Compulsary assignment 1

Charlott Kjærre Olofsson, Ole Benjamin Gauslaa, Min Jeong Cheon, Martin Bergsholm Nesse

2022-10-06

Exercise 1

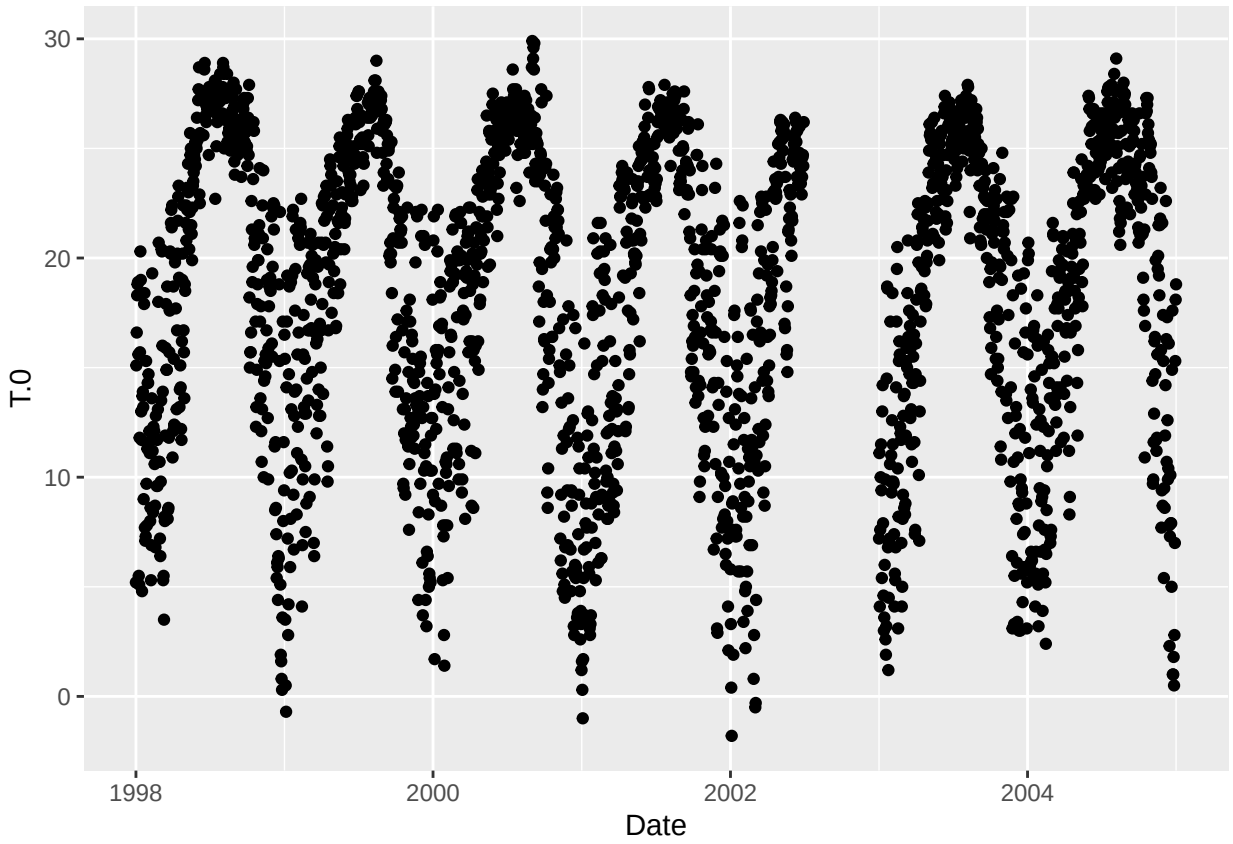
a)

```
# Load the data
library(readr)
ozone <- read_csv("/Users/martinbergsholmnesse/Documents/NMBU/DAT320/ozone.csv")
```

```
# Delete columns
ozone = subset(ozone, select = -c(WSR_PK, WSR_AV, T_PK, T_AV, T85, RH85, U85, V85,
                                  HT85, T70, RH70, U70, V70, HT70, T50, RH50, U50,
                                  V50, HT50, KI, TT, SLP, SLP_, Precp, response))
```

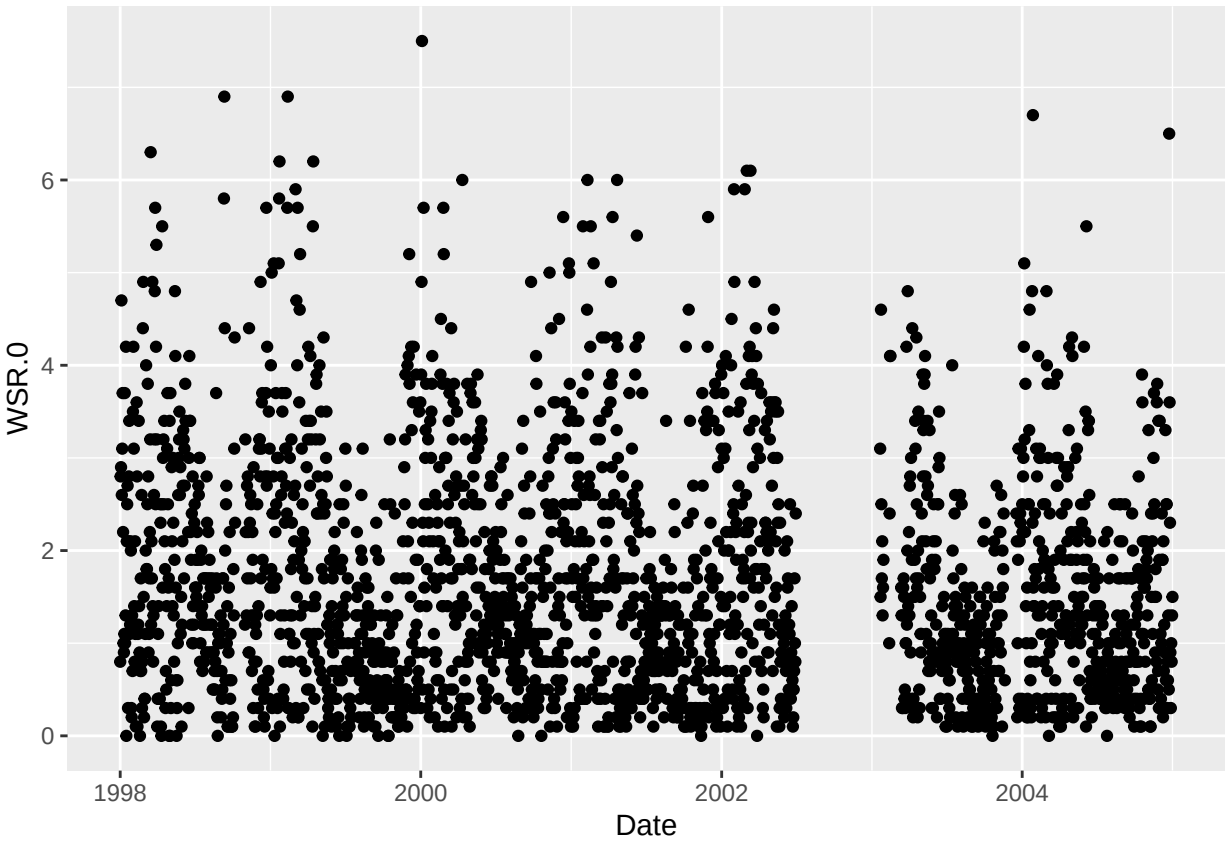
```
# Convert Date to Dato-format
library(dplyr)
library(magrittr)
mutate(ozone, Date= as.Date(Date, format= "%d.%m.%Y"))
```

```
# # Plot T.0
library(ggplot2)

ggplot(data = ozone ,
       aes(x = Date , y =
             T.0)) + geom_point()
```



We see that the average temperature between 00:00 and 00:59 varies from a few negative degrees to over 30. There are clearly seasonal changes in the temperature. It does not seem to be a trend in the temperature.

```
# Plot WSR.0
ggplot(data = ozone ,
       aes(x = Date , y =
             WSR.0)) + geom_point()
```



Also in the wind speed we see some seasonality. The yearly seasonality is not quite as distinct as it was with the temperature. In the wind speed plot we see that the peaks usually appear in the beginning of the year, but in the temperature plot the peaks are of course in the summer. As with the temperature, it is no clear trend in the data.

b)

```
# Missing values
sum(is.na(ozone))
```

```
## [1] 11458
```

The number of missing values is 11458.

```
library(tidyr)

ozone = ozone %>%
  mutate(Date = as.Date(Date)) %>%
  complete(Date = seq.Date(min(Date), max(Date), by="day"))
```

```
# Remove leap year dates
ozone = ozone[!(format(ozone$Date,"%m") == "02" & format(ozone$Date, "%d") == "29"),
              ,drop = FALSE]

# Check if number of rows is correct. It is.
nrow(ozone)
```

```
## [1] 2555
```

We see that the number of observations are now 2555, which means that we successfully removed the leap year dates. 2555 / 365 = 7.

c)

```
ozone = as.data.frame(ozone)

ozone_long = reshape(ozone, varying = list(grepl(colnames(ozone), pattern = "T"),
                                grepl(colnames(ozone), pattern = "WSR")),
        direction = "long",
        v.names = c("T", "WSR"),
        timevar = "Time",
        times = 0:23) %>%
  arrange(Date, Time)

library(lubridate)

ozone_long = mutate(ozone_long, DateTime = ymd_h(paste(Date, Time)))

head(ozone_long)
```

```
##           Date Time   T WSR id            DateTime
## 1.0 1998-01-01    0 5.2 0.8  1 1998-01-01 00:00:00
## 1.1 1998-01-01    1 6.1 1.8  1 1998-01-01 01:00:00
## 1.2 1998-01-01    2 6.1 2.4  1 1998-01-01 02:00:00
## 1.3 1998-01-01    3 6.1 2.1  1 1998-01-01 03:00:00
## 1.4 1998-01-01    4 6.1 2.0  1 1998-01-01 04:00:00
## 1.5 1998-01-01    5 5.6 2.1  1 1998-01-01 05:00:00
```

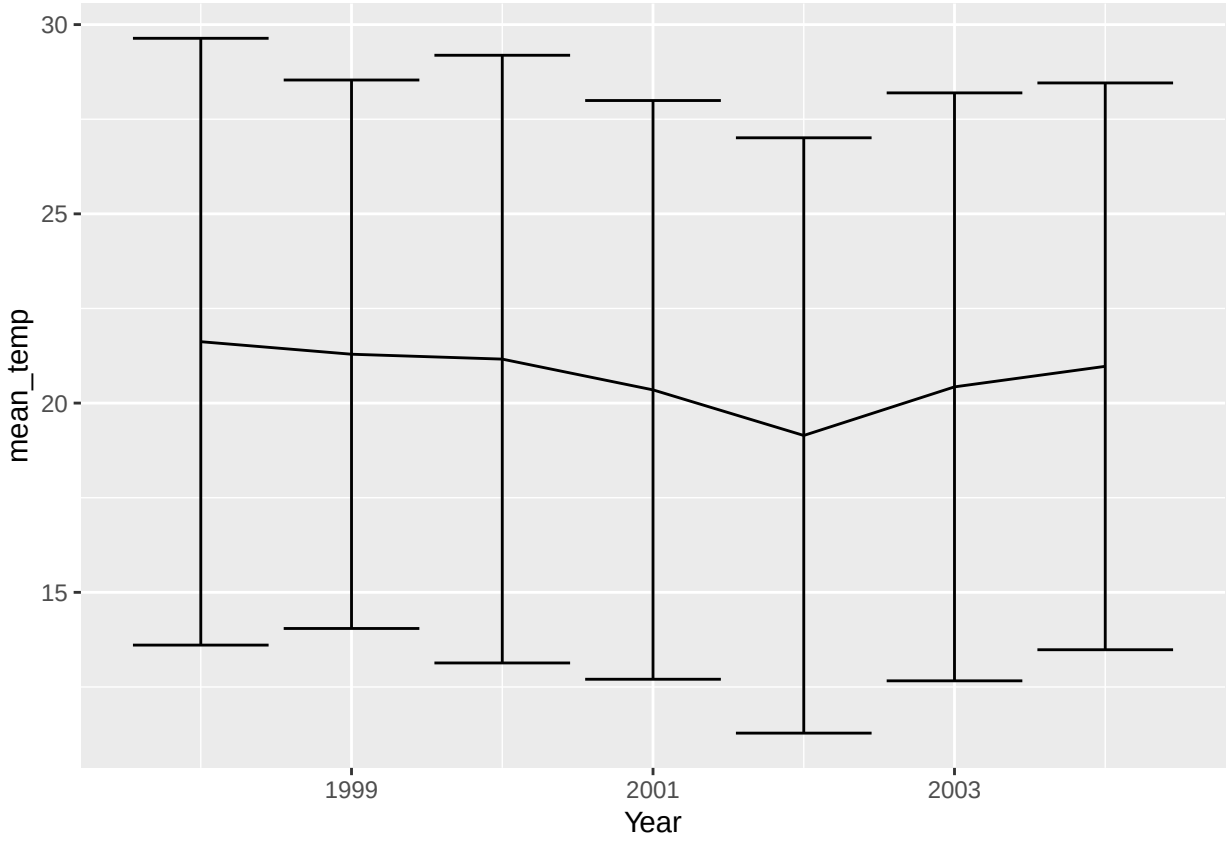
We see that the data is converted from wide to long format.

```
# Yearly average temperatures
yat = ozone_long %>%
  group_by(year(Date)) %>%
  summarize(mean_temp = mean(T, na.rm = TRUE),
            sd_temp = sd(T, na.rm = TRUE))


# Sett som data frame
yat = as.data.frame(yat)

# Endre navnet til noe som ikke inkluderer den
# innebygde funksjonen "year"...
names(yat)[names(yat) == 'year(Date)'] <- 'Year'

# Sjekk at navn er endret
colnames(yat)
```

```
## [1] "Year"      "mean_temp" "sd_temp"
```

```
# Plot of average temperature and standard deviation

ggplot(data = yat,
       mapping = aes(x = Year ,
                     y = mean_temp)) +
```

```
  geom_errorbar(aes(ymin = mean_temp - sd_temp,
                    ymax = mean_temp + sd_temp)) +
  geom_line()
```



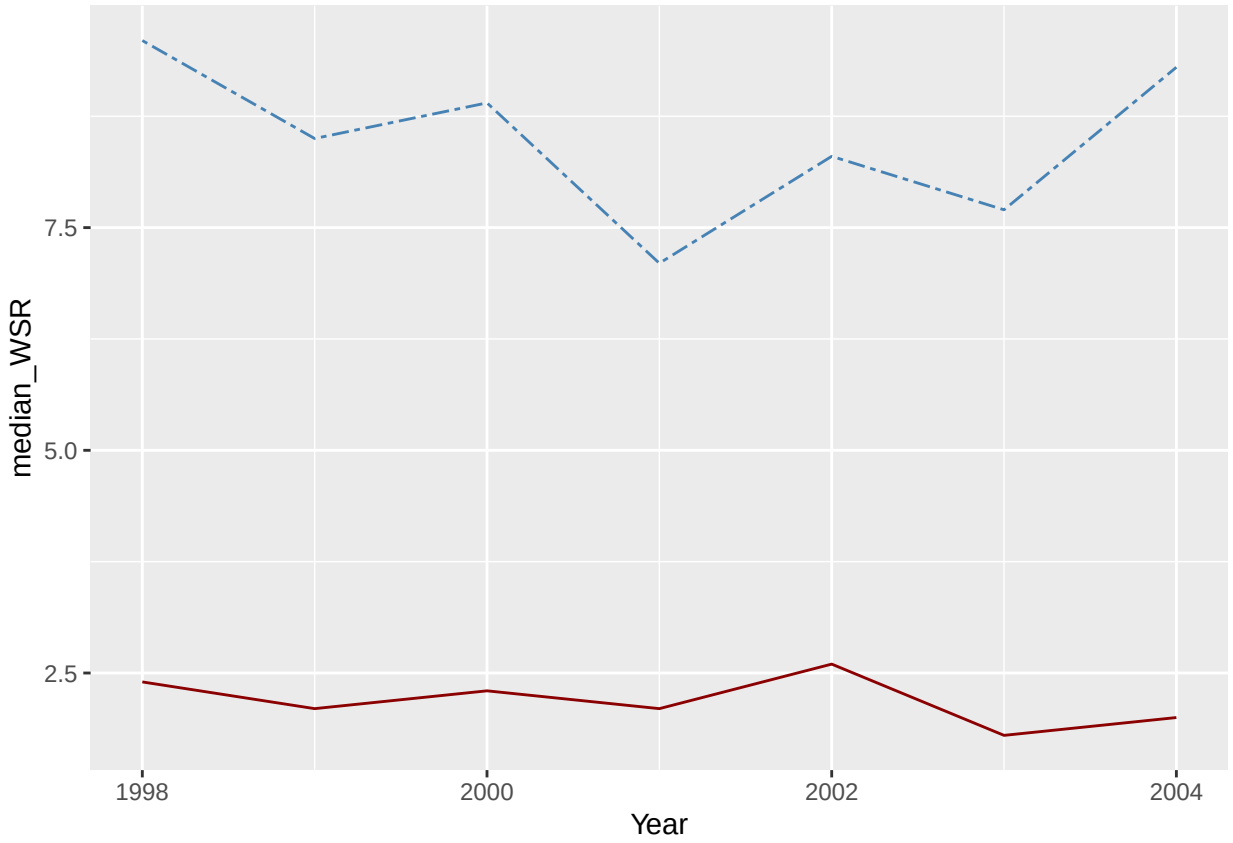
The yearly average temperature reached a minimum level in 2002.

```
# Yearly median wind
ymw = ozone_long %>%
  group_by(year(Date)) %>%
  summarize(median_WSR = median(WSR, na.rm = TRUE),
            max_wind = max(WSR, na.rm = TRUE),
            min_wind = min(WSR, na.rm = TRUE))

# Sett som data frame
ymw = as.data.frame(ymw)

# Endre navnet til noe som ikke inkluderer den
# innebygde funksjonen "year"...
names(ymw)[names(ymw) == 'year(Date)'] <- 'Year'

# Sjekk at navn er endret
colnames(ymw)
```

```
## [1] "Year"       "median_WSR" "max_wind"   "min_wind"
```

```
# Plot it!
ggplot(ymw, aes(x=Year)) +
  geom_line(aes(y = median_WSR), color = "darkred") +
```

```
  geom_line(aes(y = max_wind), color="steelblue", linetype="twodash")
```

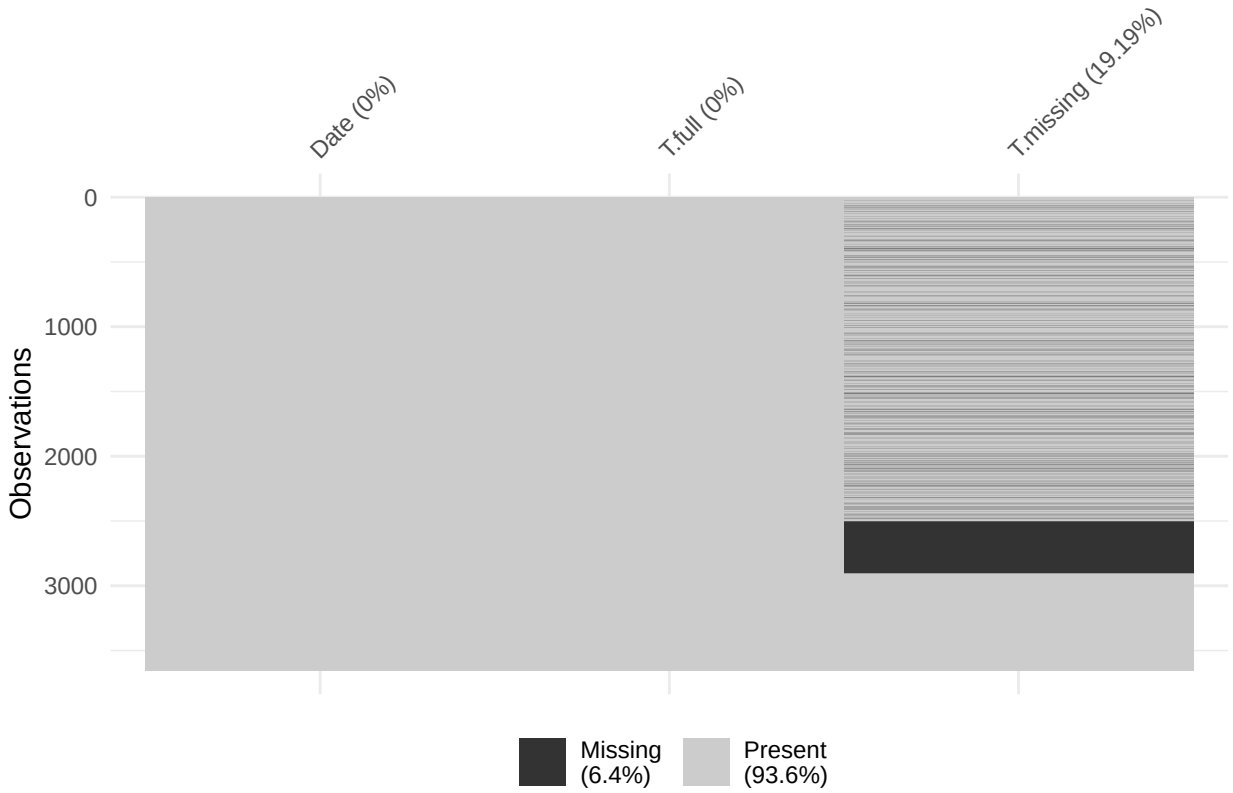


The dotted line is the maximum wind speed and the red line below is the mean wind speed. The minimum wind speed is zero for all the years.

Exercise 2

a)

```
data <- read_csv("/Users/martinbergsholmnesse/Documents/NMBU/DAT320/temperature.csv")

#install.packages("naniar")
library(naniar)
vis_miss(data)
```



The plot shows the placement of missing values. Total 701 NA values Longest list of consecutive NA are 401

```
sum_of_na = sum(is.na(data$T.missing))
list_consecutive_na = c(0)
consecutive_na = 0
for (i in 1:length(data[,1])) {
  if (is.na(data[i, 3])) {
    consecutive_na = consecutive_na + 1
    if(!is.na(data[i + 1, 3])) {
      list_consecutive_na = append(list_consecutive_na, c(consecutive_na))
      consecutive_na = 0
    }
  }
}

print(list_consecutive_na)
```

```
## [1] 0
```

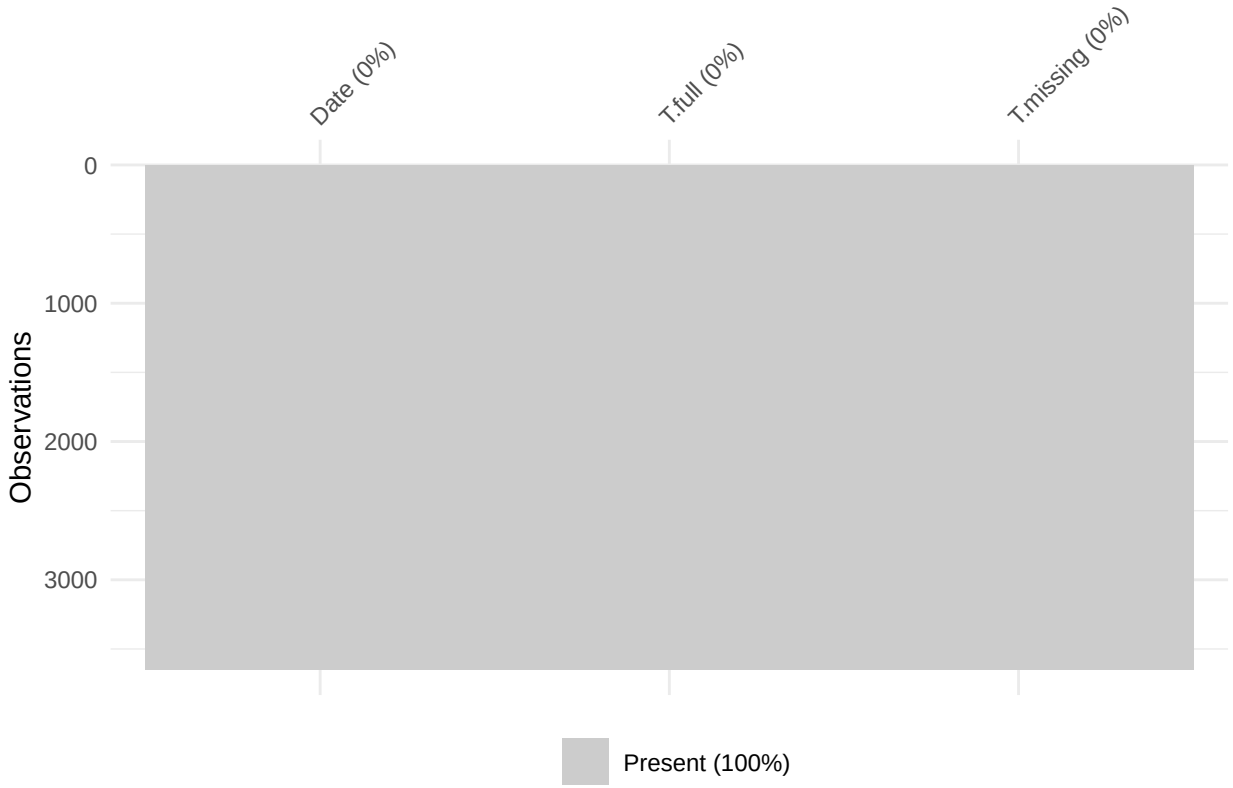
```
# Remove type caste date remove 29 of february
max(list_consecutive_na)
```

```
## [1] 0
```

```
library(lubridate)
leap = function(x){
  +   day(x) == 29 & month(x) == 2
}

data$Date <- as.Date(data$Date)
data = data[!leap(data$Date), ]
```

b)

```
# (B)
# Impute missing values by local and global mean.

library(imputeTS)
library(dplyr)
gobal_solution_data <- data
gobal_solution_data$T.missing <- na_mean(gobal_solution_data$T.missing)

locf_solution_data <- data
locf_solution_data$T.missing <- na_locf(locf_solution_data$T.missing)
vis_miss(locf_solution_data)
```

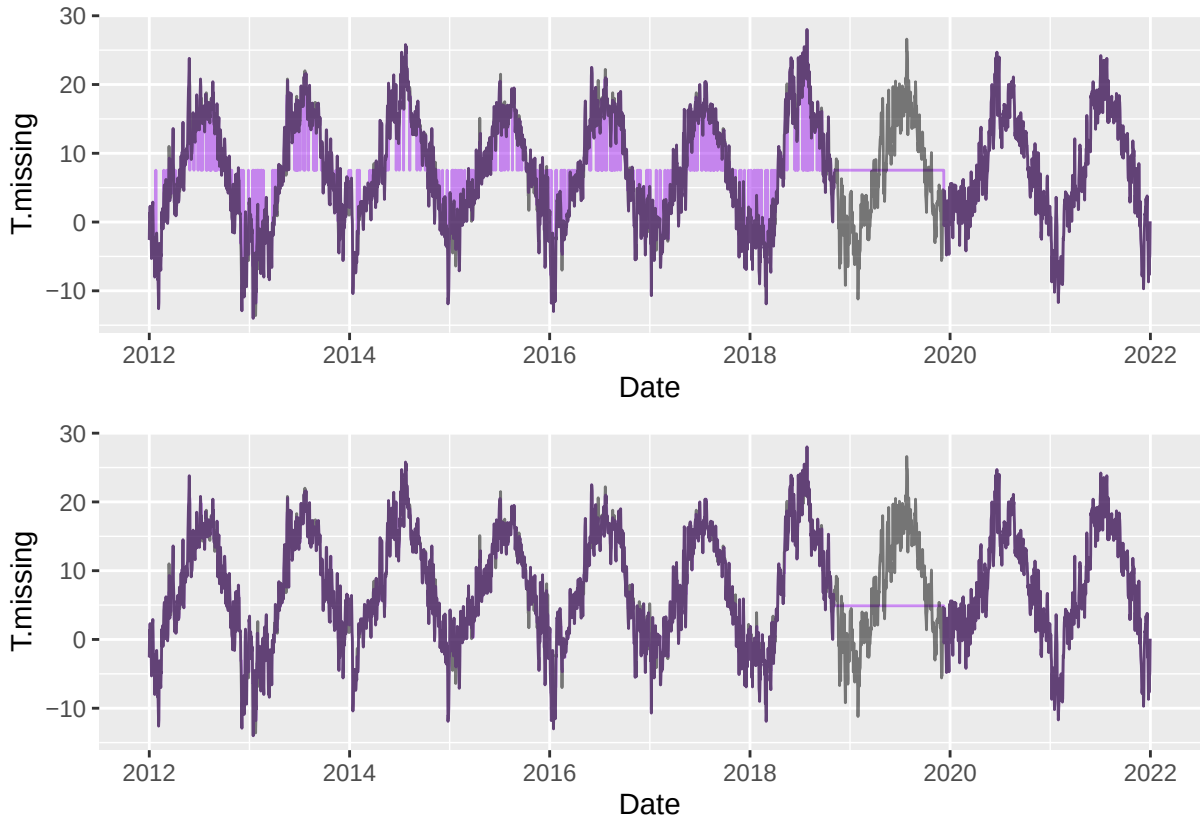



```
colnames(data)
```

```
## [1] "Date"      "T.full"    "T.missing"
```

```
library(ggplot2)
library(patchwork)
p1 <- ggplot(gobal_solution_data, aes(x=Date,
                                      y=T.missing)) +
  geom_line(color="purple", alpha=0.5) +
  geom_line(aes(x=Date, y=T.full), alpha=0.5)

p2 <- ggplot(locf_solution_data, aes(x=Date,
                                     y=T.missing)) +
  geom_line(color="purple", alpha=0.5) +
  geom_line(aes(x=Date, y=T.full), alpha=0.5)
```

```
p1 / p2
```




```
# RMSE
# By visualization and MSE we can see that local replacement is more acurate then global.
global_MSE <- mean((gobal_solution_data$T.full - gobal_solution_data$T.missing)^2)
locf_MSE <- mean((locf_solution_data$T.full - locf_solution_data$T.missing)^2)

global_MSE
```

```
## [1] 11.6178
```

```
locf_MSE
```

```
## [1] 7.067693
```

(C):

There are some clear problem when we interpolation the larger group of values. 1. most of the methods are noe suited to cover longer segments of missing values. 2. spline is prone to error caused by the direction from the two segments which we try to connect. Therefor spline perform the worst.

