K-MEDIAN CLUSTERING ALGORITHMS FOR TIME SERIES DATA

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

GÖKÇEM YİĞİT

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
INDUSTRIAL ENGINEERING

FEBRUARY 2021

Approval of the thesis:

K-MEDIAN CLUSTERING ALGORITHMS FOR TIME SERIES DATA

submitted by **GÖKÇEM YİĞİT** in partial fulfillment of the requirements for the degree of **Master of Science in Industrial Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Esra Karasakal
Head of Department, **Industrial Engineering** _____

Prof. Dr. Cem İyigün
Supervisor, **Industrial Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Özlem İlk Dağ
Statistics, METU _____

Prof. Dr. Cem İyigün
Industrial Engineering, METU _____

Prof. Dr. Serhan Duran
Industrial Engineering, METU _____

Assoc. Prof. Dr. Ceylan Yozgatlıgil
Statistics, METU _____

Assist. Prof. Dr. Fatma Yerlikaya Özkurt
Industrial Engineering, Atılım University _____

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Gökçem Yiğit

Signature :

ABSTRACT

K-MEDIAN CLUSTERING ALGORITHMS FOR TIME SERIES DATA

Yiğit, Gökçem
M.S., Department of Industrial Engineering
Supervisor: Prof. Dr. Cem İyigün

February 2021, 103 pages

Clustering is an unsupervised learning method, that groups the unlabeled data for gathering valuable information. Clustering can be applied on various types of data. In this study, we have focused on time series clustering. When the studies about time series clustering are reviewed in the literature, for the time series data, the centers of the formed clusters are selected from the existing time series samples in the clusters.

In this study, we have changed that view and have proposed clustering algorithms based on the idea of selecting the cluster centers for each timestamp. With this view, we aim to improve the clustering performance. Based on this idea four different algorithms are suggested that are called as Center Based K-Median Algorithm (CKM), CKM with Haar Wavelet decomposition, CKM with Haar Wavelet Decomposition Without Projection and Search Based CKM with Haar Wavelet Decomposition.

In the first algorithm, the raw data is used and the clustering problem is solved by the proposed optimization model. The other three algorithms are also solved by using the proposed optimization model and instead of using raw data, transformed data, which the Haar wavelet decomposition is applied to, is used. The proposed algorithms have been experimented on different data sets and evaluated by using different internal and

external indices. Due to the evaluations, successful results are obtained regarding clustering performances of the CKM based algorithms.

Keywords: Clustering, Time Series Data, Haar Wavelet Decomposition, Optimization

ÖZ

ZAMAN SERİSİ VERİLERİ İÇİN K-MEDYAN KÜMELEME ALGORİTMALARI

Yiğit, Gökçem
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Cem İyigün

Şubat 2021 , 103 sayfa

Kümeleme bir denetimsiz öğrenme metodudur ve etiketlenmemiş veriyi, bilgi elde etmek amacıyla gruplandırmayı amaçlar. Kümeleme pek çok veri çeşidine uygulanabilir. Bu çalışmada, zaman serisi kümelemesi üzerinde durulmuştur. Literatürde, zaman serisi verileri için, küme merkezleri, kümede var olan zaman serilerinden seçilmiştir.

Bu çalışmada, var olan bakış açısı değiştirilmiş ve küme merkezlerinin her bir zaman noktası için seçilmesi fikrinden yola çıkarak algoritmalar oluşturulmuştur. Bu bakış açısıyla, kümeleme performansını iyileştirmek amaçlanmıştır. Çalışmamızda bu fikir baz alınarak dört farklı algoritma önerilmiştir. Bu algoritmalar şu şekilde isimlendirilmiştir: Merkez Bazlı K-Medyan Algoritması (CKM), Haar Dalgacık Dönüşümü ile CKM, Yansıtmasız Haar Dalgacık Dönüşümü ile CKM ve Arama Bazlı Haar Dalgacık Dönüşümü ile CKM.

İlk algoritmada ham veri kullanılmış ve kümeleme problemi önerilen optimizasyon modeli ile çözülmüştür. Diğer üç algoritmada da önerilen optimizasyon modeli kullanılmıştır ve ham veri kullanmak yerine, Haar dalgacık dönüşümü uygulanmış veriler

kullanılmıştır. Önerilen algoritmalar farklı indisler kullanılarak farklı veri setlerinde denenmiş ve içsel ve dışsal indisler kullanılarak değerlendirilmiştir. Değerlendirmelere göre, CKM bazlı algoritmaların kümeleme performansları ile ilgili başarılı sonuçlar elde edilmiştir.

Anahtar Kelimeler: Kümeleme, Zaman Serisi Verileri, Haar Dalgacık Dönüşümü, Optimizasyon

To my mother and to the memory of my father...

ACKNOWLEDGMENTS

First of all, I would like to express my gratitude to my advisor Prof. Dr. Cem İyigün for all of his enlightening comments, contributions and support through this study. Both in my undergraduate and graduate studies, he guided me to explore new research areas that I enjoyed to work.

Besides, I would like to thank the members of the examining commitee Prof.Dr. Özlem İlk Dağ, Prof. Dr. Serhan Duran, Assoc. Prof. Dr. Ceylan Yozgatlıgil and Assist. Prof. Dr. Fatma Yerlikaya Özkurt for their valuable feedback and time to review this study.

I am thankful to my second family, Sezin Sarıca, Zeynep Şengil, Asu Erkoç, Ali Bakan and Deniz Aslan for being always there for me. I would like to express my gratitude to my colleague Eda Yıldız for her support throughout the undergraduate years and to my coworkers Mert Ünsal, Gizem Çakıroğlu and Emirhan Ergün for their support throughout my graduate years. I would also like to thank Nihat Atay for all of his support through the process of this thesis and for encouraging me whenever I need.

Last but not least, this thesis and everything that I accomplished is completed with the support of my family. Specially, thanks to my mother Ayşe Arzu Yiğit to all of her love and her support for every decision that I make. I am also grateful to my father Mehmet Yiğit teaching me how to chase my dreams. Lastly, I would like to thank my brother Cem Yiğit for all of his jokes, love and support.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ALGORITHMS

ALGORITHMS

LIST OF ABBREVIATIONS

CI	C Index
CKM	Center Based k-Median Algorithm
CKM-H1	CKM with Haar Wavelet Decomposition
CKM-H2	CKM with Haar Wavelet Decomposition without Projection
CKM-SH	Search Based CKM with Haar Wavelet Decomposition
DKM	Discrete k-Median Algorithm
DTW	Dynamic Time Warping
PD	Data Pool Including Periodicity
PSA	Prostate Specific Antigen
RI	Rand Index
SCD	Synthetic Control Chart Time Series Data Pool
SI	Silhouette Index
TOPSIS	Technique of Order Preference Similarity to the Ideal Solution

CHAPTER 1

INTRODUCTION

Huge amount of data is gathered everyday and the analysis of data has a vital importance on human activities. From that much data, in order to obtain valuable information and in order to transform this valuable information to organized knowledge, powerful tools are needed. Data mining has a crucial importance at that point, since it is meeting these mentioned needs.

Machine learning is a technology used in data mining and focuses on the learning methods of computers from data. The main focus of machine learning is the recognition of patterns in data and making precise decisions (Han et al., 2011). Two of the main methods used in machine learning are *supervised learning* and *unsupervised learning*.

Supervised learning methods are working on the labeled data in order to train the algorithm. Following this training process, the labels of unlabeled data are predicted. Unsupervised learning is used for obtaining information from unlabeled data.

Clustering is an unsupervised method, grouping unlabeled objects, that is used for dividing data points to similar groups. Each group is called a cluster, and each sample in a cluster is similar to each other whereas dissimilar to the samples in other clusters (Rai and Singh, 2010). Clustering is used for dividing unlabeled objects to similar groups as mentioned earlier. This process can have lots of purposes such as image segmentation, character and object recognition, information retrieval and data mining (Jain et al., 1999). Clustering can be applied to various types of data. In this study, we will be focusing on *time series clustering*.

The databases that include timestamp information are called *temporal databases* and data classified in temporal databases called *temporal data* (Mitsa, 2010). *Time series*

data is a type of temporal data. In addition to this, time series data is dynamic data since it is composed of series of data points in same time spaces. In other words, it is chronologic collection of observations. Some important features of time series data are to change continuously and to be large in data size (Fu, 2011).

Since, the time series data can be huge in size, lots of users want to see the structured way of the data in order to discover the patterns. By this way, they can get meaningful information from data and can use this information in their studies. Time series clustering has a big impact on the discovery of patterns and addresses two main things in the given data. One is to discover the frequent patterns, two is to discover the surprising patterns (Aghabozorgi et al., 2015). Discovery of patterns in stock options can be given as an example. Some other use cases are recognition of changes, which might help to find out correlation between time series data such as finding factories that has similar move in productions. In addition to this, by applying a clustering on a time series data, a prediction on future data and recommendations based on the predictions can be made such as looking at the climate data of cities of a specific region and making predictions based on that information.

Time series clustering algorithms are divided into five main groups as *Hierarchical Clustering*, *Partitional Clustering*, *Density Based Clustering*, *Model Based Clustering* and *Grid Based Clustering*. In this thesis, we propose partitional clustering algorithms and we are following two main approach. In the first one, by using the raw time series data, we are considering the clustering problem as an optimization problem and providing an optimization model for the solution of it. In the second approach, we are proposing algorithms that are also following the optimization problem approach but instead of using raw data, data with Haar Wavelet Decomposition, that is a preprocessing method applied to raw data, is used.

In Chapter 2, time series clustering algorithms will be explained, and *Partitional Clustering* will be detailed more since it will be the main focus in this study. Also in Chapter 2, since the distance measures has vital importance in time series clustering, the types of distance measures will be explained. Following that, the literature, will be reviewed since there are various studies about time series clustering.

In the literature, for the time series clustering, the centers of the clusters are selected

from the existing time series in the clusters. The algorithms proposed in this thesis is based on a different idea. In the algorithms that we propose, the cluster centers are selected for each timestamp. In other words, when the center points in timestamps are merged, it is not forming an existing time series in the data set. In Chapter 3, the algorithms developed with this idea will be reviewed and the term *Haar Wavelet Decomposition* will be detailed, since it will be used in some of the proposed algorithms in this thesis.

For the clustering algorithms, among the algorithm development, the performance measures has vital importance. In this study, for measuring the success of the algorithms, *internal* and *external indices* are used. The external indices are *Rand Index*, *Adjusted Rand Index* and *Purity* whereas the internal indices are *Silhouette Index* and *C Index*. In Chapter 4, these indices will be explained. For measuring the algorithm performances by using the internal and external indices, data sets generated from two different data pools are used. One data pool is named as Synthetic Control Chart Time Series Data Pool (SCD) and generated by using the base data taken from UCI repository (Dua and Graff, 2017). The second data pool is named as Data Pool Including Periodicity (PD), which is generated with the idea of having time series data including periodicity. These data pools and the data sets generated from these data pools will be also explained in Chapter 4. In the final part of this thesis, in Chapter 5, the findings of this study, the advantages and the disadvantages of the proposed algorithms and future research directions are detailed.

CHAPTER 2

BACKGROUND ON CLUSTERING, TIME SERIES CLUSTERING AND LITERATURE REVIEW

As it is mentioned in Chapter 1, time series clustering has wide range of uses. In this chapter, more detailed information about clustering will be given and time series clustering and the related terms used in this thesis will be explained in detail. Following that, the literature will be reviewed.

2.1 Classifications of Time Series Clustering

Clustering is an unsupervised method that aims to provide maximum similarity for the data within the cluster and minimum similarity with the data in other clusters. It has a wide ranges of uses since with clustering, information can be identified from unlabeled data set and the data set can be organized by having the similar data in the same groups. By the advancing technology, due to more powered the data storages, applications are able to store the data for a long time. Time series data, such as weather data, finance data and exchange rates are some examples for this. In the literature, time series clustering falls into three categories:

- **Whole Time Series Clustering:** It is defining a single time series as a distinct sample and applying clustering based on the similarity of distinct time series samples (Guijo-Rubio et al., 2020).
- **Subsequence Time Series Clustering:** It is clustering of subsequences of a time series. In other words, it is clustering the sections of a long time series data. Subsequence time series clustering is mostly used in discovery of patterns and

structures in time series data (Zolhavarieh et al., 2014).

- **Time Point Clustering:** It is clustering time points due to their proximity and similarity. Time point clustering is similar to subsequence clustering in terms of clustering a single time series data but there can be some points which are not assigned to any cluster and can be considered as noise (Zolhavarieh et al., 2014).

In this study, we will focus on *Whole Time Series Clustering*. Whole time series clustering can be applied in three ways as shape, feature and model based (Fu, 2011).

- **Shape Based:** It is matching the shapes of data. An appropriate distance measure is used and following this a conventional clustering algorithm is applied (Hautamaki et al., 2008).

- **Feature Based:** Under this method, the original time series is converted into a lower dimension vector (feature vector) and a distance vector is calculated. Following this, a conventional clustering algorithm is applied to these lower dimension vectors (Hautamaki et al., 2008).

- **Model Based:** In this method, the original time series data are converted to the model parameters and following the distance measurement, a classic clustering algorithm can be applied (Liao, 2005b).

2.2 Distance Measures Used in Time Series Clustering

Time series clustering is highly dependent on distance measures. The most common clustering measures that will be mentioned in this thesis are rectilinear, euclidean and dynamic time warping (DTW) distances. Time series distance measures can be classified in four categories that are shape based (including lock-step and elastic measures), feature based, edit based and structure based distances (Esling and Agon, 2012).

The *shape based distances* compare the time series by using their actual values and including to categories as elastic and lock-step. For the lock-step measures (in which

we will be mostly focused on under the study of this thesis), the time series should be in same length whereas the elastic measures do not have such a constraint.

For the *feature based distances*, at first it is decided which features to extract and then the distance of these features are calculated. *Edit based distances* are based on the calculation of dissimilarity between time series. The main idea is to use minimum number of transformations, in order to transform the initial time series data to the other. *Structure based distances* aims to compare time series on more global scale.

Lock-step measures have two categories that are *Minkowski distance* and *Pearson correlation coefficient*. The formula of Minkowski distance is as below:

$$d_{minkowski}(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

If p=2, then the distance is named as *Euclidean distance* and if the p=1, it is named as *Rectilinear distance*. In our study, as the distance measure, rectlinear distance is used.

The elastic measures are including two different measures that are *Dynamic Time Warping* and *Longest Common Subsequence*. Dynamic Time Warping is warping two sequences non linearly with the aim of dealing with deformations in time. It was a distance measure introduced by Berndt and Clifford (1994) with the aim of comparing time series that are different in length.

In terms of similarity measures between time series, the elastic distance measures mostly defeats the lock step distance measures. However, the disadvantageous part of it is to cause increment in the calculation time that makes the lock step measures more suitable due to the length and size of the time series. Also, in their paper Keogh and Lin (2005) showed that the difference between the DTW and Euclidean distance diminish when the classification error rates of the data are compared. Xi et al. (2006) have also indicated that for the small data sets, DTW is beating Euclidean distance, that is due to the increment in the requirement of warping for matching with the closest neighbor but for the larger data sets, warping requirement is decreasing since it is more common to find a closer match. And Keogh and Lin (2005) summarized this while DTW warps, it resembles to simple Euclidean distance.

2.3 Classifications of Time Series Clustering Algorithms

Time series clustering algorithms can be classified on to five main groups.

2.3.1 Hierarchical Clustering

Hierarchical clustering aims to form clusters by formation of hierarchy. For hierarchical clustering, the clusters are organized as a tree (Kaufman and Rousseeuw, 2009). It has two types as agglomerative and divisive. *Agglomerative clustering* is starting from the bottom and goes through to top, that is each cluster is merging to a higher cluster whereas *divisive clustering* is dividing a larger cluster to smaller ones.

For hierarchical clustering, k is not needed to be predefined. For any k, the clusters can be obtained by a cut in the dendrogram. While this is the advantageous part of hierarchical clustering, the disadvantage is to calculate distance between each sample and also having no flexibility of applying an adjustment to the tree after a merge or a split is applied (Wang et al., 2006).

2.3.2 Partitional Clustering

The partitional clustering is based on clustering of n unlabeled samples to k clusters in which each cluster contains at least one object and $k \leq n$. Sardá-Espinosa (2017) stated that the partitional clustering algorithms can be perceived as combinatorial optimization problems since the purpose of a partitional clustering algorithm is to minimize the distances of samples in a cluster whereas to maximize the distances between the formed clusters. The partition is named as *crisp* when each sample is a member of exactly one cluster and named as *fuzzy* as a sample can be in distinct clusters with a specific degree.

In partitional clustering algorithms, after definition of k, the next step is to assign k random initial centers. In this thesis, the *random partitioning* method will be used as the initialization method that is based on making random assignment of samples to k clusters and for each randomly formed cluster finding means or medians based on the

algorithm used to assign them as cluster centers (Hamerly and Elkan, 2002).

2.3.2.1 Crisp Partition

The most well known partitional clustering algorithms under this category are *k-means* (MacQueen, 1967) and *k-medoids* (PAM) (Kaufman and Rousseeuw, 1990) algorithms. For k-means the cluster center is the mean value of the samples in the cluster whereas in k-medoid, the cluster center is represented by the most centrally located sample in the cluster.

In k-means algorithm, after the random initial center selection and assignment of samples to the nearest centers, the cluster centers are updated based on the samples assigned to a given cluster. This procedure repeatedly continues until the membership of a sample can not be updated. The procedure is the same for the PAM, but instead of mean, the cluster centers are taken as the sample that has the minimum distance to all of the samples in the cluster.

In k-means algorithm, the cluster center can be affected from the outliers, whereas under k-medoids the cluster center is less reactive to the outlier data. For k-means algorithms, the distance measures used should be meaningful for mean interpretation whereas for k-medoids the distance measure is more flexible.

2.3.2.2 Fuzzy Partition

While the k-means and k-medoid algorithms are giving exact results about an object belonging to a cluster, Fuzzy c-means algorithm (Bezdek et al., 1984) is giving softer results that means each sample has ratios of belonging to a cluster.

The advantage of partitional clustering to hierarchical clustering is the rapidness and as a result they are more suitable for clustering the time series data.

2.3.3 Density Based Clustering

Density based clustering is applied for identifying the clusters in data according to the idea of a cluster is being a region with high density and distinctive from the other such clusters by regions of low density (Sander, 2010). Since the density based clustering algorithms have high complexity, they are not much used in time series clustering.

2.3.4 Model Based Clustering

Under this approach, for each cluster a model is assumed and the data is clustered according to fitting to the models. Model based clustering algorithms mostly have scalability problems and once the clusters become close to each other, the performance of the model based clustering algorithms reduces (Liao, 2005a).

2.3.5 Grid Based Clustering

The grid based clustering is different from the other clustering algorithms since it is focusing on the value space around the data points rather than focusing on data points. The grid based clustering algorithm is starting with formation of grid structure and calculation of the cell density for each of the cells. Following that, the cells are sorted due to their densities and the cluster centers are found (Schikuta, 1993).

2.4 Literature Review for Clustering and Time Series Clustering

Clustering problems are tried to be solved in various ways in the literature. Bradley et al. (1996) try to solve this problem by concentrating on a concave minimization formulation which, they claimed, is a fast and finite algorithm. They define their problem explicitly as: for m points in n-dimensional space (R^n), given number of clusters as k, specify k centers in R^n such that the sum of distances of points to the closest center is minimized. They depict that the solution of this problem is not easy since a minimum local point might not be a global minimum point and as a result, conversion of this problem to a bilinear program, that is called “a fast successive

linearization k-median Algorithm" can give a more accurate result. Bradley et al. (1996) also indicate that the k-median algorithm is giving better results than k-means algorithm since the outliers affect the k-median algorithm less.

The first algorithm that they have proposed is as below in 2.1. In the below formulation, the matrix $A \in R^{mxn}$ is representing a set A of m point in R^n, k is the number of clusters, C_l is the center of cluster $l \in \{1, 2, 3...k\}$, D_{il} is indicated as a dummy variable ($i \in \{1, 2, 3...m\}$), which is used for bounding the 1-norm distance between A_i and C_l. In this algorithm, it is aimed to find the cluster centers to minimize the sum of the distances between a point and the cluster center that the point belongs to.

$$\text{Minimize} \quad \sum_{i=1}^{m} \min_{l=1,2,...k} \left(e^T D_{il}\right) \tag{2.1}$$

$$\text{subject to:} \quad -D_{il} \leq A_i^T - C_l \leq D_{il}. \quad i \in \{1, 2, 3...m\}, l \in \{1, 2, 3...k\} \tag{2.2}$$

In this algorithm, since the objective is minimization of k linear and concave functions, it is named as a *piecewise linear concave function* (Bradley et al., 1996). For an effective solution of this problem, they propose reformulating the problem as a bilinear program. The bilinearized version of the problem can be seen in below.

$$\text{Minimize} \quad \sum_{i=1}^{m} \sum_{l=1}^{k} e^T D_{il} T_{il} \tag{2.3}$$

$$\text{subject to:} \quad -D_{il} \leq A_i^T - C_l \leq D_{il}, \quad i \in \{1, 2, 3...m\}, l \in \{1, 2, 3...k\} \tag{2.4}$$

$$\sum_{l=1}^{k} T_{il} = 1, \qquad i \in \{1, 2, 3...m\}, l \in \{1, 2, 3...k\} \tag{2.5}$$

$$T_{il} \geq 0. \qquad i \in \{1, 2, 3...m\}, l \in \{1, 2, 3...k\} \tag{2.6}$$

For the solution of this problem, Bradley et al. (1996) mention the term *uncoupled bilinear programming* (UBPA), which is firstly defined by Bennett and Mangasarian (1993) in the literature. Bennett and Mangasarian (1993) define UBPA as an algorithm that decomposes the existing problem into two and solving them by fixing a variable alternatively until an optimal solution is found.

In their paper, Charikar et al. (2002) mention that the k-median algorithm has similar points with the uncapacitated facility location problem. In this problem type, for the opened facilities, there is no limit and the aim is to minimize the cost which is composed of the cost of opening facilities and the cost of assigning a location to the nearest facility.

For p-median facility location problem, the first formulation is done by ReVelle and Swain (1970) in which there the two decision variables are binary: one is the selection variable whereas the other one is the allocation variable.

