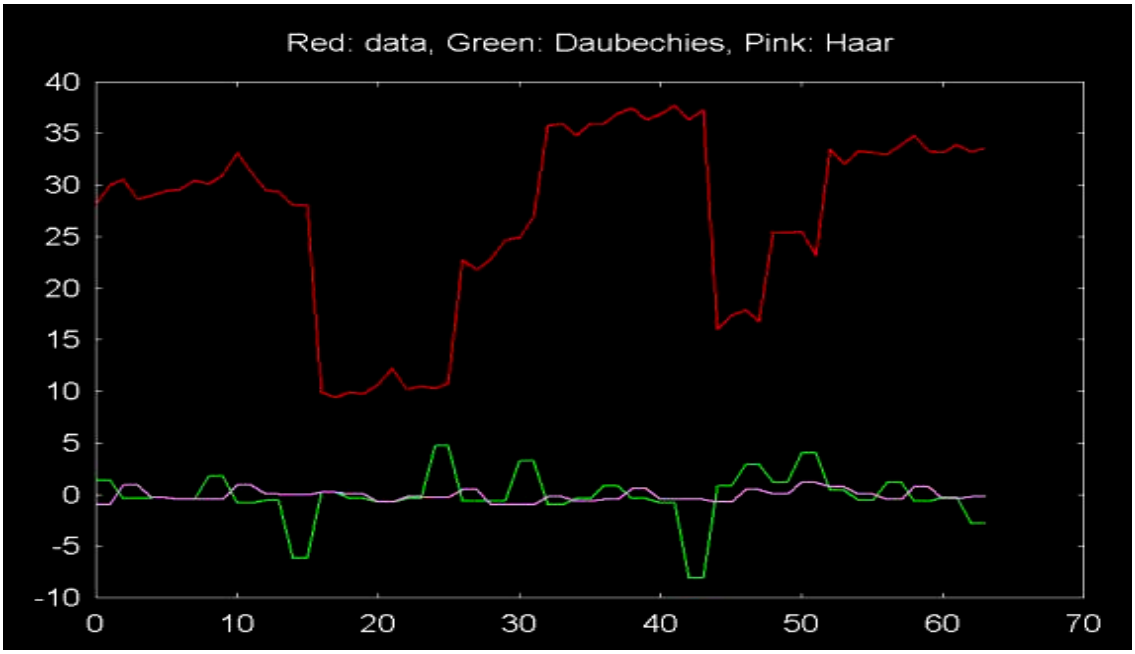

Figure 2.4. Haar wavelet transform example

In this version of the Haar transform, the coefficients show the average change between odd and even elements of the signal. Since the large changes fall between even and odd elements in this sample, these changes are missed in this wavelet coefficient spectrum. These changes would be picked up by lower frequency (smaller) Haar wavelet coefficient bands.

Wavelet transforms have been used for sequence matching[30-33]. These transforms create features that describe properties of the sequence both at various locations and at varying

time granularities. From a knowledge discovery viewpoint, the spectral methods are somewhat indirect.

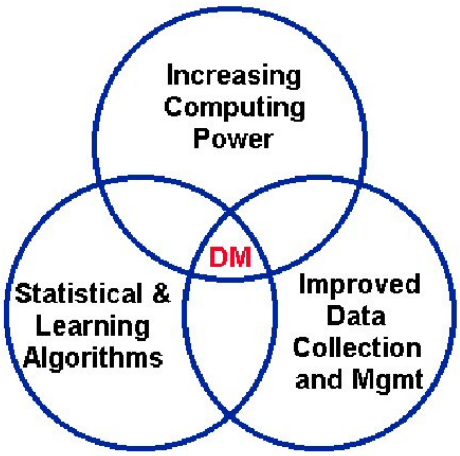

Figure 2.5. Data mining interrelated fields

Data Mining

Data mining is also known as Knowledge-Discovery in Databases (KDD). Data mining is defined by Weiss and Indurkhya as “the search for valuable information in large volumes of data. Predictive data mining is a search for very strong patterns in big data that can generalize to accurate future decisions[5]”. Frawley [34] defined data mining as "the nontrivial extraction of implicit, previously unknown, and potentially useful information from data". Hand, et al, [35] also defined data mining as "the science of extracting useful information from large data sets or databases." Although data mining is defined differently, it is often used in relation to analysis of data and information extraction. Data mining is also applied as a term with varied meanings in a wide range of contexts.

Data mining evolved from several fields (Figure 2.5), including computer science, machine learning and statistics, and database. The goal of data mining is to discover previously

unsuspected relationships, patterns, and feature, which are of interest of value to their owners. In general, data mining can be categorized into event-based data mining and relationship-based data mining. Relationship-based data mining is based on associations. It includes spatial, temporal, and coincidence associations. Spatial associations identify events (for example, astronomical objects) at the same location in the sky. The temporal associations find events occurring during the same or related periods of time; and the coincidence associations use clustering techniques to identify events that are co-located within a multi-dimensional parameter space. The event-based data mining is the research based upon events or trend in data. The definition of events will be discussed later in the Chapter Three of this dissertation.

If the degree of knowledge understood is expressed as a gradient from white to black, where black means unknown and white means known, the event-based data mining can be illustrated in knowledge space as shown in figure 2.6 based on the algorithms used and the events.

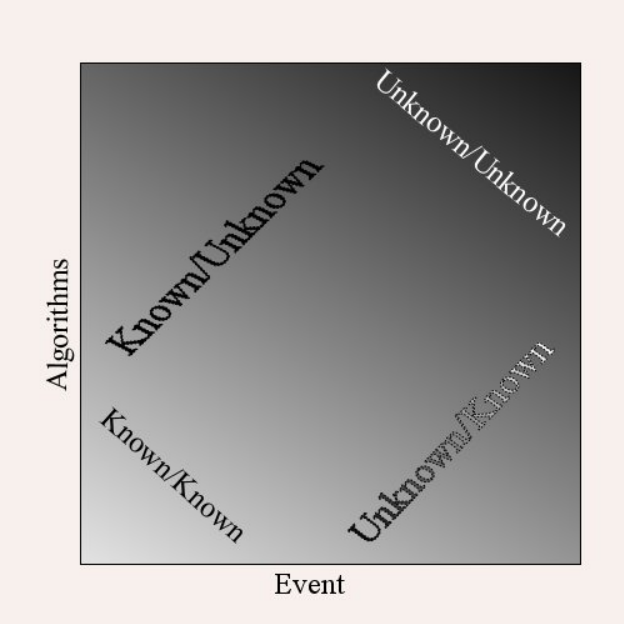

Figure 2.6. Event-based data mining based on algorithms and event

Known events/known algorithms constitute the methodology that uses existing (descriptive) models (descriptive) to locate known phenomena of interest (events) either spatially or temporally within a large data set or database; known events/unknown algorithms is the method that uses recognition and clustering properties of data to discover new observational relationships among known phenomena (events); unknown events /known algorithms use expected physical relationships (predictive models) among observational parameters of phenomena to predict the presence of previously unseen events within a large data set or database; and unknown events/unknown algorithms relies on thresholds or trend to identify transient or otherwise unique ("one-of-a-kind") events in an attempt to discover new phenomena.

Data mining uses techniques, among them association rules, clustering and classification, visualization, decision trees, neural networks to identify novel, hidden, and useful structures or events from data set or large data base[36]. Time series data mining have grown rapidly as an exciting subfield of data mining[37]. There are many reasons for this, including:

- A growth in the volume of data being collected and requiring analysis
- An increase in the general availability of data through the Internet and as a result of E-commerce and inter-enterprise applications
- A growing recognition of the value and commercial advantage that the result of data mining can bring

In the following section, we will review the general time series data mining concepts, techniques and applications.

Time Series Data Mining

Time Series Data Mining is also called sequence data mining. This involves mining a sequence of data that can either referenced by time (time-series, such as stock market and production process data), or simply reveals a sequence of data that are ordered in a sequence. In general, one aspect of mining time-series data focuses on the goal of identifying movements or components that exist within the data (trend analysis). These can include long-term or trend movements, seasonal variations, cyclical variations, and random movements.

Time series data mining has been attracting tremendous amount of attention in data research community [38]. Due to the advances in data storage technologies and the advent of large-scale business and scientific applications, as well as the relatively recent appearance of the Internet, the data for data mining applications have become very large and may contain over several million records; as a result, people have experienced the era of data explosion. Therefore, the sizes of normal databases nowadays might be hundreds of gigabytes or even terabytes. On the other hand, people's abilities to analyze the collected data are limited, and the enormity and complexity of the data involved make the task of data mining prohibitively expensive.

Traditionally, people tried to build models for time series data and then fit the actual observations of sequences into these models. If a model is successful in interpreting the observed time series, the future values of time series can be predicted, provided that the model's assumptions continue to hold in the future. However, these traditional data mining techniques have troubles when they are applied to the data with these characteristics, such as high

dimensionality, non-stationarity[39]. This contradiction creates the need to generate new technologies and tools to analyze the collected data intelligently and efficiently and overcome the difficulties embedded in today's data, which sparks the emergence of knowledge discovery in databases (KDD) and data mining.

Other techniques that are used on these kinds of data include similarity search, sequential-pattern mining, and periodicity analysis. Similarity search is concerned with the identification of a pattern sequence that is close or similar to a given pattern. Sequential-pattern mining has as its focus the identification of sequences that occur frequently in a timer series or sequence of data. Periodicity analysis attempts to analyze the data from the perspective of identifying patterns that repeat or reoccur in a time series.

CHAPTER 3

CONCEPTS IN TIME SERIES DATA MINING

In previous chapters, events, temporal patterns, structural change and similarity matching are presented in Time Series Data Mining (TSDM). In this chapter, these concepts are explained in further detail. Current research on time series data mining is reviewed in relation to each concept, with clear definitions of the concepts presented in the research framework defined.

Event

An event is an important occurrence in a time series. However, the definition of an event depends on the Time Series Data Mining goal and application [40, 41]. For example, in the mining of the weather time series data, a hurricane could be defined as an event. In telecommunication industry time series data, switch failures over time may be defined as events. Other examples of events in the financial time series mining might include sharp rises or falls of a price of the equity being studied or fraudulent credit card transactions.

Given a time series $X = \{x_t, t = 1, ..., n\}$, an event E_t is a time-stamped observation that occurs at time *t* and is described by a set of feature-value pairs defined by the goal of time series data mining. Therefore, an event sequence is a time-ordered sequence of events, $E = \{E_{t_1}, E_{t_2}, ..., E_{t_n}\}$, which includes all events at time t_n. In general, the events are associated with a domain (object D), which is the source or generator of the events[18, 42].

Cox and Lewis[6] proposed the statistical analysis of series of events. Based on the time-dependent Poisson process, generalizations of renewal processes are introduced to study the marginal distribution of the intervals and of the second-order properties of these intervals and also account for factors such as doubly stochastic Poisson process, Wold's Markov process of intervals, Branching renewal processes, and the Semi-Markov process.

Weiss and Hirsh [18, 42, 43] described a learning system that uses a genetic algorithm-based machine learning system to predict rare events with categorical features derived from time-series data by identifying predictive temporal and sequential patterns within time series data. Due to the inherent difficulty of the problem and the limited information revealed by the observed data, it is generally not possible to make predictions with high accuracy. The authors proposed an algorithm to learn noise-tolerant rules and return multiple solutions, trading off precision and recall in different ways.

Another type of research that uses the genetic algorithm method to search for optimal temporal pattern clusters that are significant for characterizing and predicting events (the important occurrences) is the study done by Povinelli and Feng [41, 44] and Povinelli [45]. They have proposed a specific time series data mining framework aimed at a priori identification of the future events via the pre-defined (desired) properties. The key idea is the selection of a number of patterns from past evolution that maximize the event characterization function defined a priori on the phase space.

Gao, Kinouchi, and Zhao [46] presented an event detection method using neural network for time series analysis to capture homeostatic dynamics of the system under the influence of exogenous event. The results showed that the BP neural networks could identify the properties of homeostatic dynamics and model the dynamic relation between endogenous and exogenous variables in financial time series. Therefore, the authors concluded that the neural network is useful for event extraction from time series by capturing and modeling the predictable deterministic and unpredictable random components.

Knovalov [47] proposed a new general procedure for a priori selection of more predictable events from a time series of an observed variable. The procedure first implies the creation of two neural network-based forecasting modes, one of which is aimed at the prediction of conditional mean and others – conditional dispersion, and then elaboration on the rule for future event selection into groups of more and less predictable events. The main idea of the method is creation of the special neural network model aimed to predict absolute errors of forecasts, then discriminating between more and less predictable events, achieved by a priori assessing an error of a forecast of a given event.

Guralnik and Srivastava [48] presented an approach for event detection from time series data. The iterative algorithm fits a model to a time segment and uses a likelihood ratio to determine whether the segment should be partitioned further. The experimental results suggest that the algorithms are able to correctly identify change-points in cases where signal-to-noise ratio is not too low.

Wu, Salzberg and Zhang [49] defined an event as a new potential end point which is being identified, with no pruning needed by their 3-tier online simultaneous segmentation and pruning algorithm over a massive data stream. The algorithm produces a piecewise linear representation of the data stream that features high sensitivity and accuracy. The event-driven subsequent matching and search in the time series is then performed based on the similarity definition that is based on a permutation followed by a metric distance function.

Genetic algorithms and neural networks have been used in the above research to identify, learn. or detect events from a time series, e.g., researches that have been conducted by Weiss and Hirsh [18, 42, 43], Povinelli and Feng[41, 44] and Povinelli [45], Gao, Kinouchi, and Zhao[46] Knovalov[47]. These methodologies used works well in the processing of offline time series data for the model to learn and function. Povinelli, Feng and Knovalov have tried to mine the internal structure change of time series itself, while Povinelli and Feng have attempted to identify the structure in the phase space on which the event prediction is based. Knovalov has proposed to predict absolute errors of forecasts of neural networks with collaboration of another neural network to discriminate high predictable and low predictable events. Guralnik and Srivastava and Wu's algorithms are able to identify and predict events in both incremental and batch (online and offline) modes for time series event detection; neither not requires a prior model. The determination of the threshold in Guralnik and Srivastava's method is performed by trial and error, and the segmentations of time series revealed by Wu's method might not connect with the internal structure change after the time series data is transferred into %b indicator and a time delay exists in the transferred time series.

Temporal Pattern

Pattern is a general term for any recognizable regularity in the data. The Merriam-Webster dictionary [50] defines pattern as “a natural or chance configuration such as frost pattern, the pattern of events” or “a reliable sample of traits, acts, tendencies, or other observable characteristics of a person, group, or institution”. The pattern is often used in combination with time, sound and space. Pattern also helps to describe the structures behind our thinking, such as in the financial time series data; e.g., a head-and-shoulders pattern is defined to describe the shape of the price move like that of head and shoulders. However, there are two opposites of the term “pattern”. If we are not able to recognize any pattern, we may state that there *is* no pattern. The other way often used to describe a pattern is regarded as a separate class, the random pattern.

Temporal patterns are those that may or may not be stable over time. Temporal patterns are probably best explained by using the language of music. They can be defined as consecutive, organized arrangements of sounds on an abstract level. Povinelli and Feng[44, 45, 51] defined a temporal pattern as a hidden structure in a time series that is characteristic and predictive of events; and it is represented in a Q-dimensional real metric space. In time series, a temporal pattern may be described as a set of points that describes the location of these points in terms of the relative distances between one point and another, arranged temporally in terms of time and/or scale. Generally, a temporal pattern can be defined over a multidimensional domain of which at least one dimension is time.

It is very important to identify temporal patterns because in them very little is stable and unchanging, particularly when the objective is to detect some event or action, process it with a

computer, and act upon the event or action in real life. Temporal pattern processing is a challenging task because the information is embedded in time and inherently dynamic [52]. One example of a temporal pattern is a traffic build-up in a telephone switch that may cause the switch to become overloaded.

Temporal pattern processing and identification is a challenging topic, because of the information is embedded in time and is therefore inherently dynamic and not simultaneously available. However, the two characteristics of temporal pattern – (a) temporal order (b) time duration – provide keys to temporal pattern identification and processing. Povinelli, etc. [44, 45, 51, 53] also conducted extensive research on temporal pattern identification, characterization, and matching using pattern wavelets and genetic algorithm. They defined temporal pattern as hidden structure in a time series that is characteristic and predictive of events defined as an important occurrence. In order to link a temporal pattern (past and present) with an event (future), the authors introduced event characterization function that represents the value of future 'eventness' for the current time index. The event characterization function is defined such that its value at *t* correlates highly with the occurrence of an event at some specified time in the future. The authors also presented a new method for temporal pattern matching by pattern wavelets and genetic algorithm. A problem-specific fitness factor is introduced in the feature space. Pattern wavelets yield high fitness value for temporal pattern matching through a threshold process.

Lin and Keogh [54] and Keogh [54, 55] have attempted to find and locate specified previously known patterns in a time series. The temporal features are extracted by Markov

models and expressed via a suffix tree. Then the algorithm that detects surprising patterns in a time series occurring in linear space and time are addressed. Lin further defined the 'motif' as patterns previously unknown but frequently occurring. After the time series are discretized, the authors use Enumeration of Motifs through Matrix Approximation (EMMA) and modified ADM algorithms to search the temporal pattern from time series efficiently.

Singh and Stuart [56] described a pattern recognition-based tool for forecasting. The pattern- matching approach is based on the premise that current structures may be matched with old structures to generate a future prediction. Then, the time series is mapped into a binary series based on the defined direction of change, high or low. The authors showed that the pattern-matching method based on structure-matching performs very well when compared with other established statistical methods used frequently in practice.

Dasgupta and Forrest [57] incorporate the ideas from Immunology into novelty detection in time series data. In the Immunology, "self "is defined as normal data patterns and "non-self" as any deviation exceeding an allowable variation. The novelty detection algorithm is based on the negative-selection mechanism of the immune system that discriminates between self and other. The algorithm proposed relies on a large enough sample of normal data to generate a diverse set of detectors that probabilistically notice any deviation from the normal. The detection system may also be updated by generating shifts due to aging, system modifications, change in operating environments, etc.

Elfeky, etc [58, 59] focused on incremental mining of partial periodic patterns that had been defined by Han [60] in time series. The partial periodic patterns are those that specify the behavior of time series at some but not all points in time. In the papers, several algorithms for incremental mining of partial periodic patterns in time series are proposed and analyzed empirically. The new algorithms allow for online adaptation of the thresholds in order to produce interactive mining of partial periodic patterns.

Wang, etc. [52, 61, 62] have conducted extensive studies on temporal pattern processing with neural networks. A neural network that has a capacity of short-term memory is outlined for temporal pattern processing and recognition. The anticipation model is developed to learn multiple complex sequences sequentially; i.e., the new training does not take place until existing sequences are acquired. The authors discovered that the interference properties of the anticipation model are consistent with human retroactive interference well documented in psychology. The recognition scheme is achieved via Hebbian learning.

Structural time series and Structural Change

Structural time series models are models with are formulated directly in terms of components of interest[19]. Structural time series have a considerable intuitive appeal, particularly for economics and social time series. Furthermore, they provide a clear link with regression models, both in their technical formulation and in the model selection methodology. Structural time series models are appropriate to many subjects, including economics, sociology, and management science, as well as operational research, geography, meteorology, and engineering.

The typical time series of many economic and social time series displays characteristics such as a trend, which represents the long-run movements in the series, and a seasonal pattern that repeats itself more or less every year. A model of the series will be needed to capture these characteristics. There are many ways in which such a model may be formulated, but a useful starting point is to assume that the series may be composed in following way:

Observed series = trend + seasonal + irregular

(where the 'irregular' component reflects non-systematic movements in the series.) The model is an additive and/or a multiplicative form:

Observed series = trend x seasonal x irregular

However, a multiplicative model may be handled within the additive framework by the simple expedient of taking logarithms.

A structural time series model is set up in terms of components that have a direct interpretation. A univariate structural model is not intended to represent the underlying data generation process. Rather, it aims to present the 'stylized facts' of a series in terms of a decomposition into components such as trend, seasonal, and cycle. These quantities themselves are of interest. Furthermore, they highlight the features of a series that must be accounted for by a properly formulated behavioral model. Prediction from a univariate mode is naïve in the sense that it is simply an extrapolation of past movements. Nevertheless, it is often effective and provides a yardstick against which the performance of more elaborate models can be assessed.

The statistical formulation of the trend component is a structural model that needs to be flexible enough to allow it to respond to general changes in the direction of the series. A trend is not seen as a deterministic function of time beyond which the series is constrained to move forevermore. In a similar way, the seasonal component must be flexible enough to respond to changes in the seasonal pattern. A structural time series model therefore needs to be set up in such a way that its components are stochastic; in other words, they are regarded as being driven by random disturbances.

In general, in a given observed time series, all of these components (trend, cycle, seasonal variations, and irregular fluctuation) can occur together or in any combinations (Arnold (1974) [20], Medhi (1982) [21]). One example is the US Nominal Gross National Product (GNP) over the period 1890-1974 in figure 3.1 [63]. There is a clearly detectable trend; in addition, the earlier part of the series shows marked cyclical behavior as the economy moves from boom to recession and back again. Indeed, we would probably have realized this from the economic history of the period without even looking at the graph. Incorporating a cyclical component in a model for US Nominal GNP will therefore play an important role in providing a description of this series, at least in its early stages.

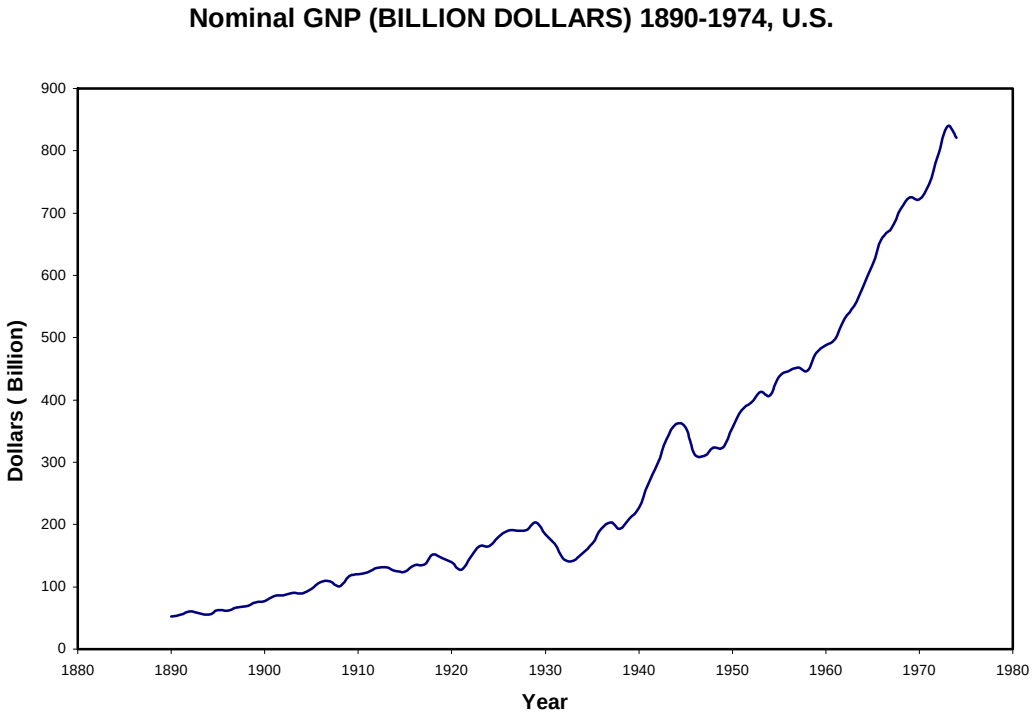

Figure 3.1 Nominal GNP (Billion Dollars) in U.S. from 1890 to 1974

As indicated in the structural time series, we find an internal generation mechanism that produces the seasonal or cyclical patterns in the time series. These patterns often manifest as identified perceptually important points, and the points are often perceptually important in the human identification process; therefore, they should be considered of higher importance than is often accorded to them. This situation is often presented in the financial analysis, such as head-and-shoulder pattern, which is characterized by a few critical points; a head point, two shoulder points, and a pair of neck points are shown as '◯' in Figure 3.2[64].

Figure 3.2. Perceptually Important Points: Head-and-shoulder pattern

If we denote a mechanism that generates a time series, then the structure change shown in the series results from the drifting of the generating mechanism during the evolution of the time series. Statistically, a structure change (a *priori*) can be defined as the discovery of level that change all continuous observations within one level or all continuous observations with more or less the same gradient. Given a cognitive system, such as one created by the human brain, a structure change in a time series means the loss of the patterns that we're used to living with and the appearance of the novel pattern with which we have no experience [65]. Although the structure change defined above is not conventional in statistics or econometrics, it is usual to and compatible with our intuition of structure change, even though, in this case, the defined structure change might not exhibit any precise presentation or specific mathematical form.

Structural changes in a system are often the 'surprise,' 'shock,' or sometimes 'perceptually important' elements that have extensively changed the operating mechanism of the system but can't be dealt with by some intelligent cognition methods such as dynamic adaptive or self-organized training. Let's assume that the operating mechanism of a system that describes that activity of time series by function $F(X_t)$. The cognitive function of the operating mechanism of the system is denoted as $\hat{C}(F(X_t))$. The system has undergone a structure change at time t at a given tolerance level $\delta>0$ if and only if

$$D(F(X_t), \hat{C}(\bullet)) > \delta,$$

where $D(\bullet)$ is a function of distance measure and $\hat{C}(\bullet)$ represents possible cognitive functions.

Detection of structure changes of any observed time series data has a long history in statistics and econometrics, but many issues remainto be settled. Recent extensive use of nonlinear and nonstationary time series models seems to have made many of these issues even more obscure. For example, it is difficult to distinguish a nonstationary time series from a stationary time series with structure breaks [66]. Detecting structural changes of observed time series is a daunting task when the model class is extended from linear to nonlinear and stationary to nonstationary models [65].

A large variety of structural change point detection problems have occurred in time series analysis. The literature on this topic is rapidly growing, mainly due to applications in engineering, financial engineering, and econometrics. The problem is known under different names in these applications, such as segmentation, failure detection, quality control, change-

point detection, and shock detection. The underlying changes are the operating mechanism of the time-series-generated changes, no matter what the applications are. These changes have increased interest in using data mining techniques to extract interesting temporal patterns from temporal sequence [67] and study the actual structural change of the series itself to reveal the internal mechanism of changes. The internal structure and its meaning should be analyzed and modeled for several reasons. First, the model provides a tool for forecasting and monitoring a time series. Second, it helps to understand the underlying changes produced by the corresponding time series. And, third, the structure change often arises from new information involved; therefore, the information embedded in time series could be mined and prove to be not only interesting but even strategically important.

A number of approaches have been proposed to solve the change-point detection problem in the field of statistics [68-72]. The standard assumptions under these algorithms are:

- a known, stationary (usually linear) model can be used to describe the phenomenon, and
- the number of change-points is known *a priori*

The general various change-point models can be classified as likelihood ratio tests, non-parametric approaches, and linear model approaches. These approaches are summarized below.

Parametric Likelihood Ratio Test (LRT) Method.

Assume that $X_1, X_2, ..., X_n$ are independent normal observations with parameters (μ_1, σ^2), (μ_2, σ^2),…, and (μ_n, σ^2). Under hypothesis H_0: $\mu_1 = \mu_2 = ... = \mu_n$ and under the alternative hypothesis, a change point occurs at k^* such that $\mu_1 = \mu_2 = ... = \mu_{k^*} \neq \mu_{k^*+1}, = \mu_n$.

The variance is an unknown. Further, it can be assumed that both the variance and the mean may change at an unknown time, which means that then the maximally selected likelihood ration test is $\max_{1\le k<n} L_k$, where

$$L_k = n\log\hat{\sigma}_n^2 - k\log\hat{\sigma}_k^2 - (n-k)\log\hat{\sigma}_{k^*}^2, \text{ where}$$

$$\hat{\sigma}_n^2 = \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X}_n)^2,$$

$$\hat{\sigma}_k^2 = \frac{1}{n}\left\{\sum_{i=1}^{k}(X_i - \hat{X}_k)^2 + \sum_{i=k+1}^{n}(X_i - X_{k^*})^2\right\}.$$

$$\hat{\sigma}_{k^*}^2 = \frac{1}{n-k}\sum_{i=k+1}^{n}(X_i - X_{k^*})^2,$$

$$\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

$$\hat{X}_k = \frac{1}{k}\sum_{i=1}^{k} X_i, \text{ and}$$

$$X_{k^*} = \frac{1}{n-k}\sum_{i=k+1}^{n} X_i$$

The distribution of $\max_{1\le k<n} L_k$ is approximated to the normal distribution for large sample sizes [73].

Non-parametric Pettitt test:

Pettitt[72, 74] proposed a non-parametric change-point approach in 1979; this approach is summarized below:

Assume that a change-point at k^* that separates a sequence of random variables $X_1, X_2, ..., X_n$ if X_i for $i = 1, 2, ..., k^*$ which share a common distribution function, $F_1(x)$ and X_i for

$i = k^* + 1, k^* + 2, ..., n$ have a common distribution $F_2(x)$ and $F_1(x) \neq F_2(x)$. Under the null hypothesis of no-change, H_0: $k^* = n$, and the alternative hypothesis of change, H_1: $1 \leq k^* < n$ uses a non-parametric statistic.

An appealing non-parametric test to detect a change would be to use a version of the Mann-Whitney two-sample test. Pettitt pointed out that Mann-Whitney type statistics have remarkably stable distribution and provide a robust test of the change point resistant to outliers. Let $D_{ij} = sign(X_i - X_j)$,

Where $sign(x) = \begin{cases} 1, if & x > 0 \\ 0, if & x = 0 \\ -1, if & x < 0 \end{cases}$, then the statistic $U_{k,n} = \sum_{i=1}^{k} \sum_{j=k+1}^{n} D_{ij}$ is equivalent to a Man-Whitney statistic for testing that the two sample $X_1, X_2, ..., X_k$ and $X_{k+1}, X_{k+2}, ...X_n$ come from the same population. The statistic $U_{k,n}$ is considered for the values of *k*, such that $1 \leq k < n$ for the test of null hypothesis H_0, which is change and alternative hypothesis with the statistic

$$K_n = \max_{1 \leq k < n} \left| U_{k,n} \right|$$

The limiting distribution of K_n is approximate to $2 \exp\left\{-6k^2/(n^2 + n^3)\right\}$ for a large n.

Non-parametric F-Chow Test:

The Chow test [75] involves a non-parametric, linear approach. Quandt (1960) [76] extends the analysis of Chow (1960) to the case of unknown break date by computing a sequence of Chow statistics for all possible break dates contained in a restricted interval. Andrews (1993) [77] and the related "average" and "exponential" tests of Andrews and Ploberger(1994) [78] derived the asymptotic distributions which are useful to provide approximations to the finite

sample distribution. The basic Chow test is summarized below. The method is a test for structural breaks that are for stationary variables and a single change point at k^*. Consider the linear regression model with *p* variables as follow:

$$Y_i = \beta_0 + \beta_1 X_{i,1} + ... + \beta_p X_{i,p} + \varepsilon_i, \quad i = 1, 2, ..., n$$

where Y_i is the value of the response variable in the i^{th} observation; $\beta_0, \beta_1, ..., \beta_p$ are parameters; $X_{i,1}, X_{i,2}, ..., X_{i,p}$ are the values of the independent variables in the observations; ε_i is an identically independent random error term that follows the Gaussian distribution of mean of zero and variance of σ^2.

Consider the two linear regressions for the two subsets of the data modeled separately:

$$Y_i^{(1)} = \beta_0^{(1)} + \beta_1^{(1)} X_{i,1} + ... + \beta_p^{(1)} X_{i,p} + \varepsilon_i^{(1)}, \quad i = 1, 2, ..., k^*$$

$$Y_i^{(2)} = \beta_0^{(2)} + \beta_1^{(2)} X_{i,1} + ... + \beta_p^{(2)} X_{i,p} + \varepsilon_i^{(2)}, \quad i = k^* + 1, k^* + 2, ..., n$$

The Chow statistic is used to test the null hypothesis of no change point k^*

H_0: $\beta_0^{(1)} = \beta_0^{(2)}, \beta_1^{(1)} = \beta_1^{(2)}, ... \beta_p^{(1)} = \beta_p^{(2)}$ with the assumption of same variance $Var(\varepsilon^{(1)}) = Var(\varepsilon^{(2)})$. The Chow test statistic is computed using three residual sums of squares errors as:

$$F_{chow} = \frac{\left(\sum_{i=1}^{n} \hat{\varepsilon}_i^2 - \sum_{i=1}^{k^*} (\hat{\varepsilon}_i^{(1)2}) - \sum_{i=k^*+1}^{n} (\hat{\varepsilon}_i^{(2)2}) \right) \Big/ p}{\left(\sum_{i=1}^{k^*} (\hat{\varepsilon}_i^{(1)2}) + \sum_{i=k^*+1}^{n} (\hat{\varepsilon}_i^{(2)2}) \right) \Big/ (n - 2p)}$$

The above methods detect a possible change point in the time series. Once the change point is detected through the test, the time series dataset is divided into two intervals. The intervals before and after the change point form two homogeneous groups that take

heterogeneous characteristics from each other. Recursively, multiple change-points can be obtained under the binary segmentation method with the null hypothesis H_0 described above such that θ changes in the parameters are assumed, where θ is a known integer. The alternative hypothesis can be expressed as:

$H_1^{\theta} : \mu_1 = ... = \mu_{k_1^*} \neq \mu_{k_1^*+1} = ... = \mu_{k_2^*} \neq \mu_{k_2^*+1} = ... = \mu_{k_\theta^*} \neq \mu_{k_\theta^*+1} = ... = \mu_n$ where the integers $1 < k_1^* < k_2^* < ... < k_\theta^* < n$ are series of change points. The binary segmentation method works as follows:

First, use the change-point detection test, if H_0 is rejected; then find $\hat{k}_1$ (the time where the maximum of L_k, or F_{chow} is reached).

Next, divide the random sample into two subsamples: $\left\{ X_i : 1 \le i \le \hat{k}_1 \right\}$ and $\left\{ X_i : \hat{k}_1 + 1 < i \le n \right\}$.

Third, recursively test subsamples for further changes until no subsamples contain further change points. If exactly θ changes are found, then one rejects H_0 and accepts H_1^{θ}.

As discussed previously, the methods stated above need to determine the number of change points beforehand. It is difficult to know how many change points exist before knowledge extraction in data mining. Therefore, the methods are not usable directly in data mining techniques. On the other hand, these algorithms are not suitable for processing time series data online.

Keogh [79, 80] undertakes the extensive review and empirical comparison of all proposed techniques, including Sliding Windows algorithm, Top-Down algorithm and Bottom-

up algorithm, on the online time series data mining. The sliding windows algorithm allows a segment to grow until it exceeds some error bound. The process repeats with the next data point not included in the newly approximated segment; the top-down algorithm partitions the time series recursively until some stopping criteria are met; and the bottom-up algorithm starts from the finest possible approximation and merges segments of time series until some stopping criteria are met. The sliding window algorithm can't look ahead and lacks the global view of its offline counterparts. Although the bottom-up and the top-down approaches produce better results, they are offline and require scanning of the entire data set, thus making it impractical or even unfeasible in a data mining context, where the data are in the order of terabytes or arrive in continuous streams. The authors proposed the Sliding Window and Bottom-up (SWAB) algorithm to capture the online nature of sliding windows and retain the superiority of Bottom-up algorithm. The authors claim that the SWAB algorithm scales linearly with the size of the dataset,requires only constant space, and produces high quality approximations of the data.

Chu [81] further summarized the time series segmentation using sliding window techniques to detect nonstatinarity based on parameter fluctuations and change point localization. Chu classified the sliding windows techniques into the following four identification schemes:

- A fixed reference window and sliding testing windows; e.g., the reference model is identified in a fixed window of size h, and a sequence of test models is identified from sliding testing windows of the same size
- Growing reference windows and sliding test windows, e.g., reference models are updated in such a way that the reference window is always adjacent to the test window of size h.

- A fixed (Global) reference window and growing test window, e.g., the reference model, is estimated from the whole sample, and the sequence of test models is identified from test windows of increasing size.
- Growing reference windows and shrinking test windows, e.g., a sample split point k is used to determine the location of the reference window (containing all the pre-k observations), and the test window (containing all the post-k observations) such that the size of reference and test windows always sums up to the size of the whole sample.

According to Chu, only the first two methods permit online segmentation, while the last two do not. Based on the analysis of the asymptotic properties of the first two algorithms (FSW and GSW) based on AR model. The author concluded the following findings [81]:

- When the time series exhibits autocorrelation, the choice of the window size is not crucial in the sense of controlling the false alarm rate. On the other hand, if the time series is strongly auto-correlated, the FSW seems to have better size performance than the GSW.
- Segmentation with a larger window size performs better than with smaller window size, provided that the reference window is not contaminated.
- FSW is more powerful and accurate than the GSW when the change point is located near the end of the first reference window
- Localization accuracy deteriorates as the change point moves toward the end of the sample.

Li and Yu [82] discussed an optimization method to estimate a piecewise polynomial function with unknown change-points. A piecewise function is expressed by the addition of some absolute terms on which a piecewise regression model is then formulated based on these terms to minimize the estimation of errors subjected to a number of change-points. By utilizing a modified Least Absolute Deviation linear programming method, the algorithm estimates the piecewise regression with automatic change-point detection without pre-specifying the positions and number of change-points. The authors also pointed out that the computation burden is high as more 0-1 variables are involved. Yu and Li [83] further explored the general fuzzy piecewise regression analysis with an automatic change-point detection. The fuzzy piecewise possibility and necessity regression models are employed when the function behaves differently in different parts of the range of crisp input variables. The proposed method can deal with outliers by automatically segmenting the data. The result obtained is the global optimal rather than the local optimal by means of employing the mixed integer programming.

Kumar and Wu [84] presented an integrated identification procedure for change-point detection based on fuzzy logic. The membership function of each datum corresponding to the cluster centers is computed and is used for performance index grouping. The authors also suggested tests of the change in level and the change in slope for testing a hypothesis about change points. According their research, the algorithm based on the fuzzy entropy works more effectively than other techniques such as MPAGE and CUSUM methods.

Raimondo and Tajvidi [85] proposed peaks over the threshold model for change-point detection using wavelets. This method detects discontinuities in an otherwise smooth curve or

even in the presence of noise by checking the absolute value of wavelet coefficients. The authors combined wavelet methods and extreme value theory to test the presence of an arbitrary number of discontinuities in an unknown function observed with noise. This research suggested that lower frequencies should be used to detect change-points in case of heavy-tailed perturbations.

Wu, Salzberg and Zhang [49] presented a piecewise linear representation of the time series data. A tiered online segmentation and pruning strategy was adopted in the algorithm. The piecewise linear representation uses pre-processed time series before mapping into raw time series, in which pruning on the line segments with criteria based on the raw data stream was performed. The segmentation algorithm uses a sliding window with varying sizes. This rule-based pruning was performed to determine more than localized optimization of segmentation in a given time series. In this research, one finding is that the linear representation algorithm exits potential delay in response to online data; the second finding is that the threshold decision requires some trial and error, which makes it difficult to handle parameter drifting time series.

Most of the previous research for the change-point detection focused on the finding of unknown change points for the past, not on forecasting the future or dealing with the time series in an off-line approach; most methods discussed before also require strict assumptions on statistical distribution of data or residuals. As more requirements and needs arise in order to analyze time series data in real time, it is more promising to develop an algorithm that can handle time series data online with fewer restrictions on data and data distribution. The structure change defined in the dissertation sounds natural, and depends on its formalization, especially the way it

formalizes an adaptive cognitive system. In the research, each structure change corresponds to an event defined in the dissertation. This temporal pattern is expressed by dynamic gray systems, and its characteristics are fully utilized during the temporal pattern discovery by using these dynamic gray systems. In the following chapter, a dynamic adaptive cognitive whitening system – adaptive gray system – is introduced and discussed in detail on how the system is whitened and cognized.

Similarity Matching

Similarity matching in time series data mining poses the problem of solving questions such as finding all time series in the database or in data sets that are similar to the query series. There have been many research efforts targeted to similarity searching and matching. Similarity matching and searching are very useful in their own right as tools for exploring time series data sets, and also provide an important subroutine in many knowledge discovery and data mining applications such as clustering [86], classification [87], and mining of association rules [88]. Many different similarity measures have been developed by researchers to identify the similarities among time series. In this section, in order to facilitate our discussion later, the Euclidean distance measure is used as an example. The summaries of recent researches on similarity matching are also included. Our event-driven approach for time series similarity measure that considers these internal structures of time series will be discussed in a later chapter.

Humans are adept at recogizing the similarities between time series by simply looking at their plots. Such knowledge must be encoded in the computer if we want to automate the detection of similarity patterns among time series. In general, given any pair of time series, their similarity is usually measured by their correlation or distance. If we treat a time series as high-

dimensional points, which in time series they are, the Euclidean distance appears to be a natural choice for distance between time series. The Euclidean distance is defined as:

Given two time series sequences $X = x_1, x_2, ..., x_n$ and $Y = y_1, y_2, ..., y_m$ with n=m, their Euclidean distance is defined as:

$$D(X,Y) \equiv \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

We define that the two sequences X and Y are in δ-match if *D(X,Y)* is less than or equal to δ. We define n-dimensional distance computation as the operation that computes the distance between two sequences of length n. Basically, there are essentially two ways this data might be organized [89]:

- Whole sequence matching: In whole sequence matching, all the time series that are assumed to be comparable are of the same length; the query time series *q* is of length *n* as well. The Euclidean distance between the query time series and any time series it is to be compared with can be computed in linear time. Given a query threshold δ, the answer to a whole sequence similarity query for *q* is all the time series in the data set whose Euclidean distances from *q* are less than the threshold δ.
- Subsequence Matching: Here the time series in the data set can have different lengths. The lengths of these candidate time series are usually larger than the length of the query time series. The answer to a subsequence query derives from any subsequence of any candidate time series whose distance with *q* is less than δ.