Result: By comparing plots and MSE there are one solution that perform as we want, and that is seasonal imputation by the function: na_seasplit. by trial and error we find out that the function is not suited to find the right seasonality, this can be fixed by transforming the feature vector into a time series object and specify the right FQ for the series.

```
library(imputeTS)

# Linear imputation
```

```r
LI_data <- data
LI_data$T.missing <- na_interpolation(LI_data$T.missing, option = "linear")

p1 <- ggplot(LI_data, aes(x=Date,
                          y=T.missing)) +
  geom_line(color="purple", alpha=0.5) +
  geom_line(aes(x=Date, y=T.full), alpha=0.5)

linear_MSE <- mean((LI_data$T.full - LI_data$T.missing)^2)
linear_MSE
```

```
## [1] 10.12044
```

```r
# Spinel imputation

SPINEL_data <- data
SPINEL_data$T.missing <- na_interpolation(SPINEL_data$T.missing, option = "spline")

p2 <- ggplot(SPINEL_data, aes(x=Date,
                              y=T.missing)) +
  geom_line(color="purple", alpha=0.5) +
  geom_line(aes(x=Date, y=T.full), alpha=0.5)

SPINEL_MSE <- mean((SPINEL_data$T.full - SPINEL_data$T.missing)^2)
SPINEL_MSE
```

```
## [1] 1077.645
```

```r
# Stine imputation
Stine_data <- data
Stine_data$T.missing <- na_interpolation(Stine_data$T.missing, option = "stine")

p3<- ggplot(Stine_data, aes(x=Date,
                            y=T.missing)) +
  geom_line(color="purple", alpha=0.5) +
  geom_line(aes(x=Date, y=T.full), alpha=0.5)

STINE_MSE <- mean((Stine_data$T.full - Stine_data$T.missing)^2)
STINE_MSE
```

```
## [1] 25.13848
```

```r
# kalman
kalman_data <- data
kalman_data$T.missing <- na_kalman(kalman_data$T.missing)

p4 <- ggplot(kalman_data, aes(x=Date, y=T.missing)) +
  geom_line(color="purple", alpha=0.5) +
  geom_line(aes(x=Date, y=T.full), alpha=0.5)

KALMAN_MSE <- mean((kalman_data$T.full - kalman_data$T.missing)^2)
KALMAN_MSE
```

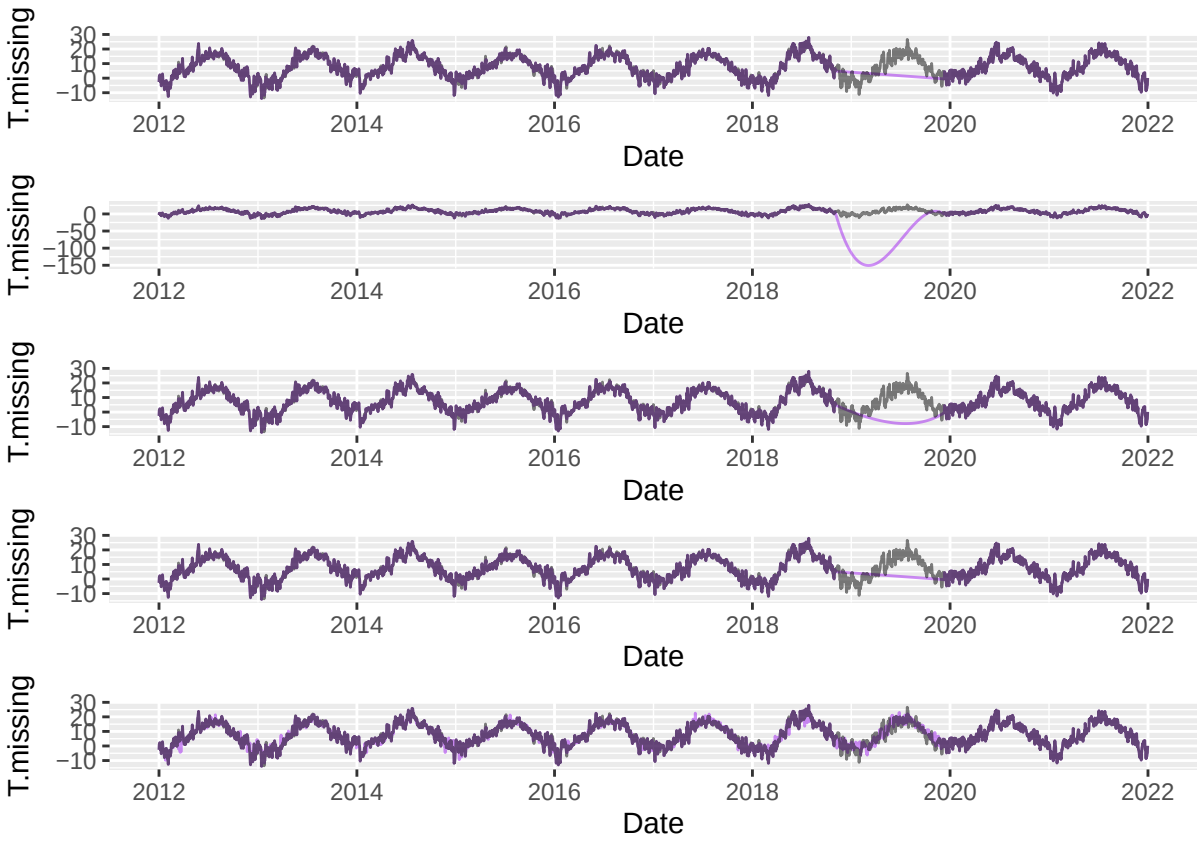
```
## [1] 10.02509
```

```r
# seasplit interpolation
seasplit_data <- data
```

```
seasplit_data$T.missing <- ts(seasplit_data$T.missing, start=2012, frequency = 365)
seasplit_data$T.missing <- na_seasplit(seasplit_data$T.missing,
                                       algorithm = "interpolation",
                                       find_frequency = FALSE)

p5 <- ggplot(seasplit_data,
             aes(x=Date, y=T.missing)) +
  geom_line(color="purple", alpha=0.5) +
  geom_line(aes(x=Date, y=T.full), alpha=0.5)

SEASSPLIT_MSE <- mean((seasplit_data$T.full - seasplit_data$T.missing)^2)
SEASSPLIT_MSE
```

```
## [1] 3.282046
```

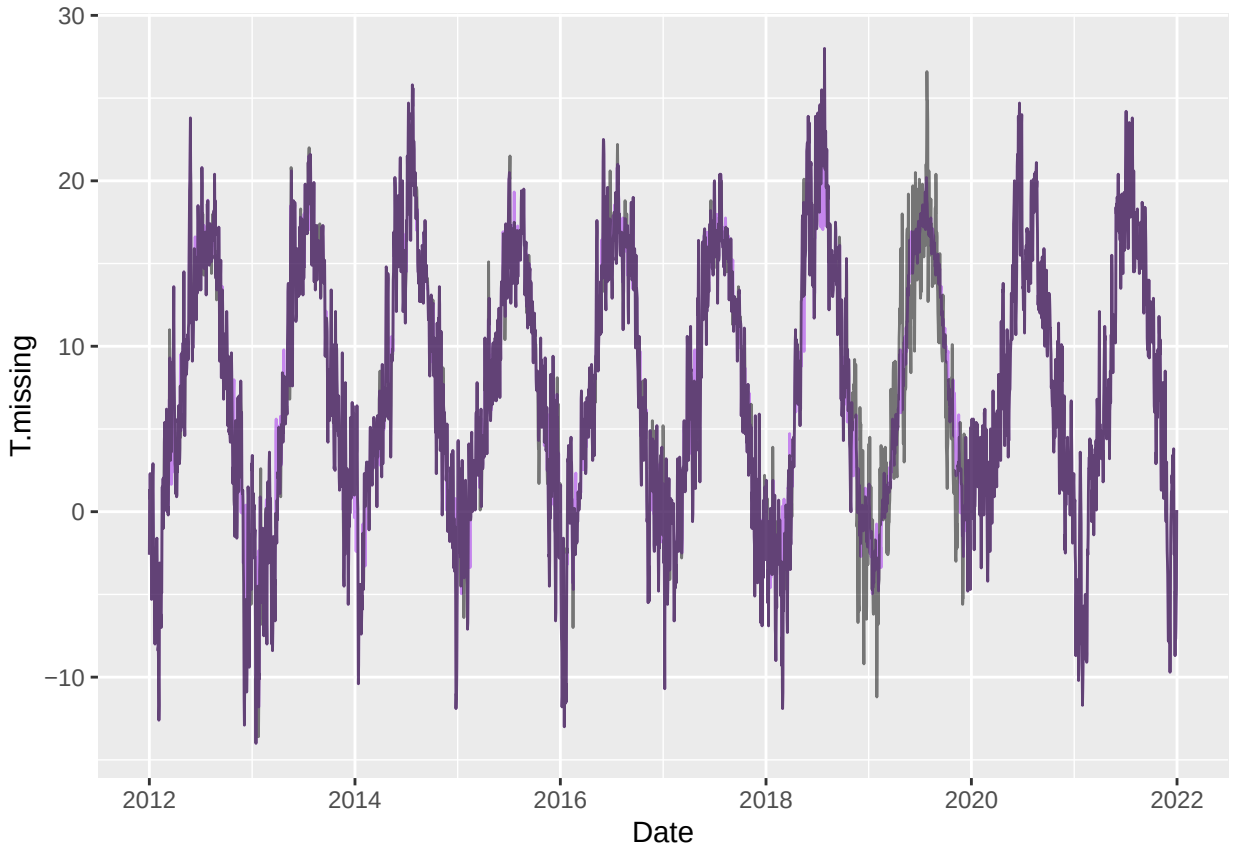
```
p1 / p2 / p3 /p4 /p5
```



d)

```
manuel_data <- data

manuel_data$month <- as.integer(format(manuel_data$Date, "%m"))
manuel_data$day <- as.integer(format(manuel_data$Date, "%d"))

for (val in 1:31)
{
```

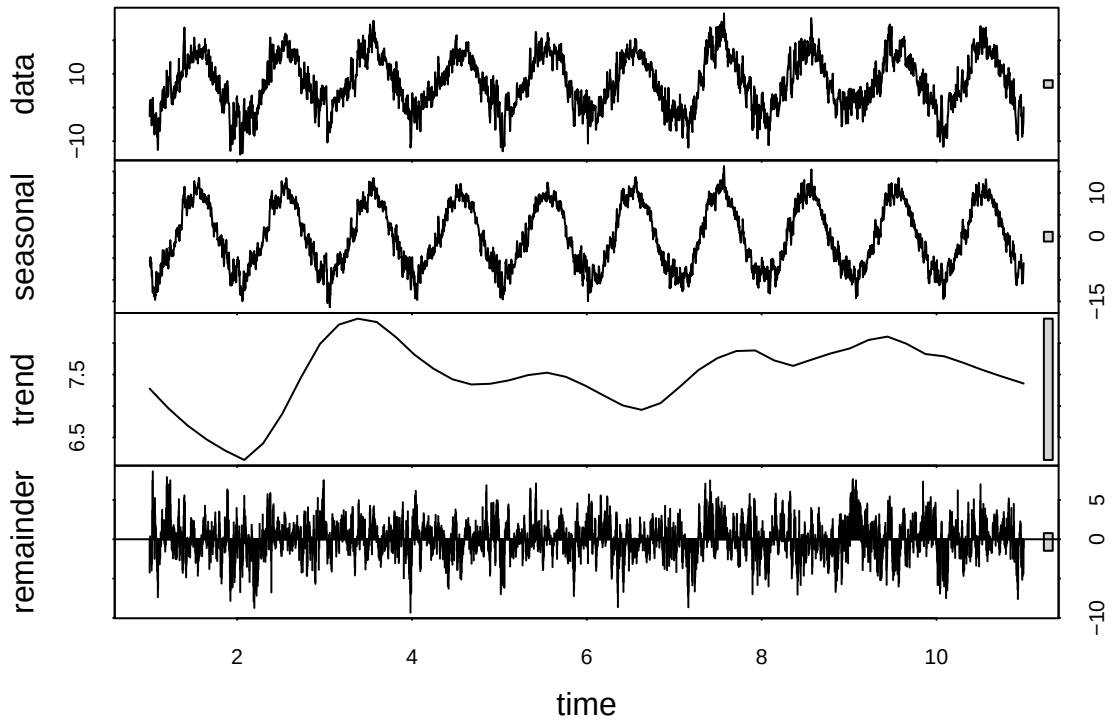
```
  for (val2 in 1:12)
  {
    manuel_data$T.missing[manuel_data$day == val & manuel_data$month == val2] <- na_mean(manuel_data$T.
  }
}

p_manuel <- ggplot(manuel_data, aes(x=Date, y=T.missing)) +
  geom_line(color="purple", alpha=0.5) +
  geom_line(aes(x=Date, y=T.full), alpha=0.5)
p_manuel
```



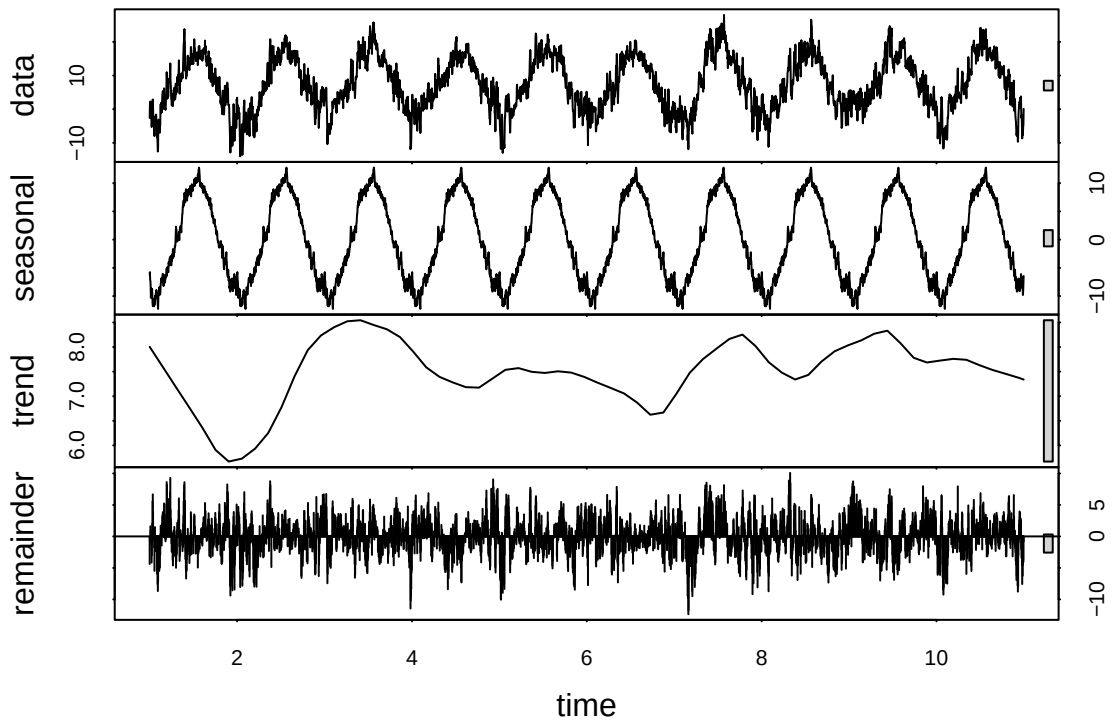

```
manuel_MSE <- mean((manuel_data$T.full - manuel_data$T.missing)^2)
manuel_MSE
```

```
## [1] 2.177454
```

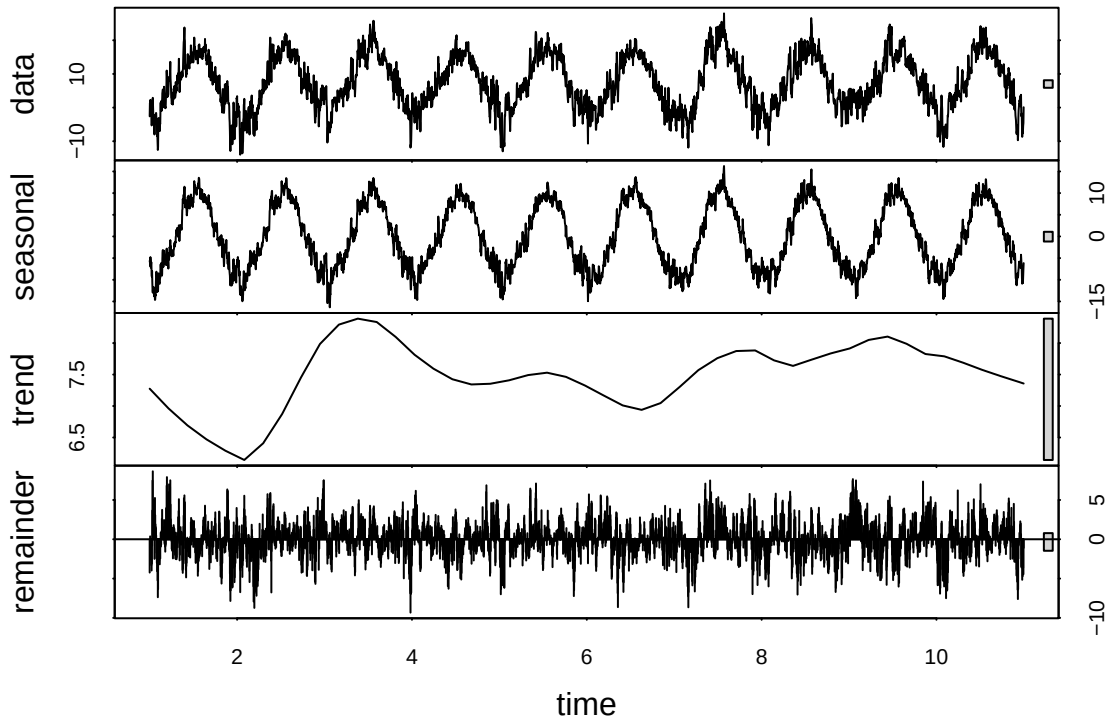
```
maunel.full.stl.w5 <- stl(ts(manuel_data$T.full, frequency = 365),
                          s.window = 5)
plot(maunel.full.stl.w5)
```



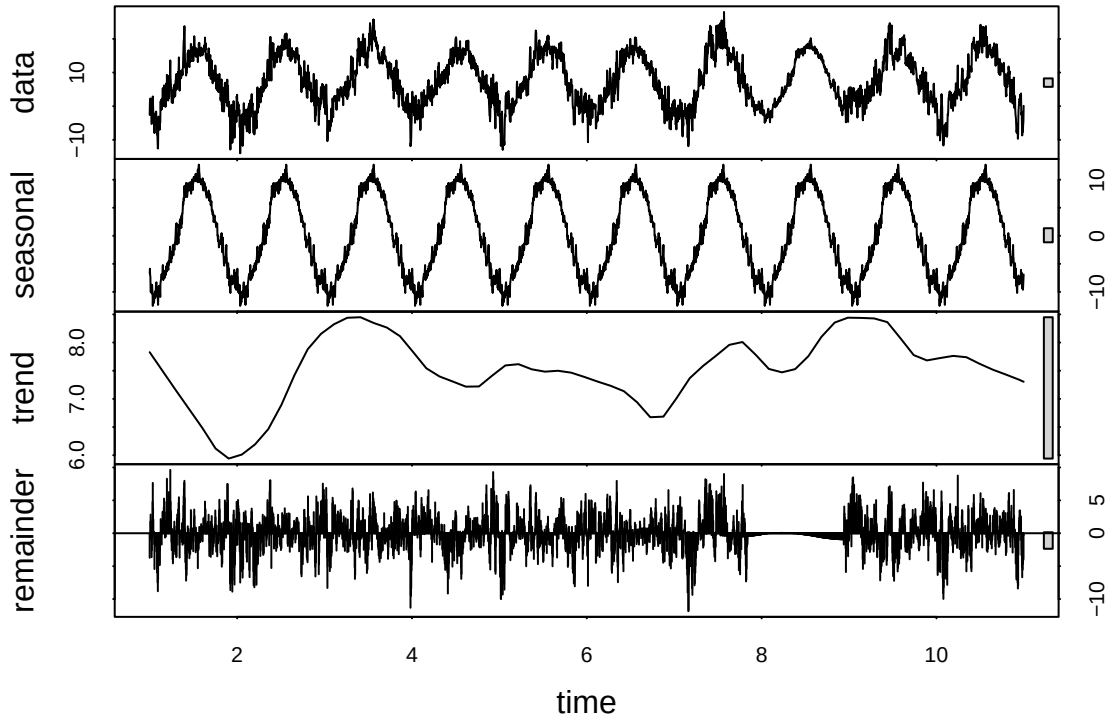

```
maunel.full.stl.wp <- stl(ts(manuel_data$T.full, frequency = 365),
                          s.window = "periodic")
plot(maunel.full.stl.wp)
```



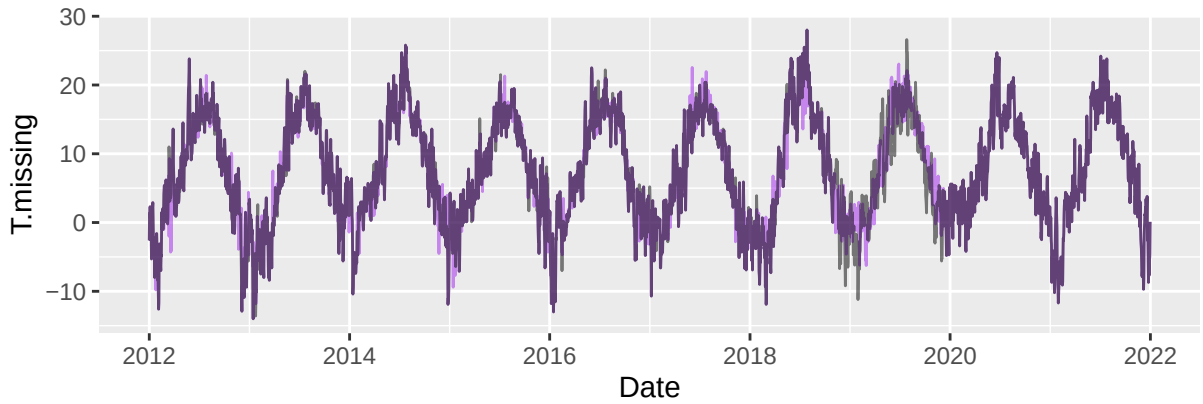

```
maunel.missing.stl.w5 <- stl(ts(manuel_data$T.missing, frequency = 365),
                             s.window = 5)
plot(maunel.full.stl.w5)
```

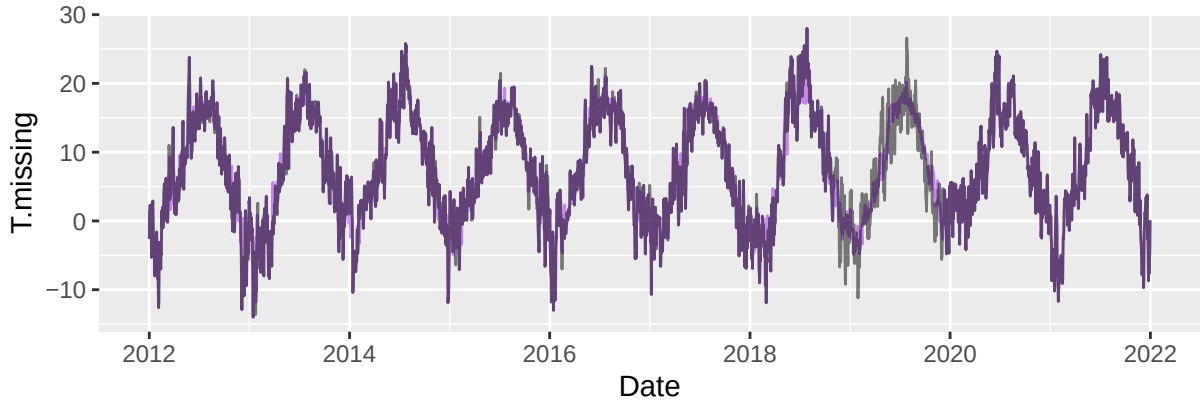



```
maunel.missing.stl.wp <- stl(ts(manuel_data$T.missing, frequency = 365),
                             s.window = "periodic")
plot(maunel.missing.stl.wp)
```




```
# Results

p5 / p_manuel
```






```
# error between manuel and na_seasplit
SEASSPLIT_MSE
```

```
## [1] 3.282046
```

```
manuel_MSE
```

```
## [1] 2.177454
```

```
diff_MSE <- SEASSPLIT_MSE - manuel_MSE
diff_MSE
```

```
## [1] 1.104592
```

By compering na_seasplit and our implementation we can se that they both follow the the full data, but with some difference in error.

