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

Assignment 6 - Due date 03/26/21

Keyang Xue

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_A06_Sp21.Rmd"). Submit this pdf using Sakai.

Set up

```
library(forecast)
library(tseries)
library(dplyr)
library(lubridate)
library(Kendall)
```

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.

```
electricity_generation <- read.csv(file="../Data/Net_generation_United_States_all_sectors_monthly.csv",
                                   header=TRUE,skip=4)
nvar <- ncol(electricity_generation) - 1
nobs <- nrow(electricity_generation)
```

```r
#create date object and rename columns
electricity_generation_processed <-
  electricity_generation %>%
  mutate( Month = my(Month) ) %>%
  rename( All.fuels = all.fuels..utility.scale..thousand.megawatthours ) %>%
  rename( Coal = coal.thousand.megawatthours ) %>%
  rename( NaturalGas = natural.gas.thousand.megawatthours ) %>%
  rename( Nuclear = nuclear.thousand.megawatthours ) %>%
  rename( ConventionalHydro = conventional.hydroelectric.thousand.megawatthours ) %>%
  arrange( Month )

head(electricity_generation_processed)
```

```
##        Month All.fuels     Coal NaturalGas  Nuclear ConventionalHydro
## 1 2001-01-01  332493.2 177287.1   42388.66 68707.08          18852.05
## 2 2001-02-01  282940.2 149735.5   37966.93 61272.41          17472.89
## 3 2001-03-01  300706.5 155269.0   44364.41 62140.71          20477.19
## 4 2001-04-01  278078.9 140670.7   45842.75 56003.03          18012.99
## 5 2001-05-01  300491.6 151592.9   50934.21 61512.44          19175.63
## 6 2001-06-01  327694.0 162615.8   57603.15 68023.10          20727.63
```

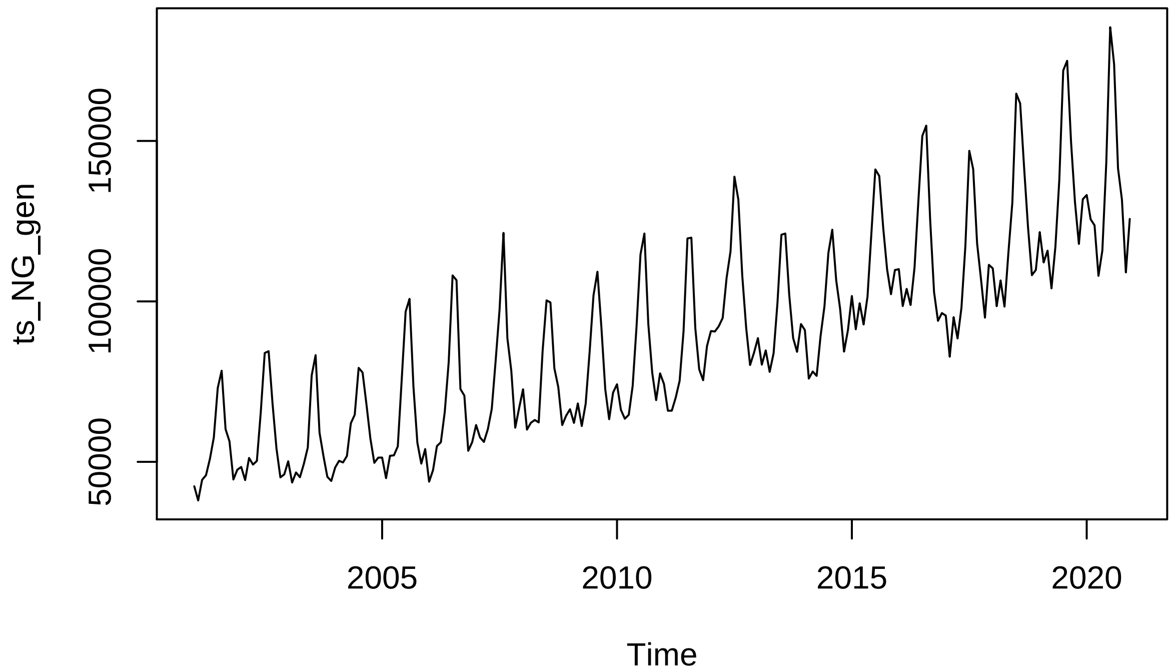
```r
ts_NG_gen <- ts(electricity_generation_processed[,4],
                start=c(year(electricity_generation_processed$Month[1]),
                        month(electricity_generation_processed$Month[1])),
                frequency=12)

head(ts_NG_gen,15)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2001 42388.66 37966.93 44364.41 45842.75 50934.21 57603.15 73030.14 78409.80
## 2002 48412.83 44308.43 51214.46
##           Sep      Oct      Nov      Dec
## 2001 60181.14 56376.44 44490.62 47540.86
## 2002
```

