ENV 797 - Time Series Analysis for Energy and Environment Applications | Spring 2025

Assignment 7 - Due date 03/06/25

Rosie Wu

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_Sp25.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.

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

Set up

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

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

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

```
library(ggplot2)
library(forecast)
```

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

```r
library(Kendall)
library(tseries)
library(outliers)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr   1.1.4     v stringr 1.5.1
## v forcats 1.0.0     v tibble  3.2.1
## v purrr   1.0.2     v tidyr   1.3.1
## v readr   2.1.5

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(cowplot)
```

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

```r
library(trend)
```

Importing and processing the data set

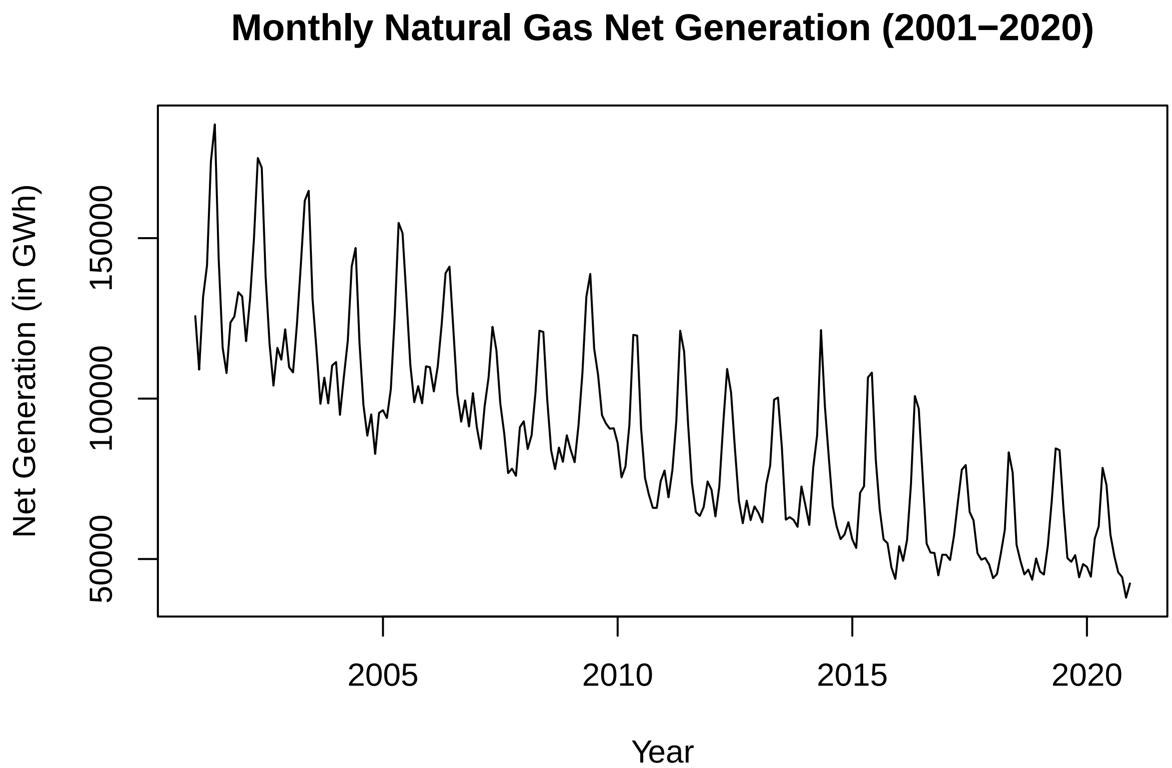
Consider the data from the file "Net_generation_United_States_all_sectors_monthly.csv". The data corresponds to the monthly net generation from January 2001 to December 2020 by source and is provided by the US Energy Information and Administration. **You will work with the natural gas column only**.

Q1

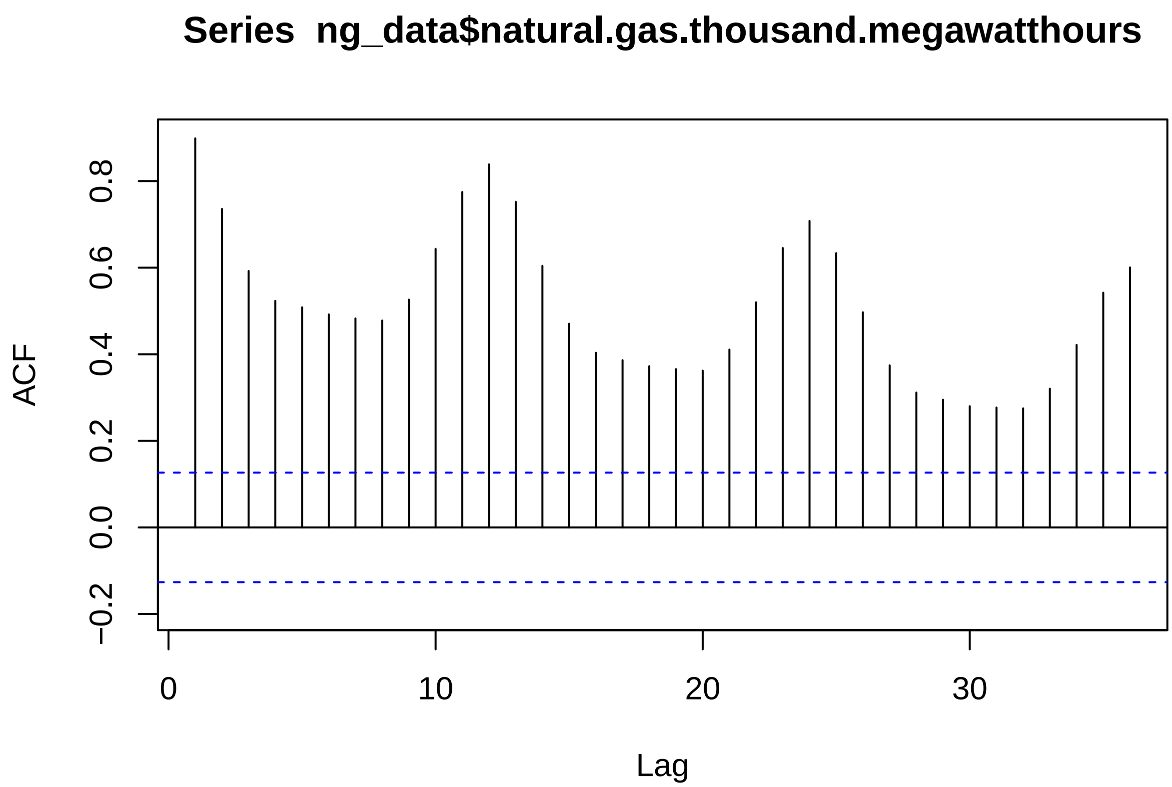
Import the csv file and create a time series object for natural gas. Make you sure you specify the **start=** and **frequency=** arguments. Plot the time series over time, ACF and PACF.

```r
net_gen_data <- read.csv(file="./Data/Net_generation_United_States_all_sectors_monthly.csv",header=TRUE

ng_data <- net_gen_data[, c("Month", "natural.gas.thousand.megawatthours")]

# convert the month column to a proper format
ng_data$Month <- parse_date_time(ng_data$Month, orders = "my")
```

