A New Method for Outlier Detection on Time Series Data

by

Lin Liu
M.Eng., Beijing University of Posts and Telecommunications, 2013
B.Eng., Minzu University of China, 2010

a Thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in the
School of Computing Science
Faculty of Applied Sciences

© Lin Liu 2015
SIMON FRASER UNIVERSITY
Summer 2015

All rights reserved.
However, in accordance with the Copyright Act of Canada, this work may be reproduced without authorization under the conditions for “Fair Dealing.” Therefore, limited reproduction of this work for the purposes of private study, research, criticism, review and news reporting is likely to be in accordance with the law, particularly if cited appropriately.
APPROVAL

Name: Lin Liu
Degree: Master of Science
Title of Thesis: A New Method for Outlier Detection on Time Series Data

Examination Committee: Dr. Arrvindh Shriraman,
Assistant Professor, Chair

Dr. Jian Pei,
Professor, Senior Supervisor

Dr. Qianping Gu,
Professor, Supervisor

Dr. Jiangchuan Liu,
Professor, External Examiner

Date Approved: June 9th, 2015
Abstract

Time series outlier detection has been attracting a lot of attention in research and application. In this thesis, we introduce the new problem of detecting hybrid outliers on time series data. Hybrid outliers show their outlyingness in two ways. First, they may deviate greatly from their neighbors. Second, their behaviors may also be different from that of their peers in other time series. We propose a framework to detect hybrid outliers, and two algorithms based on the framework are developed to show the feasibility of our framework. An extensive empirical study on both real data and synthetic data verifies the effectiveness and efficiency of our algorithms.

Keywords: Outlier detection; time series; hybrid outlier; distance; prediction
To my family.
“The only true wisdom is in knowing you know nothing.”

— Socrates, (470 - 399 BC)
Acknowledgments

I would like to express my sincerest gratitude to my senior supervisor, Dr. Jian Pei, for his great patience, wealth of knowledge, warm encouragement and continuous support throughout my Master's studies. He taught me not only valuable knowledge, but also the wisdom of life, which gave me interest and confidence in accomplishing this thesis.

I am very thankful to Dr. Qianping Gu for being my supervisor and giving me helpful suggestions that helped me to improve my thesis. I am also grateful to thank Dr. Jiangchuan Liu and Dr. Arrvindh Shriraman for serving in my examining committee.

My deepest thanks to Professor Jiali Bian and Professor Jian Kuang in Beijing University of Posts and Telecommunications for educating and training me for my career.

I thank Dr. Huaizhong Lin, Dr. Aihua Wu, Dr. Fuyuan Cao, Dr. Kui Yu, Dr. Dongwan Choi, Dr. Guanting Tang, Xiao Meng, Juhua Hu, Xiangbo Mao, Chuancong Gao, Xiaoning Xu, Yu Tao, Xuefei Li, Yu Yang, Jiaxing Liang, Beier Lu, Li Xiong, Lumin Zhang, Xiang Wang, Zicun Cong, and Xiaojian Wang for their kind help during my study at SFU. I am also grateful to my friends at Fortinet. I thank Kai Xu, for his guide and insight suggestions.

Last but not least, I am very grateful to my parents, my husband, my sister, and my brother, for their endless love and support through all these years. Without them, never can I accomplish this thesis.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approval</td>
<td>ii</td>
</tr>
<tr>
<td>Abstract</td>
<td>iii</td>
</tr>
<tr>
<td>Dedication</td>
<td>iv</td>
</tr>
<tr>
<td>Quotation</td>
<td>v</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>vi</td>
</tr>
<tr>
<td>Contents</td>
<td>vii</td>
</tr>
<tr>
<td>List of Tables</td>
<td>ix</td>
</tr>
<tr>
<td>List of Figures</td>
<td>x</td>
</tr>
</tbody>
</table>

1 Introduction

1.1 Motivations ............................................. 1
1.2 Contributions ........................................... 3
1.3 Organization of the Thesis ............................... 3

2 Related Work

2.1 Detecting Outliers within a Single Time Series ................. 4
  2.1.1 Time Points as Outliers ................................ 4
  2.1.2 Subsequences as Outliers ............................... 5
2.2 Detecting Outliers in Multiple Time Series .................... 6
  2.2.1 Parametric Methods .................................... 7
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.2</td>
<td>Discriminative Methods</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>Problem Definition and Framework</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>Problem Definition</td>
<td>9</td>
</tr>
<tr>
<td>3.2</td>
<td>Framework</td>
<td>13</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Grouping Phase</td>
<td>14</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Score Computation Phase</td>
<td>14</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Distance Weight Adjustment Phase</td>
<td>15</td>
</tr>
<tr>
<td>3.2.4</td>
<td>Convergence</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>Two Algorithms</td>
<td>21</td>
</tr>
<tr>
<td>4.1</td>
<td>The Distance-based Algorithm (DA)</td>
<td>21</td>
</tr>
<tr>
<td>4.2</td>
<td>The Prediction-based Algorithm (PA)</td>
<td>23</td>
</tr>
<tr>
<td>4.3</td>
<td>Example</td>
<td>26</td>
</tr>
<tr>
<td>4.3.1</td>
<td>The DA Result</td>
<td>27</td>
</tr>
<tr>
<td>4.3.2</td>
<td>The PA Result</td>
<td>29</td>
</tr>
<tr>
<td>5</td>
<td>Experiments</td>
<td>31</td>
</tr>
<tr>
<td>5.1</td>
<td>Results on a Real Data Set</td>
<td>31</td>
</tr>
<tr>
<td>5.2</td>
<td>Results on Synthetic Data Sets</td>
<td>35</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Effectiveness</td>
<td>37</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Efficiency</td>
<td>38</td>
</tr>
<tr>
<td>5.3</td>
<td>Summary of Results</td>
<td>38</td>
</tr>
<tr>
<td>6</td>
<td>Conclusions</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>Bibliography</td>
<td>47</td>
</tr>
</tbody>
</table>
List of Tables

3.1 The summary of symbols ........................................... 13
3.2 Notations used in the algorithm framework ...................... 15
3.3 2-NN segments during iterations ................................ 18
3.4 The outlier degree rank of time points during iterations ......... 18

4.1 Notations used in the distance-based algorithm ................... 22
4.2 Three time series .................................................. 22
4.3 Notations used in the prediction-based algorithm ................. 24
4.4 2-NN segments during iterations for DA ......................... 27
4.5 Outlier score of each time point during iterations for DA ........ 27
4.6 Distance weight of each time point during iterations for DA ...... 27
4.7 Final outlier degree rank of time points for DA .................. 28
4.8 2-NN segments during iterations for PA ........................ 28
4.9 Outlier scores of time points during iterations for PA .......... 28
4.10 Distance weight of time points during iterations for PA ........ 29
4.11 Average score of all time point in all time series during iterations for PA . . . 29
4.12 Final outlier degree rank of time points for PA ................ 30

5.1 Top 10 hybrid outliers detected by our two algorithms ........... 32
5.2 Top 10 hybrid outliers detected by the baselines ................. 32
5.3 Top 10 self-trend and peer-wise outliers ........................ 33
5.4 The description of 4 real data sets ................................. 35
5.5 Original time series ............................................... 36
5.6 Synthetic time series .............................................. 36
List of Figures

2.1 4 time series data ................................................................. 6
3.1 An example of time series ...................................................... 10
3.2 Small time series data ......................................................... 12
3.3 Overview of the framework ................................................... 14
5.1 Results on the real data set (w = 5, k = 4) ............................... 34
5.2 Monthly exchange rates of 7 currencies w.r.t. USD .................... 39
5.3 Results on synthetic temperature data set ................................. 40
5.4 Results on synthetic exchange rate data set .............................. 41
5.5 Results on synthetic interest rate data set ............................... 42
5.6 Results on synthetic government GDP data set ......................... 43
5.7 Results of DA on 4 synthetic data sets with respect to r (x=4, y=40, k=5, w=6) ................................................................. 44
Chapter 1

Introduction

In this chapter, we first discuss several interesting applications that motivate the problem of hybrid outlier detection on time series data, which will be studied in this thesis. Then, we will summarize the major contributions and describe the structure of the thesis.

1.1 Motivations

Time series data are becoming increasingly important in many areas, such as finance, signal processing, and weather forecasting. Time series data are observations collected sequentially over time. Some examples of time series data are the daily stock price of a firm, the daily temperature for a region, the altitude of an airplane every second, the exchange rate of a currency every month, and so on. In all these instances, a measure is associated with a timestamp. By analyzing time series data, people can extract useful information from the data and gain insight into the data.

Despite the large amounts of data, people are very interested in unusual values (or outliers) out of the whole time series data. Usually time series outliers are informally defined as somehow unexpected or surprising values in relation to the rest of the series, often the immediately preceding and following observations [37]. A great number of methods have been proposed to detect different types of time series outliers, and most of the existing methods work on detecting outlier points or subsequences in a single time series [20] [21] [38] or finding entire time series as outliers in multiple time series [43] [31] [42].

In this thesis, we pose the new problem of detecting hybrid outliers on time series data. Hybrid outliers show their outlyingness in two ways. First, they may deviate greatly from
their neighbors. Second, their behaviors may also be different from that of their peers in other time series. If two observations in different time series have the same timestamp, they are peers of each other. In general, a hybrid outlier has a combination of self-trend outlier behavior and peer-wise outlier behavior. Hybrid outliers are valuable in various application areas. Some motivating examples are as follows.

**Example 1 (Motivation - temperature):** Temperature data are a kind of time series data, which are commonly collected in climate and environment research. Anomalies in temperature data provide significant insights about hidden environmental trends, which may have caused such anomalies.

In temperature data, each time series usually represents the temperature values of a geographical region over time, and the time series from close regions are likely to have similar trends. In many cases, the interestingness of a particular temperature value of a region not only depends on its past temperature values, but also depends on the data from close regions. For example, suppose there is a sudden temperature increase in Vancouver, and we know other close regions, like Richmond and Burnaby, do not have temperature rise at the same time, then this temperature change in Vancouver would be very interesting and worth investigation. On the other hand, if the temperature increase in Vancouver is along with the rising temperatures in other close regions, this change would be more likely some common climate trends.

Clearly, outlier detection on time series data from both the self-trend aspect and the peer-wise aspect is practically useful.

**Example 2 (Motivation - exchange rate):** The exchange rate of a currency partly reflects the economic health of the country. By analyzing the change of currency exchange rate, people can predict economic trends of the country to some extent. In currency exchange rate data, each time series represents a sequence of exchange rate values (usually with respect to US dollar) of a currency. Some currencies often have similar trends, like GBP (British Pound), EUR (Euro), and CHF (Swiss Franc), because the countries have frequent economic cooperation.

When analyzing the exchange rate of a currency, people are not only interested in the trend with respect to its historical data, but also interested in how the trend is different from those of some other close related currencies. For example, if the exchange rate of GBP drops greatly this month and the exchange rates of EUR and CHF also drop at the same time, then this change is probably caused by some economic problem of the whole European
continent. However, if the GBP is the only currency going down, this change would be more interesting and it deserves to be further analyzed.