They have used the following notations: y_i indicates if i is selected as median, z_{ij} indicates whether sample j is assigned to the cluster which has i as median. In the objective function, they have the matrix $D \in R^{nxn}$, where d_{ij} is the distance between sample i and j. ReVelle and Swain (1970) form the following model:

$$\text{Minimize} \quad \sum_{j=1}^{n}\sum_{i=1}^{n} d_{ij} z_{ij} \tag{2.7}$$

$$\text{subject to:} \quad \sum_{i=1}^{n} z_{ij} = 1, \qquad j \in \{1, 2, 3...n\} \tag{2.8}$$

$$\sum_{j=1}^{n} z_{ij} \leq y_i, \qquad i \in \{1, 2, 3...n\} \tag{2.9}$$

$$\sum_{j=1}^{n} y_j = k, \tag{2.10}$$

$$z_{ij} \in \{0, 1\}, \tag{2.11}$$

$$y_j \in \{0, 1\}. \tag{2.12}$$

The objective of this model is to minimize the sum of distances from samples to cluster centers. The first, second and third constraints are respectively indicating each sample should be assigned to a cluster, samples should be assigned to selected centers, there are k clusters.

In this study, the proposed algorithms, that will be detailed in Chapter 3, are under the category of Whole Time Series Clustering. In the algorithms that we proposed, both raw and transformed data are used and rectilinear distance is selected as the distance

measure. In the literature, there are various studies that are focusing on Whole Time Series Clustering which differ due to the representation of the data used, distance measures used for measuring the distance between time series and proposed clustering algorithms.

Košmelj and Batagelj (1990) propose a clustering method in which time series data are used as raw time series and the distance measure is euclidean. In the proposed method, a procedure that includes iterative relocation clustering is used. Their method starts with initial partition. Following the distance calculation between time series, the clusters are updated by swapping units between clusters until the best solution can be found. Another study, in which the raw time series data and the euclidean distance is used, is proposed by Golay et al. (1998). In their study, fuzzy c-means algorithm is used in order to cluster brain signals due to applications of different stimuli. Policker and Geva (2000) also propose a study that is using raw time series data and fuzzy partition. In their study, as the distance measure, Dynamic Time Warping (DTW) is used and their aim is to cluster time series with similar patterns. Ratanamahatana and Keogh (2004) is also proposed a clustering method that uses raw time series data. In their study, multimedia data are represented as time series data and the k-medoids algorithm is applied to this data by using DTW as the distance measure.

Hautamaki et al. (2008) emphasize the importance of cluster prototype calculation in shape based time series clustering by using DTW as a distance measure. In their paper, they mention that, in the k-medoids problem, the medoid is selected as a prototype which is defined as the time series in the cluster in which the distance of other time series in the same cluster is minimized. In their paper, they suggest alternative prototype calculations such as averaging and prototype by local search. Averaging is simply taking the mean of the time series in each time point. In prototype by local search, the method includes combining the time series in sequential or hierarchical order. One negative side is summarized as the final prototype is dependent on the order of pairing.

Besides the raw time series, there are literature studies that are using transformed time series data for the application of clustering algorithms. Lin et al. (2004) suggest a new method by applying k-means algorithm to a transformed time series data. For k-means

algorithm, the initial center selection has an effect on the quality of the resultant clusters. Method suggested by Lin et al. (2004) clarifies this effect. Under this method for each time series, the *Haar wavelet decomposition* is calculated. This calculation is a one time process. The original data is in *level 0* of the Haar Wavelet decomposition and the levels are getting higher in number as the data gets simpler with decomposition. After the Haar Wavelet decomposition, starting with the highest level of transformation (lower resolution data), the k-means algorithm is applied gradually to the levels. The *I-kmeans algorithm*, that they suggest, is standing for interactive k-means. Due to Lin et al. (2004), time series' shape are preserved well in very beginning levels of the wavelet decomposition so that the results of the clusters can be obtained in the low levels which don't need the full decomposition of the data. Ratanamahatana et al. (2005) propose applying k-means algorithm with the clipped time series representation of the initial data. In their study, they prove that this method is giving faster results than using DTW as a distance measure for time series data. Bagnall and Janacek (2005) also defend the advantages of using clipping time series data in time series clustering by using euclidean distance as the distance measure and applying k-means and k-medoids clustering algorithms. Aghabozorgi et al. (2012) focus on studying with transformed time series by using Discrete Wavelet Transform. In their study, an iterative clustering algorithm, that is called Incremental Fuzzy C-Mean Clustering for time series, is proposed by aiming the algorithm to have the ability of accepting new time series and updating clusters. As the distance measure Longest Common Subsequence is used.

Seref et al. (2014) define the discrete k-median clustering (DKM), in which the cluster centers are selected among an existing sample in the clusters. The algorithm that they suggest in their paper is based upon the formulation defined by Bradley et al. (1996). In the algorithm of Bradley et al. (1996) only 1-norm distance can be used as a distance measure but Seref et al. (2014) expand this work in their algorithm in which any distance measure can be used. Since the model of Bradley et al. (1996) is a bilinear model, Seref et al. (2014) propose to solve it with uncoupled bilinear program that is proposed in the paper of Bennett and Mangasarian (1993) that solves the bilinear problem by dividing it into two alternating linear programs that is solved in assignment of samples and update of medians steps. In this thesis, the proposed

algorithms will be compared with DKM algorithm. Since as the distance measure, the rectilinear distance is used in our algorithms, it is also used in DKM algorithm.

The mentioned literature studies, that are focusing on *Whole Time Series Clustering*, are also summarized in Table 2.1.

Table 2.1: Literature studies about whole time series clustering

Article	**Representation Methods**	**Distance Measures**	**Clustering Algorithms**
Košmelj and Batagelj (1990)	Raw Time Series	Euclidean	Modified Relocation Clustering
Golay et al. (1998)	Raw Time Series	Euclidean	Fuzzy c-Means
Policker and Geva (2000)	Raw Time Series	DTW	Fuzzy Clustering
Ratanamahatana and Keogh (2004)	Raw Time Series	DTW	k-Medoids
Lin et al. (2004)	Wavelets	Euclidean	k-Means
Ratanamahatana et al. (2005)	Clipped Time Series Representation	Lower Bound Clipped	k-Means
Bagnall and Janacek (2005)	Clipped Time Series Representation	Euclidean	k-Means, k-Medoids
Hautamaki et al. (2008)	Raw Time Series	DTW	k-Medoids
Aghabozorgi et al. (2012)	Discrete Wavelet Transform	Longest Common SubSequence	Fuzzy c-Means Clustering
Seref et al. (2014)	Raw Time Series	Arbitrary Pairwise Distance Matrices	DKM

CHAPTER 3

PROBLEM DEFINITION

In the previous chapters, it is mentioned about the background of clustering, the terms used for clustering are introduced and the taxonomy of clustering is detailed. Following these, the time series algorithm types are briefly summarized and the literature review that is focusing on partitional clustering, which will be our main focus in this thesis, is given. In this chapter, the features of the problem defined in this study will be detailed. Following that, proposed algorithms will be discussed.

3.1 Problem Statement

As it was mentioned in the previous studies in the literature review, for the k-median clustering algorithm applied to time series data, the centers of the clusters are selected among the existing time series in each cluster. This can cause the selected cluster center to miss the local changes of the time series data in the cluster.

In our solution approach, this is not followed and the cluster centers are selected for each timestamp. In other words, when the resultant cluster centers in each timestamp is combined, the resultant vector is not an available time series in the data set, so it is allowed the cluster centers to be selected from different time series data for each timestamp. Due to this timestamp based center selection approach, the local changes in the formed clusters can be identified. In Figure 3.1, it can be seen that when a cluster center is selected from an existing time series, the center might miss the local changes in the data. With the timestamp based center selection approach, the center can represent the local behaviors of data.

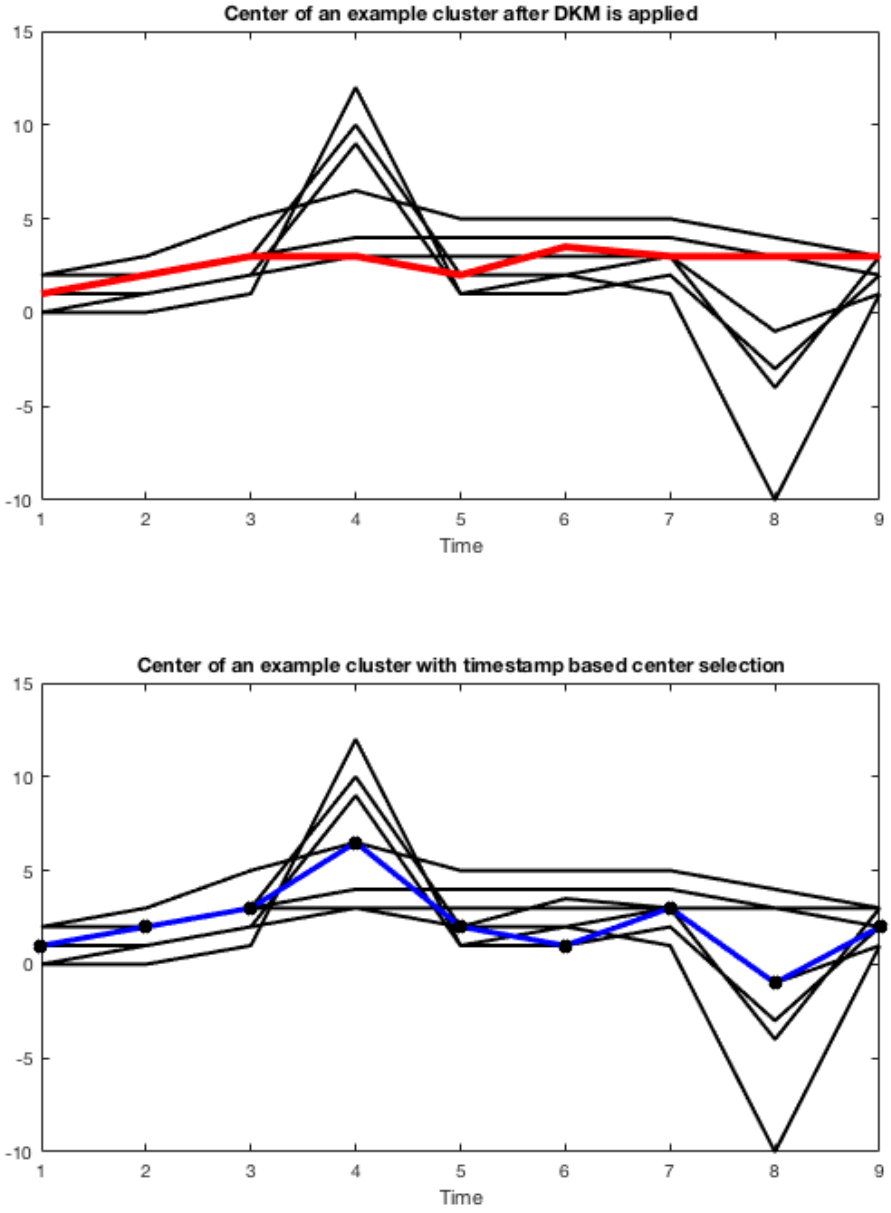

Figure 3.1: Example cluster with different center selection approaches

For the solution of the clustering problem, by following the defined center selection approach, the steps of the problem formulated and solved as an optimization problem.

Since in this study each timestamp has different centers, the distance of a time series to a cluster center is computed by the summation of distances between the value that time series is taking in timestamp q and the cluster center in timestamp q in which $q \in t$. As the similarity measure, rectilinear distance is used and a distance matrix D_{jiq} is formed, that includes rectilinear distance between time series j and time series i in timestamp q.

The calculation can be formulated as follows:

Let S_q^i be the value that time series i is taken in timestamp q.

$$D_{jiq} = \left| S_q^j - S_q^i \right|$$

The main objective in this study is to assign n samples, that is the number of time series data, to the k clusters by minimizing the summation of the distances between n samples to the nearest cluster centers.

3.2 Proposed Algorithms

Under this chapter, four different models, suggested for the solution of time series clustering problem, are defined. In each algorithm, the main purpose is to select cluster centers for each timestamp such that the summation of the distances between n samples and the nearest cluster centers are minimized.

For solving the defined clustering problem, mainly, two approaches are followed. In the first approach, we formulated the model as an optimization model. In the first algorithm, the proposed optimization model is solved by using the raw time series data. In the second approach followed, that is used for the three other algorithms, for the solution of the problem, the proposed optimization model is solved with using the transformed time series data with Haar Wavelet Decomposition, that will be detailed later.

3.2.1 Center Based k-Median Algorithm for Time Series Data

Consider a data set where I represents the set of time series, Q represents the timestamps and P represents the set of clusters. We try to assign time series data to the given number of clusters by selecting the appropriate centers of the clusters for each timestamp so that the summation of the distances between samples and the nearest cluster centers are minimized.

For the solution of this problem, an optimization problem is formulated. The following problem is to solve the k-median problem in which the objective function (3.1) aims to assign n samples to the k clusters by minimizing the summation of the distances between n samples to the nearest cluster centers.

$$(\textbf{CKM})\ \text{Minimize}\ \sum_{i=1}^{n} \min_{p=1,2,\ldots k} \left(\sum_{q=1}^{t} \left(\sum_{j=1}^{n} D_{jiq} X_{jpq} \right) \right) \tag{3.1}$$

$$\text{subject to:}\ \sum_{i=1}^{n} X_{ipq} = 1, \qquad \forall p \in P, \forall q \in Q \tag{3.2}$$

$$\sum_{p=1}^{k} X_{ipq} \leq 1, \qquad \forall i \in I, \forall q \in Q \tag{3.3}$$

$$X_{ipq} \in \{0, 1\}. \qquad \forall i \in I, \forall p \in P, \forall q \in Q \tag{3.4}$$

The binary decision variable in this formulation is X_{ipq} that is equal to 1 if time series i in timestamp q (S_q^i) is selected as median to cluster p in timestamp q.

The inner summation is the sum of the distances of the time series i (S_q^i) to the centers of cluster p in each timestamp q. In the first inner minimization, we are trying to minimize the value gathered in inner summation for different values of p.

The constraint 3.2 indicates, for a given timestamp q and cluster p, there can be at most one center. The constraint 3.3 indicates time series i in timestamp q can be at most center to one cluster.

In Table 3.1, the notation used for mathematical formulations is provided.

Table 3.1: Notation used for mathematical formulations

Sets	
P	Set of clusters ($p = 1, 2, ..., k$)
Q	Set of timestamps ($q = 1, 2, ..., t$)
I	Set of time series ($i = 1, 2, ..., n$)
Parameters	
D_{jiq}	Rectilinear distance between time series j and time series i in timestamp q.
Decision Variables	
X_{ipq}	Binary decision variable equal to 1 if time series i in cluster p is selected as center for timestamp q.
T_{ip}	Binary decision variable equal to 1 if time series i is in cluster p.

The inner minimization in 3.1 is equal to the following problem:

$$\text{Maximize} \quad U \tag{3.5}$$

$$\text{subject to:} \quad U \leq \sum_{q=1}^{t} \left(\sum_{j=1}^{n} (D_{jiq} X_{jpq}) \right). \qquad \forall p \in P, i \tag{3.6}$$

And when the dual of the problem above is taken, the following formulation is found:

$$\text{Minimize} \quad \sum_{p=1}^{k} \left(\sum_{q=1}^{t} \left(\sum_{j=1}^{n} (D_{jiq} X_{jpq}) \right) \right) \tag{3.7}$$

$$\text{subject to:} \quad \sum_{p=1}^{k} T_{ip} = 1. \qquad \forall i \in I \tag{3.8}$$

When 3.7 is replaced with the first inner minimization in 3.1 the following final formulation of CKM is gathered:

$$(\textbf{CKM}) \quad \text{Minimize} \quad \sum_{i=1}^{n}\left[\sum_{p=1}^{k}\left(\sum_{q=1}^{t}\left(\sum_{j=1}^{n} D_{jiq}X_{jpq}\right)\right)T_{ip}\right] \tag{3.9}$$

$$\text{subject to:} \quad \sum_{i=1}^{n} X_{ipq} = 1, \qquad \forall p \in P, \forall q \in Q \tag{3.10}$$

$$\sum_{p=1}^{k} X_{ipq} \leq 1, \qquad \forall i \in I, \forall q \in Q \tag{3.11}$$

$$\sum_{p=1}^{k} T_{ip} = 1, \qquad \forall i \in I \tag{3.12}$$

$$X_{ipq} \in \{0,1\}, \qquad \forall i \in I, \forall p \in P, \forall q \in Q \tag{3.13}$$

$$T_{ip} \in \{0,1\}. \qquad \forall i \in I, \forall p \in P \tag{3.14}$$

As it can be realized from the formulation of the problem, in this problem, there are two unknown variables that are X_{ipq} and T_{ip}. Seeing that there are two unknown variables, in order to solve this bilinear problem, the approach that Bradley et al. (1996) is followed for solving a bilinear problem, that is called Uncoupled Bilinear Program (UBPA) is followed. UBPA can be defined as an algorithm that decomposes the existing problem into two and solving them by fixing a variable alternatively until an optimal solution is found.

For the solution of this problem, two subproblems are solved one after the other, in which one variable is set and the problem is solved to find the other variable alternatively.

Before the first subproblem is solved, at first the initial cluster center assignments X_0 are made for each timestamp q. For this, random clusters are formed for the given number of clusters. And based on the cluster members, the cluster centers for each timestamp are determined based on selecting the sample that has the closest total distance to the other data points.

In the given timestamp q, $X_{0,mpq}$=1 for m is a member of cluster p, and the minimum value of the given calculation is found for $j = m$ for the formulation 3.15 , in which I_p indicates the samples that are members of cluster p.

$$\min_{j \in I_p} \sum_{j \neq i, i \in I_p} D_{jiq} \tag{3.15}$$

After this initialization, the subproblem 1 is formulated as below (**CKM-S1**). In this problem, the cluster centers are assumed to be known (X_{jpq}) and the cluster assignments (T_{ip}) are taken as the decision variable.

$$(\textbf{CKM-S1}) \quad \text{Minimize} \quad \sum_{i=1}^{n} \left[\sum_{p=1}^{k} \left(\sum_{q=1}^{t} \left(\sum_{j=1}^{n} D_{jiq} X_{jpq} \right) \right) T_{ip} \right] \tag{3.16}$$

$$\text{subject to:} \quad \sum_{p=1}^{k} T_{ip} = 1, \qquad \forall i \in I \tag{3.17}$$

$$T_{ip} \in \{0, 1\}. \qquad \forall i \in I, \forall p \in P \tag{3.18}$$

This problem aims to assign the time series i to cluster p in which its sum of distances between the time series i and the cluster centers in each timestamp q is minimized. Based on the solution of this subproblem (**CKM-S1**) the second subproblem (**CKM-S2**) is solved in order to find new medians for the given cluster members.

$$(\textbf{CKM-S2}) \quad \text{Minimize} \quad \sum_{q=1}^{t} \left[\sum_{p=1}^{k} \left(\sum_{i=1}^{n} \left(\sum_{j=1}^{n} T_{jp} D_{jiq} \right) X_{ipq} \right) \right] \tag{3.19}$$

$$\text{subject to:} \quad \sum_{i=1}^{n} X_{ipq} = 1, \qquad \forall p \in P, \forall q \in Q \tag{3.20}$$

$$\sum_{p=1}^{k} X_{ipq} \leq 1, \qquad \forall i \in I, \forall q \in Q \tag{3.21}$$

$$X_{ipq} \in \{0, 1\}. \qquad \forall i \in I, \forall p \in P, \forall q \in Q \tag{3.22}$$

The inner summation indicates that, for given timestamp q and for given cluster p, if time series i is selected as cluster center, the total distances of the other time series data in the same cluster to center i. So, by minimizing the total sum and based on this assignment cost problem, the new centers for each cluster are determined. Following that subproblem 1 (**CKM-S1**) can be solved again.

Under our algorithm, in order to prevent excess calculation, i and p are restricted as the samples in cluster p that is denoted as I_p. In other words, the objective function in 3.19 is reformulated as follows:

$$\sum_{q=1}^{t}\left[\sum_{p=1}^{k}\left(\sum_{i\in I_p}\left(\sum_{j\in I_p}T_{jp}D_{jiq}\right)X_{ipq}\right)\right] \tag{3.23}$$

Since the solution of this sequential algorithm can change for each initial assignment, we have conducted the initialization 500 times and results are investigated for each different initialization. Among 500 initialization, the initialization in which the cluster and center assignments are providing the minimum optimal objective function is selected in order to evaluate the algorithm based on different indices. This approach is followed for all of the algorithms defined in chapters 3.2.2, 3.2.3 and 3.2.4.

Pseudocode of CKM is provided in Algorithm 1.

Algorithm 1: Center Based k-Median Algorithm

Procedure `CKM()`
- **input** : k, I
- **output** : T_{ip}, X_{ipq}

1 Initialize $X_{0,ipq}$
2 $Z_0 \leftarrow \infty$
3 $\delta = 1000000$
4 **repeat**
5 **CKM-S1**
- **input** : $D_{jiq}, X_{0,ipq}$
- **output** : T_{ip}

6 $T_{0,ip} \leftarrow T_{ip}$
7 **CKM-S2**
- **input** : $D_{jiq}, T_{0,ip}$
- **output** : X_{ipq}

8 $X_{0,ipq} \leftarrow X_{ipq}$
9 $Z \leftarrow \sum_{q=1}^{t}\left[\sum_{p=1}^{k}\left(\sum_{i\in I_p}\left(\sum_{j\in I_p}T_{jp}D_{jiq}\right)X_{ipq}\right)\right]$
10 $\delta = Z_0 - Z$
11 $Z_0 \leftarrow Z$
12 **until** $\delta < 0.000001$

end

CKM is visualized in Figure 3.2 with a small data set that has 6 time series data, $t = 10$ and given $k = 2$. In the figure, the initial centers are visualized with points in the **Data Set and Initial Centers (1)** graph. An iteration of CKM is completed by the successive applications of CKM-S1 and CKM-S2. In the first iteration, after the application of CKM-S1, the clusters based on the initial centers are found, that is visualized in the **Iteration 1: Clusters after CKM-S1 Applied (2)** graph. Following the new cluster, the new centers are found by applying CKM-S2. The found centers are merged with a black line and once the graph **Iteration 1: Clusters after CKM-S2 Applied (3)** is checked, it can be realized that the black line do not represents an existing time series. After the new centers are found, the algorithm comes back to the CKM-S1 in iteration 2 and new clusters are found (Graph 4). CKM-S2 followed that process by finding the cluster centers (Graph 5). Seeing both the centers and cluster assignments have not been changed, the algorithm is stopped and the final version of the clusters and cluster centers are visualized in **Final Clusters (6)**.