Exercise 3

a)

```
# Loading libraries
#-------------------------------------------------------------------------------
library('readr')
library('dplyr')
library('ggplot2')
library('tidyverse')
library('lubridate')
library('zoo')
library('reshape2')
```

```
library('devtools')
library('factoextra')
```

```
covid <- read_csv("/Users/martinbergsholmnesse/Documents/NMBU/DAT320/covid.csv")
# View(covid)
# str(covid)

# Selecting time-window
covid <- covid[covid$date >= '2020-03-16' & covid$date <= '2022-01-01', ]

# Selecting countries and columns
countries <- c('SWE', 'DNK', 'NOR', 'GBR', 'ITA', 'IND')
covid <- covid[covid$iso_code %in% countries, c('date', 'iso_code', 'new_cases_per_million')]

# Checking that date-column is in correct format
class(covid$date)
```

```
## [1] "Date"
```

```
# Summary statistics of data.frame
summary(covid)
```

```
##       date              iso_code         new_cases_per_million
##  Min.   :2020-03-16   Length:3942        Min.   :-344.21
##  1st Qu.:2020-08-27   Class :character   1st Qu.:  16.10
##  Median :2021-02-07   Mode  :character   Median :  58.47
##  Mean   :2021-02-07                      Mean   : 168.43
##  3rd Qu.:2021-07-21                      3rd Qu.: 180.49
##  Max.   :2022-01-01                      Max.   :7058.81
```

```
# Reshaping data.frame
date <- covid$date
date <- date[!duplicated(date)]

swe <- covid[covid$iso_code == 'SWE', ]
swe <- select(swe, new_cases_per_million)
dnk <- covid[covid$iso_code == 'DNK', ]
dnk <- select(dnk, new_cases_per_million)
nor <- covid[covid$iso_code == 'NOR', ]
nor <- select(nor, new_cases_per_million)
gbr <- covid[covid$iso_code == 'GBR', ]
gbr <- select(gbr, new_cases_per_million)
ita <- covid[covid$iso_code == 'ITA', ]
ita <- select(ita, new_cases_per_million)
ind <- covid[covid$iso_code == 'IND', ]
ind <- select(ind, new_cases_per_million)

df <- data.frame(date, swe, dnk, nor, gbr, ita, ind)

# Renaming data.frame
names(df) <- c('Date', 'Sweden', 'Denmark', 'Norway',
               'Britain', 'Italia', 'India')
```

```
# Exploratory analysis
# ------------------------------------------------------------------------------
# Dimensionality
```

```r
nrow(df)
```

```
## [1] 657
```

```r
ncol(df)
```

```
## [1] 7
```

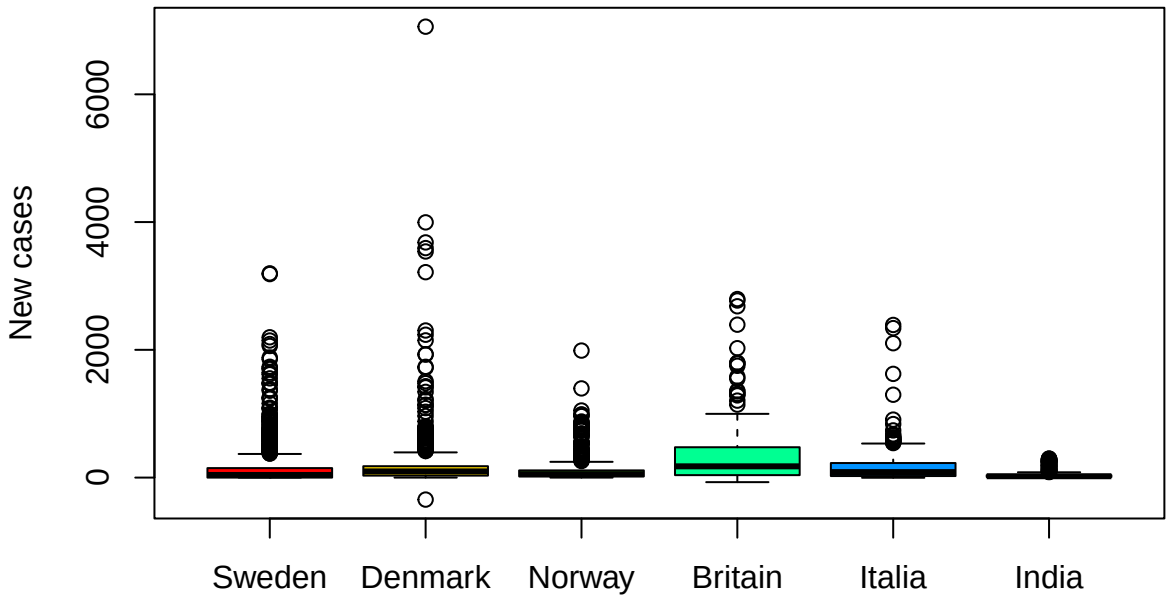
```r
dim(df)
```

```
## [1] 657   7
```

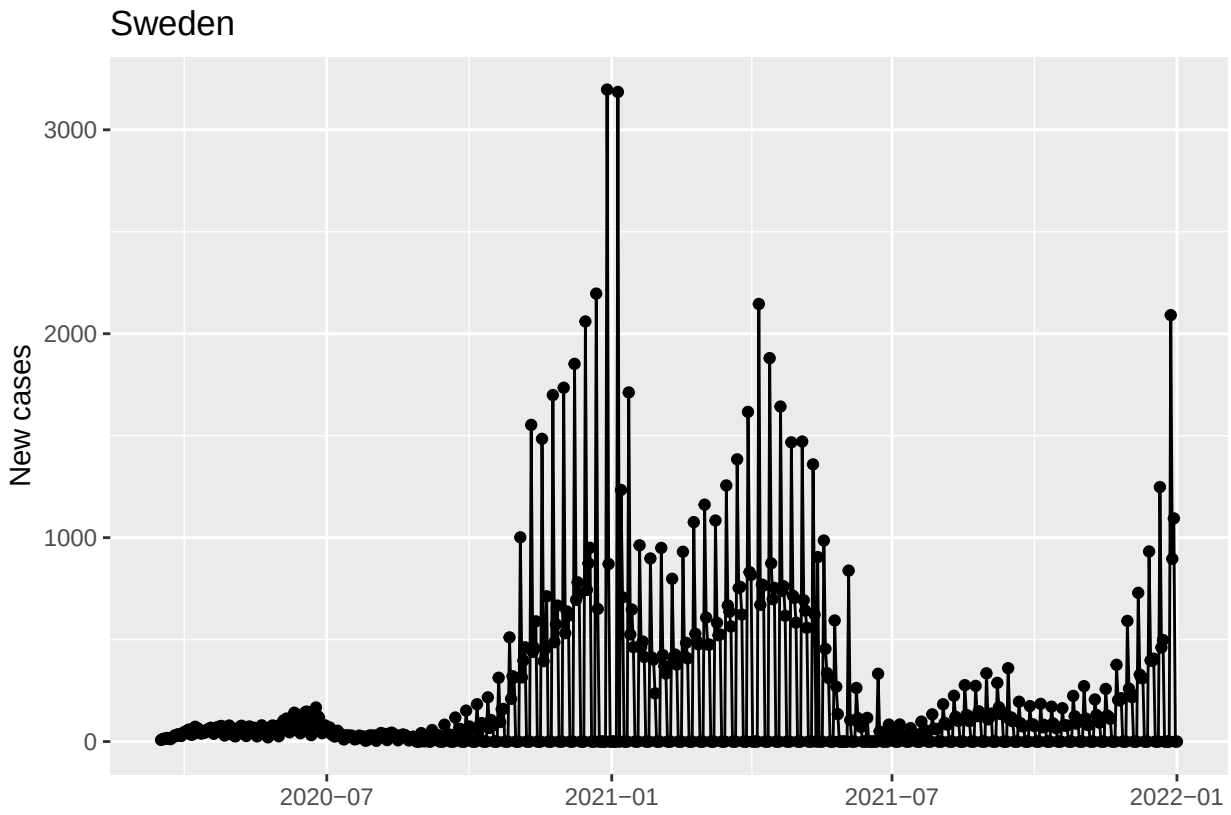
```r
# Summary statistics
summary(df)
```

```
##       Date                Sweden           Denmark            Norway
##  Min.   :2020-03-16   Min.   :   0.00   Min.   :-344.21   Min.   :   0.00
##  1st Qu.:2020-08-27   1st Qu.:   0.00   1st Qu.:  30.79   1st Qu.:  18.11
##  Median :2021-02-07   Median :  40.94   Median :  96.85   Median :  53.61
##  Mean   :2021-02-07   Mean   : 196.81   Mean   : 215.33   Mean   : 110.05
##  3rd Qu.:2021-07-21   3rd Qu.: 148.72   3rd Qu.: 178.38   3rd Qu.: 112.89
##  Max.   :2022-01-01   Max.   :3197.29   Max.   :7058.81   Max.   :1988.24
##     Britain            Italia             India
##  Min.   : -71.25   Min.   :  -2.452   Min.   :  0.000
##  1st Qu.:  38.46   1st Qu.:  24.185   1st Qu.:  8.817
##  Median : 176.39   Median :  85.940   Median : 21.552
##  Mean   : 292.89   Mean   : 157.387   Mean   : 38.111
##  3rd Qu.: 475.08   3rd Qu.: 227.871   3rd Qu.: 38.803
##  Max.   :2795.06   Max.   :2389.615   Max.   :297.248
```

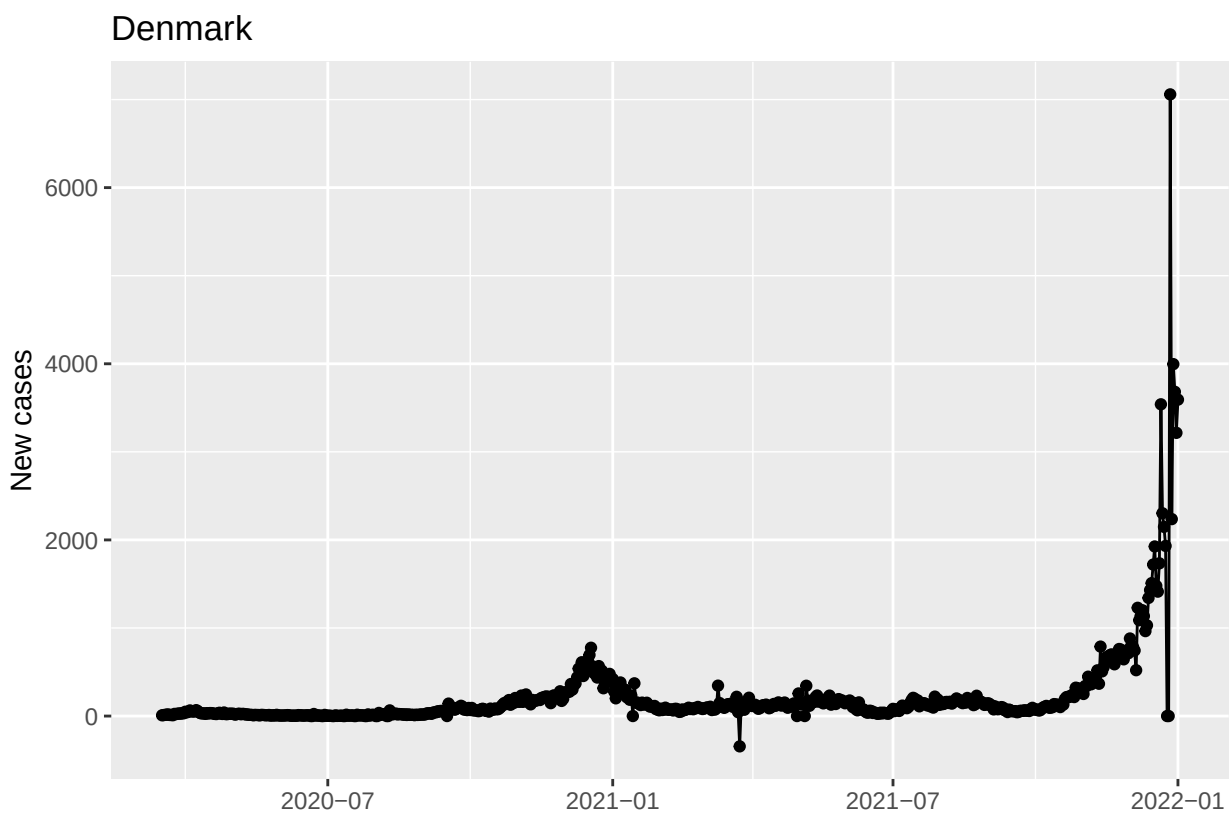
```r
# Plots
# ------------------------------------------------------------------------------

# Distribution
boxdf <- select(df, Sweden, Denmark, Norway, Britain, Italia, India)
stacked_df <- stack(boxdf)
boxplot(stacked_df$values ~ stacked_df$ind,
        col = rainbow(ncol(df)), xlab='', ylab='New cases')
```



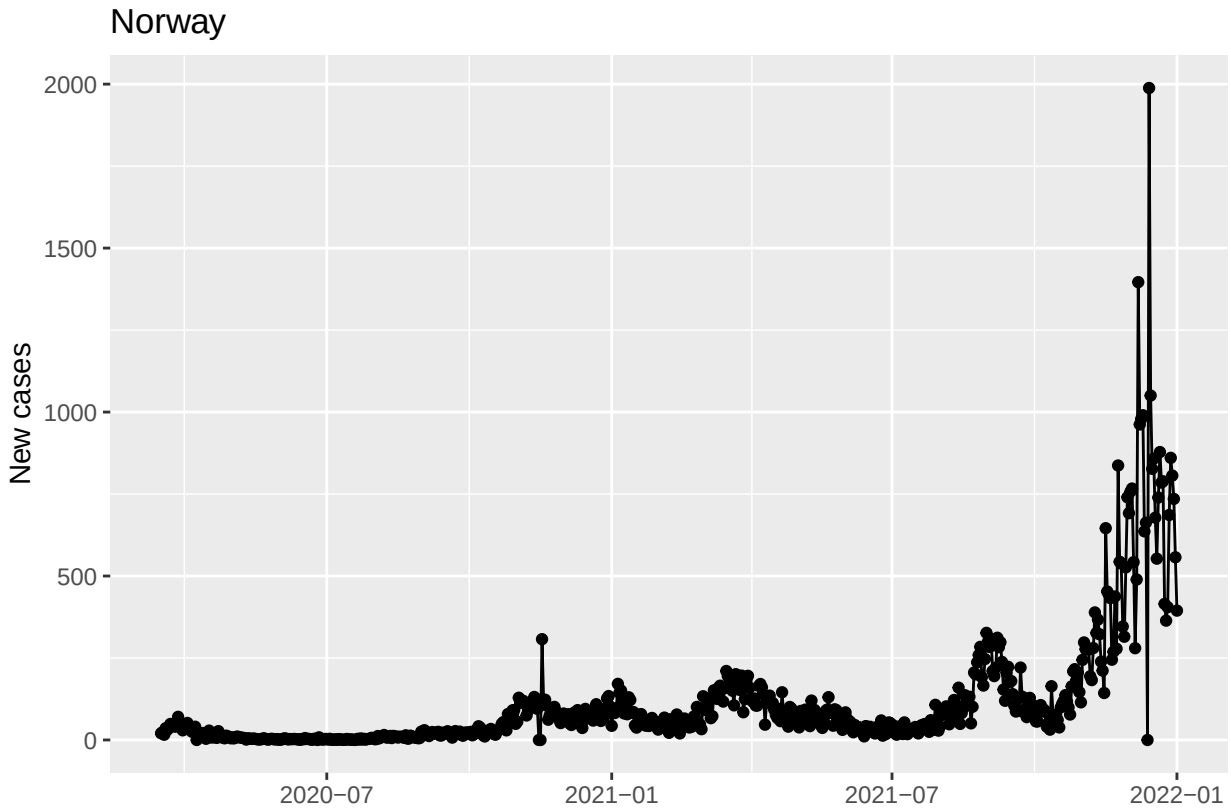

```
# Line plot for the countries separately
# Sweden
p <- ggplot(df, aes(x=Date, y=Sweden)) +
  geom_line() +
  geom_point() +
  labs(x='', y='New cases', title='Sweden')
p
```



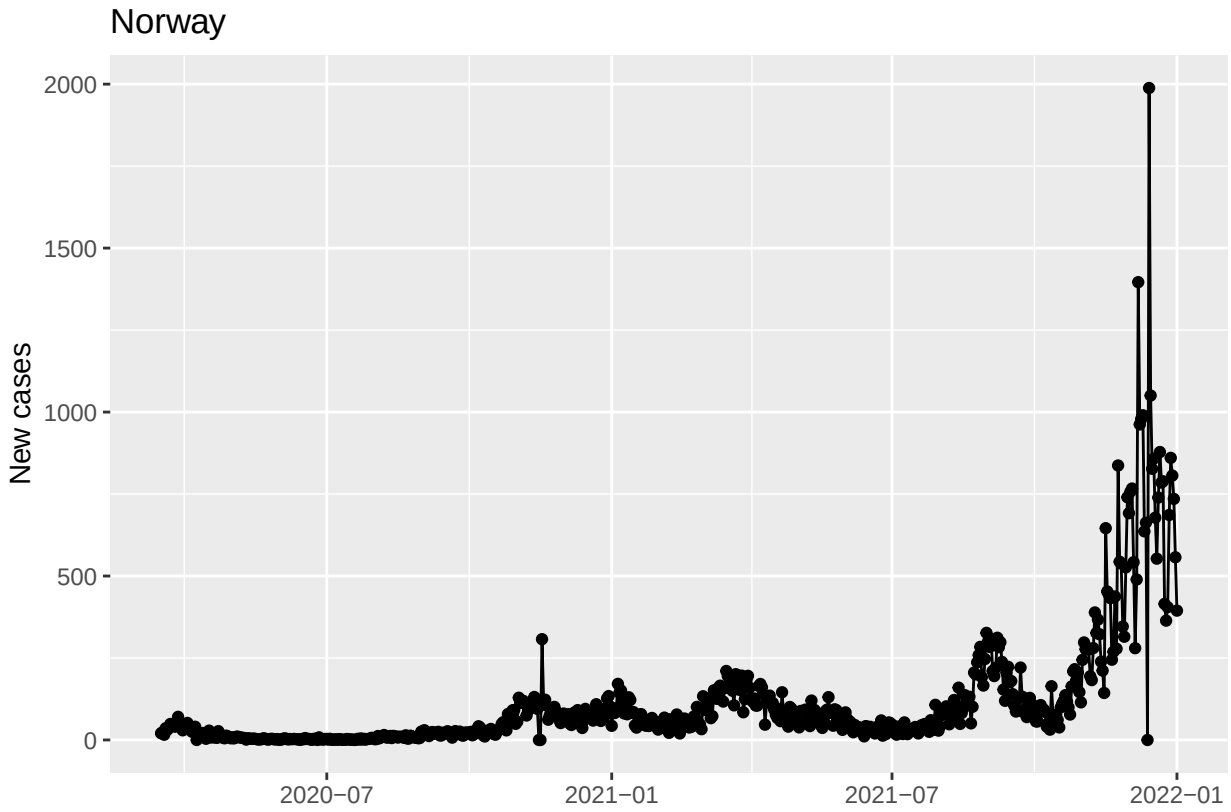
```
# Denmark
p <- ggplot(df, aes(x=Date, y=Denmark)) +
  geom_line() +
  geom_point() +
  labs(x='', y='New cases', title='Denmark')
p
```



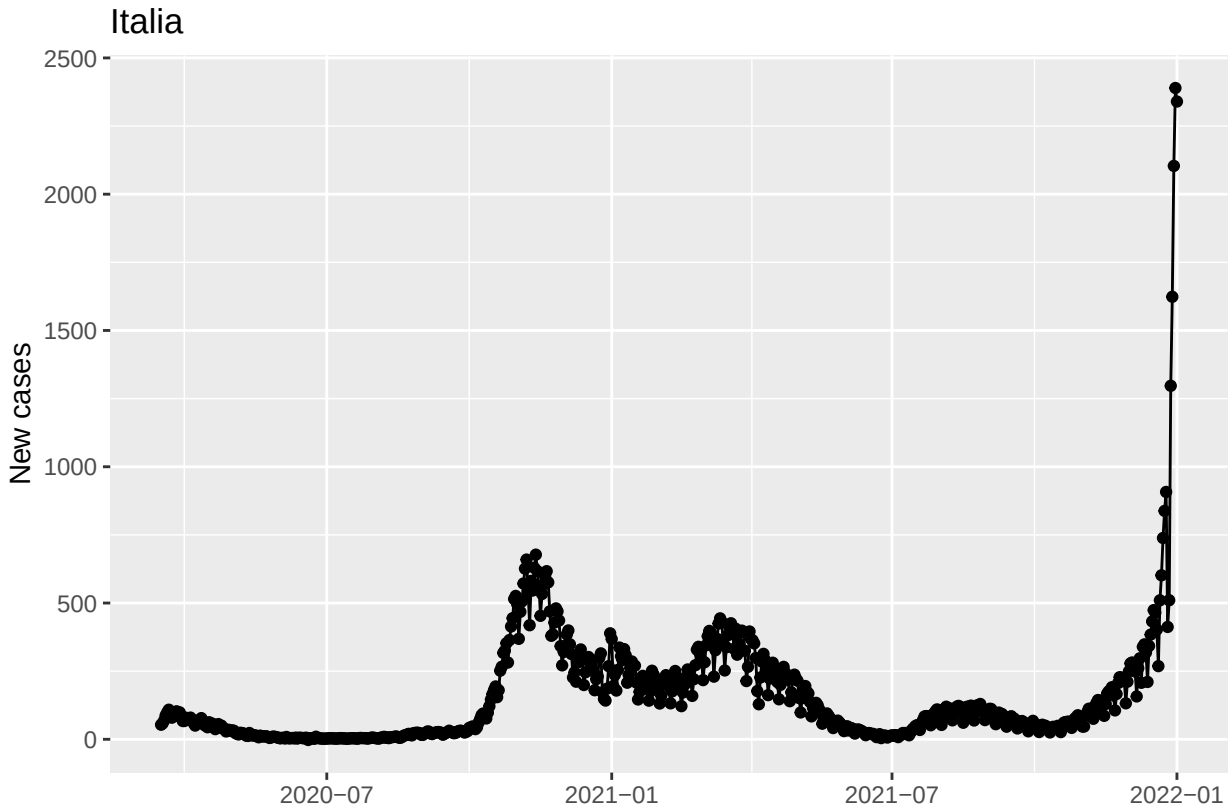

```
# Norway
p <- ggplot(df, aes(x=Date, y=Norway)) +
  geom_line() +
  geom_point() +
  labs(x='', y='New cases', title='Norway')
p
```



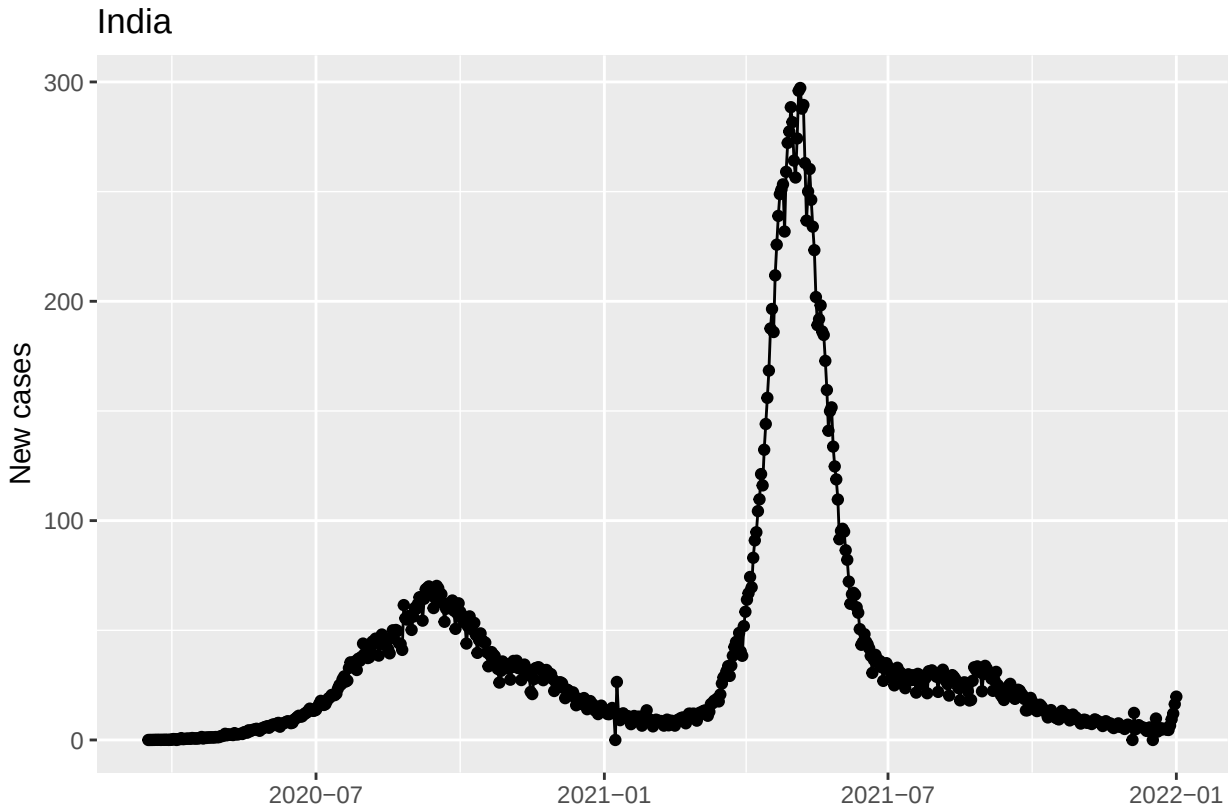

```
# Great Britain
p <- ggplot(df, aes(x=Date, y=Norway)) +
  geom_line() +
  geom_point() +
  labs(x='', y='New cases', title='Norway')
p
```



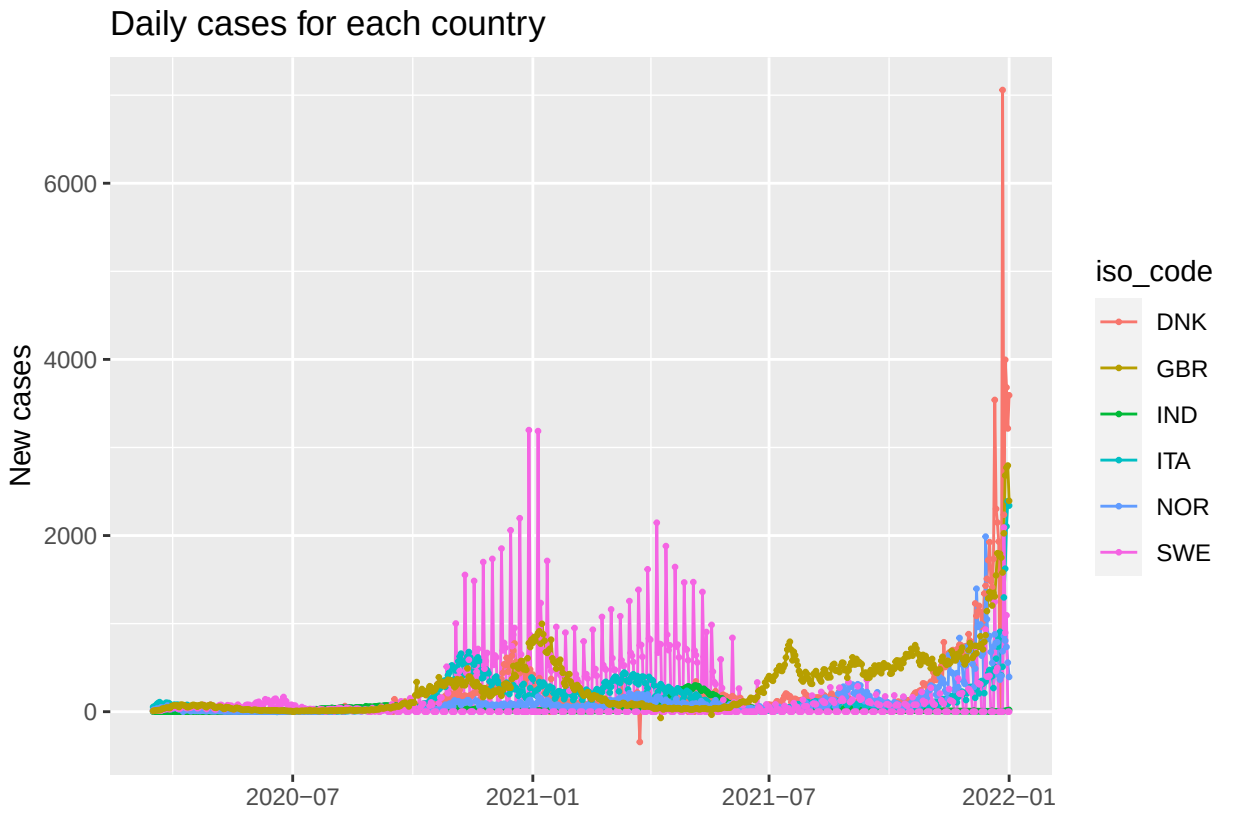

```
# Italia
p <- ggplot(df, aes(x=Date, y=Italia)) +
  geom_line() +
  geom_point() +
  labs(x='', y='New cases', title='Italia')
p
```



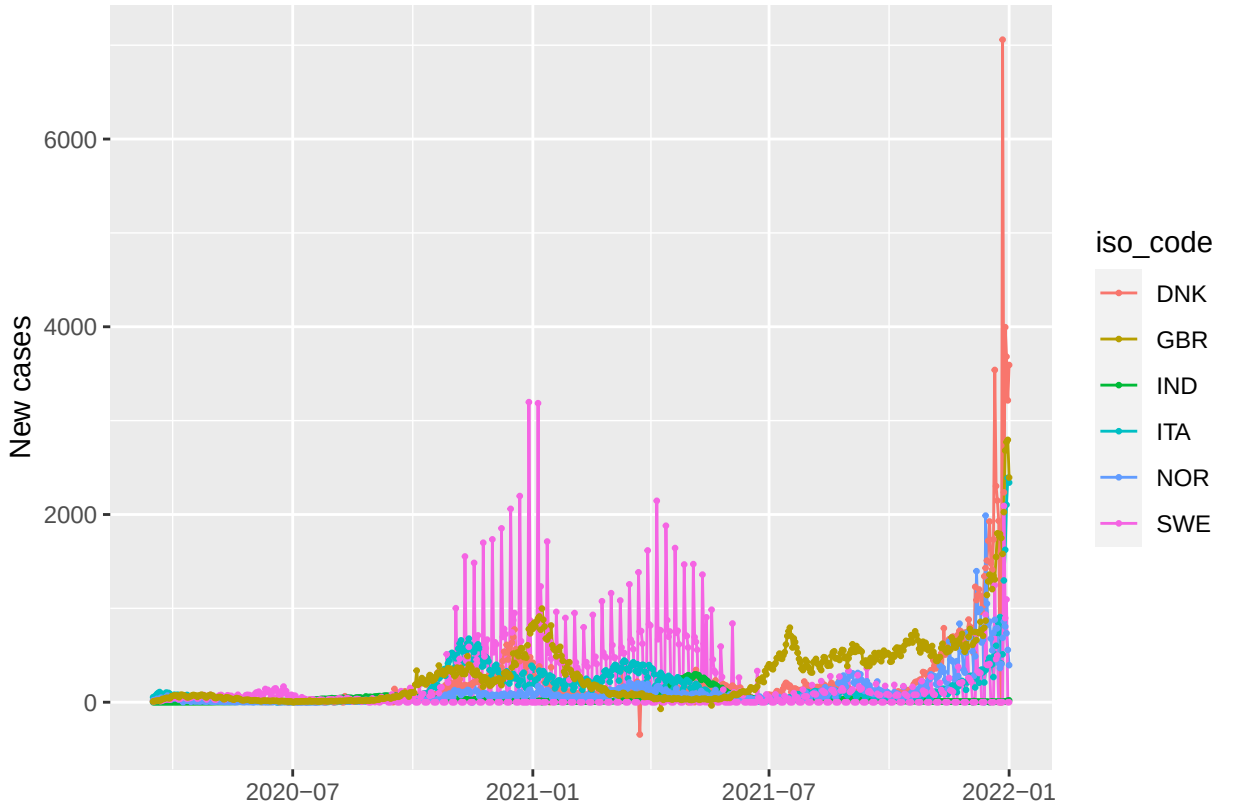

```
# India
p <- ggplot(df, aes(x=Date, y=India)) +
  geom_line() +
  geom_point() +
  labs(x='', y='New cases', title='India')
p
```




```
# Line plot, all countries combined
covid_cases <- melt(covid, id.vars=c('iso_code', 'date'),
                    value.name='Value')

covid_plot <- ggplot(data=covid_cases, aes(x=date,
                                           y=Value,
                                           group=iso_code,
                                           colour=iso_code)) +
  geom_line() +
  geom_point(size=0.5) +
  labs(y='New cases', x='')
covid_plot + ggtitle('Daily cases for each country')
```




```
covid_plot
```



After restricting the dataset to the specified column, the series is a univariate time series with only one

variable that is varying over. The time domain is selected to be from 16.03.2020 to 01.01.2022. There is one measurement each day, so the time series has daily resolution.