Again, outlier detection on time series data from both the self-trend aspect and the peer-wise aspect is meaningful.

Besides, hybrid outliers are interesting and useful in many other applications, like medical diagnosis, network intrusion detection systems, stock market analysis, interesting sensor events, etc. To the best of our knowledge, there does not exist any systematic study on mining hybrid outliers.

1.2 Contributions

In this thesis, we tackle the problem of hybrid outlier detection on time series data, and make the following contributions.

- We pose the new problem of detecting hybrid outliers.
- We propose a heuristic framework for detecting hybrid outliers.
- We develop two algorithms based on the framework. The first one is a distance-based algorithm and the other one is a prediction-based algorithm.
- We report an empirical evaluation on both synthetic and real-world data sets, which validates the effectiveness and efficiency of our algorithms.

1.3 Organization of the Thesis

The rest of the thesis is organized as follows. In Chapter 2, we review the related work on time series outlier detection. We then formulate the problem of hybrid outlier detection on time series data and propose our framework in Chapter 3. In Chapter 4, two algorithms based on the framework are presented. We report our experimental results in Chapter 5, and conclude the thesis in Chapter 6.
A significant amount of work has been performed in the area of time series outlier detection. The existing techniques on time series outlier detection can be divided into two categories: the techniques to detect outliers within a single time series, and the techniques over multiple time series, which are reviewed in Section 2.1 and Section 2.2, respectively.

### 2.1 Detecting Outliers within a Single Time Series

Given a single time series, an outlier can be either a time point or a subsequence. In this subsection, we will discuss methods for both these cases.

#### 2.1.1 Time Points as Outliers

Various methodologies were proposed to find outlier time points in a time series. Deviant detection and prediction models are two of them.

Deviants are outlier time points in time series from a minimum description length (MDL) point of view [20]. If the removal of a time point from the time series leads to an improved compressed representation of the remaining items, then the point is a deviant [26]. A dynamic programming mechanism for deviant detection was developed by Jagadish et al. [20]. Muthukrishnan et al. [26] proposed an approximation method to the dynamic programming based solution.

The general idea of prediction models [1] is to predict the value of a time point and compute the deviation from the predicted value to its real value as the outlier score for the
CHAPTER 2. RELATED WORK

time point. Basu et al. [4] predicted the value of the time point at time $t$ as a median of the values of the time points in the size of $2w$ window from $t - w$ to $t + w$. Hill et al. [19] used single-layer linear network predictor (or AR model) which predicts the value at time $t$ as a linear combination of the $q$ previous values. A vector auto-regressive integrated moving average (or ARIMA) model was proposed to identify 4 types of outliers from multi-variate time series by Tsay et al. [38]. ARIMA is a traditional and frequently used methodology for time series forecasting, which was first popularized by Box and Jenkins [17]. An ARIMA model predicts a value in a time series as a linear combination of its own past values and past errors.

2.1.2 Subsequences as Outliers

A bunch of methods were proposed to discover discord, which is a type of outlier subsequences in a time series. Keogh et al. [22] gave the definition of discord: given a time series $T$, the subsequence $D$ of length $n$ beginning at position $l$ is said to be the discord (or outlier) of $T$ if $D$ has the largest distance to its nearest non-overlapping match. The brute force algorithm for finding discord is to consider all possible subsequences $s \in S$ of length $n$ in $T$ and compute the distance of each such $s$ with each other non-overlapping $s' \in S$. To make the computation efficient, effective pruning techniques implemented by smartly ordering subsequence comparisons are used in many methods, like heuristic reordering of candidate subsequences using SAX (HOT SAX) [21], Haar wavelet and augmented tries (WAT) [9], and locality sensitive hashing (LSH) [40].

Chen et al. [12] defined the subsequence outlier detection problem for an unequal interval time series which is a time series with values sampled at unequal time intervals. For such a time series, a pattern is defined as a subsequence of two neighbor points. If a pattern is infrequent or rare in the time series, then the pattern is an outlier. The Haar wavelet transform is applied to identify outlier patterns. Shahabi et al. [35] proposed trend and surprise abstractions (TSA) tree to store trends and surprises in terms of Haar wavelet coefficients. Wei et al. [41] defined a subsequence as anomalous if its similarity with a fixed previous part of the sequence is low.

The methods discussed in this section only focus on detecting self-trend outliers which are different from other elements in the same time series. However, our problem in this thesis is to detect hybrid outliers, which have a combination of self-trend outlier behavior and peer-wise outlier behavior. Example 2.1 gives an example of hybrid outliers. We can
apply some techniques reviewed in this section to detect the self-trend outlier behavior of time points, but those techniques can not solve our problem completely.

**Example 2.1** (Hybrid Outlier). *There are 4 time series $s_1, s_2, s_3, s_4$ in Figure 2.1, and $a = 35, b = 33, c = 32, d = 34, e = -42, f = -42, g = -43$ are time points in them. Time points $d, e, f, g$ dramatically deviate from their neighbor time points, and they are self-trend outliers. Time points $a, b, c, d$ are peer-wise outliers because of their great difference from the time points with the same timestamps in other time series. Our problem is to find hybrid outliers. $d$ is a hybrid outlier, which is not only different from its neighbors, but also different from the time points with the same timestamps in other time series.*

### 2.2 Detecting Outliers in Multiple Time Series

Given multiple time series, assuming that most of the time series are normal while a few are anomalous, the problem is to find all anomalous time series. In most of the methods for solving such problems, a model is first learned based on all the time series, and an outlier score for each time series is then computed with respect to the model. The model could be supervised or unsupervised, and we mainly review unsupervised models in this section.
2.2.1 Parametric Methods

A summary model is created on the base data in unsupervised parametric methods. For a test sequence, if the probability of generation of it from the model is super low, then the sequence is marked abnormal. The outlier score of the entire time series is computed based on the probability of each element. Markov chain models and Finite state automata (FSA) are two frequently used models.

Markov methods estimate the conditional probability for each symbol in a test sequence conditioned on the symbols preceding it. Most of the techniques utilize the short memory property of sequences [33] and they store conditional information for a history size $k$. Ye et al. [43] proposed a technique where a Markov model with $k = 1$ is used. A probabilistic suffix tree (PST) is a tree representation of a variable-order markov chain [36], which was used in [36] and [42] for efficient computations.

A fixed length Markovian technique (FSA) [25] determines the probability of a symbol, conditioned on a fixed number of preceding symbols, and the techniques were used for outlier detection in [11], [24], and [25]. The approach employed by FSA uses a Finite State Automaton to estimate the conditional probabilities [11]. FSA can be learned from length $n$ subsequences in training data. During testing, all length $n$ subsequences can be extracted from a test sequence and fed into the FSA. If the FSA reaches a state from where there is no outgoing edge corresponding to the last symbol of the current subsequence, then an anomaly is detected.

2.2.2 Discriminative Methods

Discriminative methods first define a distance function that measures the distance between two time series. Clustering algorithms are then applied to cluster time series, and the outlier score of a time series is computed as the distance to the centroid of the closest cluster. There are various distance measures and clustering algorithms.

The most straightforward distance measure for time series is the Euclidean Distance [16] and its variants, based on the common $L_p$-norms [44]. Dynamic time warping (DTW) was introduced by Berndt et al. [6], which allows a time series to be stretched or compressed to provide a better match with another time series. Another group of distance measures for time series are developed based on the concept of the edit distance for strings. The best known such distance is the LCSS distance, utilizing the longest common subsequence
model [2], [39].

The majority of clustering algorithms can be applied on time series data, including k-Means [13], EM [28], phased k-Means [31], dynamic clustering [34], k-medoids [10], one-class SVM [15], etc. Since different clustering methods have different complexity, the selection of a clustering method depends on specific application domain.

The outliers identified by the methods reviewed in this section are entire time series, while our problem in this thesis is to detect hybrid outliers which are time points. Thus, those methods are obviously not suitable for solving our problem.

**Example 2.2 (Outlier Time Series).** *Let us consider the same time series data in Example 2.1. The methods introduced in this section will find time series \( s_1 \) as an outlier, which is different from the other time series.*

To the best of our knowledge, we are the first to detect hybrid outliers on time series data.
Chapter 3

Problem Definition and Framework

In this chapter, we first present the formal definition of our problem in Section 3.1. Then, we propose a heuristic framework for detecting hybrid outliers in a time series database in Section 3.2.

3.1 Problem Definition

Definition 3.1 (Time Data Point). A time data point (or time point for short) consists of a timestamp and an associated data value.

Definition 3.2 (Time Series). A time series $s$ is a sequence of time points. The data values are ordered in timestamp ascending order. We assume that all timestamps take positive integer values. We denote by $s[j]$ the time point of time series $s$ at timestamp $j$.

To keep our discussion simple, in this thesis, we assume that all time series are of the same length, denoted by $m$, i.e., each time series $s$ has a time point $s[j]$ at timestamp $1 \leq j \leq m$. When the time series are not of the same length, alignments or dynamic warping can be applied as the preprocessing step. We also assume that all time series are normalized by Z-normalization [18].

Example 3.1 (Time Point and Time Series). In Figure 3.1, $s[1], s[2], s[3], s[4], s[5], s[6]$, and $s[7]$ are time points. The value of time point $s[3]$ is 1, and its timestamp is 3. $s$ is a time series, and it is a sequence of those 7 time points.
Definition 3.3 (Segment). Given a time series $s$ of length $m$, $s[j:e] = s[j]s[j+1] \ldots s[e]$ $(1 \leq j \leq e \leq m)$ is the segment at timestamp interval $[j:e]$. The length of $s[j:e]$ is $l = e - j + 1$.

When $l$ is 1, i.e., $j = e$, segment $s[j:e]$ is a time point.


In this thesis, among all time points in time series, what we are particularly interested in is outlier points.

Definition 3.4 (Neighbor). Given a window size $w > 0$, for a time point $s[j]$ in time series $s$, a time point $s[j']$ is called a neighbor of $s[j]$ if $0 < |j - j'| \leq w$.


A self-trend outlier is a time point that deviates remarkably from its neighbors.

Definition 3.5 (Self-Trend Outlier). Suppose there is an outlyingness function $F_s$, which measures the difference between a time point $s[j]$ and its neighbors as the self-trend outlier degree $sdeg[j] = F_s(s,j)$. Given a time series $s$ and a self-trend outlier degree threshold $\delta_s > 0$, a time point $s[j]$ is a self-trend outlier if its self-trend outlier degree $sdeg[j] \geq \delta_s$.

Example 3.4 (Self-Trend Outlier). Assume we have a self-trend outlyingness function, which computes the smallest distance from a time point to its neighbors as the self-trend outlier degree. Suppose $\delta_s = 5$ and the window size $w$ is 2, for time series $s$ in Figure 3.1, $s[4]$ is a self-trend outlier because its self-trend outlier degree $sdeg[4] = 8 > \delta_s$. 

Figure 3.1: An example of time series
Definition 3.6 (Time Series Database). A time series database $S$ consists of $n$ time series, $S = \{s_i | 1 \leq i \leq n\}$, where $s_i$ is the $i$-th time series in $S$.

