ENV 790.30 - Time Series Analysis for Energy Data | Spring 2023

Assignment 7 - Due date 03/20/23

Kelsey Husted

Directions

You should open the .rmd file corresponding to this assignment on RStudio. The file is available on our class repository on Github. And to do so you will need to fork our repository and link it to your RStudio.

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_A07_Sp23.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).

When you have completed the assignment, **Knit** the text and code into a single PDF file. Submit this pdf using Sakai.

Set up

```
#Load/install required package here
library(gridExtra)
require(scales)
library(readxl)
library(openxlsx)
library(forecast)
library(tseries)
library(ggplot2)
library(Kendall)
library(lubridate)
library(tidyverse)
```

Importing and processing the data set

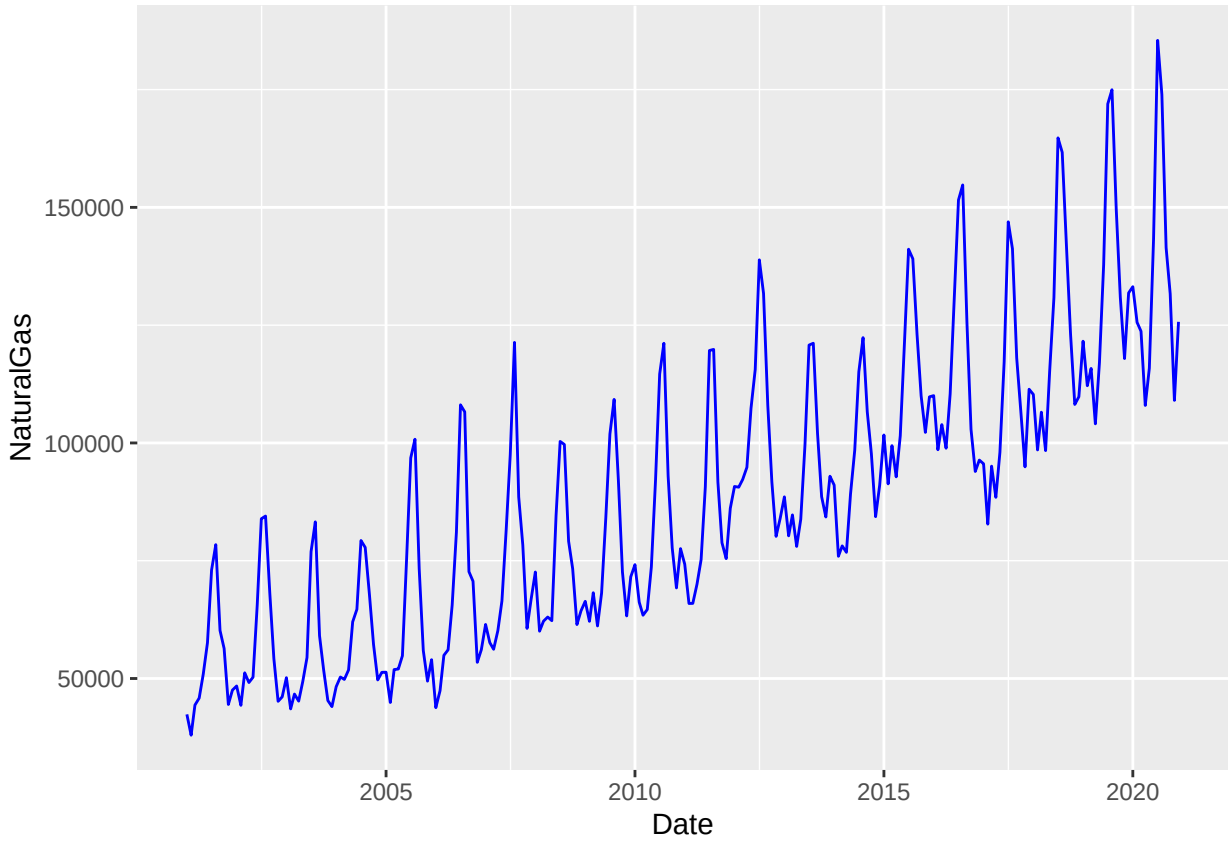
Consider the data from the file "Net_generation_United_States_all_sectors_monthly.csv". The data corresponds to the monthly net generation from January 2001 to December 2020 by source and is provided by the US Energy Information and Administration. **You will work with the natural gas column only**.

Packages needed for this assignment: "forecast","tseries". Do not forget to load them before running your script, since they are NOT default packages.\

