Clustering Classification Final Report

Wei Zhang

Min Feng

Atilla Ay

May 1 2018

Contents

Abstract

Time series clustering has been shown a wide application in the real world dataset. For example, in the stock market, people could cluster thousands of stocks into different groups based on time series data on different characters [1]. In this article, we will explain some basic ideas of time series clustering such as distance measures and clustering algorithm. (Seminar Paper) Moreover, we will apply the methods we discussed to interpret the "DJIA 30 Stock Time Series" dataset.

Keywords: Time Series; Clustering; Distance measure; Partioned Algorithm; Hierarchical Clustering

1 Data and Motivation

1.1 DJIA 30 Stock Time Series" Dataset

Stock market dataset is always good for time series analysis since the value of stocks differ from time to time. In this project, our group is going to measure the distance between each company in the dataset, and apply different clustering algorithm to divide the dataset into different groups based on their distances. To analysis this dataset, we used several R packages: `dplyr`, `readr`, `xts`, `TSclust`, `dtwclust`, and `TSdist`.

```
library(dplyr)
library(readr)
library(ggplot2)
library(gridExtra)
library(xts)
library(dtw)
library(TSclust)
library(dtwclust)
library(TSdist)
```

1.1.1 Data Explanation

The dataset we use to apply the methods of distance measures and different clustering algorithm is the "DJIA 30 Stock Time Series" dataset from Kaggle (https://www.kaggle.com/szrlee/stock-time-series-20050101-to-20171231). This data contains past year's stock data from 31 companies that posted on Dow Jones Industrial Average. Here are the explanations of each variable:

- Date - in format: yy-mm-dd
- Open - price of the stock at market open (this is NYSE data so all in USD)
- High - Highest price reached in the day
- Low Close - Lowest price reached in the day
- Volume - Number of shares traded
- Name - the stock's ticker name

We added two new variables *diff_OC* and *diff_HL* which are measure the difference between open price and close price and the difference between highest price and lowest price reached in a day. In order to analyze data more convinient, we then combined *Volume*, *diff_OC*, *diff_HL*, *Name* and *Date* to create a new dataset. Finally, we separated this new dataset by company.

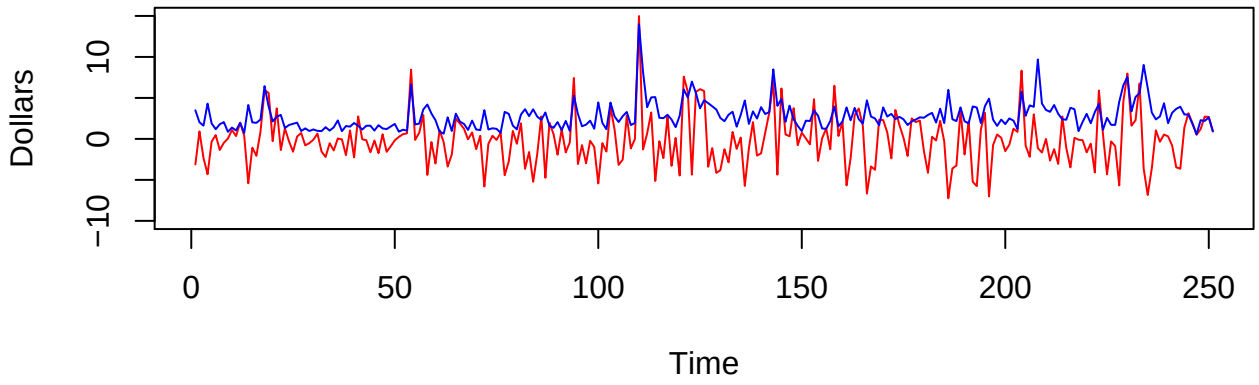
```
## We create a new datase that only includes difference of open-close price
## difference of high-low price and stock volume of each company.

data$diff_OC<-data$Open-data$Close
data$diff_HL<-data$High-data$Low

data.new<-data[,c(1,7,6,8,9)]

## Data cleaning and normalized, in this part we set all missing data
```

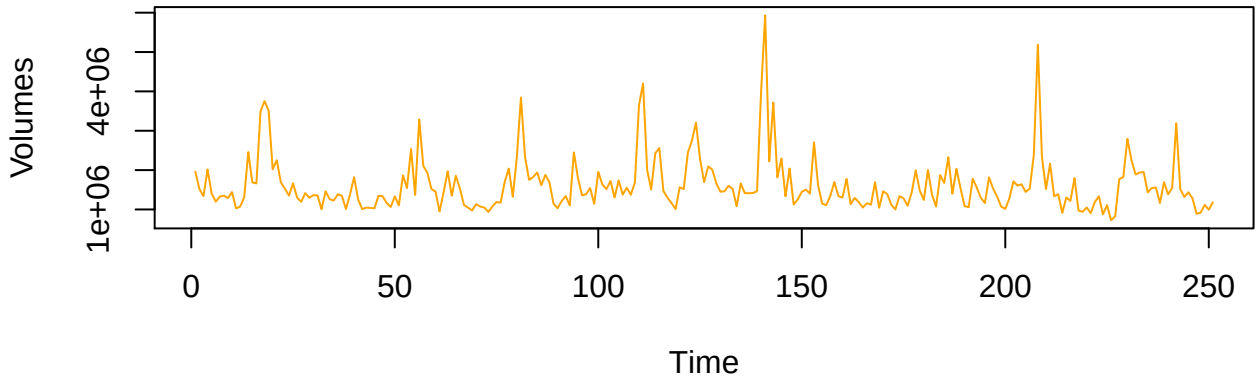
```
## equal to 0.

data.new[is.na(data.new)]<-0
data.new$Date <- as.Date(data.new$Date, format = "%Y-%m-%d")

data.new[,c(4,5)]<-scale(data.new[,c(4,5)])

## In this part, we seperate 31 companies into a stock_list.

stock_list <- vector(mode="list", length=length(unique(data.new$Name)))
i = 1
for (c in unique(data.new$Name)){
    stock_list[[i]] <- filter(data.new, Name == c)
    i <- i+1
}
```

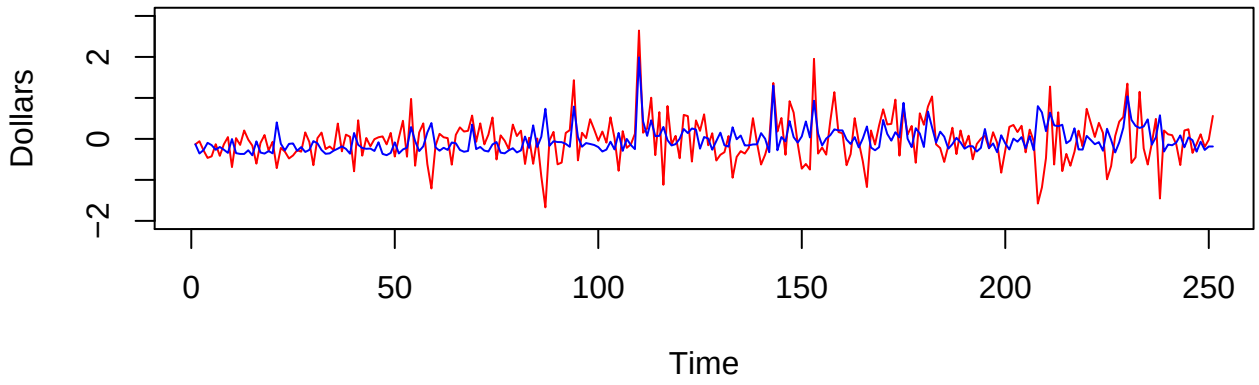
1.1.2 Time Series Plots of Some Companies

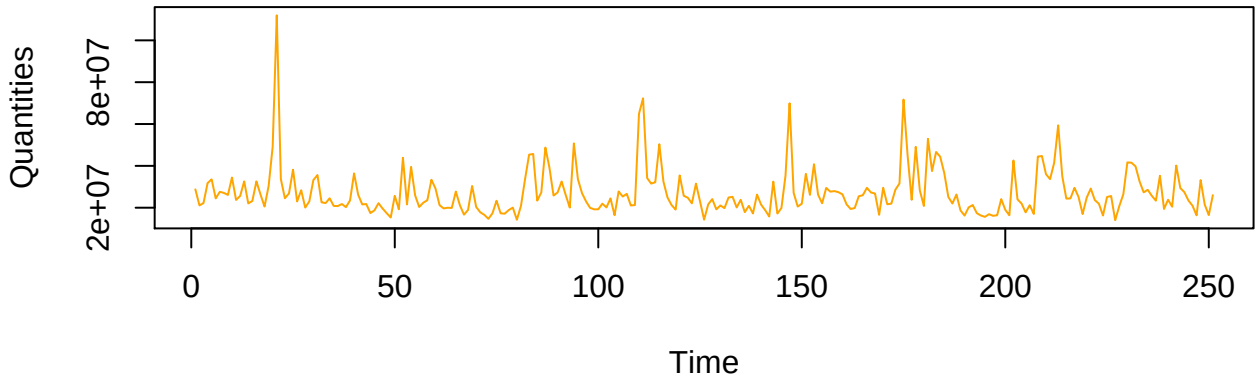
In order to get a detail understanding of data, we plot time series plots of three large companies, Google, Apple and Amazon.

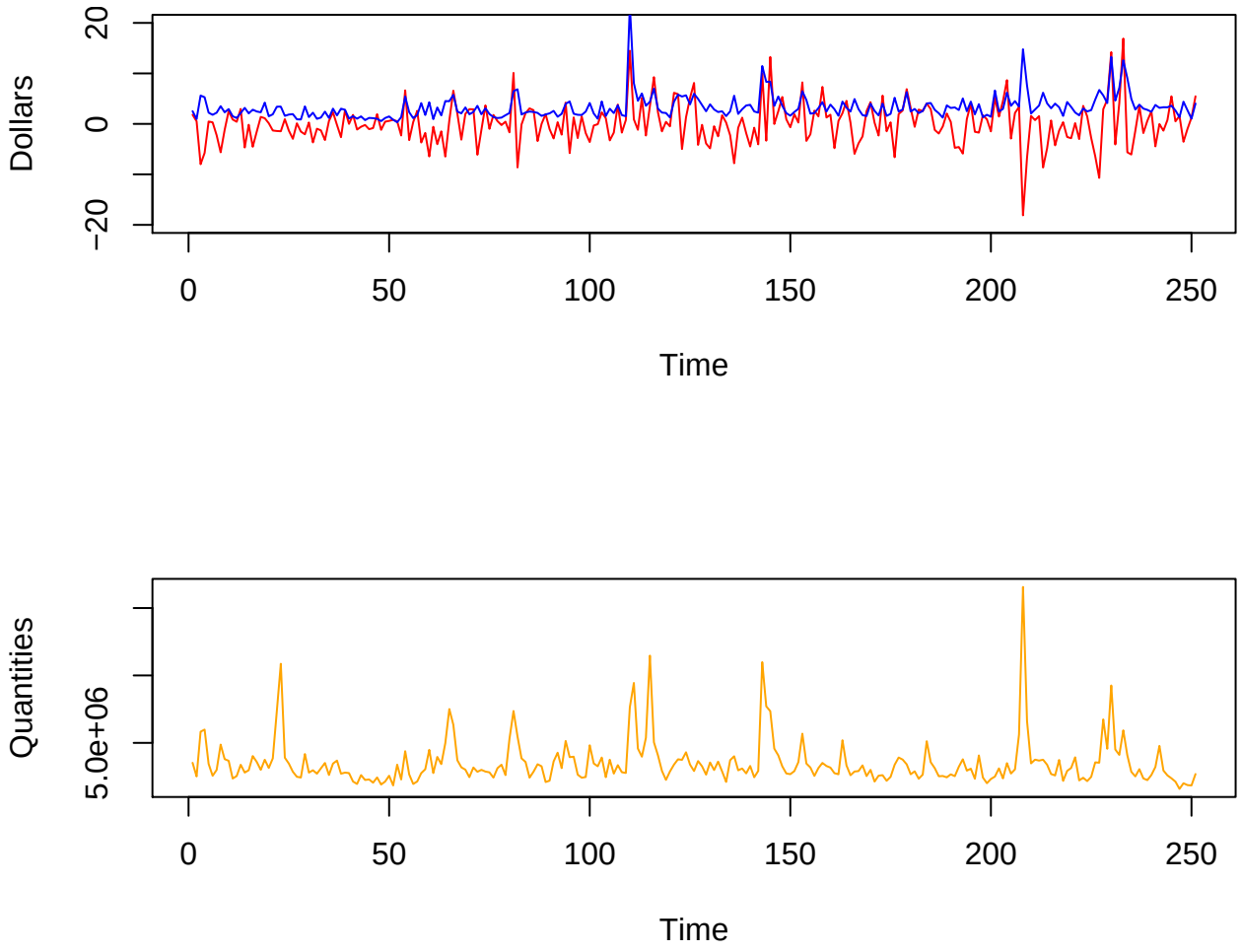
```
## Time Series Plot for Google

par(mfrow=c(2,1))
a=stock_list[[29]]
xts1<-xts(a$diff_OC,order.by=a$Date)
xts2<-xts(a$Volume,order.by=a$Date)
ts.plot(xts1,col="red",ylim=c(-10,15),ylab="Dollars")
lines(a$diff_HL,col="blue",ylab="Dollars")
ts.plot(xts2,col="orange",ylab="Volumes")
```