Shasha and Zhu [90] pointed out that the Euclidean distance measure is not adequate as a flexible similarity measure between/among time series because:

- Two time series can be very similar even though they have different base lines or amplitude scales.

- The Euclidean distance between two time series of different lengths is undefined even though the time series are similar to each other.
- Two time series could be very similar even though they are not perfectly synchronized. The Euclidean distance that sums up the difference between each pair of corresponding data points between two time series is too rigid and will amplify the difference(s) between time series.

Faloutsos, et. al., [91] summarized highly desirable properties for the similarity search as follows:

- It should be faster than sequential scanning
- The method should require a small space overhead
- It should be 'correct'
- The method should be able to handle queries of varying length

However, Euclidean distance is a good "gold standard" for comparing different approaches [89].

The most direct approach to solving the similarity matching in time series is to compute the Euclidean distance between the query time series and all the candidate time series, meaning that only those whose distance is less than δ are selected for the final result. This approach scales poorly because the computing time increases linearly with the size of the database. A time series of length *n* can be considered as a point in *n*-dimensional space. This suggests that time series could be indexed by Spatial Access Methods (SAMs) such as R-tree and its many variants

[92]. However, most SAMs begin to degrade rapidly as the dimensionality increases. A realistic query can easily contain 20 to 1,000 dimensions per series. In order to use the advantages of SAMs, it is necessary to perform dimension reduction.

There are two basic strategies to cope with high-dimensional problems. The first is simply to use a subset of relevant variables to construct the model. That is, in other words, to find a subset of p' variables where $p' << p$. The second approach is to transform the original p variables into a new set of p'' variables, where $p'' << p$.

Researchers have made many efforts to reduce the dimensionality of time series while increasing performance of similarity matching. The original work by Agrawal, et. al., [24] utilizes the Discrete Fourier Transform (DFT) to perform the dimensionality reduction on the data, and then uses spatial access methods to index the data in the transformed space to speed up whole sequence similarity searching process. Faloutsos, et. al., [91] extended the work of Agrawal further to perform subsequence similarity matching using a Discrete Fourier Transformation. The idea of Faloutsos, et.al., is to map each data sequence into a small set of multidimensional rectangles in feature space; then, these rectangles can be readily indexed using traditional spatial access methods, like R-tree. A sliding window over the data sequence was also used to extract features. Moon, et.al., [93] uses a generalized window to reduce false negatives (from Faloutso's method) due to a lack of point-filtering effect. The general match method divides data sequences into generalized sliding windows and the query sequence into generalized disjoint windows to reduce false dismissal. The authors also proposed a method to estimate the optimal value of the sliding factor that minimizes the number of page access.

Because of its efficiency, wavelet transform is also used for feature extraction functions. Chan and Wu [33] proposed to use Haar wavelet transform for time series indexing and showed that Euclidean distance is preserved in the Haar transformed domain while no false dismissal will occur. The research also showed that the Haar transform outperforms DFT. Also, the method accommodates vertical shift of time series. Gilbert, et. al., [94] presented techniques for computing small space representation of massive data streams using wavelet-based approximations that consist of specific linear projections of underlying data. By capturing various linear projections of the data and using them to provide pointwise and rangesum estimation of data stream, the method uses only a small amount of space and per-item time as well as providing accurate representation of data. Huhtala, et. al., [31] proposed using a wavelet transformation of a time series to produce a natural set of features for the sequence in order to mine similarities in aligned time series. The features generated by wavelet transformations describe properties of the time series both at various locations and at varying time granularities such that they are insensitive to changes in the vertical position, scaling, and overall trend of the time series. The authors also examined how the similarity between time series changes as a function of time or as a function of time granularity. For more details of DFT and Haar wavelet transform, please refer to the Chapter 2 discussion of frequency domain time series analysis.

Wu, et. al., [95] compared the feature vector extraction using Single Value Decomposition (SVD) and DFT. The results showed that the SVD overall outperforms DFT in cases of a given query for a large number of neighbors or those which address a large radius. SVD also provided the best linear least squares error to data ratios. Korn, et. al., [96] proposed

SVD with Deltas (SVDD) based on SVD algorithm that supports ad hoc queries in large time sequence datasets. The SVDD algorithm achieves excellent dimensionality reduction and requires only three passes over the dataset while preserving distances.

Piecewise Aggregate Approximation (PAA) [97, 98] method was also proposed in the time series representation with dimensionality reduction. The PAA method reduced the data from *n* dimensions into *N* dimensions by dividing the time series data into *N* equi-sized "frames". The mean value of the data falling within a frame is calculated, and a vector of these values becomes the data reduced representation. In general, the transformation produces a piecewise constant approximation of the original sequence; see figure 3.3 [97, 98]. Keogh, et. al. [99] further modified the method and proposed adaptive piecewise constant approximation (APAC). APCA approximates each time series by a set of constant value segments of varying lengths such that their individual reconstruction errors are minimal. The authors showed how APCA can be indexed using a multidimensional index structure and proposed two distance measures in the indexed space that exploit the high fidelity of APCA for fast searching: a lower bounding Euclidean distance approximation, and a non-lower bounding, but very tight Euclidean distance approximation.

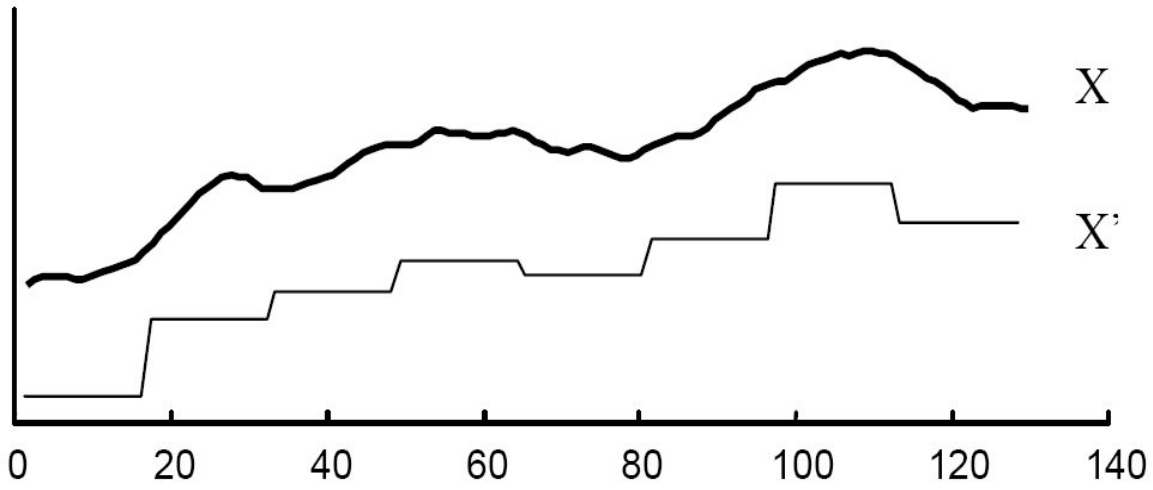

Figure 3.3. Piecewise Aggregate Approximation representation of time series

New distance measures , such as dynamic time warping,[90, 100] and longest common sequence[101] are also explored by researchers. Dynamic time warping algorithm allows stretching or squeezing time axis in the matching the similarity of time series. The method proposed by Das[101] also takes into account outliers, different scaling functions, and variable sampling rates while measuring the similarity between two time series. Wu, et. al., [102] proposed a comprehensive solution for analysis, clustering, and online prediction of respiratory motion using subsequence similarity matching. In the system, a motion signal is captured in real time as a data stream and is analyzed immediately by a piecewise linear representation generated by a finite state model to capture the relative importance of amplitude, frequency, and proximity in time.

There are extensive researches in the search for similarity in data streams. Some recent papers include [16, 103-111]. When dealing with data stream, traditional methods for time series are inefficient. Although the dimensionality reduction methods are very costly, they are applied

only once. The index methods are static because the index is constructed only once. Therefore, in order to utilize the advantages provided by the traditional methods, the dimensionality reduction methods must be applied each time a value arrives, and the index must be updated each time a value emerges. The research cited above adopts the incremental approach to reducing the computation time needed, or to incrementally maintain stream statistics over data streams.

Most research on streams still focuses on basic statistics and on how to define and evaluate continuous query. Also, the studies do not address these feature points that are important to human cognition and perception during the clustering or similarity matching. Current research also does not concern relative position of corresponding end points in the time series in the selected distance measures. Even though the best distance measure can depend on the dataset, task, and user, it is still possible that the "best" distance measure to use and change when the user interactively explores a database or data sets. In our research framework, an event-driven similarity matching algorithm is proposed. The algorithm is especially for online data stream processing and similarity matching, and the internal structures of time series are considered during the similarity marching. There is also no need to maintain incremental statistical maintenance over time. In the proposed algorithm, it also can handle both online data streams and off-line time series data. We introduced an cognitive dynamic system model that identify the structural changing points in the time series adaptively to capture internal structures of time series as well as to reduce the dimensionality of time series. Based on the structural changing points identified, a new distance measure, which addresses relative position of the corresponding data points, is introduced and used for time similarity matching and clustering. The detailed

algorithms are introduced in Chapter 5 after the introduction of dynamic system gray model presented in Chapter 4.

CHAPTER 4

DYNAMIC SYSTEM AND GRAY MODEL

Basically, a dynamic system may have any precise form of representation or any definite mathematical form [8]. A dynamic system evolves though time according to given evolutionary rules. Most common dynamic systems are represented mathematically in terms of differential equations that provide the element of dynamics when we consider mining information to assess the structural change. The information reflected by the change of structure in the time series is generated by the operation of the dynamic system that generates that time series. A dynamic adaptive cognitive system is needed to act as an intelligent cognitive process designed to identify these changes in order to mine this structural change information. On the basis of the cognitive process, a diagnostic set of statistics is defined to detect structure change in the system proposed. In the following sections, a general static gray system that is modified later to serve the need of the cognition process is introduced and discussed in detail, and later an adaptive rolling dynamic gray model method is discussed and used in the research as a cognition system.

Gray Systems and Gray Model Review

The gray system theory is based on the assumption that a system is uncertain and that the information regarding the system is insufficient to build or construct a model that will depict the evolution of the system exactly. The gray theory was initially presented by Deng[11, 12, 112] in 1982. The gray forecasting model adopts the essential part of the gray system theory and has been successfully used in finance, physical control, engineering and economics [113-117].

The gray model comes from the control theory that was established by a mathematician – Wiener [118]. Wiener proved that the self-adjustment mechanism between adjustment of machine and self-adjustment of biology was same. Control theory breaks the limitation between the animal and machine, and is used in many fields, such as engineering control theory, economical control theory, management control theory, etc.

Wiener first proposed the 'black box', which is the system defined by the fact that its internal structure, characteristics, and parameters are unknown (See figure 4.1.). However, when a system is being studied, the human observer(s) and the system (black box) couple together to form a machine with feedbacks. Based on different input, many shadow boxes will be obtained, as well as many possible outputs that might affect the methods of experiment. Information relating to the box will be obtained via study of input and output, and the mechanism of the 'black-box' can be somehow whitened. In contrast to the black box, in a white box, everything is known.

The gray system is an extension of the black box concept (see figure 4.2). According to the concept of the black box, a system, containing both known and unknown information is called a gray system. For example, the human body, economy, agriculture, etc., are gray systems. The aims of gray system theory are to provide theory, techniques, notions and ideas for resolving or analyzing latent and intricate systems [12]. The essential contents and topics of gray system theory encompass the following areas : gray relational space, gray generating space, gray forecasting, gray decision making, gray control, gray mathematics, and gray theory.

Knowledge acquiring procedure is based on converting information that is black into a form that is a human understandable and controllable format, which is white or gray. This is the procedure that converts black into gray or even into white. The process is called the whitening procedure. It is also a procedure through which the new knowledge, hidden pattern, or information is discovered. In natural science or social science, not all systems can be completely whitened during a study. Some mechanisms of the system can be understood precisely and accurately; for example, the relationships among electric resistance, electric current, and voltage can be expressed exactly by formulae in physics under certain conditions. However, in most cases of social, ecological, economical, and agricultural science, only partial information can be whitened; for example, in the economical supply and demand curve, researchers somehow understand the relationship between these factors, However, only partial information can be studied and converted into gray or white states because of the incompleteness of information, such as the psychological elements, behavior, etc., of the mass market. In general, the understandability in depicted in the white box is complete, while the gray box is partial , and the black box is nothing; i.e., white box >> gray box > black box.

From the aspect of predictability, the black box is unpredictable in general. the predictability of the gray box is greater than that of the black box, and the white box has a very good level of predictability.

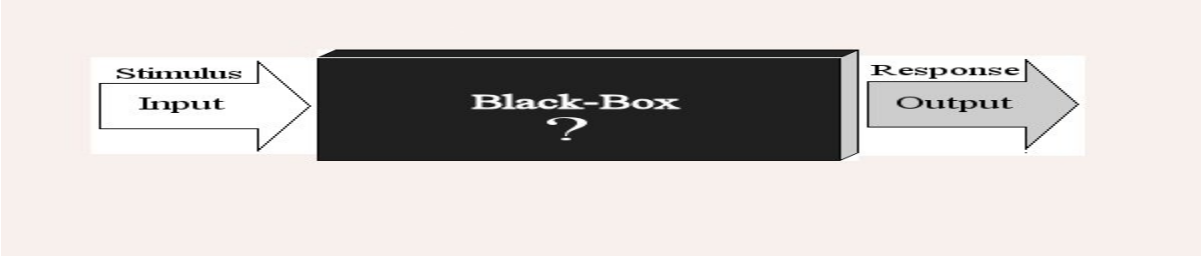

Figure 4.1 Black-box model

The gray model is the core of the gray system theory. It is the basic of gray forecasting, gray decision-making, and gray control. The gray model is the model that fits with differential equations based on a gray module (the module is the zone formed by the continuous curve or asymptotic curve of a series related to a time axis on a time-data plane). The establishment of a gray differential equation involves a procedure that uses a series to approximate the ordinary differential equation. A dynamic model with a group of differential equations called a gray differential model is then developed according to Deng [12].

Figure 4.2 Gray-box model

The gray model has been used successfully in geography, hydrology, management, engineering, agriculture, ecology, medicine, and social science because of its computational simplicity and effectiveness [12, 13, 119, 120]. The advantage of the gray predicting system is that only a few discrete data are sufficient to characterize an unknown system that is depicted by a first-order or higher-order differential equation. Thus, the gray predicting system is suitable for predicting a system in which only limited historical data are available and a quick and reliable resolution is needed for the decision-maker's reference. The gray theory avoids some inherent defects of conventional statistical methods such as regressive analysis or traditional time series analysis that require certain constraints on the data.

The gray prediction system uses a gray predictor to anticipate the system behavior and feed predicting information back to the decision-making mechanism to indicate an appropriate control action. This involves three basic procedures:

- Accumulated generation operation (AGO)
- Inverse accumulated generation operation (IAGO)
- Gray modeling

The gray prediction system uses accumulated generation to build differential equations. The gray model (GM) is often expressed as GM(n, h), where n is the order of the differential equation and h is the number of variables. The most commonly used gray model is GM(1,1) model, which is the first-order differential equation on one variable. Another commonly used approach is the GM(1, N) model, which is the first order differential equation on N variables. The following sections will discuss in detail how to build a gray model of the first order on one variable.

GM(1,1) Model

The algorithm [12] that builds a first order one-variable gray model is summarized as follows; this algorithm can predict the value of $x^{(0)}(n+i)$, where *i* is a positive number, and n is the number of data points used to build the gray model.

Step 1: The initial time series is

$$x^{(0)} = \left\{ x^{(0)}(1), x^{(0)}(2), ..., x^{(0)}(n) \right\} \qquad (1)$$

where $x^{(0)}(i)$ is the time series data at time *i*, for *i=1,2,…,n.*

Step 2: Based on the initial time series $x^{(0)}$, a new time series $x^{(1)}$ is generated by the first

accumulated generating operation (1-AGO).

where $x^{(1)} = \left\{ x^{(1)}(1), x^{(1)}(2), ..., x^{(1)}(n) \right\}$. The $x^{(1)}(k)$ is derived as follows:

$$x^{(1)}(k) = \sum_{i=1}^{k} x^{(0)}(i) \qquad (2)$$

Step 3: Assume that $x^{(1)}$ can by modeled by a first-order differential equation:

$$\frac{dx^{(1)}(t)}{dt} + ax^{(1)}(t) = b \qquad (3)$$

where *a* and *b* are constants, the parameter *a* is called the developing coefficient, and *b* is the gray input.

Step 4: Discretize to obtain the parameters *a* and *b* by using the least-squares error method to establish *GM(1,1)* prediction model, so that we have

$$A = \begin{bmatrix} a \\ b \end{bmatrix} = (B^T B)^{-1} B^T Y_n \qquad (4)$$

$$B = \begin{bmatrix} -0.5(x^{(1)}(1) + x^{(1)}(2)), & 1 \\ -0.5(x^{(1)}(2) + x^{(1)}(3)), & 1 \\ \cdots & \cdots \\ -0.5(x^{(1)}(n-1) + x^{(1)}(n)), & 1 \end{bmatrix} \qquad (5)$$

$$Y_n = (x^{(0)}(2), x^{(0)}(3), ..., x^{(0)}(n))^T \qquad (6)$$

Step 5: From (2), the initial condition for $x^{(1)}$ is $x^{(1)}(1) = x^{(0)}(1)$. Then the solution of ordinary differential equation (3) can be obtained

$$\hat{x}^{(1)}(k+1) = \frac{b}{a} + (x^{(0)}(1) - \frac{b}{a})e^{-ak} \qquad (7)$$

where $\hat{x}^{(1)}(k+1)$ is the predicted value of $x^{(1)}(k+1)$ at time *k+1*. Hence, the predicted data for an original time series can be obtained via inverse accumulated generation operation (IAGO) process as

$$\hat{x}^{(0)}(k+1) = x^{(1)}(k+1) - x^{(1)}(k)$$

$$= (x^{(0)}(1) - \frac{b}{a})(1-e^{a})e^{-ak} \qquad (8)$$

The predicting value of $x^{(0)}(k+i)$ can be calculated by

$$x^{(0)}(k+i) = (x^{(0)}(1) - \frac{b}{a})(1-e^{a})e^{-a(k+i+1)} \qquad (9)$$

In the algorithm, only small amounts of data are needed to estimate the two parameters in the model for prediction.

Fourier Residual Correction GM(1,1) Model

Although the gray model has outperformed some well-known modeling methods in our research [121], its modeling accuracy may not be very high. There have been many other methods proposed in the literature to increase the accuracy of the gray prediction by modeling the residual of the prediction. Most of them use multiple-order AGOs [12]. In order to illustrate the wave and cyclic behavior of financial time series, Fourier series to model the residual of gray model had been explored.[116]. Because of its good performance of modeling the residual of the gray model, we will adopt this technique to improve the *GM(1,1)*. The algorithm can be described as follows:

Let the residual time series R_r be defined as

$$R_r = \{R_r(2), R_r(3), ..., R_r(n)\}^T$$

where

$$R_r(k) = x(k) - \hat{x}(k), \text{ for } k = 2, 3, \ldots, n$$

Assume that residual time series is described by discrete Fourier series as:

$$R_r(k) = a_0 / 2 + \sum_{i=i}^{k_a} \left[a_i \cos(2\pi ik / T) + b_i \sin(2\pi ki / T) \right]$$

where $T = n - 1$, and $k_a = \lfloor (n-1)/a \rfloor - 1$. The parameters a_0, a_i and b_i for $i=1,2,\ldots,k_a$ can be estimated by the least-squares method. The coefficients a_0, a_i and b_i for $i=1, 2, \ldots k_a$ can be estimated as $C = (M^T M)^{-1} M^T E_r$, where $C = \left[a_0, a_1, b_1, a_2, b_2 \cdots a_{k_a}, b_{k_a} \right]^T$ and M

$$M = \begin{bmatrix} \frac{1}{2} & \cos\left(\frac{2\pi \times 2}{T}\right) & \sin\left(\frac{2\pi \times 2}{T}\right) & \cos\left(\frac{2\pi \times 2 \times 2}{T}\right) & \sin\left(\frac{2\pi \times 2 \times 2}{T}\right) & \ldots & \cos\left(\frac{2\pi \times 2 \times k_a}{T}\right) & \sin\left(\frac{2\pi \times 2 \times k_a}{T}\right) \\ \frac{1}{2} & \cos\left(\frac{2\pi \times 3}{T}\right) & \sin\left(\frac{2\pi \times 3}{T}\right) & \cos\left(\frac{2\pi \times 3 \times 2}{T}\right) & \sin\left(\frac{2\pi \times 3 \times 2}{T}\right) & \ldots & \cos\left(\frac{2\pi \times 3 \times k_a}{T}\right) & \sin\left(\frac{2\pi \times 3 \times k_a}{T}\right) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{2} & \cos\left(\frac{2\pi \times n}{T}\right) & \sin\left(\frac{2\pi \times n}{T}\right) & \cos\left(\frac{2\pi \times n \times 2}{T}\right) & \sin\left(\frac{2\pi \times n \times 2}{T}\right) & \ldots & \cos\left(\frac{2\pi \times n \times k_a}{T}\right) & \sin\left(\frac{2\pi \times n \times k_a}{T}\right) \end{bmatrix}$$

The new Fourier-corrected residual gray model can be rewritten as:

$$\hat{x}^{(0)}(k+1) = \hat{x}^{(0)}(k+1) + R_r(k+1) \text{ for } k=2,3,\ldots,n,..$$

with the initial condition being $\hat{x}(1) = x^{(0)}(1)$.

Modified GM(1,1) Model

What has been done so far to improve on the gray prediction model is to take the first term of $x^{(1)}$ as the initial condition of the first ordinary differential equation, thereby incorporating the error terms to the model, or, alternatively, to optimize the parameters via other methods. In the time series date, if data value is the reflection of information, then new information contained in terms other than the first term of $x^{(1)}$ is more significant; i.e., the recent

value $x^{(1)}(n)$ is of greater interest to the researchers. In our newly developed gray model, we chose the n^{th} term of $x^{(1)}$ as the starting condition to solve the gray differential equation. Therefore, the new information is more adequately incorporated into the gray model; as a result, the accuracy of prediction should be improved and residual errors should decrease correspondingly. Theorem 4.1 proves our assumption and gives the formula to for creating the gray model.

Theorem 4.1 Assuming that A, B and Y are the same as the algorithm in the gray model, the following statements are true:

(i) The discretized solution of the first-order differential equation

$$\frac{dx^{(1)}(t)}{dt} + ax^{(1)}(t) = b$$

at the initial value at $x^{(1)}(t)\Big|_{t=n} = x^{(1)}(n)$ is

$$x^{(1)}(k) = (x^{(1)}(n) - \frac{b}{a})e^{-a(k-n)} + \frac{b}{a}$$

(ii) The original raw data can be restored as

$$\hat{x}^{(0)}(k+1) = (x^{(1)}(n) - \frac{b}{a})(e^{-a} - 1)e^{-a(k-n)}$$

Proof:

(i) The general solution of the first-order ODE

$$\frac{dx^{(1)}(t)}{dt} + ax^{(1)}(t) = b \text{ is}$$

$x^{(1)}(t) = \frac{b}{a} + ce^{-at}$, where c is a constant.

Assume that the initial condition $x^{(1)}(t)\Big|_{t=n} = x^{(1)}(n)$ holds true, and substitute the value into:

$x^{(1)}(t) = \frac{b}{a} + ce^{-at}$, we have

$x^{(1)}(n) = \frac{b}{a} + ce^{-an}$

solve for *c*, we obtain

$c = (x^{(1)}(n) - \frac{b}{a})e^{an}$

so, the final solution of the ODE is

$x^{(1)}(t) = (x^{(1)}(n) - \frac{b}{a})e^{-a(t-n)} + \frac{b}{a}$

Using $t=k$ to discretize the equation solution, we have

$x^{(1)}(k) = (x^{(1)}(n) - \frac{b}{a})e^{-a(k-n)} + \frac{b}{a}$ (2)

(ii) It is trivial that the raw time series can be obtained from the inversed accumulating generation operation (IAGO), where

$\hat{x}^{(0)}(k+1) = x^{(1)}(k+1) - x^{(1)}(k)$, substituting the values from prove (*i*), we have:

$\hat{x}^{(0)}(k+1) = (x^{(1)}(n) - \frac{b}{a})(e^{-a} - 1)e^{-a(k-n)}$

In the equation above, for $k>n$, the calculation will produce a forecasting value based mostly upon on the newly generated information that has been reflected in the time series movement.

Adaptive Rolling Gray Model Method

As discussed in chapter three, a temporal pattern extends over a time period. The temporal pattern can be identified by its characteristics such as temporal order and time duration Therefore, a short-term memory is necessary to accommodate the inherent dynamics of temporal pattern. In general, a human short-term memory has a limited capacity (7±2)[52]. This pattern-based short-term memory decays over time corresponding to the decay theory of forgetting in human short-term memory. Theoretically, time information can be precisely recovered from the current value. However, due to rapid decay and noise, only a limited number of the most recent items can be reliably discerned from short-term memory. Therefore, a sliding window algorithm was adopted into the gray model building procedure in order to discover the temporal pattern and dynamics of the system. The gray model is able to have a dynamic allocated memory for the current structure that the dynamic system is on.

The sliding window algorithm is illustrated in figure 4.3. Let's assume that the time series data has N numbers of data points. Assume that each window contains W_1 data, and that each slide of window will move N_2 points to form the next window, whcih contains W_2 data. Let $W_1, W_2, ..., W_k$ be the series of windows formed by the sliding window technique. The determination of each window size and the N_2 will be discussed in a later chapter, because it closely relates to the goal of mining and the structure change. The memory size required depends upon which way the structure change is defined.

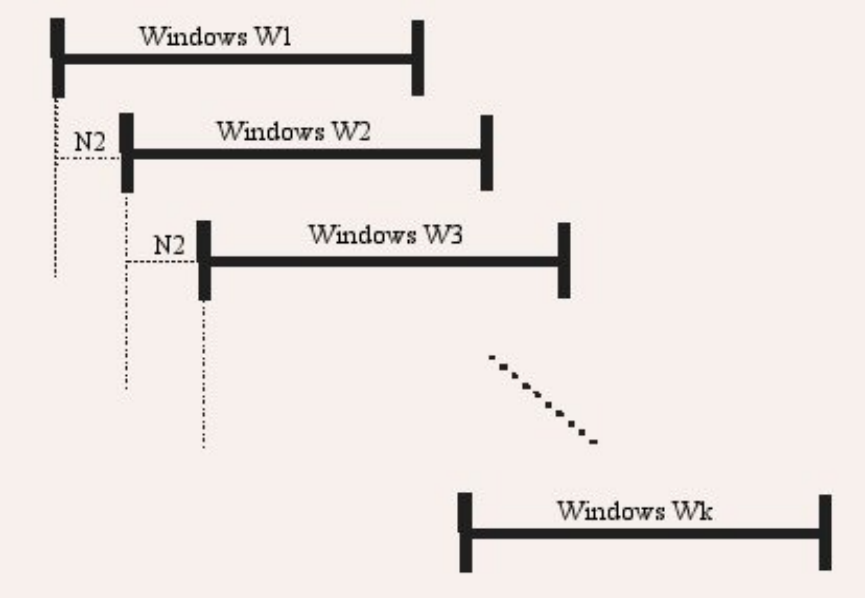

Figure 4.3. Sliding Window Algorithm Concept

For the purpose of simplification, let's assume that each window size is fixed with data size of N_1 and say that $N_1 = 4$ and $N_2 = 1$ to illustrate how a dynamic rolling window gray model is built (see figure 4.4). The dynamic sliding window gray model algorithm is explained as follows:

Assume that there are N original data:

$$X^{(0)} = \left\{ X^{(0)}(1), X^{(0)}(2), X^{(0)}(3), \ldots X^{(0)}(n) \right\}$$

After extrapolating $\hat{X}^{(0)}(n+1)$, slide the window the first time at N_2

Remove $X^{(0)}(1)$ and add the newly obtained data point $X^{(0)}(n+1)$, resulting in a new series: $X^{(0)} = \left\{ X^{(0)}(2), X^{(0)}(3), \ldots, X^{(0)}(n+1) \right\}$

which still maintains the series of size N; next, continuously extrapolate $\hat{X}^{(0)}(n+2)$; then slide the window a second time, and analogize from this information..

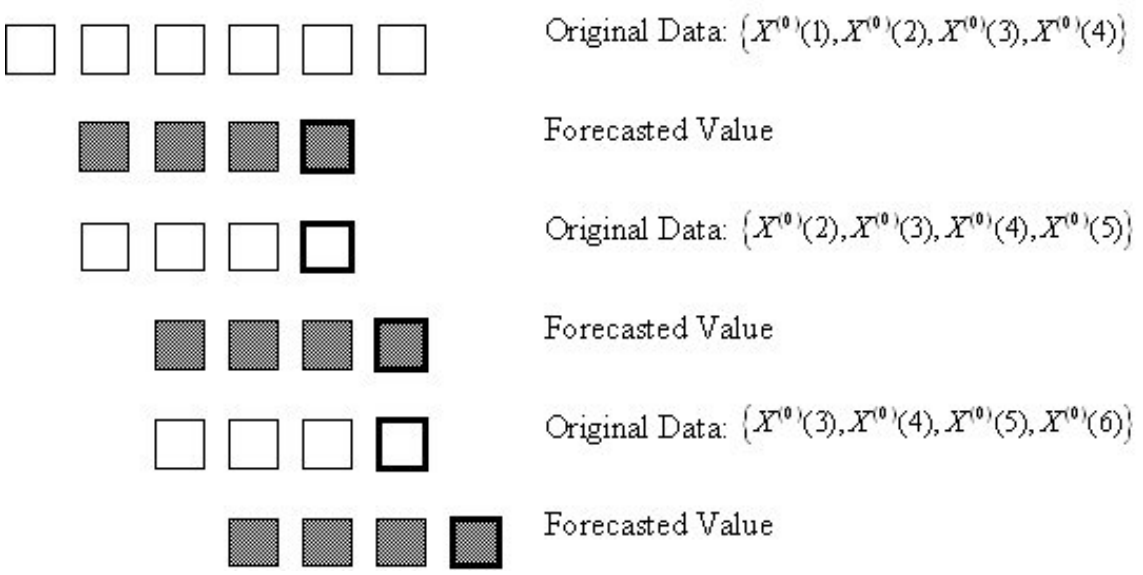

Figure 4.4. Example of sliding window algorithm with fixed window size of 4

Model Evaluation Methods

There are many ways to evaluate the forecasting performance of a model, ranging from directional measures to magnitude measures to distributional measures. The selection of evaluation criteria for forecasting techniques was conducted by Yokum and Armstrong [122]. Accuracy was their most important criterion, followed by the cost savings generated from improved decisions. In particular, execution issues such as ease of both interpretation and use were also highly rated.

Three measures of magnitude were incorporated into the research. The first measure is the mean square error (MSE), which measures the overall performance of a model. The calculation for MSE is

$$MSE = \frac{1}{n}\sum_{k=1}^{n}[x(k) - \hat{x}(k)]^2$$

where $\hat{x}(k)$ is the predicted value for time *k*. $x(k)$ is the actual value at time *k*, and *n* is the number of data used for prediction.

The second measure is the mean absolute error (MAE). It is a measure of the average error for all points and is computed as

$$MAE = \frac{1}{n}\sum_{k=1}^{n}\left|x(k) - \hat{x}(k)\right| \tag{14}$$

where the meaning of the $\hat{x}(k)$ and $x(k)$ is as MSE.

If we express the MAE into percentile format, we call it Mean Absolute Percentage Error (MAPE).

$$MAPE = \frac{1}{n}\sum_{k=1}^{n}\left|\frac{x(k) - \hat{x}(k)}{x(k)}\right| \tag{15}$$

The definition of $\hat{x}(k)$ and $x(k)$ is the same as that of MSE. MAPE is the more objective statistic indicator because the measure is in relative percentage and will not be affected by the unit of the forecasting series. The closer MAPE approaches zero, the better the forecasting results. According to Lewis [123], the performance of the model is categorized in Table 4.1:

Table 4.1. Performance measure by MAPE

MAPE (%)	**Performance**
<10	High precision forecast
10-20	Good forecast
20-50	Reasonable forecast
>50	Imprecise forecast

Although these three measures are fairly accurate for deriving the deviations of the predicted values from the actual values, they do not provide much information about the power of the models in predicting the turning points or direction of the predicted value. For many applications of time series analysis, the direction is as important as the magnitude; e.g., in the financial market, traders and analysts market direction and turning points besides the value of predictability. In these markets, money can be made simply by knowing the direction in which the time series moves. A correct directional prediction requires that

$$sign(\hat{x}(k+1)-x(k)) = sign(x(k+1)-x(k))$$

where $x(k)$ is the time series data point at time *k*, $x(k+1)$ is the time series data point at time *k+1*, and $\hat{x}(k+1)$ is the estimated time series data point at time *k+1*. Therefore, the direction accuracy (DA) is computed as

$$DA = \frac{1}{n}\sum_{i=1}^{n} a_i$$

where

$$a_i = \begin{cases} 1, if\ (x(k+1)-x(k))(\hat{x}(k+1)-x(k)) > 0 \\ 0, otherwise \end{cases}$$

This means that a_i takes the value 1 if the actual change and the predicted change have the same sign and 0 if they have opposite signs. If $(S_{i+1}-S_i)(\hat{S}_{i+1}-S_i) > 0$ for all *i*, then the value of DA will be 1, implying that the model predicts the accurate change on all occasions.

Theil's inequality coefficient [124] measures the forecasting power of a model relative to the random walk model. This relationship is calculated by dividing the RMSE of the model by the RMSE of the random walk model. Hence,

$$U = \frac{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(S_{i+1} - \hat{S}_{i+1})^2}}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(S_{i+1} - S_i)^2}}$$

The denominator is calculated from the actual change in the exchange rate between *i* and *i+1*. The numerator is derived from the difference between the actual change and the predicted change, which results in

$$(S_{i+1} - S_i) - (\hat{S}_{i+1} - S_i) = (S_{i+1} - \hat{S}_{i+1})$$

where $(S_{i+1} - S_i)$ is the actual change and $(\hat{S}_{i+1} - S_i)$ is the predicted change. Another version of Theil's inequality coefficient is based on relative changes. Hence it is calculated as:

$$U = \frac{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\frac{S_{i+1} - \hat{S}_{i+1}}{S_i})^2}}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\frac{S_{i+1} - S_i}{S_i})^2}}$$

The implications of the numerical values of U are as follow:

Value	**Implication**
U=0	The model produces perfect forecasts
0<U<1	The model provides less than perfect forecasts but out performs the random walk model
U=1	The model is as good as the random walk model

CHAPTER 5

FUNDAMENTAL TIME SERIES STRUCTURE MINING

This chapter details fundamentals of the time series structural change mining framework. After reviewing the goal of structural change mining, we discuss the framework. The chapter presents a framework based on the dynamic sliding window gray model that serves as a cognitive system to whiten the operating mechanism that generates a time series. The structural change is monitored based on the defined distance measure between the generating system and the cognitive system. When a sequence of structural change points are identified, a similarity-matching algorithm can be performed based on the reduced dimensionality, where the original time series is represented by a series of important structural change points.

Frame the TSDM Goal

In our research framework, the goal of the time series data mining is to identify a series of structural changes of operating mechanisms during the evolution of a time series in an online fashion; i.e., as a new data point arrives, a sequence of structural change points will be identified based on the cognitive model that dynamically identifies structural changes. In order to dynamically whiten the operating mechanism that generates the time series, a dynamic sliding window gray model acts as a cognitive function that has an adaptive short memory.

When a sequence of data can be broken into structures with observations that within each structure the data present homogeneous results, and between structures the data present

heterogeneous evidence.. This is especially true when the parameter shifts during the time series generating operations. In order to identify the changes in structure, several issues are involved: the choice of boundaries for changes in structure, how to cope with newly arrived data points, and what memory size is: i.e., how large the window should be during the dynamic cognitive model moving process.

Determine Temporal Structure Repulsion Factor

In order to identify the structural change via a cognitive model, distance measure is used to estimate the difference between the cognitive models as the new data point comes. Let's define a hypothetical cognitive model before change as a reference model, and a cognitive model after change as a test model. The questions underlying the structural change are: first, how to identify the reference and test models; second, how to measure the difference between the two models; third how to define a threshold of difference between these two models that can generate results that can be best explained by these models; forth is how we are going to measure and compare our structural changed discovered via different thresholds.

The first question is which model is going to be used to produce the time series, because the both the models before and after change are typically unknown. The second question is how we can identify the changes from the model we selected. In our research, the dynamic gray model is deployed to serve this purpose of modeling because of its ability to obtain the underlying mechanism of system operations and its power to whiten black information, which is unknown, into a certain level of gray. Thus, the information discovered can be partially explainable by gray numbers and models leading to white information. In order to discover the

changes, we introduced dual-model system – reference and test models to fully utilize the information whitened, accuracy, and precision provided by the dynamic gray model system.

We can reformulate the second question as follows: assume at current time *t*, where the newly identified data point is x_t, the closest recent structural change point to time *t* is at time *c*, and we need to decide whether newly arrived value x_t is a new structural change point; i.e., our cognitive system identifies a 'shock' or 'surprise', the internal structure of time series changes. If *t* is a structural change point, then the series $X_t = (x_c, x_{c+1}, ..., x_{t-1}, x_t)$ will be split into two series $X_{t1} = (x_c, x_{c+1}, ..., x_{t-1})$ and $X_{t2} = (x_t)$.

***Lemma 5.1*:** For all $x \in X$, where *X* is vector presentation of time series, there exists an asymptotic model *W*, such that the residual series

$$e = x - \hat{x}$$

follows the normal Gaussian Distribution.

Proof:

The proof is rather trivial: for example, the most commonly used multivariable linear regression model satisfied the requirements above.

Based on the *Lemma 5.1*, we have already found the gray model that asymptotically whitens the generating mechanism of time series both as the reference model and test model which residuals of the models follows the normal Gaussian distribution. The questions now is how we identify the structure change, i.e., split X_t into X_{t1} and X_{t2}. As proved, that residual follows Gaussian distribution with mean and variance such as:

$$X_i \sim N(\mu_j, \sigma_j^2),$$

The log likelihood of the sequence $X_{c+1},...,X_m$ for a generic μ, σ is

$$\sum_{i=c+1}^{m}\left[-\frac{(Xi-\mu)^2}{2\sigma^2}-\log\left(\sqrt{2\pi}\sigma\right)\right]$$

The maximized likelihood gives:

$M(c,m) = (m-c)\left[1+\log\left\{W(c,m)/(m-c)\right\}\right]$, further redefine

$M(c,m) = (m-c)\log\left[W(c,m)/(m-c)\right]$, because the omitted term sums to a constant over the entire sample. Let's now consider the residual series $e_t = (e_c, e_{c+1},..., e_{t-1}, e_t)$ and $e \square N(0,\sigma^2)$. We have the maximum likelihood *M(1,m)* that does not split, and *M(2,m)* that splits m into two series, we define the distance measure:

$$d = \frac{(m-2)\left[M(1,m)-M(2,m)\right]}{M(2,m)} > \delta,$$

then the *e* will be split from *m* into two series. δ is the threshold and $\delta > 0$. Statistically speaking, the *d* follows the *F*-distribution with degree of freedom of *m*, and *(m-1)*, the confidence level of 0.10, the value of *F* approximately lies from 1 to 10. Therefore threshold value δ could be range from 1 to 10. In our experiments both with synthetic and real data, the δ value within that range works well in most cases.

Search for Structural Change Points

Our approach to identify structural change points uses a sliding window algorithm that varies size adaptively. The sliding window can initially contain the minimum *m* data points. The sliding window that begins after the last identified structure change point and before ends right before the current data point is shown in figure 5.1. Suppose the current data point is $P_i(x_i, t_i)$,

the previous structure change point is $P_c(x_c, t_c)$, then the current point is the structure change point if and only if the distance measure between reference model and test model *d* satisfies:

$$d = \frac{(i-2)\left[M(1,i) - M(2,i)\right]}{M(2,i)} > \delta$$

Figure 5.1 shows an example of the sliding window after the latest structure change point is identified as P_3. Here, P_i=P_{13} ,which is the current newly determined data point. In the example, we assume a minimum initial sliding window of size 5. Therefore, after the first structure change point is found as P_3, the new data continues to arrive until the initial window is full enough to build a reference and test cognitive model; say at P_8, the sliding window is of size 6, but the distance measure reveals that there is no structure change, and data continues to be fed into our short term memory – the size of the sliding window until the current point P_{13}, where the distance measure indicates that the new data P_{13} caused the change of the structure. Therefore, the algorithm will mark P_{12} as the structure change point: i.e,. the end of the old structure and the beginning of a new structure in the next data point. Within the data points from P_3 to P_{12} , this structure is considered to be whitened by the cognitive gray model.

It can be noted that if a structure change point is not detected for a long time, the computations of the models become increasingly expensive. The solution is to consider a sliding window of a fixed size, say the last *w* data points of the data stream.

