# ENV 797 - Time Series Analysis for Energy and Environment Applications | Spring 2024

Assignment 7 - Due date 03/07/24

Jaimie Wargo

## 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_Sp24.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(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- 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
```

```
library(cowplot)
```

```
##
## Attaching package: 'cowplot'
```

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

```
library(here)
```

```
## here() starts at C:/Users/jaimi/OneDrive/Documents/Duke/Spring_2024/TSA_Sp24
```

```
library(forecast)
```

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

```
library(tseries)
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**.

### 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.

```
raw_data <- read_csv(here('Data',
                          'Net_generation_United_States_all_sectors_monthly.csv'),
                     skip = 4)
```

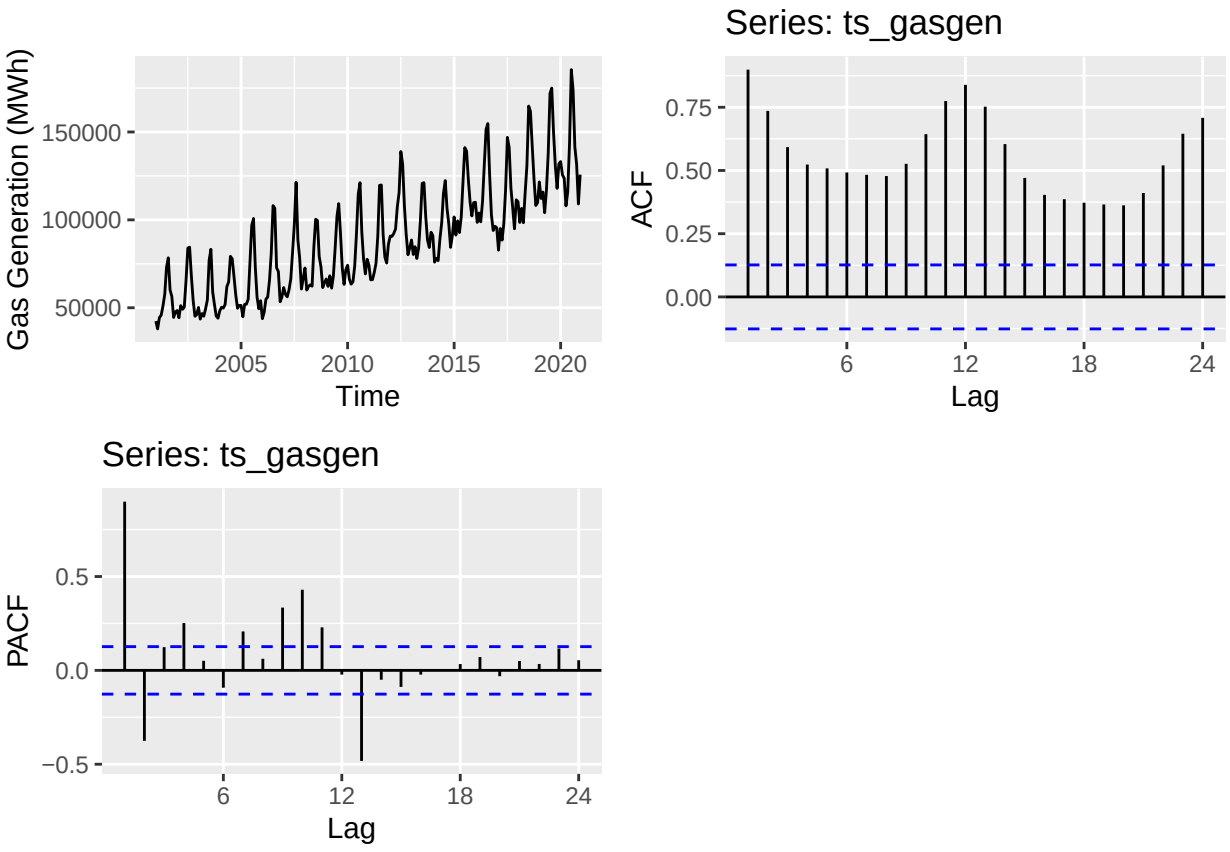
```
## Rows: 240 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Month
## dbl (5): all fuels (utility-scale) thousand megawatthours, coal thousand meg...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
raw_data$Month <- mdy(raw_data$Month)

raw_data <- raw_data[order(raw_data$Month),]

ts_gasgen <- ts(raw_data[,'natural gas thousand megawatthours'],
                start=c(2001,1), frequency = 12)
```

```
plot_grid(autoplot(ts_gasgen, y='Gas Generation (MWh)'),
          autoplot(Acf(ts_gasgen, plot=F)),
          autoplot(Pacf(ts_gasgen, plot=F)))
```