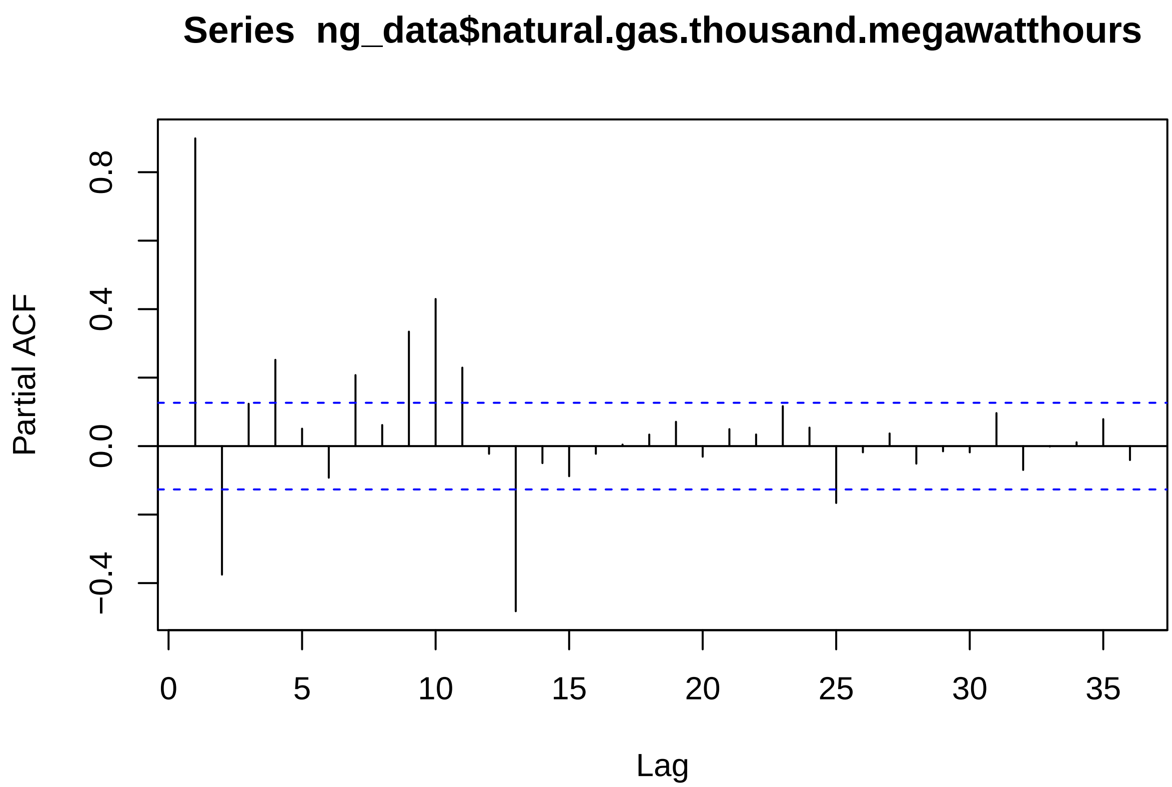
```r
# Extract the Natural Gas values and create a time series
natural_gas_ts <- ts(ng_data$natural.gas.thousand.megawatthours, start = c(2001, 1), frequency = 12)

# Plot the time series
ts_plot1 <- plot(natural_gas_ts, main = "Monthly Natural Gas Net Generation (2001-2020)",
     xlab = "Year", ylab = "Net Generation (in GWh)")
```




```
# ACF Plot
acf_plot1 <- Acf(ng_data$natural.gas.thousand.megawatthours,
    lag = 36,  plot = TRUE)
```



