ENV 790.30 - Time Series Analysis for Energy Data | Spring 2024

Assignment 3 - Due date 02/01/24

Emma Kaufman

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 file open on your local machine the first thing you will do is rename the file such that it includes your first and last name (e.g., "LuanaLima_TSA_A02_Sp24.Rmd"). Then change "Student Name" on line 4 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. 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_by_Source.xlsx".

The data comes from the US Energy Information and Administration and corresponds to the December 2022 **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(forecast)
```

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

```
library(tseries)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

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

```r
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

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

```r
library(readxl)
library(ggplot2)
library(cowplot)
```

```
##
## Attaching package: 'cowplot'

## The following object is masked from 'package:lubridate':
##
##     stamp
```

```r
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
#Importing data set without changing the original file using read.xlsx
energy_data1 <- read_excel(path=
                        "./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xlsx",
                          skip = 12,
                          sheet="Monthly Data",
                          col_names=FALSE)
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
## * `` -> `...12`
## * `` -> `...13`
## * `` -> `...14`
```

```
#Now let's extract the column names from row 11
read_col_names <- read_excel(path=
                          "./Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source.xlsx
                             skip = 10,
                             n_max = 1,
                             sheet="Monthly Data",
                             col_names=FALSE)
```

```
## New names:
## * `` -> `...1`
## * `` -> `...2`
## * `` -> `...3`
## * `` -> `...4`
## * `` -> `...5`
## * `` -> `...6`
## * `` -> `...7`
## * `` -> `...8`
## * `` -> `...9`
## * `` -> `...10`
## * `` -> `...11`
## * `` -> `...12`
## * `` -> `...13`
## * `` -> `...14`
```

```
colnames(energy_data1) <- read_col_names
```

```
#creating a dataframe of the columns we are interested in
energy_interest <- select(energy_data1,
                          'Total Biomass Energy Production',
                          'Total Renewable Energy Production',
                          'Hydroelectric Power Consumption')
#creating a timeseries object
energy_ts <- ts(energy_interest, start = c(1973,1), frequency = 12)
biomass_ts <- energy_ts[,1]
renewable_ts <- energy_ts[,2]
hydro_ts <- energy_ts[,3]
```

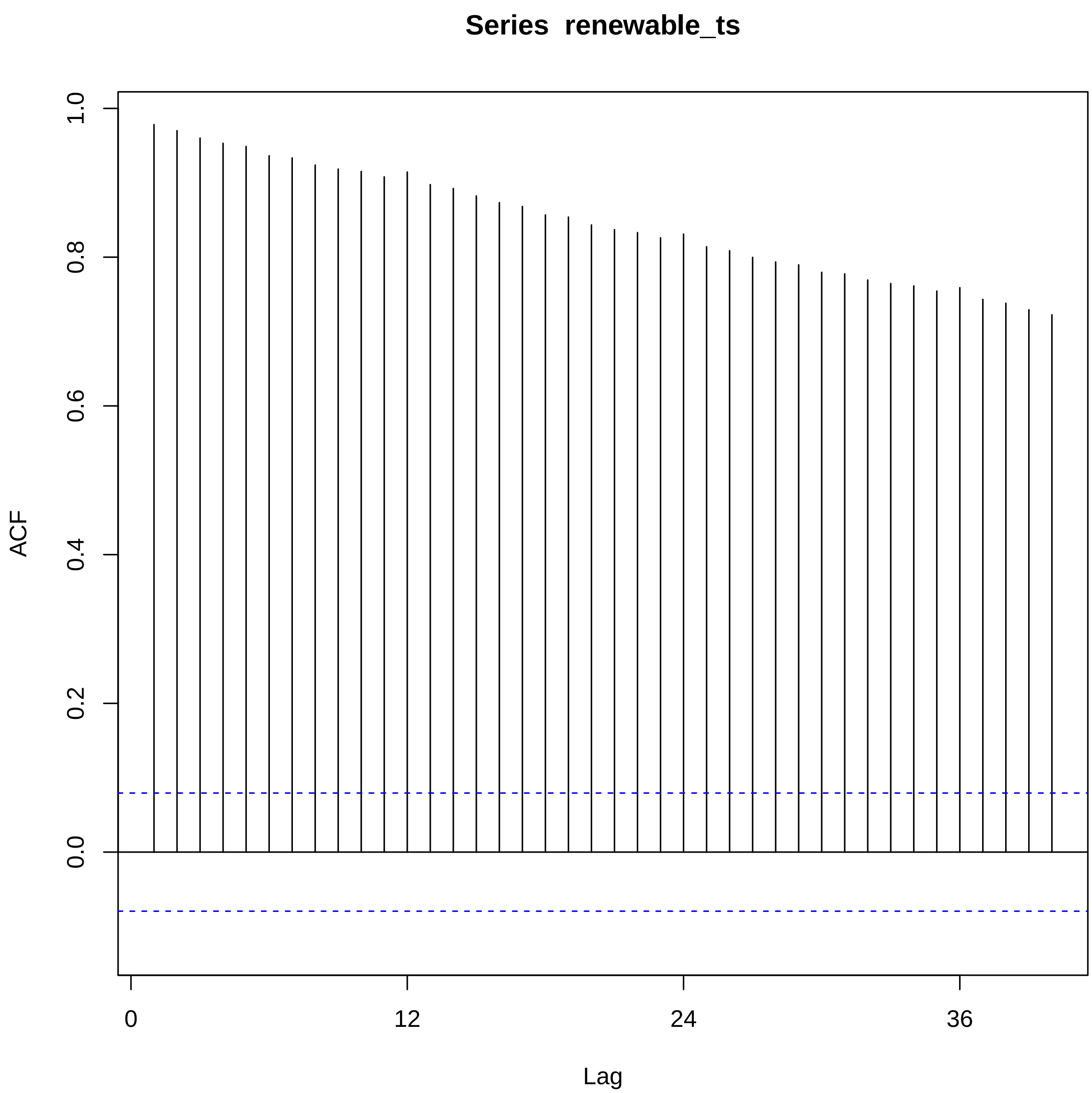
##Trend Component

Q1

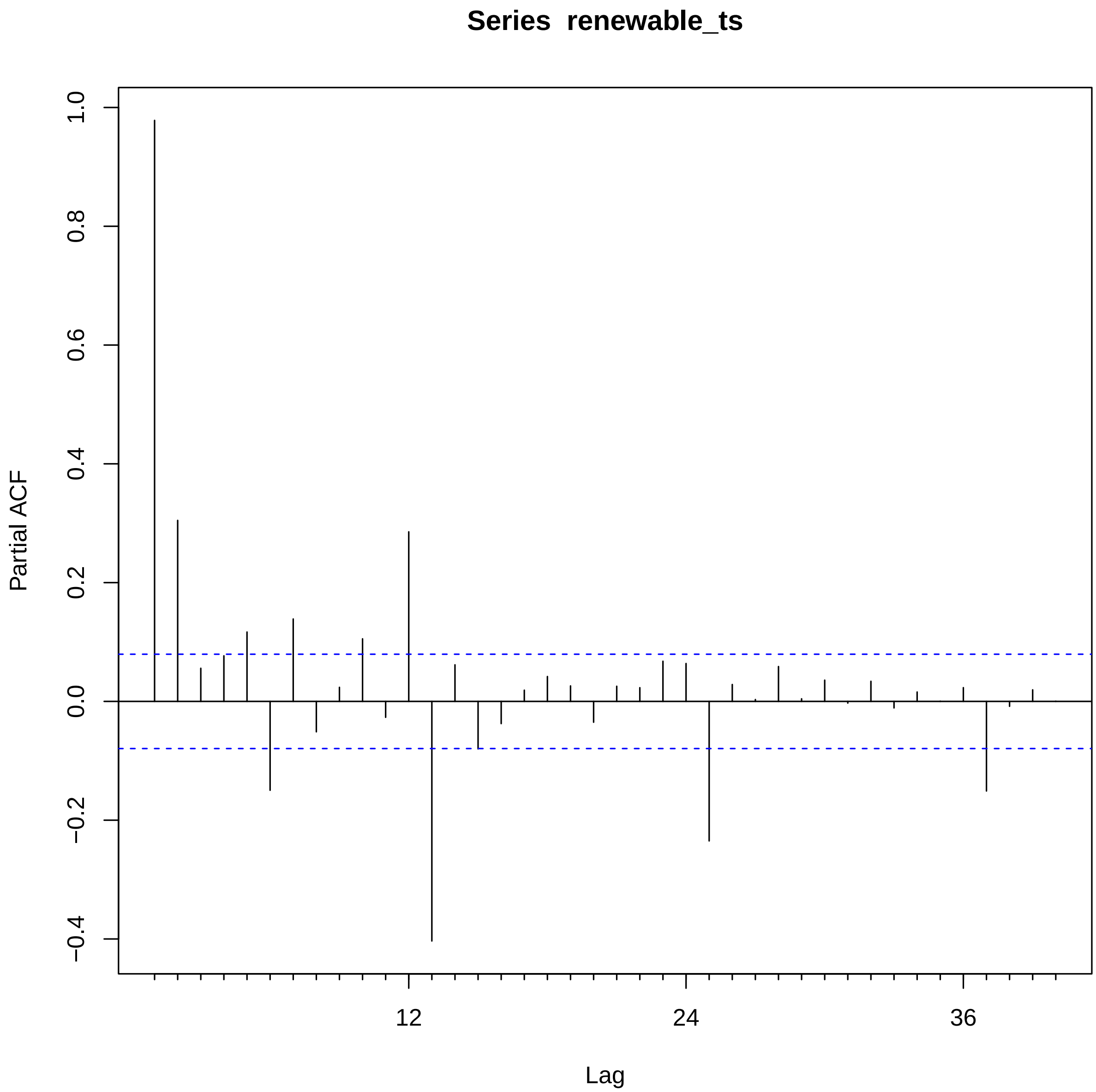
For each time series, i.e., Renewable Energy Production and Hydroelectric Consumption create three plots: one with time series, one with the ACF and with the PACF. You may use the some code form A2, but I want all the three plots side by side as in a grid. (Hint: use function `plot_grid()` from the `cowplot` package)

Renewable Energy Production

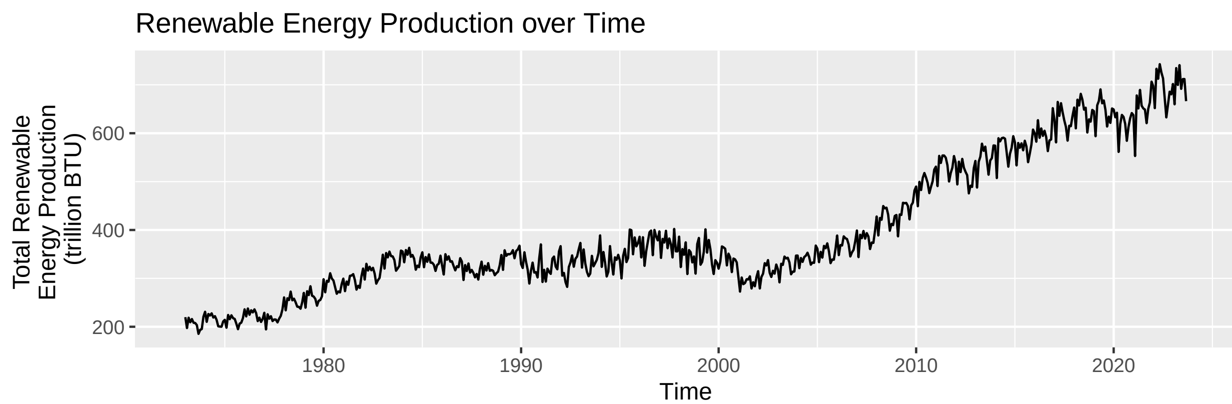
```r
#time series plot
Renew_ts_plot<- autoplot(renewable_ts) +
  labs(x = "Time",
       y = "Total Renewable \nEnergy Production \n(trillion BTU)",
       title = "Renewable Energy Production over Time")

#acf plot
Renewable_acf= autoplot(Acf(renewable_ts,lag.max=40))
```

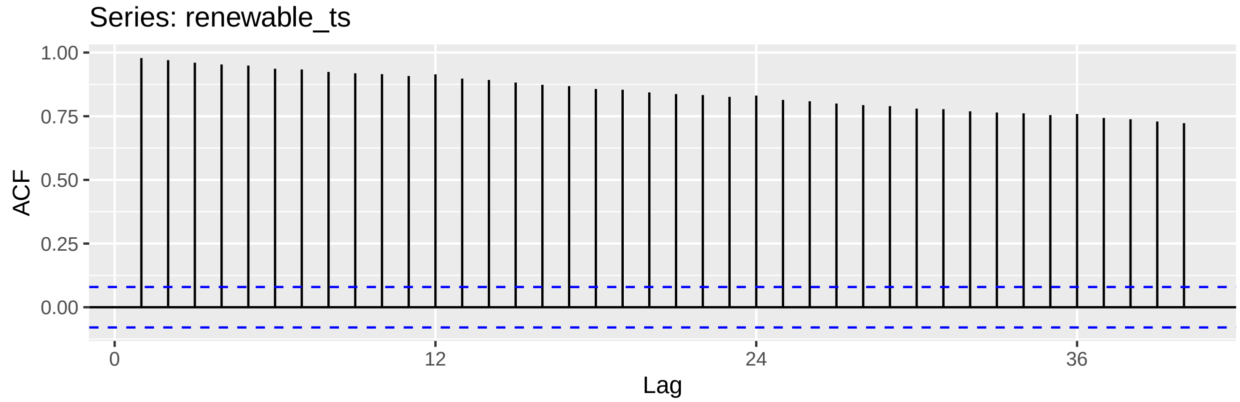


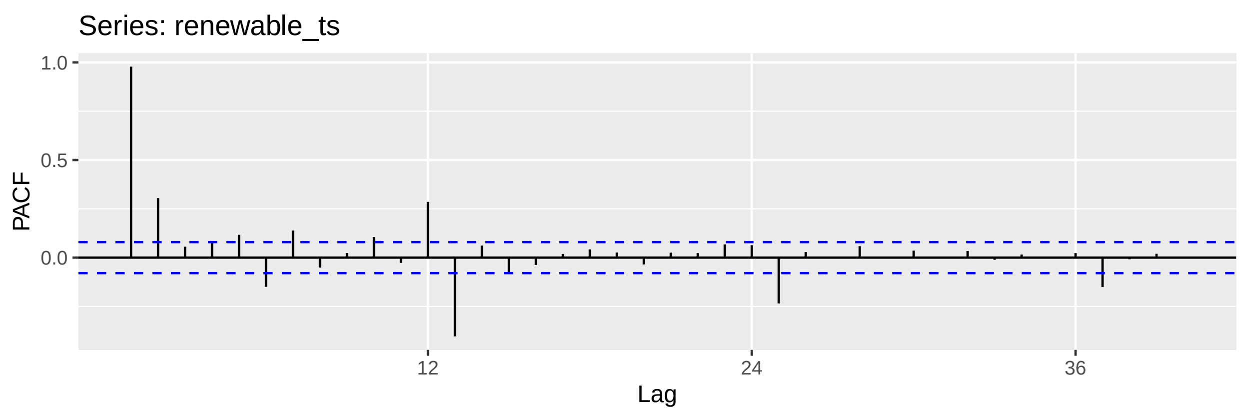