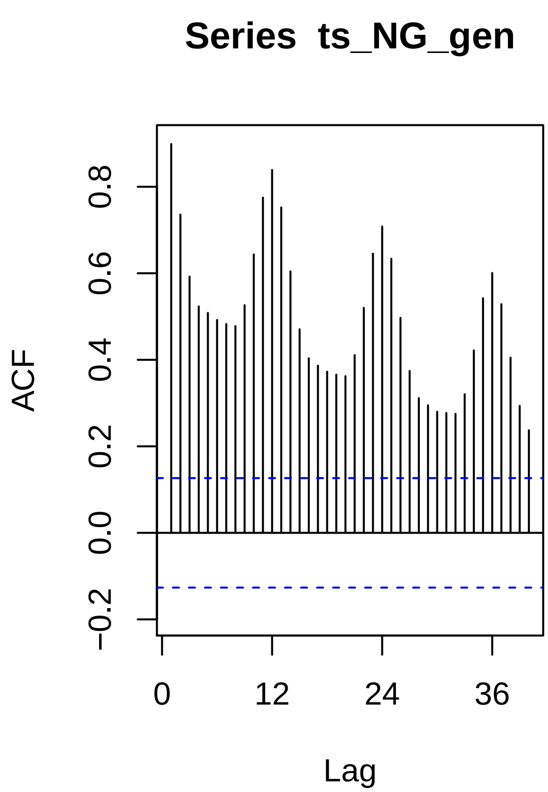
```r
tail(ts_NG_gen,15)
```

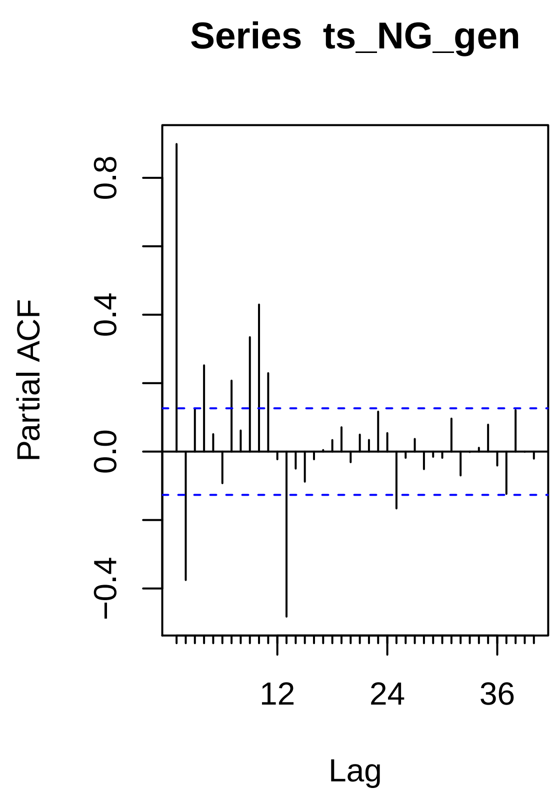
```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2019
## 2020 133157.6 125593.9 123697.0 107960.0 115870.9 143245.4 185444.8 173926.6
##           Sep      Oct      Nov      Dec
## 2019          130947.6 117910.5 131838.9
## 2020 141452.7 131658.2 109037.2 125703.7
```

```r
ts.plot(ts_NG_gen)
```




```
par(mfrow=c(1,2))
NG_ACF <- Acf(ts_NG_gen, lag.max = 40, plot = TRUE)
NG_PACF <- Pacf(ts_NG_gen, lag.max = 40, plot = TRUE)
```