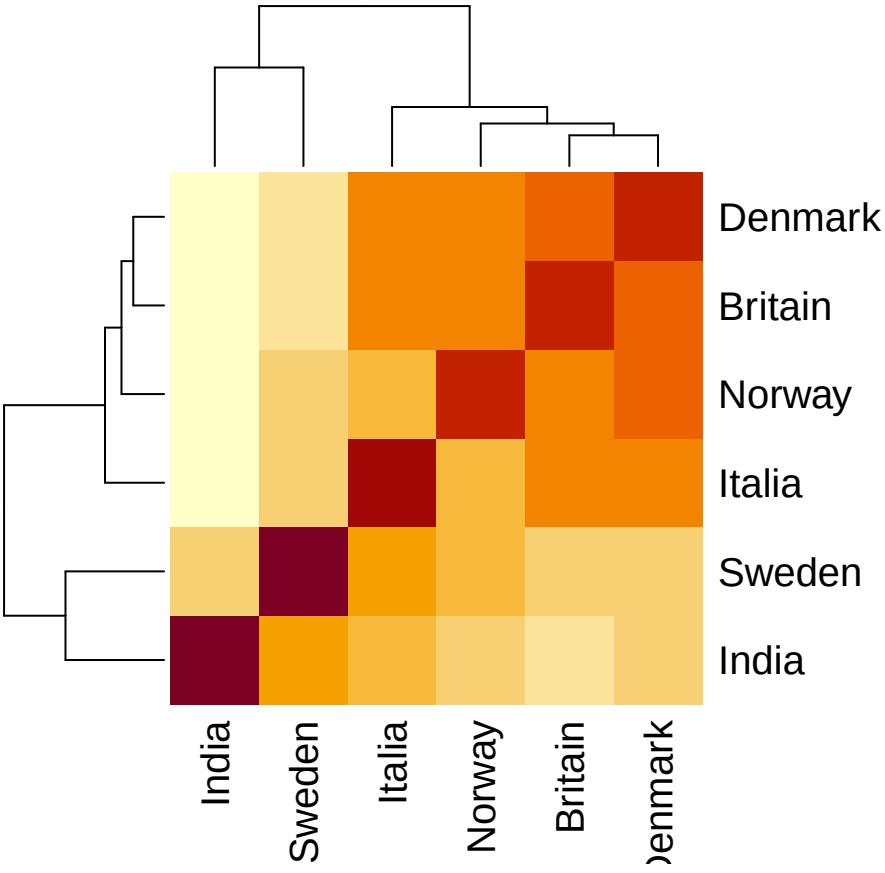
The dataset looks complete, since either of the countries had missing values for the measurement of new cases. However, since Sweden did not test for covid every day, the series for this country contains many values that are 0. Denmark, Great Britain and India have some entries that are below 0. This can be seen from the minimum value in the summary statistics and also in the line plot.

b)

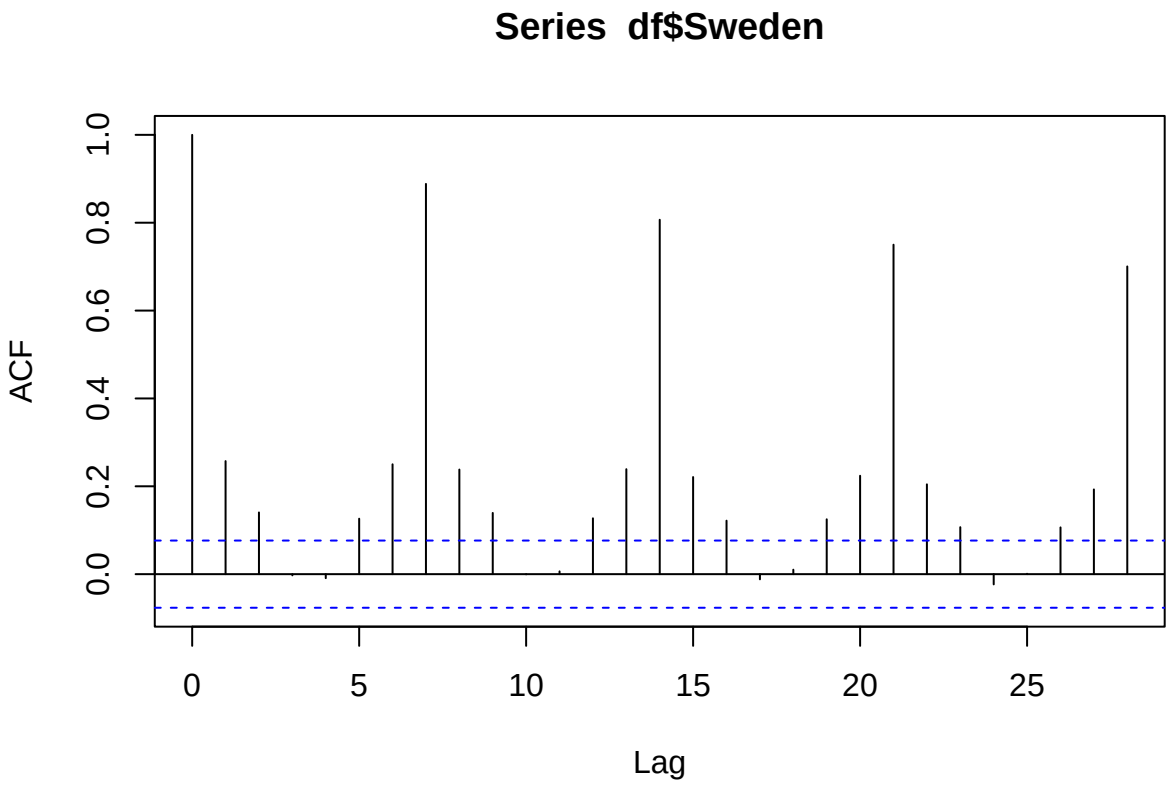
```
# ------------------------------------------------------------------------------
# Correlation-matrix
dfCorr <- select(df, Sweden, Denmark, Norway, Britain, Italia, India)
corr <- cor(dfCorr, method='pearson')
round(corr, 3)
```

```
##         Sweden Denmark Norway Britain Italia  India
## Sweden   1.000   0.162  0.213   0.116  0.307  0.154
## Denmark  0.162   1.000  0.664   0.729  0.624 -0.098
## Norway   0.213   0.664  1.000   0.631  0.429 -0.119
## Britain  0.116   0.729  0.631   1.000  0.625 -0.277
## Italia   0.307   0.624  0.429   0.625  1.000 -0.025
## India    0.154  -0.098 -0.119  -0.277 -0.025  1.000
```

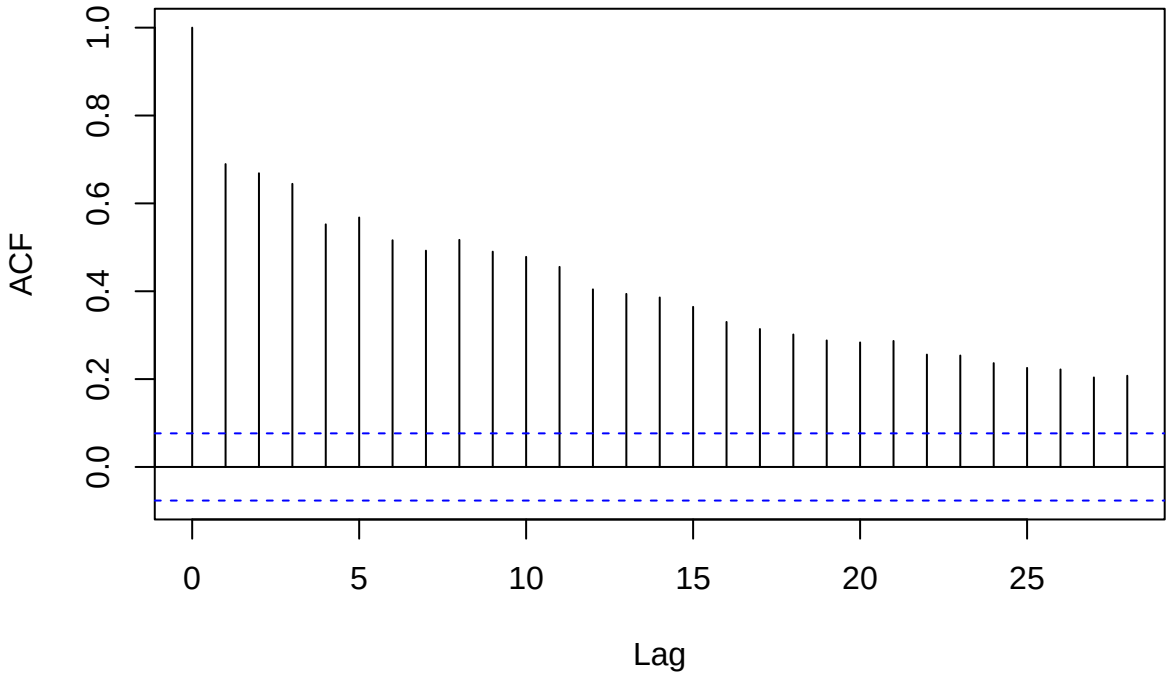
```
heatmap(corr)
```



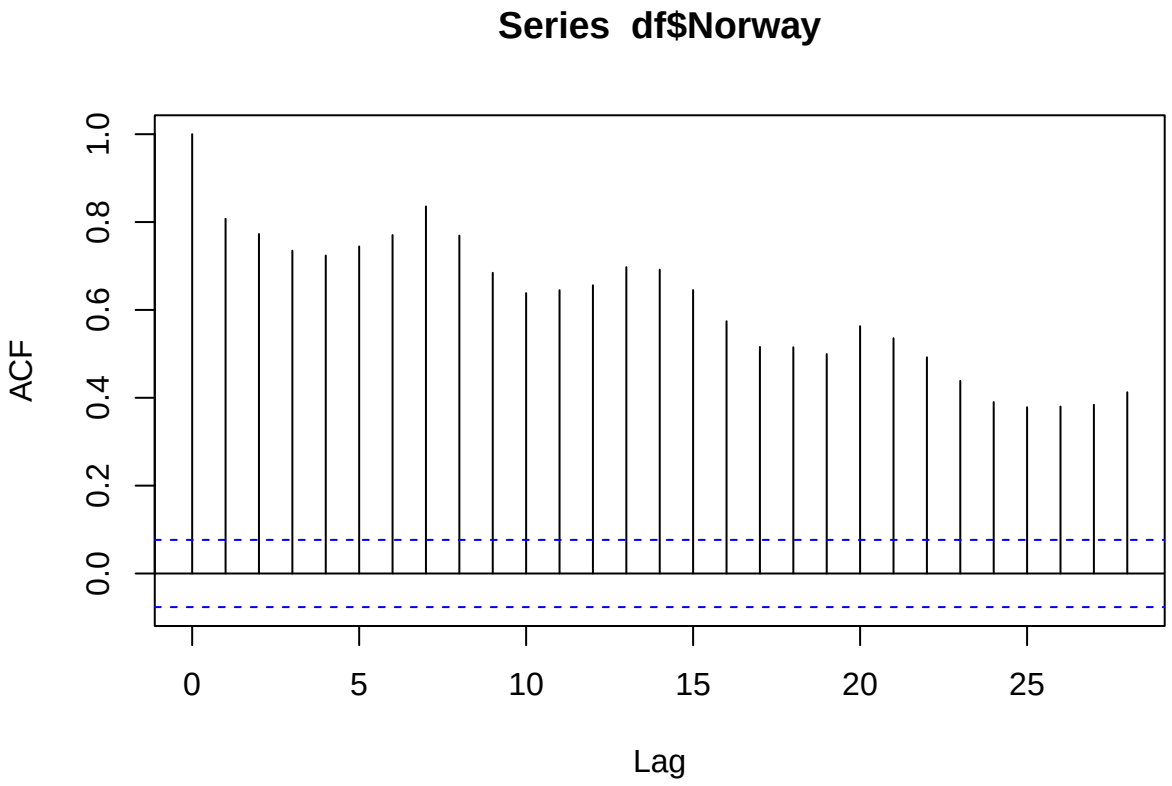
```
# Autocorrelation
acf(df$Sweden)
```



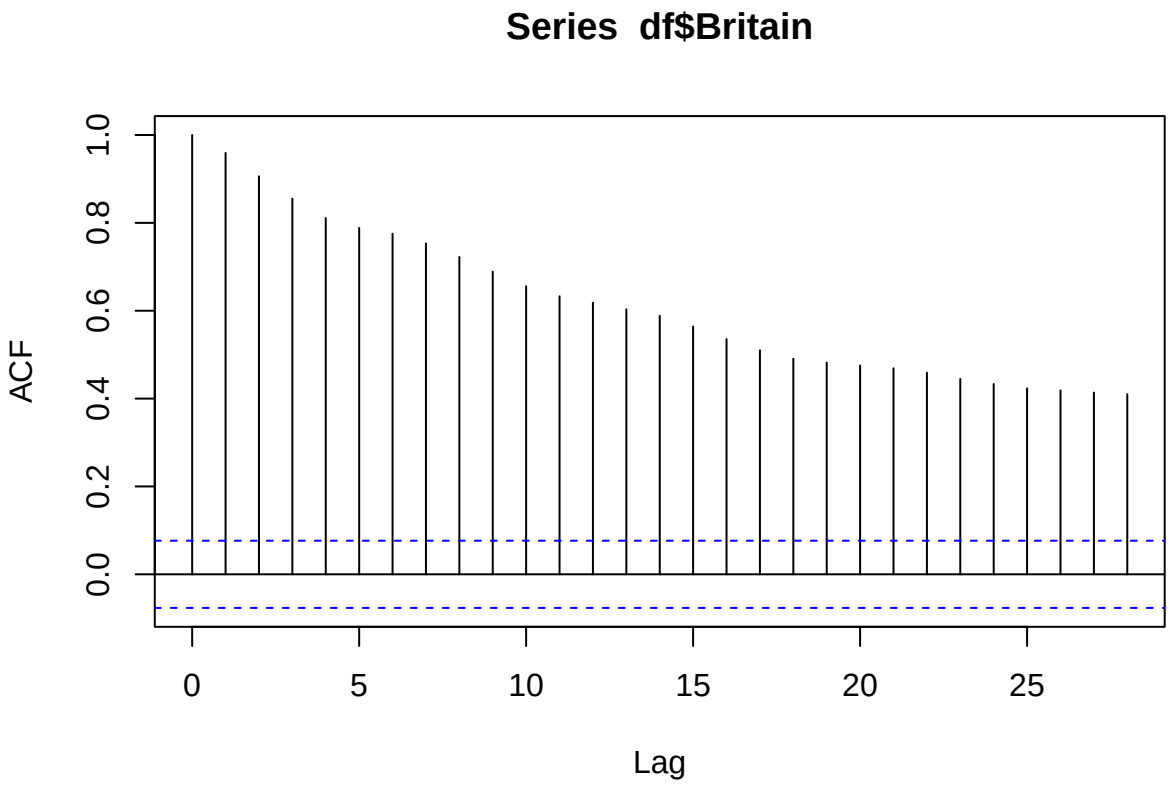

```
acf(df$Denmark)
```



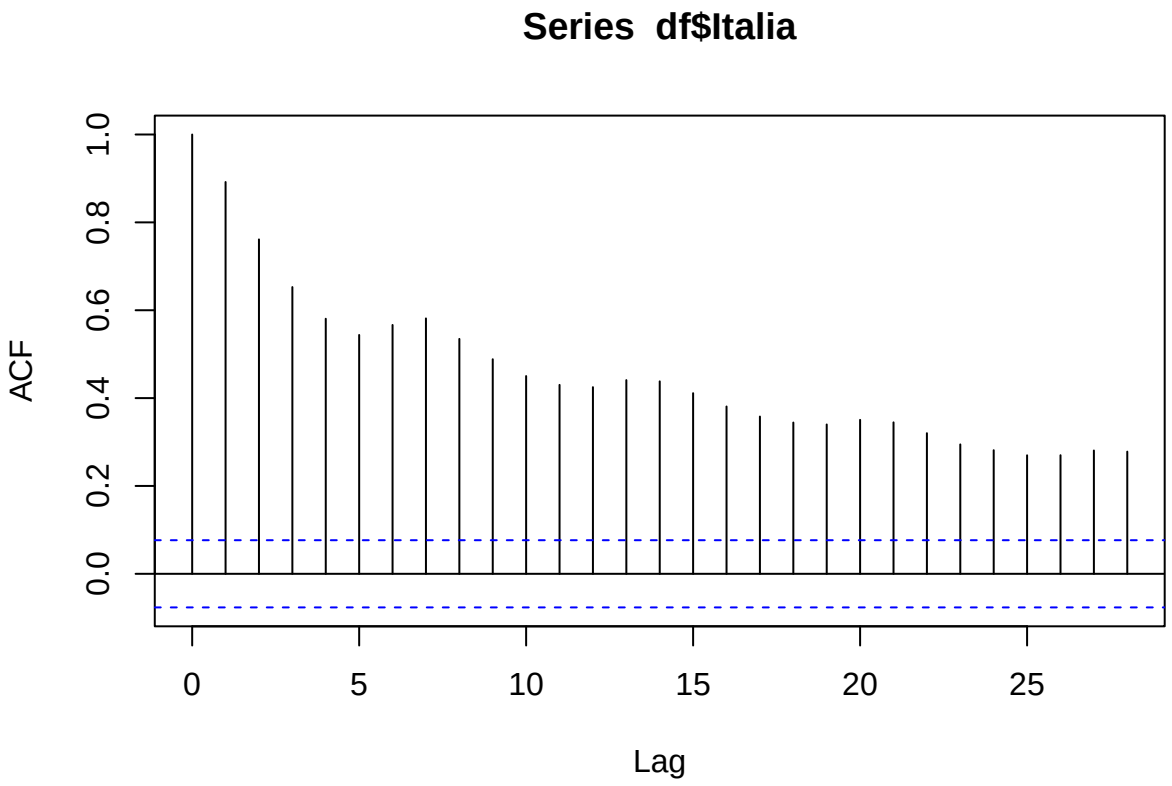
```
acf(df$Norway)
```



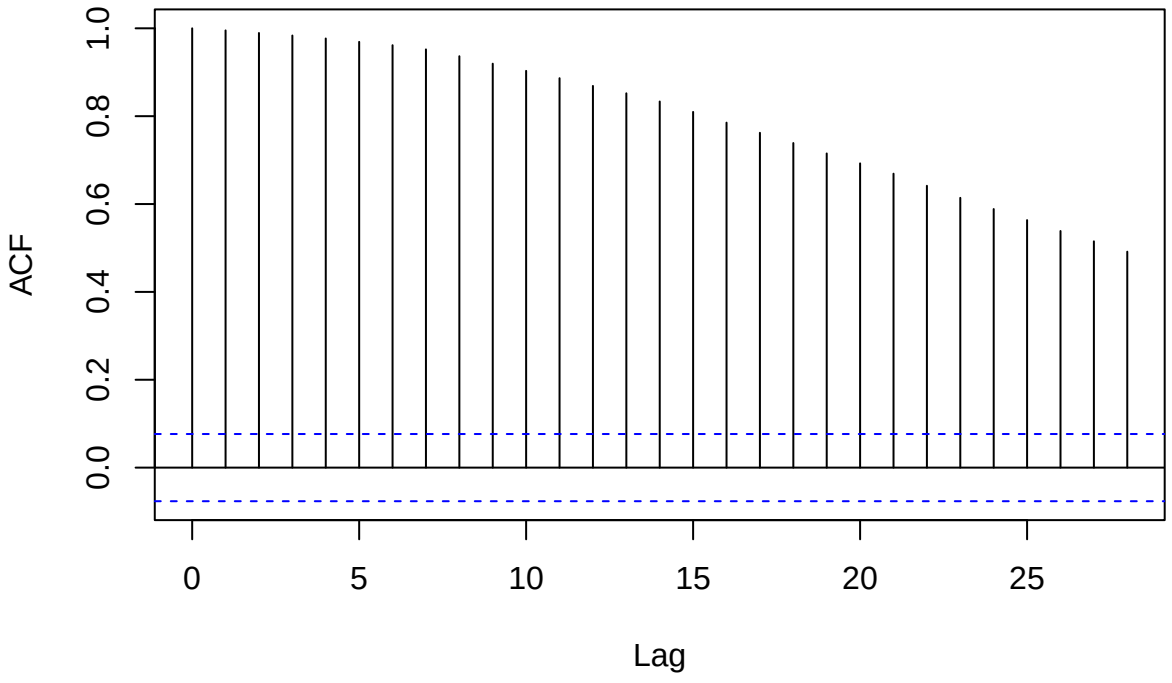

```
acf(df$Britain)
```



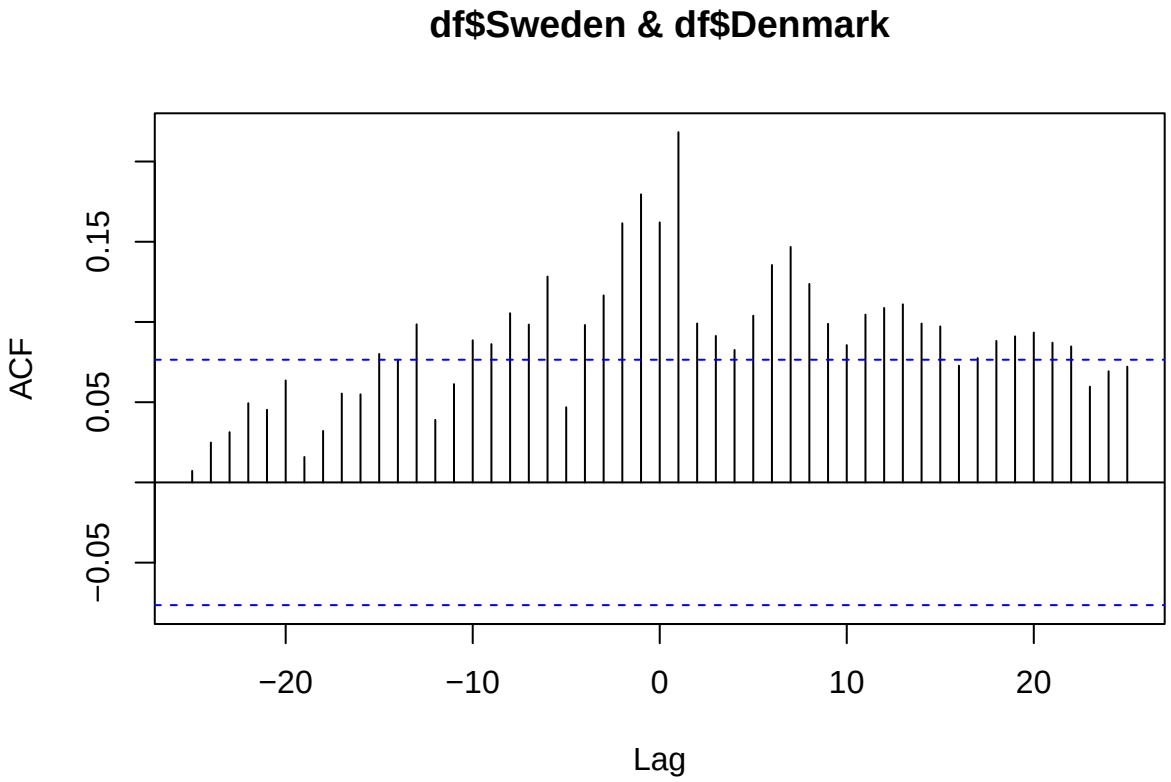
```
acf(df$Italia)
```



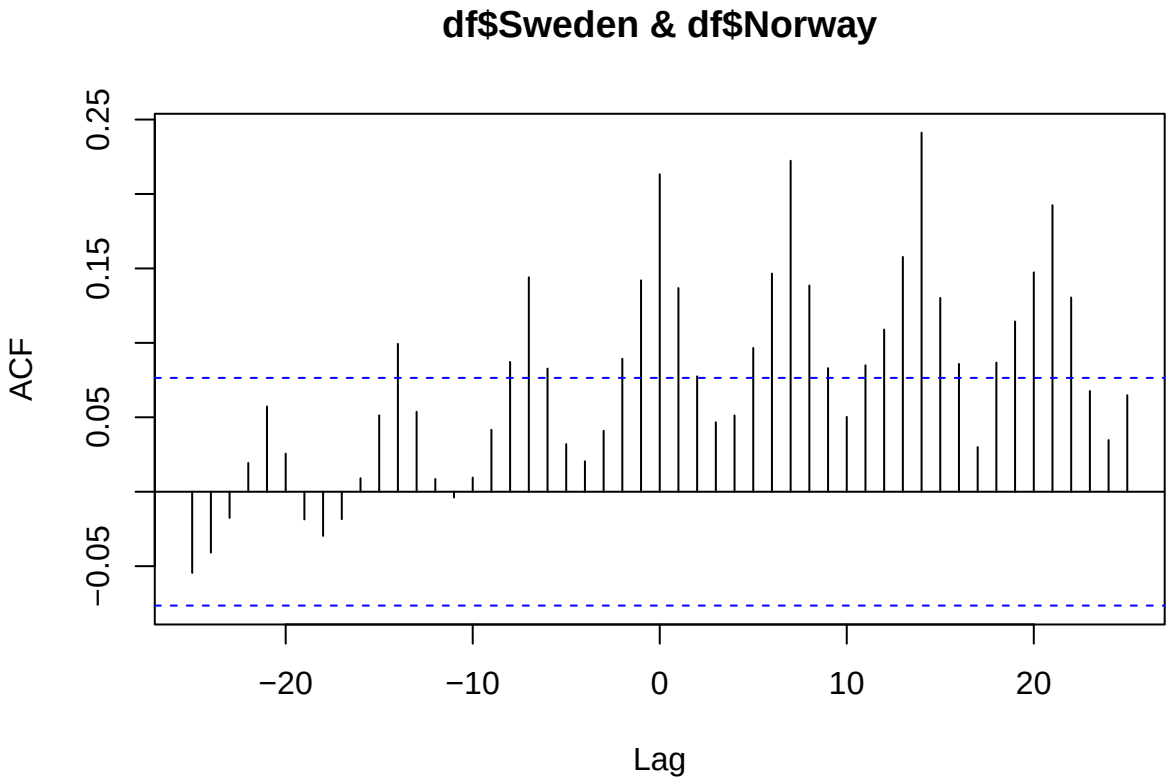

```
acf(df$India)
```



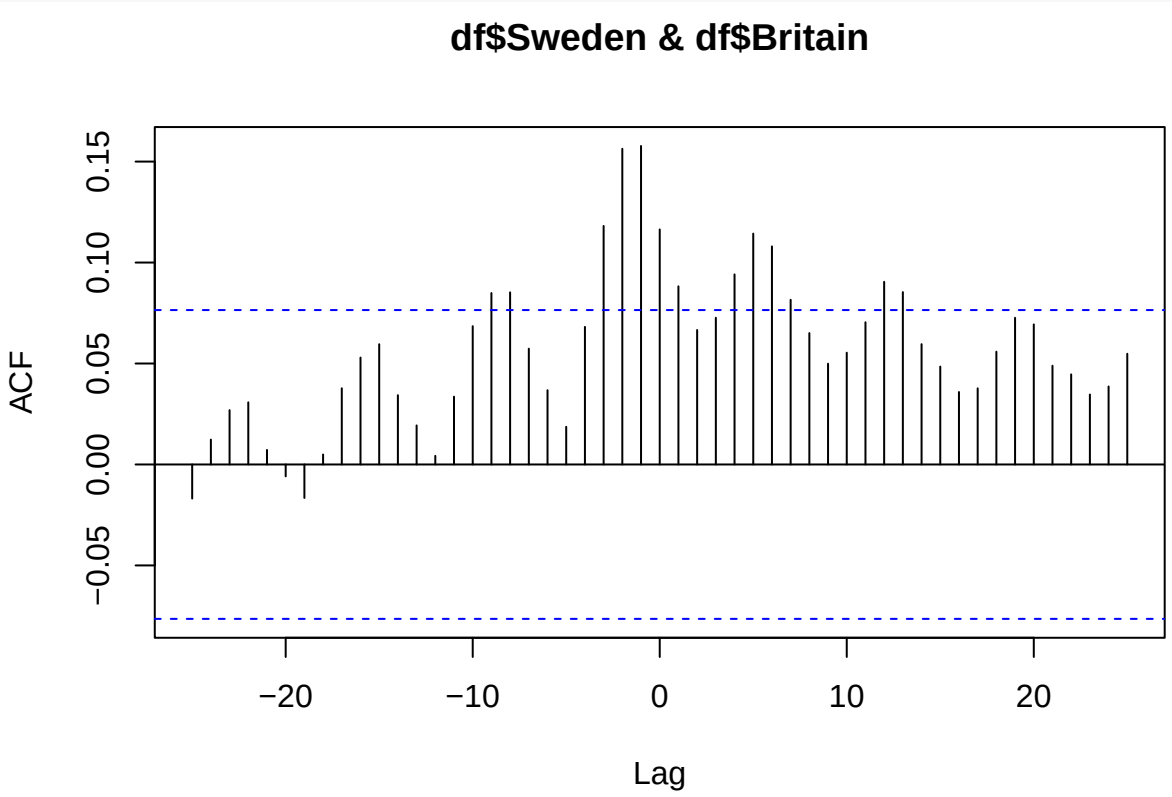
```r
# Cross-correlation
# Sweden
ccf(df$Sweden, df$Denmark)
```