```
#pacf plot
Renewable_pacf= autoplot(Pacf(renewable_ts,lag.max=40))
```




```r
#plotting all three as grid could do 1 by 3
plot_grid(Renew_ts_plot,Renewable_acf, Renewable_pacf, nrow=3,align= 'h',
          rel_heights = c(2, 2, 2))
```

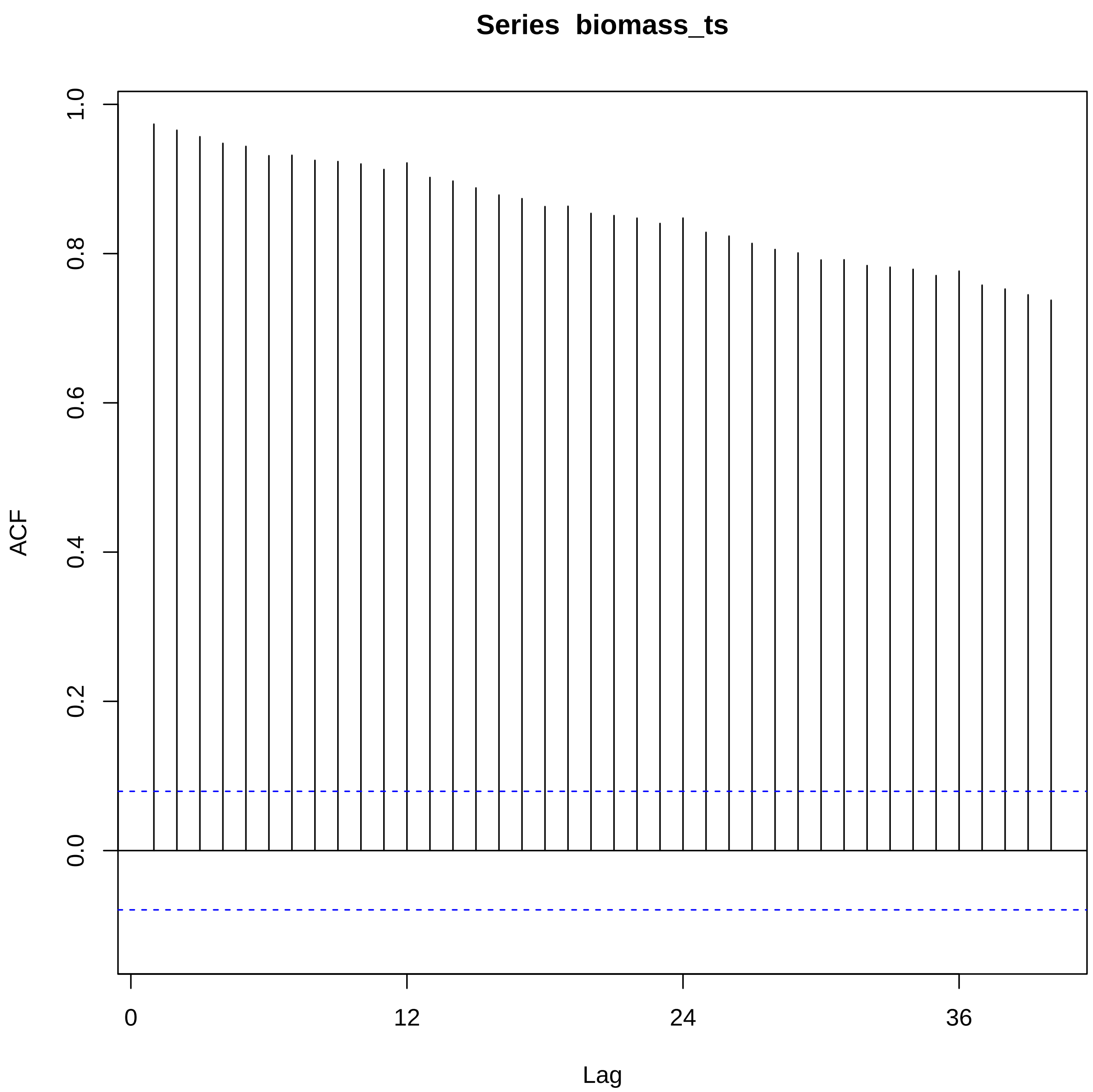

Renewable Energy Production over Time
Total Renewable Energy Production (trillion BTU)
600
400
200
1980
1990
2000
2010
2020
Time


Series: renewable_ts
1.00
0.75
0.50
0.25
0.00
ACF
0
12
24
36
Lag

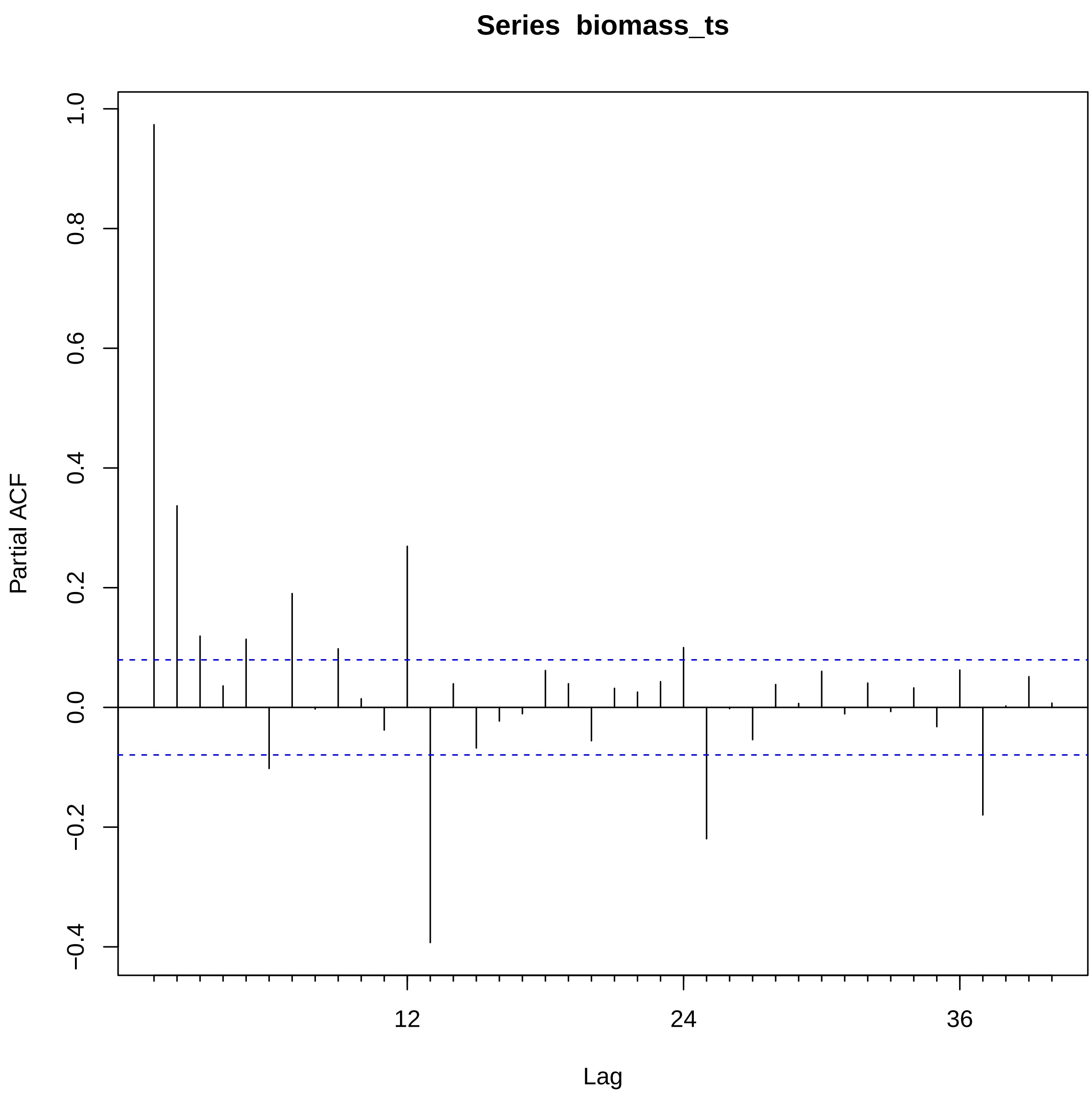

Series: renewable_ts
1.0
0.5
0.0
PACF
12
24
36
Lag

Biomass Energy Production

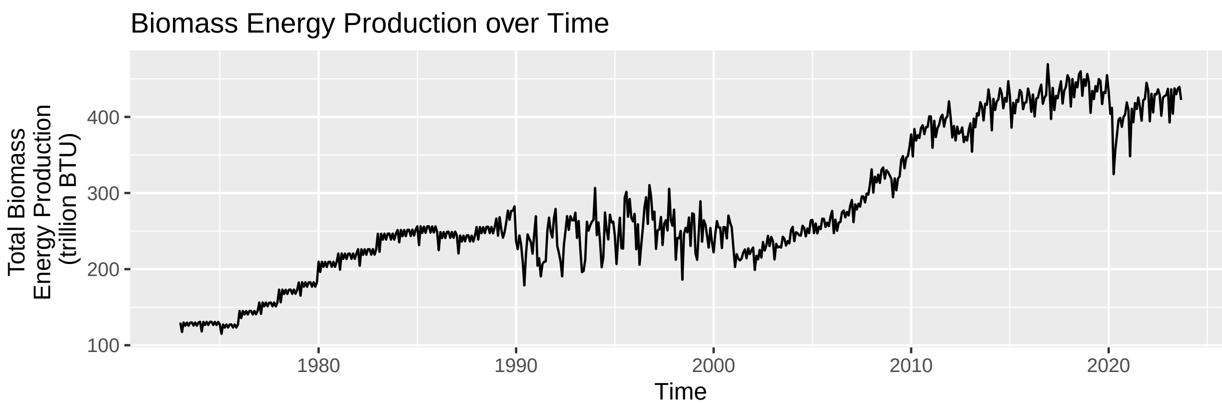
```r
#time series plot
Biomass_ts_plot<-autoplot(biomass_ts) +
  labs(x = "Time",
       y = "Total Biomass \nEnergy Production \n(trillion BTU)",
       title = "Biomass Energy Production over Time")

#acf plot
Biomass_acf= autoplot(Acf(biomass_ts,lag.max=40))
```

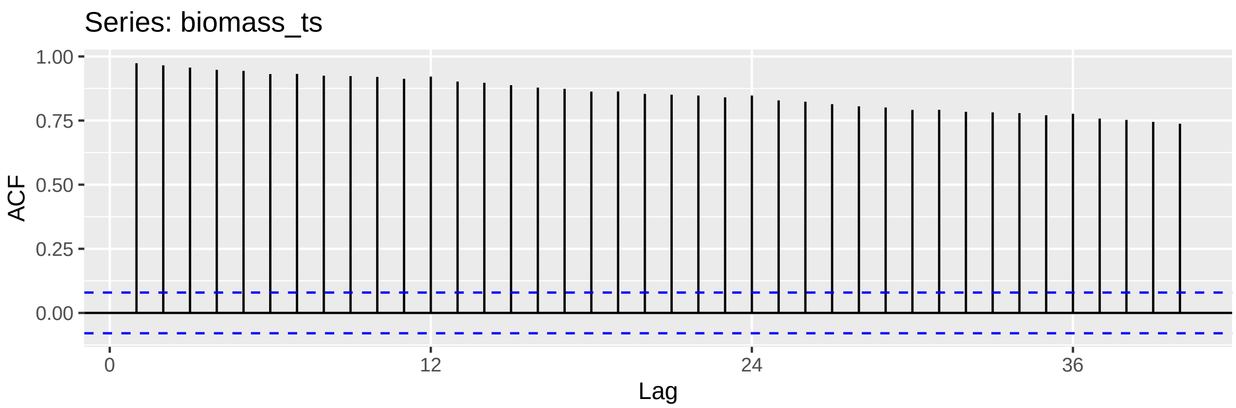


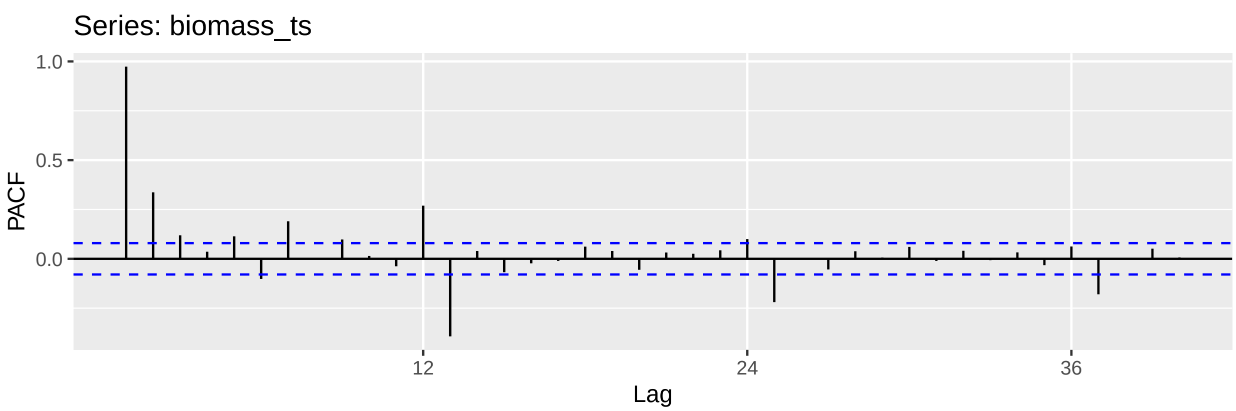
```r
#pacf plot
Biomass_pacf= autoplot(Pacf(biomass_ts,lag.max=40))
```




```r
#plotting all three as grid
plot_grid(Biomass_ts_plot,Biomass_acf, Biomass_pacf, nrow=3,align= 'h',
          rel_heights = c(2, 2, 2))
```

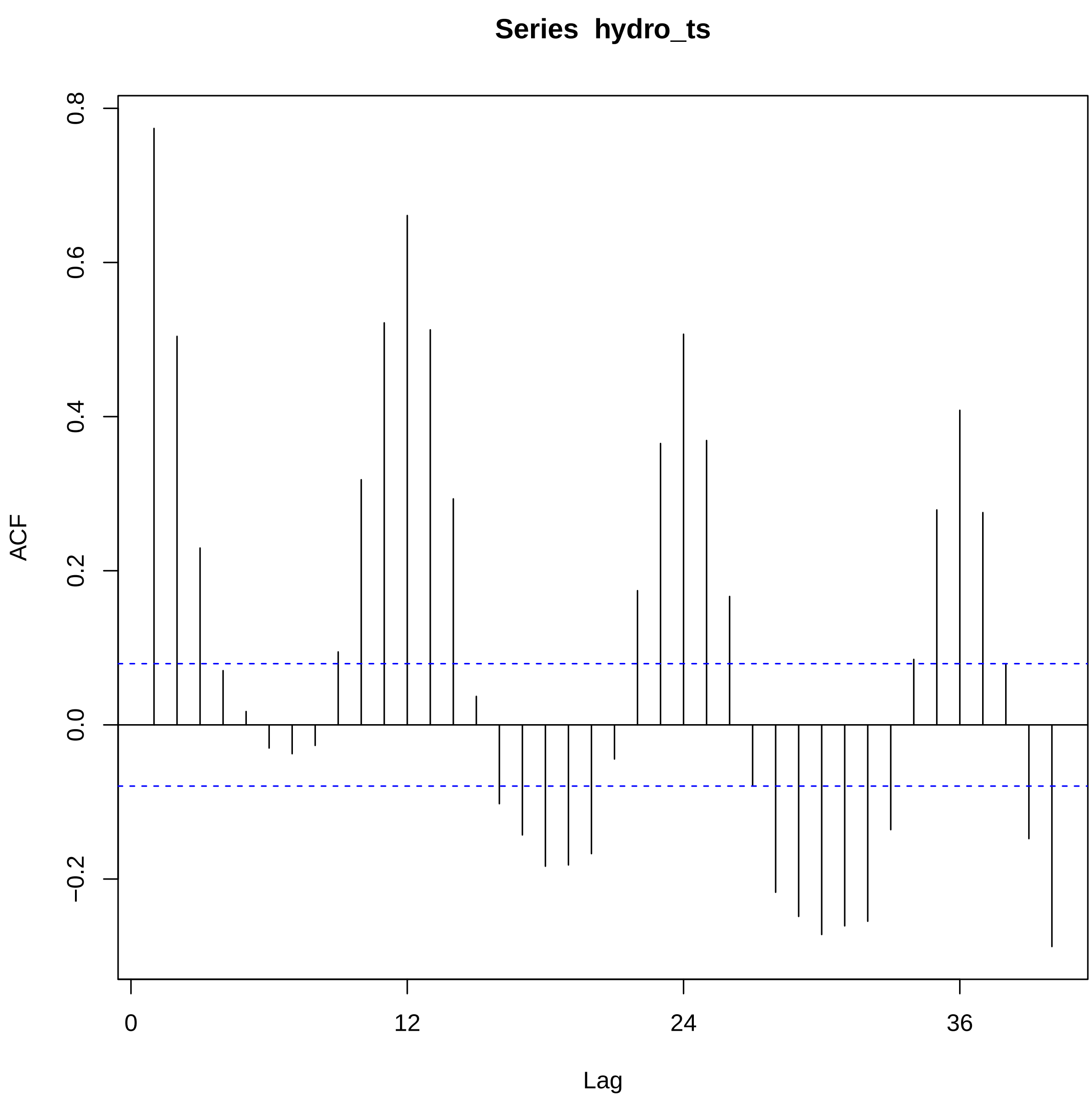

Biomass Energy Production over Time
Total Biomass
Energy Production
(trillion BTU)
100
200
300
400
1980
1990
2000
2010
2020
Time


Series: biomass_ts
ACF
0.00
0.25
0.50
0.75
1.00
0
12
24
36
Lag

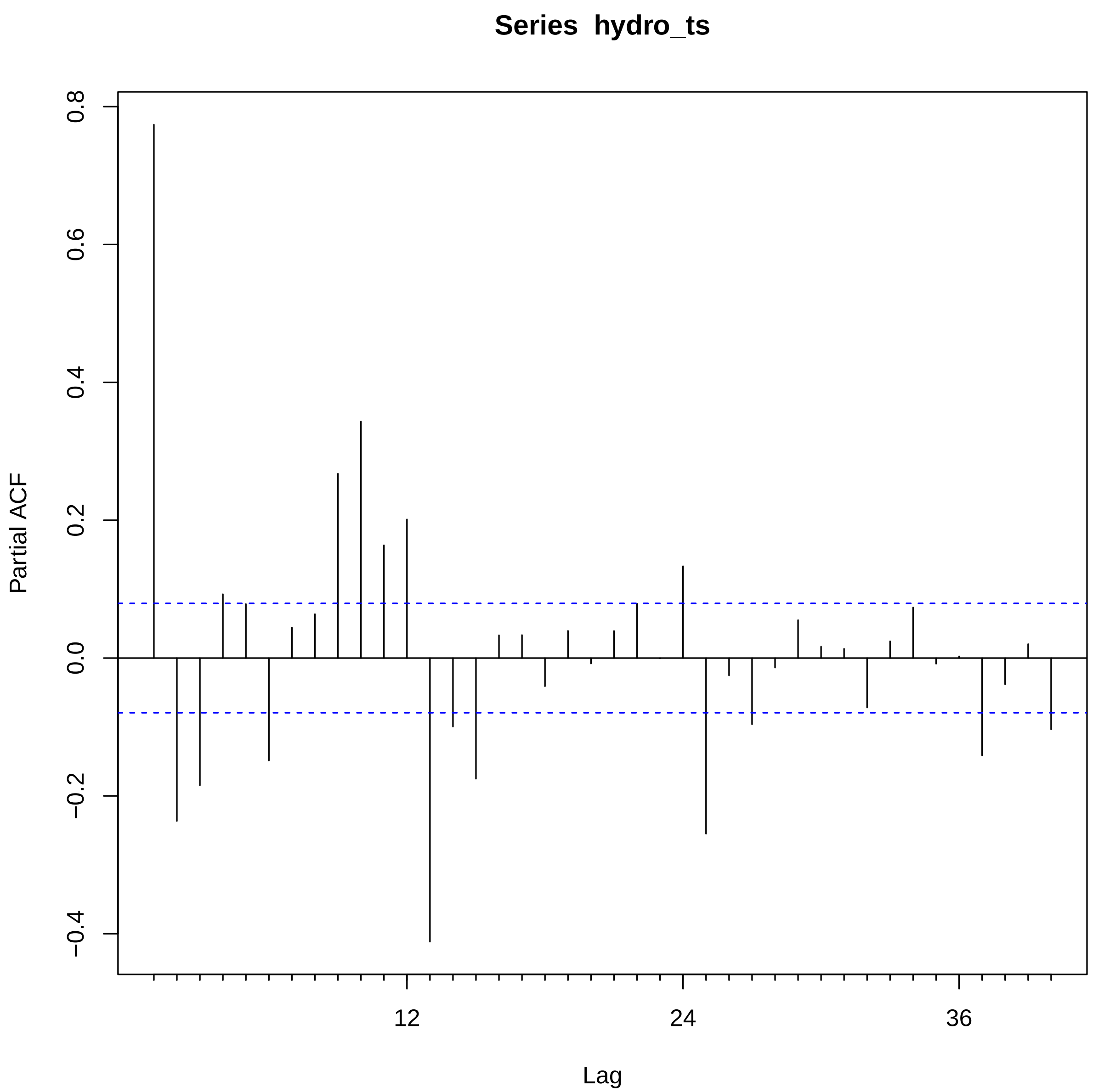

Series: biomass_ts
PACF
0.0
0.5
1.0
12
24
36
Lag

Hydroelectric Consumption

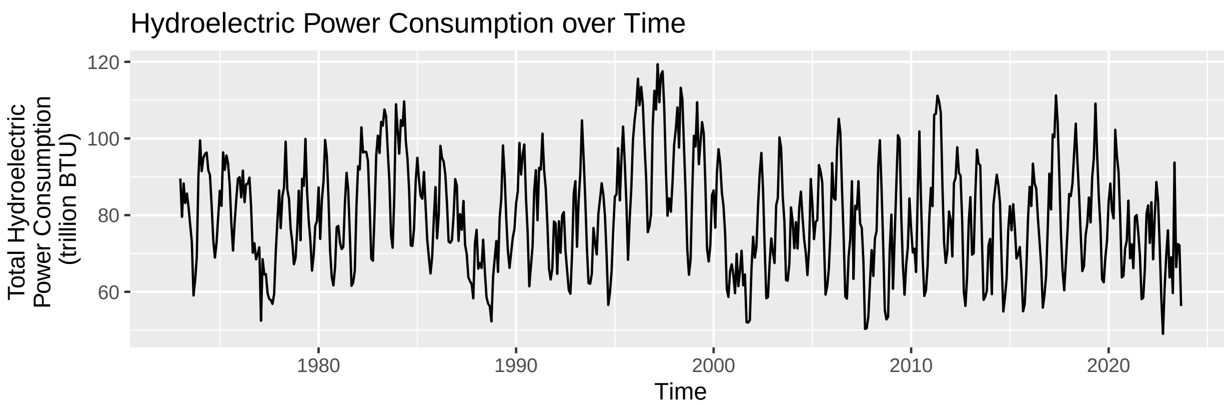
```
#time series plot
Hydro_ts_plot<-autoplot(hydro_ts) +
  labs(x = "Time",
       y = "Total Hydroelectric \nPower Consumption \n(trillion BTU)",
       title = "Hydroelectric Power Consumption over Time")