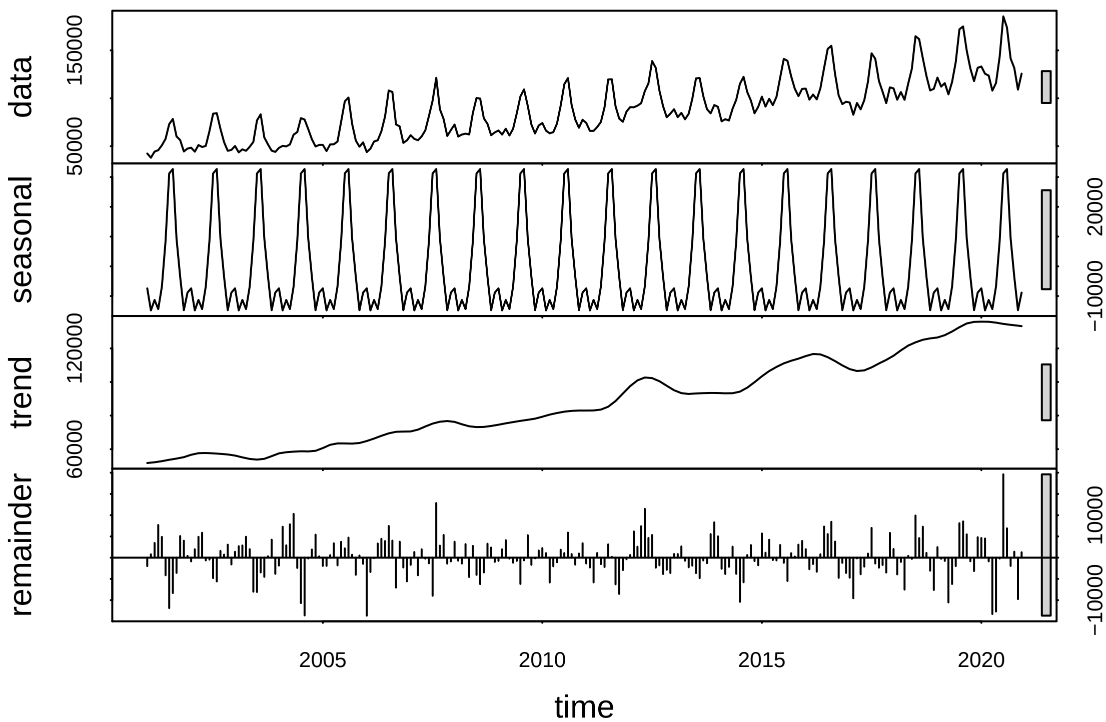




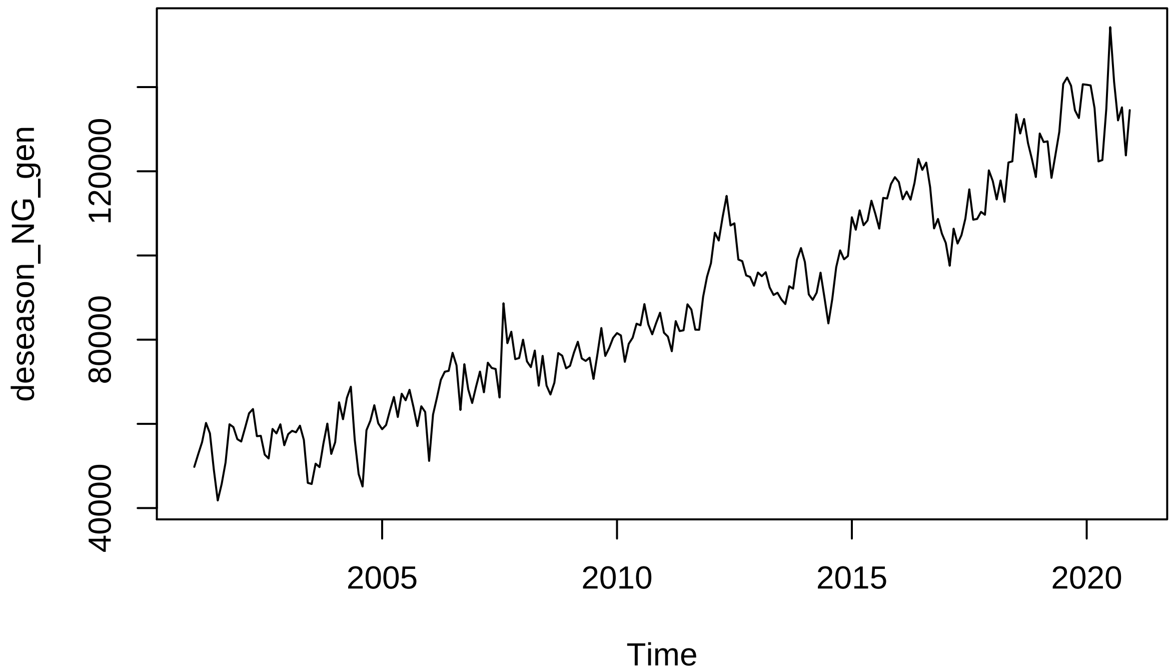
Q2

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.

