ENV 790.30 - Time Series Analysis for Energy Data | Spring 2022

Assignment 6 - Due date 03/25/22

Colin Lee

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 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).

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_A07_Sp22.Rmd"). Submit this pdf using Sakai.

Set up

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.1.2
```

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

```
library(tseries)
library(sarima)
```

```
## Warning: package 'sarima' was built under R version 4.1.2
```

```
## Loading required package: stats4
```

```
##
## Attaching package: 'sarima'
```

```
## The following object is masked from 'package:stats':
##
##     spectrum
```

```
library(xlsx)
library(ggplot2)
library(Kendall)
library(lubridate)
```

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

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

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
```

```
library(astsa)
```

```
##
## Attaching package: 'astsa'

## The following object is masked from 'package:sarima':
##
##     sarima

## The following object is masked from 'package:forecast':
##
##     gas
```

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**.

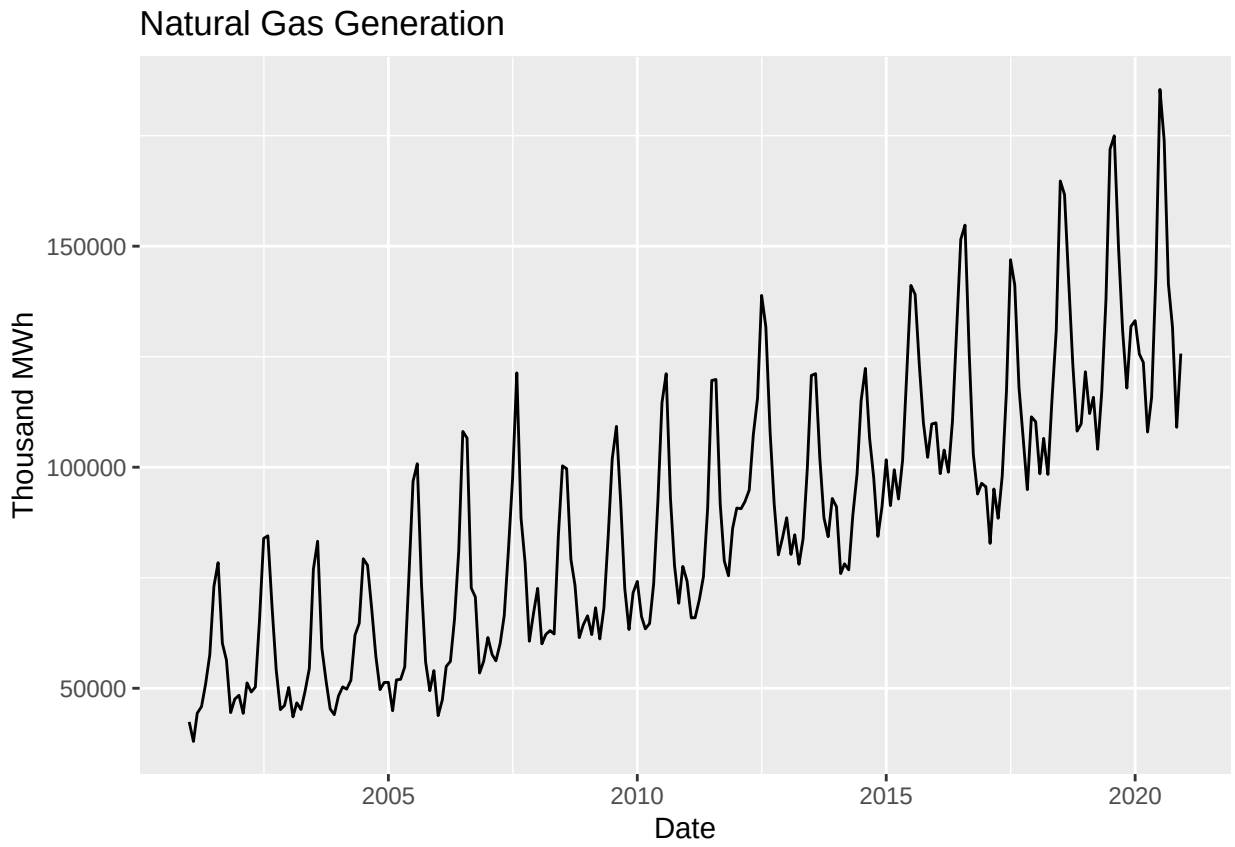
Packages needed for this assignment: "forecast","tseries". Do not forget to load them before running your script, since they are NOT default packages.\

Q1

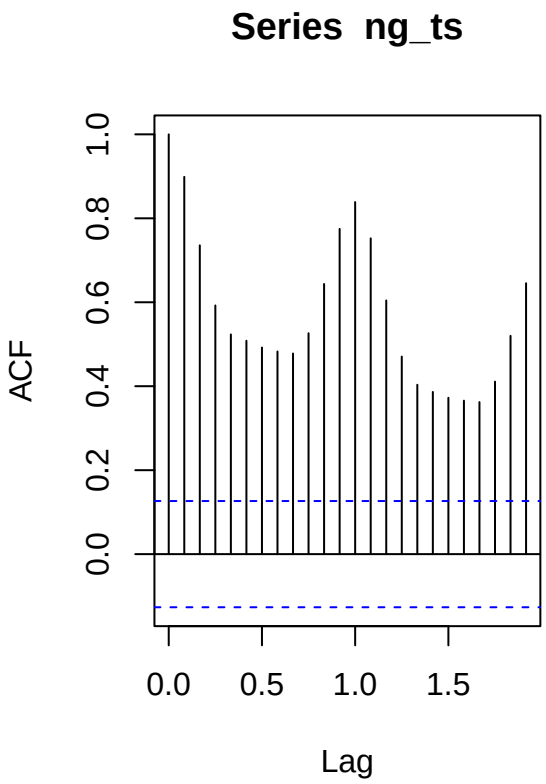
Import the csv file and create a time series object for natural gas. Make you sure you specify the **start=** and **frequency=** arguments. Plot the time series over time, ACF and PACF.