Q1

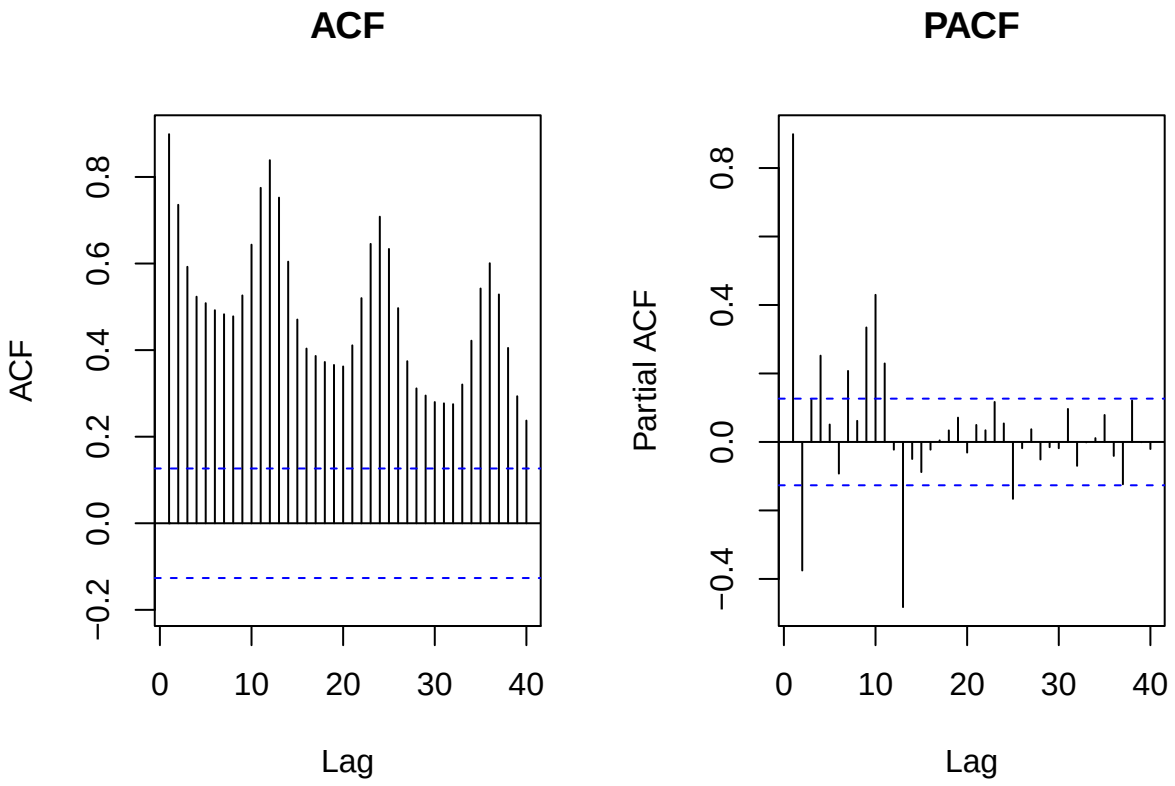
Import the csv file and create a time series object for natural gas. Make you sure you specify the **start=** and **frequency=** arguments. Plot the time series over time, ACF and PACF.

```r
#Import data into environment
energy_data <- read.csv(file = "./Data/Net_generation_United_States_all_sectors_monthly.csv",
                        header = TRUE,
                        skip = 4)


#Filter data for specific columns
energy_data_clean <-
  energy_data %>%
  mutate( Date = my(Month)) %>%
  select("Date", "natural.gas.thousand.megawatthours") %>%
  rename(NaturalGas = natural.gas.thousand.megawatthours) %>%
  arrange (Date)


#Create a time series object
ts_energy <- ts(energy_data_clean[,2], frequency = 12, start = c(2001,1))

#TS Plot
TS_Plot <-
  ggplot(energy_data_clean, aes(x=Date, y=NaturalGas)) +
      geom_line(color="blue")
plot(TS_Plot)
```




```r
#ACF and PACF plots
par(mfrow=c(1,2))
ACF_Plot <- Acf(energy_data_clean$NaturalGas, lag = 40, plot = TRUE,main="ACF")
```

```
PACF_Plot <- Pacf(energy_data_clean$NaturalGas, lag = 40, plot = TRUE,main="PACF")
```