Dollars
10
0
−10
0
50
100
150
200
250
Time


Volumes
4e+06
1e+06
0
50
100
150
200
250
Time


Dollars
2
0
−2
0
50
100
150
200
250
Time


Quantities
8e+07
2e+07
0
50
100
150
200
250
Time



As we can see, the open-close difference and high-low difference are varying.

2 Methodology

In the lecture of Dr Glotzer [2], he mentioned three necessary approaches of clustering time series data:

- Distance measures
- Clustering algorithm
- Criteria for evaluating clusters

Our project will focus on the first two problem. In this section, we will describe several methods of distance measures and clustering algorithms. These methods will be used in analyze the dataset in our project.

2.1 Distance Measure

Distance measure is an important problem of time series clusting. Many approaches that have already been applied for the real-world data. According to the categorization introduced by Esling and Agon, time series data are usually divided into four categories: shape based, edit based, features based and structure based [3]. Different distance measures methods can be applied based on different categories of data. In our project, we focused on the univariate data measure and also gave an attempt to measure multivariate distance.

For univariate dataset, we can simply use Euclidean Distance measure. Then formula of eclidean measure is $\sqrt{\sum_{i=0}^{N-1}(x_i - y_i)^2}$. However, only apply the Euclidean Distance method may not be a good idea of measuring distance since the datasets are vary from time to time. Hence, we intend to apply Dynamic Time Warping method to measure univariate distance and attempt to use Spectrum-Based likelihood ratio method to measure the multivariate distance of our data. First, we will explain the theory behind each methond.

2.1.1 Dynamic Time Warping (DTW)

The basic idea of DTW is given two time series data, we stretch or compress them locally to make them resemble to each other as much as possible [4]. This means that if the distance values of two time series data are closed, we consider to put them into one cluster. So how could this method work? Assume we want to compare two series: a test,$X = (x_1, x_2, ..., x_N)$; and a reference $Y = (y_1, y_2, ...y_M)$. We used i and j as the indexes of X and Y. *Local dissimilarity function* is defined as $d(i, j) = f(x_i, y_j)0$. The core technique of this method lies the *Warping curve* $\phi(k)$:

$$\phi(k) = (\phi_x(k), \phi_y(k))$$

$$\phi_x(k)1...N,$$

$$\phi_y(k)1...M$$

We then apply the warping curve into local dissimilarity function to get the *Warping functions*. Given ϕ,we compute the average accumulated distortion between the warped

time series X and Y:

$$d_\phi(X,Y) = \sum_{k=1}^{T} d(\phi_x(k), \phi_y(k)) m_\phi(k) / M_\phi$$

where $m_\phi(k)$ is a per-step weighting coefficient and M_ϕ is the corresponding normalization constant, which ensures that the accumulated distortions are comparable along different paths [5].

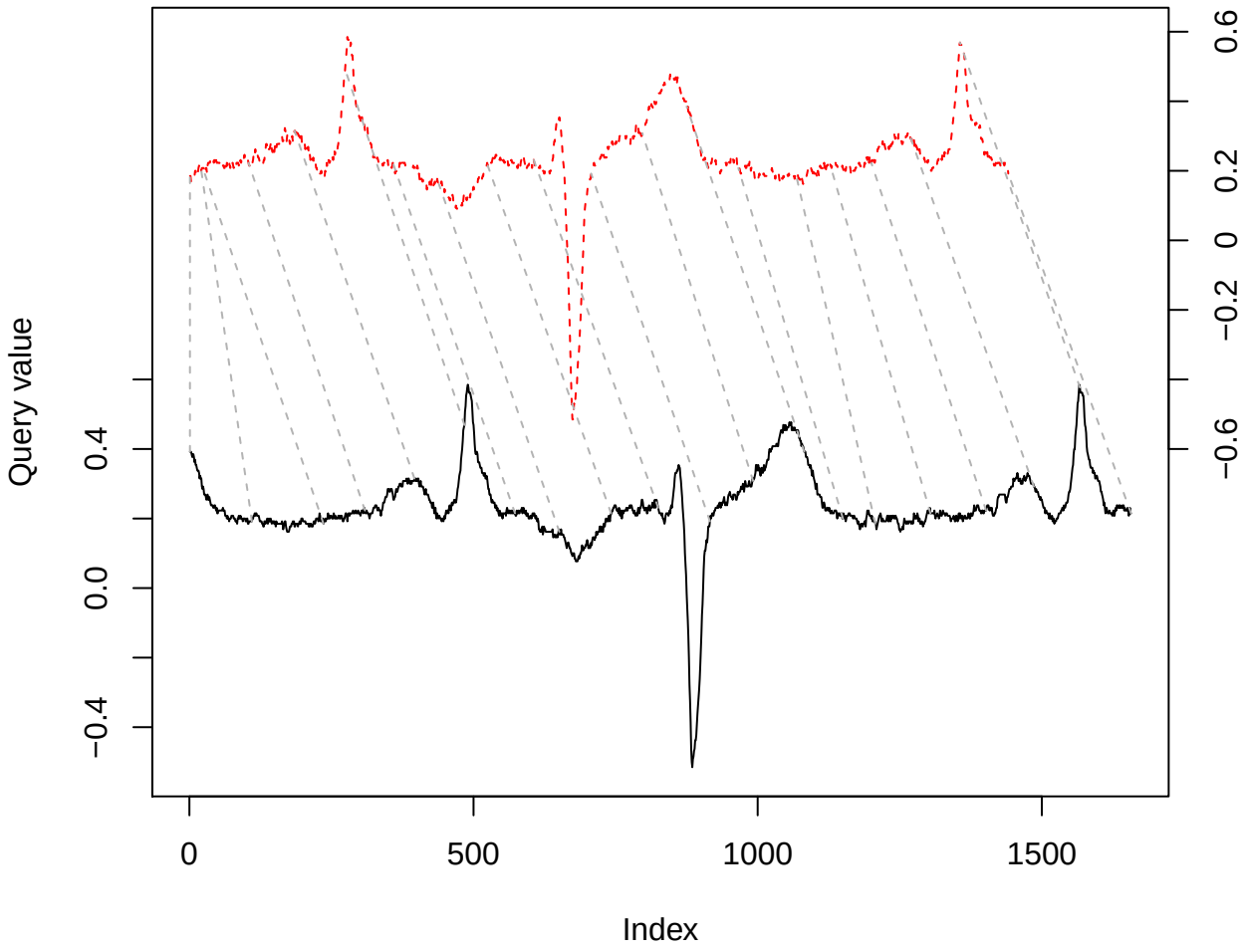
Idea is to find the optimal alignment ϕ, such that $D(X,Y) = min_\phi d_\phi(X,Y)$. And we can use dynamic programming algorithm to find the Optimal alignment [6].

Here we try an example, the "aami3a" which included in the R package `dtw` contains a reference electrocardiogram from the PhysioBank dataset [7].

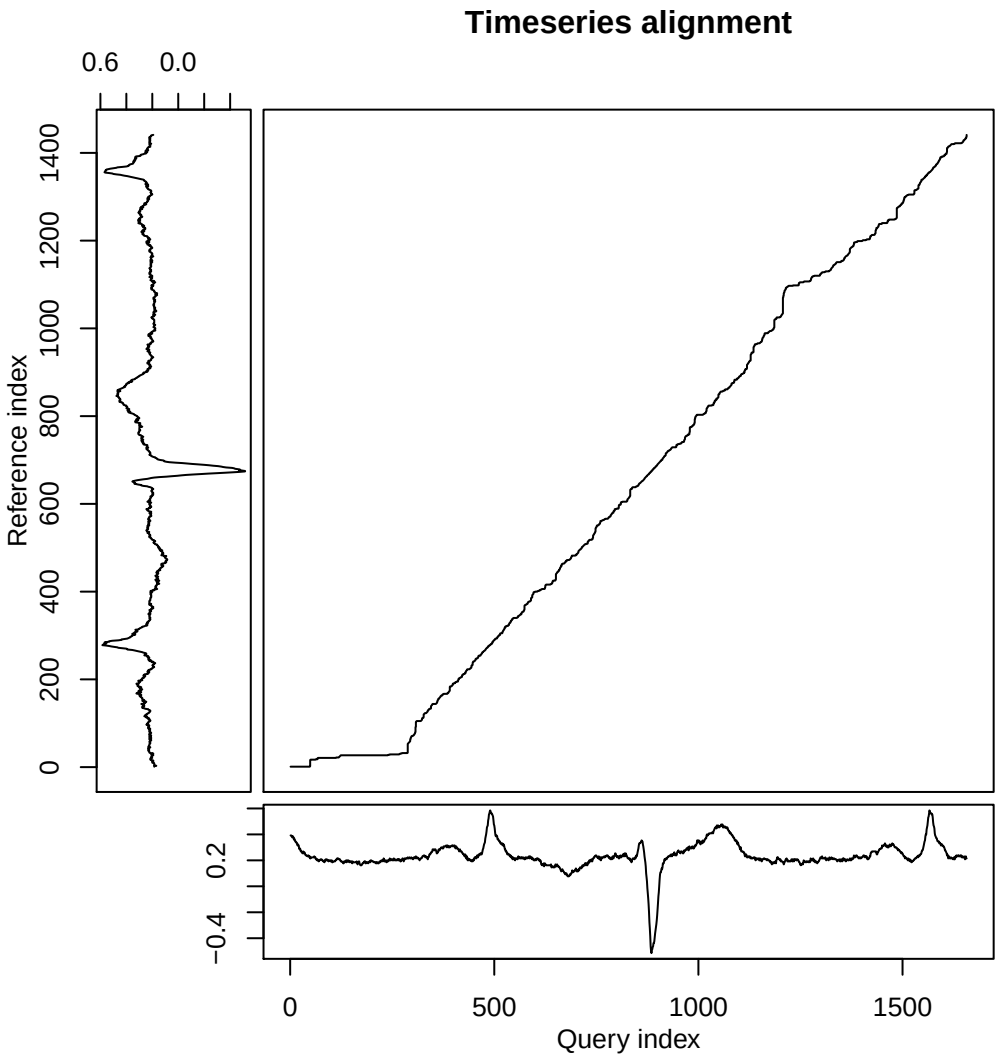
```
data("aami3a")
ref <- window(aami3a, start = 0, end = 2)
test <- window(aami3a, start = 2.7, end = 5)
alignment <- dtw(test, ref,keep=TRUE)

## Cumulative cost for the alignment
alignment$distance
```

```
## [1] 15.824
```

```
## Alignment plot of two simple time series data
plot(
    dtw(test,ref,keep=TRUE),
    type="two",offset=1,match.lty=2,match.indices=20)
```



