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

Assignment 7 - Due date 03/07/24

Student Name

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

```r
#Load/install required package here
library(forecast)
```

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

```r
library(tseries)
library(ggplot2)
library(Kendall)
library(lubridate)
```

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

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

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr   1.1.3     v stringr 1.5.0
## v forcats 1.0.0     v tibble  3.2.1
## v purrr   1.0.2     v tidyr   1.3.0
## v readr   2.1.4
```

```
## -- 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 error
```

```r
library(ggthemes)
library(sarima)
```

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

```r
library(dplyr)
library(cowplot)
```

```
##
## Attaching package: 'cowplot'
##
## The following object is masked from 'package:ggthemes':
##
##     theme_map
##
## 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
#read in the data
net_generation_data <- read.csv(
  file = "./Data/Net_generation_United_States_all_sectors_monthly.csv",
  skip = 5,
  header = FALSE,
  dec = ".",
  sep=",",
  stringsAsFactors = TRUE)

#assign column names
colnames(net_generation_data) <- read.csv( file = "./Data/Net_generation_United_States_all_sectors_montl
                          nrows = 1,
                          skip = 4,
                          header = FALSE)
#check that data are imported correctly
head(net_generation_data)
```

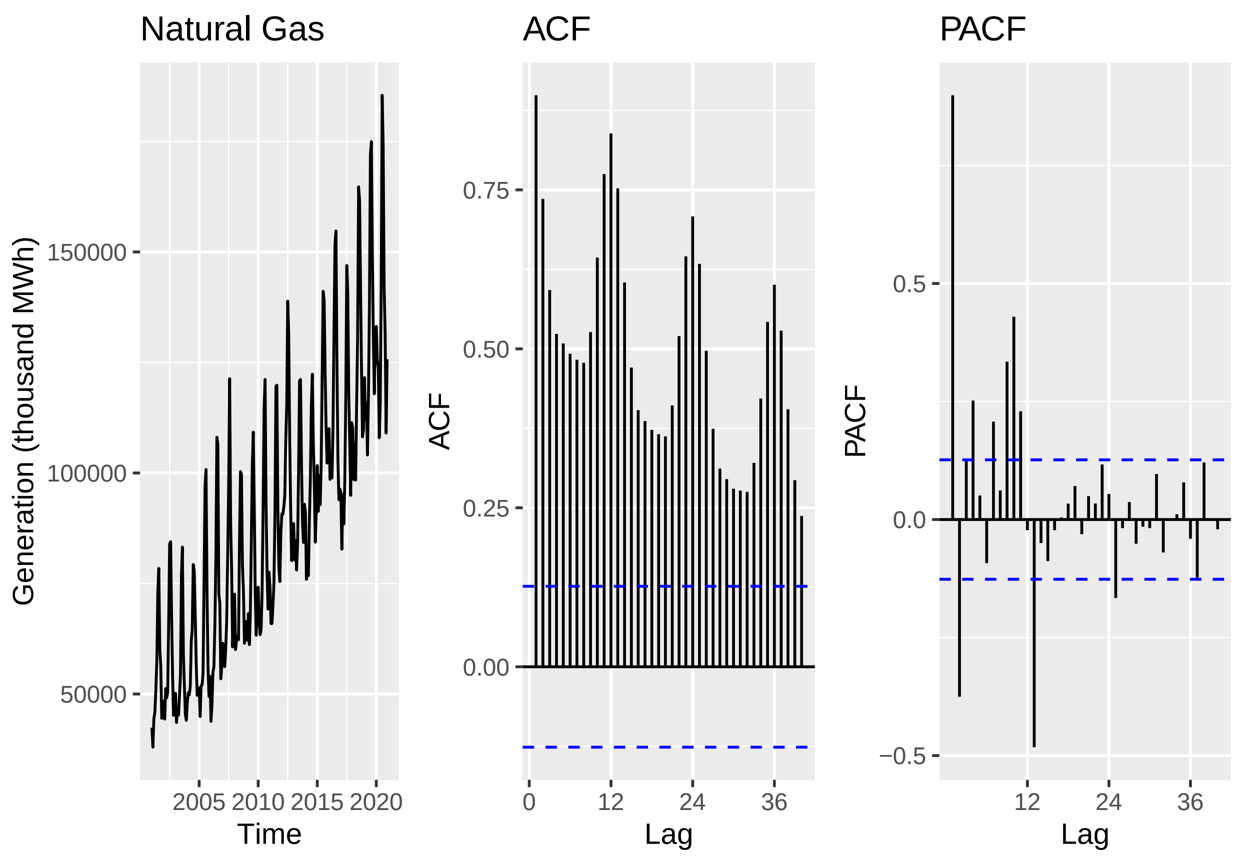
```
##      Month all fuels (utility-scale) thousand megawatthours
## 1 Dec 2020                                        344970.4
## 2 Nov 2020                                        302701.8
## 3 Oct 2020                                        313910.0
## 4 Sep 2020                                        334270.1
## 5 Aug 2020                                        399504.2
## 6 Jul 2020                                        414242.5
##   coal thousand megawatthours natural gas thousand megawatthours
## 1                    78700.33                           125703.7
## 2                    61332.26                           109037.2
## 3                    59894.57                           131658.2
## 4                    68448.00                           141452.7
## 5                    91252.48                           173926.6
## 6                    89831.36                           185444.8
##   nuclear thousand megawatthours
## 1                       69870.98
## 2                       61759.98
## 3                       59362.46
## 4                       65727.32
## 5                       68982.19
## 6                       69385.44
##   conventional hydroelectric thousand megawatthours
## 1                                          23086.37
## 2                                          21831.88
## 3                                          18320.72
## 4                                          19161.97
## 5                                          24081.57
## 6                                          27675.94
```

```r
#natural gas
ng <- as.data.frame(net_generation_data[,4])
num_rows <- nrow(ng)

#making the data start from 2001
reversed_ng <- ng[nrow(ng):1, ]
```

```r
#create time series object for natural gas
ng_ts <-ts(reversed_ng, start=c(2001,1), frequency=12) #question: why are there no more decimals?
```

```r
#ACF and PACF
plot_grid(
  autoplot(ng_ts, main = "Natural Gas", ylab= "Generation (thousand MWh)"),
  autoplot(Acf(ng_ts, lag = 40, plot=FALSE),
                main = "ACF"),
   autoplot(Pacf(ng_ts, lag = 40, plot=FALSE),
                main = "PACF"),
  ncol=3
  )
```

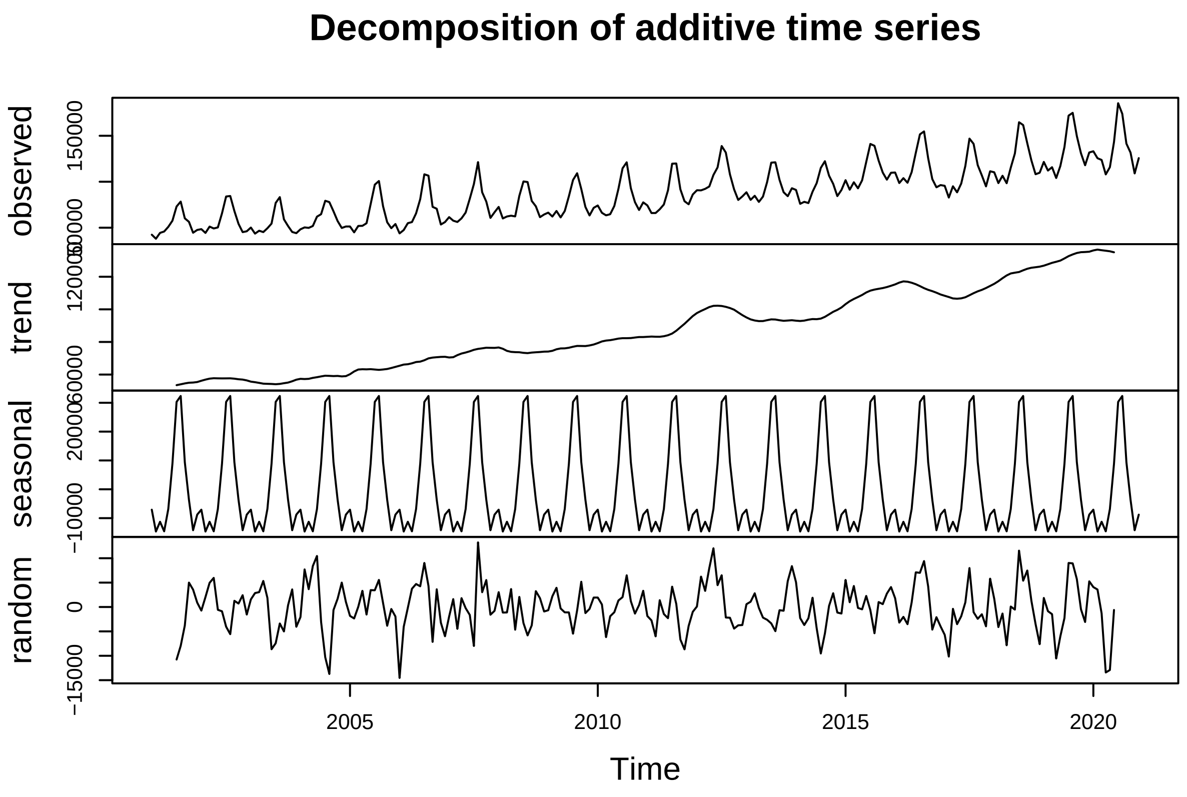
```
## Warning in ggplot2::geom_segment(lineend = "butt", ...): Ignoring unknown parameters: ‘main‘
## Ignoring unknown parameters: ‘main‘
```



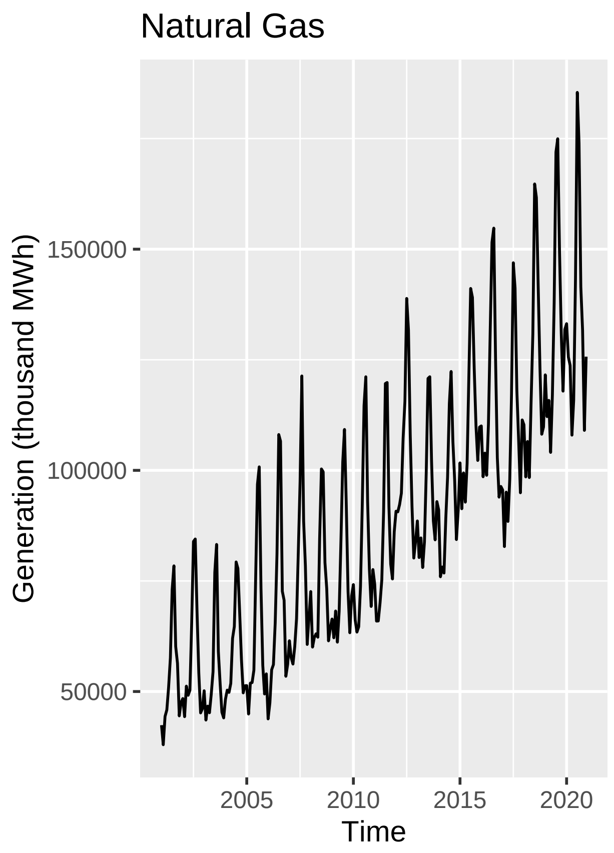
Q2

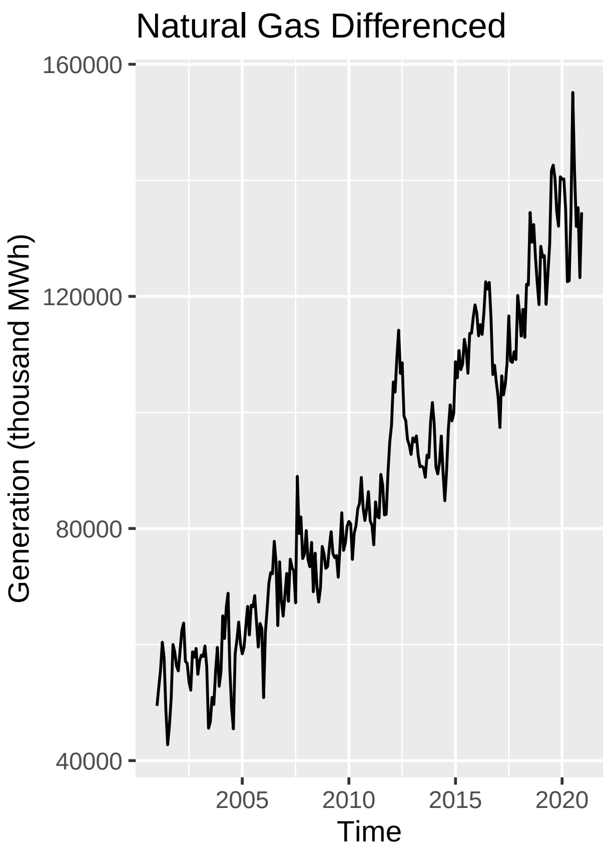
Using the *decompose*() or *stl*() and the *seasadj*() functions create a series without the seasonal component, i.e., a deseasonalized natural gas series. Plot the deseasonalized series over time and corresponding ACF and PACF. Compare with the plots obtained in Q1.

