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

Assignment 7 - Due date 03/07/24

Faustin Kambale

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

```
library(lubridate)
library(ggplot2)
library(forecast)
library(Kendall)
library(tseries)
library(outliers)
library(tidyverse)
library(cowplot)
library(sarima)
library(patchwork)
library(dplyr)
library(forecast)
```

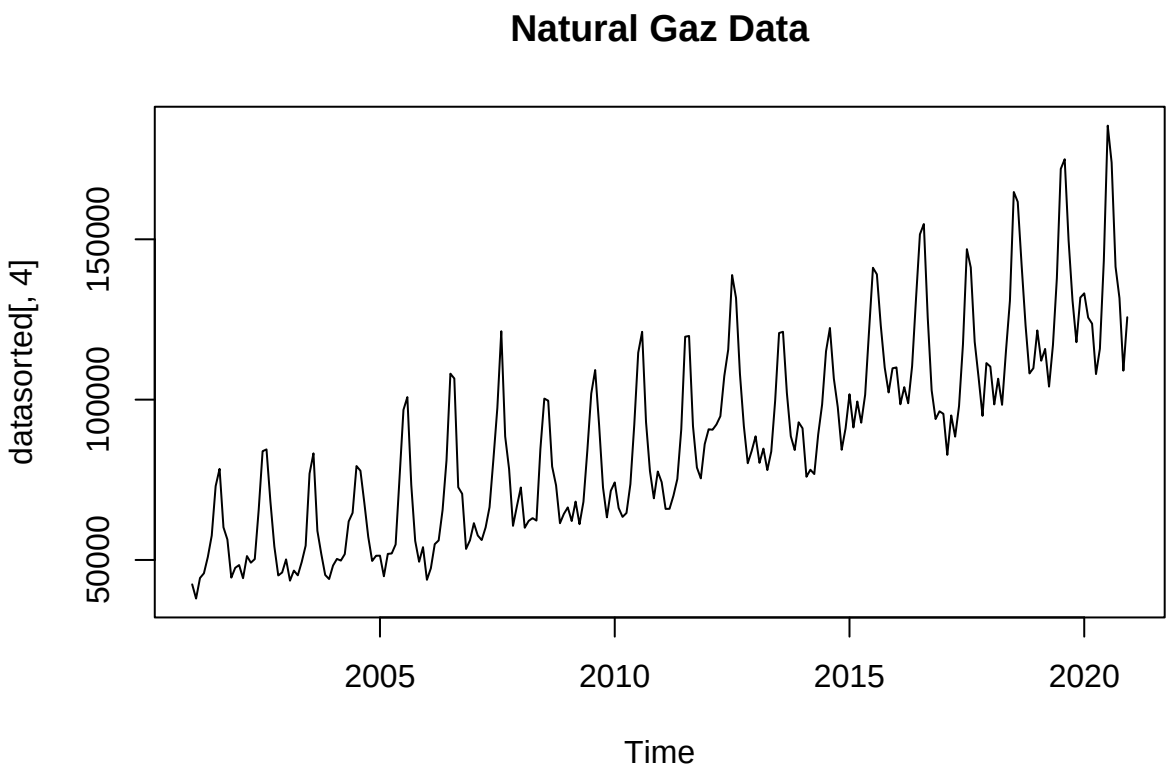
Importing and processing the data set

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.

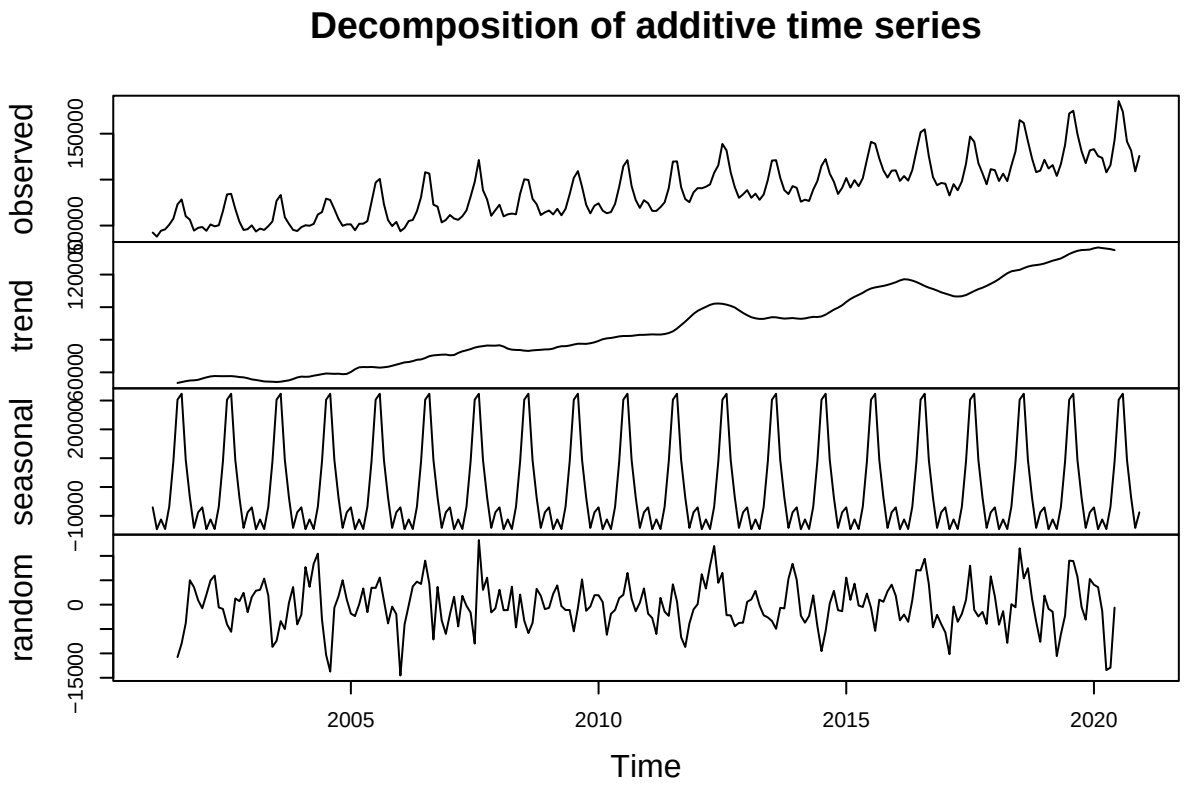
```
raw_data <- read.csv(file ="/Users/faustinkambale/Library/CloudStorage/OneDrive-DukeUniversity/Spring 2
                     header=TRUE,skip=4)
#Processing Data
datasorted <-
  raw_data %>%
  mutate( Month = my(Month) ) %>%
  arrange( Month )

#Set variables to work with
data <- as.data.frame(datasorted[, 4])
tsdata <- ts(data, start=c(2001,1), frequency=12)
plot(tsdata, main = "Natural Gaz Data")
```