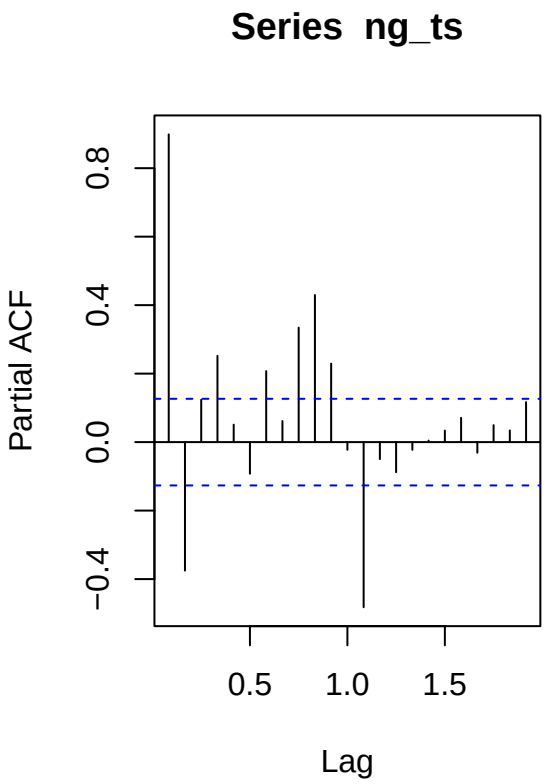
I turned the data into .xlsx file so that I could then reorder the data so it starts with 2001 instead of 2020 and descending.

```
energy_data <- read.xlsx(file="/Users/colinlee/Documents/Duke/Spring 2022/ENV790/ENV790_TimeSeriesAnaly

ng_data <- as.numeric(energy_data[,4])

ng_ts <- ts(ng_data,start = c(2001,1), frequency = 12)

ggplot(energy_data) +
  geom_line(aes(x = as.Date(energy_data[,1],origin = "2001-01-00"), y = (as.numeric(energy_data[,4])))) +
  labs(y = "Thousand MWh",
            x = "Date",
            title = "Natural Gas Generation",color="") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```