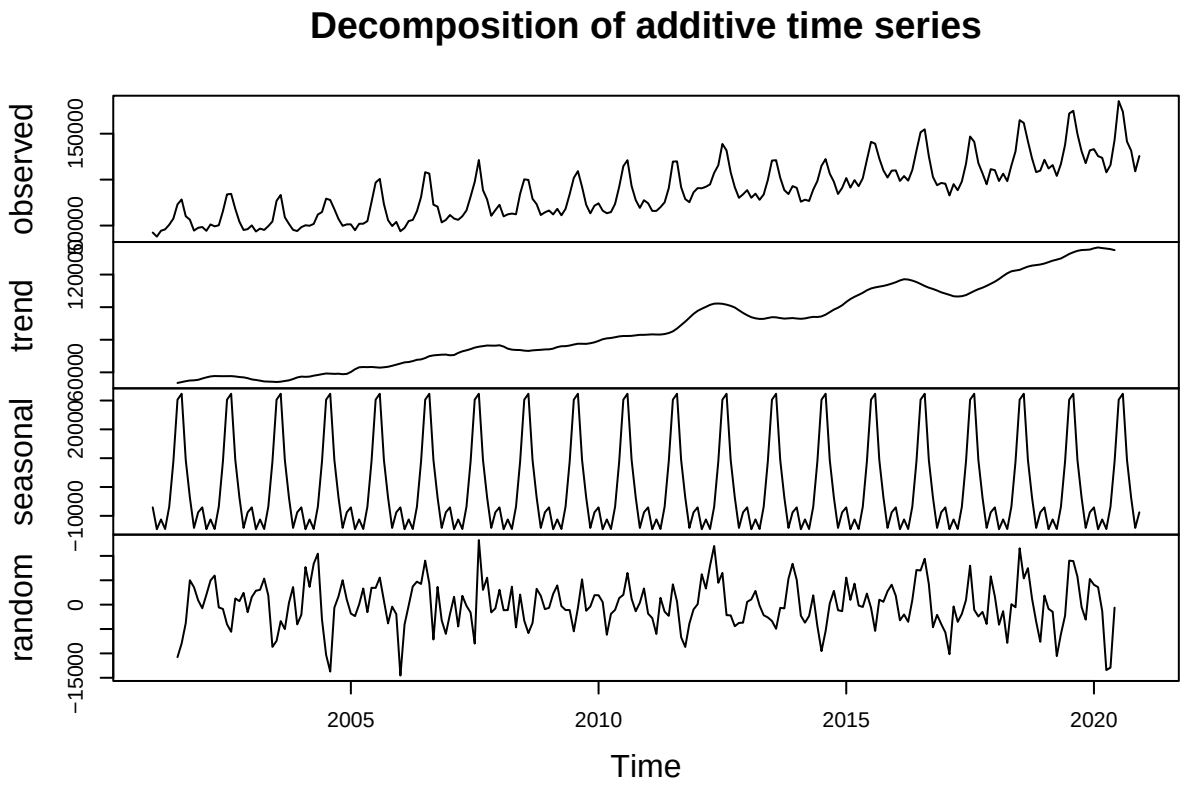


Q2

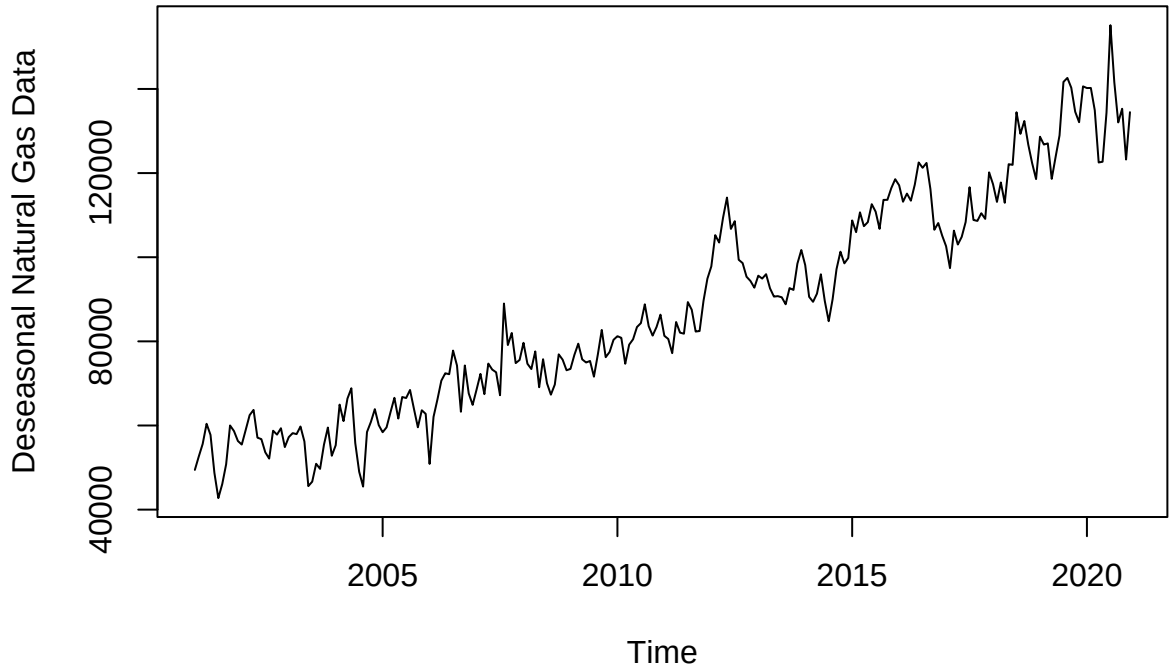
Using the *decompose*() or *stl*() and the *seasadj*() functions create a series without the seasonal component, i.e., a deseasonalized natural gas series. Plot the deseasonalized series over time and corresponding ACF and PACF. Compare with the plots obtained in Q1.

> Plot Comparison for Q1 & Q2: When looking at the Q1 plots, the scalloped top of the ACF graph indicates the presence of seasonality. Alternatively, the Q2 plots do not display this scalloped pattern (i.e., the ACF plot) which indicates that seasonality was removed.

```
#Decompose data
decompose_energy <- decompose(ts_energy, type = "additive")
plot(decompose_energy)
```

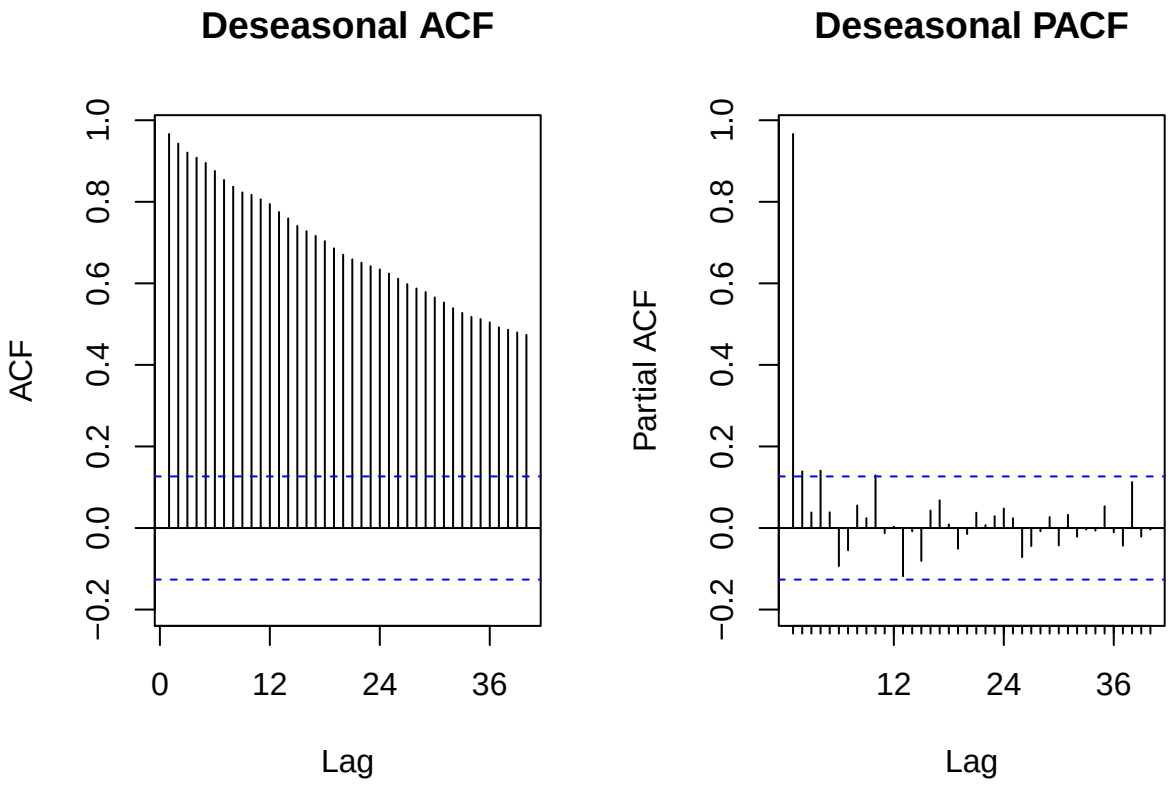



```
#Remove seasonality because some models cannot handle seasonality
deseasonal_energy <- seasadj(decompose_energy)

#Deseasonal Plot
plot(deseasonal_energy, ylab="Deseasonal Natural Gas Data")
```




```
#ACF and PACF plots
par(mfrow=c(1,2))
```

```r
deseasonal_ACF_Plot <- Acf(deseasonal_energy, lag = 40, plot = TRUE,main="Deseasonal ACF")
deseasonal_PACF_Plot <- Pacf(deseasonal_energy, lag = 40, plot = TRUE,main="Deseasonal PACF")
```



Modeling the seasonally adjusted or deseasonalized series

Q3

Run the ADF test and Mann Kendall test on the deseasonalized data from Q2. Report and explain the results.