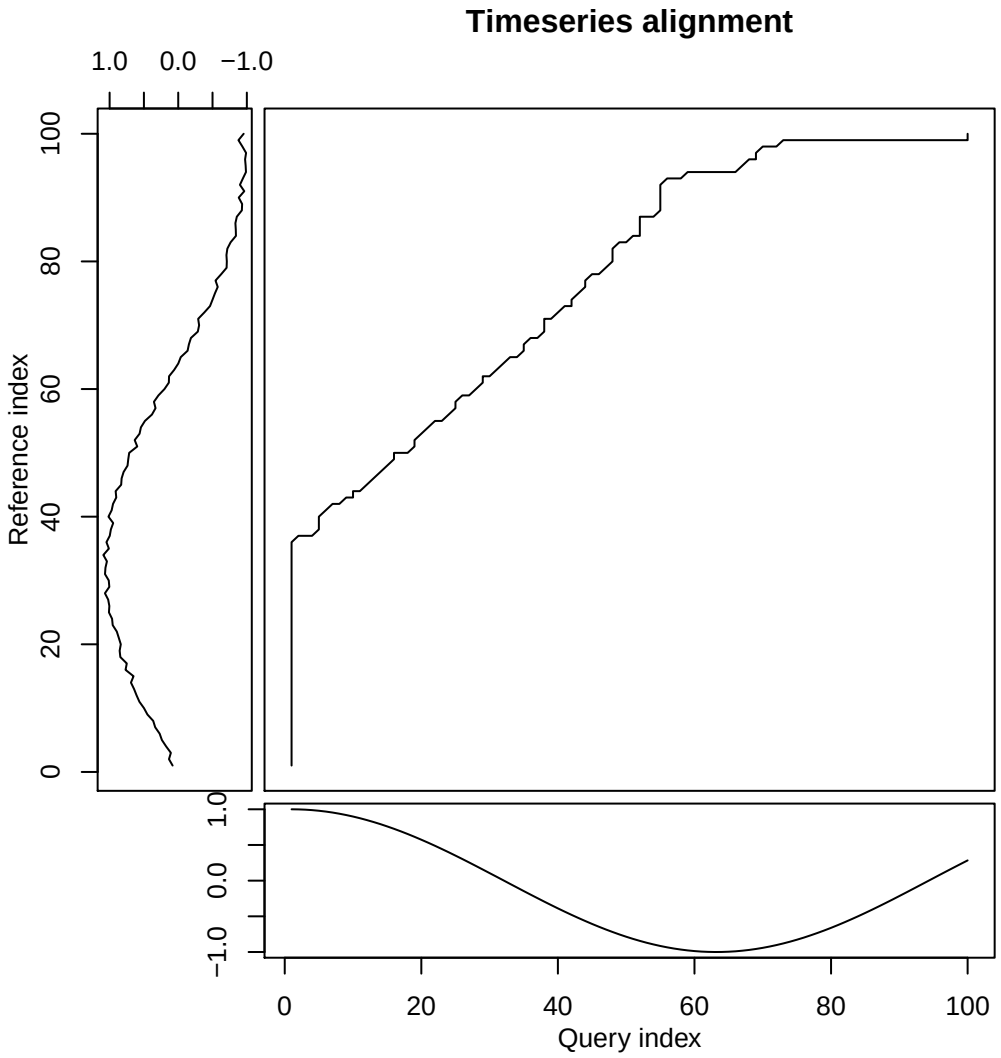

```
## A three-way plot of the alignment
plot(alignment,type="threeway")
```



Here we can have a brifely idea of how this method is applied. Below is another example of the application of DTW algorithm:

```
## A noisy sine wave as query
idx<-seq(0,5,len=100)
ref2<-sin(idx)+runif(100)/10

## A cosine is for template; sin and cos are offset by 25 samples
test2<-cos(idx)

## Find the best match using DTW method
alignment2<-dtw(test2,ref2,keep=TRUE)
```

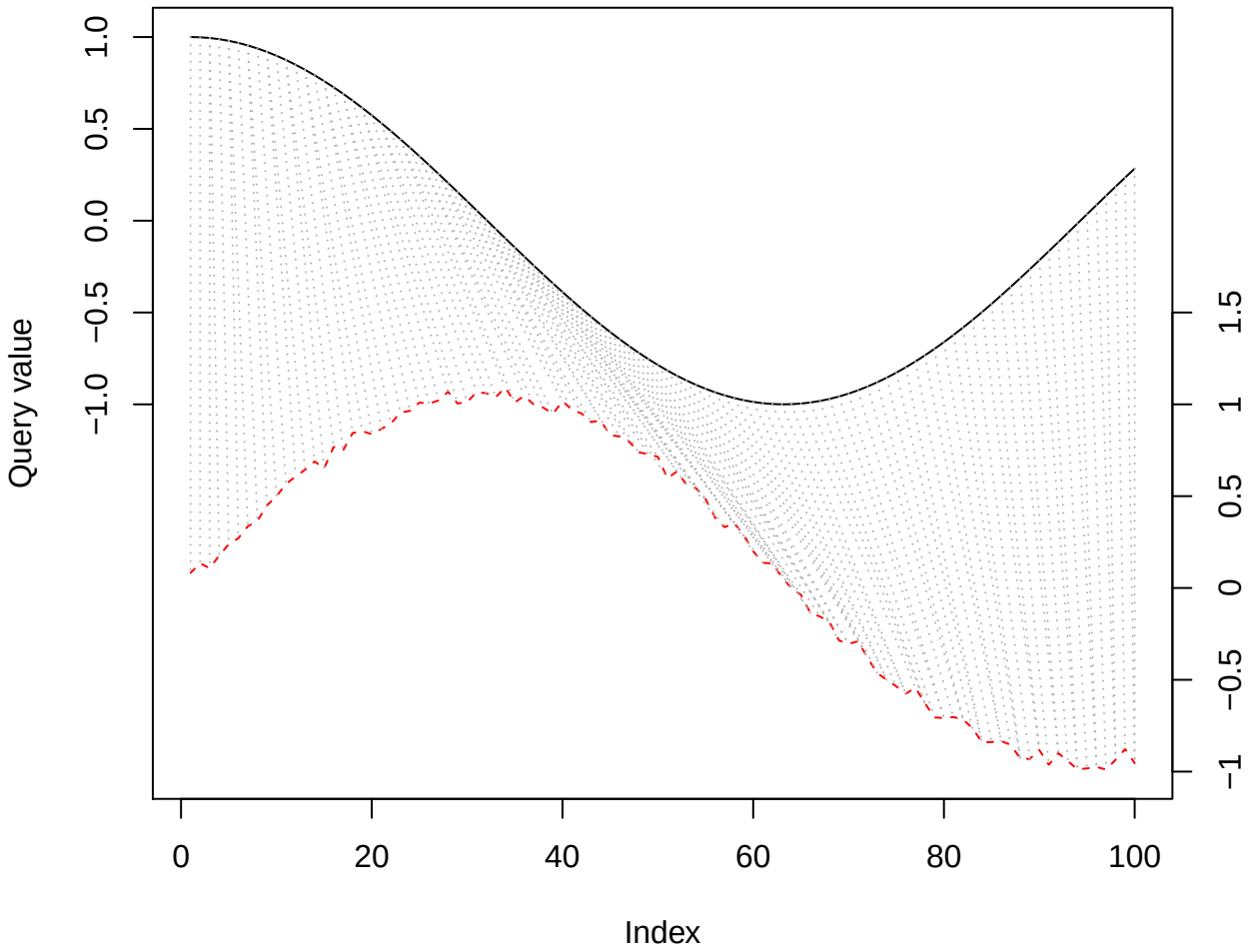
```
## Display the warping curve, i.e. the alignment curve
plot(alignment2,type="threeway")
```




```
alignment2$distance
```

```
## [1] 28.75576
```

```
## Align and plot with the Rabiner-Juang type VI-c unsmoothed recursion
plot(
    dtw(test2,ref2,keep=TRUE,
        step=rabinerJuangStepPattern(6,"c")),
    type="twoway",offset=-2)
```



In Section 4, we will apply this method to the stock market data.

2.1.2 A Brife Introduction of Spectrum-Based Similarity

Multivariate data analysis is more difficult than univariate data analysis. For measure the distance of multivariate time series data, we introduce an idea called Spectrum-Based likelihood ratio test method on the estimated cross-spectra of the series yields a quasi-distance between series [8]. Basic idea of this mehtod is based on the likelihood ratio test with $H_0 : \gamma_X(\omega) = \gamma_Y(\omega)$, where γ_X,γ_Y are the spectral density matrices of multivariate time series Xt,Yt. The likelihood ratio is defined as $Q_{XY}(\omega)$ [9].

The next step is to cluster the n-dimentional time series into different groups use a pairwise quasi-distance between two multivariate time series data x_t and y_t based on the likelihood ratio test we mentioned above. The quasi-distance $Q^{\star}_{XY} = 1/M \sum_{j=1}^{M} Q_{XY}(\omega_(j))$ is the average of the M smallest values of the likelihood ratio, where M is to take $\sqrt{(T)} <=$

$M <= T/10$ [10].

After deriving the quasi-distance, we can apply the hierarchical clustering algorithm campared with several time sereis data given the quasi-distance [11].

2.2 Time Series Clustering Algorithm

Similiar to the static data clustering, the time series data can apply clustering algorithm to recognize the unlabeled data. The clustering methods can classify a set of unlabeled data into different groups based on the availiable data and the purpose of the application. The most important elements to consider are the (dis)similarity or distance measure, the prototype extraction function (if applicable), the clustering algorithm itself, and cluster evaluation [13].

since the dynamic data of time series, algorithms are developed to transformed the data to the static data, or modify the similiarity definiation to the apporate one.

The methods can distinct different type of data, such as, univariate data, multivariate data, and equal or unequaled length data. There are lost of algorithms are developed to deal with time series data. Basically, there are three different approachs algorithm to work for static data (a) Raw-data-based, (b)featured-based, (c)model-based. For time series data, hierachical methods, partitioning methods and model based methods are used the most commonly. In this section, we will only talk about hierachical methods and partitioning methods.

2.2.1 Hierachical Methods

Hierachical clustering is working by merging data into a similiar group, then formed a tree-based clusteirng diagram. The clustering start from a point them split the data or merge the data according to the similiarity. The most two popular hierachical methods is agglomerative and divisive method. The difference of these two methods is the order they token to classify data.

The Agglomerative method is working from bottom to the top. It starts from each unite data, which is also cluster, then merging the cluters into a larger one. Untill all data is

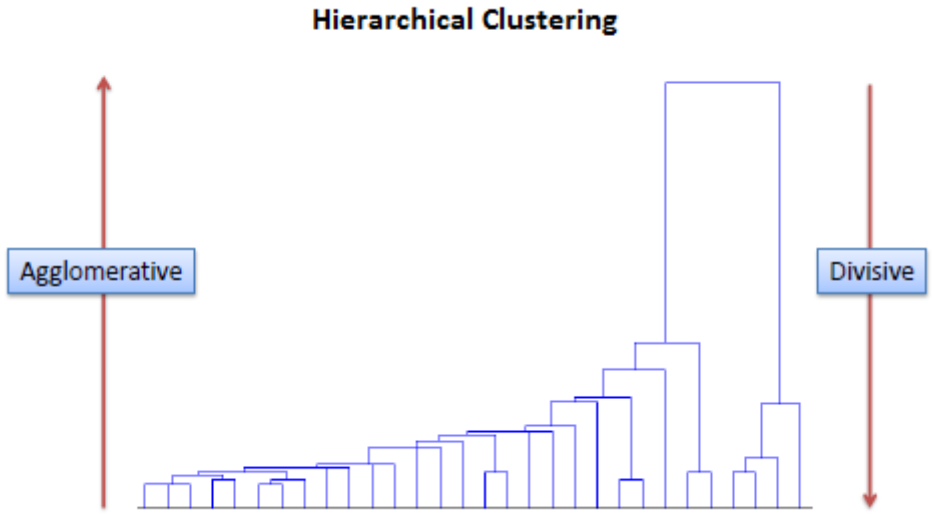


Figure 1: Agglomerative vs. Divisive

merged into one large cluster.