```
par(mfrow=c(1,2))
acf(ng_ts)
pacf(ng_ts)
```



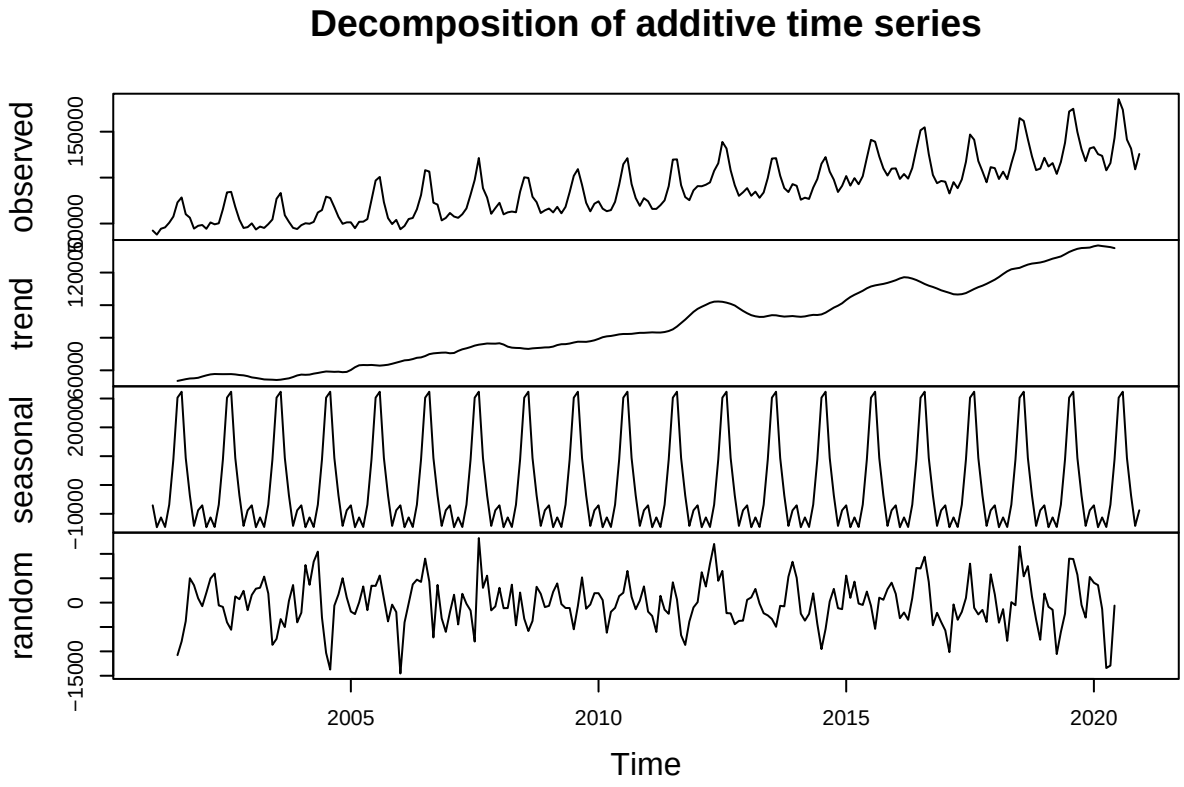


Q2

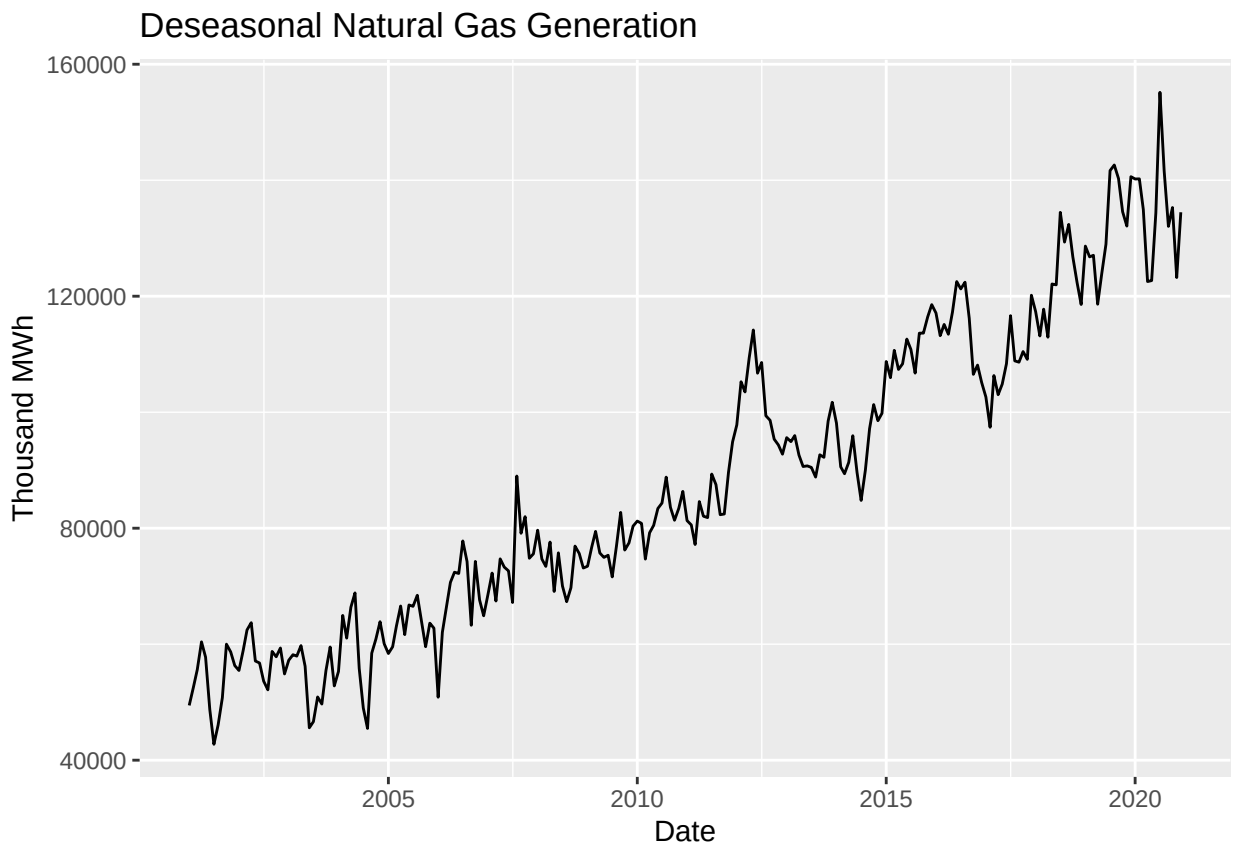
Using the *decompose*() or *stl*() and the *seasadj*() functions create a series without the seasonal component, i.e., a deseasonalized natural gas series. Plot the deseasonalized series over time and corresponding ACF and PACF. Compare with the plots obtained in Q1.

In the plot, we can see the seasonal trend removed with the long peaks and troughs that are indicative of seasonality removed in the graph. Furthermore, in the deseasonal ACF, we can see the wave pattern disappear and the ACF trend is just monotonically decreasing. The PACF is a little different in that the deseasonal PACF values are lower but no large difference in pattern.

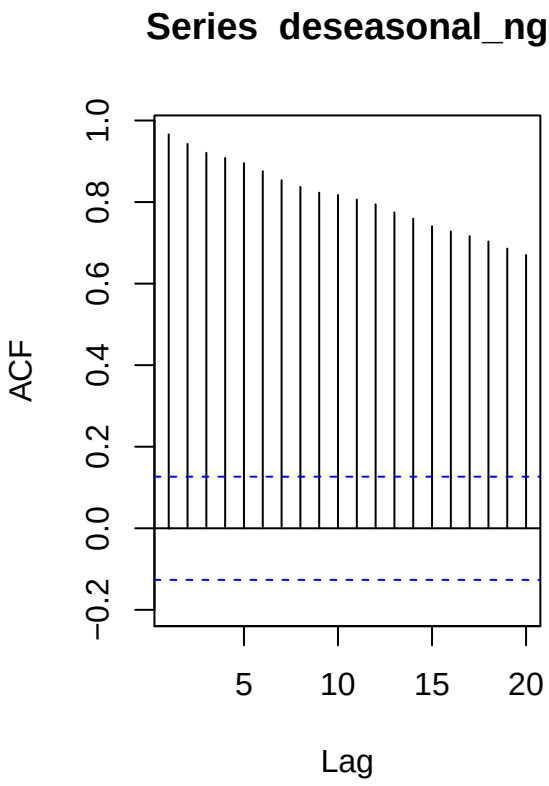
```
decomposed_ng <- decompose(ng_ts)
plot(decomposed_ng)
```

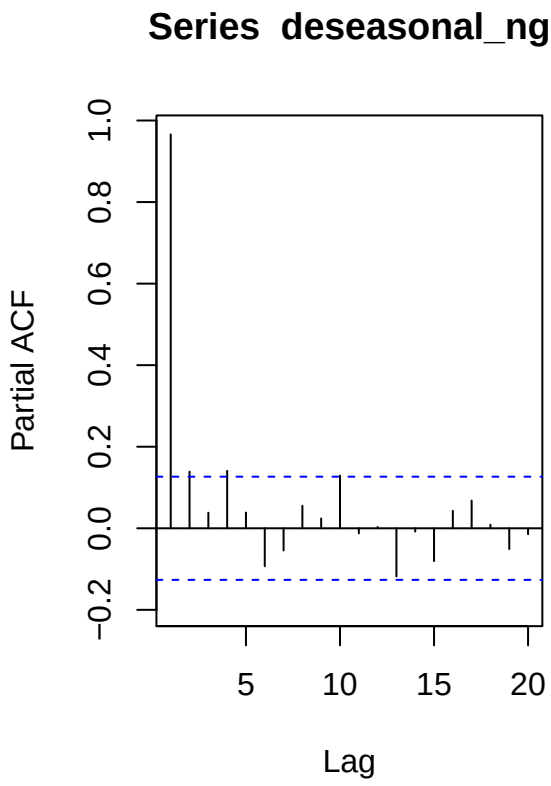