> Answer: The ADF test had a p-value less than 0.05 which indicates a stationary trend. For the ADF test, the p-value = 0.01 so we reject the null hypothesis (i.e., stochastic trend). A Mann-Kendall Test is used to determine whether or not a trend exists in a time series data. Since the results of the Mann-Kendall Test had a p-value less than 0.05, the null hypothesis is rejected which indicates a trend present in the data. For the Mann Kendall test, p-value=<2.22e-16. Overall, both of these tests indicate the presence of a trend that is deterministic (i.e., not stochastic).

```r
#ADF test; checking for stochastic or deterministic trend
print(adf.test(deseasonal_energy, alternative = "stationary")) #p-value = 0.01
```

```
## Warning in adf.test(deseasonal_energy, alternative = "stationary"): p-value
## smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  deseasonal_energy
## Dickey-Fuller = -4.0271, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```r
#Mann Kendall Test
MKtest <- MannKendall(deseasonal_energy)
print(summary(MKtest)) # p-value=<2.22e-16
```

```
## Score =  24186 , Var(Score) = 1545533
## denominator =  28680
## tau = 0.843, 2-sided pvalue =< 2.22e-16
## NULL
```

Q4

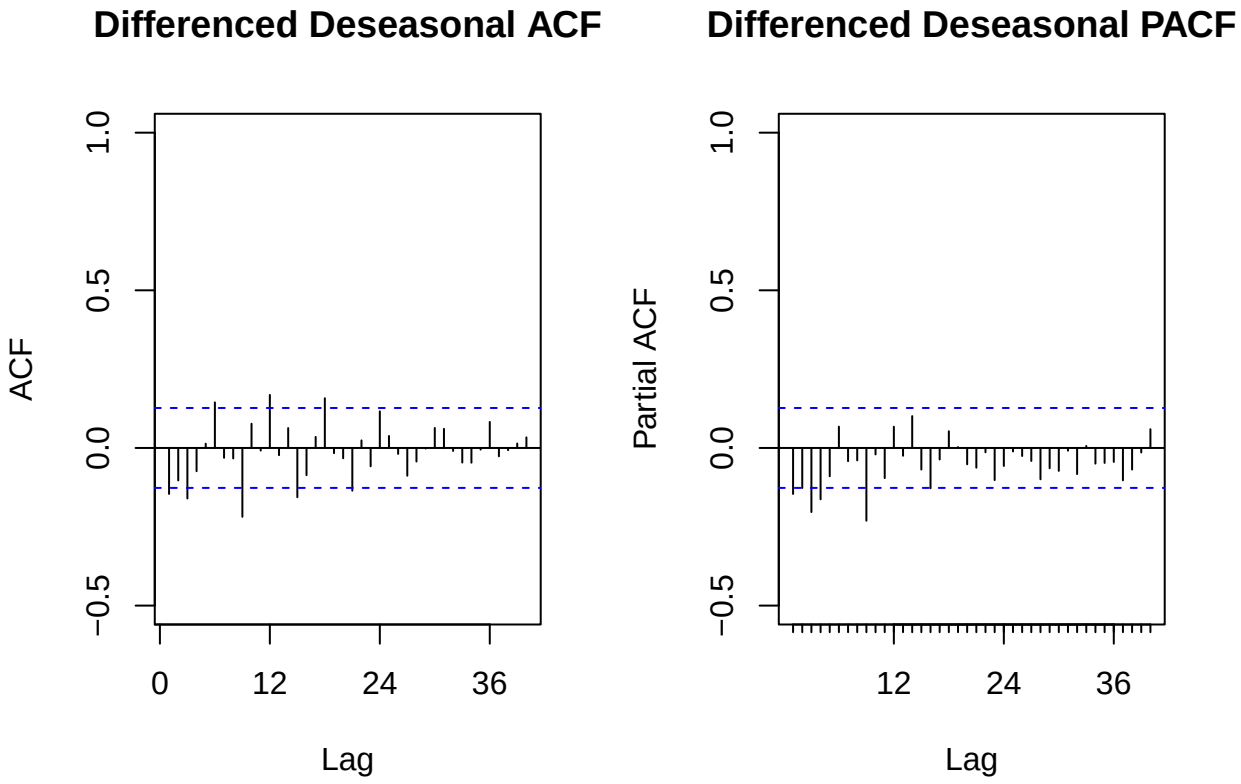
Using the plots from Q2 and test results from Q3 identify the ARIMA model parameters p, d and q. Note that in this case because you removed the seasonal component prior to identifying the model you don't need to worry about seasonal component. Clearly state your criteria and any additional function in R you might use. DO NOT use the *auto.arima*() function. You will be evaluated on ability to can read the plots and interpret the test results.

> Plot Interpretations/Reasonings: As seen in Q1, the ACF is geometric and diminishing unlike the PACF. To me, this indicates an AR model instead of a MA model (i.e., p=1 & q=0). The cutoff seen on the PACF occurs after lag 1 so maybe p=1. In the code chunk below, The function ndiffs() was used to confrm the number of differences required which was 1 (i.e., d=1). ARIMA(1,1,0)

```r
#check number of differences
deseasonal_ns_diff <- ndiffs(deseasonal_energy)
cat("Number of seasonal differencing needed: ",deseasonal_ns_diff)
```

```
## Number of seasonal differencing needed:  1
```

```r
deseason_diff <- diff(deseasonal_energy,differences=1,lag=1)

par(mfrow=c(1,2))
Acf(deseason_diff, lag.max=40, ylim=c(-.5,1), main="Differenced Deseasonal ACF")
Pacf(deseason_diff, lag.max=40, ylim=c(-.5,1), main="Differenced Deseasonal PACF")
```



Q5

Use *Arima*() from package "forecast" to fit an ARIMA model to your series considering the order estimated in Q4. You should allow constants in the model, i.e., $include.mean = TRUE$ or $include.drift = TRUE$. **Print the coefficients** in your report. Hint: use the *cat*() function to print.

```
#ARIMA Model
Model_110 <- Arima(deseasonal_energy,order=c(1,1,0),include.mean = TRUE, include.drift=TRUE)
print(Model_110)
```

```
## Series: deseasonal_energy
## ARIMA(1,1,0) with drift
##
## Coefficients:
##           ar1     drift
##       -0.1479  348.3927
## s.e.   0.0644  308.8385
##
## sigma^2 = 30254066:  log likelihood = -2396.54
## AIC=4799.07   AICc=4799.18   BIC=4809.5
```

```
#print coefficients
cat("Model Coefficients: ", Model_110$coef)
```