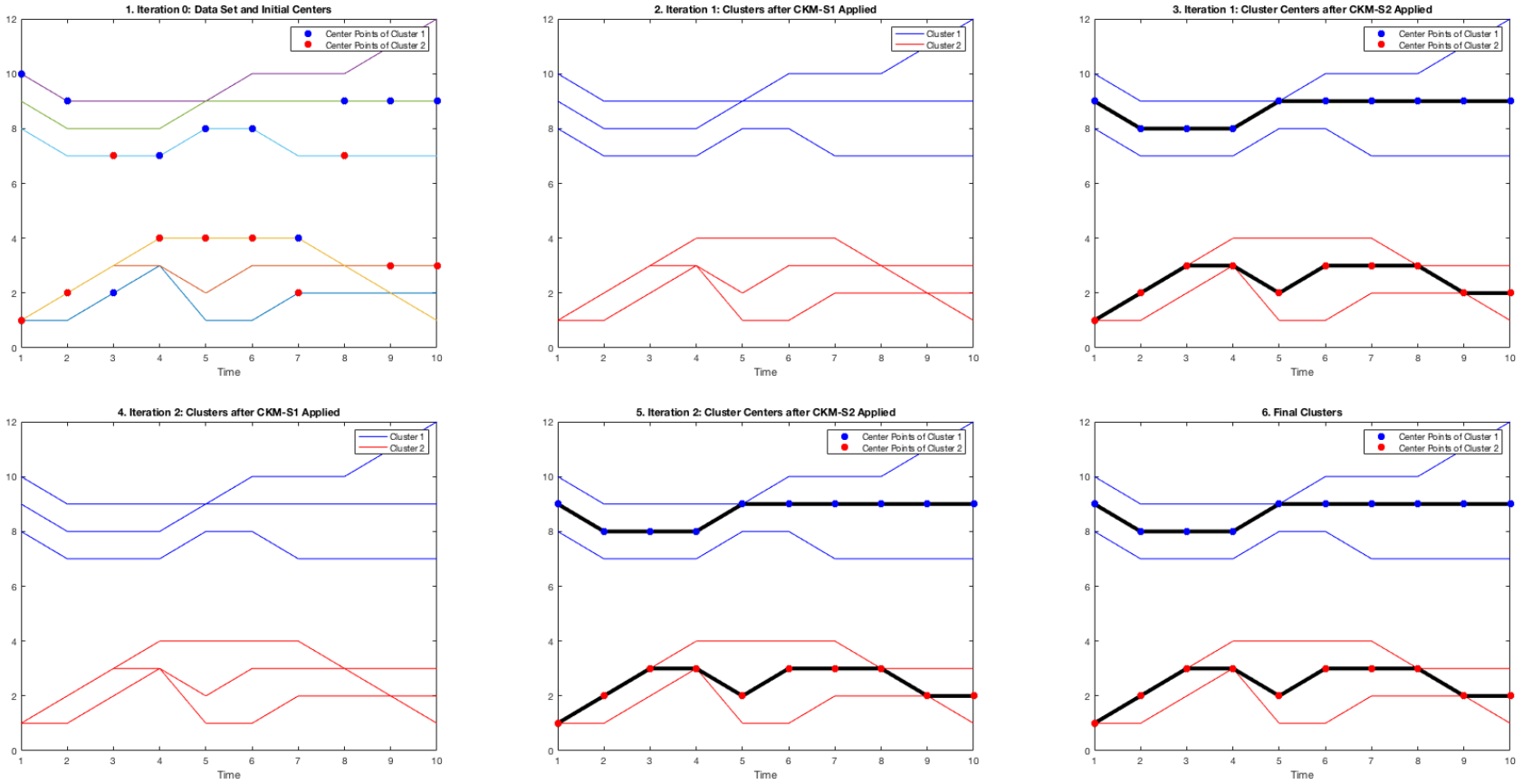

Figure 3.2: CKM applied on an example data set

3.2.2 CKM with Haar Wavelet Decomposition

In the previous chapter, we have defined the formulation of the proposed optimization problem. In the provided algorithm, CKM, the problem is solved by using the raw data. Besides the algorithms using raw data, while applying the clustering algorithms on time series data, the data can be preprocessed or transformed. In the proposed algorithms in this part, the optimization model approach, followed in the solution of the first algorithm, is applied by using transformed data. For this transformation, the method *"Haar wavelet decomposition"* is used, since the transformed data, which the Haar wavelet decomposition is applied to, still preserves the shape of data and do not loose the characteristics of the original data. These are important features of the Haar wavelet decomposition, since we are working on timestamp based center selection approach.

The Haar transform is a term to be mentioned before Haar wavelet decomposition. The Haar transform is decomposing a discrete signal (in our case a time series) to different subsignals that are *trend* (running average) and *fluctuation* (running difference). The fluctuation subsignal is small in magnitude when it is compared with the original subsignal.

The trend subsignal is calculated by taking the running average of two successive values and then multiplying it by $\sqrt{2}$. Let's call the discrete signal (in our case a single time series data) as $f = f_1, f_2, ...f_N$. The trend subsignals will be calculated as below.

$$a_m = \frac{f_{2m-1} + f_{2m}}{2} \cdot \sqrt{2} \quad , \qquad m = 1, 2, 3, ..., \frac{N}{2} \tag{3.24}$$

The fluctuation subsignal is calculated by taking the averages of differences of two successive values and then multiplying it with $\sqrt{2}$.

$$d_m = \frac{f_{2m-1} - f_{2m}}{2} \cdot \sqrt{2} \quad , \qquad m = 1, 2, 3, ..., \frac{N}{2} \tag{3.25}$$

Wavelets are defined as the mathematical functions used in order to make the representation of data in terms of differences (fluctuation) and averages (trend) of a prototype function called the mother wavelet (Daubechies and Bates, 1993). The Haar wavelet representation can be summarized as approximating time series with a linear combination of the basis functions (Lin et al., 2004).

In order to show 1-level fluctuation subsignals in a simpler form, the scalar products of the 1-level Haar wavelets can be used and 1-level Haar wavelets are defined as:

$$W_1^1 = \left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 0, 0, 0, 0..., 0\right)$$
$$W_2^1 = \left(0, 0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 0, 0..., 0\right)$$
$$...$$
$$W_{\frac{N}{2}}^1 = \left(0, 0, 0, 0, .., 0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right)$$

The the 2-level Haar wavelets can be defined as:

$$W_1^2 = \left(\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2}, 0, 0, 0, 0, ., 0\right)$$
$$W_2^2 = \left(0, 0, 0, 0, \frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2}, .., 0\right)$$
$$...$$
$$W_{\frac{N}{2}}^2 = \left(0, 0, 0, 0, ..., \frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2}\right)$$

The fluctuation subsignal in 3.25 can be written in terms of scalar products of the discrete signal and 1-level Haar wavelet.

$$d_m = f.W_m^1, \qquad m = 1, 2, 3, ..., \frac{N}{2} \tag{3.26}$$

In order to express, 1-level trend values by scalar product of certain elementary signals, 1-level Haar scaling signals can be used and they are defined as:

$$V_1^1 = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, 0, ., 0\right)$$
$$V_2^1 = \left(0, 0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, ., 0\right)$$
$$...$$
$$V_{\frac{N}{2}}^1 = \left(0, 0, 0, .., \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$$

2-level Haar scaling signals are defined as:

$$V_1^2 = \left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, 0, 0, 0, ., 0\right)$$
$$V_2^2 = \left(0, 0, 0, 0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, .., 0\right)$$
$$...$$
$$V_{\frac{N}{2}}^2 = \left(0, 0, 0, 0, ..., \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$$

The trend subsignal in 3.24 can be written in terms of scalar products of the discrete signal and 1-level Haar scaling signal.

$$a_m = f.V_m^1, \qquad m = 1, 2, 3, ..., \frac{N}{2} \tag{3.27}$$

A single time series data $f = f_1, f_2, ...f_N$ can be expressed by the summation of two signals that are called the first averaged signal (A^1) and the first detail signal (D^1). By formulation:

$$f = A^1 + D^1 \tag{3.28}$$
$$A^1 = a_1 V_1^1 + a_2 V_2^1 + ... + a_{N/2} V_{N/2}^1 \tag{3.29}$$
$$D^1 = d_1 W_1^1 + d_2 W_2^1 + ... + d_{N/2} W_{N/2}^1 \tag{3.30}$$

As an example, let say the time series data is:
$f = (10, 12, 4, 6, 6, 8)$
The trend subsignal of this time series data is:
$a_1 = (11\sqrt{2}, 5\sqrt{2}, 7\sqrt{2})$

The first averaged signal of this time series data is:

$A^1 = (11, 11, 5, 5, 7, 7)$

The fluctuation subsignal of this time series data is:

$d_1 = (-\sqrt{2}, -\sqrt{2}, -\sqrt{2})$

The first detail signal of this time series data is:

$D^1 = (-1, 1, -1, 1, -1, 1)$

As the Haar wavelet representation of this data in level 1, the first averaged signal will be used. The representation of data in the other levels are calculated with this logic.

To summarize, Haar wavelet decomposition can help in representation of time series data in different resolutions. Additionally, by applying the Haar wavelet decomposition, in a time series, there are coefficients stored, that are the averages and the averages of the differences of two successive values, which are important for storing the information about the transformed times series sample. In Figure 3.3, a time series data with different levels of Haar wavelet decomposition can be seen.

If the time series has the length as N, the decomposition levels can be maximum $\log_2 N$, if N is power of 2 and $floor(\log_2 N/2)$ otherwise.

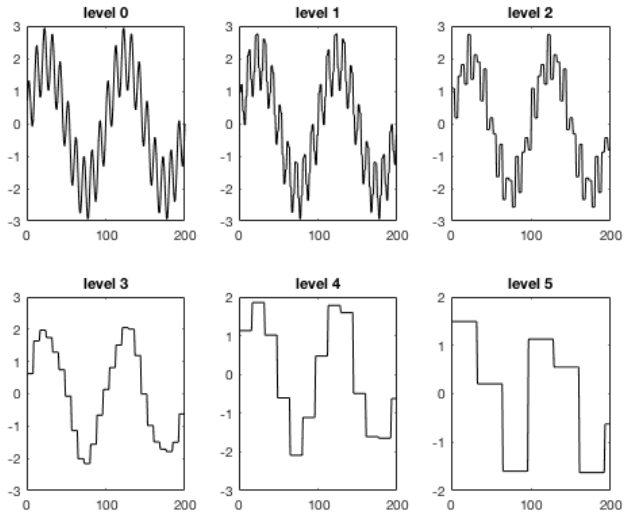

Figure 3.3: The application of Haar wavelet decomposition on an example time series data

For the defined algorithm in this chapter, CKM is combined with the idea proposed in the paper of Lin et al. (2004) and the algorithm is called CKM with Haar wavelet decomposition (CKM-H1). The additional notation used in CKM-H1 formulation can be seen in Table 3.2.

Table 3.2: Notation used for mathematical formulations of CKM-H1

Sets	
H	Set of Haar wavelet decomposition levels ($h = 1, 2, ..., r$) (h=1 indicates level 0, h=r indicates the highest level)
Parameters	
M_{iqh}	Matrix in which the Haar wavelet decomposition of each time series is stored.
D_{jiqh}	Rectilinear distance between time series j and time series i in timestamp q for Haar wavelet decomposition level h. ($\lvert M_{jqh} - M_{iqh} \rvert$)
Decision Variables	
X_{ipqh}	Binary decision variable equal to 1 if time series i is selected as median to cluster p in timestamp q for Haar wavelet decomposition level h.
T_{iph}	Binary decision variable equal to 1 if time series i is in cluster p for Haar wavelet decomposition level h.

Under the method proposed by Lin et al. (2004), for each time series, the *Haar wavelet decomposition* is calculated before applying the algorithm. The data in *level 0* of the Haar wavelet decomposition is the original data and the levels are getting higher in number as the data gets coarser. After the application of the Haar wavelet decomposition for all levels, starting with the highest level of transformation (lower resolution data), the k-means algorithm is applied. The resultant cluster centers are projected as the initial centers of the data in the next haar level. The algorithm continues recursively until it is found out that the cluster assignments are not changing after applying the k-means algorithm for the successive haar levels.

In this chapter, we will be defining the main implementation of the combination of

CKM and the idea proposed by Lin et al. (2004). In the following chapters, 3.2.3 and 3.2.4, two different approaches that are based on that idea will be defined.

Before starting the solution of CKM-H1, at first the Haar wavelet decomposition for each time series is calculated. The transformed set of time series are stored in a matrix that has the size of n x t x r.

The formulation for minimizing the objective function in level h is as below:

$$(\textbf{CKM-H1}) \text{ Minimize } \sum_{i=1}^{n}\left[\sum_{p=1}^{k}\left(\sum_{q=1}^{t}\left(\sum_{j=1}^{n} D_{jiqh}X_{jpqh}\right)\right)T_{iph}\right] \tag{3.31}$$

$$\text{subject to: } \sum_{i=1}^{n} X_{ipqh} = 1, \qquad \forall p \in P, \forall q \in Q \tag{3.32}$$

$$\sum_{p=1}^{k} X_{ipqh} \leq 1, \qquad \forall i \in I, \forall q \in Q \tag{3.33}$$

$$\sum_{p=1}^{k} T_{iph} = 1, \qquad \forall i \in I \tag{3.34}$$

$$X_{ipqh} \in \{0,1\}, \qquad \forall i \in I, \forall p \in P, \forall q \in Q \tag{3.35}$$

$$T_{iph} \in \{0,1\}. \qquad \forall i \in I, \forall p \in P \tag{3.36}$$

When the optimization problem 3.31 is divided into two subproblems the formulations are as below:

$$(\textbf{CKM-H1-S1}) \text{ Minimize } \sum_{i=1}^{n}\left[\sum_{p=1}^{k}\left(\sum_{q=1}^{t}\left(\sum_{j=1}^{n} D_{jiqh}X_{jpqh}\right)\right)T_{iph}\right] \tag{3.37}$$

$$\text{subject to: } \sum_{p=1}^{k} T_{iph} = 1, \qquad \forall i \in I \tag{3.38}$$

$$T_{iph} \in \{0,1\}. \qquad \forall i \in I, \forall p \in P \tag{3.39}$$

$$(\textbf{CKM-H1-S2})\ \text{Minimize} \sum_{q=1}^{t}\left[\sum_{p=1}^{k}\left(\sum_{i=1}^{n}\left(\sum_{j=1}^{n} T_{jph} D_{jiqh}\right) X_{ipqh}\right)\right] \tag{3.40}$$

$$\text{subject to:} \sum_{i=1}^{n} X_{ipqh} = 1, \qquad \forall p \in P, \forall q \in Q \tag{3.41}$$

$$\sum_{p=1}^{k} X_{ipqh} \leq 1, \qquad \forall i \in I, \forall q \in Q \tag{3.42}$$

$$X_{ipqh} \in \{0, 1\}. \qquad \forall i \in I, \forall p \in P, \forall q \in Q \tag{3.43}$$

For this version of the algorithm, at first, it is started with Haar wavelet decomposition level r, that is the highest level in number, but the simplest and most transformed level of the data. Following this, the initialization is done as it is mentioned in the previous part. Then, new cluster assignments are made due to the distance matrix D_{jiqh} in level r (3.37). Due to new clusters, new centers in each timestamp q are calculated (3.40). The found centers are used as the initial centers for the next run and calculations continue until no better objective function value for level r can be found.

For the center and cluster assignments that give the minimum objective function value for level r, the centers in which $X_{ipqr} = 1$ become the new initial center for the cluster p at same timestamp q for Haar wavelet decomposition level $r - 1$. And the values of centers in each timestamp computed at the end of level r are projected onto level $r - 1$.

For the distance matrix in the iteration 2 of the algorithm, the distance matrix $D_{jiq(r-1)}$ is recalculated by forming the distance matrix between data in level $r - 1$ with the version of projected values of centers in each timestamp from level r to level $r - 1$ as $M_{iq(r)}$ to $M_{iq(r-1)}$.

This iteration continues until the end of level 1 and the indices are calculated based on the final assignments. Steps of CKM-H1 can be seen in Figure 3.4. The Pseudocode of the CKM-H1 is provided in Algorithm 2.

Step 0. Set $h = r$, Initialize $X_{0,ipqh}$.

Step 1. Run CKM-H1 for h. Find X_{ipqh} and T_{iph}.

Step 2. Use the final center assignments (X_{ipqh}) that gives the minimum objective function value (Z) as the initial center assignments for $h = h - 1$ $(X_{0,ipq(h-1)})$.

Step 3. If $X_{ipqh} = 1$, project the value of center points from M_{iqh} to $M_{iq(h-1)}$. Recalculate $D_{jiq(h-1)}$.

Step 4. $h = h - 1$, check h.

If $h \neq 1$, set $h = h - 1$, go to **Step 1** and continue.

If $h = 1$, repeat **Step 1** and STOP.

Figure 3.4: Steps of CKM-H1

Algorithm 2: CKM with Haar Wavelet Decomposition

Procedure `CKM-H1()`

input : k, I

output : T_{ip1}, X_{ipq1}

$h = r$

Initialize $X_{0,ipqh}$

repeat

  $Z_0 \leftarrow \infty$

  $\delta = 1000000$

  repeat

    CKM-H1-S1

    input : $D_{jiqh}, X_{0,ipqh}$

    output : T_{iph}

    $T_{0,iph} \leftarrow T_{iph}$

    CKM-H1-S2

    input : $D_{jiqh}, T_{0,iph}$

    output : X_{ipqh}

    $X_{0,ipqh} \leftarrow X_{ipqh}$

    $Z \leftarrow \sum_{q=1}^{t} \left[\sum_{p=1}^{k} \left(\sum_{i \in I_p} \left(\sum_{j \in I_p} T_{jph} D_{jiqh} \right) X_{ipqh} \right) \right]$

    $\delta = Z_0 - Z$

    $Z_0 \leftarrow Z$

  until $\delta < 0.000001$

  $X_{0,ipq(h-1)} \leftarrow X_{ipqh}$

  $M_{iq(h-1)} \leftarrow M_{iqh}$

  Recalculate D_{jiqh-1}

  $h \leftarrow h - 1$

until $h = 0$

end

The visualization of the solution of this problem can be seen in Figure 3.5 . As it can be seen in figure, the algorithm is started to be solved from the simplest level of the data and the complexity of the data improved gradually. In the clustering applied for the example data set in Figure 3.5, $k = 3$ and time series data in different clusters are represented with different colors. The black line represents the combination of centers in each timestamp.

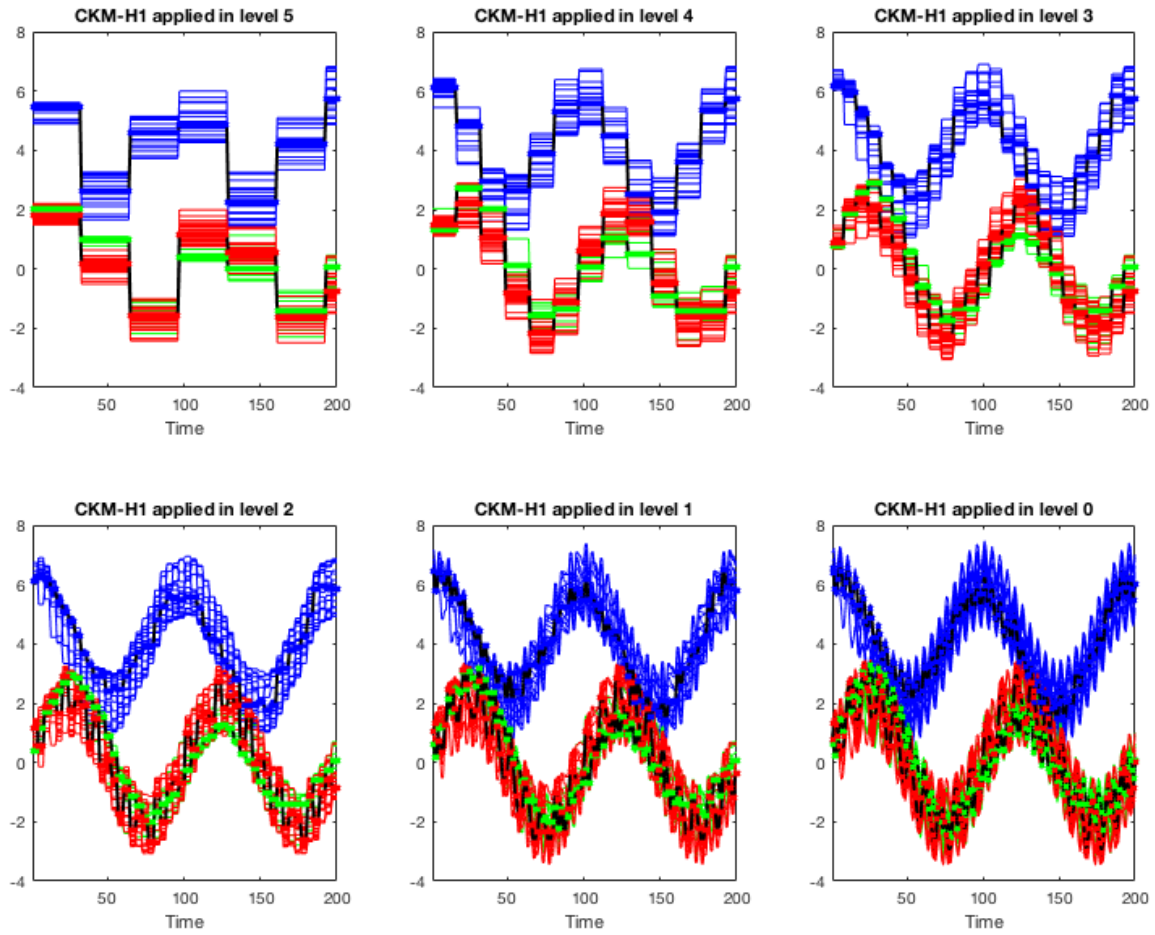

Figure 3.5: Application of CKM-H1 on an example data set

3.2.3 CKM with Haar Wavelet Decomposition without Projection

This method is very similar with CKM-H1 but different in **Step 3** of the algorithm, since the value of centers in level h are not projected to level $h-1$ and the original $D_{jiq(h-1)}$ matrix is used in the calculations. Steps of CKM with Haar Wavelet Decomposition without Projection (CKM-H2) algorithm can be seen in Figure 3.6. Also, the Pseudocode of the CKM-H2 is provided in Algorithm 3.