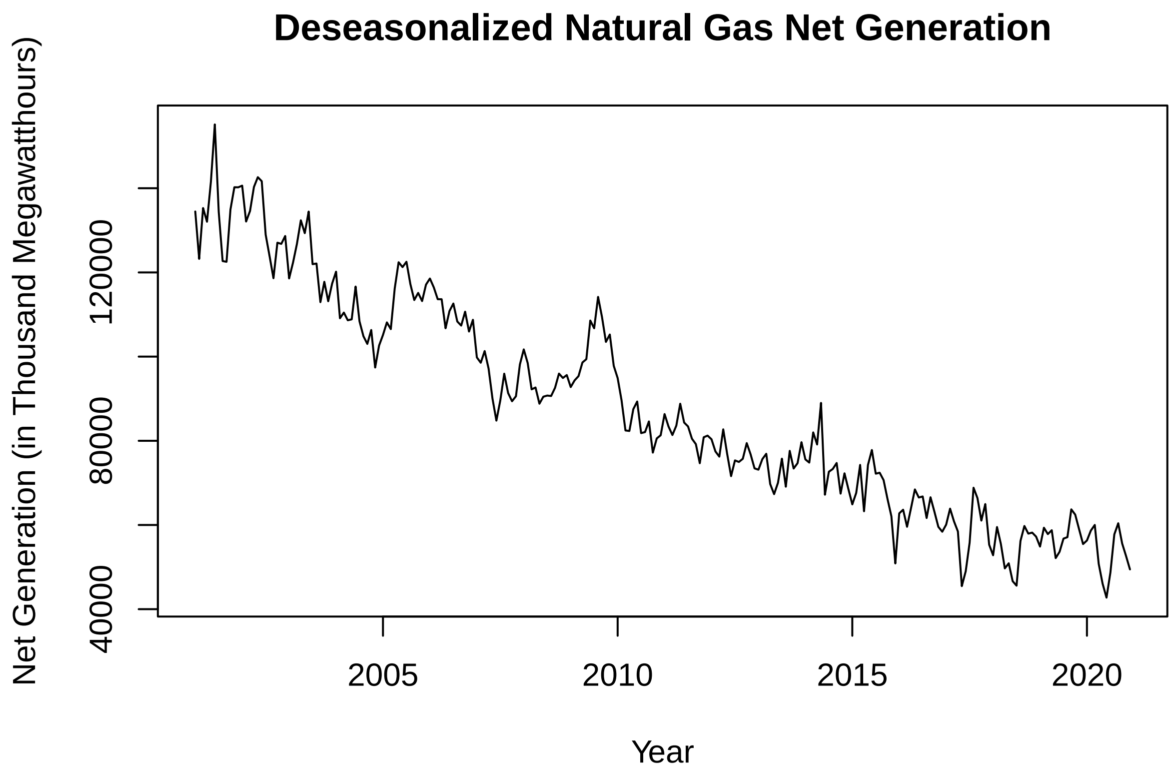

```
# PACF Plot
pacf_plot1 <-  Pacf(ng_data$natural.gas.thousand.megawatthours,
    lag = 36,  plot = TRUE)
```



Q2

Using the *decompose*() 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.

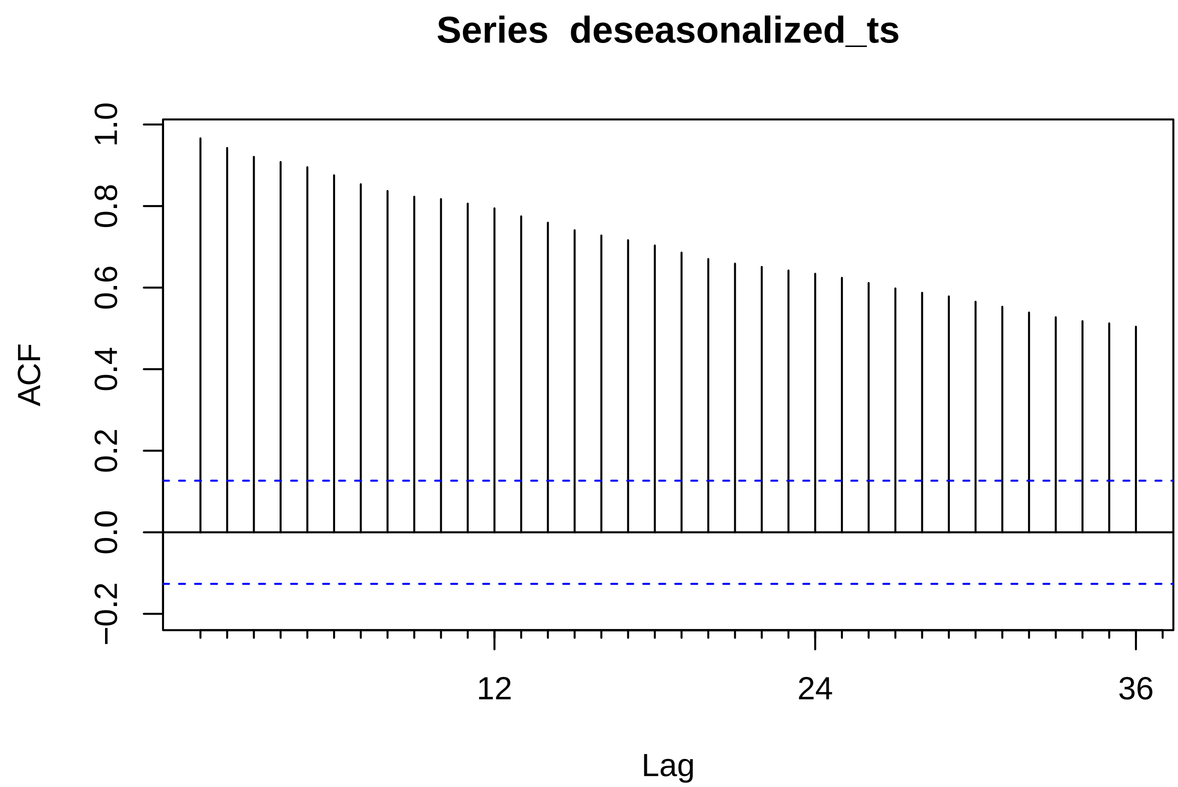
```r
# Decompose the time series
decomposed_ts <- decompose(natural_gas_ts, type = "additive")

# Deseasonalize the time series using seasadj()
deseasonalized_ts <- seasadj(decomposed_ts)

# Plot the deseasonalized series
deseasoned_plot <- plot(deseasonalized_ts, main = "Deseasonalized Natural Gas Net Generation",
     xlab = "Year", ylab = "Net Generation (in Thousand Megawatthours)")
```




```r
print(deseasoned_plot)
```

```
## NULL
```