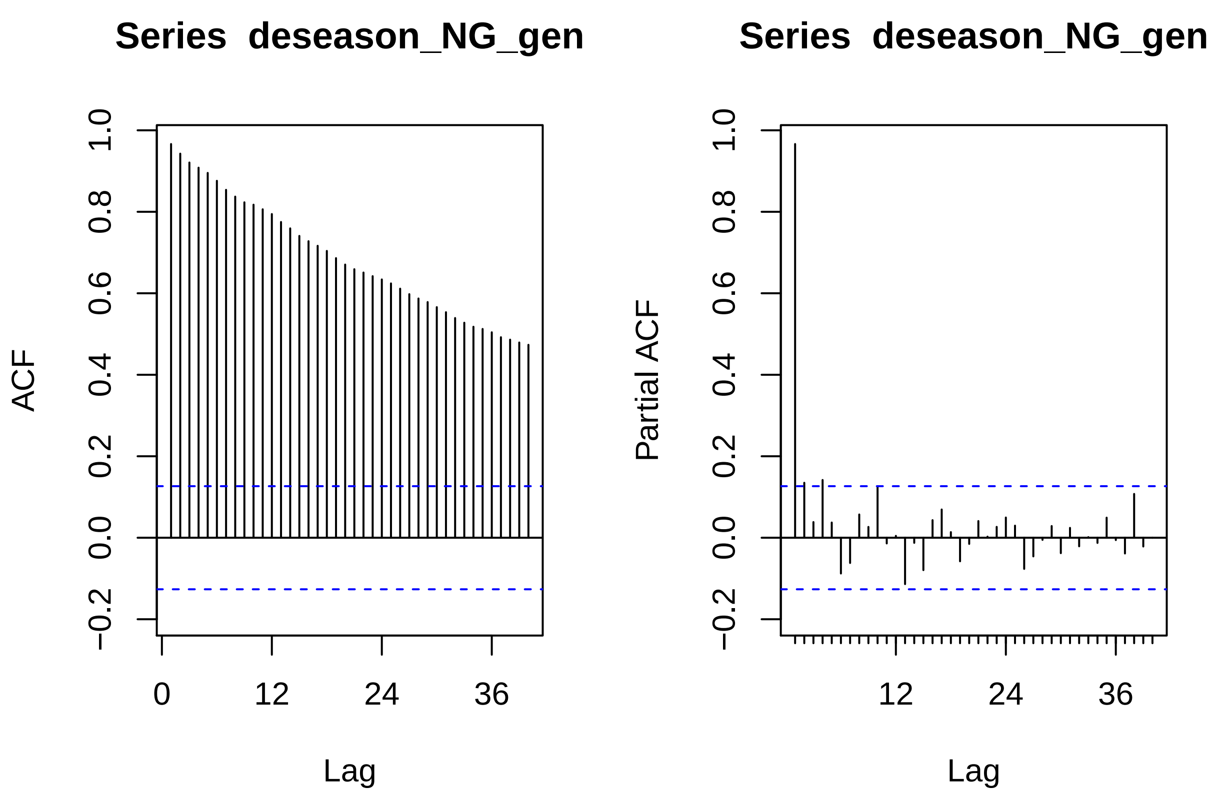
```
decompose_NG_gen <- stl(ts_NG_gen, s.window = "periodic")
plot(decompose_NG_gen)
```




```
deseason_NG_gen <- seasadj(decompose_NG_gen)
ts.plot(deseason_NG_gen)
```




```r
par(mfrow=c(1,2))
deseason_NG_ACF <- Acf(deseason_NG_gen, lag = 40, plot = TRUE)
deseason_NG_PACF <- Pacf(deseason_NG_gen, lag = 40)
```



The time series plot does not show seasonality any more because it does not have cyclical/seasonal pattern like the plot in Q1 does. Similarly, the ACF plot does not have spikes anymore, showing no seasonality. The PACF plot shows more values fall into the significant range throughout the 40 lags compared to that in Q1.

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.

```
summary(MannKendall(deseason_NG_gen))
```

```
## Score =  24196 , Var(Score) = 1545533
## denominator =  28680
## tau = 0.844, 2-sided pvalue =< 2.22e-16
```

```
adf.test(deseason_NG_gen, alternative = "stationary")
```

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

```
##
##  Augmented Dickey-Fuller Test
##
## data:  deseason_NG_gen
## Dickey-Fuller = -4.01, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

The results of the Mann Kendall test for deseasoned NG generation series reject the null (the series is stationary) and conclude that this series is not stationary and has a increasing trend (p<=2.22e-16, score=24196, n=240). The ADF test rejects the null hypothesis and reaches the same conclusion that this series is not stationary (p=0.01).

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.

```
#how many difference needed
ndiffs(deseason_NG_gen)
```

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

The series has an increasing trend, so it needs a difference. Since the ndiffs() function gives a result of 1, d=1. The ACF plot decays exponentially, and the PACF cuts off after lag 1 (lag 2 only marginally significant, so assume cuts off after lag 1), indicating this is a AR process. Hence, p=1,q=0. –> ARIMA(1,1,0)

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.

```
arimaNG_deseas <- Arima(deseason_NG_gen, order = c(1,1,0),include.drift=TRUE)