```
deseasonal_ng <- seasadj(decomposed_ng)
ggplot() +
  geom_line(aes(x = as.Date(energy_data[,1],origin = "2001-01-00"), y = (as.numeric(deseasonal_ng)))) +
  labs(y = "Thousand MWh",
            x = "Date",
            title = "Deseasonal Natural Gas Generation",color="") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")
```




```
par(mfrow=c(1,2))
Acf(deseasonal_ng, lag = 20)
Pacf(deseasonal_ng, lag = 20)
```





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.

According to the ADF test, we have a p-value of 0.01, indicating that there the series is stationary.

The p-value for the MK test is very low with p-value of 2.22e-16, indicating that there is a trend, which is very clear since the data is strongly monotonically increasing (upwards trend).

```
ng_ADF <- adf.test(deseasonal_ng,alternative="stationary")
```

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

```
print((ng_ADF))
```

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

```
ng_MK <- MannKendall(deseasonal_ng)
print(ng_MK)
```

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

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.

Based on the ADF, we do not have a unit root, but the MK test shows that we do have a deterministic trend. Thus, because a deterministic trend is present, q = 1.

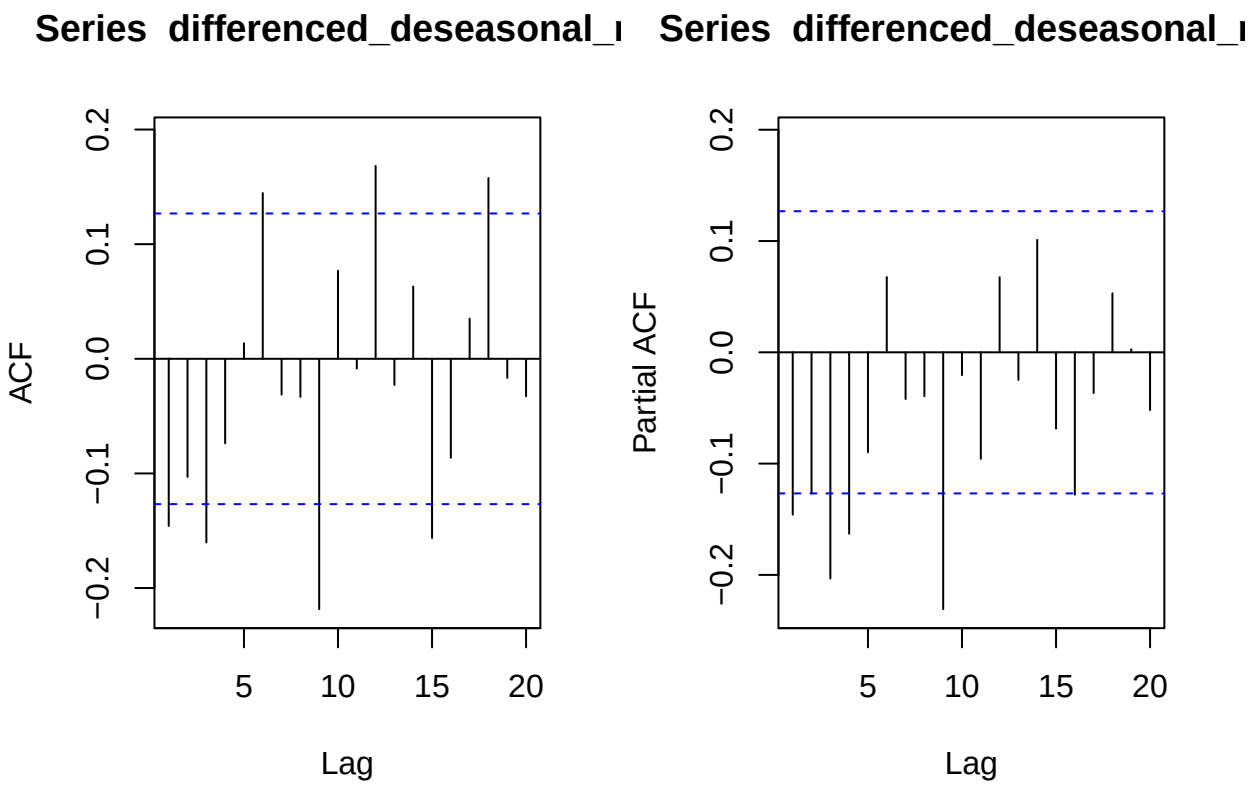
Based on the plots for Q2, we can see that there is a cut-off at lag 1 in the PACF plot, indicating p = 1. However, in the ACF plot, there is a slow decay, and there is no clear cut-off, making it very hard to identify the q value. Because there is no cut-off, I will have to assume q = 0.

Thus, I initially predict the ARIMA model is ARIMA(1,1,0).

However, if I implement the differencing and then do another ACF and PACF, I find that the plots are different and it is clearer to find the p and q values. The ACF plot no longer has a slow decay, but rather, there is a small cut-off at lag 1. For the PACF plot, there still is no slow decay, and we can see another small cut-off at lag 1 as well.

Thus, after differencing, I can now make a more educated guess that the ARIMA model here is ARIMA(1,1,1)

```
differenced_deseasonal_ng <- diff(deseasonal_ng, differences = 1)
par(mfrow=c(1,2))
Acf(differenced_deseasonal_ng, lag = 20)
Pacf(differenced_deseasonal_ng, lag = 20)
```



Q5

Use *Arima*() from package "forecast" to fit an ARIMA model to your series considering the order estimated in Q4. Should you allow for 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.

I should not use include.mean = TRUE since d = 1, indicating the series is differenced in the ARIMA model.

We may use include.drift = TRUE since the series is not differenced twice, and it allows for a small deviation in the mean from zero.