```
#can use additive because our seasonal component is not changing in magnitude with time?
ng_decomposed_a <- decompose(ng_ts, type = "additive")
plot(ng_decomposed_a)
```



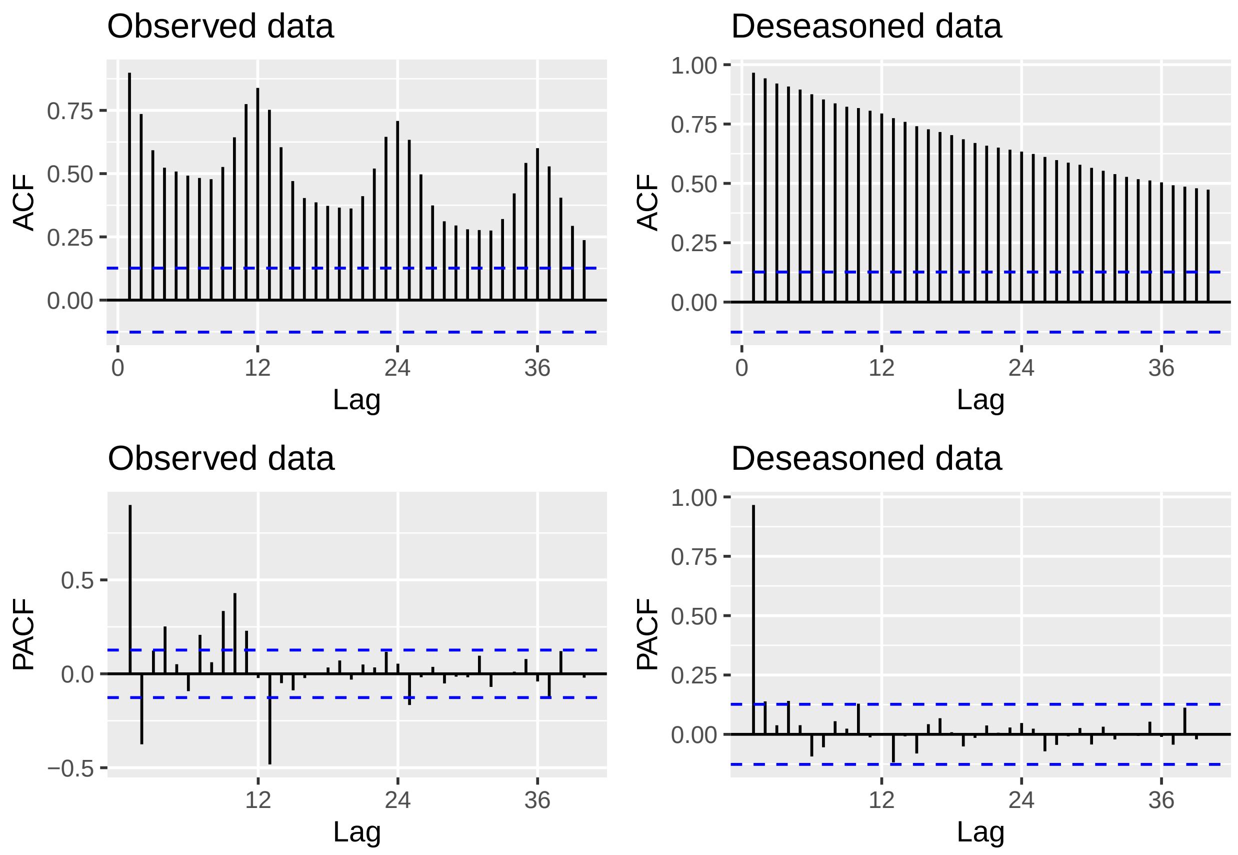
```
#Creating non-seasonal natural gas generation ts
#seasonally adjusted data constructed by removing the seasonal component from the decomposed function
#helps specify p and q
deseasonal_ng <- seasadj(ng_decomposed_a)



#comparing observed data
plot_grid(
  autoplot(ng_ts, main = "Natural Gas", ylab= "Generation (thousand MWh)"),
  autoplot(deseasonal_ng, main = "Natural Gas Differenced", ylab= "Generation (thousand MWh)"))
```





```r
#comparing ACFs and PACFs
plot_grid(
  autoplot(Acf(ng_ts, lag = 40, plot=FALSE),
                main = "Observed data"),
  autoplot(Acf(deseasonal_ng, lag = 40, plot=FALSE),
                main = "Deseasoned data"),
  autoplot(Pacf(ng_ts, lag = 40, plot=FALSE),
                main = "Observed data"),
  autoplot(Pacf(deseasonal_ng, lag = 40, plot=FALSE),
                main = "Deseasoned data"),
  ncol=2
)
```

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



Answer: The new ACF plot show a slow decay which is a sign of non-stationarity. Need to difference the data! But we can confirm if there is non-stationarity with the ADF and MK tests.

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 test
print(adf.test(deseasonal_ng))
```

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

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

```r
#Run MK test
print(summary(MannKendall(deseasonal_ng)))
```

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

> Answer: The p-value for the ADF test is 0.01 which is less than 0.05 so we can reject the null hypothesis. The data are either stationary or deterministic. The p-value for the MK test is much less than 0.05 so we can reject the null for the MK test. This means that the data follow a deterministic trend. We can use the MK test instead of the SMK because we have deseasoned the data. Becuase we have confirmed that the data have a trend, we need to difference our data.

