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

Assignment 7 - Due date 03/06/25

Jessalyn Chuang

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 errors
```

```r
library(cowplot)
```

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

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
#Importing data
net_generation <- read.csv(file="./Data/Net_generation_United_States_all_sectors_monthly.csv",header=TRU

#Preparing the data - create date object and rename columns
ng_gen <-
  net_generation %>%
  mutate( Month = my(Month) ) %>%
  select(Month, natural.gas.thousand.megawatthours) %>%
  rename(gen_thousand_MWh = natural.gas.thousand.megawatthours) %>%
  arrange(Month)

head(ng_gen)
```

```
##        Month gen_thousand_MWh
## 1 2001-01-01         42388.66
## 2 2001-02-01         37966.93
## 3 2001-03-01         44364.41
## 4 2001-04-01         45842.75
## 5 2001-05-01         50934.21
## 6 2001-06-01         57603.15
```

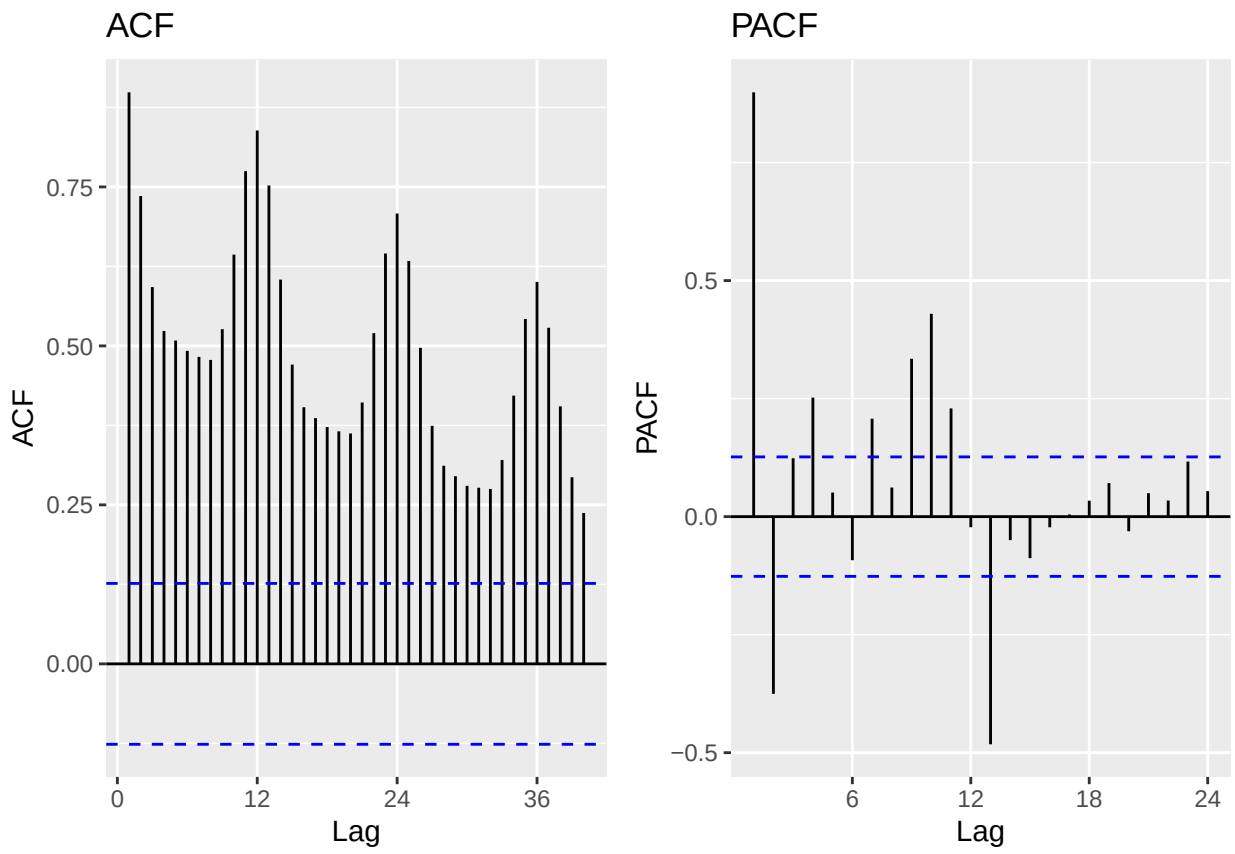
```
summary(ng_gen)
```

```
##      Month             gen_thousand_MWh
##  Min.   :2001-01-01   Min.   : 37967
##  1st Qu.:2005-12-24   1st Qu.: 62245
##  Median :2010-12-16   Median : 84415
##  Mean   :2010-12-16   Mean   : 88028
##  3rd Qu.:2015-12-08   3rd Qu.:108385
##  Max.   :2020-12-01   Max.   :185445
```

```
sum(is.na(ng_gen))
```