Definition 3.7 (Peer). Given a time series database $S$, for a time point $s_i[j]$ in time series $s_i$, a time point $s_{i'}[j]$ is called a peer of $s_i[j]$ if $i \neq i'$.

Example 3.5 (Peer). In Figure 3.2, there is a time series database which consists of 7 time series $s_1, s_2, \ldots, s_7$, and $a, b, c, \ldots, i, j$ are time points in them. Time points $d, h$ and $i$ are peers of each other.

A peer-wise outlier is a time point whose behavior is exceptional comparing to its peers in a given set of time series.

Definition 3.8 (Peer-Wise Outlier). Suppose there is an outlyingness function $F_p$, which measures the difference between a time point $s_i[j]$ and its peers in a given set of time series as the peer-wise outlier degree $\text{pdeg}_i[j] = F_p(S, j, i)$. Given a time series $s_i$ in a time series database $S$ and a peer-wise outlier degree threshold $\delta_p > 0$, a time point $s_i[j]$ is a peer-wise outlier if the peer-wise outlier degree $\text{pdeg}_i[j] \geq \delta_p$.

Example 3.6 (Peer-Wise Outlier). Assume we have a peer-wise outlyingness function, which computes the smallest distance from a time point to its peers in a given set of time series as the peer-wise outlier degree. Let us consider the time series database $S$ in Figure 3.2. When $\delta_p = 20$, $s_1[5]$ is a peer-wise outlier, because the smallest distance from it to the peers $(s_2[5], s_3[5], s_4[5])$ is 35, which is greater than $\delta_p$, i.e., $\text{pdeg}_1[5] = 35 > \delta_p$.

Self-trend outlier and peer-wise outlier are two different types of outliers, and they have different outlier behaviors. In practice, some outliers show their outlyingness in two ways. First, they may deviate greatly from their neighbors. Second, their behaviors may also be different from that of their peers. For example, in Figure 3.2, time point $j$ is different from that of their peers. We call this type of outlier hybrid outlier. In general, a hybrid outlier has a combination of those two types of outlier behaviors.

Definition 3.9 (Hybrid Outlier Score). Hybrid outlier score (or outlier score for short) $\alpha_i[j]$ is a non-negative value which measures the hybrid outlier degree of a time point $s_i[j]$.

Definition 3.10 (Hybrid Outlier). Suppose there is an outlyingness function $F$, which computes the expectation of a time point $s_i[j]$ based on its neighbors and a set of its peers,
Figure 3.2: Small time series data

and gives the difference degree between $s_i[j]$ and its expectation as the hybrid outlier score $o_i[j] = F(S, j, i)$. Given a time series database $S$ and a hybrid outlier threshold $\delta > 0$, a time point $s_i[j]$ is called a hybrid outlier if the hybrid outlier score $o_i[j] \geq \delta$.

**Example 3.7** (Hybrid Outlier). Assume we have a hybrid outlyingness function, which computes the smallest distance from a time point to the peers in a given set of time series and its neighbors as the outlier degree. Suppose $\delta = 10$ and $w = 1$. For the time series database in Figure 3.2, the smallest distance from time point $s_7[20]$ to the peers ($s_1[20]$, $s_5[20]$) and its neighbors ($s_7[19]$, $s_7[21]$) is 16. Therefore, the outlier score $o_7[20] = 16 > \delta$, and $s_7[20]$ is a hybrid outlier.

In this thesis, we tackle the problem of finding top $N$ hybrid outliers in a time series database.

**Problem Definition.** Given a time series database $S = \{s_i|1 \leq i \leq n\}$ and $N$, where time series $s_i = \{s_i[j]|1 \leq j \leq m\}$, the problem of **hybrid outlier detection on time series**
Table 3.1: The summary of symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$</td>
<td>the time series database</td>
</tr>
<tr>
<td>$s_i$</td>
<td>the $i$-th time series in $S$</td>
</tr>
<tr>
<td>$s_i[j]$</td>
<td>the $j$-th time point in $s_i$</td>
</tr>
<tr>
<td>$n$</td>
<td>the number of time series in $S$</td>
</tr>
<tr>
<td>$m$</td>
<td>the number of time points in a time series</td>
</tr>
<tr>
<td>$s_i[j:e]$</td>
<td>the segment which starts at timestamp $j$ and ends at timestamp $e$ in $s_i$</td>
</tr>
<tr>
<td>$l$</td>
<td>the length of a segment</td>
</tr>
<tr>
<td>$w$</td>
<td>the span of tracking window</td>
</tr>
<tr>
<td>$o_i[j]$</td>
<td>the hybrid outlier score of $s_i[j]$</td>
</tr>
<tr>
<td>$N$</td>
<td>the number of hybrid outliers to be detected</td>
</tr>
</tbody>
</table>

data is to find $N$ hybrid outliers with the highest outlier scores.

Table 3.1 summaries some frequently used symbols in this thesis.

## 3.2 Framework

In this section, a heuristic algorithm framework is proposed to detect hybrid outliers in a time series database. As discussed in Section 3.1, a hybrid outlier has a combination of self-trend outlier behavior and peer-wise outlier behavior. Heuristically, our framework iteratively compares each time point with the neighbors and a set of peers in the locally similar time series to compute the hybrid outlier score.

In Figure 3.3, we show an overview of our framework. Basically, to compute the outlier score of a time point $s_i[j]$, our framework first finds out a set of segments $S' = \{s_{i'}[j - w : j + w]\}$, where $s_{i'}[j - w : j + w]$ is similar to $s_i[j - w : j + w]$ and $i \neq i'$. We refer to $w$ as the tracking window size. Then an intermediate outlier score of $s_i[j]$ is calculated based on the time points in $s_i[j - w : j + w]$ and its peers in $S'$. After that, the distance between segments will be adjusted and $S'$ will be updated accordingly. This process repeats until $S'$ is stable, and the intermediate outlier score in the last iteration is returned as the final outlier score.

To be specific, our framework mainly consists of three iterative phases: grouping phase, outlier score computation phase, and distance weight adjustment phase.
3.2.1 Grouping Phase

As discussed in Section 3.1, the hybrid outlier degree of a time point is affected by its self-trend outlier behavior and peer-wise outlier behavior. For a time point \( s_i[j] \), we compute its outlier score based on its neighbor time points in \( s_i[j - w : j + w] \) and the peer time points in a given set of time series. Because of that, we need to first find out a set of peers of \( s_i[j] \) which \( s_i[j] \) will be compared with. In this thesis, the k-Nearest Neighbors algorithm (or k-NN for short) is applied to get \( k \) segments \( s_{i'}[j - w : j + w] \), denoted as \( KNN(i, j, w) \), which are most similar to \( s_i[j - w : j + w] \) by distance, where \( i \neq i' \). A general way to calculate the k-NN distance \( dist_{knn}(s_i[j : e], s_{i'}[j : e]) \) between two segments \( s_i[j : e] \) and \( s_{i'}[j : e] \) is shown in Equation 3.1.

\[
dist_{knn}(s_i[j : e], s_{i'}[j : e]) = \sum_{t=j}^{e} (s_i[t] - s_{i'}[t])^2
\] (3.1)

3.2.2 Score Computation Phase

After we get the \( KNN(i, j, w) \) of \( s_i[j - w : j + w] \) for time point \( s_i[j] \), its outlier score will be computed based on its neighbors and the peers in \( KNN(i, j, w) \). Different algorithms which apply our framework can specify their own methods to calculate outlier scores. We will develop two such algorithms in Chapter 4 and the detailed methods of how to compute outlier scores will be discussed there.
Table 3.2: Notations used in the algorithm framework

<table>
<thead>
<tr>
<th>Notation</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$</td>
<td>the number of nearest neighbor segments of each $s_i[j - w : j + w]$</td>
</tr>
<tr>
<td>$w_i[j]$</td>
<td>the distance weight of time point $s_i[j]$</td>
</tr>
<tr>
<td>$KNN(i,j,w)$</td>
<td>the k-NN segment set of $s_i[j - w : j + w]$</td>
</tr>
<tr>
<td>$KNN(i,j,w)^*$</td>
<td>the k-NN segment set of $s_i[j - w : j + w]$ in the previous iteration</td>
</tr>
<tr>
<td>$\bar{o}$</td>
<td>the average outlier score of all time points of all time series in $S$</td>
</tr>
<tr>
<td>$topNO$</td>
<td>the hybrid outlier set with top $N$ highest outlier scores</td>
</tr>
</tbody>
</table>

3.2.3 Distance Weight Adjustment Phase

An outlier score $o_i[j]$ is generated for $s_i[j]$ after the first two phases. One may use this score as its final outlier score. However, since the peers in $KNN(i,j,w)$ are only a subset of the peers of $s_i[j]$, it may inaccurately gain a very high outlier score. For example, in Figure 3.2, suppose $w = 4$ and the 2-NN segments of $s_2[10 : 18]$ are $s_3[10 : 18]$ and $s_4[10 : 18]$. The time point $d$ ($s_2[14]$) has a very high outlier score, because it is so different from its neighbors and the peers ($s_3[14], s_4[14]$). However, $s_2[10 : 18]$ is also similar to $s_5[10 : 18]$ and $s_6[10 : 18]$, and $d$ ($s_2[14]$) is not supposed to have a very high outlier score because of its similarity to $h$ ($s_5[14]$) and $i$ ($s_6[14]$).

To get more accurate outlier scores, the k-NN distance between segments will be adjusted iteratively based on the outlier scores. In this way, a time point would have a chance to be compared with its peers in different k-NN segments during the iterations. To achieve that, we assign a distance weight $w_i[j]$ to each time point $s_i[j]$ when calculating the k-NN distance between segments. The formal definition of $w_i[j]$ is as follows.

Definition 3.11 (Distance Weight). Given a time series database $S$ and a time point $s_i[j]$, the distance weight $w_i[j]$ of $s_i[j]$ is defined as

$$w_i[j] = \frac{\bar{o}}{o_i[j]}$$

where $o_i[j]$ is the outlier score of $s_i[j]$, and $\bar{o}$ is the average outlier score of all time points of all time series in $S$, i.e.,

$$\bar{o} = \frac{\sum_i \sum_j o_i[j]}{mn}.$$
The higher outlier score \( o_i[j] \) is, the less distance weight \( w_i[j] \) is. If \( o_i[j] \) is more than \( \bar{o} \), the distance between two points would be shrunk; otherwise, the distance would be enlarged.

**Definition 3.12 (Segment k-NN Distance).** Given two segments \( s_i[j : e] \) and \( s'_i[j : e] \), the k-NN distance between \( s_i[j : e] \) and \( s'_i[j : e] \) is defined as

\[
\text{dist}_{k\text{nn}}(s_i[j : e], s'_i[j : e]) = \sum_{t=j}^{e} (w_i[t]w'_i[t](s_i[t] - s'_i[t]))^2.
\]

The length of k-NN segment is \( 2w + 1 \). The window size \( w \) affects the similarity of two segments \( s_i[j - w : j + w] \) and \( s'_i[j - w : j + w] \) for each time point \( s_i[j] \). With \( w \) increasing, the similarity will be determined by more time points in \( s_i \) and \( s'_i \), and \( s_i[j - w : j + w] \) will have more globally similar trends with the k-NN segments, i.e., the time point \( s_i[j] \) will be compared with the peers in the more globally similar time series. When \( w \) decreases, \( s_i[j - w : j + w] \) and the k-NN segments are more locally similar.