Step 0. Set h=r. Initialize $X_{0,ipqh}$.

Step 1. Run the CKM-H2 for h. Find T_{iph} and X_{ipqh}.

Step 2. Use the final center assignments (X_{ipqh}) that gives the minimum objective function value Z as the initial center assignments for $h = h-1$ $(X_{0,ipq(h-1)})$.

Step 3. $h = h-1$, check h.

If $h \neq 1$, go to **Step 1** and continue.
If $h = 1$, repeat **Step 1** and STOP.

Figure 3.6: Steps of CKM-H2

In Figure 3.7, there is a small data set with 5 time series. Since in CKM-H1, the values of centers found in level h is projected to level $h-1$ while in CKM-H2 the original values of the centers are preserved from level h to level $h-1$, there is a difference in the steps of both algorithms. In the figure, this change can be observed from the location of the centers.

Algorithm 3: CKM with Haar Wavelet Decomposition without Projection

Procedure `CKM-H2()`

input : k, I

output : T_{ip1}, X_{ipq1}

$h = r$

Initialize $X_{0,ipqh}$

repeat

  $Z_0 \leftarrow \infty$

  $\delta = 1000000$

  repeat

    CKM-H2-S1

    input : $D_{jiqh}, X_{0,ipqh}$

    output : T_{iph}

    $T_{0,iph} \leftarrow T_{iph}$

    CKM-H2-S2

    input : $D_{jiqh}, T_{0,iph}$

    output : X_{ipqh}

    $X_{0,ipqh} \leftarrow X_{ipqh}$

    $Z \leftarrow \sum_{q=1}^{t} \left[\sum_{p=1}^{k} \left(\sum_{i \in I_p} \left(\sum_{j \in I_p} T_{jph} D_{jiqh} \right) X_{ipqh} \right) \right]$

    $\delta = Z_0 - Z$

    $Z_0 \leftarrow Z$

  until $\delta < 0.000001$

  $X_{0,ipq(h-1)} \leftarrow X_{ipqh}$

  $h \leftarrow h - 1$

until $h = 0$

end

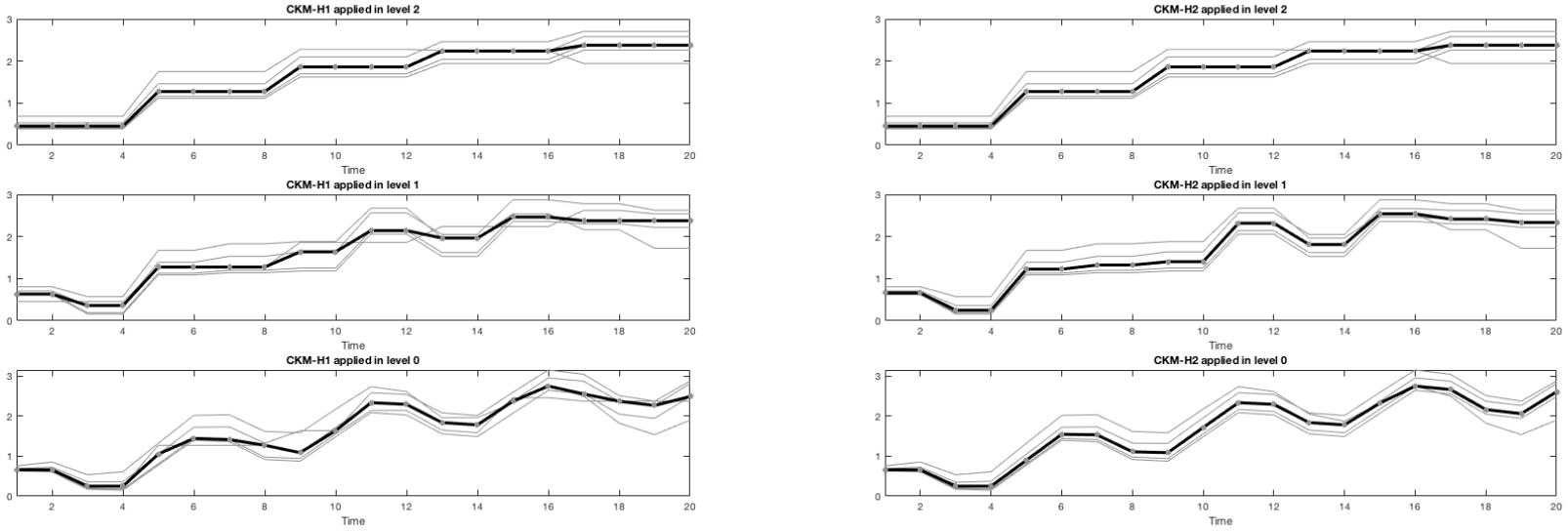

Figure 3.7: Application of CKM-H1 and CKM-H2 algorithms on an example data set

3.2.4 Search Based CKM with Haar Wavelet Decomposition

Another proposed algorithm is named search based CKM with Haar wavelet decomposition (CKM-SH). The main difference of CKM-SH to CKM-H1 is while projecting the values of centers in each timestamp q from Haar wavelet decomposition level h to $h-1$, a threshold level is used.

If $X_{ipqh} = 1$, check $\frac{\left|M_{iq(h-1)} - M_{iqh}\right|}{\left|M_{iqh}\right|}$. If it is lower than the threshold level, $M_{iq(h-1)}$ will take the value of M_{iqh} . In other words, M_{iqh} will be projected directly to $M_{iq(h-1)}$ and the distance matrix $D_{jiq(h-1)}$ will be recalculated based on this. But if this value is higher than the threshold level, $M_{iq(h-1)}$ will stay as it is. As the threshold level 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and 1 are used.

It is searched which threshold level provides the clustering that provides the best result in terms of used indices for the given value of k (Highest Silhouette Index, lowest C Index). The steps of algorithm can be seen in Figure 3.8

The steps of the algorithm can be also viewed from the flow chart in Figure 3.9. Also, the pseudocode of the algorithm can be seen in Algorithm 4.

Step 0. Initialize, $X_{0,ipqr}$ and take $Iteration = 1$. ($Iteration$ increase by one once the algorithm comes to Step 0.

Step 1. Set $threshold = 0.1 * Iteration$.

Step 2. Run the CKM-H1 algorithm for $h = r$ representation of the data. Find X_{ipqh} and T_{iph}.

Step 3. Use the final center assignments (X_{ipqh}) that gives the minimum objective function value Z as the initial center assignments for $h = h - 1$ $(X_{0,ipq(h-1)})$.

Step 4. If $X_{ipqh} = 1, \forall i, \forall p, \forall q$, check $\frac{\left|M_{iq(h-1)} - M_{iqh}\right|}{\left|M_{iqh}\right|}$.

If $\frac{\left|M_{iq(h-1)} - M_{iqh}\right|}{\left|M_{iqh}\right|} \leq threshold$, $M_{iq(h-1)}$=M_{iqh}.

If $\frac{\left|M_{iq(h-1)} - M_{iqh}\right|}{\left|M_{iqh}\right|} > threshold$, $M_{iq(h-1)}$ will stay as it is.

In either step after the check is done for $\forall i, \forall p, \forall q$, recalculate $D_{jiq(h-1)}$.

Step 5. $h = h - 1$, check h.

If $h \neq 1$, go to **Step 2** and continue.

If $h = 1$, run the CKM-H1 algorithm for $h = 1$, save X_{ipq1} and T_{ip1} and calculate Silhouette Index and C Index, then continue with **Step 6**.

Step 6. Check $threshold$.

If $threshold < 1$, go back to **Step 0**.

If $threshold = 1$, STOP. Take the X_{ipq1} and T_{ip1} as the optimal cluster and center assignments for the $threshold$ that gives the maximum Silhouette Index and minimum C Index.

Figure 3.8: Steps of CKM-SH

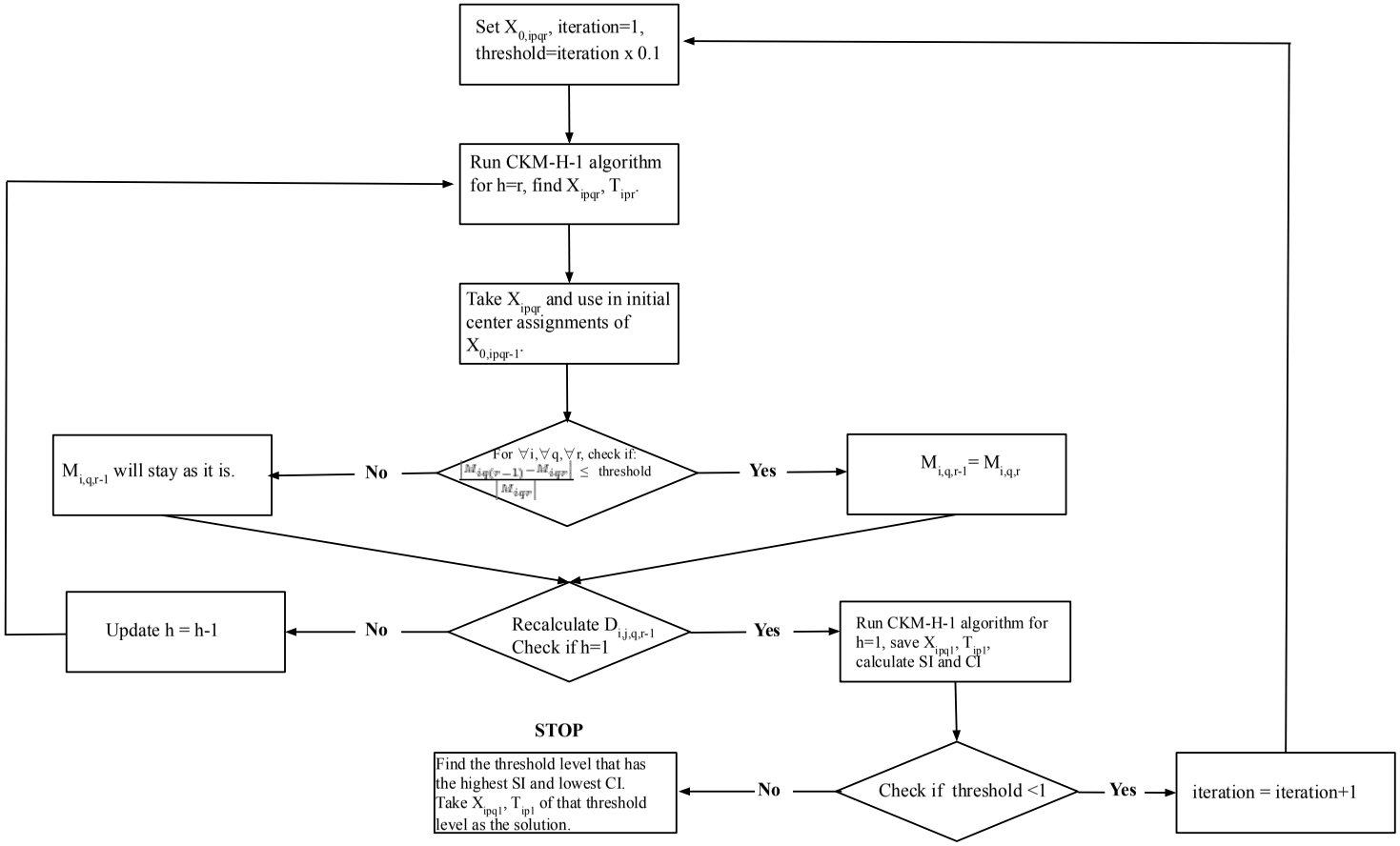

Figure 3.9: Flow chart of CKM-SH

Algorithm 4: Search Based CKM with Haar Wavelet Decomposition

Procedure `CKM-H2()`

input : k, I

output : T_{ip1}, X_{ipq1}

1 **for** *threshold=0:0.1:1*

2 $h = r$

3 Initialize $X_{0,ipqh}$

4 **repeat**

5 $Z_0 \leftarrow \infty$

6 $\delta = 1000000$

7 **repeat**

8 **CKM-H1-S1**

input : $D_{jiqh}, X_{0,ipqh}$

output : T_{iph}

9 $T_{0,iph} \leftarrow T_{iph}$

10 **CKM-H1-S2**

input : $D_{jiqh}, T_{0,iph}$

output : X_{ipqh}

11 $X_{0,ipqh} \leftarrow X_{ipqh}$

12 $Z \leftarrow \sum_{q=1}^{t} \left[\sum_{p=1}^{k} \left(\sum_{i \in I_p} \left(\sum_{j \in I_p} T_{jph} D_{jiqh} \right) X_{ipqh} \right) \right]$

13 $\delta = Z_0 - Z$

14 $Z_0 \leftarrow Z$

15 **until** $\delta < 0.000001$

16 $X_{0,ipq(h-1)} \leftarrow X_{ipqh}$

17 **if** $\frac{\left| M_{iq(r-1)} - M_{iqr} \right|}{|M_{iqr}|} \leq threshold$

18 $M_{iq(r-1)} \leftarrow M_{iqr}$

19 **end**

20 Recalculate $D_{jiq(h-1)}$

21 $h \leftarrow h - 1$

22 **until** $h = 0$

23 Calculate $SI(threshold)$, $CI(threshold)$

24 Save T_{ip1}, X_{ipq1}

25 **end for**

26 For threshold that gives $max(SI(threshold))$, $min(CI(threshold))$, T_{ip1}, X_{ipq1} is the solution.

end

CHAPTER 4

COMPUTATIONAL STUDY AND RESULTS

4.1 Data Sets

For the evaluation of the algorithms, data sets generated from two different data pools are used. The formation of the first data pool is initiated by the data set taken from the UCI repository (Dua and Graff, 2017), which is named as *Synthetic Control Chart Time Series Data Set* and generated by the process defined in Alcock et al. (1999). In addition to the time series data taken from UCI repository that includes six different patterns, time series data that includes eight other patterns are generated and added to the data set. The different patterns in the data set can be seen in Figure 4.1. The data with patterns 1, 2, 3, 4, 5 and 6 are taken from the repository and the others are generated. The new formed data pool will be called as *Synthetic Control Chart Time Series Data Pool (SCD)* in the later parts of this thesis. Each pattern includes 50 time series data. The common behavior of the patterns is the data points in time series in each pattern can take a value between -2 and 2.

The second data pool is generated with the idea of having time series data that includes periodicity. For this, three *trend terms*, three *periodicity terms* and two *error terms* are generated as the first step.

- **Trend Terms:** Each trend term has an increasing behavior. The graph of trend terms can be seen in Figure 4.2.
  Trend term 1: $T_1(t) = t^{\frac{31}{48}}$
  Trend term 2: $T_2(t) = t^{\frac{28}{48}}$
  Trend term 3: $T_3(t) = t^{\frac{24}{48}}$

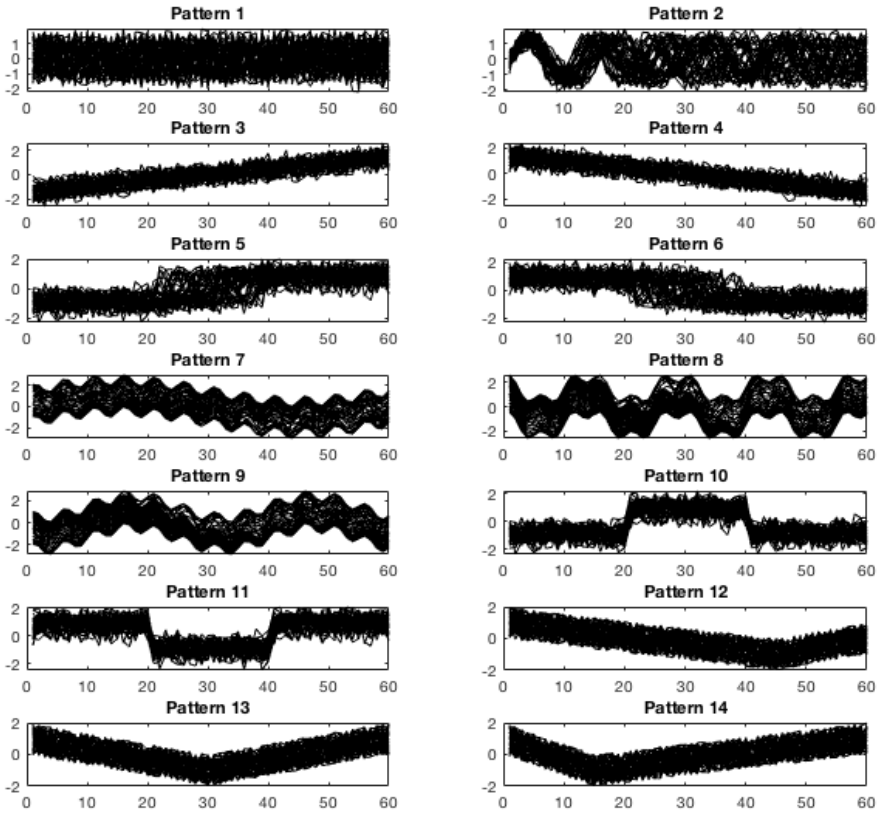

Figure 4.1: Synthetic Control Chart Time Series Data Pool

- **Periodicity Terms:** At first, sine and cosine waves are generated with sine term as $4sin\left(\frac{2\pi t}{75}\right)$ and cosine term as $cos\left(\frac{\pi t}{4}\right)$. Following that, the periodicity terms are formed. The graph of periodicity terms can be seen in Figure 4.2.

  Periodicity term 1: $P_1(t) = 4sin\left(\frac{2\pi t}{75}\right) + cos\left(\frac{\pi t}{4}\right)$
  Periodicity term 2: $P_2(t) = 4sin\left(\frac{2\pi t}{75}\right) * cos\left(\frac{\pi t}{4}\right)$
  Periodicity term 3: $P_3(t) = (4 + 4sin\left(\frac{2\pi t}{75}\right))^{\frac{1}{5}}$

- **Error Terms:**

  Error term 1: $E_1 \sim N(0, 1.5)$
  Error term 2: $E_2 \sim N(0, 3)$

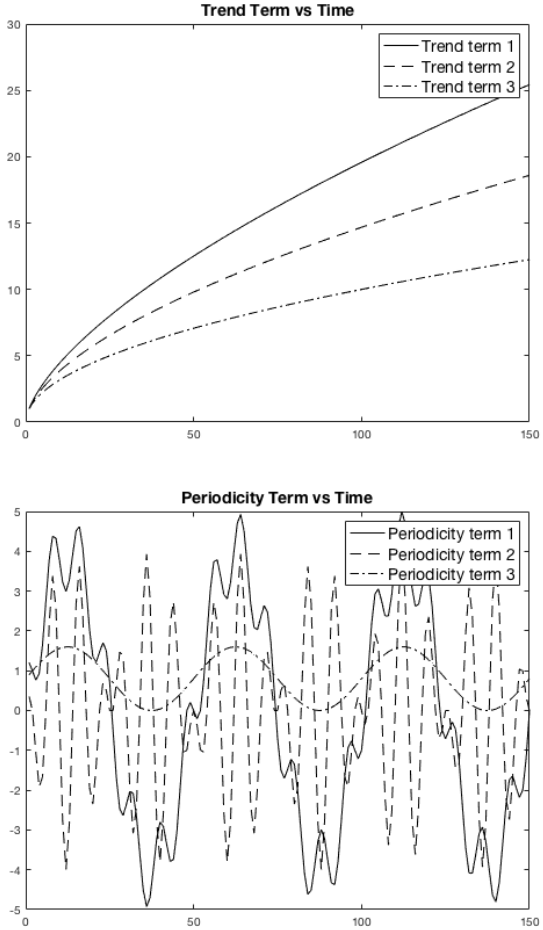

Figure 4.2: Trend terms and periodicity terms used in the generation of the data

Following the generation of trend, periodicity and error terms, time series data pool that includes 18 different patterns, in which each data in different pattern is formed

by the combination of the one trend term, one periodicity term and one error term are formed. In Figure 4.3, it can be seen the possible combinations of trend, periodicity and error terms for the generation of data in different patterns. Each pattern includes 20 time series data. This data pool will be called as *Data Pool Including Periodicity (PD)* in the later parts of this thesis.