In our research, we have found that there exist false structural change points that are discovered by the algorithm discussed above. Therefore, an improved algorithm that verifies each newly structural change point detected is also proposed. This algorithm validates each newly discovered changing point by keeping the previously segments of structure and testing

that the distance measure between the reference and test model is still valid as the new data comes. If the next newly arrived data does satisfies the distance measure between the reference and test models for the new structure, then the newly discovered structural change point is validated. Otherwise, the newly discovered structural change point is a false structural change point *per se*, and the algorithm will discard this structural change and keep on reading the new data until the distance measures between the reference model and test model is satisfied. The algorithm that validates each structural change point is summarized as follows. By default, the starting point is a structural change point and is validated.

```
For each data point arrived
    If accumulated number of data points > min. window size
        Build reference and test model of these data
        Check whether the distance measure is satisfied
        If Yes
            Mark the data as not validated change point
            Push the newly arrived data point into the queue of change point
        Else
            Continue
    Else
        Continue
    Validate newly discovered structural change point if it is not validated
    If the distance measure is still satisfied, push it back to queue
    Else discard the recently discovered change point
End for
```

The last question is how to evaluate the structural change points discovered so far. How to evaluate the structural change detected helps to solve following problems. One is to test whether the results obtained are reasonable and sound; the second is how to compare the results obtained from different algorithms and parameters. Hawkins[68, 125] is interested in evaluating the change-points detection based on expert judgements. In data mining area, researchers either use human expert judgements or minimize the sum of square errors of model from each episode[48]. The sum of square errors is defined as

$E = \sum_{i=1}^{k} e_i = \sum_{i=1}^{k} \sum_{j=0}^{m_i} \left(x - f_i(x) \right)^2$, where k is the number of episodes discovered by the algorithm, m_i is the number of data points in the i^{th} episode, x is the true value of time series and $f_i(x)$ is the estimated value from the model.

Guralnik [48] proposed that the optimized event discovery is to minimize the square error sum. However, the square error sum has its own limitations. If we divide a time series x of size N into N/2 pieces (i.e., each piece contains only two data points) ece), and we model each piece according to a linear model, we could then obtain a minimized square error sum and a piecewise representation that would be useless and nonsensical. On the other hand, if the time series presents a horizontal line, then there should no event detected; the performance of the algorithm is totally dependent on how well the model can describe the data. In our research, we proposed a two-dimensional measure that compares the structural change points discovered. One dimension is the square error of sum,; the other dimension is the number of structures discovered. The ratio of square sum errors ranges from 0 to 1, where 0 means that the model generates no error and could describe the time series perfectly, and 1 means that the model can't describe the system very well. The number of structures discovered ranges from 1 to N/2, where N is the number of data points in the time series, and 1 means that only one structure has been discovered so far;

i.e.,. the time series is linear. N/2 is the maximum number of structures that could be discovered by any algorithm. The ratio of square error sum is defined as

$$ratio = E / \sum_{i=1}^{n} \left(x - \bar{x} \right)^2$$

where E is defined as above and $\sum_{i=1}^{n} \left(x - \bar{x} \right)^2$ is the variance of the time series. The ratio equals zero when $\sum_{i=1}^{n} \left(x - \bar{x} \right)^2 = 0$; in this special case, the time series is a horizontal line; i.e., the time series should have only one structure.

As shown in Figure 5.2, the plane created by the dimension ratio and the dimension of structures can be divided into four regions at any time. These four regions are marked as I, II, III, and IV in the figure. The results that fall in the region IV always outperform these that fall into these other regions such as I, II, and III. Region IV has a lesser ratio than that of region I and has a comparable number of structures; i.e., the results in region IV have acquired a lesser error factor in depicting the data. Region IV contains fewer structures than that of region III, yet has a comparable ratio to region III; therefore, the results in region IV have accomplished modeling the data with a smaller number of structures. Comparing the results in region IV to these at region II, region IV has both a smaller number of structures and a lower ration of square sum of errors.

Let's further discuss the two-dimension performance measure in the following situation. In case of $\sum_{i=1}^{n} \left(x - \bar{x} \right)^2 = 0$, the time series is a horizontal line. The best result is no further structures will be detected. In the two-dimension measure, the ratio is zero by definition, and the

performance of the results from different algorithms is decided only by the number of structures detected; in other words, the less structure found, the better the algorithm or parameter is. If the number of structures is *n/2*, where *n* is the number of data points, the lower the ratio obtained by the model, the better the performance of the algorithm or parameter. In the following chapter, this two-dimension measure is used to analyze and compare different algorithms and parameters used in our research.

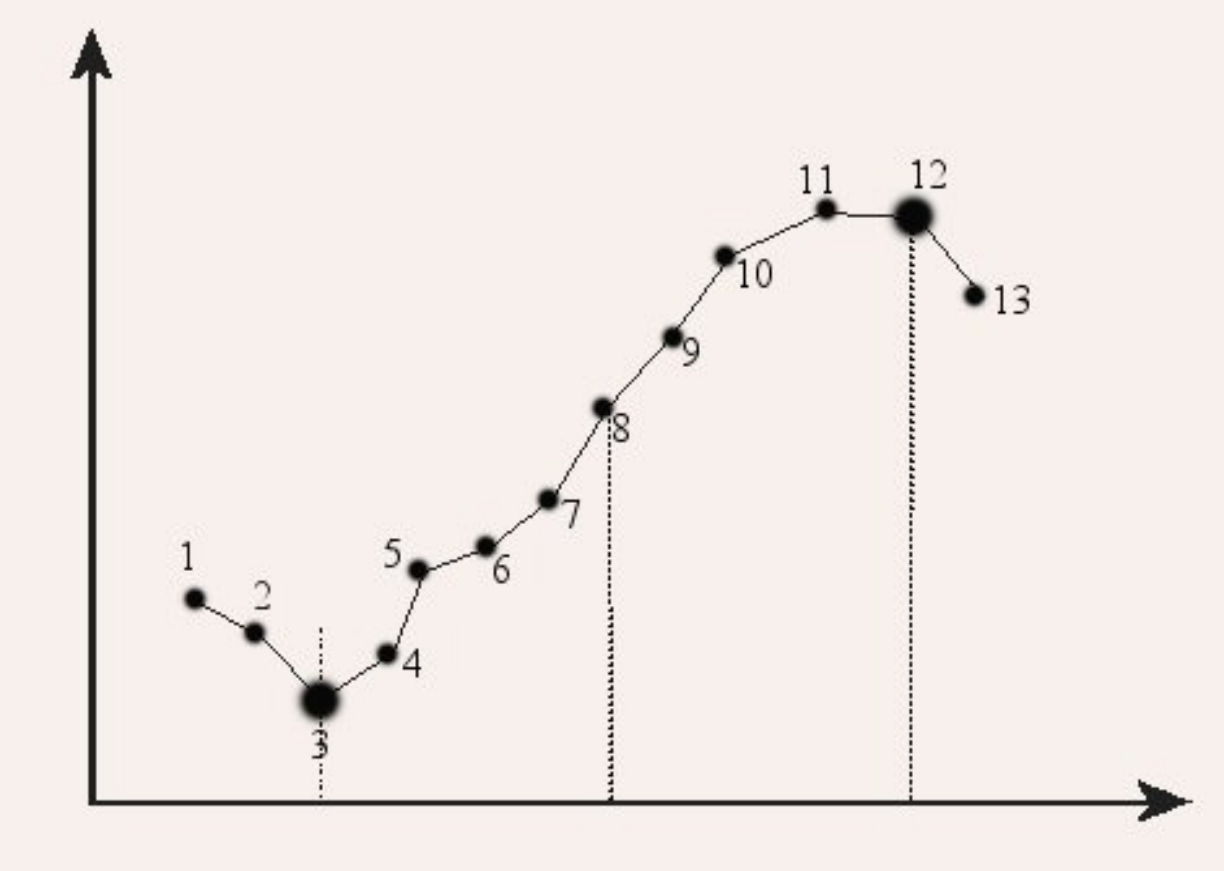

Figure 5.1 A sliding window that identifies the latest structural change point

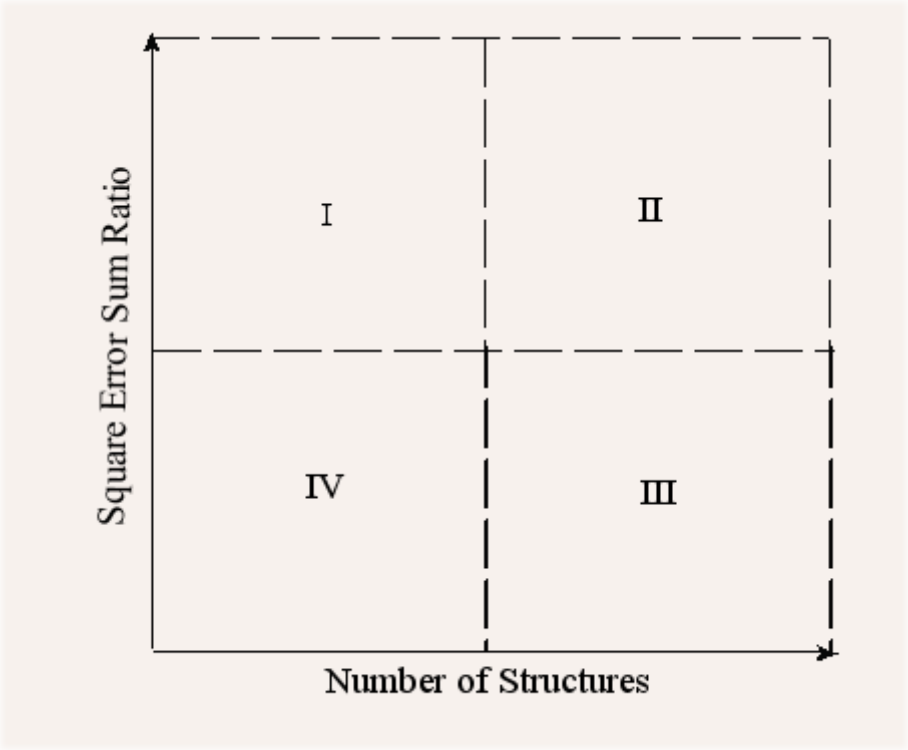

Figure 5.2 Two-dimensional evaluation matrix

Time Series Data Clustering

Once a series of historical events has been identified, we could perform our online event-driven similarity matching. In this section, we discuss our definition of similarity and the structure-driven online similarity matching is introduced thereafter.

There have been many research efforts for efficient similarity search. For example, the researches that use Euclidean distance and its variations based on the reduced dimension of time series [24, 33, 89, 91, 93]. Other distance measures include Dynamic Time Warping [90, 100], Longest Common Sequence Distance[101], and Wu's similarity measure based on permutation [49] and model-based, multi-layer, weighted subsequence similarity measure [102]. All methods above except Wu's similarity measure did not address the relative position of corresponding data

points. Unlike Wu's similarity measure based on the relative positions defined as permutation, our approach addresses the relative changes of each data point within the time series being compared among time series.

Our similarity measure is event-driven; i.e., the similarity matches only consider if we consider the data points that have been defined as the structure change points. The advantages of this approach are as follow:

- The time series data are presented dynamically by the cognitive whitening function
- The dimensionality reduction is based on the internal structure of a time series; i.e., where the meaning of the time series is given and the less volatile the time series is, the more reduction in dimensionality.
- It provides another means to reduce noise effect that might affect the similarity matches and also produces more accurate approximate similarity matches.

Our similarity distance function based on the relative positions of the series of events discovered from time series is defined as follows:

Define time series: $X = \{(x_1, t_1), (x_2, t_2), ..., (x_n, t_n)\}$ and $Y' = \{(y_1, t_1), (y_2, t_2), ..., (y_n, t_n)\}$ in which each has its own sequence of events discovered so far $E_x = \{(e_1, t_1), (e_2, t_2), ..., (e_m, t_m)\}$ and $E_y = \{(e'_1, t'_1), (e'_2, t'_2), ..., (e'_m, t'_m)\}$, X and Y are similar if and only if they satisfy two conditions:

- $d(E_x, E_y) < \gamma$
- $d(E_x, E_y) = \sqrt{\sum_{i=1}^{m-1}\left[(e_{i+1} - e_i) - (e'_{i+1} - e'_i)\right]^2 + \sum_{i=1}^{m-1}\left[(t_{i+1} - t_i) - (t'_{i+1} - t'_i)\right]^2}$

where $\gamma \geq 0$ and is a user-defined parameter

In order to use metric distance indexing method for a faster search, we need to prove that our defined distance measure is a metric function.

THEOREM 5.1. For event sequences E_x and E_y (with the same length), the distance $d(E_x, E_y)$ is a metric function.

Proof. To prove that $d(E_x, E_y)$ is metric, we need to prove it is non-negative, symmetric, reflexive, and that it satisfies the triangle inequality. Obviously, $d(E_x, E_y) \geq 0$ and $d(E_x, E_y) = d(E_y, E_x)$, also $d(E_x, E_x) = 0$, so $d(E_x, E_y)$ is non-negative, symmetric, and reflexive. Now we need to prove that $d(E_x, E_y)$ satisfies the triangle inequality, i.e., $d(E_x, E_y) \leq d(E_x, E_z) + d(E_z, E_y)$.

Given the sequences of events $E_x = \{(e_1, t_1), (e_2, t_2), ..., (e_m, t_m)\}$ and $E_y = \{(e'_1, t'_1), (e'_2, t'_2), ..., (e'_m, t'_m)\}$, we transform them into relative difference space of the sequence of events respectively as E_x to E and E_y into E' as:

$$E = \{(\Delta e_1, \Delta t_1), (\Delta e_2, \Delta t_2), ..., (\Delta e_{m-1}, \Delta t_{m-1})\} \text{ and}$$

$$E' = \{(\Delta e'_1, \Delta t'_1), (\Delta e'_2, \Delta t'_2), ..., (\Delta e'_{m-1}, \Delta t'_{m-1})\}$$

where $\Delta e_i = (e_{i+1} - e_i)$ and $\Delta e'_i = (e'_{i+1} - e'_i)$. Thus, it shown that the triangle inequality is satisfied based on the Pythagorean theorem and Euclidean space.

CHAPTER 6

SIMULATION OF SYNTHETIC TIME SERIES MODELS

This chapter presents two time series generating mechanisms that help to analyze the capabilities and limitations of the power of whitening and structural change detection algorithms. The first example characterizes the smooth sine function with a constant frequency, while the second example focuses on synthetic Mackey-Glass chaotic time series used to examine the algorithmic ability to whiten determined chaotic system change. After discussion of these generating mechanisms of time series, the four gray models are first evaluated based on MAE, MSE, MAPE, and Theil's inequality coefficient. An example is used to illustrate and compare out cluster and similarity distance measure, which does not need to maintain statistical variables over time. Finally, the structural change mining algorithms are then applied to these time series separately and the results are discussed.

Sinusoidal Time Series

In this sinusoidal time series evaluation, we examine a smooth sine function with constant frequency. This function is defined as

$$f(t) = a\sin(\varpi_1 t) + (1-a)\sin(\varpi_2 t)$$

where ϖ_1 and ϖ_2 are the frequencies of generated time series, and a is a constant. The observed time series that is scaled up to 100 generated by the above function where $\varpi_1 = 2\pi/250$, $\varpi_2 = 2\pi/25$ $a = 0.80$ and N=100 is illustrated in figure 6.1. As stated in Chapter 5, the final goal of structural changes mining is to identify these structural changes, such

as a dynamic gray model that cannot be handled by the cognitive system, and hope that these change in structures match with these visually important turning points that indicate the cyclic/inteal change of internal change occurring in the time series. The perceptually important points are indicated as 'O' in figure 6.1. In order to present the structure changes of the generating mechanism, a phase space diagram is shown in figure 6.2, and corresponding perceptually important points are circled with ''O'.

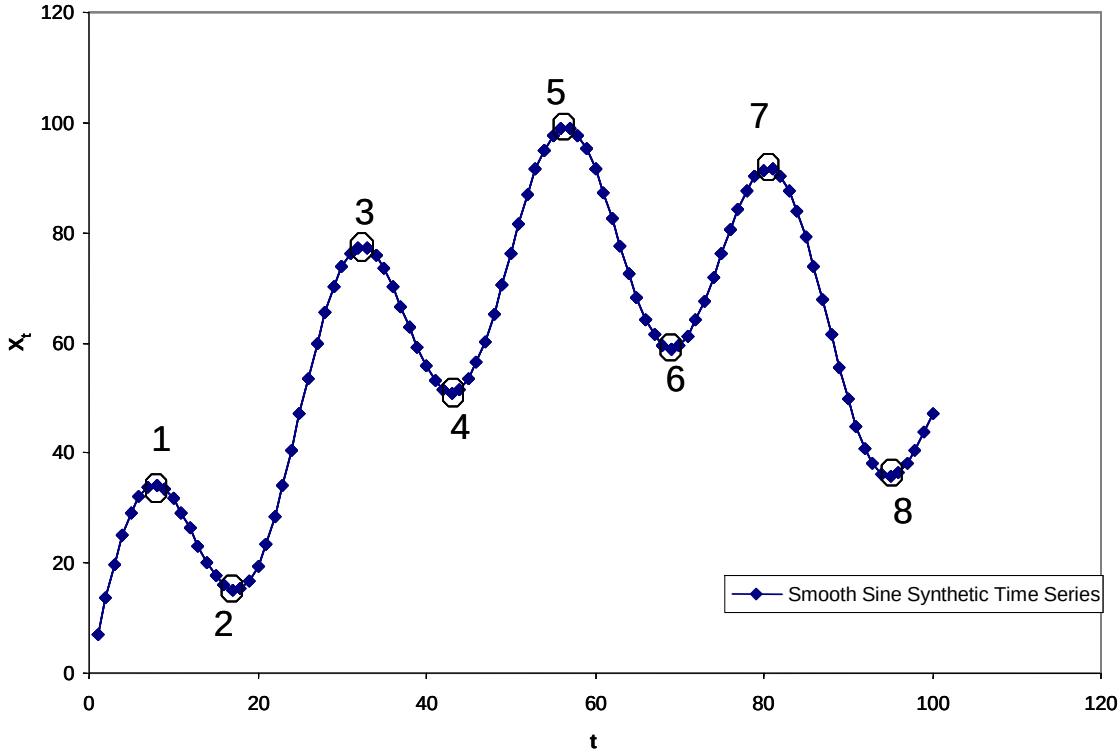

Figure 6.1. Synthetic Sinusoidal Time Series without noise

It is an easy task for the human brain to identify these perceptually important points in figure 6.1 that are marked with 'O' and numbered from 1 to 8. It can be seen that in the phase space of figure 6.2, these turning points are located at the narrow ends of the ovals. We recall that the phase space of sine curve on x_t and x_{t+1} is an oval.

The results of structure change-mining algorithm without validation are shown in figure 6.3. (and obtained by using our algorithm with validation) are shown in figure 6.4. Because of the constant frequency property of this synthetic time series, we use this time series to measure the accuracy level of structure discovery. As indicated in figure 6.3, the algorithm without validation contains two extra structural change points and misses two real structural change point as compared to our perceptual justifications, where these changes are marked one through eight in figure 6.1. Figure 6.4 shows that the algorithm with validation obtains only one extra structural change point and misses one change point. In order to compare further, we use a four cell contingency table for each category to evaluate our results, summarized in table 6.1.

Table 6.1 contingency table for structures discovered

	Data that belong to structure	Data that do not belong to structure
Data assigned to structure	TP	FP
Data not assigned to structure	FN	TN

The total number of data points in testing is N = TP + FP + FN + TN, where TP is the number of data points correctly assigned as structure changes, while FP represents the number of data points incorrectly assigned to these changes. FN is the number of data points incorrectly rejected from valid structure changes, and TN is the number of data points correctly rejected from these changes.

Our evaluation measures are summarized in Table 6.2. For each class, we count false positives (FP), true positives (TP), false negatives (FN), and true negatives (TN). The precision

is the proportion of correct positive class assignments; recall is the proportion of class members which are correctly labeled; fallout is the proportion of non-members which are incorrectly labeled, and the error rate assesses the proportion of errors.

Table 6.2 Evaluation measure for structure discovery

Recall = TP / (TP + FN)
Precision = TP / (TP + FP)
Fallout = FP / (FP + TN)
Error rate = (FP + FN)/(TP + TN + FP + FN)

Table 6.3 lists the results from the above contingency table of the results obtained from general algorithm and the algorithm with validation. Table 6.3 reveals that the algorithm with validation improves the recall, precision, fallout, and error rate compared to algorithm without validation.

Table 6.3 contingency table for algorithm with/without validation

	Recall	Precision	Fallout	Error rate
Algorithm without validation	75%	75%	2.1%	4%
Algorithm with validation	87.5%	87.5%	1.1%	2%

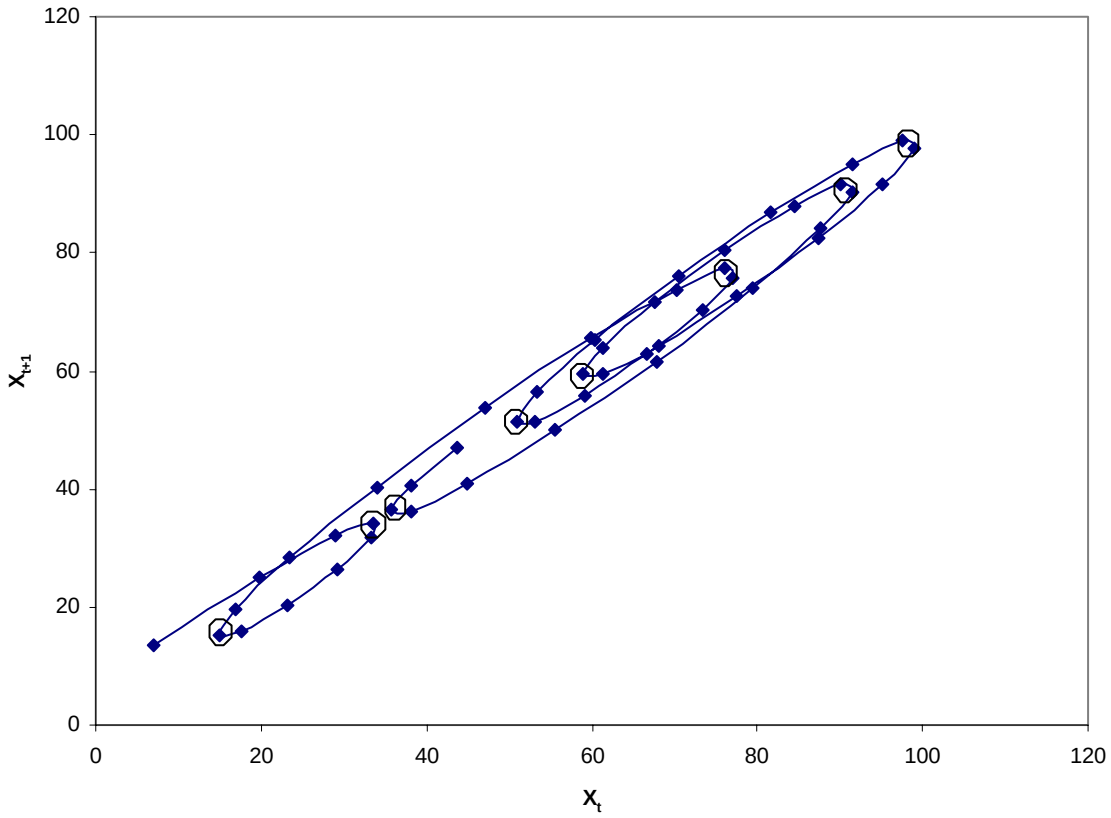

Figure 6.2. Phase space diagram of Sinusoidal Time Series with x_t and x_{t+1}

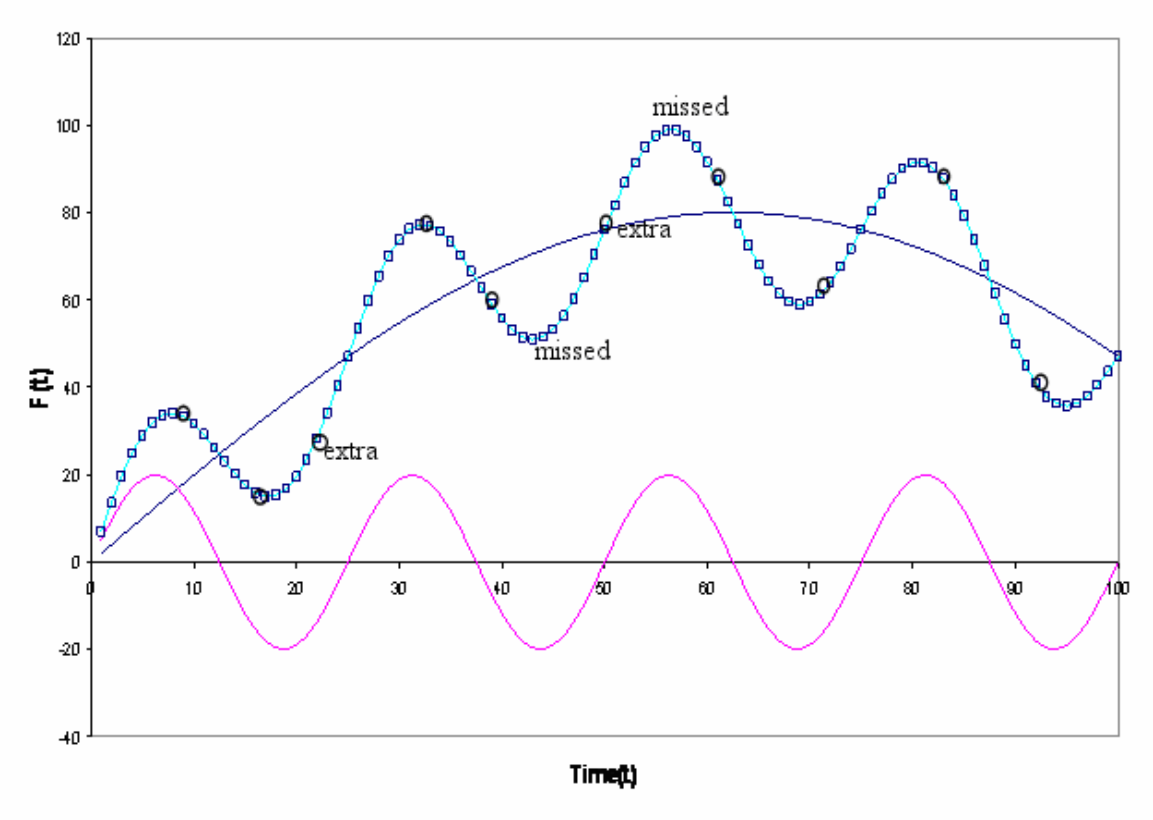

Figure 6.3. Structural change points identified by algorithm without validation

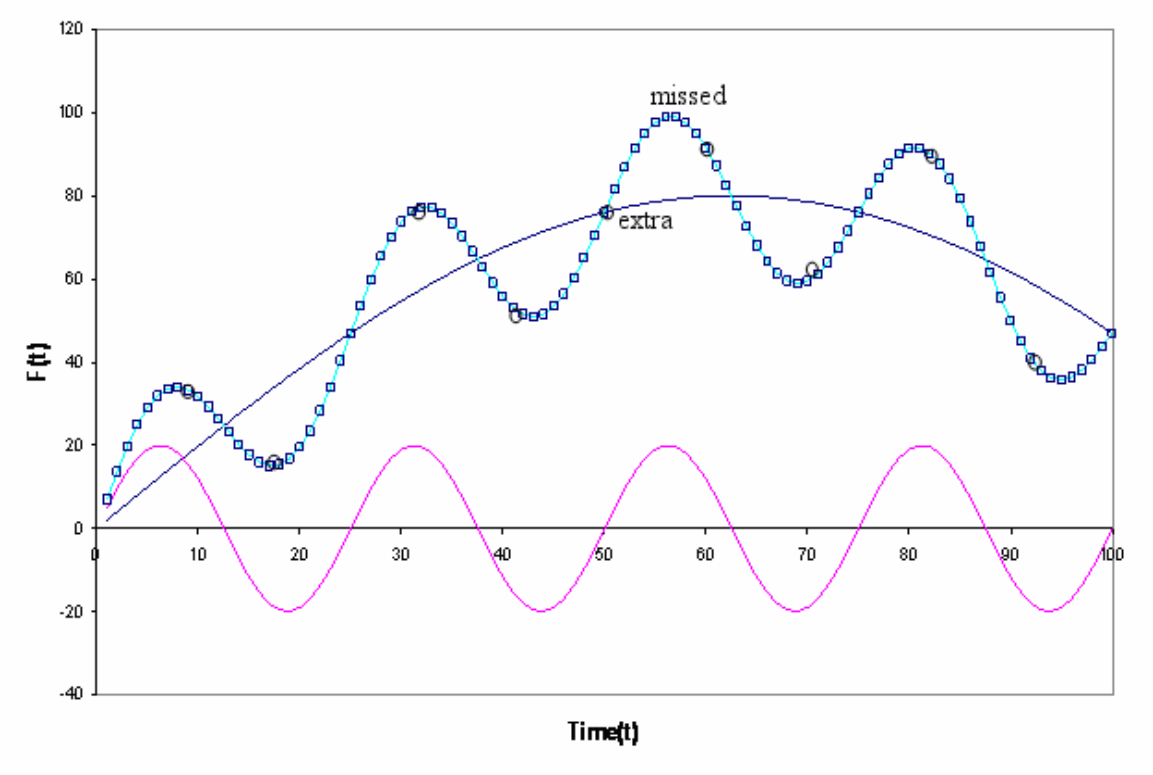

Figure 6.4. Structural changes with enhanced validation algorithm

Figure 6.5 presents the comparison between algorithm with and without validation based on the dimension of number of structures discovered and the error ration defined. It can be concluded that the algorithm with validation outperforms the algorithm without validation, because the error ration is close between these two; however, then the number of structures used to express the time series used by algorithm with validation is less than that of the algorithm without validation.

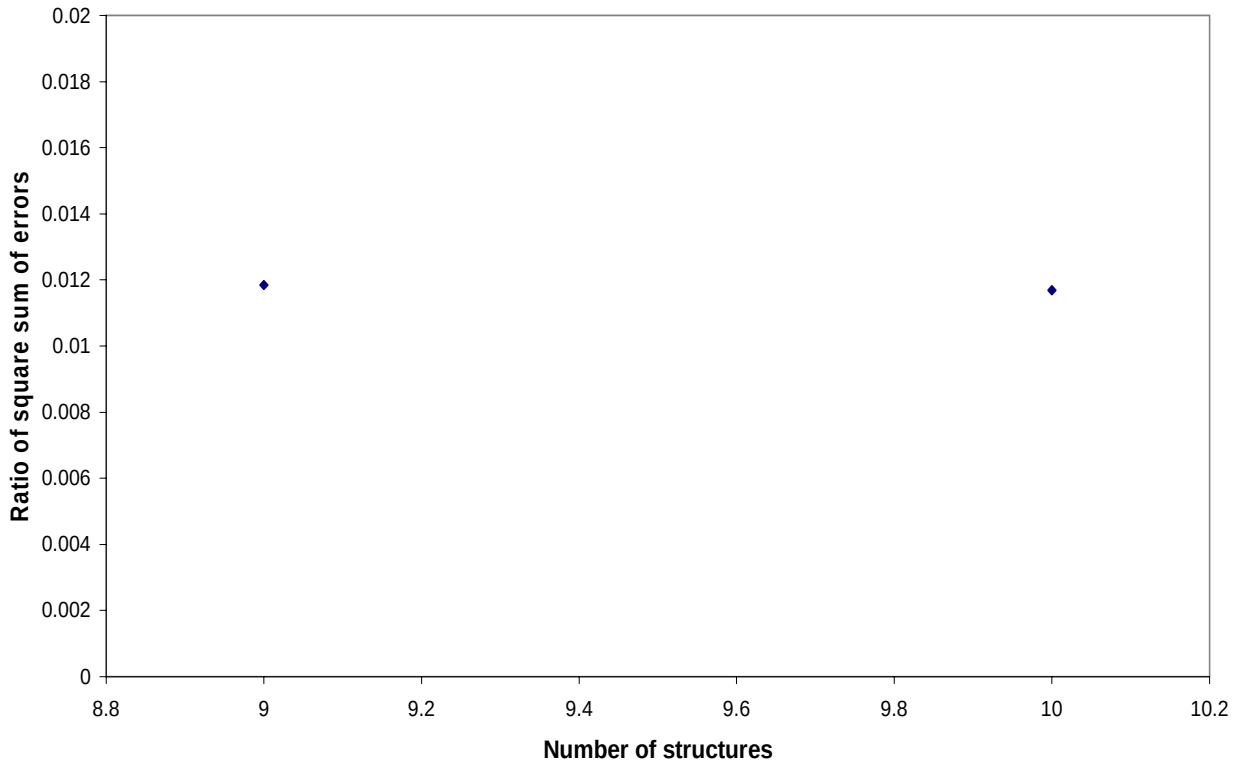

Figure 6.5 Two dimensional measures of algorithm with/without validation

Sinusoidal Time Series with Noise

Observed data are frequently contaminated by a noise, which is defined as '*irrelevant or meaningless data or output occurring along with desired information*'[126]. The presence of noise can substantially affect invariant system parameters as a dimension or entropy. Kostelich, et. al., [127] showed that even 2% of noise can make a dimension calculation misleading. Statistically, the term "white noise" is often commonly applied to a noise signal in the spatial domain that has zero autocorrelation with itself over the relevant space dimension. Gaussian white noise is a good approximation of many real-world situations and generates mathematically tractable models [128]. It is defined as random uncorrelated observations with mean zero, and

some standard deviations, which are added onto the real measurements. In order to study the noise impact on our algorithm, the additive white Gaussian noise (AWGN) is introduced into the Sinusoidal Time Series as:

$$f(t) = a\sin(\varpi_1 t) + (1-a)\sin(\varpi_2 t) + \varepsilon,$$

where ε is the white noise that follows Gaussian distribution with zero mean and unit variance, and ϖ_1 and ϖ_2 are the frequency of generated time series, and a is a constant. The observed time series that is scaled up to 100 generated by above function where $\varpi_1 = 2\pi/250$, $\varpi_2 = 2\pi/25$ $a = 0.80$ and *N=100* is illustrated in figure 6.6. The structure change points are marked as 'O' in the figure.

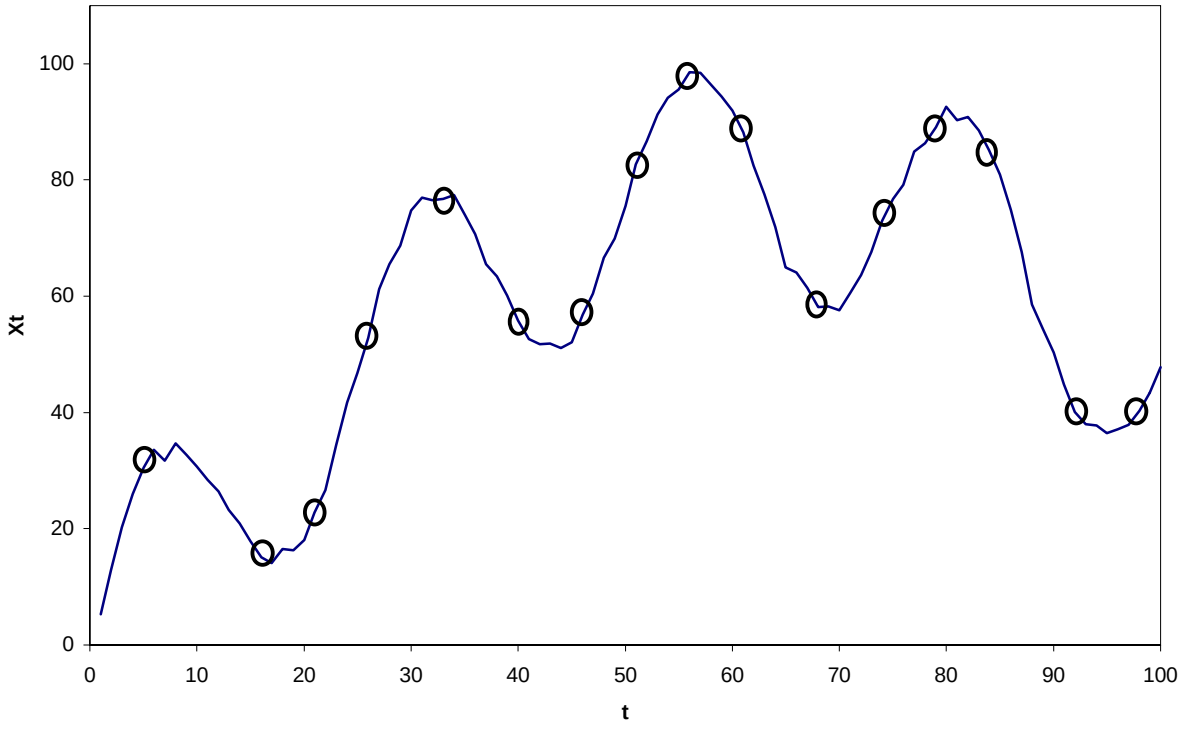

Figure 6.6. Structure change points identified with presence of noise

With the presence of noise in the generation function, our algorithm performs well at identifying these structure changes that are perceptually important to human comprehension. It can be concluded that without the noise, the dynamic incremental version of the algorithm could mine inherit structure changes correctly, and the result obtained is very close to the visual judgment of a human being. As noise presents in the synthetic signal, the algorithm can dynamically identify the flat regions that are significant, such as region 83 - 92. More structural regions can be obtained/accessed because noise has changed the internal time series structure.

Chaotic Time Series

The most interesting time series in this dissertation may be classified as chaotic. In this section, we will discuss its definition.

"Chaos comprised a class of signals intermediate between regular sinusoidal or quasiperiodic motions and unpredictable, truly stochastic behavior [129]." The chaotic time series is also defined as one generated by a nonlinear, deterministic process (highly sensitive to initial conditions) that has a broadband frequency spectrum [129].

The language for describing chaotic time series can be traced to dynamical systems theory, which studies the trajectories described by flows (differential equations), maps (difference equations), and nonlinear dynamics, an interdisciplinary field that applies dynamical systems theory in numerous scientific fields.[9, 10]

Chaos occurs as a feature of orbits *x(t)* arising from nonlinear evolution rules, which are systems of differential equations:

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

with three or more degrees of freedom $x(t) = [x_1(t), x_2(t), x_3(t), ..., x_d(t)]$ or invertible discrete time maps

$$x(t+1) = F(x(t))$$

with two or more degrees of freedom. Degrees of freedom are systems characterized by ordinary differential equations, which determine the number of required first-order autonomous ordinary differential equations**.** In discrete time systems that are described by maps $x(t) \to F(x(t)) = x(t+1)$, the number of degrees of freedom is the same as the number of components in the state vector *x(t)*. The requirement for a minimum size of state space to realize chaos is geometric. For differential equations in the plane (*d=2*), it has been know for a long time that only fixed-point (time independent solutions) or limit cycles (periodic orbits) are possible. Chaos, as a property of orbits *x*(*t*), manifests itself as complex time traces with continuous, broadband Fourier spectra, nonperiodic motion, and exponentials sensitive to small changes in the orbit.

As a class of observed signals *x*(*t*), chaos lies logically between:

- the well studied domain of predictable, regular, or quasi-periodic signals which have been the mainstay of signal processors or decades, and
- the total irregular stochastic signals we call “noise” and which are completely unpredictable.

With conventional linear tools such as Fourier transforms, chaos looks like “noise”, but chaos has structure in an appropriate state or phase space. That structure means that there are numerous potential engineering applications of sources of chaotic time series which can take advantage of the structure to predict and control those sources. Chaos is irregular in time and slightly predictable. Chaos also has structure in phase space.

One of the most commonly used chaotic time series generation function, the Mackey-Glass chaotic time series, is defined by the following delay-differential equation [130].

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t)$$

where τ is an adjustable delay term.

This delay differential equation can be extremely chaotic and display a wide variety of behaviors because its value at any time may depend on its entire previous history. Farmer and Sidorowich [131] presented a forecasting technique for chaotic time series. They introduced nonlinear mapping using a local approximation after embedding a time series in a state space using delay coordinates. In order to evaluate the ability of the algorithm to identify structural change of chaotic time series that periodicity is not apparent or might not exist at all, a Mackey-Glass chaotic time series is used with $\tau = 30$ in the simulated evaluation.

The following experiments are conducted based on the Mackey-Glass chaotic time series: forecast precision measures test, initial point sensitivity test, and structural mining parameters sensitivity test. In the following sections, these three experiments are discussed in detail.

Forecast Precision Measure

Three statistical measures: MSE, MAE and MAPE are used to measure predictability of our four models GM11, GM11XN, MFGM11, and MFGM11XN. Directional accuracy (DA) is used to measure the directional predictability. Thiel's Inequality Coefficient is used to compare with random walk model. The figures 6.7 to figure 6.11 present the analyzed results based on the in-sample, out-sample, and parameters such as window size of the models.

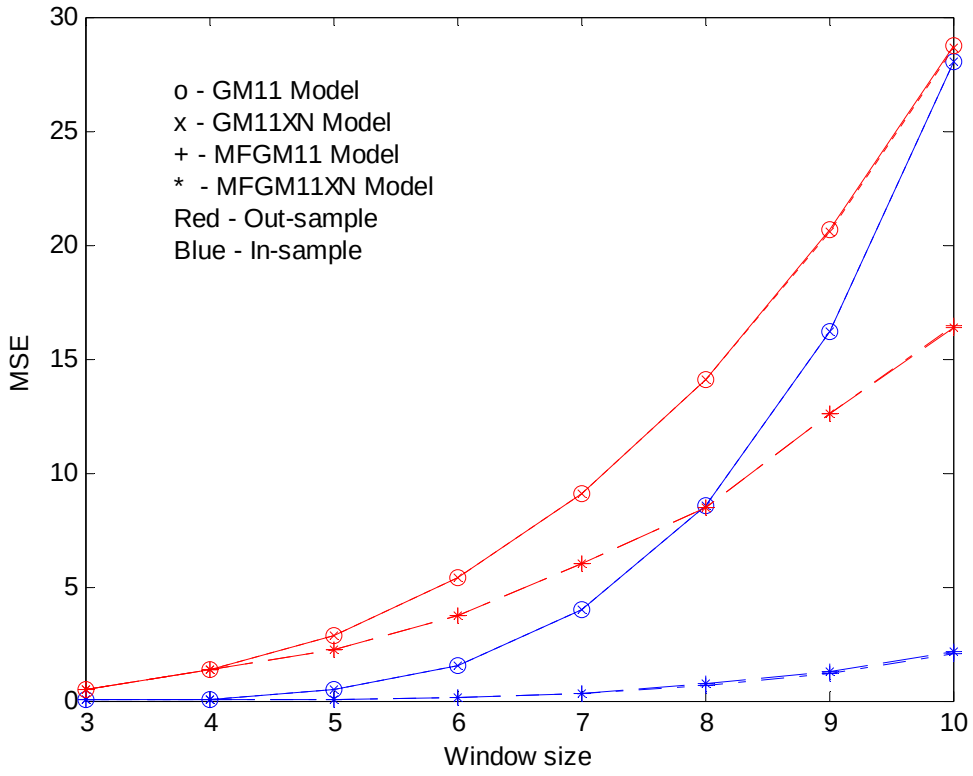

Figure 6.7. Mackey-Glass MSE vs. Window Size

Figure 6.7 shows that the MSE increases as window size used by models increases; i.e., the more historical data used to build the model, the higher the MSE will be. The result confirms the research [52] that the short-term memory is around 7±2. Fourier Modified Gray Models tend to have a smaller MSE than these of non-Fourier error corrections. Also, MSE from out-sample is higher than that of in-sample. The latest data values used as the initial condition to solve differential equation do not seem to improve the MSE. The bigger the window size used to build the model, the worse is the model based on MSE.