The three phases of our framework, i.e., grouping phase, outlier score computation phase, and distance weight adjustment phase, will repeat until every \( KNN(i, j, w) \) is equal to \( KNN(i, j, w)^* \), where \( KNN(i, j, w)^* \) is the k-NN segment set of \( s_i[j - w : j + w] \) in the previous iteration. □

Algorithm 1 shows the pseudo code of the framework for detecting hybrid outliers in a time series database. Table 3.2 lists the symbols used in the framework for ease of presentation. Algorithm 1 iteratively computes the k-NN segments according to Definition 3.12, calculates outlier scores of all time points in all time series based on the difference from them to their neighbors and a set of their peers, and adjusts distance weight for every time point using Definition 3.11. The outlier score of each time point may change with the new k-NN in each iteration, and the outlier score in each iteration is the minimal one so far. Finally top \( N \) hybrid outliers can be identified according to their outlier scores.

**Example 3.8 (Algorithm Framework).** A time series database \( S \) which consists of 7 time series \( s_1, s_2, \ldots, s_7 \) is shown in Figure 3.2. Suppose \( k = 2, N = 1 \) and \( w = 4 \), we detect the top 1 hybrid outlier in \( S \).

We first get the initial 2-NN segments of \( s_i[j - w : j + w] \) for each \( s_i[j] \) by segment k-NN distances, as shown in the second column of Table 3.3. Then an intermediate outlier score of each time point \( s_i[j] \) is calculated as the second smallest distance from it to its neighbors and the peers in 2-NN segments. For instance, \( d(s_2[14]) \) will be compared with the time points.
Algorithm 1: Hybrid Outlier Detection Algorithm Framework

**Input:** $S; N; k; w.$

**Output:** topNO

1. Normalize each $s_i \in S$ if necessary by ZNormalization;

2. foreach $s_i \in S$ do

3.     foreach $s_{i[j]} \in s_i$ do

4.         $w_{i[j]} \leftarrow 1$;

5.         Initialize $KNN(i, j, w)$ and $KNN(i, j, w)^*$;

6.     do

7.     foreach $s_i \in S$ do

8.         foreach $s_{i[j]} \in s_i$ do

9.             $KNN(i, j, w)^* \leftarrow KNN(i, j, w)$;

10.            Compute $KNN(i, j, w)$ based on Definition 3.12;

11.     foreach $s_i \in S$ do

12.         foreach $s_{i[j]} \in s_i$ do

13.             Compute $o_{i[j]}$;

14.      Calculate $\bar{o}$;

15.     foreach $s_i \in S$ do

16.         foreach $s_{i[j]} \in s_i$ do

17.             Calculate $w_{i[j]}$ by Definition 3.11;

18. while any $KNN(i, j, w) \neq KNN(i, j, w)^*$;

19. Compute topNO based on the final outlier scores;

20. return topNO;

in $s_2[10:18]$ except for itself and the time points ($s_3[14], s_4[14]$). Since $d$ deviates greatly from those time points, it has a very high outlier score in this iteration. The second row of Table 3.4 shows the rank of time points ordered by their outlier scores in the first iteration, where $d$ is the top 1 outlier. When the first iteration is done, the distance weight of each time point will be calculated according to the intermediate outlier score by Definition 3.11. Since $d$ has the highest outlier score, its distance weight will be the smallest.

In the second iteration, with the change of distance weight, the $k$-NN distance between segments is adjusted by Definition 3.12, and each 2-NN segment set is updated accordingly as shown in the third column of Table 3.3. The outlier score of each time point is then recalculated based on its neighbors and the peers in the new 2-NN segments. The third row of Table 3.4 shows the rank of time points ordered by their outlier scores in the second iteration. Interestingly, the 2-NN segments of $s_2[10:18]$ for time point $d$ ($s_2[14]$) change to be $s_5[10:18]$ and $s_6[10:18]$, and its outlier score goes down due to its similarity to $h$ and
Table 3.3: 2-NN segments during iterations

<table>
<thead>
<tr>
<th>Time Point</th>
<th>1st Iteration</th>
<th>2nd Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a ,(s_1[5])$</td>
<td>$s_3[1 : 9], s_5[1 : 9]$</td>
<td>$s_3[1 : 9], s_7[1 : 9]$</td>
</tr>
<tr>
<td>$b ,(s_1[7])$</td>
<td>$s_3[3 : 11], s_5[3 : 11]$</td>
<td>$s_3[3 : 11], s_7[3 : 11]$</td>
</tr>
<tr>
<td>$c ,(s_1[9])$</td>
<td>$s_3[5 : 13], s_4[5 : 13]$</td>
<td>$s_3[5 : 13], s_7[5 : 13]$</td>
</tr>
<tr>
<td>$d ,(s_2[14])$</td>
<td>$s_3[10 : 18], s_4[10 : 18]$</td>
<td>$s_5[10 : 18], s_6[10 : 18]$</td>
</tr>
<tr>
<td>$g ,(s_4[17])$</td>
<td>$s_2[13 : 21], s_3[13 : 21]$</td>
<td>$s_1[13 : 21], s_4[13 : 21]$</td>
</tr>
<tr>
<td>$h ,(s_5[14])$</td>
<td>$s_1[10 : 18], s_6[10 : 18]$</td>
<td>$s_2[10 : 18], s_6[10 : 18]$</td>
</tr>
<tr>
<td>$i ,(s_6[14])$</td>
<td>$s_1[10 : 18], s_5[10 : 18]$</td>
<td>$s_2[10 : 18], s_5[10 : 18]$</td>
</tr>
<tr>
<td>$j ,(s_7[20])$</td>
<td>$s_1[16 : 24], s_5[16 : 24]$</td>
<td>$s_1[16 : 24], s_4[16 : 24]$</td>
</tr>
</tbody>
</table>

Table 3.4: The outlier degree rank of time points during iterations

<table>
<thead>
<tr>
<th>Rank</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Iteration</td>
<td>$d$</td>
<td>$h$</td>
<td>$i$</td>
<td>$j$</td>
<td>$a$</td>
<td>$c$</td>
<td>$b$</td>
<td>$f$</td>
<td>$e$</td>
<td>$g$</td>
</tr>
<tr>
<td>2nd Iteration</td>
<td>$j$</td>
<td>$a$</td>
<td>$c$</td>
<td>$b$</td>
<td>$h$</td>
<td>$i$</td>
<td>$d$</td>
<td>$f$</td>
<td>$e$</td>
<td>$g$</td>
</tr>
</tbody>
</table>

i. After the second iteration, the algorithm terminates because there are no more changes in 2-NNs, and $j$ is returned as the top 1 hybrid outlier.

The algorithms for detecting hybrid outliers in a time series database can apply this framework by specifying the outlier score calculation methods. To show the feasibility of our framework, two algorithms based on the framework are developed in Chapter 4.

3.2.4 Convergence

Since our framework is iterative, it is necessary to discuss its property of convergence.

Lemma 3.1. Let $KNN_t = \langle KNN(1,1,w), \ldots, KNN(1,m,w), KNN(2,1,w), \ldots, KNN(2,m,w), \ldots, KNN(n,1,w), \ldots, KNN(n,m,w) \rangle$ be the vector of k-NN segment sets for all time points in the $t$-th iteration. $O_t = \langle o_1[1], \ldots, o_1[m], o_2[1], \ldots, o_2[m], \ldots, o_n[1], \ldots, o_n[m] \rangle$ denotes the vector of outlier scores for all time points in the $t$-th iteration. If $p < q$ and $KNN_p = KNN_q$, then $O_q = O_{q-1}$.

Proof. We prove the lemma by contradiction. Assume $O_q \neq O_{q-1}$ when $p < q$ and $KNN_p = KNN_q$. $o_t[j]^t$ denotes the outlier score of time point $s_t[j]$ in the $t$-th iteration. In this case,
there must exist a time point \( s_i[j] \) whose outlier score \( o_i[j]^q \) in the \( q \)-th iteration is not equal to the outlier score \( o_i[j]^{(q-1)} \) in the \( (q-1) \)-th iteration, i.e., \( o_i[j]^q \neq o_i[j]^{(q-1)} \). Because the outlier score in each iteration is the minimal one so far and \( p \leq (q-1) \), we can get \( o_i[j]^p \geq o_i[j]^{(q-1)} \). Since \( KNN_p = KNN_q \), \( i.e., \) the KNN segment set for time point \( s_i[j] \) in the \( p \)-th iteration is the same as that in the \( q \)-th iteration, \( o_i'[j]^q = o_i'[j]^p \geq o_i[j]^p \), where \( o_i'[j]^q \) is the intermediate outlier score computed according to the peers in \( KNN(i, j, w) \) and the neighbors in the \( t \)-th iteration. Since \( o_i[j]^q = \min(o_i'[j]^q, o_i[j]^{(q-1)}) \), \( o_i'[j]^q \geq o_i[j]^p \), and \( o_i[j]^p \geq o_i[j]^{(q-1)} \), we can get \( o_i[j]^q = o_i[j]^{(q-1)} \), which is a contradiction.

**Lemma 3.2.** If \( O_q = O_{q-1} \), then the framework will terminate in the \((q+1)\)-th iteration.

**Proof.** By definition, we have

\[
\text{dist}_{\text{knn}}(s_i[j : e], s_{i'}[j : e]) = \sum_{t=j}^{e} (w_i[t]w_{i'}[t](s_i[t] - s_{i'}[t]))^2,
\]

\[
w_i[j] = \frac{\bar{o}}{o_i[j]},
\]

\[
\bar{o} = \frac{\sum_{i}^{n} \sum_{j}^{m} o_i[j]}{mn}.
\]

Using the above three equations, we have

\[
\text{dist}_{\text{knn}}(s_i[j : e], s_{i'}[j : e]) = \sum_{t=j}^{e} \left( \frac{\left( \sum_{i}^{n} \sum_{j}^{m} o_i[j] \right)^2}{m^2n^2o_i[t]o_{i'}[t](s_i[t] - s_{i'}[t])} \right)^2.
\]

Since \( O_q = O_{q-1} \), \( i.e., \) \( o_i[j]^q = o_i[j]^{(q-1)} \), every \( KNN(i, j, w) \) in the \((q+1)\)-th iteration will be the same as that in the \( q \)-th iteration based on the above equation. Our framework will terminate in the \((q+1)\)-th iteration according to the termination condition of the framework.