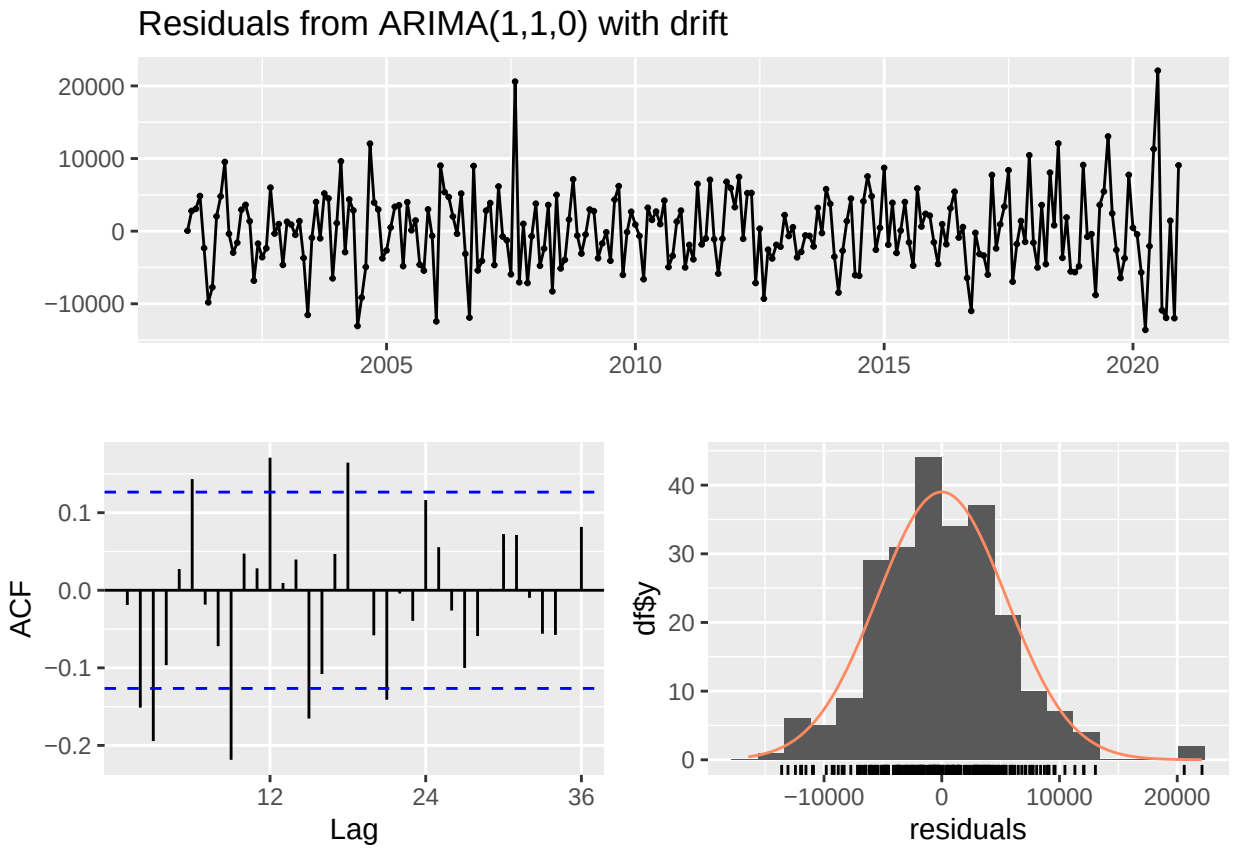
```
## Model Coefficients:  -0.1478609 348.3927
```

Q6

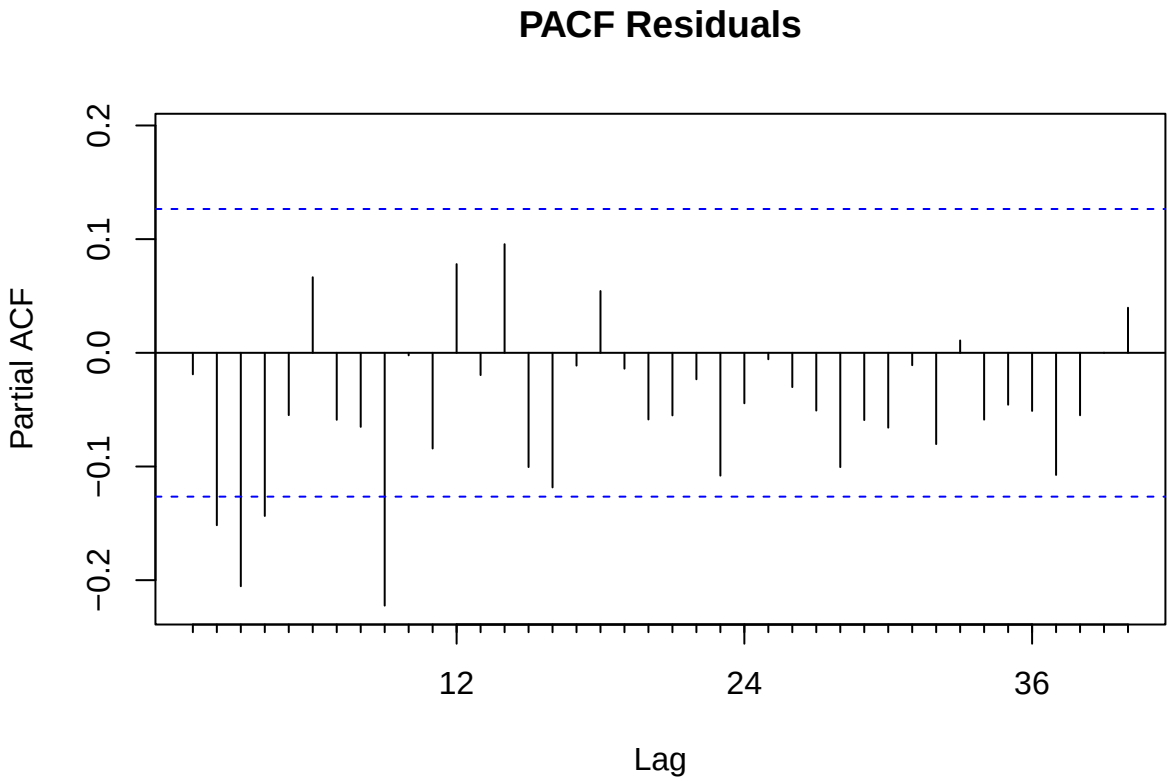
Now plot the residuals of the ARIMA fit from Q5 along with residuals ACF and PACF on the same window. You may use the *checkresiduals*() function to automatically generate the three plots. Do the residual series look like a white noise series? Why?