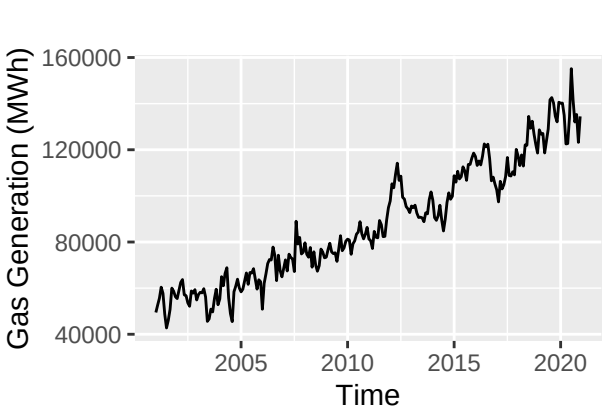


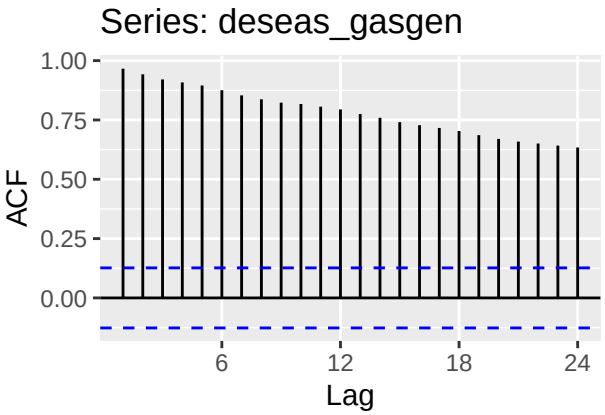

**Q2**

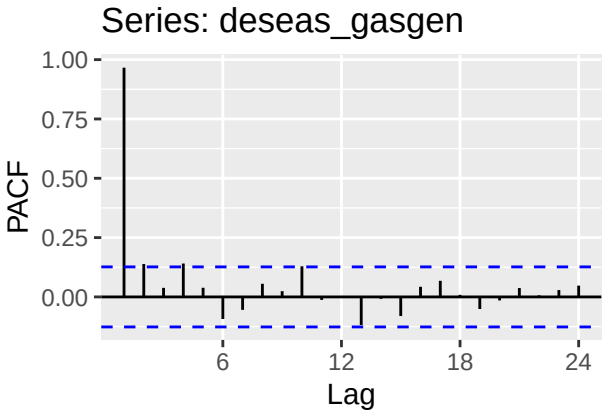
Using the *decompose*() or *stl*() and the *seasadj*() functions create a series without the seasonal component, i.e., a deseasonalized natural gas series. Plot the deseasonalized series over time and corresponding ACF and PACF. Compare with the plots obtained in Q1.

```
decompose_gasgen <- decompose(ts_gasgen)
deseas_gasgen <- seasadj(decompose_gasgen)

plot_grid(autoplot(deseas_gasgen, y='Gas Generation (MWh)'),
          autoplot(Acf(deseas_gasgen, plot=F)),
          autoplot(Pacf(deseas_gasgen, plot=F)))
```




The ACF and PACF clearly have the seasonality removed, as the ACF has the wave pattern removed. The PACF also now only has significant lags at lag 1. The general time series also has more randomness now that the seasonal component is removed.