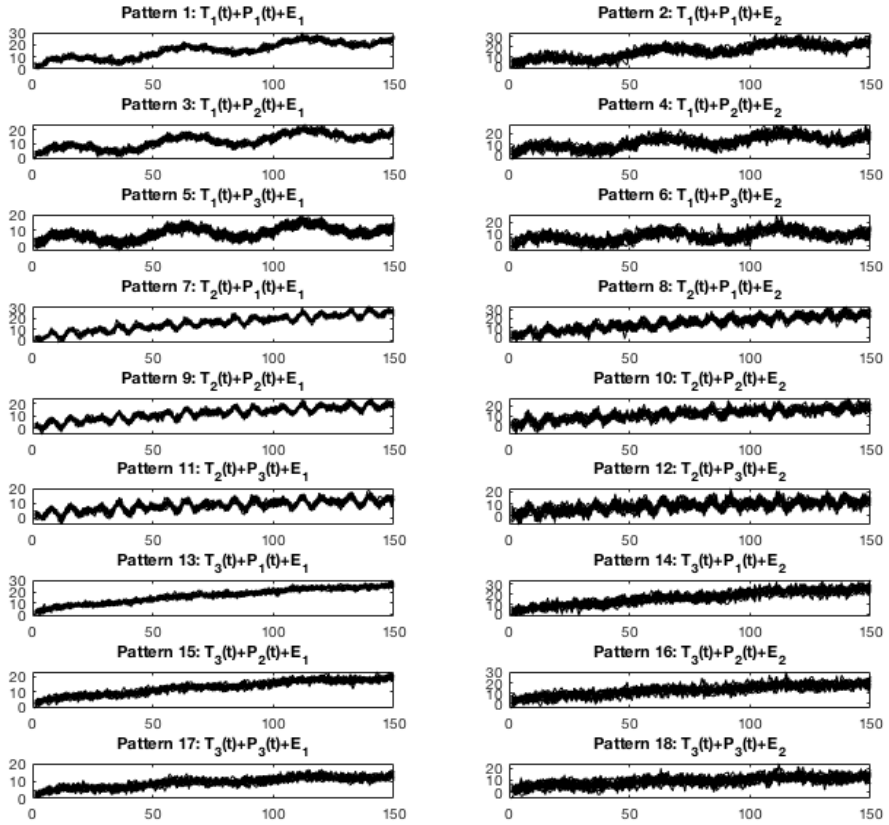

Figure 4.3: Generated Data Pool

For the evaluation of the algorithms, from both data pools, SCD and PD, new data sets with different true number of clusters are formed by taking combinations of 2, 3, 4 and 5 different data patterns. In other words, by combining data that follows the known patterns, we have an information of true cluster labels and this can help in the evaluation part of the algorithms due to both internal and external indices, that will be explained in Chapter 4.2. Data sets that are composed of 2, 3, 4 and 5 patterns will be called as the member of SCD-2, SCD-3, SCD-4, SCD-5 and PD-2, PD-3, PD-4, PD-5

respectively. For each data set, the internal and external indices, that will be detailed in Chapter 4.2, are calculated for comparing the performances of proposed algorithms CKM, CKM-H1, CKM-H2, CKM-SH and DKM. In tables 4.1 and 4.2, the number of patterns used in order to generate smaller data sets from SCD and PD, the number of data sets generated, the number of time series in each data set, and length of time series can be seen.

Table 4.1: Details of the data sets generated from SCD

Number of Combined Patterns	Number of Data Sets Generated	Number of Time Series in Each Data Set	Length of Each Time Series
2	91	100	60
3	364	150	60
4	1001	200	60
5	2002	250	60

For PD, there are some exclusions in the combined time series data with different patterns since it is not allowed to put groups of patterns 1-2, 3-4, 5-6, 7-8, 9-10, 11-12, 13-14, 15-16 and 17-18 in a data set and the reason is these groups include same trend and periodicity patterns but just different error terms.

Table 4.2: Details of the data sets generated from PD

Number of Combined Patterns	Number of Data Sets Generated	Number of Time Series in Each Data Set	Length of Each Time Series
2	144	40	150
3	672	60	150
4	2016	80	150
5	4032	100	150

The number of time series in the combined patterns to form the data sets generated from SCD and PD are same, that makes the formed data sets balanced. In order to test the algorithms, for the unbalanced data sets, 50 unbalanced data sets are generated by combining 3 and 4 patterns for both SCD and PD. The groups of data sets that are composed of 3 unbalanced patterns from SCD are called SCD-3-U and from PD are called PD-3-U. The same approach is followed for 4 patterns and the groups of data sets are called as SCD-4-U and PD-4-U. The summary of the formed unbalanced data sets can be seen in Table 4.3.

Table 4.3: Details of the unbalanced data sets

Data Sets in:	Number of Data Sets Generated	Number of Data Sets in Randomly Combined Patterns			
		Pattern A	Pattern B	Pattern C	Pattern D
SCD-3-U	50	40	30	20	-
PD-3-U	50	15	10	20	-
SCD-4-U	50	30	40	20	30
PD-4-U	50	15	10	20	5

In addition to the data sets generated from the explained data pools, a real data set, that will be called as PSA Data Set, is used in order to test the algorithms . The data set is taken from the study of Petricoin III et al. (2002) and includes time series data that includes samples of patients with prostate diseases. For each time series, t is equal to 15154. The data includes 4 true labels since it is taken from 4 patient groups that are:

- cancer with PSA level greater than 10 ng/ml (43 time series)
- cancer with PSA level within 4 ng/ml and 10 ng/ml (26 time series)
- benign with PSA level greater than 4 ng/ml (190 time series)
- no evidence of disease (63 time series)

The PSA data set can be seen in Figure 4.4.

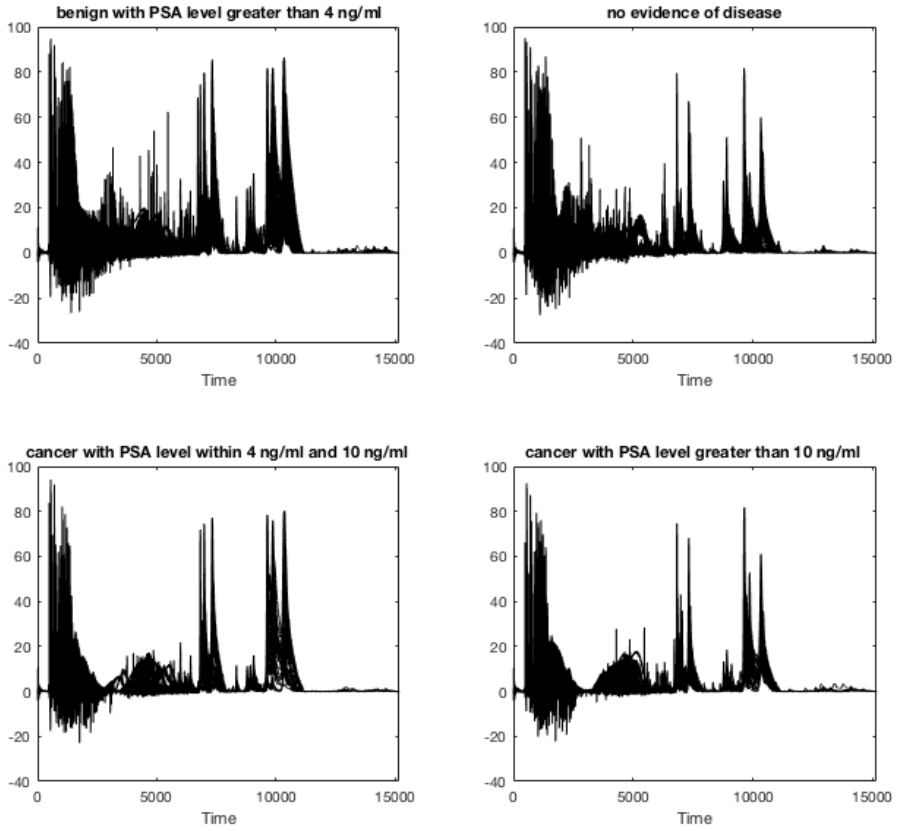

Figure 4.4: PSA Data Set

4.2 Performance Measures

As it was mentioned in the earlier chapters, clustering aims to group the unlabeled data into clusters in which the similarity of the data in the same clusters are maximized and the dissimilar samples are separated to different clusters. For measuring the cluster quality, there are two types of indices as *internal* and *external* indices. The external indices evaluates the resultant clusters by comparing it with pre-specified cluster labels that are supplied externally (Halkidi et al., 2001). The internal indices are evaluating the resultant clusters internally without the need of an external information. In this study, for the evaluation of proposed algorithms, three internal and two external measures are used.

4.2.1 External Measures

As the external measures, that are used for evaluating the resultant clusters with pre-specified cluster labels, Rand Index, Adjusted Rand Index and Purity are used.

4.2.1.1 Rand Index

Rand Index (RI) is used for measuring the consistency of clustering algorithm results (X) and real clusters (Y). Rand Index can be calculated with the following formula:

$$RI = \frac{a + d}{a + b + c + d}$$

a: Pairs that have same label in Y and also are in the same cluster in X.
b: Pairs that have same label in Y but are assigned to different clusters in X.
c: Pairs that are assigned to the same cluster in X but have different labels in Y.
d: Pairs that are assigned to different clusters in X and also have different labels in Y.

The value of Rand Index is changing between 0 and 1, in which, 1 indicates that the algorithm clustered the data set same as its actual labels. In other words, the greater the Rand Index, the better that the algorithm results fit with the actual result (Ansari et al., 2015a).

4.2.1.2 Adjusted Rand Index

Adjusted Rand Index (ARI) is suggested since it was found out that, in an accurate clustering, the RI is not allowing randomness. ARI can take negative values too and like RI, the more the ARI value, the more successful the clustering algorithm. When ARI takes negative values, that indicates random partition is better than the clustering applied.

Based of the contingency table of two partitions X and Y, the ARI is formulated as below:

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i.}}{2} * \sum_j \binom{n_{.j}}{2} / \binom{N}{2}\right]}{\frac{1}{2}\left[\sum_i \binom{n_{i.}}{2} * \sum_j \binom{n_{.j}}{2}\right] - \left[\sum_i \binom{n_{i.}}{2} * \sum_j \binom{n_{.j}}{2}\right] / \binom{N}{2}}$$

n_{ij}: The number of points that are common in cluster i of partition X and cluster j of partition Y.
N: Total number of samples in the dataset.
$n_{i.}$: Column sums are taken for row i in the contingency table.
$n_{.j}$: Row sums are taken for column j in the contingency table.

4.2.1.3 Purity

Purity measures how well the data with the same labels are clustered (Rendón et al., 2011). The value of purity can be between 0 and 1. If purity equals to 1, it indicates that, data with the same labels are in the same clusters without any exception.

$$P_j = \frac{1}{N_j} \max_{i \epsilon C} n_j^i$$

$$Purity = \sum_j \frac{N_j}{N} P_j$$

C: Given set of classes of samples (known from the real data).
N: Total number of samples in the dataset.
N_j: Total number of samples in cluster j.

n_j^i: The total number of samples in cluster j with class i.
P_j: Calculation the fraction of the samples whose class are majority in the cluster j.

4.2.2 Internal Measures

As the internal measures, that are used for evaluating the resultant clusters without a need of an external information, Silhouette Index and C Index are used.

4.2.2.1 Silhouette Index

Silhouette Index is used for measuring clustering quality when the real clusters of data are not known. It is both measuring how well the data is clustered and how well the compactness of the clustered data (Han et al., 2011). It is formulated in Rousseeuw (1987) as below:

$$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

$S(i)$: The Silhouette Index for sample i.
$a(i)$: The average dissimilarity of i to the other samples in the cluster that i is in.
$d(i, C)$: The average dissimilarity of i to the samples in another cluster C.
$b(i)$: $\min_C d(i, C)$

The value of the Silhouette Index is changing between -1 and 1. When $S(i)$ is close to 1, that means, the dissimilarity of i to the other samples in the same cluster ($a(i)$) is smaller than the minimum dissimilarity of i with the samples in another cluster ($b(i)$). When $a(i)$ is much more larger than $b(i)$, the worst $S(i)$ is found, that is -1 and it is indicating the within cluster distance is much more higher than the between cluster distance, that means the sample i is more close to the samples in another cluster rather than the samples that it's in the same cluster (Rousseeuw, 1987).

So, it can be concluded that sample i is clustered well, when the Silhouette Index is close to 1. For measuring how well the data set is clustered, mean $S(i)$ can be used. Han et al. (2011) also stated that for measuring the clustering quality of an algorithm on a data set, the average Silhouette Index value of samples can be taken. For the

average Silhouette Index, the more it is close to 1, the more clusters are compact and clearly separated.

In their paper, Wang et al. (2009) used the Silhouette Index for comparing the performances of different clustering algorithms. In their study, they formed a graph including the number of clusters (k) versus the average Silhouette Index obtained by applying different clustering algorithms. As the most successful algorithm, they have selected the algorithm that has the highest average Silhouette Index value in the formed graph, since the quality of clustering gets better with a larger average Silhouette Index. Chen et al. (2002) also used average Silhouette Index for comparing the clustering quality of different algorithms in their data set and they have indicated that if a clustering algorithm has smaller average Silhouette Index, it can be said that for this algorithm there are more misclassified data in comparison to the algorithms that have larger average Silhouette Index.

Another use case of average Silhouette Index versus number of clusters (k) graph is to find the optimal number of clusters by selecting the k that has the largest average Silhouette Index value (Ansari et al., 2015b).

4.2.2.2 C Index

The C Index is firstly defined in Hubert and Levin (1976) and it is calculated by the formula below:

$$C = \frac{S - S_{min}}{S_{max} - S_{min}}$$

S: Sum of distances of all pairs of samples in clusters (let's say m is the number of these pairs).
S_{min}: Sum of m smallest distances of pairs for the samples in the dataset.
S_{max}: Sum of m largest distances of pairs for the samples in the dataset.

C Index becomes close to 0 when S is very close to S_{min}, that indicates the sum of within cluster distances are close to S_{min}, which is an indicator of accurate clustering. C Index becomes close to 1 when S is very close to S_{max}, that indicates the clustering could not be done for the nearest samples. In their paper, (Roux, 2006) indicated that

for C Index, only the within cluster distances are taken into account, which makes the C Index a compactness measure and the lower the C Index for an algorithm, the better the data partitioned.

4.3 Computational Experiments for Measuring the Performances of the Algorithms

For the given data sets, the performances of the algorithms are compared in two ways. Firstly, the proposed algorithms are compared with the algorithm called discrete k-median clustering (DKM) proposed by Seref et al. (2014), in which the cluster centers are selected among an existing sample in the clusters. In the second way, the algorithm performances are compared within each other.

The number of patterns used for the generation of the data set are giving the true cluster numbers and for the true cluster numbers, since it is known from which pattern the data is taken, the external indices can be calculated. In Table 4.10, it can be seen which indices are calculated for data sets with different k values. In the table, "x" denotes, the related index is calculated.

Table 4.4: Indices calculated for different true number of clusters and k values

Indices	True Number of Clusters=2 (SCD-2, PD-2)						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
Silhouette Index	x	x	x	x	x	x	x
C Index	x	x	x	x	x	x	x
Rand Index	x						
Adjusted Rand Index	x						
Purity	x						

Indices	True Number of Clusters=3 (SCD-3, PD-3)						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
Silhouette Index	x	x	x	x	x	x	x
C Index	x	x	x	x	x	x	x
Rand Index		x					
Adjusted Rand Index		x					
Purity		x					

Indices	True Number of Clusters=4 (SCD-4, PD-4)						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
Silhouette Index	x	x	x	x	x	x	x
C Index	x	x	x	x	x	x	x
Rand Index			x				
Adjusted Rand Index			x				
Purity			x				

Indices	True Number of Clusters=5 (SCD-5, PD-5)						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
Silhouette Index	x	x	x	x	x	x	x
C Index	x	x	x	x	x	x	x
Rand Index				x			
Adjusted Rand Index				x			
Purity				x			

4.4 Initialization of the Algorithms

For each data set for given k, at first 500 initialization are made for the first center assignments and each algorithm is run 500 times with the same 500 initialization. Since the center selection in DKM is conducted by selecting an existing time series as the initial center while in CKM, CKM-H1, CKM-H2 and CKM-SH the centers of each cluster can be selected from different time series for each timestamp, in order to imitate the initialization, the steps in Figure 4.5 below are followed.

Step 1. For the given number k and $the\ initialization\ number\ =\ 1$, random partitioning of time series data to the k clusters are conducted.

Step 2. For **DKM**, the cluster centers of each clusters are selected as the time series that has the closest total distance to the other time series in the cluster.

Step 2. For **CKM**, **CKM-H1**, **CKM-H2**, **CKM-SH**, the cluster centers are selected for each timestamp by minimizing the total distances of points of time series in each timestamp to the center points.

Step 3. Check $the\ initialization\ number$.

If $the\ initialization\ number$ < 500, go back to **Step 1** by increasing the initialization number by 1 and continue.

If $the\ initialization\ number$ = 500, STOP.

Figure 4.5: Steps of initialization

The results of the 500 initializations are saved, and each algorithm run 500 times by starting with these initializations. As the final center and cluster assignments for each algorithm, the runs starting with the initialization that gives the minimum objective function value is taken. Following that, the indices are calculated.

In Figure 4.6, it can be seen how the center selection can be different in a cluster between DKM and CKM based algorithms. The bold line in DKM graph is indicating the center of the cluster, the dots in the CKM based graph is indicating the centers in each timestamp.

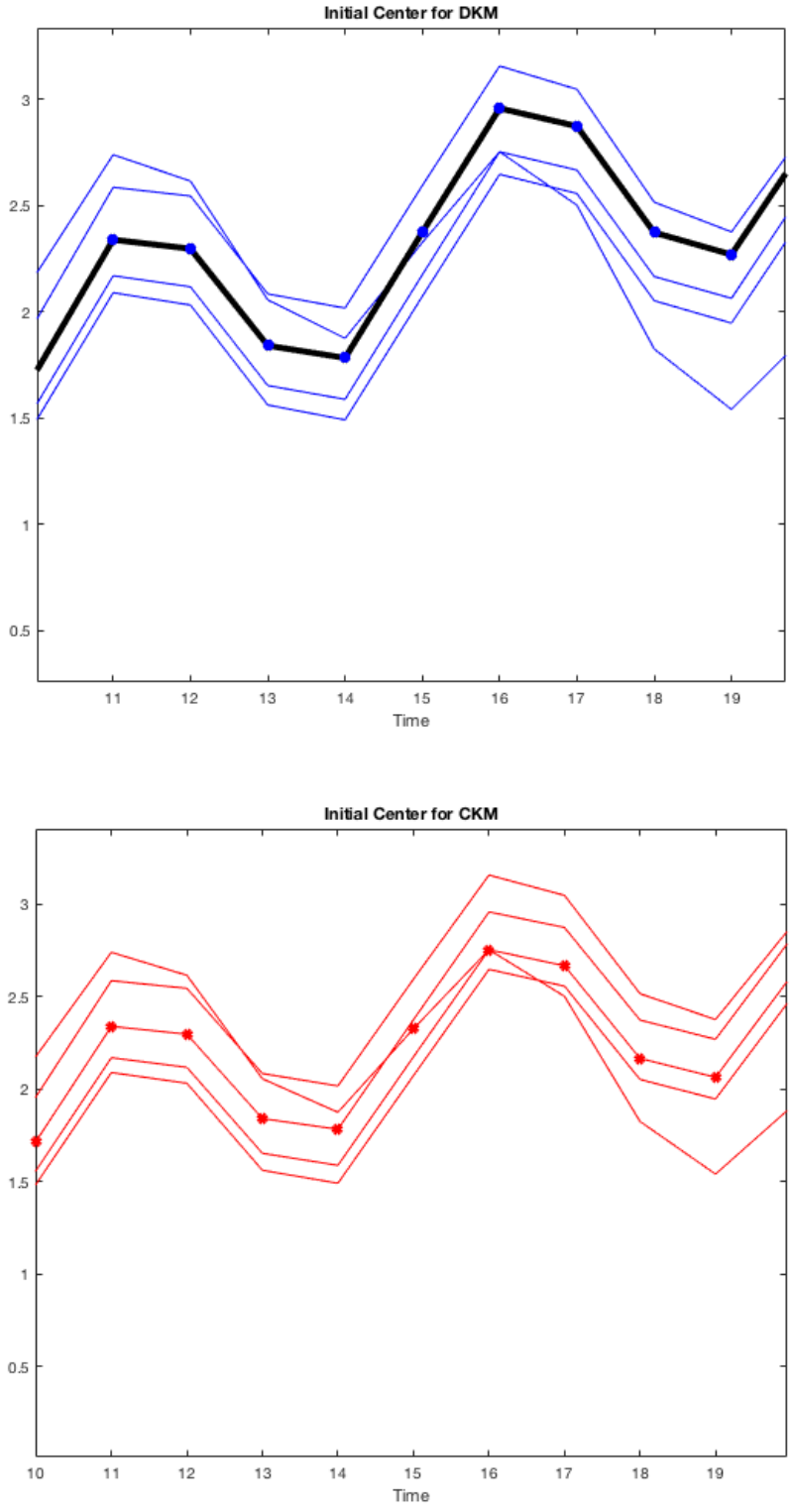

Figure 4.6: Initial center selection in an example cluster for DKM and CKM

4.5 Computational Results

Experimental studies were carried out on 64-bit Windows 10 PC with 3.6 GHz 6 core Intel Xeon E-2246G processor and 16 GB RAM. All of the algorithms are coded in MATLAB.

As it was mentioned in Section 4.3, after the application of the algorithms, the indices are calculated for the formed data sets in SCD-2, SCD-3, SCD-4, SCD-5 and PD-2, PD-3, PD-4, PD-5.

Since there is a huge number of data (3458 total data sets generated from SCD and 6844 total data sets generated from PD), in order to express and evaluate the results for Silhouette Index and C Index, a ranking approach is used.

For each data set, the Silhouette Index matrix is formed that includes the Silhouette indices for DKM, CKM, CKM-H1, CKM-SH and CKM-H2 in $k = 2$, $k = 3$, $k = 4$, $k = 5$, $k = 6$, $k = 7$ and $k = 8$. For each Silhouette Index matrix, for each column (for a k value), the Silhouette indices are ranked from 1 to 5. This ranking is done by giving 1 to the cell that has the highest Silhouette Index and going on until 5, that is the ranking score given to the lowest Silhouette Index. Same approach is also followed for the C Index. For C Index, in a column, the smallest number is ranked as 1, while the largest number is ranked as 5 since the smaller the C Index, the better the clustering algorithm applied. In order to see the ranking method with examples for Silhouette Index and C Index for a data set from SCD-2, Table 4.5 and Table 4.6 can be checked respectively. This ranking approach is followed for all of the data sets in SCD-2, SCD-3, SCD-4, SCD-5 and PD-2, PD-3, PD-4, PD-5.