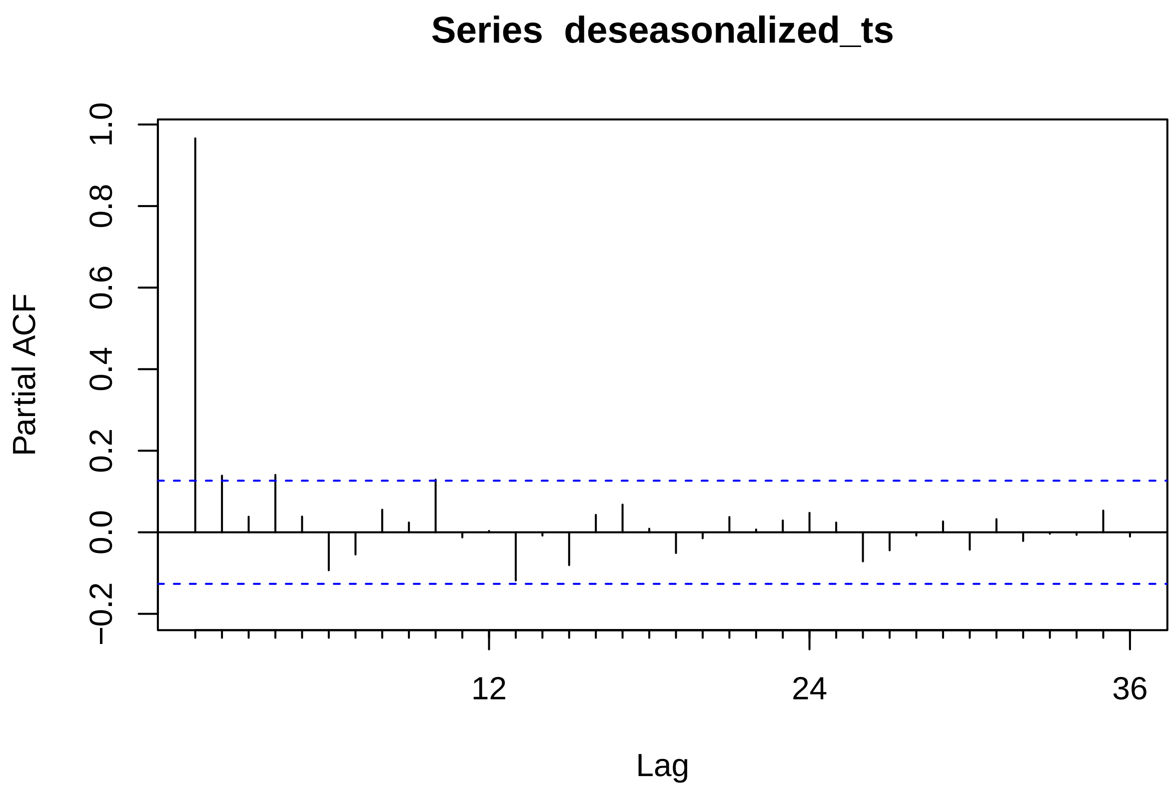
```r
# updated ACF Plot
acf_plot2 <- Acf(deseasonalized_ts, lag = 36, plot = TRUE)
```




```r
print(acf_plot2)
```

```
##
## Autocorrelations of series 'deseasonalized_ts', by lag
##
##     0     1     2     3     4     5     6     7     8     9    10    11    12
## 1.000 0.966 0.943 0.921 0.908 0.895 0.876 0.854 0.837 0.823 0.817 0.806 0.794
##    13    14    15    16    17    18    19    20    21    22    23    24    25
## 0.775 0.759 0.741 0.728 0.716 0.703 0.686 0.670 0.659 0.651 0.642 0.634 0.624
##    26    27    28    29    30    31    32    33    34    35    36
## 0.611 0.598 0.587 0.578 0.565 0.553 0.539 0.527 0.518 0.512 0.504
```

```r
# updated PACF Plot
pacf_plot2 <- Pacf(deseasonalized_ts, lag = 36, plot = TRUE)
```




```r
print(pacf_plot2)
```

```
##
## Partial autocorrelations of series 'deseasonalized_ts', by lag
##
##      1      2      3      4      5      6      7      8      9     10     11
##  0.966  0.139  0.038  0.141  0.039 -0.093 -0.054  0.055  0.024  0.129 -0.013
##     12     13     14     15     16     17     18     19     20     21     22
##  0.003 -0.118 -0.008 -0.081  0.043  0.068  0.009 -0.051 -0.015  0.037  0.007
##     23     24     25     26     27     28     29     30     31     32     33
##  0.029  0.048  0.024 -0.071 -0.044 -0.008  0.027 -0.043  0.032 -0.021 -0.004
##     34     35     36
## -0.006  0.053 -0.011
```

The deseasonalized times series in this question shows a more random pattern than the times series plot in question 1. Yet both of the time series are sloping down. The deseasonalized ACF in Q2 shows less seasonal patterns/ peaks, while the original ACF graph showed more seasonal patterns and peaks. The Deseasoned PACF also showed an immediate drop after the first lag, but still being positive, while the original PACF shows that the value of PACF drop to negative after lag 1, while could indicate a AR model?