#acf plot
Hydro_acf= autoplot(Acf(hydro_ts,lag.max=40))
```

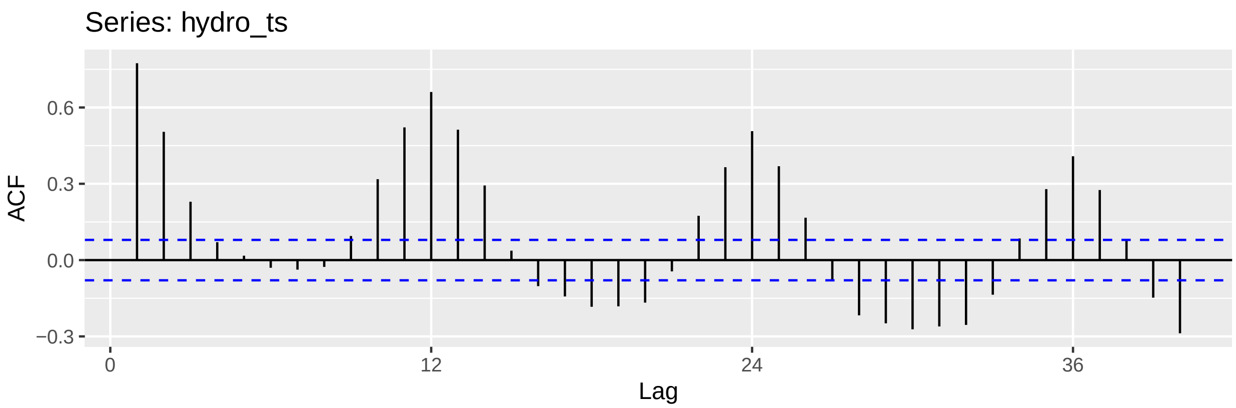


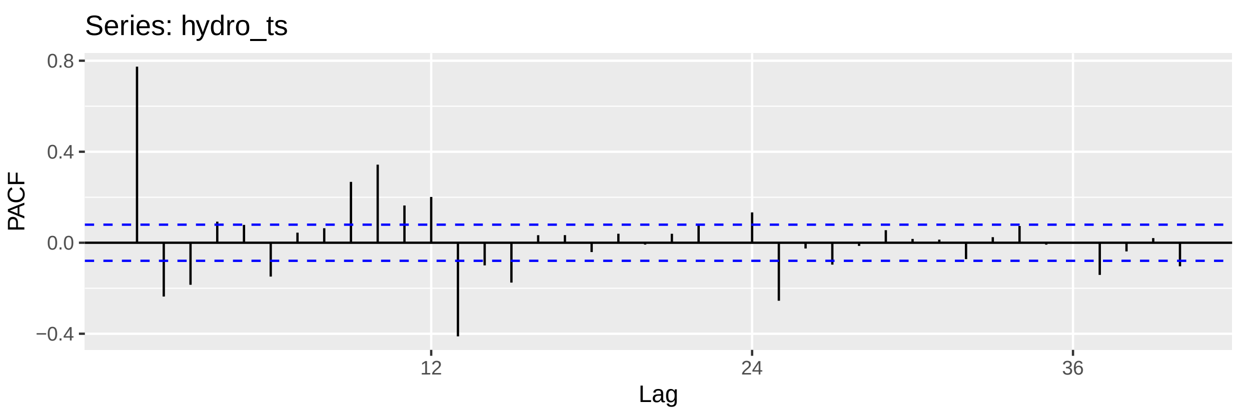
```r
#pacf plot
Hydro_pacf= autoplot(Pacf(hydro_ts,lag.max=40))
```




```r
#plotting all three as grid
plot_grid(Hydro_ts_plot,Hydro_acf, Hydro_pacf, nrow=3,align= 'h',
          rel_heights = c(2, 2,2))
```


Hydroelectric Power Consumption over Time
Total Hydroelectric Power Consumption (trillion BTU)
120
100
80
60
1980
1990
2000
2010
2020
Time


Series: hydro_ts
ACF
0.6
0.3
0.0
−0.3
0
12
24
36
Lag


Series: hydro_ts
PACF
0.8
0.4
0.0
−0.4
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?

> Hydroelectric Power Consumption does not appear to have a trend. It oscillates above and below the mean consistently over time, and the mean appears to be consistent. Biomass and energy production both have upwards trends. They both follow a somewhat polynomial upwards trend, that could be broken up into piecewise linear trends.

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.

```r
#number of observations
nobs <- nrow(energy_interest)

#Create vector t
t<- 1:nobs

#from tibble to df
energy_interest<- as.data.frame(energy_interest)

#empty vectors
beta0 <- numeric(3)
beta1 <- numeric(3)

for (i in 1:3) {
  #define linear trend based on interested energy type
  linear_trend <- lm(energy_interest[, i] ~ t)
  print(summary(linear_trend))
  beta0[i] <- coef(linear_trend)[1]  # intercept
  beta1[i] <- coef(linear_trend)[2]  # slope
}
```

```
##
## Call:
## lm(formula = energy_interest[, i] ~ t)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -102.344  -23.754    5.491   31.980   83.154
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 134.27841    3.18601   42.15   <2e-16 ***
## t             0.47713    0.00905   52.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 39.26 on 607 degrees of freedom
## Multiple R-squared:  0.8208, Adjusted R-squared:  0.8205
## F-statistic:  2780 on 1 and 607 DF,  p-value: < 2.2e-16
##
##
## Call:
## lm(formula = energy_interest[, i] ~ t)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -148.27  -35.63   11.58   41.51  144.27
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 180.98940    4.90151   36.92   <2e-16 ***
## t             0.70404    0.01392   50.57   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 60.41 on 607 degrees of freedom
## Multiple R-squared:  0.8081, Adjusted R-squared:  0.8078
## F-statistic:  2557 on 1 and 607 DF,  p-value: < 2.2e-16
##
##
## Call:
## lm(formula = energy_interest[, i] ~ t)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.818 -10.620  -0.669   9.357  39.528
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 82.734747   1.140265  72.557  < 2e-16 ***
## t           -0.009849   0.003239  -3.041  0.00246 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.05 on 607 degrees of freedom
## Multiple R-squared:  0.015,  Adjusted R-squared:  0.01338
## F-statistic: 9.247 on 1 and 607 DF,  p-value: 0.002461
```

```r
#intercepts
beta0
```

```
## [1] 134.27841 180.98940  82.73475
```

```r
#slopes
beta1
```

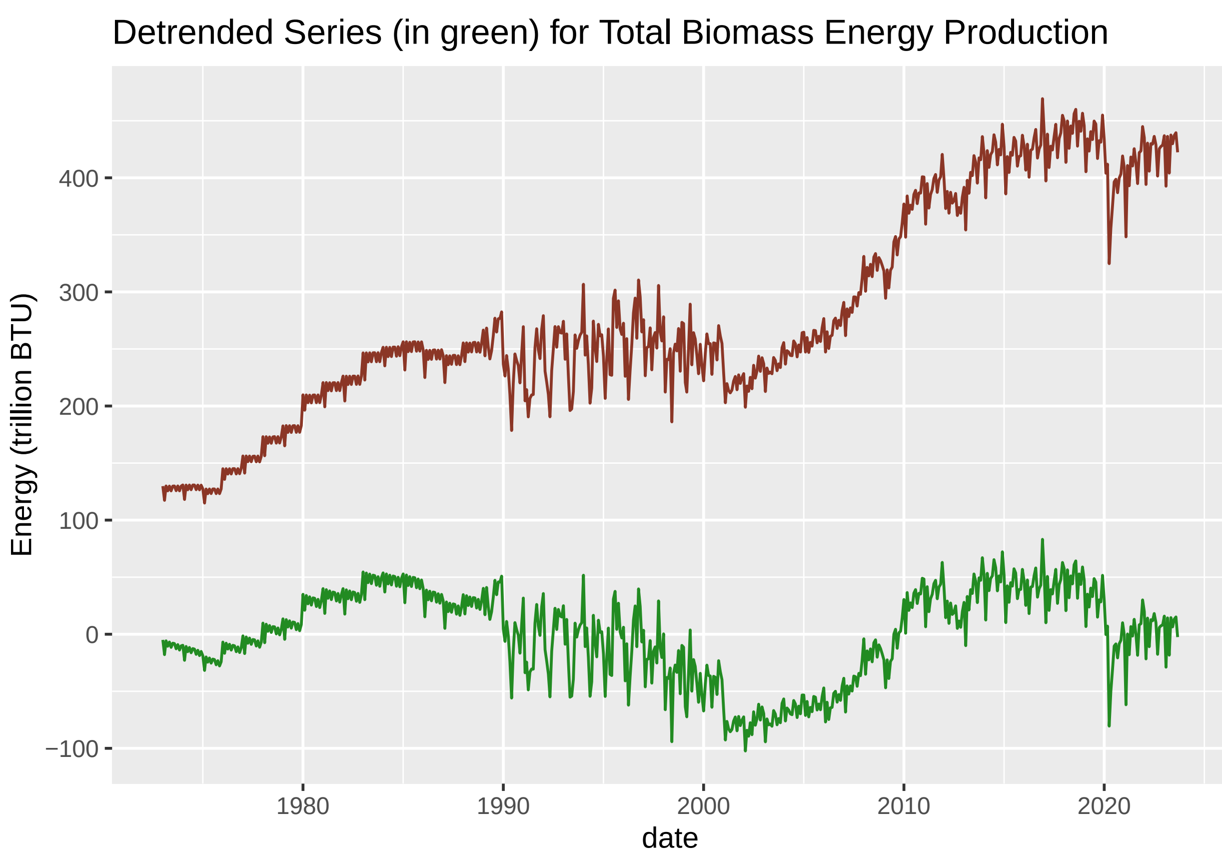
```
## [1]  0.477134961  0.704039133 -0.009849298
```

The slopes stored in beta1 represent how much energy production and consumption is increasing for each time step. For Biomass Energy Production the p-value is <0.05 so the slope of 0.477 is significant, and the r2 of 0.82 means that this linear trend accounts for 82% of the variability in the data. The trend is defined by a linear model of y=0.477x+134.3. For Total Renewable Energy Production the p-value is also <0.05 so the slope of 0.7 is significant, and the r2 of 0.807 means that this linear trend accounts for 80.7% of the variability in the data. The trend is defined by a linear model of y=0.7x+181. For the Hydroelectric Power Consumption the p-value is <0.05 so the slope of -0.009 is significant (though the magnitude is very small). But the r2 is 0.01, so this linear trend accounts for only 1% of the variability in the data. The trend is defined by a linear model of y=-0.009x+82.7.

Q4

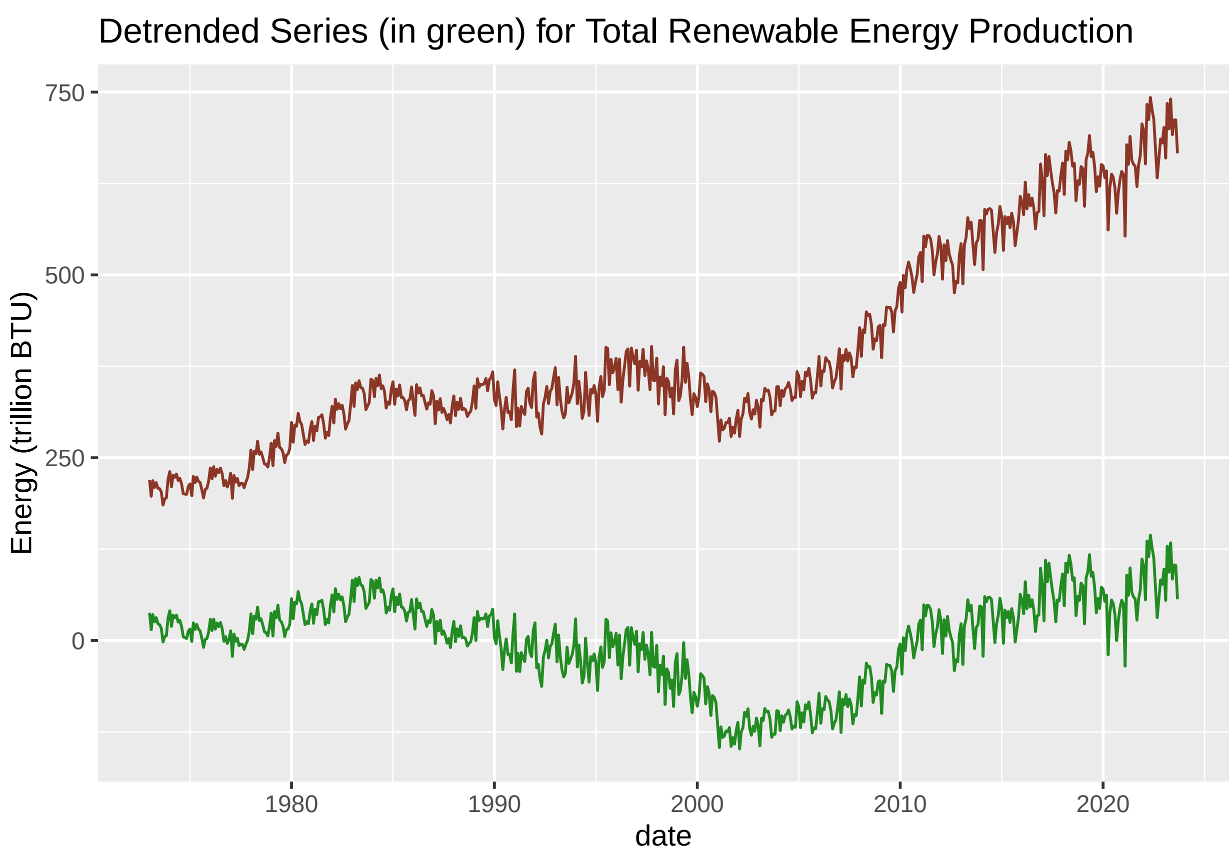
Use the regression coefficients from Q3 to detrend the series. Plot the detrended series and compare with the plots from Q1. What happened? Did anything change?

