ENV 790.30 - Time Series Analysis for Energy Data | Spring 2022

Assignment 7 - Due date 03/25/22

Kristen Pulley

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 project open the first thing you will do is change "Student Name" on line 3 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. Rename the pdf file such that it includes your first and last name (e.g., "LuanaLima_TSA_A07_Sp22.Rmd"). Submit this pdf using Sakai.

Set up

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**.

Packages needed for this assignment: "forecast","tseries". Do not forget to load them before running your script, since they are NOT default packages.\

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.

```
netgen <- read_csv("Data/Net_generation_United_States_all_sectors_monthly.csv", skip = 4)
```

```
## Rows: 240 Columns: 6

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Month
## dbl (5): all fuels (utility-scale) thousand megawatthours, coal thousand meg...

##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

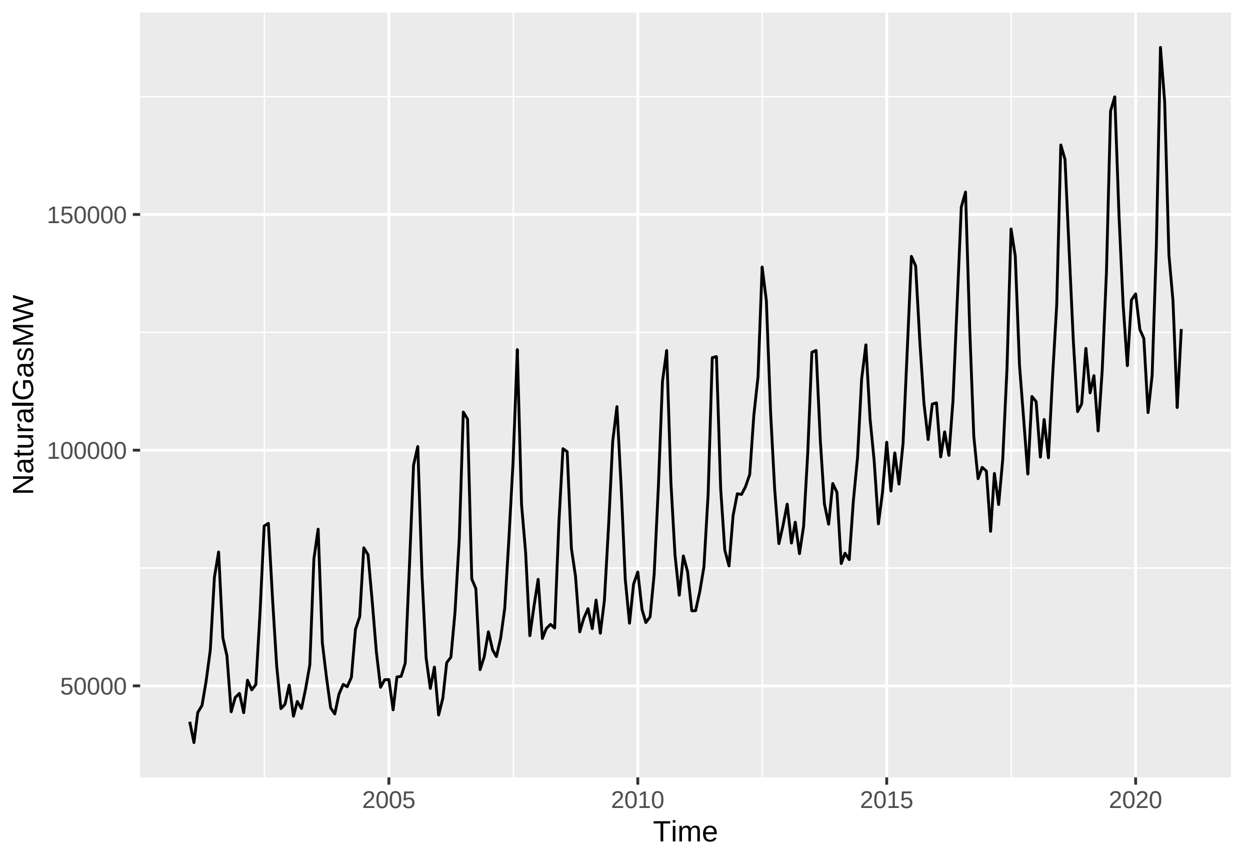
```
gen.df<-data.frame(netgen)
my_date <- paste(gen.df[,1], sep="-")
my_date <- my(my_date)
head(my_date)
```