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.

```r
#identifying differencing (d) model parameters
#Lets difference the series once at lag 1 to remove the trend.
deseasonal_ng_diff <- diff(deseasonal_ng,differences=1,lag=1)

#now we can run the MK and ADF tests again to see if d=1 was enough to remove the trend
#Run ADF test
print(adf.test(deseasonal_ng_diff))
```

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

##
##  Augmented Dickey-Fuller Test
##
## data:  deseasonal_ng_diff
## Dickey-Fuller = -6.9137, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```r
#Run MK test
print(summary(MannKendall(deseasonal_ng_diff)))
```

```
## Score =  -299 , Var(Score) = 1526334
## denominator =  28441
## tau = -0.0105, 2-sided pvalue =0.80939
## NULL
```

```r
ndiffs(deseasonal_ng)
```

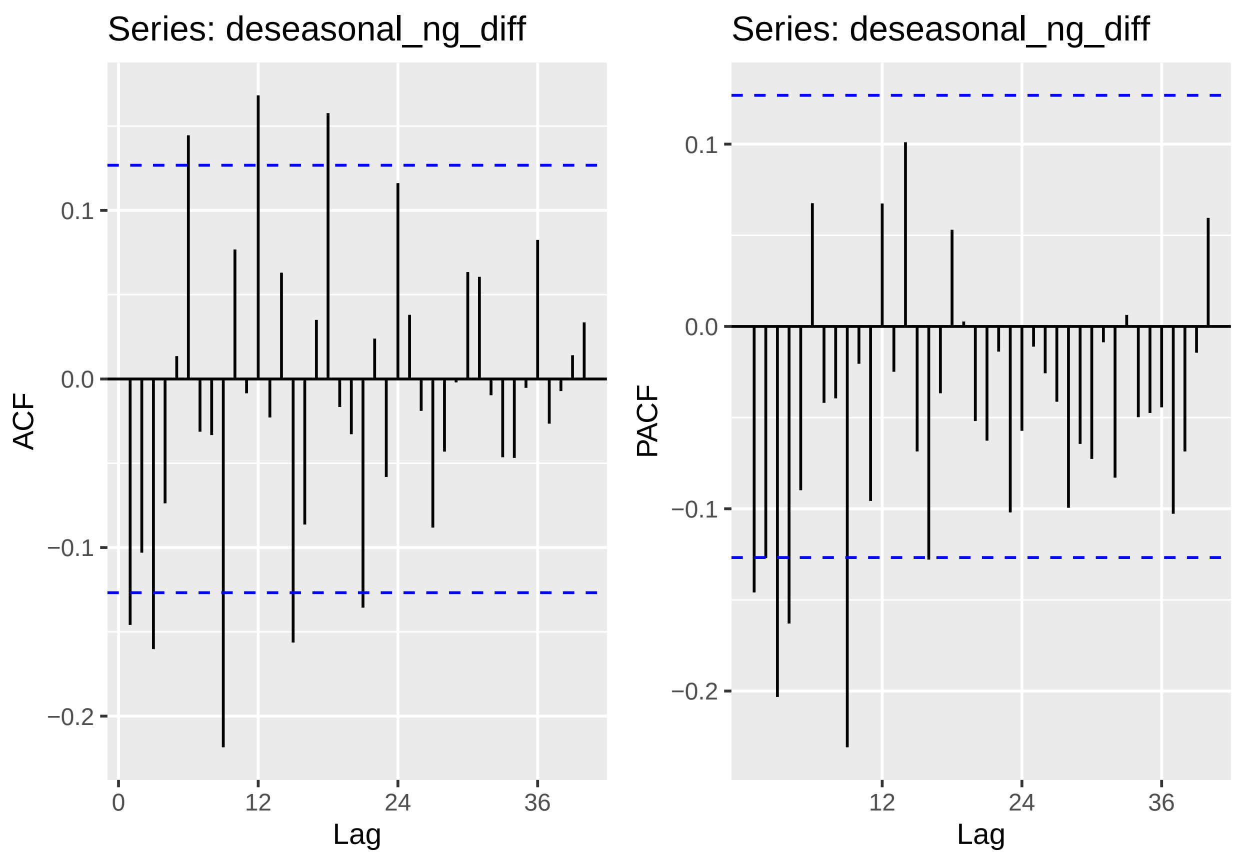
```
## [1] 1
```

```r
ndiffs(deseasonal_ng_diff)
```

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

Based on the results from q3 I differenced the data once. Then I ran the tests again to see if one difference was enough. The MK test produces a p-value of 0.8, which is greater than 0.05 so we cannot reject the null hypothesis that the data are stationary. This means that differencing once was enough. This is confirmed when using the ndiffs() function. On the deseasonal data it tells us we only need to difference once. After differencing once, it returns we don't need to difference anymore! We know that d=1. Now we can examine p and q.

```r
#ACF and PACF for differenced and desesaoned data
deseason_difference_acf <- autoplot(Acf(deseasonal_ng_diff, lag = 40, plot=FALSE))
deseason_difference_pacf <- autoplot(Pacf(deseasonal_ng_diff, lag = 40, plot=FALSE))

plot_grid(
 deseason_difference_acf,
 deseason_difference_pacf)
```

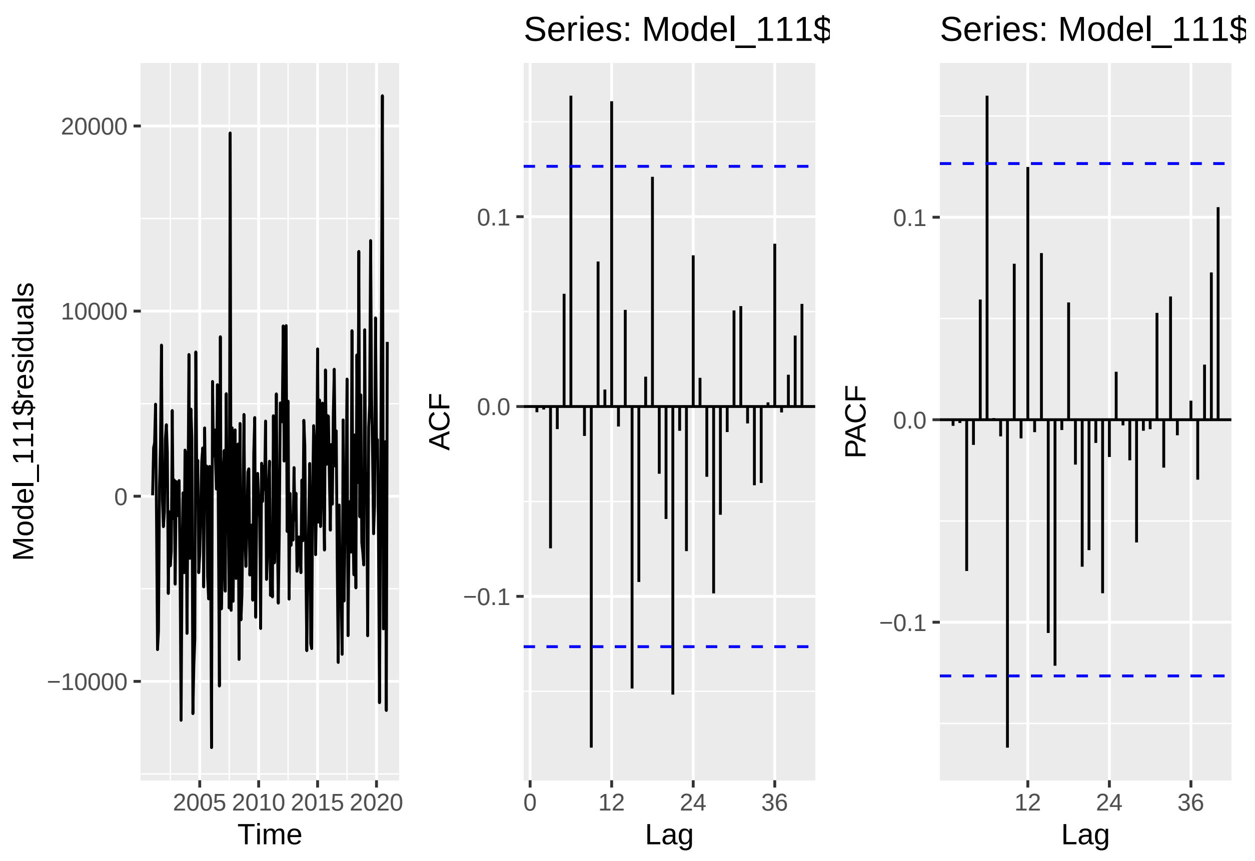


Looking at the ACF and PACF for the differenced and deseasoned data there is no obvious cut off or slow decay for either of the ACF or PACF plots... Let's try some different ARIMA models and see what sticks. Likely there are both MA and AR components.

```r
#so lets try ARIMA(1,1,1) on the non-seasonal generation ng data before differencing
Model_111 <- Arima(deseasonal_ng,order=c(1,1,1), include.drift=TRUE)
print(Model_111)
```

```
## Series: deseasonal_ng
## 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
```

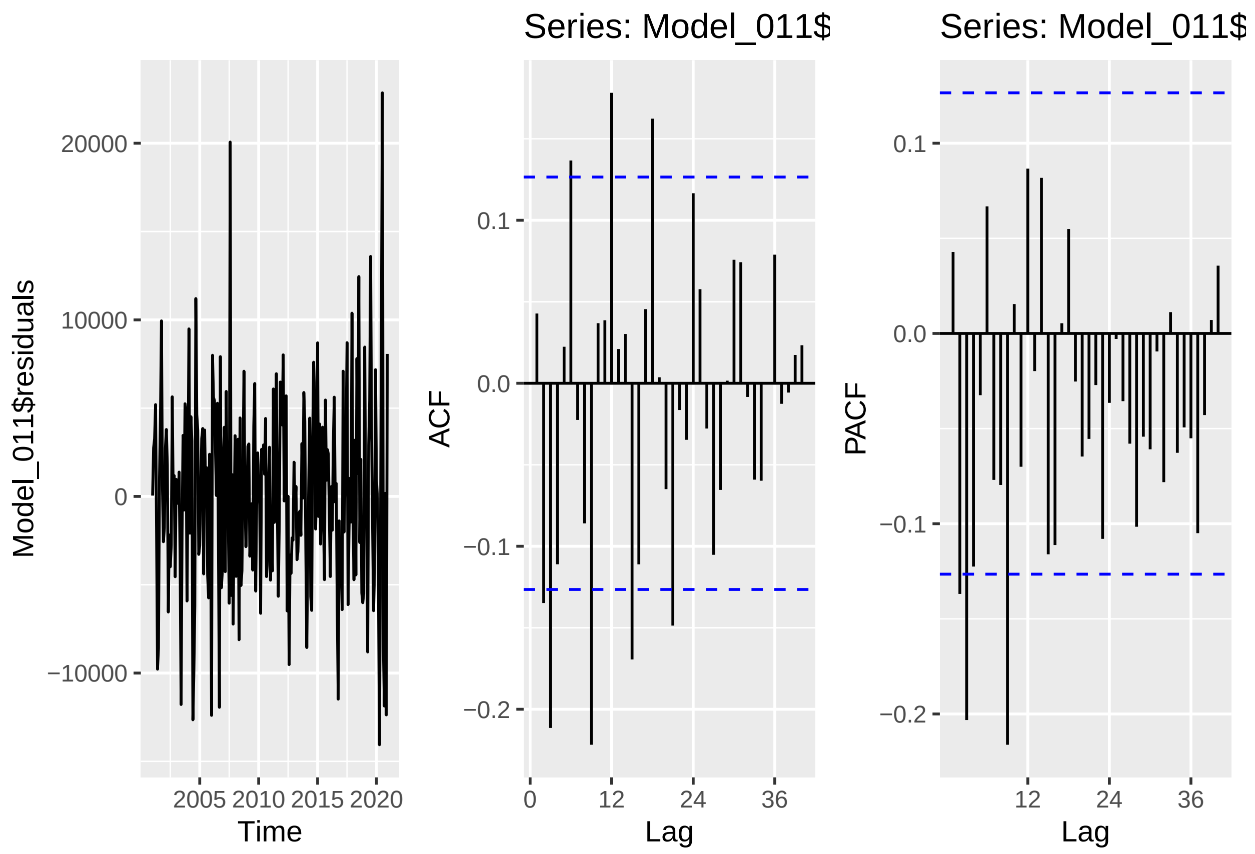
```r
compare_aic <- data.frame(Model_111$aic)

