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

Assignment 7 - Due date 03/07/24

Cara Kuuskvere

Directions

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)
library(ggplot2)
library(forecast)
library(Kendall)
library(tseries)
library(outliers)
library(tidyverse)
library(cowplot)
#install.packages("smooth")
library(smooth)
```

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**.

```
data_raw <- read.csv(file="Data/Net_generation_United_States_all_sectors_monthly.csv",header=TRUE,skip=

NG_data <- c(data_raw[,1],data_raw[,4])

#Preparing the data - create date object and rename columns
data_processed <-
  data_raw %>%
  mutate( Month = my(Month) ) %>%
  rename( All.sectors = all.fuels..utility.scale..thousand.megawatthours ) %>%
  rename( Coal.k.MWh = coal.thousand.megawatthours ) %>%
  rename( NG.k.MWh = natural.gas.thousand.megawatthours ) %>%
  rename( Nuc.k.MWh = nuclear.thousand.megawatthours ) %>%
```

```
  rename( Hydro.k.MWh = conventional.hydroelectric.thousand.megawatthours ) %>%
  arrange( Month )

head(data_processed)
```

```
##        Month All.sectors Coal.k.MWh NG.k.MWh Nuc.k.MWh Hydro.k.MWh
## 1 2001-01-01    332493.2   177287.1 42388.66  68707.08    18852.05
## 2 2001-02-01    282940.2   149735.5 37966.93  61272.41    17472.89
## 3 2001-03-01    300706.5   155269.0 44364.41  62140.71    20477.19
## 4 2001-04-01    278078.9   140670.7 45842.75  56003.03    18012.99
## 5 2001-05-01    300491.6   151592.9 50934.21  61512.44    19175.63
## 6 2001-06-01    327694.0   162615.8 57603.15  68023.10    20727.63
```

```
summary(data_processed)
```

```
##      Month              All.sectors       Coal.k.MWh        NG.k.MWh
##  Min.   :2001-01-01   Min.   :276127   Min.   : 40624   Min.   : 37967
##  1st Qu.:2005-12-24   1st Qu.:309142   1st Qu.:113769   1st Qu.: 62245
##  Median :2010-12-16   Median :328297   Median :141665   Median : 84415
##  Mean   :2010-12-16   Mean   :336382   Mean   :135391   Mean   : 88028
##  3rd Qu.:2015-12-08   3rd Qu.:357671   3rd Qu.:161751   3rd Qu.:108385
##  Max.   :2020-12-01   Max.   :421797   Max.   :190135   Max.   :185445
##    Nuc.k.MWh      Hydro.k.MWh
##  Min.   :54547   Min.   :14743
##  1st Qu.:62651   1st Qu.:19568
##  Median :65775   Median :22307
##  Mean   :66037   Mean   :22639
##  3rd Qu.:70438   3rd Qu.:25475
##  Max.   :74649   Max.   :32607
```

```
# Checking for missing data
Days <- as.data.frame(seq.Date(from=as.Date("2001/01/01"),to=as.Date("2020/12/01"), by ="month"))
colnames(Days) <- "Month"


new_data <-left_join(Days, data_processed, by="Month")

new_data <- na.exclude(new_data)
#no NAs!
```

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.

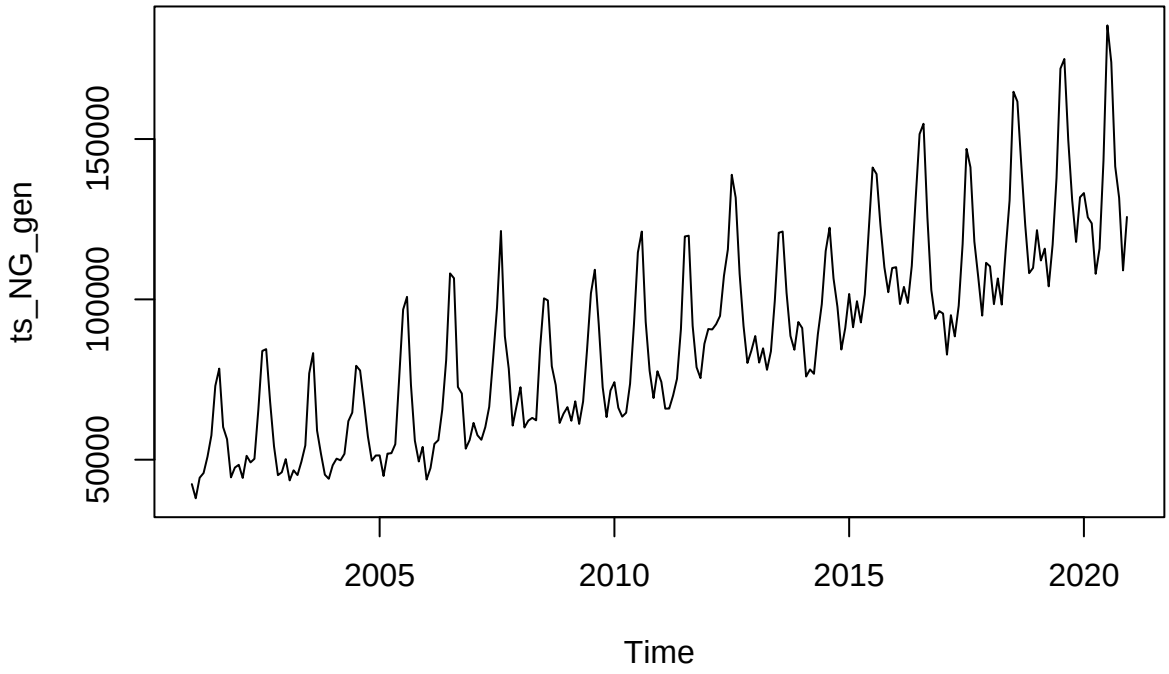
```
# create the time series for our dataset
ts_NG_gen <- ts(data_processed$NG.k.MWh, start=c(year(data_processed$Month[1]),month(data_processed$Mon
                          frequency=12)