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.

```r
adf_test3 <- adf.test(deseasonalized_ts)
```

```
## Warning in adf.test(deseasonalized_ts): p-value smaller than printed p-value
```

```r
print(adf_test3)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  deseasonalized_ts
## Dickey-Fuller = -4.0574, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```r
mk_test3 <- mk.test(deseasonalized_ts)
print(mk_test3)
```

```
##
##  Mann-Kendall trend test
##
## data:  deseasonalized_ts
## z = -19.454, n = 240, p-value < 2.2e-16
## alternative hypothesis: true S is not equal to 0
## sample estimates:
##             S          varS           tau
## -2.418600e+04  1.545533e+06 -8.433054e-01
```

> ADF test: Null for this test is that the time series of natural gas generation has a unit root (it's non-stationary). Alternative: The time series is stationary. The p-value for this test turns out to be 0.01 (<0.05), which means we can reject the null and state that the deseasoned time series of natural gas generation could be stationary. Mann-Kendall trend test: Null: no trend in time series. Alternative: there is a trend in time series. p-value in test is much less than 0.05, so we reject the null and state that there is a significant trend in the deseasoned time series. The negative value of S also indicates that the trend is slowly decreasing, which matches the previous results of the ts plots.

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 understand the ACF/PACF plots and interpret the test results.

> From the ADF test, we know that the series is stationary, so d=0 From the PACF plot of the deseasonalized series, the PACF value cuts off at lag 2, so the p = 2 From the ACF plot of the deseasonalized series, the ACF has no significant cut offs/ spikes, so q=0.

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()` or `print()` function to print.

```
# values from observing the previous plots
p <- 2
d <-0
q <- 0
# Fit the ARIMA model with a constant term
arima_model5 <- Arima(deseasonalized_ts, order = c(p, d, q), include.mean = TRUE)

# Print the coefficients of the fitted model
cat("ARIMA Model Coefficients:\n")
```

```
## ARIMA Model Coefficients:
```

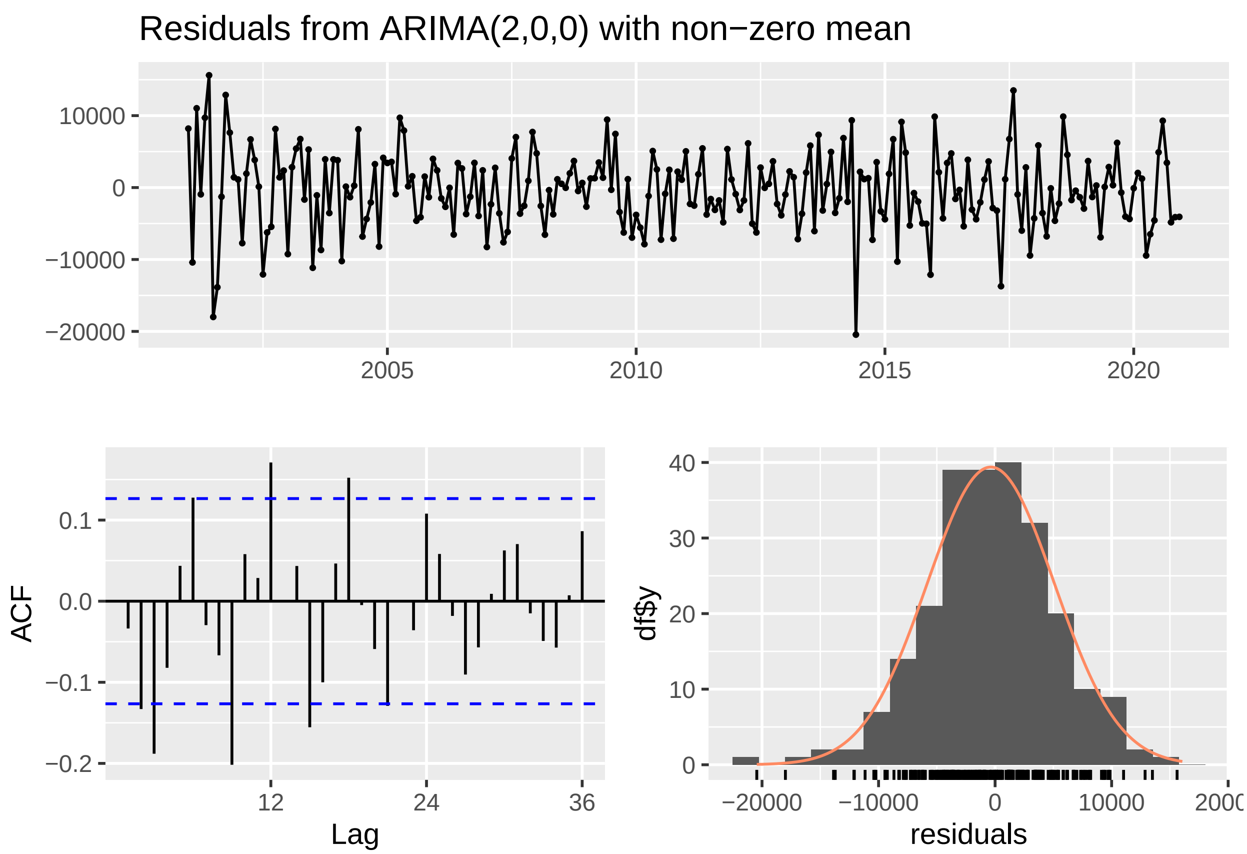
```
print(arima_model5$coef)
```

```
##          ar1          ar2    intercept
## 8.499034e-01 1.360714e-01 8.834881e+04
```

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?

```
# Use checkresiduals() to generate the residual plots
checkresiduals(arima_model5)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,0) with non-zero mean
## Q* = 64.249, df = 22, p-value = 5.143e-06
##
## Model df: 2.   Total lags used: 24
```

The deseasoned series do display some white noise. The residuals plot show some randomness overtime, as it fluctuates around 0 with no visible patterns or trends. ACF plot shows that there are some significant values (not very significant though) in some lags, so not all autocorrelations lie within the confidence intervals, so this doesn't necessarily indicate white noise, but it's relatively close. Accoding to the histogram of residuals, it shows a normal distribution around mean of zero, which indicates white noise.

