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

Assignment 7 - Due date 03/06/25

Chloe Young

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

```
library(lubridate)
```

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

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

```
library(readr)
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

## -- 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 errors
```

```r
library(cowplot)
```

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

```r
library(sarima)
```

```
## Loading required package: stats4
##
## Attaching package: 'sarima'
##
## The following object is masked from 'package:stats':
##
##     spectrum
```

```r
library(openxlsx)
library(readxl)
library(dplyr)
library(zoo)
```

```
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

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.

```
#Importing data set
generation_data <- read_csv("/home/guest/TSA_Sp25/Data/Net_generation_United_States_all_sectors_monthly
```

```
## Rows: 233 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Month
## dbl (5): All_Fuels, Coal, Natural_Gas, Nuclear, Hydroelectric
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

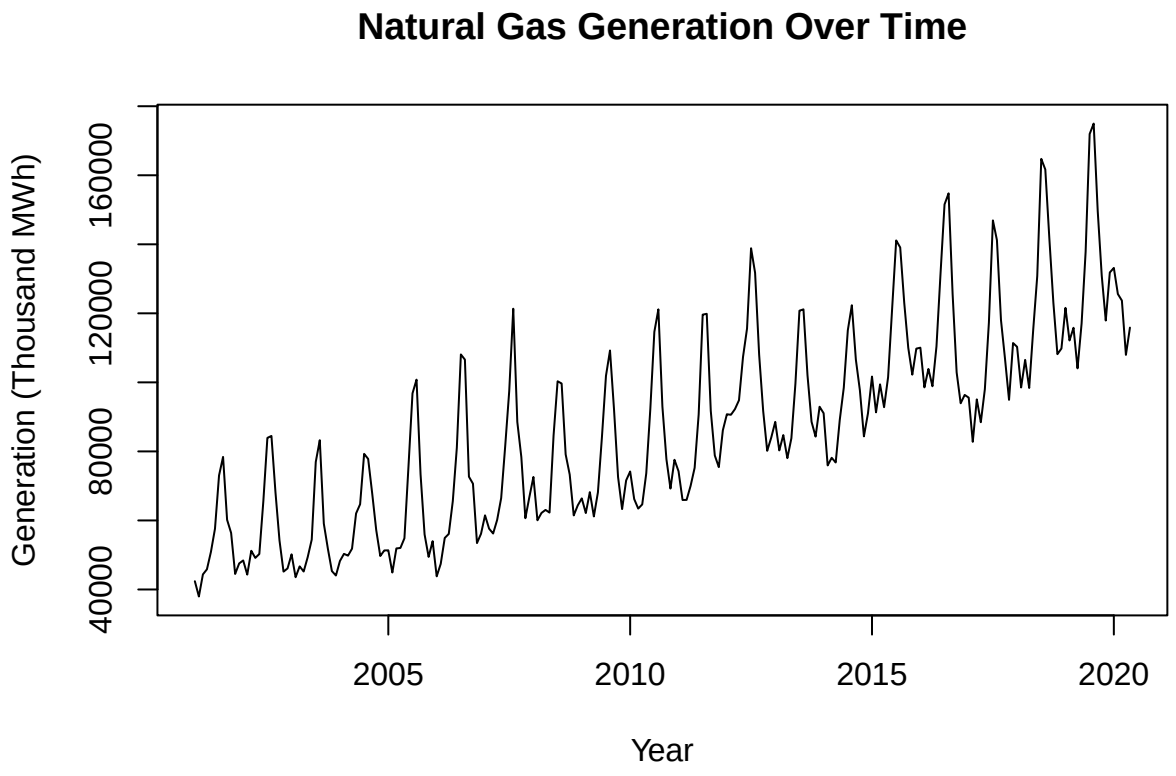
```
generation_data <- generation_data %>% filter(Month != "Month")

generation_data <- generation_data %>%
  mutate(Month = as.Date(paste("01", Month), format = "%d %b %Y"),
         Natural_Gas = as.numeric(Natural_Gas))

generation_data <- generation_data %>% arrange(Month)

#Inspect data
head(generation_data)
```

```
## # A tibble: 6 x 6
##   Month      All_Fuels    Coal Natural_Gas Nuclear Hydroelectric
##   <date>         <dbl>   <dbl>       <dbl>   <dbl>         <dbl>
## 1 2001-01-01   332493. 177287.      42389.  68707.        18852.
## 2 2001-02-01   282940. 149735.      37967.  61272.        17473.
## 3 2001-03-01   300707. 155269.      44364.  62141.        20477.
## 4 2001-04-01   278079. 140671.      45843.  56003.        18013.
## 5 2001-05-01   300492. 151593.      50934.  61512.        19176.
## 6 2001-06-01   327694. 162616.      57603.  68023.        20728.
```

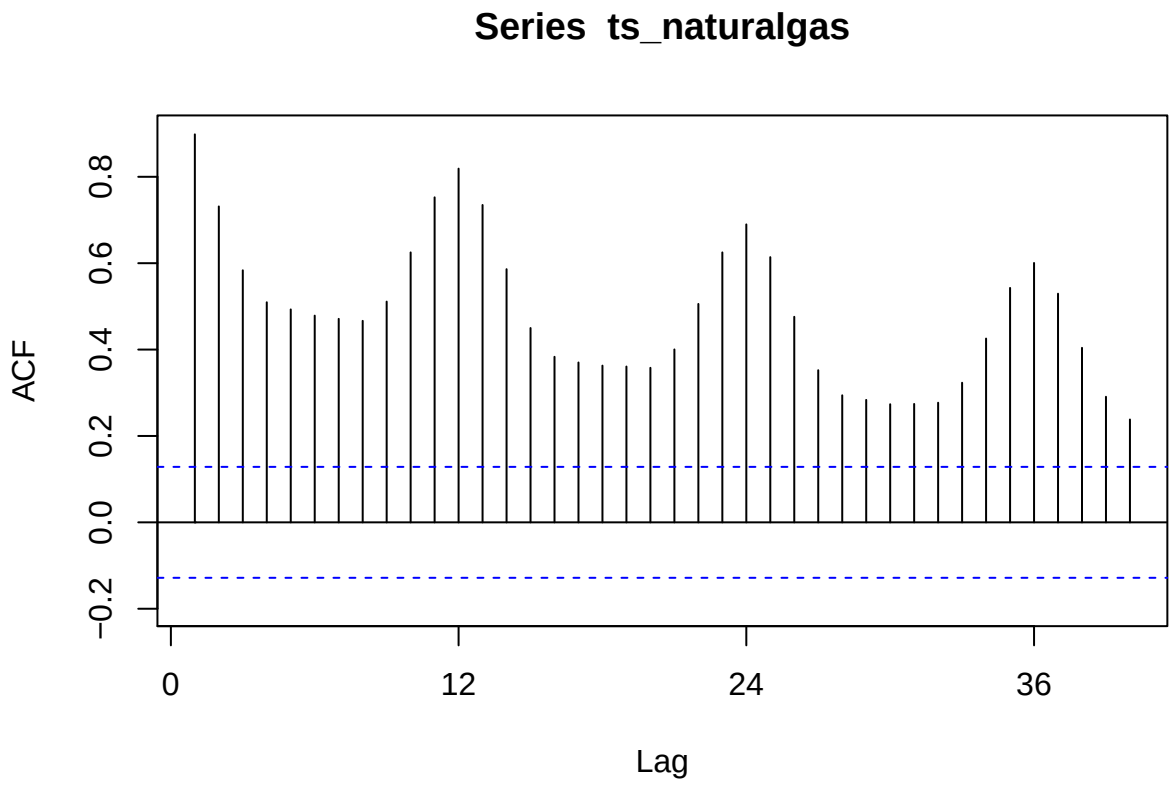
```
nvar <- ncol(generation_data) - 1
nobs <- nrow(generation_data)

head(generation_data)
```

```
## # A tibble: 6 x 6
##   Month      All_Fuels    Coal Natural_Gas Nuclear Hydroelectric
##   <date>         <dbl>   <dbl>       <dbl>   <dbl>         <dbl>
## 1 2001-01-01   332493. 177287.      42389.  68707.        18852.
## 2 2001-02-01   282940. 149735.      37967.  61272.        17473.
## 3 2001-03-01   300707. 155269.      44364.  62141.        20477.
## 4 2001-04-01   278079. 140671.      45843.  56003.        18013.
## 5 2001-05-01   300492. 151593.      50934.  61512.        19176.
## 6 2001-06-01   327694. 162616.      57603.  68023.        20728.
```

