ENV 790.30 - Time Series Analysis for Energy Data | Spring 2021

Assignment 3 - Due date 02/12/21

Traian Nirca

Directions

You should open the .rmd file corresponding to this assignment on RStudio. The file is available on our class repository on Github.

Once you have the project open the first thing you will do is change "Student Name" on line 3 with your name. Then you will start working through the assignment by **creating code and output** that answer each question. Be sure to use this assignment document. Your report should contain the answer to each question and any plots/tables you obtained (when applicable).

Please keep this R code chunk options for the report. It is easier for us to grade when we can see code and output together. And the tidy.opts will make sure that line breaks on your code chunks are automatically added for better visualization.

When you have completed the assignment, **Knit** the text and code into a single PDF file. Rename the pdf file such that it includes your first and last name (e.g., "LuanaLima_TSA_A01_Sp21.Rmd"). Submit this pdf using Sakai.

Questions

Consider the same data you used for A2 from the spreadsheet "Table_10.1_Renewable_Energy_Production_and_Consumption The data comes from the US Energy Information and Administration and corresponds to the January 2021 Monthly Energy Review. Once again you will work only with the following columns: Total Biomass Energy Production, Total Renewable Energy Production, Hydroelectric Power Consumption. Create a data frame structure with these three time series only.

R packages needed for this assignment:"forecast","tseries", and "Kendall". Install these packages, if you haven't done yet. Do not forget to load them before running your script, since they are NOT default packages.\

```
#Load/install required package here
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.0.3

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.0.3
```

```r
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.0.3

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
library(Kendall)
```

```
## Warning: package 'Kendall' was built under R version 4.0.3
```

```r
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 4.0.3
```

```r
library(xlsx)
```

```
## Warning: package 'xlsx' was built under R version 4.0.3
```

```r
#Importing data
energy_data <- read.xlsx(file="../Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source

read_col_names <- read.xlsx(file="../Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Sou

colnames(energy_data) <- read_col_names

work_data <- data.frame("Total Biomass Energy Production"=energy_data[,4], "Total Renewable Energy Prod

time_series <- ts(work_data, frequency = 12, start = c(1973,1))
#It is a monthly data so the frequency is 12, while the first point is January 1973
head(time_series)
```

```
##          Total.Biomass.Energy.Production Total.Renewable.Energy.Production
## Jan 1973                         129.787                           403.981
## Feb 1973                         117.338                           360.900
## Mar 1973                         129.938                           400.161
## Apr 1973                         125.636                           380.470
## May 1973                         129.834                           392.141
## Jun 1973                         125.611                           377.232
##          Hydroelectric.Power.Consumption
## Jan 1973                         272.703
## Feb 1973                         242.199
## Mar 1973                         268.810
## Apr 1973                         253.185
## May 1973                         260.770
## Jun 1973                         249.859
```

```r
the_date <- energy_data[,1]
my_date <- ymd(the_date)
head(my_date)
```

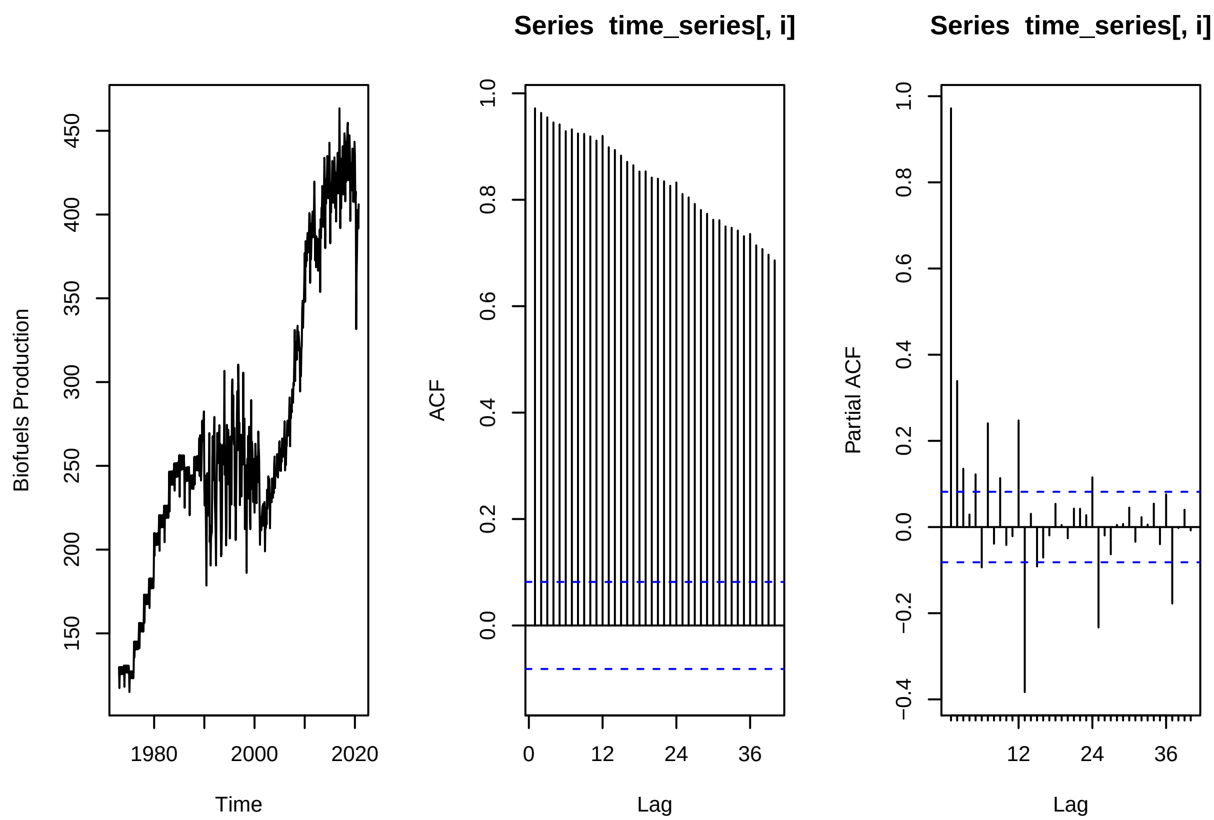
```
## [1] "1973-01-01" "1973-02-01" "1973-03-01" "1973-04-01" "1973-05-01"
## [6] "1973-06-01"
```