```
## [1] "2020-12-01" "2020-11-01" "2020-10-01" "2020-09-01" "2020-08-01"
## [6] "2020-07-01"
```

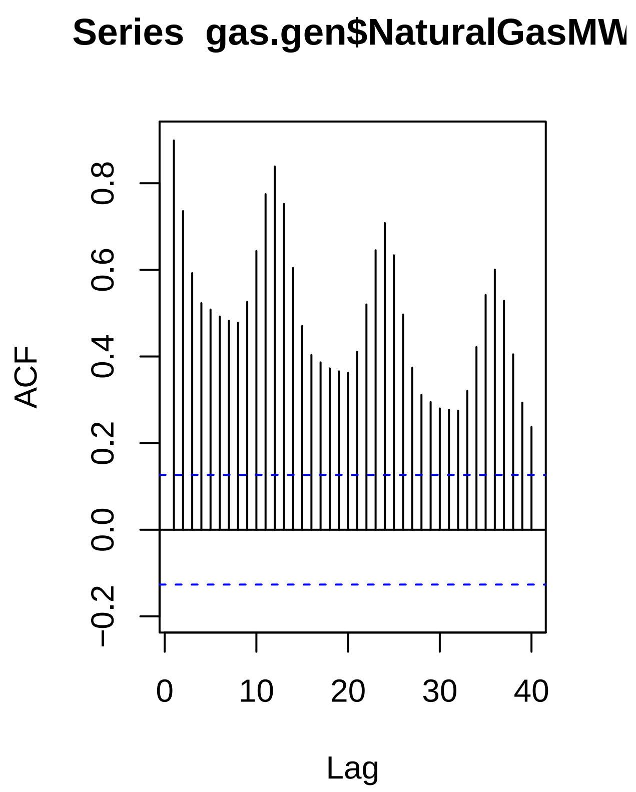
```
gas.gen<- data.frame(my_date,gen.df$natural.gas.thousand.megawatthours)
names(gas.gen)<-c('Time','NaturalGasMW')
head(gas.gen)
```

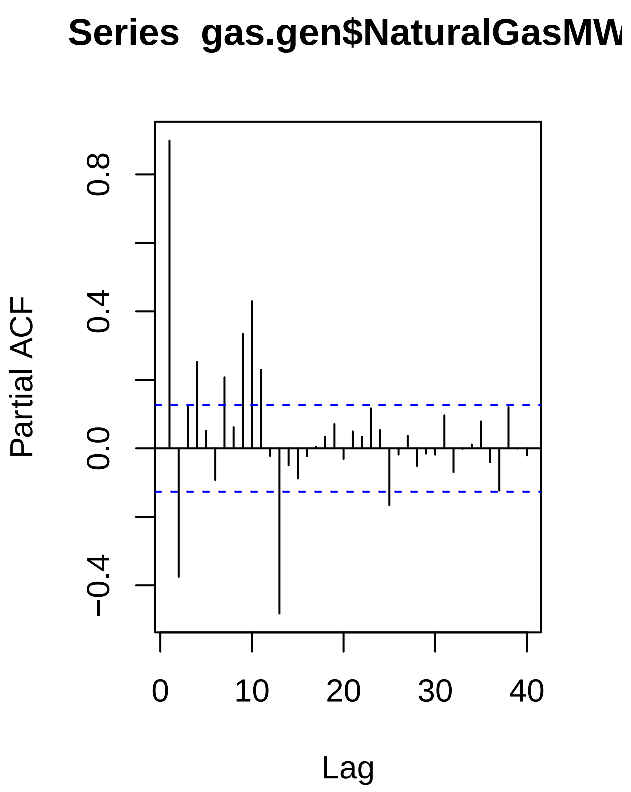
```
##         Time NaturalGasMW
## 1 2020-12-01     125703.7
## 2 2020-11-01     109037.2
## 3 2020-10-01     131658.2
## 4 2020-09-01     141452.7
## 5 2020-08-01     173926.6
## 6 2020-07-01     185444.8
```

```
gas.ts<-ts(gas.gen[,2],
         start=c(year(gas.gen$Time[1]),month(gas.gen$Time[1])),
                           frequency=12)
TS_Plot <-
  ggplot(gas.gen, aes(x=Time, y=NaturalGasMW)) +
      geom_line()
plot(TS_Plot)
```