```
#Total Biomass
i=1
  y_detrend_bio <- energy_interest[, i] - (beta0[i]+beta1[i]*t)
  df_detrend_bio <- data.frame("date"=energy_data1$Month,
                          "observed"=energy_interest[, i],
                          "detrend"= y_detrend_bio)
  detrended_series_bio <- ggplot(df_detrend_bio, aes(x=date))+
  geom_line(aes(y=observed), color="tomato4")+
  geom_line(aes(y=detrend), color= "forestgreen") +
  ggtitle(paste("Detrended Series (in green) for", colnames(energy_interest)[i])) +
    labs(y= "Energy (trillion BTU)")
  plot(detrended_series_bio)
```

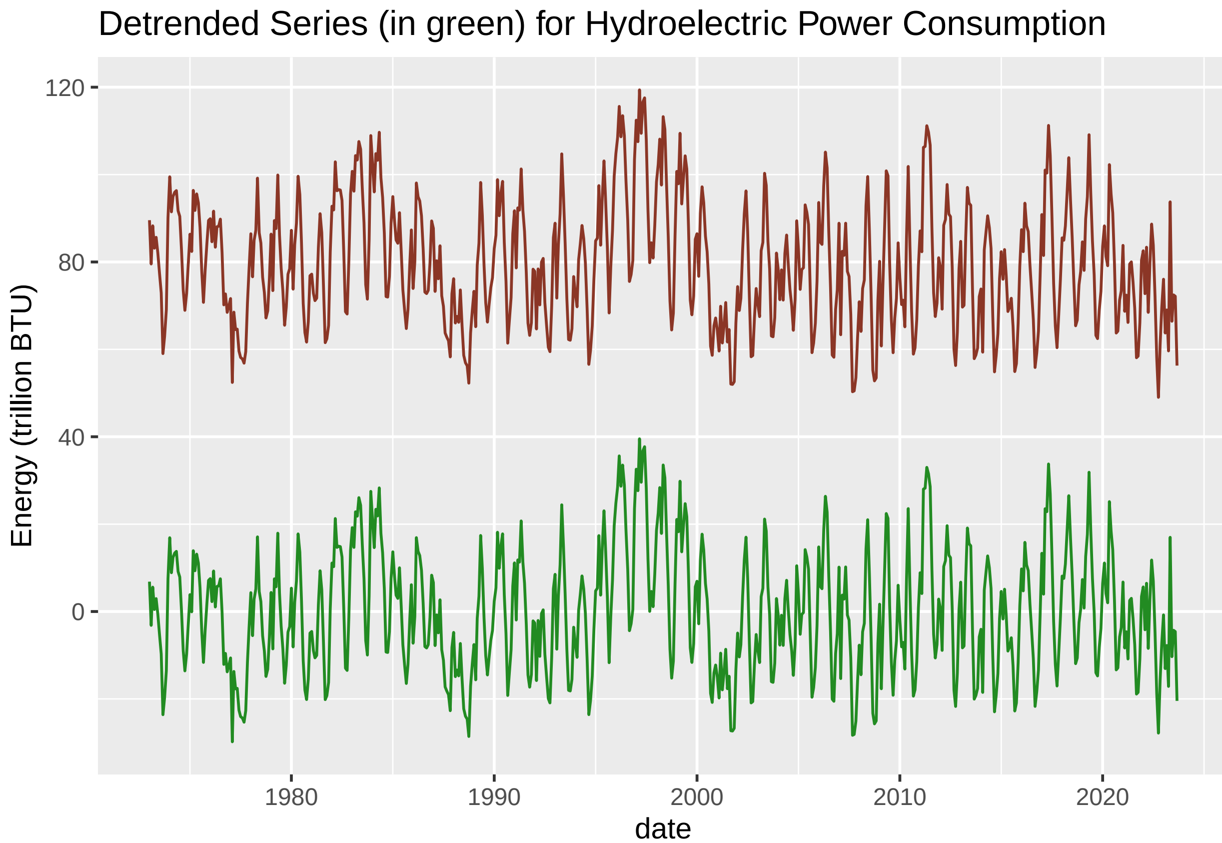



```
#Total Renewable
  i=2
  y_detrend_renew <- energy_interest[, i] - (beta0[i]+beta1[i]*t)
  df_detrend_renew <- data.frame("date"=energy_data1$Month,
                          "observed"=energy_interest[, i],
                          "detrend"= y_detrend_renew)
  detrended_series_renew <- ggplot(df_detrend_renew, aes(x=date))+
```

```r
  geom_line(aes(y=observed), color="tomato4")+
  geom_line(aes(y=detrend), color= "forestgreen") +
  ggtitle(paste("Detrended Series (in green) for", colnames(energy_interest)[i])) +
    labs(y= "Energy (trillion BTU)")
  plot(detrended_series_renew)
```




```r
#Hydroelectric
  i=3
  y_detrend_hydro <- energy_interest[, i] - (beta0[i]+beta1[i]*t)
  df_detrend_hydro <- data.frame("date"=energy_data1$Month,
                           "observed"=energy_interest[, i],
                           "detrend"= y_detrend_hydro)
  detrended_series_hydro <- ggplot(df_detrend_hydro, aes(x=date))+
  geom_line(aes(y=observed), color="tomato4")+
  geom_line(aes(y=detrend), color= "forestgreen") +
  ggtitle(paste("Detrended Series (in green) for", colnames(energy_interest)[i])) +
    labs(y= "Energy (trillion BTU)")
  plot(detrended_series_hydro)
```



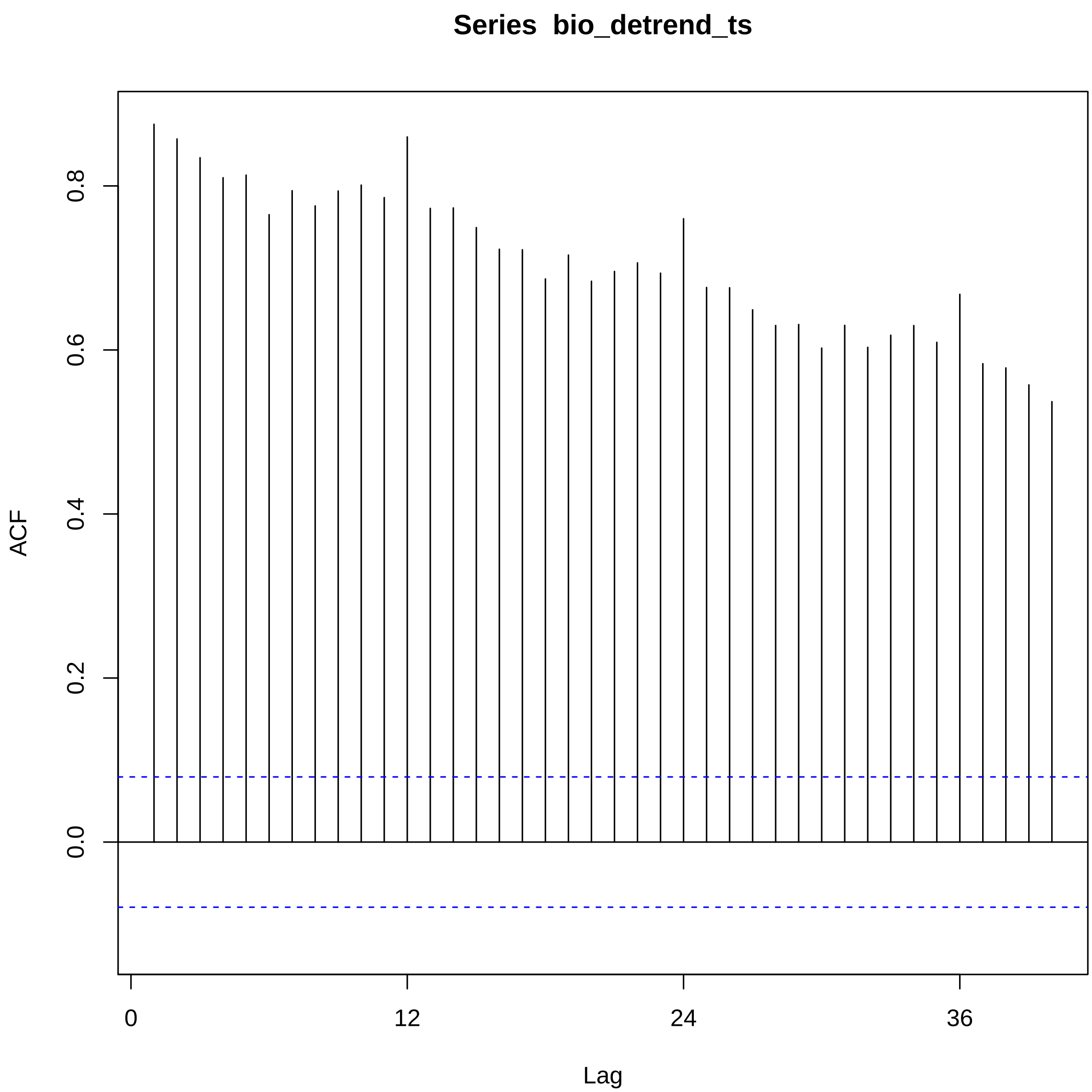
> The detrended series for total renewable has shifted down, the intercept is now close to zero, and the range has decreased (the difference between the maximum and minimum values). > The detrended series for hydroelectric power consumption has also shifted down with the intercept being closer to zero. The data seem to have a mean of around 0 now. But the range between the max and min values seems to have stayed the same.

Q5

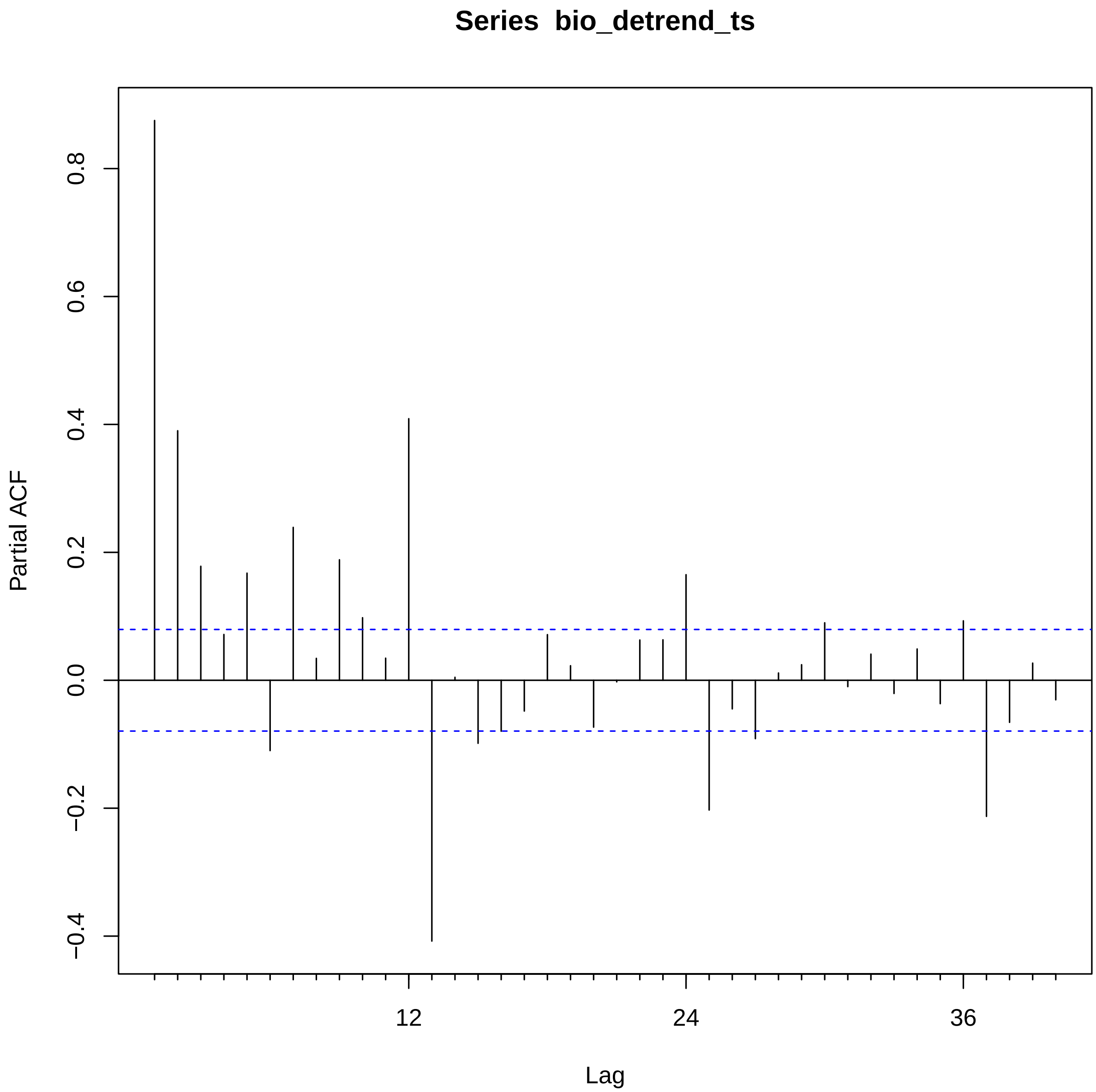
Plot ACF and PACF for the detrended series and compare with the plots from Q1. You may use plot_grid() again to get them side by side. but not mandatory. Did the plots change? How?

```
#creating timeseries objects for all of the detrended series
bio_detrend_ts <- ts(y_detrend_bio, start = c(1973,1), frequency = 12)
renew_detrend_ts <-ts(y_detrend_renew, start = c(1973,1), frequency = 12)
hydro_detrend_ts <- ts(y_detrend_hydro, start = c(1973,1), frequency = 12)
```

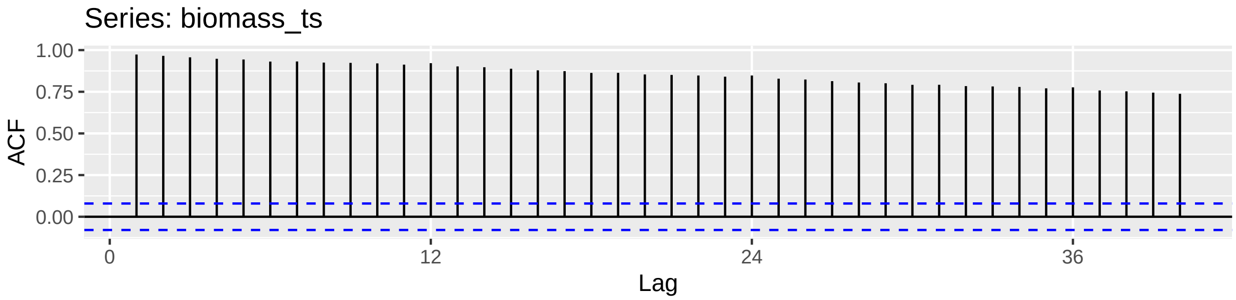
```
#acf detrended plot for biomass
bio_acf_detrend= autoplot(Acf(bio_detrend_ts,lag.max=40))
```

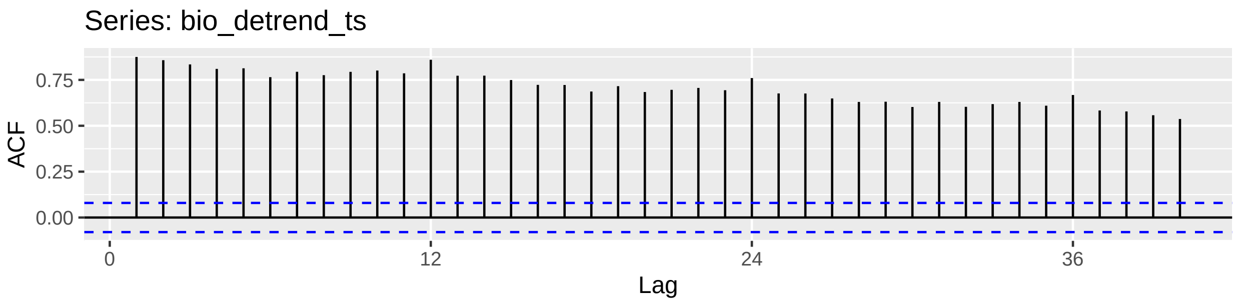


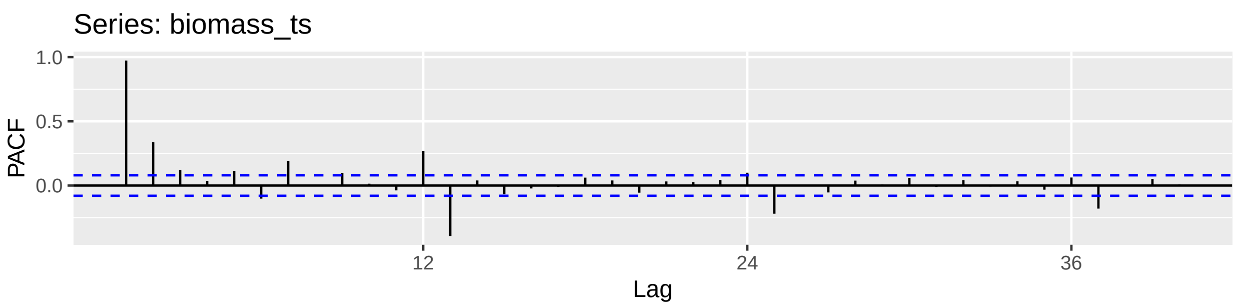
```
#pacf detrendedplot for biomass
bio_pacf_detrend= autoplot(Pacf(bio_detrend_ts,lag.max=40))
```

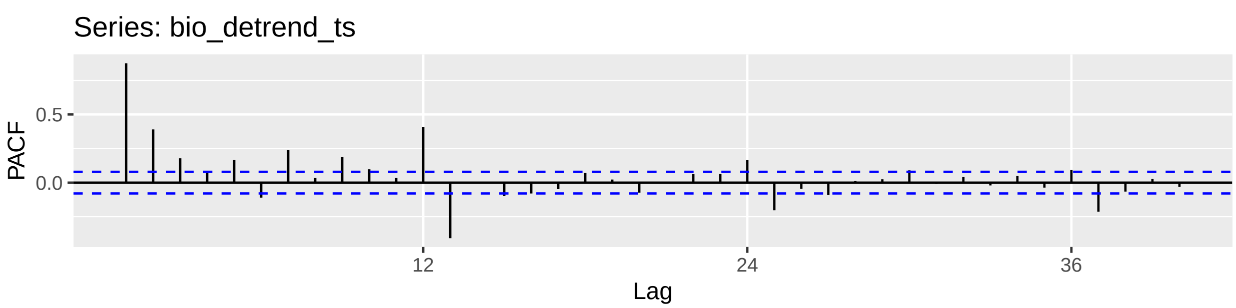