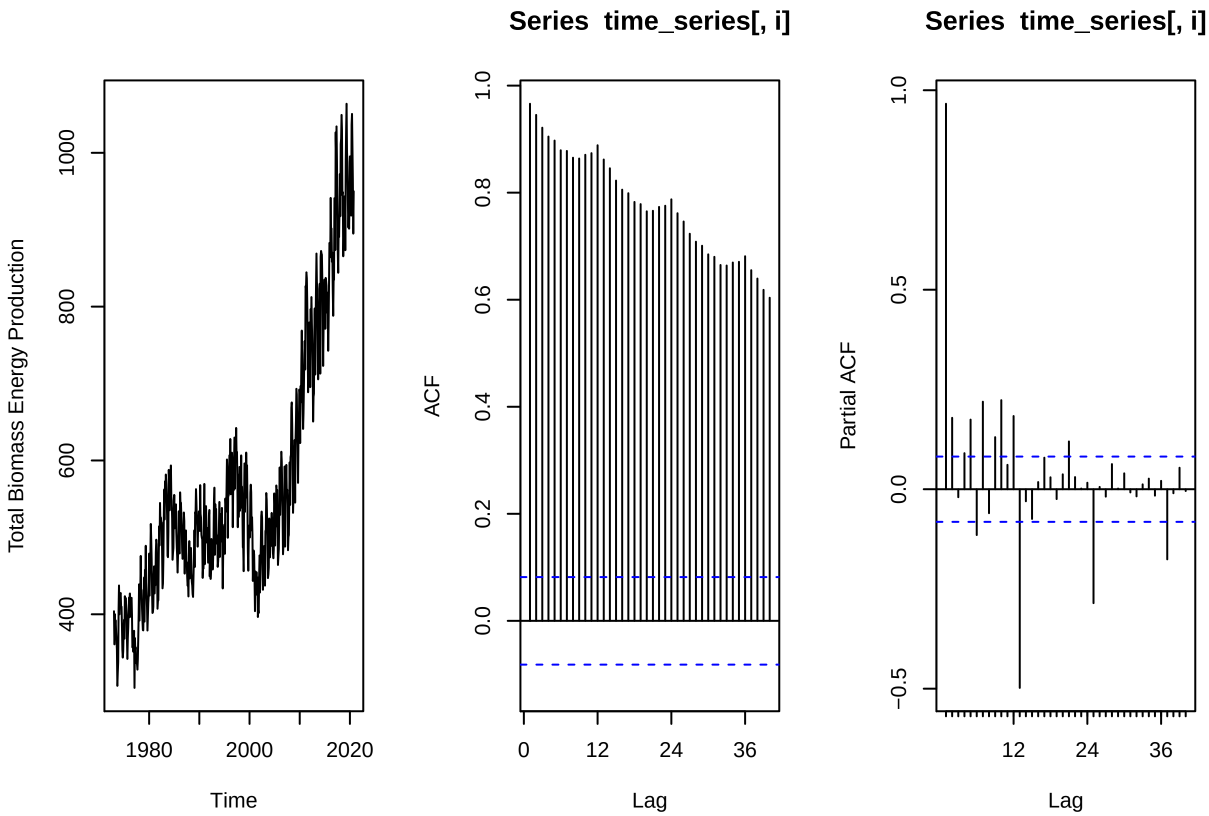
##Trend Component

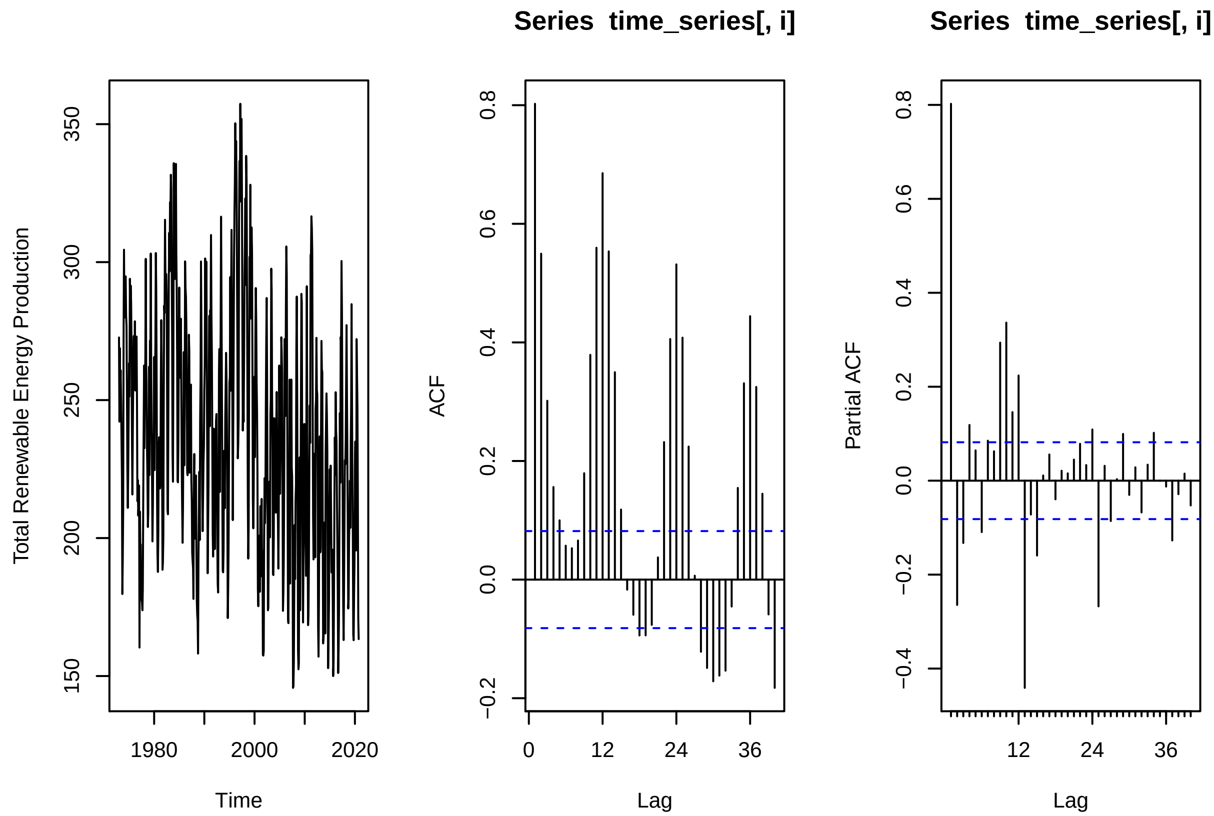
Q1

Create a plot window that has one row and three columns. And then for each object on your data frame, fill the plot window with time series plot, ACF and PACF. You may use the some code form A2, but I want all three plots on the same window this time. (Hint: watch videos for M4)

```
#Plot the three plots in the same window
for(i in 1:3){
  par(mfrow=c(1,3))
  plot(time_series[,i], ylab=colnames(energy_data)[i+2])
  Acf(time_series[,i], lag.max = 40)
  Pacf(time_series[,i], lag.max = 40)

}
```




Series time_series[, i]
Series time_series[, i]
Total Biomass Energy Production
400
600
800
1000
1980
2000
2020
Time
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
12
24
36
Lag
Partial ACF
−0.5
0.0
0.5
1.0
12
24
36
Lag



Q2

From the plot in Q1, do the series Total Biomass Energy Production, Total Renewable Energy Production, Hydroelectric Power Consumption appear to have a trend? If yes, what kind of trend?

Based on the plots in Q1, Total Biomass Energy Production and Total Renewable Energy Production seem to have a linear increasing trend, while Hydroelectric Power Consumption seems to have a seasonal trend. Based on its ACF, Total Renewable Energy Production also seems to have a small seasonal approach.

Q3

Use the *lm()* function to fit a linear trend to the three time series. Ask R to print the summary of the regression. Interpret the regression output, i.e., slope and intercept. Save the regression coefficients for further analysis.

We start by fitting a linear model to

```
nobs <- nrow(work_data)
t <- c(1:nobs)

#For Total Biomass Energy Production

linear_trend_model=lm(time_series[,1]~t)
summary(linear_trend_model)
```

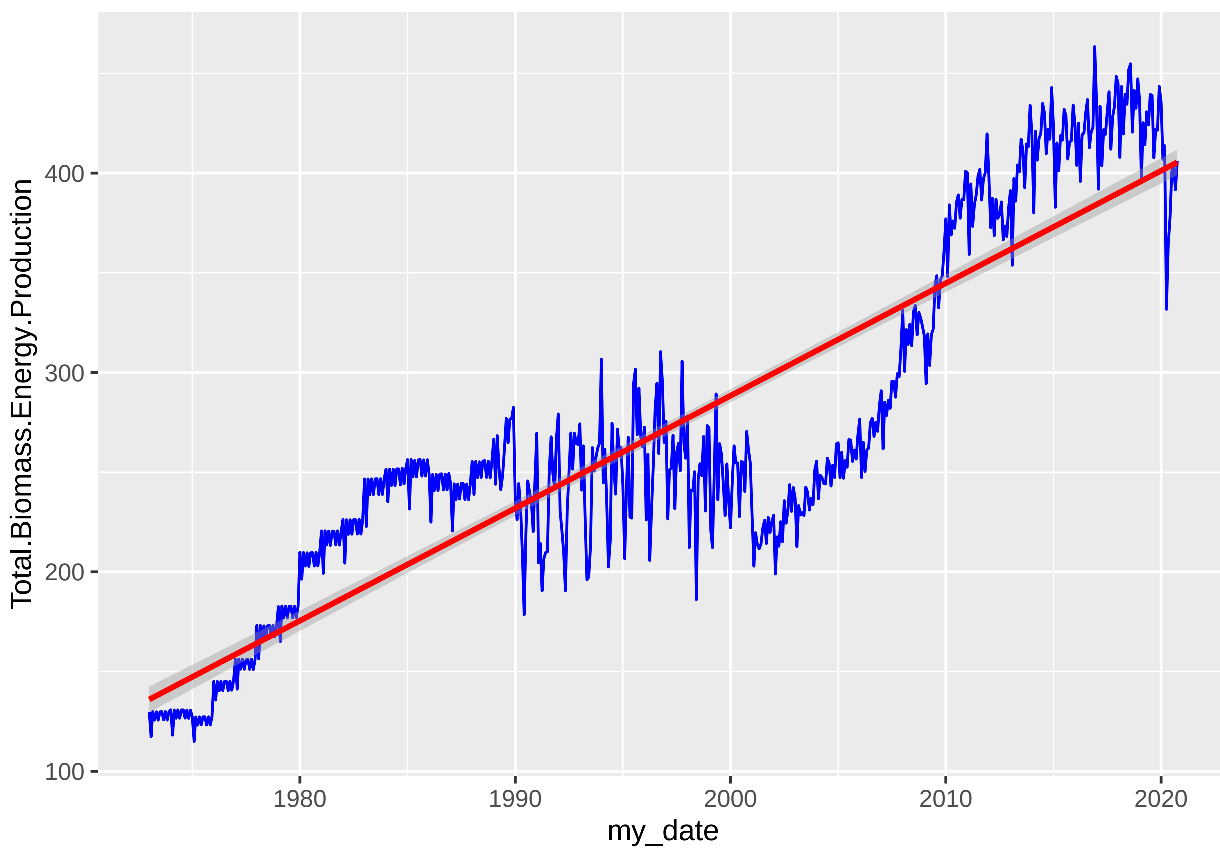
```
##
## Call:
## lm(formula = time_series[, 1] ~ t)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -101.149  -25.456    4.985   33.353   79.634
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.355e+02  3.296e+00   41.11   <2e-16 ***
## t           4.702e-01  9.934e-03   47.33   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.44 on 572 degrees of freedom
## Multiple R-squared:  0.7966, Adjusted R-squared:  0.7962
## F-statistic:  2240 on 1 and 572 DF,  p-value: < 2.2e-16
```

```
beta10 = as.numeric(linear_trend_model$coefficients[1]) #the first coefficient is the intercept term or
beta11 = as.numeric(linear_trend_model$coefficients[2]) #the second coefficient is the slope or beta1

ggplot(time_series[,1], aes(x=my_date, y=time_series[,1])) +
  geom_line(color="blue") +
  ylab(colnames(work_data[1])) +
  geom_smooth(color="red",method="lm")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```