head(ts_NG_gen,12)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2001 42388.66 37966.93 44364.41 45842.75 50934.21 57603.15 73030.14 78409.80
##           Sep      Oct      Nov      Dec
## 2001 60181.14 56376.44 44490.62 47540.86
```

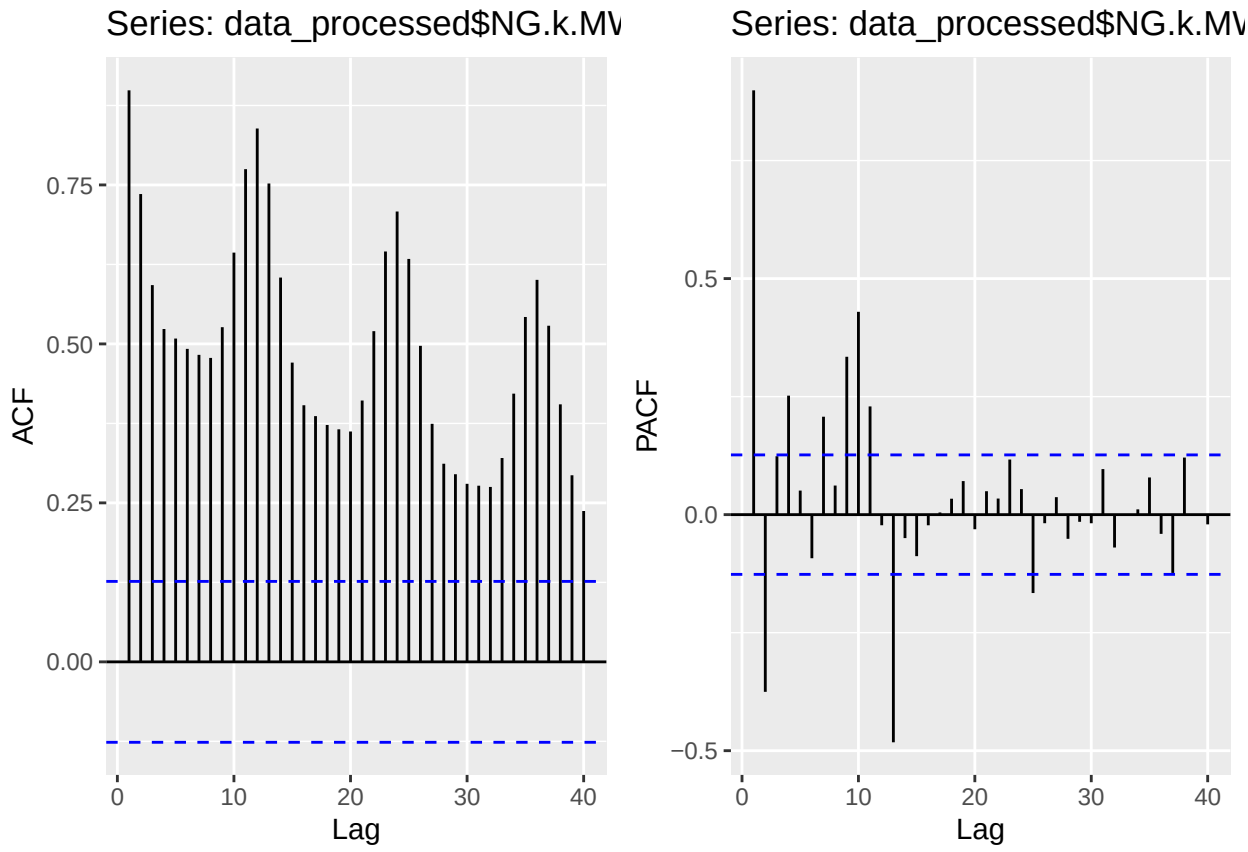
```
tail(ts_NG_gen,12)
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2020 133157.6 125593.9 123697.0 107960.0 115870.9 143245.4 185444.8 173926.6
##           Sep      Oct      Nov      Dec
## 2020 141452.7 131658.2 109037.2 125703.7
```