With the ranking method, for each data set, we have ranking matrices for Silhouette indices and C indices. The results are evaluated by taking average of the ranking matrices of the data sets in SCD-2, SCD-3, SCD-4, SCD-5 and PD-2, PD-3, PD-4, PD-5. In other words, for each data pool SCD and PD, we have 4 Silhouette Index average ranking matrices and 4 C Index average ranking matrices. Due to the resultant average matrices, it can be evaluated for which algorithm and given value of k, the Silhouette indices and C indices give the highest values and lowest values. At that point, taking the average of ranking matrices indicates, what is the rank that the related

Table 4.5: Silhouette Index Table and Silhouette Index Ranking Table for an Example Data Set in SC-2

Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	0.1996	0.1746	0.1492	0.1532	0.1767	0.1816	0.1510
CKM	0.2272	0.1916	0.1592	0.1771	0.1908	0.1895	0.1799
CKM-H1	0.2272	0.1933	0.1654	0.1540	0.1792	0.1840	0.1748
CKM-H2	0.2215	0.1933	0.1644	0.1595	0.1879	0.1736	0.1793
CKM-SH	0.2272	0.1935	0.1644	0.1821	0.1973	0.1975	0.1902

Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3	5	4	5	5	4	5
CKM	1	4	3	2	2	2	2
CKM-H1	1	3	1	4	4	3	4
CKM-H2	2	2	2	3	3	5	3
CKM-SH	1	1	2	1	1	1	1

Table 4.6: C Index Table and C Index Ranking Table for an Example Data Set in SC-2

Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	0.2314	0.1805	0.1923	0.1986	0.1771	0.1835	0.2154
CKM	0.1716	0.1580	0.1642	0.1578	0.1483	0.1474	0.1575
CKM-H1	0.1716	0.1555	0.1534	0.1581	0.1554	0.1472	0.1547
CKM-H2	0.1804	0.1576	0.1521	0.1595	0.1525	0.1794	0.1664
CKM-SH	0.1716	0.1544	0.1521	0.1492	0.1414	0.1384	0.1409

Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3	5	4	5	5	5	5
CKM	1	4	3	2	2	3	3
CKM-H1	1	2	2	3	4	2	2
CKM-H2	2	3	1	4	3	4	4
CKM-SH	1	1	1	1	1	1	1

algorithm is taking on average.

For SCD-2, SCD-3, SCD-4, SCD-5, the average of rankings of Silhouette indices can be seen in Table A.1 and the rankings of averages of Silhouette indices' rankings can be seen in Table 4.7. The average of rankings of C indices for SCD-2, SCD-3, SCD-4, SCD-5 can be seen in Table A.2 and the rankings of averages of C indices' rankings can be seen in Table 4.8.

For PD-2, PD-3, PD-4, PD-5, the average of rankings of Silhouette indices can be seen in Table A.3 and the rankings of averages of Silhouette indices' rankings can be seen in Table 4.9. The average of rankings of C indices for PD-2, PD-3, PD-4, PD-5 can be seen in Table A.4 and the rankings of averages of C indices' rankings can be seen in Table 4.10.

For SCD-3-U, SCD-4-U and PD-3-U, PD-4-U, the average of rankings of Silhouette indices can be seen in tables 4.11 and 4.12 and the rankings of averages of Silhouette indices' rankings can be seen in tables A.5 and A.6. The average of rankings of C indices for SCD-3-U, SCD-4-U and PD-3-U, PD-4-U, can be seen in tables 4.13 and 4.14 and the rankings of averages of C indices' rankings can be seen in tables A.7 and A.8.

For PSA data set, the real Silhouette Index and C Index values for different k values can be seen in tables 4.15 and 4.16.

Table 4.7: Rankings of averages of Silhouette Indices' rankings for data sets in SCD-2, SCD-3, SCD-4, SCD-5

Rankings of averages of Silhouette Indices' rankings for data sets in SCD-2							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	4	2	2	2	2	2	3
CKM-H1	3	4	4	4	4	4	4
CKM-H2	2	3	3	3	3	3	2
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of Silhouette Indices' rankings for data sets in SCD-3							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	2	3	3	4	4	4	4
CKM-H1	4	4	2	2	3	3	3
CKM-H2	3	2	4	3	2	2	2
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of Silhouette Indices' rankings for data sets in SCD-4							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	4	2	2	2	2	2	4
CKM-H1	2	3	4	3	3	4	3
CKM-H2	3	4	3	4	4	3	2
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of Silhouette Indices' rankings for data sets in SCD-5							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	3	2	2	2	2	3	4
CKM-H1	2	3	3	3	3	2	3
CKM-H2	4	4	4	4	4	4	2
CKM-SH	1	1	1	1	1	1	1

Table 4.8: Rankings of averages of C Indices' rankings for data sets in SCD-2, SCD-3, SCD-4, SCD-5

Rankings of averages of C Indices' rankings for data sets in SCD-2							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	4	2	2	2	2	2	2
CKM-H1	3	4	4	4	4	4	4
CKM-H2	2	3	3	3	3	3	3
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of C Indices' rankings for data sets in SCD-3							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	2	3	2	2	3	2	2
CKM-H1	4	4	3	4	4	4	4
CKM-H2	3	2	4	3	2	3	3
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of C Indices' rankings for data sets in SCD-4							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	2	2	2	2	2	2	2
CKM-H1	4	4	4	4	4	4	4
CKM-H2	3	3	3	3	3	3	3
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of C Indices' rankings for data sets in SCD-5							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	2	2	2	2	2	2	2
CKM-H1	4	4	4	4	4	4	4
CKM-H2	3	3	3	3	3	3	3
CKM-SH	1	1	1	1	1	1	1

Table 4.9: Rankings of averages of Silhouette Indices' rankings for data sets in PD-2, PD-3, PD-4, PD-5

Rankings of averages of Silhouette Indices' rankings for data sets in PD-2							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2	4	4	4	4	4	4
CKM	1	2	2	2	2	2	2
CKM-H1	1	5	5	5	5	5	5
CKM-H2	1	3	3	3	3	3	3
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of Silhouette Indices' rankings for data sets in PD-3							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	3	5	5	5	5	5
CKM	4	1	2	2	2	2	2
CKM-H1	3	2	4	4	4	4	4
CKM-H2	2	1	3	3	3	3	3
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of Silhouette Indices' rankings for data sets in PD-4							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	4	4	4	2	2	2	2
CKM-H1	2	3	3	4	4	3	3
CKM-H2	3	2	2	3	3	4	4
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of Silhouette Indices' rankings for data sets in PD-5							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	4	4	4	2	2	2	2
CKM-H1	2	3	3	4	4	3	3
CKM-H2	3	2	2	3	3	4	4
CKM-SH	1	1	1	1	1	1	1

Table 4.10: Rankings of averages of C Indices' rankings for data sets in PD-2, PD-3, PD-4, PD-5

Rankings of averages of C Indices' rankings for data sets in PD-2							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3	5	5	4	4	4	3
CKM	2	2	2	2	2	2	2
CKM-H1	1	4	4	5	5	5	5
CKM-H2	1	3	3	3	3	3	4
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of C Indices' rankings for data sets in PD-3							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	3	4	2	2	2	2	2
CKM-H1	4	3	3	4	4	4	4
CKM-H2	2	2	4	3	3	3	3
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of C Indices' rankings for data sets in PD-4							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	4	4	4	2	2	2	2
CKM-H1	2	3	2	4	3	3	3
CKM-H2	3	2	3	3	4	4	4
CKM-SH	1	1	1	1	1	1	1

Rankings of averages of C Indices' rankings for data sets in PD-5							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	5	5
CKM	4	4	3	2	2	2	2
CKM-H1	3	3	2	3	3	3	3
CKM-H2	2	2	4	4	4	4	4
CKM-SH	1	1	1	1	1	1	1

Table 4.11: Rankings of averages of Silhouette Indices' rankings for data sets in SCD-3-U, SCD-4-U

Rankings of averages of Silhouette Indices' rankings for data sets in SCD-3-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	5	5	5	4	3
CKM	4	2	4	4	4	3	1
CKM-H1	1	4	2	3	3	2	2
CKM-H2	3	3	3	2	2	3	2
CKM-SH	2	1	1	1	1	1	1

Rankings of averages of Silhouette Indices' rankings for data sets in SC-4-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	4	4	4	4	5	4	5
CKM	3	3	3	3	4	2	4
CKM-H1	1	2	3	2	2	3	2
CKM-H2	2	3	2	2	3	2	3
CKM-SH	1	1	1	1	1	1	1

Table 4.12: Rankings of averages of Silhouette Indices' rankings for data sets in PD-3-U, PD-4-U

Rankings of averages of Silhouette Indices' rankings for data sets in PD-3-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3	2	5	5	3	4	5
CKM	2	1	3	4	4	5	2
CKM-H1	4	3	4	3	2	2	3
CKM-H2	2	5	2	2	3	3	4
CKM-SH	1	4	1	1	1	1	1

Rankings of averages of Silhouette Indices' rankings for data sets in PD-4-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	5	5	4	4	5	4	5
CKM	4	4	3	3	4	3	4
CKM-H1	3	1	2	2	3	2	3
CKM-H2	2	3	3	2	2	2	2
CKM-SH	1	2	1	1	1	1	1

Table 4.13: Rankings of averages of C Indices' rankings for data sets in SCD-3-U, SCD-4-U

Rankings of averages of C Indices' rankings for data sets in SCD-3-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3	5	3	5	5	3	4
CKM	1	2	2	3	3	2	3
CKM-H1	5	4	4	4	4	4	4
CKM-H2	4	3	3	2	2	2	2
CKM-SH	2	1	1	1	1	1	1

Rankings of averages of C Indices' rankings for data sets in SCD-4-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3	5	4	5	5	4	5
CKM	1	1	2	3	2	2	2
CKM-H1	4	3	5	4	4	3	4
CKM-H2	3	4	3	2	3	2	3
CKM-SH	2	2	1	1	1	1	1

Table 4.14: Rankings of averages of C Indices' rankings for data sets in PD-3-U, PD-4-U

Rankings of averages of C Indices' rankings for data sets in PD-3-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3	3	5	4	4	3	3
CKM	2	1	4	3	4	5	2
CKM-H1	4	4	2	2	2	2	3
CKM-H2	2	3	3	4	3	4	4
CKM-SH	1	2	1	1	1	1	1

Rankings of averages of C Indices' rankings for data sets in PD-4-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3	2	4	5	5	5	5
CKM	1	2	3	3	4	4	4
CKM-H1	4	3	5	4	3	3	3
CKM-H2	2	1	2	2	2	2	2
CKM-SH	2	1	1	1	1	1	1

Table 4.15: Silhouette Index Table for PSA data set

Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	0.3787	0.3618	0.3140	0.2977	0.2668	0.2459	0.2148
CKM	0.4044	0.3766	0.3689	0.3022	0.2692	0.2461	0.2253
CKM-H1	0.4069	0.3771	0.3223	0.2984	0.2687	0.2302	0.2056
CKM-H2	0.4044	0.3766	0.3223	0.2984	0.2565	0.2306	0.2053
CKM-SH	0.4080	0.3771	0.3223	0.2984	0.2697	0.2759	0.2375

Table 4.16: C Index Table for PSA data set

Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	0.1667	0.0763	0.0706	0.0750	0.0792	0.0847	0.0947
CKM	0.1196	0.0695	0.0527	0.0520	0.0819	0.0840	0.0927
CKM-H1	0.1165	0.0695	0.0680	0.0763	0.0812	0.0932	0.0983
CKM-H2	0.1196	0.0695	0.0680	0.0763	0.0889	0.0931	0.1025
CKM-SH	0.1156	0.0695	0.0680	0.0763	0.0812	0.0580	0.0684

For evaluation of the performances of the algorithms based on the external indices, that are only calculated for the true number of clusters for a data set, the average values of the indices for each data set in SCD-2, SCD-3, SCD-4 and SCD-5 and for each data set in PD-2, PD-3, PD-4 and PD-5 are calculated. The resultant tables can be seen in Table 4.17 and Table 4.18.

For the unbalanced data sets, the average values of the indices for data sets in SCD-3-U, SCD-4-U, PD-3-U and PD-4-U can be seen in Table 4.19.

Also for PSA data set, the external indices are calculated for k=4, that is the true cluster number. The results can be seen in Table 4.20

Table 4.17: Averages of external indices for data sets in SCD-2, SCD-3, SCD-4, SCD-5

Algorithms	Average Adjusted Rand Index for the data sets in:			
	SCD-2	SCD-3	SCD-4	SCD-5
DKM	0.5440	0.5507	0.4912	0.4223
CKM	0.6868	0.6568	0.6306	0.5937
CKM-H1	0.7136	0.7008	0.6792	0.6491
CKM-H2	0.6867	0.6683	0.6456	0.5986
CKM-SH	0.6857	0.6601	0.6382	0.5892

Algorithms	Average Rand Index for the data sets in:			
	SCD-2	SCD-3	SCD-4	SCD-5
DKM	0.7718	0.7958	0.8024	0.8062
CKM	0.8432	0.8408	0.8535	0.8611
CKM-H1	0.8566	0.8625	0.8747	0.8808
CKM-H2	0.8431	0.8477	0.8615	0.8635
CKM-SH	0.8426	0.8432	0.8579	0.8594

Algorithms	Average Purity for the data sets in:			
	SCD-2	SCD-3	SCD-4	SCD-5
DKM	0.9332	0.9133	0.7049	0.6352
CKM	0.9755	0.9933	0.7948	0.7556
CKM-H1	0.9753	0.9867	0.8285	0.7943
CKM-H2	0.9767	0.9933	0.8091	0.7640
CKM-SH	0.9743	0.9933	0.8050	0.7573

Table 4.18: Averages of external indices for data sets in PD-2, PD-3, PD-4, PD-5

Algorithms	Average Adjusted Rand Index for the data sets in:			
	PD-2	PD-3	PD-4	PD-5
DKM	0.8385	0.8334	0.6847	0.6106
CKM	0.9442	1.0000	0.9575	0.8101
CKM-H1	0.9448	1.0000	0.9887	0.7801
CKM-H2	0.9457	1.0000	0.9837	0.7653
CKM-SH	0.9447	1.0000	0.9850	0.7800

Algorithms	Average Rand Index for the data sets in:			
	PD-2	PD-3	PD-4	PD-5
DKM	0.9192	0.9254	0.8762	0.8645
CKM	0.9721	1.0000	0.9821	0.9305
CKM-H1	0.9721	1.0000	0.9953	0.9199
CKM-H2	0.9724	1.0000	0.9932	0.9146
CKM-SH	0.9717	1.0000	0.9937	0.9195

Algorithms	Average Purity for the data sets in:			
	PD-2	PD-3	PD-4	PD-5
DKM	0.9332	0.9117	0.8003	0.7262
CKM	0.9755	1.0000	0.9681	0.8482
CKM-H1	0.9753	1.0000	0.9922	0.8307
CKM-H2	0.9767	1.0000	0.9890	0.8226
CKM-SH	0.9743	1.0000	0.9892	0.8270

Table 4.19: Averages of external indices for data sets in SCD-3-U, SCD-4-U, PD-3-U, PD-4-U

Algorithms	Average Adjusted Rand Index for the data sets in			
	SCD-3-U	SCD-4-U	PD-3-U	PD-4-U
DKM	0.1172	0.2844	0.8388	0.0186
CKM	0.2845	0.4220	0.9165	0.1098
CKM-H1	0.4412	0.4262	0.9165	0.2171
CKM-H2	0.3257	0.4022	0.9165	0.1062
CKM-SH	0.2798	0.4039	0.9165	0.1171

Algorithms	Average Rand Index for the data sets in			
	SCD-3-U	SCD-4-U	PD-3-U	PD-4-U
DKM	0.5998	0.7233	0.9261	0.5998
CKM	0.6722	0.7770	0.9885	0.6511
CKM-H1	0.7296	0.7707	0.9885	0.6712
CKM-H2	0.6912	0.7699	0.9885	0.6506
CKM-SH	0.6707	0.7702	0.9885	0.6552

Algorithms	Average Purity for the data sets in			
	SCD-3-U	SCD-4-U	PD-3-U	PD-4-U
DKM	0.5333	0.6150	0.9111	0.4360
CKM	0.6400	0.6917	0.9817	0.5400
CKM-H1	0.7556	0.6800	0.9817	0.6200
CKM-H2	0.6778	0.6800	0.9817	0.5320
CKM-SH	0.6511	0.6783	0.9817	0.5360

Table 4.20: External indices for PSA data set

Algorithms	External Indices		
	ARI	RI	Purity
DKM	0.0065	0.5471	0.5901
CKM	0.0540	0.5571	0.6242
CKM-H1	0.0102	0.5492	0.5901
CKM-H2	0.0102	0.5492	0.5901
CKM-SH	0.0102	0.5492	0.5901

4.6 The Overall Rankings of the Algorithms and Evaluation

In order to evaluate the algorithms based on the internal indices, the ranking data in Table 4.7 and Table 4.8 for the data sets generated from SCD and the ranking data in Table 4.9 and Table 4.10 for the data sets generated from PD, are formed. For making a comparison between the algorithms, the overall rankings of the Silhouette Indices and the overall rankings of the C Indices of different data groups in each data pool should be considered. Realising that the decision of selecting the best algorithm and ranking the algorithms due to their performances, multiple criteria should be considered, in order to compare the algorithm performances, *TOPSIS (Technique of Order Preference Similarity to the Ideal Solution)* which is a multi criteria decision making method, is used.

Under this approach, the best algorithm is the one that is the closest alternative to the ideal solution and furthest alternative to the nadir solution (Ishizaka and Nemery, 2013). In the application of TOPSIS, the decision is given due to the *relative closeness coefficient* of the alternatives, that is in between 0 and 1. If the relative closeness coefficient of an alternative approaches to 1, that means the alternative is closer to the ideal solution. For applying TOPSIS, the median rankings of averages of the Silhouette indices' rankings and the median rankings of averages of the C indices' rankings for each algorithms in tables 4.7 and 4.8 respectively for SCD and in tables 4.9 and 4.10 for PD are calculated.

In the first applied TOPSIS for SCD, the alternatives are the algorithms DKM, CKM, CKH-H1, CKM-SH and CKM-H2, and 4 criteria with equal weights (0.25) are the median rankings of averages of the Silhouette indices' rankings of data sets in SCD-2, SCD-3, SCD-4 and SCD-5. The criteria will be named as S1, S2, S3, S4 in tables. The values of criteria for each alternative and the relative closeness coefficients found by applying TOPSIS can be seen in Table 4.21.

In the second applied TOPSIS for SCD, the alternatives are the same and 4 criteria with equal weight (0.25) are the median rankings of averages of the C indices' rankings of data sets in SCD-2, SCD-3, SCD-4 and SCD-5. The criteria will be named as C1, C2, C3, C4 in tables. The related values can be seen in Table 4.22.

In the final TOPSIS applied for SCD, the criteria in firstly and secondly applied TOPSIS are combined with equal weight and the weight is taken as 0.125 for each criteria. The related values can be seen in Table 4.23.

For the data sets in PD-2, PD-3, PD-4 and PD-5, for deciding which algorithm is the best alternative, again TOPSIS is applied three times with the same logic that is explained for the data sets generated from SCD above. The related tables for the first, second and final applied TOPSIS can be seen in tables 4.24, 4.25 and 4.26 respectively.

Table 4.21: Performances of the alternatives for criteria S1, S2, S3, S4 and Relative Closeness Scores of the algorithms for the data sets generated from SCD

Alternatives	Criteria				The Relative Closeness Coefficient
	S1	S2	S3	S4	
DKM	5	5	5	5	0.00
CKM	2	4	2	2	0.61
CKM-H1	4	3	3	3	0.44
CKM-H2	3	2	3	4	0.50
CKM-SH	1	1	1	1	1.00

Table 4.22: Performances of the alternatives for criteria C1, C2, C3, C4 and Relative Closeness Scores of the algorithms for the data sets generated from SCD

Alternatives	Criteria				The Relative Closeness Coefficient
	C1	C2	C3	C4	
DKM	5	5	5	5	0.00
CKM	2	2	2	2	0.75
CKM-H1	4	4	4	4	0.25
CKM-H2	3	3	3	3	0.50
CKM-SH	1	1	1	1	1.00

Table 4.23: Performances of the alternatives for criteria S1, S2, S3, S4, C1, C2, C3, C4 and Relative Closeness Scores of the algorithms for the data sets generated from SCD

Alternatives	Criteria								The Relative Closeness Coefficient
	S1	S2	S3	S4	C1	C2	C3	C4	
DKM	5	5	5	5	5	5	5	5	0.00
CKM	2	4	2	2	2	2	2	2	0.67
CKM-H1	4	3	3	3	4	4	4	4	0.36
CKM-H2	3	2	3	4	3	3	3	3	0.50
CKM-SH	1	1	1	1	1	1	1	1	1.00

Table 4.24: Performances of the alternatives for criteria S1, S2, S3, S4 and Relative Closeness Scores of the algorithms for the data sets generated from PD

Alternatives	Criteria				The Relative Closeness Coefficient
	S1	S2	S3	S4	
DKM	4	5	5	5	0.11
CKM	2	2	2	2	0.75
CKM-H1	5	4	3	3	0.35
CKM-H2	3	3	3	3	0.50
CKM-SH	1	1	1	1	1.00

Table 4.25: Performances of the alternatives for criteria C1, C2, C3, C4 and Relative Closeness Scores of the algorithms for the data sets generated from PD

Alternatives	Criteria				The Relative Closeness Coefficient
	C1	C2	C3	C4	
DKM	4	5	5	5	0.11
CKM	2	2	2	2	0.75
CKM-H1	5	4	3	3	0.35
CKM-H2	3	3	3	4	0.44
CKM-SH	1	1	1	1	1.00

Table 4.26: Performances of the alternatives for criteria S1, S2, S3, S4, C1, C2, C3, C4 and Relative Closeness Scores of the algorithms for the data sets generated from PD

Alternatives	Criteria								The Relative Closeness Coefficient
	S1	S2	S3	S4	C1	C2	C3	C4	
DKM	4	5	5	5	4	5	5	5	0.11
CKM	2	2	2	2	2	2	2	2	0.75
CKM-H1	5	4	3	3	5	4	3	3	0.35
CKM-H2	3	3	3	3	3	3	3	4	0.47
CKM-SH	1	1	1	1	1	1	1	1	1.00

When the performances of the algorithms, due to internal indices for the data sets generated from SCD, are reviewed from tables 4.21, 4.22 and 4.23, for all of the TOPSIS methods applied, the best alternative among the algorithms is CKM-SH since it has the highest relative closeness coefficient. The performances of CKM, CKM-H2, CKM-H1 and DKM follows it respectively.

Similarly, when the TOPSIS applied due to the internal indices for the data sets generated from PD, the performances of the algorithms are in the same order. The alternative that has the highest relative closeness coefficient is CKM-SH and CKM, CKM-H2, CKM-H1 and DKM follows it respectively. It is an expected result that CKM-SH is the best alternative among the other CKM based algorithms, since it is a search-based algorithm. Also it can be observed that CKM is more successful than the proposed algorithms that includes Haar wavelet decomposition and if the ones that includes Haar wavelet decomposition are compared, it is apparent that CKM with Haar Wavelet Decomposition without Projection (CKM-H2) performs better than CKM with Haar Wavelet Decomposition (CKM-H1).

