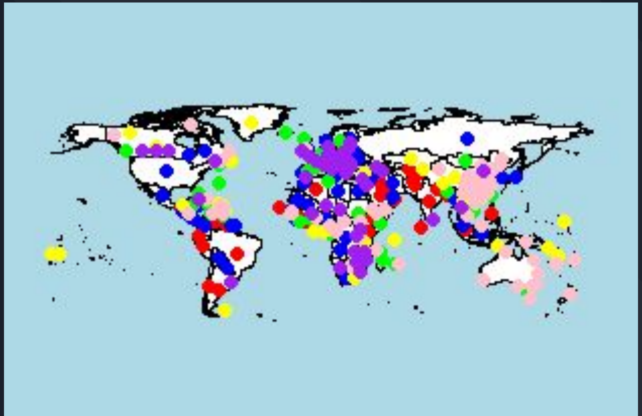


1091 Data Science - Group 9

Clustering Countries
By Their COVID-19 Time Series

AND
Plotting Covid-19 Time series (ggplot)

Choi Pui Shan 蔡佩珊(109753158)
Armando 王神鐸(107753048)

Table Of Contents

1. Motivations

- COVID-19 spread around the world
- Want to know if there are similar patterns between the time series of different countries
- But what we learnt from class are not for analyzing time series….
- We also do not familiar with neural networks...
- WHAT CAN WE DO??

1. Motivations

- A passage talking about how to cluster stocks by their time series in R for investment

Stock Clustering with Time Series Clustering in R

Yin-Ta Pan Aug 10, 2018 · 7 min read

IMPORTANT: THIS IS NOT INVESTMENT ADVICE.

As a newbie in stock market, the amount of choices available always prevents us from moving forward. It would be much easier if there is a tool can classify different stocks based on their historical stock price and then we can determine my investment strategy. For this purpose, time series clustering with `dtwclust` package in R is perfect. It can compare different stock prices and group them together, with few lines of R code.

1. Motivations

“Data visualizations make big and small data easier for the human brain to understand, and visualization also makes it easier to detect patterns, trends, and outliers in groups of data.”

=> We can also make some visualization of the COVID-19 Data!!

2. Goals

- Clustering Countries By Their Covid-19 Confirmed/Death/Recovered Time Series
- Plotting Covid-19 Data

- Ultimate Goal :
  To Understand Which Countries Have Similar Time Series Of Covid-19 Cases & Their Trends
  =>Make A Reference When Doing Business

3. Introduction of Time Series Clustering & Required R library

3. Introduction of Time Series Clustering

- **Clustering:**
  Unsupervised Learning to form groups of object with high similarity, where inter-groups have a high dissimilarity.

- **Time Series Clustering**
  A type of clustering algorithm made to handle dynamic data
  -Common Approaches:
   ->Hierarchical Clustering
   ->Partitional Clustering
   ->Fuzzy Clustering
  -Types:
   ->Shape-based
   ->Feature-based
   ->Model-based

2.5. Summary of distance measures

The distances described in this section are the ones implemented in **dtwclust**, which serve as basis for the algorithms presented in Section 3 and Section 4. Table 1 summarizes the salient characteristics of these distances.

Distance	Computational cost	Normalized	Symmetric	Multivariate support	Support for length differences
LB_Keogh	Low	No	No	No	No
LB_Improved	Low	No	No	No	No
DTW	Medium	Can be*	Can be*	Yes	Yes
GAK	High	Yes	Yes	Yes	Yes
Soft-DTW	High	Yes	Yes	Yes	Yes
SBD	Low	Yes	Yes	No	Yes

Table 1: Characteristics of time-series distance measures implemented in **dtwclust**. Regarding the cells marked with an asterisk: the DTW distance can be normalized for certain step patterns, and can be symmetric for symmetric step patterns when either no window constraints are used, or all time-series have the same length if constraints are indeed used.