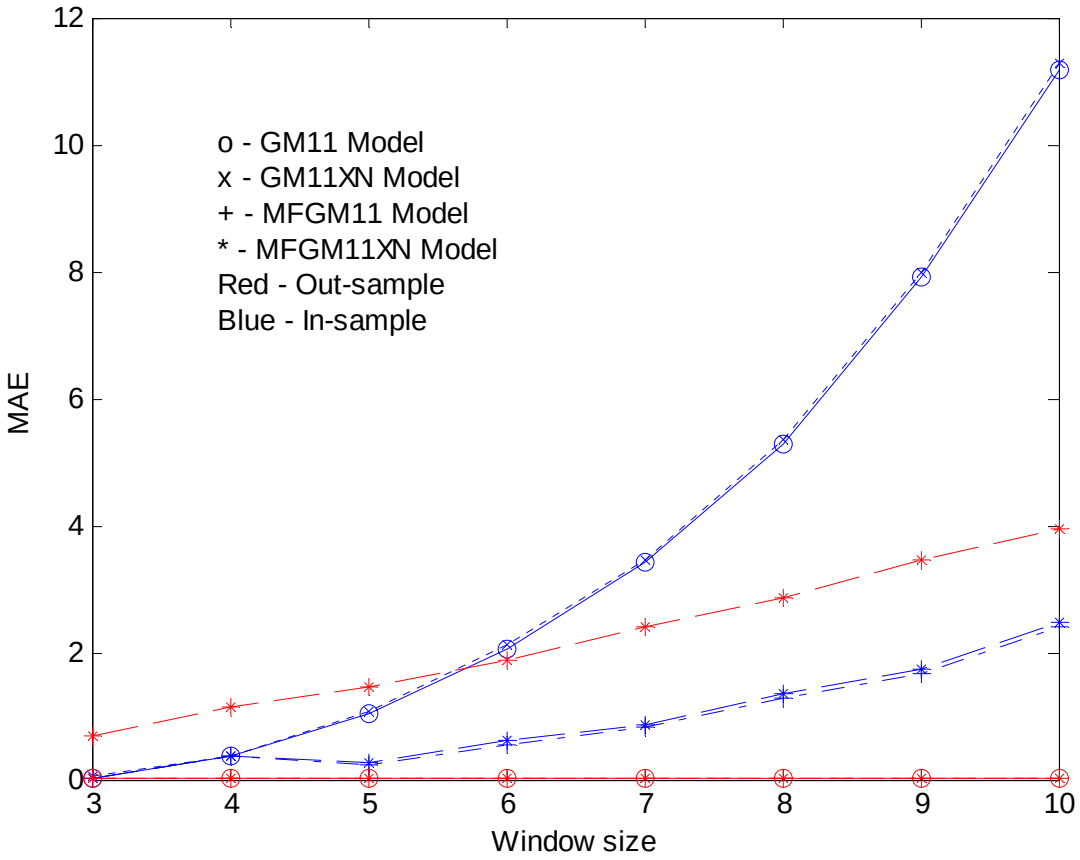

Figure 6.8 Mackey-Glass MAE vs. Window Size

Figure 6.8 shows the MAE changes with the change of window size used by the models. It is clear that our sample MAE is smaller than that of an in-sample. The Fourier- corrected error models decrease MAE compared to non-Fourier modified errors on in-samples. The out-sample MAEs are smaller than those of the in-sample.

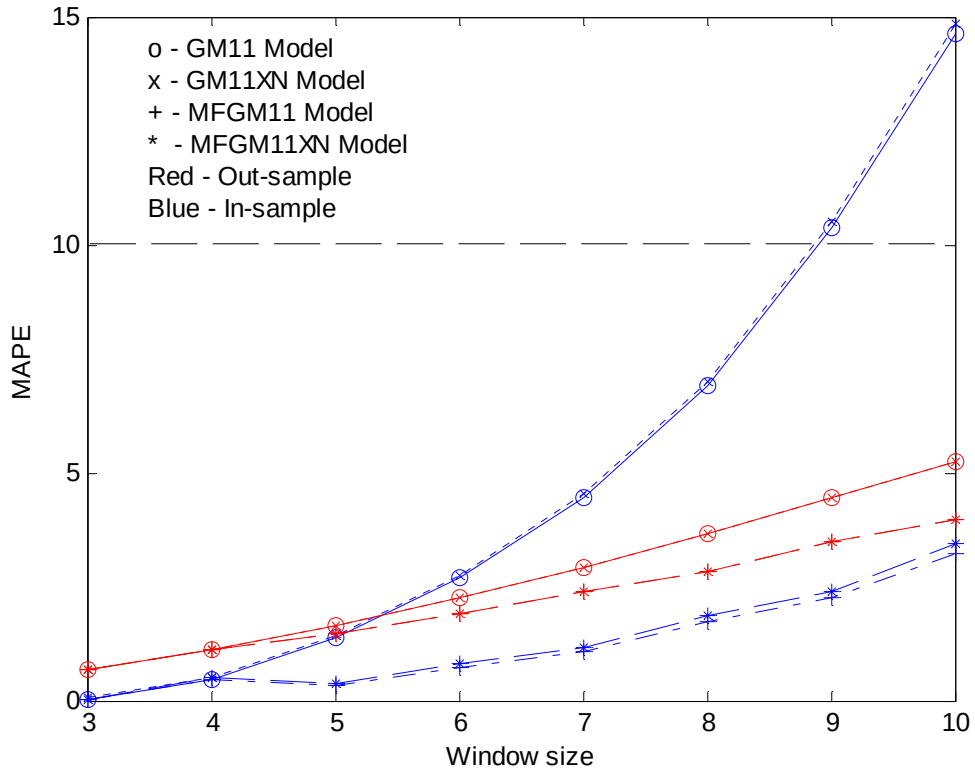

Figure 6.9 Mackey-Glass MAPE vs. Window Size

Figure 6.9 presents the change of MAPE against the windows size of the model. Models with MAPE values of less than ten can be classified as high-precision models. We can see that when the window size is less than nine, all these four models can be categorized as high-precision models. In contrast, models modified with Fourier transform have lower MAPE values than those of non-Fourier modified models in the in-sample comparison. However, the out-sample comparisons reveal that all four models achieve similar MAPE values. As window size increases, the MAPE value increases also.

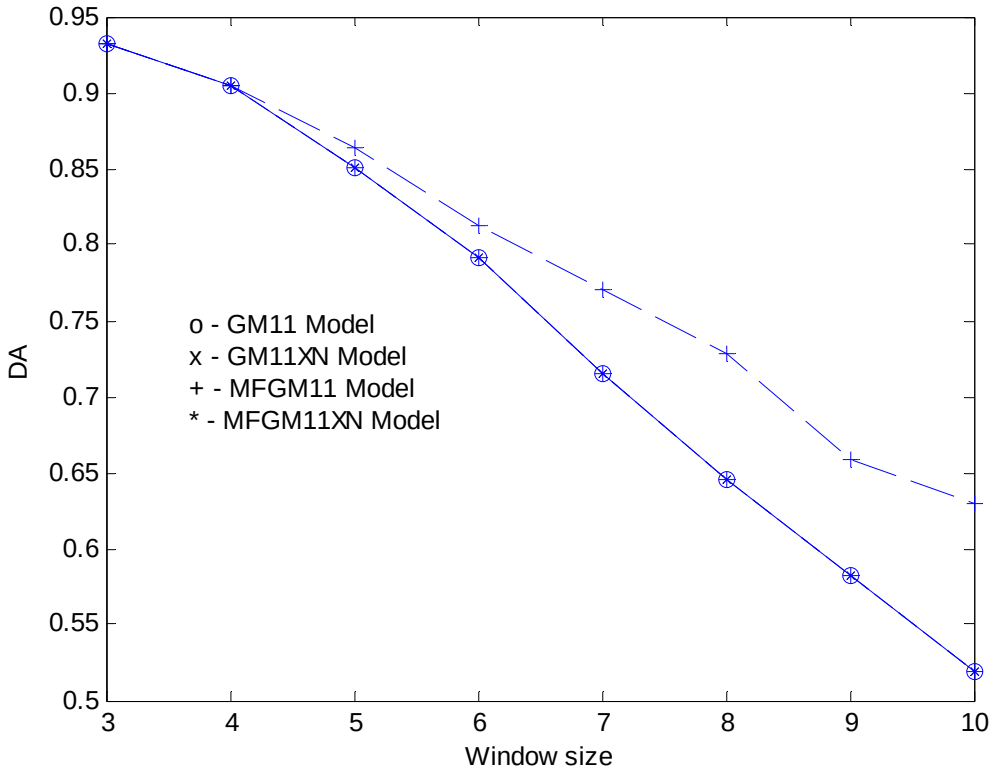

Figure 6.10 Mackey-Glass DA vs. Window Size

Figure 6.10 shows that the increasing of window size used by the models tends to decrease the model's ability to forecast directional moves. Fourier modified models do not change the DA significantly, and usage *Xn* as initial conditional value does not cause major improvement. It is interesting to see the Fourier-modified gray model produce better DA predictability as window size increases.

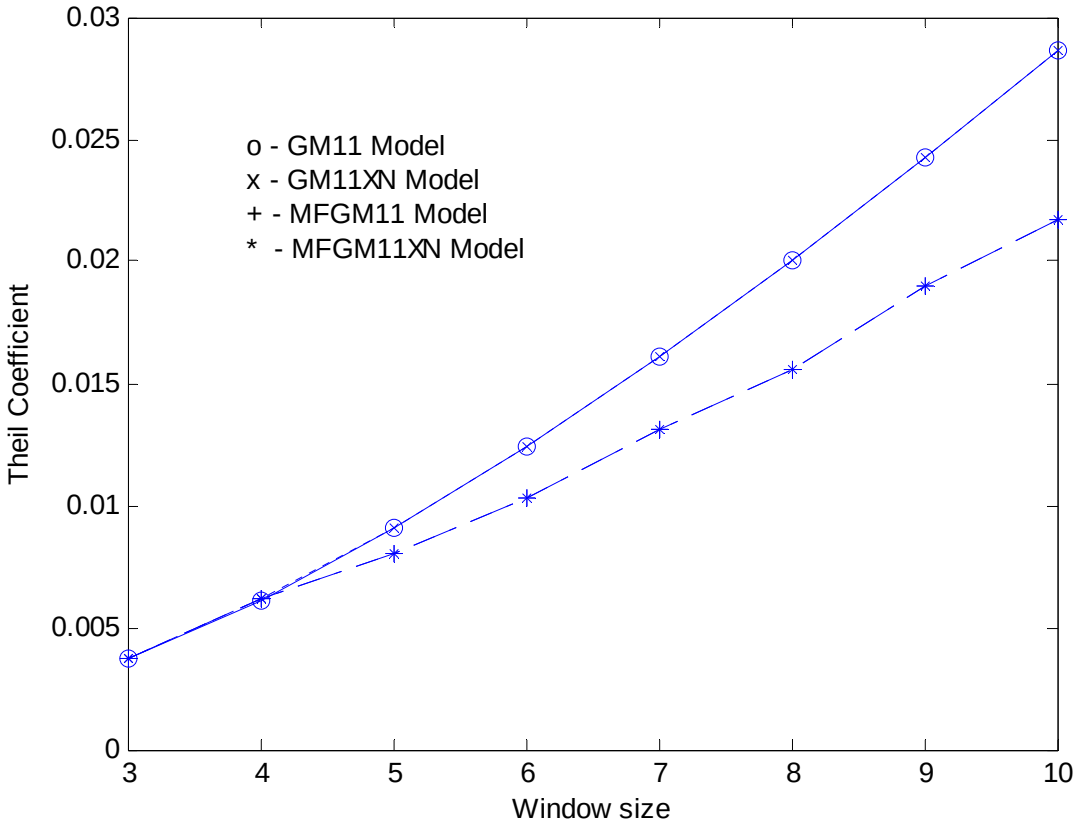

Figure 6.11 Mackey-Glass Theil's Inequality Coefficient vs. Window Size

Figure 6.11 illustrates the change of Theil's Inequality Coefficient with the change of window size. Clearly, the increasing of window size also increases the Theil's Inequality Coefficient. Fourier modification of models decreases the Theil's Inequality Coefficients, while initial condition change does not affect the Coefficient very much.

Overall, it can be concluded that a smaller window size is better suited to building models. Based on our experiments, a window size less than seven is suggested in order to achieve better values of MSE, MAE, MAPE, DA and Theil's Inequality Coefficient. All four models perform very well, and high-precision models based on these measures outperform the random walk model.

Algorithm Initial Data Point Sensitivity

The results with threshold value of 1.0 and 2.0 are shown separately in figures 6.12 and 6.13. We can conclude that the dynamic incremental version of the algorithm with gray system could mine the inherited structural change correctly, and the result obtained would be very close to the visual judgment of a trained human being. With the critical value of 1.0, all the peaks and valleys in the Mackey-Glass chaotic time series data stream are identified as structure changing points. The two flat structures, region 10 to 20 and region 103 to 112, are also detected correctly. The two dramatic drops of the time series, regions 30 to 61 and 131 to 164, are also further subdivided into two finer structures based on the algorithm of critical value 1.0. When the critical value increased from 1.0 to 2.0, the structure changes detected by the algorithm decreased from 28 to 22 because higher critical values give the algorithm less restriction on structure detection. These two flat regions are combined with structures nearby. There are five peaks, marked as A, B, C, D and E, that are not detected exactly as human visual inspection did. The reason is, perhaps, because the algorithm is based on local optimization rather than global optimization. Another reason is that humans often tend to focus on straight-line in visual examinations [132].

However, we are also interested in whether the structural change points discovered are sensitive to the initial data point. We designed the experiment with the fixed window size of five, and a threshold value of 1.0, and the model; the initial data point is shifted forward one at a time until the first structural change point discovered by the algorithm. After a series of structural points are discovered, these points are then compensated back to their original sequence for the purpose of comparison. The final results are listed in the table 6.4, where the first row indicates the number of data points shifted forward, and each column is the sequential number of data

points that represent the structural changing points. We can see that the structural change points are not sensitive to the initial starting points other than the first structural change point discovered. The algorithm converges quickly after the first structural points as the data points shift forward. The errors of each structural change point discovered are within ±2. Therefore, we conclude that our structural change mining algorithm is not sensitive to an initial starting point.

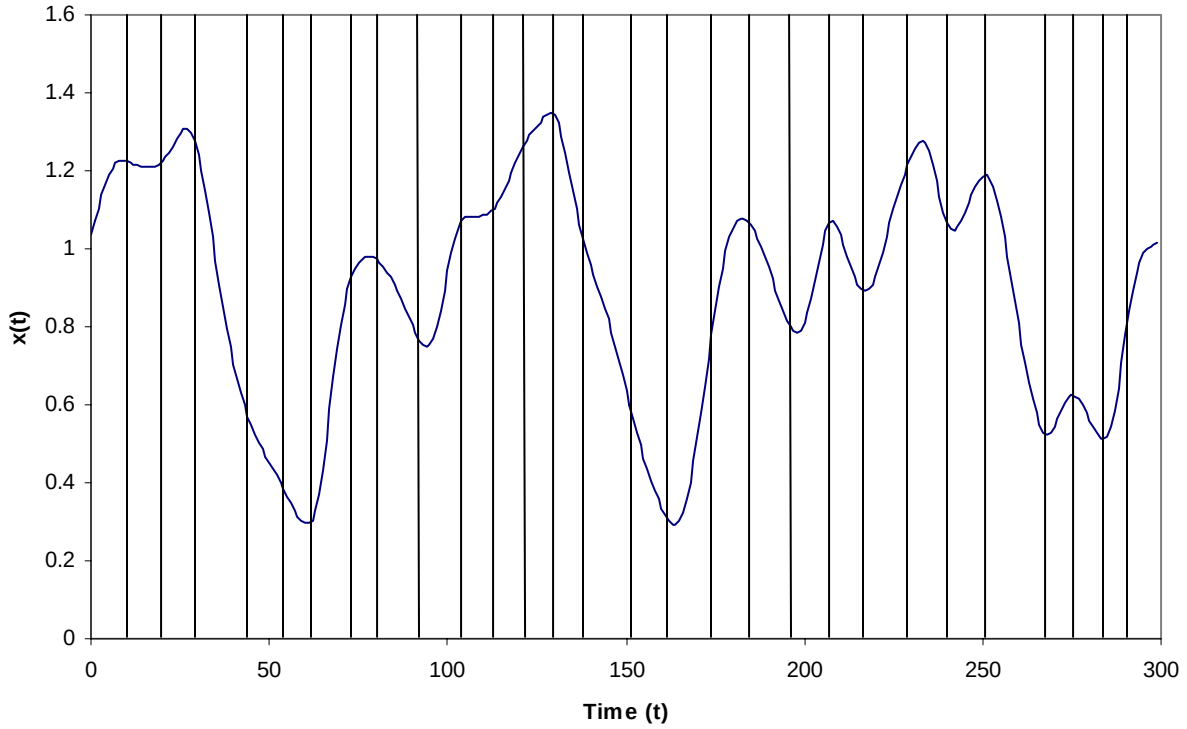

Figure 6.12. Structural change points with threshold of 1.0

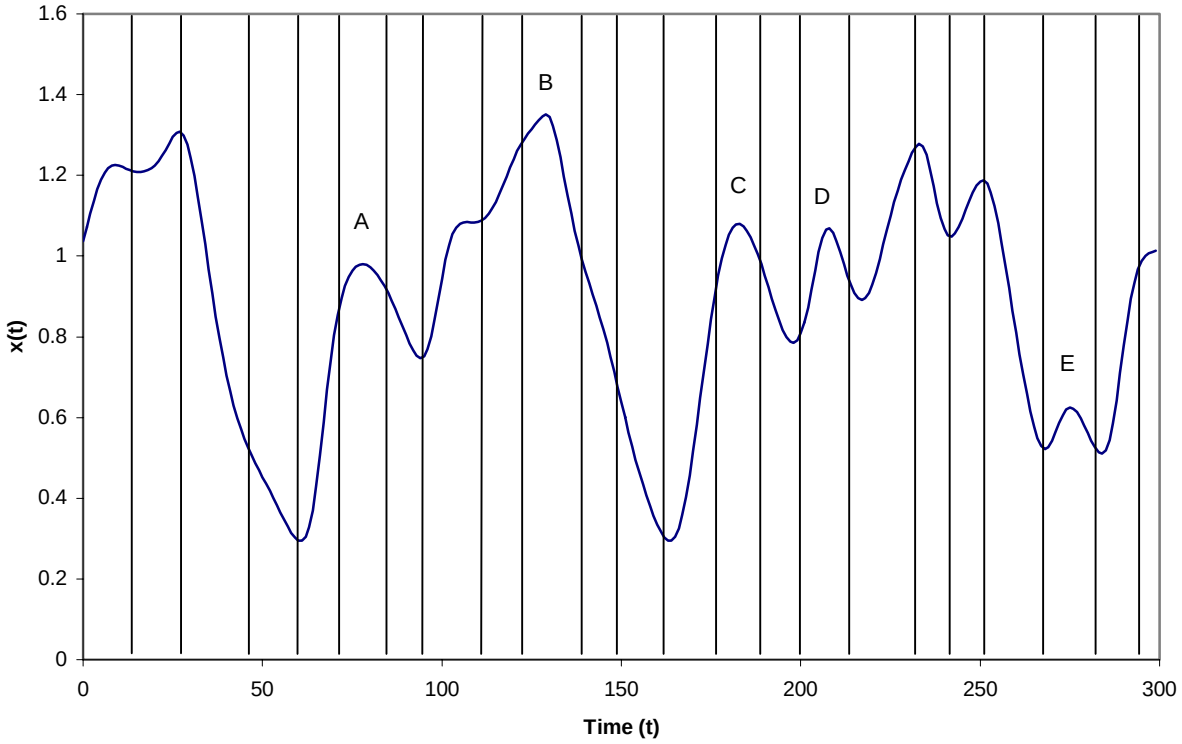

Figure 6.13. Structural change points with threshold value of 2.0

Table 6.4 Initial Data Point Sensitivity of Structural Change Algorithm

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
16	15	15	14	15	14	13	12								
26	25	25	26	25	26	25	26	26	26	27	26	27	26	25	26
42	43	43	42	43	42	43	42	42	42	39	42	39	42	43	42
50	51	51	50	51	50	51	50	50	50	51	50	51	50	51	50
71	70	70	71	70	71	70	71	71	71	70	71	70	71	70	71
92	93	93	92	93	92	93	92	92	92	93	92	93	92	93	92
100	101	101	100	101	100	101	100	100	100	101	100	101	100	101	100
110	109	109	110	109	110	109	110	110	110	109	110	109	110	109	110
120	121	121	120	121	120	121	120	120	120	121	120	121	120	121	120
155	156	156	155	156	155	156	155	155	155	156	155	156	155	156	155
175	174	174	175	174	175	174	175	175	175	174	175	174	175	174	175
196	195	195	196	195	196	195	196	196	196	195	196	195	196	195	196
206	205	205	206	205	206	205	206	206	206	205	206	205	206	205	206
214	213	213	214	213	214	213	214	214	214	213	214	213	214	213	214
226	225	225	226	225	226	225	226	226	226	225	226	225	226	225	226
240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240
246	246	246	246	246	246	246	246	246	246	246	246	246	246	246	246
267	267	267	267	267	267	267	267	267	267	267	267	267	267	267	267
283	283	283	283	283	283	283	283	283	283	283	283	283	283	283	283
289	289	289	289	289	289	289	289	289	289	289	289	289	289	289	289

Structural Change Algorithm Parameter Sensitivity Experiment

In our previous experiments, we have roughly determined that the window size used to build the model is better less than seven in order to achieve better MSE, MAE, MAPE, DA and Theil's Inequality Coefficient, be classified as high-precision models and outperform random walk model. In this experiment, we are going to study other parameters that influence the structural change detection result/threshold value. Four different models and different threshold values are used in the experiments. The structural change results are measured under the two-dimension measure matrix discussed in chapter four. The results are presented from figures 6.14 to 6.16. In this experiment, only the first hundred data points are used.

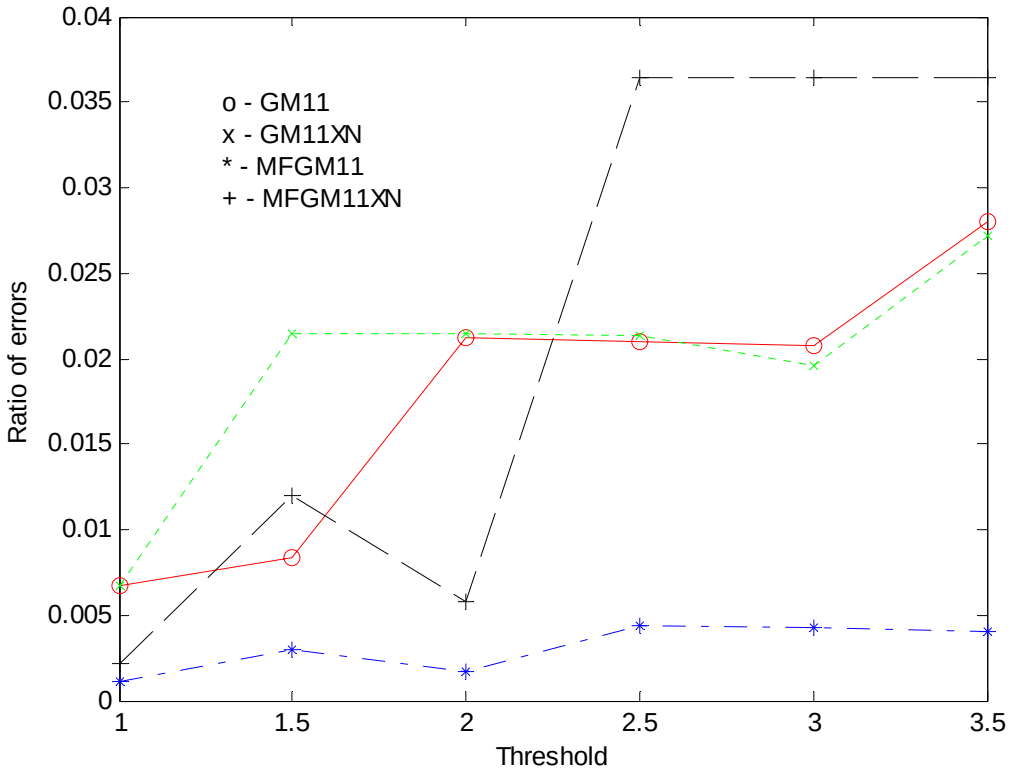

Figure 6.14 Threshold vs. Ratio of Error from Algorithm

Figure 6.14 presents the influence of threshold values to the ratio of errors. The ratio of errors increases as the threshold value increases for all four models. Non-Fourier corrected models move closely as the threshold value increases. However, the Fourier corrected models diverge. The MFGM11XN model tends to have a higher ratio error than the MFGM11 model, which looks more stable to the change of threshold. This is because the MFGM11 model converges to a stable number of structures discovered faster as the threshold increases (as shown in figure 6.15).

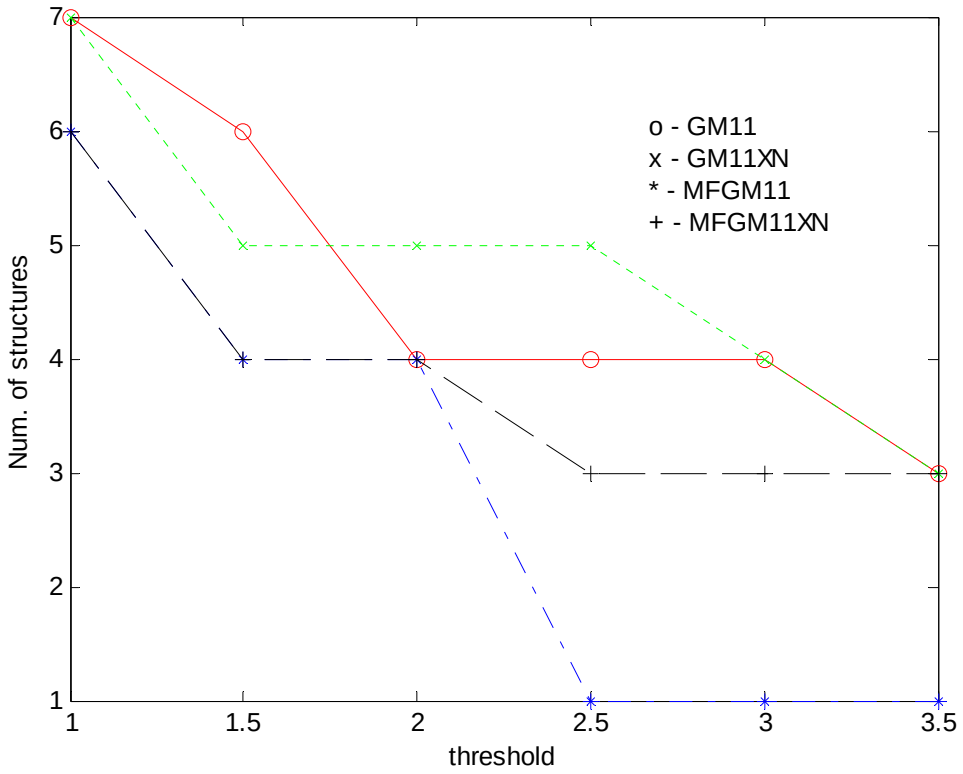

Figure 6.15 Threshold vs. Number of Structures from Algorithm

Figure 6.15 shows the change of threshold vs. the number of structures discovered. As we have expected, the higher the value of threshold, the lower the number of structures discovered. The model GM11, GM11XN, and MFGM11XN converges to the three structures as the threshold increases, which is close to human justification. The MFGM11 model does not perform very well as the threshold increases and the number of structures detected drop quickly from 3 to 1 as the threshold increases from 2 to 2.5.

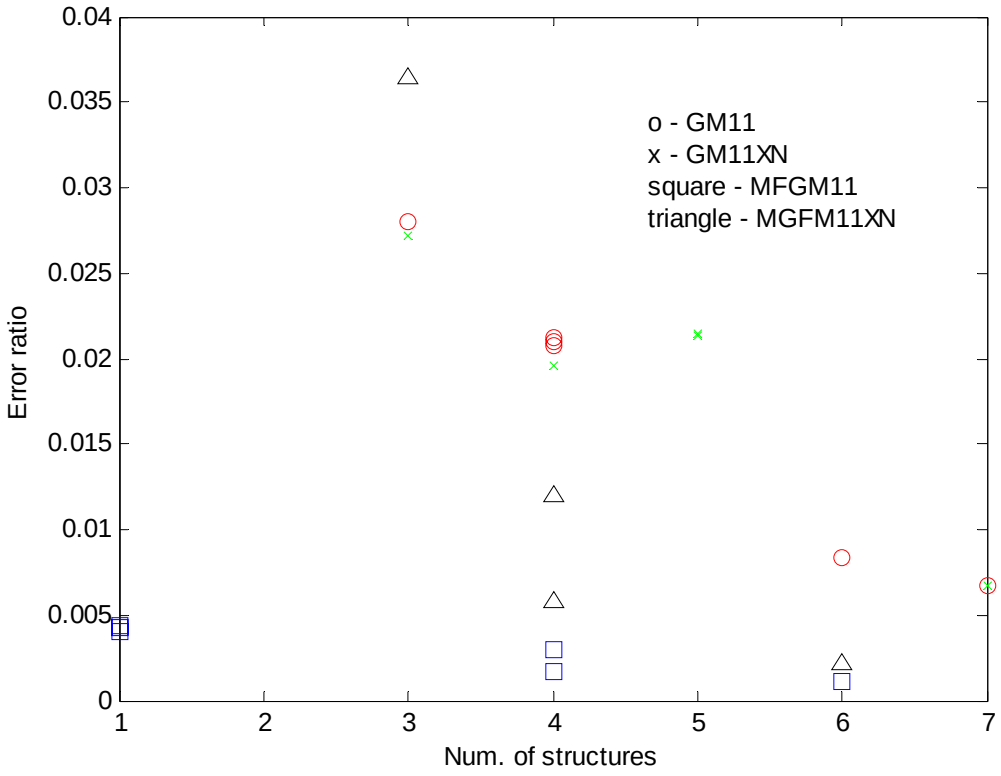

Figure 6.16 2D Comparison Domain analysis of Models

Figure 6.16 displays the comparison based on two-dimensional space of error ration and number of structures of all four models and different thresholds. All of these four models present a trend: i.e., that a smaller threshold value has a smaller ratio of error but a higher number of structures discovered.

Synthetic Similarity Cluster Evaluation

We have proved that our newly developed distance measure for time series similarity matches is a metric function. Shasha, et al. [90] showed that Euclidean distance alone does not provide an intuitive measure of similarity when the time series compared are of different baselines and scales. Therefore, shifting transforms or scaling transforms are often performed

before measuring Euclidean distance. Here the shifting transform is defined as the transformation of the old time series by adding some real numbers to each item, thereby creating a new time series. Scaling transform of a time series leads to a new time series by multiplying some real number to each item in the old time series. A simple way to make a similarity measure invariant to shifting and scaling is to normalize the time series.

Definition 6.1. (Normal Form) The normal form $\overrightarrow{X}$ of a time series $\overrightarrow{X}$ is transformed from $\overrightarrow{X}$ by shifting the time series by its mean and then scaling by its standard deviation.

$$\overrightarrow{X} = \frac{\overrightarrow{X} - avg(\overrightarrow{X})}{std(\overrightarrow{X})}$$

It is trivial that the normalized time series have the properties $avg(\overrightarrow{X}) = 0$ and $std(\overrightarrow{X}) = n$. The Euclidean distance between the normal forms of two time series is a similarity measure between time series that is invariant to shifting and scaling because they have the same baseline and scale[90]. We use four sets of time series data that have different standard deviations as listed in table 6.5.

Table 6.5 Sample time series for similarity matching

Series	A	B	C	D
1	100	50	200	50
2	110	55	250	62.5
3	90	45	150	37.5
4	100	50	200	50
Average	100	50	200	50
Standard Deviation	8.1649	4.0824	40.8248	10.2062

The plots of the raw time series data and the normalized time series are shown in Figure 6.17 and 6.18 separately.

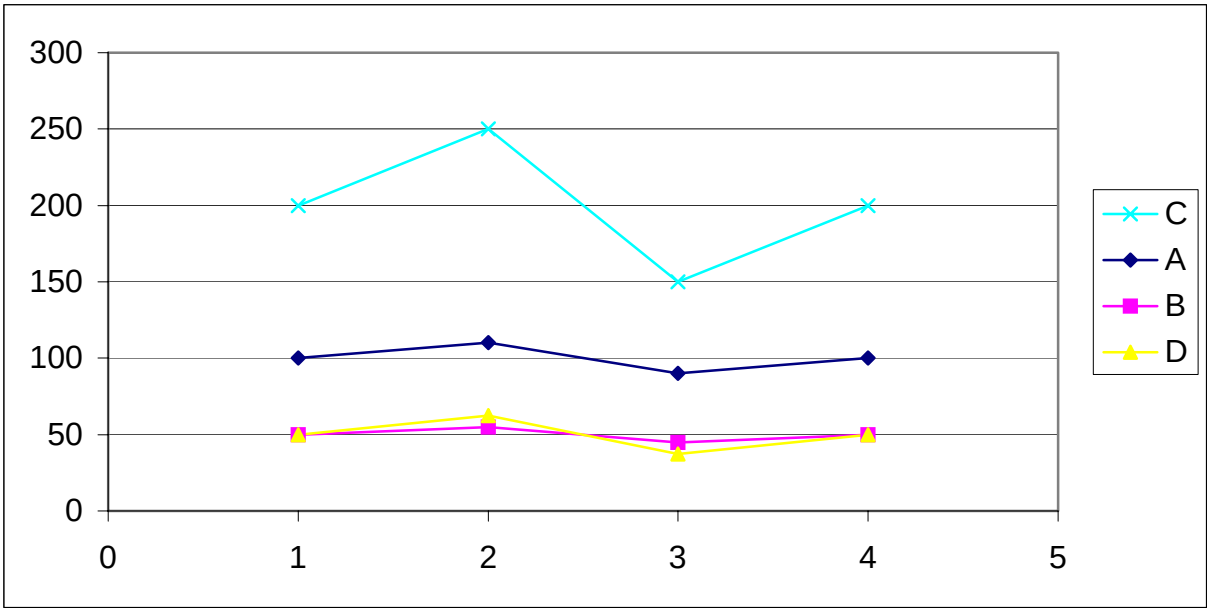

Figure 6.17. Time series data plot without normalization

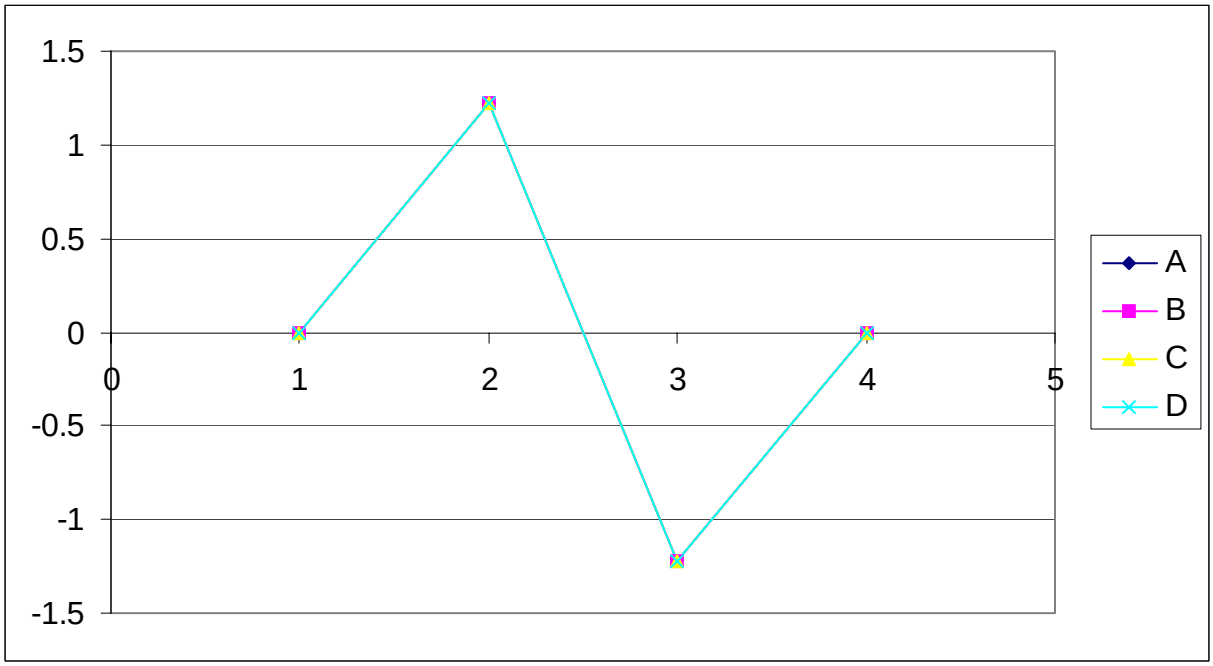

Figure 6.18. Time series plot in the normalized form

It is clear that after the normalization, the Euclidean distance measure will never distinguish these four time series because they are all overlaid on top of each other. Basically, it means that all these four series are considered as 'same'. However, from the point view of our perception, time series 'C' varies greatly from the other three, which can be easily seen from the time series plot in figure 6.17. It also can be confirmed from the standard deviation, which is often used to represent the 'risk' of investment or 'volatility' of the time series.

The distance measure computed from our proposed metric function is shown in table 6.6 in the increasing order, which the higher value means less similar.

Table 6.6. Similarity distance measure

	AD	AB	BD	CD	AC	BC
Distance	6.1237	12.2474	18.3711	91.8558	97.9795	110.3711

It is clear that our proposed metric measure works better than directly use of Euclidean distance. Particularly, the result is close to human visual judgment. In the example, time series 'A' and 'D' are complete parallel to each other, and in our result, they are clustered as closest. 'B' is a little bit away from 'B'; therefore, 'AB' and 'BD' are closely grouped together. 'C' is furthest from the group 'A', 'B', and 'D'. In the similarity clustering, our metric measure uses raw date directly without any shifting and scaling; even the data are of different baselines and scales and the model performs close to human perceptual judgment. The clusters of these four time series are presented in figure 6.19 and figure 6.20. In figure 6.19, the cluster that is based on Euclidean distance shown, and figure 6.20 shows the cluster based on our distance measure.

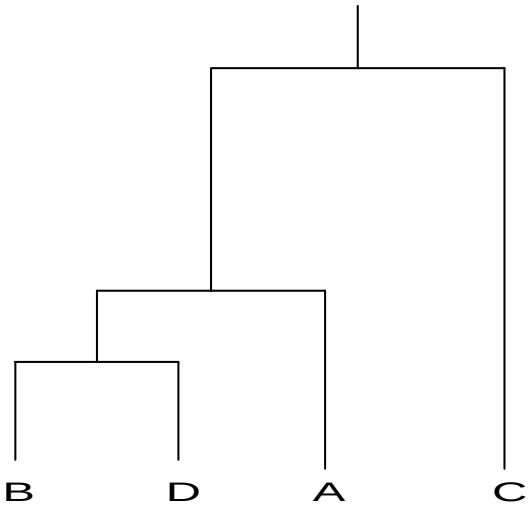

Figure 6.19: Cluster based on Euclidean distance of series A, B, C, and D

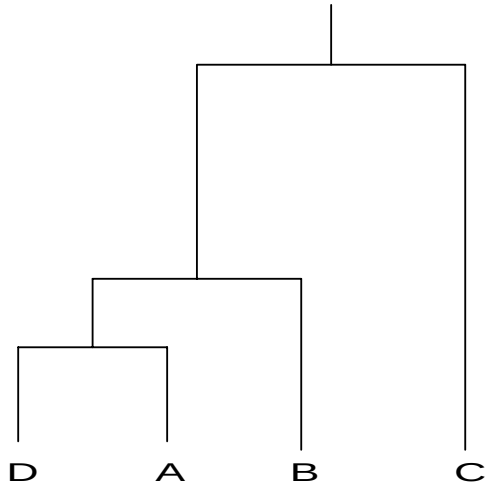

Figure 6.20. Cluster based on our distance measure of series A, B, C, and D

CHAPTER 7

REAL TIME SERIES EVALUATION

Structural Change Mining for Financial Data and Similarity Matches

Recent time series data mining tasks have been focusing on how to identify a given pattern from time series and pattern matching on a given set of pattern templates [44, 45, 52, 90, 133-135]. However, the use of the fixed template or pattern matching or identification simplified the inherent structural change of the time series itself, and the characteristics of the financial time series such as high noise-to-signal ratio, nonstationarity and limitations on data availability make the problem more complicated in the pattern (*a priori*) to be identified or matched.

