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

Assignment 7 - Due date 03/06/25 - Extension Granted

Rachael Stephan

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(forecast); library(tseries); library(tidyverse)
library(lubridate); library(cowplot); library(ggpubr)
library(Kendall)
```

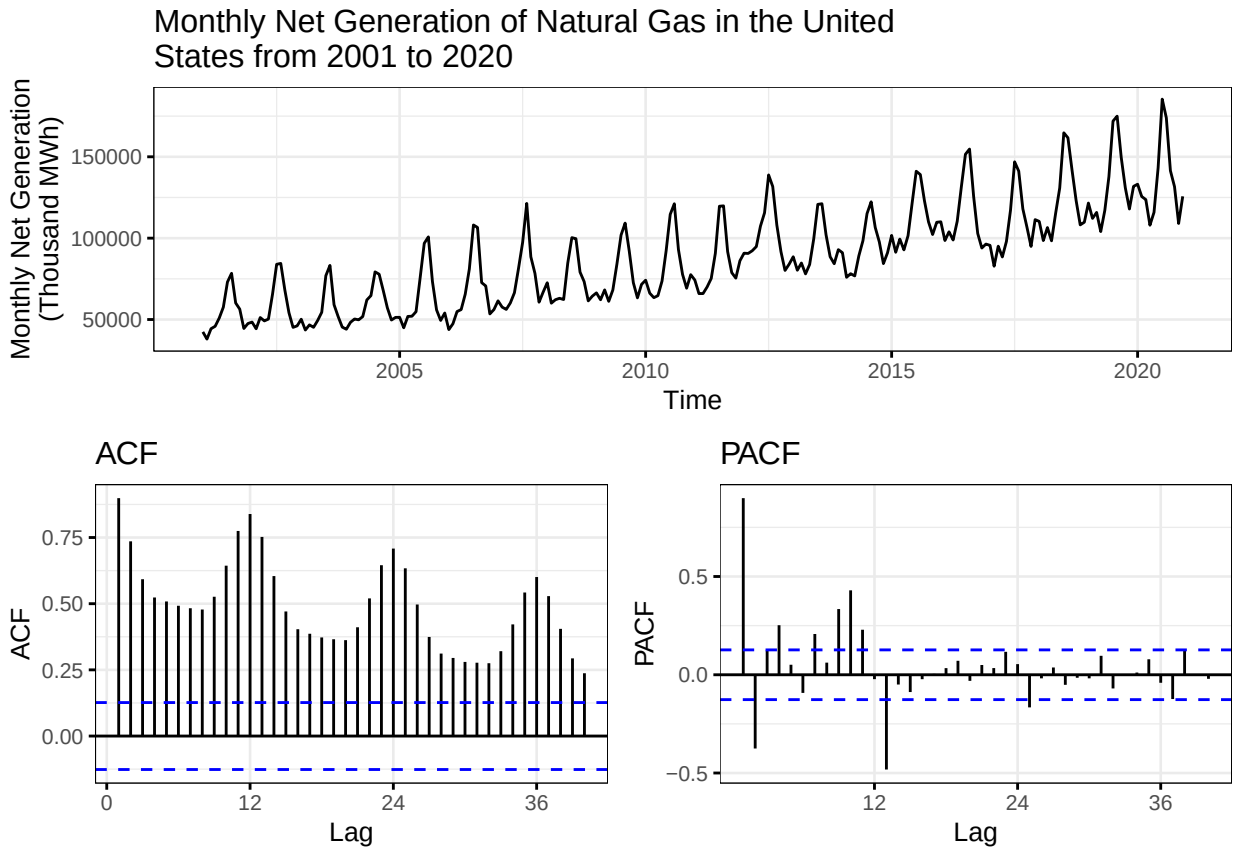
Importing and processing the data set

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.