```
q5_arima <- Arima(deseasonal_ng, order = c(1,1,1), include.drift = TRUE)
q5_arima
```

```
## Series: deseasonal_ng
## 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
```

```
cat("coefficients", q5_arima$coef)
```

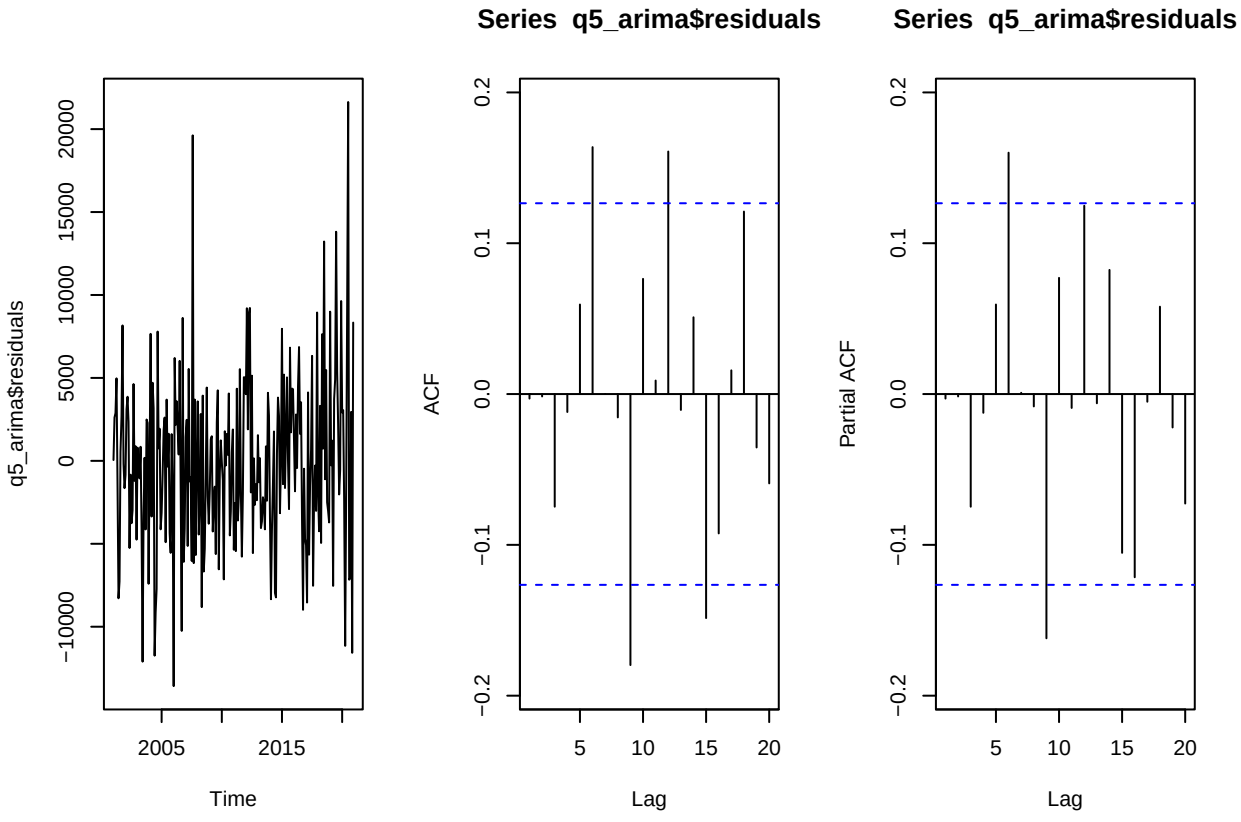
```
## coefficients 0.7065237 -0.9794655 359.5052
```

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?

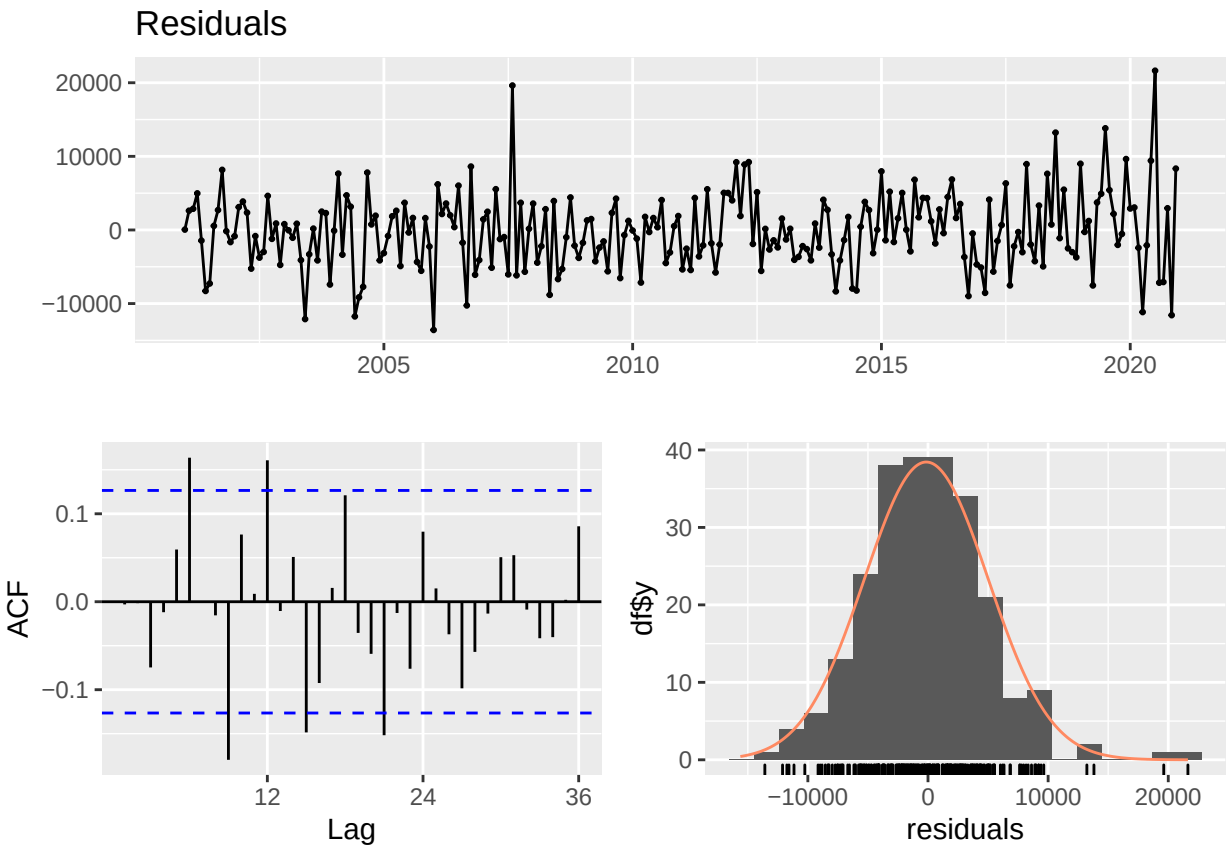
The residual series look like a white noise series because there is no clear cutoff for the ACF or PACF plots, indicating that the model does not follow any ARIMA or AR or MA model.