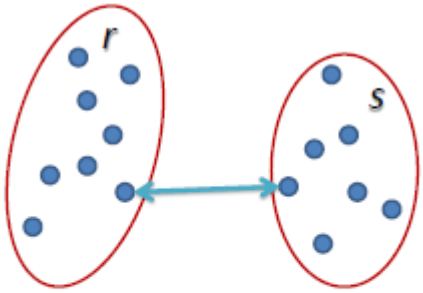
The divisive method distinc data using the opposite direction. It takes from top to bottom. All data are in one cluster, the divisive method works by spliting data from the top cluster, and data are grouped my similiarity, unitll each data is distinced in one clusrer.

2.2.1.1 Linkage

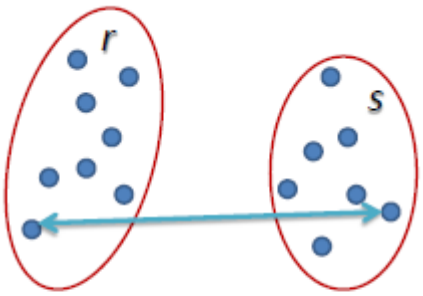
When use hierachical methods, it is important to decide where to merge or split cluster. Similiarity measurement is most used in clustering algorithm. Linkage is the most common method to measure the similiarity. It measures distance between clusters and inter-clusters, then cluster will merge the closest two clusters.

a. Single linkage is used to measure the distance between two clusters that is defined as the shortest distance between two points in each cluster.

b. Complete linkage calculates the longest distance between two points in each clusters.



$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$



$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

2.2.1.2 Steps

(1). Start by assigning each item to its own cluster, so that if you have N items, you now have N clusters, each containing just one item. Let the distances (similarities) between the clusters equal the distances (similarities) between the items they contain.

(2). Find the closest (most similar) pair of clusters and merge them into a single cluster, so that now you have one less cluster.

(3). Compute distances (similarities) between the new cluster and each of the old clusters.

(4). Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

2.2.2 Partitioning Methods

Besides hierachical clusteirng, another popular clustering method is partitioning method. It calculates the distance between data to all clusters' medoid. Medoid is simply a representative objec from a cluster. Based the distances calculated, data will be grouped to the closest cluster. The main idea of this method is it minimize an objective funtion, which the total distance between all patterns from their representive cluster centers [14] method, the number of clusters is needed at begining.

2.2.2.1 Steps

Let $X={x_1, x_2, x_3,….,x_n} $ be the set of data points and $V = \{v_1, v_2, v_3, ..., v_c\}$ be the set of centers.

(1). Randomly select 'c' cluster centers.

(2). Calculate the distance between each data point and cluster centers.

(3). Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.

(4). Recalculate the new cluster center using objective function.

(5). Recalculate the distance between each data point and new obtained cluster centers.

(6). If no data point was reassigned then stop, otherwise repeat from step (3).

2.2.2.2 K-Means and Fuzzy C-Means

The different objection function lead to different partitioning methods, K-Means and Fuzzy C-Means.

K-means algorithm is calcuating the hard partition, which each member of the data belongs to only one cluster, and clusters are mutually exclusive. The objective funtion of hard partition is

$$\sum \sum ||x_i^j - c_j||^2$$

Fuzzy clustering creates a fuzzy or soft partition in which each member belongs to each cluster to a certain degree. For each member of the data, the degree of belongingness is constrained so that its sum equals 1 across all clusters.[15] The objective function of soft partition is

$$min \sum \sum u_{p,c}^m d_{p,c}^2$$

and

$$\sum u_{p,c} = 1, u_{p,c} \geq 0$$

3 Seminal Research Paper Discussion

In this section, we are going to discuss two seminar research paper about time series classification and clustering. The first paper we will discuss is "Clustering of time series data-a survey" by T. Warren Liao. This paper was published in 2005 in the Journal of the Pattern Recognition Society. It gives a good overview and summarizes previous works that

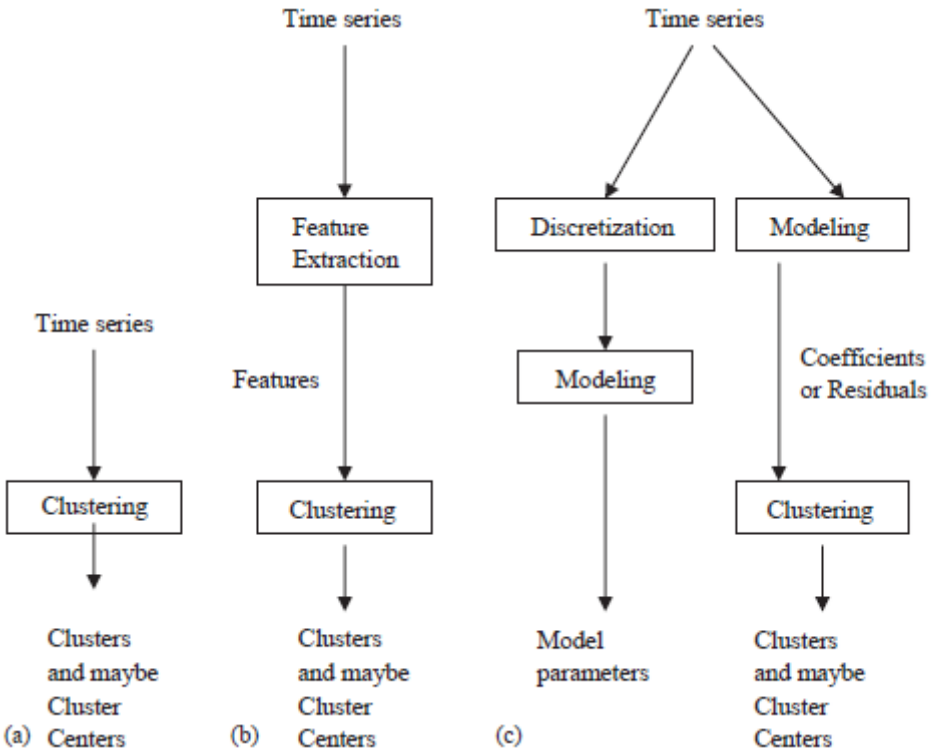


Figure 2: FIGURE 4.1

investigated the clustering of data in various application domains.

Liao starts the paper with a clear definition for the goal of clustering. The purpose of clustering is to identify structure in an unlabeled data set by organizing data into homogeneous groups where the within-group-object dissimilarity is minimized and the between-group-object dissimilarity is maximized. Then, Liao explains that the logic of all the clustering algorithms are the same; they all either try to modify the existing clustering algorithms for static data to make it applicable for time series data or they try to convert time series data into the form of static data. Liao then mentions three time series clustering approaches which are raw-data-based, feature-based and model-based which are shown in the following figure.

The first approach usually works with the raw time series data and therefore it is called raw-data-based approach. The major modification for this approach lies in replacing the distance/similarity measure for static data with an appropriate one for time series. The second and third approach converts a time series data into either a feature vector of lower dimension or a number of model parameters and then applies conventional clustering algorithms.

Afterwards, Liao explains four different Clustering algorithms which are Relocation clustering, Agglomerative hierarchical clustering, k-Means clustering and Self-organizing maps.

Relocation clustering initially starts with prescribed k number of clusters. Then, for each time point, it computes the dissimilarity matrix. If it is able to find a clustering that is better than the current clustering in terms of the generalized Ward criterion function, it swaps the members between those two clusters. If no such better clustering exists, then the algorithm stops.

Agglomerative hierarchical clustering works by grouping the time series into a tree of clusters. It starts by placing each series into its own cluster and then merges the clusters into larger clusters depending on their similarities. In other words, initially each series has its own cluster. Then, depending on the distance of these atomic clusters, the most similar clusters merge into a larger cluster until all the objects are in a single cluster or until a certain termination condition holds.

The main idea of K means clustering is to solve a minimization problem in which the objective function is the total distance between all patterns from their respective cluster centers.

Self-organizing map, also sometimes called as Kohonen map or network, is a class of neural networks with neurons arranged in a two dimensional structure and trained by an iterative procedures. During the training process, the weight vectors are updated. Self-organizing maps do not work well with time series of unequal lengths because of difficulty in defining the dimension of weight vectors.

Later, Liao explains different similarity/distance measures including Euclidean, Pearson's correlation coefficient and related distance, Short time series distance and Dynamic time warping distance.

He shows for a P-dimensional vector, how to compute the Euclidean distance and how it can be generalized to Mikowski distance. He then shows how to calculate the correlation factor for same P-dimensional vector. Then he explains dynamic time warping distance which is explained in details in section 2.1.1 of this project.

Liao finally discusses about the clustering result evaluation criteria. He introduces two different categories of evaluation criteria: known ground truth and unknown ground truth.

One of the main difference between these categories is that the number of clusters is usually known in the former and not known in the latter. He gives how the cluster similarity measure is defined for both known and unknown ground truth.

In conclusion, Liao in this paper studies three key components of time series clustering that are the clustering algorithm, the similarity/dissimilarity measure and the evaluation criterion. He also mentions some potential topics for future study such as implementation of genetic algorithm to time series clustering, integrating an unsupervised clustering algorithm with a supervised classification algorithm and developing scaled-up clustering algorithms to handle larger data sets.

The second research paper we want to discuss is "Comparing Time-Series Clustering Algorithm in R Using the dtwclust Package" by Alexis Sarda-Espinosa. Although there are many different R packages written for time series clustering such as flexclust, cluster, TSdist, TSclust and pdc, we used an R package called dtwclust in our project as it is very user friendly and has a very useful and recently updated manuscript with examples and explanations. Dr. Pipiras, in his time series lecture notes also use this dtwclust package. Therefore, we decided to discuss Espinosa's paper about comparison of time series clustering algorithms and their implementation in R using the dtwclust package. In the previous paper, Liao introduced different dissimilarity/distance measures including dynamic time warping distance. The main focus of our second paper is the dynamic time warping distance and implementation of different clustering algorithms in R.

Espinosa first starts with introducing different common time series such as stock data, medical data and machine monitoring and the challenges/high complexity because of the length of the series. He emphasizes on the importance of dissimilarity or distance measure, clustering algorithms and cluster evaluation. In the next section, he mentions l1 and l2 vector norms and how common it is to use these distance measures. However, he also mentions that those measures can only be efficiently computed for series of equal length and that they are very sensitive to noise, scale and time shifts. Therefore, he introduces a new distance, Dynamic Time Warping, whose functions are included in dtwclust package. He also gives example of different paper and researchers that use DTW measure for clustering to emphasize on how common it is.

Espinosa then introduces the complexity of DTW which is quadratic when the length of two time series are similar. Since it is more useful but more complex than some other measure distance, he warns about how slow using DTW distance measure can be in non-complied programming languages as it uses dynamic programming steps. After explaining DTW, Espinosa mentions two possible modifications of DTW which are Global DTW constraint and Lower Bound for DTW. Global DTW constraint provides an upper bound for the Local Cost Matrix. He mentions Sakoe-Chiba window which constraints the allowed region along the diagonal of the Local Cost Matrix. He also shows how to implement Sakoe-Chiba window in R using dtwclust package. Another modification Espinosa mentioned is Lower Bound for DTW that guarantees being less than or equal to the corresponding DTW distance. As a result, DTW becomes less expensive to compute and it increases the effectiveness of the measure.