**Lemma 3.3.** Given a time series database \( S \) which consists of \( n \) time series of length \( m \), and let \( KNN_t = \langle KNN(1,1,w), \ldots, KNN(1,m,w), KNN(2,1,w), \ldots, KNN(2,m,w), \ldots, KNN(n,1,w), \ldots, KNN(n,m,w) \rangle \) be the vector of k-NN segment sets for all time points in the \( t \)-th iteration, the possible number of distinct \( KNN_t \) is \((C_n^k)^{mn}\).
CHAPTER 3. PROBLEM DEFINITION AND FRAMEWORK

Proof. There are \( C_{n-1}^k \) possible different k-NN segment sets \( KNN(i,j,w) \) for each time point \( s_i[j] \). Since there are \( m \times n \) time points in total, the possible number of distinct \( KNN_t \) is \( (C_{n-1}^k)^{mn} \). \qed

Theorem 3.1. After at most \( (C_{n-1}^k)^{mn} + 2 \) iterations, our framework terminates.

Proof. According to Lemma 3.3, there must exist two iterations \( p \) and \( q \) such that \( KNN_p = KNN_q \) and \( 1 \leq p < q \leq (C_{n-1}^k)^{mn} + 1 \). Then, by Lemma 3.1, we can get \( O_q = O_{q-1} \). In this way, the framework will terminate in the \((q + 1)\)-th iteration by Lemma 3.2. Since \( q + 1 \leq (C_{n-1}^k)^{mn} + 2 \), our framework terminates after at most \( (C_{n-1}^k)^{mn} + 2 \) iterations. \qed

Corollary 3.1. Since \( m, n, k \) are finite, the upper bound on the number of iterations \( (C_{n-1}^k)^{mn} + 2 \) is finite. As a result, the framework must converge in a finite number of iterations.
Chapter 4

Two Algorithms

In Section 3.2 we propose a heuristic framework to detect hybrid outliers. In this chapter, two algorithms based on the framework are developed by specifying outlier score computation methods. The implementation of those two algorithms verifies the feasibility of our framework.

In Section 4.1, a distance-based algorithm which calculates outlier scores according to distances between time points is presented. We also propose a prediction-based algorithm using the deviation of the actual value of a time point from its predicted value as the outlier score in Section 4.2. In Section 4.3, we use a small data example to further describe our two algorithms.

4.1 The Distance-based Algorithm (DA)

Basically, to get the outlier score of a time point $s_i[j]$, the distance-based algorithm (or DA for short) calculates the distances from $s_i[j]$ to the following time points.

- The neighbors of $s_i[j]$, i.e., the time points in $s_i[j - w : j + w]$ except for $s_i[j]$ itself.
- The peers of $s_i[j]$ in $KNN(i, j, w)$.

In the DA algorithm, the distance between two time points is the absolute value of their difference, which is defined as:

$$\text{dist}(s_i[j], s_i'[j']) = |s_i[j] - s_i'[j']|. \quad (4.1)$$

We formally define the hybrid outlier score in the DA algorithm as follows.
### Table 4.1: Notations used in the distance-based algorithm

<table>
<thead>
<tr>
<th>Notation</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$KNN(i, j, w)$</td>
<td>the k-NN segment set of $s_i[j - w : j + w]$</td>
</tr>
<tr>
<td>$r^{th} \min_i[j]$</td>
<td>the $r$-th smallest distance from $s_i[j]$ to its peers in $KNN(i, j, w)$ and the neighbors</td>
</tr>
<tr>
<td>$dist(s_i[j], s_i'[j'])$</td>
<td>the distance between two time points $s_i[j]$ and $s_i'[j']$</td>
</tr>
</tbody>
</table>

### Table 4.2: Three time series

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_1$</td>
<td>11</td>
<td>12</td>
<td>12</td>
<td>19</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>$s_2$</td>
<td>12</td>
<td>11</td>
<td>10</td>
<td>11</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>$s_3$</td>
<td>13</td>
<td>11</td>
<td>11</td>
<td>12</td>
<td>11</td>
<td>13</td>
</tr>
</tbody>
</table>

**Definition 4.1** (Hybrid Outlier Score in the DA Algorithm). Given a time series database $S$, a time point $s_i[j]$, the k-NN segments $KNN(i, j, w)$, and a parameter $1 \leq r \leq 2w + k$, the outlier score $o_i[j]$ of $s_i[j]$ in DA is defined as

$$o_i[j] = r^{th} \min_i[j],$$

where $r^{th} \min_i[j]$ is the $r$-th smallest distance from $s_i[j]$ to its peers in $KNN(i, j, w)$ and the neighbors in $s_i$ [23] [30] [5] [27] [7].

The outlier score computation method in DA is shown in Algorithm 2, and Table 4.1 lists the symbols used in the DA algorithm.

**Example 4.1** (Outlier Score Computation in DA). In Table 4.2, there are three time series. Suppose $k = 2$, $w = 1$, $r = 2$, and $KNN(1, 4, 1) = \{s_2[3 : 5], s_3[3 : 5]\}$, we compute the outlier score $o_1[4]$ of time point $s_1[4]$.

First we calculate the distances from $s_1[4]$ to its neighbors $(s_1[3], s_1[5])$ by Equation 4.1,

$$dist(s_1[4], s_1[3]) = |19 - 12| = 7,$$

$$dist(s_1[4], s_1[5]) = |19 - 11| = 8.$$

Then, the distances from $s_1[4]$ to the peers $(s_2[4], s_3[4])$ are calculated,

$$dist(s_1[4], s_2[4]) = |19 - 11| = 8,$$
Algorithm 2: Outlier Score Calculation Method in DA

\begin{algorithm}
\begin{algorithmic}[1]
\State \textbf{Input}: $s_i[j]$; $s_i$; $KNN(i, j, w)$; $w$; $r$.
\State \textbf{Output}: $o_i[j]$.
\State $r^{th}\text{min}_i[j] \leftarrow +\infty$;
\ForEach{$s_i[j'] \in s_i$}
\If{$1 \leq |j - j'| \leq w$}
\State Compute $dist(s_i[j], s_i[j'])$ by Equation 4.1;
\If{$dist(s_i[j], s_i[j']) < r^{th}\text{min}_i[j]$}
\State Update $r^{th}\text{min}_i[j]$;
\EndIf
\EndIf
\EndFor
\EndFor
\State $o_i[j] \leftarrow r^{th}\text{min}_i[j]$;
\Return $o_i[j]$;
\end{algorithmic}
\end{algorithm}

dist($s_1[4], s_3[4]$) = $|19 - 12| = 7$.

Thus, the 2nd smallest distance from $s_1[4]$ to its peers in $KNN(1, 4, 1)$ and the neighbors is $2^{th}\text{min}_1[4] = 7$, i.e., $o_1[4] = 7$.

4.2 The Prediction-based Algorithm (PA)

The prediction-based algorithm (or PA for short) predicts the value of a time point $s_i[j]$ by applying the least squares technique for multi-variate linear regression [29], and computes the deviation of the actual value $s_i[j]$ from its predicted value $s_i[j]^*$ as its outlier score. Table 4.3 gives a list of notations used in this section.

For a time point $s_i[j]$ with $w < j \leq m - w$, where $w$ is a parameter that we refer to as the tracking window size and $m$ is the length of time series $s_i$, the PA algorithm estimates its value by using two sources of information.

- The neighbors of $s_i[j]$, i.e., $s_i[j - w]$, $\ldots$, $s_i[j - 1]$, $s_i[j + 1]$, $\ldots$, $s_i[j + w]$.

- Its peers in $KNN(i, j, w)$. 

CHAPTER 4. TWO ALGORITHMS

Table 4.3: Notations used in the prediction-based algorithm

<table>
<thead>
<tr>
<th>Notation</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_i[j]^*$</td>
<td>the predicted value of time point $s_i[j]$</td>
</tr>
<tr>
<td>$KNN(i,j,w)$</td>
<td>the k-NN segment set of $s_i[j-w:j+w]$</td>
</tr>
<tr>
<td>$v$</td>
<td>the number of independent variables in multi-variate regression</td>
</tr>
<tr>
<td>$A_i$</td>
<td>the optimal regression coefficient matrix of time series $s_i$</td>
</tr>
<tr>
<td>$X_i$</td>
<td>the matrix of independent variables for $s_i$, each row is sample values of the corresponding independent variables</td>
</tr>
<tr>
<td>$Y_i$</td>
<td>the column vector of desired values for $s_i$</td>
</tr>
</tbody>
</table>

For a time point $s_i[j]$ with $j \leq w$, its value is estimated based on the time points in $s_i[1:2w+1]$ except for $s_i[j]$ itself and its peers in $KNN(i,w+1,w)$; for a time point $s_i[j]$ with $j > m - w$, we predict its value according to the time points in $s_i[m-2w:m]$ except for $s_i[j]$ and the peers in $KNN(i,m-w,w)$. To be specific, we try to predict the value of $s_i[j]$ as a linear combination of its neighbor time points and the peer time points in $KNN(i,j,w) = \{s_i[j-w,j+w], s_i[j-w,j+w], \ldots, s_i[j-w,j+w]\}$. Mathematically, we have the following equation for $w < j \leq m-w$:

$$s_i[j]^* = a_1s_i[j-1] + a_2s_i[j-2] + \cdots + a_ws_i[j-w] + a_{w+1}s_i[j+1] + a_{w+2}s_i[j+2] + \cdots + a_{2w}s_i[j+w] + a_{2w+1}s_i[j] + a_{2w+2}s_i[j] + \cdots + a_{2w+k}s_{ik}[j]$$  \hspace{1cm} (4.2)

Equation 4.2 is a collection of linear equations for $j = w + 1, \ldots, m - w$, with $s_i[j]$ being the dependent variable, $s_i[j-1], \ldots, s_i[j-w], s_i[j+1], \ldots, s_i[j+w], s_{i_1}[j], s_{i_2}[j], \ldots, s_{ik}[j]$ the independent variables. The number of independent variables is

$$v = 2w + k.$$  \hspace{1cm} (4.3)

Each $a_x$ is called a regression coefficient. The least square solution, i.e., regression coefficients which minimize the sum of squares errors

$$SSE = \sum_{j=1}^{m} (s_i[j] - s_i[j]^*)^2,$$

is given by the multi-variate regression [14]. With this set up, the optimal regression coefficients $A_i$ for time series $s_i$ are given by [8]

$$A_i = (X_i^T \times X_i)^{-1} \times (X_i^T \times Y_i),$$  \hspace{1cm} (4.4)
Algorithm 3: Outlier Score Calculation Method in PA

**Input:** $s_i[j]; s_i; \text{KNN}(i, j, w); w.$

**Output:** $o_i[j].$

1. if $j = 1$ then
2. Set $X_i;$
3. Set $Y_i;$
4. Calculate $A_i$ by Equation 4.4;
5. Calculate $s_i[j]^*$ by Equation 4.5;
6. $o_i[j] \leftarrow |s_i[j] - s_i[j]^*|;$
7. return $o_i[j].$

where $X_i$ is an $m \times v$ matrix, the $j$-th row of the matrix $X_i$ consists of sample values of the corresponding independent variable for $s_i[j]$ in Equation 4.2. $Y_i$ is a column vector of actual values $s_i[j]$, i.e., $Y_i$ is an $m \times 1$ matrix. $A_i$ is a $v \times 1$ matrix.