```
par(mfrow=c(1,3))
ts.plot(q5_arima$residuals)
Acf(q5_arima$residuals, lag = 20)
Pacf(q5_arima$residuals, lag = 20)
```




```
#just trying the checkresiduals() here
checkresiduals(q5_arima$residuals)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```



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.

I ran the ADF and SMK tests to assess whether there is a need for differencing. The results are the same as the previous ADF and SMK tests, where the ADF test states the series is stationary, and the SMK (seasonal Mann Kendall) states the series has a deterministic trend.

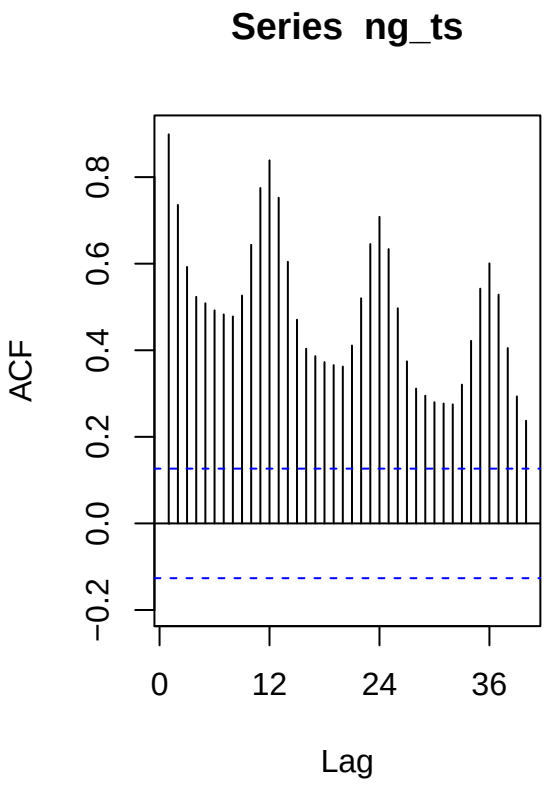
Thus, d = 1.

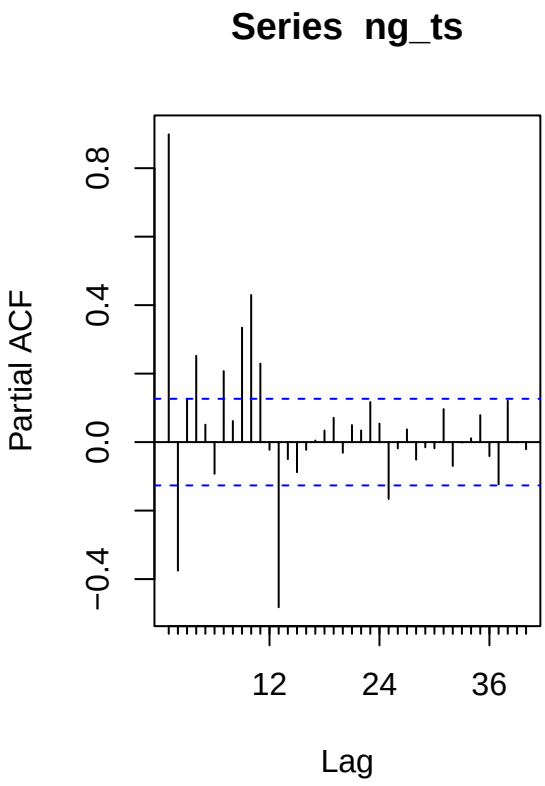
For the ACF and PACF plots, the ACF has no cut-off and is decaying while the PACF has a cut-off at lag 1. Thus, this is an AR model with p = 1. However, we need to difference since there is a trend. After differencing, the ACF and PACF reveal cutoffs at lag 1, indicating an ARIMA model with p, d, and q equaling 1. Thus, for the non-seasonal component, I predict the model to be ARIMA(1,1,1)

For the seasonal component, we see multiple spikes in intervals of 12 in the ACF plot and a single positive spike in the PACF plot at lag 12, indicating a SAR process with P = 1. Finally, I ran the nsdiffs function to find the number of seasonal differencing required to achieve stationarity, which is 1. This indicates D = 1.