- **Metrics**
  -Dynamic Time Warping(DTW) distance : Dissimilarity measure

4. Importing Data

	Province.State	Country.Region	Lat	Long	X1.22.20	X1.23.20	X1.24.20	X1.25.20	X1.26.20	X1.27.20
1	Afghanistan	Afghanistan	33.93911	67.709953	0	0	0	0	0	0
2	Albania	Albania	41.15330	20.168300	0	0	0	0	0	0
3	Algeria	Algeria	28.03390	1.659600	0	0	0	0	0	0
4	Andorra	Andorra	42.50630	1.521800	0	0	0	0	0	0
5	Angola	Angola	-11.20270	17.873900	0	0	0	0	0	0
6	Antigua and Barbuda	Antigua and Barbuda	17.06080	-61.796400	0	0	0	0	0	0
7	Argentina	Argentina	-38.41610	-63.616700	0	0	0	0	0	0
8	Armenia	Armenia	40.06910	45.038200	0	0	0	0	0	0
9	Australian Capital Territory	Australia	-35.47350	149.012400	0	0	0	0	0	0
10	New South Wales	Australia	-33.86880	151.209300	0	0	0	0	3	1
11	Northern Territory	Australia	-12.46340	130.845600	0	0	0	0	0	0
12	Queensland	Australia	-27.46980	153.025100	0	0	0	0	0	0
13	South Australia	Australia	-34.92850	138.600700	0	0	0	0	0	0
14	Tasmania	Australia	-42.88210	147.327200	0	0	0	0	0	0

```
##Importing Data from source(updated daily)
#---
Main <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covi

confirmed_Path <-  file.path(Main,"time_series_covid19_confirmed_global.csv")
Deaths_Path <- file.path(Main,"time_series_covid19_deaths_global.csv")
Recovered_Path <- file.path(Main,"time_series_covid19_recovered_global.csv")

#Read data from stored links:
ConfirmedData <- read.csv(confirmed_Path,stringsAsFactors = FALSE)
ConfirmedData<-as.data.frame(ConfirmedData)
DeathData<- read.csv(Deaths_Path,stringsAsFactors = FALSE)
DeathData<-as.data.frame(DeathData)
RecoveredData <- read.csv(Recovered_Path,stringsAsFactors = FALSE)
RecoveredData<-as.data.frame(RecoveredData)
#---
```

5. Preprocessing

```
##Data Proprocessing
#---
#Change the data from accumulated cases into daily change cases
for(i in 1:nrow(ConfirmedData)){
  ConfirmedData[i,6:ncol(ConfirmedData)]<-diff(as.numeric(ConfirmedData[i,5:ncol(ConfirmedData)]),1)
  #plot(as.numeric(ConfirmedData[1,3:ncol(ConfirmedData)]),type='l')
}

for(i in 1:nrow(DeathData)){
  DeathData[i,6:ncol(DeathData)]<-diff(as.numeric(DeathData[i,5:ncol(DeathData)]),1)
  #plot(as.numeric(DeathData[1,3:ncol(DeathData)]),type='l')
}

for(i in 1:nrow(RecoveredData)){
  RecoveredData[i,6:ncol(RecoveredData)]<-diff(as.numeric(RecoveredData[i,5:ncol(RecoveredData)]),1)
  #plot(as.numeric(RecoveredData[1,3:ncol(RecoveredData)]),type='l')
}

#Check if there is any NAs
#table(unique(is.na(ConfirmedData)))
#table(unique(is.na(DeathData)))
#table(unique(is.na(RecoveredData)))
```

5. Preprocessing

```r
#Found that 43th,53th row of ConfirmedData and Death Data missed the value for Lat and Long, and the n
#table(unique(is.na(ConfirmedData)))
#table(unique(is.na(DeathData)))

#But in the Recovered Data, there are Lat and Long for Canada, so take them to replace the missing val
ConfirmedData[c(43,53),3]<-RecoveredData[40,3]
ConfirmedData[c(43,53),4]<-RecoveredData[40,4]
DeathData[c(43,53),3]<-RecoveredData[40,3]
DeathData[c(43,53),4]<-RecoveredData[40,4]

#Found that 42th row of ConfirmedData and Death Data are strange, and required case is only 1
#as.numeric(ConfirmedData[42,5:ncol(ConfirmedData)])
#as.numeric(DeathData[42,5:ncol(ConfirmedData)])
#Remove 42th row of ConfirmedData and Death Data
ConfirmedData<-ConfirmedData[-42,]
DeathData<-DeathData[-42,]

#Check again
#table(unique(is.na(ConfirmedData)))
#table(unique(is.na(DeathData)))
#table(unique(is.na(RecoveredData)))
```