Modeling the original series (with seasonality)

Q7

Repeat Q3-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.

```r
# first do the tests on the original series
adf_test7 <- adf.test(natural_gas_ts)
```

```
## Warning in adf.test(natural_gas_ts): p-value smaller than printed p-value
```

```r
print(adf_test7)
```

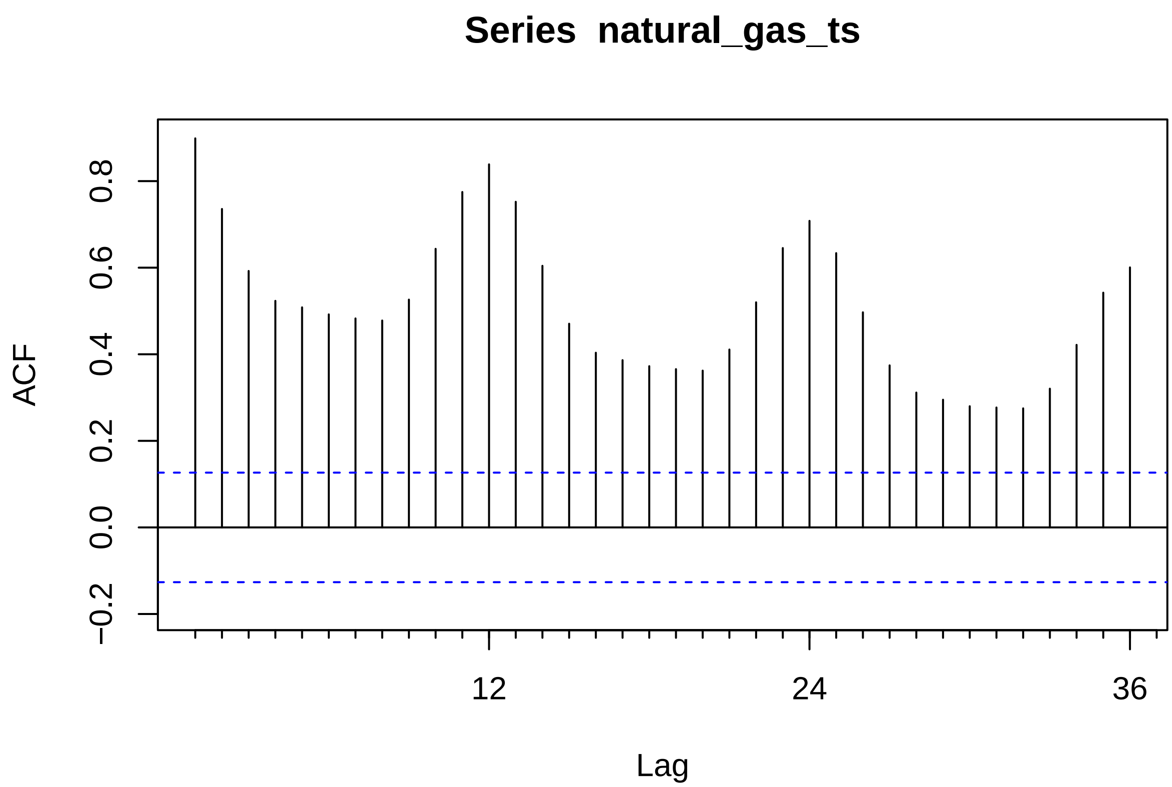
```
##
##  Augmented Dickey-Fuller Test
##
## data:  natural_gas_ts
## Dickey-Fuller = -8.8795, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```r
mk_test7 <- mk.test(natural_gas_ts)
print(mk_test7)
```

```
##
##  Mann-Kendall trend test
##
## data:  natural_gas_ts
## z = -15.007, n = 240, p-value < 2.2e-16
## alternative hypothesis: true S is not equal to 0
## sample estimates:
##             S          varS           tau
## -1.865800e+04  1.545533e+06 -6.505579e-01
```

> ADF test: Null for this test is that the time series of natural gas generation has a unit root (it's non-stationary). Alternative: The time series is stationary. The p-value for this test turns out to be 0.01 (<0.05), which means we can reject the null and state that the time series of natural gas generation could be stationary. Mann-Kendall trend test: Null: no trend in time series. Alternative: there is a trend in time series. p-value in test is much less than 0.05, so we reject the null and state that there is a significant trend in the time series. The negative value of S also indicates that the trend is slowly decreasing, which matches the previous results of the ts plots.