```
#plotting comparison with plots from #1 as grid
plot_grid(Biomass_acf, bio_acf_detrend,Biomass_pacf, bio_pacf_detrend, nrow=4,
          align= 'h', rel_heights = c(2, 2,2,2))
```

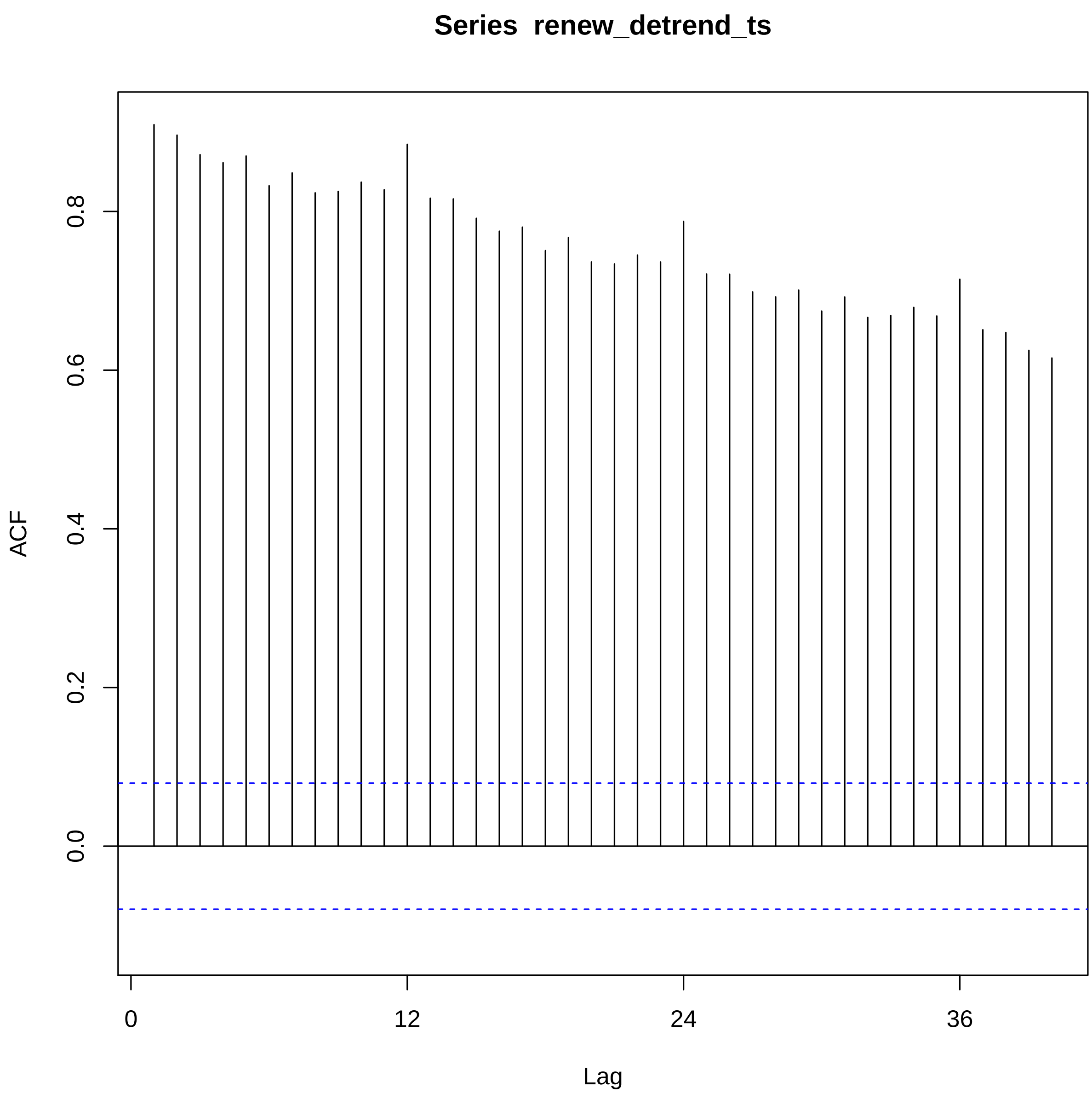

Series: biomass_ts
ACF
1.00
0.75
0.50
0.25
0.00
0
12
24
36
Lag


Series: bio_detrend_ts
ACF
0.75
0.50
0.25
0.00
0
12
24
36
Lag

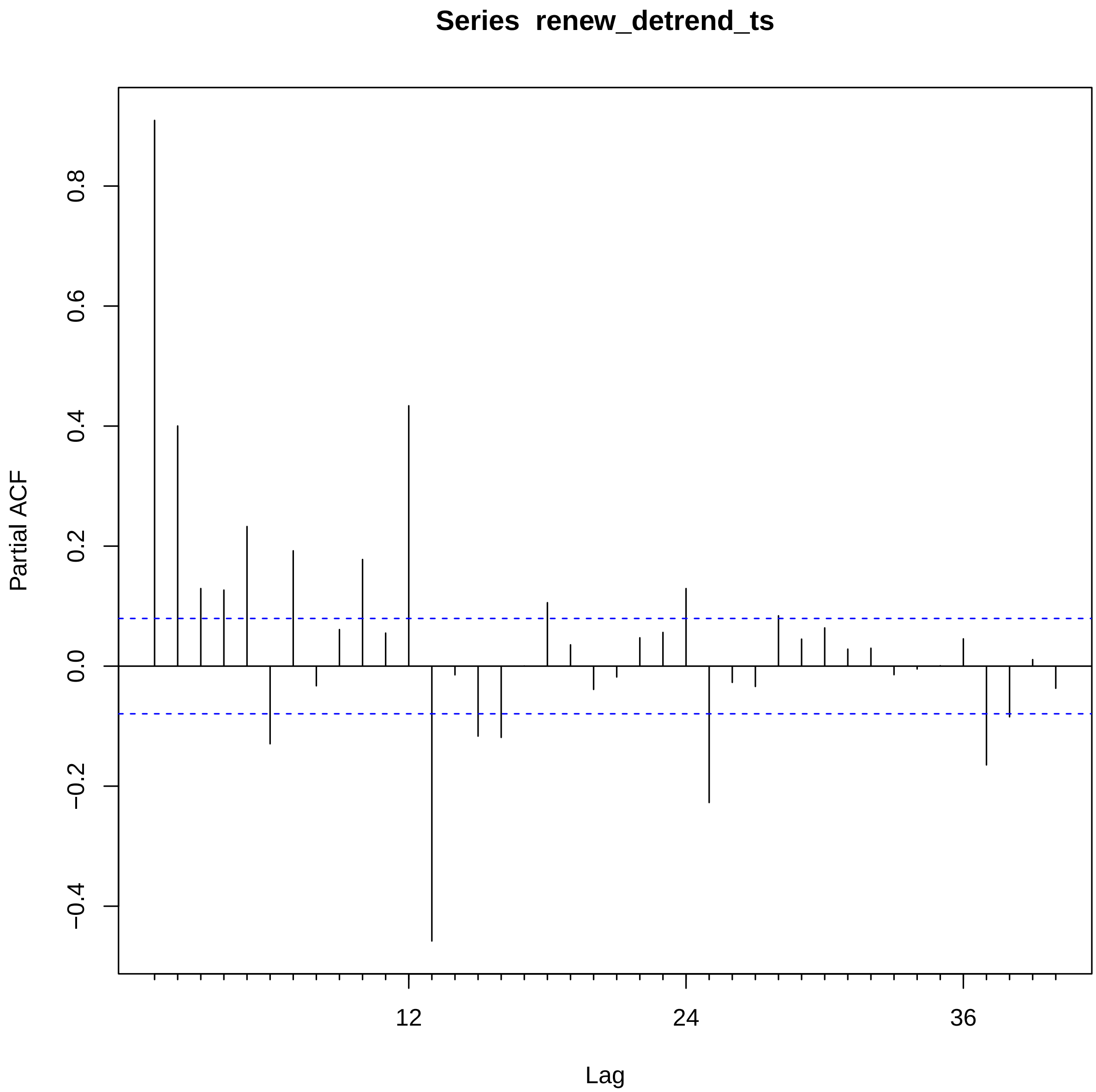

Series: biomass_ts
PACF
1.0
0.5
0.0
12
24
36
Lag


Series: bio_detrend_ts
PACF
0.5
0.0
12
24
36
Lag

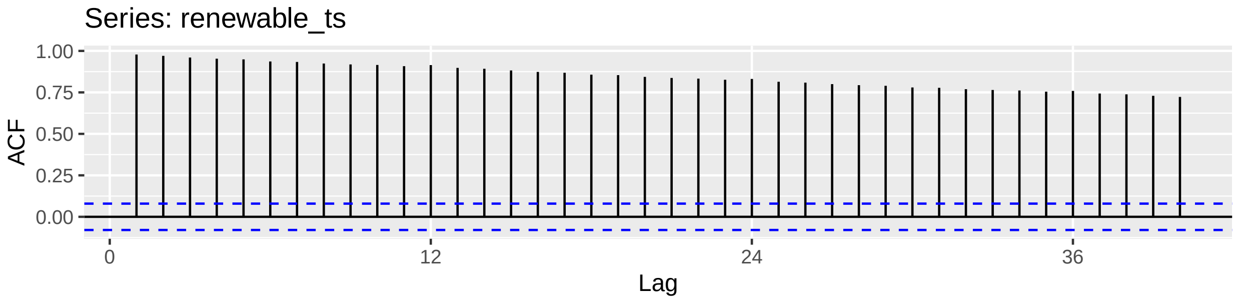
```
#acf detrended plot for renewables
Renew_acf_detrend= autoplot(Acf(renew_detrend_ts,lag.max=40))
```

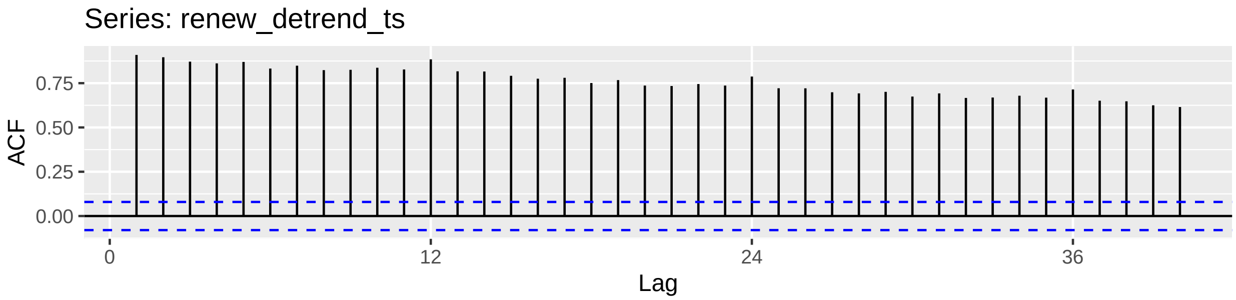


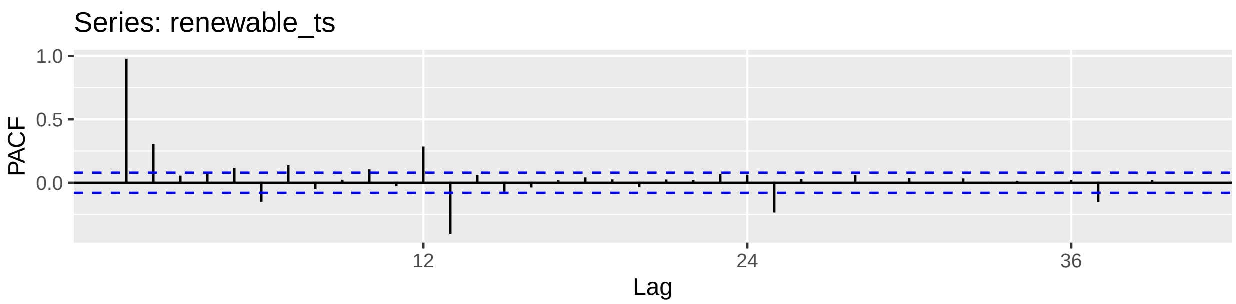

```
#pacf detrendedplot for renewables
Renew_pacf_detrend= autoplot(Pacf(renew_detrend_ts,lag.max=40))
```

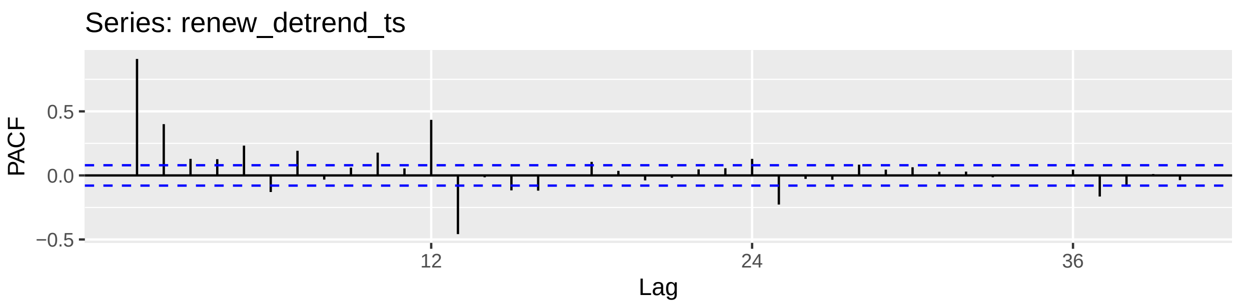