```r
#ACF and PACF plots
par(mfrow=c(1,2))
ACF_Plot <- Acf(gas.gen$NaturalGasMW, lag = 40, plot = TRUE)
PACF_Plot <- Pacf(gas.gen$NaturalGasMW, lag = 40)
```



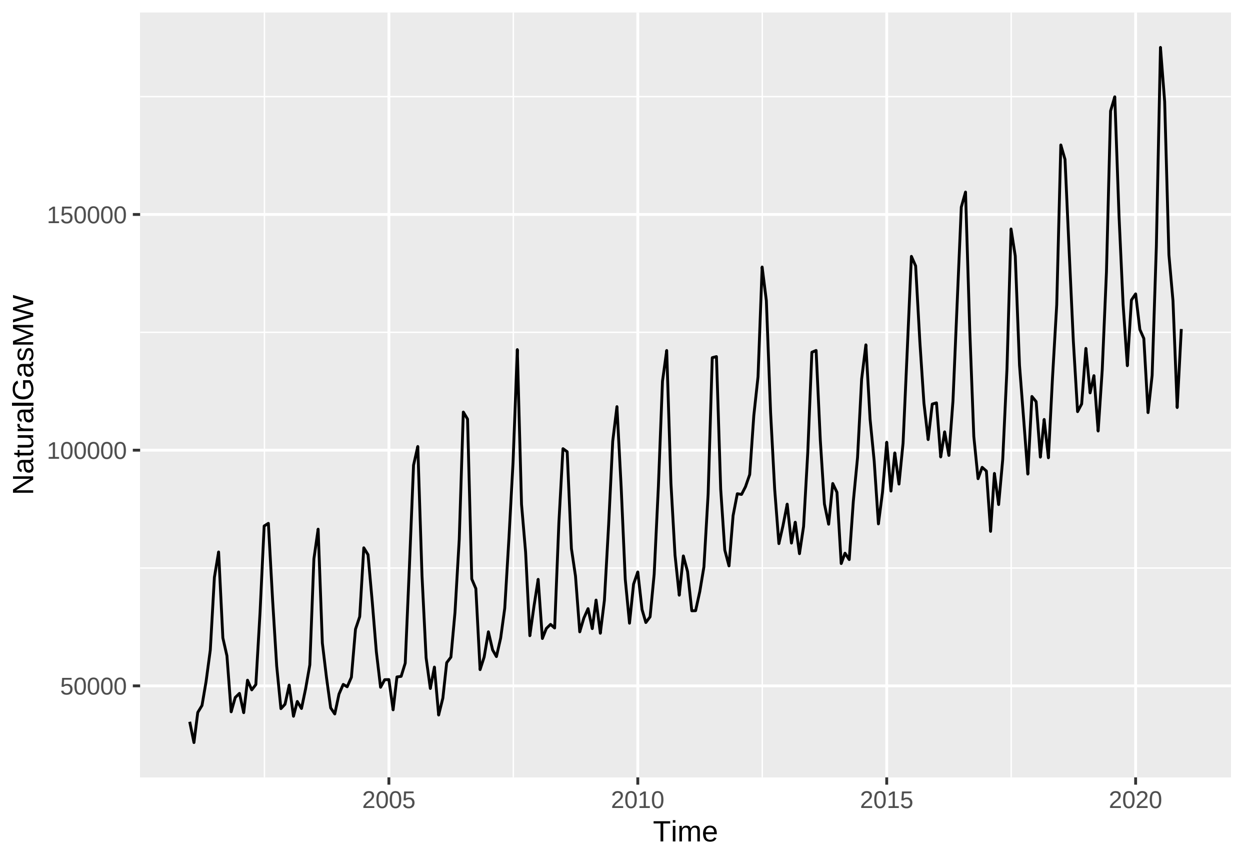


Q2

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.

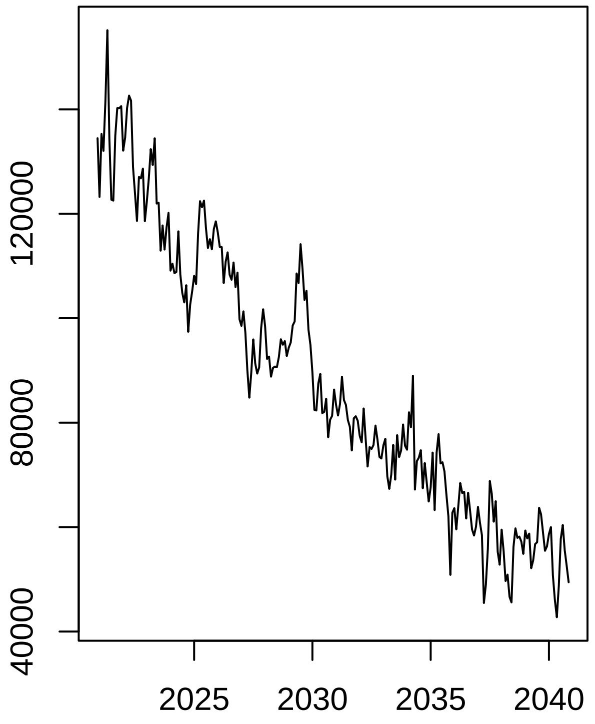
```
decomp.gas<-decompose(gas.ts,"additive")
deseason.gas<-seasadj(decomp.gas)