Answer: The residual series does look like white noise. Residuals that display a white noise pattern are good and indicates that th model is a good fit.

```
checkresiduals(Model_110, test=FALSE, plot=TRUE)
```




```
Pacf(Model_110$residuals,lag.max=40, main="PACF Residuals")
```



Modeling the original series (with seasonality)

Q7

Repeat Q4-Q6 for the original series (the complete series that has the seasonal component). Note that when you model the seasonal series, you need to specify the seasonal part of the ARIMA model as well, i.e., P, D and Q.

> Answer: Using the function nsdiff(), I can tell that 1 seasonal differencing is needed (i.e., D=1). For P and Q, I am looking at the lags at 12, 24, 36, etc. from the ACF/PACF plots from Q1. Both ACF/PACF plots look like they could be geometric so maybe P=1 and Q=1. The differenced ACF and PACF plots display a decreasing geometric pattern so maybe p=1 and q=1. Ndiff() showed that differencing was needed (d=1). ARIMA(1,1,1)(1,1,1)

```
#ADF test; checking for stochastic or deterministic trend
print(adf.test(ts_energy, alternative = "stationary")) #p-value = 0.01
```

```
## Warning in adf.test(ts_energy, alternative = "stationary"): p-value smaller than
## printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_energy
## Dickey-Fuller = -8.9602, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```
#Mann Kendall Test
SMKtest <- SeasonalMannKendall(ts_energy)
print(summary(MKtest)) # p-value=<2.22e-16
```

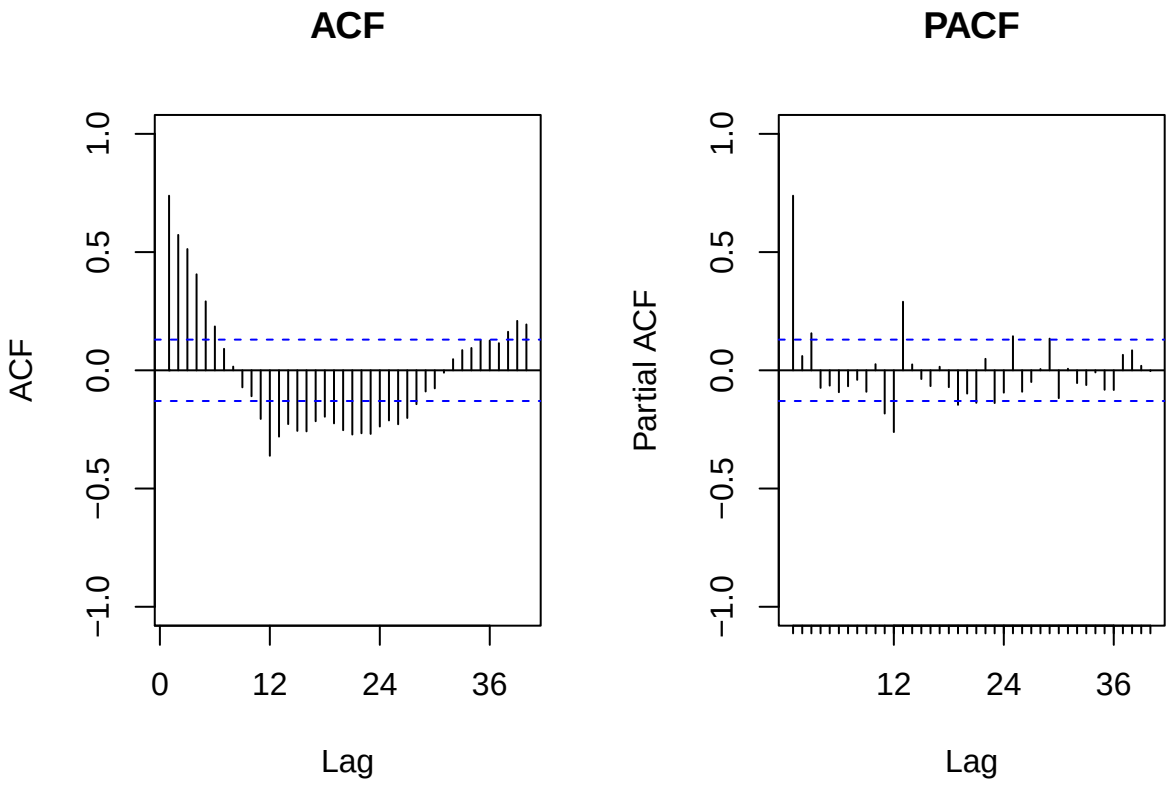
```
## Score =  24186 , Var(Score) = 1545533
```

```
## denominator =  28680
## tau = 0.843, 2-sided pvalue =< 2.22e-16
## NULL
```

```
#check number of differences
ns_diff <- nsdiffs(ts_energy)
cat("Number of seasonal differencing needed: ",ns_diff) #D = 1
```

```
## Number of seasonal differencing needed:  1
```

```
#difference the seasonal series
energy_diff <- diff(ts_energy,lag =12, differences=1) #diff done on orig series

#check the differenced plots
par(mfrow=c(1,2))
Acf(energy_diff,lag.max=40,main="ACF",ylim=c(-1,1))
Pacf(energy_diff,lag.max=40,main="PACF",ylim=c(-1,1))
```




```
SARIMA_Model <- Arima(deseasonal_energy,order=c(1,1,1),seasonal =c(1,1,1),include.mean = TRUE, include.
```