#check if differenced series needs another difference
ndiffs(arimaNG_deseas$residuals)
```

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

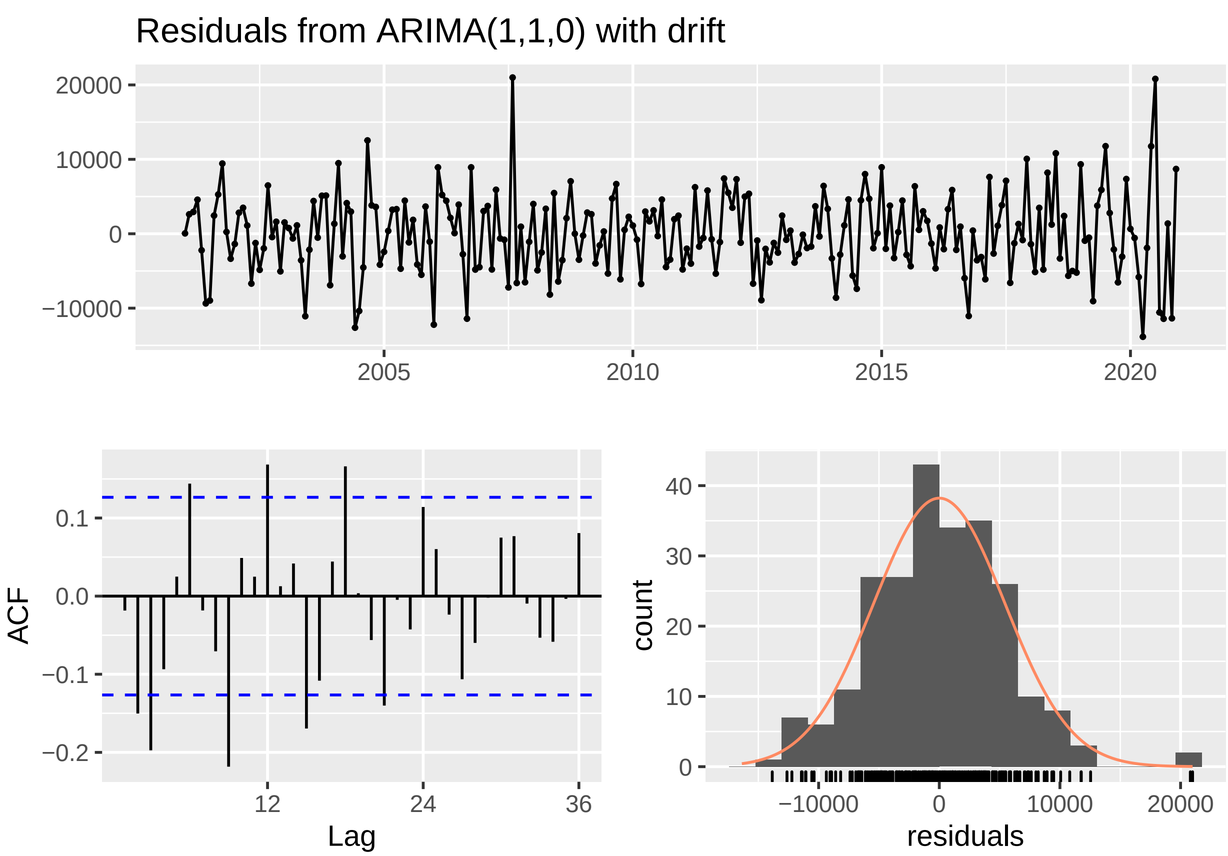
```
cat("The coefficients are",arimaNG_deseas$coef)
```

```
## The coefficients are -0.1453268 347.6758
```

Q6

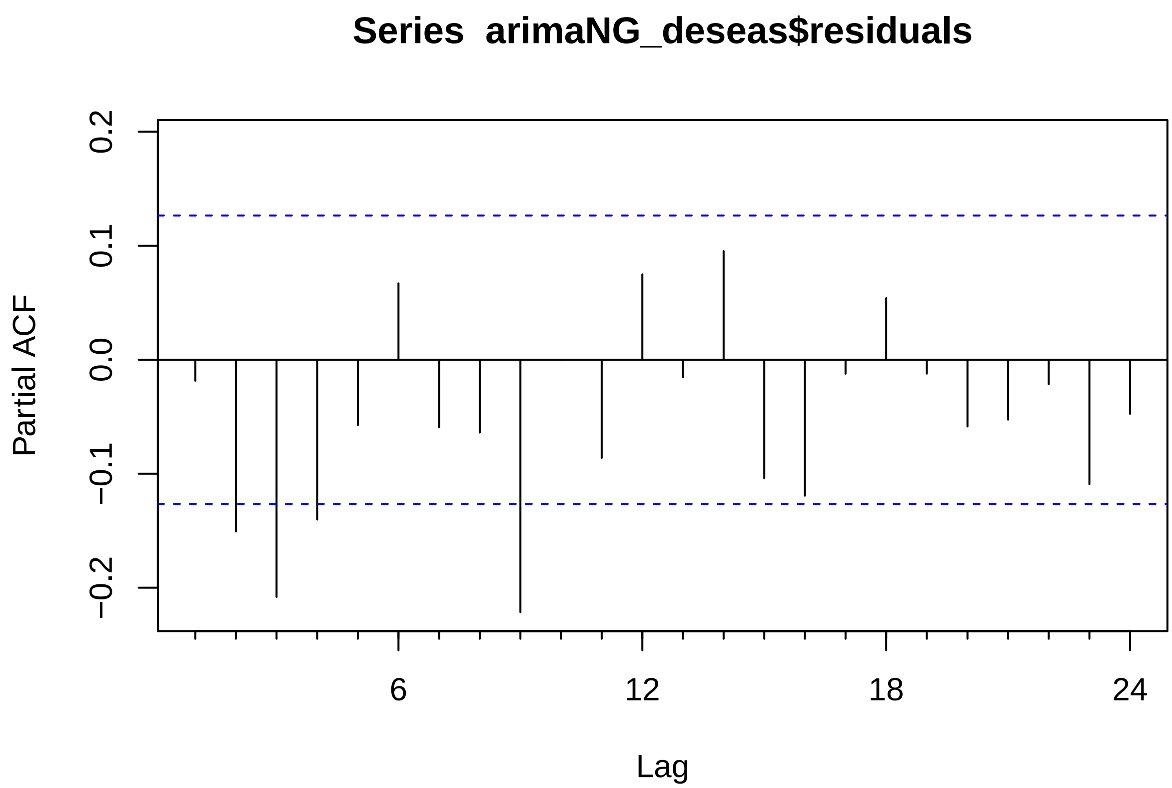
Now plot the residuals of the ARIMA fit from Q5 along with residuals ACF and PACF on the same window. You may use the $checkresiduals()$ function to automatically generate the three plots. Do the residual series look like a white noise series? Why?