```r
#create dataframe with the month column in date format and from oldest to newest date
energy_data <- read_csv(file = "./Data/Net_generation_United_States_all_sectors_monthly.csv",
                        skip = 4) %>%
  select(Month, `natural gas thousand megawatthours`) %>%
  mutate(Month = my(Month)) %>%
  arrange(Month)

#convert data frame to timeseries
energy_ts <- ts(energy_data[,2],
                start = c(2001,1),
                frequency = 12)

original_plot <- autoplot(energy_ts)+
  labs(y = "Monthly Net Generation\n(Thousand MWh)",
       title = "Monthly Net Generation of Natural Gas in the United\nStates from 2001 to 2020")

original_acf <- Acf(energy_ts,
                    lag.max=40,
                    type="correlation",
                    plot=FALSE) %>%
  autoplot()+
  labs(title = "ACF")

original_pacf <- Pacf(energy_ts,
                    lag.max=40,
                    type="correlation",
                    plot=FALSE) %>%
  autoplot()+
  labs(title = "PACF")

#plot the timeseries, acf, and pacf
ggarrange(original_plot,
          ggarrange(original_acf, original_pacf, ncol = 2),
          nrow = 2)
```

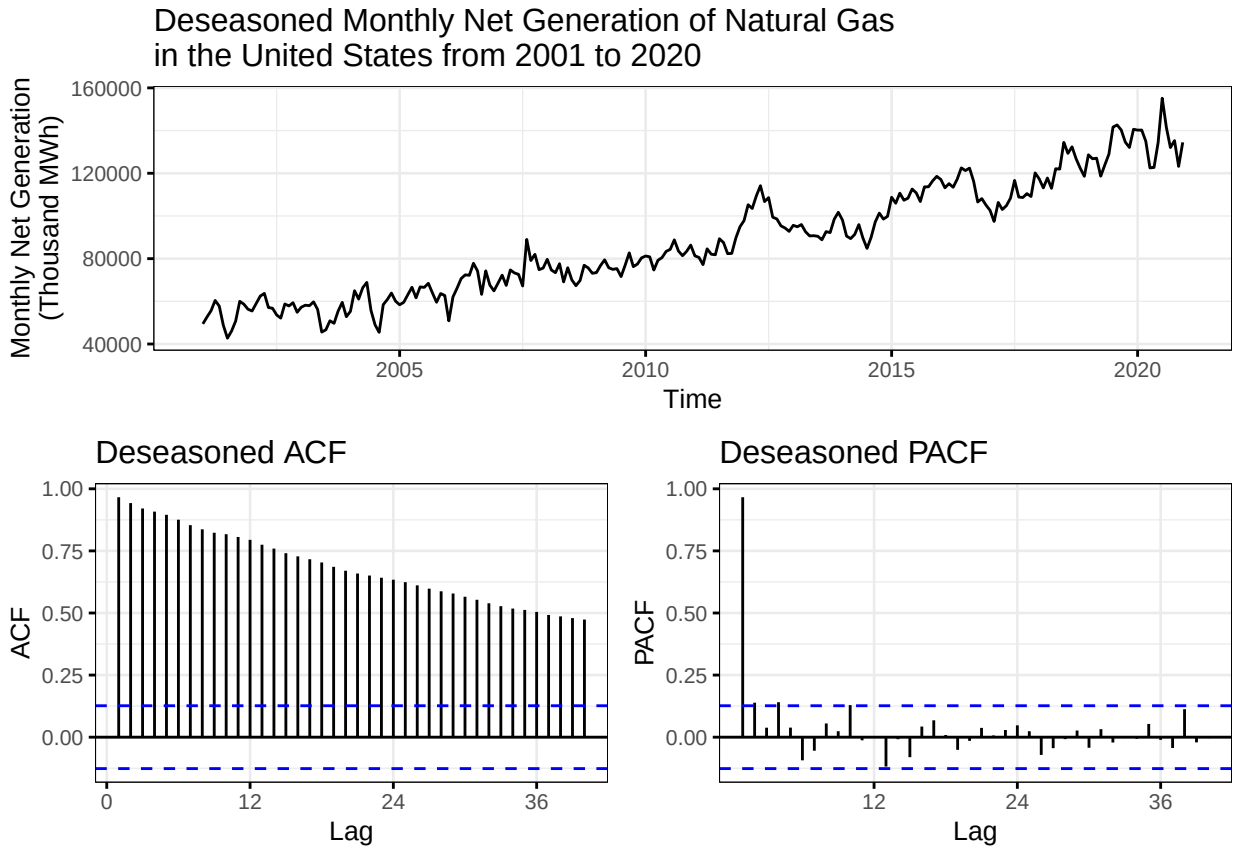


Q2

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.