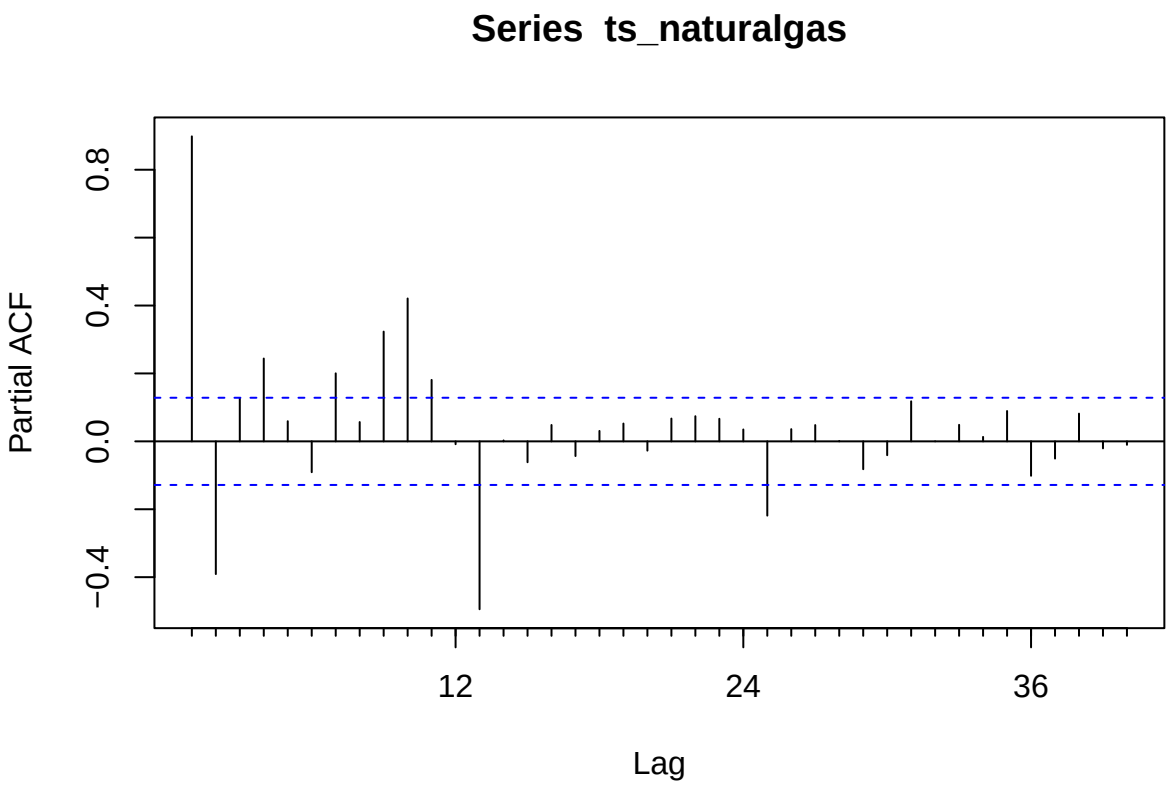
```r
ts_naturalgas <- ts(generation_data$Natural_Gas,
                           start=c(year(generation_data$Month[1]),month(generation_data$Month[1])),
                           frequency=12)

#Plots
plot(ts_naturalgas, main = "Natural Gas Generation Over Time",
     ylab = "Generation (Thousand MWh)",
     xlab = "Year")
```




```r
ACF_Plot <- Acf(ts_naturalgas, lag = 40, plot = TRUE)
```



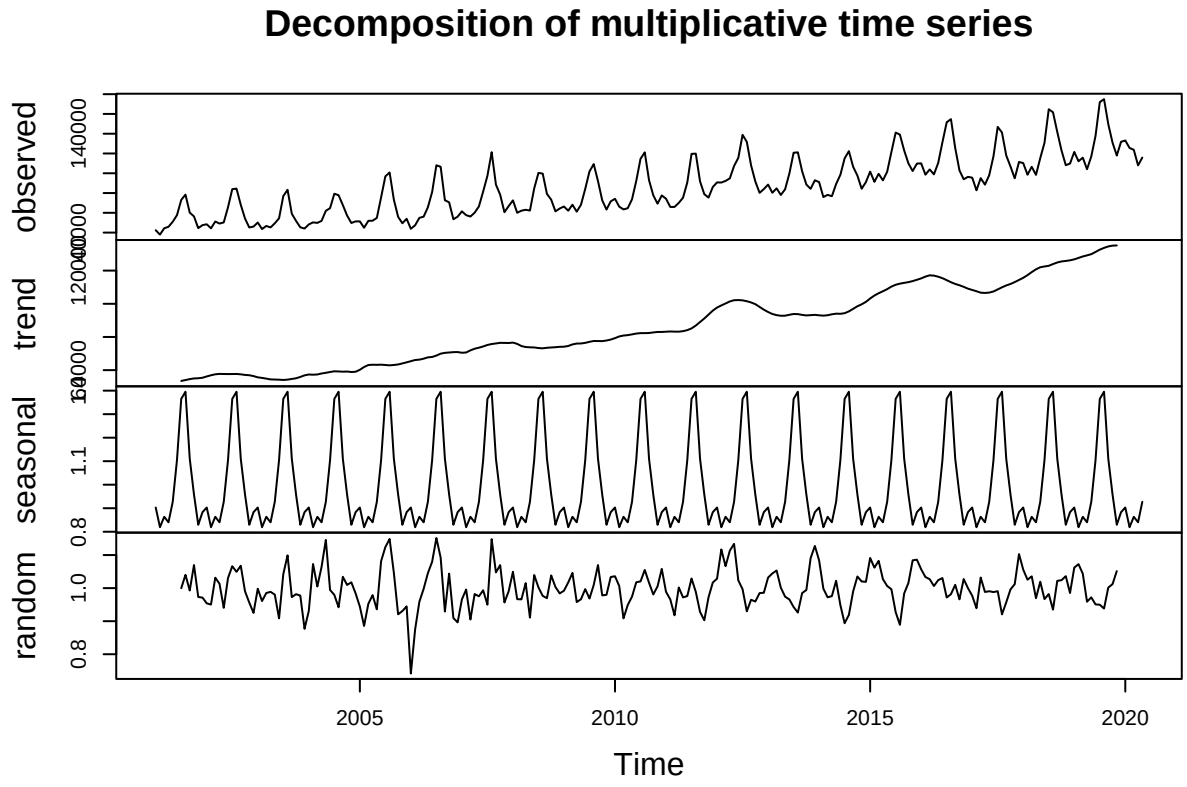

```
PACF_Plot <- Pacf(ts_naturalgas, lag = 40)
```