```
checkresiduals(arimaNG_deseas, plot = TRUE)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0) with drift
## Q* = 72.598, df = 22, p-value = 2.564e-07
##
## Model df: 2.   Total lags used: 24
```

```
Pacf(arimaNG_deseas$residuals)
```



It looks like a white noise series from the time series plot since the values are oscillating randomly around 0. However, some of the ACF and PACF values are still beyond the significant range, indicating it is not a perfect white noise series.

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.

```
#check how many differences needed
nsdiffs(ts_NG_gen)
```

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

```
#try fitting with 1 seasonal differencing
arimaNG <- Arima(ts_NG_gen, order = c(2,0,0),seasonal = c(1,1,0),include.drift=TRUE)

#check if needs further differencing
ndiffs(arimaNG$residuals)
```

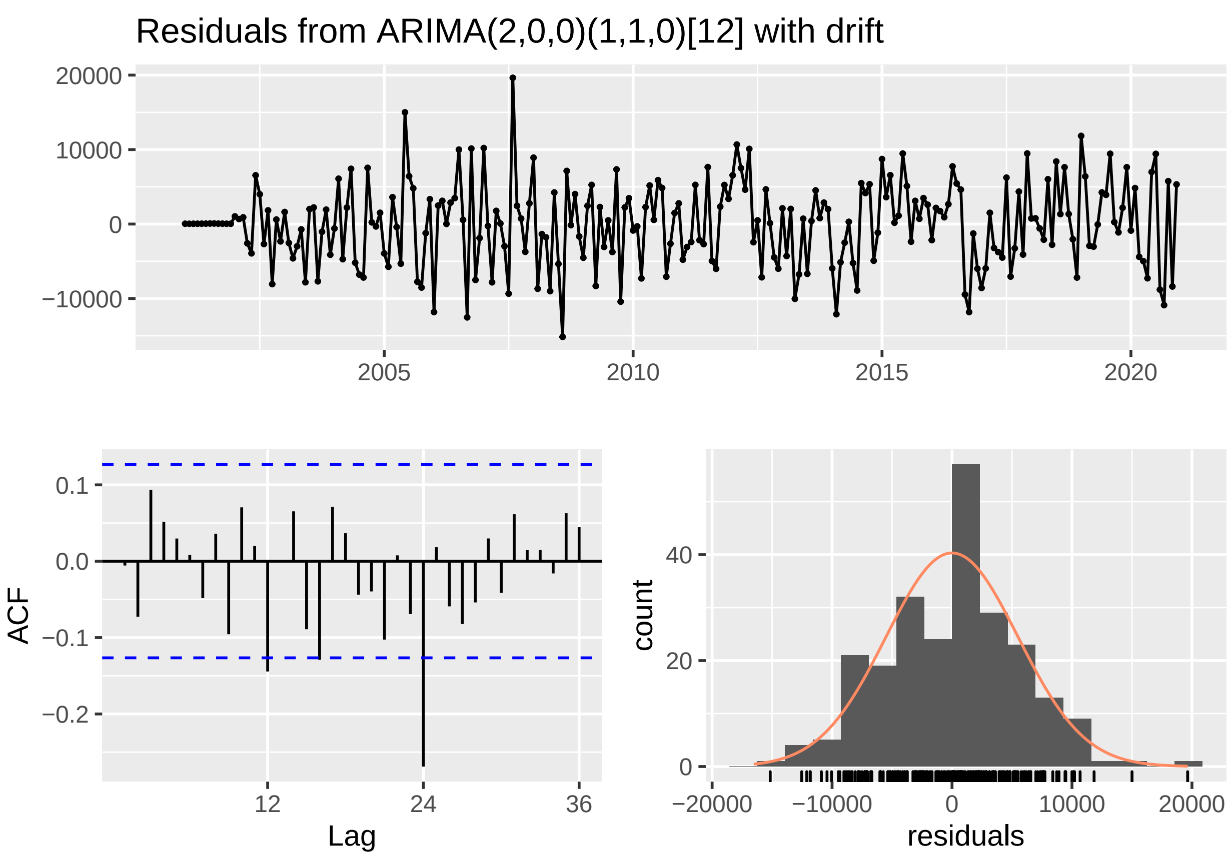
```
## [1] 0
```