Espinosa then introduces different time series clustering algorithms which were also discussed in Liao's paper. The main addition of this paper to Liao's article is that Espinosa also gives R implementation examples and their graphical outcomes using the dataset in dtwclust package. He gives different examples using hierarchical clustering, partitional clustering and fuzzy clustering. He emphasizes the main difference of dtwclust package from other packages like TSdist, cluster and clue is that those packages are more specialized, dealing with specific tasks. They can be only focusing on distance calculations or only on hierarchical clustering or so on. Whereas dtwclust provides a more general purpose, having enough flexibility to allow for the most common approaches to be used from same package. Espinosa gives some future package extensibility ideas: he states that although the most common time series prototypes are implemented, it is possible to provide custom centroid algorithms and implement a custom clustering algorithms if desired. Finally, Espinosa stated that as dtwclust is hosted on an open source platform, GitHub, it is possible to make significant modification although it is not an easy task.

4 Documentation of R code

4.1 Univariate Analysis

4.1.1 Distance Measure

For univariate distance measures, we choose the open-close price difference as the univariate variable. We first apply the DTW methods to calculate the distance of open-close price difference between the companies.

```
stock_list2 <- vector(mode="list", length=length(unique(data.new$Name)))
i=1
for (c in unique(data.new$Name)){
    stock_list2[[i]] <- stock_list[[i]][,4]
    i=i+1
}
```

```
## We use different packages to apply the dtw method, the result is the same
DM_DTW<-TSDatabaseDistances(stock_list2,distance="dtw")
dist_DTW <- dist(stock_list2, stock_list2, method = "dtw_basic")
```

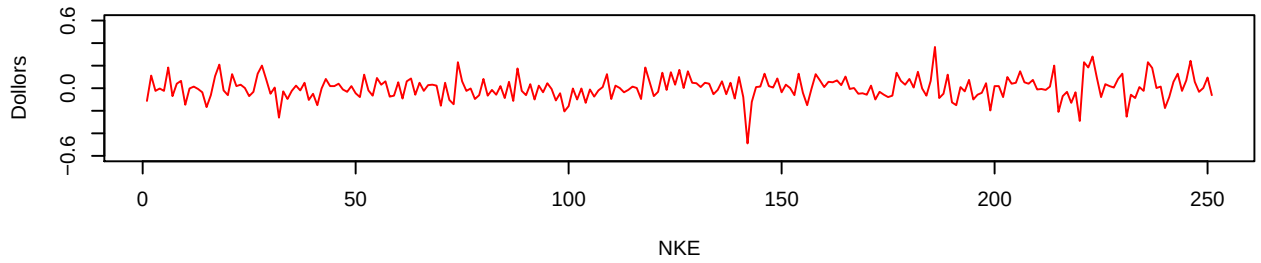
```
#DM_DTW
diag(dist_DTW) <- NA
```

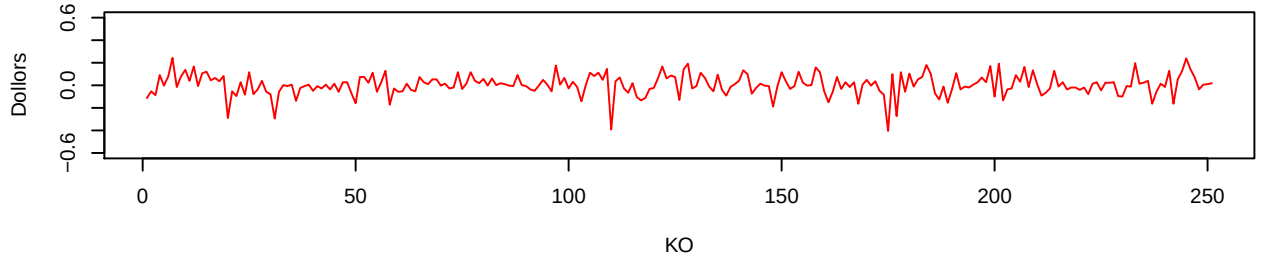
```
## This step shows the closest distance between companies based on
## the one step distance value
NN_DTW <- apply(dist_DTW, 1, which.min)
NN_DTW
```

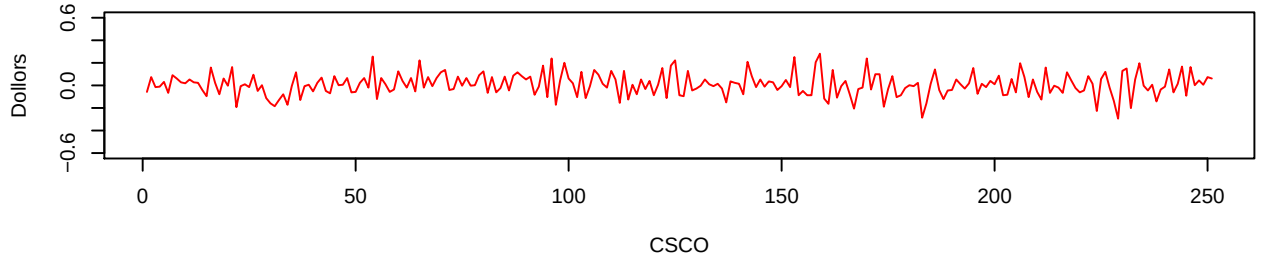
```
##  [1] 28 15 31 16 21 27 22 22 27  7 22 18 31 19 22 22 23 28 22  8  7  8  8
## [24] 20 19 31  7 15  4 29  7
```

As we can see the eighth company and the twenty-second company have the closest distance. So we plot the time series graph of the two companies. Besides, the 7th company in the dataset also shows very close to the companies I just mentioned. The plots are below.

```
c1=as.ts(stock_list2[[8]])
c2=as.ts(stock_list2[[22]])
c3=as.ts(stock_list2[[7]])

par(mfrow=c(3,1))
ts.plot(c1,col="red",ylim=c(-.6,.6),ylab="Dollors",xlab="NKE")
ts.plot(c2,col="red",ylim=c(-.6,.6),ylab="Dollors",xlab="KO")
ts.plot(c3,col="red",ylim=c(-.6,.6),ylab="Dollors",xlab="CSCO")
```







4.1.2 Clustering Algorithm

```
ret = function(i){
  l = subset(data,data$Name==comp[i])
  l=l$Close-l$Open
  l=scale(l)
  l[145] = 0
  l = l[1:251]
}
```

```
comp = levels(data$Name)
```

To be able to use dist function id dtwclust library, we created a list called list as follows:

```
list = list(ret(1),ret(2),ret(3),ret(4),ret(5),ret(6),ret(7),ret(8),ret(9),ret(10),
            ret(11),ret(12),ret(13),ret(14),ret(15),ret(16),ret(17),ret(18),ret(19),ret
            ret(21),ret(22),ret(23),ret(24),ret(25),ret(26),ret(27),ret(28),ret(29),ret
            )
```

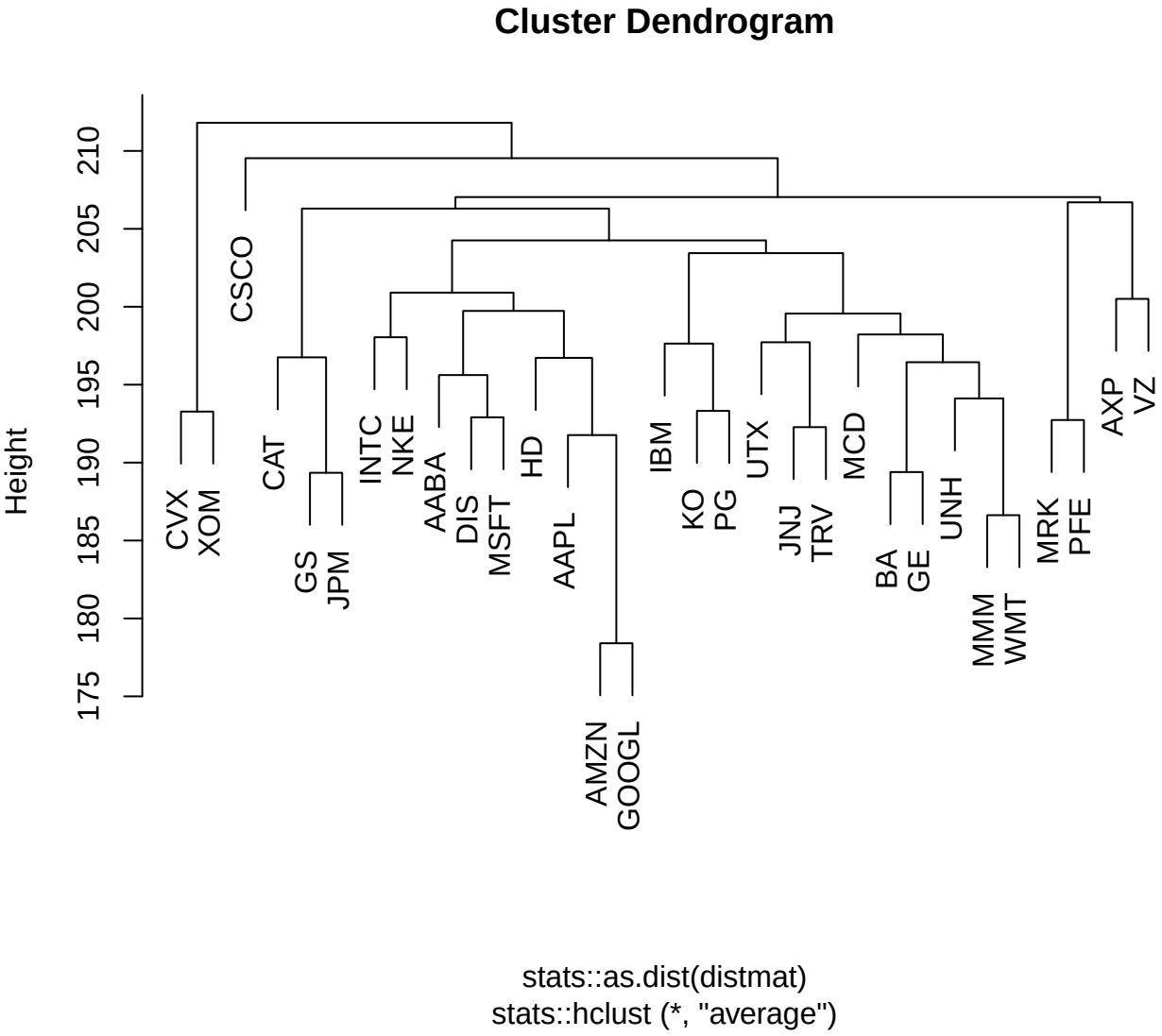
```
list=setNames(list,comp)
```

To perform a hierarchical clustering, we used tsclust function in dtwclust package. We have done it for two different distance measurement.

```
hc_DTW = tsclust(list, type="h", k=5, distance= "dtw_basic")
hc_EUC = tsclust(list, type="h", k=5, distance= "euclidean")
```