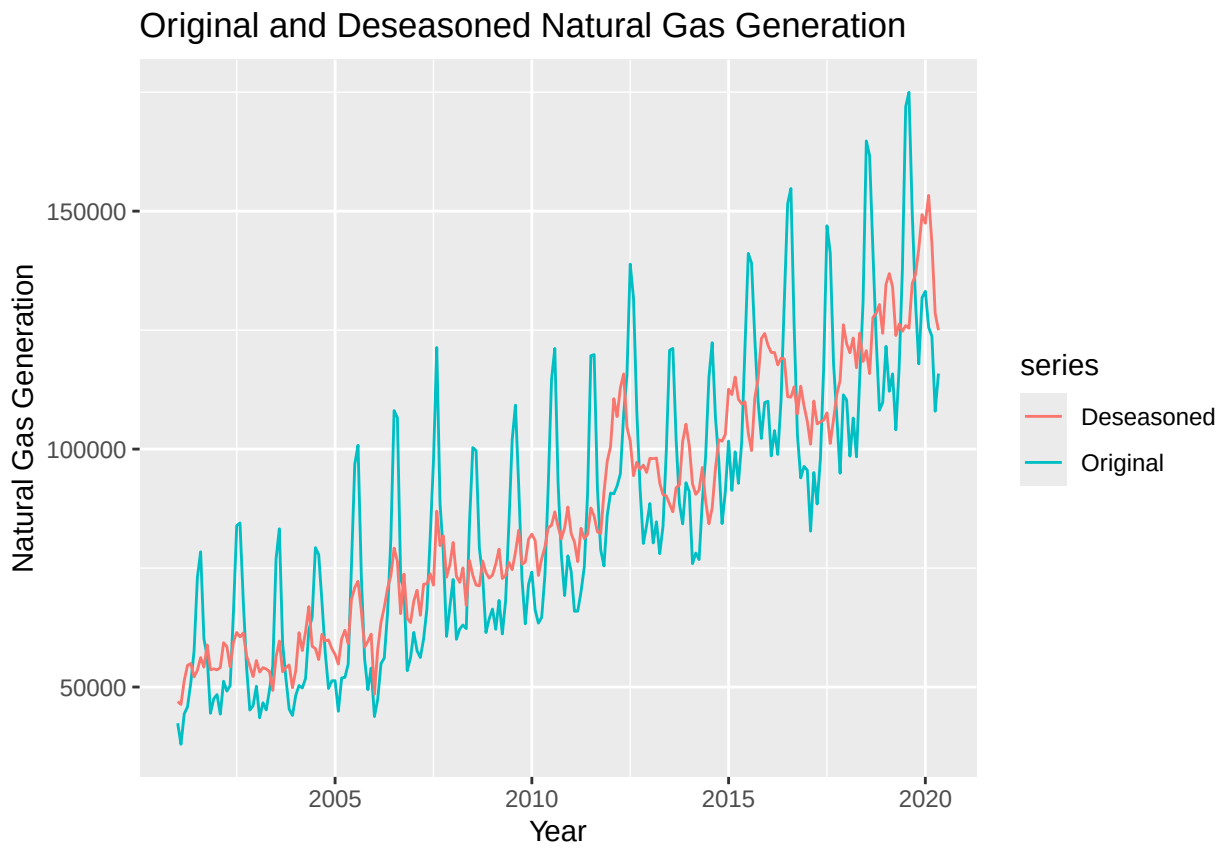
Q2

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.

```
#Decompose
decompose_naturalgas <- decompose(ts_naturalgas, "multiplicative")
plot(decompose_naturalgas)
```

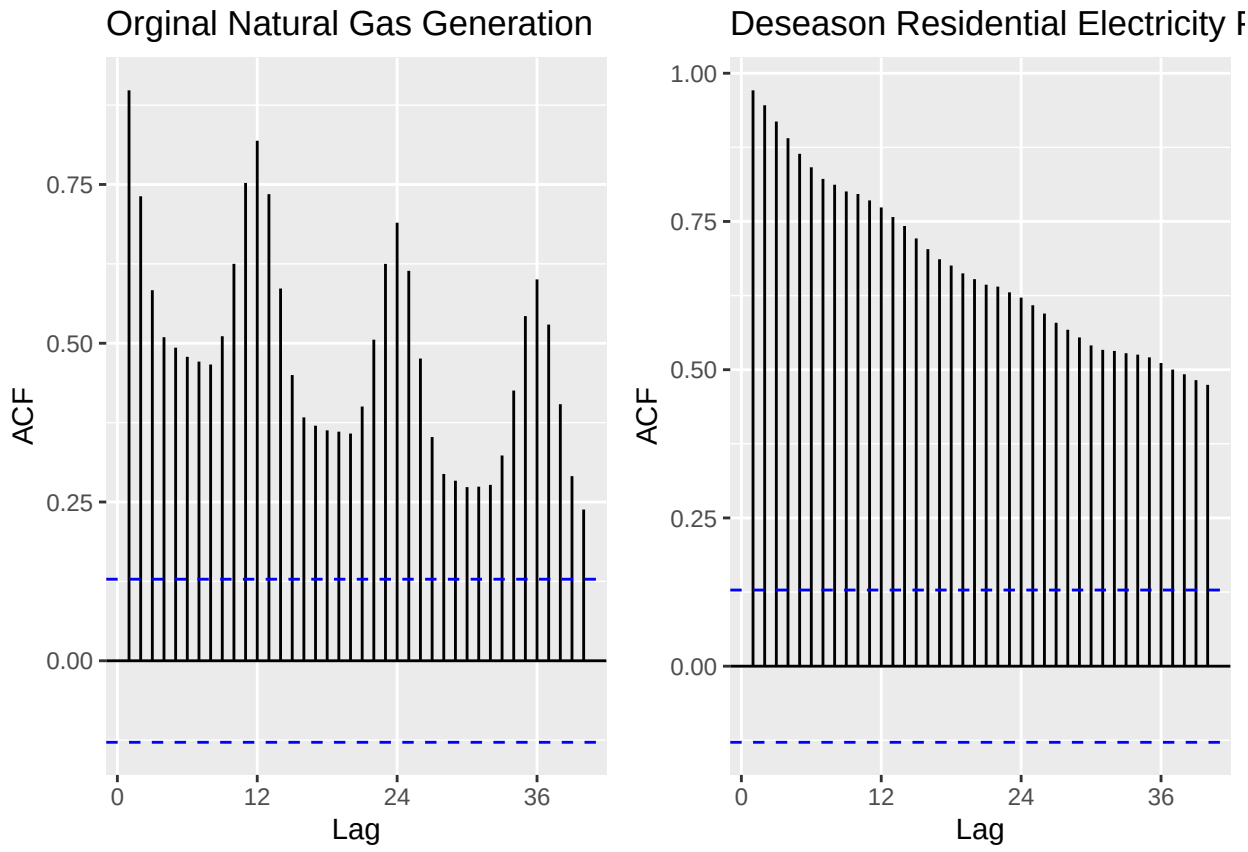



```
#Creating non-seasonal time series
deseasonal_naturalgas <- seasadj(decompose_naturalgas)

#Plotting original, deseasoned and differenced series
autoplot(ts_naturalgas, series="Original") +
  autolayer(deseasonal_naturalgas, series="Deseasoned") +
  xlab("Year") + ylab("Natural Gas Generation") +
  ggtitle("Original and Deseasoned Natural Gas Generation")
```