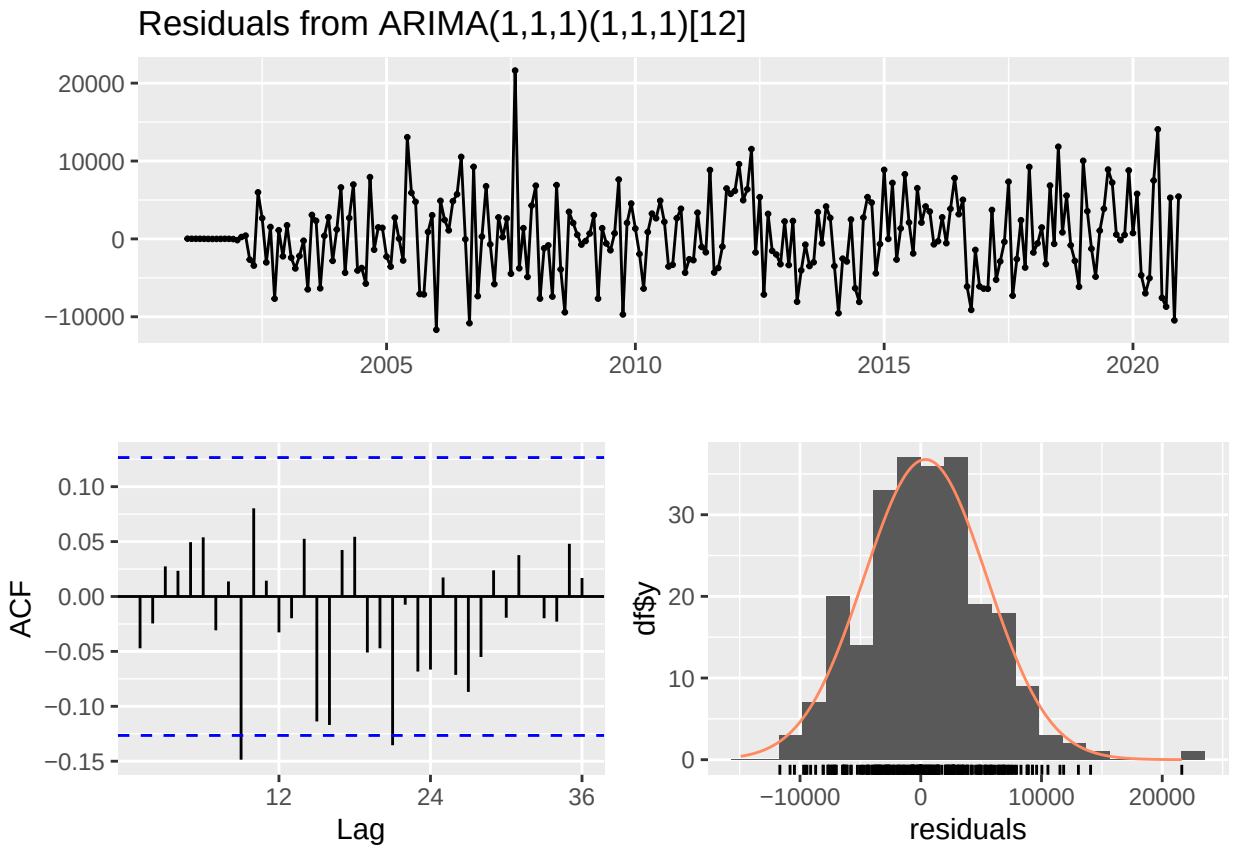
```
## Warning in Arima(deseasonal_energy, order = c(1, 1, 1), seasonal = c(1, : No
## drift term fitted as the order of difference is 2 or more.
```

```
print(SARIMA_Model)
```

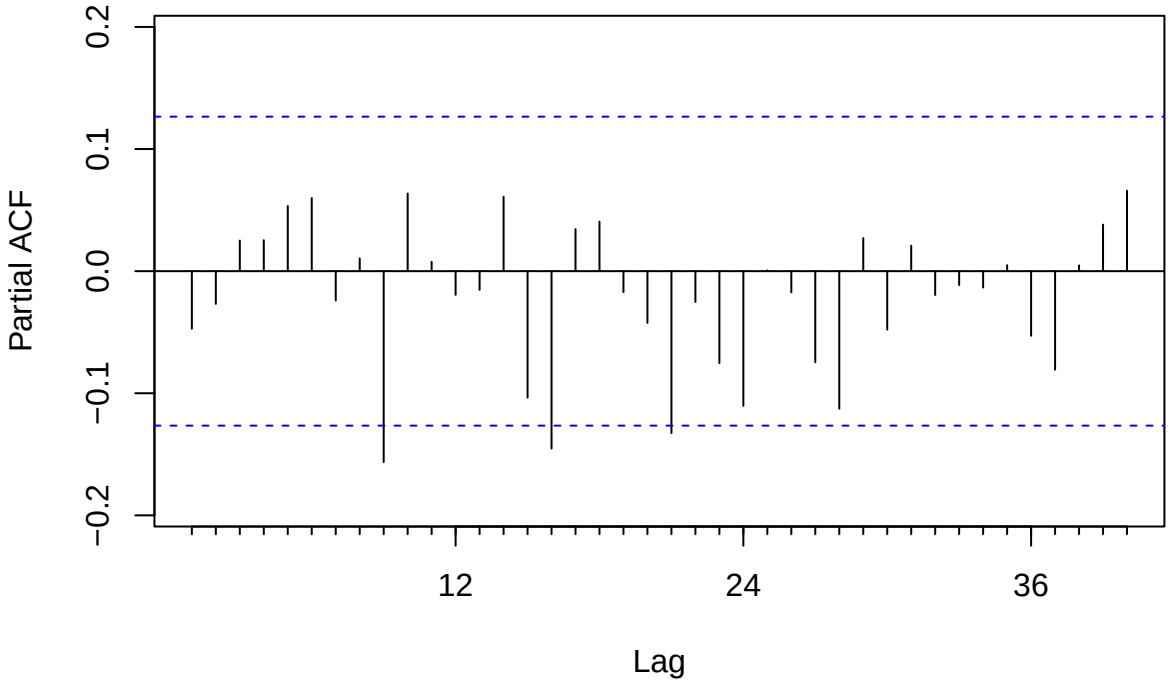
```
## Series: deseasonal_energy
## ARIMA(1,1,1)(1,1,1)[12]
##
## Coefficients:
##          ar1      ma1     sar1     sma1
##       0.7328  -0.9819  -0.0202  -0.6909
## s.e.  0.0504   0.0182   0.0949   0.0772
```

```
##
## sigma^2 = 28044701:  log likelihood = -2272.17
## AIC=4554.35   AICc=4554.62   BIC=4571.47
```

```
checkresiduals(SARIMA_Model, test=FALSE, plot=TRUE)
```




```
Pacf(SARIMA_Model$residuals,lag.max=40, main="PACF Residuals")
```



Q8

Compare the residual series for Q7 and Q6. Can you tell which ARIMA model is better representing the Natural Gas Series? Is that a fair comparison? Explain your response.