## 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.

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

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

```
print(adf.test(deseas_gasgen,alternative = "stationary"))
```

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

##
##  Augmented Dickey-Fuller Test
##
## data:  deseas_gasgen
```

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

```
print("Results of Mann Kendall on average yearly series")
```

```
## [1] "Results of Mann Kendall on average yearly series"
```

```
print(summary(MannKendall(deseas_gasgen)))
```

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

The ADF test reports a p-value less than 0.01, meaning that we accept the alternative hypothesis that the trend does not contain a unit root. Mann Kendall also has a p-value less than 0.01, so we accept the alternative hypothesis that this series follows a deterministic trend. These conclusions are evident when looking at the time series plot in Q2.

### 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.

```
ndiffs(deseas_gasgen)
```

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

```
Model_610 <- Arima(deseas_gasgen,order=c(6,1,0),include.drift=TRUE, include.mean = TRUE)
compare_aic <- data.frame(Model_610$aic)

Model_110 <- Arima(deseas_gasgen,order=c(1,1,0),include.drift=TRUE, include.mean = TRUE)
compare_aic <- data.frame(compare_aic, Model_110$aic)

print(compare_aic)
```

```
##   Model_610.aic Model_110.aic
## 1      4784.964      4799.075
```

To do this, I am using the AIC method and trying different iterations of p, d, and q. The Mann Kendall test indicated lag order = 6, so I determined the AR component was 6. Additionally, because there is a trend present as interpreted from the MK, the series does need to be differenced. I checked to make sure d!=2 by using ndiffs, which returned d=1. The ACF chart showed an exponential decay, so I interpreted that there was not a moving average component. Visually, I would have determined a lag order of 1 for the auto-regressive order, but I tested this by comparing the AICs of model 110 and model 610 and the AIC was lower for 610.

### 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()` r `print()` function to print.

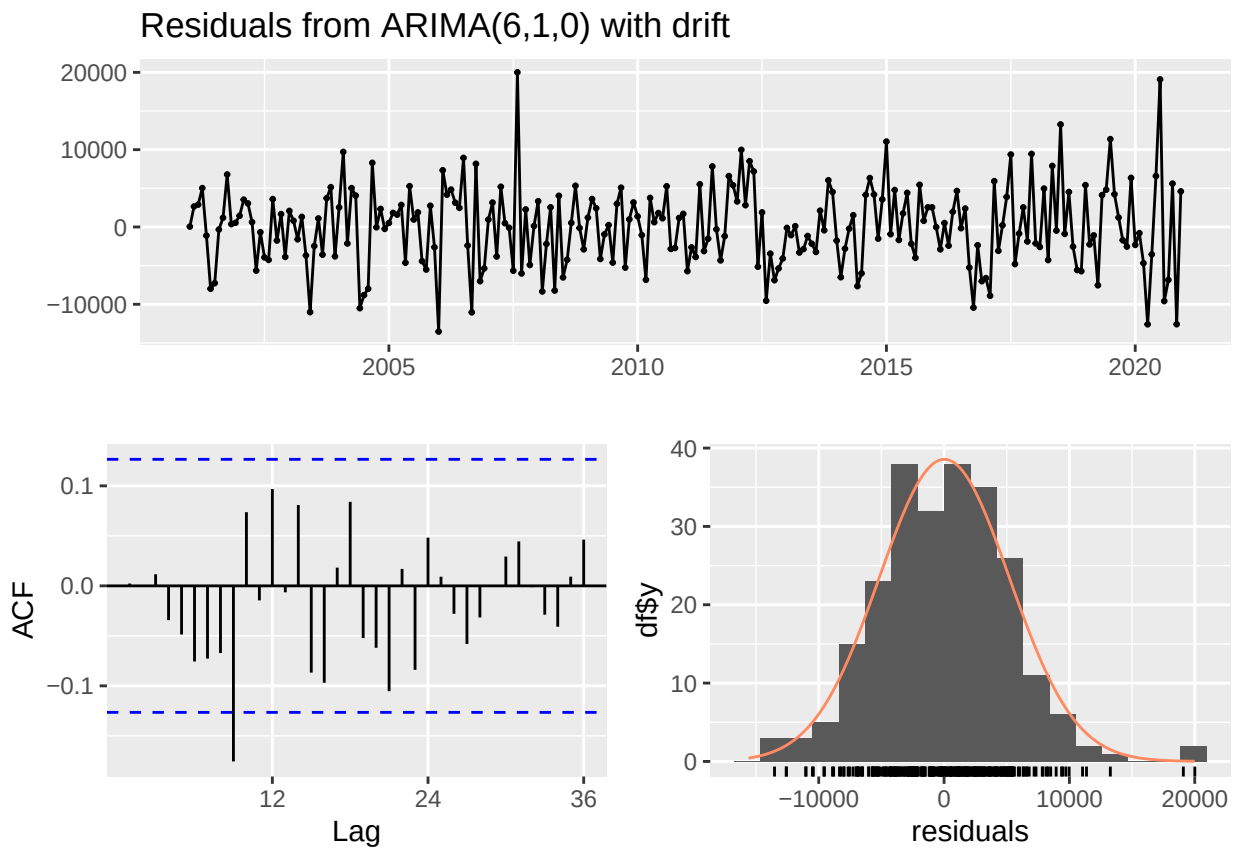
```
#Already created this model in Q4
print(Model_610$coef)
```