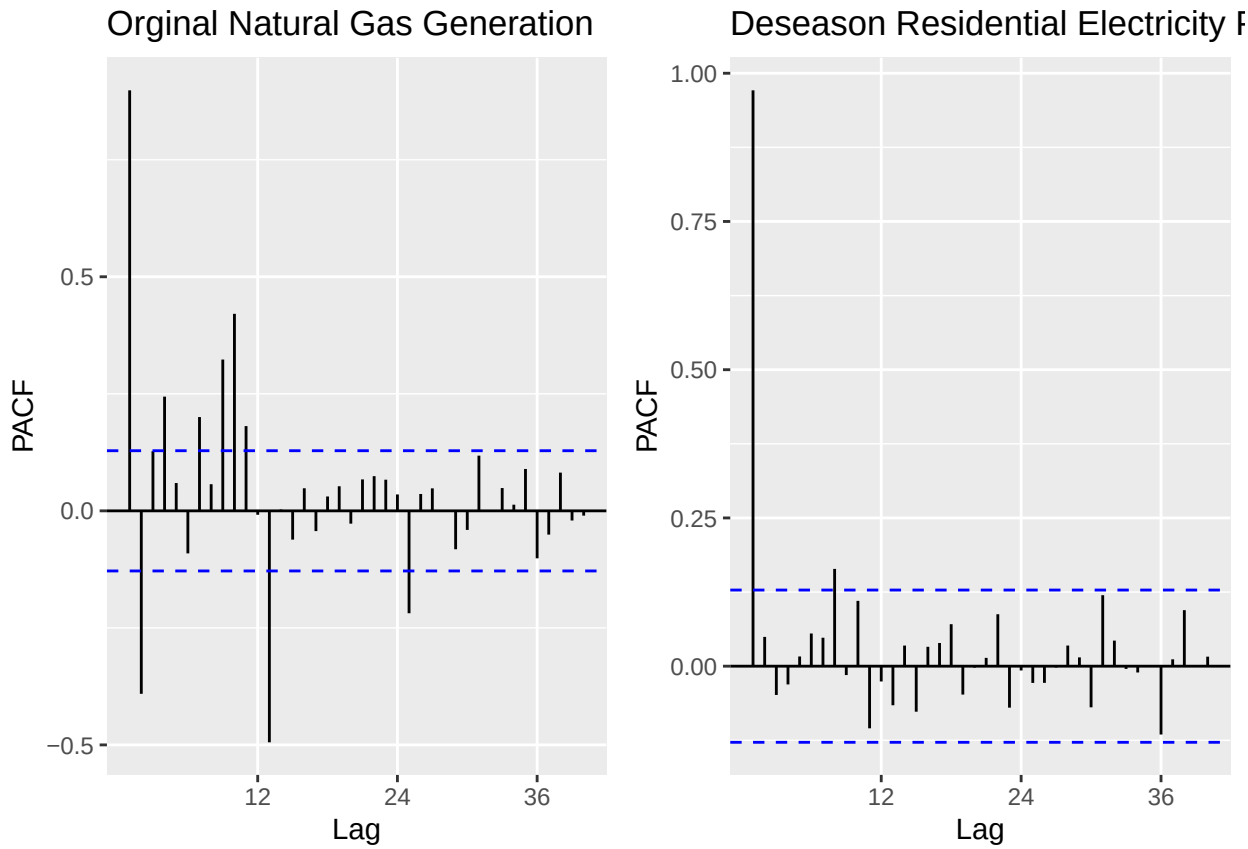

```
#Check autocorrelation plot again
#Comparing ACFs
plot_grid(
  autoplot(Acf(ts_naturalgas, lag = 40, plot=FALSE),
                main = "Orginal Natural Gas Generation"),
  autoplot(Acf(deseasonal_naturalgas, lag = 40, plot=FALSE),
                  main = "Deseason Residential Electricity Price"),
  nrow=1
)
```

```
## Warning in ggplot2::geom_segment(lineend = "butt", ...): Ignoring unknown parameters: `main`
## Ignoring unknown parameters: `main`
```




```
#Comparing PACFs
plot_grid(
  autoplot(Pacf(ts_naturalgas, lag = 40, plot=FALSE),
                main = "Orginal Natural Gas Generation"),
  autoplot(Pacf(deseasonal_naturalgas, lag = 40, plot=FALSE),
                 main = "Deseason Residential Electricity Price"),
  nrow=1
)
```

```
## Warning in ggplot2::geom_segment(lineend = "butt", ...): Ignoring unknown parameters: `main`
## Ignoring unknown parameters: `main`
```



The original series in Q1 has a lot larger peaks and troughs than the deseasoned series. The deseasoned series follows the same upward trend without the seasonal trend and fluctuations. The original ACF plot shows spikes at lags 12, 24, and 36, representing the seasonality, while the ACF for the deseasoned data shows a gradual decline as the seasonality has been removed. A similar trend can be observed in the PACF where in the original data there are seasonal spikes at lags 12, 24, and 36 that aren't seen in the deseasoned 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.

```
#ADF test
print("Results for ADF test")
```

```
## [1] "Results for ADF test"
```

```
print(adf.test(deseasonal_naturalgas,alternative = "stationary")) #stationary over stochastic trend
```

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

```
##
##  Augmented Dickey-Fuller Test
##
## data:  deseasonal_naturalgas
## Dickey-Fuller = -4.5195, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```
#Mann Kendall
print("Results for Mann Kendall Test")
```

```
## [1] "Results for Mann Kendall Test"
```

```
summary(MannKendall(deseasonal_naturalgas))
```

```
## Score =  22610 , Var(Score) = 1414465
## denominator =  27028
## tau = 0.837, 2-sided pvalue =< 2.22e-16
```

For the ADF test, the p-value of 0.01 is less than 0.05, so the null hypothesis is rejected and the series is stationary and doesn't need to be differenced.

For the Mann Kendall test, the p-value is 2.22e-16 which is less than 0.05, so the null hypothesis is rejected and it can be concluded that there is statistically significant trend. Tau is 0.837 which indicates a strong increasing trend since the value is positive and relatively close to 1. The large score of 22,610 also indicates that there is a strong increasing trend.

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.

The ADF test indicated that the data is stationary so no more differencing needs to occur. This means that d = 0.

The model is an AR model since the ACF of the deseasoned data decays exponentially. There is only one significant lag in the PACF, which indicates that the order is 1. Therefore, p = 1, d = 0, and q = 0. Therefore, the model is ARIMA (1,0,0).