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

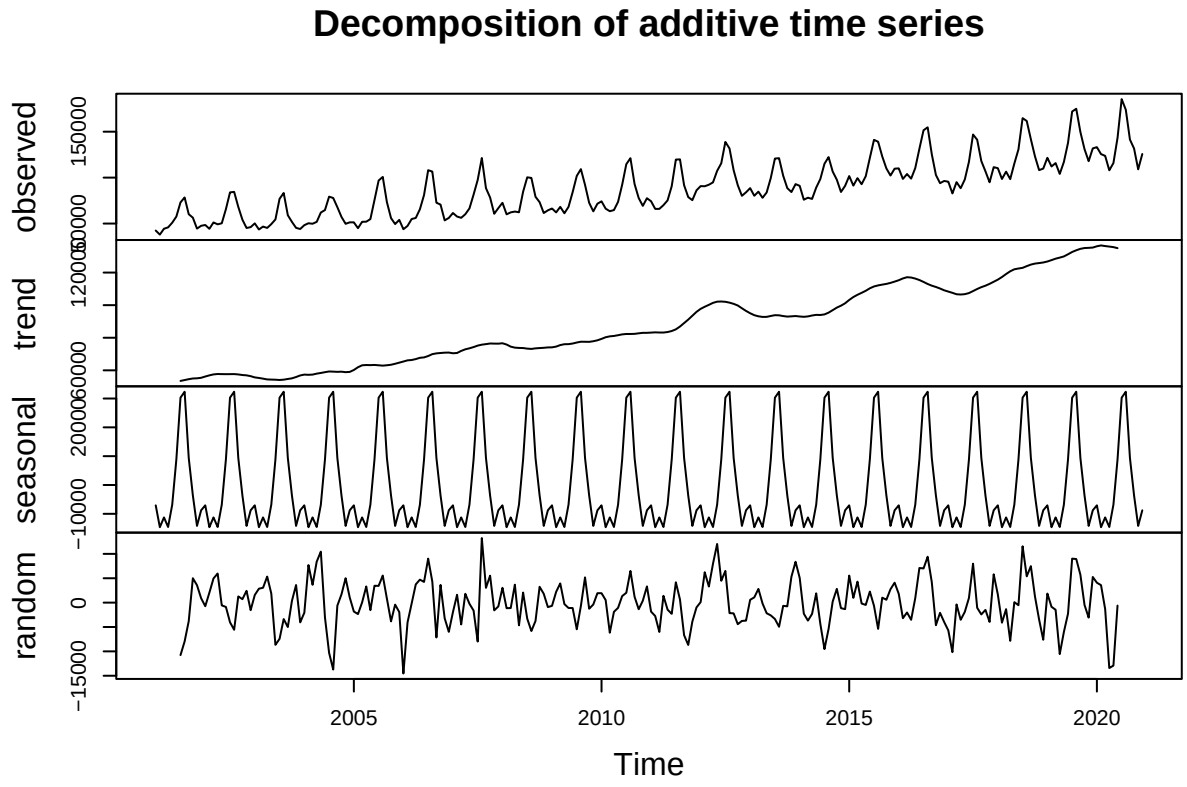
```
#No NAs so don't need to worry about missing values

#Create time series
ts_ng_gen <- ts(ng_gen[,2],
                           start=c(year(ng_gen$Month[1]),month(ng_gen$Month[1])),
                           frequency=12)

#ACF and PACF plots
acf_plot <- autoplot(Acf(ts_ng_gen, lag = 40, plot = FALSE)) +
  ggtitle("ACF")

pacf_plot <- autoplot(Pacf(ts_ng_gen, plot = FALSE)) +
  ggtitle("PACF")

# Combine the plots in one row
plot_grid(acf_plot, pacf_plot, nrow = 1)
```