```
#TS plot
ts.plot(ts_NG_gen)
```




```
#ACF and PACF plots
plot_grid(
autoplot(Acf(data_processed$NG.k.MWh, lag= 40, main = "ACF Natural Gas Gen", plot=FALSE)),
autoplot(Pacf(data_processed$NG.k.MWh, lag= 40, main = "PACF Natural Gas Gen", plot= FALSE)))
```

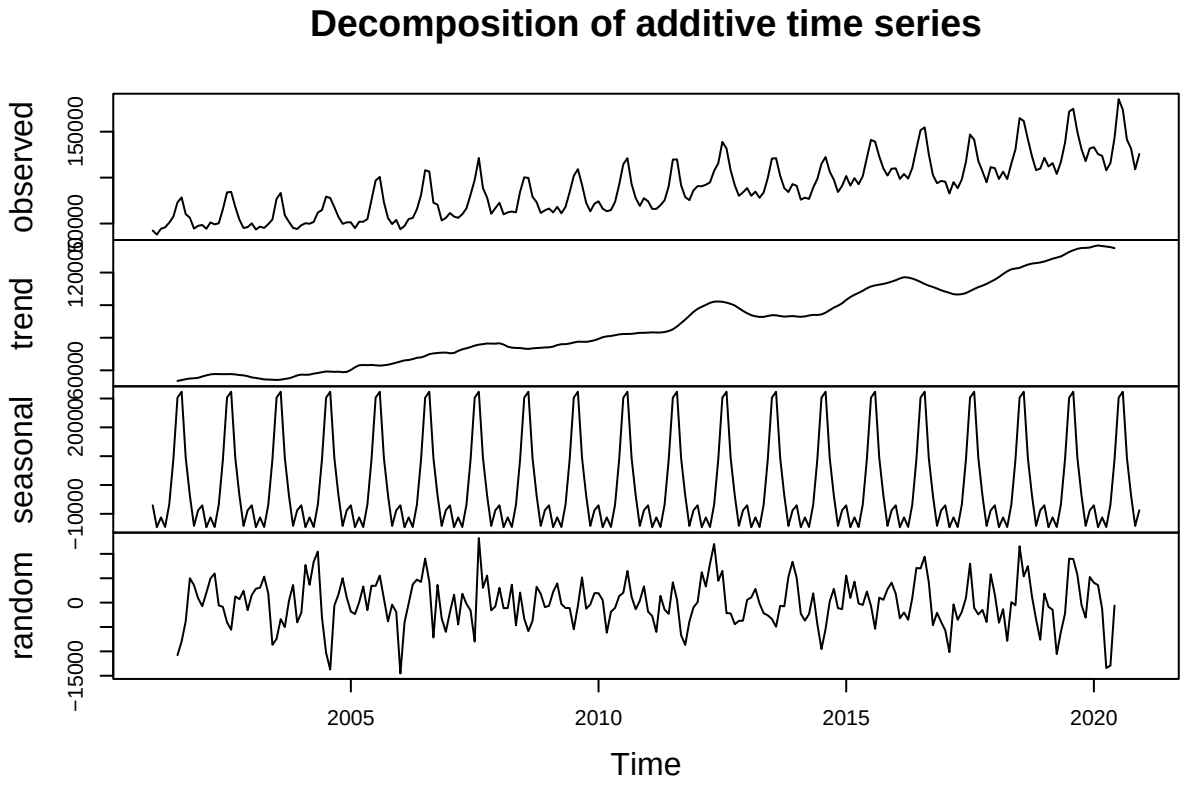


Q2

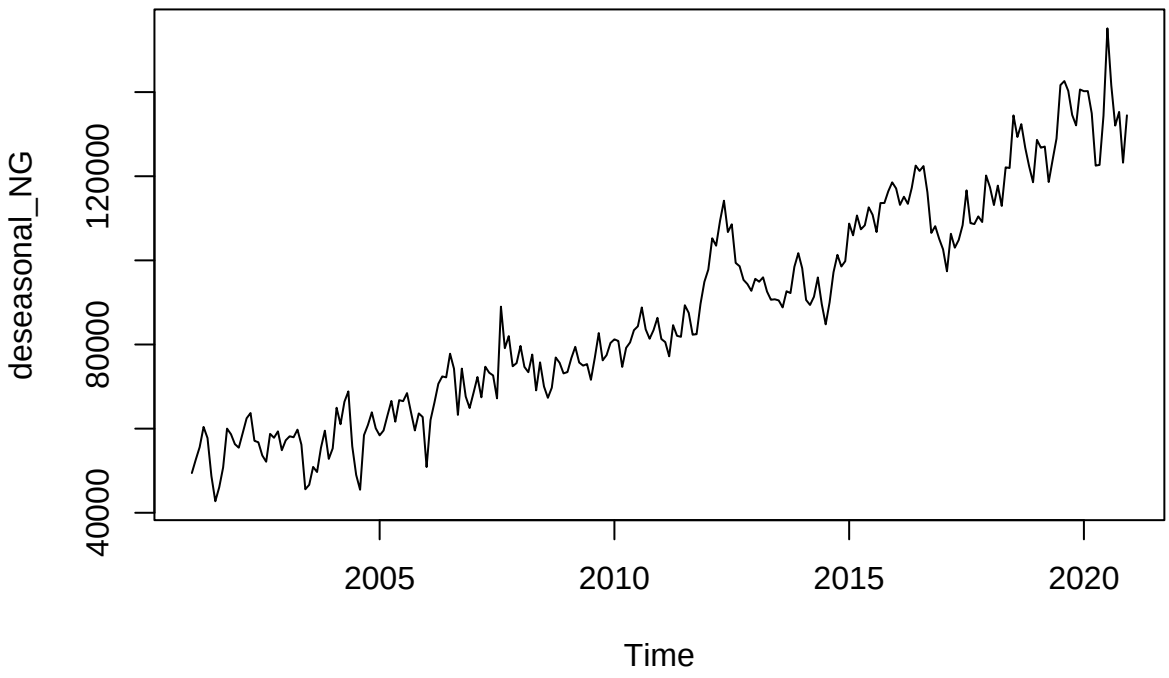
Using the *decompose*() or *stl*() and the *seasadj*() functions create a series without the seasonal component, i.e., a deseasonalized natural gas series. Plot the deseasonalized series over time and corresponding ACF and PACF. Compare with the plots obtained in Q1.