```
## `geom_smooth()` using formula 'y ~ x'
```



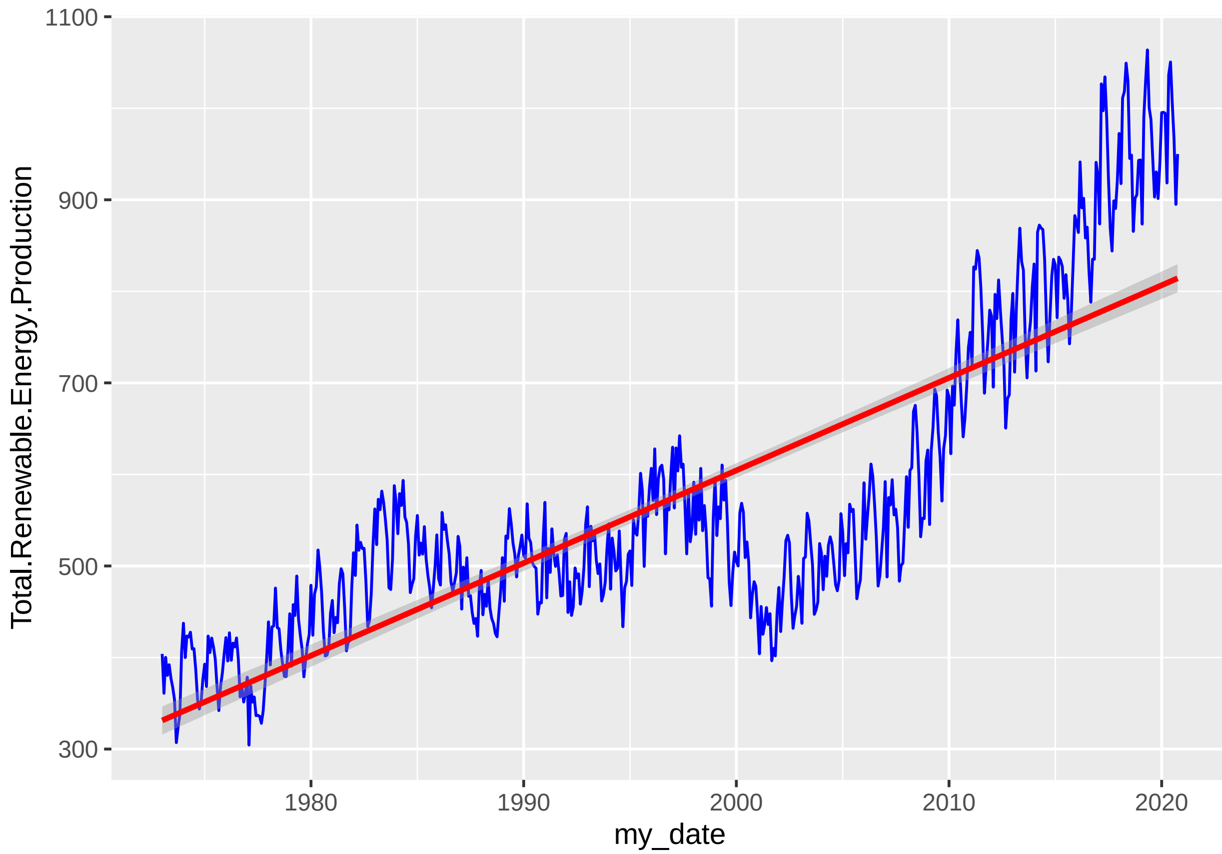
```r
#For Total Renewable Energy Production

linear_trend_model=lm(time_series[,2]~t)
summary(linear_trend_model)
```

```
##
## Call:
## lm(formula = time_series[, 2] ~ t)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -224.735  -55.673    5.418   60.453  263.849
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 330.37156    7.86270   42.02   <2e-16 ***
## t             0.84299    0.02369   35.58   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 94.07 on 572 degrees of freedom
## Multiple R-squared:  0.6887, Adjusted R-squared:  0.6882
## F-statistic:  1266 on 1 and 572 DF,  p-value: < 2.2e-16
```

```r
beta20 = as.numeric(linear_trend_model$coefficients[1]) #the first coefficient is the intercept term or
beta21 = as.numeric(linear_trend_model$coefficients[2]) #the second coefficient is the slope or beta1

ggplot(time_series[,2], aes(x=my_date, y=time_series[,2])) +
  geom_line(color="blue") +
  ylab(colnames(work_data[2])) +
  geom_smooth(color="red",method="lm")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```

```
## `geom_smooth()` using formula 'y ~ x'
```



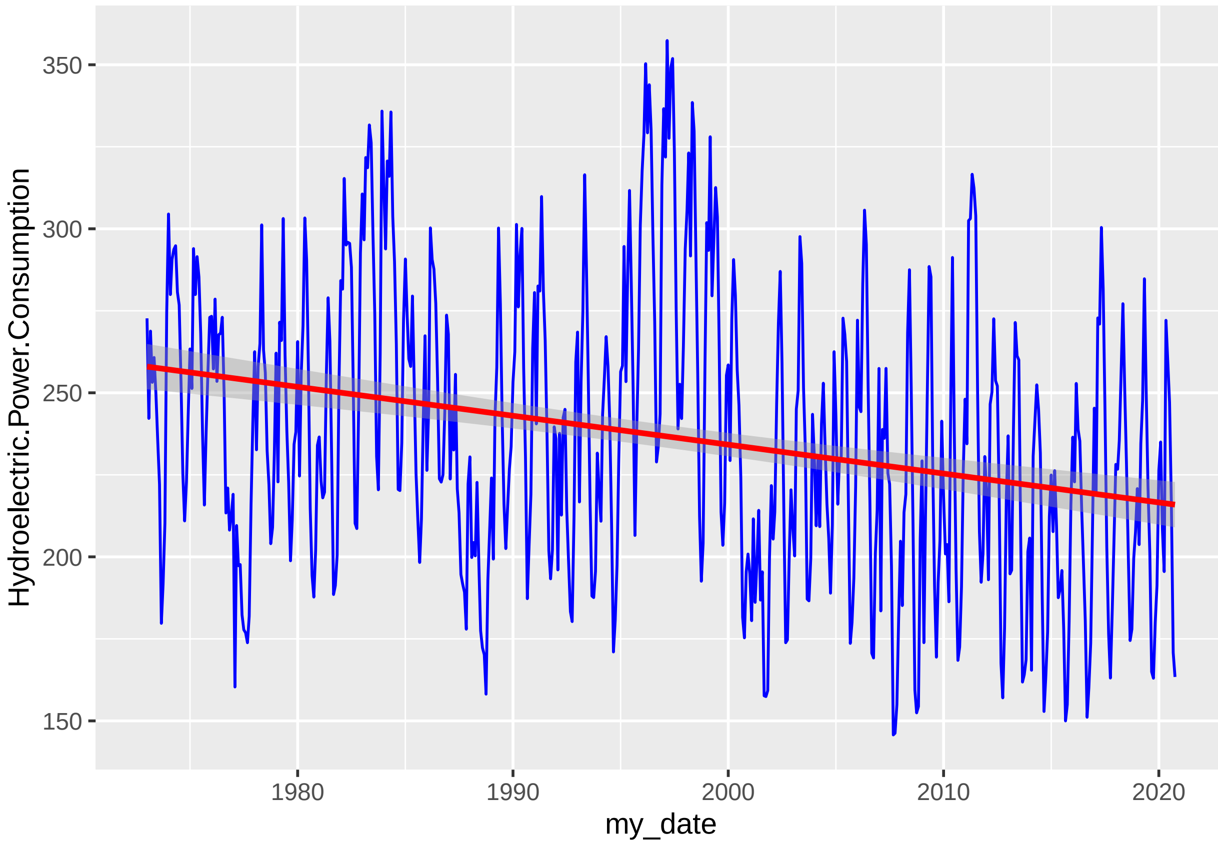

```r
#For Hydroelectric Power Consumption

linear_trend_model=lm(time_series[,3]~t)
summary(linear_trend_model)
```

```
##
## Call:
## lm(formula = time_series[, 3] ~ t)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -94.06 -31.57  -1.63  27.73 120.69
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 258.05622    3.52899  73.125  < 2e-16 ***
## t            -0.07341    0.01063  -6.903 1.36e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 42.22 on 572 degrees of freedom
## Multiple R-squared:  0.07689,    Adjusted R-squared:  0.07528
## F-statistic: 47.64 on 1 and 572 DF,  p-value: 1.361e-11
```

```r
beta30 = as.numeric(linear_trend_model$coefficients[1]) #the first coefficient is the intercept term or
beta31 = as.numeric(linear_trend_model$coefficients[2]) #the second coefficient is the slope or beta1
```

```r
ggplot(time_series[,3], aes(x=my_date, y=time_series[,3])) +
  geom_line(color="blue") +
  ylab(colnames(work_data[3])) +
  geom_smooth(color="red",method="lm")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.

## `geom_smooth()` using formula 'y ~ x'
```



The first two plots have positive slopes, while the third one has a negative slope. This is consistent with the trend that we observe visually: first two increasing and teh alst decreasing. The intercepts seem to be close to the initial values of the curves.