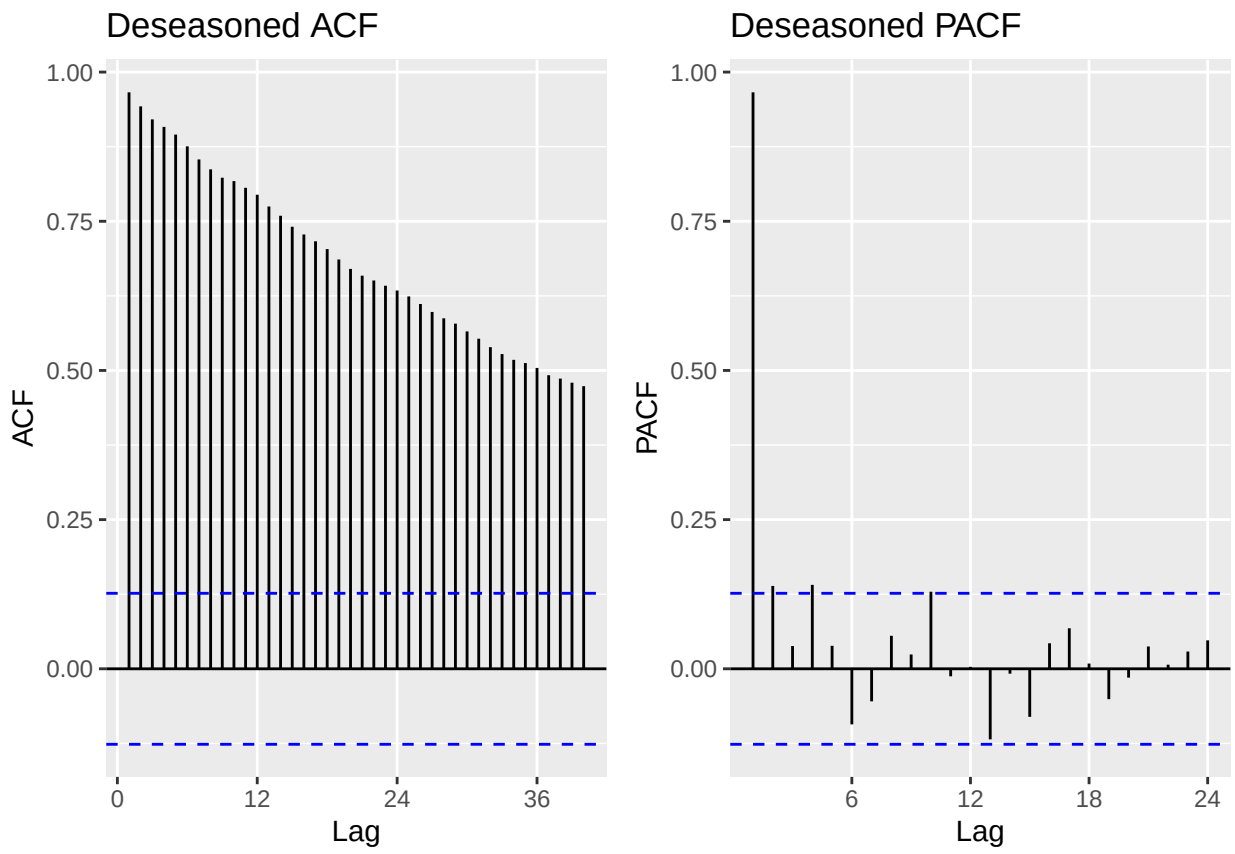
Q2

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.

```
#Decomposing
decompose_ng_gen <- decompose(ts_ng_gen,"additive")
plot(decompose_ng_gen)
```




```r
#Creating non-seasonal ng_gen time series
deseason_ng_gen <- seasadj(decompose_ng_gen)

#ACF and PACF plots
acf_plot_deseason <- autoplot(Acf(deseason_ng_gen, lag = 40, plot = FALSE)) +
  ggtitle("Deseasoned ACF")

pacf_plot_deseason <- autoplot(Pacf(deseason_ng_gen, plot = FALSE)) +
  ggtitle("Deseasoned PACF")

# Combine the plots in one row
plot_grid(acf_plot_deseason, pacf_plot_deseason, nrow = 1)
```



In these plots, the ACF is much smoother with a gradual decline, indicating that the seasonal component has been removed. The PACF also now drops off quickly after lag 1, making the non-seasonal AR characteristics of the de-seasoned data clearer.