#Check residuals series, if white noise we got a good fit
plot_grid(
  autoplot(Model_111$residuals), #want white noise, mean to be centered at 0
  autoplot(Acf(Model_111$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_111$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```



```r
#Now let's try ARIMA(0,1,1) #differencing data and have a MA term
Model_011 <- Arima(deseasonal_ng,order=c(0,1,1),include.drift=TRUE)
print(Model_011)
```

```
## Series: deseasonal_ng
## ARIMA(0,1,1) with drift
##
## Coefficients:
##           ma1     drift
##       -0.2296  344.5131
## s.e.   0.0866  271.9198
##
## sigma^2 = 29946343:  log likelihood = -2395.33
## AIC=4796.66   AICc=4796.77   BIC=4807.09
```

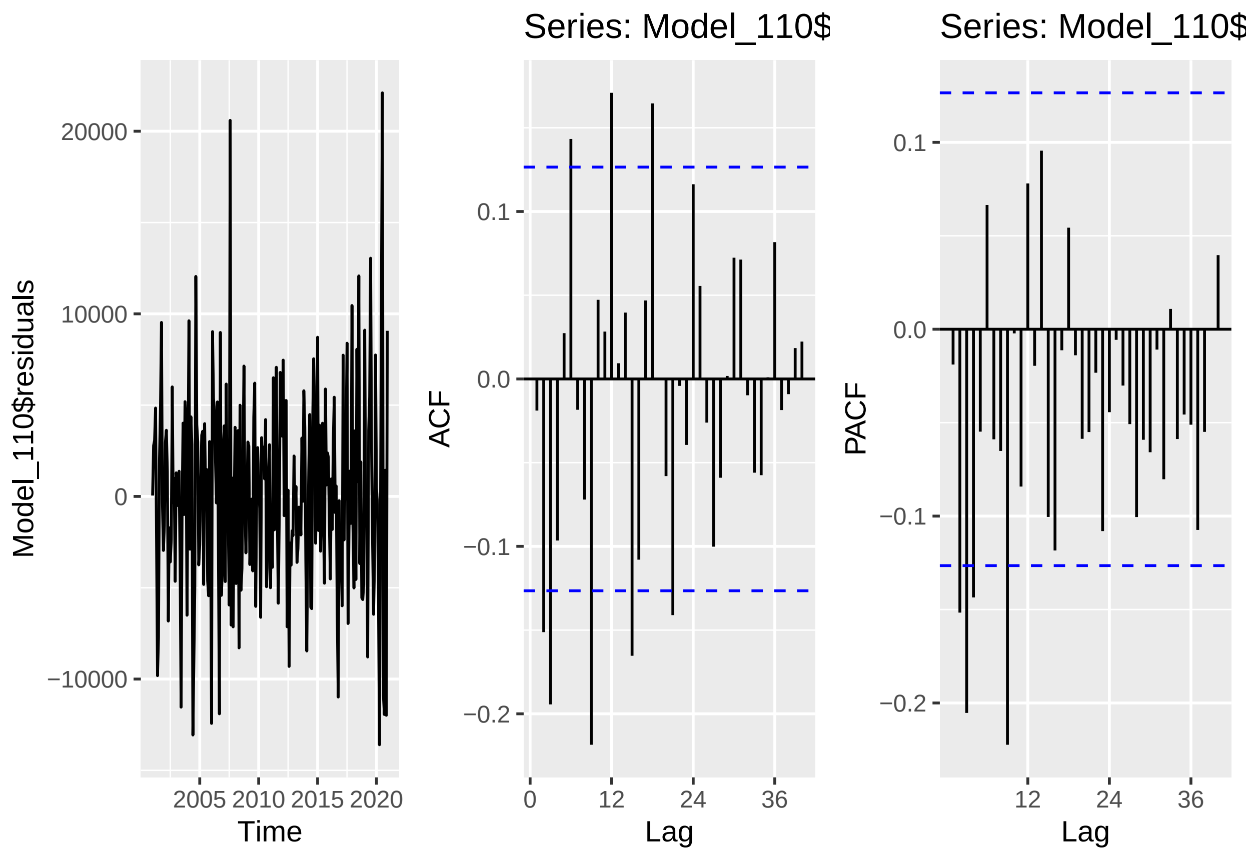
```r
compare_aic <- data.frame(compare_aic,Model_011$aic)

plot_grid(
  autoplot(Model_011$residuals),
  autoplot(Acf(Model_011$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_011$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```



```r
#Now let's try ARIMA(1,1,0) #differencing data and have a AR term
Model_110 <- Arima(deseasonal_ng,order=c(1,1,0),include.drift=TRUE)
print(Model_110)
```

```
## Series: deseasonal_ng
## 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
```

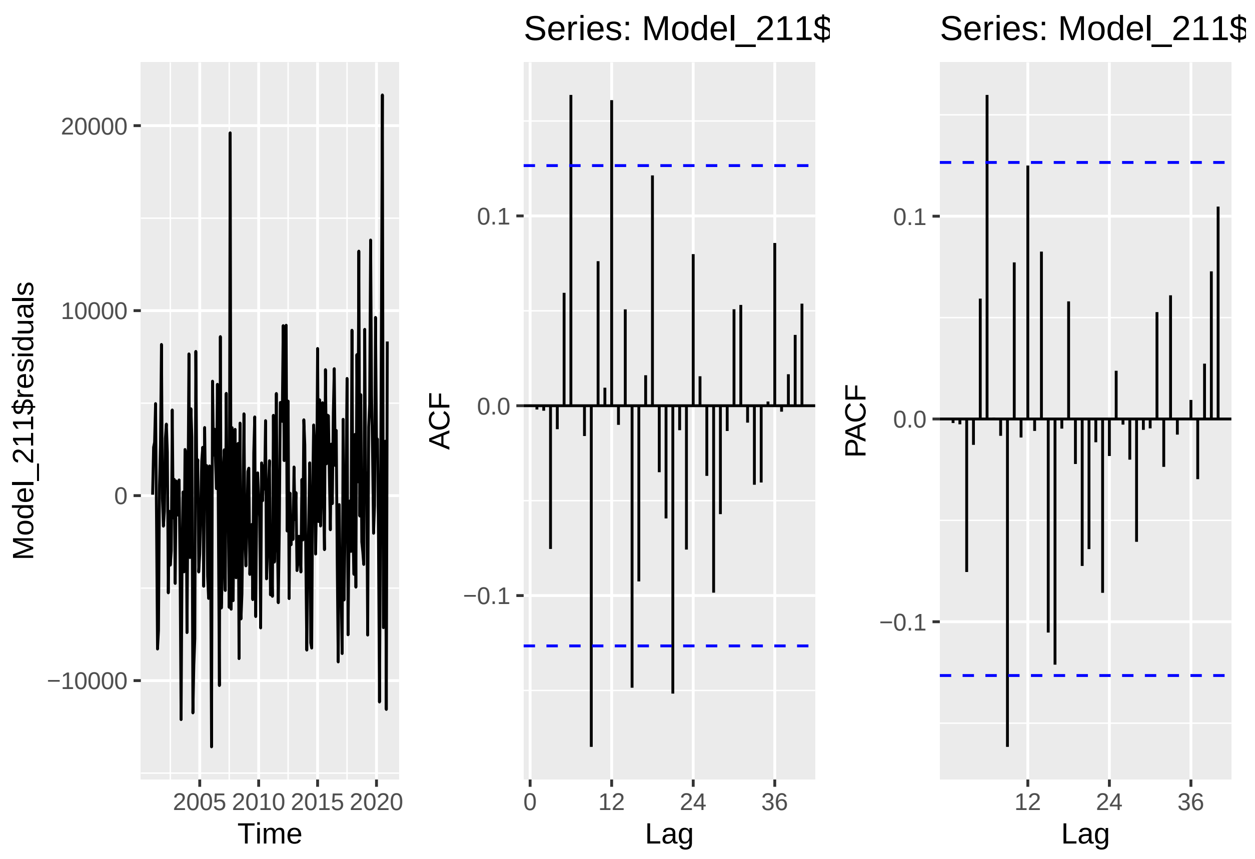
```r
compare_aic <- data.frame(compare_aic,Model_110$aic)

plot_grid(
  autoplot(Model_110$residuals),
  autoplot(Acf(Model_110$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_110$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```



```r
#Now let's try ARIMA(2,1,1)
Model_211 <- Arima(deseasonal_ng,order=c(2,1,1),include.drift=TRUE)
print(Model_211)
```

```
## Series: deseasonal_ng
## ARIMA(2,1,1) with drift
##
## Coefficients:
##          ar1     ar2      ma1     drift
##       0.7057  0.0017  -0.9798  359.4921
## s.e.  0.0710  0.0707   0.0360   29.3046
##
## sigma^2 = 27094287:  log likelihood = -2383.11
## AIC=4776.21   AICc=4776.47   BIC=4793.59
```