par(mar=c(3,3,3,0));par(mfrow=c(1,2))
plot(TS_Plot)
```

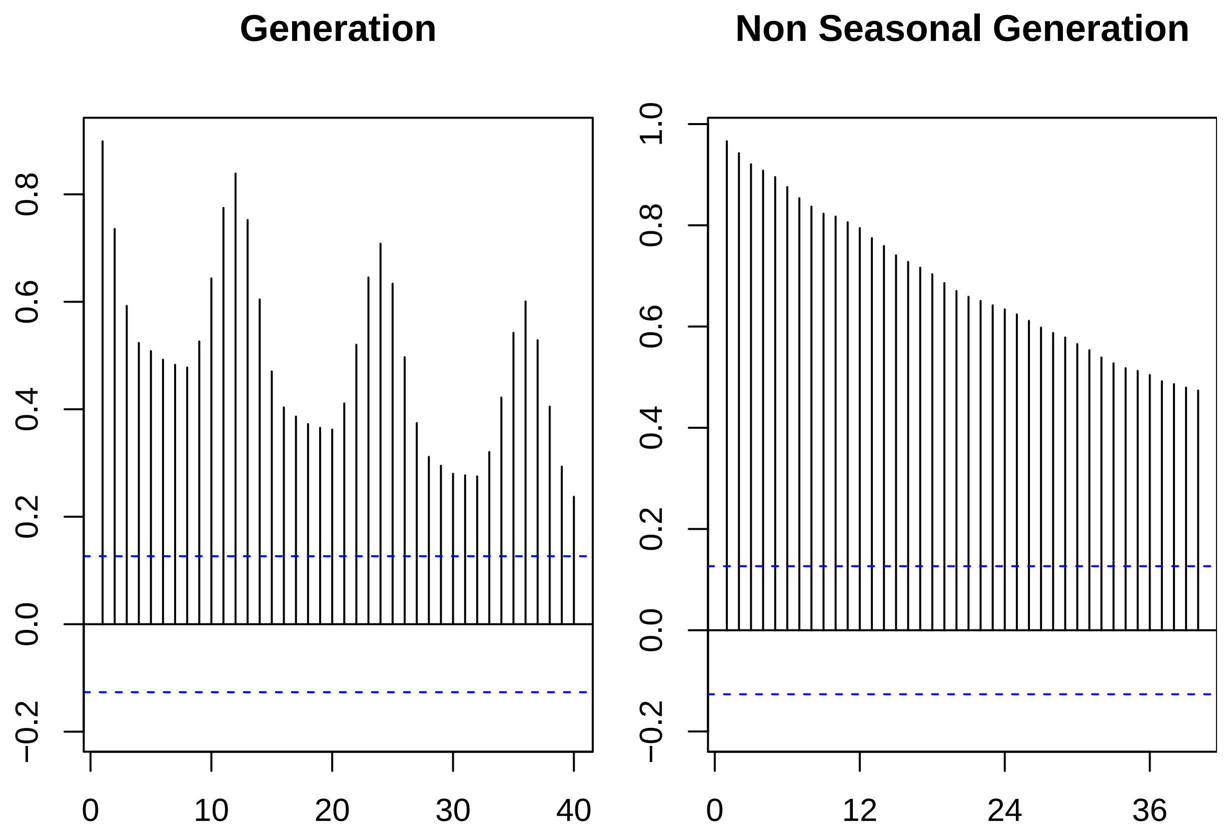