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.

```
#Fitting the model
Model_100 <- Arima(deseasonal_naturalgas,order=c(1,0,0),include.drift=TRUE)
print(Model_100)
```

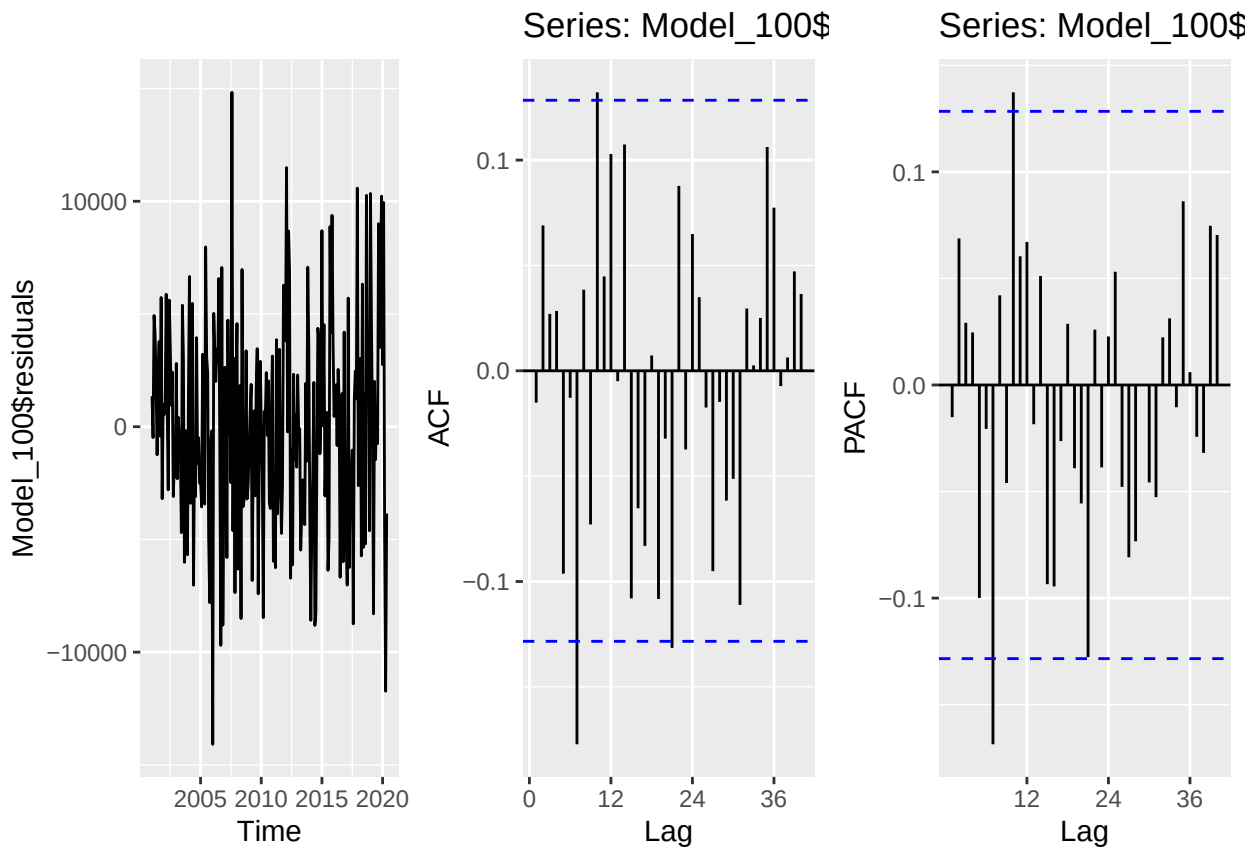
```
## Series: deseasonal_naturalgas
## ARIMA(1,0,0) with drift
##
## Coefficients:
##          ar1  intercept     drift
##       0.7793  44421.191  361.8149
## s.e.  0.0405   2691.915   19.7989
##
## sigma^2 = 22290500:  log likelihood = -2300.71
## AIC=4609.42   AICc=4609.6   BIC=4623.23
```

```
#Let's store aic for further comparison
#The lowest the aic the better the model
compare_aic <- data.frame(Model_100$aic)
```

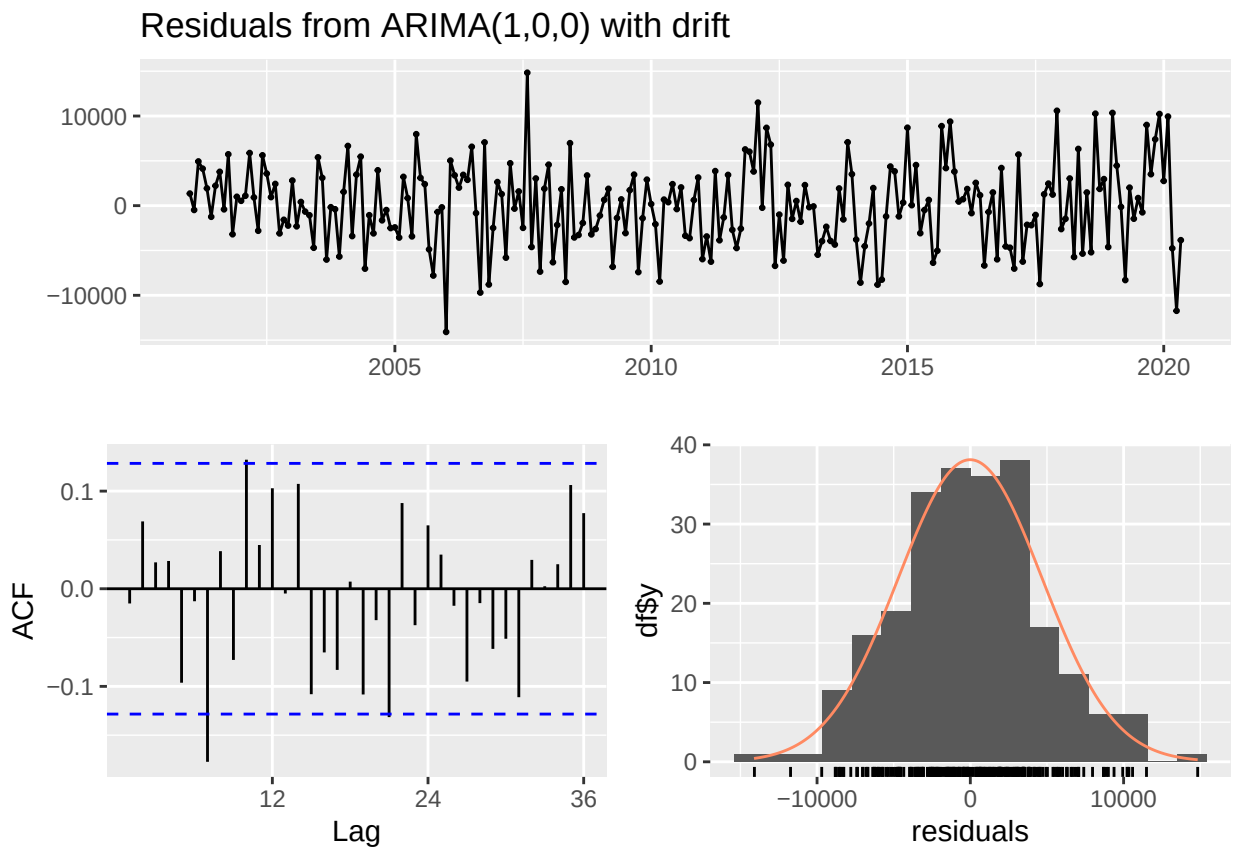
Q6

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?

```
plot_grid(
  autoplot(Model_100$residuals),
  autoplot(Acf(Model_100$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_100$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```




```
checkresiduals(Model_100)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0) with drift
## Q* = 40.358, df = 23, p-value = 0.01401
##
## Model df: 1.   Total lags used: 24
```

Although there are some anomalies, it looks like the trend fluctuates around 0 for the residual plot, indicating that the model is a good fit. Moreover, most of the ACF values fall within the significance window which indicates that the residual series is a white noise series since the residuals are uncorrelated.

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.