Thus, $s_i[j]^*$ is computed based on Equation 4.2 as follows.

$$s_i[j]^* = \sum_{c=1}^{v} X_i[j][c] A_i[c]. \quad (4.5)$$

**Definition 4.2** (Hybrid Outlier Score in the PA Algorithm). *For a time point $s_i[j]$, the outlier score $o_i[j]$ of $s_i[j]$ in PA is defined as

$$o_i[j] = |s_i[j] - s_i[j]^*|,$$

where $s_i[j]^*$ is the predicted value of $s_i[j].$*

The method for computing outlier scores in the PA algorithm is presented in Algorithm 3. For all time points in time series $s_i$, we only need to calculate $A_i$ once, i.e., it is only calculated when we compute the predicted value of $s_i[1]$. The detailed computation of the outlier score of a time point is shown in Example 4.2.

**Example 4.2** (Outlier Score Computation in PA). *Let us consider the same time series as in Example 4.1, and we compute the outlier score $o_1[4]$ of time point $s_1[4]$. The span of tracking window size $w$ is 1, and the 2-NN segments of each $s_1[j-1, j+1]$ are $s_2[j-1, j+1]$ and $s_3[j-1, j+1].$ Since $w = 1$ and $k = 2$, the number of independent variables for each time point is $v = 2 \times 1 + 2 = 4$ by Equation 4.3. First, the matrix $X_1$ is set, each row of $X_1$ consists of*
the sample values of the corresponding independent variable for each time point in \( s_1 \). For example, the independent variables for \( s_{1[4]} \) are in the 4-th row of \( X_1 \), where \( X_1 \) is

\[
X_1 = \begin{bmatrix} 12 & 12 & 12 & 13 \\ 11 & 12 & 11 & 11 \\ 12 & 19 & 10 & 11 \\ 12 & 11 & 11 & 12 \\ 19 & 12 & 10 & 11 \\ 19 & 11 & 12 & 13 \end{bmatrix}.
\]

The matrix \( Y_1 \) is also set, which actually is a column vector of real values of time points in \( s_1 \), shown as following:

\[
Y_1 = \begin{bmatrix} 11 \\ 12 \\ 12 \\ 19 \\ 11 \\ 12 \end{bmatrix}.
\]

Then we compute the optimal regression coefficients \( A_1 \) by Equation 4.4 and get

\[
A_1 = \begin{bmatrix} -0.27 \\ -0.02 \\ -1.21 \\ 2.55 \end{bmatrix}.
\]

Finally, based on \( A_1 \) and the 4-th row of \( X_1 \), the predicted value of \( s_{1[4]} \) is

\[
s_{1[4]}^* = 12 \cdot (-0.27) + 11 \cdot (-0.02) + 11 \cdot (-1.21) + 12 \cdot (2.55) = 13.83
\]

by Equation 4.5. Therefore, the outlier score of \( s_{1[4]} \) is \( o_{1[4]} = |19 - 13.83| = 5.17 \).

### 4.3 Example

A time series database which consists of 7 time series \( s_1, s_2, \ldots, s_7 \) is shown in Figure 3.2. Each time series has 24 time points, where \( a = 35, b = 33, c = 32, d = 34, e = -42, f = \ldots \)
Table 4.4: 2-NN segments during iterations for DA

<table>
<thead>
<tr>
<th>Time Point</th>
<th>1st Iteration</th>
<th>2nd Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>a (s₁[5])</td>
<td>s₃[1 : 9], s₅[1 : 9]</td>
<td>s₃[1 : 9], s₇[1 : 9]</td>
</tr>
<tr>
<td>b (s₁[7])</td>
<td>s₃[3 : 11], s₅[3 : 11]</td>
<td>s₃[3 : 11], s₇[3 : 11]</td>
</tr>
<tr>
<td>c (s₁[9])</td>
<td>s₃[5 : 13], s₄[5 : 13]</td>
<td>s₃[5 : 13], s₇[5 : 13]</td>
</tr>
<tr>
<td>d (s₂[14])</td>
<td>s₃[10 : 18], s₄[10 : 18]</td>
<td>s₅[10 : 18], s₆[10 : 18]</td>
</tr>
<tr>
<td>h (s₅[14])</td>
<td>s₁[10 : 18], s₆[10 : 18]</td>
<td>s₂[10 : 18], s₆[10 : 18]</td>
</tr>
<tr>
<td>i (s₆[14])</td>
<td>s₁[10 : 18], s₅[10 : 18]</td>
<td>s₂[10 : 18], s₅[10 : 18]</td>
</tr>
<tr>
<td>j (s₇[20])</td>
<td>s₁[16 : 24], s₅[16 : 24]</td>
<td>s₁[16 : 24], s₄[16 : 24]</td>
</tr>
</tbody>
</table>

Table 4.5: Outlier score of each time point during iterations for DA

<table>
<thead>
<tr>
<th>Time Point</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
<th>i</th>
<th>j</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Iteration</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>33</td>
<td>1</td>
<td>1</td>
<td>33</td>
<td>30</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>2nd Iteration</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>17</td>
<td></td>
</tr>
</tbody>
</table>

-42, g = -43, h = 35, i = 34, j = 20 are 10 time points among the 168 time points. We apply the DA and PA algorithms to iteratively compute the outlier scores of the time points, and the results are presented in Subsection 4.3.1 and Subsection 4.3.2, respectively.

### 4.3.1 The DA Result

Set \( k = 2, r = 2, \) and \( w = 4. \) Initially, the outlier score \( o_{i}[j] \) of each time point \( s_{i}[j] \) is set to infinity and the distance weight \( w_{i}[j] \) is 1. We compute \( KNN(i, j, 4), o_{i}[j], \) and \( w_{i}[j] \) iteratively.

- **The 1st iteration**
  
  After calculating the k-NN distances between segments by Definition 3.12, we get the

Table 4.6: Distance weight of each time point during iterations for DA

<table>
<thead>
<tr>
<th>Time Point</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
<th>i</th>
<th>j</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Iteration</td>
<td>0.43</td>
<td>0.65</td>
<td>0.43</td>
<td>0.04</td>
<td>1.29</td>
<td>1.29</td>
<td>1.29</td>
<td>0.04</td>
<td>0.04</td>
<td>0.08</td>
</tr>
<tr>
<td>2nd Iteration</td>
<td>0.20</td>
<td>0.31</td>
<td>0.20</td>
<td>0.62</td>
<td>0.62</td>
<td>0.62</td>
<td>0.62</td>
<td>0.62</td>
<td>0.04</td>
<td></td>
</tr>
</tbody>
</table>
Table 4.7: Final outlier degree rank of time points for DA

<table>
<thead>
<tr>
<th>Rank</th>
<th>Time Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>j</td>
</tr>
<tr>
<td>2</td>
<td>a</td>
</tr>
<tr>
<td>3</td>
<td>c</td>
</tr>
<tr>
<td>4</td>
<td>b</td>
</tr>
<tr>
<td>5</td>
<td>h</td>
</tr>
<tr>
<td>6</td>
<td>i</td>
</tr>
<tr>
<td>7</td>
<td>d</td>
</tr>
<tr>
<td>8</td>
<td>f</td>
</tr>
<tr>
<td>9</td>
<td>e</td>
</tr>
<tr>
<td>10</td>
<td>g</td>
</tr>
</tbody>
</table>

Table 4.8: 2-NN segments during iterations for PA

<table>
<thead>
<tr>
<th>Time Point</th>
<th>1st Iteration</th>
<th>2nd Iteration</th>
<th>3rd Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>a (s₁[5])</td>
<td>s₃[2 : 8], s₅[2 : 8]</td>
<td>s₅[2 : 8], s₆[2 : 8]</td>
<td>s₅[2 : 8], s₆[2 : 8]</td>
</tr>
<tr>
<td>b (s₁[7])</td>
<td>s₃[4 : 10], s₅[4 : 10]</td>
<td>s₅[4 : 10], s₆[4 : 10]</td>
<td>s₅[4 : 10], s₆[4 : 10]</td>
</tr>
<tr>
<td>c (s₁[9])</td>
<td>s₃[6 : 12], s₅[6 : 12]</td>
<td>s₂[6 : 12], s₆[6 : 12]</td>
<td>s₂[6 : 12], s₆[6 : 12]</td>
</tr>
<tr>
<td>e (s₂[17])</td>
<td>s₃[14 : 20], s₄[14 : 20]</td>
<td>s₁[14 : 20], s₆[14 : 20]</td>
<td>s₁[14 : 20], s₆[14 : 20]</td>
</tr>
<tr>
<td>f (s₃[17])</td>
<td>s₂[14 : 20], s₄[14 : 20]</td>
<td>s₂[14 : 20], s₄[14 : 20]</td>
<td>s₂[14 : 20], s₄[14 : 20]</td>
</tr>
<tr>
<td>g (s₄[17])</td>
<td>s₂[14 : 20], s₃[14 : 20]</td>
<td>s₂[14 : 20], s₃[14 : 20]</td>
<td>s₂[14 : 20], s₃[14 : 20]</td>
</tr>
<tr>
<td>j (s₇[20])</td>
<td>s₁[17 : 23], s₅[17 : 23]</td>
<td>s₅[17 : 23], s₆[17 : 23]</td>
<td>s₅[17 : 23], s₆[17 : 23]</td>
</tr>
</tbody>
</table>

2-NN segments of sᵢ[ j – w : j + w] for each sᵢ[ j], as shown in the second column of Table 4.4. The second row of Table 4.5 demonstrates the outlier scores of a, b, . . . , j calculated based on the distances from each time point to its peers in KNN(i, j, 4) and the neighbors. For instance, the outlier score of a (s₁[5]) is the 2nd smallest distance from it to the time points in s₁[1 : 9] except for itself and the time points (s₃[5], s₅[5]), i.e., o₁[5] = 3. The average outlier score of all time points of all time series is 1.29 in this iteration. Then the distance weight is adjusted by Definition 3.11, as shown in the second row of Table 4.6.