```
plot(deseason.gas)

#Comparing ACFs
par(mar=c(3,3,3,0));par(mfrow=c(1,2))
```

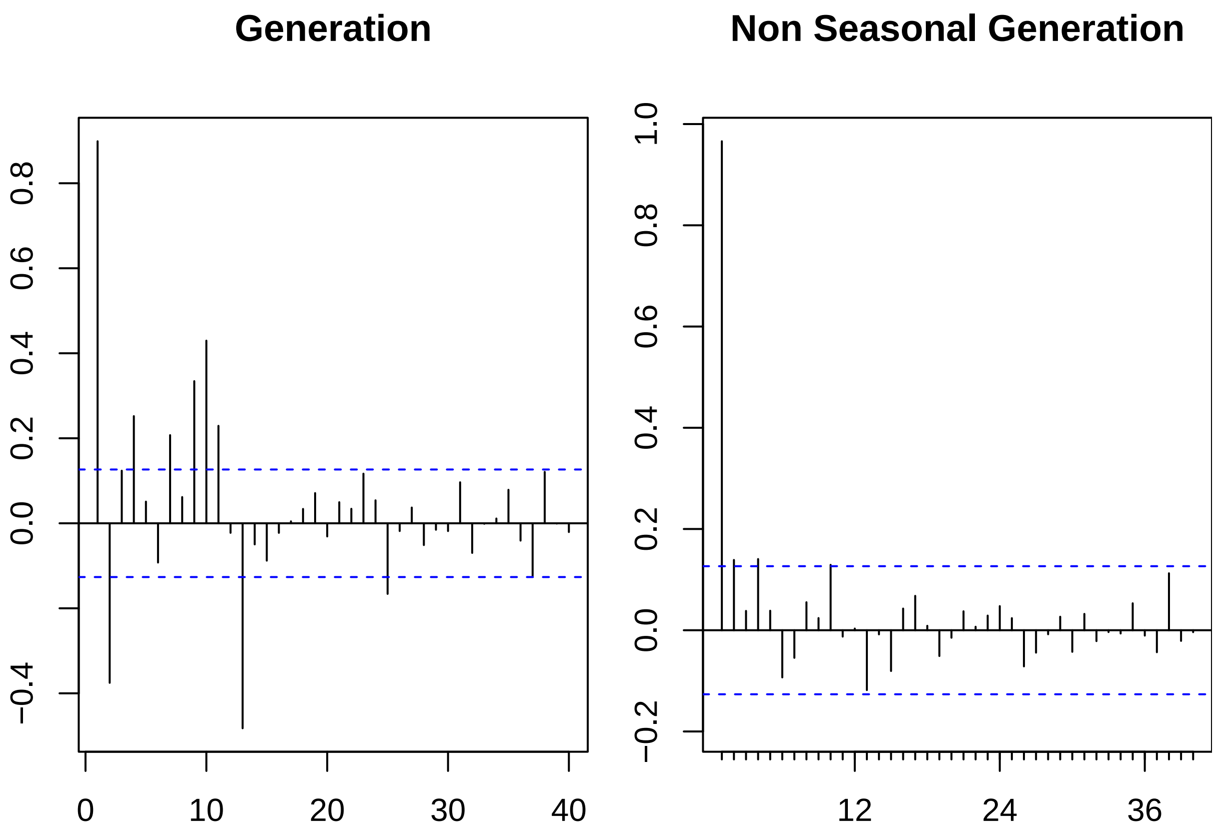


```
Acf(gas.gen$NaturalGasMW,lag.max=40,main="Generation")
Acf(deseason.gas,lag.max=40,main="Non Seasonal Generation")
```



```
#Note seasonality is gone!

#Comparing PACFs
par(mar=c(3,3,3,0));par(mfrow=c(1,2))
Pacf(gas.gen$NaturalGasMW,lag.max=40,main="Generation")
Pacf(deseason.gas,lag.max=40,main="Non Seasonal Generation")
```



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.

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

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

```
##
##  Augmented Dickey-Fuller Test
##
## data:  deseason.gas
## Dickey-Fuller = -4.0574, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