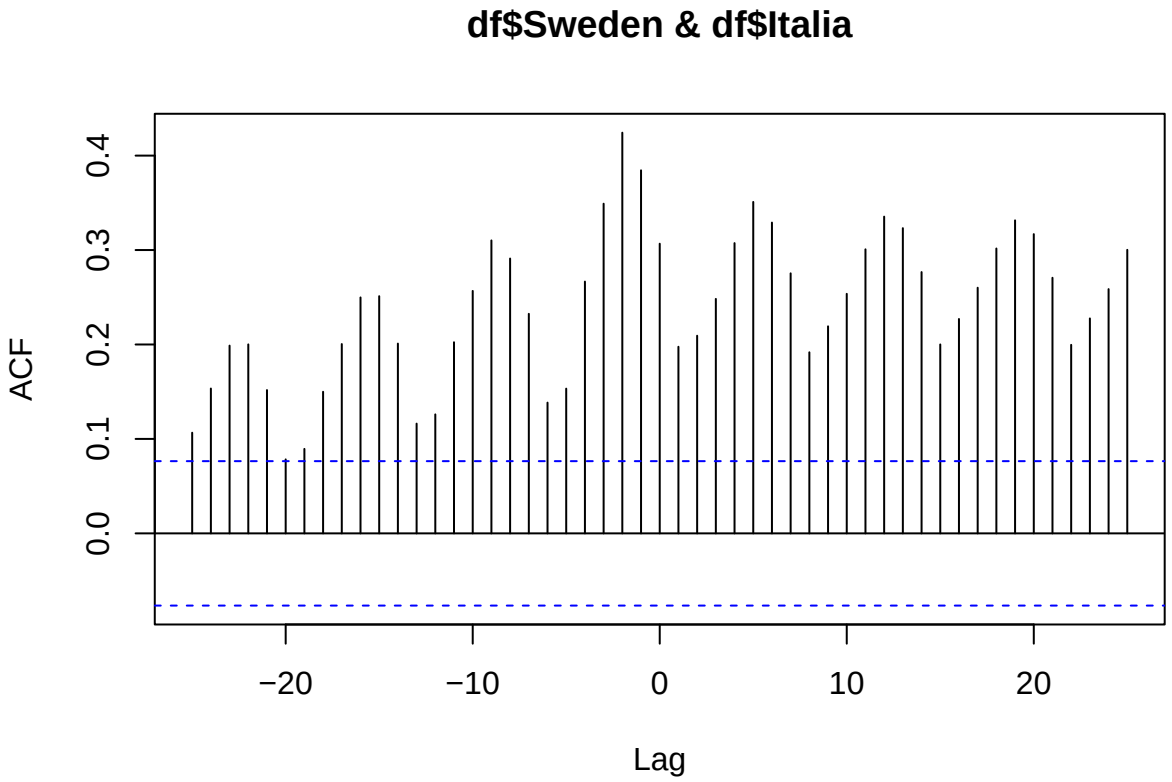

```r
ccf(df$Sweden, df$Norway)
```



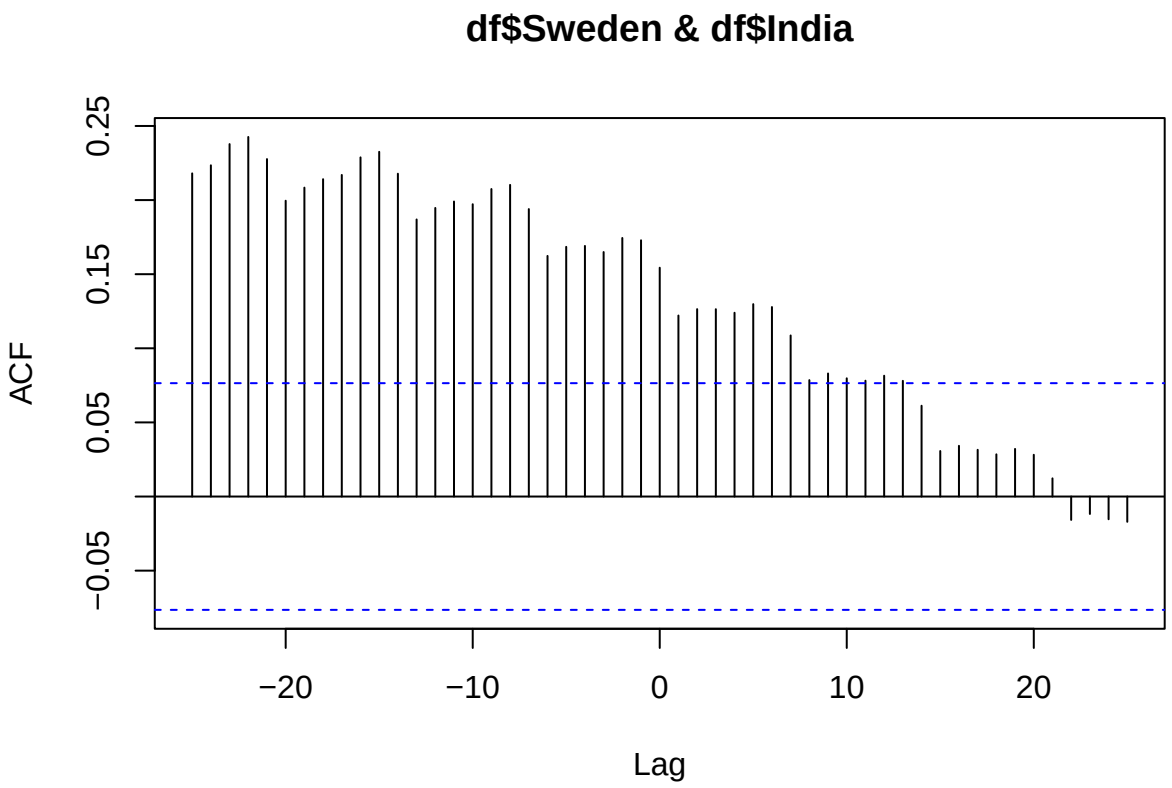

```
ccf(df$Sweden, df$Britain)
```



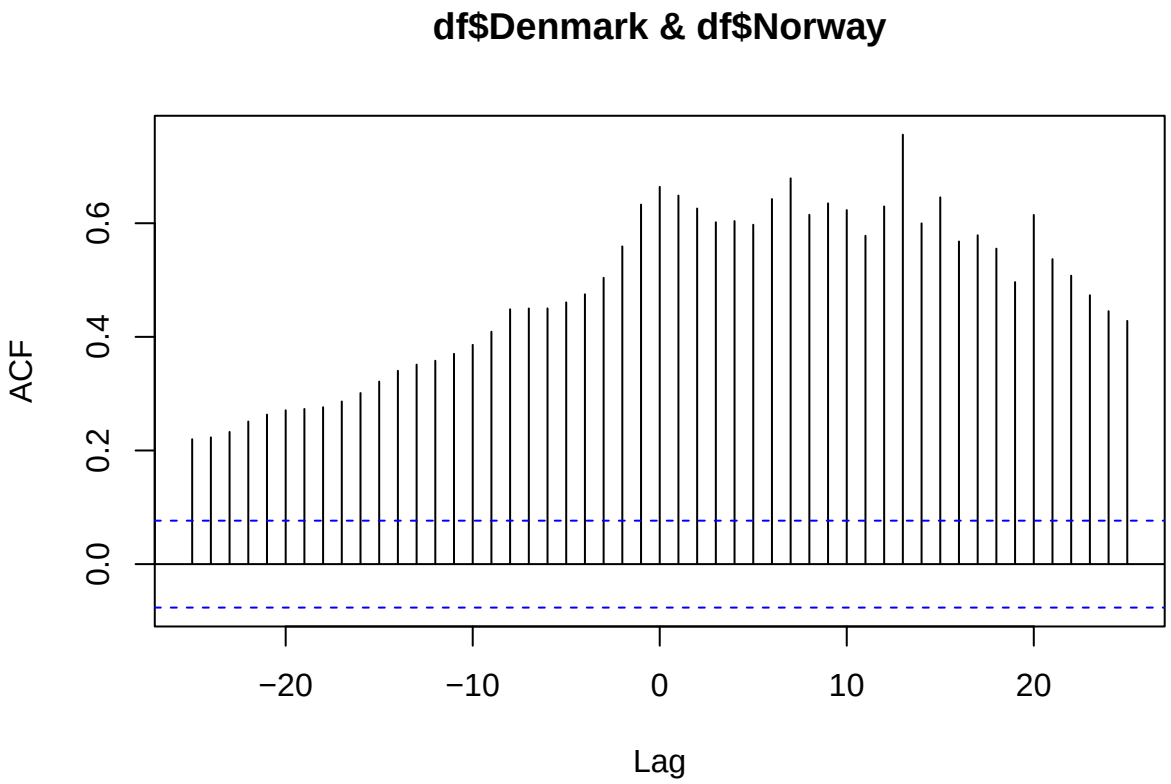

```
ccf(df$Sweden, df$Italia)
```



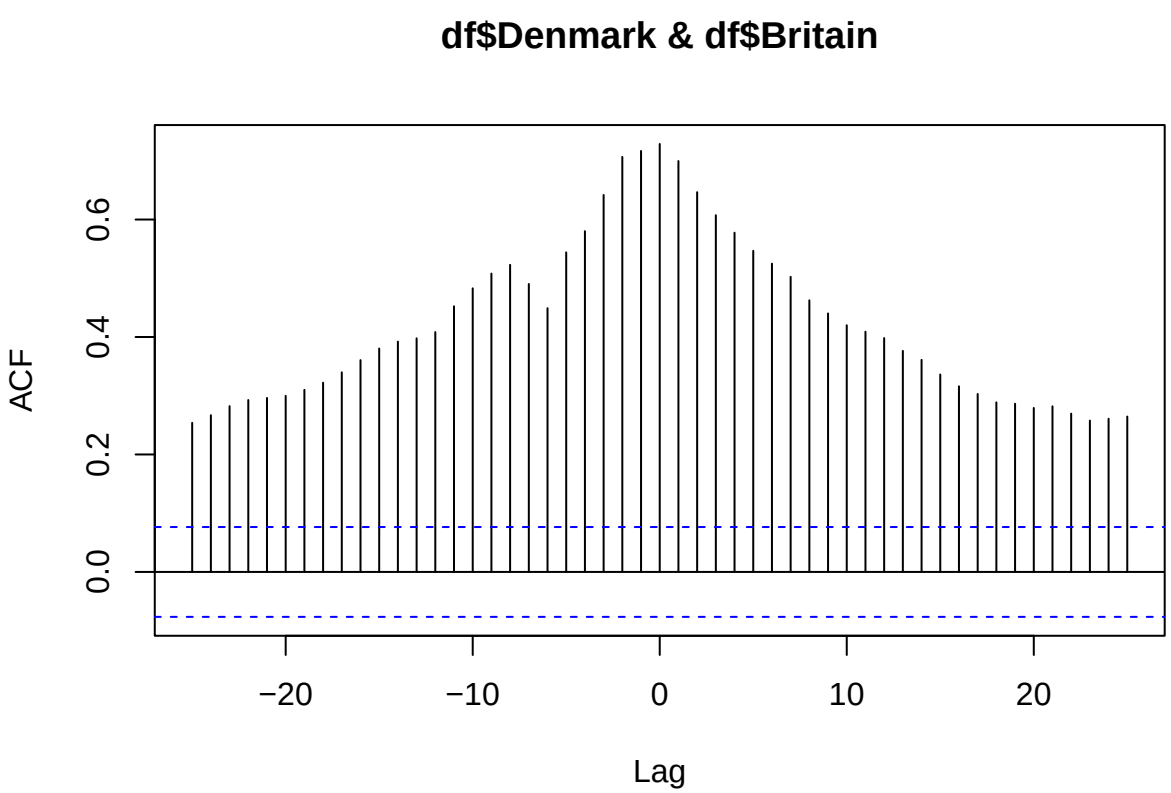

```
ccf(df$Sweden, df$India)
```



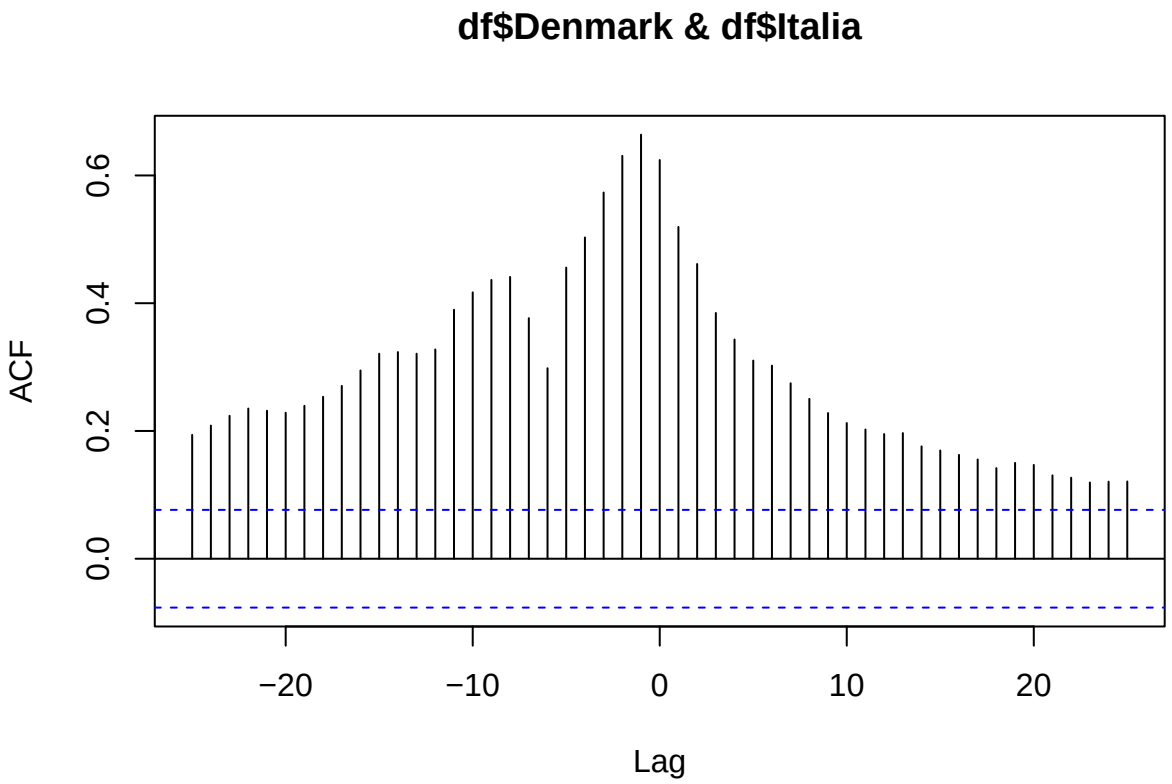

```
# Denmark
ccf(df$Denmark, df$Norway)
```



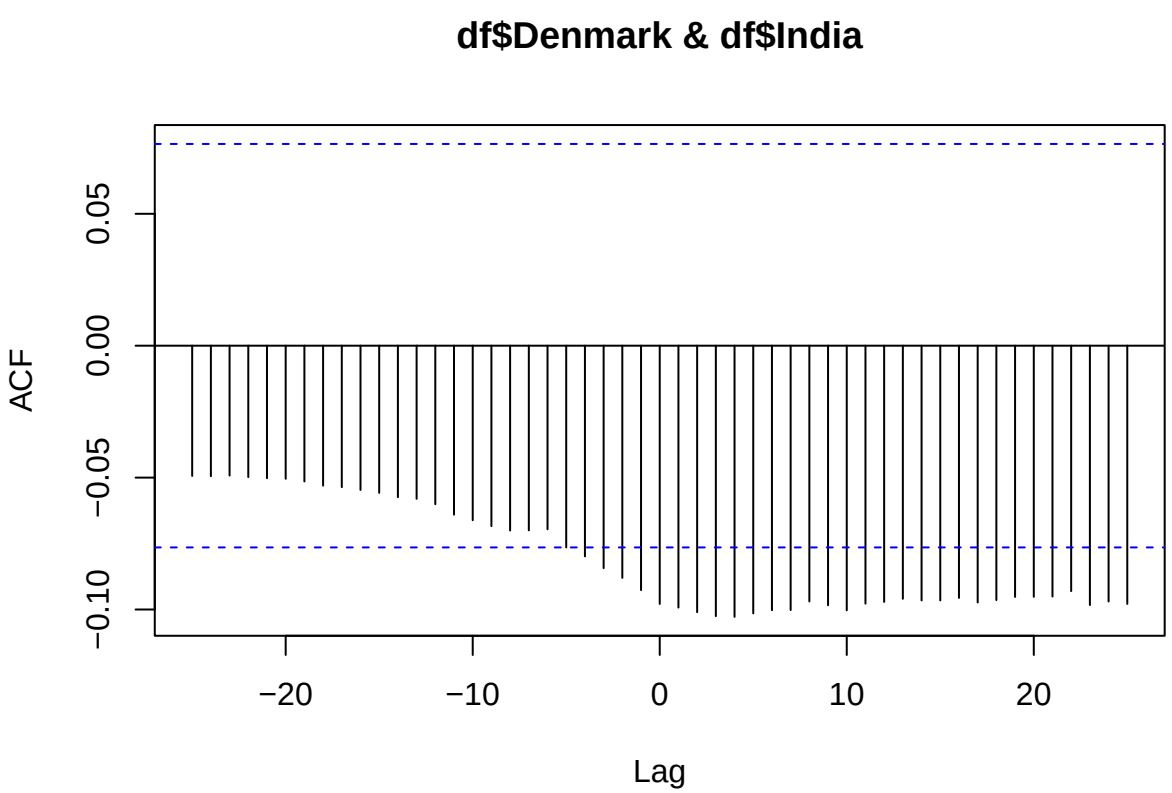

```
ccf(df$Denmark, df$Britain)
```



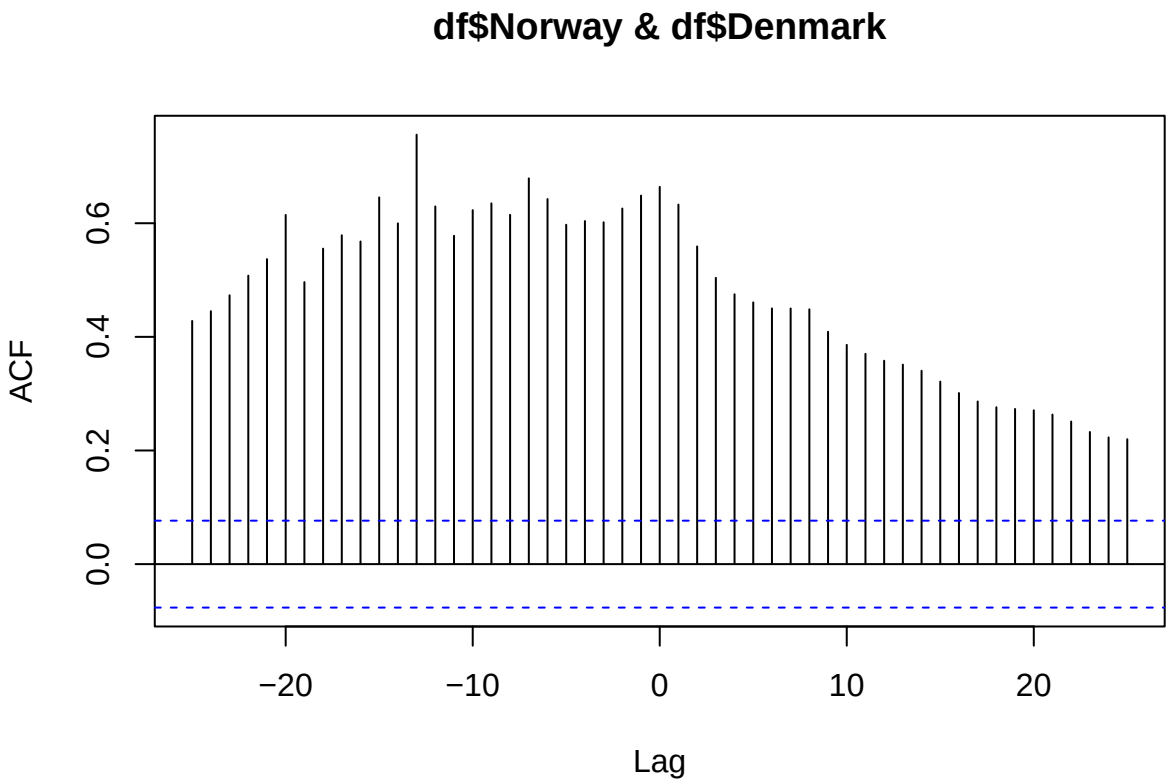

```
ccf(df$Denmark, df$Italia)
```



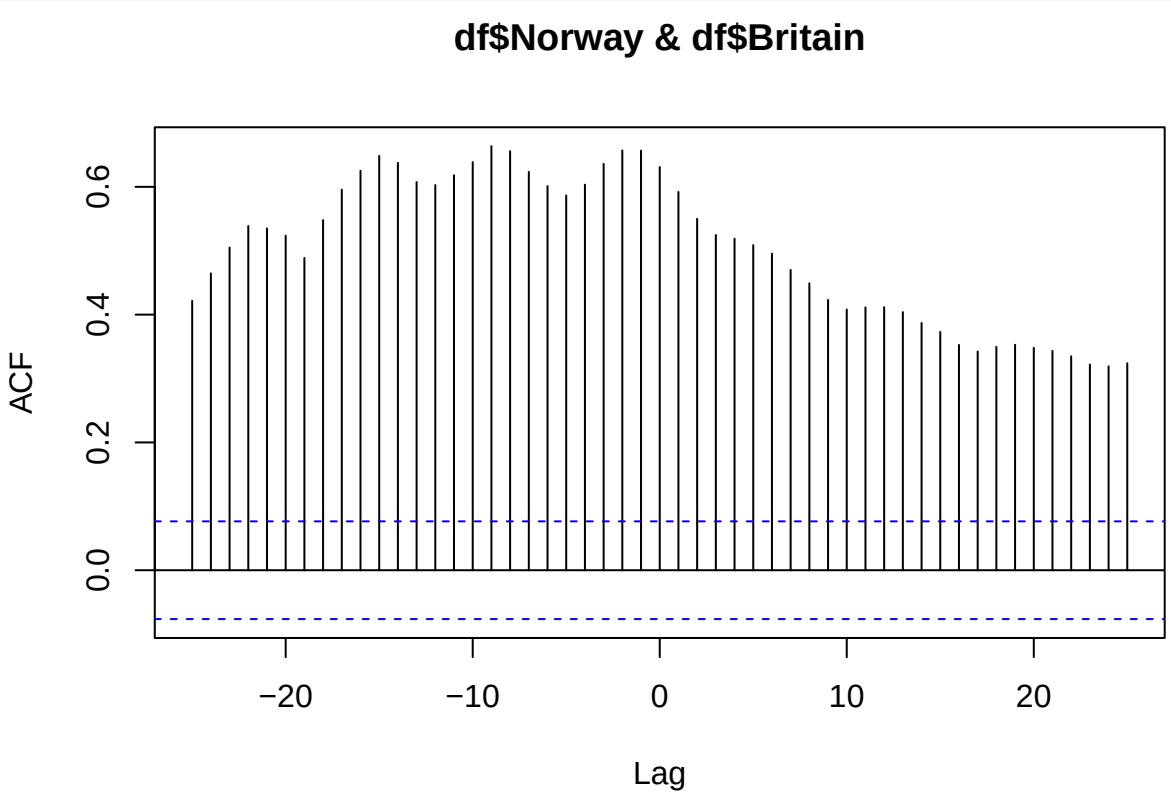

```
ccf(df$Denmark, df$India)
```



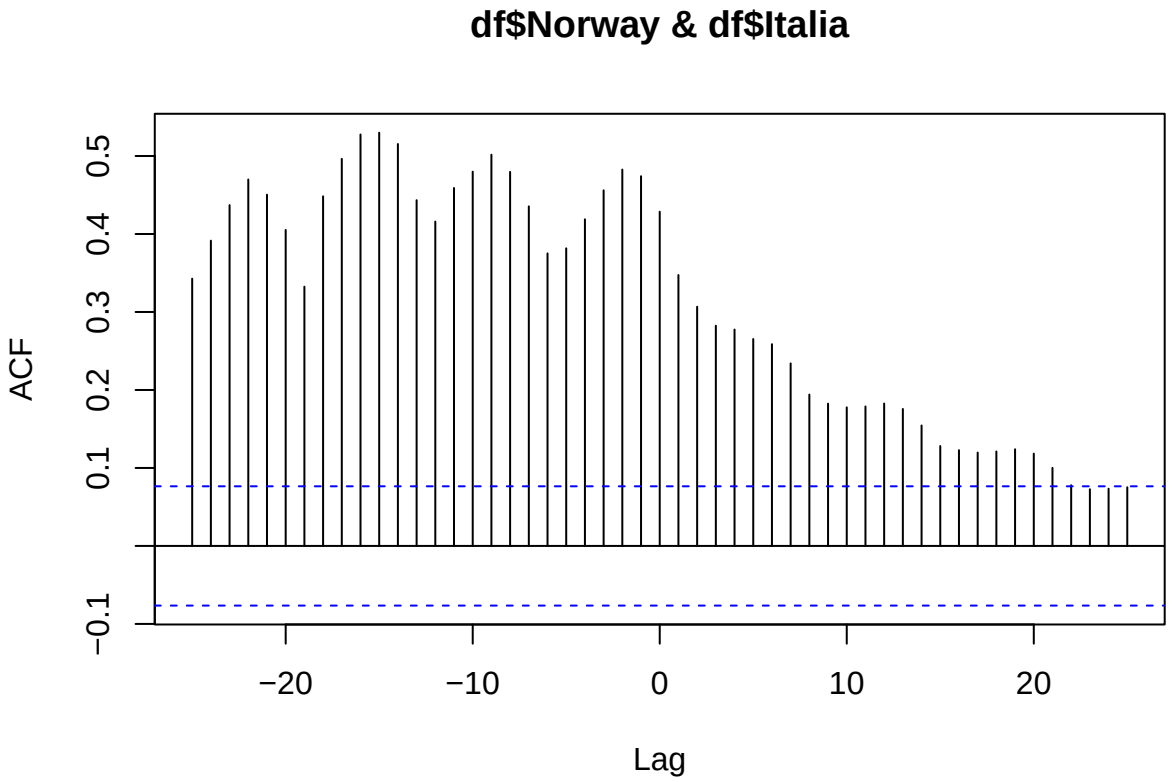

```
# Norway
ccf(df$Norway, df$Denmark)
```