- The 2nd iteration

Table 4.9: Outlier scores of time points during iterations for PA

<table>
<thead>
<tr>
<th>Time Point</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
<th>i</th>
<th>j</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Iteration</td>
<td>0.05</td>
<td>0.02</td>
<td>0.03</td>
<td>3.74</td>
<td>0.3</td>
<td>0.03</td>
<td>0.01</td>
<td>0.14</td>
<td>0.03</td>
<td>9.88</td>
</tr>
<tr>
<td>2nd Iteration</td>
<td>0.05</td>
<td>0.02</td>
<td>0.03</td>
<td>0.25</td>
<td>0.3</td>
<td>0.03</td>
<td>0.01</td>
<td>0.14</td>
<td>0.02</td>
<td>8.33</td>
</tr>
<tr>
<td>3rd Iteration</td>
<td>0.05</td>
<td>0.02</td>
<td>0.03</td>
<td>0.075</td>
<td>0.3</td>
<td>0.03</td>
<td>0.01</td>
<td>0.01</td>
<td>0.02</td>
<td>8.33</td>
</tr>
</tbody>
</table>
Table 4.10: Distance weight of time points during iterations for PA

<table>
<thead>
<tr>
<th>Time Point</th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
<th>(e)</th>
<th>(f)</th>
<th>(g)</th>
<th>(h)</th>
<th>(i)</th>
<th>(j)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Iteration</td>
<td>9.2</td>
<td>23</td>
<td>15.33</td>
<td>0.12</td>
<td>1.53</td>
<td>15.33</td>
<td>46</td>
<td>3.29</td>
<td>15.33</td>
<td>0.05</td>
</tr>
<tr>
<td>2nd Iteration</td>
<td>7.8</td>
<td>19.5</td>
<td>13</td>
<td>1.56</td>
<td>1.3</td>
<td>13</td>
<td>39</td>
<td>2.79</td>
<td>19.5</td>
<td>0.05</td>
</tr>
<tr>
<td>3rd Iteration</td>
<td>7.2</td>
<td>18</td>
<td>12</td>
<td>4.8</td>
<td>1.2</td>
<td>12</td>
<td>36</td>
<td>36</td>
<td>18</td>
<td>0.04</td>
</tr>
</tbody>
</table>

Table 4.11: Average score of all time point in all time series during iterations for PA

<table>
<thead>
<tr>
<th></th>
<th>1st Iteration</th>
<th>2nd Iteration</th>
<th>3rd Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\bar{o})</td>
<td>0.46</td>
<td>0.39</td>
<td>0.36</td>
</tr>
</tbody>
</table>

According to the distance weight of time points in the first iteration, the 2-NN segments are updated, shown in the third column of Table 4.4. The outlier score of each time point is then recalculated based on its neighbors and the peers in the new 2-NN segments, as shown in the third row of Table 4.5. The third row of Table 4.6 shows the recalculated distance weight. In addition, the average outlier score is 0.62.

After two iterations, there are no more changes in all 2-NN segment sets, and the outlier scores in the 2nd iteration are returned as the final outlier scores. Table 4.7 shows the final rank of time points ordered by their outlier scores in the DA algorithm. From the final result, we can see that our algorithm indeed finds out the outlier point \(j\) which is not only different from its neighbors, but also different from the peers.

The result also shows how our algorithm can get more accurate outlier scores by adjusting distance weight in each iteration. For example, in the first iteration, the 2-NN segments of \(s_2[10:18]\) are \(s_3[10:18]\) and \(s_4[10:18]\). Time point \(d\) \((s_2[14])\) gets a very high outlier score, because \(d\) deviates remarkably from the time points in \(s_2[10:18]\) except for itself and the peers \((s_3[14], s_4[14])\). However, after the distance weight adjustment, the 2-NN segments become \(s_5[10:18]\) and \(s_6[10:18]\) in the second iteration, and the outlier score of \(d\) drops because it is similar to the time points \(h\) \((s_5[14])\) and \(i\) \((s_6[14])\).

4.3.2 The PA Result

Set \(k = 2, w = 3\). Similar to the DA algorithm, we compute \(KNN(i, j, 3), o_i[j], \) and \(w_i[j]\) iteratively in the PA algorithm. There are 3 iterations for PA. The results of \(KNN(i, j, 3), o_i[j], w_i[j]\), and \(\bar{o}\) during iterations are shown in Tables 4.8, 4.9, 4.10, and 4.11, respectively. \(\bar{o}\) is the average outlier score of all time points of all time series. Table 4.12 shows the final
Table 4.12: Final outlier degree rank of time points for PA

<table>
<thead>
<tr>
<th>Rank</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Point</td>
<td>j</td>
<td>e</td>
<td>d</td>
<td>a</td>
<td>c</td>
<td>f</td>
<td>b</td>
<td>i</td>
<td>g</td>
<td>h</td>
</tr>
</tbody>
</table>

outlier degree rank of time points in our PA algorithm. The outlier point $j$ is also identified by our PA algorithm.
Chapter 5

Experiments

To validate the effectiveness and efficiency of our two algorithms named DA and PA, we conducted experiments on a real data set and a series of synthetic data sets. The experimental results on a real currency exchange rate data set are reported in Section 5.1, and we present the results on synthetic data sets in Section 5.2.

All methods were implemented in Java and all experiments were conducted on a PC computer with an Intel Core i5-2400 3.10GHz CPU and 4GB main memory running 64-bit CentOS 6.5.

We cannot identify any existing method that solves the exact same problem. The focus of our methods is to find hybrid outliers on time series data. Consequently, this thesis does not intend to compete with the existing methods.

5.1 Results on a Real Data Set

We obtained an exchange rate data set from the Pacific Exchange Rate Service of University of British Columbia\(^1\). The data set consists of monthly exchange rates of 62 currencies w.r.t. USD from 1974 to 2013. We normalized the data set by Znormalization [18].

As a case study, Figure 5.2 shows 7 currency time series in 2009-2013, and they are CAD, EUR, GBP, JPY, HKD, CHF, and RUB. A time point is denoted as the currency name concatenating its timestamp. For instance, the exchange rate of CAD in January 2009 is denoted as CAD2009.01. We set \(k = 2\), \(w = 6\), and \(r = 2\) for DA, and \(k = 2\) and \(w = 6\)

\(^1\)http://fx.sauder.ubc.ca/
Table 5.1: Top 10 hybrid outliers detected by our two algorithms

<table>
<thead>
<tr>
<th></th>
<th>DA</th>
<th>PA</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHF2011.08</td>
<td>RUB2009.02</td>
<td></td>
</tr>
<tr>
<td>GBP2009.05</td>
<td>GBP2013.03</td>
<td></td>
</tr>
<tr>
<td>JPY2013.01</td>
<td>CHF2011.08</td>
<td></td>
</tr>
<tr>
<td>EUR2009.02</td>
<td>HKD2012.02</td>
<td></td>
</tr>
<tr>
<td>GBP2013.12</td>
<td>EUR2009.02</td>
<td></td>
</tr>
<tr>
<td>CHF2011.07</td>
<td>HKD2011.11</td>
<td></td>
</tr>
<tr>
<td>RUB2009.03</td>
<td>JPY2009.01</td>
<td></td>
</tr>
<tr>
<td>RUB2009.02</td>
<td>GBP2013.09</td>
<td></td>
</tr>
<tr>
<td>GBP2009.03</td>
<td>GBP2013.12</td>
<td></td>
</tr>
<tr>
<td>EUR2009.11</td>
<td>JPY2012.03</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.2: Top 10 hybrid outliers detected by the baselines

<table>
<thead>
<tr>
<th></th>
<th>DA Baseline</th>
<th>PA Baseline</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHF2011.08</td>
<td>CHF2011.08</td>
<td></td>
</tr>
<tr>
<td>GBP2009.05</td>
<td>RUB2009.02</td>
<td></td>
</tr>
<tr>
<td>JPY2013.01</td>
<td>CHF2011.09</td>
<td></td>
</tr>
<tr>
<td>EUR2009.02</td>
<td>GBP2013.03</td>
<td></td>
</tr>
<tr>
<td>GBP2013.12</td>
<td>HKD2011.03</td>
<td></td>
</tr>
<tr>
<td>CHF2011.07</td>
<td>HKD2012.02</td>
<td></td>
</tr>
<tr>
<td>RUB2009.03</td>
<td>HKD2010.12</td>
<td></td>
</tr>
<tr>
<td>RUB2009.02</td>
<td>HKD2011.10</td>
<td></td>
</tr>
<tr>
<td>GBP2009.03</td>
<td>EUR2010.10</td>
<td></td>
</tr>
<tr>
<td>RUB2012.06</td>
<td>GBP2011.08</td>
<td></td>
</tr>
</tbody>
</table>

for PA. The top 10 outliers in the 7 time series detected by our two algorithms are shown in Table 5.1. By observation those time points are indeed not only different from their neighbors, but also different from the peers. CHF2011.08 has a very high hybrid outlier score for both algorithms. It is a locally highest point of CHF time series, while other series have downward trends at the same time. Interestingly, we found a news article\(^2\) about the strength of CHF during summer 2011. According to another article\(^3\), the rise of CHF was caused by the European debt crisis, and investors increasingly sought secure investments in

\(^3\)http://www.thelocal.ch/20110923/1277
Table 5.3: Top 10 self-trend and peer-wise outliers

<table>
<thead>
<tr>
<th>Self-trend</th>
<th>Peer-wise</th>
</tr>
</thead>
<tbody>
<tr>
<td>GBP2009.05</td>
<td>HKD2011.08</td>
</tr>
<tr>
<td>RUB2009.02</td>
<td>HKD2011.07</td>
</tr>
<tr>
<td>CHF2011.08</td>
<td>HKD2011.03</td>
</tr>
<tr>
<td>HKD2010.06</td>
<td>HKD2011.06</td>
</tr>
<tr>
<td>EUR2010.05</td>
<td>HKD2011.09</td>
</tr>
<tr>
<td>RUB2009.04</td>
<td>JPY2013.12</td>
</tr>
<tr>
<td>CHF2011.07</td>
<td>HKD2011.02</td>
</tr>
<tr>
<td>HKD2010.11</td>
<td>JPY2013.09</td>
</tr>
<tr>
<td>HKD2010.05</td>
<td>JPY2013.11</td>
</tr>
<tr>
<td>GBP2010.05</td>
<td>JPY2013.10</td>
</tr>
</tbody>
</table>

Switzerland that is considered a safe haven, during times of crisis.

As the baseline methods, we compute k-NN segments without distance weight assignment for each time point, and use the outlier scores in the first iteration as the final outlier scores. The top 10 hybrid outliers found by the baseline methods are shown in Table 5.2. The results of the PA baseline contain four top 10 hybrid outliers identified by PA. For the results of the DA baseline, RUB2012.06 is the only time point which is not in the top 10 hybrid outliers detected by DA. This is because RUB2012.06 is compared with its peers in different k-NN segments during the iterations in our DA algorithm. The 2-NN segments for RUB2012.06 are the segments of time series GBP and CHF in the first iteration of DA. However, since time series RUB is also similar to time series EUR during 2012, the 2-NN segments change to be the segments of time series EUR and GBP, and the outlier score of RUB2012.06 drops in the second iteration. Therefore, our algorithms can get more accurate outlier scores comparing to the baselines.

Table 5.3 shows the top 10 self-trend outliers and peer-wise outliers in the 7 currency time series. The $r$-th smallest distance from each time point to its neighbors is computed as the self-trend outlier degree, and we compute the $r$-th smallest distance from each time point to the peers as the peer-wise outlier degree, where $w = 6$, $k = 2$ and $r = 2$. Interestingly, we observe that time series EUR, GBP, and HKD are similar to each other from January 2010 to December 2010. Because of that, even though GBP2010.05, EUR2010.05, and HKD2010.05 are top 10 self-trend outliers, their hybrid outlier scores are not high. On the other hand, JPY2013.09, JPY2013.10, JPY2013.11 and JPY2013.12 are peer-wise outliers,
however, they are neighbors and similar to each other, so they are also not in the final top 10 hybrid outliers.