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 to check stationarity
print(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 = -4.0271, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```r
#Mann Kendall Test to check stationarity
summary(MannKendall(deseason_ng_gen))
```

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

Both tests results with p value being less than 0.05. For both, the null hypotheses were rejected in favor for the alternate hypothesis. From the ADF test, the alternative hypothesis is that the data is stationary, and from the Mann Kendall test the alternative hypothesis is that the data follows a trend. These conclusions suggest that the data has a deterministic trend, but no stochastic 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.

p = 1 This is because the significant lags in the PACF determine the order of the AR and the PACF cuts off sharply after lag 1. d = 0 Data does not seem like it needs to be differenced because the ADF test suggested that there is stationarity and differencing is not required for stationary data. No seasonal differencing is needed either because the data has been deseasoned. q = 0 This data looks like it is an AR process (gradual decline in ACF, quick cut off in PACF) and not an MA process.

Different values for p,d,and q can be tested, and I can compare the AIC values of each model to further help with determining what the best parameters are without *auto.arima*(). A lower AIC value indicates a better model. It can also be helpful to look at the residuals of the model. If the residual time series appears random, and the ACF and PACF of residuals show no significant autocorrelation, this suggests a good model fit, meaning that the remaining noise is white noise.

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
Model_100 <- Arima(deseason_ng_gen,order=c(1,0,0),include.mean = TRUE, include.drift=TRUE)
print(Model_100)
```

```
## Series: deseason_ng_gen
## ARIMA(1,0,0) with drift
##
## Coefficients:
##          ar1  intercept     drift
##       0.7182  44800.489  359.3965
## s.e.  0.0445   2295.974   16.4299
##
## sigma^2 = 26630969:  log likelihood = -2391.11
## AIC=4790.22   AICc=4790.39   BIC=4804.14
```

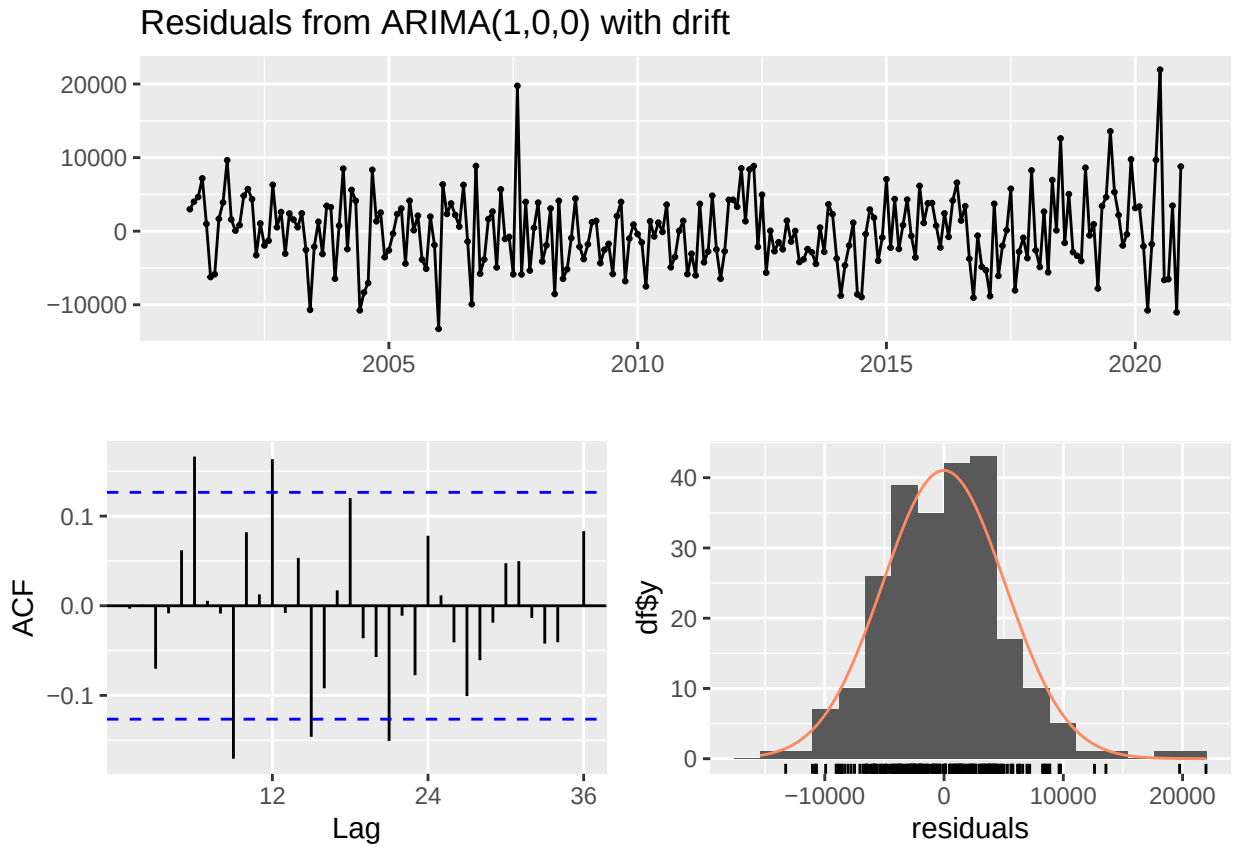
```
cat("phi =", 0.7182, ", intercept =", 44800.489, ", drift =", 359.3965, "\n")
```

```
## phi = 0.7182 , intercept = 44800.49 , drift = 359.3965
```