In financial time series data mining applications, the detection of patterns and trading signals has been explored to obtain abnormal gains in investment [136]. Many time series may present the structural change phenomenon that is critical to capital management and financial time series data mining. In structural time series, the generating system of the series is assumed to consist of four components: trend, cycle, seasonal variations, and irregular fluctuations. Kovalerchuk [137] pointed out that financial data mining techniques are designed to discover hidden trends and patterns in a financial database, e.g., in stock market data for market prediction. However, the question is how to separate real trends and patterns from mirages. Otherwise, it is equally dangerous to follow any of them. Financial forecasting has been widely studied as a case of time-series prediction problem. The difficulty posed by this problem is due

to the following factors: Low signal-to-noise ratio, non-Gaussian noise distribution, nonstationary, and non linearity. After Osborne [138] proposed the random walk characteristic of stock market in 1959. The random walk model has been widely considered as a statistical model for the movement of the logged stock price. Burton [139] suggested that in order to save on hefty broker and management fees, investors should entrust their stock-market portfolios to a blindfolded monkey tossing darts at the financial pages of newspaper. However, Lo and MacKinlay [140] marshal the most sophisticated techniques of financial theory to show that the market is not completely random after all; that it has, so to speak, a memory. In the short run, broad market returns are positively correlated, like the weather. Just as a fair day is more likely to be followed by another fair one than by rain, a positive return in one week is more likely to be followed by a positive return in the next. In the long run, however, individual stock returns display negative serial correlation: winners over the past three years are somewhat more likely to be losers over the next three years.

Artificial intelligence techniques such as artificial neural network (ANN) and genetic algorithms (GA) have been applied to forecast stock prices as the computation power increased dramatically[141-143]. Those approaches are based on the training data that includes those far away from the present to train the model and thus produce prediction. Thus, the data are not fully considered as in the time series because the all data are treated without any preference, and those economical and fundamentals of the market might have differed greatly from these of the prediction is made. Kim and Han [144] also showed that ANNs had some limitations in learning the patterns because stock price data contains tremendous noise and complex dimensionality; also, the sheer quantity of stock data sometimes interferes with the learning of patterns.

Samuelson [145] and Mandelbrot [146] stated that the unexpected price changes reflect new information. However, if today's stock prices reflect all available information, then tomorrow's price movements must be unforeseeable, since any information that might be used to forecast has already been incorporated by traders into today's prices. If the stock price change reflected its market information, then the impact and strength of the information related to price change is relatively gray. Intuitively, traders believe that "good" or "favorable" information tends to drive the price up, that "bad" information would push the price down, and that competition will drive all information into the price quickly, causing the stocks to sell for their *true values.* The heterogeneity of market participants weights their opinions and evaluation of information in respect to their wealth, expectations, risk reversion, and forecast accuracy. Overall market price will reflect a mixture of all these diverse opinions weighed by the amount of money that supports it. Given these complex factors, it is extremely difficult to measure all the relevant information and true stock prices.

Mining financial data presents special challenges. For one, the rewards for finding successful patterns are potentially enormous, but so are the difficulties and sources of confusion. On the other hand, in an efficient market, prices reflect all available information. The implication of this definition is that it is not possible to predict price movements from available information, since this information is already reflected in prices. Because the arrival of information is random, and given that the new information is reflected in prices very quickly, the period-to-period changes in prices tend to be random. Another implication is that it is not

possible to earn abnormal (higher) returns via active trading as compared to what can be obtained from a passive buy-and-hold strategy.

However, we have good evidence that short-term trends do exist and that programs can be written to find them. The data miners' challenge is to find the trends quickly while they are valid, as well as to recognize the time when the trends are no longer effective. The other challenge of financial time series data mining is how to find the group of products that have a similar historical trading pattern. Additional challenges of financial time series data mining are to take into account the abundance of domain knowledge that describes the intricately inter-related world of global financial markets and to deal effectively with time series and calendar effects.

In the following section, three main indices in the financial market of the United States are used for structural change mining study. They are Dow Jones Industrial Index (DJI), Nasdaq Index (IXIC), and Standard & Poor 500 Index (SP500). The time frame is from January 2000 to December 2004. Four different models are studied in the research: general gray model GM(1,1) (GM11), modified GM(1,1) that Xn is initial condition (GM11XN), Fourier enhanced residual GM(1,1) model (MFGM11) and Fourier enhanced residual GM11XN model (MFGM11XN).

The forecast precision experiment and structure parameter sensitivity experiment are performed based on IXIC time series data. The algorithm used is the one without validation. In order to save space here and the design suggested by the committee, only forecast precision experiment is conducted based on the DJI time series and SP500 time series. In the later section,

clustering and similarity matching experiment are tested based on eight randomly picked stocks and results are discussed.

IXIC Forecast Precision Study

The IXIC data are fed into the algorithm in a real-time fashion. We have discussed the model and parameter selection regarding the Mackey-Glass chaotic time series in chapter six. In this experiment, a similar design is used except that another variable is added – period of moving average. We already know that financial data have a high noise-to-signal ratio, therefore, aggregation over raw data streams is both necessary and important for practical applications. Aggregation makes sure there is a unique value for each time instance over a fixed time interval. We need to filter out the noise before further data processing. In our research, the standard moving average, which is widely used in the financial market, is used to smooth the data:

$$MA_p(i) = \frac{1}{p} \sum_{j=i-p+1}^{p} x(j)$$

Where *x(i)* is the value for *i=1, 2,…n* and *p* is the number of periods. $MA_p(i)$ computes the *p-interval* moving average time series that assigns equal weight to every data point in the average interval. Short-term noise will be filtered out and a clear temporal pattern signal is generated by smoothing through the moving average. In the experiments, the *p* ranges from 2 to 10. The window size used by models is same as the Mackey-Glass study. The raw IXIC time series from Jan. 2000 to Dec. 2004 are shown in figure 7.1. Because the difficulties to put all these data into a graph and for the purpose of easy visualization, only 300 data points are chosen; these are marked in light-blue in figure 7.1. These data cover a time frame from Jan. 2001 to March 2002, which includes a well-known 9/11 special event. These 300 data points are plotted in figure 7.2, and the results of prediction precision are shown in figures 7.3 to 7.9.

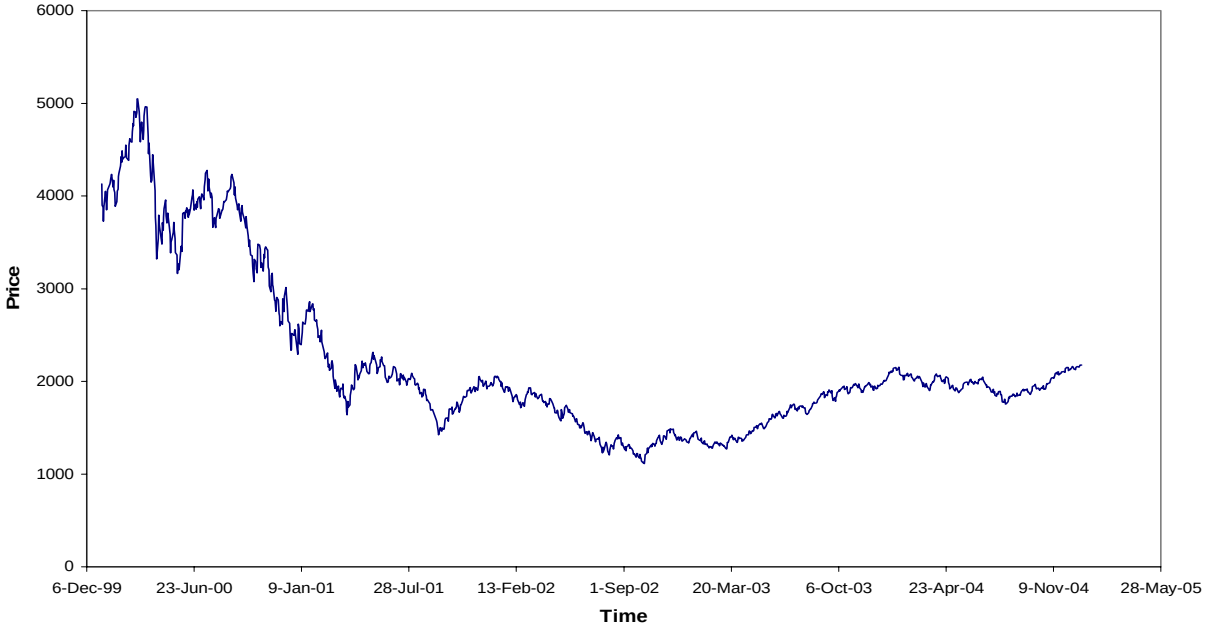

Figure 7.1 IXIC complete data plot from Jan. 2000 to Dec. 2004

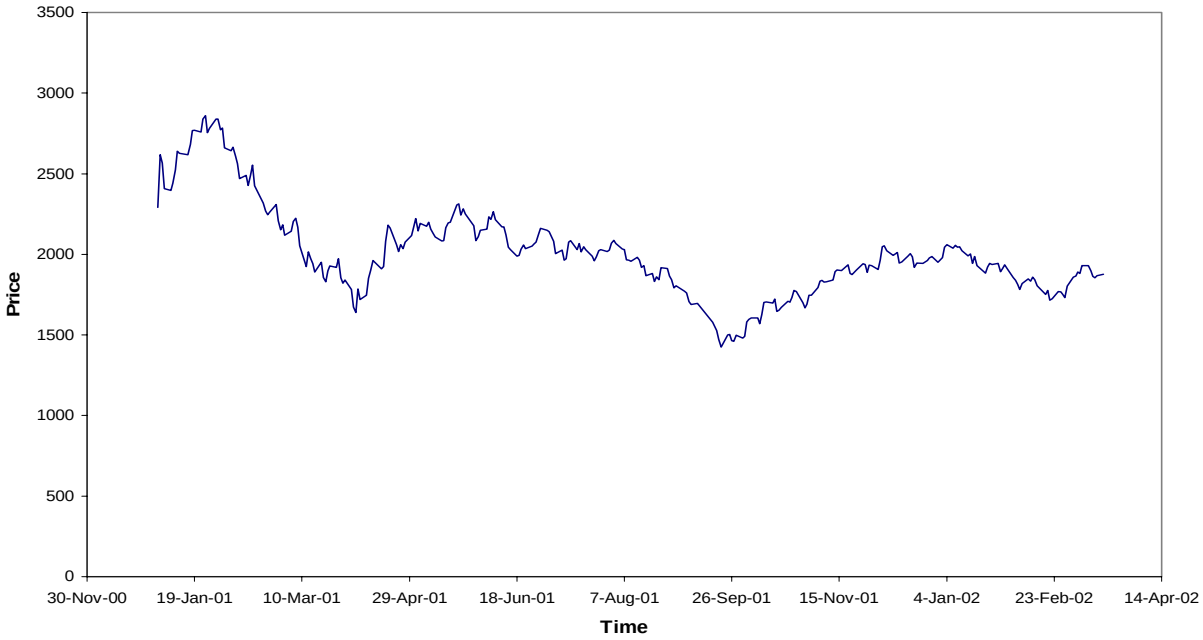

Figure 7.2 IXIC partial data (300 data points)

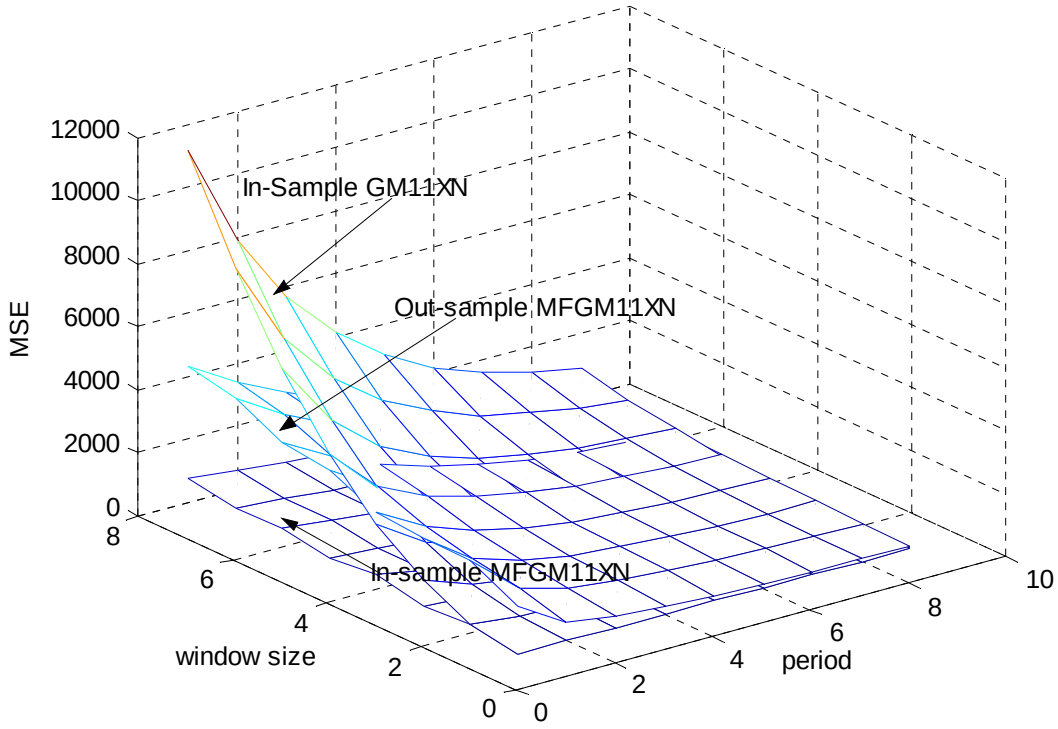

Figure 7.3 IXIC MSE changes vs. period and window size

Figure 7.3 shows the change of MSE in relation to changes of periods and windows size used by models both in-samples and out-samples. The Fourier corrected gray model has a far small MSE values than that of non-Fourier corrected models. The more noises that are being filtered out, the less the MSE values are; i.e., the filtering out the noises helps to increase the prediction performance of all models. The bigger window size is, the higher the MSE values are for models and in-samples/out-samples. The Fourier corrected model has very small variations on MSE changes with window size and period. The out-sample of models demonstrates relatively smaller MSEs compared to in-sample models.

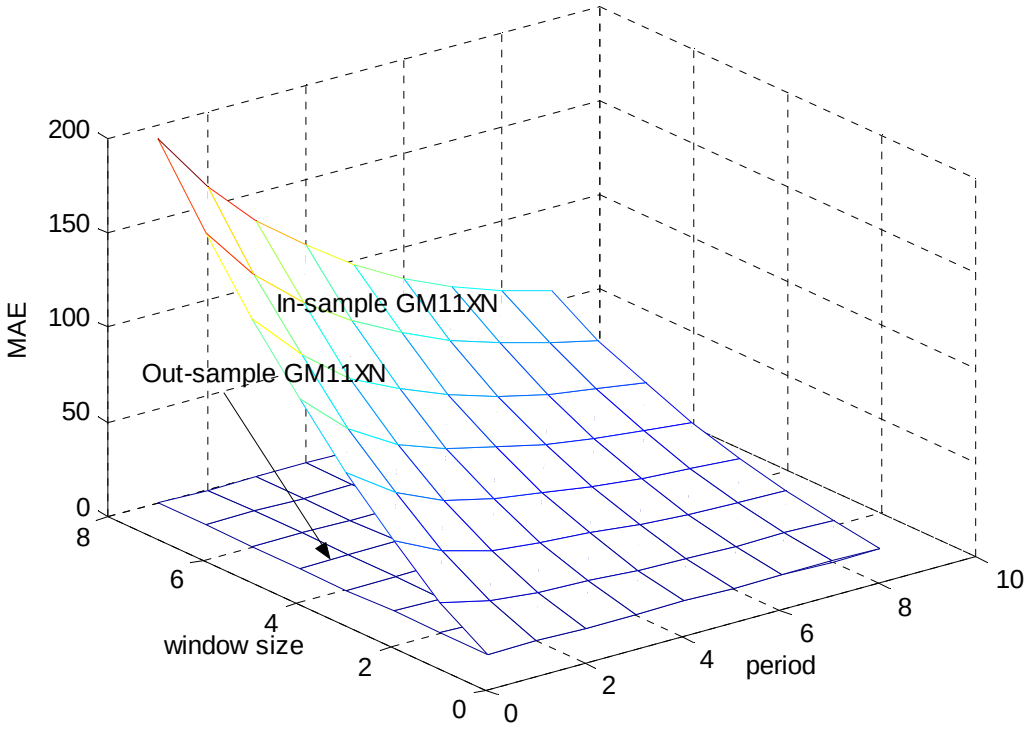

Figure 7.4 IXIC MAE changes vs. period and window size (in, out sample)

Figure 7.4 presents the change of MAE corresponding to periods and window size. It is obvious that the out-sample MAE of the model is much smaller than that of in-sample. The variation of MAE on out-samples is very small. For in-samples, MAE increases as the window size increases. The filtering out of noise decreases the MAE values.

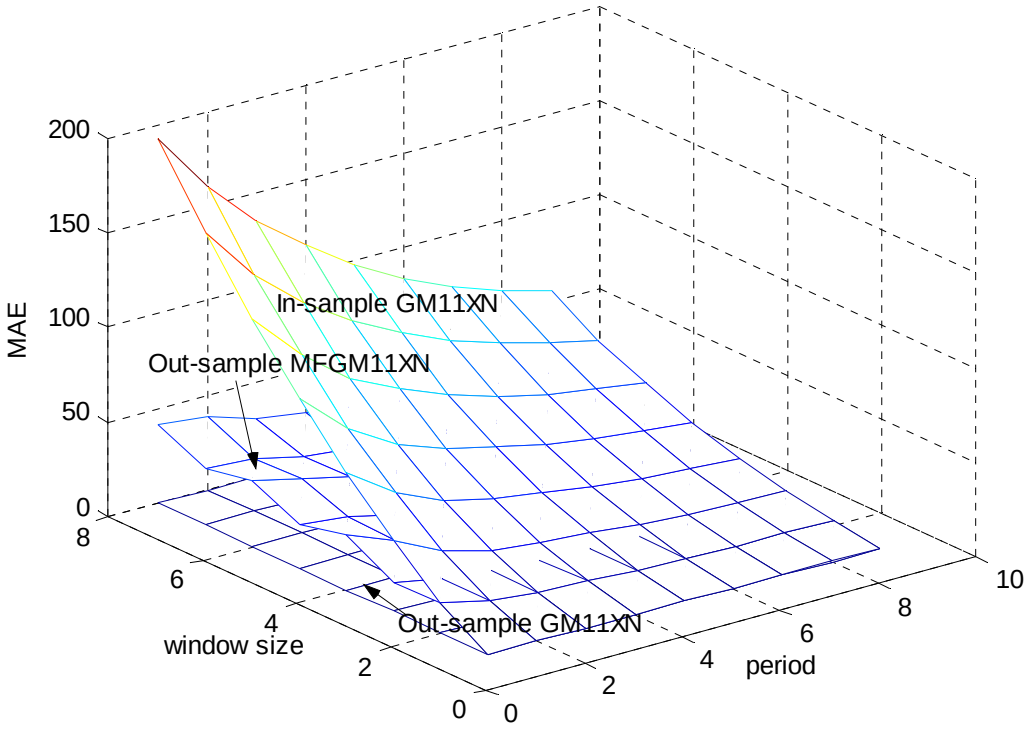

Figure 7.5 IXIC MAE changes vs. period and window size (Model)

Figure 7.5 further illustrates the change of MAE corresponds to different models. Even the Fourier corrected model – MFGM11XN has a higher MAE values compare to the out-sample of GM11XN model. It also shows that the window size increases tends to increase the MAE also. Smoothing out noises decreases the MAE. Out-sample MAE is far smaller than in-sample.

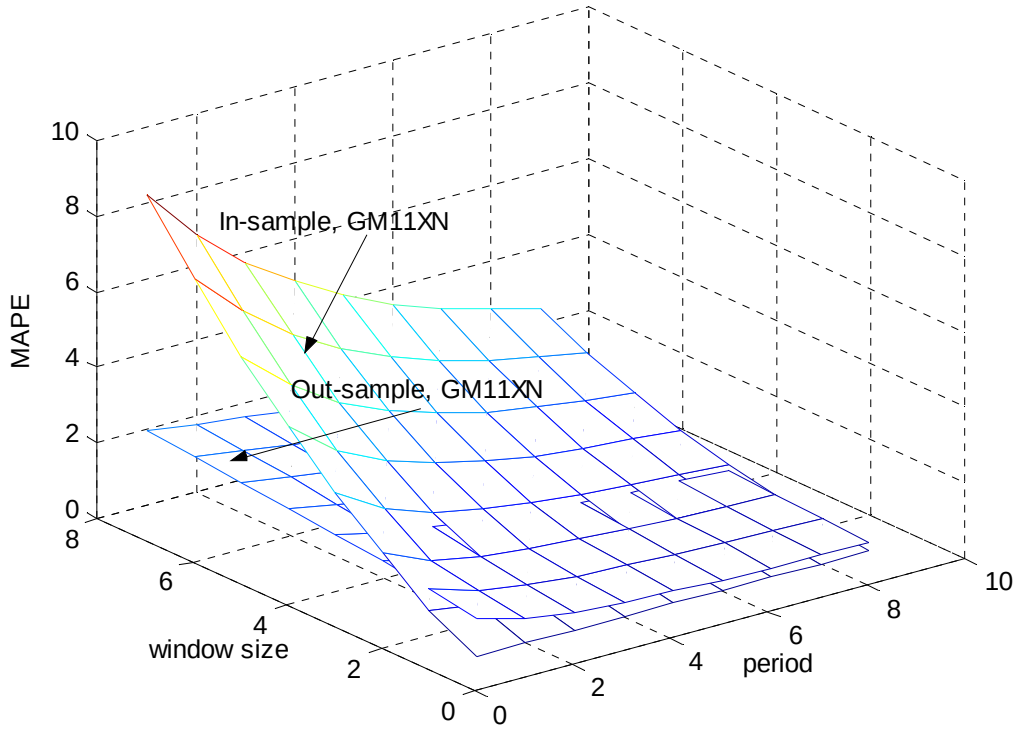

Figure 7.6 IXIC MAPE changes in related to period and window size

Figure 7.6 presents the MAPE change of corresponding window size and periods. All of these models can be categorized as high-precision models based on the threshold of MAPE, both for in-sample and out-sample. Out-sample has a lower MAPE than that of in-sample. The models that use a higher window size tend to have a higher MAPE value. The more noise being filtered out, the lower the MAPE values could be.

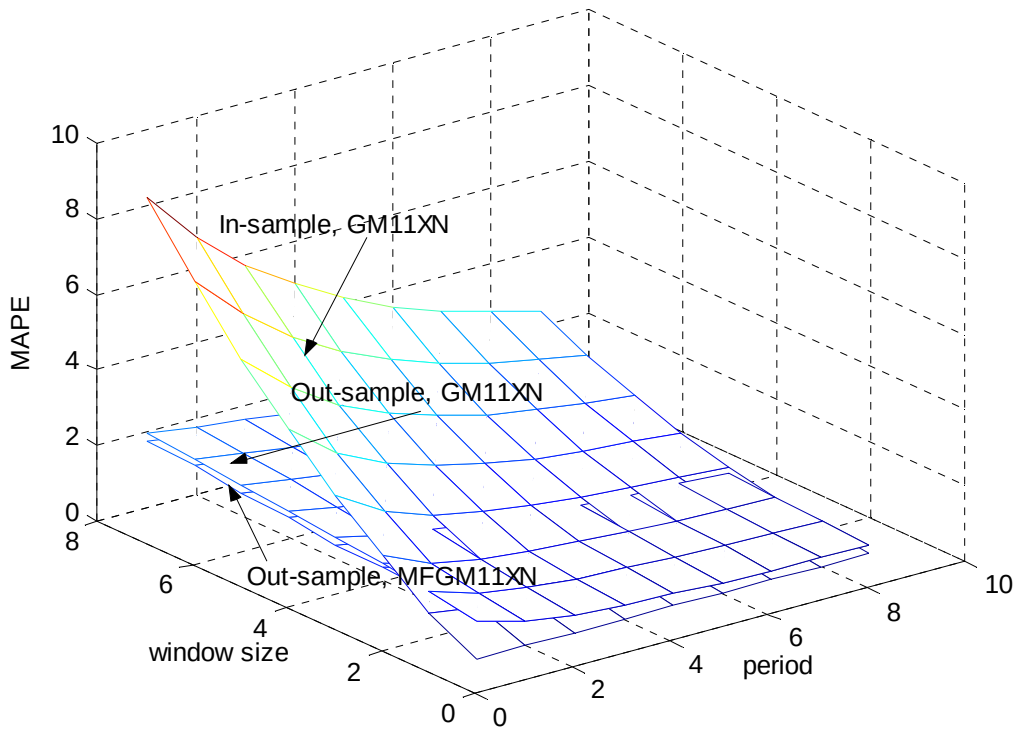

Figure 7.7 IXIC MAPE vs. window size, period, and models

Figure 7.7 further explains how MAPE changing corresponds to the different models. All models have a similar MAPE values at out-sample. In-sample MAPEs are still higher than those of out-samples, except that a smaller window size (such as two) is being used. In out-sample cases, the window size change does not affect the MAPE very much; however, the more noises that are filtered out, the lower MAPEs are for the out-sample also.

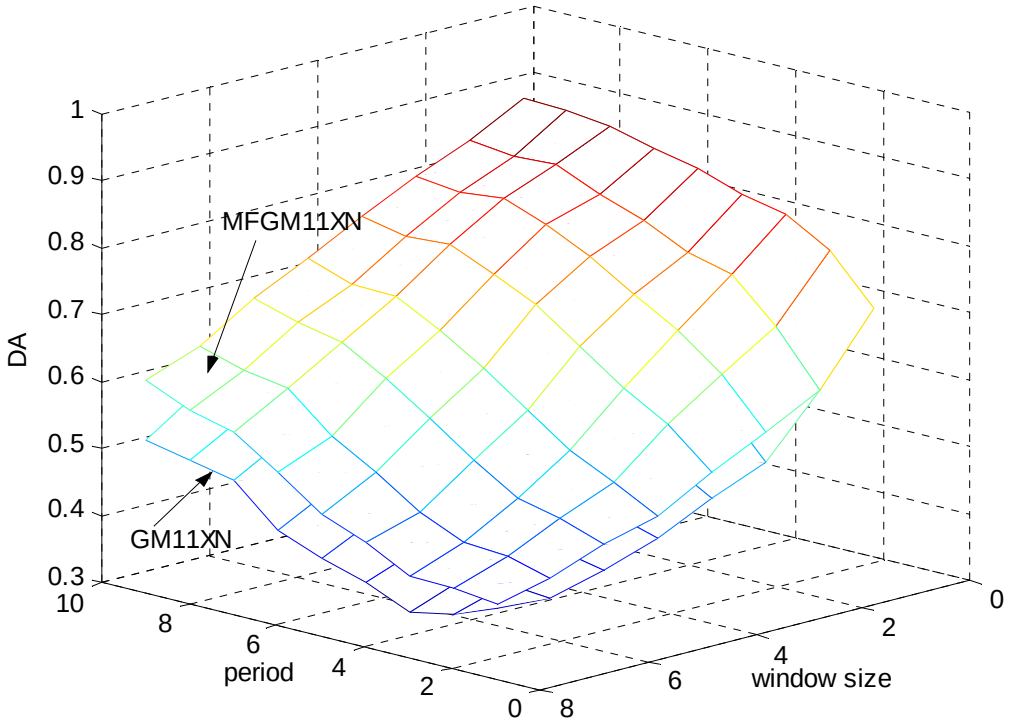

Figure 7.8 IXIC DA changes in related to period and window size

Figure 7.8 presents the direction forecast accuracy changes v.s. the window size and periods. The removal of noises from data helps models to achieve a better directional forecast. That is because of the noise filtering discloses the true temporal trend and patterns, and makes it easier for the model to grasp these temporal patterns. A smaller window size used, a better DA value is. This confirms that the next new data is closely related to the one preceding it. MFGM11XN has a higher DA comparing to the GM11XN model, but when a smaller window size is used, the DA performance of these two models does not differ very much.

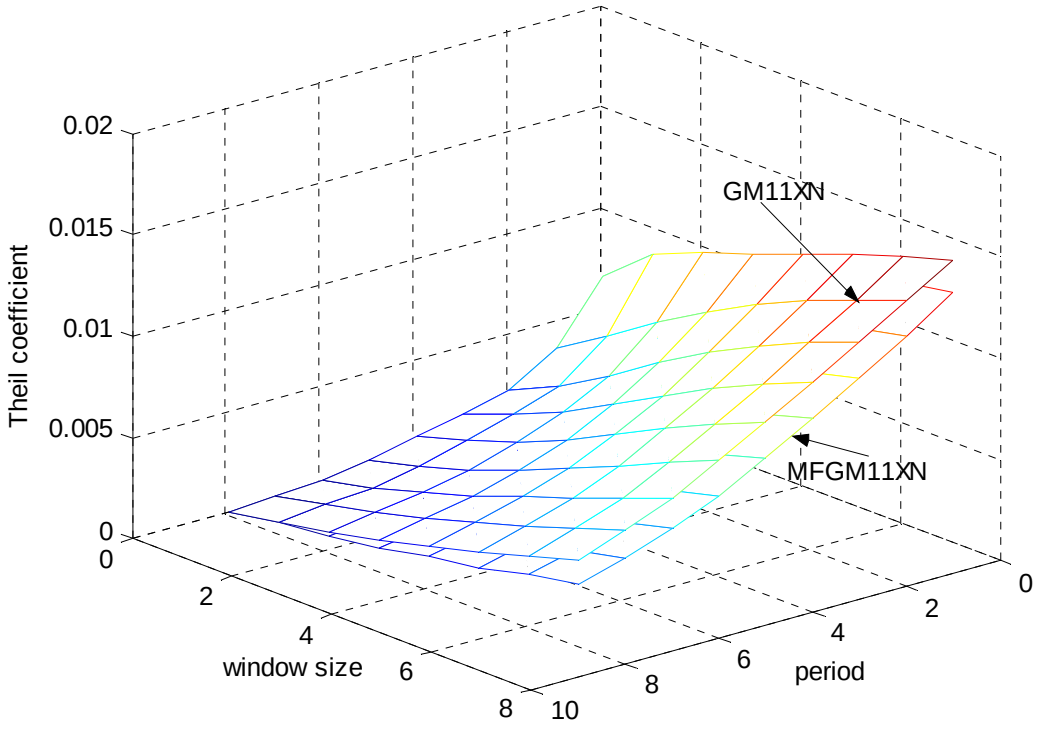

Figure 7.9 IXIC Theil's coefficient changes in related to period and window size

Figure 7.9 shows that increasing of window size also increases the Theil's inequality coefficient for all models. The filter of noise from data stream improves the Theil's inequality coefficient. All of these four models outperform the random walk model based on Theil's inequality coefficient. Fourier-corrected models have a better performance than these not Fourier corrected.

Actually, all of these four gray models could forecast the next day's stock prices changes with relatively high accuracy rates, and a high direction if the right window size and period are used. The experiments suggest that the models perform better if a smaller window size is used

and there are some degrees of noise filtering. The models require a smaller amount of data to achieve high precision and accuracy both on direction and magnitude, and do not depend on statistical distribution assumptions.

IXIC Structural Mining Algorithm Parameter Study

The total 1256 data points are plotted in figure 7.1. In order to visually compare and identify the structure changes mined by the algorithm, the partial 300 data series are depicted in figure 7.2 and figures 7.10 to 7.15. We can see that the number of structures decreases when the window size increases with the same threshold. The higher the threshold value used by the algorithm, the smaller number of structures that will be discovered. The threshold of 1.0 and 1.5 combines with the window size of 5 to achieve a better match with our perceptual identification of turning points. A threshold of 1.5 and a window size of 5 produce a smaller number of structures than that of a threshold of 1.0 and a window size of 5; this is because the higher threshold accommodates higher volatility of the data and noise. However, both of them identify the structures and come close to human justification successfully. From these experiments, we can conclude that smaller window size and threshold generate results that closely approximate human justification. More quantitative comparisons are performed and illustrated in figures 7.16 through 7.21.

We can conclude that the structure changes mining algorithm successfully identified the changes and closely approximated perceptual view when smaller threshold and window size were used. The bigger movements of stock are identified successfully; for example, the event of 9/11 is successfully identified in figures 7.10 through 7.15.

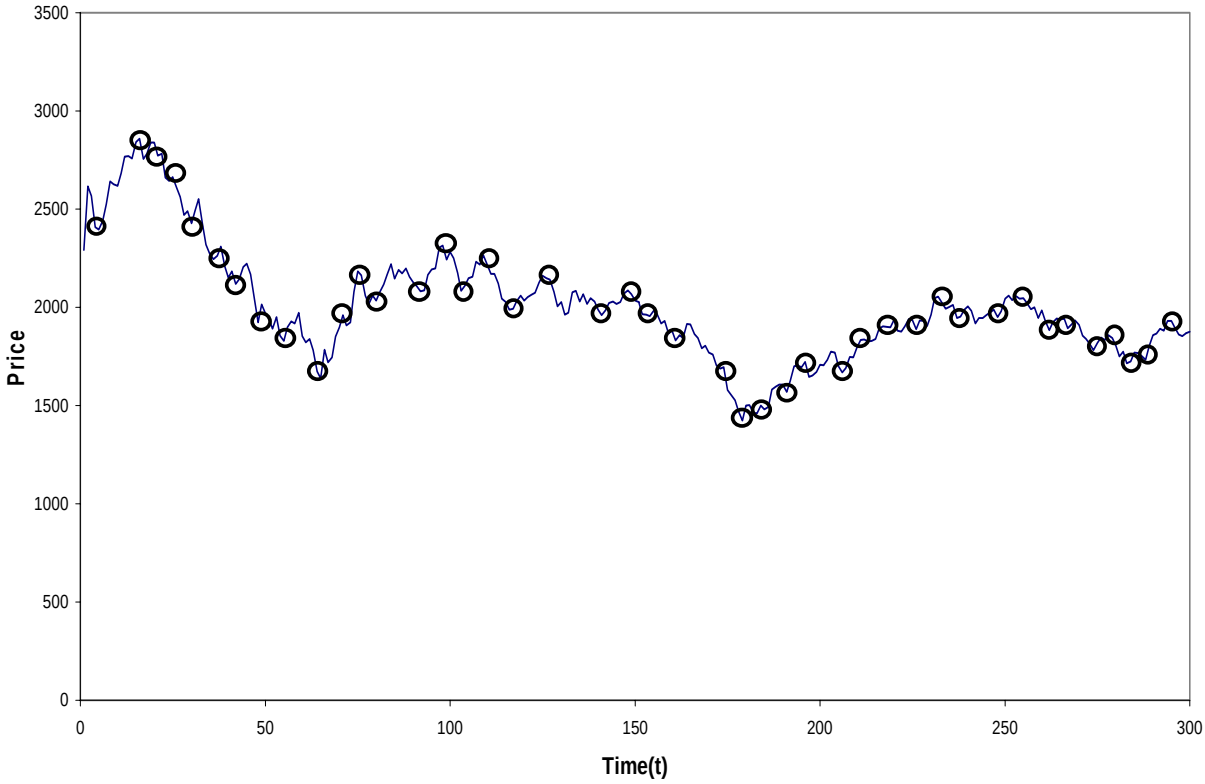

Figure 7.10. IXIC structural change mining result with threshold 1.0 and window size of 5.

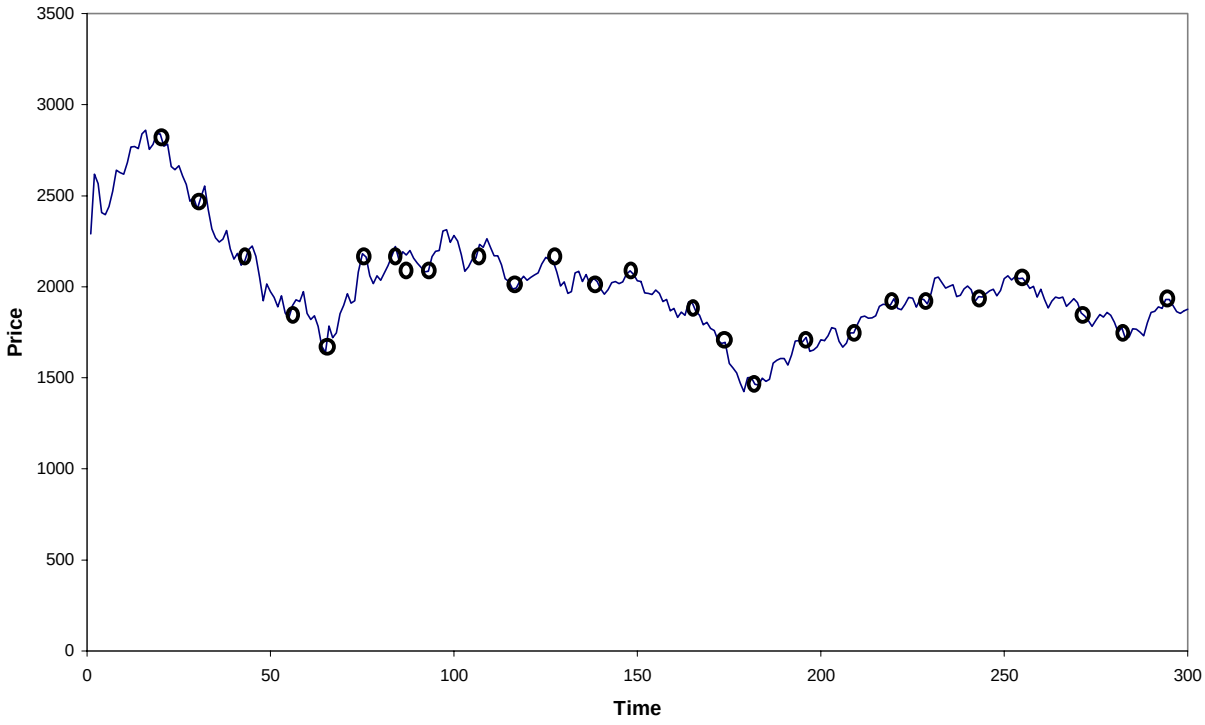

Figure 7.11. IXIC structural change mining result with threshold 1.0 and window size of 10.