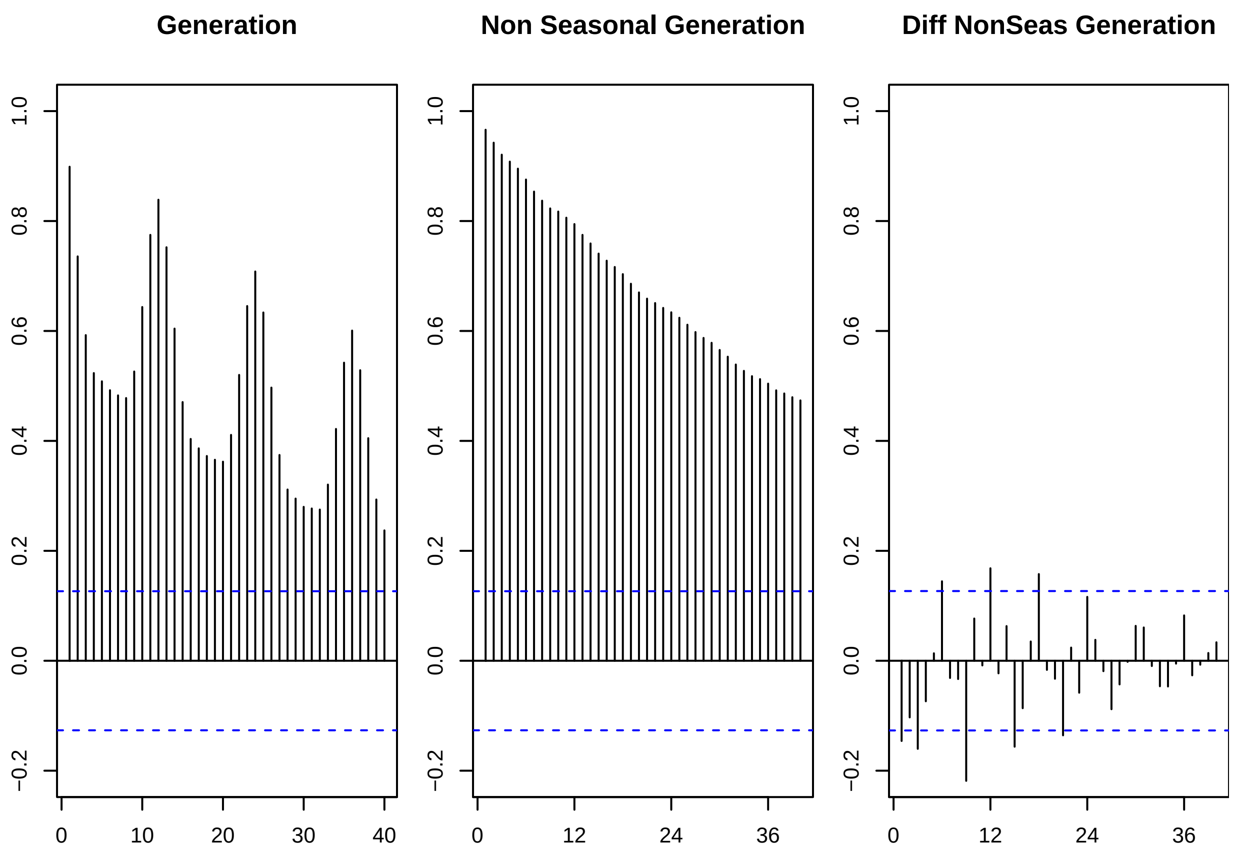
```
print(MannKendall(deseason.gas))
```