5. Preprocessing

```
#As some of the 1st column are empty, we insert the 2nd column into 1st column as the Province.State c
for(j in 1:nrow(ConfirmedData)){
  if(ConfirmedData[j,1]==""){
    ConfirmedData[j,1]<-ConfirmedData[j,2]
  }
}

for(j in 1:nrow(DeathData)){
  if(DeathData[j,1]==""){
    DeathData[j,1]<-DeathData[j,2]
  }
}

for(j in 1:nrow(RecoveredData)){
  if(RecoveredData[j,1]==""){
    RecoveredData[j,1]<-RecoveredData[j,2]
  }
}

#Check again
#ConfirmedData[,1]==""
#---

##Time Series Clustering for Confirmed Cases
```

5. Preprocessing

Filteringdata per country/region

```
# Data clean/ filter by country/region
DataClean <- function(data, region, CaseType) {
  CleanedData <- data %>%
    pivot_longer(cols = starts_with("X"),
                 names_to = "Date",
                 names_prefix = "X",
                 names_ptypes = list(week = integer()),
                 values_to = CaseType,
                 values_drop_na = TRUE)   %>%
    mutate(Province.State = ifelse(Province.State %in% "", Country.Region, Province.State)) %>%
    mutate(Date = as.Date(Date, "%m.%d.%y")) %>%
    filter(Province.State == region) %>%
    arrange(Date) %>%
    mutate(ID = row_number())
  return(CleanedData)
}

# Clean Data
ConData <- DataClean(ConfirmedData,Region,"Confirmed")
RecData <- DataClean(RecoveredData,Region,"Recovered") %>% select(ID,Recovered)
DeData <-DataClean(DeathData,Region,"Deaths")          %>% select(ID,Deaths)

# Merge cleaned data with ID column
AllData <- list(ConData, RecData, DeData)              %>% reduce(left_join, by = "ID")
```

6. Time Series Clustering for Confirmed Cases

```
##Time Series Clustering for Confirmed Cases
#---
#Hierarchical clustering of Time Series Clustering for Confirmed Cases
hc_dtw_Confirmed <- tsclust(as.ts(ConfirmedData[,5:ncol(ConfirmedData)]),type = "h",k = 20L,preproc = 

#From the dendrogram, we can basically divide them into 6 clusters
plot(hc_dtw_Confirmed)

#Average intra-cluster distance
tsclust(as.ts(ConfirmedData[,5:ncol(ConfirmedData)]), type = "h", k = 6L,
        preproc = zscore,
        seed = 899,
        distance = "dtw_basic",
        centroid = shape_extraction,
        control = hierarchical_control(method = "complete"), #complete = maximal intercluster dissimil
        args = tsclust_args(dist = list(window.size = 7L))) #for every 7 days

hc_dtw_Confirmed <- tsclust(as.ts(ConfirmedData[,5:ncol(ConfirmedData)]),type = "h",k = 6,preproc = zs
#Centroids Series Plot of Hierarchical clustering for Confirmed Cases
plot(hc_dtw_Confirmed, type = "centroids")
#plot(hc_dtw_Confirmed, type = "series", clus = 1L) #clus=1L : plot for the 1st cluster
#plot(hc_dtw_Confirmed, type = "series", clus = 2L)
#plot(hc_dtw_Confirmed, type = "series", clus = 3L)
#plot(hc_dtw_Confirmed, type = "series", clus = 4L)
#plot(hc_dtw_Confirmed, type = "series", clus = 5L)
```