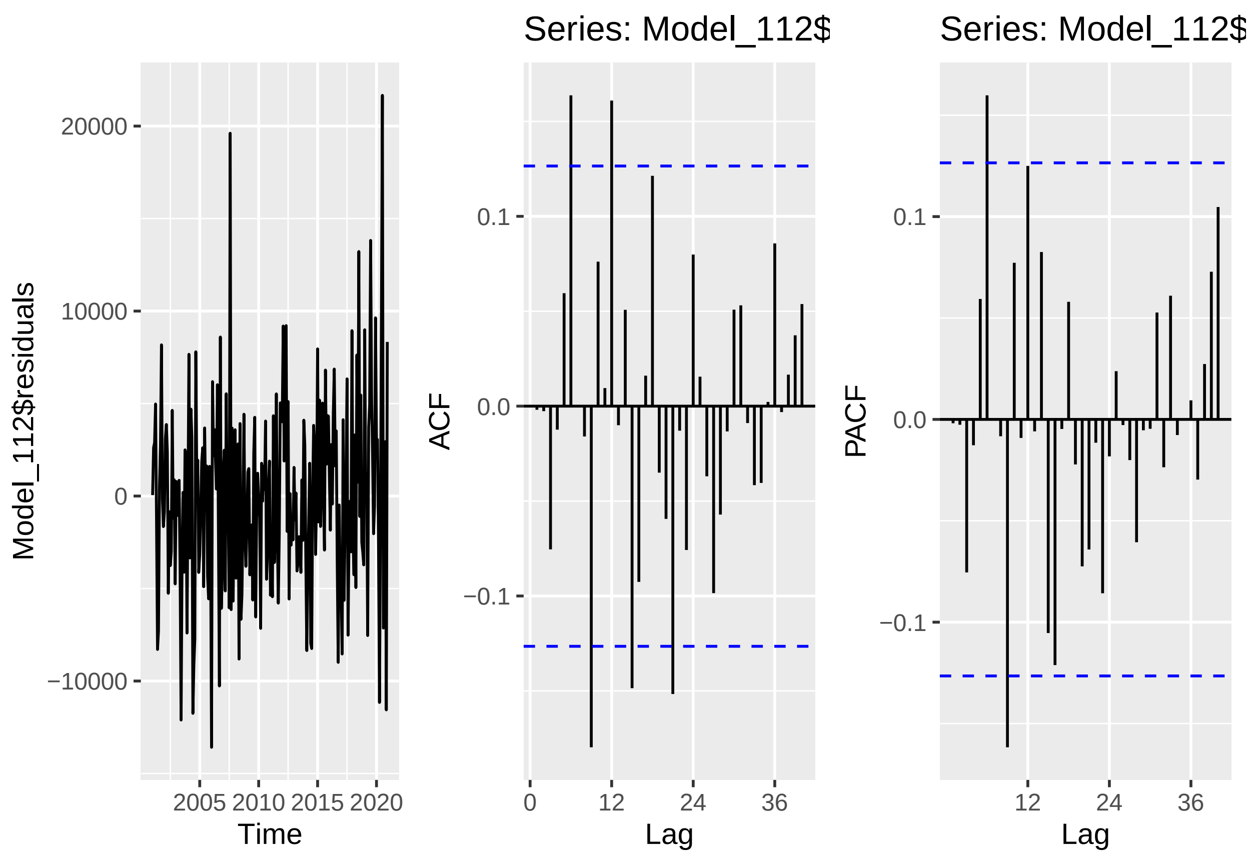
```r
compare_aic <- data.frame(compare_aic,Model_211$aic)

plot_grid(
  autoplot(Model_211$residuals),
  autoplot(Acf(Model_211$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_211$residuals,lag.max=40, plot = FALSE)),
  nrow=1)
```



```
#we still see some significant coefficients outside the blue line, so we can keep incorperating more te

#Now let's try ARIMA(1,1,2)
Model_112 <- Arima(deseasonal_ng,order=c(1,1,2),include.drift=TRUE)
print(Model_112)
```

```
## Series: deseasonal_ng
## ARIMA(1,1,2) with drift
##
## Coefficients:
##          ar1      ma1     ma2     drift
##       0.7081  -0.9823  0.0025  359.4980
## s.e.  0.0939   0.1189  0.0982   29.3122
##
## sigma^2 = 27094333:  log likelihood = -2383.11
## AIC=4776.21   AICc=4776.47   BIC=4793.59
```

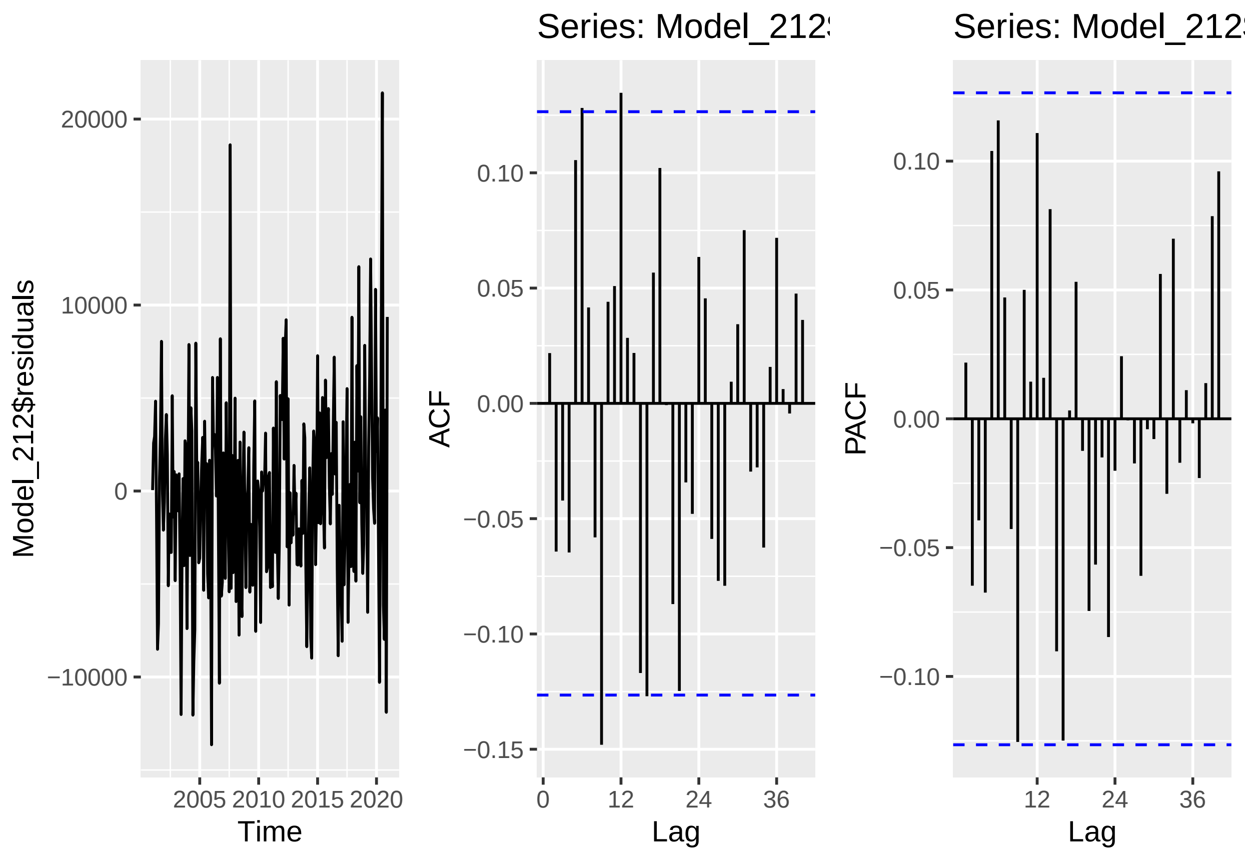
```
compare_aic <- data.frame(compare_aic,Model_112$aic)

plot_grid(
  autoplot(Model_112$residuals),
  autoplot(Acf(Model_112$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_112$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```



```r
#Now let's try ARIMA(2,1,2)
Model_212 <- Arima(deseasonal_ng,order=c(2,1,2),include.drift=TRUE)
print(Model_212)
```

```
## Series: deseasonal_ng
## ARIMA(2,1,2) with drift
##
## Coefficients:
##           ar1     ar2      ma1      ma2     drift
##       -0.2339  0.7177  -0.0574  -0.9425  359.3591
## s.e.   0.0541  0.0478   0.0494   0.0484   17.2526
##
## sigma^2 = 26505822:  log likelihood = -2381.07
## AIC=4774.13   AICc=4774.49   BIC=4794.99
```

```r
compare_aic <- data.frame(compare_aic,Model_212$aic)

plot_grid(
  autoplot(Model_212$residuals),
  autoplot(Acf(Model_212$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_212$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```



```r
#yay everything is nearly within the blue lines! can stop adding terms.

#Let's check AIC
print(compare_aic)
```

```
##   Model_111.aic Model_011.aic Model_110.aic Model_211.aic Model_112.aic
## 1      4774.213      4796.664      4799.075      4776.212      4776.212
##   Model_212.aic
## 1      4774.131
```

> Looking at all of the iterations of difference p and q values, the model with p=2 ,q=2 and d=1 has the lowest AIC and the most promising ACF and PACF plots (with all coefficients nearly within the blue lines). Additionally our residuals look like white noise centered around a mean of 0. (There is no serial correlation or time dependence of errors, and insignificant coefficients)

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.

```r
Model_212 <- Arima(deseasonal_ng,order=c(2,1,2),include.drift=TRUE, include.mean = TRUE)
print(Model_212)
```

```
## Series: deseasonal_ng
## ARIMA(2,1,2) with drift
##
## Coefficients:
##           ar1     ar2      ma1      ma2     drift
##       -0.2339  0.7177  -0.0574  -0.9425  359.3591
## s.e.   0.0541  0.0478   0.0494   0.0484   17.2526
##
## sigma^2 = 26505822:  log likelihood = -2381.07
## AIC=4774.13   AICc=4774.49   BIC=4794.99
```