```
cat("The coefficients are", arimaNG$coef)
```

```
## The coefficients are 0.7013958 0.08265526 -0.4561543 356.9487
```

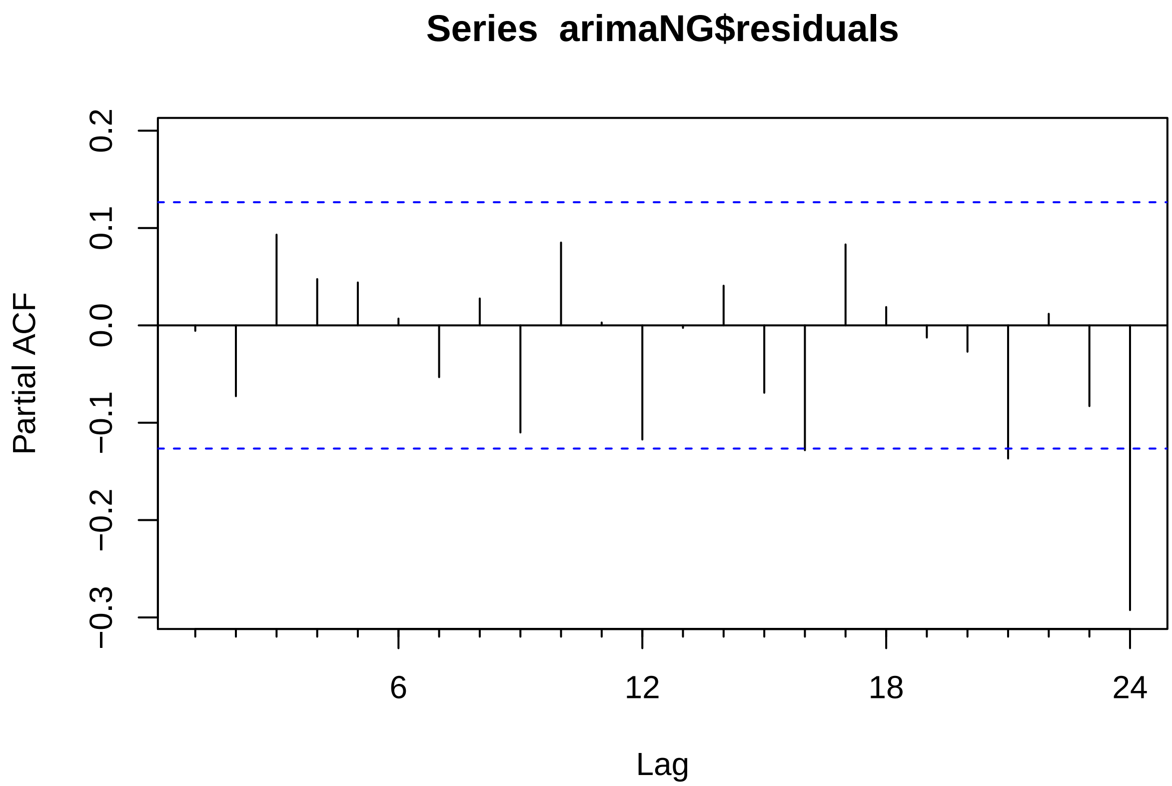
Based on the results from nsdiffs() and ndiffs(), d=0, D=1.For the non-seasonal lags, the ACF plot has a slow decay, and the PACF plot cuts off after lag 2. Therefore, this is an AR process, and p=2.For the seasonal lags, the ACF plot has multiple spikes and the PACF plot only has one spike, indicating this is a SAR process with P=1. -> ARIMA(2,0,0)(1,1,0)[12]

```
checkresiduals(arimaNG)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,0)(1,1,0)[12] with drift
## Q* = 47.771, df = 20, p-value = 0.000458
##
## Model df: 4.   Total lags used: 24
```

```
Pacf(arimaNG$residuals)
```



It looks like a white noise series because the time series plot looks random and has a mean of 0 while most of the ACF nad PACF values are within the significant range.

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.

```
AIC(arimaNG)
```

```
## [1] 4599.438
```

```
AIC(arimaNG_deseas)
```

```
## [1] 4797.119
```

More ACF and PAF values of the second ARIMA model with seasonality are within the significant range compared to the first model without seasonality, meaning that the second one is a better model. Using AIC, I found that the second model has a lower AIC, indicating this is a fair comparison and the second model is a better 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!).

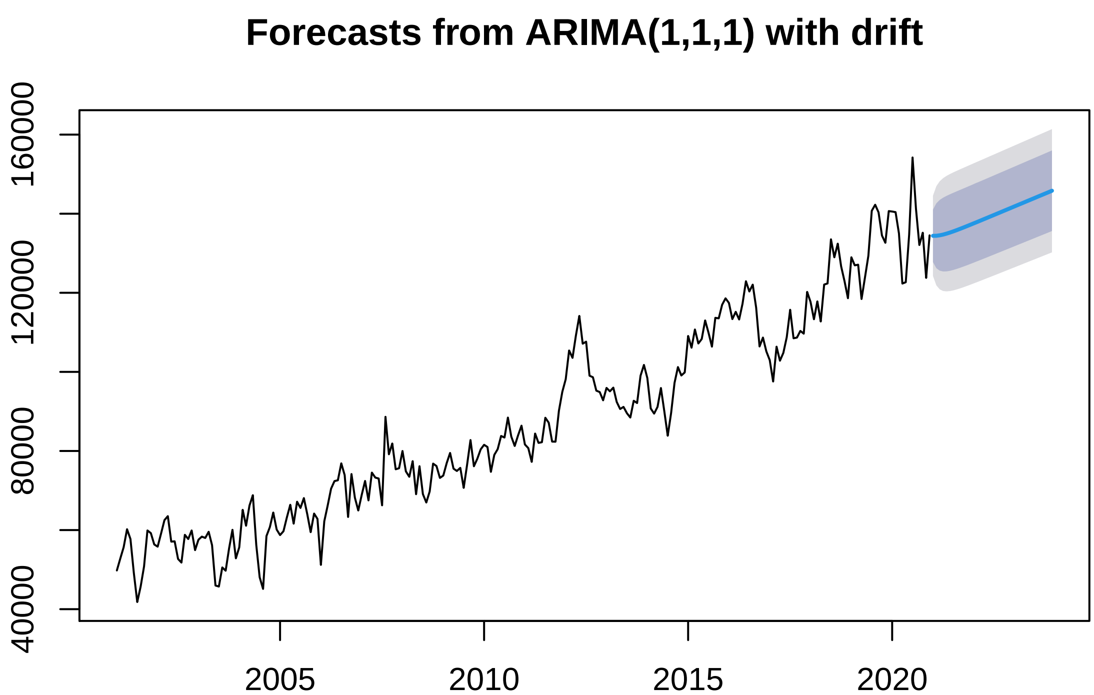
Q9

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?