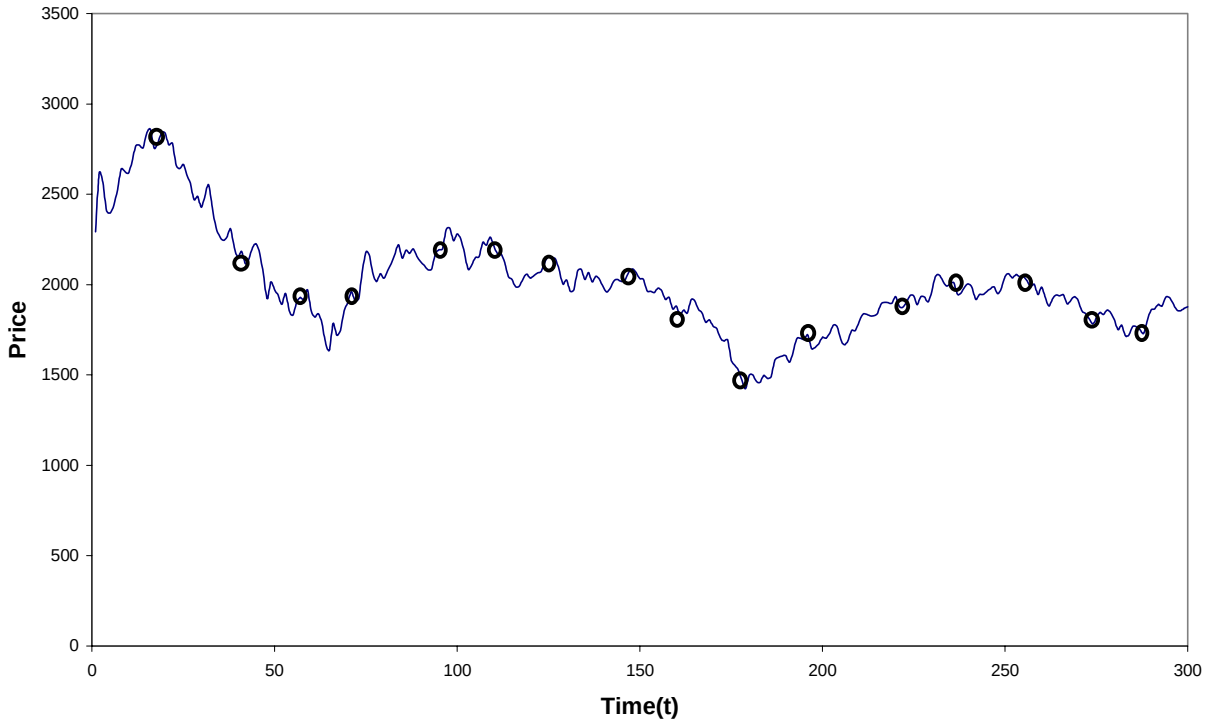

Figure 7.12. IXIC structural change mining result with threshold 1.0 and window size of 15

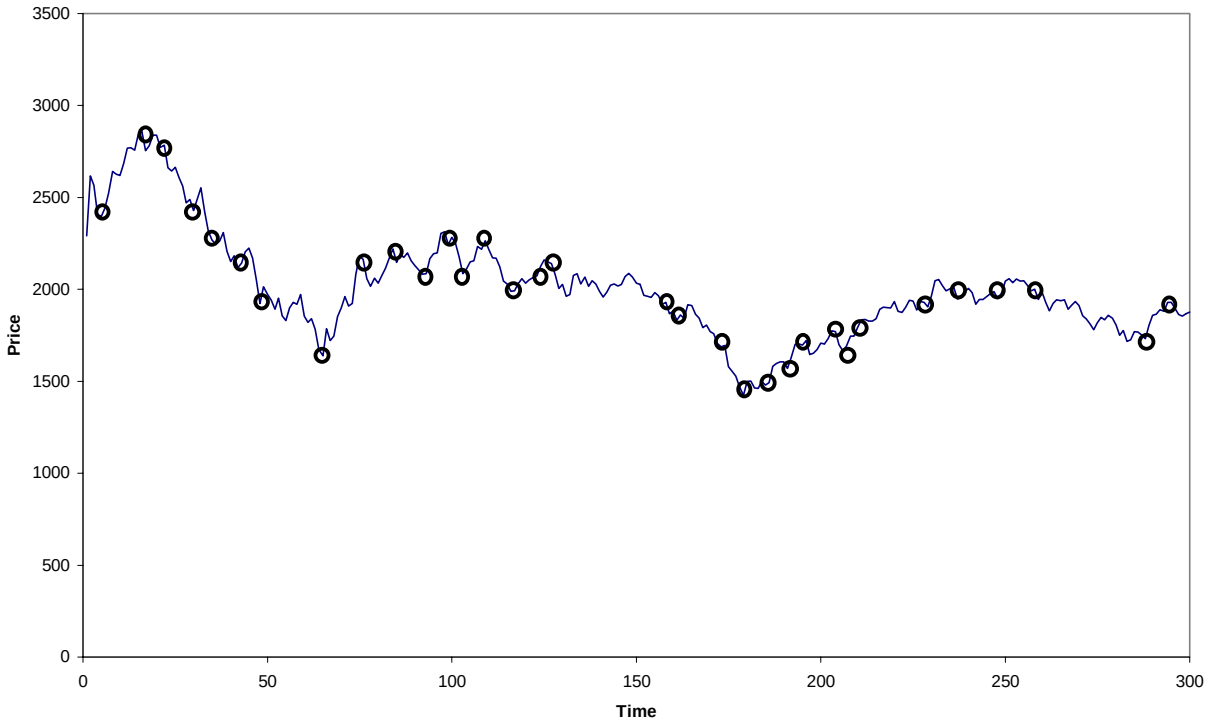

Figure 7.13. IXIC structural change mining result with threshold 1.5 and window size of 5

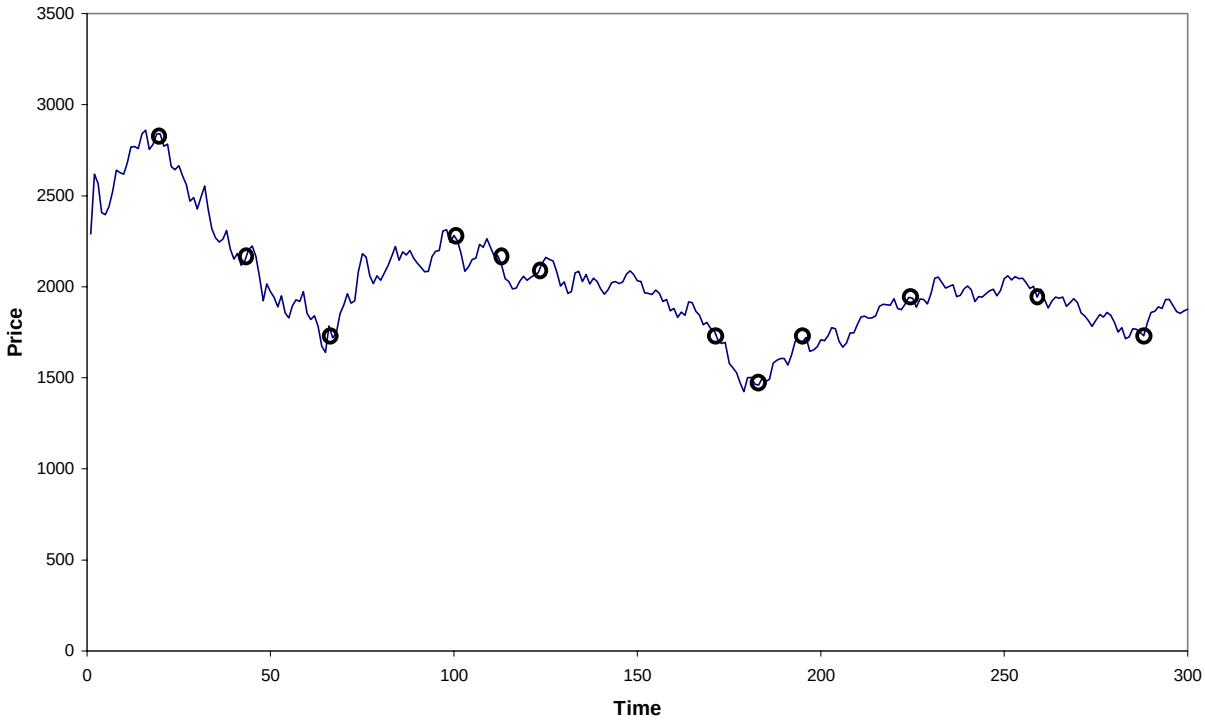

Figure 7.14. IXIC structural change mining result with threshold 1.5 and window size of 10

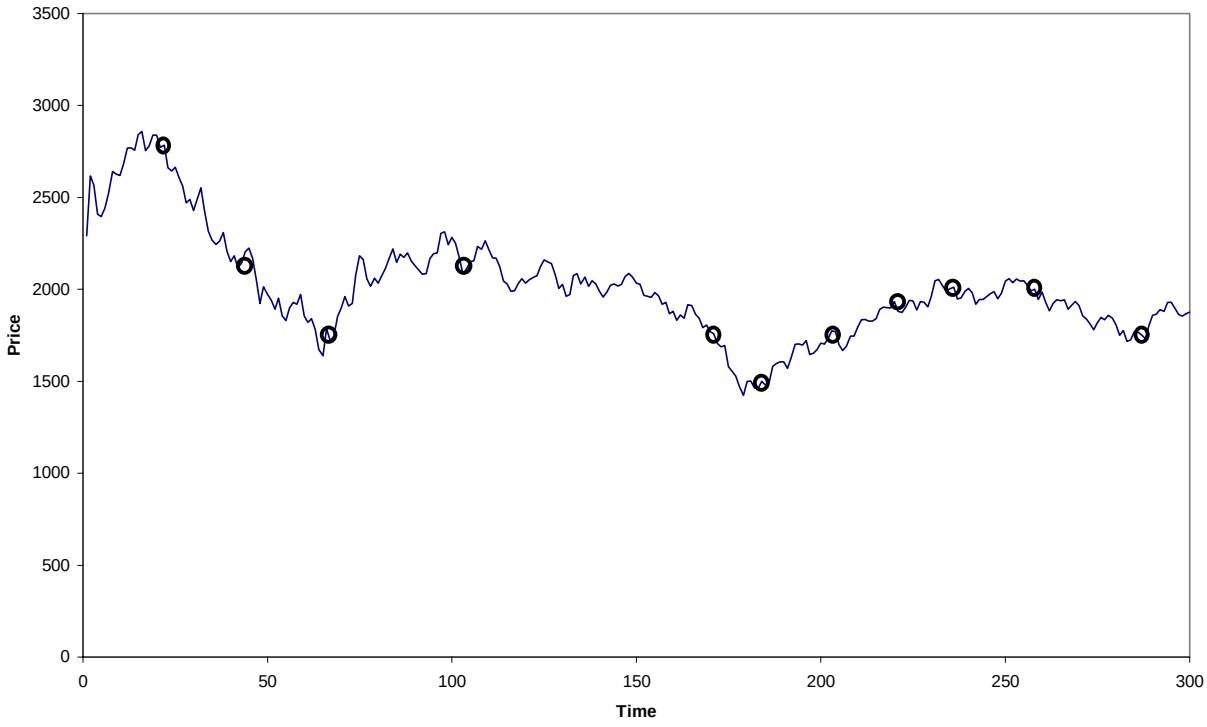

Figure 7.15. IXIC structural change mining result with threshold 1.5 and window size of 15

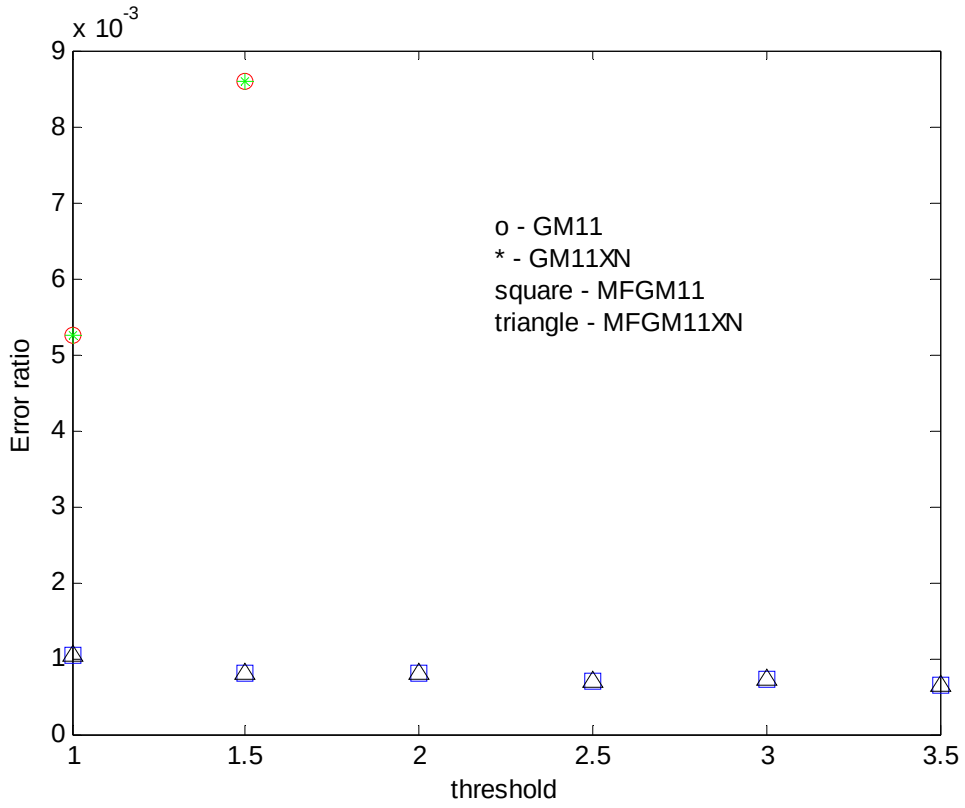

Figure 7.16 IXIC Threshold changes vs. Error ratio on different models

In the experiments to study the parameter sensitivity of the algorithm, the threshold values changes from 1.0 to 3.5 with an incremental of 0.5. Four models, GM11, GM11XN, MFGM11, and MFGM11XN, are used to mine structural changes. For the comparison of the window size effects, only the GM11 model is used in the study. The window size changes from 5, 10, to 15. (All other experiments use the same window size of 5.)

Figure 7.16 shows that when threshold increases, the ratio of errors des not change very much for the Fourier-corrected models such as MFGM11 and MFGM11XN. GM11 and GM11XN models have a higher ratio of errors when a higher threshold is used in the algorithm.

It clearly shows that the MFGM11 and MFGM11XN behave similarly when the threshold changes. The same situation occurs for GM11 and GM11XN.

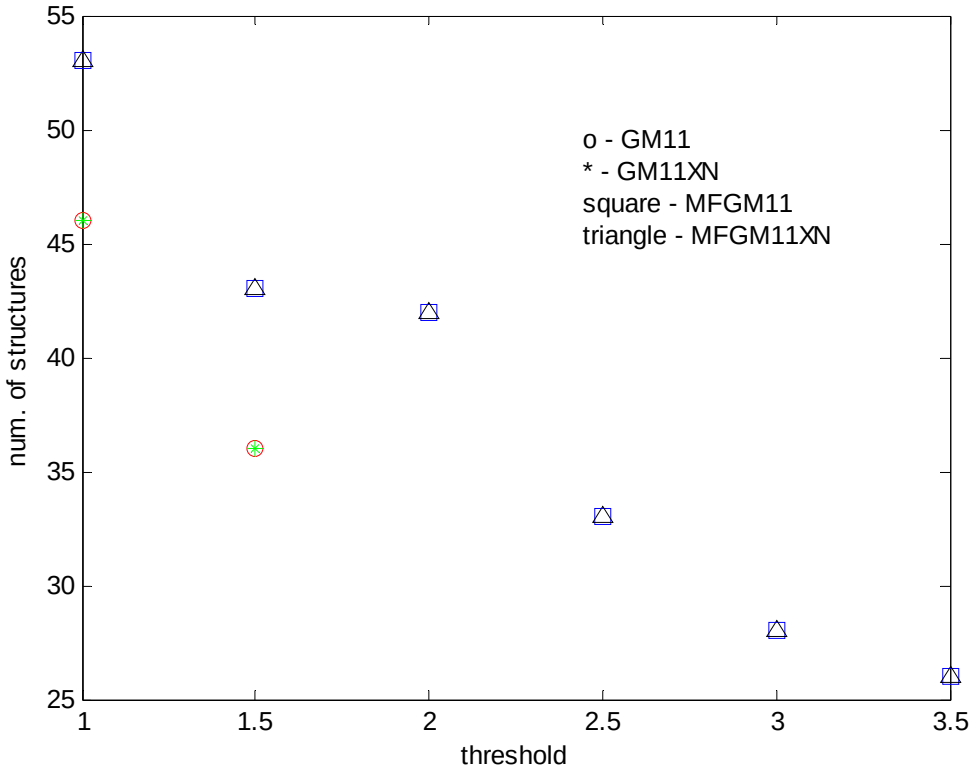

Figure 7.17 IXIC threshold changes vs. number of structures

Figure 7.17 illustrates how the number of structure changes increases as the threshold increases. For all four models used in the algorithm, increasing the threshold decreases the number of structures detected. That is because the higher threshold accommodates more variation and noise of data; therefore, fewer structures are detected. Similarly, MFGM11 and MFGM11XN behave closely in response to changes of threshold. This also applies to GM11 and GM11XN models.

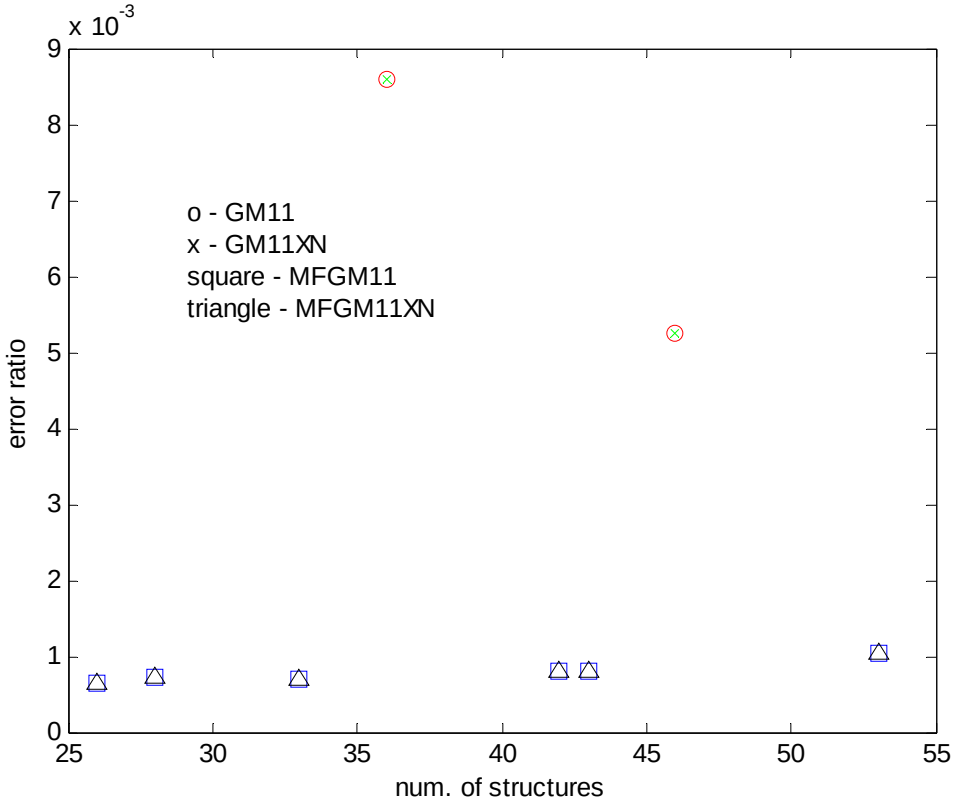

Figure 7.18 IXIC number of structures vs. error ratio on models against threshold

Figure 7.18 illustrates the comparison of algorithm on the dimension of error ratio and number of structures. The algorithm that uses MFGM11 and MFGM11XN models acts closely, so do the model GM11 and GM11XN. The error ratio of the algorithm that uses MFGM11 and MFGM11XN does not change significantly as the threshold changes; however, the number of structures discovered decreases.

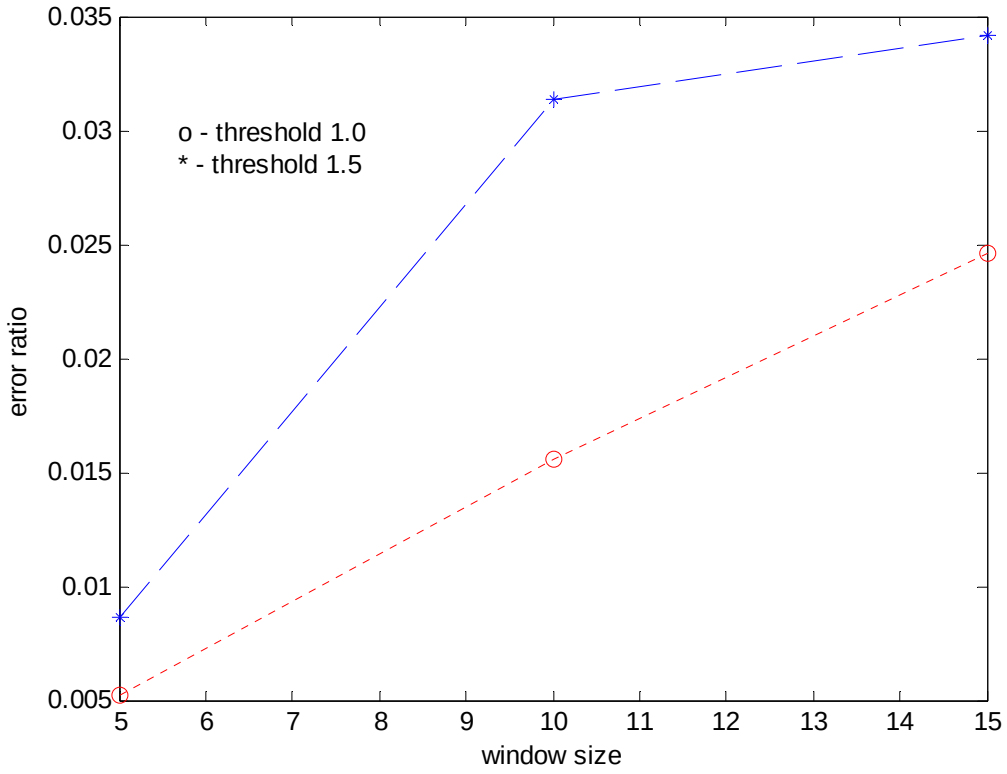

Figure 7.19 IXIC window size vs. error ratio

Figure 7.19 reveals how the window size changes affect the algorithm. In this experiment, the GM11 model is used. We can see that as window size increases, the error ratio increases also; however, the smaller threshold has a lower error ratio as the window size increases.

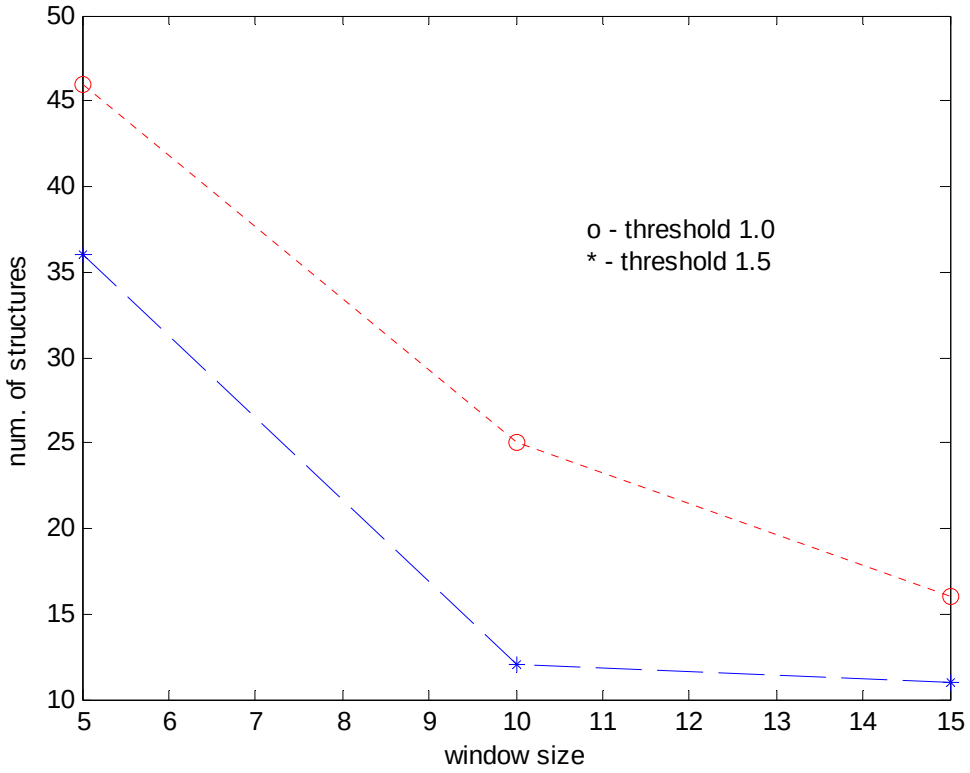

Figure 7.20 IXIC window size vs. number of structures

Figure 7.20 illustrates the number of structures discovered when window size changes from 5, 10, and 15. The larger the window size used in the algorithm to build the model, the smaller number of structures detected. This is because we use percentage change to measure the structure changes. Again, the larger the threshold used, the smaller the number of structures that could be discovered by the algorithm.

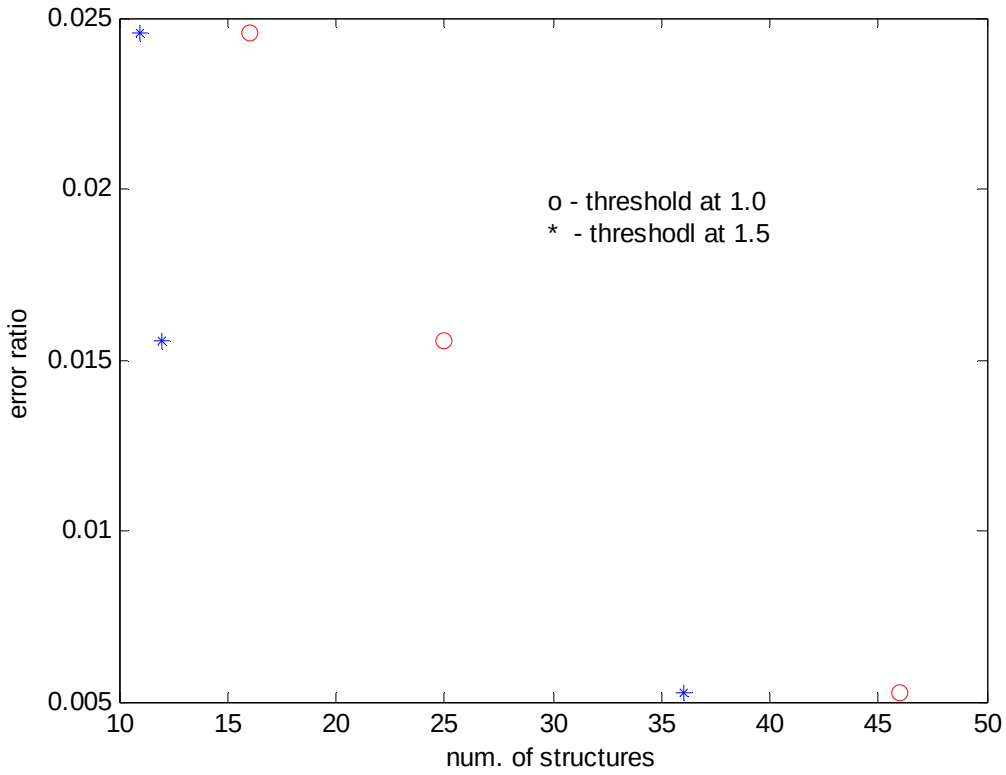

Figure 7.21 IXIC number of structures vs. error ratio against window size

Figure 7.21 manifests the comparison on the effects of different threshold and window size in the space of error ratio and number of structures. Bigger window size has a higher error ratio and moves the result towards region I when threshold 1.0 is used. It is also true for the threshold of 1.5. Therefore, consider the case of smaller error ratio, in which the smaller threshold tends to have a greater number of structures discovered.

DJI Forecast Precision Study

In this forecast precision experiment, the Dow Jones Industrial Index from Jan 2000 to Dec. 2004 data is used. In order to visually compare and identify the structure changes mined by the algorithm, the partial 300 data series are depicted in figure 7.22. The structure mining algorithm uses a threshold of 1.0, window size of 5, and the GM11XN model. We can see the famous 9.11 event is identified as successfully with these parameters as the IXIC data set does. Again, the structures identified are close to human justification. More quantitative explanation of the forecast precision are performed and illustrated in figures 7.23 through 7.32. In order to express the relationship more clearly, in this study, only the GM11XN model is depicted in the graphs below.

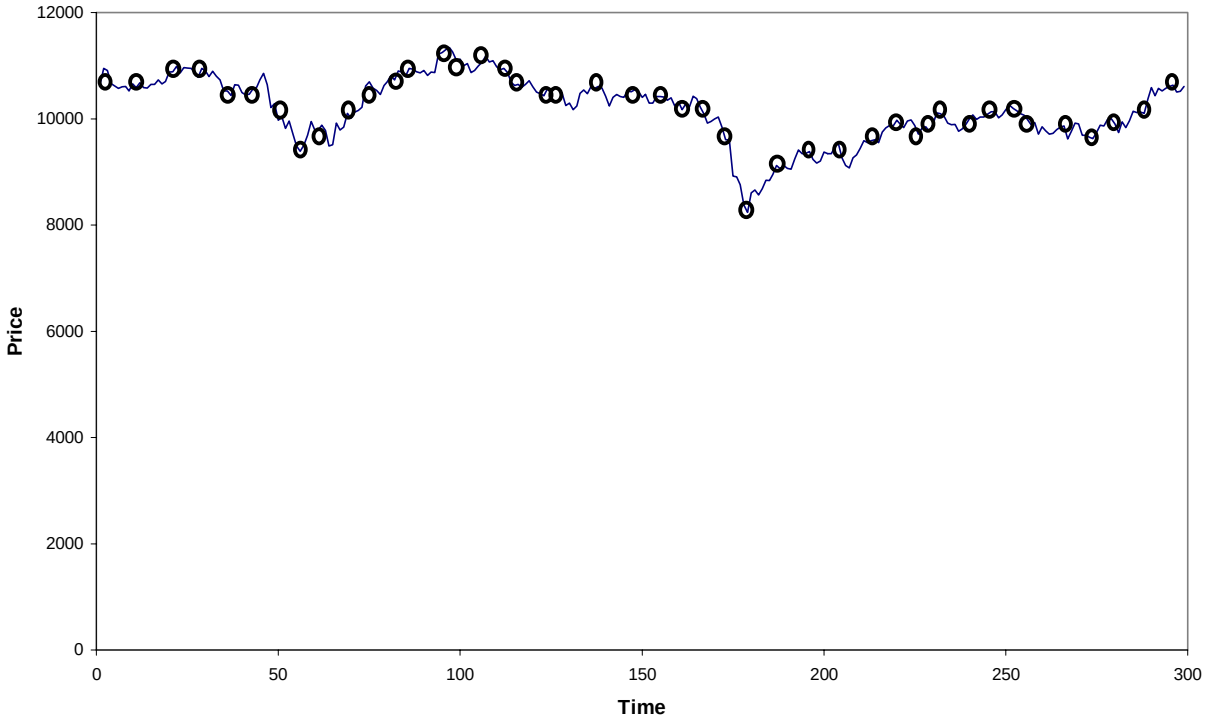

Figure 7.22 DJI structures discovered with threshold 1.0 and window size 5.

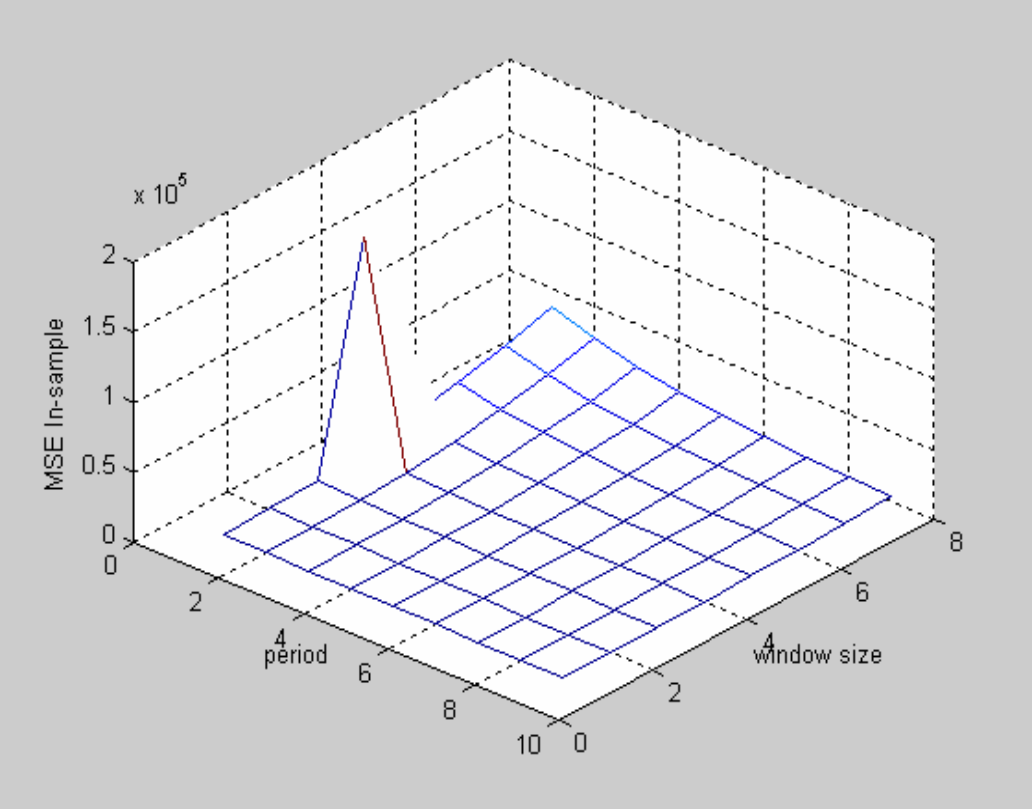

Figure 7.23 DJI MSE in-sample changes

Figure 7.23 shows the change of MSE and its relationship to the periods that filter noise and the window size that is used to build model. Like what we have seen in the case of IXIC data, a higher period removes more noises from data and therefore improves the in-sample MSE. The smaller the window size used to build the model, the smaller in-sample MSE is, except that in this DJI data, an exception, which I don't know how to explain, occurs.

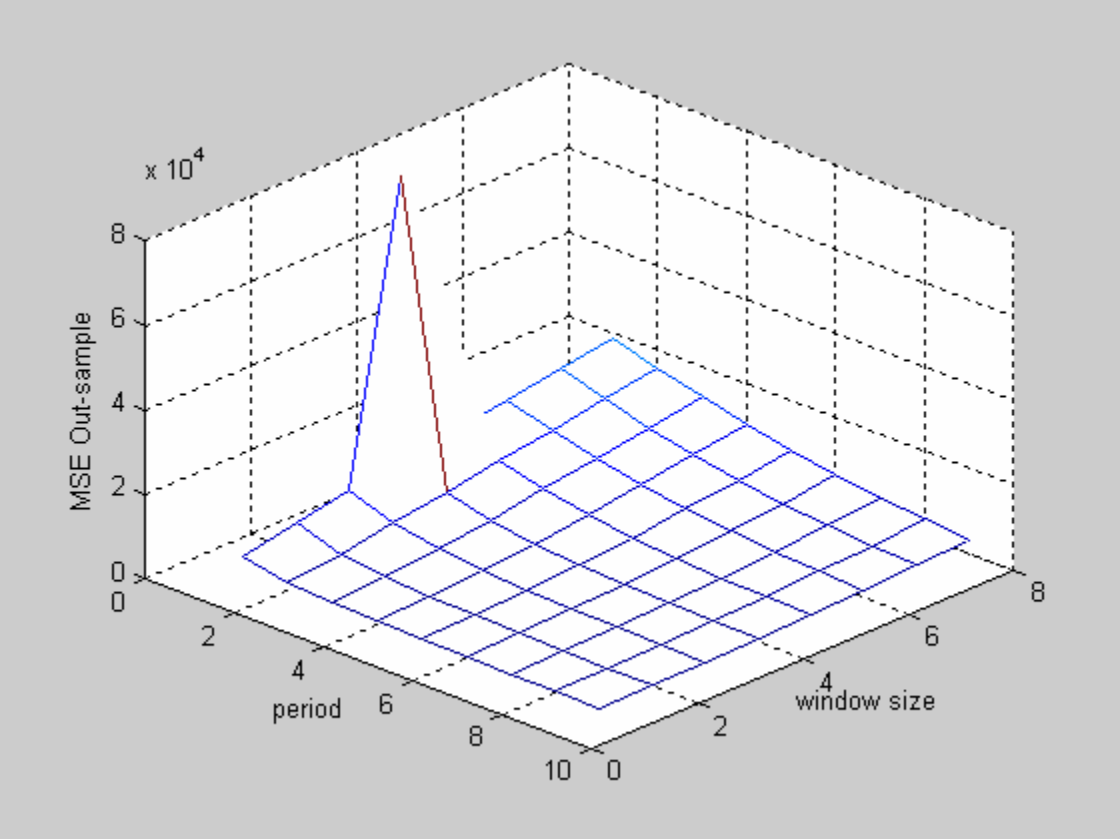

Figure 7.24 DJI (01-04) MSE Out-sample

Figure 7.24 show the out-sample MSE changes and the relationship to changes of periods and window size. As in the in-sample MSE case, the higher a period is, the smaller the out-sample MSE is; and the higher window size is, the higher the out-sample MSE will be.

From discussion of in-sample and out-sample MSE changes, we can see that when we use our model to predict, a smaller window size and some degree of noise reduction could improve our model precision.

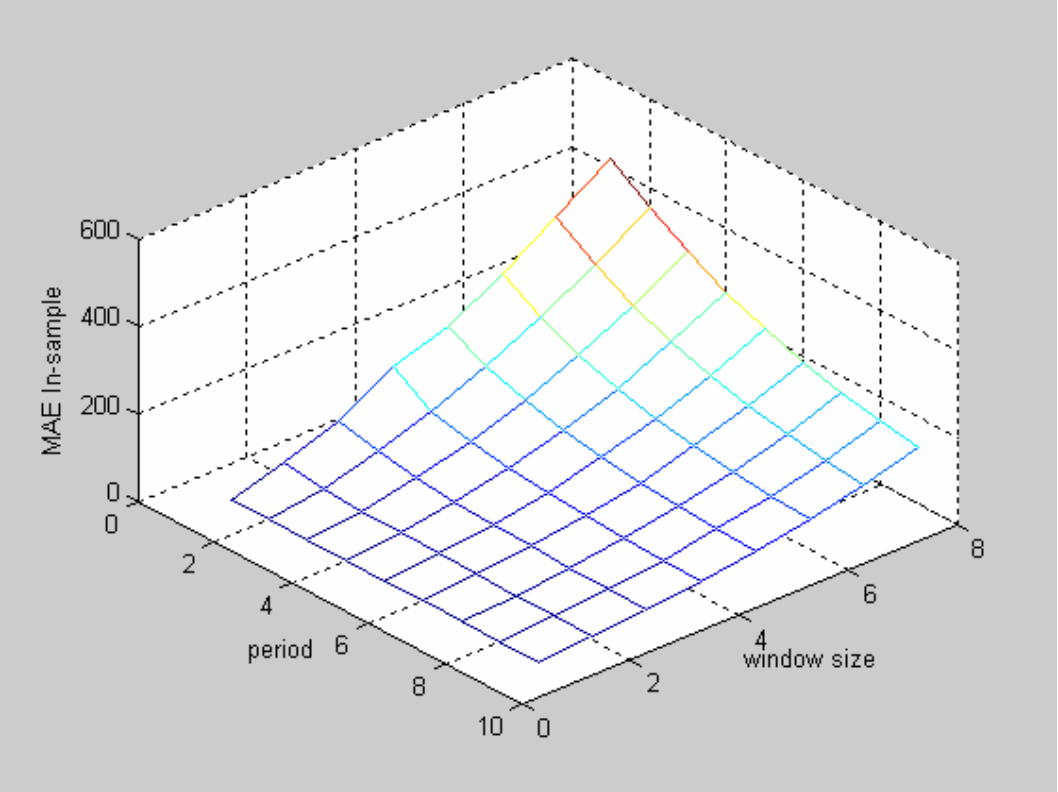

Figure 7.25 DJI MAE In-sample

Figure 7.25 presents how the in-sample MAE changes with different periods and window sizes. The in-sample becomes highest when the largest window size and the smallest period are used in the algorithm; conversely, the in-sample becomes smallest when the smallest window size and the smallest period are used in the algorithm. The increasing of window size also increases the in-sample MAE, and on the contrary, The increasing of period decreases the in-sample MAE, and on the contrary, the increasing of period decreases the in-sample MAE.

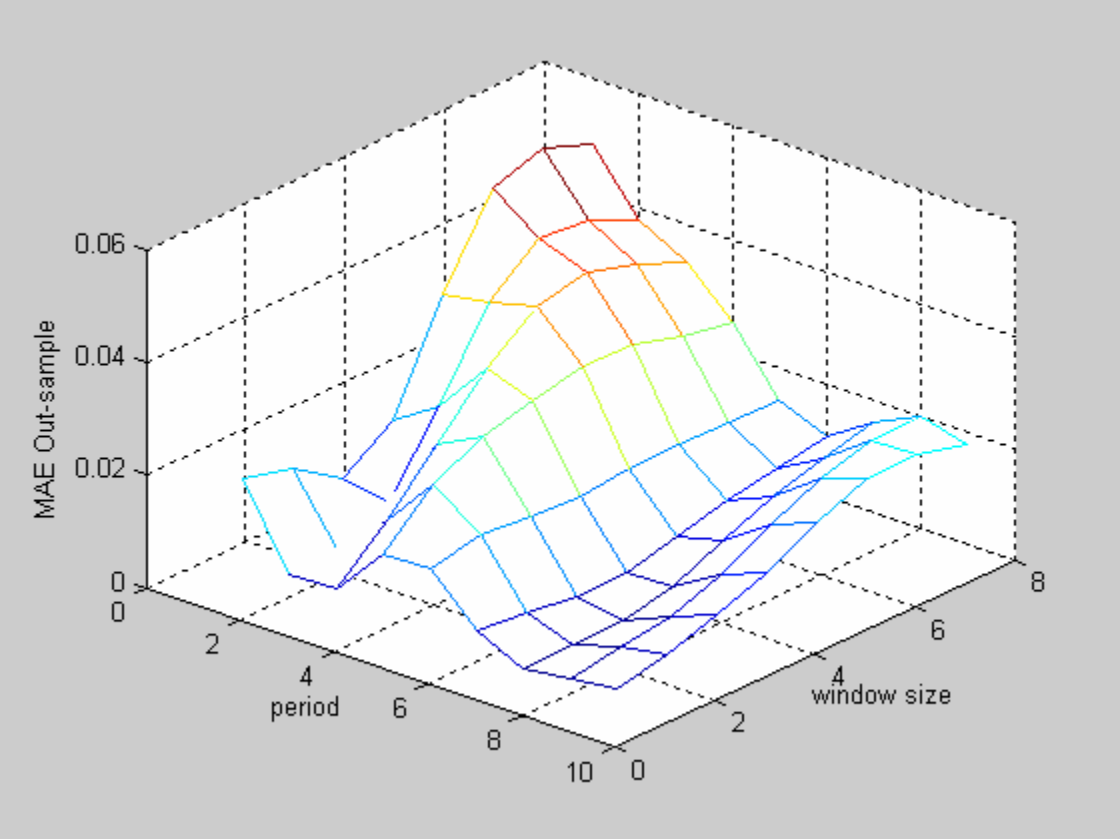

Figure 7.26 DJI MAE Out-sample

Figure 7.26 demonstrates how out-sample MAE changes with different periods and window sizes. It is clear that a higher window size increases out-sample MAE value. In general, a higher period minimizes out-sample somehow, but it is not always considered in our experiment. By comparing the in-sample and out-sample MAE, we can see that out-sample MAE values are much smaller than these of in-samples, as shown in figure 7.27.