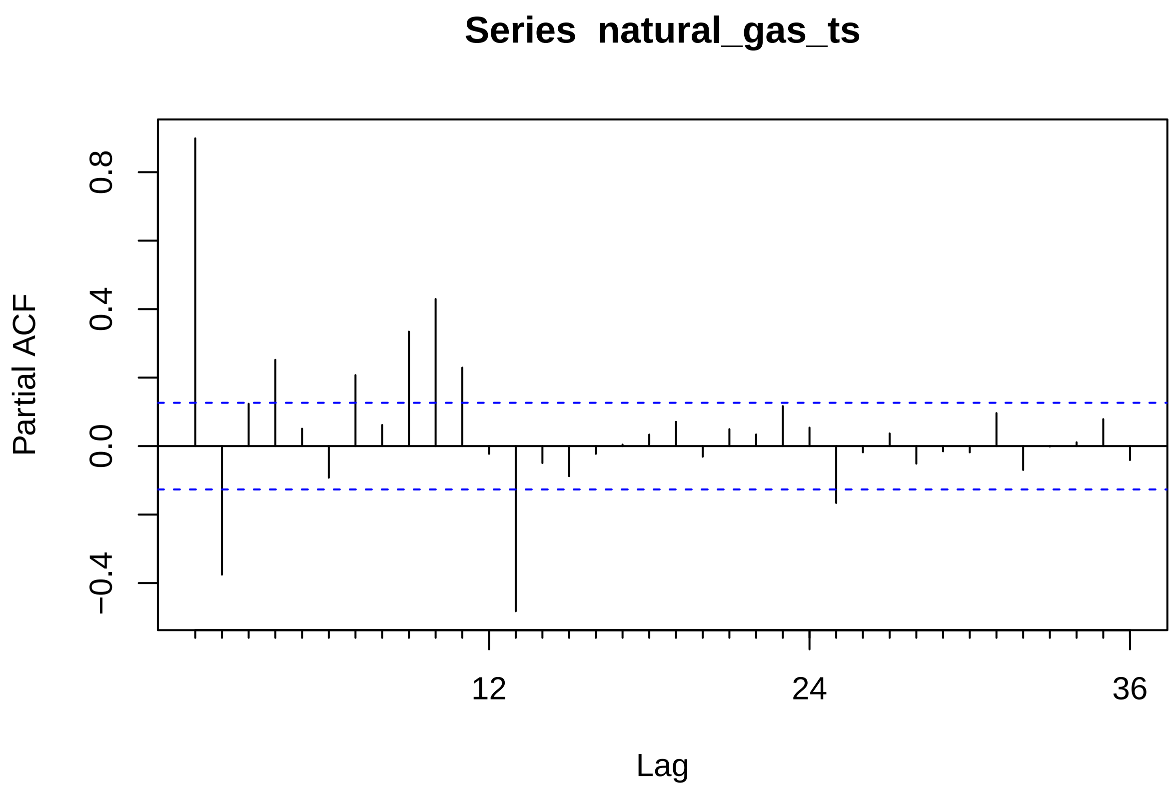
```r
# updated ACF Plot
acf_plot7 <- Acf(natural_gas_ts, lag = 36, plot = TRUE)
```




```r
print(acf_plot7)
```

```
##
## Autocorrelations of series 'natural_gas_ts', by lag
##
##     0     1     2     3     4     5     6     7     8     9    10    11    12
## 1.000 0.899 0.736 0.592 0.523 0.508 0.492 0.483 0.478 0.526 0.644 0.775 0.839
##    13    14    15    16    17    18    19    20    21    22    23    24    25
## 0.752 0.604 0.471 0.404 0.386 0.373 0.366 0.362 0.411 0.520 0.645 0.708 0.634
##    26    27    28    29    30    31    32    33    34    35    36
## 0.497 0.374 0.312 0.295 0.280 0.277 0.275 0.321 0.422 0.542 0.601
```

```r
# updated PACF Plot
pacf_plot7 <- Pacf(natural_gas_ts, lag = 36, plot = TRUE)
```




```
print(pacf_plot7)
```

```
##
## Partial autocorrelations of series 'natural_gas_ts', by lag
##
##      1      2      3      4      5      6      7      8      9     10     11
##  0.899 -0.375  0.124  0.252  0.051 -0.092  0.207  0.062  0.334  0.430  0.229
##     12     13     14     15     16     17     18     19     20     21     22
## -0.022 -0.482 -0.050 -0.088 -0.023  0.005  0.034  0.071 -0.031  0.050  0.034
##     23     24     25     26     27     28     29     30     31     32     33
##  0.117  0.054 -0.166 -0.018  0.037 -0.051 -0.015 -0.018  0.096 -0.070 -0.001
##     34     35     36
##  0.011  0.079 -0.041
```

From the ADF test, we know that the series is stationary, so d=0 From the PACF plot of the original series, the PACF value cuts off at lag 2, so the p = 2 From the ACF plot of the original series, the ACF has no significant cut offs/ spikes, so q=0.

```
# values from observing the previous plots
p <- 2
d <-0
q <- 0
# define seasonal and non-sesonal arima model
SARIMA_manual <- Arima(natural_gas_ts,
                       # non-seasonal
```