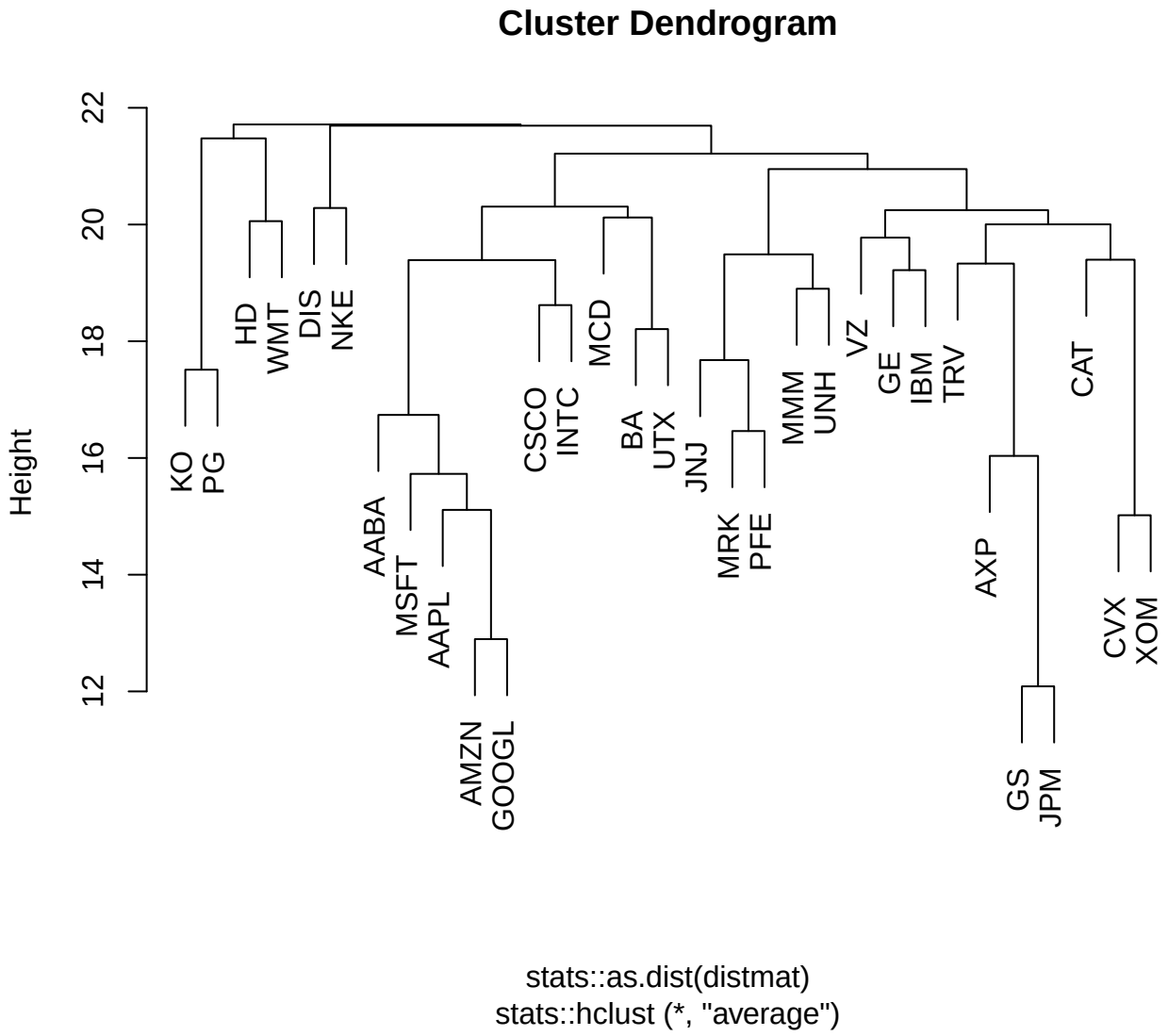
This is the dendogram for dtw distance

```
 plot(hc_DTW)
```



The dendogram for euclidean distance

```
plot(hc_EUC)
```



By looking at the dendograms, we decided to use 5 clusters for both distance measures. Here are the members of each cluster:

```
hc_DTW@cluster
```

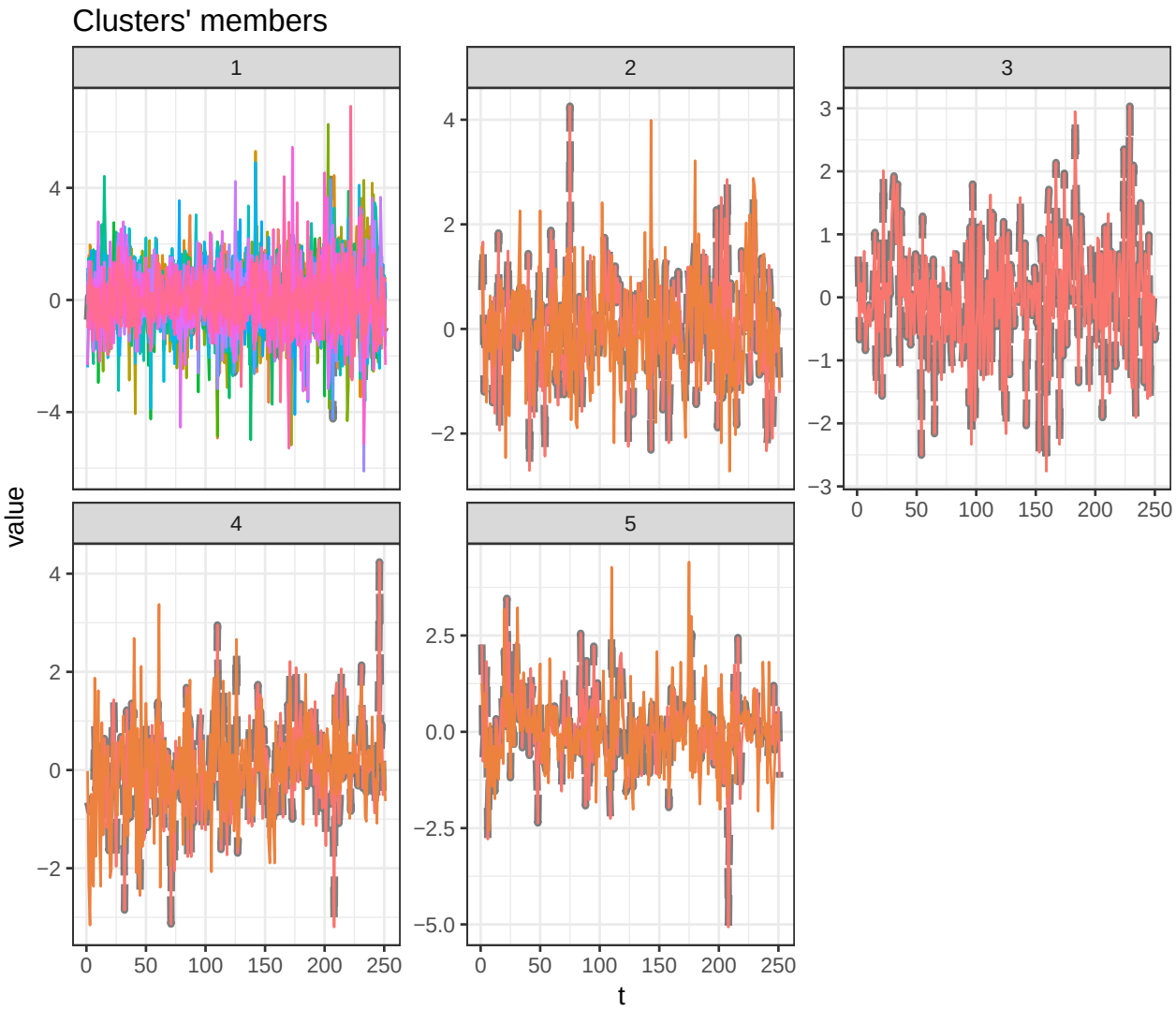
```
##  AABA  AAPL  AMZN   AXP    BA   CAT  CSCO   CVX   DIS    GE GOOGL    GS
##     1     1     1     2     1     1     3     4     1     1     1     1
##    HD   IBM  INTC   JNJ   JPM    KO   MCD   MMM   MRK  MSFT   NKE   PFE
##     1     1     1     1     1     1     1     1     5     1     1     5
##    PG   TRV   UNH   UTX    VZ   WMT   XOM
##     1     1     1     1     2     1     4
```

```
  hc_EUC@cluster
```

```
##   AABA   AAPL   AMZN    AXP     BA    CAT   CSCO    CVX    DIS     GE GOOGL     GS
##      1      1      1      2      1      2      1      2      3      2      1      2
##     HD    IBM   INTC    JNJ    JPM     KO    MCD    MMM    MRK   MSFT    NKE    PFE
##      4      2      1      2      2      5      1      2      2      1      3      2
##     PG    TRV    UNH    UTX     VZ    WMT    XOM
##      5      2      2      1      2      4      2
```

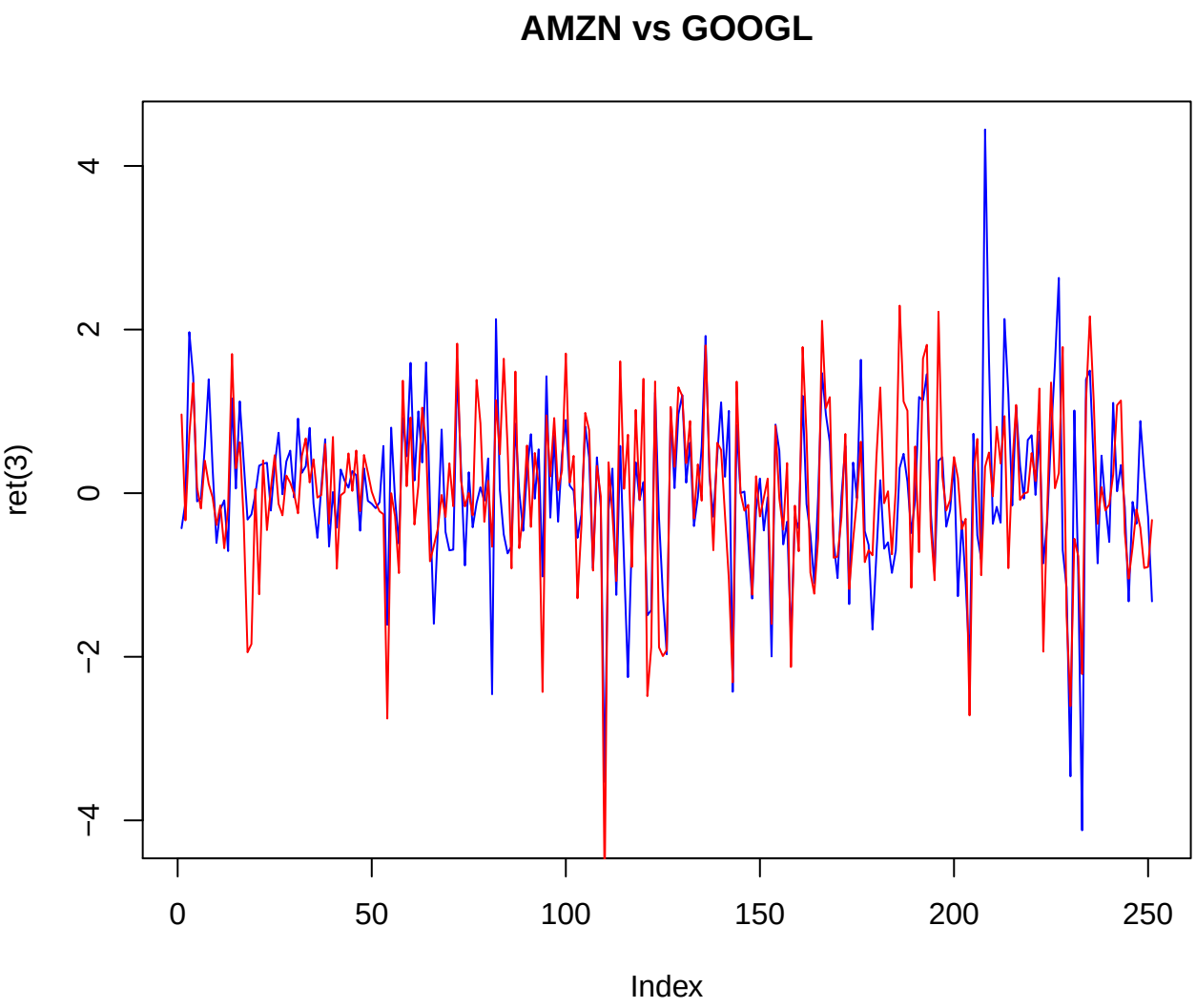
The plot of behavior of the members of each clusters for DTW is as follows:

```
plot(hc_DTW,type="sc")
```