The ACF and PACF show a much cleaner decay than in the Q1 plots without the jumps at the seasonal lags. There is a significant spike at lag 1 of the PACF and decay in the ACF. This suggests and ARIMA model with both an autoregressive and moving average components may be appropriate. There is also still an increasing trend in the data that is more clear to see rather than that of the time plot in Q1.

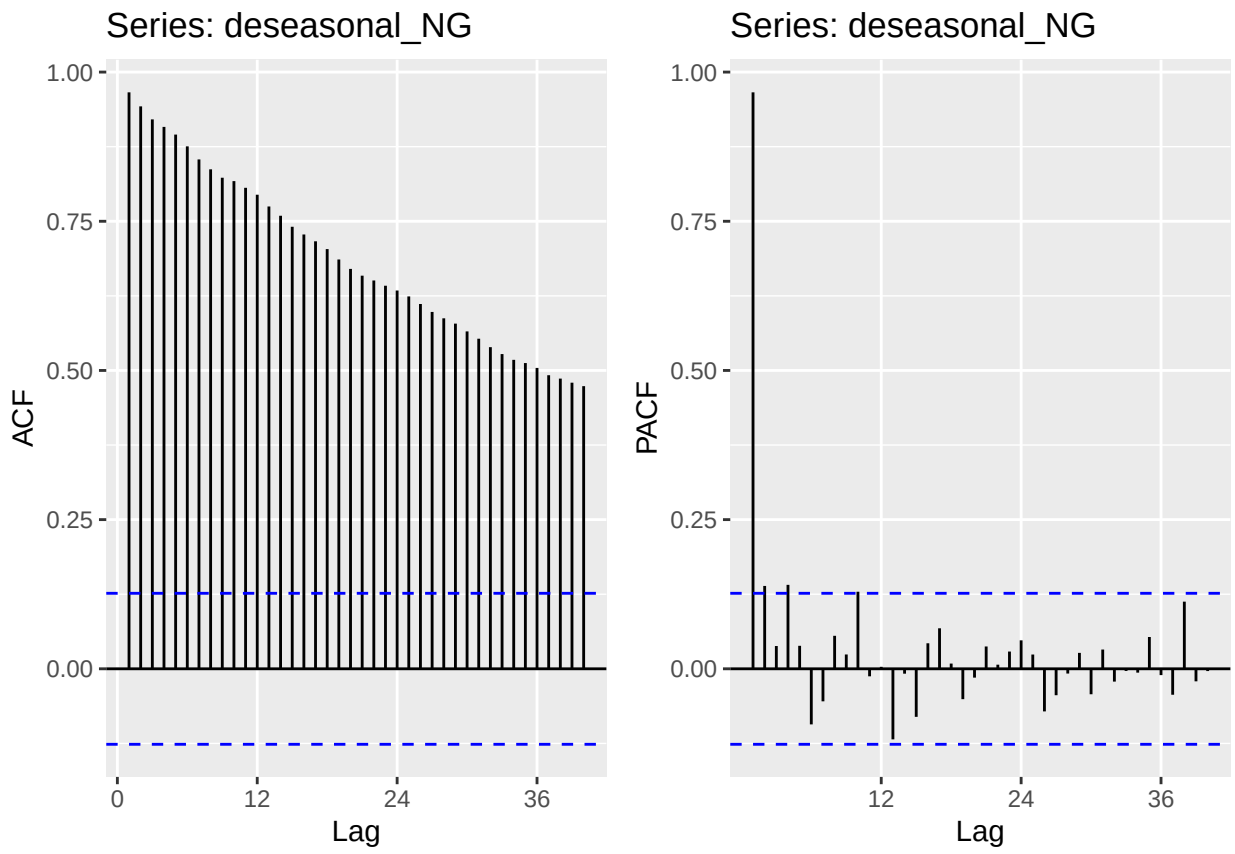
```
#Using R decompose function
decompose_NG <- decompose(ts_NG_gen,"additive")
plot(decompose_NG)
```