```
#Q3:
#ADF test
print("Results for ADF test")
```

```
## [1] "Results for ADF test"
```

```r
print(adf.test(ts_naturalgas,alternative = "stationary")) #stationary over stochastic trend
```

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

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ts_naturalgas
## Dickey-Fuller = -8.9753, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```r
#Mann Kendall
print("Results for Mann Kendall Test")
```

```
## [1] "Results for Mann Kendall Test"
```

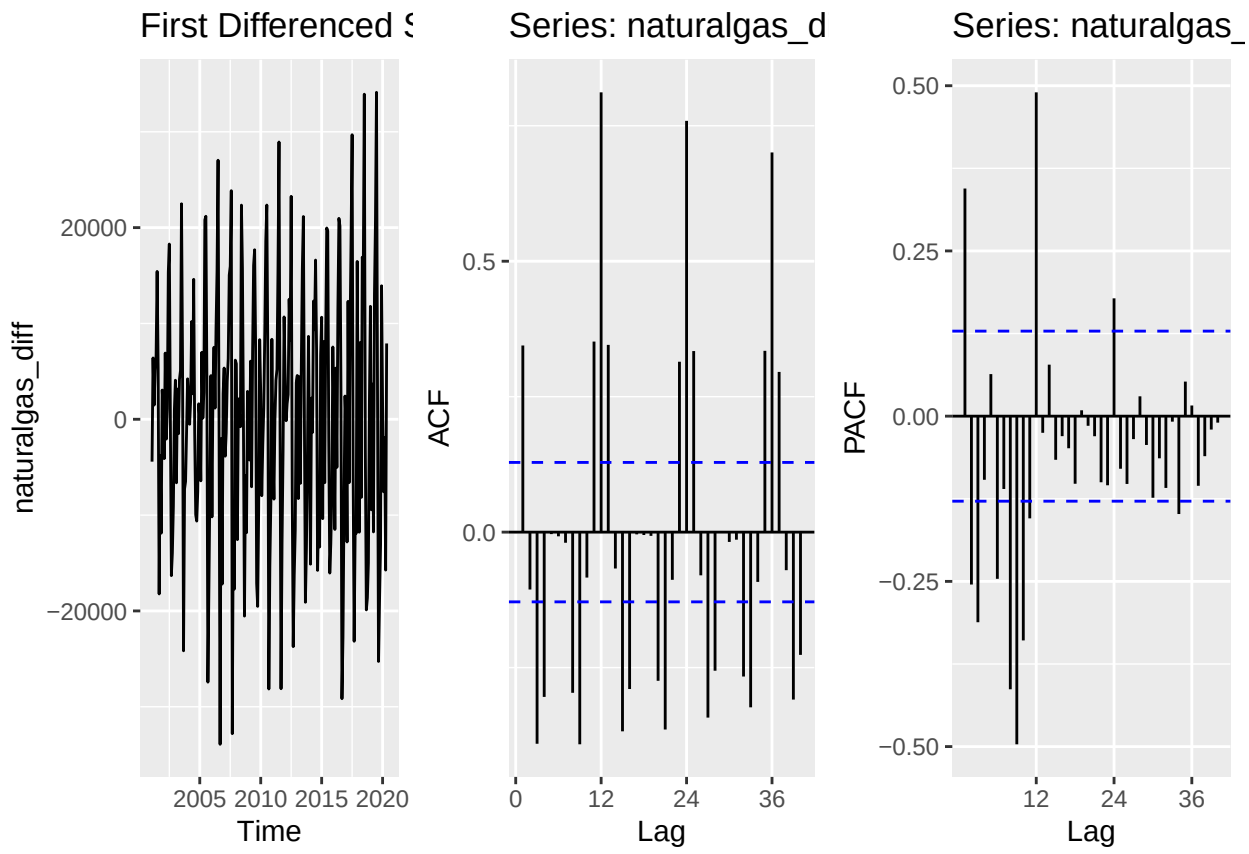
```r
summary(MannKendall(ts_naturalgas))
```

```
## Score =  17258 , Var(Score) = 1414465
## denominator =  27028
## tau = 0.639, 2-sided pvalue =< 2.22e-16
```

```r
#Q4:
#Find out how many time we need to difference
n_diff <- ndiffs(ts_naturalgas)
cat("Number of differencing needed: ",n_diff)
```

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

```r
#Differencing the Series
naturalgas_diff <- diff(ts_naturalgas,differences=1,lag=1)

plot_grid(
  autoplot(naturalgas_diff, main = "First Differenced Series"),
  autoplot(Acf(naturalgas_diff, lag = 40, plot=FALSE, main = "ACF of Differenced Series")),
  autoplot(Pacf(naturalgas_diff, lag = 40, plot=FALSE, main = "PACF of Differenced Series")),
  nrow=1
  )
```

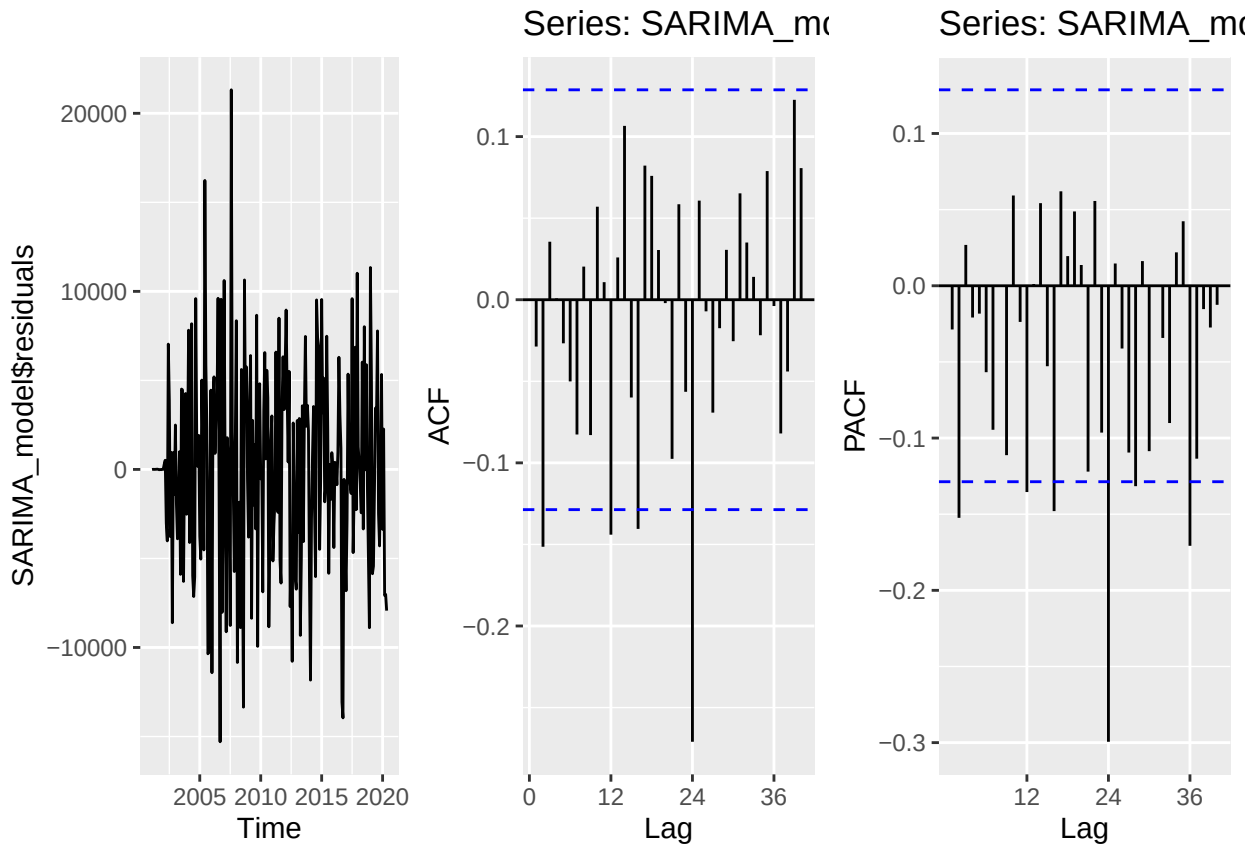



```
#Q5:

#Fitting the model
SARIMA_model <- Arima(naturalgas_diff,
                      order=c(1,0,0),
                      seasonal=c(1,1,0),
                      include.drift=TRUE)
print(SARIMA_model)
```