TOPSIS is also applied to the unbalanced data sets generated from SCD and PD. For SCD, the criteria are the median rankings of averages of the Silhouette indices' rankings of data sets in SCD-3-U and SCD-4-U, that are named as S1-U, S2-U and the median rankings of averages of the C indices' rankings of data sets in SCD-3-U and SCD-4-U, that are named as C1-U, C2-U. For all of the applied TOPSIS, the criteria have equal weights. The related results can be seen in tables 4.27, 4.28 and 4.29. The same approach is also followed for PD-3-U and PD-4-U. The results can be seen in tables 4.30, 4.31 and 4.32.

When the performances of the algorithms, due to internal indices for the unbalanced data sets, it can be realised that CKM-SH is again the best algorithm while DKM gives the smallest relative closeness coefficient which indicates it is the least preferred algorithm. While for the balanced data sets, CKM is the second best algorithm among the alternatives, for the unbalanced data sets, it is realised that CKM-H1 or CKM-H2 can give better results than CKM in tables 4.27, 4.30, 4.31 and 4.32 .

According to the Silhouette Index table (Table 4.15) and C Index table (Table 4.16) for PSA data set, CKM based algorithms performs better than DKM since in most of the given k values, CKM based algorithms are giving higher Silhouette Indices and lower C Indices. For k=4, that is the true number of clusters, CKM has the highest Silhouette Index and lowest C Index, that makes it the best alternative, and CKM-H1, CKM-H2 and CKM-SH follow it with equal values of indices.

Table 4.27: Performances of the alternatives for criteria S1-U, S2-U and Relative Closeness Scores of the algorithms for the data sets in SCD-3-U and SCD-4-U

Alternatives	Criteria		The Relative Closeness Coefficient
	S1-U	S2-U	
DKM	5	4	0.00
CKM	4	3	0.29
CKM-H1	2	2	0.71
CKM-H2	3	2	0.58
CKM-SH	1	1	1.00

Table 4.28: Performances of the alternatives for criteria C1-U, C2-U and Relative Closeness Scores of the algorithms for the data sets in SCD-3-U and SCD-4-U

Alternatives	Criteria		The Relative Closeness Coefficient
	C1-U	C2-U	
DKM	4	5	0.00
CKM	2	2	0.71
CKM-H1	4	4	0.18
CKM-H2	2	3	0.57
CKM-SH	1	1	1.00

Table 4.29: Performances of the alternatives for criteria S1-U, S2-U, C1-U, C2-U and Relative Closeness Scores of the algorithms for the data sets in SCD-3-U and SCD-4-U

Alternatives	Criteria				The Relative Closeness Coefficient
	S1-U	S2-U	C1-U	C2-U	
DKM	5	4	4	5	0.00
CKM	4	3	2	2	0.49
CKM-H1	2	2	4	4	0.45
CKM-H2	3	2	2	3	0.57
CKM-SH	1	1	1	1	1.00

Table 4.30: Performances of the alternatives for criteria S1-U, S2-U and Relative Closeness Scores of the algorithms for the data sets in PD-3-U and PD-4-U

Alternatives	Criteria		The Relative Closeness Coefficient
	S1-U	S2-U	
DKM	4	5	0.00
CKM	3	4	0.28
CKM-H1	3	2	0.57
CKM-H2	3	2	0.57
CKM-SH	1	1	1.00

Table 4.31: Performances of the alternatives for criteria C1-U, C2-U and Relative Closeness Scores of the algorithms for the data sets in PD-3-U and PD-4-U

Alternatives	Criteria		The Relative Closeness Coefficient
	C1-U	C2-U	
DKM	3	5	0.00
CKM	3	3	0.39
CKM-H1	2	3	0.50
CKM-H2	3	2	0.53
CKM-SH	1	1	1.00

Table 4.32: Performances of the alternatives for criteria S1-U, S2-U, C1-U, C2-U and Relative Closeness Scores of the algorithms for the data sets in PD-3-U and PD-4-U

Alternatives	Criteria				The Relative Closeness Coefficient
	S1-U	S2-U	C1-U	C2-U	
DKM	4	5	3	5	0.00
CKM	3	4	3	3	0.34
CKM-H1	3	2	2	3	0.54
CKM-H2	3	2	3	2	0.55
CKM-SH	1	1	1	1	1.00

When the algorithms DKM, CKM, CKM-H1, CKM-SH and CKM-H1 are evaluated based on external indices, Table 4.17 and Table 4.18 can be checked for the average external indices for the data sets generated from SCD and PD respectively. In Table 4.17, for the average Adjusted Rand Index, Rand Index and Purity, values of each indices of CKM based algorithms (CKM, CKM-H1, CKM-SH, CKM-H2) are very close to each other and their value is higher than DKM, which indicates, based on the external indices, CKM based algorithms are better alternatives, when they are compared with DKM.

The same comment and performance evaluation can be also done due to the values of external indices in Table 4.18. Again, when the performances of the algorithms are checked due to the external indices for the data sets generated from PD, the CKM based algorithms performs better than DKM.

When the resultant clusters formed by the algorithms applied on data sets are checked, it can be also realised that the CKM based algorithms are better in terms of compactness of similar data in the same cluster and separation of dissimilar data in different clusters. The visualisation of the resultant clusters after the algorithms applied on example data sets can be seen in figures 4.7 and 4.8. In the figures, each color represents a resultant cluster.

For the unbalanced data sets the results of the external indices can be seen in Table 4.19. In the table, it can be realised that the CKM based algorithms have very close results and they have higher values than DKM which makes us conclude CKM based algorithms performed better than DKM based on external indices for the unbalanced data sets.

For the real PSA data set, by knowing the true labels of the clusters, the external indices are also calculated. Due to the results in Table 4.20, based on all external indices, CKM performs better than the other algorithms and CKM-H1, CKM-H2 and CKM-SH follow it with equal values.

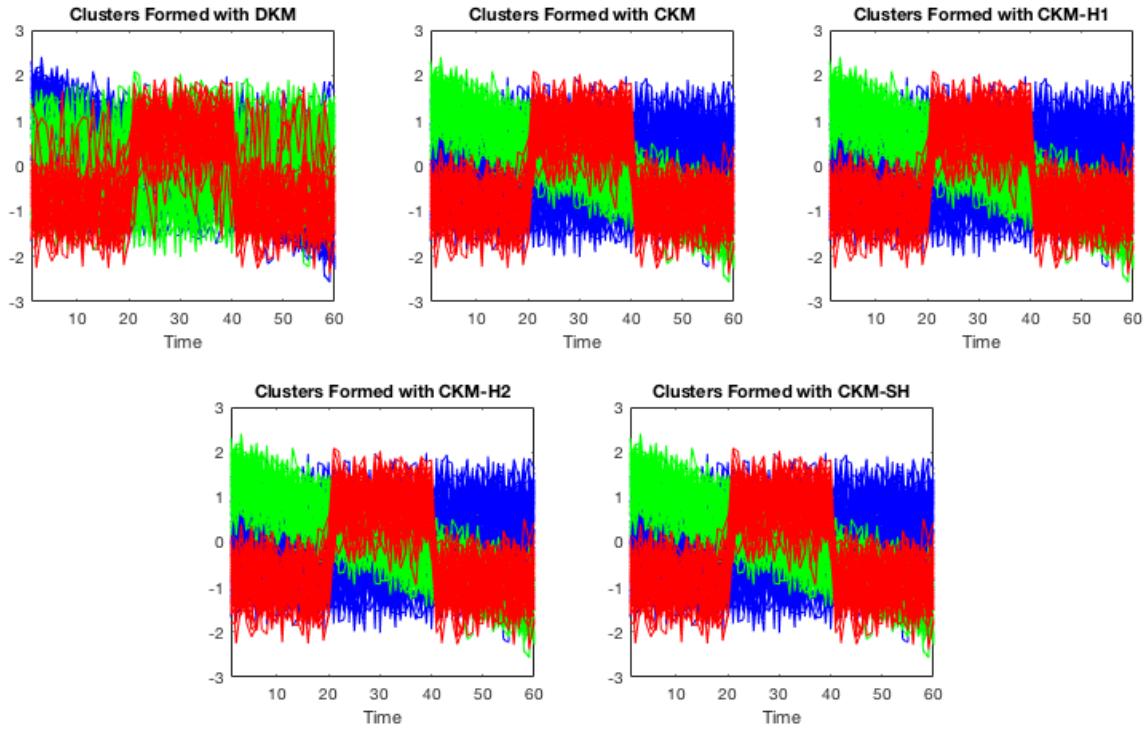

Figure 4.7: Resultant clusters of an example data set from SCD-3

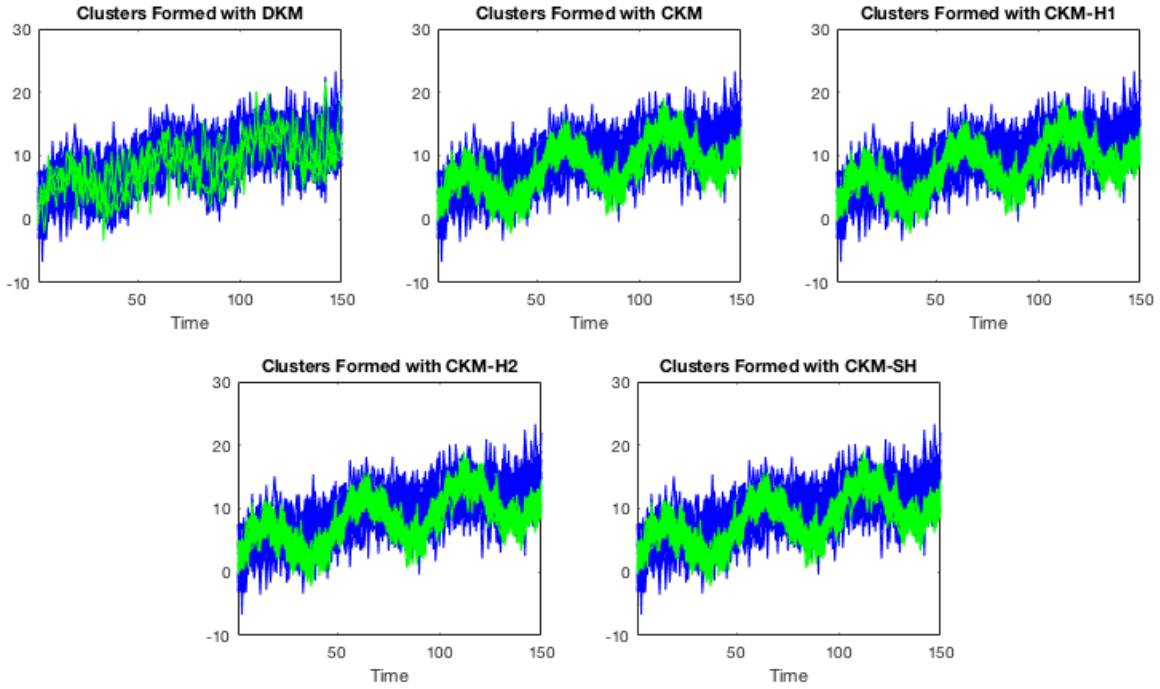

Figure 4.8: Resultant clusters of an example data set from PD-2

Average computing times of the algorithms DKM, CKM, CKM-H1, CKM-H2 and CKM-SH for data sets in SCD-2, SCD-3, SCD-4, SCD-5 and PD-2, PD-3, PD-4, PD-5 can be seen in tables 4.33 and 4.34, that summarizes how computational times differ due to the applied algorithms. Due to the tables, execution time of DKM is shorter than the CKM based algorithms since in the approach that DKM follows, centers of clusters are selected in a global view, as an existing time series in the cluster. CKM based approaches are selecting the centers for each timestamp, with a local approach, that makes the execution time longer.

When CKM is compared with the algorithms that are using Haar wavelet decomposition, CKM-H1, CKM-H2 and CKM-SH, it can be realised that the application of CKM in each Haar level increases the execution time. The algorithm that has the highest execution time is CKM-SH, since it is following a search based approach and applying the clustering for 11 threshold levels to find the best result.

It can be also realised that as the number of time series in the data set increases, the execution time also increases too since the average execution times of the algorithms from SCD-2 to SCD-5 in Table 4.33 and from PD-2 to PD-5 in Table 4.34 are increasing.

Table 4.33: Average computational times (in seconds) of the algorithms for the data sets in SCD-2, SCD-3, SCD-4, SCD-5

Number of Clusters	Average computational time of the algorithms in seconds for data sets in SCD-2					Average computational time of the algorithms in seconds for data sets in SCD-3				
	Algorithms					Algorithms				
	DKM	CKM	CKM-H1	CKM-H2	CKM-SH	DKM	CKM	CKM-H1	CKM-H2	CKM-SH
k=2	0.021	0.132	0.185	0.136	1.390	0.045	0.367	0.916	0.380	7.038
k=3	0.018	0.124	0.223	0.231	2.352	0.033	0.298	0.843	0.535	7.982
k=4	0.015	0.114	0.238	0.327	2.587	0.031	0.379	1.166	0.766	11.820
k=5	0.013	0.118	0.265	0.349	3.195	0.028	0.468	1.208	0.947	12.644
k=6	0.012	0.127	0.282	0.439	2.949	0.027	0.518	1.268	1.160	11.914
k=7	0.012	0.133	0.242	0.450	3.098	0.025	0.578	1.374	1.328	12.971
k=8	0.013	0.153	0.246	0.500	2.973	0.026	0.617	1.526	1.556	13.098
Average	**0.015**	**0.129**	**0.240**	**0.347**	**2.649**	**0.031**	**0.461**	**1.186**	**0.953**	**11.067**

Number of Clusters	Average computational time of the algorithms in seconds for data sets in SCD-4					Average computational time of the algorithms in seconds for data sets in SCD-5				
	Algorithms					Algorithms				
	DKM	CKM	CKM-H1	CKM-H2	CKM-SH	DKM	CKM	CKM-H1	CKM-H2	CKM-SH
k=2	0.139	0.824	2.187	1.055	12.703	0.186	0.816	1.793	1.608	22.587
k=3	0.087	0.874	2.244	1.322	22.947	0.122	1.059	2.284	2.195	23.329
k=4	0.079	1.486	2.361	1.882	19.610	0.107	1.459	2.911	2.981	39.261
k=5	0.089	1.437	2.832	2.306	21.128	0.102	2.197	3.566	3.760	42.72
k=6	0.069	1.363	4.128	2.760	36.874	0.117	2.355	4.451	4.427	47.644
k=7	0.063	1.491	3.390	3.264	33.644	0.093	2.571	5.074	5.337	53.946
k=8	0.062	1.631	3.632	3.727	38.908	0.090	2.905	5.698	6.163	70.134
Average	**0.084**	**1.301**	**2.968**	**2.331**	**26.545**	**0.117**	**1.909**	**3.682**	**3.782**	**42.803**

Table 4.34: Average computational times (in seconds) of the algorithms for the data sets in PD-2, PD-3, PD-4, PD-5

Number of Clusters	Average computational time of the algorithms in seconds for data sets in PD-2					Average computational time of the algorithms in seconds for data sets in PD-3				
	Algorithms					Algorithms				
	DKM	CKM	CKM-H1	CKM-H2	CKM-SH	DKM	CKM	CKM-H1	CKM-H2	CKM-SH
k=2	0.004	0.056	0.159	0.062	1.788	0.007	0.460	0.553	0.1495	5.032
k=3	0.003	0.025	0.135	0.129	1.992	0.006	0.390	0.510	0.210	5.311
k=4	0.003	0.028	0.122	0.145	2.241	0.006	0.346	0.601	0.304	6.342
k=5	0.003	0.032	0.134	0.182	2.301	0.005	0.236	0.430	0.375	4.317
k=6	0.002	0.038	0.129	0.210	2.353	0.005	0.171	0.310	0.454	4.924
k=7	0.003	0.047	0.150	0.243	2.998	0.005	0.161	0.243	0.531	3.111
k=8	0.003	0.062	0.169	0.284	3.320	0.005	0.152	0.209	0.615	2.418
Average	**0.003**	**0.041**	**0.143**	**0.179**	**2.428**	**0.005**	**0.274**	**0.408**	**0.377**	**4.494**

Number of Clusters	Average computational time of the algorithms in seconds for data sets in PD-4					Average computational time of the algorithms in seconds for data sets in PD-5				
	Algorithms					Algorithms				
	DKM	CKM	CKM-H1	CKM-H2	CKM-SH	DKM	CKM	CKM-H1	CKM-H2	CKM-SH
k=2	0.016	0.462	3.216	3.342	40.163	0.015	0.485	3.975	3.443	37.169
k=3	0.015	0.577	2.668	3.459	37.948	0.012	0.676	5.573	5.380	39.216
k=4	0.013	1.041	2.996	3.738	32.459	0.016	1.301	4.505	4.272	47.418
k=5	0.020	1.211	2.999	3.978	41.890	0.030	1.960	3.197	4.560	42.612
k=6	0.012	1.355	3.053	3.848	39.195	0.018	1.674	3.033	4.013	38.923
k=7	0.012	1.347	2.868	3.021	37.754	0.019	1.838	3.223	3.251	37.273
k=8	0.013	1.817	3.260	3.723	39.435	0.029	2.043	3.307	4.329	54.782
Average	**0.014**	**1.116**	**3.009**	**3.587**	**38.406**	**0.020**	**1.425**	**3.830**	**4.178**	**42.485**

CHAPTER 5

CONCLUSION

Clustering is a widely used unsupervised learning method that aims to group similar data in the same clusters and separate the dissimilar data to the different clusters. Clustering can be applied to various data types. In this thesis, we address the application of clustering algorithms on time series data that can be defined as the chronologic collection of data.

In this thesis, for the solution of time series clustering problem we are proposing two main approaches. In the first approach, the clustering problem is considered as an optimization problem and an optimization model is proposed for the solution of it. In the defined algorithm, the raw time series data is used and the problem is solved by using the proposed optimization problem. In the second approach, three other algorithms are proposed which are also following the optimization problem approach but instead of using raw data, transformed data is used.

In the literature, the studies about time series clustering select the centers of the clusters among the existing time series. In this study, we have developed a different center selection method. In our method, the center points of a cluster are selected for each timestamp and when these center points are combined, it is not forming an existing time series. For comparing the performances of the proposed center based algorithms, an algorithm called Discrete k-Median Algorithm (DKM), that is proposed in the paper of Seref et al. (2014) and follows the selection of centers from the existing time series, is used.

By following the logic of selecting the cluster center points for each timestamp, four algorithms are proposed, that are Center Based k-Median Algorithm (CKM)

in which the raw data is used, CKM with Haar Wavelet Decomposition (CKM-H1), CKM with Haar Wavelet Decomposition without Projection (CKM-H2) and CKM with Search Based Haar Wavelet Decomposition (CKM-SH). In CKM, that is the first proposed algorithm, at first the optimization problem formulation is given, in which the objective function aims to assign the given n samples to the k clusters, by minimizing the distance between time series data to the center points in the clusters which data are member of. CKM is solved by dividing the problem into two subproblems that are CKM-S1 and CKM-S2. In CKM-S1, the cluster centers are assumed to be known and the problem is solved to find the cluster assignments of the data. Following that, CKM-S2 is solved to find the new centers from the resultant cluster assignments of CKM-S1. The algorithm goes back to CKM-S1 and recursively continue to be solved, until no better objective function (no better cluster assignments) can be done. In CKM-H1, at first the Haar wavelet decomposition of each time series is calculated. Haar wavelet decomposition can be defined as smoothing the time series by taking average of two successive values in the sample in a defined resolution level. Following this step, the same two subproblem logic applied in CKM is followed. The difference is the algorithm is started to be solved with the lowest resolution (the most transformed time series) of the data, and after the application of CKM, the new centers are projected and used as the initial center assignments for the next lowest resolution which CKM will be applied again. The algorithm continue to be applied like that until it is solved for the highest resolution level (original data). CKM-H2 is different than CKM-H1 in the center projection step. The last proposed algorithm, CKM-SH is the search based version of CKM-H2.

For evaluating the performances of the algorithms, data sets that are generated from two different data pools namely Synthetic Control Chart Time Series Data Pool (SCD) and Data Pool Including Periodicity (PD) are used. In the data pools, there are 14 and 18 different patterns respectively. By taking combination of 2, 3, 4 and 5 patterns, new data sets are generated, that are named as the data sets in SCD-2, SCD-3, SCD-4, SCD-5 and PD-2, PD-3, PD-4, PD-5. In addition to the formed balanced data sets, unbalanced data sets, which have patterns that include different number of time series are formed too. The unbalanced data sets, which are composed of 3 patterns, are named as data sets in SCD-3-U and PD-3-U and the ones formed with 4 patterns are

named as data sets in SCD-4-U and PD-4-U. Finally, the tests are conducted to a real data set that is called PSA Data Set, which is formed by time series data that includes 4 labels.

In order to evaluate the performances of the algorithms, the compactness inside the cluster and the separation of the data in different clusters needed to be checked. For this purpose, as the performance measures, three external and two internal indices are used. For the calculation of the external indices, true cluster information is needed. As a result, the performance evaluation due to external indices are made for the data sets which k is equal to the combined pattern number used for the formation of the data set. The used external indices are Rand Index, Adjusted Rand Index and Purity. The internal indices, Silhouette Index and C Index, do not need any additional information other than the resultant clusters and evaluate the compactness and separation internally. For each data set, for the resultant clusters of the algorithms, the internal indices are calculated for $k = 2, 3, 4, 5, 6, 7$ and 8.

For evaluating the internal indices, TOPSIS (Technique of Order Preference Similarity to the Ideal Solution), that is a multi criteria decision making method is used. From best to the worst alternative, the performances of the algorithms are sorted as CKM-SH, CKM, CKM-H2, CKM-H1 and DKM for the balanced data sets in SCD-2, SCD-3, SCD-4, SCD-5 and PD-2, PD-3, PD-4, PD-5. When the evaluation of the algorithms is made, based on the calculated internal indices for the unbalanced data sets in SCD-3-U, SCD-4-U and PD-3-U, PD-4-U, it can be concluded that CKM-SH is the best alternative again and CKM-H1 and CKM-H2 give better results than CKM. For the PSA data set, it can be also concluded that CKM-SH is the best alternative while DKM is the least preferred.