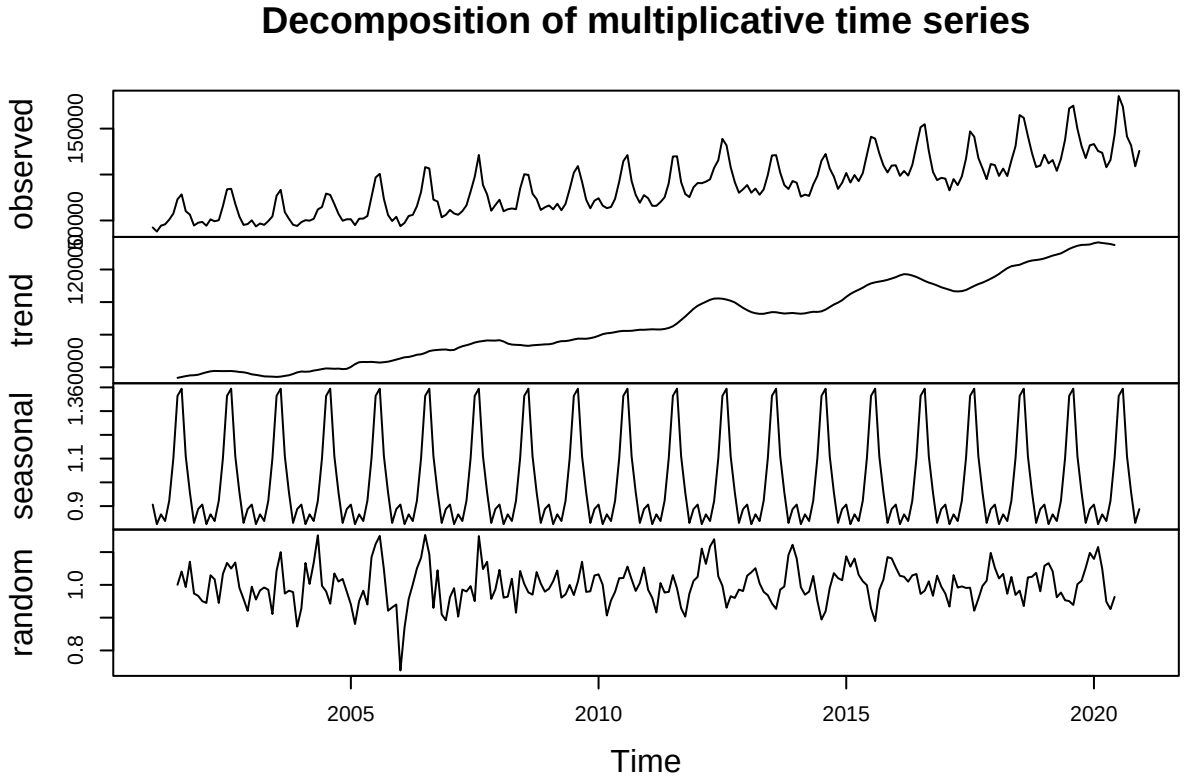
Q2

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.

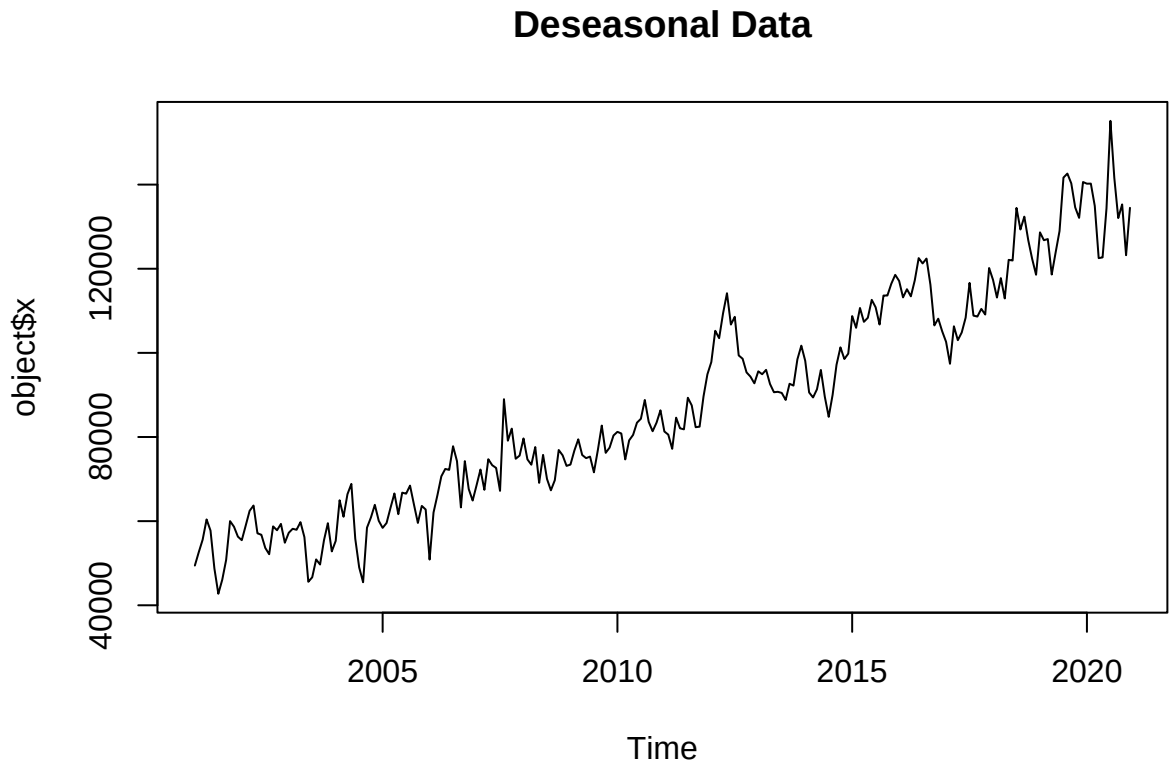
```
#Decomposing with additive function
datadecomp_add <- decompose(tsdata, type = "additive")
plot(datadecomp_add)
```