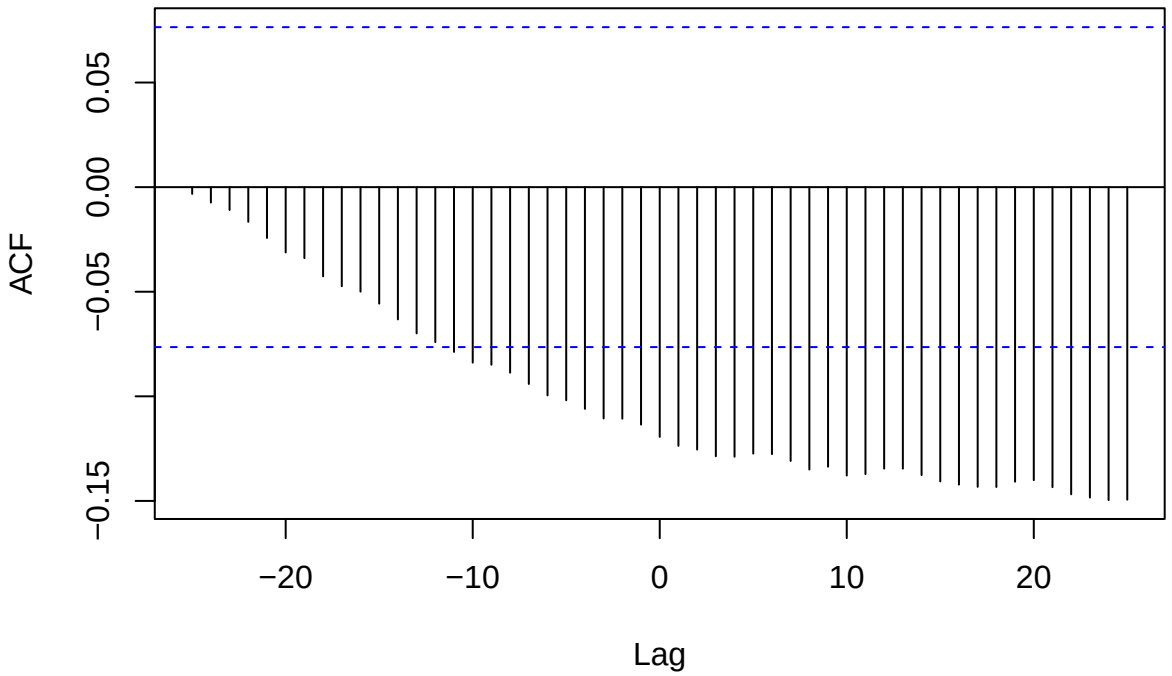

```
ccf(df$Norway, df$Britain)
```



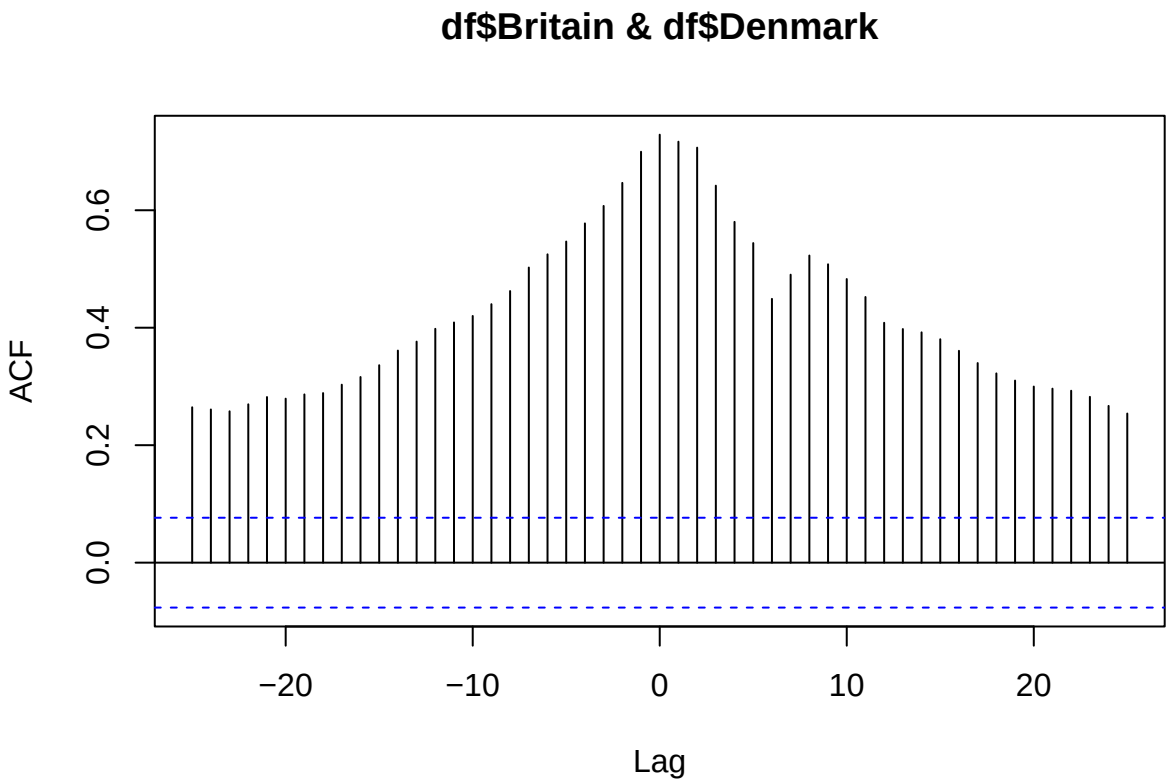

```
ccf(df$Norway, df$Italia)
```



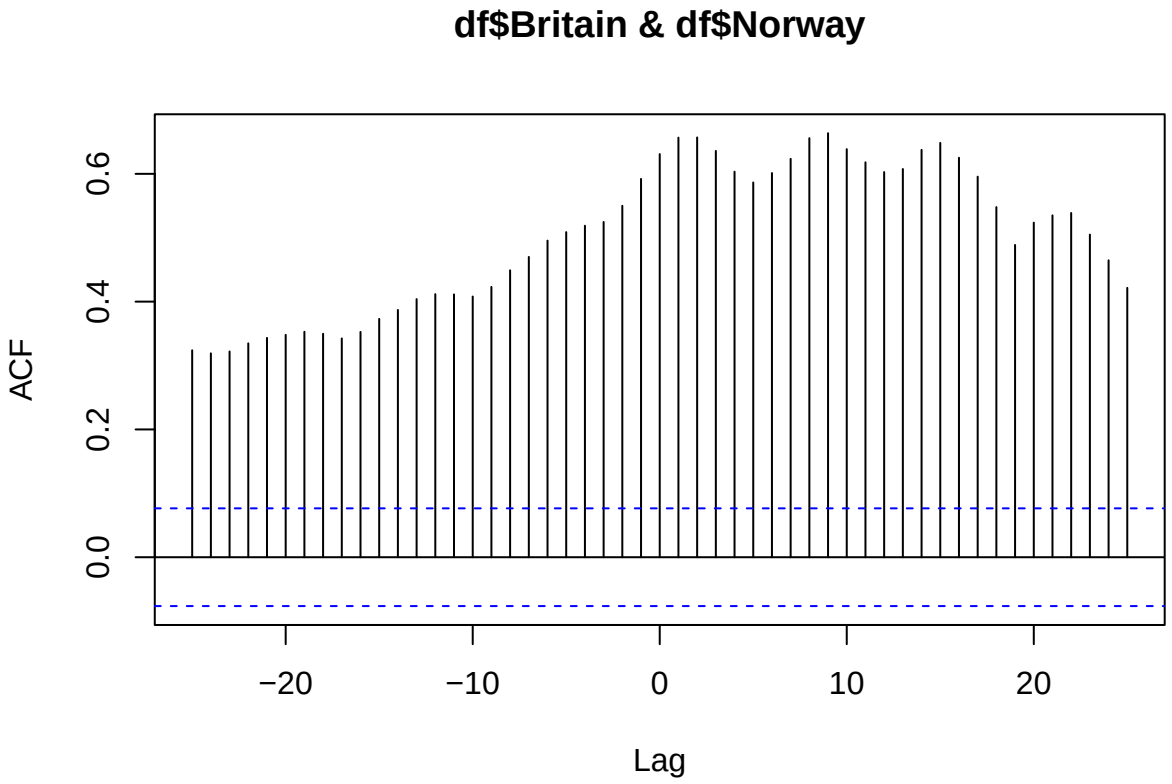

```
ccf(df$Norway, df$India)
```



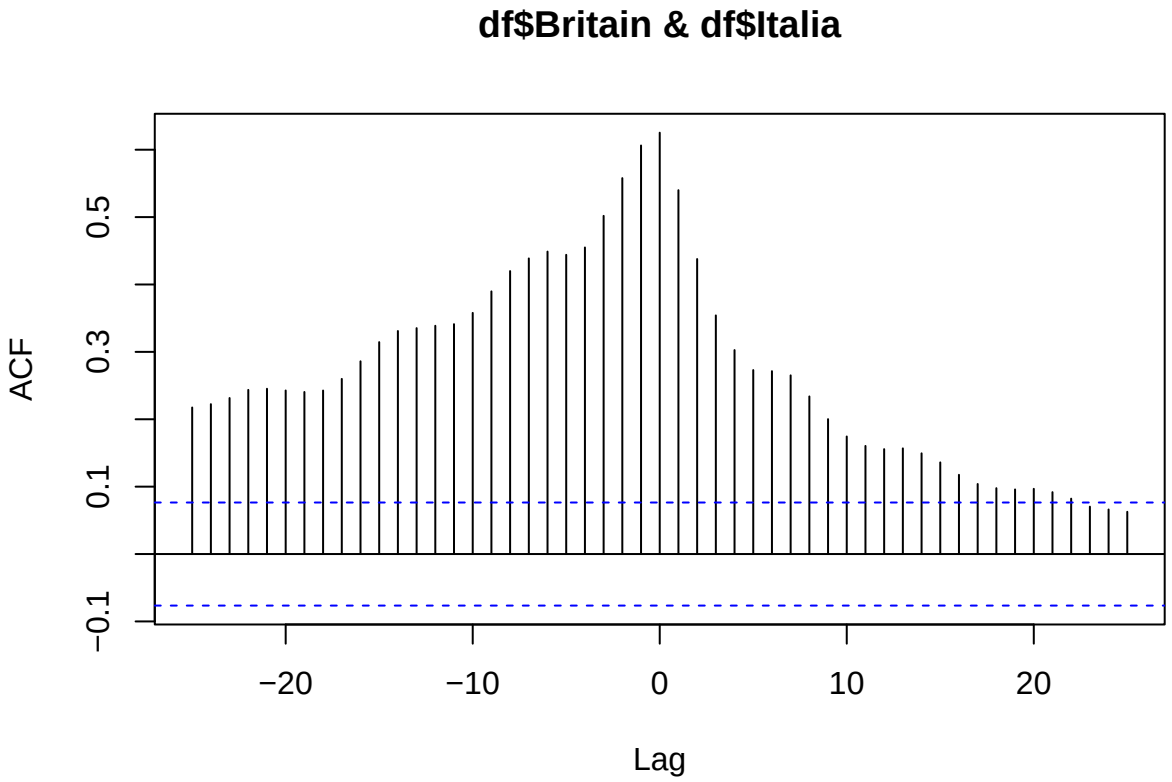

```
# Britain
ccf(df$Britain, df$Denmark)
```



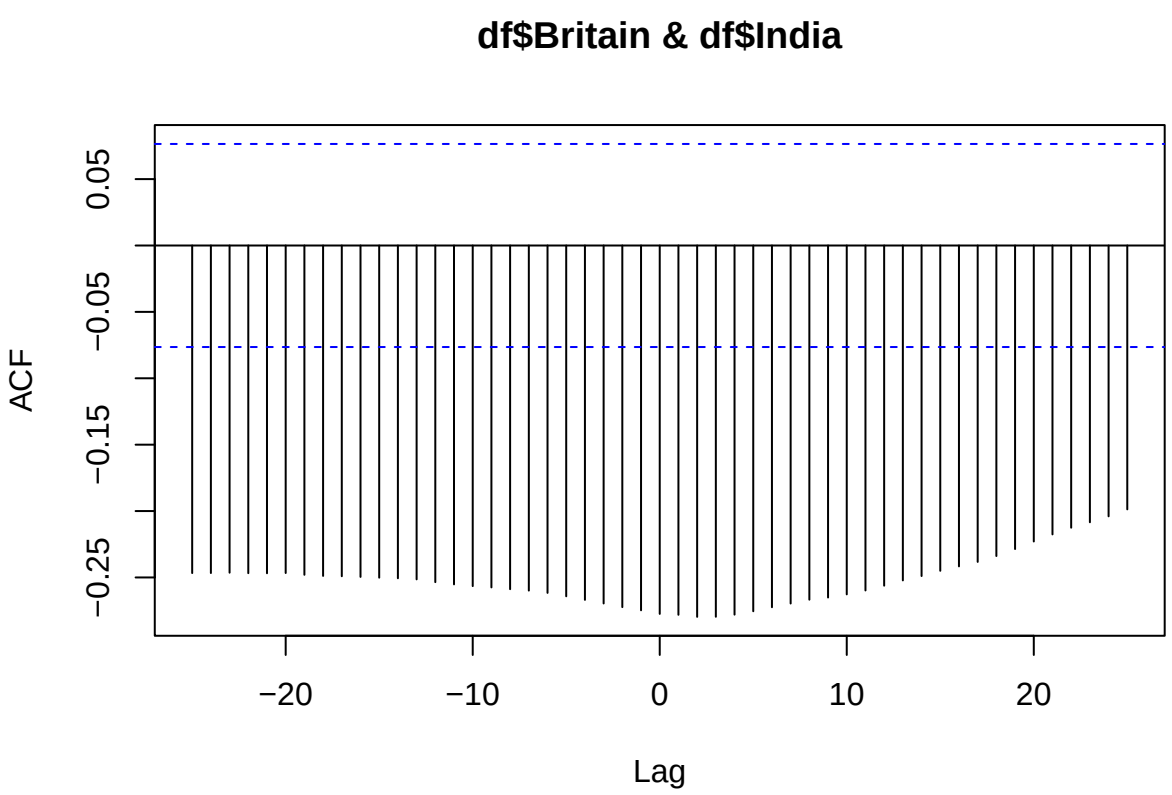

```
ccf(df$Britain, df$Norway)
```



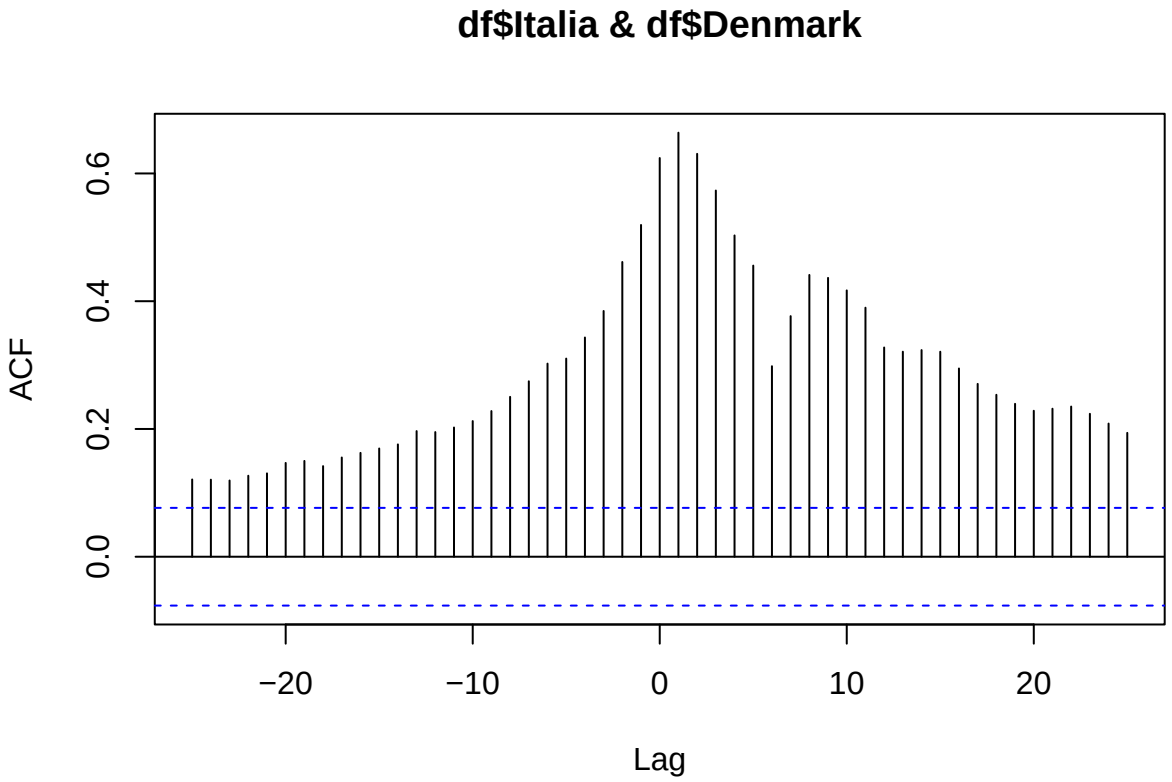

```
ccf(df$Britain, df$Italia)
```



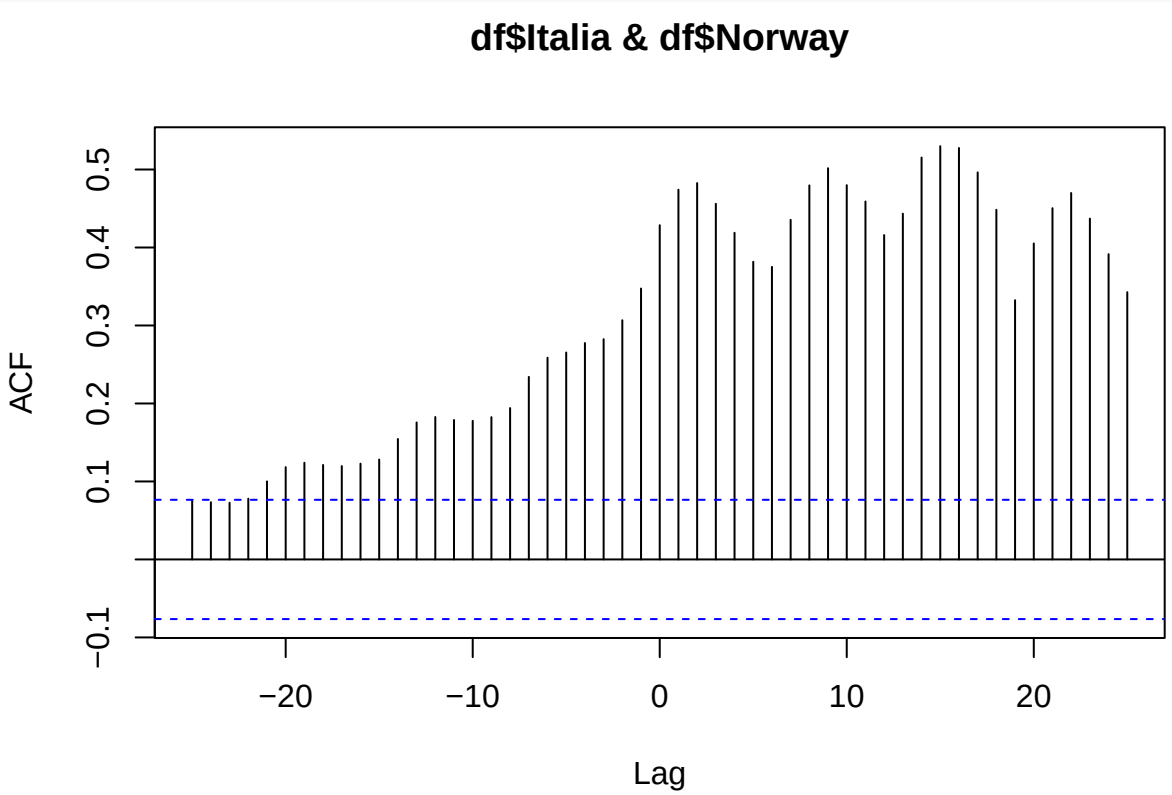

```
ccf(df$Britain, df$India)
```



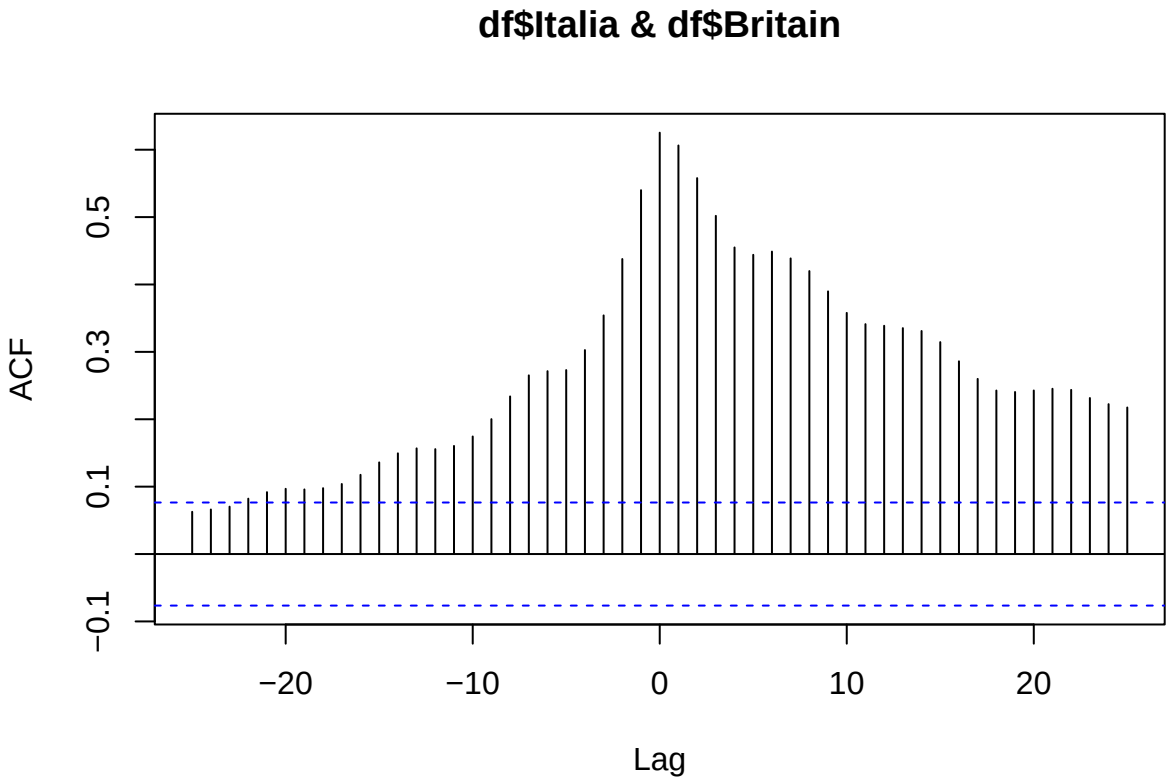

```
# Italia
ccf(df$Italia, df$Denmark)
```