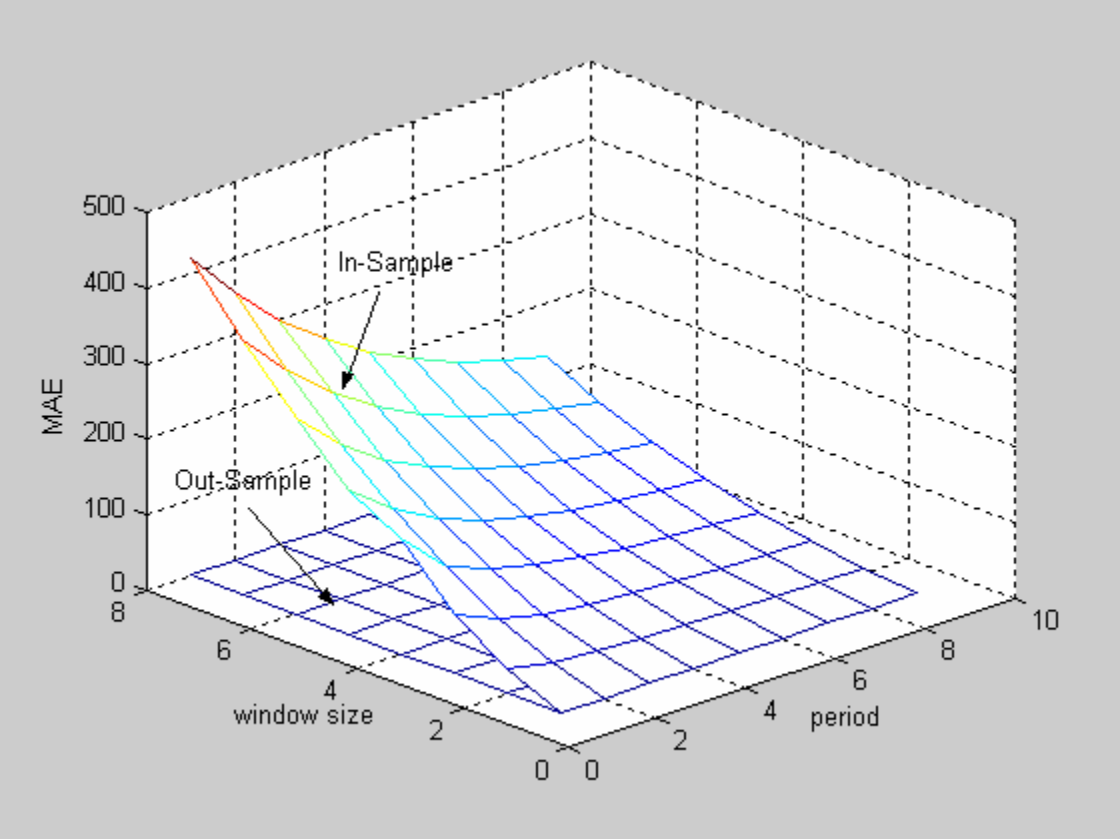

Figure 7.27 DJI MAE out-sample/ in-sample comparison

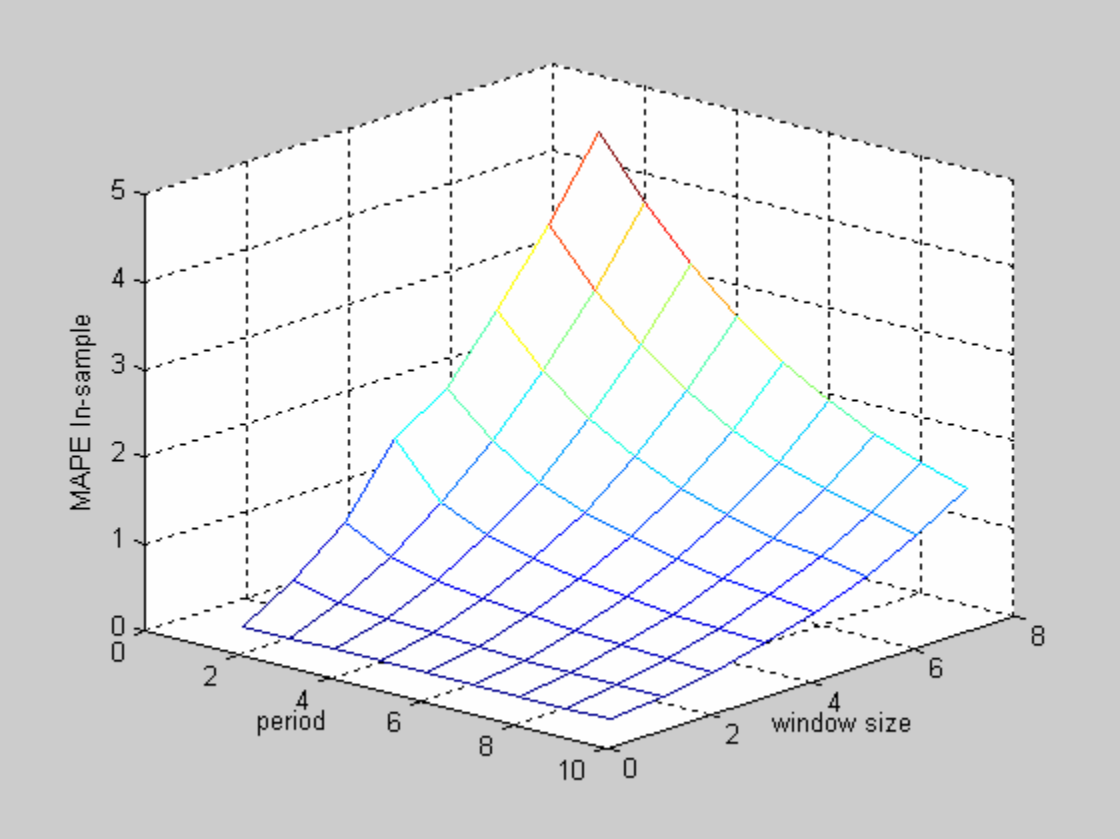

Figure 7.28 DJI MAPE In-sample changes

Figure 7.28 exhibits the in-sample MAPE changes with different periods and window sizes. Overall, the model is categorized as high-precision regarding the in-sample data because the in-sample MAPE is less than ten. A higher window size creates a higher in-sample MAPE, and a higher period produces a smaller in-sample MAPE.

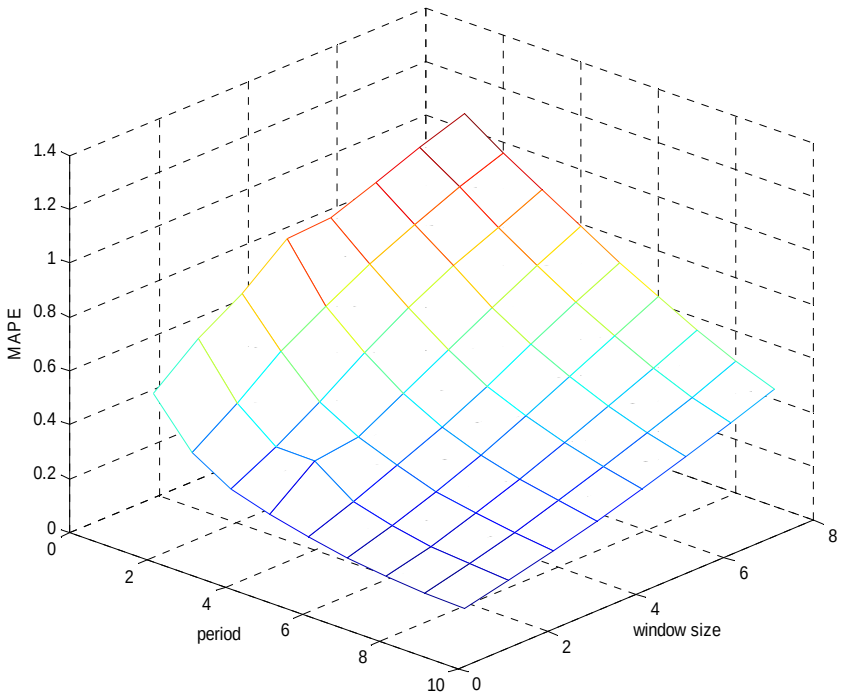

Figure 7.29 DJI MAPE Out-sample

Figure 7.29 introduces the out-sample MAPE changes with the periods and window sizes. The out-sample MAPE is highest at the largest window size and the smallest period. A higher period will generate a smaller out-sample MAPE; while a bigger window size produces a higher out-sample MAPE.

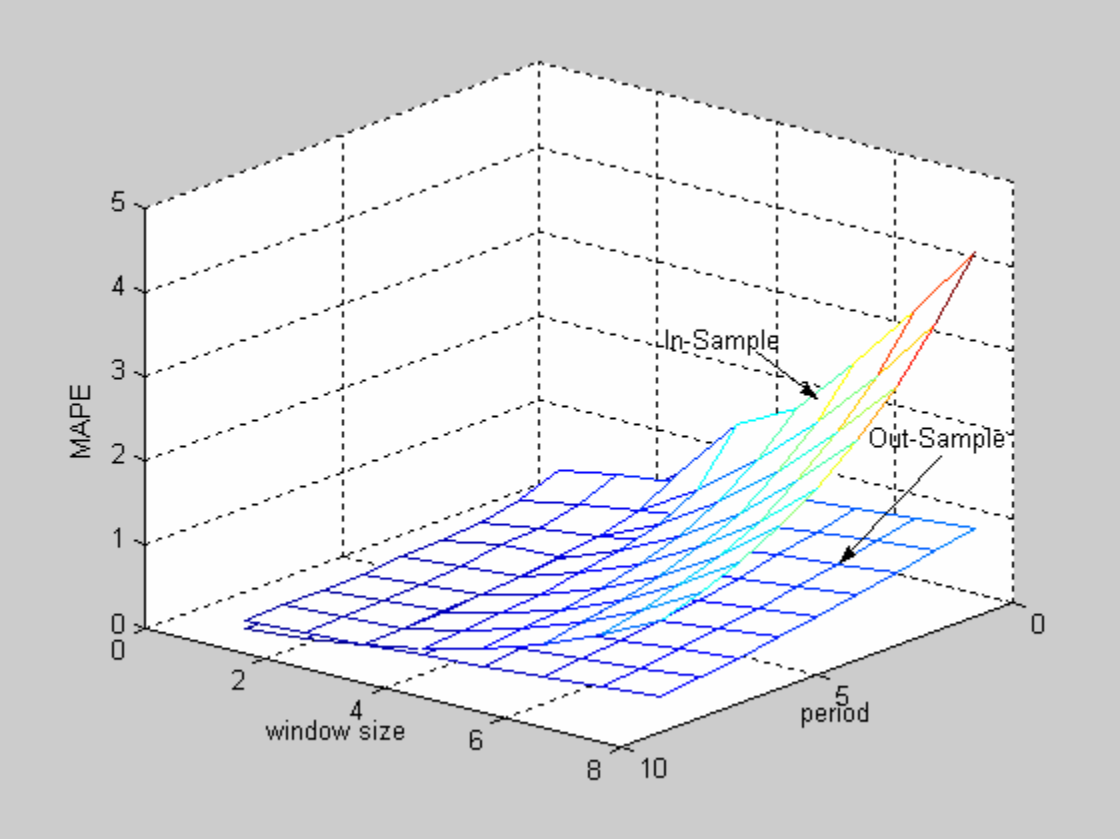

Figure 7.30 DJI MAPE compare

Figure 7.30 compares the in-sample and out-sample MAPE changes with the periods and window sizes. The in-sample MAPE increases much faster than out-sample MAPE as the window size increases and the period decreases. Overall, the out-sample MAPE is smaller than that of the in-sample MAPE. However, based on the criteria, the model can be categorized as a high-precision model both at in-sample and out-sample levels.

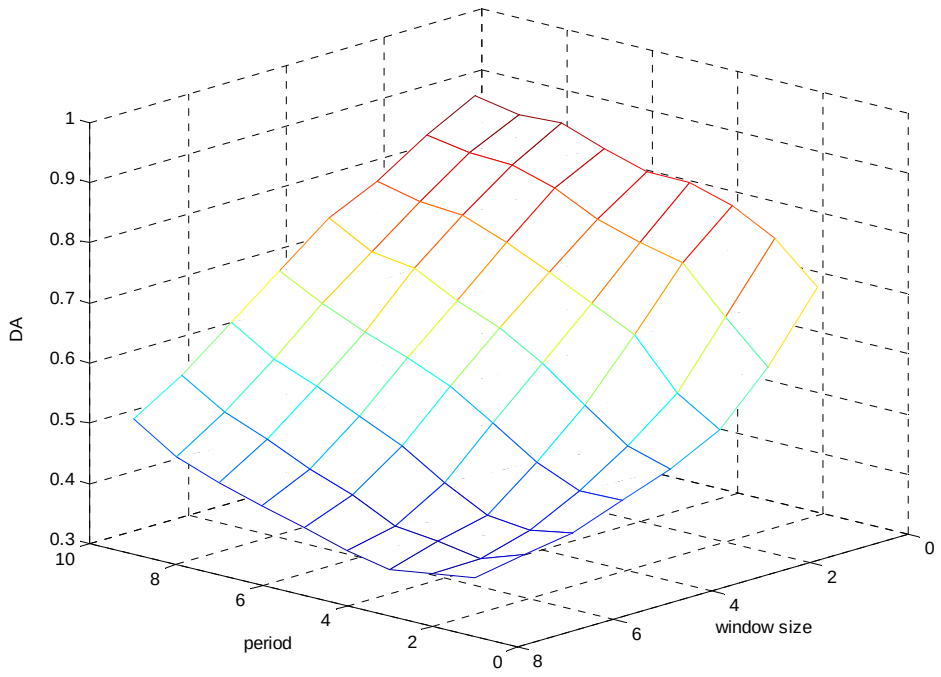

Figure 7.31 DJI DA changes against period and window size

Figure 7.31 presents the DA changes v.s. the change of period and window size. DA reaches a highest point when the biggest period and the smallest window size used in the experiment, and reaches lowest with the reverse parameters. A bigger the period is , a better DA is, since filtering out the noise from the data stream makes the model much more effective at catching the true movement. Similarly, the larger the window size used, the lower the DA is. Similar to what we have discovered before, data has a short-term memory, which means that the more data are involved in predicting the short-term future, the less the accuracy level will be.

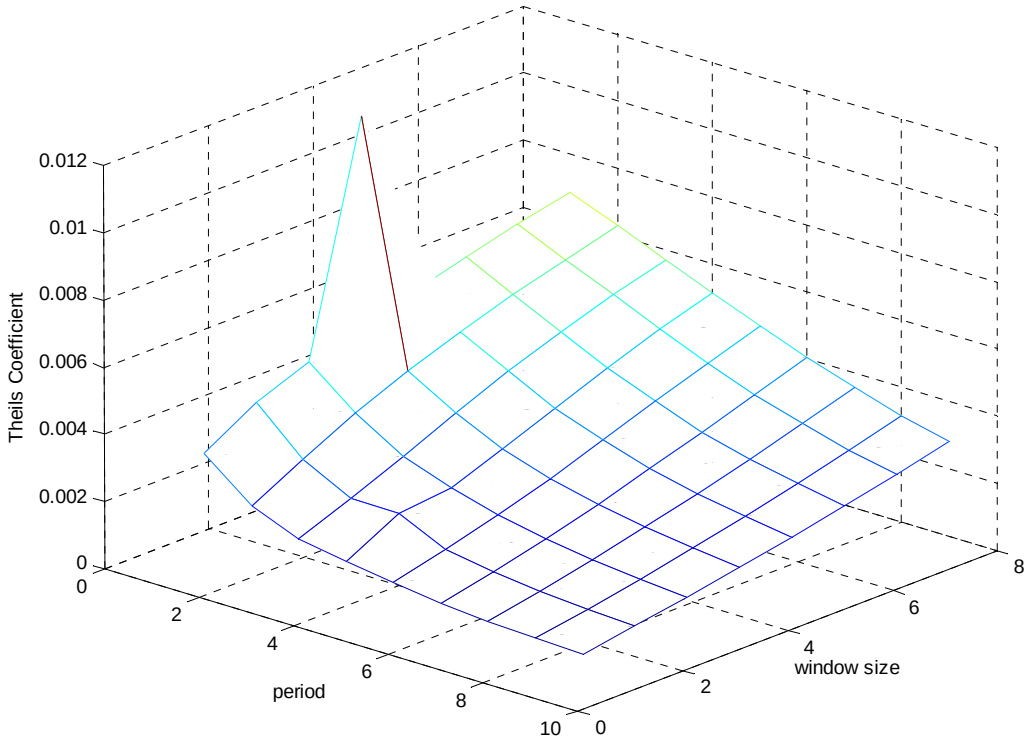

Figure 7.32 DJI Theil's Coefficient

Figure 7.32 shows the comparison between our model and the random walk model based on the Theil's inequality coefficient. It can be concluded that our model outperforms the random walk model because all of Theil's inequality coefficients are less than one. The lower the degree removal of the noise (i.e., a higher period used), the higher Theil's inequality coefficient is; and the larger the window size is, the higher the Theil's inequality coefficient is also. Additionally, there is an outlier which I can't explain in this figure.

SP500 Forecast Precision Study

The last experiment with the real world time series is the Standard & Poor 500 (SP500) index. In this forecast precision experiment, the SP500 from Jan 2000 to Dec. 2004 data is used. In order to visually compare and identify the structure changes mined by the algorithm, the partial 300 data series are depicted in figure 7.33. The structure mining algorithm uses threshold of 1.0, window size of 5, and GM11XN model. We can see that the famous 9/11 event is identified successfully with these parameters much as the IXIC data set and DJI do. Again, the structures identified are close to human justification. More quantitative explanation of the forecast precision are performed and illustrated from figure 7.34 to figure 7.39. All four models are used in this experiment. For the clarity of the figures, only relationships that show significant differences are illustrated.

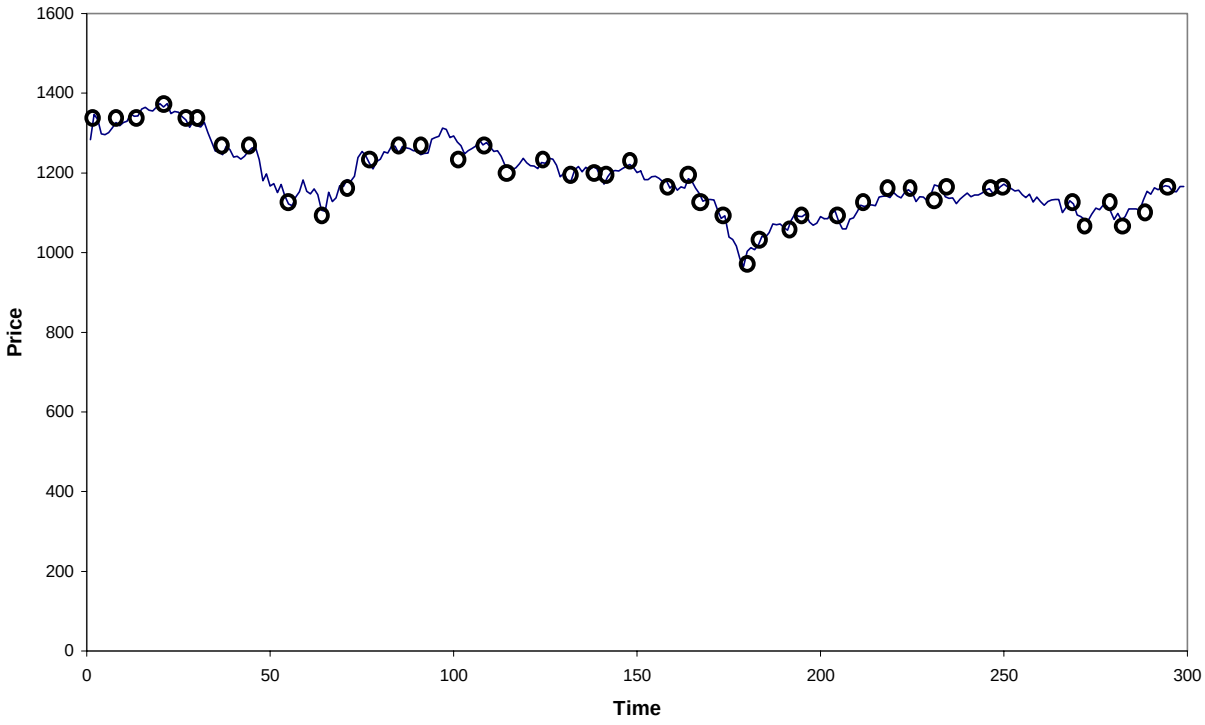

Figure 7.33. SP500 structural mining result with threshold of 1.0 and window size of 5

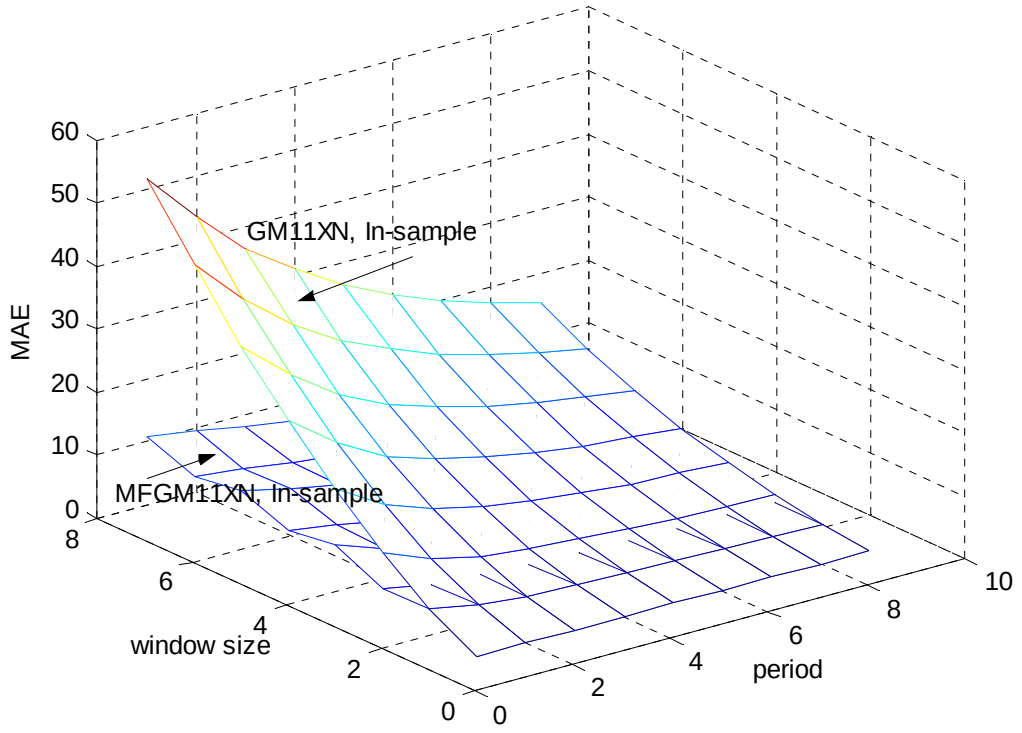

Figure 7.34 SP500 MAE changes v.s. period and window size against Models (In- Sample)

Figure 7.34 shows the in-sample MAE changes v.s. the different periods, window size, and models. The Fourier-corrected models have much smaller in-sample MAE that that of non-Fourier corrected models. A bigger window size produces a higher in-sample MAE; and a smaller period also generates a higher in-sample MAE.

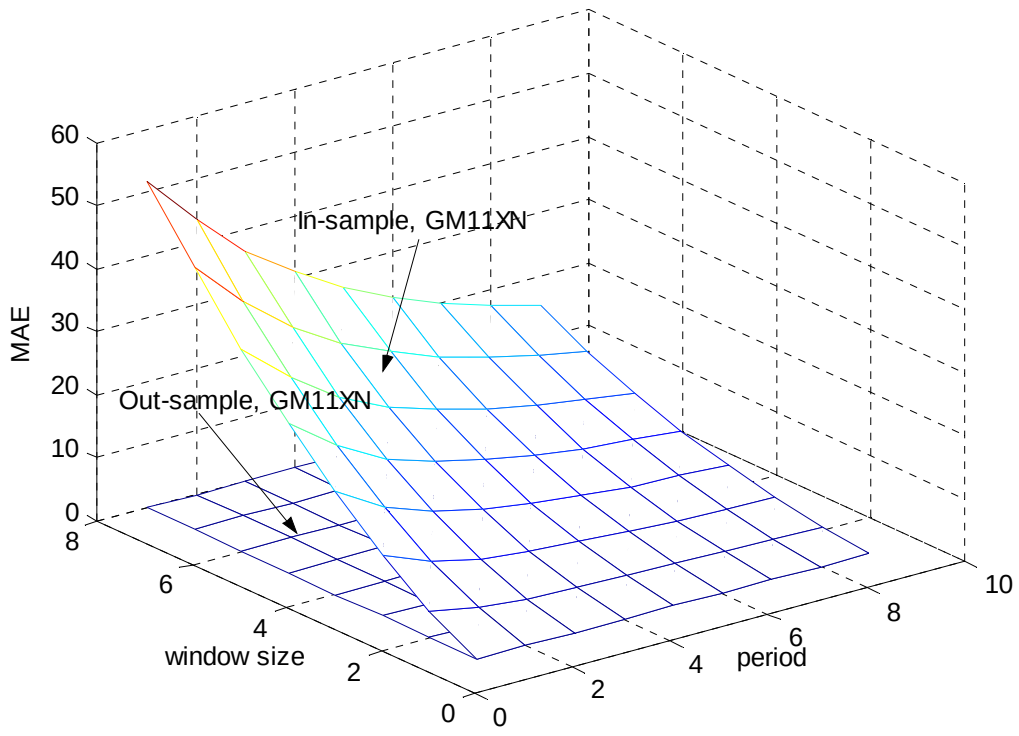

Figure 7.35 SP500 MAE changes v.s. period and window size (in- and out-sample comparison)

Figure 7.35 presents in-sample and out-sample MAE comparison results against the same model on changes of period and window size. The out-sample MAEs are much smaller than these of in-sample.

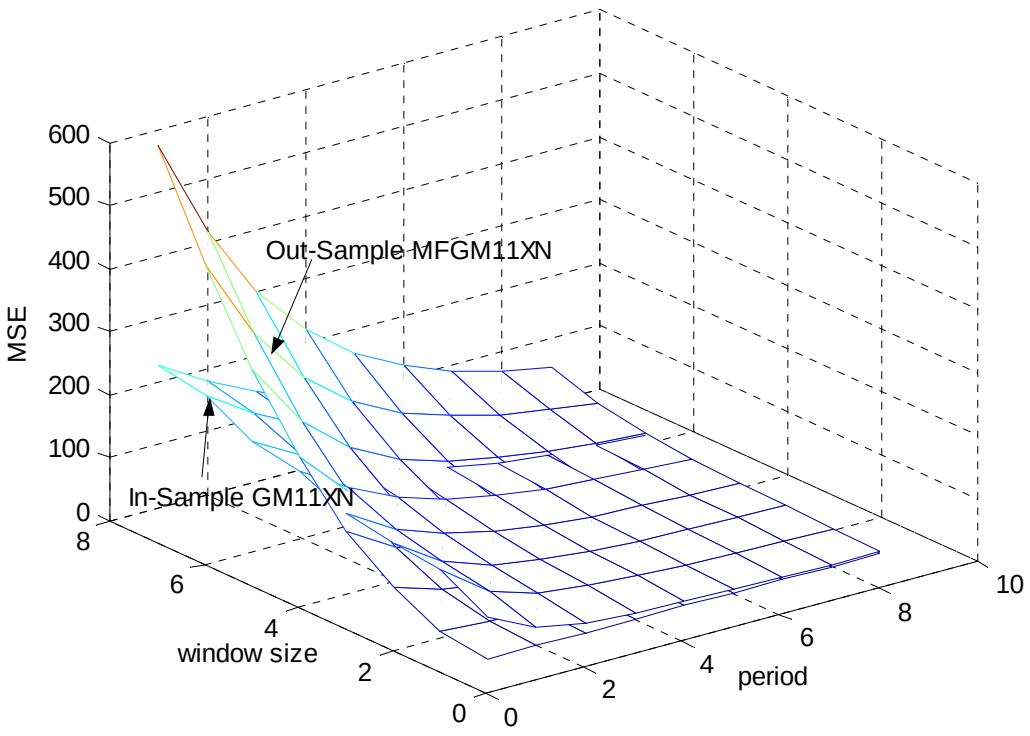

Figure 7.36 SP500 MSE changes vs. period and window size

Figure 7.36 shows that a higher window size is, the larger MSE is in both in-sample and out-sample cases. It is the same for the change of out-sample MSE corresponding to period and window size. The out-sample MSEs are generally higher than these in-samples even though the Fourier correction does not help to reduce MSE at out-sample level.

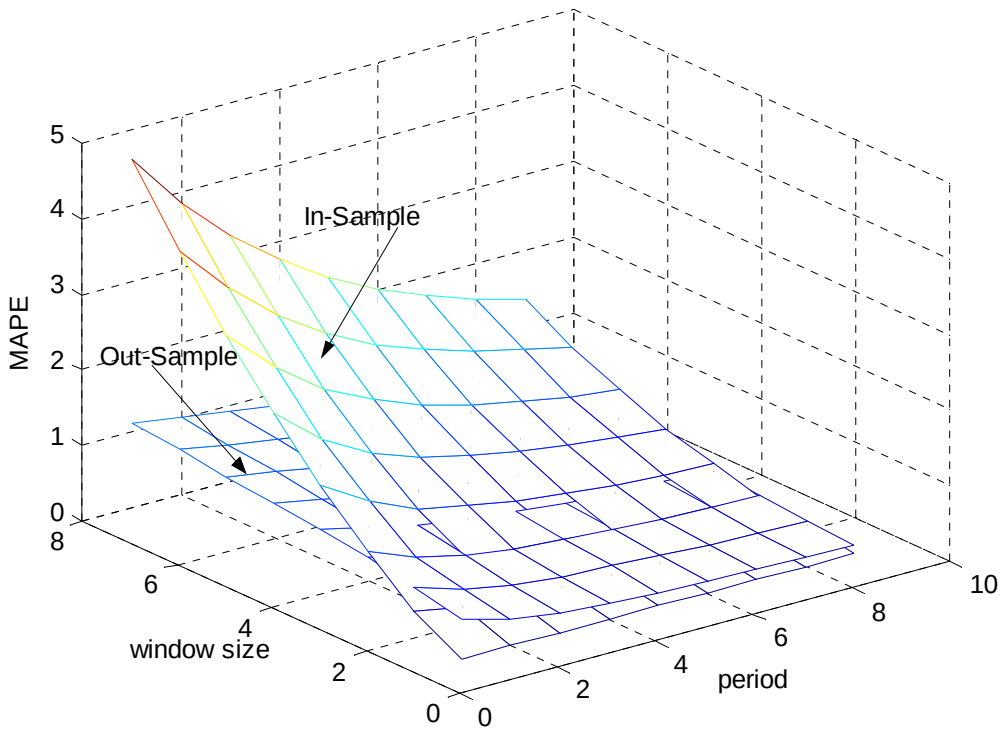

Figure 7.37 SP500 MAPE changes vs. period and window size

Figure 7.37 proves that the models are highly precise in the range of parameters chosen for both in-sample and out-sample prediction. This is because MAPE values are all less than ten. The out-sample MAPE values here are much smaller than the in-sample values. Similarly, the higher a window size is, the larger the MAPE values are for both in-samples and out-samples , and the larger the period is, the smaller the MAPE values are.

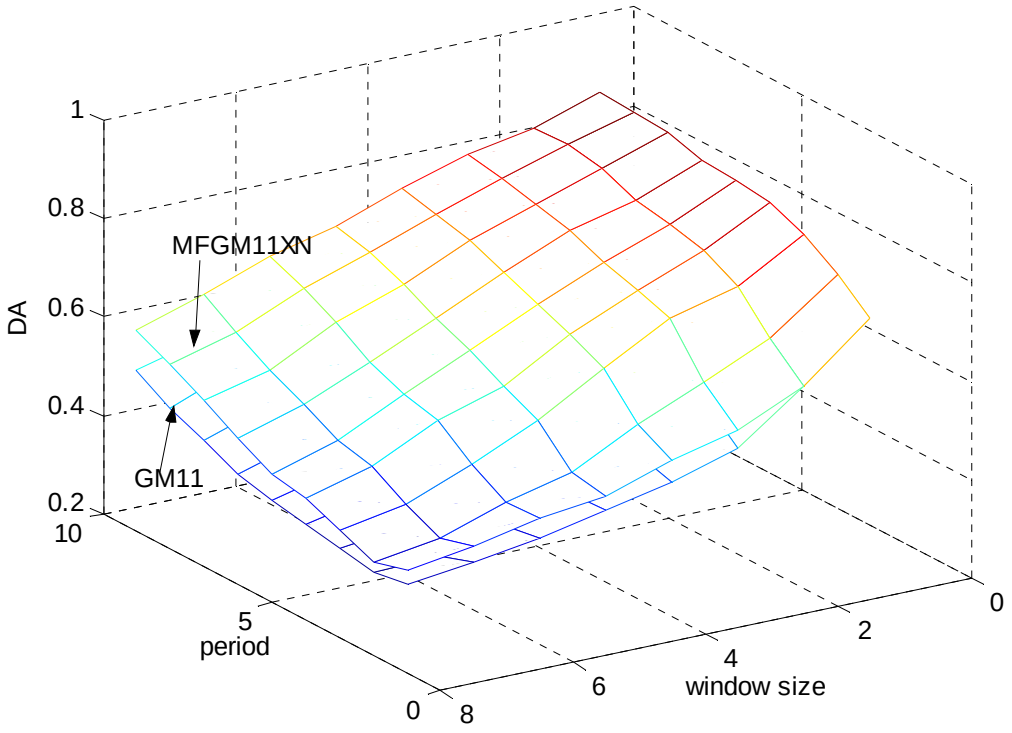

Figure 7.38 SP500 DA changes vs. period and window size on different model

Figure 7.38 illustrates how the DA changes with corresponding periods and window size used in the experiment. More noises are filtered; i.e, the bigger the period, the higher the DA values are; and the less historical data used to build the model, the higher the DA values are. The Fourier correction helps to improve the directional predictability of the model.

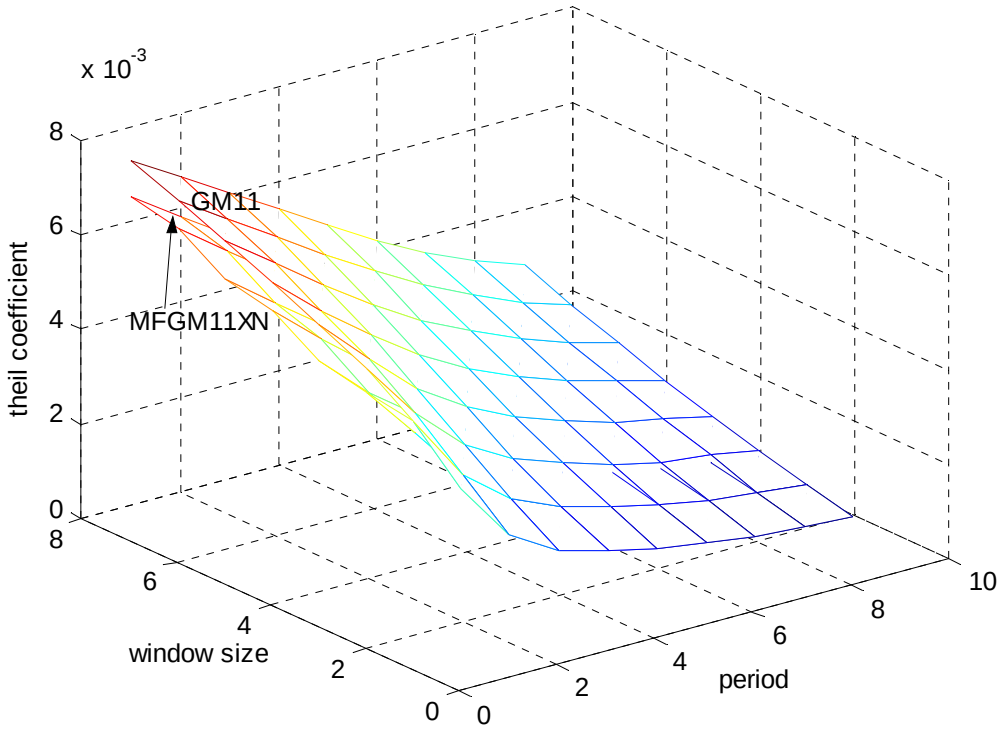

Figure 7.39 SP500 Theil's Coefficient change vs. period, window size, and models

Figure 7.39 proves that all our models outperform the random walk model. Fourier correction of errors helps to improve the performance of models even further. A larger window size degrades the performance of a model, while a bigger period boots the model's performance.

Cluster and Similarity Study

Eight stocks such as EOG, CDIS, SCOX, NOVL, SM, HAL, WMT and FDG are picked from April 2005 to October 2005 for similarity matching based on event history. The results of structure changes mining algorithm are shown in figures 7.40 and 7.43 separately. The normalized data are depicted in figure 7.41 and 7.43. Based on event-driven similarity matching, two clusters are obtained. One cluster group A composes of EOG, SCOX, and CDIS that contains two identified events. The other cluster, group B, consists NOVL, SM and HAL that has three discovered events. The similarity distance measure of each group is summarized in increasing order among items in table 7.1 and 7.2. Our similarity measure on group A revealed same results as our visual perception, i.e. EOG and CDIS are most closed to each other than any other pairs in the group. It is further confirm by the normalized data plot in figure 7.41, even though at this time, it is a little bit harder for human to decide which pair of these three are closest to each other. In the raw data plot of group B, it looks like that SM is close to both NOVL and HAL. However, the normalized data plot in figure 7.43 revealed immediately that SM and HAL are much close to each other. Our newly developed similarity distance measure disclosed same results summarized in table 7.2.

It can be concluded that our event-driven similarity matching approach works very well with greatly reduced dimensionality of time series data. In our example here, the reduction rate of dimensionality is approximately 95%. Our experiments with other data sets tells us that the reduction of dimensionality is generally around 90%.