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_100)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0) with drift
## Q* = 47.775, df = 23, p-value = 0.001787
##
## Model df: 1.   Total lags used: 24
```

The residual series does not look perfectly like a white noise series. Ideally, the residuals should fluctuate randomly around zero with no clear pattern, but in the top plot of the time series, there are large spikes towards the later years, suggesting some heteroscedasticity. So this could mean that my model hasn't

captured all the patterns in the data. There should also be no significant auto-correlations in the ACF, but there are some lags that lie outside of the significant bounds. Lastly the histogram does appear roughly normal as it should, but there are some outliers and a little bit of skewness.

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.

Repeat of Q3

```
#ADF Test to check stationarity
print(adf.test(ts_ng_gen, alternative = "stationary"))
```

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

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

```
#Mann Kendall Test to check stationarity
summary(MannKendall(ts_ng_gen))
```

```
## Score =  18658 , Var(Score) = 1545533
## denominator =  28680
## tau = 0.651, 2-sided pvalue =< 2.22e-16
```

Again, both tests results with p value being less than 0.05. For both, the null hypotheses were rejected in favor for the alternate hypothesis. From the ADF test, the alternative hypothesis is that the data is stationary, and from the Mann Kendall test the alternative hypothesis is that the data follows a trend. These conclusions suggest that the original data has a deterministic trend, but no stochastic trend.

Repeat of Q4

Non-Seasonal: p = 1 d = 0 q = 0 I am expecting a non-seasonal AR process based on the generally steady decline in the ACF prior to the seasonal lag and the steep drop off in the PACF after the first two lags. Because the alternative hypothesis of the ADF test was accepted that the data is stationary, I am not including differencing.

Seasonal: P = 1 D = 1 Q = 0 I am expecting a seasonal AR process because the PACF shows a significant spike near the seasonal lag and multiple significant algs in the ACF. I am including differencing because the ACF shows strong spikes at seasonal lags, suggesting seasonal non-stationarity.

Repeat of Q5