```r
#plotting comparison with plots from #1 as grid
plot_grid(Renewable_acf, Renew_acf_detrend,Renewable_pacf, Renew_pacf_detrend,
          nrow=4,align= 'h', rel_heights = c(2, 2,2,2))
```

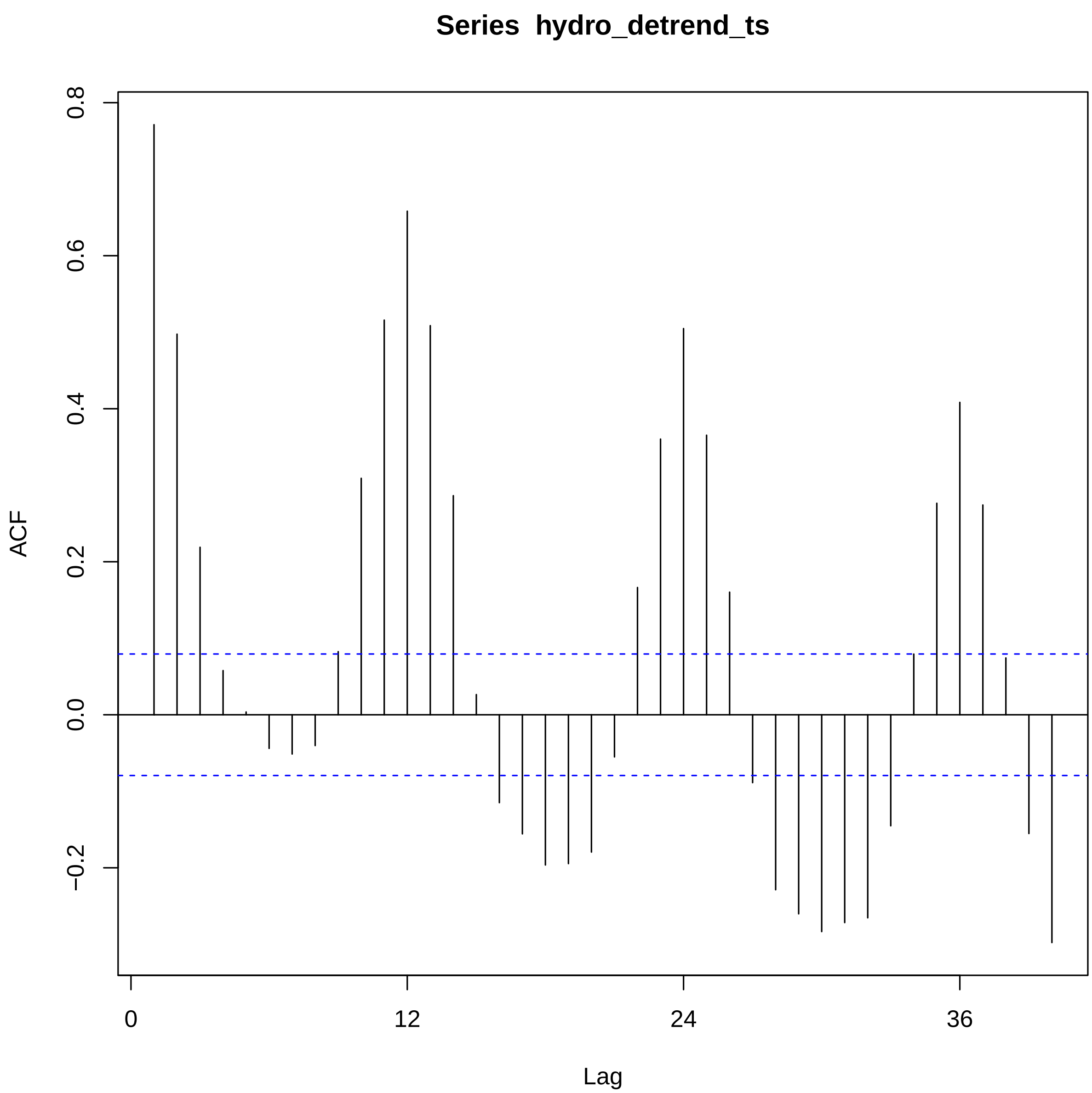

Series: renewable_ts
ACF
1.00
0.75
0.50
0.25
0.00
0
12
24
36
Lag


Series: renew_detrend_ts
ACF
0.75
0.50
0.25
0.00
0
12
24
36
Lag

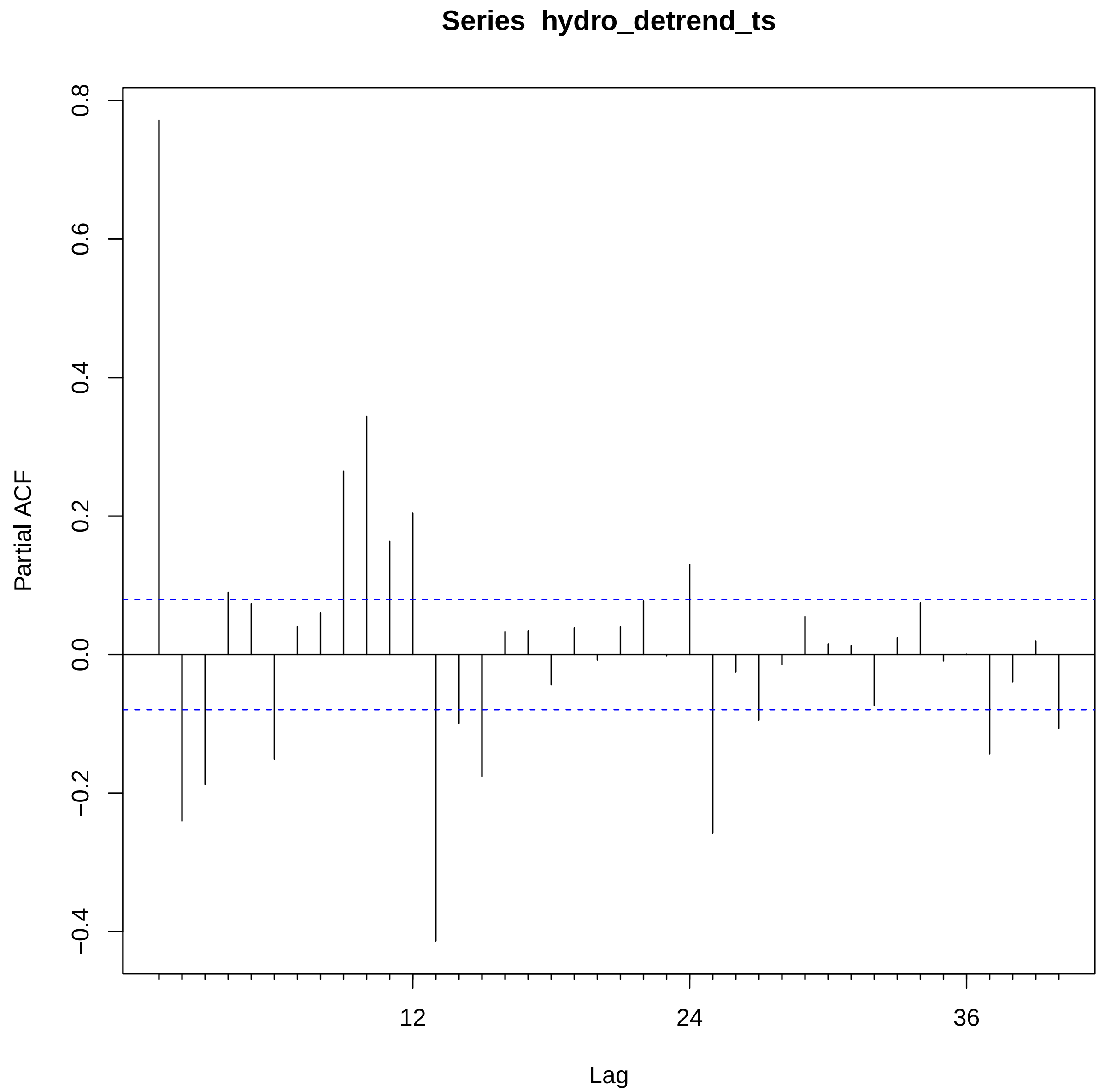

Series: renewable_ts
PACF
1.0
0.5
0.0
12
24
36
Lag


Series: renew_detrend_ts
PACF
0.5
0.0
−0.5
12
24
36
Lag

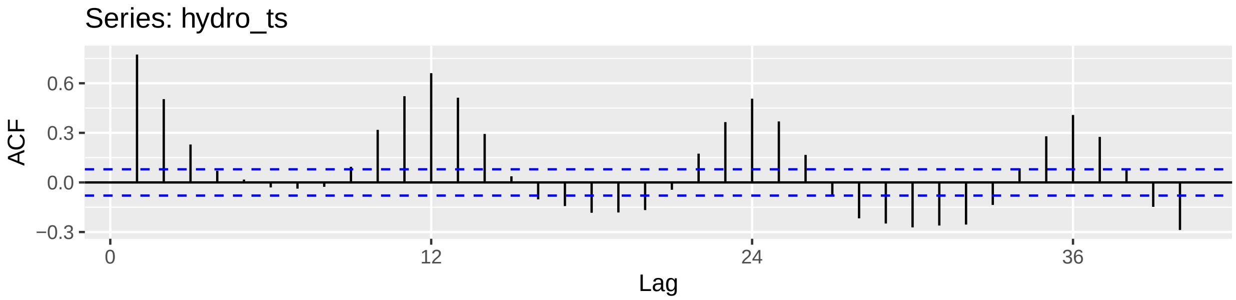
```r
#acf detrended plot for hydro
Hydro_acf_detrend= autoplot(Acf(hydro_detrend_ts,lag.max=40))
```

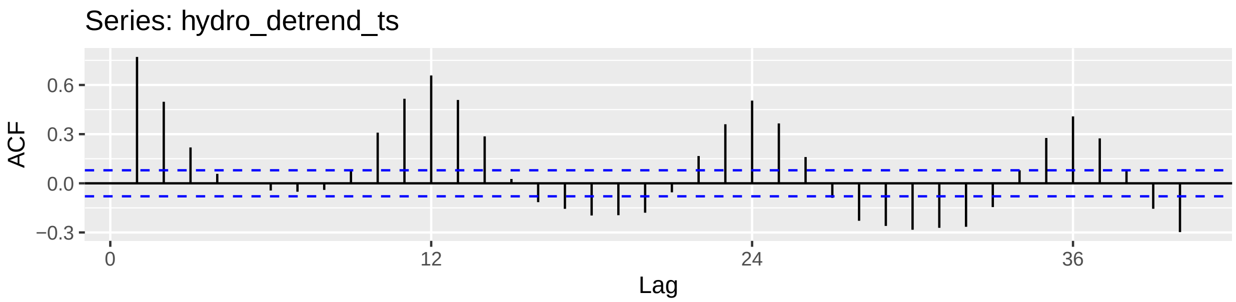


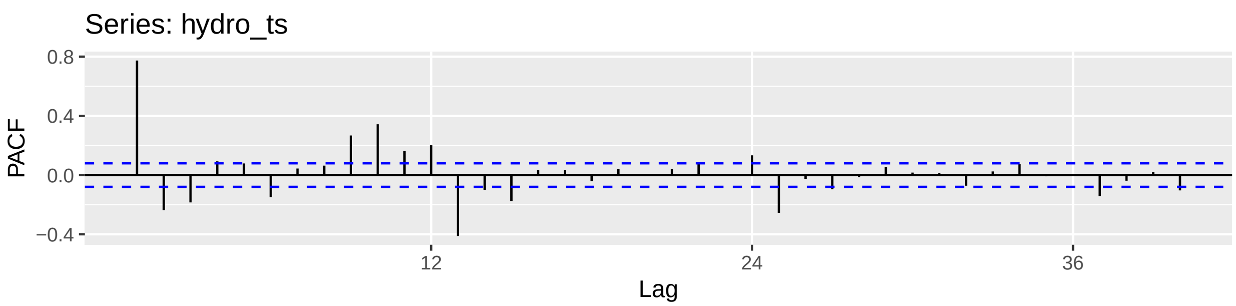

```r
#pacf detrendedplot for hydro
Hydro_pacf_detrend= autoplot(Pacf(hydro_detrend_ts,lag.max=40))
```

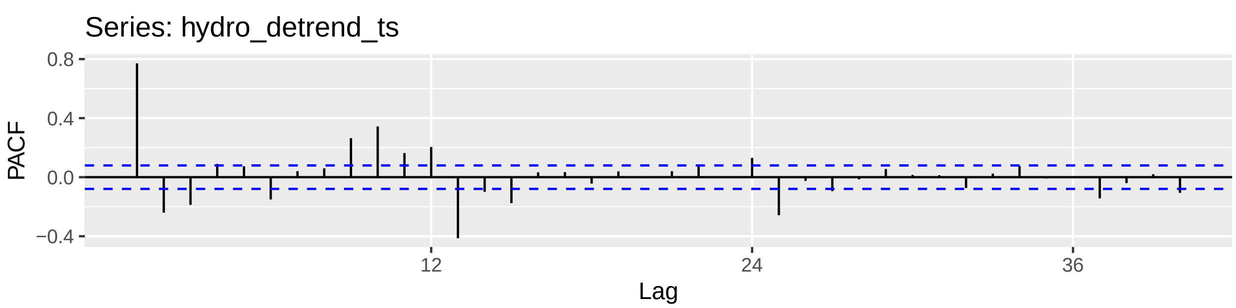



```r
#plotting comparison with plots from #1 as grid
plot_grid(Hydro_acf, Hydro_acf_detrend,Hydro_pacf, Hydro_pacf_detrend,
          nrow=4,align= 'h', rel_heights = c(2, 2,2,2))
```









The renewable graph changed a little bit, because we removed a signficant amount of variation from the data when removing the trend. We are now able to see really weak seasonality in the renewable ACF detrended data because there are very small spikes at 12, 24, 36. For the hydro graphs we dont have a strong upward or downward trend, and see strong seasonaility in the ACF. For the pacf we see the highest magnitude partial autocorrelation at consistently spaced lags (13, 25, 37) which indicates there is a seasonal component. But there is no notable change in these graphs from the original to the detrended data.

Seasonal Component

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

Q6

Just by looking at the time series and the acf plots, do the series seem to have a seasonal trend? No need to run any code to answer your question. Just type in you answer below.

> The Renewable energy series does not appear to have a strong seasonal trend. There are small peaks in the ACF graph that repeat every 12 months, but I predict this is due to weak seasonality. The Hydroelectric series does seem to have a strong seasonal trend. There are equally spaced peaks and troughs. The ACF plot also looks periodic, with cycles repeating every 12 month lag.

Q7

Use function *lm()* to fit a seasonal means model (i.e. using the seasonal dummies) the two time series. Ask R to print the summary of the regression. Interpret the regression output. From the results which series have a seasonal trend? Do the results match you answer to Q6?

```r
#seasonal dummies for hydroelectric and total renewable
dummies_hydro = seasonaldummy(hydro_ts)
dummies_renew = seasonaldummy(renewable_ts)

#creating seasonal means model for both
linear_trend_seasonal_hydro <- lm(hydro_ts ~ dummies_hydro)
  print(summary(linear_trend_seasonal_hydro))
```

```
##
## Call:
## lm(formula = hydro_ts ~ dummies_hydro)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -31.323  -5.849  -0.468   6.243  32.290
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        80.282      1.470  54.601  < 2e-16 ***
## dummies_hydroJan    4.807      2.069   2.323  0.02050 *
## dummies_hydroFeb   -2.725      2.069  -1.317  0.18831
## dummies_hydroMar    6.825      2.069   3.298  0.00103 **
## dummies_hydroApr    5.319      2.069   2.571  0.01039 *
## dummies_hydroMay   13.922      2.069   6.729 4.02e-11 ***
## dummies_hydroJun   10.650      2.069   5.147 3.60e-07 ***
## dummies_hydroJul    3.912      2.069   1.891  0.05914 .
## dummies_hydroAug   -5.677      2.069  -2.744  0.00626 **
## dummies_hydroSep  -16.797      2.069  -8.118 2.72e-15 ***
## dummies_hydroOct  -16.468      2.079  -7.920 1.17e-14 ***
## dummies_hydroNov  -10.885      2.079  -5.235 2.29e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 10.4 on 597 degrees of freedom
## Multiple R-squared:  0.4697, Adjusted R-squared:  0.4599
## F-statistic: 48.07 on 11 and 597 DF,  p-value: < 2.2e-16
```

```
linear_trend_seasonal_renew <- lm(renewable_ts ~ dummies_renew)
  print(summary(linear_trend_seasonal_renew))
```