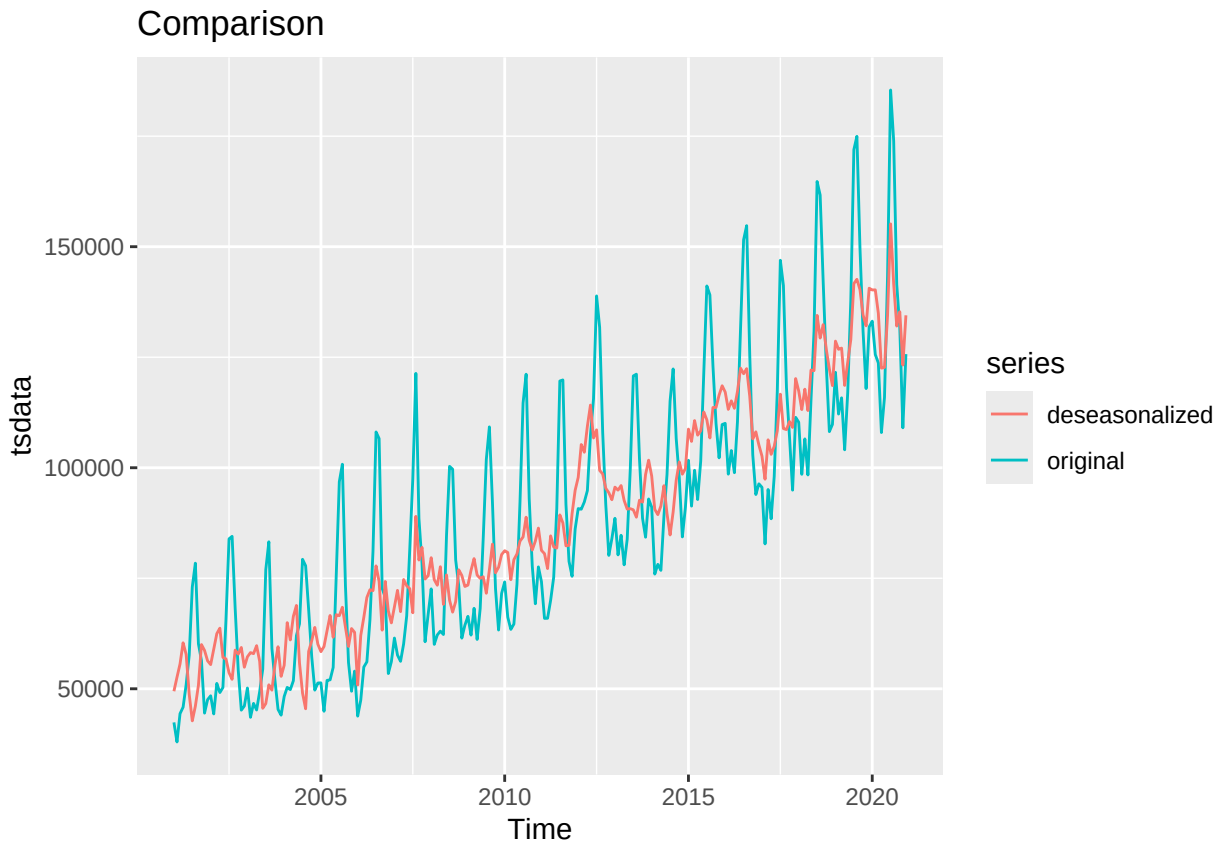

```
datadecomp_mult <- decompose(tsdata, type = "multiplicative")
plot(datadecomp_mult)
```



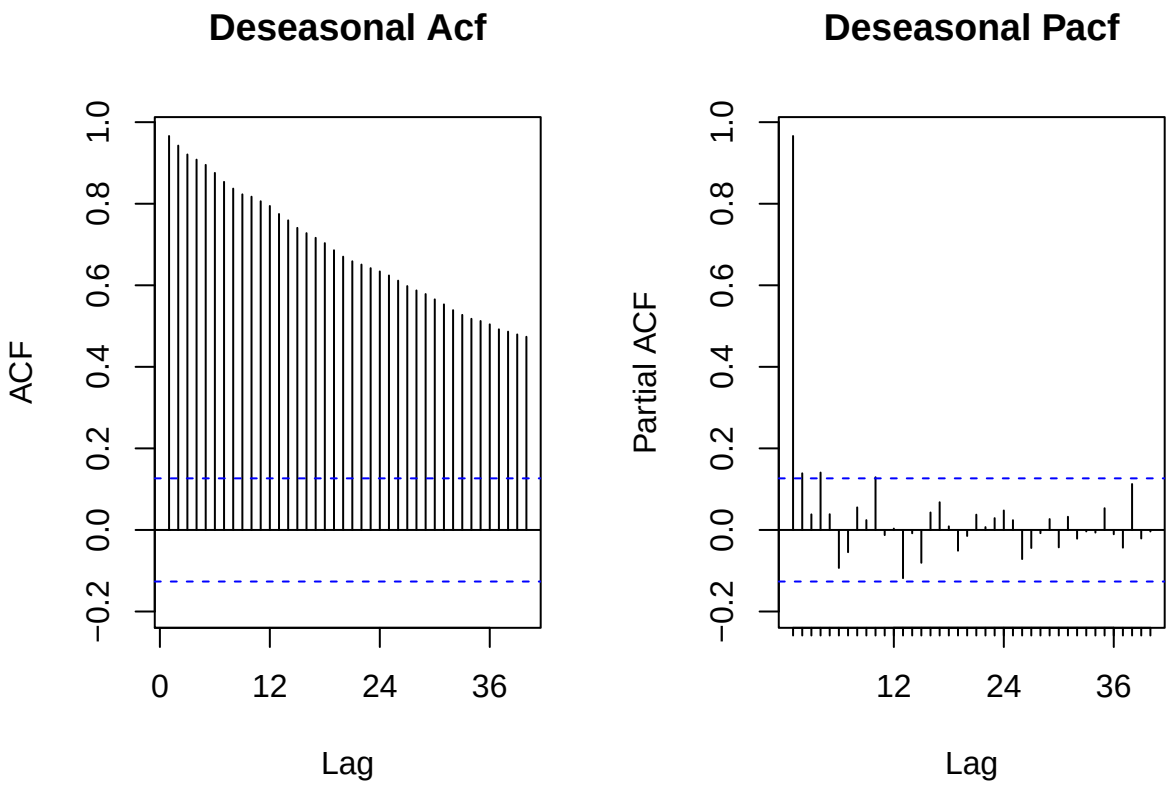
```r
datadeseasonal <- seasadj(datadecomp_add)
plot(datadeseasonal, main = "Deseasonal Data")
```




```r
#Comparing plot
autoplot (tsdata, series = "original", main = "Comparison")+
  autolayer (datadeseasonal, series = "deseasonalized")
```