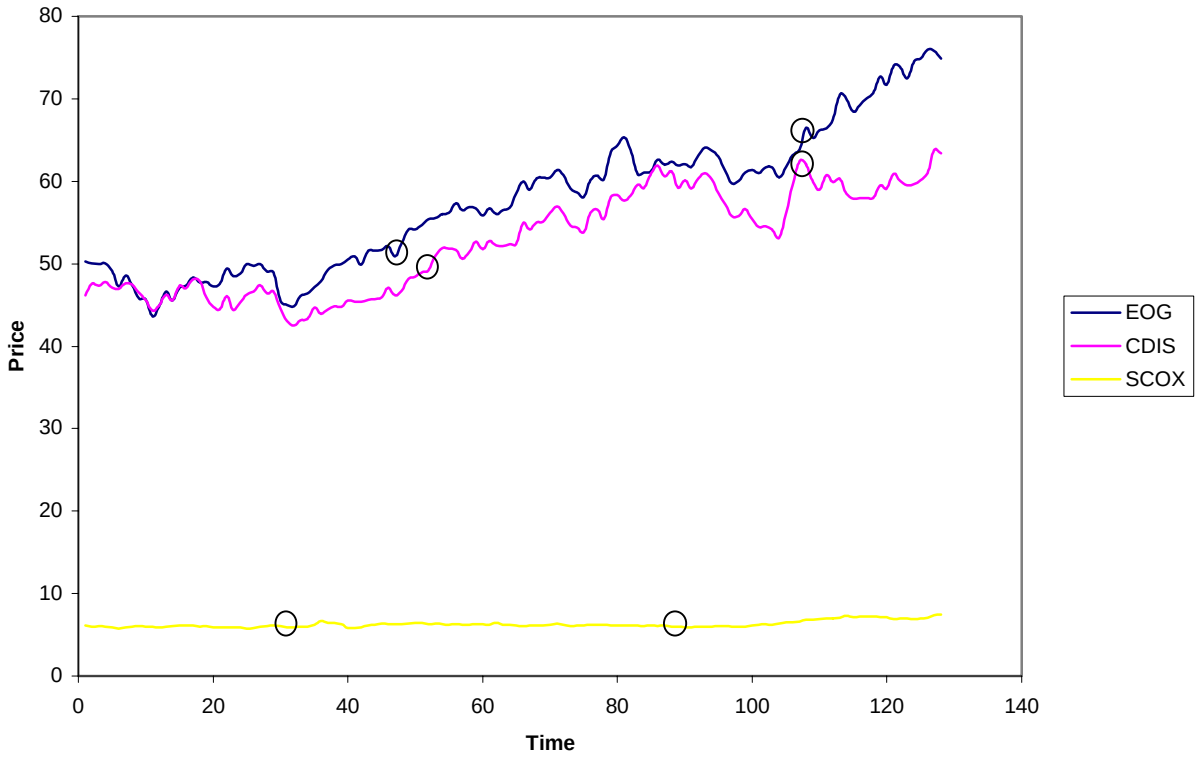

Figure 7.40. Structural change of stock EOG, CDIS, SCOX from April 2005 to Oct. 2005

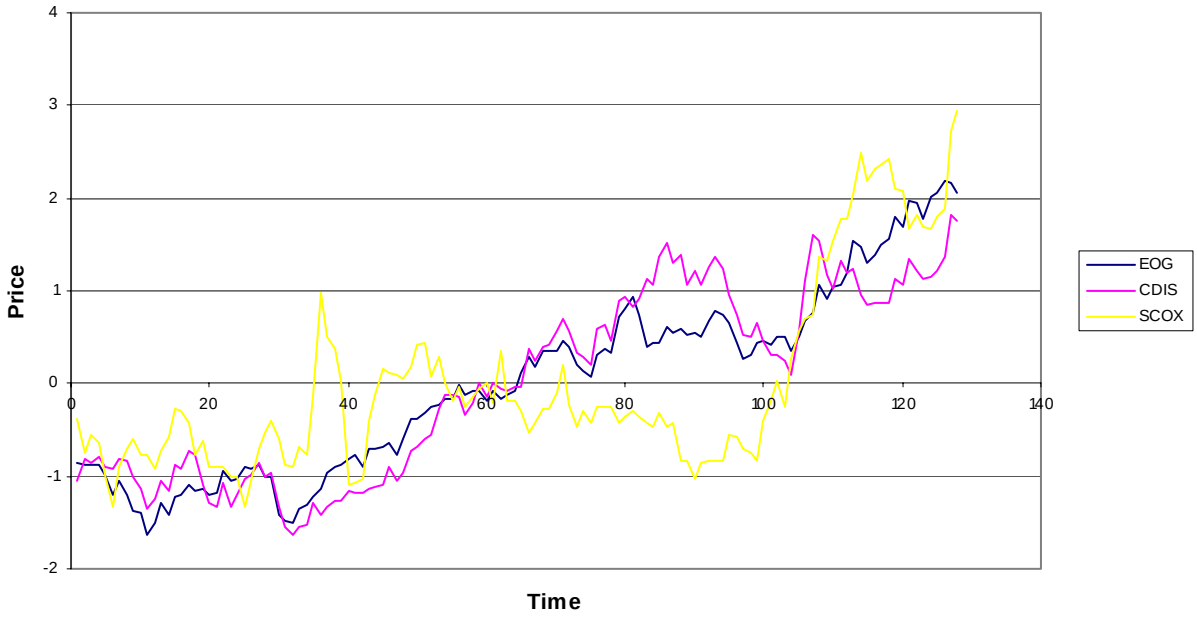

Figure 7.41. Normalized stock price of EOG, CDIS, and SCOX

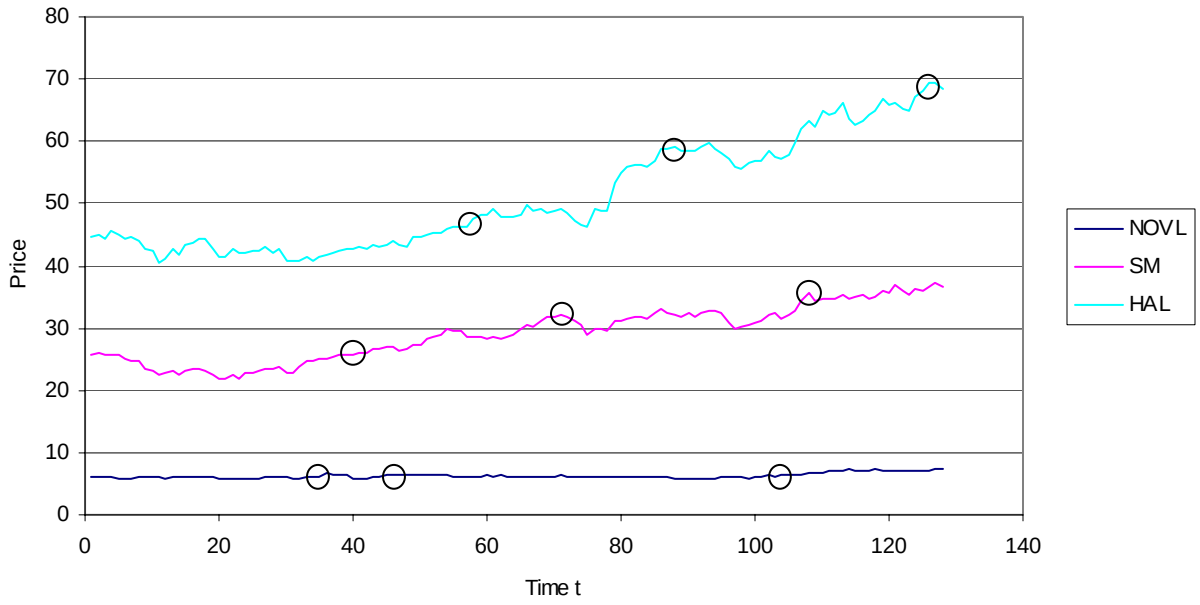

Figure 7.42. Structural changes of stock NOVL, SM, and HAL from April 2005 to Oct. 2005

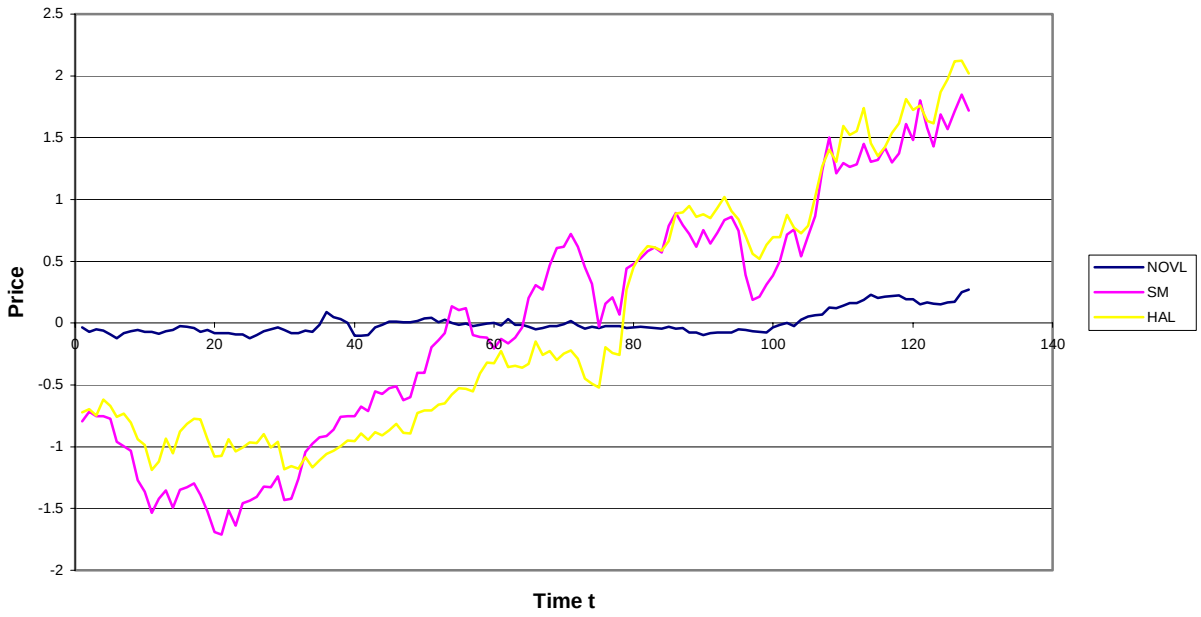

Figure 7.43. Normalized stock price of NOVL, SM, and HAL from April 2005 to Oct. 2005

Table 7.1. Similarity distance among EOG, CDIS, and SCOX

	EOG-CDIS	EOG-SCOX	SCOX-CDIS
Distance	9.9715	30.7219	31.6819

Table 7.2, Similarity distance among NOVL, SM, and HAL

	SM-HAL	NOVL-SM	NOV-HAL
Distance	26.669	29.792	42.296

Structure Changes Mining of Network Traffic

The predictability of network traffic is of significant interest in many applications, such as dynamic capacity planning, predictive congestion control, network management etc. The initial study toward long-term forecasting of IP network traffic is performed by Groschwitz[147]. The authors compute a single value for the aggregate number of bytes follow over the NSFNET, and model it using linear time series models. The results show that the time series obtained can be accurately modeled by a low-order ARIMA model, offering highly accurate forecast (within 10% of the actual behavior) for up to two years in the future. Other work in the area of Internet traffic forecasting typically addresses small time scale, such as seconds or minutes, that are relevant to dynamic resource allocation [148, 149]. Qiao [150] show that one-step ahead prediction at a coarse resolution is a prediction of the average behavior over a long interval using a wide range of linear and nonlinear time series models. Papagiannaki [151] introduced a methodology to predict when and where link additions/upgrades have to take place in an IP backbone network. The aggregate demand and its evolution at time scales larger than one hour were studied. The results show that IP backbone traffic exhibits visible long- term trend, strong periodicities, and variability at multiple time scales. Xue, et. al. [152] proposed network traffic forecast based on a seasonal neural network model. Their research indicates that the seasonal neural network prediction model produces higher accuracy than the seasonal ARIMA model and common time series of neural network on monitoring network traffic. Cheng, et. al., [153] compared the Autoregressive (AR) model and Window Mean (WM) model in prediction of session throughput of constant bit rate streams in wireless data networks. They show that AR and WM models have similar performance in prediction and concluded that session throughput

in wireless data networks can be modeled and predicted to a useful degree from past values by using linear time series analysis.

From the research we have done, we find little research that focuses on the internal changes or temporal patterns of network traffic. In our project, we conducted research on internal structure changes of network traffic and predict the changes when it happens. In this research, the UC Berkeley Home IP Web trace data is used in the study. The Home IP service is offered by UC Berkeley to its students, faculty, and staff. The Home IP provides dial-up PPP/SLIP IP connectivity using 2.4kb/s, 9.6kb/s, 14.4 kb/s, or 28.8 kb/s wireline models or 20-30 wireless modems. Only traffic destined for port 80 was traced and all non-HTTP protocols and HTTP connections for other ports were excluded from these traces. The trace file that generated from Fri Nov 1 15:18:59 1996 through Wed Nov 6 12:46:59 1996 that contains 2,599,049 requests and 6037 unique clients seen is used in our experiments. The raw trace file was parsed into database, and data are cleaned before they are used in our experiments. The final hourly network traffic data that was processed with our structure changes mining algorithm is shown in figure 7.44. The structure changes are marked with 'O'. The GMXN(1,1) model is used for one-step ahead network traffic forecasting, the results are show in table 7.3

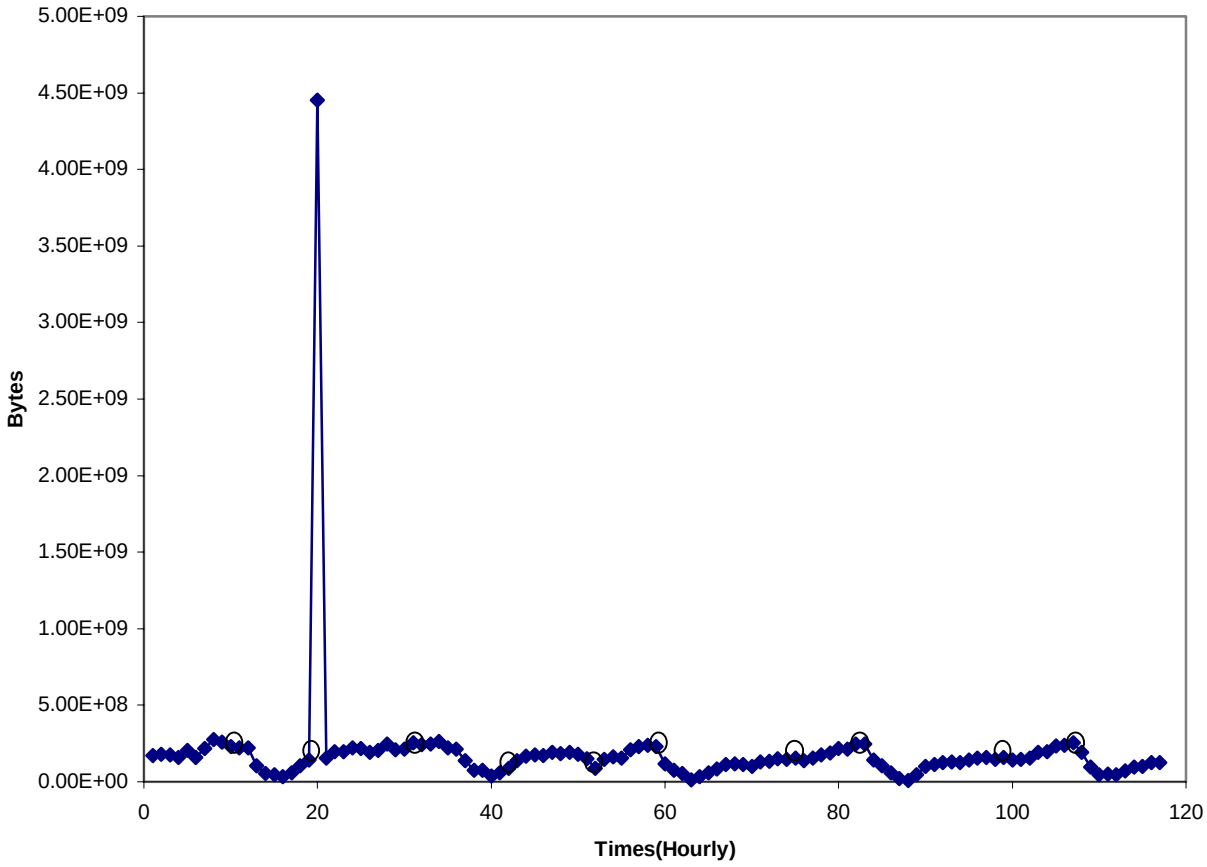

Figure 7.44. Structural changes of UC Berkeley Home IP hourly network traffic

Table 7.3. Network traffic forecasting with GMXN(1,1) model

	MSE	MAE
In-Sample	6.9506970942655955E19	5.733702015674582E8
Out-Sample	5.2068541177883208E18	3.0671614187936205E8
DA	0.53153	
Theil's Coeff.	1.42177246	

From our results, we can see that our structure changes identification algorithm successfully identified the network traffic burst at data point 20. The direction accuracy is slightly better than 50%, and the forecast did not outperform the random walk model in our research. However, our structure-changes mining algorithm identified almost all of the perceptually important turning of the network traffic.

CHAPTER 8

CONCLUSIONS AND FUTURE EFFORTS

Through clear definition of structural changes in time series, this dissertation has made an original and fundamental contribution to the field of time series analysis and data mining. Chapters 4 and 5 developed time series data mining concepts and a framework for white hidden structure of the generating operation of time series using a dynamic gray model. The later chapters shows that the framework developed based on dynamic system successfully characterizes and discovers complex structures, nonperiodic, irregular, and chaotic time series even with the presence of noise. The framework was then successfully applied to discover complex, nonstationary, chaotic time series from both financial and computer science domains. All of our four models are high-precision models based on MAPE at in-sample and out-sample; and all of them outperform the random walk model according to Theil's Inequality Coefficient. A smaller window size and a considerable degree of filtering out noises in data help to improve the precision of the model. Dynamic gray models work well to recognize structures without any assumptions of statistical distribution of data. Under the conditions of the small window size used and a reasonable threshold, the structures discovered are close to human justification. Our proposed two-dimension measure helps to distinguish the structural changes produced by different algorithms and parameters, and our defined similarity matching distance measure successfully clustered randomly selected time series into structure-driven clusters. Online structure-driven time series clustering and similarity search based on our newly defined distance measure also perform well and closely approximate human perceptual justification. The

dynamic gray model system is capable of explaining the internal structure very well without statistical assumptions that are usually required in general time series analysis.

Future efforts will fall into three categories: theoretical, application, and performance. Theoretical research will be conducted to determine the order of optimal whitening function given an arbitrary time series and optimal measure and threshold that reflect human judgments in online fashion. As the time series data sets grow larger, the computational effort required to find structure changes also grows, especially when a structure change is not found for a long time. One suggestion for coping with this issue is to use a fixed window size; however, high-performance implementation of the methods might be another solution. In this dissertation, the applications of the framework in the domains of financial and network traffic are explored. It would be more promising to explore the actual profit gain in financial markets with proper trading strategies incorporated with structure changes mining algorithm. Also, it is well worth exploring the possible application of the algorithm to network resource allocation through network control mechanisms.

Because of the structure changes mining algorithm, it is possible to study the 'semantics' of the changes reflected in the internal structure of time series. Using defined ontology domain knowledge, it is possible now to explore the meaning and causes of changes, as well as the interrelationships among variables if the structural change algorithm is expanded into multivariable version.

REFERENCES

[1] G. E. P. Box, G. N. Jenkins, and G. C. Reinsel, *Time series analysis: forecasting and control*, 3rd ed. Englewood Cliffs, N.J.: Prentice Hall, 1994.

[2] J. R. Thompson, E. E. Williams, and I. M. Chapman Findlay, *Models for Investors in Real World Markets*: Wiley-Interscience, 2003.

[3] D. R. Cox, D. V. Hinkley, and O. E. Barndorff-Nielsen, *Time series models: in econometrics, finance and other fields*. London: Chapman & Hall, 1996.

[4] U. M. Fayyad, *Advances in knowledge discovery and data mining*. Menlo Park, Calif.: AAAI Press: MIT Press, 1996.

[5] S. M. Weiss and N. Indurkhya, *Predictive data mining: a practical guide*. San Francisco: Morgan Kaufmann Publishers, 1998.

[6] D. R. Cox and P. A. W. Lewis, *The statistical analysis of series of events*. London: Methuen, 1966.

[7] R. Cawley, G.-H. Hsu, and L. W. Salvino, "Detection and Diagnosis of Dynamics in Time Series Data: Theory of Noise Reduction," *Proceedings of the Chaos Paradigm: Developments and Applications in Engineering and Science, Mystic, Connecticut*, pp. 861-871, 1993.

[8] D. G. Luenberger, *Introduction to dynamic systems: theory, models, and applications*. New York: Wiley, 1979.

[9] S. Wiggins, *Introduction to applied nonlinear dynamical systems and chaos*. New York: Springer-Verlag, 1990.

[10] C. Grebogi and J. A. Yorke, *The impact of chaos on science and society*. Tokyo; New York: United Nations University Press, 1997.

[11] J. Deng, "Control problems of grey systems," *Systems & Control Letters*, vol. 1, pp. 288-294, 1982.

[12] J. Deng, "Introduction to Grey System Theory," *The Journal of Grey System*, vol. 1, pp. 1-24, 1989.

[13] Y.-P. Huang and T. M. Yu, "The Hybrid Grey-Based Models for Temperature Prediction," *IEEE Transactions on Systems, Man. and Cybernetics*, vol. 27, pp. 284-292, 1997.

[14] C. C. Aggarwal, "A Framework for Diagnosing Changes in Evolving Data Streams," presented at SIGMOD, San Diego, CA, 2003.

[15] S. Ahmad, T. Taskays-Temizel, and K. Ahmad, "Summarizing Time Series: Learning Patterns in 'Volatile' Series," presented at IDEAL 2004, Springer-Verlag Berlin Heidelberg, 2004.

[16] Y. Zhu and D. Shasha, "Efficient Elastic Burst Detection in Data Streams," in *SIGKDD*. Washington, DC, USA: ACM, 2003.

[17] H. D. Zhou, "Nonlinearity or Structural Break? - Data mining in Evolving Financial Data Sets from a Bayesian Model Combination Perspective," presented at Proceedings of the 38th Hawaii International Conference on System Science, Hawaii, 2005.

[18] G. M. Weiss and H. Hirish, "Learning to Predict Extremely Rare Events," *AAAI Workshop on Learning from Imbalanced Data Sets, Technical Report WS-00-05, AAAI Press, Menlo Park, CA*, pp. 64-68, 2000.

[19] A. Harvey, *Forecasting, structural time series models and the Kalman filter*, 1st paperback ed. Cambridge; New York: Cambridge University Press, 1990.

[20] L. Arnold, *Stochastic differential equations: theory and applications*. New York: Wiley, 1974.

[21] J. Medhi, *Stochastic processes*. New York: Wiley, 1982.

[22] G. Dorffner, "Neural Networks for Time Series Processing," *Nerual Network World*, vol. 6, pp. 447-468, 1996.

[23] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation fo complex Fourier series," *Mathematical Compuations*, vol. 19, 1965.

[24] R. Agrawal, C. Faloutsos, and A. Swami, "Efficient Similarity Search in Sequence Databases," *Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithm, New York:Springer, 1993.*, 1993.

[25] E. A. Robinson and M. T. Silvia, *Digital signal processing and time series analysis*. San Francisco: Holden-Day, 1978.

[26] J. S. Walker, *A primer on wavelets and their scientific applications*. Boca Raton, Fla.: Chapman & Hall/CRC, 1999.

[27] D. B. Percival and A. T. Walden, *Wavelet methods for time series analysis*. Cambridge; New York: Cambridge University Press, 2000.

[28] A. Jensen and A. La Cour-Harbo, *Ripples in mathematics: the discrete wavelet transform*. Berlin; New York: Springer, 2001.

[29] I. Kaplan, "http://www.bearcave.com/misl/misl_tech/wavelets/daubechies/index.html," 2003.

[30] Z. R. Struzik and A. Siebes, "The Haar Wavelet Transform in the Time Series Similarity Paradigm," *3rd European Conference on Principles and Practice fo Knowledge Discovery in Databases, Prague, Czeck Republic*, 1999.

[31] Y. Huhtala, Kärkkäinen, J. & Toivonen, H., " Mining for similarities in aligned time series using wavelets," *Data Mining and Knowledge Discovery: Theory, Tools, and Technology, SPIE Proceedings Series, Orlando, FL*, vol. 3695, pp. 150-160, 1999.

[32] Z. R. Struzik and A. Siebes, "Wavelet transform in similarity paradigm I," 1998.

[33] K.-p. Chan and A. W.-c. Fu, "Efficient Time Series Matching by Wavelets," *Proceedings of Internation Conference on Data Engineering (ICDE '99), Sydney*, pp. 126, 1999.

[34] W. Frawley, G. Piatesky-Shapiro, and C. Matheus, "Knowledge Discovery in Database: An Overview," *AI Magazine*, pp. 213-228, Fall, 1992.

[35] D. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining*. Cambridge, MA: MIT Press, 2001.

[36] M. Kantardzic, *Data mining: concepts, models, methods, and algorithms*. Hoboken, NJ: Wiley-Interscience: IEEE Press, 2003.

[37] J. F. Roddick and K. Hornsby, *Temporal, spatial, and spatio-temporal data mining: first international workshop, TSDM 2000, Lyon, France, September 12, 2000: revised papers*. Berlin; New York: Springer, 2001.

[38] M. Last, A. Kandel, and H. Bunke, *Data mining in time series databases*. New Jersey; London: World Scientific, 2004.

[39] H. Kargupta, *Data mining: next generation challenges and future directions*. Menlo Park, Calif.; London: AAAI, 2004.

[40] R. J. Povinelli, "Time Series Data Mining: Identifying Temporal Patterns for Characterization and Prediction on Time Series Events," in *Electrical and Computer Engineering Department*. Milwaukee, Wisconsin: Marquette University, 1999.

[41] 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, pp. 339-352, 2003.

[42] G. M. Weiss and H. Hirsh, "Learning to Predict Rare Events in Categorical Time-Series Data," *AAAI Workshop, Predicting the Future: AI Approaches to Time-Series Problems*, pp. 83-90, 1998.

[43] G. M. Weiss and H. Hirish, "Learning to Predict Rare Events in Event Sequences," presented at Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, 1998.

[44] R. J. Povinelli and X. Feng, "Temporal Pattern Identification of Time Series Data using Pattern Wavelets and Genetic Algorithms," *Artificial Neural Networks in Engineering, Proceedings*, pp. 691-696, 1998.

[45] R. J. Povinelli, "Identifying Temporal Patterns for Characterization and Prediction fo Financial Time Series Events," *Temporal, Spatial and Spatio-Temporal Data Mining: First International Workshop; revised papers / TSDM2000*, pp. 46-61, 2000.

[46] D. Gao, Y. Kinouchi, K. Ito, and X. Zhao, "Nerual networks for event extraction form time series: a back propagation algorithm approach," *Future Generation Computer Systems*, 2005.

[47] I. B. Konovalov, "Selection of future events from a time series in relation to estimations of forecasting uncertainty," *Neural and Evolutionary Computing*, 2002.

[48] V. Guralnik and J. Srivastava, "Event Detection from Time Series Data," *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, C.A.*, 1999.

[49] H. Wu, B. Salzberg, and D. Zhang, "Online Event-driven Subsequence Matching over Financial Data Stream," *International Conference on Management of Data, Proceedings of the 2004 ACM SIGMOD international conference on Management of data, Paris, France*, pp. 23 - 34, 2004.

[50] Merriam-Webster, "Pattern," in *http://www.m-w.com/dictionary/pattern*, 2005.

[51] R. J. Povinelli, "Using Genetic Algorithms to Find Temporal Patterns Indicative of Time Series Events," *GECCO 2000 Workshop: Data Mining with Evolutionary Algorithms*, pp. 80-84, 2000.

[52] D. Wang, "Temporal Pattern Processing," in *The Handbook of Brain Theory and Neural Networks, 2nd. 1163-1167*: MIT Press, Cambridge, MA, 2003.

[53] D. H. Diggs and R. J. Povinelli, "A Temporal Pattern Approach for Predicting Weekly Financial Time Series," *Artificial Neural Networks in Engineering, St. Louis, Missouri*, pp. 707-712., 2003.

[54] J. Lin, E. Keogh, S. Lonardi, and P. Patel, "Finding Motifs in Time Series," presented at SIGKDD, Edmonton, Alberta, Canada, 2002.

[55] E. Keogh, S. Londardi, and B. Y.-c. Chiu, "Finding Surperising Patterns in a Time Series Database in Linear TIme and Space," presented at SIGKDD, Edmonton, Alberta, Canada, 2002.

[56] S. Singh and E. Stuart, "A Pattern Matching Tool for Time-Series Forecasting," presented at Proc. 14th International Conference on Pattern Recognization, Brisbane, 1998.

[57] D. Dasgupta and S. Forrest, "Novelty Detection in Time Series Data using Ideas from Immunology," *Proceedings of The International Conference on Intelligent Systems*, 1999.

[58] W. G. Aref, M. G. Elfeky, and A. K. Elmagarimid, "Incremental, Online, and Merge Mining of Partial Periodic Patterns in Time-Series Databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, pp. 1-11, 2004.

[59] M. G. Elfeky, "Incremental Mining of Partial Periodic Patterns in Time-Series Databases," 2000.

[60] J. Han, G. Dong, and Y. Yin, "Efficient Mining of Partial periodic Patterns in Time Series Database," presented at Proc. 1999 Int. Conf. Data Engineering (ICDE'99), Sydney, Australia, 1999.

[61] D. Wang and B. Yuwono, "Incremental learning of complex temporal patterns," *IEEE Transactions on Neural Net.*, vol. 7, pp. 1465-1481, 1999.

[62] D. Wang and M. A. Arbib, "Complex temporal sequence learning based on short-term memory," *Proc. IEEE*, vol. 78, pp. 1536-1543, 1990.

[63] Hipel and Mcleod, "Nominal GNP (billion dollars), U.S., 1890 - 1974.," *http://www-personal.buseco.monash.edu.au/~hyndman/TSDL/roberts/gnpn.dat*, 1994.

[64] StockCharts.com, "Oceaneering International, Inc.," *http://www.stockcharts.com*, 2005.

[65] S.-H. Chen and C.-H. Yeh, "Using Genetic Programming to Model Volatility in Financial Time Series," *Genetic Programming, 1997, Proceedings of the second annual conference, Stanford, CA, Morgan Kaufmann, San Francisco, CA*, 1997.

[66] P. Perron, "The Great Crash, the Oil Price Shock and the Unit Root Hypothesis," *Econometrica*, pp. 1361-1401, 1989.

[67] H. Mannila, H. Toivonen, and A. I. Verkamo, "Discovery of frequent episodes in event sequences," *Data Mining and Knowledge Discover*, vol. 1, pp. 259-289, 1997.

[68] D. M. Hawkins, "Point estimation of parameters of piecewise regression models," *The Journal of the Royal Statistical Society Series C (Applied Statistics)*, vol. 25, pp. 51-57, 1976.

[69] D. M. Hawkins and D. F. Merriam, "Optimal zonation of digital sequential data," *Mathematical Geology*, vol. 5, pp. 389-395, 1973.

[70] S. B. Guthery, "Partition regression," *Journal of the American Statistical Association*, vol. 69, pp. 945-947, 1974.

[71] N. Sugiura and T. Ogden, "Testing change-points with linear trend," *Communications in Statistics B: Simulation and Computation*, pp. 287-322, 1994.

[72] A. N. Pettitt, "Some Results on Estimating a Change-Point using Non-Parametric Type Statistics," *J. Statist. Comput. Simul.*, vol. 11, pp. 261-272, 1980.

[73] M. Csèorgèo, L. Horvâath, and NetLibrary Inc., *Limit theorems in change-point analysis*. Chichester; New York: Wiley, 1997.

[74] A. N. Pettitt, "A non-parametric appraoch to the change-point problem," *Applied Statistics*, vol. 28, pp. 126-135, 1979.

[75] G. C. Chow, "Tests of Equality Between Sets of Coefficients in Two Linear Regressions," *Econometrica*, vol. 28, pp. 591-605, 1960.

[76] R. E. Quandt, "Tests of Hypothesis that a Linear Regression Obeys Two Separate Regimes," *Journal of the American Statistical Association*, vol. 55, pp. 324-330, 1960.

[77] D. W. K. Andrews, "Tests for Parameter Instability and Structural Change with Unknown Change Point," *Econometrica*, vol. 61, pp. 821-856, 1993.

[78] D. W. K. Andrews and W. Ploberger, "Optimal Tests when a Nuisance Paramter is Present Only Under the Alternative," *Econometrica*, vol. 62, pp. 1383-1414, 1994.

[79] E. Keogh, S. Chu, D. Hart, and M. Pazzani, " An Online Algorithm for Segmenting Time Series," *Proceedings of IEEE International Conference on Data Mining*, pp. 289-296, 2001.

[80] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "Segmenting Time Series: A Survey and Novel Approach," in *Data Mining in Time Series Databases*: World Scientific Publishing Company, 1993.

[81] C.-S. J. Chu, "Time Series Segmentation: A Sliding Window Approach," *Information Sciences*, vol. 85, pp. 147-173, 1995.

[82] H. L. Li and J. R. Yu, "A piecewise regression analysis with automatic change-point detection," *Intelligent Data Analysis*, vol. 3, pp. 75-85, 1999.

[83] J. R. Yu, G.-H. Tzeng, and H. L. Li, "General fuzzy piecewise regression analysis with automatic change-piont detection," *Fuzzy Sets and Systems*, vol. 119, pp. 247-257, 2001.

[84] K. Kumar and B. Wu, "Detection of change points in time series analysis with fuzzy statistics," *International Journal of Systems Science*, vol. 32, pp. 1185-1192, 2001.

[85] M. Raimondo and N. Tajvidi, "A PEAKS OVER THRESHOLD MODEL FOR CHANGE-POINT DETECTION BY WAVELETS," *Statistica Sinia*, vol. 14, pp. 395-412, 2004.

[86] A. J. Bagnall, G. Janacek, I. B. d. la, and M. Zhang, "Clustering Time Series from Mixture Polynomial Models with Discretised Data," presented at Proceedings of the second Australasian Data Mining Workshop, 2003.

[87] 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.

[88] J.-M. Adamo, *Data mining for association rules and sequential patterns: sequential and parallel algorithms*. New York: Springer, 2001.

[89] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehoritra, "Dimensional Reduction for Fast Similarity Search in Large Time Series Databases," *Knowledge and Information Systems*, vol. 3, pp. 263-286, 2001.

[90] D. E. Shasha and Y. Zhu, *High performance discovery in time series: techniques and case studies*. New York: Springer, 2004.

[91] C. Faloutsos, M. Rangenathan, and Y. Manolopoulos, "Fast Subsequence Matching in Time-Series Database," presented at Proc. ACM SIGMOD Conf., Minneapolis, 1994.

[92] A. Guttman, "R-trees: A dynamic index structure for spatial searching," presented at Proc. ACM SIGMOD Conf., 1984.

[93] Y.-S. Moon, K.-Y. Whang, and W.-S. Han, "General Match: A Subsequence Matching Method in Time-Series Databases Based on Generalized Windows," *SIGMOD*, pp. 382-393, 2002.

[94] A. C. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. J. Strauss, "Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries," presented at Proceedings of the 27th VLDB Conference, Roma, Italy, 2001.

[95] D. Wu, D. Agrawal, and A. E. Abbadi, "Efficient Retrieval for Browsing Large Image Databases," presented at Proc. CIKM, Rockville, MD., 1996.

[96] F. Korn, H. V. Jagadish, and C. Falouts, "Efficient Supporting Ad Hoc Queries in Large Datasets of Time Sequences," presented at SIGMOD, 1997.

[97] E. Keogh and M. Pazzani, "Scaling up Dynamic Time Warping for Datamining applications," presented at KDD, Boston, MA, 2000.

[98] B.-K. Yi and C. Falouts, "Fast time sequence indexing fro arbitrary Lp norms," presented at Proceedings of the 26th International Conference on Very Large Databases, Cairo, Egypt, 2000.

[99] E. Keogh, K. Chakrabarti, S. Mehoritra, and M. Pazzani, "Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases," presented at Proc. SIGMOD, Santa Barbara, California, 2001.

[100] B.-K. Yi, H. V. Jagadish, and C. Falouts, "Efficient Retrieval of Similar Time Sequences under Time Warping," *ICDE*, pp. 201-208, 1998.

[101] G. Das, D. Gunopulos, and H. Mannila, "Finding Similar Time Series," presented at PKDD, 1997.

[102] H. Wu, B. Salzberg, and G. C. Sharp, "Subsequence Matching on Structured Time Series Data," presented at SIGMOD, Baltimore, Maryland, USA, 2005.

[103] M. Datar, A. Gionis, P. Indyk, and R. Motwani, "Maintaining Stream Statistics over Sliding Windows," *SIAM Journal on Computing*, vol. 31, pp. 1794-1813, 2002.

[104] R. Jin and G. Agrawal, "Efficient Decision Tree Construction on Streaming Data," presented at Conference on Knowledge Discovery in Data archive Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, Washington, D.C., 2003.

[105] S. Muthukrishnan, R. Shah, and J. S. Vitter, "Mining Deviants in Time Series Data Streams," presented at Proceedings of the 16th International Conference on Scientific and Statistical Database Management (SSDBM 2004), Santorini Island, Greece, 2004.

[106] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and Issues in Data Stream Systems," presented at Proceedings of 21st ACM Symposium on Principles of Database Systems (PODS 2002), Madison, Wisconsin, 2002.

[107] L. Gao, Z. Yao, and X. S. Wang, "Evaluating continuous nearest neighbor queries for streaming time series via prefetching," presented at CIKM, 2002.

[108] L. Gao and X. S. Wang, "Continually Evaluating Similarity-based Pattern Queries on a Streaming Time Series," presented at SIGMOD, 2002.

[109] X. Liu and H. Ferhatosmanoglu, "Efficient k-NN Search on Streaming Data Series," *SSTD*, pp. 289-300, 2003.

[110] L. O'Callaghan, N. Mishra, A. Meyerson, and S. Guha, "Streaming-data algoritihms for high-quality clustering," *ICDE*, pp. 685-, 2002.

[111] J. Gehrke, F. Korn, and D. Srivastava, "On Computing Correlated Aggregates Over Continual Data Streams," *SIGMOD*, pp. 126-133, 2001.

[112] J. Qu, "Notes on Grey System and Its Application, East China University," 1990.

[113] W. Hu, B. Hua, and C. Yang, "Building thermal process analysis with grey system method," *Building and Environment*, pp. 599-605, 2002.

[114] W. Hu and C. Yang, "Grey model of direct solar radiation intensity on the horizontal plane for cooling loads calculation," *Building and Environment*, pp. 587-593, 2000.

[115] F.-M. Tseng, H.-C. Yu, and G.-H. Tzeng, "Applied Hybrid Grey Model to Forecast Seasonal Time Series," *Technological Forecasting & Social Change*, vol. 67, pp. 291-302, 2001.

[116] C.-B. Lin, S.-F. Su, and Y.-T. Hsu, "High-precision forecast using grey models," *International Journal of Systems Science*, vol. 32, pp. 609-619, 2001.

[117] C. F. Fan, Z. Jin, and W. Tian, "A novel hybrid grey-based strategy for improving the model precision of a dynamicallly tuned gyroscope," *Measurement Science and Technology*, pp. 759-765, 2003.

[118] F. L. Lewis, *Applied Optimal Control and Estimation*: Prentice-Hall, 1992.

[119] C.-H. Tsai, C.-L. Chang, and L. Chen, "Applying Grey Relational Analysis to the Vendor Evaluation Model," *International Journal of The Computer, The Internet and Management*, vol. 11, pp. 45-53, 2003.

[120] C.-L. Chang, C.-H. Tsai, and L. Chen, "Applying Grey Relational Analysis to the Decathlon Evaluation Model," *International Journal of The Computer, The Internet and Management*, vol. 11, pp. 54-62, 2003.

[121] J. Qu and H. R. Arabnia, "A novel short-term stock price predicting system," *The 2005 International Conference on Information and Knowledge Engineering, Las Vegas, Nevada, USA*, 2005.

[122] J. T. Yokum and J. S. Armstrong, "Beyond accuracy: Comparison of criteria used to select forecasting methods," *International Journal of Forecasting*, pp. 591-591, 1995.

[123] C. D. Lewis, *Industrial and Business Forecasting Method*. London: Butterworth Scientific, 1982.

[124] H. Theil, *Economic Forecasts and Policy*. Amsterdam: North-Holland Publishing Company, 1961.

[125] D. M. Hawkins, "Fitting Multiple Change-point Models to Data," *Computational Statistics & Data Analysis*, vol. 37, pp. 323-341, 2001.

[126] Merriam-Webster, "Noise," in *http://www.m-w.com/cgi-bin/dictionary*, 2005.

[127] E. J. Kostelich and T. Schreiber, "Noise reduction in chaotic time series data: a survey of common methods," *Phys. Rev. E*, vol. 48, pp. 1752, 1993.

[128] Wikipedia, "Wikipedia Encyclopedia," http://en.wikipedia.org/wiki/White_noise, 2005.

[129] H. D. I. Abarbanel, *Analysis of observed chaotic data*. New York: Springer, 1996.

[130] A. Lapedes and R. Farber, "Nonlinear signal processing using neural networks: prediction and system modelling," *Report LA-UR-87-2662*, 1987.

[131] J. D. Farmer and J. J. Sidorowich, "Predicting Chaotic Time Series," *Physical Review Letters*, vol. 59, pp. 845-848, 1987.

[132] F. Attneave, "Some informational aspects of visual perception," *Psychological review*, vol. 61, pp. 183-193, 1954.

[133] S. Sigh, "A Pattern Matching Tool for Forecasting," *Proceedings of 14th International Conference on Pattern Recognition (ICPR'98), Brisbane, IEE Press*, vol. 1, pp. 103-105, 1998.

[134] W.-G. Teng, M.-S. Chen, and P. S. Yu, "A Regression-Based Temporal Pattern Mining Scheme for Data Stream," *Proceedings of the 29th VLDB Conference, Berlin, Germany, 2003*, 2003.

[135] C. J. Neely, "The temporal pattern of trading rule returns and exchange rate intervention: intervention does not generate technical trading profits," *Journal of International Economics*, pp. 211-232, 2002.

[136] H. Bozdogan, *Statistical data mining and knowledge discovery*. Boca Raton, Fla.: Chapman & Hall/CRC, 2004.

[137] B. Kovalerchuk and E. Vityaev, *Data Mining in Finance: advances in relational and hybrid methods*: Kluwer Adademic Publishers, 2000.

[138] M. Osborne, "Brownian Motion in the Stock Market," *Operations Research*, vol. 7, pp. 145-173, 1959.

[139] M. Burton, *A Random Walk Down Wall Street*: W.W. Norton Company Ltd., 1985.

[140] A. Lo, W. MacKinlay, and A. Craig, *A Non-Random Walk Down Wall Street*: Princeton University Press, 2001.

[141] S.-H. Chen and C.-H. Yeh, "Using Genetic Programming to Model Volatility in Financial Time Series: The Cases of Nikkei 224 and S&P 500," *The 4th JAFEE International Conference on Investments and Derivatives (JIC'97), Royal Park Hotel, Tokyo, Japan*, 1997.

[142] E. Kalyvas, "Using Neural Networks and Genetic Algorithms to Predict Stock Market Returns," University of Manchester, 2001.

[143] R. J. Frank, N. Davey, and S. P. Hunt, "Time Series Prediction and Neural Networks," *Journal of Intelligent and Robotic Systems*, pp. 91-103, 2000.

[144] K. J. Kim and I. Han, "Genetic algorithm approach to feature discretization in artificial neural network for the prediction of stock price index," *Expert systems with applications*, vol. 19, pp. 125-13, 2000.

[145] P. A. Samuelson, "Proof that properly anticipated prices fluctuate randomly," *Industrical Management Review*, vol. 6, pp. 41-50, 1965.

[146] B. Mandelbrot, "Forecasts fo future prices, unbiased market and martingale models," *Journal of Business*, vol. 39, pp. 242-255, 1966.

[147] N. K. Groschwitz and G. C. Polyzos, "A Time Series Model of Long-Term NSFNET Backbone Traffic," in *IEEE Conference on Communications (ICC)*, 1994.

[148] S. Basu, A. Mukherjee, and S. Klivansky, "Time Series Models for Internet Traffic," Georgia Institute of Technology, 1996.

[149] A. Sang and S.-q. Li, "A Predictability Analysis of Netowork Traffic," *INFOCOM, Tel Aviv, Israel*, 2000.

[150] Y. Qiao, J. Skicewicz, and P. Dinda, "An Empirical Study of the Multiscale Predictability of Network Traffic," presented at IEEE Proceedings of HPDC, 2003.

[151] K. Papagiannaki, N. Taft, Z.-L. Zhang, and C. Diot, "Long-Term Forecasting of Internet Backbone Traffic," presented at Proc. IEEE INFOCOM 2003, 2003.

[152] L. Xue and G. Cheng, "Network Traffic Forecast Based on Seasonal Neural Network Model," presented at APAN Network Research Workshop, 2004.

[153] L. Cheng and I. Marsic, "Modeling and prediction of session throughput of constant bit rate streams in wireless data networks," *Proceedings of the 2003 IEEE Wireless Communications and Networking Conference(WCNC'03), New Orleans*, vol. 3, pp. 1733-1741, 2003.