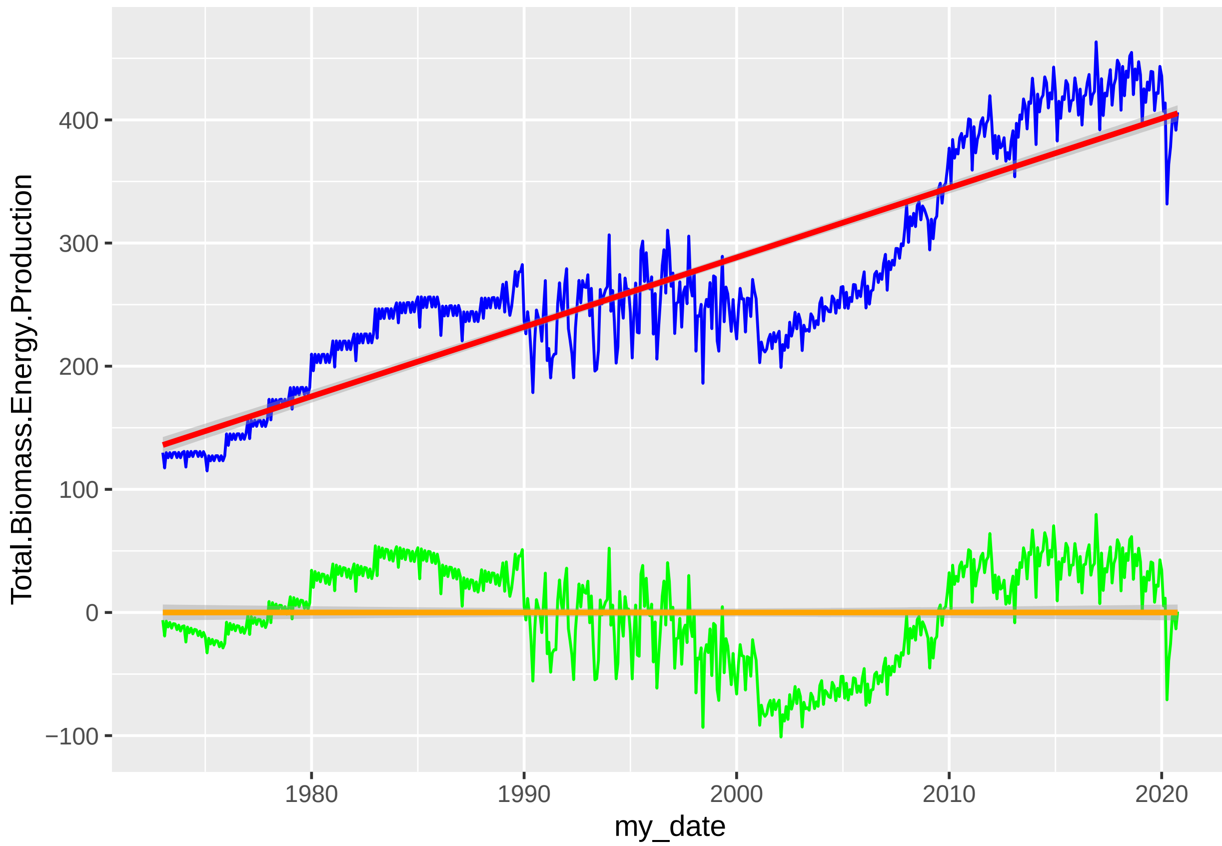
Q4

Use the regression coefficients from Q3 to detrend the series. Plot the detrended series and compare with the plots from Q1. What happened? Did anything change?

```r
detrend_data1 <- time_series[,1]-(beta10+beta11*t)

ggplot(time_series[,1], aes(x=my_date, y=time_series[,1])) +
  geom_line(color="blue") +
  ylab(colnames(work_data[1])) +
  geom_smooth(color="red",method="lm") +
  geom_line(aes(y=detrend_data1), col="green")+
  geom_smooth(aes(y=detrend_data1),color="orange",method="lm")
```

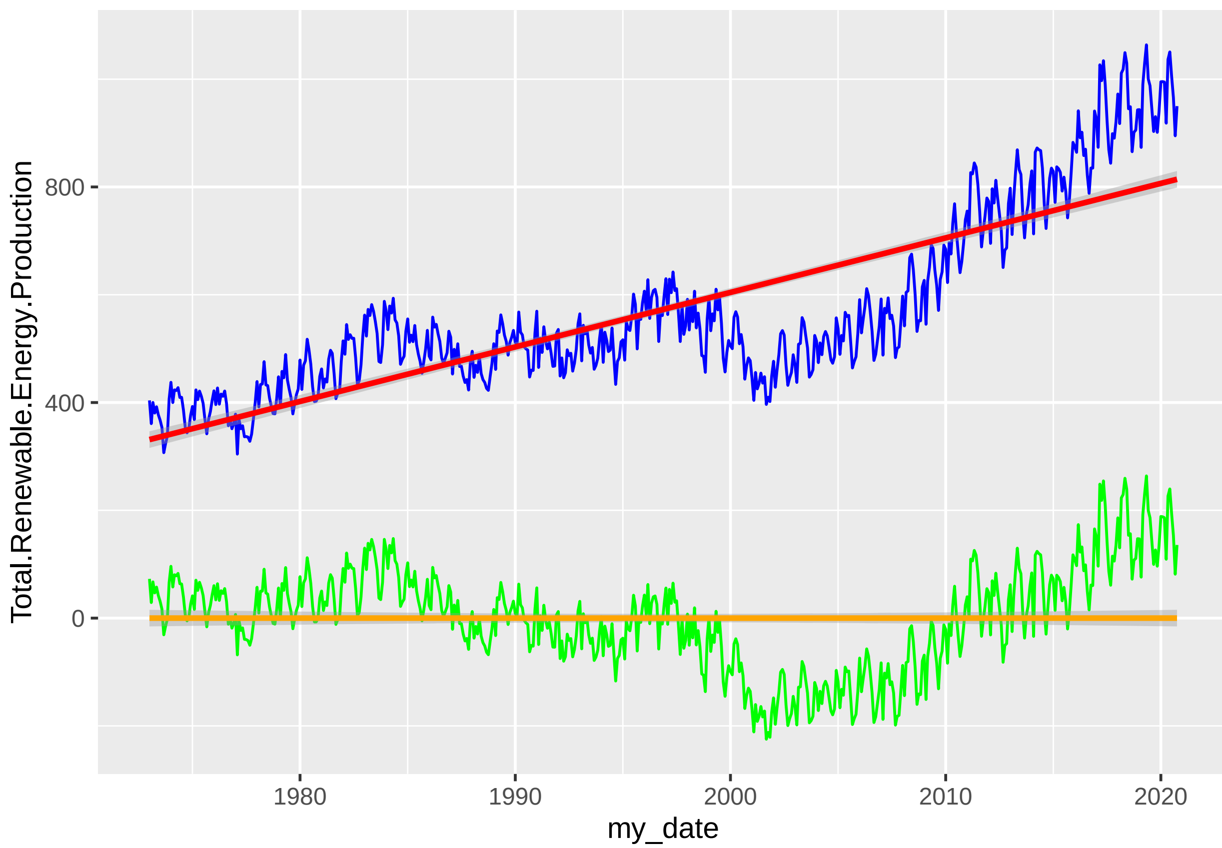
```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



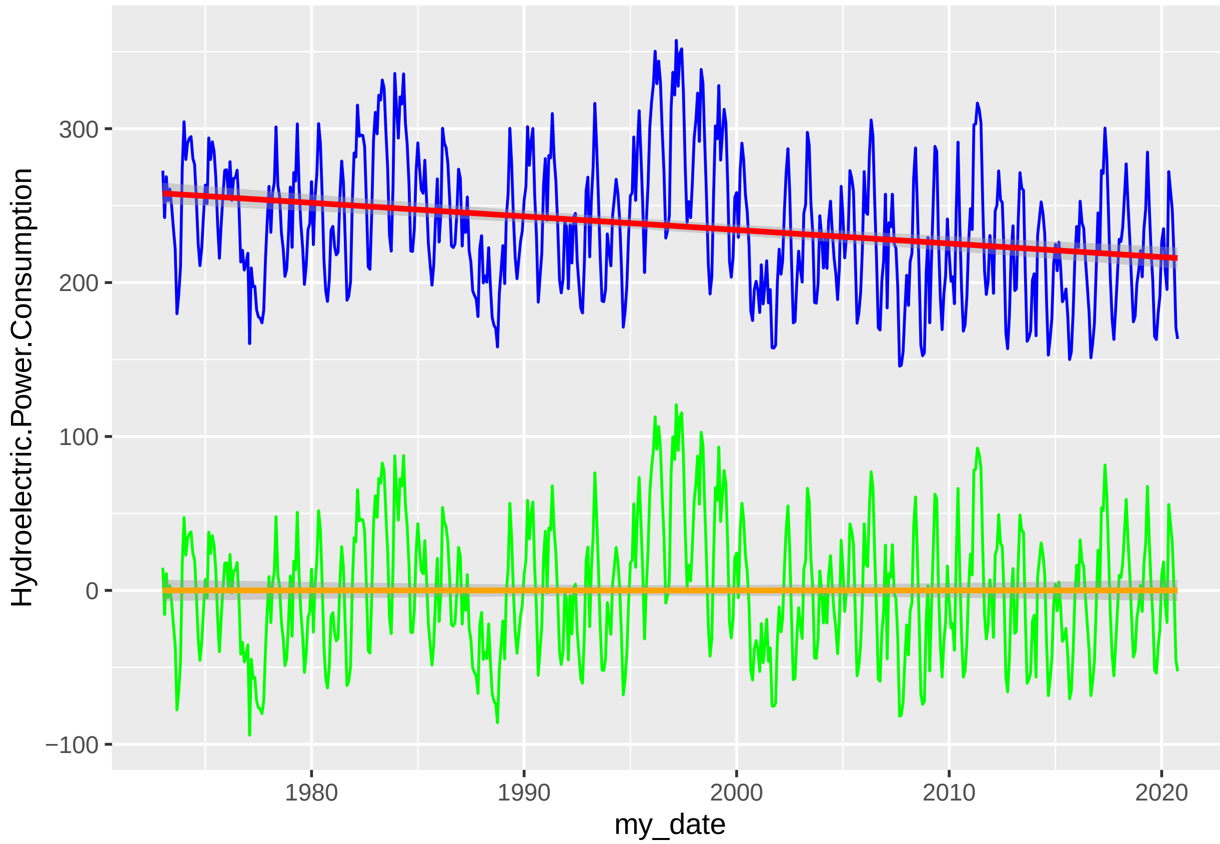

```r
detrend_data2 <- time_series[,2]-(beta20+beta21*t)

ggplot(time_series[,2], aes(x=my_date, y=time_series[,2])) +
  geom_line(color="blue") +
  ylab(colnames(work_data[2])) +
  geom_smooth(color="red",method="lm") +
  geom_line(aes(y=detrend_data2), col="green")+
  geom_smooth(aes(y=detrend_data2),color="orange",method="lm")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.

## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```




```
detrend_data3 <- time_series[,3]-(beta30+beta31*t)

ggplot(time_series[,3], aes(x=my_date, y=time_series[,3])) +
  geom_line(color="blue") +
  ylab(colnames(work_data[3])) +
  geom_smooth(color="red",method="lm") +
  geom_line(aes(y=detrend_data3), col="green")+
  geom_smooth(aes(y=detrend_data3),color="orange",method="lm")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.

## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```

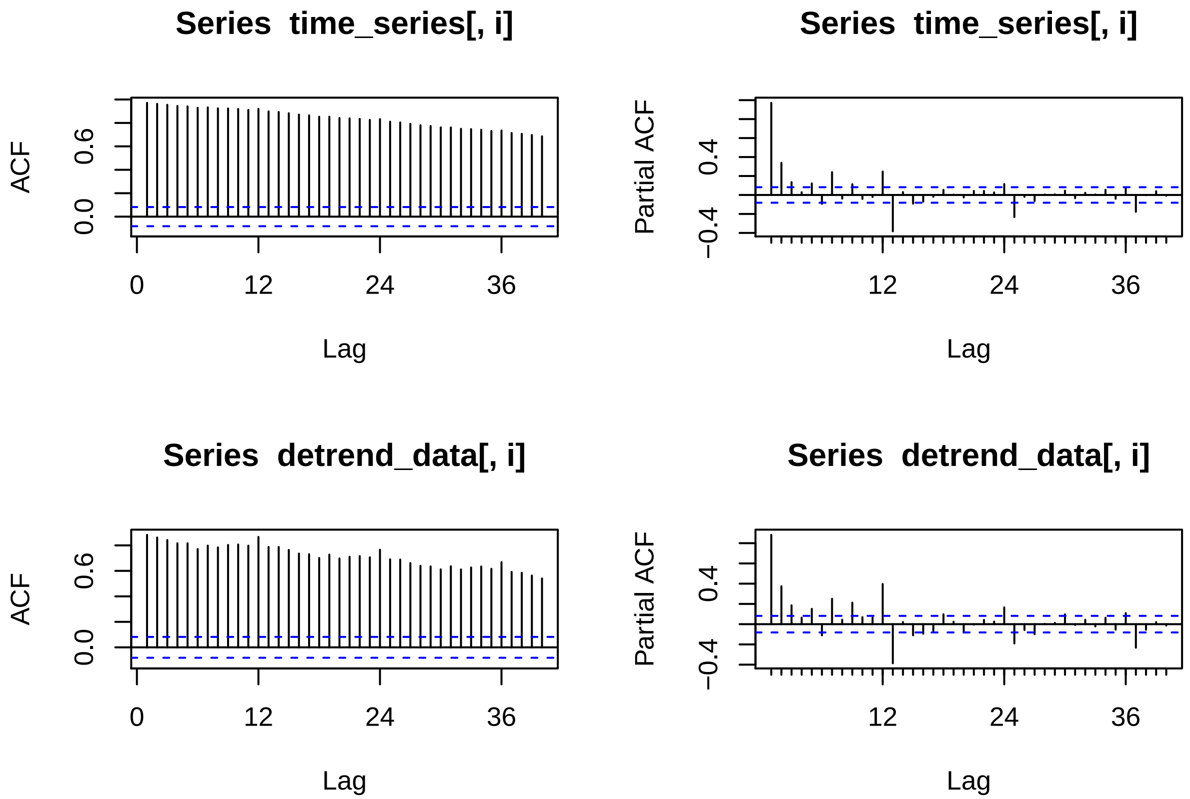


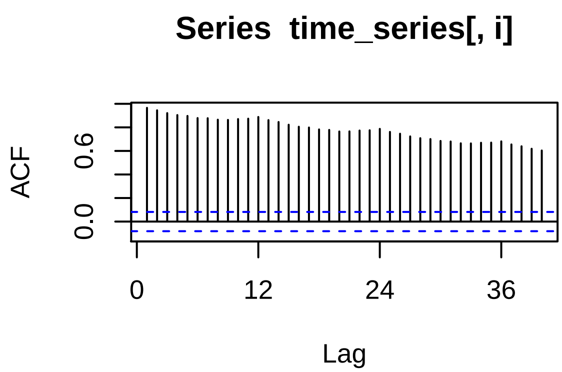
In each plot the linear trends are suppressed. We now have 3 detrended curves that we can study without worrying about the linear trends.

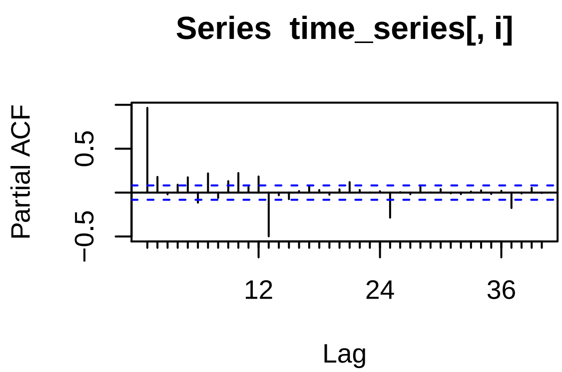
Q5

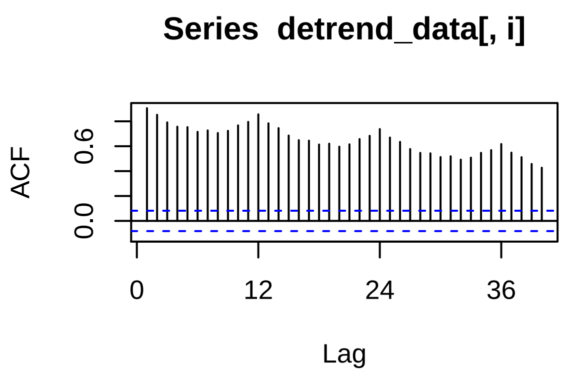
Plot ACF and PACF for the detrended series and compare with the plots from Q1. Did the plots change? How?

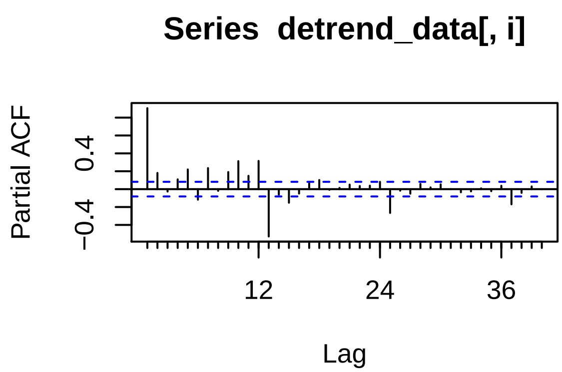
```
detrend_data <- data.frame(detrend_data1, detrend_data2, detrend_data3)