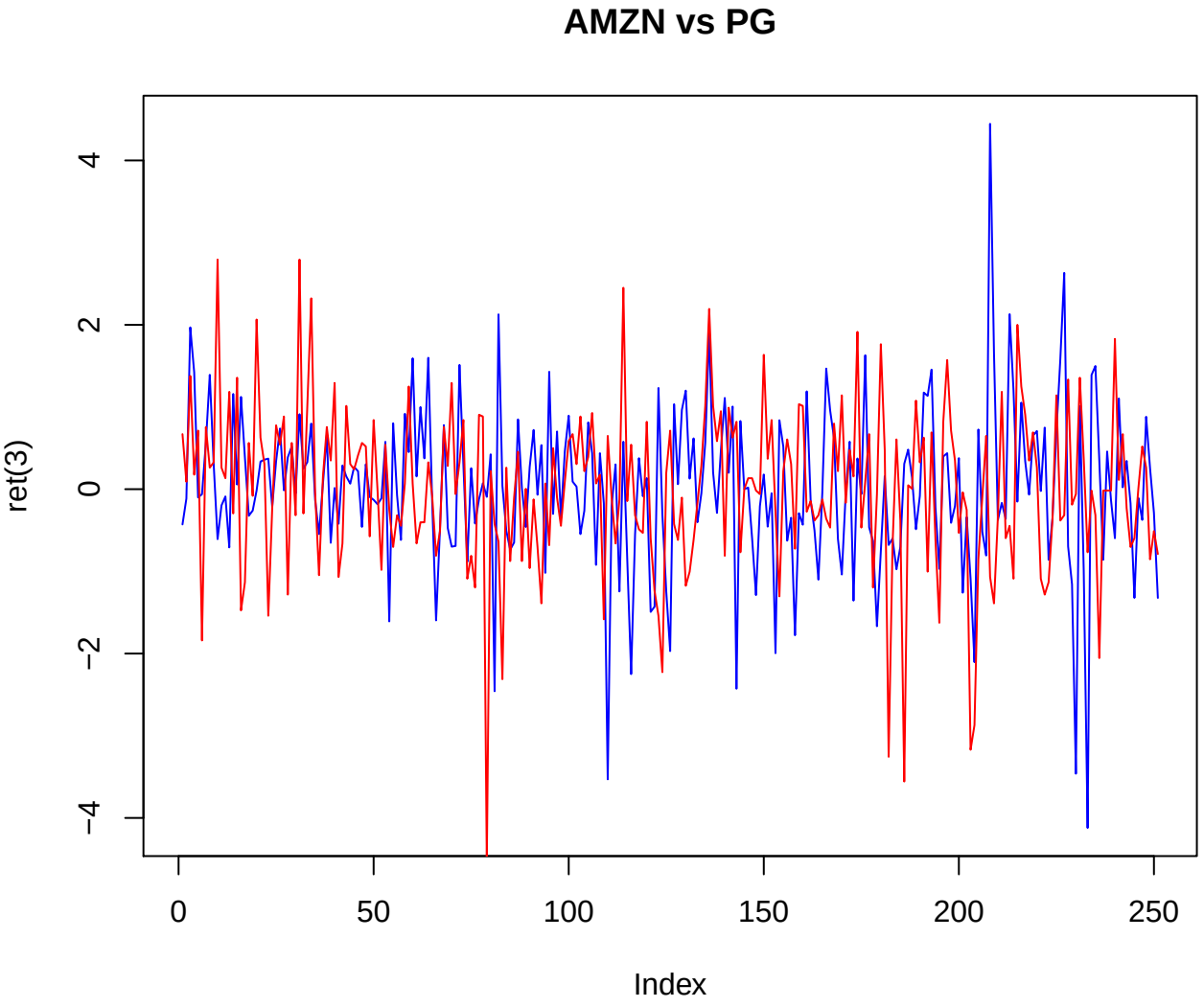


We observed that companies in the same sector are usually in the same cluster and have similar return behavior.

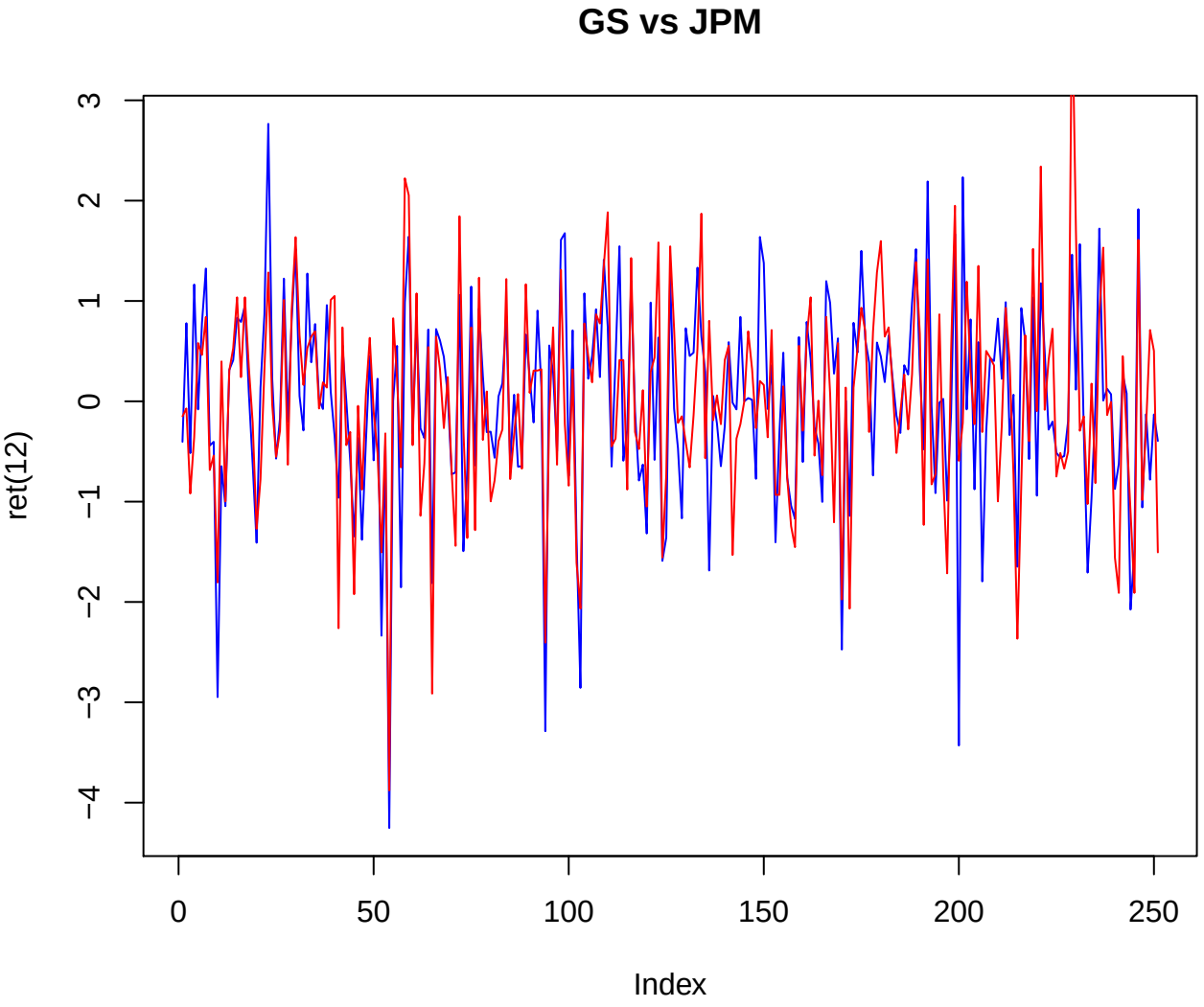
```
plot(ret(3),type='l',col='blue', main='AMZN vs GOOGL')
lines(ret(11),col='red')
```



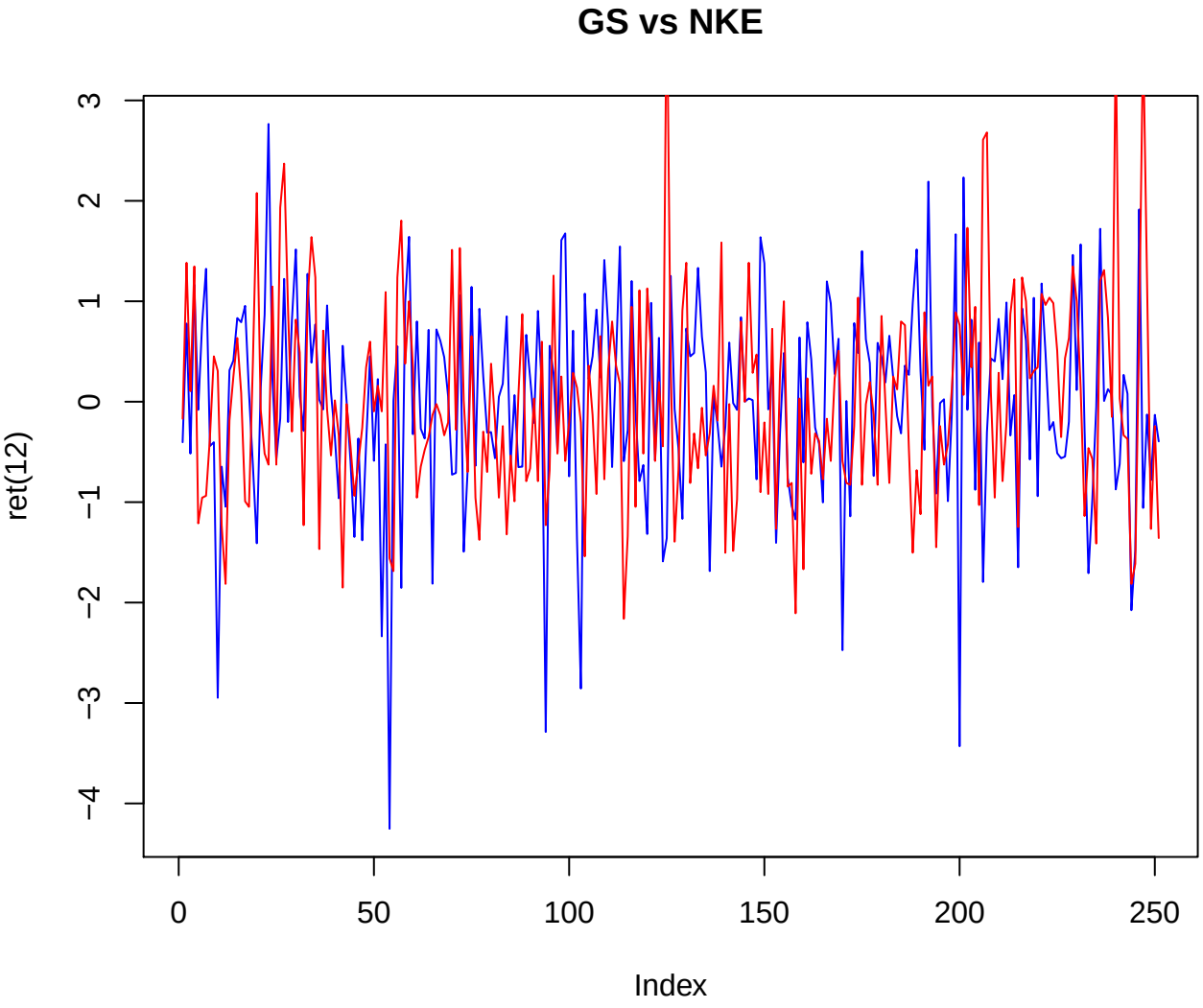

```
plot(ret(3),type='l',col='blue', main='AMZN vs PG')
lines(ret(25),col='red')
```




```
plot(ret(12),type='l',col='blue', main='GS vs JPM')
lines(ret(17),col='red')
```




```
plot(ret(12),type='l',col='blue', main='GS vs NKE')
lines(ret(23),col='red')
```



4.1.3 Multivariate Distance Measure

For the multivariate distance analysis, we revised the method that provided by Dr Glotzer [12]. We measure the quasi-distance between company MMM and AXP as well as company AAPL AND BA.