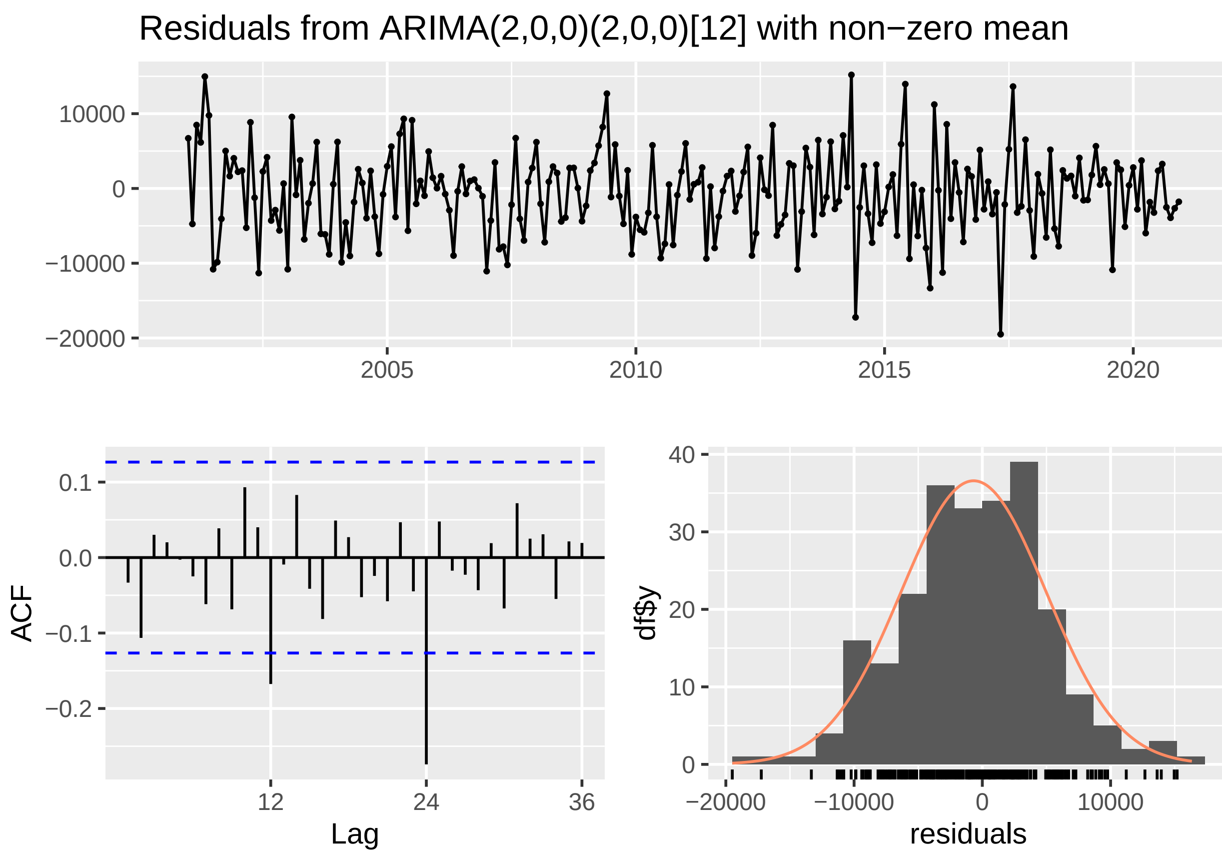
```r
                        order=c(2,0,0),
                        # seasonal
                        seasonal=c(2,0,0),
                        include.drift=FALSE)

# Print the coefficients of the fitted model
cat("ARIMA Model Coefficients:\n")
```

```
## ARIMA Model Coefficients:
```

```r
print(SARIMA_manual$coef)
```

```
##          ar1          ar2         sar1         sar2    intercept
## 7.729491e-01 9.056015e-02 5.239017e-01 4.232200e-01 8.500836e+04
```

```r
# Use checkresiduals() to generate the residual plots
checkresiduals(SARIMA_manual)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,0)(2,0,0)[12] with non-zero mean
## Q* = 43.614, df = 20, p-value = 0.001694
##
## Model df: 4.   Total lags used: 24
```

The original time series of the natural gas generation do display some white noise. The residuals plot show some randomness overtime, as it fluctuates around 0 with no siginificant patterns or trends. ACF plot shows significant values at lag 12 and 24, which displays some seasonality, so not all autocorrelations lie within the confidence intervals, so the autocorrelation doesn't necessarily indicate white noise. Accoding to the histogram of residuals, it shows a normal distribution around mean of zero, which indicates white noise.

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. > The residuals visuals for Q7 and Q6 are similar, and I think the seasonal series ARIMA model can be slightly better for representing the Natural Gas generation. Because the original time series, the original ACF, as well as the PACF of the residuals of the original natural gas time series displays some degree of seasonality, so it might be reasonable to consider the seasonal components for representing the ARMA of the original natural gas series. Yet, it's not fair to compare the deasonalized and the seasonal ARIMA model.Seasonal models are more complex because they include additional parameters to model seasonality. Deseasonalized models are simpler but rely on the assumption that the seasonal component has been fully removed.

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 ha ving 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?

```r
# deseasoned arima
auto.arima(deseasonalized_ts)
```

```
## Series: deseasonalized_ts
## ARIMA(3,1,0)(1,0,1)[12] with drift
##
## Coefficients:
##           ar1      ar2      ar3    sar1     sma1      drift
##       -0.2028  -0.1851  -0.1378  0.6609  -0.4698  -331.8138
## s.e.   0.0645   0.0655   0.0682  0.1918   0.2120   328.8248
##
## sigma^2 = 27791547:  log likelihood = -2384.89
## AIC=4783.79   AICc=4784.27   BIC=4808.12
```

The order of the deseasonlized series for the non-seasonal part is p=3, d=1, q=0, which didn't match my results/ interpretation of the plots unfortunately.

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?

```r
# original series
auto.arima(natural_gas_ts)
```

```
## Series: natural_gas_ts
## ARIMA(2,0,1)(2,1,2)[12] with drift
##
## Coefficients:
##          ar1      ar2      ma1     sar1     sar2     sma1    sma2      drift
##       1.1650  -0.2834  -0.4837  -0.0667  -0.0785  -0.6371  0.0072  -357.8435
## s.e.  0.4164   0.3180   0.3993   1.3222   0.1014   1.3215  0.9215    44.1285
##
## sigma^2 = 27958166:  log likelihood = -2278.46
## AIC=4574.91   AICc=4575.74   BIC=4605.77
```

The order of the original series for the non-seasonal part is p=2, d=0, q=1, and for the seasonal component was P=2, D=1, Q=2, which didn't match my results/ interpretation of the plots unfortunately.