```
##          ar1          ar2          ar3          ar4          ar5          ar6
##  -0.22891427  -0.20215730  -0.23318930  -0.17122967  -0.07890997   0.08575309
##        drift
## 343.69139520
```

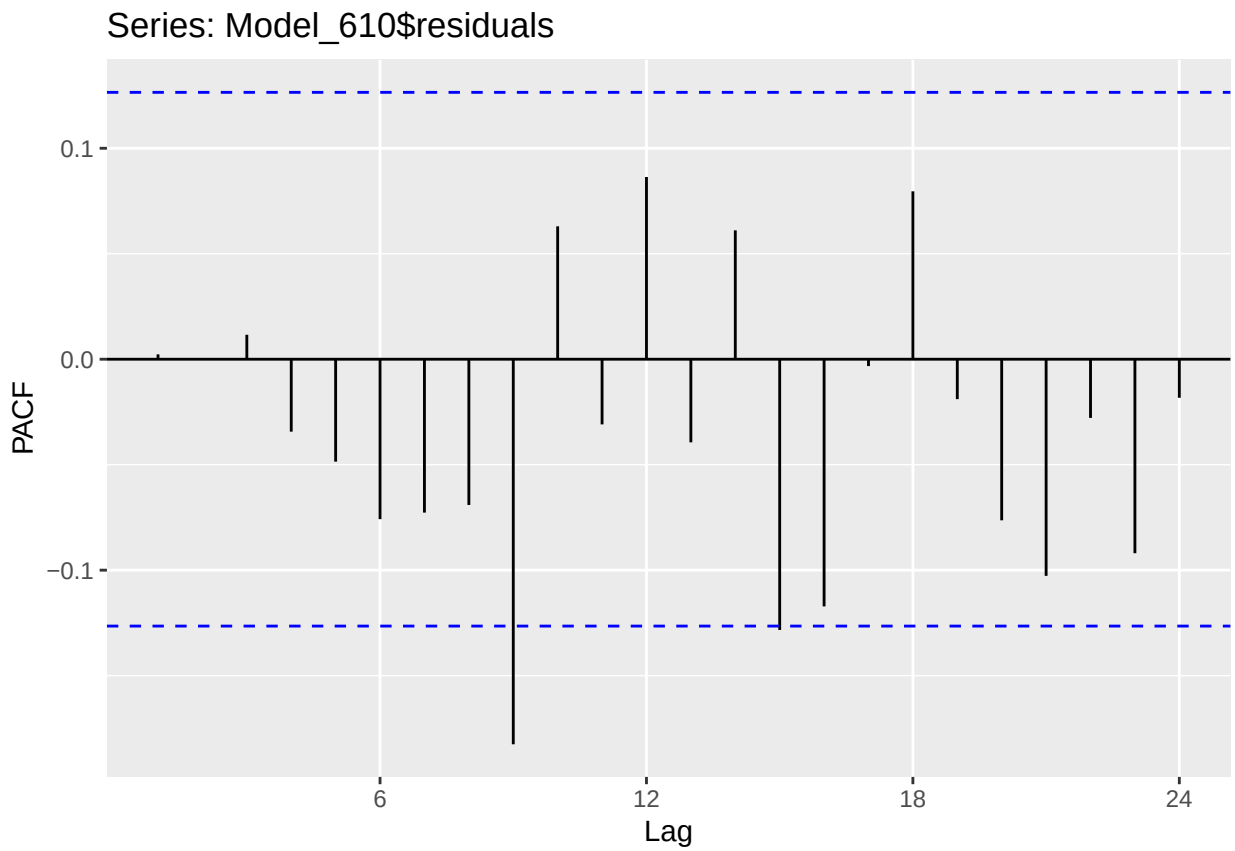
### Q6

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(Model_610, test=F)
```




```
autoplot(Pacf(Model_610$residuals, plot=F))
```