```r
SARIMA_ng_gen <- Arima(ts_ng_gen,
                       order=c(1,0,0),
                       seasonal=c(1,1,0),
                       include.drift=TRUE)
print(SARIMA_ng_gen)
```

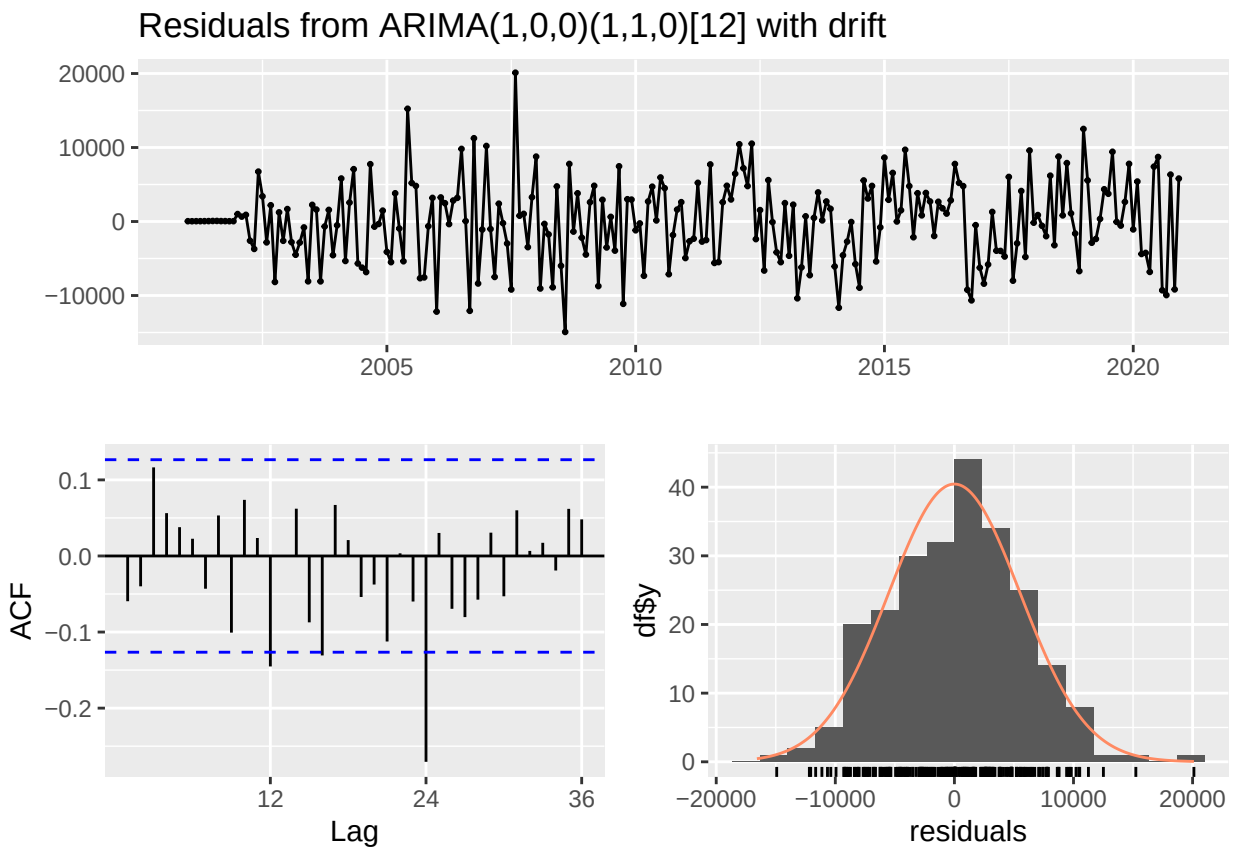
```
## Series: ts_ng_gen
## ARIMA(1,0,0)(1,1,0)[12] with drift
##
## Coefficients:
##          ar1     sar1     drift
##       0.7646  -0.4542  358.3892
## s.e.  0.0424   0.0593   91.4518
##
## sigma^2 = 32457520:  log likelihood = -2295.5
## AIC=4598.99   AICc=4599.17   BIC=4612.71
```

```r
cat("phi =", 0.7646, ", seasonal phi =", -0.4542, ", drift =", 358.3892, "\n")
```

```
## phi = 0.7646 , seasonal phi = -0.4542 , drift = 358.3892
```

Repeat of Q6

```r
checkresiduals(SARIMA_ng_gen)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0)(1,1,0)[12] with drift
## Q* = 50.251, df = 22, p-value = 0.0005424
##
## Model df: 2.   Total lags used: 24
```

The residual series does not perfectly resemble a white noise series. In the first time series plot, there are some clusters of higher volatility and some slight characteristics of heteroskedasticity. The ACF should have insignificant autocorrelations across lags, but the 24th lag falls outside of this. Lastly, in the histogram, it seems roughly normally distributed though, which is closer to what we expect.

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.

It wouldn't be fair to compare the models by comparing the residuals as it wouldn't be fair to compare the two models in the first place as they are taking in different input data. The ARIMA was fitted on deseasoned data while the SARIMA was fitted on the original data.

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(deseason_ng_gen,max.D=0,max.P = 0,max.Q=0)
arima_autofit
```

```
## Series: deseason_ng_gen
## 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 of the best ARIMA is (1,1,1). This does not match what I specified in Q4, which was (1,0,0).

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_ng_gen)
sarima_autofit
```

```
## 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 = 27569124:  log likelihood = -2279.54
## AIC=4567.08   AICc=4567.26   BIC=4580.8
```

The order of the best SARIMA is (1,0,0)(0,1,1)[12]. This does not match what I specified in Q4, which was (1,0,0)(1,1,0)[12].