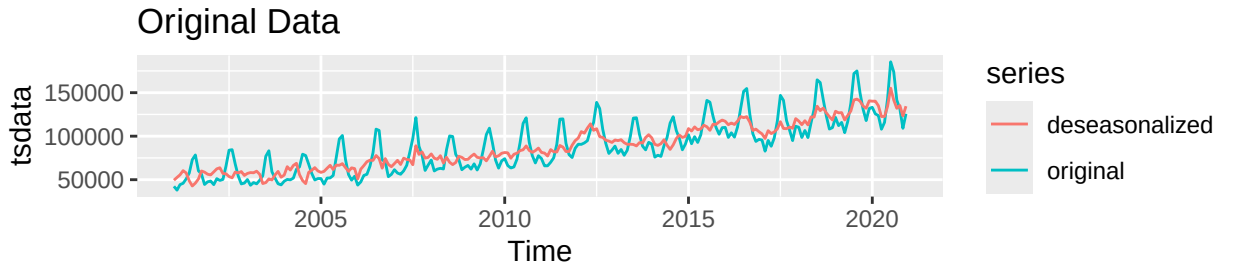


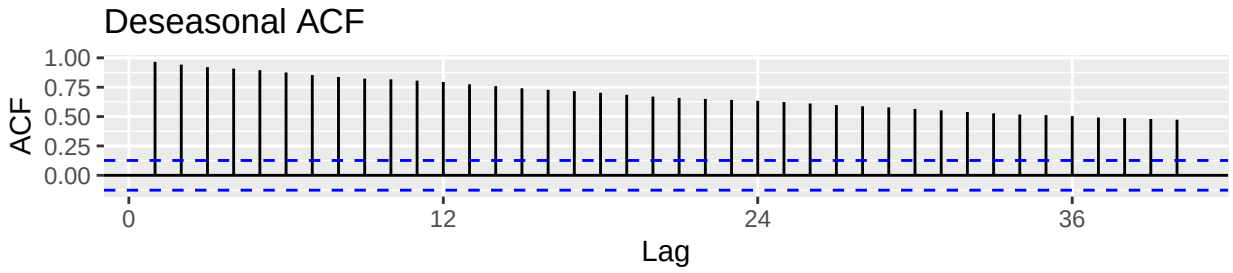

```
#ACF and PACF plots
par(mfrow=c(1,2))
deseason_acf <-Acf(datadeseasonal, lag = 40, main = "Deseasonal Acf", plot=TRUE)
deseason_pacf <-Pacf(datadeseasonal, lag = 40, main = "Deseasonal Pacf", plot=TRUE)
```

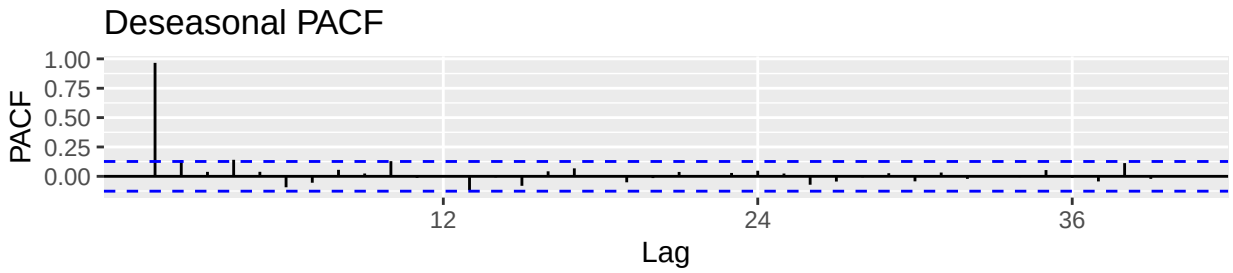



```
par(mfrow=c(1,1))

#Comparing ACF and PACF plots from the original
plot_grid(
  autoplot(tsdata, series = "original", main = "Original Data")+
  autolayer (datadeseasonal, series = "deseasonalized"),
  autoplot(Acf(datadeseasonal, lag = 40, plot=FALSE),
           main = "Deseasonal ACF"),
  autoplot(Pacf(datadeseasonal, lag = 40, plot=FALSE),
           main = "Deseasonal PACF"),
  nrow = 3, ncol = 1
  )
```

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







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
adf_result <- adf.test(datadeseasonal)
```

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

```
print(adf_result)
```

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

```
# Mann Kendall test
mk_result <- MannKendall(datadeseasonal)
print(mk_result)
```

```
## tau = 0.843, 2-sided pvalue =< 2.22e-16
```

The Dickey-Fuller of -4.0271, Lag order = 6, p-value = 0.01 indicate that we can reject the null hypothesis and confirm the alternative that our series are stationary. Result statistically significant at 10% level. Mann Kendall test : tau = 0.843 (close to 1) and a 2-sided p-value =< 2.22e-16 (close to zero) indicate that our series *have a strong positive trend*. This result is *statistically significant*.

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.