I would say these series look very much like a white noise series, as there is not a clear pattern that is being followed. This is because we removed seasonality from the series and accounted for the trend in the ARIMA.

## 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 $Q$.

```
p <- 6

ndiffs(ts_gasgen)
```

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

```
d <- 1
q <- 0

nsdiffs(ts_gasgen)
```

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

```r
D <- 1
P <- 0
Q <- 1
s <- 12
```

I kept the same values for the non-seasonal components, checking with ndiffs that non-seasonal differencing was still applicable. nsdiffs also indicated seasonal differencing was necessary for this series. From looking at the ACF and PACF in Q1, there is a clear pattern occurring every 12 lags, indicating that the s term is 12. We also see that the PACF lag spikes are negative, which suggests that we should include an SMA component. For this reason, I included Q=1 and P=0.

```r
sModel <- Arima(ts_gasgen, order=c(6,1,0), seasonal = c(0,1,1),
                include.mean = TRUE, include.drift = TRUE)
```

```
## Warning in Arima(ts_gasgen, order = c(6, 1, 0), seasonal = c(0, 1, 1),
## include.mean = TRUE, : No drift term fitted as the order of difference is 2 or
## more.
```

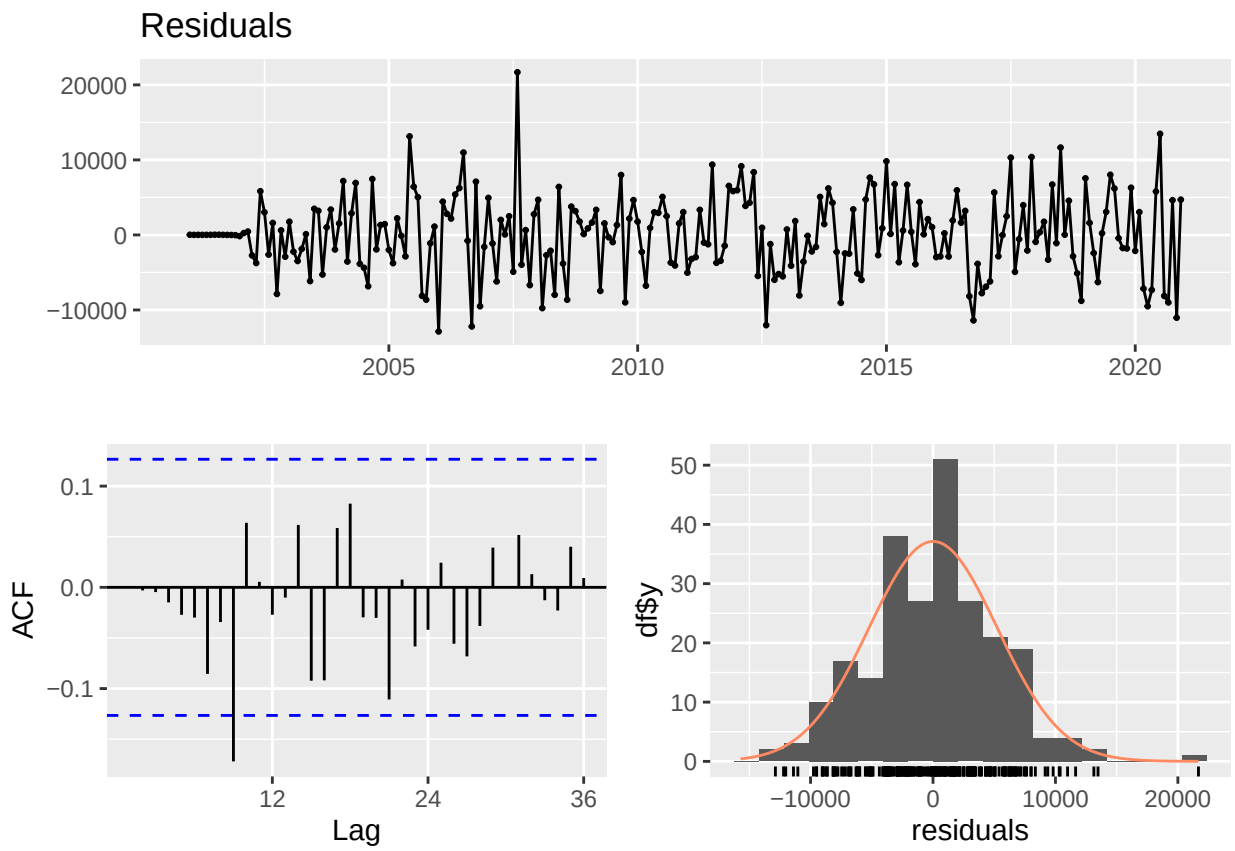
```r
print(sModel)
```

```
## Series: ts_gasgen
## ARIMA(6,1,0)(0,1,1)[12]
##
## Coefficients:
##           ar1      ar2      ar3      ar4      ar5     ar6     sma1
##       -0.2337  -0.2006  -0.1274  -0.0942  -0.0337  0.0109  -0.7142
## s.e.   0.0664   0.0686   0.0728   0.0710   0.0694  0.0706   0.0597
##
## sigma^2 = 29849875:  log likelihood = -2276.42
## AIC=4568.84   AICc=4569.51   BIC=4596.24
```