```
#create a deseasoned timeseries
energy_ts_decomp <- decompose(energy_ts, type = "additive")
energy_ts_deseason <- seasadj(energy_ts_decomp)

#create deseasoned plot
deseason_plot <- autoplot(energy_ts_deseason)+
  labs(y = "Monthly Net Generation\n(Thousand MWh)",
       title = "Deseasoned Monthly Net Generation of Natural Gas\nin the United States from 2001 to 2020

#create and plot ACF and PACF
deseason_acf <- Acf(energy_ts_deseason,
                    lag.max=40,
                    type="correlation",
                    plot=FALSE) %>%
  autoplot()+
  labs(title = "Deseasoned ACF")

deseason_pacf <- Pacf(energy_ts_deseason,
                    lag.max=40,
```

```
                  type="correlation",
                  plot=FALSE) %>%
  autoplot()+
  labs(title = "Deseasoned PACF")

#plot the deseasoned timeseries, acf, and pacf
ggarrange(deseason_plot,
          ggarrange(deseason_acf, deseason_pacf, ncol = 2),
          nrow = 2)
```



The timeseries both appear to have a similar upward trend with time. However, the deseasoned timeseries has much smaller peaks and valleys that do not recur on a yearly basis - the influence of seasonality. The seasonality can be seen in the ACF plot. The seasoned data fluctuates, although it maintains a decaying trend. The deseasoned ACF plot does not fluctuate and decays steadily. The PACF plot of the seasoned data has 9 significant lags. This is more than the deseasoned PACF, which only has one lag far above the line of significance and two lags that are just significant. Furthermore, in the deseasoned PACF, the 3 significant lags are in the first 4 lags, whereas the significant lags are more spread out in the seasonal 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.

```r
#run adf
print((adf.test(energy_ts_deseason,alternative="stationary")))
```

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

The p-value of the ADF test is less than 0.05. This indicates that the alternative hypothesis is accepted over the null hypothesis. The alternative hypothesis indicates that the data is stationary over stochastic. Therefore, this time series does not purely follow a random walk.

```r
summary(MannKendall(energy_ts_deseason))
```

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

The p-value of the Mann-Kendall test is less than 0.05. This indicates that the alternative hypothesis is accepted over the null hypothesis. The alternative hypothesis indicates that the data is deterministic over stationary. Therefore, this time series has a deterministic trend. The tau value is 0.843. A positive value close to 1, like 0.843, indicates there is a strong upwards trend over time.

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.

For this assignment, differencing will not be conducted and the differenced series will not be displayed in ACF/PACF plots because the question specifies to use plots generated in Q2. The seasonal component has been removed and does not need to be included in this trend. The ACF plot for the deseasoned data decays gradually. This is indicative of an autoregressive trend. The order of the AR trend is shown in the PACF plot. The PACF plot has a sharp decrease after lag 1. Therefore, the order of p = 1. Moving average trends are shown by a gradual decay in the PACF plot and a sharp decrease in the ACF plot that signifies order. Disregarding the sharp drop from the AR component in the PACF, there is no gradual decay visible. Furthermore, I do not see any sharp decrease between lags in the ACF. Thus, I am concluding there is no MA component to the time series and q = 0. Differencing is required when there is non-stationarity to the time series. Our ADF test indicates that there was no stochastic trend, but the Mann-Kendall test indicates that there is a deterministic trend. Therefore, differencing needs to be done. This can be checked with the `ndiffs` fuction. `ndiffs` tells you how many times you need to difference a time series. This is conducted below.

```r
ndiffs(energy_ts_deseason)
```

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

`ndiffs` indicates that differencing should be conducted once. Therefore, d = 1. The final ARIMA notation is as follows: ARIMA(1,1,0).

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.