```
##
## Call:
## lm(formula = renewable_ts ~ dummies_renew)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -199.19  -86.35  -48.84  113.18  331.58
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        404.526     19.574  20.666   <2e-16 ***
## dummies_renewJan     2.962     27.546   0.108    0.914
## dummies_renewFeb   -34.476     27.546  -1.252    0.211
## dummies_renewMar     3.929     27.546   0.143    0.887
## dummies_renewApr    -8.695     27.546  -0.316    0.752
## dummies_renewMay     6.645     27.546   0.241    0.809
## dummies_renewJun    -4.198     27.546  -0.152    0.879
## dummies_renewJul     2.460     27.546   0.089    0.929
## dummies_renewAug    -5.026     27.546  -0.182    0.855
## dummies_renewSep   -29.119     27.546  -1.057    0.291
## dummies_renewOct   -20.068     27.682  -0.725    0.469
## dummies_renewNov   -20.346     27.682  -0.735    0.463
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 138.4 on 597 degrees of freedom
## Multiple R-squared:  0.009296,   Adjusted R-squared:  -0.008958
## F-statistic: 0.5093 on 11 and 597 DF,  p-value: 0.8976
```

> Hydroelectric Power Consumption has a seasonal trend, with 45.6% of variability in the series described by seasonal component. Renewable energy production does not have a strong seasonal component, only 0.8% of the variability in the series is described by the seasonal component. This matches my response to question 6.

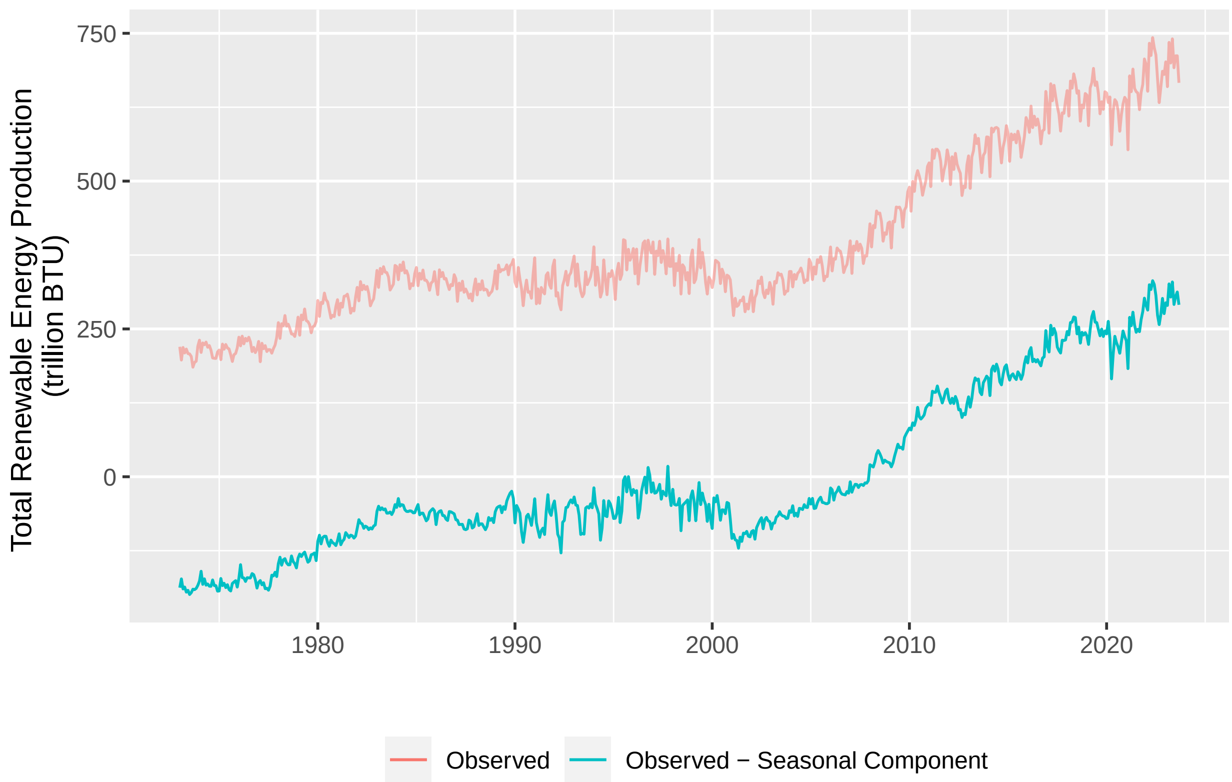
Q8

Use the regression coefficients from Q7 to deseason the series. Plot the deseason series and compare with the plots from part Q1. Did anything change?

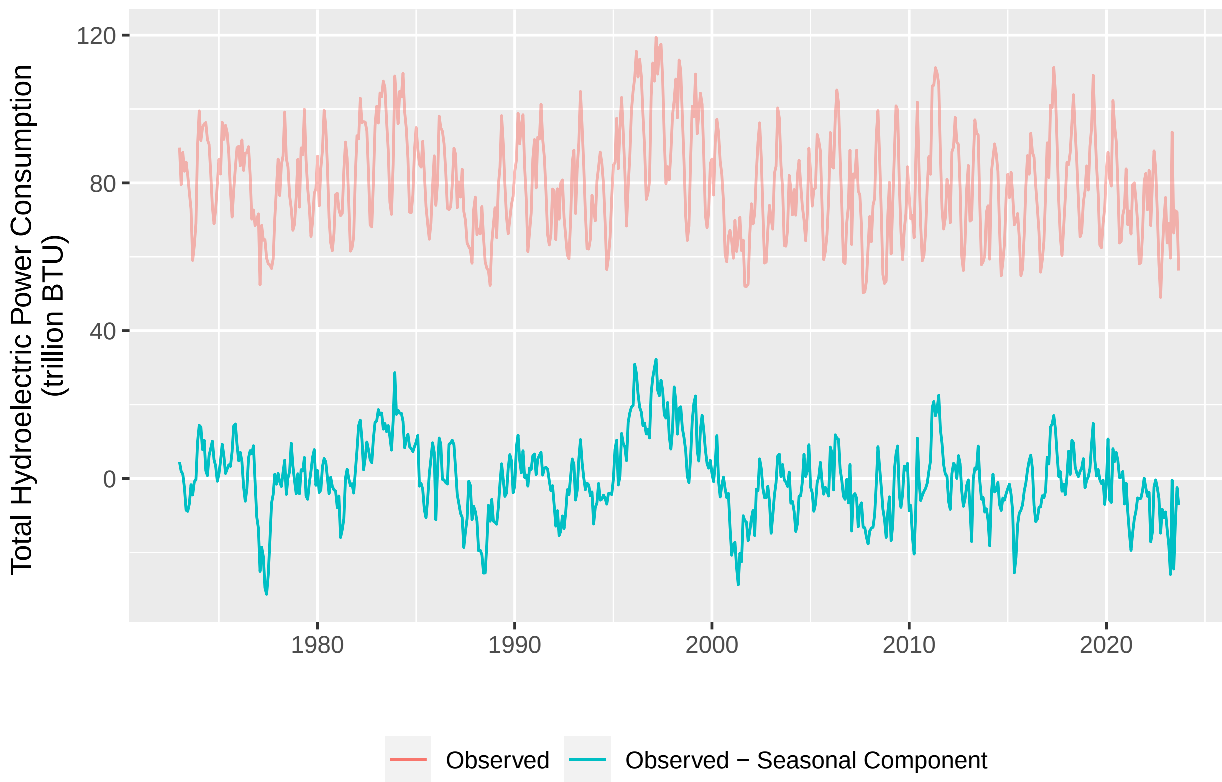
```
# get intercept and slopes and make an equation, subtract from detrended series.
beta0_season_hydro <- linear_trend_seasonal_hydro$coefficients[1]   # intercept for hydro deseason
beta0_season_renew <- linear_trend_seasonal_renew$coefficients[1]  # intercept for renew deseason
beta1_season_hydro <- linear_trend_seasonal_hydro$coefficients[2:12]  # slope for hydro deseason
beta1_season_renew <- linear_trend_seasonal_renew$coefficients[2:12] # slope for renew deseason
```

```r
#storing seasonal components
seas_component_hydro=array(0,nobs)
seas_component_renew= array(0,nobs)

for(i in 1:nobs){
seas_component_hydro[i]=(beta0_season_hydro+
                           beta1_season_hydro%*%dummies_hydro[i,])
seas_component_renew[i]=(beta0_season_renew+
                           beta1_season_renew%*%dummies_renew[i,])
}

#Transform into a ts object
ts_seasonal_component_hydro <- ts(seas_component_hydro,start = c(1973,1),
                                  frequency = 12)
ts_seasonal_component_renew <- ts(seas_component_renew,start = c(1973,1),
                                  frequency = 12)

#subtracting seasonal component from original series
y_deseason_renew <- renewable_ts-ts_seasonal_component_renew
y_deseason_hydro <- hydro_ts - ts_seasonal_component_hydro
```

```r
#Total Renewable deseason and original data
autoplot(y_deseason_renew,series="Observed - Seasonal Component") +
  autolayer(renewable_ts,series="Observed",alpha=0.5) +
  ylab("Total Renewable Energy Production \n(trillion BTU)") +
xlab("") +
labs(color="")+
theme(legend.position = "bottom")
```




```r
#Total Hydroelectric deseason and original data
autoplot(y_deseason_hydro,series="Observed - Seasonal Component") +
  autolayer(hydro_ts,series="Observed",alpha=0.5) +
  ylab("Total Hydroelectric Power Consumption \n(trillion BTU)") +
xlab("") +
labs(color="")+
theme(legend.position = "bottom")
```

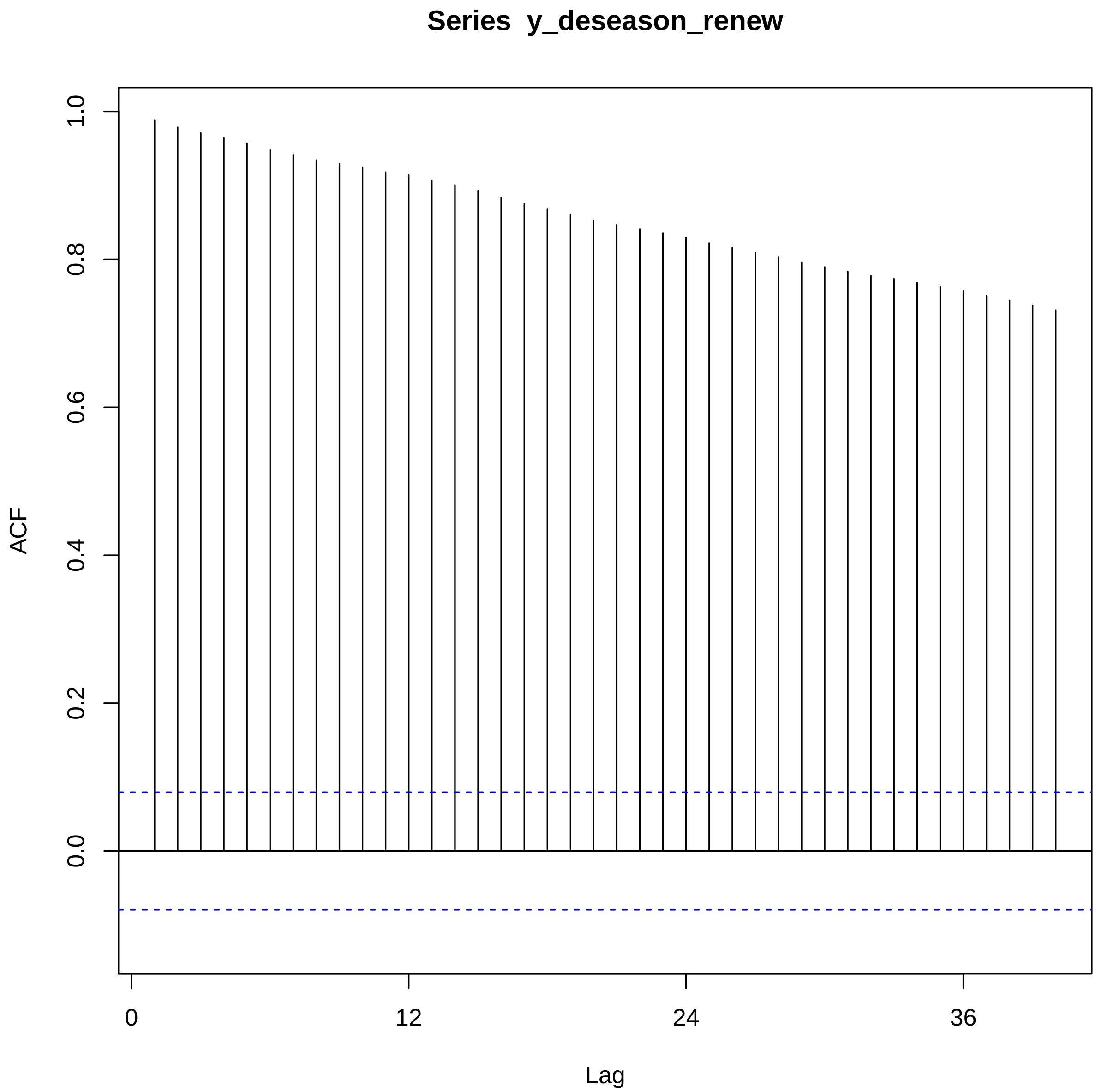


The Renewables intercept shifted down for the deseasoned data, but the graph is nearly parallel the original at any given timestep. The hydro graph intercept also shifted lower, and the magnitude between peaks and troughs has decreased in the data without the sesaonal component compared to the original data.

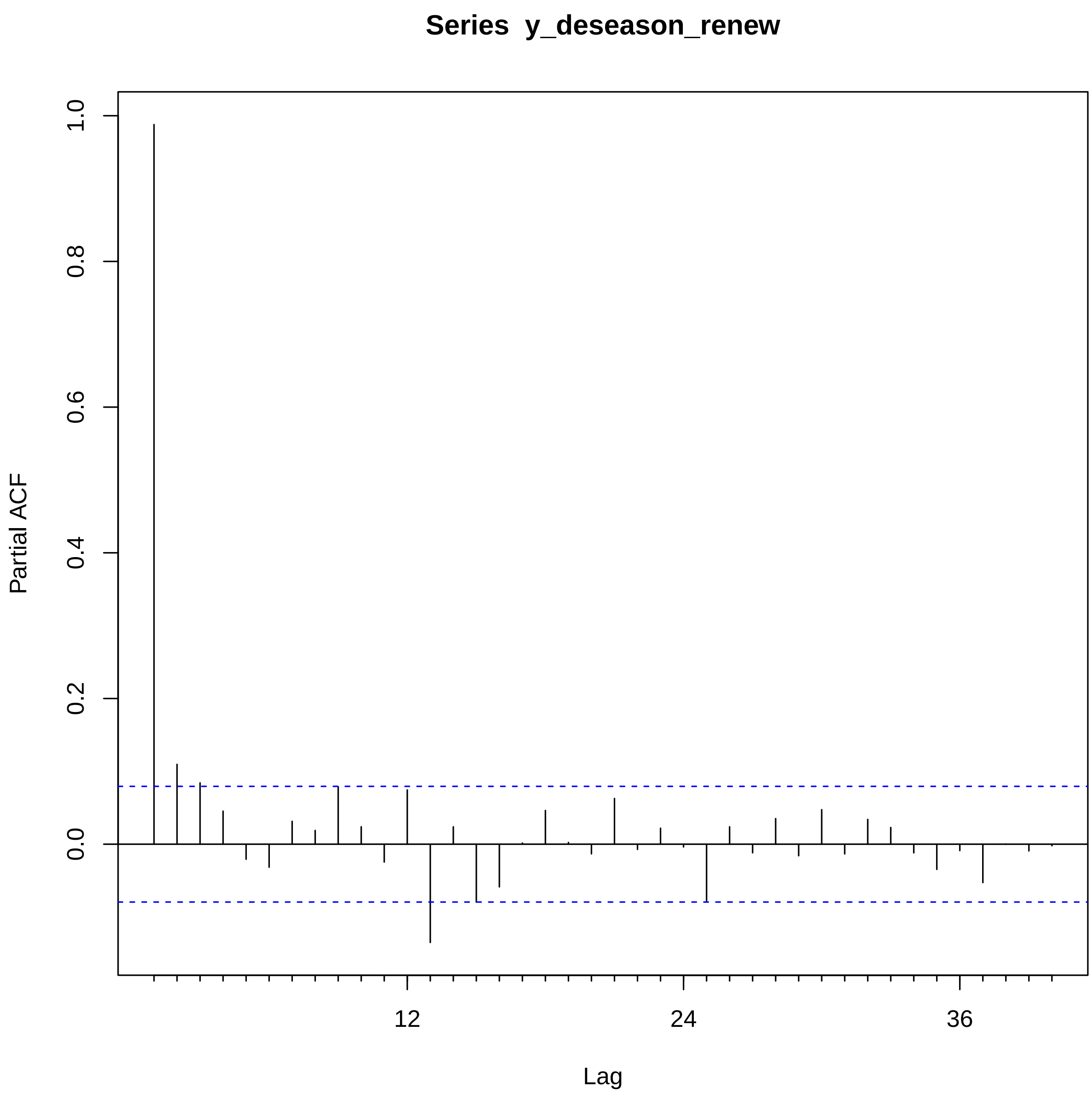
Q9

Plot ACF and PACF for the deseason series and compare with the plots from Q1. You may use plot_grid() again to get them side by side. but not mandatory. Did the plots change? How?

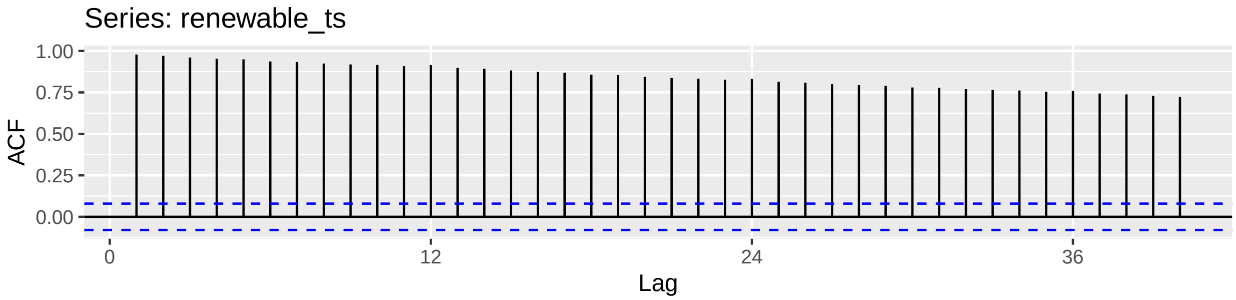
```
#renewable ACF and PACF for deseasoned data