6. Time Series Clustering for Confirmed Cases

```
print_clusters_Confirmed <- function(labels, k) {
  for(i in 1:k) {
    print(paste("cluster", i))
    print(ConfirmedData[labels==i,"Province.State"])
  }
}
groups_Confirmed <- cutree(hc_dtw_Confirmed, k=6)
#print_clusters_Confirmed(groups_Confirmed, 6)

lat_Confirmed_1=c()
long_Confirmed_1=c()
Confirmed_1<-ConfirmedData[groups_Confirmed==1,"Province.State"]
for(i in 1:nrow(ConfirmedData)){
  if(any(unique(ConfirmedData[i,1]==ConfirmedData[groups_Confirmed==1,"Province.State"])==TRUE)){
    lat_Confirmed_1[i]<-ConfirmedData[i,3]
    long_Confirmed_1[i]<-ConfirmedData[i,4]
  }
}
lat_Confirmed_1<-lat_Confirmed_1[-which(is.na(lat_Confirmed_1))]
long_Confirmed_1<-long_Confirmed_1[-which(is.na(long_Confirmed_1))]

lat_Confirmed_2=c()
long_Confirmed_2=c()
Confirmed_2<-ConfirmedData[groups_Confirmed==2,"Province.State"]
for(i in 1:nrow(ConfirmedData)){
```

6. Time Series Clustering for Confirmed Cases

```
lat_Confirmed_2=c()
long_Confirmed_2=c()
Confirmed_2<-ConfirmedData[groups_Confirmed==2,"Province.State"]
for(i in 1:nrow(ConfirmedData)){
  if(any(unique(ConfirmedData[i,1]==ConfirmedData[groups_Confirmed==2,"Province.State"])==TRUE)){
    lat_Confirmed_2[i]<-ConfirmedData[i,3]
    long_Confirmed_2[i]<-ConfirmedData[i,4]
  }
}
lat_Confirmed_2<-lat_Confirmed_2[-which(is.na(lat_Confirmed_2))]
long_Confirmed_2<-long_Confirmed_2[-which(is.na(long_Confirmed_2))]

lat_Confirmed_3=c()
long_Confirmed_3=c()
Confirmed_3<-ConfirmedData[groups_Confirmed==3,"Province.State"]
for(i in 1:nrow(ConfirmedData)){
  if(any(unique(ConfirmedData[i,1]==ConfirmedData[groups_Confirmed==3,"Province.State"])==TRUE)){
    lat_Confirmed_3[i]<-ConfirmedData[i,3]
    long_Confirmed_3[i]<-ConfirmedData[i,4]
  }
}
lat_Confirmed_3<-lat_Confirmed_3[-which(is.na(lat_Confirmed_3))]
long_Confirmed_3<-long_Confirmed_3[-which(is.na(long_Confirmed_3))]
```

6. Time Series Clustering for Confirmed Cases

```
lat_Confirmed_4=c()
long_Confirmed_4=c()
Confirmed_4<-ConfirmedData[groups_Confirmed==4,"Province.State"]
for(i in 1:nrow(ConfirmedData)){
  if(any(unique(ConfirmedData[i,1]==ConfirmedData[groups_Confirmed==4,"Province.State"])==TRUE)){
    lat_Confirmed_4[i]<-ConfirmedData[i,3]
    long_Confirmed_4[i]<-ConfirmedData[i,4]
  }
}
lat_Confirmed_4<-lat_Confirmed_4[-which(is.na(lat_Confirmed_4))]
long_Confirmed_4<-long_Confirmed_4[-which(is.na(long_Confirmed_4))]

lat_Confirmed_5=c()
long_Confirmed_5=c()
Confirmed_5<-ConfirmedData[groups_Confirmed==5,"Province.State"]
for(i in 1:nrow(ConfirmedData)){
  if(any(unique(ConfirmedData[i,1]==ConfirmedData[groups_Confirmed==5,"Province.State"])==TRUE)){
    lat_Confirmed_5[i]<-ConfirmedData[i,3]
    long_Confirmed_5[i]<-ConfirmedData[i,4]
  }
}
lat_Confirmed_5<-lat_Confirmed_5[-which(is.na(lat_Confirmed_5))]
long_Confirmed_5<-long_Confirmed_5[-which(is.na(long_Confirmed_5))]

lat_Confirmed_6=c()
long_Confirmed_6=c()
Confirmed_6<-ConfirmedData[groups_Confirmed==6,"Province.State"]
for(i in 1:nrow(ConfirmedData)){
  if(any(unique(ConfirmedData[i,1]==ConfirmedData[groups_Confirmed==6,"Province.State"])==TRUE)){
    lat_Confirmed_6[i]<-ConfirmedData[i,3]
    long_Confirmed_6[i]<-ConfirmedData[i,4]
```