```
#The ACF plot show a slow decay which is a sign of non-stationarity.

#Creating non-seasonal residential price time series because some models can't handle seasonality
deseasonal_NG <- seasadj(decompose_NG)

plot(deseasonal_NG)
```




```
plot_grid(
autoplot(Acf(deseasonal_NG, lag= 40, main = "ACF Natural Gas Gen", plot=FALSE)),
autoplot(Pacf(deseasonal_NG, lag= 40, main = "PACF Natural Gas Gen", plot= FALSE)))
```



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.

```
#Null hypothesis is that data has a unit root
print("Results for ADF test/n")
```

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

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

```
## Warning in adf.test(deseasonal_NG, alternative = "stationary"): 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
#Looking at the originl TS using SMK
SMKtest <- SeasonalMannKendall(ts_NG_gen)
print("Results for Seasonal Mann Kendall on original TS /n")
```

```
## [1] "Results for Seasonal Mann Kendall on original TS /n"
```

```r
print(summary(SMKtest))
```

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

```r
#Use deseasoned date to run Mann Kendall
print("Results of Mann Kendall on deseasoned series")
```

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

```r
print(summary(MannKendall(deseasonal_NG)))
```

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

The deseasonalized time series (deseasonal_NG) is stationary according to the ADF test due to its small p value (<0.05). Both the original and deseasoned time series exhibit significant monotonic trends according to the Mann-Kendall test, which both have a high absolute value of tau. ### 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.

In the section we will manually fit ARIMA models to the residential electricity price series using function Arima() from package *forecast*. Some important arguments for Arima() are:

y: univariate (single vector) ts object *order=c(, ,)*: three orders (p,d,q) of non-seasonal part of the ARIMA in this order *include.mean*: the default is TRUE for undifferenced series, which means the model will include a mean term, and FALSE when d > 0 *include.drift*: the default is FALSE, but changing to TRUE might lead to better fits. The drift will be necessary when the series mean is not zero even after differencing

d= 0: The time series is stationary without the need for differencing per the ADF. The PACF plot has a significant spike at lag 1 which suggests the presence of an AR component. Further, the slow decay of the ACF suggests the presence of a moving average component, as it indicates the effect of past disturbances on the current value. Therefore, I set p=1 for the AR and q=1 for the MA order.

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.

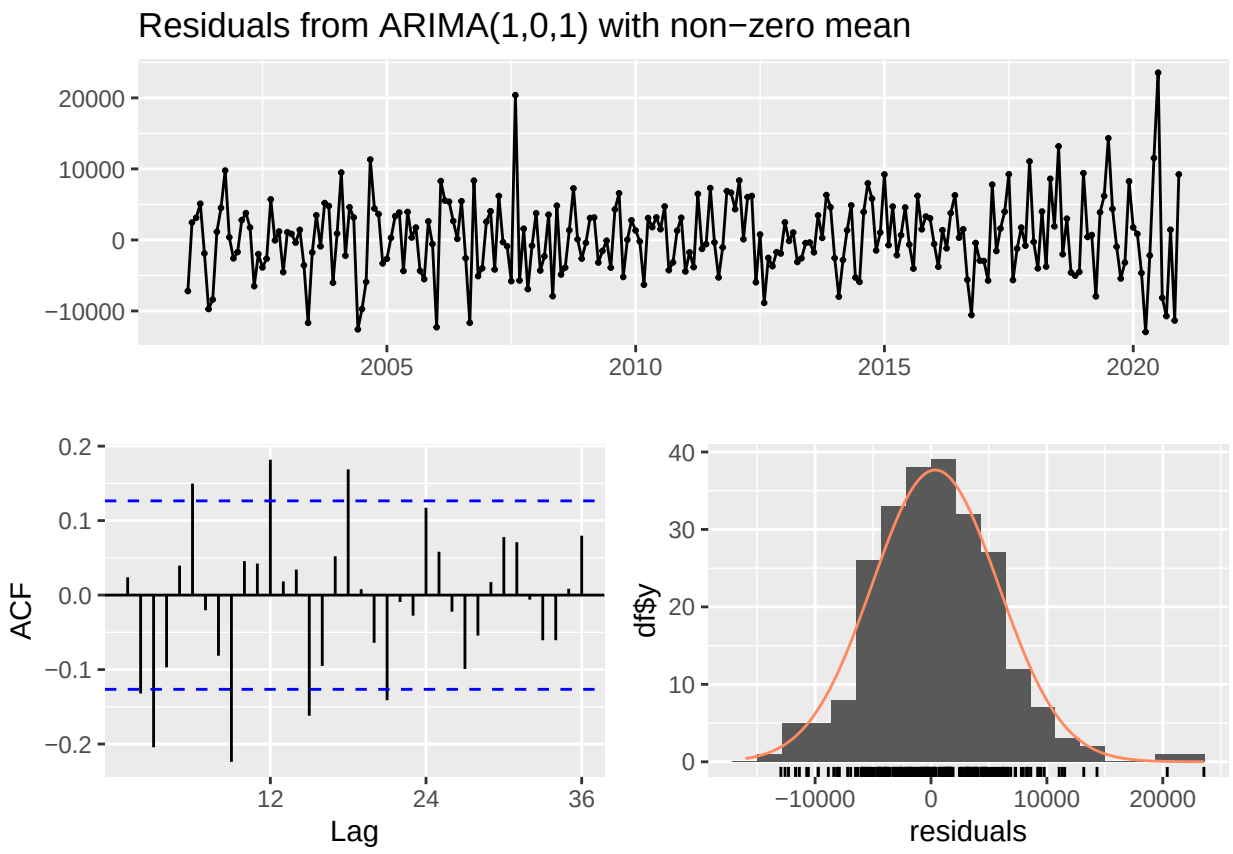
```r
arima_model_mannual <- Arima(deseasonal_NG, order = c(1, 0, 1),
                      include.mean = TRUE, include.drift = FALSE)
summary(arima_model_mannual)
```