We also test the scalability of our two algorithms by using random samples of various sizes of the exchange rate data set. Figures 5.1(a) and 5.1(b) show the scalability of the two algorithms with respect to 2 different factors, the number $n$ of time series, and the number $m$ of time points in each time series. The two algorithms are scalable as the size of data set grows. Figures 5.1(c) and 5.1(b) investigate the number of iterations of the algorithms when $n$ and $m$ vary, respectively, on the real data set. The number of iterations increases slowly with $n$ and $m$ increasing. DA converges faster than PA, which explains the reason why DA runs faster than PA to some extent.

Figure 5.1: Results on the real data set ($w = 5, k = 4$)
Table 5.4: The description of 4 real data sets

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>n</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature</td>
<td>the monthly temperature of 48 states of the USA in 1994-2013</td>
<td>48</td>
<td>240</td>
</tr>
<tr>
<td>Exchange Rate</td>
<td>the monthly exchange rate of 20 currencies w.r.t. USD in 1999-2013</td>
<td>20</td>
<td>180</td>
</tr>
<tr>
<td>Interest Rate</td>
<td>the monthly interest rate of 26 countries in 2001-2011</td>
<td>26</td>
<td>130</td>
</tr>
<tr>
<td>Government GDP</td>
<td>the yearly real government net capital stock as a percentage of real GDP of 22 countries in 1960-2001</td>
<td>22</td>
<td>42</td>
</tr>
</tbody>
</table>

5.2 Results on Synthetic Data Sets

We insert outliers into 4 real data sets to generate synthetic data sets. The real data sets are temperature data set\(^4\), exchange rate data set\(^5\), interest rate data set\(^6\), and government GDP data set\(^7\). The description of the 4 real data sets is in Table 5.4, where \(n\) is the number of time series and \(m\) is the number of time points in each time series. The outlier injection process contains following two steps.

- Step 1: Given a time series database \(S\), randomly select \(x\) time series from \(S\).
- Step 2: For each selected time series \(s_i\) in step 1, replace the value of time points \(s_i[j]\) with an extreme value for \(j = f_i, f_i + y, f_i + 2y, f_i + 3y, \ldots, f_i + \left\lfloor \frac{m-f_i}{y} \right\rfloor y\), where \(0 < y < m\) is a parameter and \(f_i\) is a random value between 0 to \(y\). The extreme value in our context is defined in Definition 5.1.

**Definition 5.1 (Extreme Value).** Given a time series \(s_i\), an interquartile range [3] of time point samples in \(s_i\) is

\[IQR = Q_3 - Q_1,\]

where \(Q_3\) is the third quartile [3] and \(Q_1\) is the first quartile [3] of \(s_i\). An extreme value \(V\)

\(^5\)http://fx.sauder.ubc.ca/data.html
\(^6\)http://www.economicswebinstitute.org/data/eurointerest-longterm.xls
\(^7\)http://www.economicswebinstitute.org/data/EU_all%20data%20for%20is-lm.zip
is a random value in the range of $[V_{\min 1}, V_{\min 2}]$ or $[V_{\max 1}, V_{\max 2}]$, where

\[
V_{\min 1} = Q_1 - 2C \cdot IQR,
\]
\[
V_{\min 2} = Q_1 - C \cdot IQR,
\]
\[
V_{\max 1} = Q_3 + C \cdot IQR,
\]
\[
V_{\max 2} = Q_3 + 2C \cdot IQR,
\]

and $C > 1$ is a parameter. $C$ is 1.5 by default.

**Example 5.1** (Outlier Injection). In Table 5.5, there is a time series database $S$ which consists of 3 time series. Suppose $x = 1$ and $y = 6$, we inject outliers into $S$. We first randomly select $x = 1$ time series to be injected outliers from $S$, and $s_1$ is selected. We then replace the values of time points $s_1[3]$ and $s_1[3+6]$ with extreme values $V$ for $f_1 = 3$. Since $Q_1 = 10$, $Q_3 = 13$ and $IQR = 3$ for $s_1$, $V$ is a random value in the range of $[1, 5.5]$ or $[17.5, 22]$ by Definition 5.1. The time series data injected outliers are shown in Table 5.5, and the values of $s_1[3]$ and $s_1[9]$ are 19 and 2, respectively.

We conduct experiments to test the effectiveness and efficiency of our two algorithms regarding the following parameters.

- $x$ - the number of time series selected to be injected outliers;
- $y$ - the interval span of time points which will be replaced by outliers;
• $k$ - the number of nearest neighbor segments of each $s_i[j - w : j + w]$;

• $w$ - the span of tracking window.

$x$ and $y$ are parameters for generating synthetic data sets, and $k$ and $w$ are parameters of our hybrid outlier detection algorithms.

Our two algorithms named DA and PA are applied to the synthetic data sets to detect top $N$ hybrid outliers, and $N$ is set to be the number of injected outliers for each data set. We use the variable controlling method to conduct our experiments. In each variable controlled test, we compare the performance of DA with that of PA.

### 5.2.1 Effectiveness

The recall in our context is defined in Definition 5.2. Figures 5.3(a), 5.4(a), 5.5(a), and 5.6(a) show the recall of the two algorithms when $x$ varies for the 4 synthetic data sets. In general, we can achieve a recall ranging from 0.6 to 0.99, and the recall is over 0.8 on the synthetic temperature data set when $y = 40$, $k = 4$, and $w = 5$. The recall first increases and then decreases with $x$ increasing for 4 synthetic data sets. Figures 5.3(b), 5.4(b), 5.5(b), and 5.6(b) provide an illustration of the trend of recall when $y$ changes. The recall first increases and then decreases, and our two algorithms have very similar trends.

**Definition 5.2 (Recall of a Method).** Suppose there are $d$ outliers injected in the original data set, and the injected outlier set is denoted as $O_d$. Given a method $A$ that returns top $N$ ($N = d$) hybrid outlier set, denoted by $\text{topNO}$, the recall of method $A$, $\text{recall}_A$, is defined as

$$\text{recall}_A = \frac{|O_d \cap \text{topNO}|}{d},$$

where $|O_d \cap \text{topNO}|$ is the number of elements in intersection of $O_d$ and $\text{topNO}$.

Besides the parameters for generating synthetic data sets, we also examine how the recall changes with respect to the parameters of our two algorithms. By default, $r = 5$ for DA. The results on the change of $k$ are shown in Figures 5.3(c), 5.4(c), 5.5(c), and 5.6(c). In general, our two algorithms can achieve a recall ranging from 0.7 to 0.99. In most cases, the recall drops with $k$ increasing for the two algorithms. The results with respect to the span of tracking window $w$ are shown in Figures 5.3(d), 5.4(d), 5.5(d), and 5.6(d). Similar to $k$, when $w$ increases, the recall decreases gradually. We also test the recall of the DA
algorithm with respect to $r$ on the 4 synthetic data sets. The recall first increases and then decreases when $r$ increases, which is shown in Figure 5.7.

In summary, DA outperforms PA in terms of recall in most cases, and PA is more sensitive to parameter selection compared to DA.

5.2.2 Efficiency

We also test the efficiency of our two algorithms. Figures 5.3(e), 5.4(e), 5.5(e), 5.6(e) and Figures 5.3(f), 5.4(f), 5.5(f), 5.6(f) show the running time in logarithmic scale of our two algorithms with $k$ and $w$ varying, respectively, on the 4 synthetic data sets. The running time of DA and PA both increases when $k$ and $w$ increase. DA runs faster than PA in all cases.

5.3 Summary of Results

We conduct experiments to test the effectiveness and efficiency of our two algorithms. There are 3 parameters that can affect the performance of DA, and the performance of PA can be affected by 2 parameters. We test our algorithms by varying one parameter in each test and compare their performance.

Generally speaking, in terms of recall, DA performs better than PA. Our two algorithms can achieve a recall ranging from 0.7 to 0.99 in most cases. For the efficiency, DA runs faster than PA, and PA converges slower than DA. DA and PA are both scalable as the size of data set grows.
Figure 5.2: Monthly exchange rates of 7 currencies w.r.t. USD
Figure 5.3: Results on synthetic temperature data set
(a) $y = 20$, $k = 4$, $w = 5$

(b) $x = 4$, $k = 4$, $w = 5$

(c) $x = 5$, $y = 80$, $w = 5$

(d) $x = 5$, $y = 80$, $k = 6$

(e) $x = 5$, $y = 80$, $w = 5$

(f) $x = 5$, $y = 80$, $k = 6$

Figure 5.4: Results on synthetic exchange rate data set
Figure 5.5: Results on synthetic interest rate data set
CHAPTER 5. EXPERIMENTS

43

(a) $y = 40$, $k = 4$, $w = 5$

(b) $x = 2$, $k = 4$, $w = 5$

(c) $x = 3$, $y = 25$, $w = 4$

(d) $x = 3$, $y = 25$, $k = 4$

(e) $x = 3$, $y = 25$, $w = 4$

(f) $x = 3$, $y = 25$, $k = 4$

Figure 5.6: Results on synthetic government GDP data set
CHAPTER 5. EXPERIMENTS

(a) Temperature data set  
(b) Exchange rate data set  
(c) Interest rate data set  
(d) Government GDP data set

Figure 5.7: Results of DA on 4 synthetic data sets with respect to $r$ (x=4, y=40, k=5, w=6)
Chapter 6

Conclusions

In this thesis, we tackle the problem of detecting hybrid outliers on time series data. Hybrid outliers are valuable in many applications, like medical diagnosis, network intrusion detection systems, stock market analysis, financial posts, interesting sensor events, earth science, etc. We proposed a heuristic framework to find hybrid outliers. To verify the feasibility of the framework, two algorithms based on the framework were developed, namely a distance-based algorithm and a prediction-based algorithm. The distance-based algorithm computes outlier scores according to distances between time points. The prediction-based algorithm first predicts the value of a time point by applying the least squares technique for multi-variate linear regression, and then computes the deviation from the predicted value to its real value as the outlier score of the time point.

We evaluated our algorithms empirically using synthetic and real data sets. The experimental results verify the effectiveness and efficiency of our algorithms.

Even though the experimental results show our algorithms are effective and efficient, there are some limitations of the algorithms. The algorithms have some parameters and the results highly depend on the parameter selection. The users who use our algorithms need to tune the parameters to get good results. In addition, when the data set is large, the data can not fit into memory. In the future, we will try to overcome these limitations to improve the algorithms.

As for future work, we can also consider the following interesting directions.

• *We can extend our techniques to multi-dimensional time series data.* In this thesis, we only tackle univariate time series data. In the future we can test our detection
methods on time series data with multiple dimensions. The problem will be more complicated and challenging.

- **We can extend our methods to discrete sequences.** The data processed in this thesis are numeric time series. In the future we can extend our framework to handle discrete sequences by utilizing discrete subsequence similarity measures, like edit distance [32], LCSS [2], etc.

- **We can speed up grouping phase.** Our framework re-calculates the k-NN segments for every time point in each iteration. Actually, the k-NN distances between some segments may not change in the next iteration. To speed up this grouping phase, we can use some extra space to store some intermediate results in the previous iteration and avoid some unnecessary k-NN distance recomputation.
Bibliography


[36] P. Sun, S. Chawla, and B. Arunasalam. Mining for outliers in sequential databases. SIAM. 7