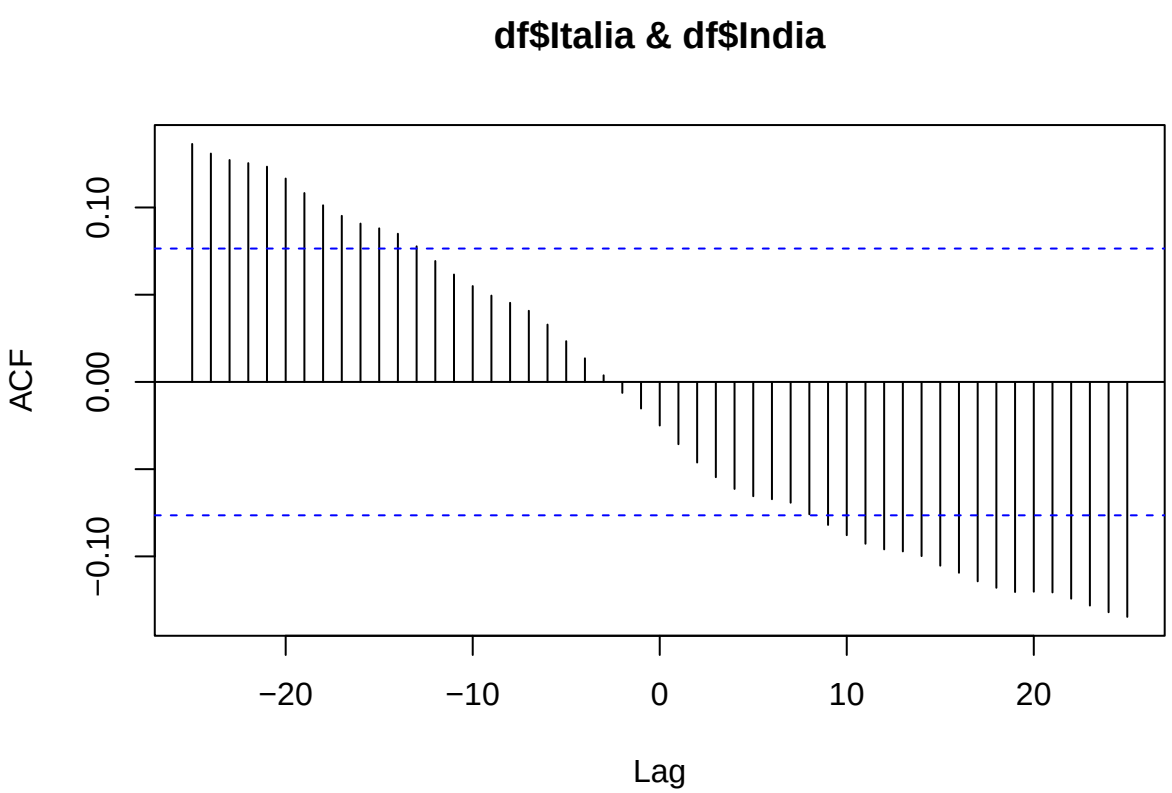

```
ccf(df$Italia, df$Norway)
```



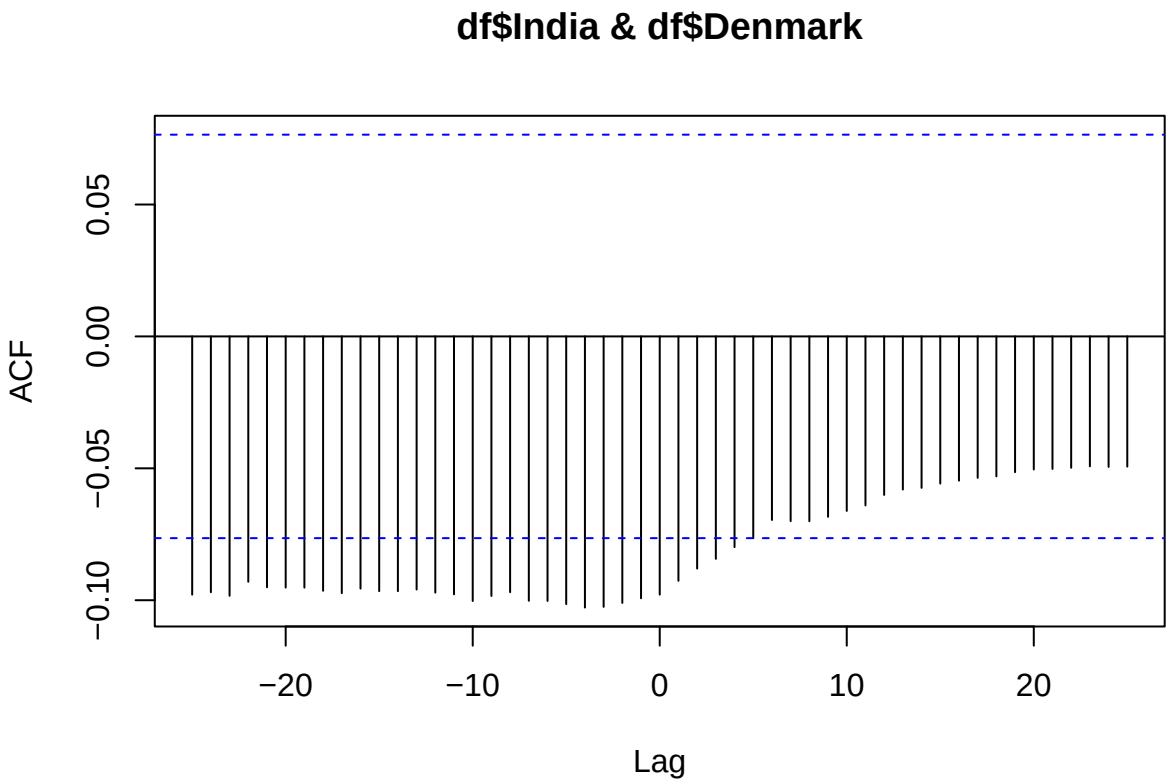

```
ccf(df$Italia, df$Britain)
```



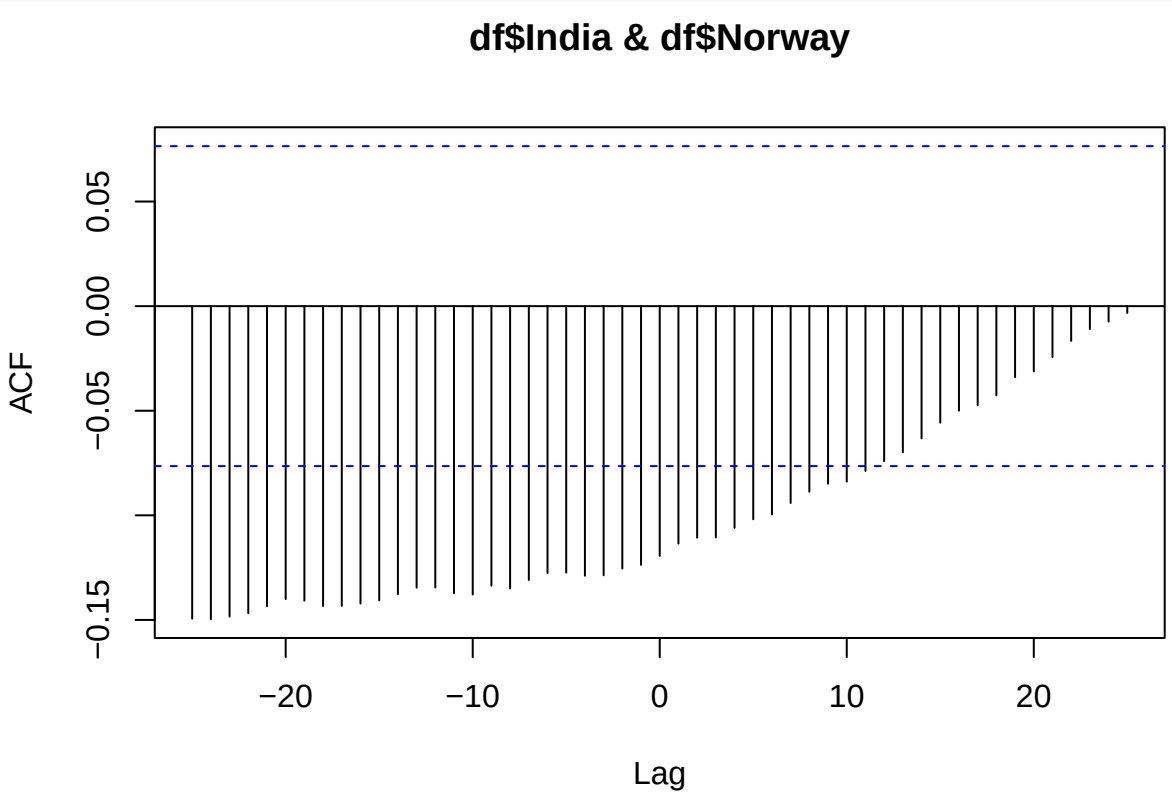

```
ccf(df$Italia, df$India)
```



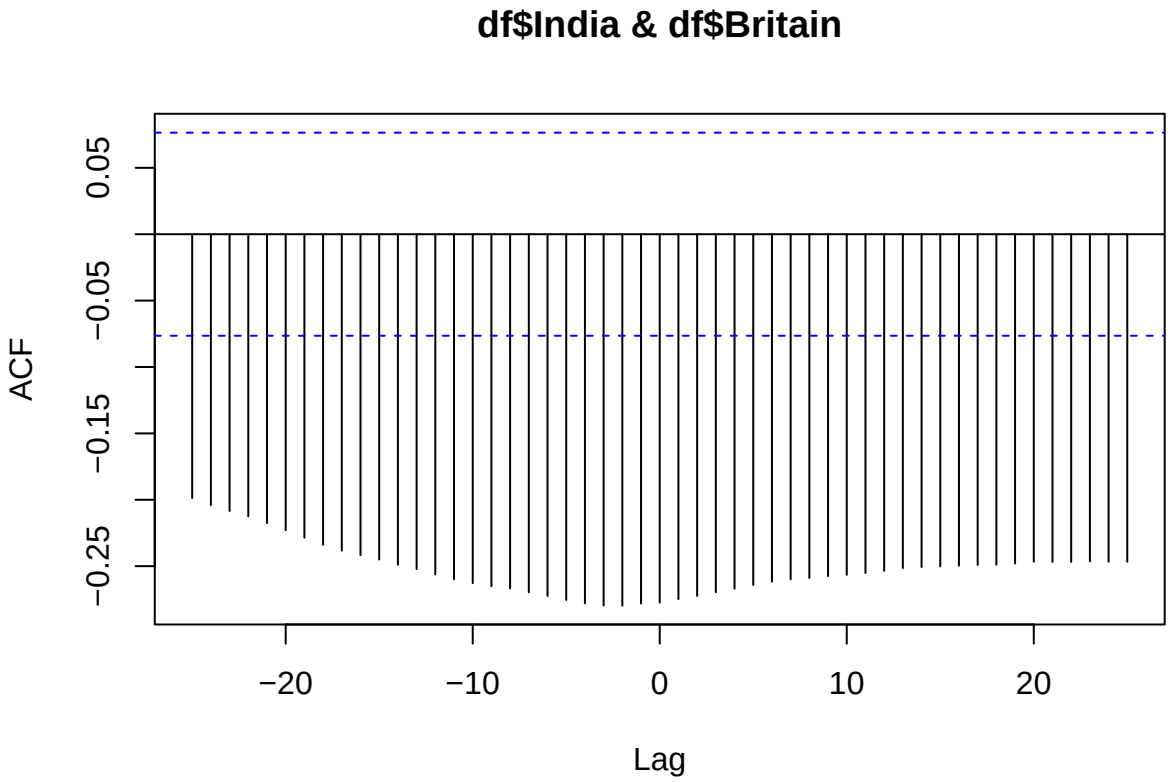

```
# India
ccf(df$India, df$Denmark)
```



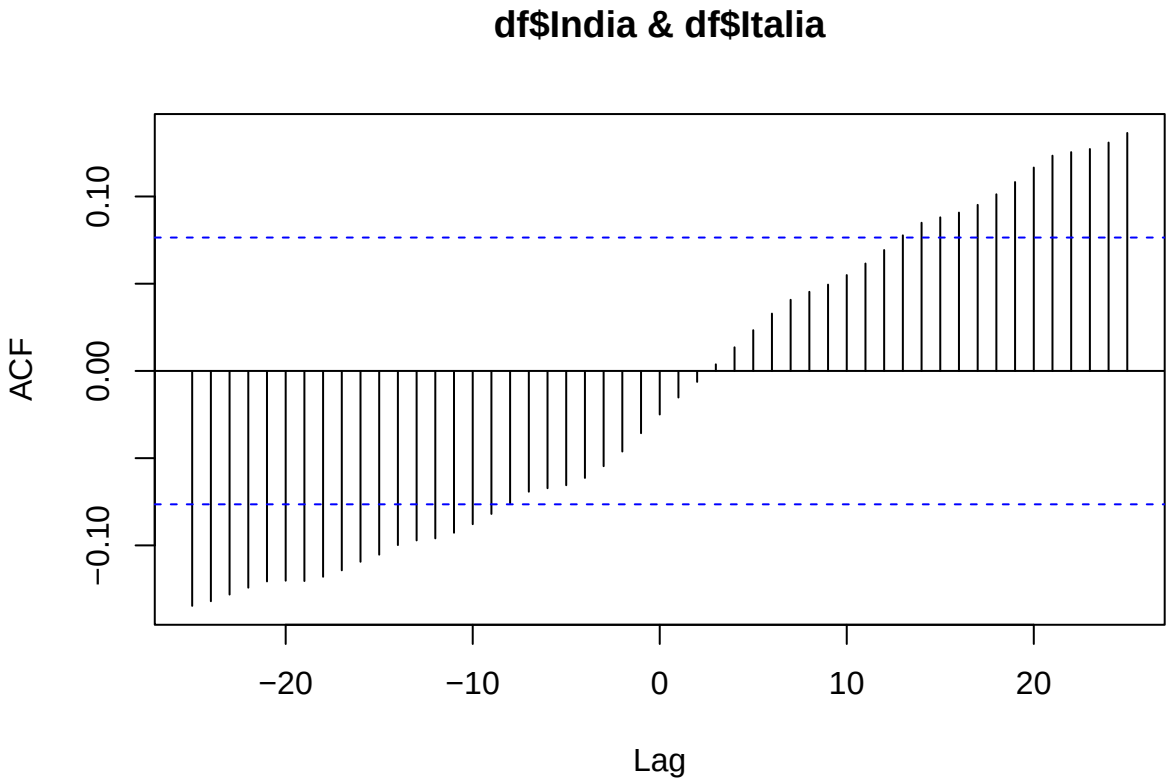

```
ccf(df$India, df$Norway)
```




```
ccf(df$India, df$Britain)
```




```
ccf(df$India, df$Italia)
```



From the correlation matrix it can be seen that Denmark, Norway, Great Britain and Italia show high, positive correlation with each other. The value for the correlation coefficient between Norway and Italia is 0.429 and above 0.6 for the other countries mentioned. Sweden show relatively low correlation with the other

countries. India is negatively correlated with the other countries, except from Sweden.

Looking at the autocorrelation plots for each country; all the countries, except Sweden, have spikes above the line for being statistically significant. This mean the measurements in the series are positively autocorrelated for lags up to 28 days. The autocorrelation plot for Sweden indicate autocorrelation for lags up to 3 days, then it alternates between 2 days of no autocorrelation and 5 days with autocorrelation.

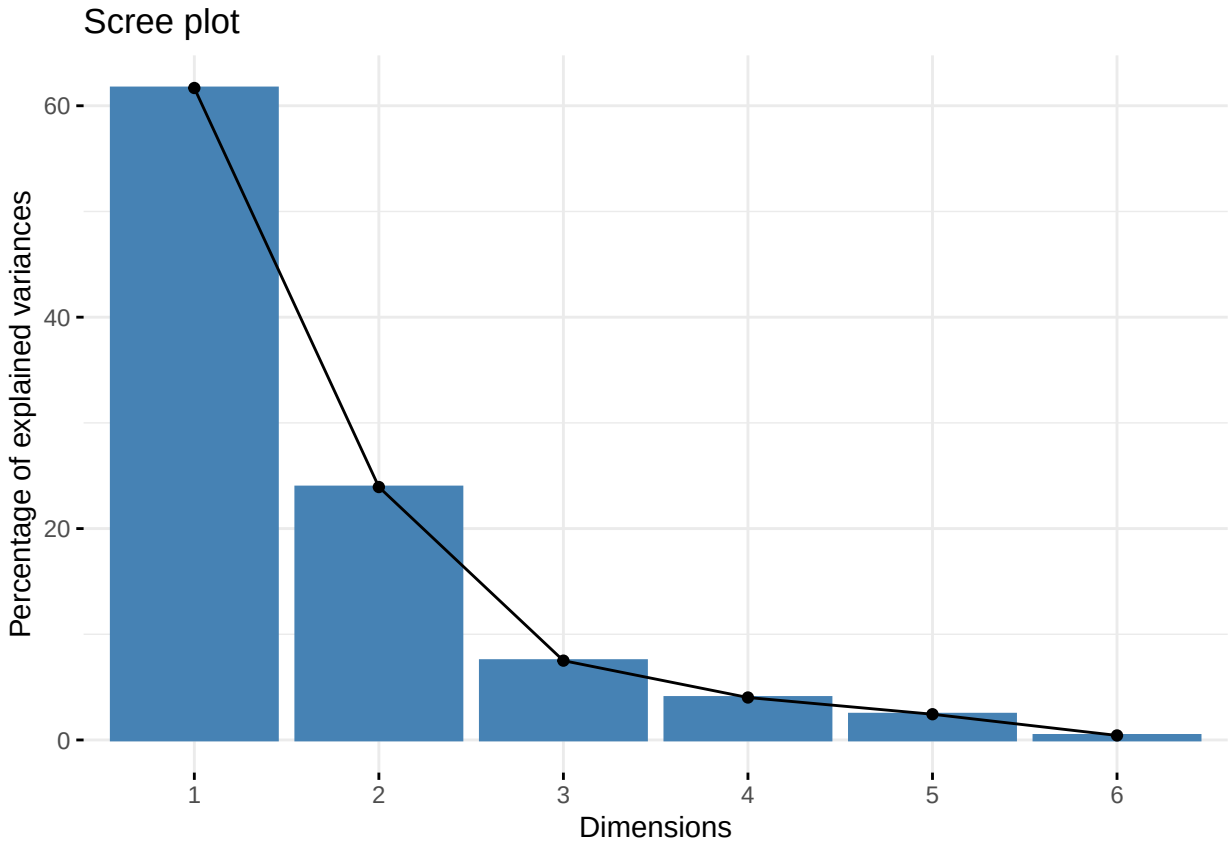
Due to the amount of plots, not all the cross-correlation plots are displayed. Denmark, Norway, Great Britain and Italia show statistically significant positive cross-correlation for the entire period visualized; -25 to +25 days lag. Sweden shows cross-correlation between Italia and with India up to a lag of 13 days. With Denmark, Norway and Great Britain, Sweden shows cyclical periods of cross-correlation and no cross-correlation. The plots for India indicate positive cross-correlation with Sweden, and both negative and positive cross-correlation with Italia. For the other countries, the plots show that India has negative cross-correlation for the entire period visualized, and mostly significantly low values.

c)

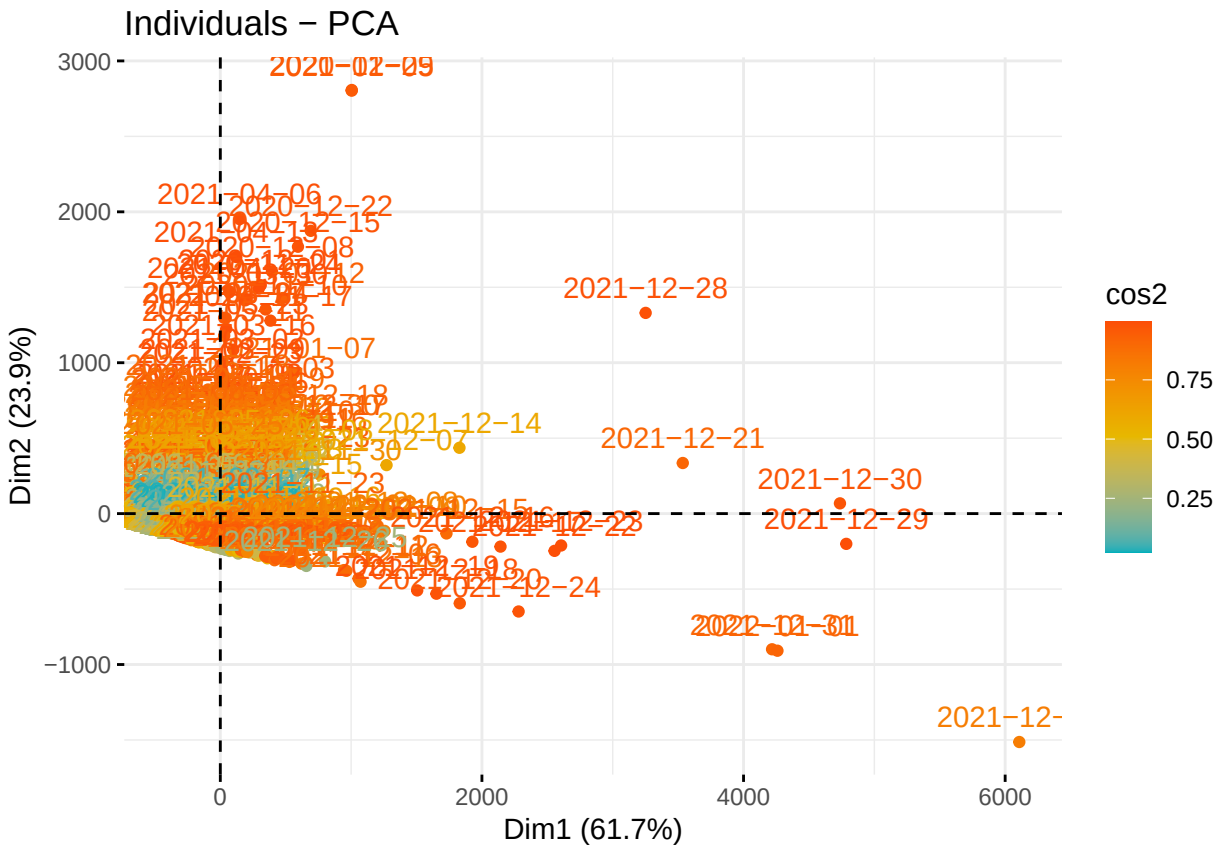
```
# Transformations
# ------------------------------------------------------------------------------

# Copy of dataframe
dfT <- df
# Setting date as index
rownames(dfT) <- dfT$Date
dfT <- select(dfT, Sweden, Denmark, Norway, Britain, Italia, India)

# Remove missing values (even though there should not be any)
dfT <- na.omit(dfT)

# PCA transform
df.pca <- princomp(dfT, scale=TRUE)

# Scores and loadings
#df.pca$scores
#df.pca$loadings

# Summary and plot
# plot(df.pca)
# biplot(df.pca)
summary(df.pca)
```

```
## Importance of components:
##                             Comp.1      Comp.2       Comp.3       Comp.4
## Standard deviation     613.7147403 382.2627759 214.07667047 156.64640995
## Proportion of Variance   0.6168102   0.2392999   0.07505111   0.04018457
## Cumulative Proportion    0.6168102   0.8561101   0.93116120   0.97134578
##                              Comp.5       Comp.6
## Standard deviation     121.97500285 51.179731727
## Proportion of Variance   0.02436465  0.004289577
## Cumulative Proportion    0.99571042  1.000000000
```

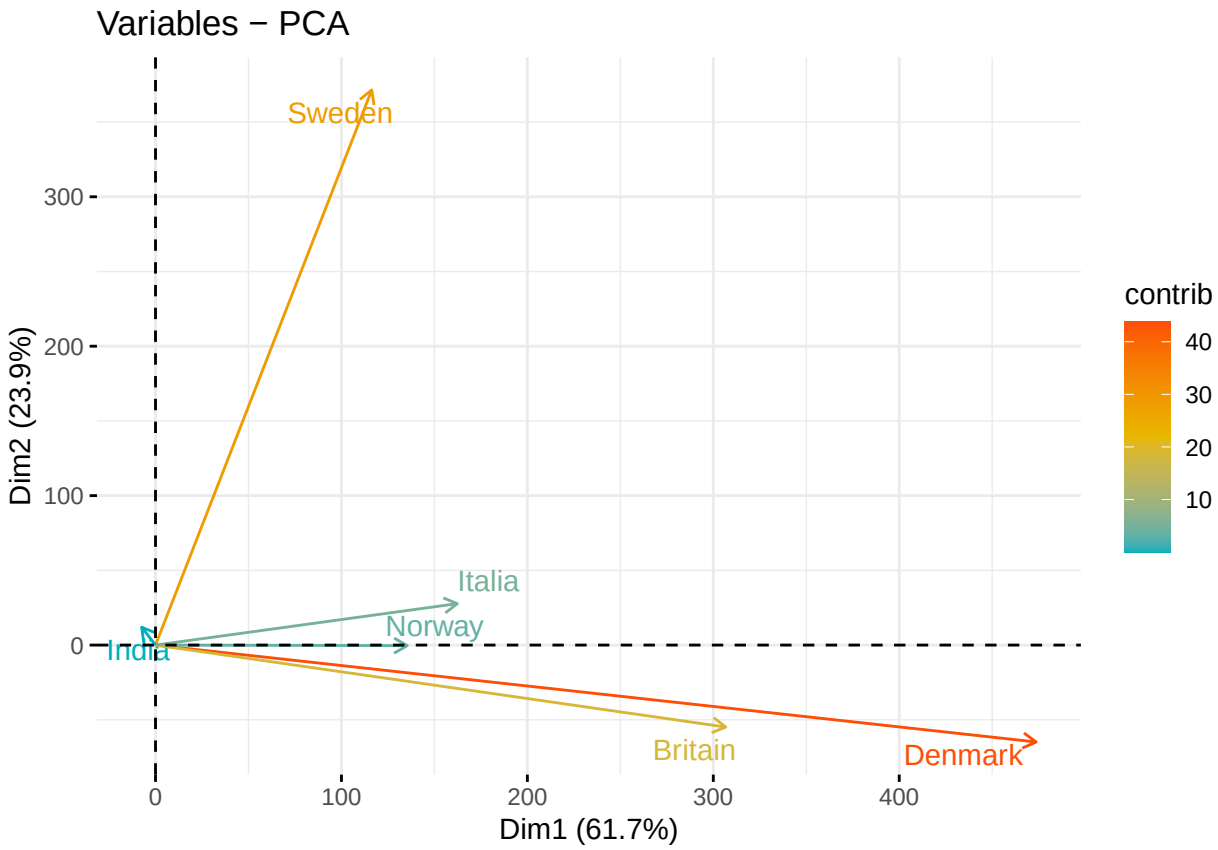
```
# Explained variance
fviz_eig(df.pca)
```



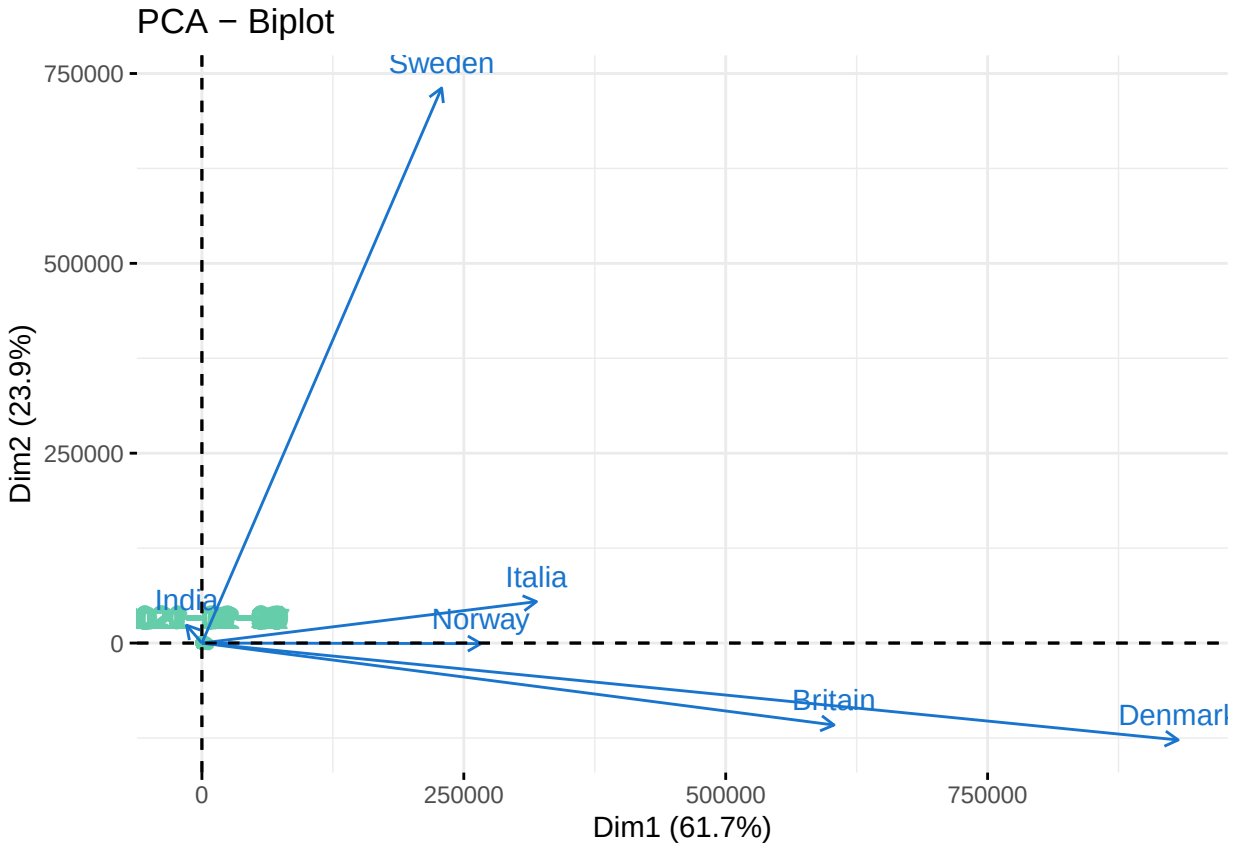

```
# Plot of scores
fviz_pca_ind(df.pca,
             col.ind = 'cos2',
             gradient.cols = c('#00AFBB', '#E7B800', '#FC4E07'),
             repel = FALSE
)
```




```
# Plot of loadings
fviz_pca_var(df.pca,
             col.var = 'contrib',
             gradient.cols = c('#00AFBB', '#E7B800', '#FC4E07'),
             repel = TRUE
)
```




```
# Biplot
fviz_pca_biplot(df.pca, repel = FALSE,
                col.var = '#1874CD',
                col.ind = '#66CDAA'
)
```