```
## Series: deseasonal_NG
## ARIMA(1,0,1) with non-zero mean
##
## Coefficients:
##          ar1      ma1      mean
##       0.9905  -0.2047  91199.78
## s.e.  0.0090   0.0850  21811.30
##
## sigma^2 = 30172540:  log likelihood = -2407.51
## AIC=4823.02   AICc=4823.19   BIC=4836.94
##
## Training set error measures:
##                    ME     RMSE      MAE        MPE     MAPE      MASE       ACF1
## Training set 363.978 5458.515 4207.197 0.01666601 5.159128 0.5143636 0.02382005
```

The summary of the values I chose show a low ACF1 value, meaning the residuals have a low autocorrelation at lag 1, which is good for the models fit. The MASE is greater than 0.5, and a value close to 1 indicates a good model. Further, the MPE is close to zero, indicating that the models predictions are reasonably accurate in percentage terms. Overall, the model used with a non-zero mean seems to provide a reasonable fit to the deseasonalized time series based upon the summary provided. ### 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
checkresiduals(arima_model_mannual)
```




```
##
##  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 residuals vs fitted values plots shows that there are no patterns in the residuals, but rather that they are largely randomly scattered around zero without any clear patterns. The ACF of residuals plot shows somewhat significant autocorrelation of the residuals about every six periods, indicating that some of the residuals are correlated and not random. The residuals are also normally distributed around a mean of zero, demonstrating that they are largely a white noise series, that they are random and uncorrelated and the ARIMA models has effectively captured the underlying patterns in the data.

Modeling the original series (with seasonality)

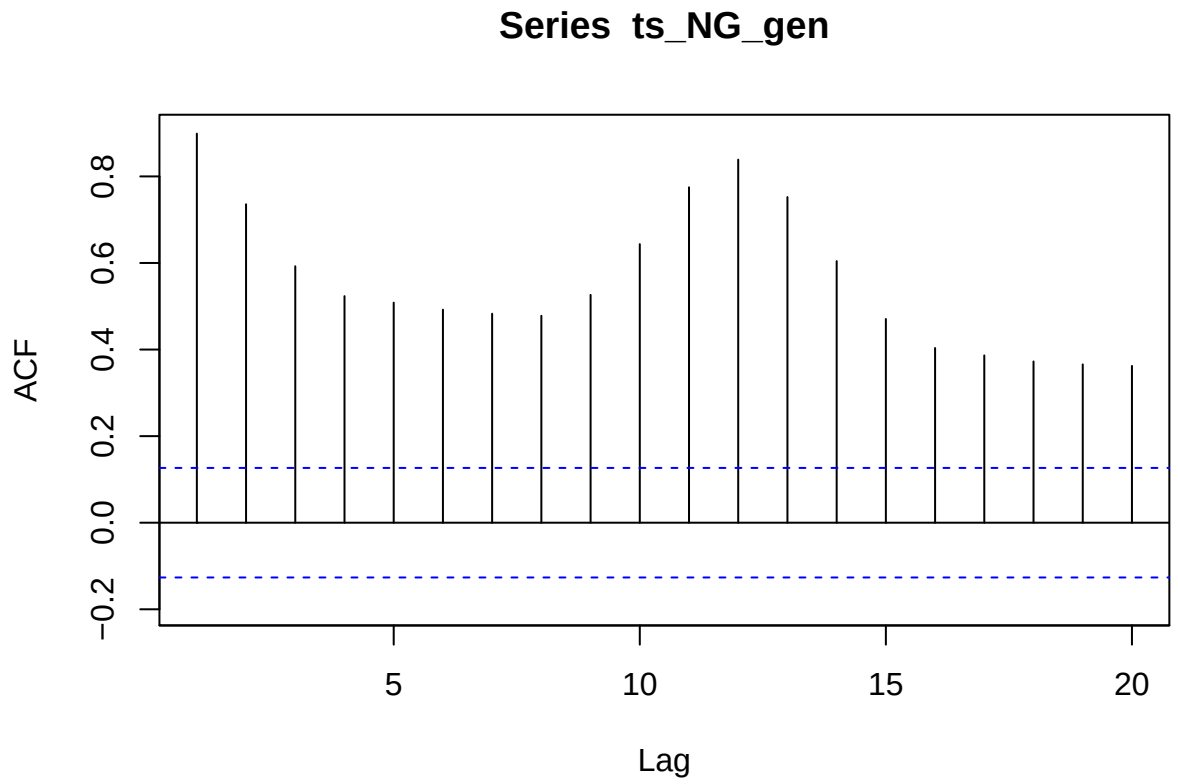
Q7

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.

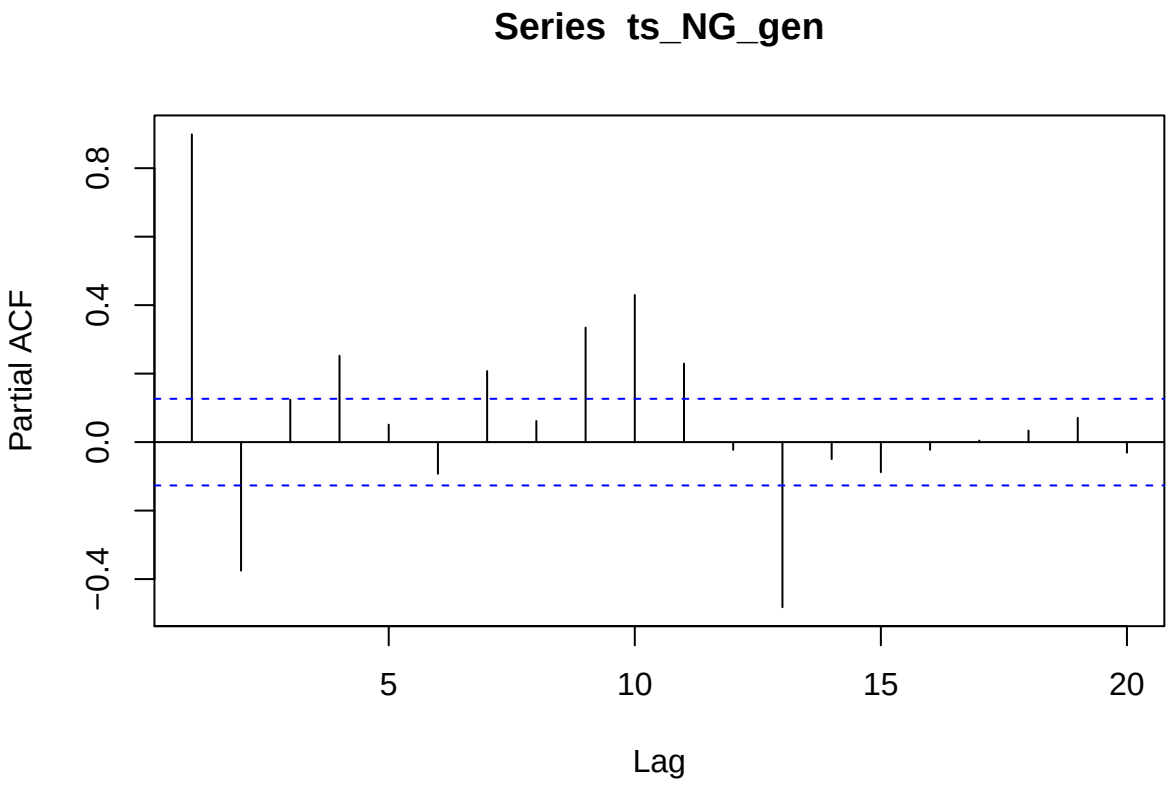
```r
# Identify the ARIMA model parameters
adf_test <- adf.test(ts_NG_gen)
```

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