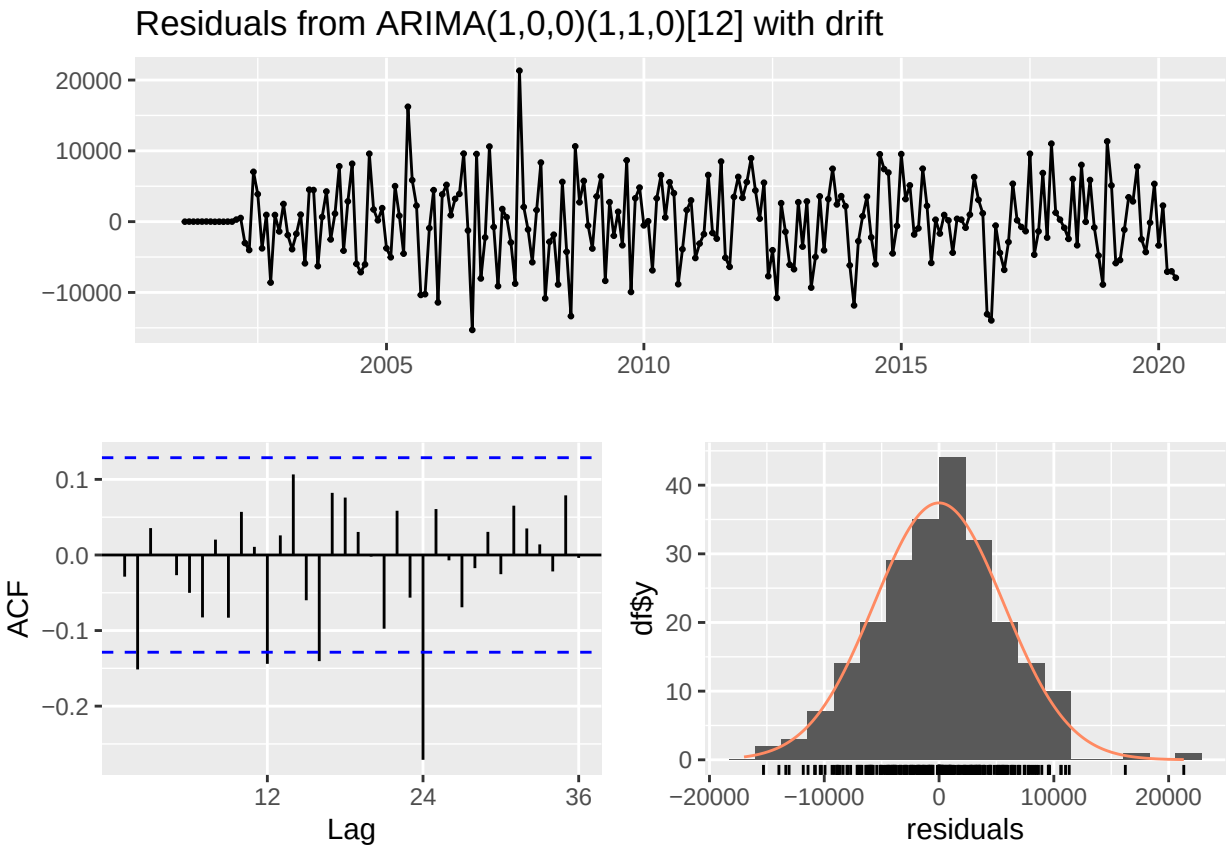
```
## Series: naturalgas_diff
## ARIMA(1,0,0)(1,1,0)[12] with drift
##
## Coefficients:
##           ar1     sar1    drift
##       -0.1767  -0.4772  -1.6043
## s.e.   0.0664   0.0584  19.1015
##
## sigma^2 = 34112052:  log likelihood = -2220.19
## AIC=4448.38   AICc=4448.57   BIC=4461.95
```

```
#Let's store aic for further comparison
#The lowest the aic the better the model
compare_aic <- data.frame(SARIMA_model$aic)

#Q6:
```

```
plot_grid(
  autoplot(SARIMA_model$residuals),
  autoplot(Acf(SARIMA_model$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(SARIMA_model$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```




```
checkresiduals(SARIMA_model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0)(1,1,0)[12] with drift
## Q* = 51.632, df = 22, p-value = 0.0003513
##
## Model df: 2.   Total lags used: 24
```

Q3: For the ADF test, the p-value of 0.01 is less than 0.05, so the null hypothesis is rejected and the series is stationary and doesn't need to be differenced.

For the Mann Kendall test, the p-value is 2.22e-16 which is less than 0.05, so the null hypothesis is rejected and it can be concluded that there is statistically significant trend. Tau is 0.639 which indicates an increasing trend since the value is positive. The large score of 17,258 also indicates that there is an increasing trend.

Q4: The ADF test indicated that the data is stationary so no more differencing needs to occur. This means that d = 0.

The model is an AR model since the ACF of the deseasoned data decays exponentially. There is only one significant lag in the PACF, which indicates that the order is 1. Therefore, p = 1, d = 0, and q = 0.

We know from nsdiffs that D=1. On the ACF, there is a positive spike at lags 12, 24, and 36 and on the PACF there's a single positive spike on lag 12, so P = 1 and Q = 0.

Therefore, the values are Seasonal ARIMA (1,0,0)(1,1,0)

Q6: This model doesn't appear to be a great fit due to the wide fluctuations in the residual series and the anomalies outside of the significance window in the ACF.

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 AIC value for the seasonal data is lower than the AIC value for the nonseasonal data. This indicates that the seasonal model is a better fit than the nonseasonal model, which doesn't correspond to the visual appearance of the models since at first glance it seems like the deseasoned model fits better. However, even so, you still can't tell which ARIMA model is better at representing data because it isn't fair to compare deseasoned and seasonal data. Deseasoning data removes some complexity so in theory it's easier to model and doesn't paint the full picture of natural gas generation.

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

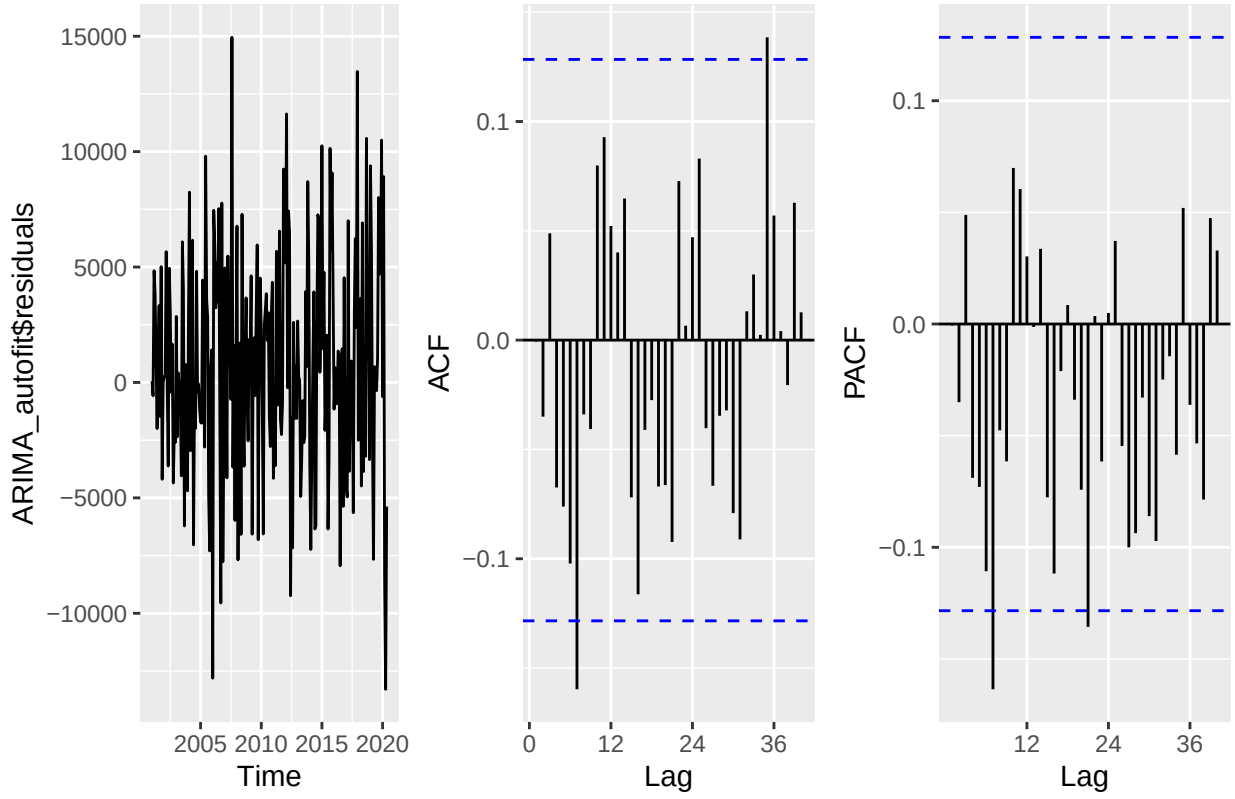
Use the *auto.arima*() command on the **deseasonalized series** to let R choose the model parameter for you. What's the order of the best ARIMA model? Does it match what you specified in Q4?

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