6. Time Series Clustering for Confirmed Cases

```
#Plot the Hierarchical clustering Result of Confirmed Cases on World Map
map("world", fill=TRUE, col="white", bg="lightblue", ylim=c(-60, 90), mar=c(0,0,0,0))
points(long_Confirmed_1,lat_Confirmed_1, col="red", pch=16)
points(long_Confirmed_2,lat_Confirmed_2, col="blue", pch=16)
points(long_Confirmed_3,lat_Confirmed_3, col="green", pch=16)
points(long_Confirmed_4,lat_Confirmed_4, col="yellow", pch=16)
points(long_Confirmed_5,lat_Confirmed_5, col="pink", pch=16)
points(long_Confirmed_6,lat_Confirmed_6, col="purple", pch=16)

#PCA to visualize the cluster
library(ggplot2)
pca_Confirmed <- prcomp(ConfirmedData[,5:ncol(ConfirmedData)])
NumberofPC_Confirmed <- 6
Projection_Confirmed <- predict(pca_Confirmed, newdata=ConfirmedData[,5:ncol(ConfirmedData)])[,1:Numbe
project.plus_Confirmed <- cbind(as.data.frame(Projection_Confirmed),cluster=as.factor(groups_Confirmed
ggplot(project.plus_Confirmed, aes(x=PC1, y=PC2))+geom_point(aes(shape=cluster))+geom_text(aes(label=s
ggplot(project.plus_Confirmed, aes(x=PC3, y=PC4))+geom_point(aes(shape=cluster))+geom_text(aes(label=s
ggplot(project.plus_Confirmed, aes(x=PC5, y=PC6))+geom_point(aes(shape=cluster))+geom_text(aes(label=s

for(i in 1:6){
  write.csv(as.data.frame(ConfirmedData[groups_Confirmed==i,"Province.State"]),paste0("Confirmed_Clust
}

#---
```

6. Time Series Clustering for Confirmed Cases

```
> tsclust(as.ts(ConfirmedData[,5:ncol(ConfirmedData)]), type = "h", k = 6L,
+         preproc = zscore,
+         seed = 899,
+         distance = "dtw_basic",
+         centroid = shape_extraction,
+         control = hierarchical_control(method = "complete"), #complete = maximal intercluster dissimilarity
+         args = tsclust_args(dist = list(window.size = 7L))) #for every 7 days
hierarchical clustering with 6 clusters
Using dtw_basic distance
Using shape_extraction centroids
Using method complete
Using zscore preprocessing

Time required for analysis:
   user  system elapsed
  34.00    3.82   16.00

Cluster sizes with average intra-cluster distance:

  size  av_dist
1   25 253.6877
2   49 133.6907
3   17 225.3176
4  137 345.7411
5   31 236.9008
6   12 198.6177
>
```

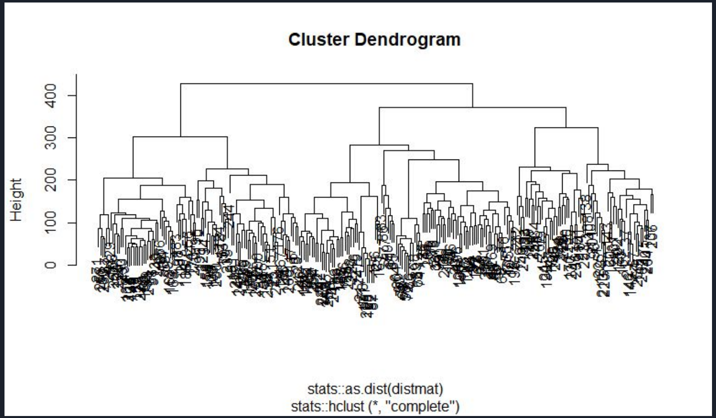
7. Time Series Clustering for Death Cases