```
# Identifying the ARIMA parameters

#d=0  because the original series are stationary and did not differentiate.
#p = 1 because there is a sharp decline after lag 1, the rest is insignificant (fitting in the blue lin
#q = 0 there is an AR process (gradual decay) with any cutt off.

#A good model for this data would be :
d=0
p=1
q=0

#Therefore parameters are :
  p <-1
  d <- 0
  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()` r `print()` function to print.

```
  # Fitting the ARIMA model
arima_model <- arima(datadeseasonal, order=c(1, 0, 0), include.mean=TRUE)
print(arima_model)
```

```
##
## Call:
## arima(x = datadeseasonal, order = c(1, 0, 0), include.mean = TRUE)
##
## Coefficients:
##          ar1  intercept
##       0.9825   90230.35
## s.e.  0.0120   16957.97
##
## sigma^2 estimated as 30594399:  log likelihood = -2410.59,  aic = 4827.17
```

```r
#trying another model
arima_model1 <- arima(datadeseasonal, order=c(1,0,1), include.mean=TRUE)
print(arima_model1) # (1,0,1) because we don't need to differentiate.
```

```
##
## Call:
## arima(x = datadeseasonal, order = c(1, 0, 1), include.mean = TRUE)
##
## Coefficients:
##          ar1      ma1  intercept
##       0.9905  -0.2047   91200.43
## s.e.  0.0090   0.0850   21811.41
##
## sigma^2 estimated as 29795382:  log likelihood = -2407.51,  aic = 4823.02
```

```r
arima_model2 <- arima(datadeseasonal, order=c(2,0,2), include.mean=TRUE)
print(arima_model2)
```

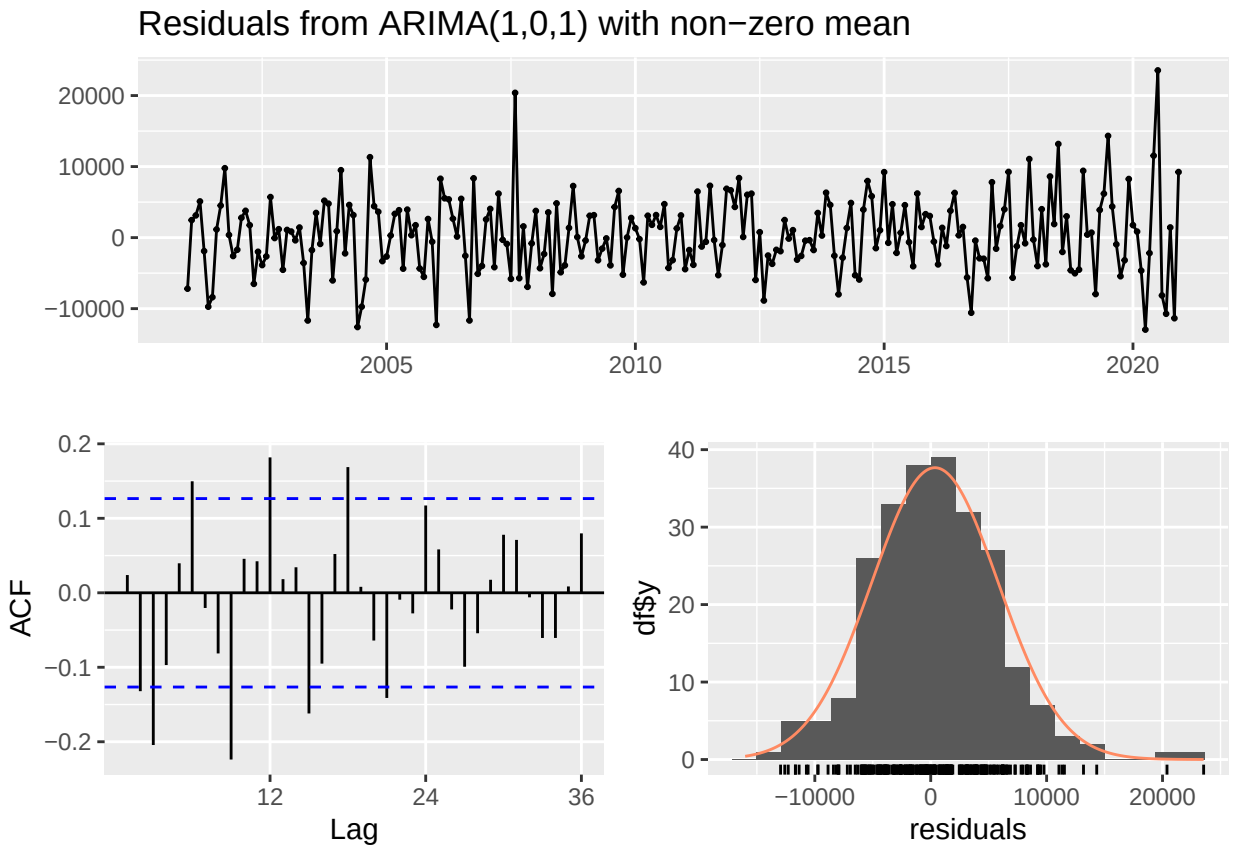
```
##
## Call:
## arima(x = datadeseasonal, order = c(2, 0, 2), include.mean = TRUE)
##
## Coefficients:
##          ar1      ar2      ma1      ma2  intercept
##       1.4742  -0.4752  -0.6988  -0.0697   95358.54
## s.e.  0.1739   0.1734   0.1753   0.0914   34120.64
##
## sigma^2 estimated as 28169257:  log likelihood = -2401.07,  aic = 4814.14
```

After fitting the model based on our parameters, meaning P=1, d=0 and q=0, but ARIMA (1,0,1) model appears to be the best fit out of the three models as it has the lowest AIC. Arima (2,0,2), though with the lowest AIC, it has higher intercept (AR1 greater than 1) and large standard error.