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

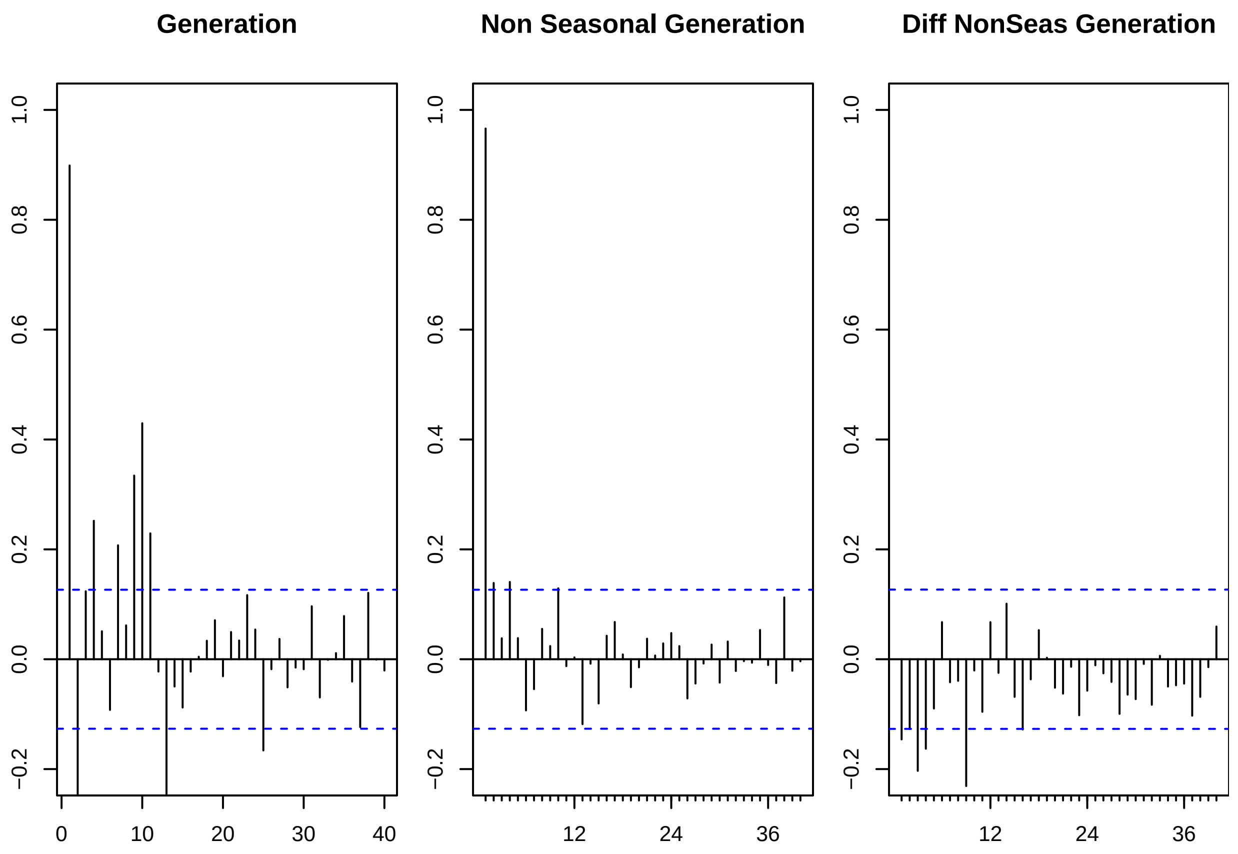
Answer: The ADF test resulted in a p-value less than 0.05, so we reject the null hypthoesis, indicating stionarity in the data. However, the results of the Mann Kendall test with a p-value significantly less than 0.5, indicate a trend present in the data. Therefore, we will difference the series at lag 1 to remove the trend.

```
diff.deseas.gas <-diff(deseason.gas, differences = 1)

#compare all 3 ACFs
par(mar=c(3,3,3,0));par(mfrow=c(1,3))
Acf(gas.gen$NaturalGasMW,lag.max=40,main="Generation",ylim=c(-.2,1))
Acf(deseason.gas,lag.max=40,main="Non Seasonal Generation",ylim=c(-.2,1))
Acf(diff.deseas.gas, lag.max=40, main="Diff NonSeas Generation",ylim=c(-.2,1))
```



```
par(mar=c(3,3,3,0));par(mfrow=c(1,3))
Pacf(gas.gen$NaturalGasMW,lag.max=40,main="Generation",ylim=c(-.2,1))
Pacf(deseason.gas,lag.max=40,main="Non Seasonal Generation",ylim=c(-.2,1))
Pacf(diff.deseas.gas, lag.max=40, main="Diff NonSeas Generation",ylim=c(-.2,1))
```



> Much better!

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 can read the plots and interpret the test results.