```
autofit_deseason_NG <- auto.arima(deseason_NG_gen, max.D = 0, max.P = 0, max.Q = 0)
print(autofit_deseason_NG)
```

```
## Series: deseason_NG_gen
## ARIMA(1,1,1) with drift
##
## Coefficients:
##          ar1      ma1     drift
##       0.7085  -0.9795  359.3879
## s.e.  0.0633   0.0327   29.5499
##
## sigma^2 estimated as 26771444:  log likelihood=-2382.17
## AIC=4772.35   AICc=4772.52   BIC=4786.25
```

```
forecast_deseason_NG <- forecast(object = autofit_deseason_NG, h = 36)
plot(forecast_deseason_NG)
```



ARIMA(1,1,1) with drift. I failed to identify the MA process and missed the q part in the model.

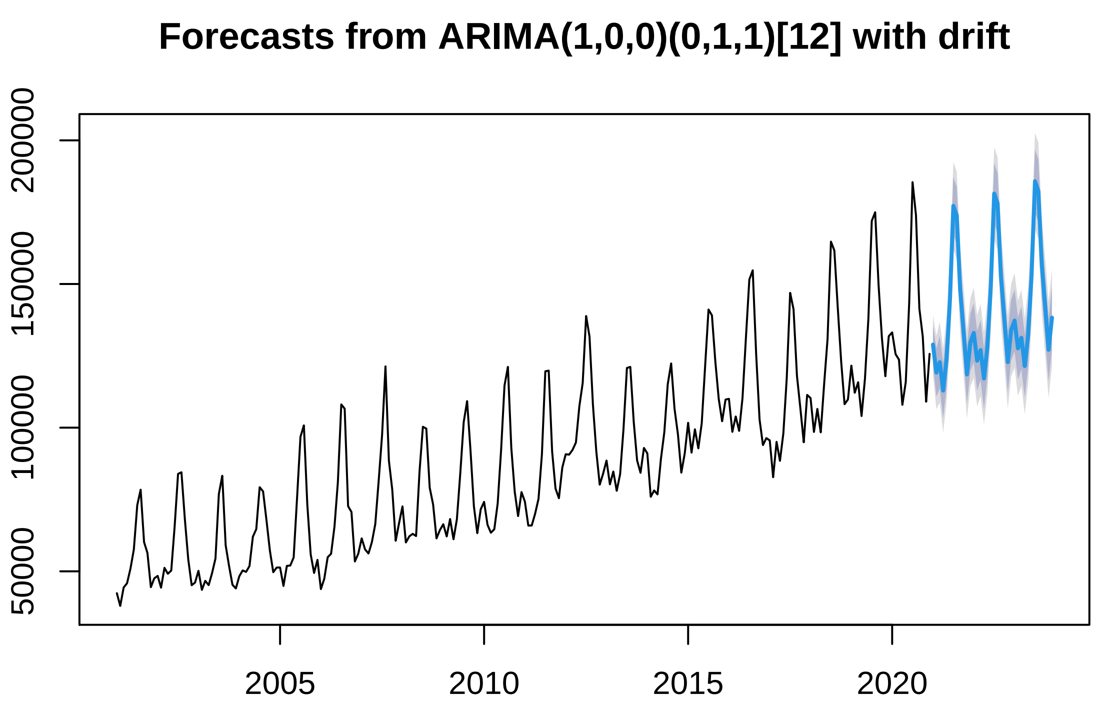
Q10

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?

```
autofit_NG <- auto.arima(ts_NG_gen)
print(autofit_NG)
```

```
## Series: ts_NG_gen
## 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 estimated as 27569124:  log likelihood=-2279.54
## AIC=4567.08   AICc=4567.26   BIC=4580.8
```

```
forecast_NG <- forecast(object = autofit_NG, h = 36)
plot(forecast_NG)
```



ARIMA(1,0,0)(0,1,1)[12] with drift. I misidentified the SMA process as SAR process. I also misidentified the order of the AR process to be 2, which should be 1.