```
## Series: deseasonal_naturalgas
## ARIMA(2,1,2)
##
## Coefficients:
##           ar1     ar2     ma1      ma2
##       -0.2055  0.7518  0.0525  -0.8366
## s.e.   0.0884  0.0842  0.0675   0.0605
##
## sigma^2 = 23679923:  log likelihood = -2297.05
## AIC=4604.1   AICc=4604.36   BIC=4621.33
```

```
plot_grid(
  autoplot(ARIMA_autofit$residuals),
  autoplot(Acf(ARIMA_autofit$residuals,lag.max=40,plot=FALSE),main=""),
  autoplot(Pacf(ARIMA_autofit$residuals,lag.max=40,plot=FALSE),main=""),
  nrow=1,ncol=3
)
```

```
## Warning in ggplot2::geom_segment(lineend = "butt", ...): Ignoring unknown parameters: `main`
## Ignoring unknown parameters: `main`
```



The order of the best ARIMA model for the deseasoned data is (2,1,2), which is not what I specified.

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?

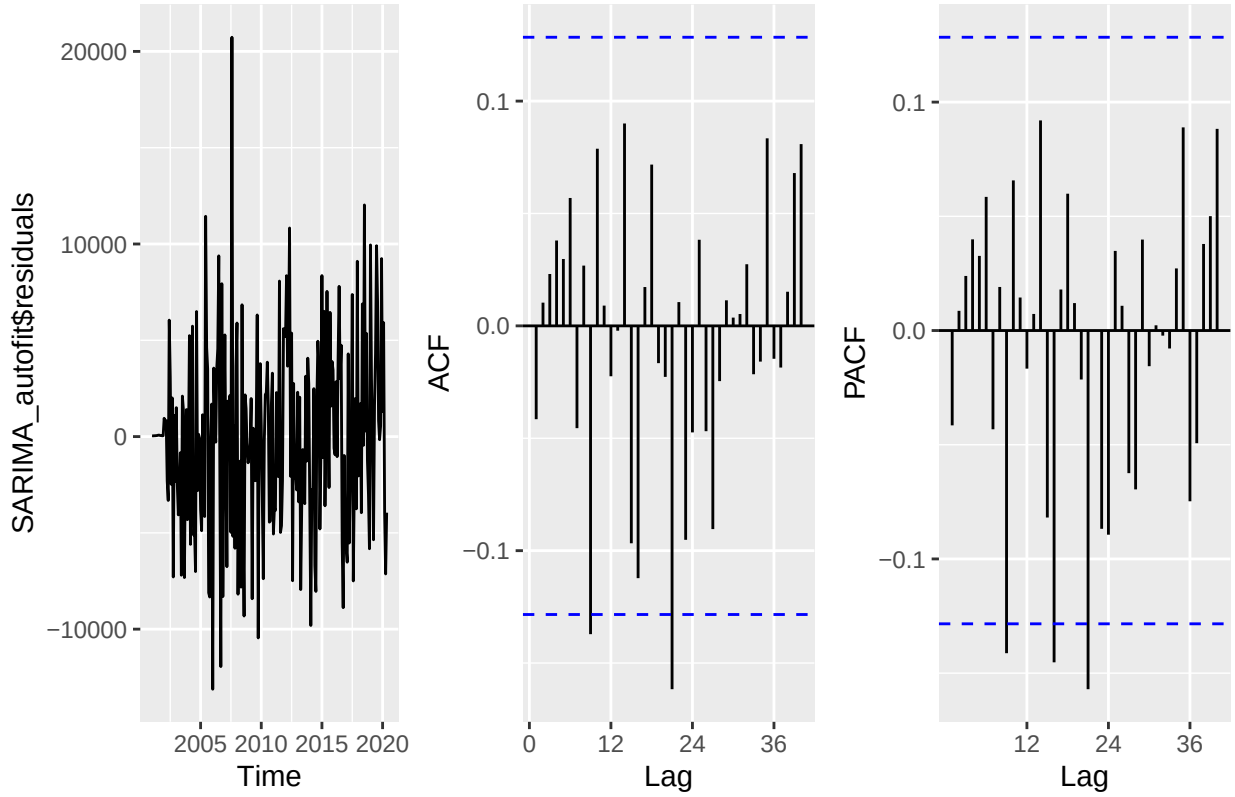
```
SARIMA_autofit <- auto.arima(ts_naturalgas)
print(SARIMA_autofit)
```

```
## Series: ts_naturalgas
## ARIMA(1,0,0)(0,1,1)[12] with drift
##
## Coefficients:
##          ar1     sma1     drift
##       0.7594  -0.7520  352.5342
## s.e.  0.0436   0.0526   34.6449
##
## sigma^2 = 25552142:  log likelihood = -2202.19
## AIC=4412.38   AICc=4412.57   BIC=4425.97
```

```
plot_grid(
  autoplot(SARIMA_autofit$residuals),
  autoplot(Acf(SARIMA_autofit$residuals,lag.max=40,plot=FALSE),main=""),
  autoplot(Pacf(SARIMA_autofit$residuals,lag.max=40,plot=FALSE),main=""),
```

```
  nrow=1,ncol=3
)
```

```
## Warning in ggplot2::geom_segment(lineend = "butt", ...): Ignoring unknown parameters: ‘main‘
## Ignoring unknown parameters: ‘main‘
```



The best order for the seasonal data is (1,0,0)(0,1,1). I was correct for p, d, q and D, but incorrect for P and Q.