Thus, I predict the SARIMA model to be ARIMA(1,1,1) x (1,1,0)12

```
par(mfrow=c(1,2))
Acf(ng_ts, lag = 40)
Pacf(ng_ts, lag = 40)
```






```
ng_ADF_seasonal <-adf.test(ng_ts,alternative="stationary")
```

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

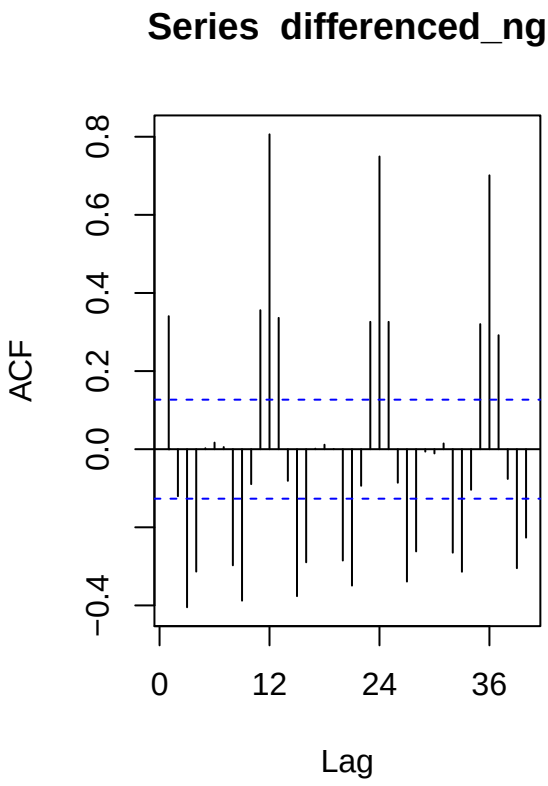
```
ng_SMK <- SeasonalMannKendall(ng_ts)

print(ng_ADF_seasonal)
```

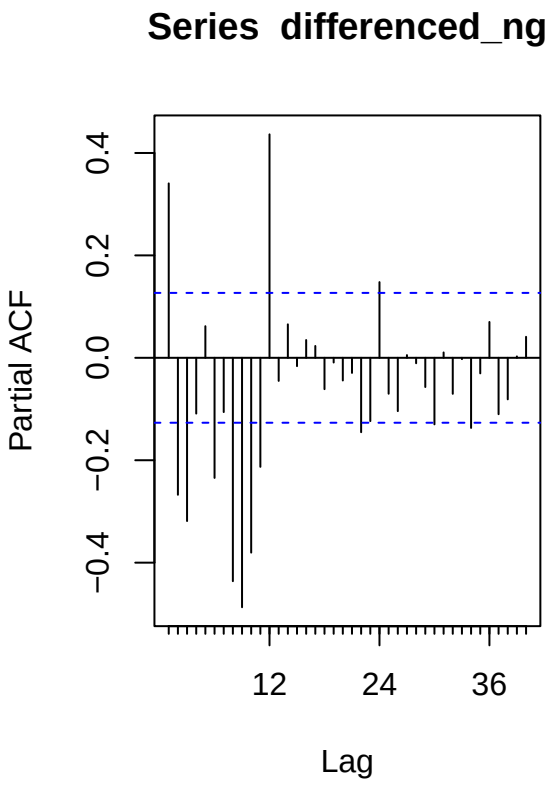
```
##
##  Augmented Dickey-Fuller Test
##
## data:  ng_ts
## Dickey-Fuller = -8.9602, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```
print(ng_SMK)
```

```
## tau = 0.887, 2-sided pvalue =< 2.22e-16
```

```
differenced_ng <- diff(ng_ts, differences = 1)
par(mfrow=c(1,2))
Acf(differenced_ng, lag = 40)
Pacf(differenced_ng, lag = 40)
```






```
nsdiffs(ng_ts)
```

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

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.

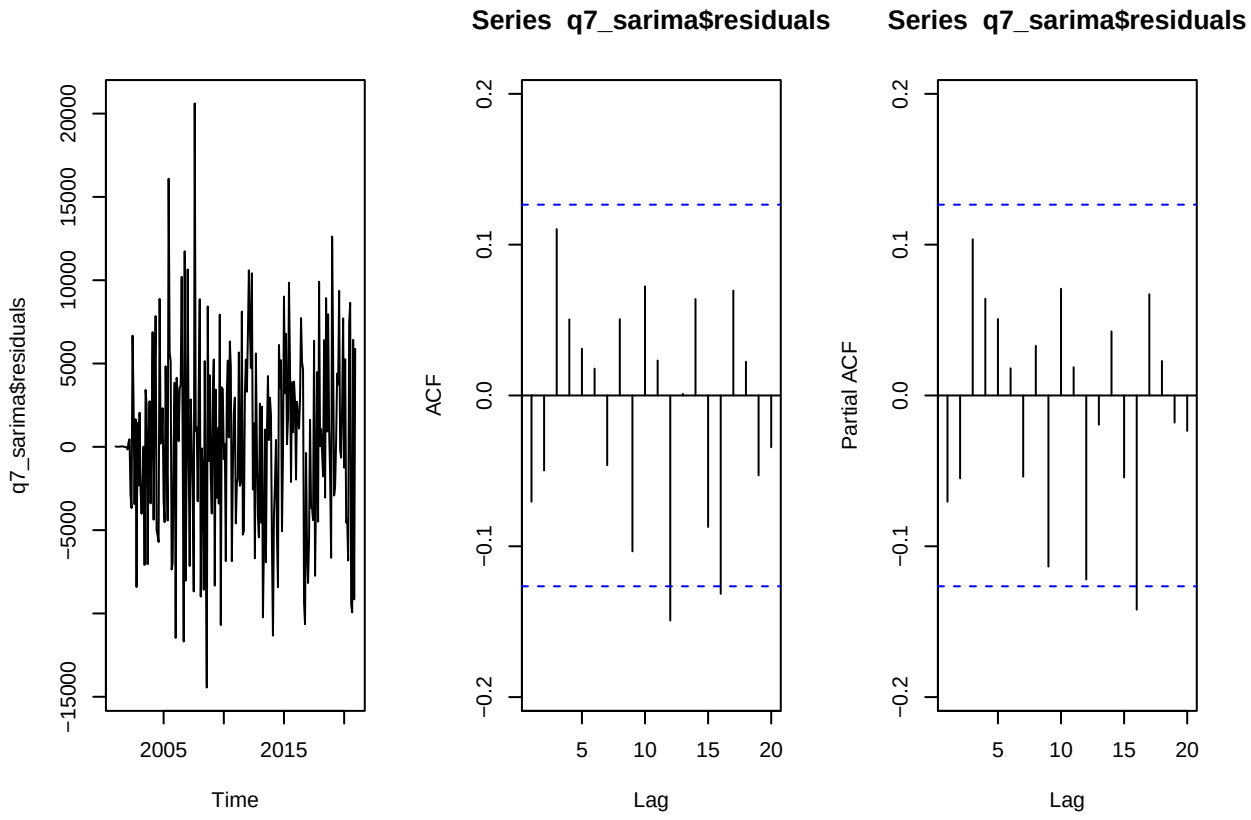
Looking at the residual series, I cannot very tell if there is a big difference, but both models appear to have randomness, except SARIMA model seems to have fewer points that are beyond the control limits. Whereas the ARIMA residual plot has big outliers at around positive 20000. Neither residual plot shows significant cycling or trend. For bias, It is hard to tell, but it appears that there is also a slight negative bias in the ARIMA model. Thus, perhaps the SARIMA model is a better representation. It is a fair comparison (or unfair depending on the point of view) since the SARIMA model has that extra level degree of granularity by being able to model different types of seasonal processes while the ARIMA model just relies on removing seasonality.

```
q7_sarima <- arima(ng_ts, order = c(1,1,1), seasonal = list(order = c(1,1,0), period = 12))
q7_sarima
```

```
##
## Call:
## arima(x = ng_ts, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 0), period = 12))
```