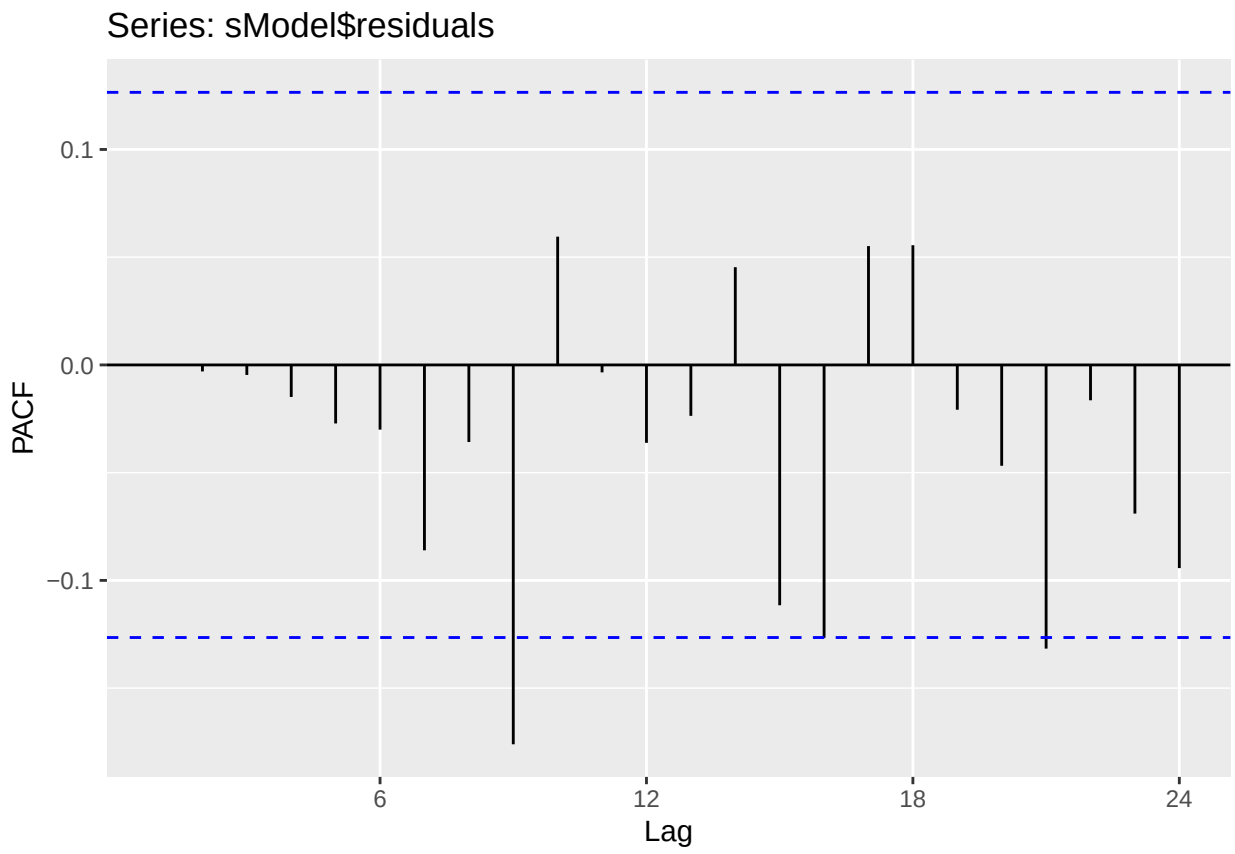
```r
print(sModel$coef)
```

```
##         ar1         ar2         ar3         ar4         ar5         ar6
## -0.23365843 -0.20064992 -0.12735653 -0.09421817 -0.03370228  0.01086204
##        sma1
## -0.71417721
```

```
checkresiduals(sModel$residuals, test=F)
```




```
autoplot(Pacf(sModel$residuals, plot=F))
```