```r
D <- 1
  #ifelse(adf_test$p.value < 0.05, 0, 1)  # Integration order

# Plot ACF and PACF for seasonal component
Acf(ts_NG_gen, lag.max = 20, plot = TRUE)
```




```r
Pacf(ts_NG_gen, lag.max = 20, plot = TRUE)
```



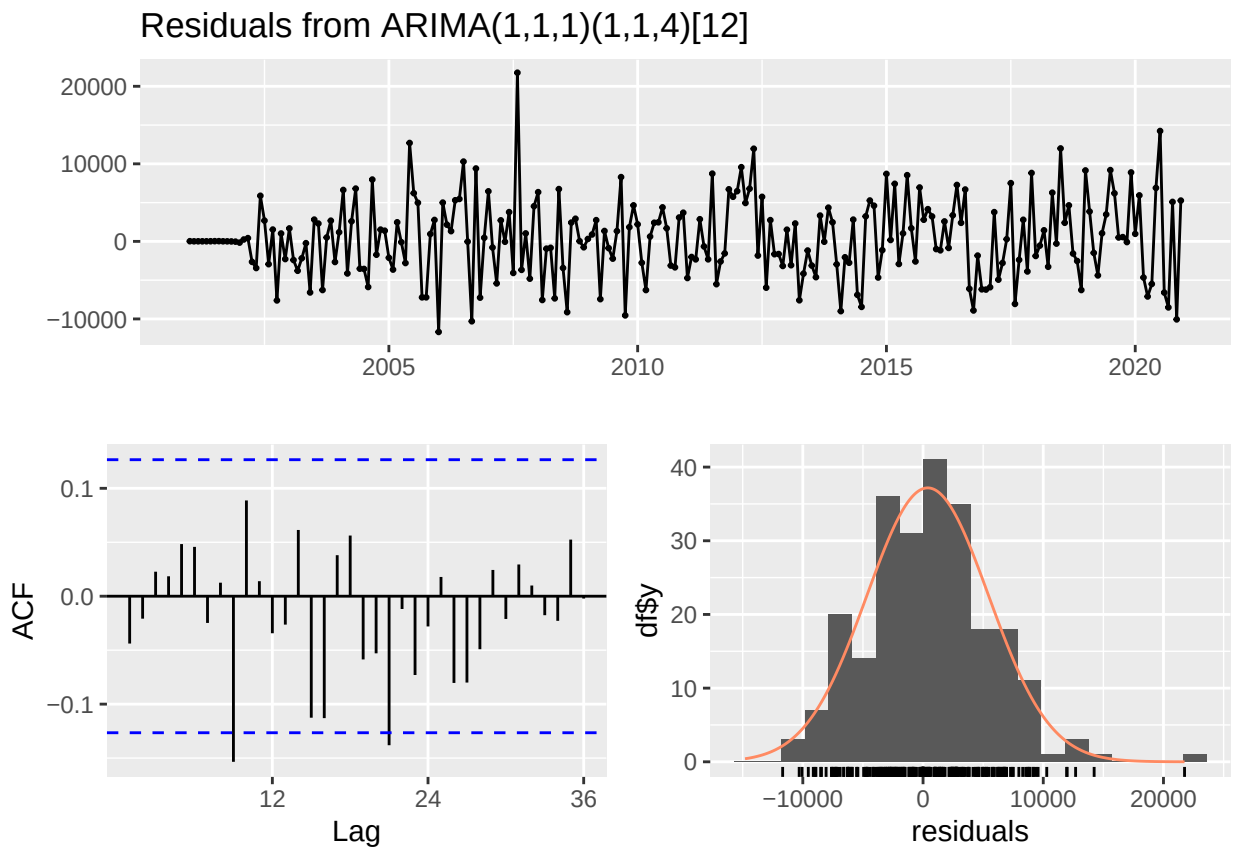

```
# Identify P and Q from ACF and PACF plots for seasonal component
P <-1 #due to significant spike at lag 1 on PACF
Q <- 4 #due to highest spike at lag 12

# Fit ARIMA model to the original series
arima_seasonal_model <- Arima(ts_NG_gen, order = c(1, D, 1), seasonal = list(order = c(P,D,Q)))

# Print coefficients of the fitted ARIMA model
cat("ARIMA Coefficients for Seasonal Series:\n")
```

```
## ARIMA Coefficients for Seasonal Series:
```

```
print(coef(arima_seasonal_model))
```

```
##         ar1         ma1        sar1        sma1        sma2        sma3
##  0.71819371 -0.98076946 -0.98643609  0.28692320 -0.73662444  0.02334485
##        sma4
##  0.09494393
```

```
# Plot residuals and check for white noise
checkresiduals(arima_seasonal_model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,1)(1,1,4)[12]
## Q* = 27.639, df = 17, p-value = 0.04934
##
## Model df: 7.   Total lags used: 24
```

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.

```
summary(arima_model_mannual)
```

```
## Series: deseasonal_NG
## ARIMA(1,0,1) with non-zero mean
##
## Coefficients:
##          ar1      ma1      mean
##       0.9905  -0.2047  91199.78
## s.e.  0.0090   0.0850  21811.30
##
## sigma^2 = 30172540:  log likelihood = -2407.51
```

```
## AIC=4823.02   AICc=4823.19   BIC=4836.94
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE       ACF1
## Training set 363.978 5458.515 4207.197 0.01666601 5.159128 0.5143636 0.02382005
```

```
summary(arima_seasonal_model)
```

```
## Series: ts_NG_gen
## ARIMA(1,1,1)(1,1,4)[12]
##
## Coefficients:
##          ar1      ma1     sar1    sma1     sma2    sma3    sma4
##       0.7182  -0.9808  -0.9864  0.2869  -0.7366  0.0233  0.0949
## s.e.  0.0528   0.0176   0.1052  0.1525   0.1236  0.0759  0.0766
##
## sigma^2 = 28031235:  log likelihood = -2271.33
## AIC=4558.66   AICc=4559.32   BIC=4586.06
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 388.8441 5069.053 3946.962 0.03273652 4.628948 0.4825477
##                     ACF1
## Training set -0.0438672
```

Based on the summary results, it appears that the ARIMA model from Q7 (with seasonal components) has a lower AIC value, lower RMSE, and lower MAPE compared to the ARIMA model from Q6. This suggests that the ARIMA model with seasonal components (Q7) may better represent the Natural Gas Series compared to the model without seasonal components (Q6). ## 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_model <- auto.arima(deseasonal_NG)

# Print the order of the best ARIMA model
print(auto_arima_model)
```

```
## 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
```

This does not match the ARIMA model I fitted in Q4, as I had a d=0 rather than the fitten d=1.

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_model_original <- auto.arima(ts_NG_gen)

# Print the order of the best ARIMA model
print(auto_arima_model_original)
```

```
## 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 ARIMA model does not match what I specified in Q7 at all! There is more that differs from my ARIMA model using auto-arima than is the same aside from my seasonal lag.