```r
energy_arima <- Arima(energy_ts_deseason,
                      order = c(1,1,0),
                      include.drift=TRUE)

print(energy_arima)
```

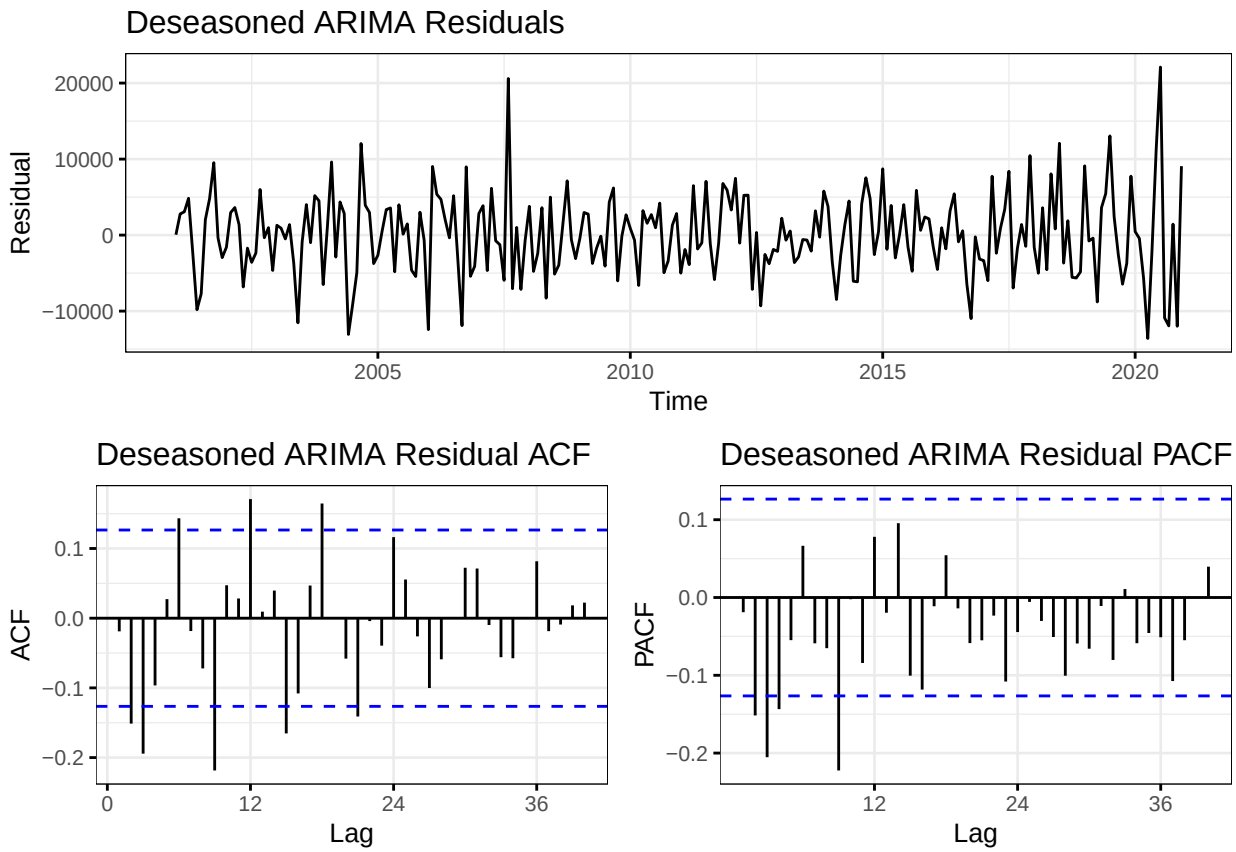
```
## Series: energy_ts_deseason
## ARIMA(1,1,0) with drift
##
## Coefficients:
##           ar1     drift
##       -0.1479  348.3927
## s.e.   0.0644  308.8385
##
## sigma^2 = 30254066:  log likelihood = -2396.54
## AIC=4799.07   AICc=4799.18   BIC=4809.5
```

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?