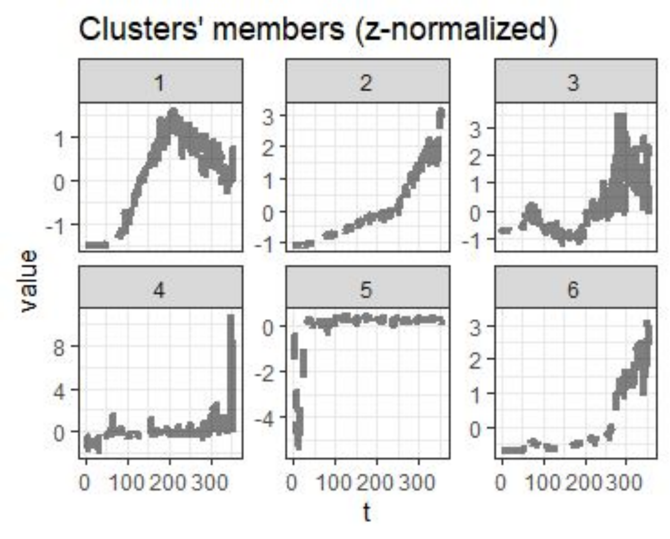
```
> tsclust(as.ts(DeathData[,5:ncol(DeathData)]), type = "h", k = 4L,
+         preproc = zscore,
+         seed = 899,
+         distance = "dtw_basic",
+         centroid = shape_extraction,
+         control = hierarchical_control(method = "complete"),
+         args = tsclust_args(dist = list(window.size = 7L))) #]for every 7 days
hierarchical clustering with 4 clusters
Using dtw_basic distance
Using shape_extraction centroids
Using method complete
Using zscore preprocessing

Time required for analysis:
   user  system elapsed
  55.58    1.48   13.42

Cluster sizes with average intra-cluster distance:

  size  av_dist
1  190 365.9079
2   47 198.4781
3   11 184.7827
4   23 187.9642
> |
```

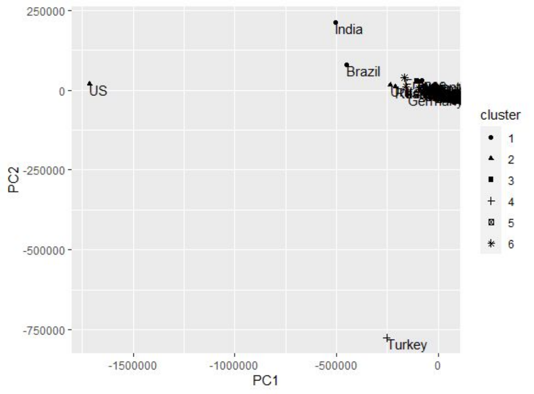
8. Time Series Clustering for Recovered Cases

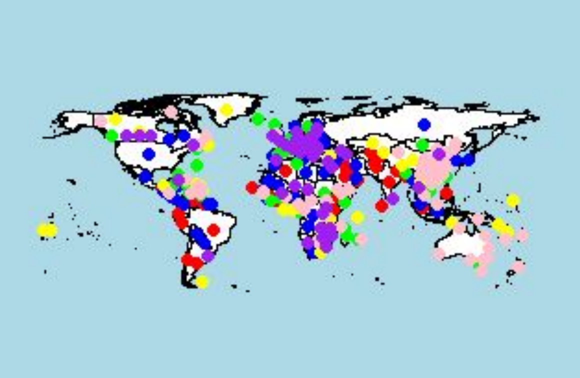
```
> tsclust(as.ts(RecoveredData[,5:ncol(RecoveredData)]), type = "h", k = 6L,
+         preproc = zscore,
+         seed = 899,
+         distance = "dtw_basic",
+         centroid = shape_extraction,
+         control = hierarchical_control(method = "complete"), #complete = maximal intercluster dissimilarity
+         args = tsclust_args(dist = list(window.size = 7L))) #for every 7 days
hierarchical clustering with 6 clusters
Using dtw_basic distance
Using shape_extraction centroids
Using method complete
Using zscore preprocessing

Time required for analysis:
   user  system elapsed
  35.61    0.88    8.36

Cluster sizes with average intra-cluster distance:

  size   av_dist
1  152 409.86688
2   52 181.69347
3   19 258.11332
4   26 250.25676
5    6 200.59672
6    2  95.04436
```