for(i in 1:3){
  par(mfrow=c(2,2))
  Acf(time_series[,i], lag.max = 40)
  Pacf(time_series[,i], lag.max = 40)
  Acf(detrend_data[,i], lag.max = 40)
  Pacf(detrend_data[,i], lag.max = 40)
}
```

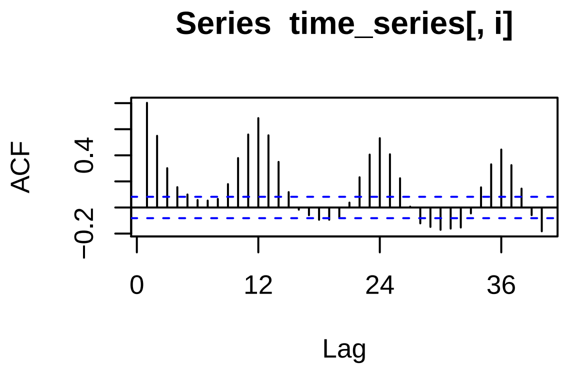

Series time_series[, i]
ACF
0.0
0.6
0
12
24
36
Lag
Series time_series[, i]
Partial ACF
−0.4
0.4
12
24
36
Lag
Series detrend_data[, i]
ACF
0.0
0.6
0
12
24
36
Lag
Series detrend_data[, i]
Partial ACF
−0.4
0.4
12
24
36
Lag

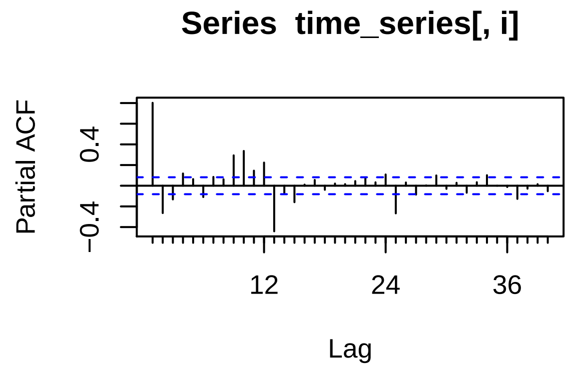

Series time_series[, i]
ACF
0.0
0.6
0
12
24
36
Lag

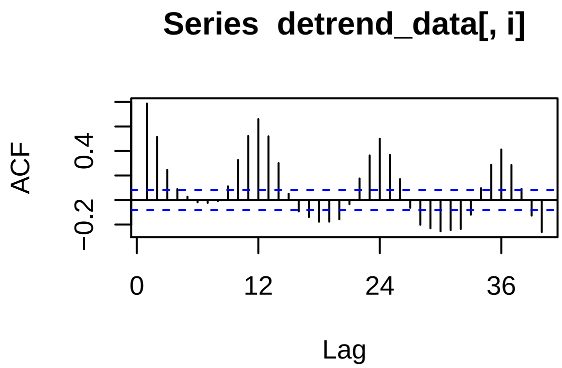

Series time_series[, i]
Partial ACF
−0.5
0.5
12
24
36
Lag

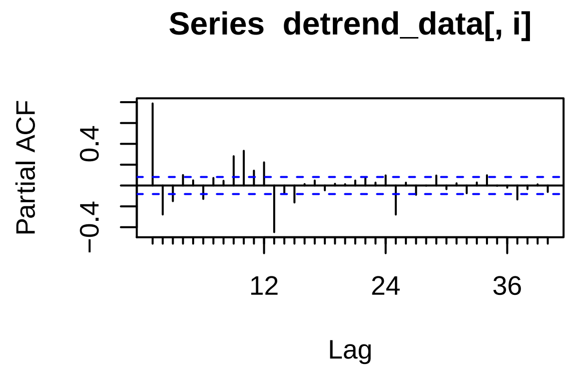

Series detrend_data[, i]
ACF
0.0
0.6
0
12
24
36
Lag


Series detrend_data[, i]
Partial ACF
−0.4
0.4
12
24
36
Lag









The PACF plots have changed very little, while the ACF plots have changed considerably to let us see very distinctive repeating patterns, which are indicating seasonality.

Seasonal Component

Set aside the detrended series and consider the original series again from Q1 to answer Q6 to Q8.

Q6

Do the series seem to have a seasonal trend? Which serie/series? Use function *lm()* to fit a seasonal means model to this/these time series. Ask R to print the summary of the regression. Interpret the regression output. Save the regression coefficients for further analysis.

Series 3 has the most pronounced seasonal trend and series 2 also has one, although less pronounced.

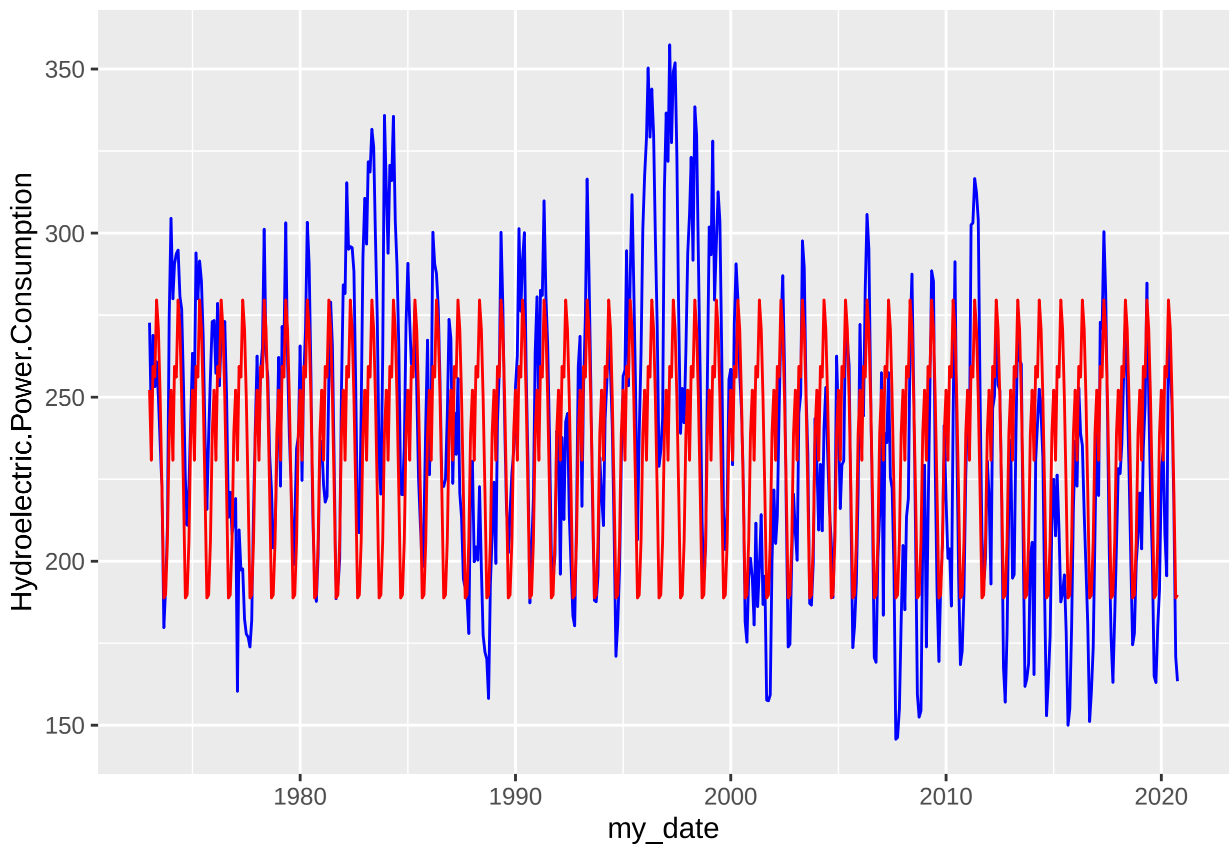
```
#For Hydroelectric power Consumption (Series 3)

#First create the seasonal dummies
dummies <- seasonaldummy(time_series[,3])

seas_means_model=lm(time_series[,3]~dummies)
summary(seas_means_model)
```

```
##
## Call:
## lm(formula = time_series[, 3] ~ dummies)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -92.064 -22.897  -2.654  20.642  98.058
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  238.887      4.863  49.125  < 2e-16 ***
## dummiesJan    13.270      6.841   1.940  0.05291 .
## dummiesFeb    -8.133      6.841  -1.189  0.23499
## dummiesMar    20.442      6.841   2.988  0.00293 **
## dummiesApr    17.199      6.841   2.514  0.01221 *
## dummiesMay    40.726      6.841   5.953 4.64e-09 ***
## dummiesJun    31.764      6.841   4.643 4.28e-06 ***
## dummiesJul    10.858      6.841   1.587  0.11306
## dummiesAug   -17.907      6.841  -2.618  0.00909 **
## dummiesSep   -50.121      6.841  -7.326 8.26e-13 ***
## dummiesOct   -49.165      6.841  -7.187 2.12e-12 ***
## dummiesNov   -32.757      6.877  -4.763 2.43e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 33.34 on 562 degrees of freedom
## Multiple R-squared:  0.4345, Adjusted R-squared:  0.4234
## F-statistic: 39.25 on 11 and 562 DF,  p-value: < 2.2e-16
```

```r
#We store regression coefficients
beta_int3=seas_means_model$coefficients[1]
beta_coeff3=seas_means_model$coefficients[2:12]


#compute seasonal component
seas_comp3=array(0,nobs)
for(i in 1:nobs){
seas_comp3[i]=(beta_int3+beta_coeff3%*%dummies[i,])
}

#Understanding what we did
ggplot(time_series[,3], aes(x=my_date, y=time_series[,3])) +
geom_line(color="blue") +
ylab(colnames(work_data[3])) +
geom_line(aes(y=seas_comp3), col="red")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