```r
#create residual plots
energy_arima_resid <- autoplot(energy_arima$residuals)+
  labs(title = "Deseasoned ARIMA Residuals",
       y = "Residual")

energy_arima_resid_acf <- autoplot(Acf(energy_arima$residuals,
                                       lag.max=40,
                                       plot = FALSE))+
  labs(title = "Deseasoned ARIMA Residual ACF")

energy_arima_resid_pacf <- autoplot(Pacf(energy_arima$residuals,
                                       lag.max=40,
                                       plot = FALSE))+
  labs(title = "Deseasoned ARIMA Residual PACF")
```

```
#print residual plots
ggarrange(energy_arima_resid,
          ggarrange(energy_arima_resid_acf,
                    energy_arima_resid_pacf,
                    ncol = 2),
          nrow = 2)
```



The ARIMA residuals series visually appears to oscillate around a 0 mean without any particular trend. There are also not many significant lags (4 significant lags) to the PACF plot. However, almost most of the lags are negative. The ACF plot has multiple significant lags that appear to follow an oscillating pattern with decay. This indicates that this series may not be a white noise series. The peaks and valleys in the residual series may be indicative of remaining seasonality or other component that was not captured as opposed to white noise. This is likely because the seasonality of the model was not added back in and accounted for in the ARIMA model. A SARIMA model may provide a better fit for the data.

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.

```
#run adf
print((adf.test(energy_ts,alternative="stationary")))
```

Stationarity tests

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

The p-value of the ADF test is less than 0.05. This indicates that the alternative hypothesis is accepted over the null hypothesis. The alternative hypothesis indicates that the data is stationary over stochastic. Therefore, this time series does not purely follow a random walk.

```
#run seasonal mk test
summary(SeasonalMannKendall(energy_ts))
```

```
## Score =  2022 , Var(Score) = 11400
## denominator =  2280
## tau = 0.887, 2-sided pvalue =< 2.22e-16
```

The p-value of the Mann-Kendall test is less than 0.05. This indicates that the alternative hypothesis is accepted over the null hypothesis. The alternative hypothesis indicates that the data is deterministic over stationary. Therefore, this time series has a deterministic trend. The tau value is 0.887. A positive value close to 1, like 0.887, indicates there is a strong upwards trend over time.

Model Order

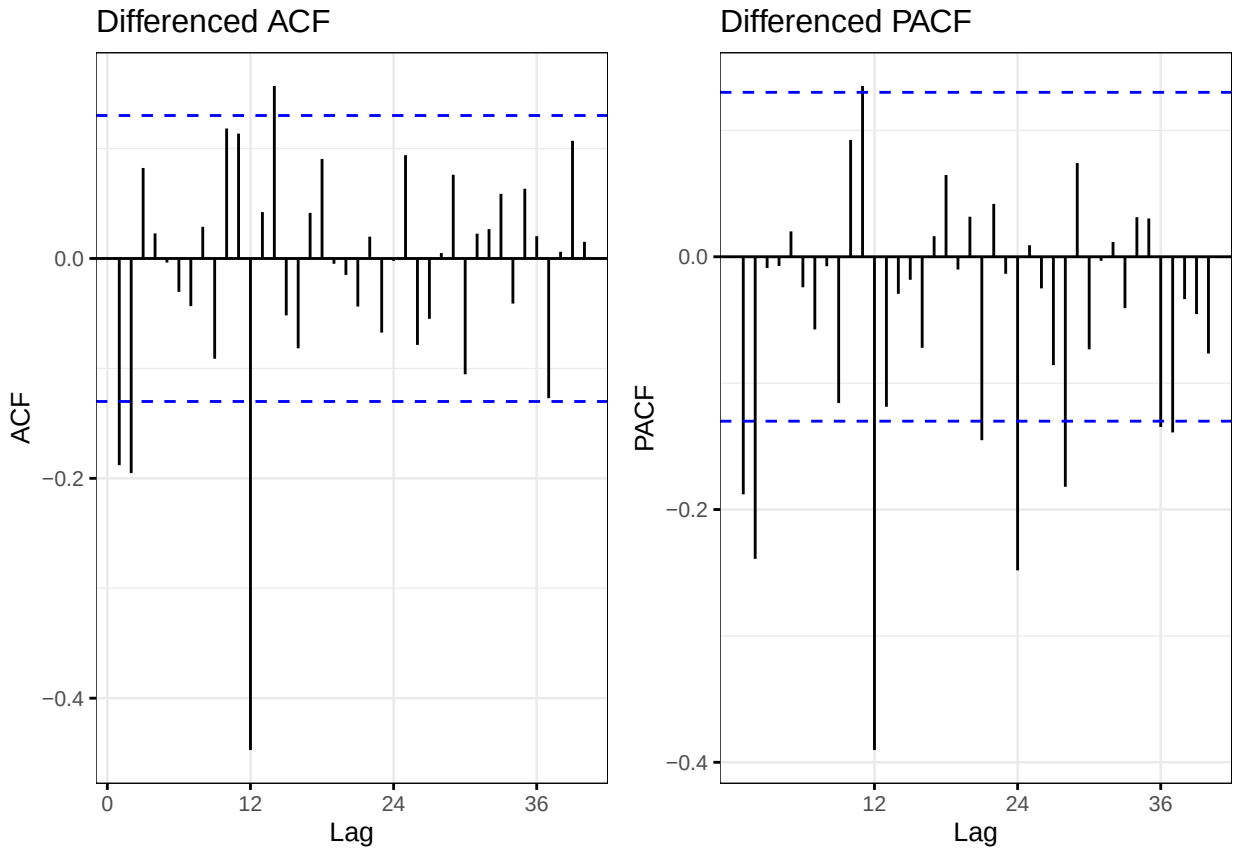
Starting with the non-seasonal components, removing the seasonality is required to determine the ARIMA orders for the non-seasonal components. This has done previously. So, I am going to retain the orders specified previousy of ARIMA (1,1,0). Moving onto the seasonal components, there is a seasonal component, and it seems like this component is stable over time. However, `nsdiffs` is going to be used to verify if the seasonal component requires differencing.