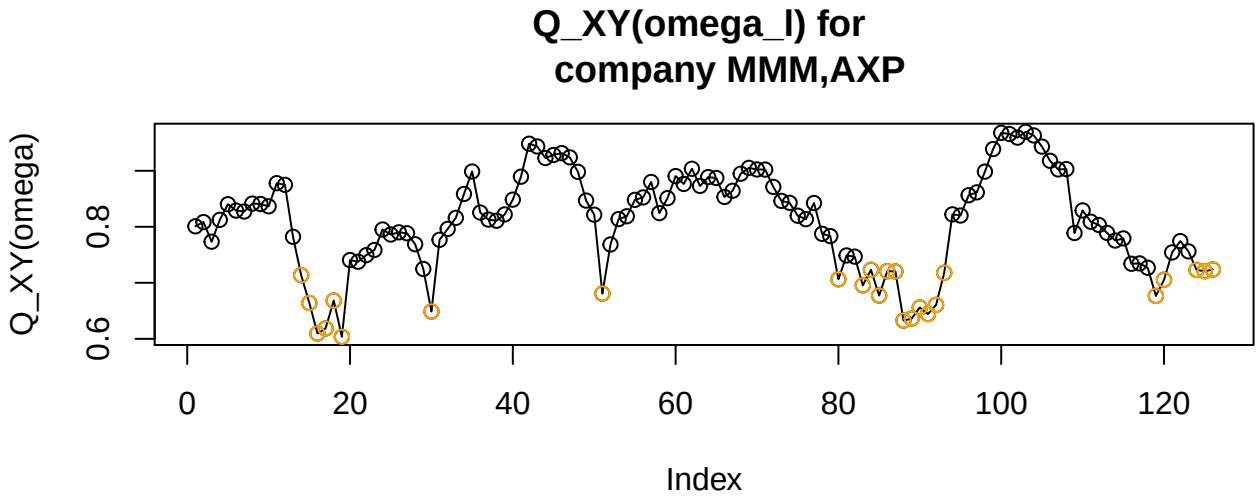
```
N=length(unique(data.new$Name))
T=251


data.multi<-NULL
for (i in 1:N){
    X=t(as.matrix(stock_list[[i]][,c(3,4,5)]))
```

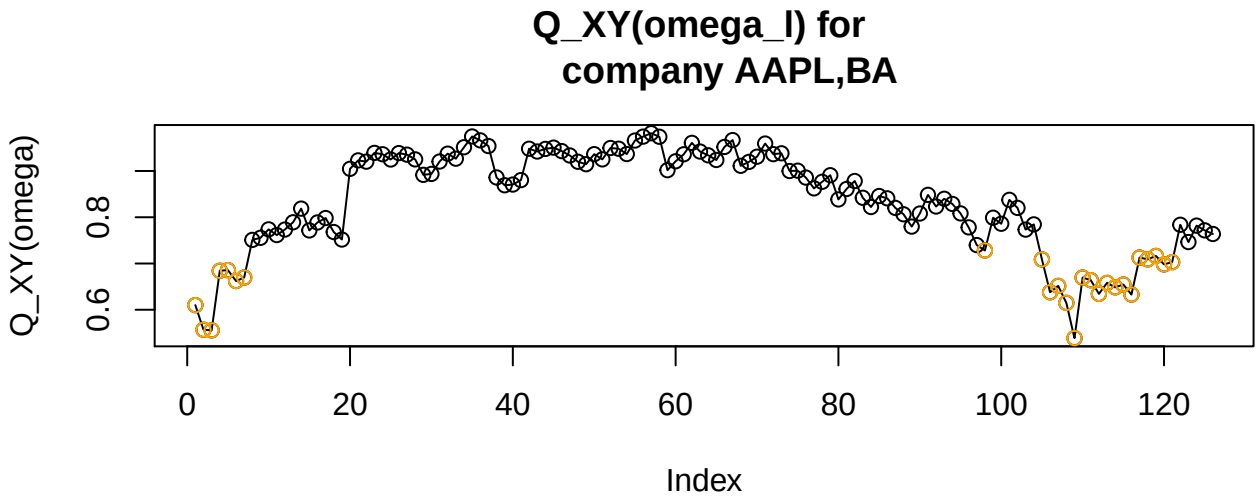
```
    data.multi<-rbind(data.multi,X)
}

data.multi2<-t(data.multi)
data.multi2 = (data.multi2 -
                    matrix(rep(1,T),ncol=1)%*%apply(data.multi2,2,mean))/ (matrix(rep(1,T
```

```
data_fft = mvfft(data.multi2)
data_fft = data_fft[c(1:ceiling(251/2)),]

data_per = (1/2/pi/T)*data_fft*Conj(data_fft)
data_per = abs(data_per)

data_cper = NULL
for (p in 1:N) {
  cper_new = (1/2/pi/T)*data_fft[,2*p-1]*Conj(data_fft[,2*p])
  data_cper = cbind(data_cper, cper_new)
}

T2 = dim(data_per)[1]
P2 = dim(data_per)[2]

T3 = dim(data_cper)[1]
P3 = dim(data_cper)[2]

# smoothing periodograms and crossperiodograms
m = 10 ;

data_per_ud = apply(data_per,2,rev) ;
data_per_2 = rbind( rep(0,P2), data_per_ud[(T2-m):(T2-1),],
```

```
                    data_per, data_per_ud[2:(m+1),] )

data_per_sm = apply(data_per_2,2,cumsum)
data_per_sm = data_per_sm[(2*m+2):dim(data_per_sm)[1],] - data_per_sm[1:(dim(data_per_sm
data_per_sm = data_per_sm/(2*m+1)

data_cper_ud = apply(data_cper,2,rev)
data_cper_2 = rbind( rep(0,P3), data_cper_ud[(T3-m):(T3-1),],
                     data_cper, data_cper_ud[2:(m+1),] )

data_cper_sm = apply(data_cper_2,2,cumsum)
data_cper_sm = data_cper_sm[(2*m+2):dim(data_cper_sm)[1],] - data_cper_sm[1:(dim(data_cp
data_cper_sm = data_cper_sm/(2*m+1)

Q = NULL ;
for (q in 1:(N-1)) {
  for (p in (q+1):N) {
    Q_new1 = abs(data_per_sm[,2*p-1] * data_per_sm[,2*p]
                 - abs(data_cper_sm[,p])^2)
    Q_new2 = abs(data_per_sm[,2*q-1] * data_per_sm[,2*q]
                 - abs(data_cper_sm[,q])^2)
    Q_new12 = abs( (data_per_sm[,2*p-1] + data_per_sm[,2*q-1]) *
                     (data_per_sm[,2*p] + data_per_sm[,2*q]) -
                     abs(data_cper_sm[,p] + data_cper_sm[,q])^2)
    Q_new = (Q_new1 * Q_new2) / (Q_new12^2)
    Q_new = 2^(2*2)*Q_new
    Q = cbind(Q, Q_new)
  }
}
```

```
Q_st = Q ;

M = T/10 ;

par(mfrow=c(2,1))
plot(Q_st[,1],main="Q_XY(omega_l) for
     company MMM,AXP",ylab="Q_XY(omega)")
lines(Q_st[,1])
smallest1 <- sort(Q_st[,1],index.return=TRUE)$ix[1:M]
points(smallest1,Q_st[smallest1,1],col="orange")

plot(Q_st[,(2*N-2)],main="Q_XY(omega_l) for
     company AAPL,BA",ylab="Q_XY(omega)")
lines(Q_st[,(2*N-2)])
smallest2 <- sort(Q_st[,(2*N-2)],index.return=TRUE)$ix[1:M]
points(smallest2,Q_st[smallest2,(2*N-2)],col="orange")
```





Here are the plots of company MMM and AXP as well as company AAPL AND BA.

```
mean(smallest1)
```

```
## [1] 73.68
```

```
mean(smallest2)
```

```
## [1] 81.88
```

5 Bibliography

[1] Ravishanker, N., J. R. M. Hosking, and J. Mukhopadhyay. 2010. "Spectrum-Based Comparison of Stationary Multivariate Time Series." Methodology and Computing in Applied Probability 12 (4): 749–62.

[2] Glotzer, Dylan. *Multivariate VI: Time Series Clustering/Classification.* 2 Mar. 2017, www.stat.unc.edu/faculty/pipiras/timeseries/Multivariate_6_-*Classification_Clustering*-_Menu.html

[3] Esling P, Agon C (2012). "Time-Series Data Mining." *ACM Computing Surveys*, 45(1), 12:1–12:34.

[4] Giorgino, T., and others. 2009. "Computing and Visualizing Dynamic Time Warping Alignments in R: The Dtw Package." *Journal of Statistical Software* 31 (7): 1–24.

[5] Giorgino, T., and others. 2009. "Computing and Visualizing Dynamic Time Warping Alignments in R: The Dtw Package." *Journal of Statistical Software* 31 (7): 1–24.

[6] Giorgino, T., and others. 2009. "Computing and Visualizing Dynamic Time Warping Alignments in R: The Dtw Package." *Journal of Statistical Software* 31 (7): 1–24.

[7] Gollmer K, Posten C (1996). Supervision of Bioprocesses Using a Dynamic Time Warping Algorithm." *Control Engineering Practice*, 4(9), 1287-1295.

[8] Ravishanker, N., J. R. M. Hosking, and J. Mukhopadhyay. 2010. "Spectrum-Based Comparison of Stationary Multivariate Time Series." *Methodology and Computing in Applied Probability* 12 (4): 749–62.

[9] Ravishanker, N., J. R. M. Hosking, and J. Mukhopadhyay. 2010. "Spectrum-Based Comparison of Stationary Multivariate Time Series." *Methodology and Computing in Applied Probability* 12 (4): 749–62.

[10] Ravishanker, N., J. R. M. Hosking, and J. Mukhopadhyay. 2010. "Spectrum-Based Comparison of Stationary Multivariate Time Series." *Methodology and Computing in Applied Probability* 12 (4): 749–62.

[11] Ravishanker, N., J. R. M. Hosking, and J. Mukhopadhyay. 2010. "Spectrum-Based Comparison of Stationary Multivariate Time Series." *Methodology and Computing in Applied Probability* 12 (4): 749–62.

[12] Ravishanker, N., J. R. M. Hosking, and J. Mukhopadhyay. 2010. "Spectrum-Based Comparison of Stationary Multivariate Time Series." *Methodology and Computing in Applied Probability* 12 (4): 749–62.

[13] Sard´a-Espinosa, Alexis. "Comparing Time-Series Clustering Algorithms in R Using the Dtwclust Package.".Cran.r-Project.org.

[14] Liao, T. Warren. "Clustering of Time Series Data—a Survey.". *Pattern Recognition*, vol. 38, no. 11, 2005, pp. 1857–1874., doi:10.1016/j.patcog.2005.01.025.

[15] Sard´a-Espinosa, Alexis. "Comparing Time-Series Clustering Algorithms in R Using the Dtwclust Package.".*Cran.r-Project*.org.