#acf deseasoned plot for renewables
Renew_acf_deseason= autoplot(Acf(y_deseason_renew,lag.max=40))
```

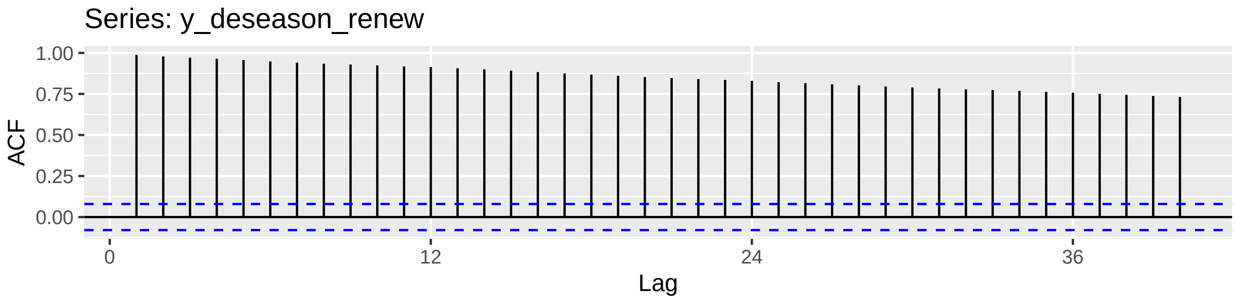


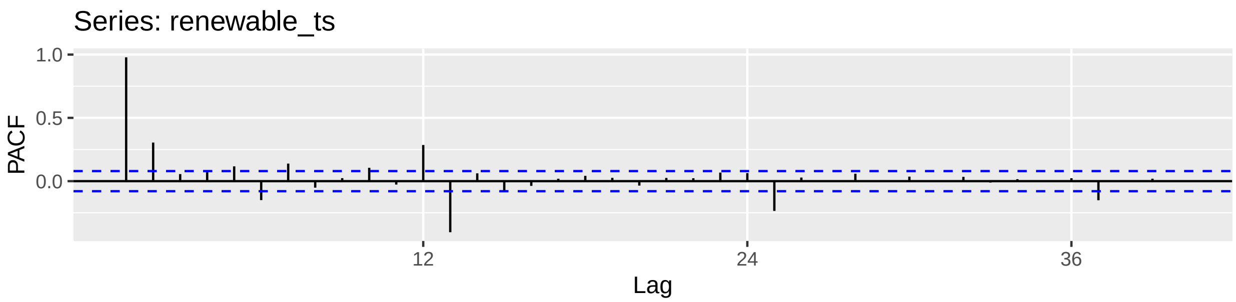

```
#pacf deseasoned plot for renewables
Renew_pacf_deseason= autoplot(Pacf(y_deseason_renew,lag.max=40))
```

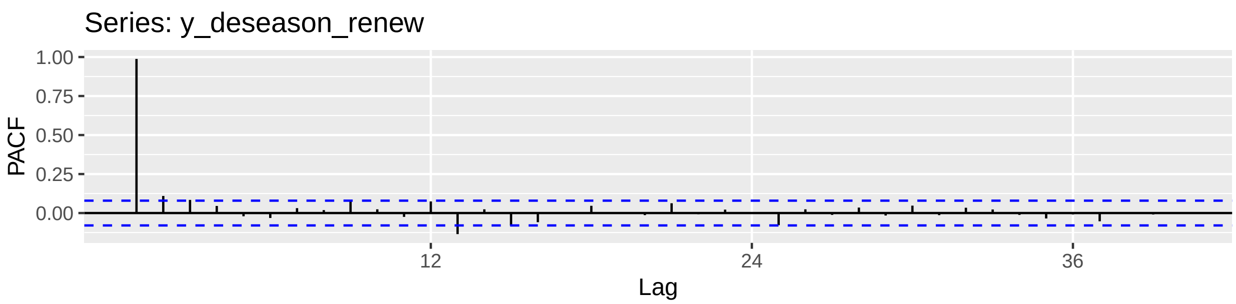



```r
#plotting comparison with plots from #1 as grid
plot_grid(Renewable_acf, Renew_acf_deseason,Renewable_pacf, Renew_pacf_deseason,
          nrow=4,align= 'h', rel_heights = c(2, 2,2,2))
```

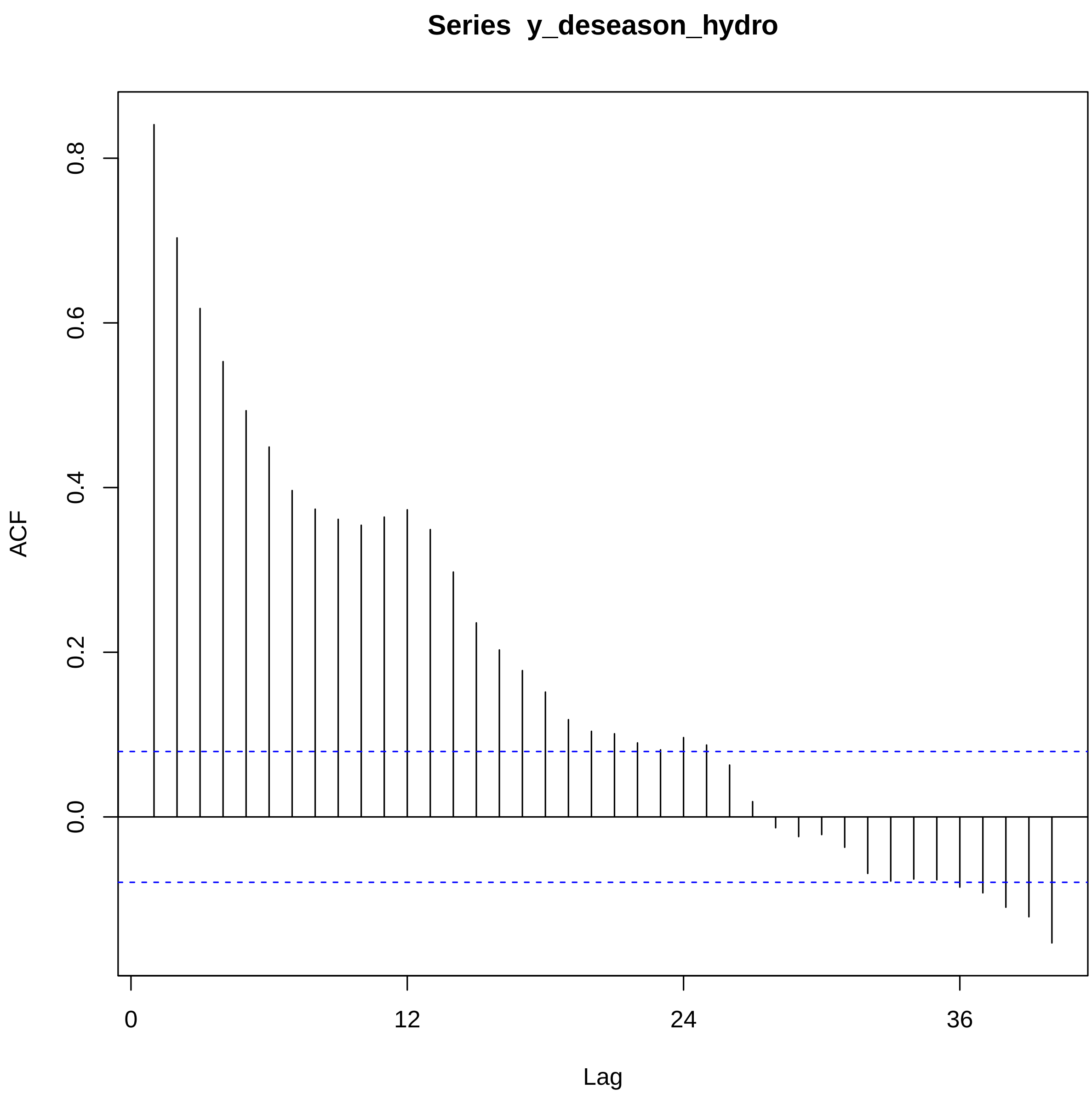




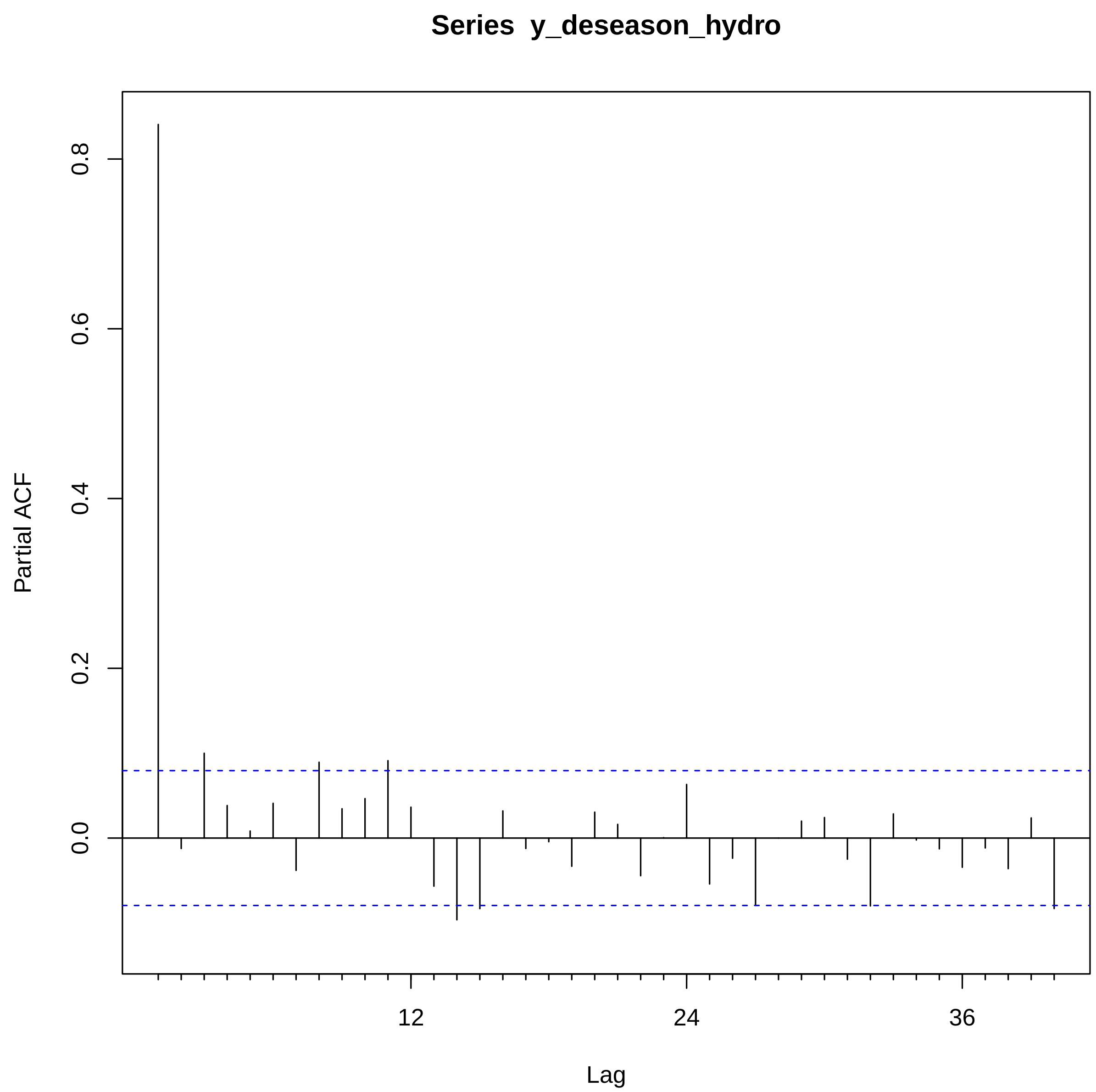




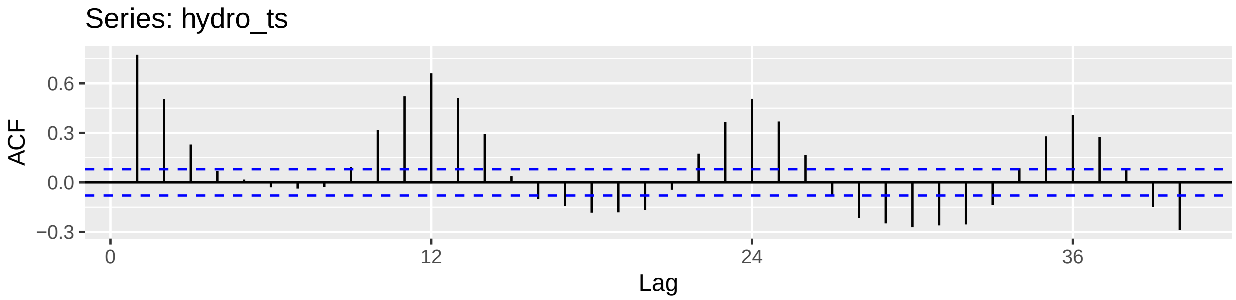

```
#hydro ACF and PACF for deseasoned data

#acf deseasoned plot for hydroelectric
hydro_acf_deseason= autoplot(Acf(y_deseason_hydro,lag.max=40))
```

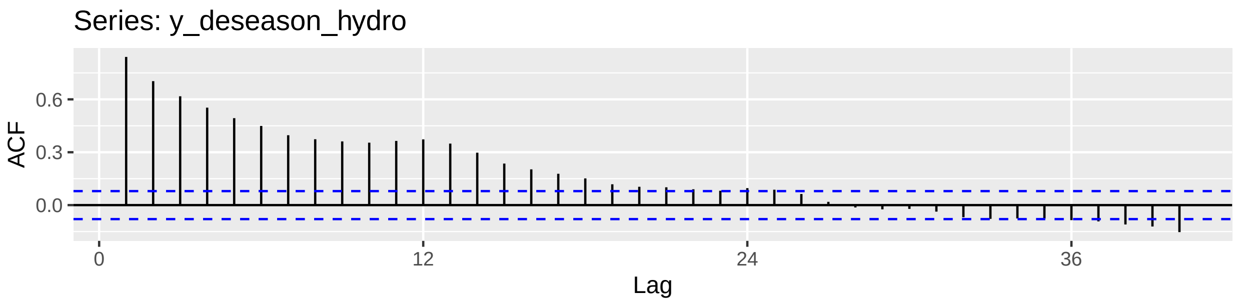


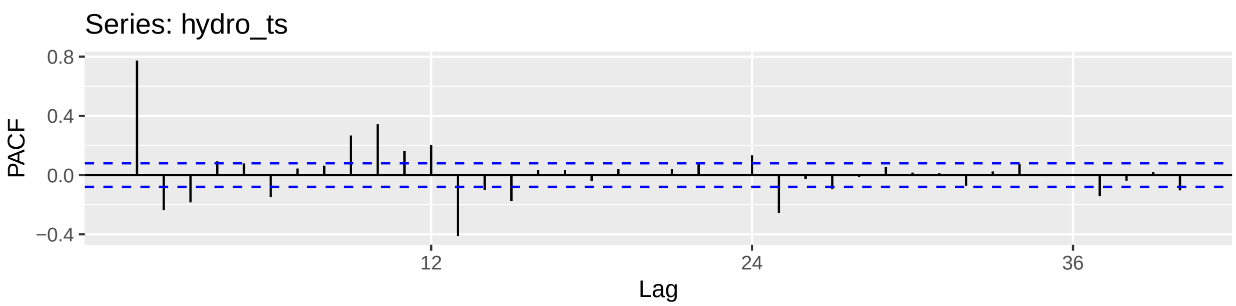

```r
#pacf deseasoned plot for hydroelectric
hydro_pacf_deseason= autoplot(Pacf(y_deseason_hydro,lag.max=40))
```

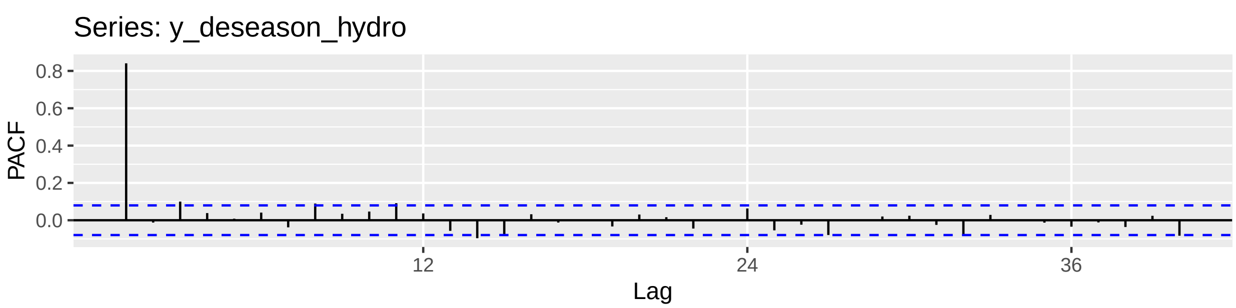



```
#plotting comparison with plots from #1 as grid
plot_grid(Hydro_acf, hydro_acf_deseason,Hydro_pacf, hydro_pacf_deseason, nrow=4,
          align= 'h', rel_heights = c(2, 2,2,2))
```









The hydro acf and pacf graphs dramatically change as a result of removing sesaonality! We no longer see the periodic pattern in the deseasoned ACF graph. The influence of later lags becomes much smaller in the ACF and PACF graphs. The renewable plots do not change much at all. This makes sense because there is very little variation caused by seasonality, as confirmed by the r2 value of .008 seen above.