```
nsdiffs(energy_ts)
```

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

```
#do the required differencing
energy_ts_diff <- diff(energy_ts, lag = 1, differences = 1) %>%
  diff(lag = 12, differences = 1)

#create new PACF/ACF plots
diff_acf <- Acf(energy_ts_diff,
                  lag.max=40,
```

```
                  type="correlation",
                  plot=FALSE) %>%
  autoplot()+
  labs(title = "Differenced ACF")

diff_pacf <- Pacf(energy_ts_diff,
                  lag.max=40,
                  type="correlation",
                  plot=FALSE) %>%
  autoplot()+
  labs(title = "Differenced PACF")

plot_grid(diff_acf, diff_pacf)
```



According to `nsdiffs`, this differencing needs to occur once. So, D = 1. Therefore, I differenced the series. To identify the seasonal components, I am going to look at the seasonal lags in the ACF and PACF. The spikes occur at lags of multiples of 12. So, s = 12. In the ACF, there is one spike in the seasonal lag. In the PACF plot, there are multiple spikes at the seasonal lags. This indicates a SMA process, so Q = 1. Therefore, P = 0, because P + Q <= 1. My final ARIMA model is: $ARIMA(1,1,0)(0,1,1)_{12}$.

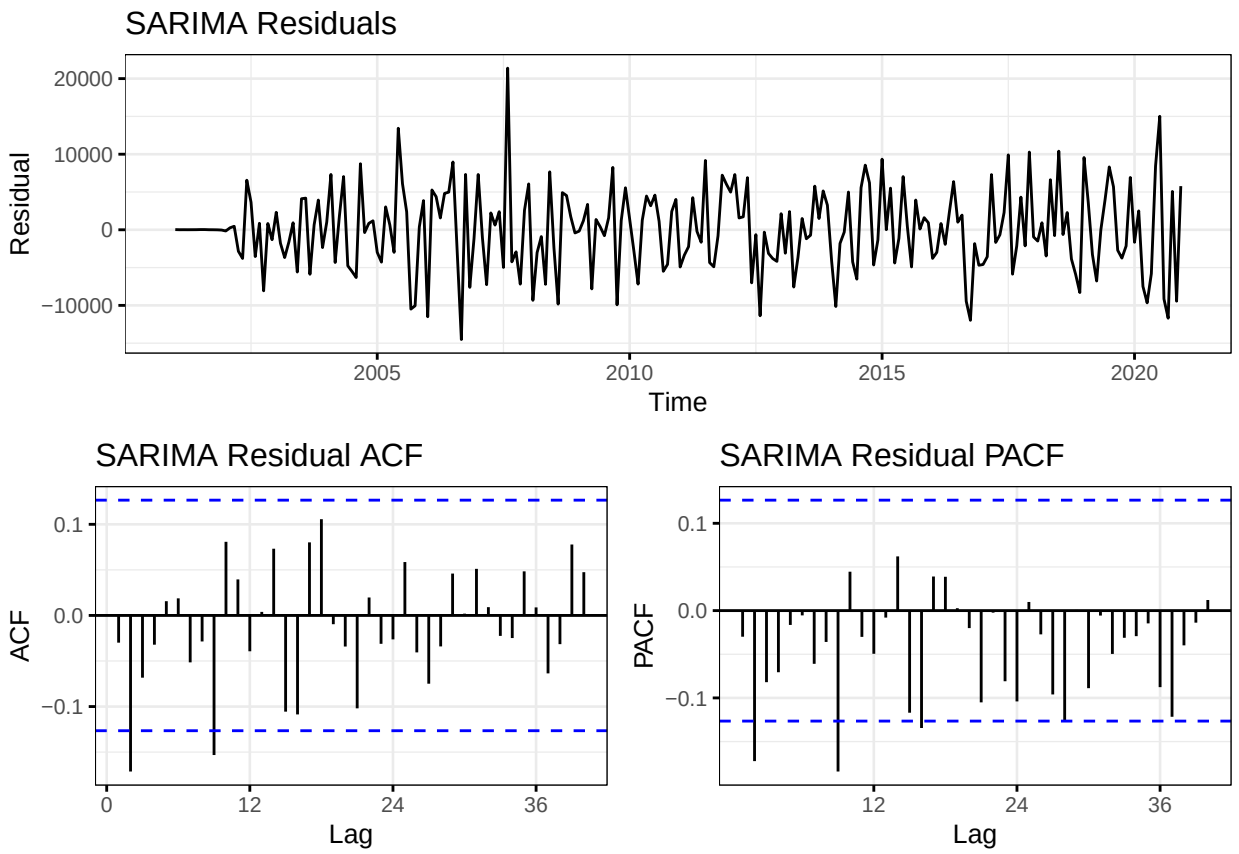
```
energy_sarima <- Arima(energy_ts,
                       order=c(1,1,0),
```