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
# Check the residuals
checkresiduals(arima_model1)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,1) with non-zero mean
## Q* = 74.619, df = 22, p-value = 1.217e-07
##
## Model df: 2.   Total lags used: 24
```

The residual time plot shows some high spikes, which may suggest spacial outliers, i.e. the residuals are not entirely white noise. The histogram displays a similar result. The ACF displays a few lags outside the confidence bounds (above and below 0) and many withing the confidence bounds, this may mean potential autocorrelation at those outbounded lags.

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.

```
#Running the ADF and Mann Kendall tests on original Data

# ADF Test
adf_result1 <- adf.test(tsdata)
```

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

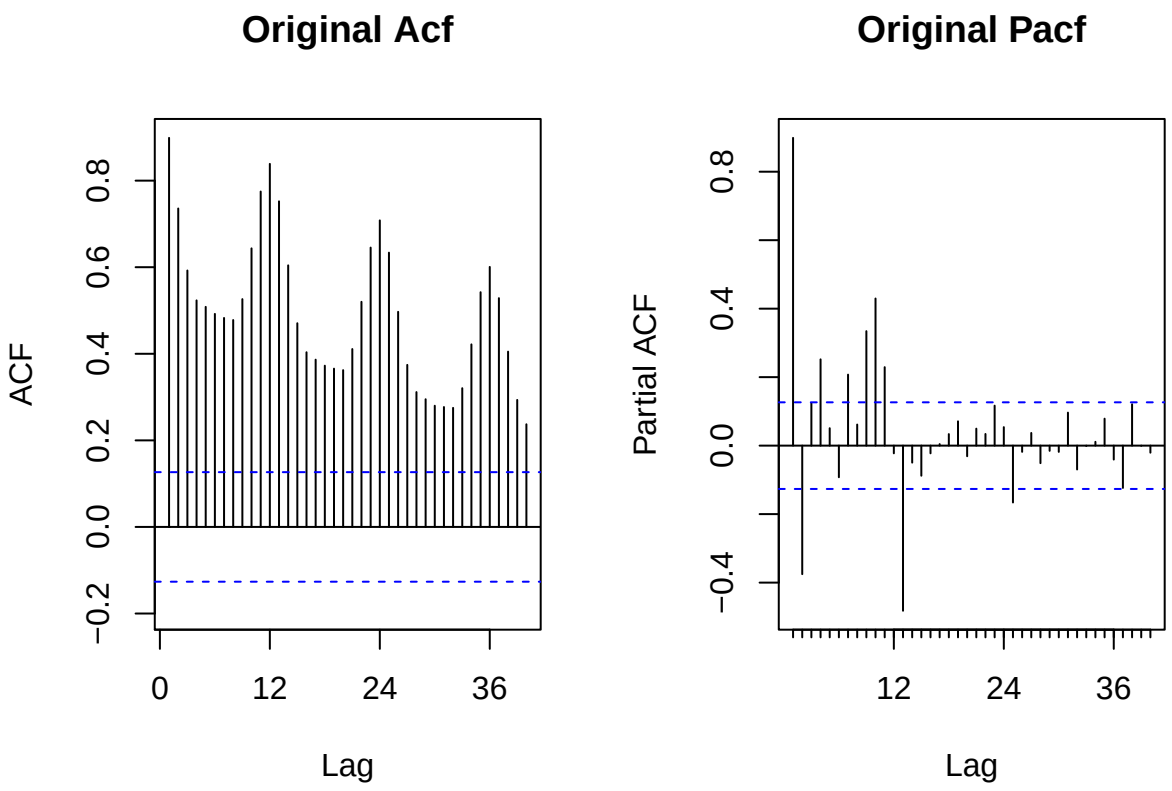
```
print(adf_result1)
```

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

```
# Mann Kendall test
mk_result1 <- MannKendall(tsdata)
print(mk_result1)
```

```
## tau = 0.651, 2-sided pvalue =< 2.22e-16
```

```
# We still can see sationarity and trend in the non seasonal data. Let's estimate the PDQ parameters.

#ACF and PACF of Orignal Data

par(mfrow=c(1,2))
acf_orig <-Acf(tsdata, lag = 40, main = "Original Acf", plot=TRUE)
pacf_orig <-Pacf(tsdata, lag = 40, main = "Original Pacf", plot=TRUE)
```



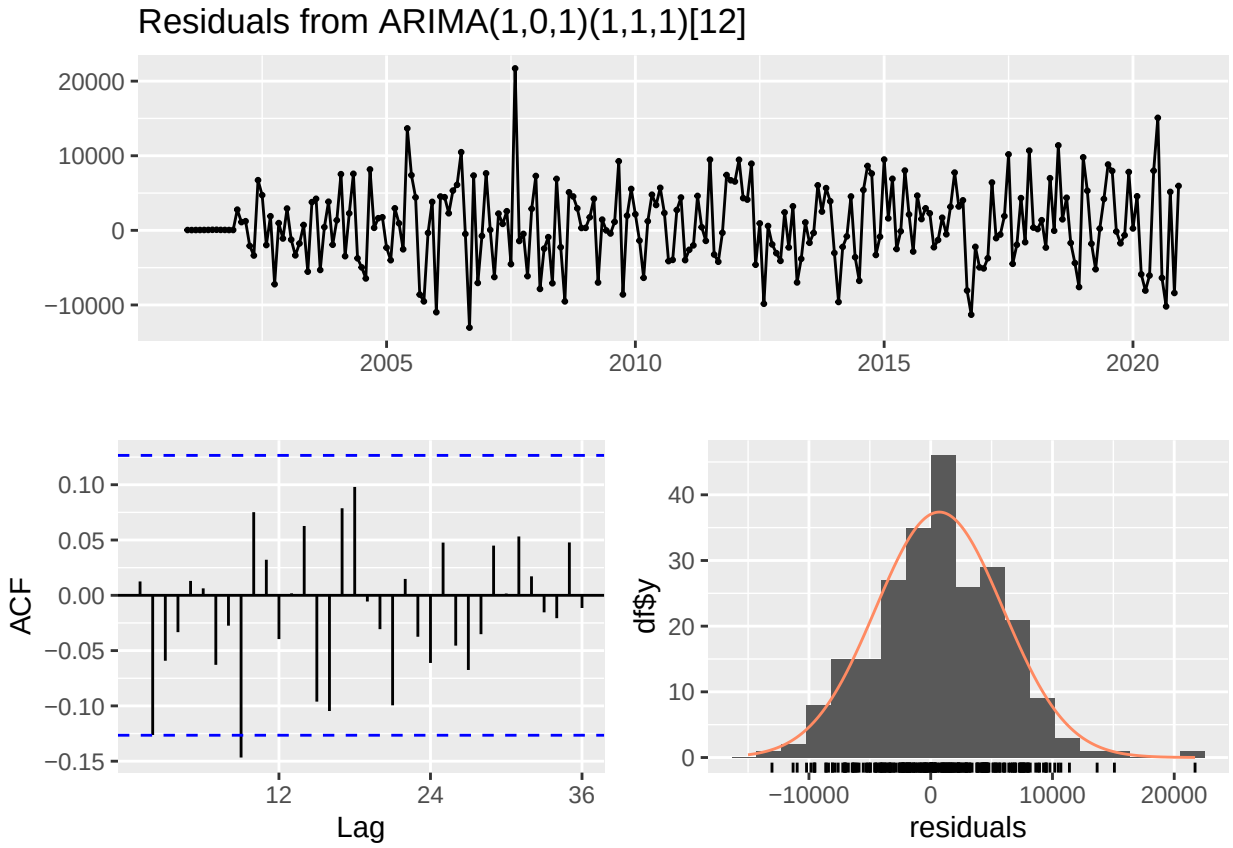
```r
par(mfrow=c(1,1))