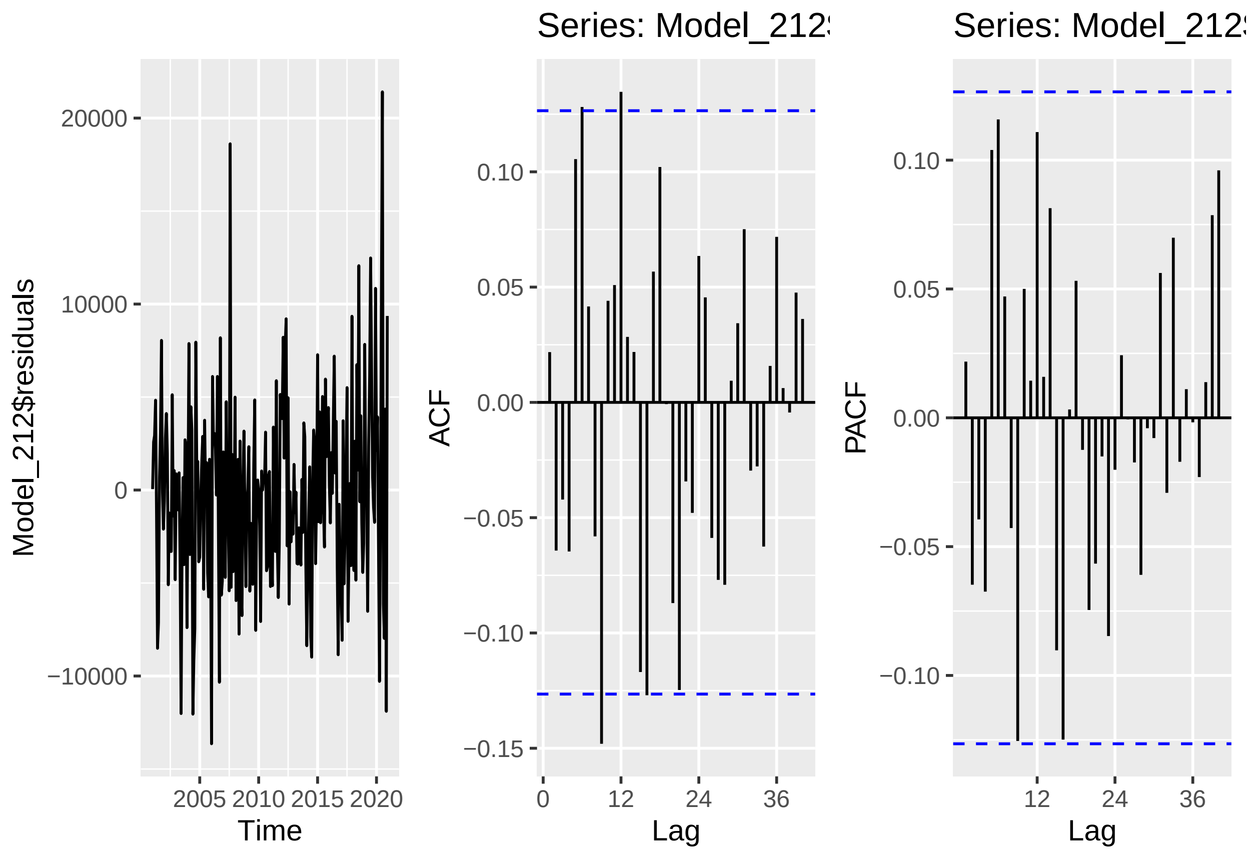
```r
cat("phi_1 is:" , Model_212$coef[1],
    "\nphi_2 is:", Model_212$coef[2],
    "\ntheta_1 is:", Model_212$coef[3],
    "\ntheta_2 is:", Model_212$coef[4],
    "\nmiu (drift) is:", Model_212$coef[5])
```

```
## phi_1 is: -0.2339103
## phi_2 is: 0.7177003
## theta_1 is: -0.05740088
## theta_2 is: -0.942483
## miu (drift) is: 359.3591
```

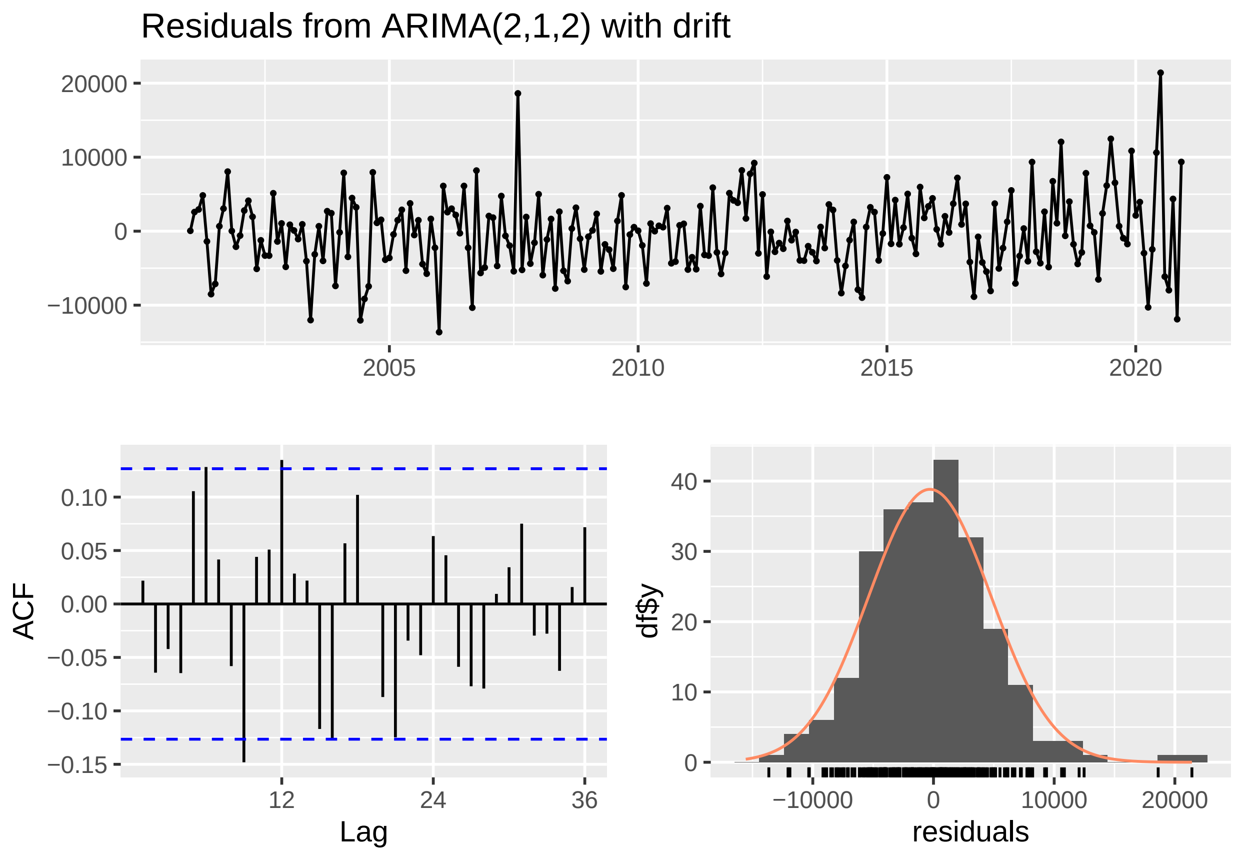
Q6

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
plot_grid(
  autoplot(Model_212$residuals),
  autoplot(Acf(Model_212$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_212$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```



```r
checkresiduals(Model_212)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,2) with drift
## Q* = 41.717, df = 20, p-value = 0.00301
##
## Model df: 4.   Total lags used: 24
```

> Answer: They do look like white noise, centered at zero. There is no pattern to the variation between maximums and minimums.

Modeling the original series (with seasonality)

Q7

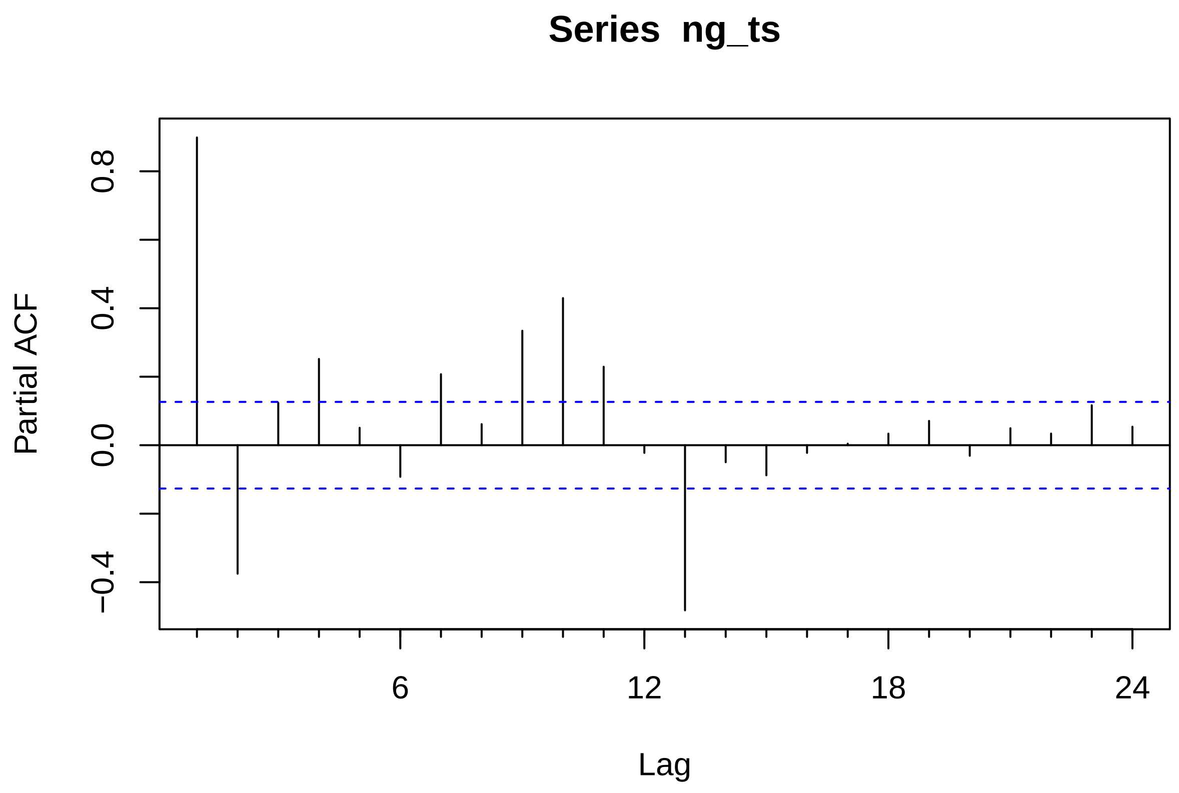
Repeat Q4-Q6 for the original series (the complete series that has the seasonal component). Note that when you model the seasonal series, you need to specify the seasonal part of the ARIMA model as well, i.e., P, D and Q.