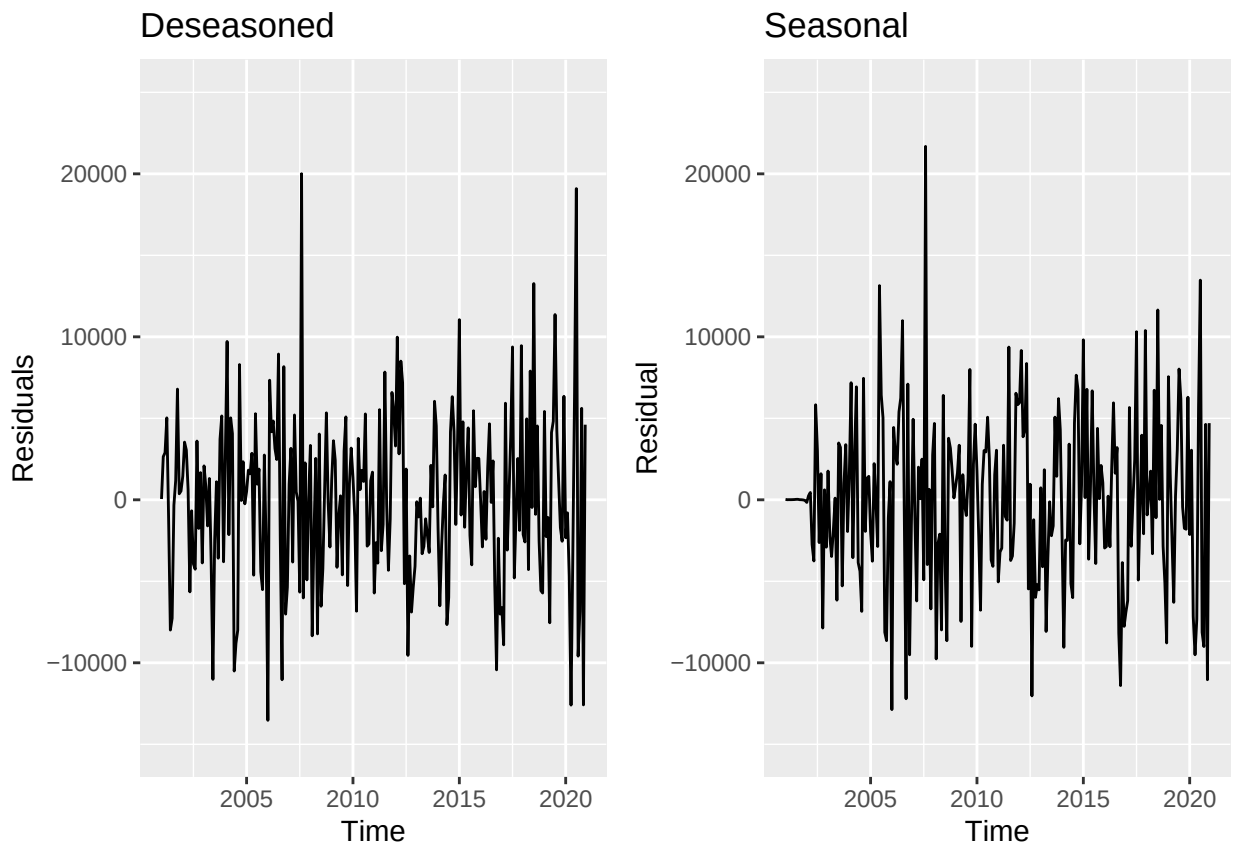

These residuals also look like a white noise series.

**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.

```
plot_grid(autoplot(Model_610$residuals, ylim=c(-15000, 25000), y="Residuals", main="Deseasoned"),
          autoplot(sModel$residuals, ylim=c(-15000, 25000), y="Residual", main="Seasonal"))
```

```
## Warning in ggplot2::geom_line(na.rm = TRUE, ...): Ignoring unknown parameters: ‘ylim‘
## Ignoring unknown parameters: ‘ylim‘
```




I can't really tell, but I feel like the seasonal model is slightly more centered around 0 overall. I'm not sure it's a fair comparison because it's harder to fit seasonal data than removing it at the top.

## 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?

```
auto.arima(deseas_gasgen)
```

```
## Series: deseas_gasgen
## 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
```

The order for the best ARIMA model is (1,1,1). It does not match what I had– this model included the MA component and ended up with 10 points lower AIC. I also used the lag order given in the MK results, but that did not appear to be the best option. I am also confused because 110 was less optimal and 610, but 111 was more optimal.

### 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?

```
auto.arima(ts_gasgen)
```

```
## Series: ts_gasgen
## 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
```

This also does not match, since I used the values from before. I don't think I understand how d and D relate and when to use BOTH or ONE when working with SARIMA. I did model the seasonal components correctly using the helpful tips in the slides.