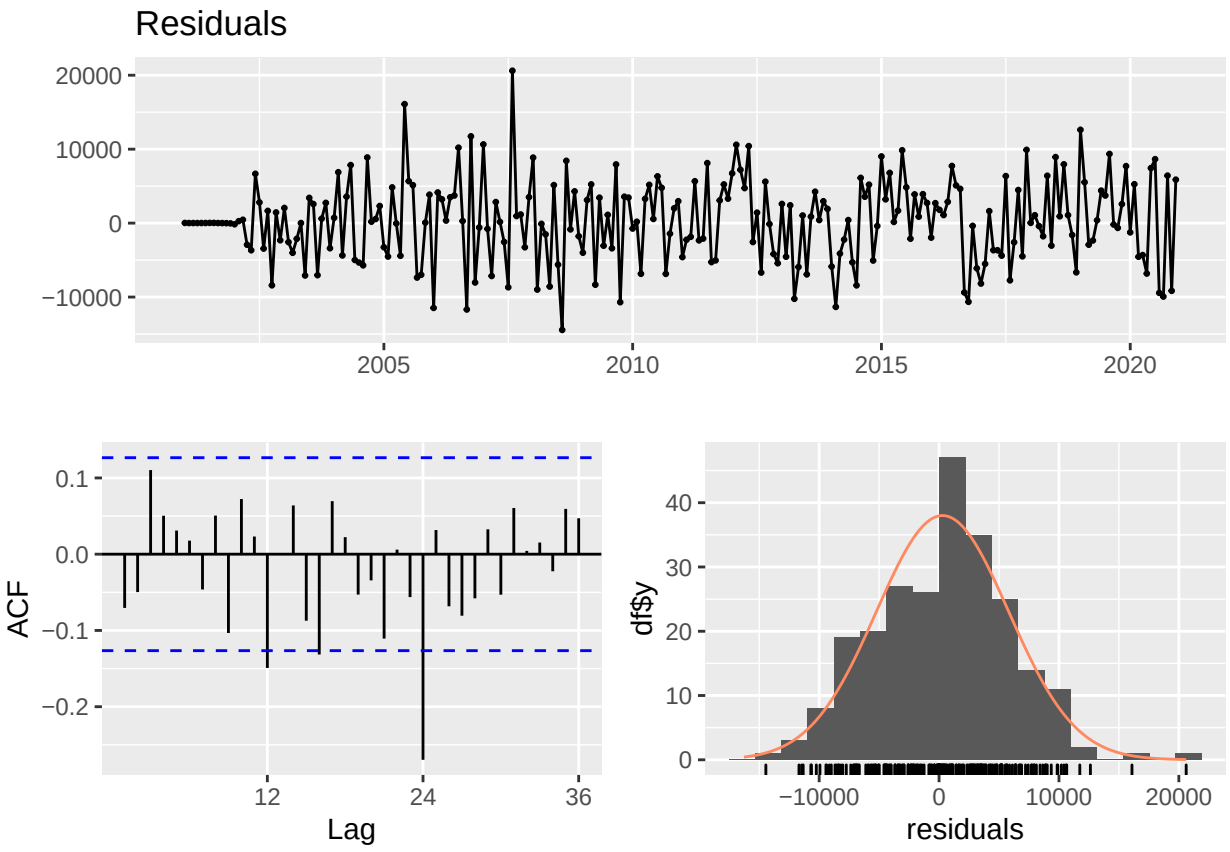
```
##
## Coefficients:
##          ar1      ma1     sar1
##       0.7722  -1.0000  -0.4526
## s.e.  0.0432   0.0213   0.0595
##
## sigma^2 estimated as 32175921:  log likelihood = -2287.56,  aic = 4583.12
```

```
par(mfrow=c(1,3))
ts.plot(q7_sarima$residuals)
Acf(q7_sarima$residuals, lag = 20)
Pacf(q7_sarima$residuals, lag = 20)
```




```
#just trying the checkresiduals() here
checkresiduals(q7_sarima$residuals)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```



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 correct orders. The intention of the assignment is to walk you to the process and help you figure out what you did wrong (if you did anything wrong!).

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?

The order here is ARIMA(1,1,1), which is exactly what I got in Q4!

```
auto.arima(deseasonal_ng)
```

```
## Series: deseasonal_ng
## 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
```

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?

The model here is ARIMA(1,0,0)X(0,1,1)12, which is not even close to what I got. I somehow got there needs to be differencing because of presence of trend, leading me away from a non-seasonal AR process and going for an ARIMA process for the non-seasonal component.

Furthermore, for the seasonal component, I am very unsure how it's an SMA process and not a SAR process since my ACF and PACF plots very clearly indicate a SAR process. I did get the D = 1 though due to nsdiffs!

```
auto.arima(ng_ts)
```

```
## Series: ng_ts
## 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
```