```
                      seasonal=c(0,1,1),
                      include.drift=TRUE)

print(energy_sarima)
```

ARIMA

```
## Series: energy_ts
## ARIMA(1,1,0)(0,1,1)[12]
##
## Coefficients:
##           ar1     sma1
##       -0.1808  -0.6898
## s.e.   0.0655   0.0557
##
## sigma^2 = 30626308:  log likelihood = -2281.43
## AIC=4568.86   AICc=4568.96   BIC=4579.13
```

```
#create residual plots
energy_sarima_resid <- autoplot(energy_sarima$residuals)+
  labs(title = "SARIMA Residuals",
       y = "Residual")

energy_sarima_resid_acf <- autoplot(Acf(energy_sarima$residuals,
                                        lag.max=40,
                                        plot = FALSE))+
  labs(title = "SARIMA Residual ACF")

energy_sarima_resid_pacf <- autoplot(Pacf(energy_sarima$residuals,
                                        lag.max=40,
                                        plot = FALSE))+
  labs(title = "SARIMA Residual PACF")

#print residual plots
ggarrange(energy_sarima_resid,
          ggarrange(energy_sarima_resid_acf,
                    energy_sarima_resid_pacf,
                    ncol = 2),
          nrow = 2)
```



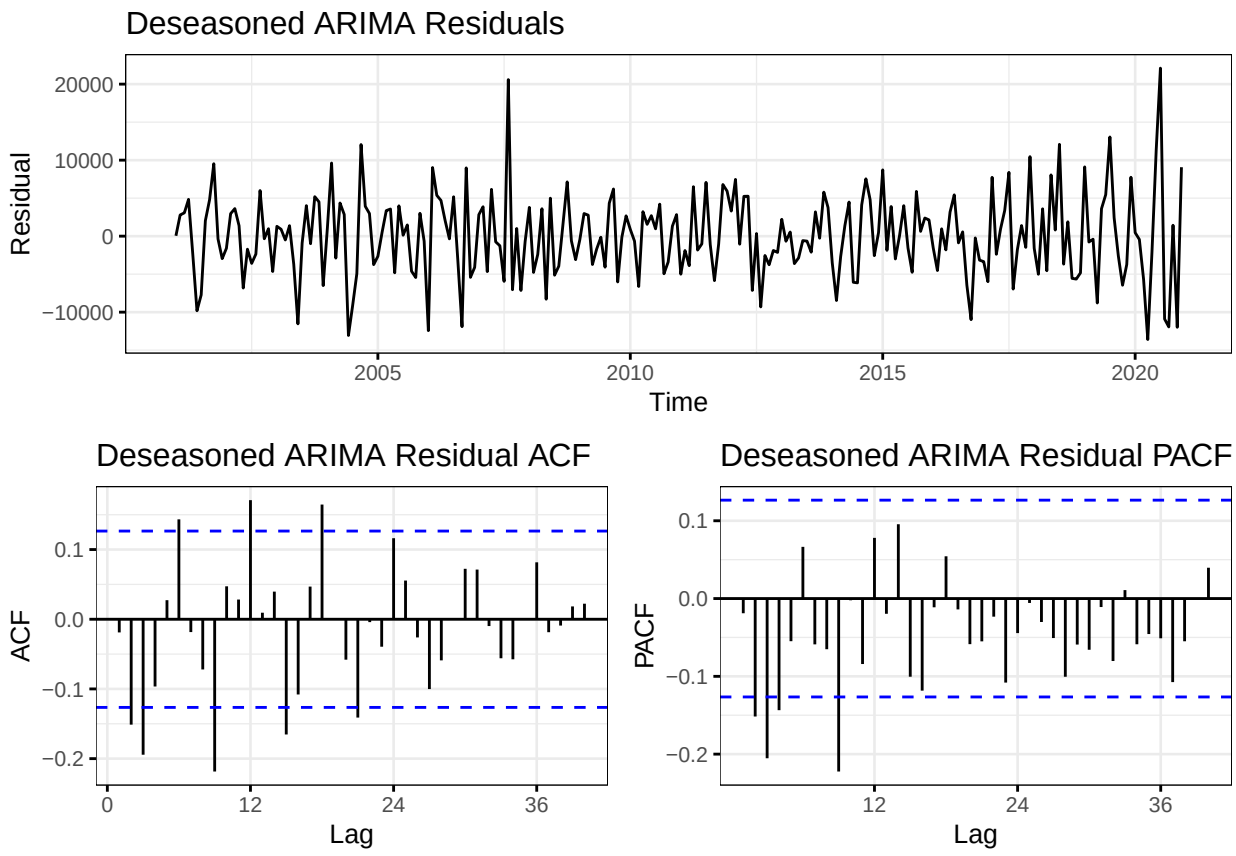
Residuals

The SARIMA residuals series visually appears to oscillate around a 0 mean without any particular trend, but there is a noticeable period of residuals = 0 at the start of the timeseries. There are also few significant lags in both the ACF (2 lags) and PACF (3 lags) plots. There also does not seem to be any patterns in either plots. Although, the PACF is still mostly negative. Therefore, I would call this 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.

The residual plots of the deseasoned series are displayed again below for convenience. This is done with the same `ggarrange` *code, but it is hidden.*



The SARIMA model is better for representing the natural gas series based on the residuals. The ARIMA model had more significant lags in the ACF and PACF plots than the SARIMA model. The ARIMA residual ACF plot also had a very apparent pattern to the residuals - a regular oscillation around an ACF of 0. The patterns and number of significant lags is perhaps indicative of a component (e.g., seasonality) that was not fully accounted for. This was not present in the SARIMA, which indicates that this model was able to account for this component missed in the ARIMA. I am more confident in labelling the SARIMA residual series as white noise.

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(energy_ts_deseason)
```

```
## Series: energy_ts_deseason
## 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 autofit chose the best model to be ARIMA(1,1,1). I decided on ARIMA(1,1,0). I did not identify the moving average component that the autofit did. Therefore, we did not match.

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(energy_ts)
```

```
## Series: energy_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
```

The autofit chose the best model to be $ARIMA(1,0,0)(0,1,1)_{12}$. I decided on $ARIMA(1,1,0)(0,1,1)_{12}$. The seasonal component matches, but the non-seasonal component does not match.