Additionally, due to the evaluation of the external indices, the proposed center based algorithms performed well when they are compared with the resultant clusters formed by DKM. Due to these results, it is concluded that the clusters formed by the proposed center based algorithms are in more compact in terms of grouping similar data and well separated in terms of separating dissimilar data when they are compared with clusters formed by the algorithm approach that is selecting the cluster centers from the existing time series.

By seeing the CKM based algorithms performed better than DKM due to the evaluations based on internal and external indices, it can be concluded that CKM based algorithms can be used in real life applications of time series clustering such as finding stock option moves that follows similar patterns, clustering test results of patients which might help to group the results due to disease levels, identification of countries that has similar increases and decreases in their population levels, determining coasts that has similar temperature moves, that can be used in geographical analysis.

While the advantage of the CKM based algorithms is to provide more compact clusters in terms of grouping similar data, the disadvantage is to have higher computing time when the CKM based algorithms are compared with DKM. The reason is since CKM based algorithms are selecting centers for each timestamp, it increases the algorithm execution time. It can be also realised that the CKM-SH has the highest computing time, since it is a search based algorithm and trying to find the best result while running the algorithm for different threshold levels.

In this study, the proposed algorithms are compared with DKM. A future study may be comparing the proposed algorithms with the algorithms proposed in TSclust package in R, such as the algorithm proposed by Maharaj (2000). TSclust package can be also used in the algorithm evaluations, in addition to the used indices in this study. The data sets used in our study are deterministic and do not contain noise. Another future research may be testing the algorithms with data sets that have stochastic natures and with data sets that contain noise. Additional future study might be conducting test of significance for the evaluation of the results.

REFERENCES

S. Aghabozorgi, M. R. Saybani, and T. Y. Wah. Incremental clustering of time-series by fuzzy clustering. *Journal of Information Science and Engineering*, 28(4):671–688, 2012.

S. Aghabozorgi, A. S. Shirkhorshidi, and T. Y. Wah. Time-series clustering–a decade review. *Information Systems*, 53:16–38, 2015.

R. J. Alcock, Y. Manolopoulos, et al. Time-series similarity queries employing a feature-based approach. In *7th Hellenic conference on informatics*, pages 27–29, 1999.

Z. Ansari, M. Azeem, W. Ahmed, and A. V. Babu. Quantitative evaluation of performance and validity indices for clustering the web navigational sessions. *arXiv preprint arXiv:1507.03340*, 1:217–226, 2015a.

Z. Ansari, M. Azeem, W. Ahmed, and A. V. Babu. Quantitative evaluation of performance and validity indices for clustering the web navigational sessions. *World of Computer Science and Information Technology Journal*, 1(5):217–226, 2015b.

A. Bagnall and G. Janacek. Clustering time series with clipped data. *Machine Learning*, 58(2-3):151–178, 2005.

K. P. Bennett and O. L. Mangasarian. Bilinear separation of two sets inn-space. *Computational Optimization and Applications*, 2(3):207–227, 1993.

D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:, 1994.

J. C. Bezdek, R. Ehrlich, and W. Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2-3):191–203, 1984.

P. Bradley, O. Mangasarian, and W. Street. Clustering via concave minimization. *Advances in neural information processing systems*, 9:368–374, 1996.

M. Charikar, S. Guha, É. Tardos, and D. B. Shmoys. A constant-factor approximation algorithm for the k-median problem. *Journal of Computer and System Sciences*, 65 (1):129–149, 2002.

G. Chen, S. A. Jaradat, N. Banerjee, T. S. Tanaka, M. S. Ko, and M. Q. Zhang. Evaluation and comparison of clustering algorithms in analyzing es cell gene expression data. *Statistica Sinica*, 12(1):241–262, 2002.

I. Daubechies and B. J. Bates. Ten lectures on wavelets, 1993.

D. Dua and C. Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

P. Esling and C. Agon. Time-series data mining. *ACM Computing Surveys (CSUR)*, 45(1):1–34, 2012.

T.-c. Fu. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1):164–181, 2011.

X. Golay, S. Kollias, G. Stoll, D. Meier, A. Valavanis, and P. Boesiger. A new correlation-based fuzzy logic clustering algorithm for fmri. *Magnetic resonance in medicine*, 40(2):249–260, 1998.

D. Guijo-Rubio, A. M. Durán-Rosal, P. A. Gutiérrez, A. Troncoso, and C. Hervás-Martínez. Time-series clustering based on the characterization of segment typologies. *IEEE Transactions on Cybernetics*, PP:1–14, 2020.

M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Journal of intelligent information systems*, 17(2-3):107–145, 2001.

G. Hamerly and C. Elkan. Alternatives to the k-means algorithm that find better clusterings. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 600–607, 2002.

J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

V. Hautamaki, P. Nykanen, and P. Franti. Time-series clustering by approximate prototypes. In *2008 19th International Conference on Pattern Recognition*, pages 1–4. IEEE, 2008.

L. J. Hubert and J. R. Levin. A general statistical framework for assessing categorical clustering in free recall. *Psychological bulletin*, 83(6):1072, 1976.

A. Ishizaka and P. Nemery. *Multi-criteria decision analysis: methods and software*, pages 215–216. John Wiley & Sons, 2013.

A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

L. Kaufman and P. J. Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

E. Keogh and J. Lin. Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and information systems*, 8(2): 154–177, 2005.

K. Košmelj and V. Batagelj. Cross-sectional approach for clustering time varying data. *Journal of Classification*, 7(1):99–109, 1990.

T. W. Liao. Clustering of time series data—a survey. *Pattern recognition*, 38(11): 1857–1874, 2005a.

T. W. Liao. Clustering of time series data—a survey. *Pattern recognition*, 38(11):1864, 2005b.

J. Lin, M. Vlachos, E. Keogh, and D. Gunopulos. Iterative incremental clustering of time series. In *International Conference on Extending Database Technology*, pages 106–122. Springer Berlin Heidelberg, 2004.

E. A. Maharaj. Cluster of time series. *Journal of Classification*, 17(2):297–314, 2000.

T. Mitsa. *Temporal data mining*. CRC Press, 2010.

E. F. Petricoin III, D. K. Ornstein, C. P. Paweletz, A. Ardekani, P. S. Hackett, B. A. Hitt, A. Velassco, C. Trucco, L. Wiegand, K. Wood, et al. Serum proteomic patterns for detection of prostate cancer. *Journal of the National Cancer Institute*, 94(20): 1576–1578, 2002.

S. Policker and A. B. Geva. Nonstationary time series analysis by temporal clustering. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 30(2): 339–343, 2000.

P. Rai and S. Singh. A survey of clustering techniques. *International Journal of Computer Applications*, 7(12):1–5, 2010.

C. Ratanamahatana, E. Keogh, A. J. Bagnall, and S. Lonardi. A novel bit level time series representation with implication of similarity search and clustering. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 771–777. Springer, 2005.

C. A. Ratanamahatana and E. Keogh. Everything you know about dynamic time warping is wrong. In *Third workshop on mining temporal and sequential data*, volume 32. Citeseer, 2004.

E. Rendón, I. M. Abundez, C. Gutierrez, S. D. Zagal, A. Arizmendi, E. M. Quiroz, and H. E. Arzate. A comparison of internal and external cluster validation indexes. In *Proceedings of the 2011 American Conference, San Francisco, CA, USA*, volume 29, pages 1–10, 2011.

C. S. ReVelle and R. W. Swain. Central facilities location. *Geographical analysis*, 2 (1):30–42, 1970.

P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

M. Roux. Which indices reveals the right number of clusters. *Data Analysis Bulletin*, 17:7–30, 2006.

J. Sander. *Density-Based Clustering*, pages 270–273. Springer US, Boston, MA, 2010.

A. Sardá-Espinosa. Comparing time-series clustering algorithms in r using the dtwclust package. *R package vignette*, 12:41, 2017.

E. Schikuta. Grid-clustering: A fast hierarchical clustering method for very large data sets. page 6, 01 1993.

O. Seref, Y.-J. Fan, and W. A. Chaovalitwongse. Mathematical programming formulations and algorithms for discrete k-median clustering of time-series data. *INFORMS Journal on Computing*, 26(1):160–172, 2014.

K. Wang, B. Wang, and L. Peng. Cvap: validation for cluster analyses. *Data Science Journal*, 8:88–93, 2009.

X. Wang, K. Smith-Miles, and R. Hyndman. Characteristic-based clustering for time series data. *Data Min. Knowl. Discov.*, 13:335–364, 2006.

X. Xi, E. Keogh, C. Shelton, L. Wei, and C. A. Ratanamahatana. Fast time series classification using numerosity reduction. In *Proceedings of the 23rd international conference on Machine learning*, pages 1033–1040, 2006.

S. Zolhavarieh, S. Aghabozorgi, and Y. W. Teh. A review of subsequence time series clustering. *The Scientific World Journal*, 2014, 2014.

APPENDIX A

APPENDIX

In this appendix, the average of rankings of Silhouette and C indices for the data sets in SCD-2, SCD-3, SCD-4, SCD-5, SCD-3-U, SCD-4-U and the average of rankings of Silhouette and C indices for the data sets in PD-2, PD-3, PD-4, PD-5, PD-3-U, PD-4-U are provided.

Table A.1: The averages of rankings of Silhouette Indices for data sets in SCD-2, SCD-3, SCD-4, SCD-5

Averages of rankings of Silhouette Indices for data sets in SCD-2							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2.4066	3.5275	4.011	4.5055	4.5604	4.5934	4.4396
CKM	1.6044	2.1868	2.4945	2.5714	2.4945	2.5495	2.7912
CKM-H1	1.5275	2.5495	2.8022	3.2857	3.2967	3.5165	3.6154
CKM-H2	1.4725	2.3077	2.5714	2.6813	2.5934	2.6154	2.4505
CKM-SH	1.1868	1.3297	1.3956	1.3736	1.3077	1.2857	1.1978

Averages of rankings of Silhouette Indices for data sets in SCD-3							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2.6896	3.4863	4.1511	4.2747	4.3214	4.4615	4.522
CKM	1.4973	2.0659	2.5357	2.7335	2.8352	2.9753	3.0412
CKM-H1	1.6456	2.0852	2.4863	2.5495	2.7418	2.8407	2.794
CKM-H2	1.6374	2.0357	2.5824	2.7005	2.6484	2.6181	2.7198
CKM-SH	1.2527	1.3379	1.4863	1.478	1.4396	1.3901	1.4038

Averages of rankings of Silhouette Indices for data sets in SCD-4							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2.9311	3.5135	4.2058	4.4016	4.4306	4.4795	4.4835
CKM	1.7732	1.9491	2.3467	2.6214	2.6593	2.7493	2.9001
CKM-H1	1.7263	2.1768	2.5564	2.6673	2.7073	2.7952	2.8492
CKM-H2	1.7473	2.2857	2.5315	2.7163	2.7992	2.7942	2.7812
CKM-SH	1.3377	1.5015	1.5295	1.5994	1.6144	1.5644	1.5165

Averages of rankings of Silhouette Indices for data sets in SCD-5							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3.1598	3.7403	4.2223	4.5045	4.5654	4.5684	4.5854
CKM	1.9211	2.1324	2.2827	2.4965	2.6289	2.7398	2.8591
CKM-H1	1.7717	2.1698	2.5654	2.7393	2.7038	2.6748	2.7797
CKM-H2	1.9261	2.3247	2.6314	2.7398	2.7507	2.7862	2.7133
CKM-SH	1.4066	1.503	1.6054	1.6119	1.5834	1.5754	1.494

Table A.2: The averages of rankings of C Indices for data sets in SCD-2, SCD-3, SCD-4, SCD-5

Averages of rankings of C Indices for data sets in SCD-2							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2.3516	3.3736	4.1209	4.5495	4.5604	4.6484	4.6264
CKM	1.6154	2.3077	2.3736	2.5495	2.5055	2.4835	2.5824
CKM-H1	1.5934	2.6154	2.8352	3.044	3.4176	3.3077	3.4396
CKM-H2	1.4066	2.4396	2.5275	2.6593	2.7802	2.9231	2.6703
CKM-SH	1.1758	1.4286	1.3736	1.3956	1.2637	1.3407	1.2967

Averages of rankings of C Indices for data sets in SCD-3							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2.3352	3.3407	4.1154	4.3434	4.4231	4.5385	4.6346
CKM	1.5797	2.1236	2.5522	2.5934	2.7308	2.6648	2.7308
CKM-H1	1.9505	2.3132	2.6154	2.7637	2.8297	2.8736	2.9203
CKM-H2	1.7170	1.9918	2.6264	2.6621	2.6593	2.7582	2.7857
CKM-SH	1.3654	1.3214	1.5055	1.4258	1.4066	1.4451	1.4313

Averages of rankings of C Indices for data sets in SCD-4							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2.5445	3.2697	4.1369	4.5065	4.5544	4.6034	4.6304
CKM	1.7183	2.0100	2.3477	2.5025	2.4655	2.5385	2.6404
CKM-H1	2.2028	2.5305	2.7842	2.7962	2.8591	2.9720	2.9780
CKM-H2	1.9431	2.2857	2.4356	2.7073	2.7772	2.8202	2.8022
CKM-SH	1.4755	1.5105	1.4905	1.5994	1.5724	1.5574	1.5045

Averages of rankings of C Indices for data sets in SCD-5							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2.9101	3.4256	4.1149	4.5240	4.6354	4.6638	4.7108
CKM	1.7742	2.0544	2.3347	2.4321	2.5340	2.5265	2.6019
CKM-H1	2.2323	2.6813	2.8127	2.9156	2.8137	2.8596	2.9216
CKM-H2	2.0579	2.3826	2.5534	2.6434	2.7522	2.7502	2.7303
CKM-SH	1.4920	1.5300	1.5544	1.5654	1.5989	1.5594	1.5165

Table A.3: The averages of rankings of Silhouette Indices for data sets in PD-2, PD-3, PD-4, PD-5

Averages of rankings of Silhouette Indices for data sets in PD-2							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.1458	3.4722	3.4861	3.5208	3.5000	3.5486	3.5625
CKM	1.0000	1.7569	2.1458	1.8819	2.0694	2.0278	1.9236
CKM-H1	1.0000	3.8958	4.1458	4.3403	4.3542	4.3819	4.4653
CKM-H2	1.0000	3.0764	2.7222	2.7917	2.7431	2.8681	2.7361
CKM-SH	1.0000	1.5833	1.4792	1.5972	1.4444	1.4514	1.5208

Averages of rankings of Silhouette Indices for data sets in PD-3							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.3125	1.3646	3.6920	3.9792	3.9940	3.8973	3.7932
CKM	1.0417	1.0074	1.9955	2.5417	2.4568	2.3795	2.3601
CKM-H1	1.0238	1.0089	3.6161	3.4048	3.4509	3.5551	3.5491
CKM-H2	1.0223	1.0074	2.9762	2.7812	2.6830	2.6339	2.6399
CKM-SH	1.0119	1.0074	1.6473	1.4821	1.4643	1.4583	1.5060

Averages of rankings of Silhouette Indices for data sets in PD-4							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.5055	1.7386	1.8611	4.0461	3.9048	3.9102	3.9018
CKM	1.1974	1.2257	1.2098	2.3547	2.3894	2.4633	2.6215
CKM-H1	1.0923	1.1404	1.1538	3.1974	3.1875	3.1220	3.0228
CKM-H2	1.1062	1.1319	1.1434	2.7946	3.0357	3.1587	3.1151
CKM-SH	1.0704	1.0987	1.0942	1.5923	1.7386	1.7019	1.6860

Averages of rankings of Silhouette Indices for data sets in PD-5							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.5320	1.8958	2.3966	3.3681	4.1014	4.1285	4.0694
CKM	1.1825	1.4504	1.7381	2.2421	2.7230	2.7475	2.7540
CKM-H1	1.0970	1.2436	1.6758	2.4695	2.9070	2.8931	2.9296
CKM-H2	1.1062	1.2262	1.6372	2.3666	2.8514	2.9137	2.9568
CKM-SH	1.0786	1.1582	1.3247	1.4601	1.4990	1.5164	1.5402

Table A.4: The averages of rankings of C Indices for data sets in PD-2, PD-3, PD-4, PD-5

Averages of rankings of C Indices for data sets in PD-2							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.3194	4.0069	3.9306	3.7222	3.5764	3.3403	2.8542
CKM	1.1736	1.8125	2.0556	1.9028	1.7778	1.8681	1.9792
CKM-H1	1.1111	3.2847	3.7431	4.1875	4.2708	4.3333	4.3056
CKM-H2	1.1111	3.2708	2.7639	2.9583	2.9722	2.9236	3.0486
CKM-SH	1.1111	1.5278	1.4653	1.5208	1.6250	1.6458	1.8611

Averages of rankings of C Indices for data sets in PD-3							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.4018	1.4271	4.0744	4.2113	4.0908	3.9330	3.8110
CKM	1.1384	1.1057	2.0298	2.0640	2.0208	2.0387	2.1324
CKM-H1	1.1429	1.0699	3.1369	3.4509	3.7039	3.7039	3.6801
CKM-H2	1.1354	1.0685	3.1637	3.1086	2.9881	2.9152	2.8318
CKM-SH	1.1265	1.0461	1.5818	1.5640	1.5506	1.5714	1.6012

Averages of rankings of C Indices for data sets in PD-4							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.5427	1.7078	1.9127	4.2564	4.3085	4.3328	4.2872
CKM	1.2158	1.2664	1.2217	2.2827	2.1935	2.1865	2.2148
CKM-H1	1.1463	1.2078	1.1860	3.0670	3.0258	2.9906	2.9554
CKM-H2	1.1483	1.1870	1.1949	2.9727	3.2019	3.3075	3.3353
CKM-SH	1.1171	1.1513	1.1285	1.5387	1.6463	1.6558	1.6776

Averages of rankings of C Indices for data sets in PD-5							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.5312	1.8827	2.5089	3.4330	4.1496	4.2450	4.2946
CKM	1.1944	1.4638	1.7245	2.1049	2.4668	2.4082	2.3748
CKM-H1	1.1471	1.2631	1.7021	2.4501	2.9000	2.9459	2.9469
CKM-H2	1.1429	1.2326	1.7416	2.5826	3.1235	3.1307	3.1967
CKM-SH	1.1210	1.1647	1.3539	1.5040	1.5657	1.5756	1.5848

Table A.5: The averages of rankings of Silhouette Indices for data sets in SCD-3-U, SCD-4-U

Averages of rankings of Silhouette Indices for data sets in SCD-3-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	4.6	5	4.6	4.8	5	4.6	5
CKM	3.6	2.6	3.6	3.4	4	2.8	2
CKM-H1	1.2	3	2.8	2.6	3	2.2	3
CKM-H2	2.6	2.8	3	2.2	2	2.8	3
CKM-SH	1.8	1.6	1	1.2	1	2	2

Averages of rankings of Silhouette Indices for data sets in SCD-4-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3.6	4.8	4.2	4.6	4.4	4.8	4.8
CKM	3	2.8	2.4	3	3.6	2.6	3.4
CKM-H1	2	2.2	2.4	2.2	1.8	2.8	2.2
CKM-H2	2.2	2.8	2.2	2.2	2.2	2.6	2.4
CKM-SH	2	1.8	1.8	1.6	1.6	1.8	1.4

Table A.6: The averages of rankings of Silhouette Indices for data sets in PD-3-U, PD-4-U

Averages of rankings of Silhouette Indices for data sets in PD-3-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.4	1.2	3.8	3.6	3.2	3	4.4
CKM	1.2	1	3	3.4	4	3.8	2
CKM-H1	1.6	1.4	3.4	3.2	2.8	2.6	2.8
CKM-H2	1.2	1.8	2.8	2.2	3.2	2.8	4
CKM-SH	1	1.6	1.2	1.8	1	1.6	1.8

Averages of rankings of Silhouette Indices for data sets in PD-4-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	4.8	4.2	4.2	4.6	4.6	4.2	4.2
CKM	3.2	3.6	2.8	3.6	3.6	3	3
CKM-H1	3	1.4	2.2	2	2.2	2	2.2
CKM-H2	2.2	2.2	2.8	2	1.8	2	1.8
CKM-SH	1.4	1.8	1.8	1.2	1.2	1.6	1.4

Table A.7: The averages of rankings of C Indices for data sets in SCD-3-U, SCD-4-U

Averages of rankings of C Indices for data sets in SCD-3-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2.6	5	3.2	4.6	4.8	4.4	4.2
CKM	2	2.2	2	2.6	2.8	2.4	2.4
CKM-H1	3.4	4	4.4	4.4	3.8	4.6	4.2
CKM-H2	3	2.4	3.2	2.4	1.8	2.4	2.2
CKM-SH	2.4	1.4	1.8	1	1	1.2	1.4

Averages of rankings of C Indices rankings for data sets in SCD-4-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	2.6	4.2	3.4	4.2	4.4	4.2	4.2
CKM	1.8	2	2	2.6	2	2.6	2
CKM-H1	3.8	2.6	3.8	3.2	3.8	4	3.4
CKM-H2	2.6	2.8	3	1.6	2.4	2.6	2.4
CKM-SH	2.4	2.2	1.6	1	1.6	1.6	1.8

Table A.8: The averages of rankings of C Indices for data sets in PD-3-U, PD-4-U

Averages of rankings of C Indices for data sets in PD-3-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	1.4	1.6	3.6	3.6	3.8	3	3.4
CKM	1.2	1	3.4	3.2	3.8	3.6	2
CKM-H1	1.6	2	2.2	2.6	2.4	2.8	3.4
CKM-H2	1.2	1.6	3.2	3.6	3.2	3.4	4.2
CKM-SH	1	1.4	1.2	1.2	1	1.6	1.6

Averages of rankings of C Indices rankings for data sets in PD-4-U							
Algorithms	Number of Clusters						
	k=2	k=3	k=4	k=5	k=6	k=7	k=8
DKM	3	2.6	4.2	4.2	4.6	4.2	4.4
CKM	1.4	2.6	2.8	3.4	3.8	3.4	3.4
CKM-H1	3.4	3.6	4.6	3.6	3.4	2.8	2.2
CKM-H2	2.2	1.2	2.2	2	2.2	2.4	2
CKM-SH	2.2	1.2	1.2	1.2	1	1.4	1.4