9. Plotting Covid-19 Data

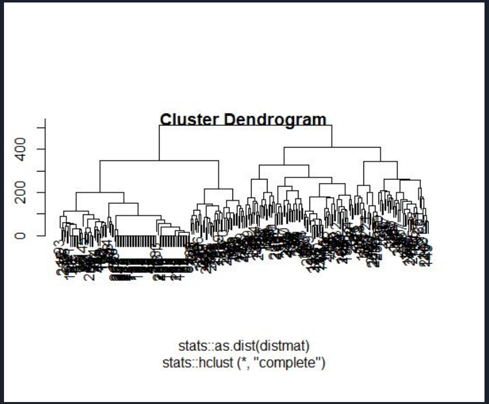


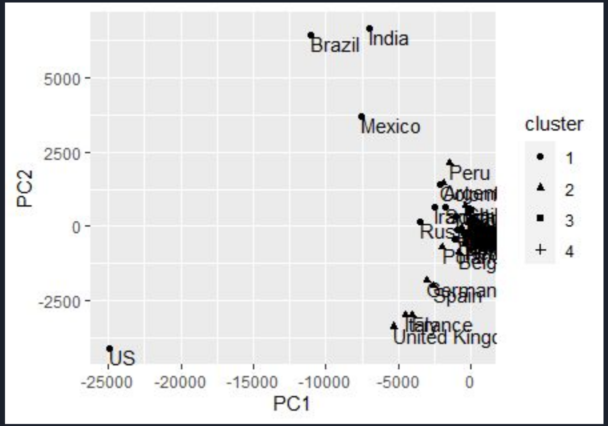


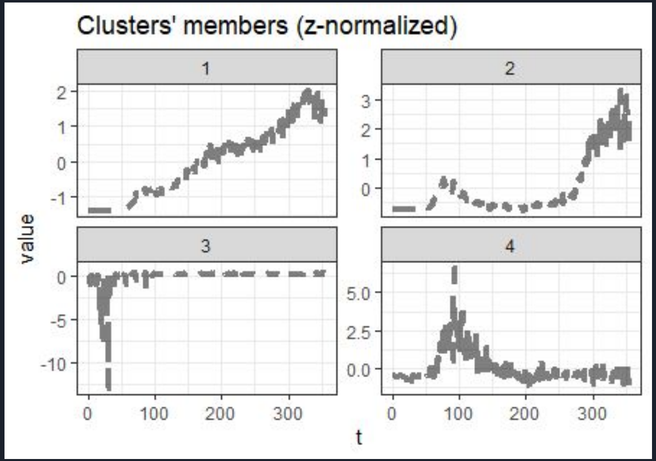


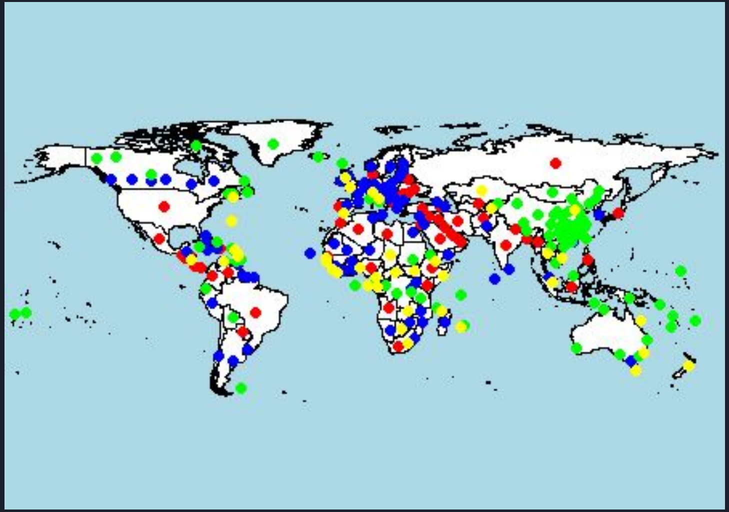


9. Plotting Covid-19 Data

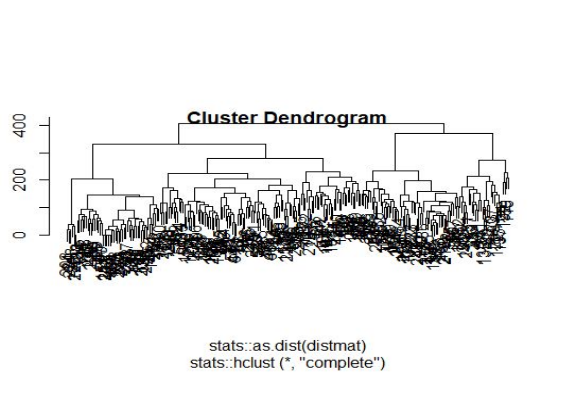


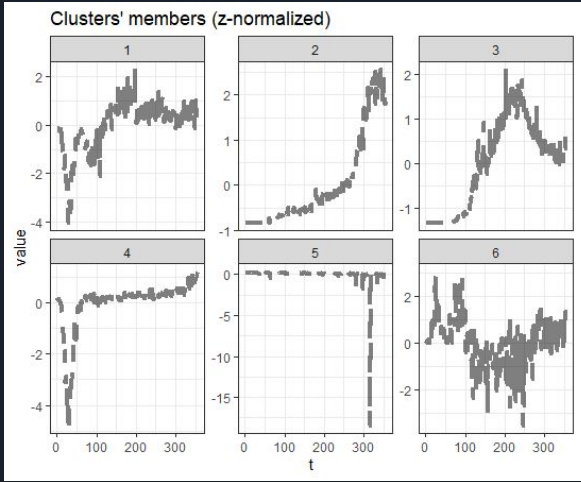


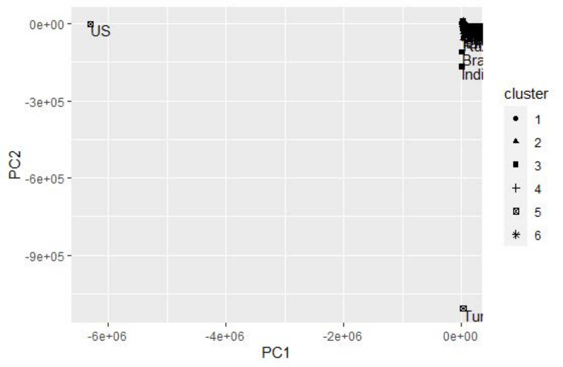


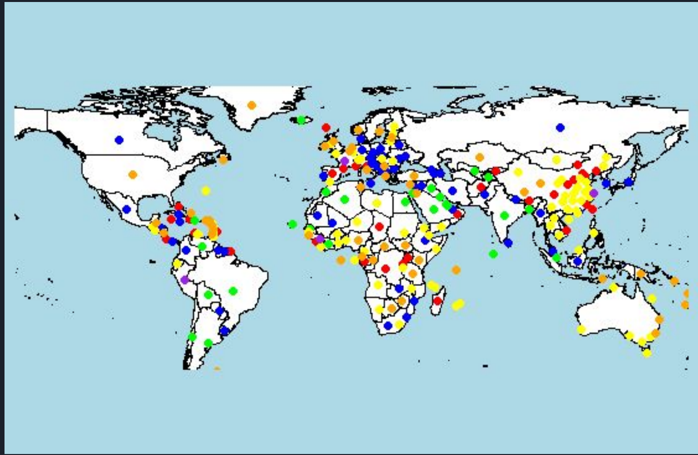


9. Plotting Covid-19 Data









What’s next?

After being able to better understand Covid-19 data by visualizing methods, we speculate that future applications and research can be done by trying do find correlations in other areas that may characterize causes and/or consequences of number of cases in a particular region. Examples of these may be the implementation of mandatory mask use (cause) or an impact in a nation’s economy (consequence).

References

https://cran.r-project.org/web/packages/dtwclust/dtwclust.pdf

https://www.rdocumentation.org/packages/dtwclust/versions/3.1.1/topics/tsclust

https://www.r-bloggers.com/2013/04/r-beginners-plotting-locations-on-to-a-world-map/

Codes Examples From This course