> Answer: I think the SARIMA model from Q7 better represents the Natural Gas Series since the residual white noise model displays less significant lags in the ACF plot in comparison as seen above. Even so, both residual series display white noise which is ideal.

Checking your model with the auto.arima()

Please do not change your answers for Q4 and Q7 after you ran the *auto.arima*(). It is **ok** if you didn't get all orders correctly. You will not loose points for not having the same order as the *auto.arima*().

Q9

Use the *auto.arima*() command on the **deseasonalized series** to let R choose the model parameter for you. What's the order of the best ARIMA model? Does it match what you specified in Q4?

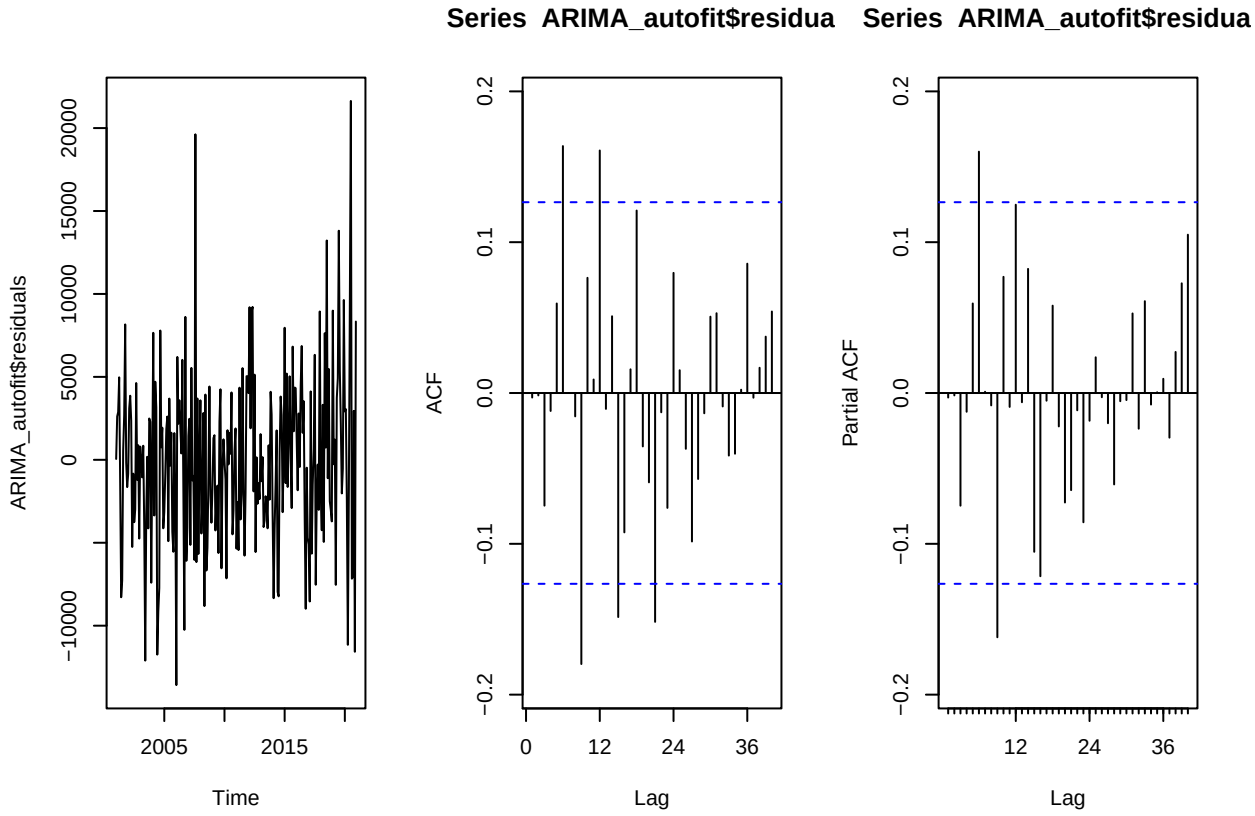
> Answer: No my model does not match. I initially thought that there was only an AR model present and no MA value present.

```
ARIMA_autofit <- auto.arima(deseasonal_energy)
print(ARIMA_autofit)
```

```
## Series: deseasonal_energy
## ARIMA(1,1,1) with drift
##
## Coefficients:
##          ar1      ma1     drift
##       0.7065  -0.9795  359.5052
```

```
## s.e.  0.0633    0.0326    29.5277
##
## sigma^2 = 26980609:  log likelihood = -2383.11
## AIC=4774.21   AICc=4774.38   BIC=4788.12
```

```
par(mfrow=c(1,3))
ts.plot(ARIMA_autofit$residuals)
Acf(ARIMA_autofit$residuals,lag.max=40)
Pacf(ARIMA_autofit$residuals,lag.max=40)
```



Q10

Use the *auto.arima*() command on the **original series** to let R choose the model parameters for you. Does it match what you specified in Q7?

Answer: This plot does not match the SARIMA model that I specified in Q7.

```
SARIMA_autofit <- auto.arima(ts_energy)
print(SARIMA_autofit)
```

```
## Series: ts_energy
## ARIMA(1,0,0)(0,1,1)[12] with drift
##
## Coefficients:
##          ar1     sma1     drift
##       0.7416  -0.7026  358.7988
## s.e.  0.0442   0.0557   37.5875
##
## sigma^2 = 27569124:  log likelihood = -2279.54
## AIC=4567.08   AICc=4567.26   BIC=4580.8
```

```
par(mfrow=c(1,3))
ts.plot(SARIMA_autofit$residuals)
Acf(SARIMA_autofit$residuals,lag.max=40)
Pacf(SARIMA_autofit$residuals,lag.max=40)
```