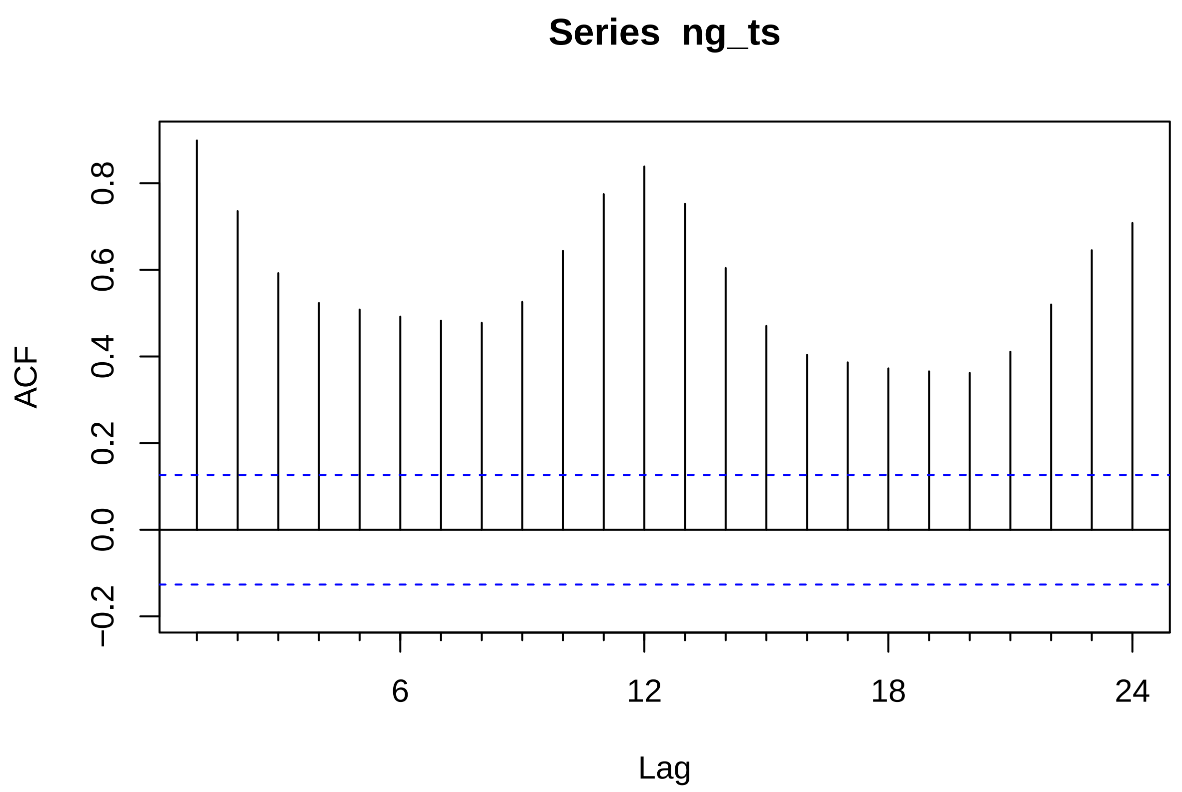
> Answer: We see multiple spikes at seas. lag for ACF and a single spike for PACF (shown below), which is indicitative of an SAR component! We know that P + Q can't be more than 1, so I will test ARIMA models with P= 1. We get that we need 1 seasonal difference with the nsdiffs() function (shown below). The PACF and ACF aren't indicitive of an MA process, but I can test for Q =1 with the ARIMA function to be sure.

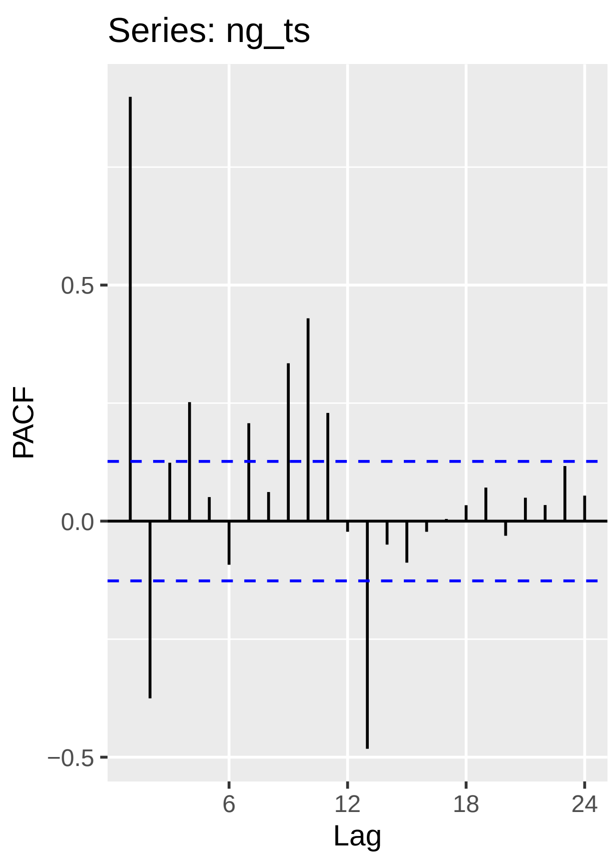
```r
D <- nsdiffs(ng_ts)
cat("The number of seasonal diffs (D) needed to achieve stationarity is:", D)
```

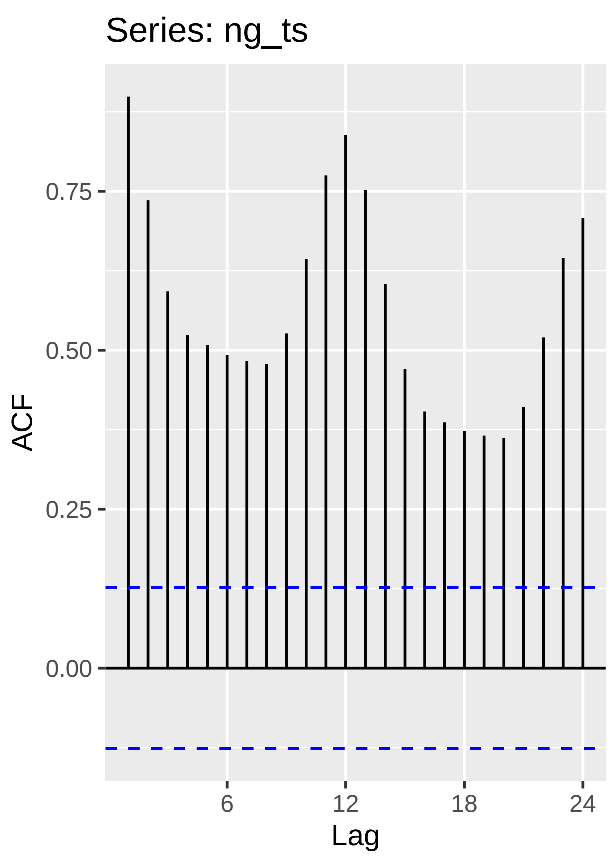
```
## The number of seasonal diffs (D) needed to achieve stationarity is: 1
```

```r
plot_grid(
autoplot(Pacf(ng_ts)),
autoplot(Acf(ng_ts)))
```




Series ng_ts
ACF
−0.2
0.0
0.2
0.4
0.6
0.8
6
12
18
24
Lag





```
Model_212110 <- Arima(ng_ts, order=c(2,1,2), seasonal =c(1,1,0), include.drift=FALSE)
print(Model_212110)
```

```
## Series: ng_ts
## ARIMA(2,1,2)(1,1,0)[12]
##
## Coefficients:
##           ar1      ar2     ma1      ma2     sar1
##       -0.2327  -0.0445  0.0041  -0.1543  -0.4483
## s.e.   0.4061   0.3144  0.4030   0.3489   0.0600
##
## sigma^2 = 34615357:  log likelihood = -2291.31
## AIC=4594.62   AICc=4595   BIC=4615.17
```

```
compare_aic <- data.frame(compare_aic,Model_212110$aic)

Model_212011 <- Arima(ng_ts, order=c(2,1,2), seasonal =c(0,1,1), include.drift=FALSE)
print(Model_212011)
```

```
## Series: ng_ts
## ARIMA(2,1,2)(0,1,1)[12]
##
## Coefficients:
##           ar1     ar2      ma1      ma2     sma1
##       -0.2162  0.7057  -0.0489  -0.9156  -0.7165
## s.e.   0.0834  0.0648   0.0767   0.0737   0.0563
##
## sigma^2 = 28013060:  log likelihood = -2271.66
## AIC=4555.33   AICc=4555.71   BIC=4575.88
```

```r
compare_aic <- data.frame(compare_aic,Model_212011$aic)

Model_111110 <- Arima(ng_ts, order=c(1,1,1), seasonal =c(1,1,0), include.drift=FALSE)
print(Model_111110)
```

```
## Series: ng_ts
## ARIMA(1,1,1)(1,1,0)[12]
##
## Coefficients:
##          ar1      ma1     sar1
##       0.7722  -1.0000  -0.4526
## s.e.  0.0432   0.0213   0.0595
##
## sigma^2 = 32606986:  log likelihood = -2287.56
## AIC=4583.12   AICc=4583.3   BIC=4596.82
```

```r
compare_aic <- data.frame(compare_aic,Model_111110$aic)

Model_111011 <- Arima(ng_ts, order=c(1,1,1), seasonal =c(0,1,1), include.drift=FALSE)
print(Model_111011)
```

```
## Series: ng_ts
## ARIMA(1,1,1)(0,1,1)[12]
##
## Coefficients:
##          ar1      ma1     sma1
##       0.7323  -0.9819  -0.7017
## s.e.  0.0504   0.0183   0.0563
##
## sigma^2 = 27922085:  log likelihood = -2272.2
## AIC=4552.39   AICc=4552.57   BIC=4566.09
```