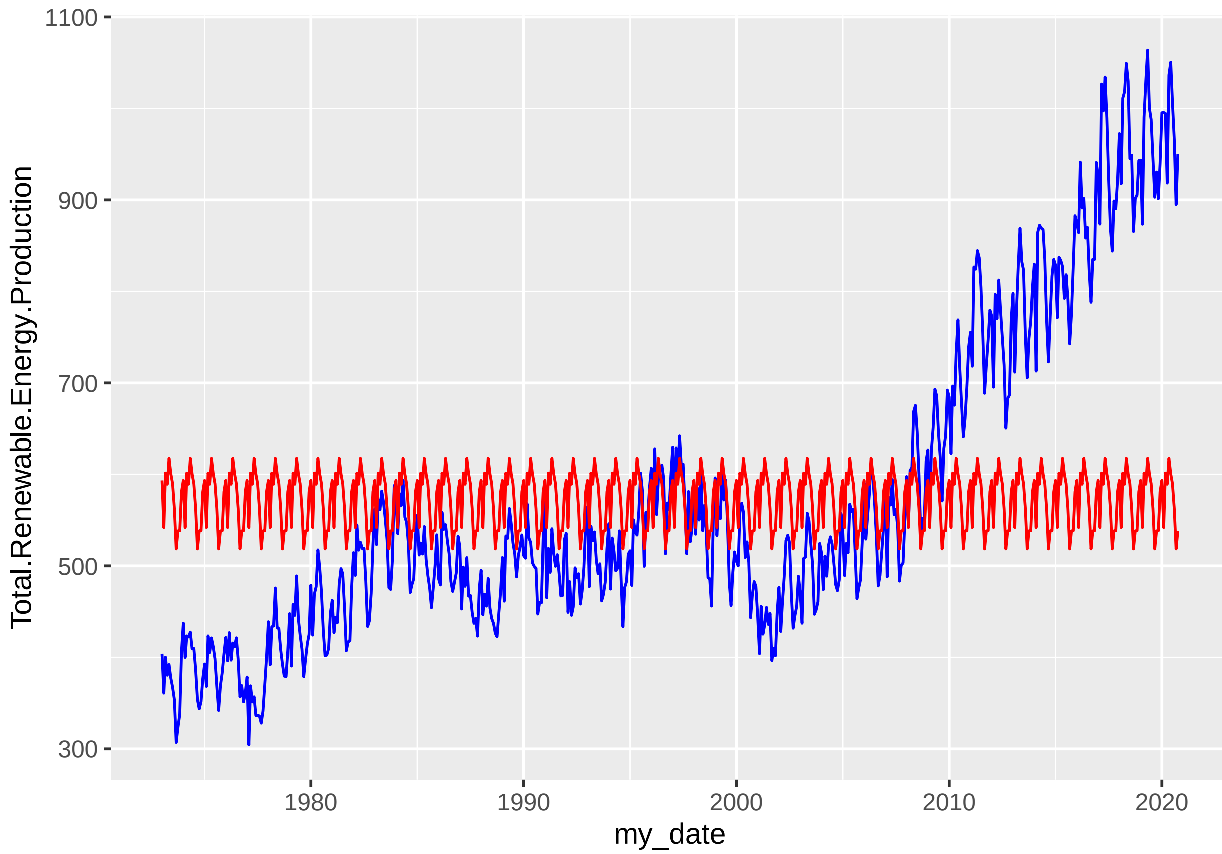

```r
#For Total Renewable Energy Production (Series 2)

#First create the seasonal dummies
dummies <- seasonaldummy(time_series[,2])

seas_means_model=lm(time_series[,2]~dummies)
summary(seas_means_model)
```

```
##
## Call:
## lm(formula = time_series[, 2] ~ dummies)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -263.99 -102.98  -52.33   36.68  453.58
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  580.912     24.406  23.802   <2e-16 ***
## dummiesJan    12.451     34.335   0.363   0.7170
## dummiesFeb   -38.964     34.335  -1.135   0.2569
## dummiesMar    20.515     34.335   0.597   0.5504
## dummiesApr     8.294     34.335   0.242   0.8092
## dummiesMay    36.628     34.335   1.067   0.2865
## dummiesJun    19.560     34.335   0.570   0.5691
## dummiesJul     8.863     34.335   0.258   0.7964
```

```
## dummiesAug   -18.480     34.335  -0.538   0.5906
## dummiesSep   -62.410     34.335  -1.818   0.0696 .
## dummiesOct   -42.649     34.335  -1.242   0.2147
## dummiesNov   -42.516     34.515  -1.232   0.2185
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 167.3 on 562 degrees of freedom
## Multiple R-squared:  0.03244,    Adjusted R-squared:  0.01351
## F-statistic: 1.713 on 11 and 562 DF,  p-value: 0.06702
```

```r
#We store regression coefficients
beta_int2=seas_means_model$coefficients[1]
beta_coeff2=seas_means_model$coefficients[2:12]


#compute seasonal component
seas_comp2=array(0,nobs)
for(i in 1:nobs){
seas_comp2[i]=(beta_int2+beta_coeff2%*%dummies[i,])
}

#Understanding what we did
ggplot(time_series[,2], aes(x=my_date, y=time_series[,2])) +
geom_line(color="blue") +
ylab(colnames(work_data[2])) +
geom_line(aes(y=seas_comp2), col="red")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



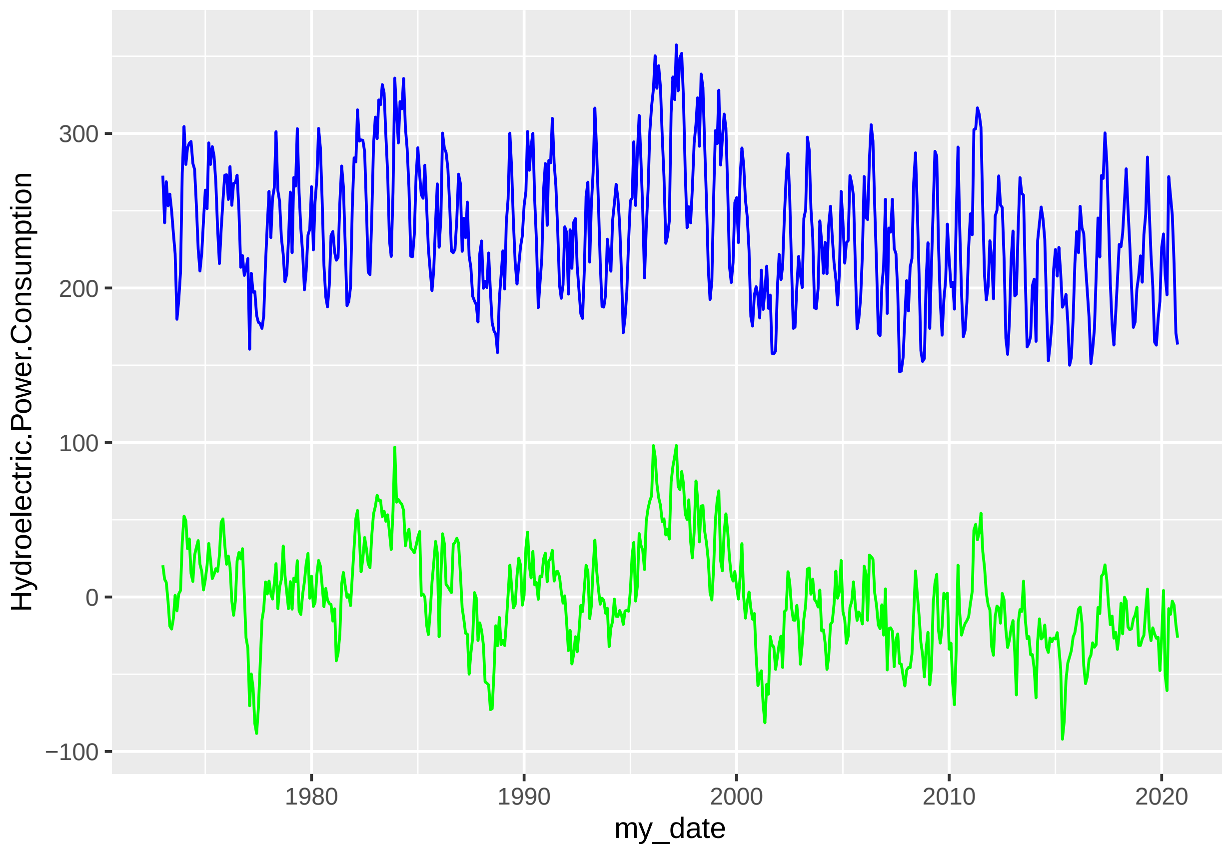
We have the values of the dummies for each month, as well as the intercept. We can now de-season the series.

Q7

Use the regression coefficients from Q6 to deseason the series. Plot the deseason series and compare with the plots from part Q1. Did anything change?

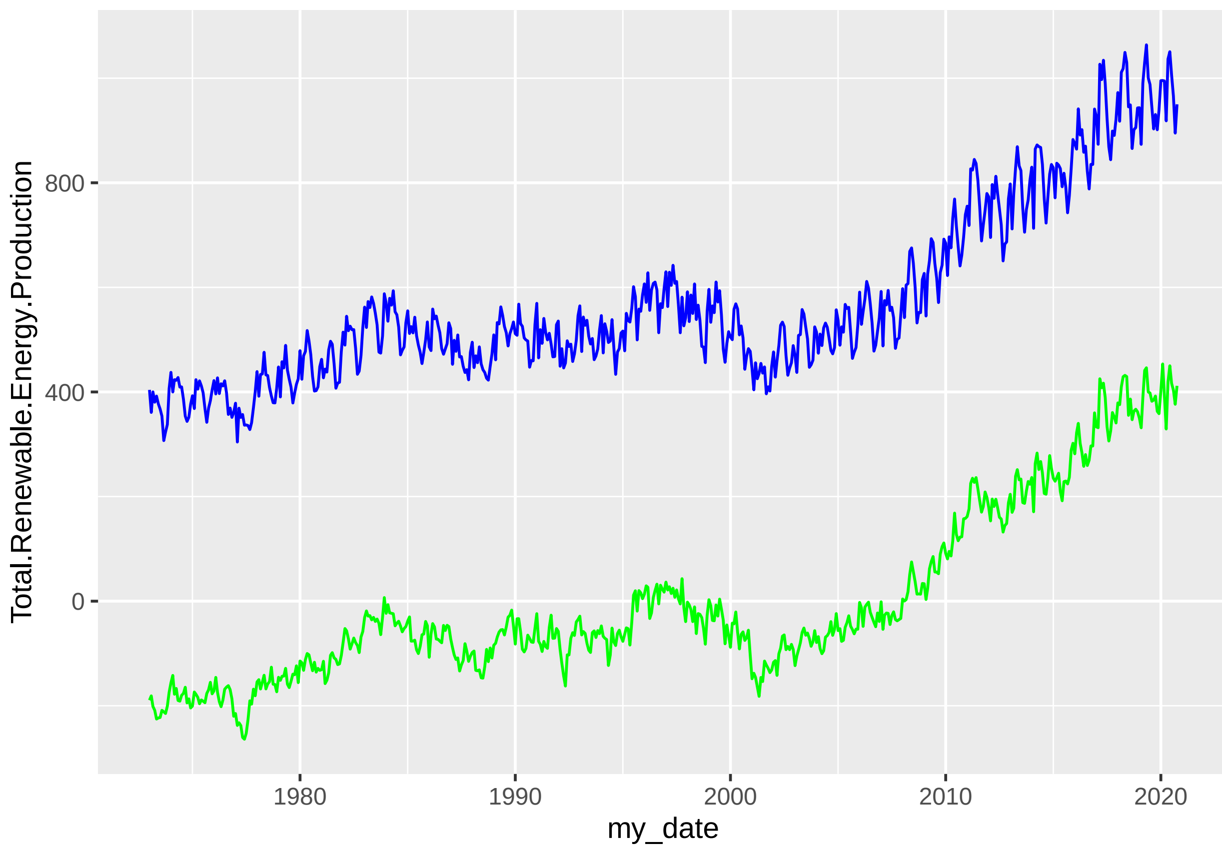
```
#For Hydroelectric Power Consumption