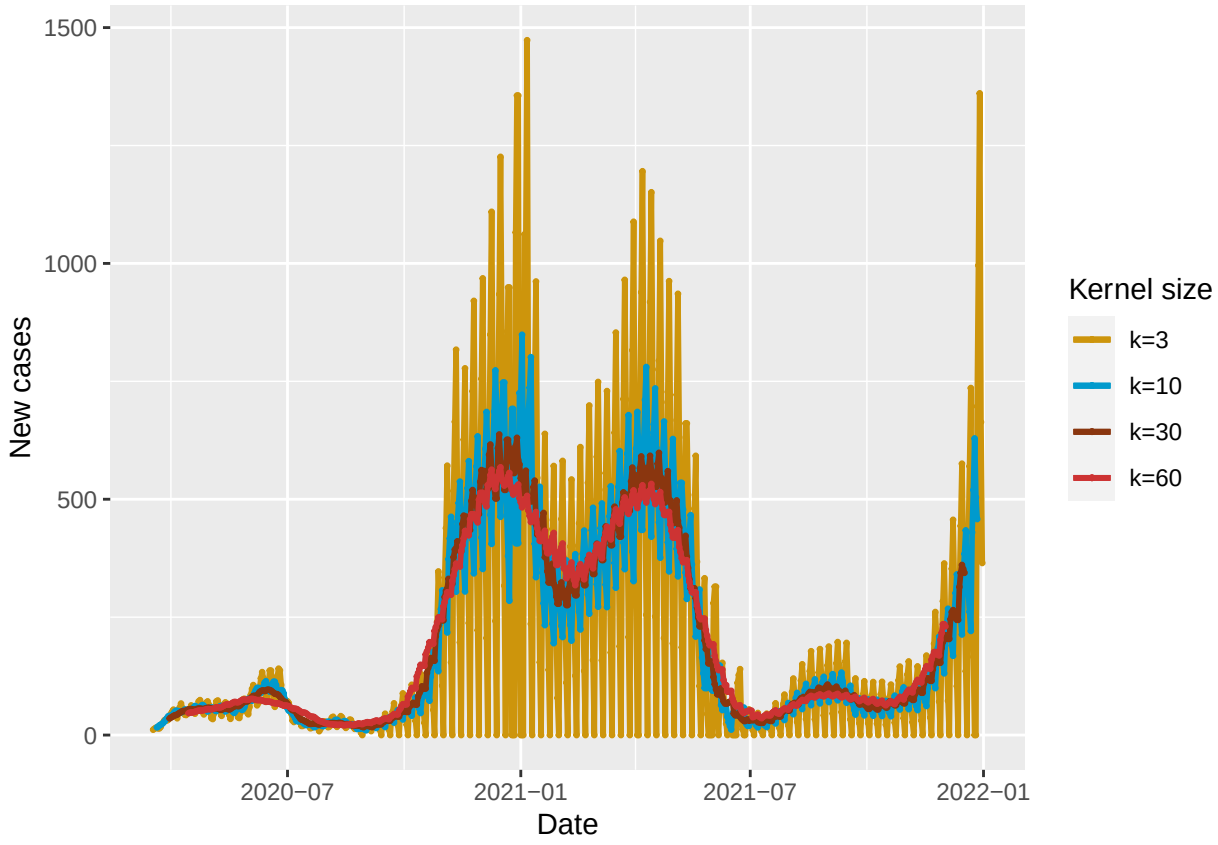


The principal components analysis shows that over 50% of the variance in the data can be explained by the first principal component. The plot of the loadings show that Denmark, Norway, Great Britain and Italy are correlated, as they lie in the same direction in along the axis for the first principal component. Denmark is contributing to most of the variance in the first principal component. India contributes to a very small amount of the variance in the dataset. Sweden can be considered different compared to the other countries and is responsible for almost all the variance in the second component.

d)

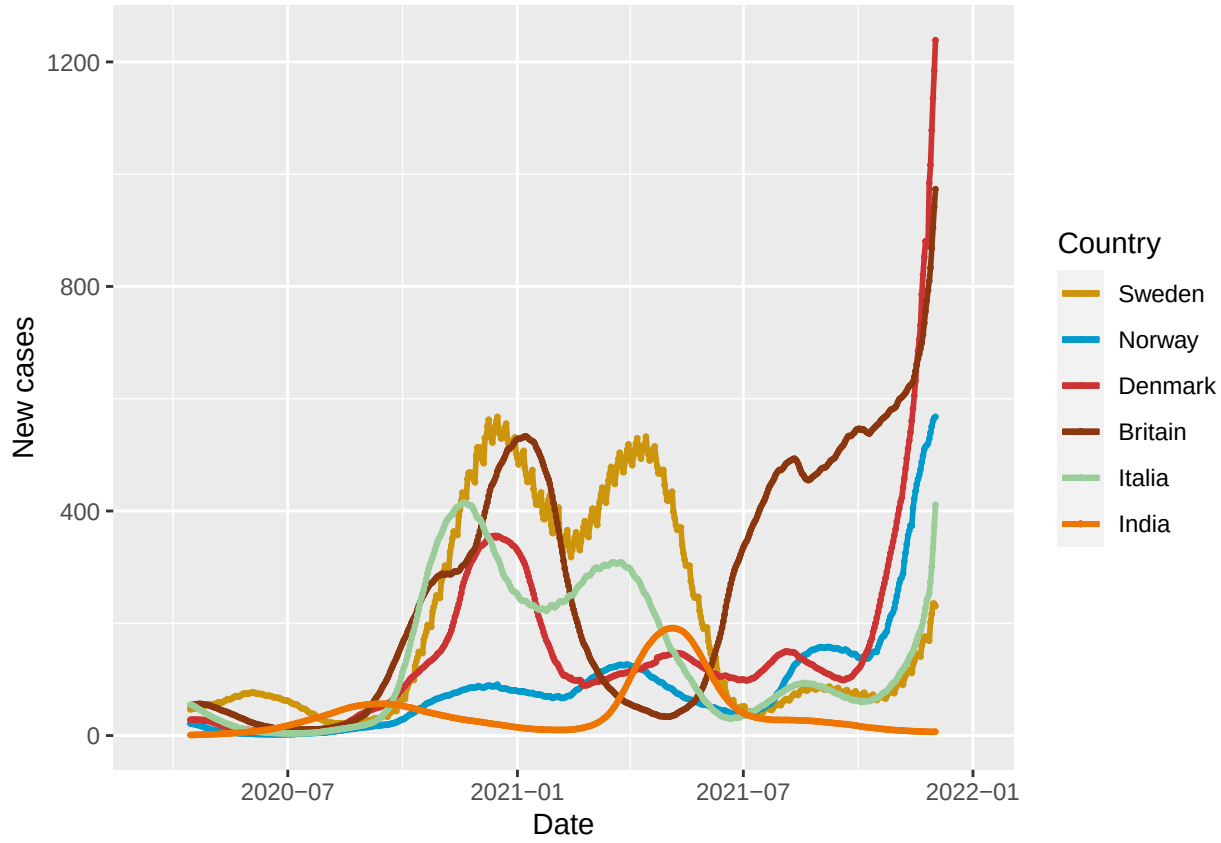
```
# Smoothing
# ------------------------------------------------------------------------------
# Sweden
swe1 <- zoo::rollmean(df$Sweden, k=3, fill=NA)
swe2 <- zoo::rollmean(df$Sweden, k=10, fill=NA)
swe3 <- zoo::rollmean(df$Sweden, k=30, fill=NA)
swe4 <- zoo::rollmean(df$Sweden, k=60, fill=NA)
dfSweden <- data.frame(date, swe1, swe2, swe3, swe4)


# Line plot of smoothing with different value for the kernel size
colors <- c('k=3'='#CD950C', 'k=10'='#009ACD', 'k=30'='#8A360F',
            'k=60'='#CD3333')

ggplot(dfSweden, aes(x=date)) +
  geom_line(aes(y=swe1, color='k=3'), size=1.0) +
  geom_point(aes(y=swe1, color='k=3'), size=0.5) +
```

```
  geom_line(aes(y=swe2, color='k=10'), size=1.0) +
  geom_point(aes(y=swe2, color='k=10'), size=0.5) +
  geom_line(aes(y=swe3, color='k=30'), size=1.0) +
  geom_point(aes(y=swe3, color='k=30'), size=0.5) +
  geom_line(aes(y=swe4, color='k=60'), size=1.0) +
  geom_point(aes(y=swe4, color='k=60'), size=0.5) +
  labs(x='Date', y='New cases', color='Kernel size') +
  scale_color_manual(values=colors)
```




```
#
# ------------------------------------------------------------------------------
date <- df$Date
dfS <- select(df, Sweden, Denmark, Britain, Norway, Italia, India)
dfS <- zoo::rollmean(dfS, k=60, fill=NA)

# New data.frame
dfSm <- data.frame(date, dfS[,'Sweden'], dfS[,'Denmark'],
                   dfS[,'Norway'], dfS[,'Britain'],
                   dfS[,'Italia'], dfS[,'India'])
# Renaming data.frame
names(dfSm) <- c('Date', 'Sweden', 'Denmark', 'Norway',
                 'Britain', 'Italia', 'India')


# Line plot of smoothed time-series
colors <- c('Sweden'='#CD950C', 'Norway'='#009ACD', 'Denmark'='#CD3333',
            'Britain'='#8A360F', 'Italia'='#9BCD9B', 'India'='#EE7600')
```

```
p <- ggplot(dfSm, aes(x=date)) +
  geom_line(aes(y=Sweden, color='Sweden'), size=1.0) +
  geom_point(aes(y=Sweden, color='Sweden'), size=0.5) +
  geom_line(aes(y=Norway, color='Norway'), size=1.0) +
  geom_point(aes(y=Norway, color='Norway'), size=0.5) +
  geom_line(aes(y=Denmark, color='Denmark'), size=1.0) +
  geom_point(aes(y=Denmark, color='Denmark'), size=0.5) +
  geom_line(aes(y=Britain, color='Britain'), size=1.0) +
  geom_point(aes(y=Britain, color='Britain'), size=0.5) +
  geom_line(aes(y=Italia, color='Italia'), size=1.0) +
  geom_point(aes(y=Italia, color='Italia'), size=0.5) +
  geom_line(aes(y=India, color='India'), size=1.0) +
  geom_point(aes(y=India, color='India'), size=0.5) +
  labs(x='Date', y='New cases', color='Country') +
  scale_color_manual(values=colors)
p
```

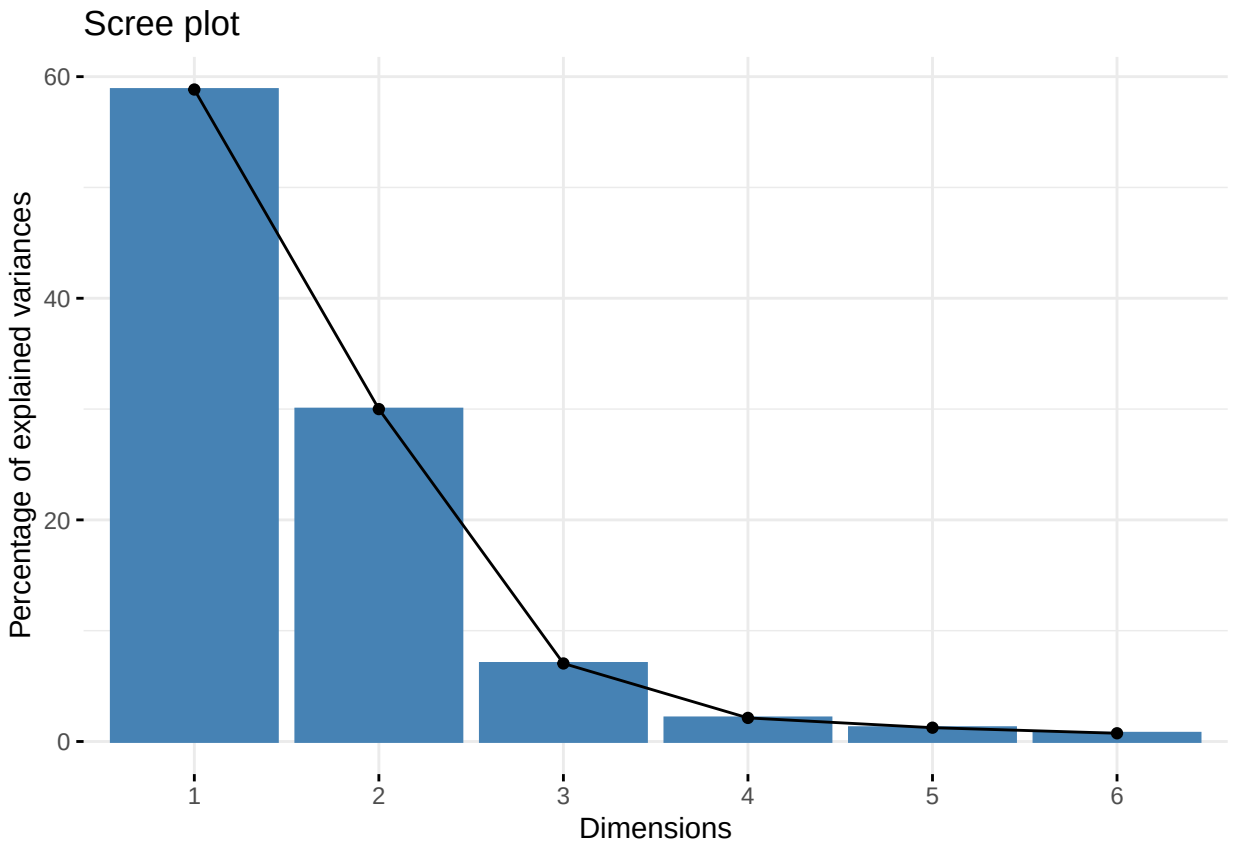



```
# PCA on smoothed time-series
# ------------------------------------------------------------------------------

# Copy of dataframe
dfT <- dfSm
# Setting date as index
rownames(dfT) <- dfT$Date
dfT <- select(dfT, Sweden, Denmark, Norway, Britain, Italia, India)

# Remove missing values
```

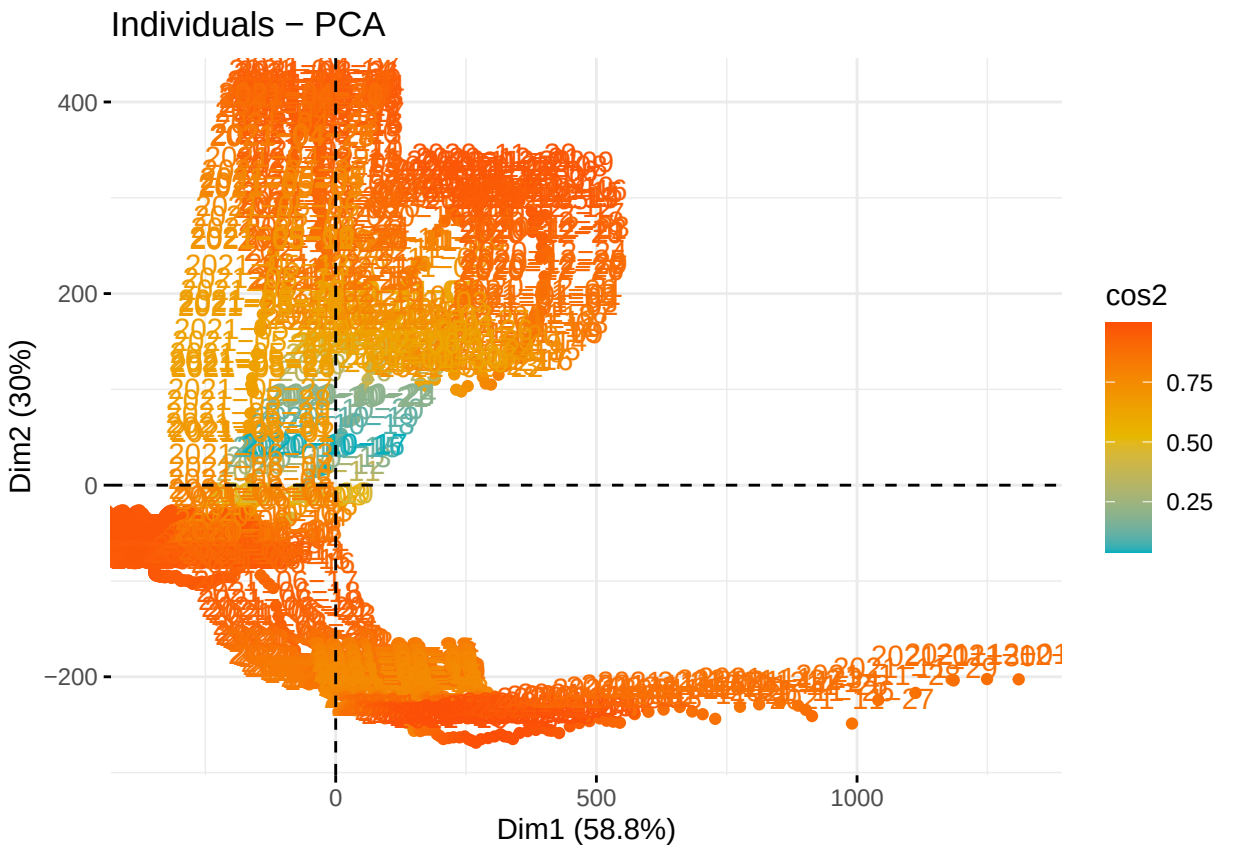
```r
dfT <- na.omit(dfT)

# PCA transform
df.pca <- princomp(dfT, scale=TRUE)

# Scores and loadings
#df.pca$scores
#df.pca$loadings

# Summary and plot
# plot(df.pca)
# biplot(df.pca)
summary(df.pca)
```

```
## Importance of components:
##                             Comp.1      Comp.2      Comp.3      Comp.4
## Standard deviation     282.8338115 201.9528270 97.85371071 53.80844438
## Proportion of Variance   0.5883794   0.2999816  0.07042866  0.02129588
## Cumulative Proportion    0.5883794   0.8883610  0.95878969  0.98008558
##                             Comp.5       Comp.6
## Standard deviation     41.18233558 31.804761717
## Proportion of Variance  0.01247432  0.007440106
## Cumulative Proportion   0.99255989  1.000000000
```

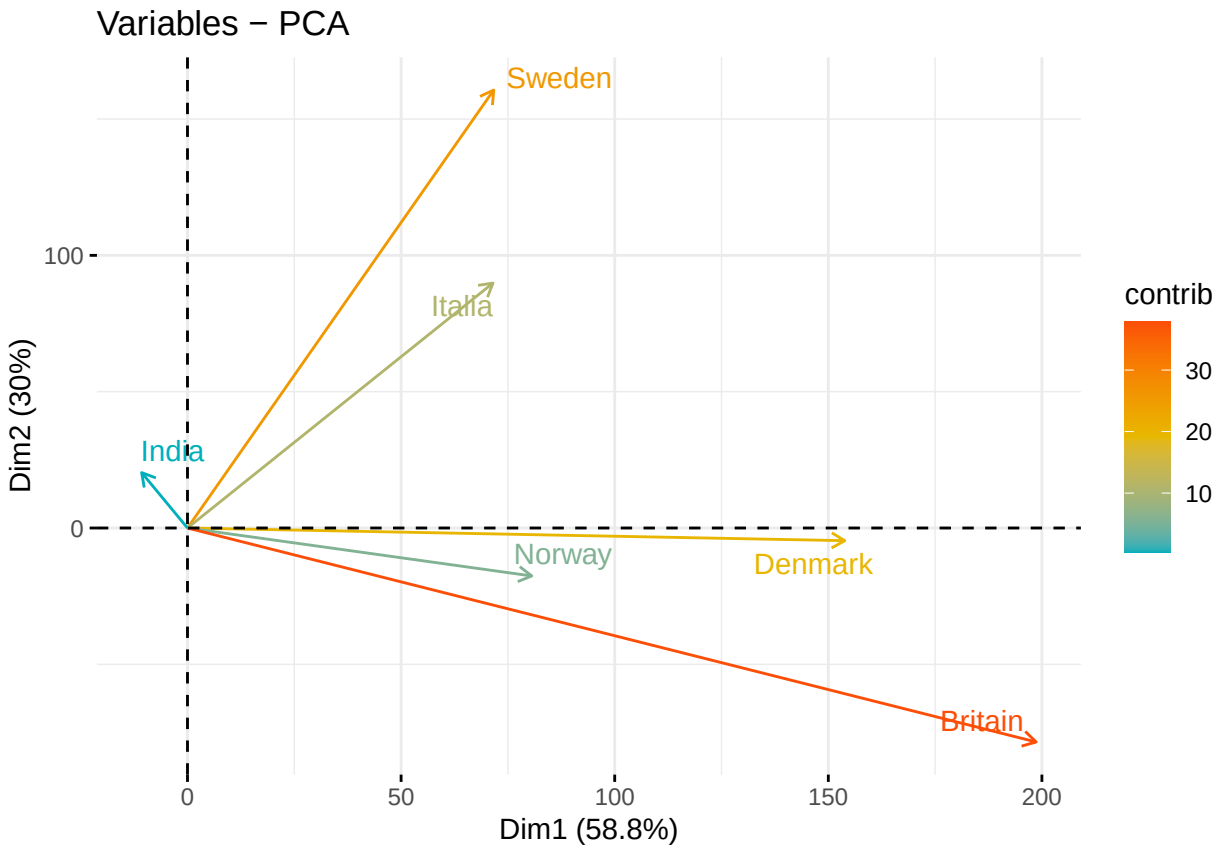
```r
# Explained variance
fviz_eig(df.pca)
```



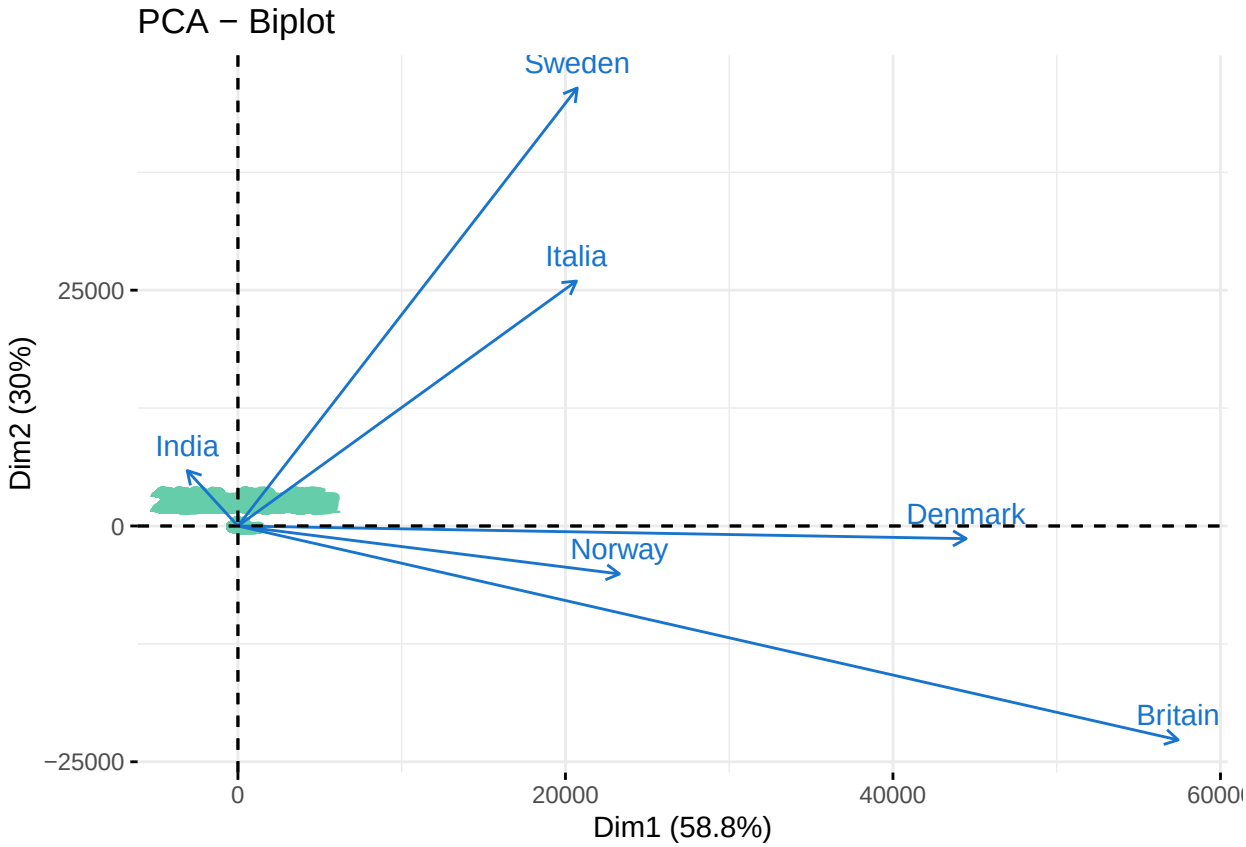
```
# Plot of scores
fviz_pca_ind(df.pca,
             col.ind = 'cos2',
             gradient.cols = c('#00AFBB', '#E7B800', '#FC4E07'),
             repel = FALSE
)
```




```
# Plot of loadings
fviz_pca_var(df.pca,
             col.var = 'contrib',
             gradient.cols = c('#00AFBB', '#E7B800', '#FC4E07'),
             repel = TRUE
)
```




```
# Biplot
fviz_pca_biplot(df.pca, repel = FALSE,
                col.var = '#1874CD',
                col.ind = '#66CDAA'
)
```



Different sizes for the kernel was tested for smoothing the series, with the example shown for Sweden. k=3 showed to be too low to filter away the measurements that were set to 0 and a kernel size of 100 could result in smoothing the curve to much. An appropriate kernel size could be 10, 30 or 60. By smoothing the entire series and looking at the line plot for the smoothed series, it showed that a kernel size of 7 or higher also filtered away the negative values. When performing the principal component analysis, the variance explained by the first component decreased some. After smoothing, the difference between Italia and Denmark, Norway and Great Britain decreased, but Italia showed to be more similar to Sweden. India still lies in the opposite direction along the first principal component after smoothing the series.

Exercise 4

a)

The autocorrelation for time series is $cor(x_t, B^k(x_t))$ where $(k < t \leq t_{max})$.
The inverse autocorrelation for time series is $cor(x_t, B^{-k}(x_t))$ where $(0 < t \leq t_{max} - k)$.
We also have that:

$$B^k(x_t) = x_{t-k}$$
$$B^-k(x_t) = x_{t+k}$$

To show that the inverted autocorrelation function is equal to the autocorrelation we have that:

$$\begin{aligned} acf(k) &= cor(x_t, B^k(x_t)) \\ &= cor(x_{t+k}, B^k(x_{t+k})) \end{aligned}$$

Replace for $B^{-k}(x_t) = x_{t+k}$

$$= cor(B^{-k}(x_t), B^k(B^{-k}(x_{t+k})$$

B^{-k} and B^k cancel each other out

$$= cor(B^{-k}(x_t), x_t) = \tilde{acf}$$

Thus the autocorrelation for time series xt is same as the inverse autocorrelation of it.

b)

The correlation between the two time series´, x and y, can be defined as $cor(x_t, B^k(y_t))$ where $(t \leq t_{max} - k)$.
The inverse correlation is then $cor(x_t, B^{-k}(y_t))$.

Use the same approach as in a) in order to check if the statement also holds for cross correlation:

$$\begin{aligned} ccf(k) &= cor(x_t, B^k(y_t)) \\ &= cor(x_{t+k}, B^k(y_{t+k})) \end{aligned}$$

Replace for $B^{-k}(x_t) = x_{t+k}$

$$= cor(B^{-k}(x_t), B^k(B^{-k}(y_t)))$$
$$= cor(B^{-k}(x_t), y_t$$

We then have that:

$$cor(B^{-k}(x_t), y_t) \neq cor(x_t, B^{-k}(y_t))$$

Meaning that the statement does not hold for cross-correlations. When we only have one time series, it does not matter if B^k or B^{-k} since the level of correlation will not be affected. When we have cross-correlation, meaning that we have two different time series, the level of correlation is affected. We see this in the $cor(B^{-k}(x_t), y_t) \neq cor(x_t, B^{-k}(y_t))$ where B^{-k} is on both sides of the $\neq$. If we instead had that $cor(B^k(x_t), y_t) = cor(x_t, B^{-k}(y_t))$ then the statement would hold, but that´s not the case. It is also logical that a backshift operator with similar exponent-signs on both sides of the $\neq$ does not make sense. Essentially, this would mean that the level of correlation would not be affected by the intervals of the two time series included in the calculation of the correlation. Which does not make sense.

c)

- Time series 1 : The ACF of Time Series 1 (TS1) is ACF3. A visual inspection of TS1 reveals that there is no apparent trend or seasonality in the plot. ACF3 shows a barely significant correlation at lag=16, but all the other lag-values are within the blue lines and therefore not significant. Due to the lack of trend and/or seasonality in TS1 and the lack of significant correlation-lag-values, ACF3 is the correct autocorrelation function for TS1.

- Time series 2 : The ACF of the time series 2 (TS2)is ACF2. TS2 has a linear upward trend, but no apparent seasonality. This fits with ACF2 which has a linear downward trend. The lack of seasonality in TS2 is clearly apparent in the linear downward trend of ACF2.

- Time series 3 : The ACF of the Time Series 3 (TS3) is ACF 4. By visually inspecting TS3 we see a clear seasonality with some noise and a slight upward trend. We also see that ACF4 shows positive correlation for lag under 12, but negative correlation for higher values. If more lag-values were included in the graph we would eventually see positive correlation values again due to the cyclical behaviour of TS3. So, the shape of ACF4 fits TS3 due to the seasonality in TS3 which results in a cyclic pattern in its autocorrelation plot.

- Time series 4 : The ACF of the Time Series 4 (TS4) is ACF 1. TS4 seems to have cyclic behavior within each season, therefore it´s autocorrelation

function should also have a cyclic pattern within a larger cyclic pattern. We see a local maximum in the autocorrelation plot when lag = 12. This fits with the distance between the peaks in TS4.

d)

A visual inspection of the graphs suggest lag = 12. Meaning that Time Series 3 is a lagged version of Time Series 5 by 12 time steps.
The autocorrelation of the two graphs would similar to ACF4 due to the seasonality of the graphs. The cross correlation between the two graphs would be identical to the autocorrelation graph since none of the data is changed, only moved along the time axis.