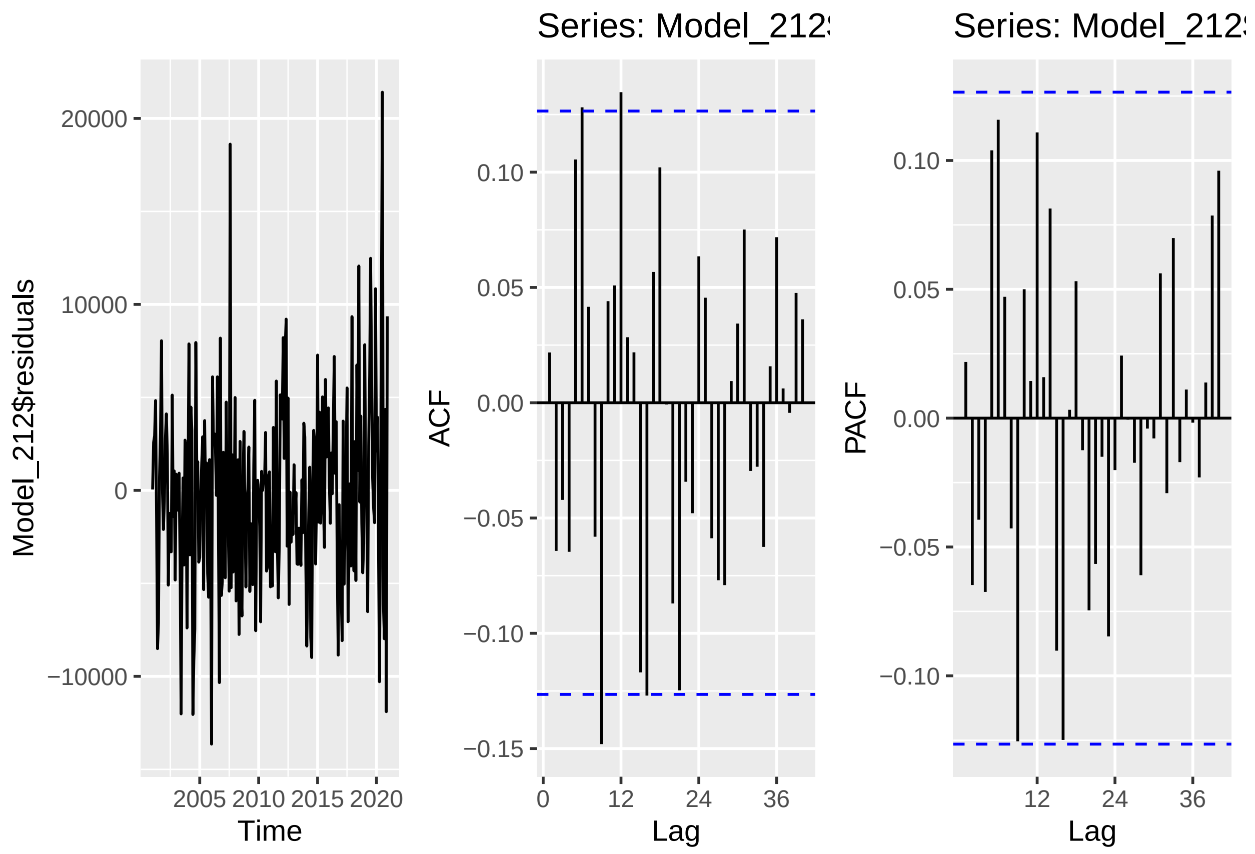
```r
compare_aic <- data.frame(compare_aic,Model_111011$aic)

print(compare_aic)
```

```
##   Model_111.aic Model_011.aic Model_110.aic Model_211.aic Model_112.aic
## 1      4774.213      4796.664      4799.075      4776.212      4776.212
##   Model_212.aic Model_212110.aic Model_212011.aic Model_111110.aic
## 1      4774.131         4594.622         4555.329         4583.117
##   Model_111011.aic
## 1         4552.393
```

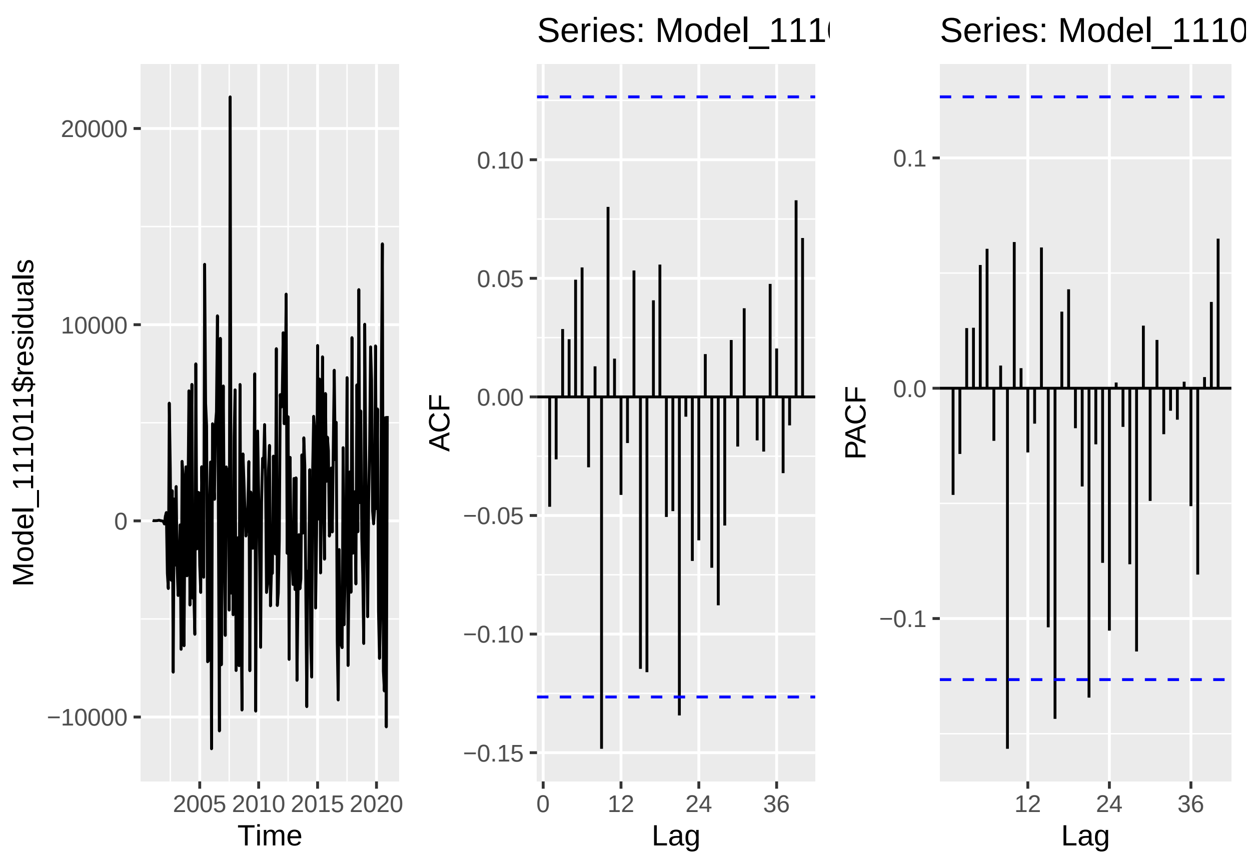
Q8

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.

```r
#from q6
plot_grid(
  autoplot(Model_212$residuals),
  autoplot(Acf(Model_212$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_212$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```



```r
#from q7
plot_grid(
  autoplot(Model_111011$residuals),
  autoplot(Acf(Model_111011$residuals,lag.max=40, plot = FALSE)),
  autoplot(Pacf(Model_111011$residuals,lag.max=40, plot = FALSE)),
  nrow=1
)
```



Answer: I can't tell which is better. Both of the residuals are white noise centered at 0. The residuals are all variable within the same magnitude for both series. Also, we are removing variability by removing the seasonal component, so it's not fair to compare errors with seasonal and non-seasonal data because they have different variability.

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?

```
deseasoned_autofit <- auto.arima(deseasonal_ng)
print(deseasoned_autofit)
```

```
## Series: deseasonal_ng
## 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
```

> Answer: When we let R choose the model parameter for us the order of the best ARIMA model is (1,1,1) with drift. That does NOT match what I specified in Q4 (2,1,2).

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?

```
deseasoned_autofit_og <- auto.arima(ng_ts)
print(deseasoned_autofit_og)
```

```
## Series: ng_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
```

> Answer: R gives model parameters (1,0,0)(0,1,1)[12] with drift for the original series. This also does NOT match what I specified in q7 (1,1,1)(0,1,1)[12].