deseason_data3 <- time_series[,3]-seas_comp3

ggplot(time_series[,3], aes(x=my_date, y=time_series[,3])) +
geom_line(color="blue") +
ylab(colnames(work_data[3])) +
geom_line(aes(y=deseason_data3), col="green")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```




```
#For Total Renewable Energy Production

deseason_data2 <- time_series[,2]-seas_comp2

ggplot(time_series[,2], aes(x=my_date, y=time_series[,2])) +
geom_line(color="blue") +
ylab(colnames(work_data[2])) +
geom_line(aes(y=deseason_data2), col="green")
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```

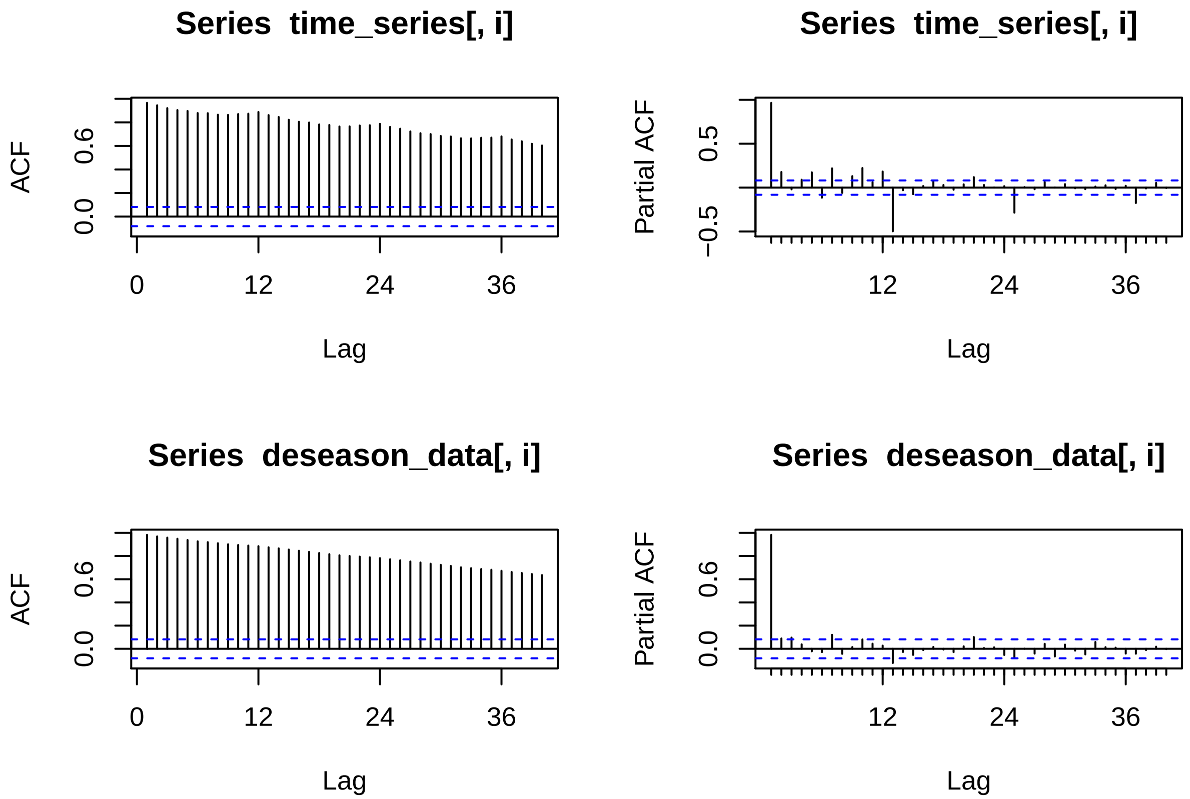


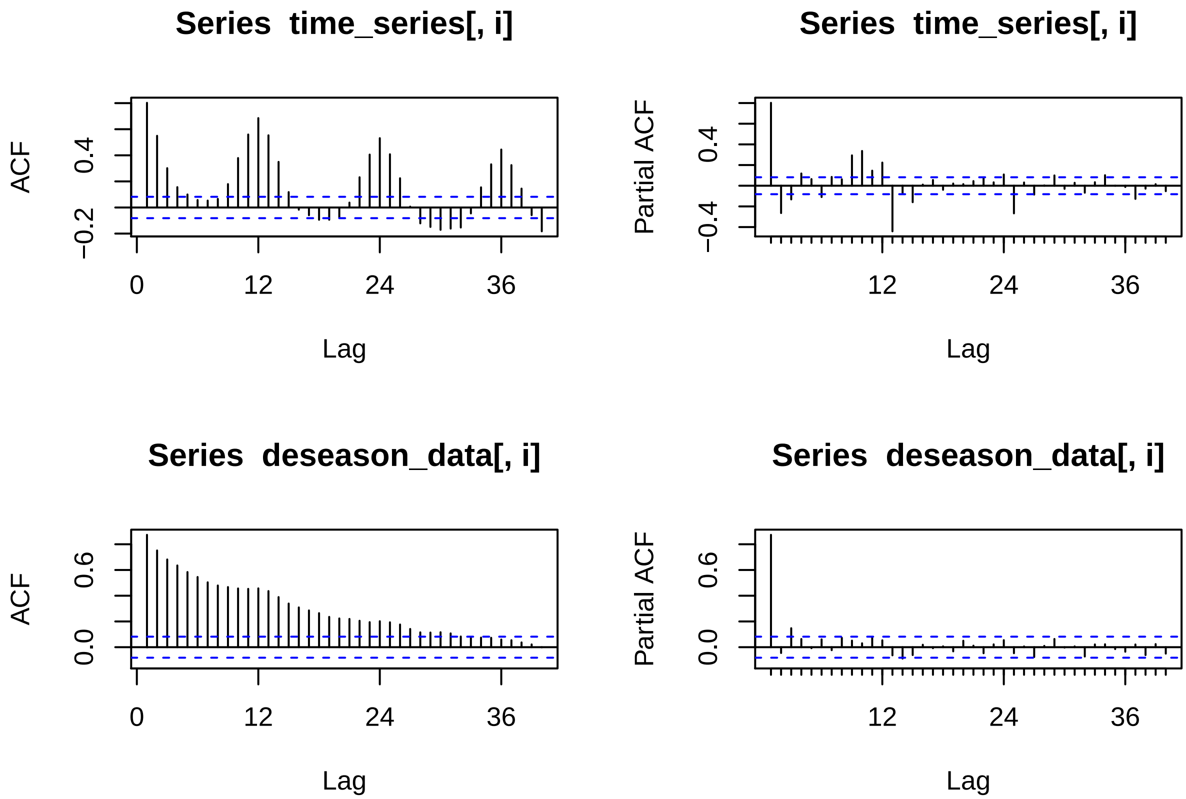
The seasonal component is gone from both plots.

Q8

Plot ACF and PACF for the deseason series and compare with the plots from Q1. Did the plots change? How?

```
deseason_data <- data.frame(detrend_data1, deseason_data2, deseason_data3)

for(i in 2:3){
  par(mfrow=c(2,2))
  Acf(time_series[,i], lag.max = 40)
  Pacf(time_series[,i], lag.max = 40)
  Acf(deseason_data[,i], lag.max = 40)
  Pacf(deseason_data[,i], lag.max = 40)
}
```


Series time_series[, i]
ACF
0.0
0.6
0
12
24
36
Lag
Series time_series[, i]
Partial ACF
−0.5
0.5
12
24
36
Lag
Series deseason_data[, i]
ACF
0.0
0.6
0
12
24
36
Lag
Series deseason_data[, i]
Partial ACF
0.0
0.6
12
24
36
Lag



The plots changed: we no longer see a seasonal component in the ACF, it is not oscillating anymore, instead it is decreasing.