#d = 0 (ADF test (-8.9602, p-value = 0.01)) indicating rationarity, need for differentiting.
#D = 1 given the significant spikes in the ACF intervals
#p = 1 given the spike at lag 1
#P = 0 no clear spikes at seasonal lage in the PACF, we can speculate on this value.
#q = 0
#Q = 1 given the significant autocorrelation at first seasonal lag (lag 12)

#Fitting ARIMA model on the seasonal data

arima_seasonal <- Arima(tsdata, order=c(1,0,0), seasonal=c(0,1,1), include.mean=TRUE)
print(arima_seasonal)
```

```
## Series: tsdata
## ARIMA(1,0,0)(0,1,1)[12]
##
## Coefficients:
##          ar1     sma1
##       0.9112  -0.6346
## s.e.  0.0333   0.0650
##
## sigma^2 = 30545109:  log likelihood = -2291.02
## AIC=4588.03   AICc=4588.14   BIC=4598.32
```

```r
#Fitting another model to speculate on the P
arima_seasonal1 <- Arima(tsdata, order=c(1,0,1), seasonal=c(1,1,1), include.mean=TRUE)
print(arima_seasonal1)
```

```
## Series: tsdata
## ARIMA(1,0,1)(1,1,1)[12]
##
## Coefficients:
##          ar1      ma1     sar1     sma1
##       0.9493  -0.2301  -0.0523  -0.6184
## s.e.  0.0264   0.0830   0.1012   0.0921
##
## sigma^2 = 29793856:  log likelihood = -2287.22
## AIC=4584.43   AICc=4584.7   BIC=4601.58
```

```r
#Check the residuals
checkresiduals(arima_seasonal1)
```



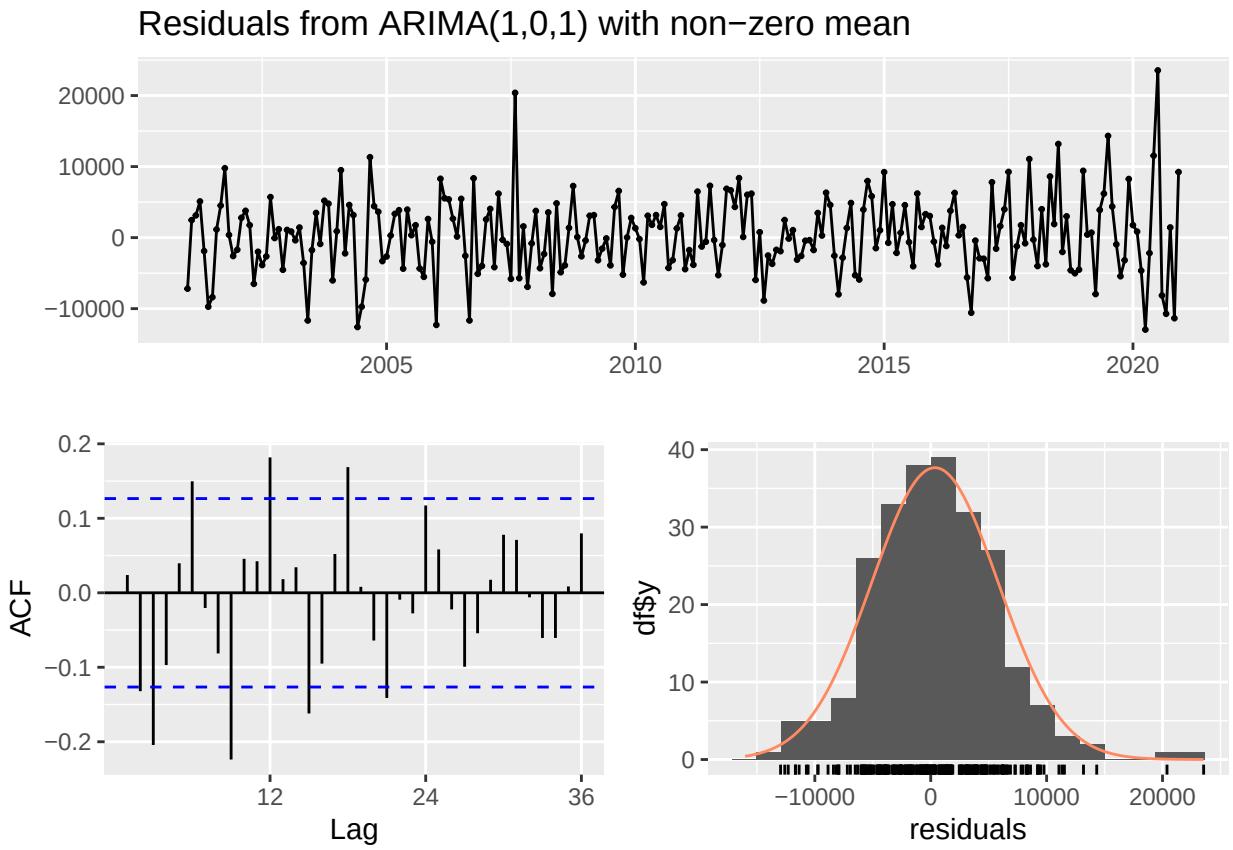

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,1)(1,1,1)[12]
## Q* = 28.467, df = 20, p-value = 0.0988
##
## Model df: 4.   Total lags used: 24
```

Again, a counterintuitive idea appears here. While we suggested the parameters (1,0,0)(0,1,10, the analysis reveals that the model (ARIMA(1,0,1)(1,1,1)[12]) would fit the data better based on lower AIC, AICc, and BIC.

Q8

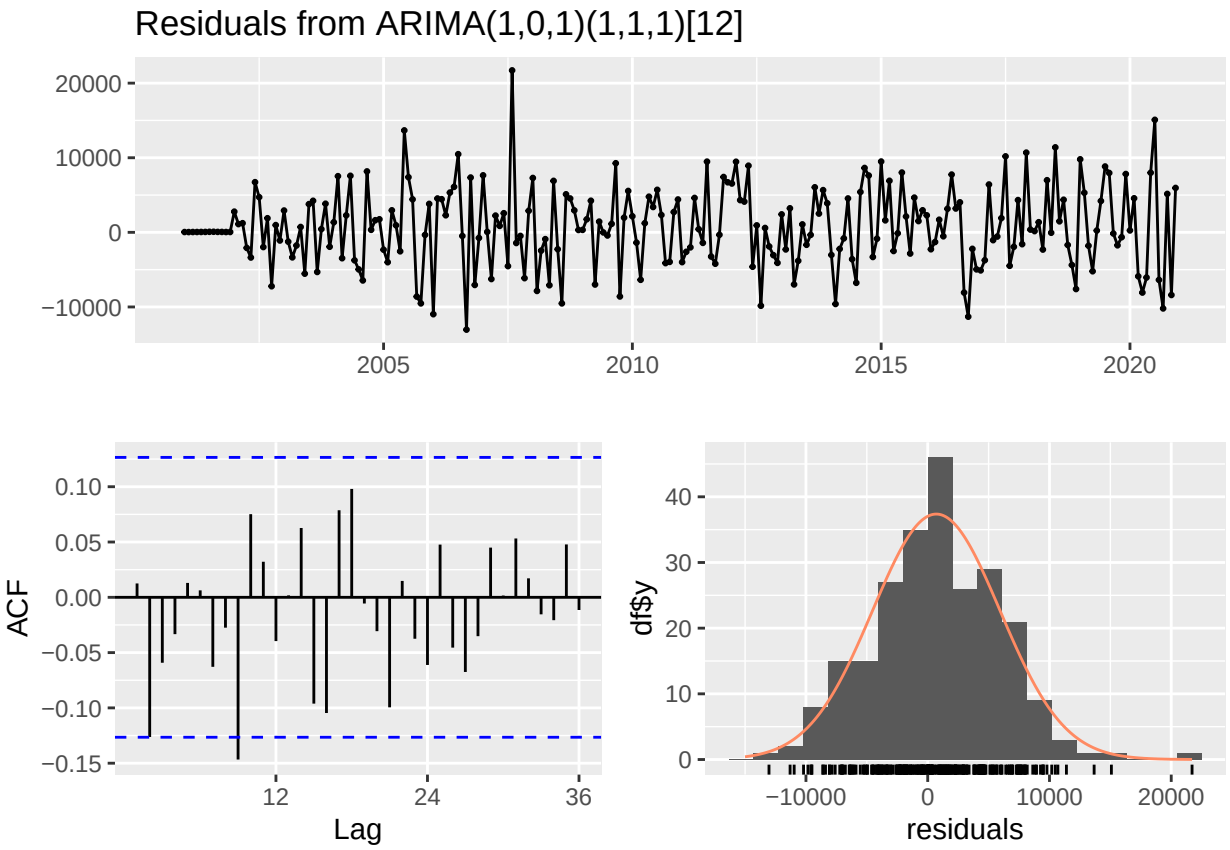
Compare the residual series for Q7 and Q6. Can you tell which ARIMA model is better representing the Natural Gas Series? Is that a fair comparison? Explain your response.

```
#Deseasonal Residuals
checkresiduals(arima_model1)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,1) with non-zero mean
## Q* = 74.619, df = 22, p-value = 1.217e-07
##
## Model df: 2.   Total lags used: 24
```

```
#Seasonal residuals
checkresiduals(arima_seasonal1)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,1)(1,1,1)[12]
## Q* = 28.467, df = 20, p-value = 0.0988
##
## Model df: 4.   Total lags used: 24
```

*Comparison

For the first plot (ARIMA(1,0,1)): The time plot shows residuals around zero with some outliers. The ACF plot shows several spikes outside the blue line (confidence intervals), suggesting some autocorrelation. For the second plot (ARIMA(1,0,1)(1,1,1)[12]): The time plot again shows residuals around zero, with fewer outliers. The ACF plot shows fewer spikes outside blue lines, suggesting data might be closer to white noise.

Therefore, the seasonal ARIMA model (ARIMA(1,0,1)(1,1,1)[12]) seems to have residuals that are closer to white noise, as indicated by fewer significant spikes in the ACF plot.

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?

```r
autoarima_deseas <- auto.arima(datadeseasonal)
print(autoarima_deseas)
```

```
## Series: datadeseasonal
## 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
```

I got ARIMA (1,0,1) and auto.arima gives me (1,1,1). However, I did not try this model (1,1,1) because I couldn't defferentiate series after they are deseasoned. That is why, trying other models in Q5, I tried ARIMA(1,0,1) because I knew we don't need to differentiate the series after deseasoning.

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
autoarima_origin <- auto.arima(tsdata)
print(autoarima_origin)
```

```
## Series: tsdata
## 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
```

I got this one. However, the function include.drift = TRUE was not running. I therefore used include.mean=TRUE.