> Answer: p: number of autoregressive terms -> 3 d: seasonality was removed -> 0 q: lag cut off in the PACF -> 1

Q5

Use $Arima()$ from package "forecast" to fit an ARIMA model to your series considering the order estimated in Q4. Should you allow for constants in the model, i.e., $include.mean = TRUE$ or $include.drift = TRUE$. **Print the coefficients** in your report. Hint: use the $cat()$ function to print.

```
Model301 <- Arima(deseason.gas,order=c(3,0,1),include.mean=TRUE, include.drift=TRUE)
print(Model301)
```

```
## Series: deseason.gas
## ARIMA(3,0,1) with drift
##
```

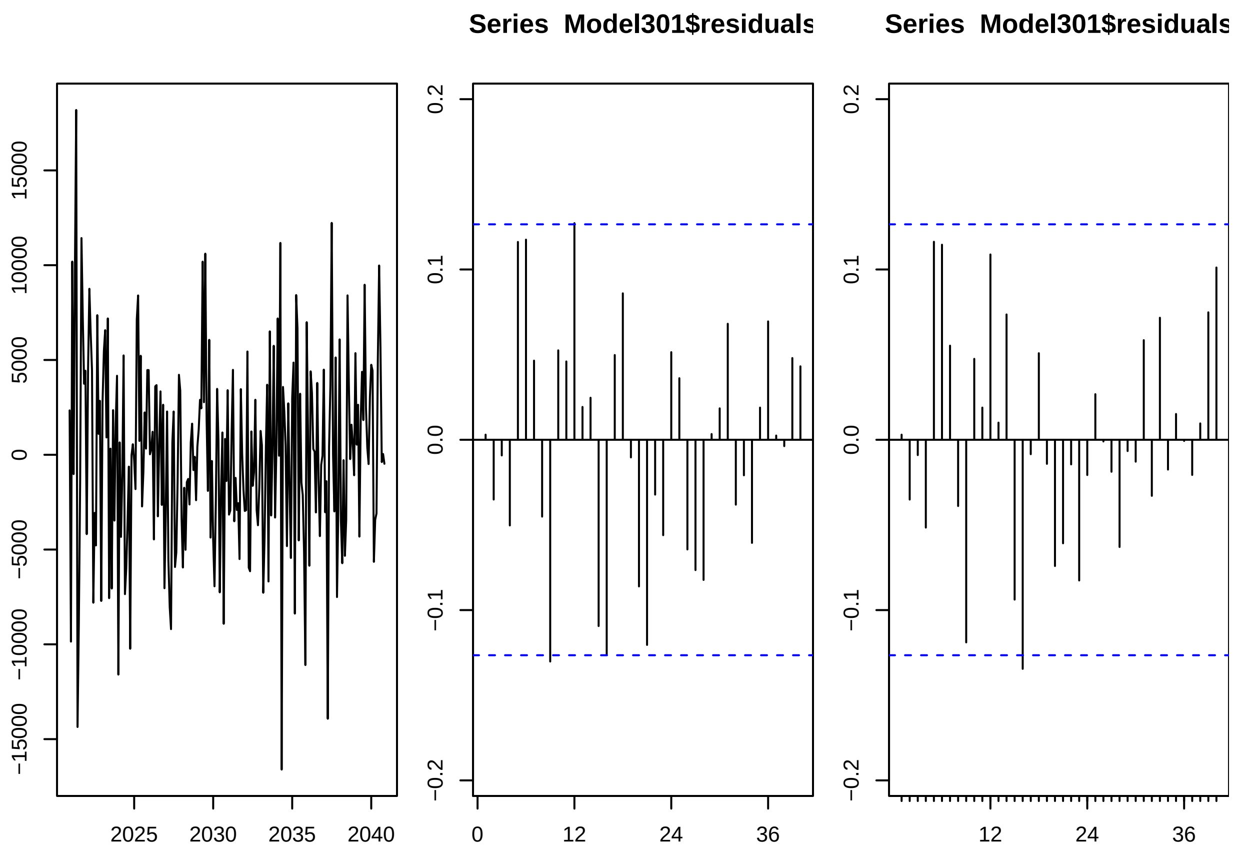
```
## Coefficients:
##           ar1     ar2      ar3     ma1   intercept      drift
##       -0.2052  0.6982  -0.0436  0.9367  131438.108  -359.5054
## s.e.   0.0771  0.0513   0.0664  0.0433    2257.241    16.1598
##
## sigma^2 = 26454293:  log likelihood = -2388.86
## AIC=4791.73   AICc=4792.21   BIC=4816.09
```

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?

> Answer: The residual series look close to a white noise series because most of the lines are within bounds and close to zero.

```r
compare_aic <- data.frame(Model301$aic)
par(mar=c(3,3,3,0));par(mfrow=c(1,3))
ts.plot(Model301$residuals)
Acf(Model301$residuals,lag.max=40)
Pacf(Model301$residuals,lag.max=40)
```



Modeling the original series (with seasonality)

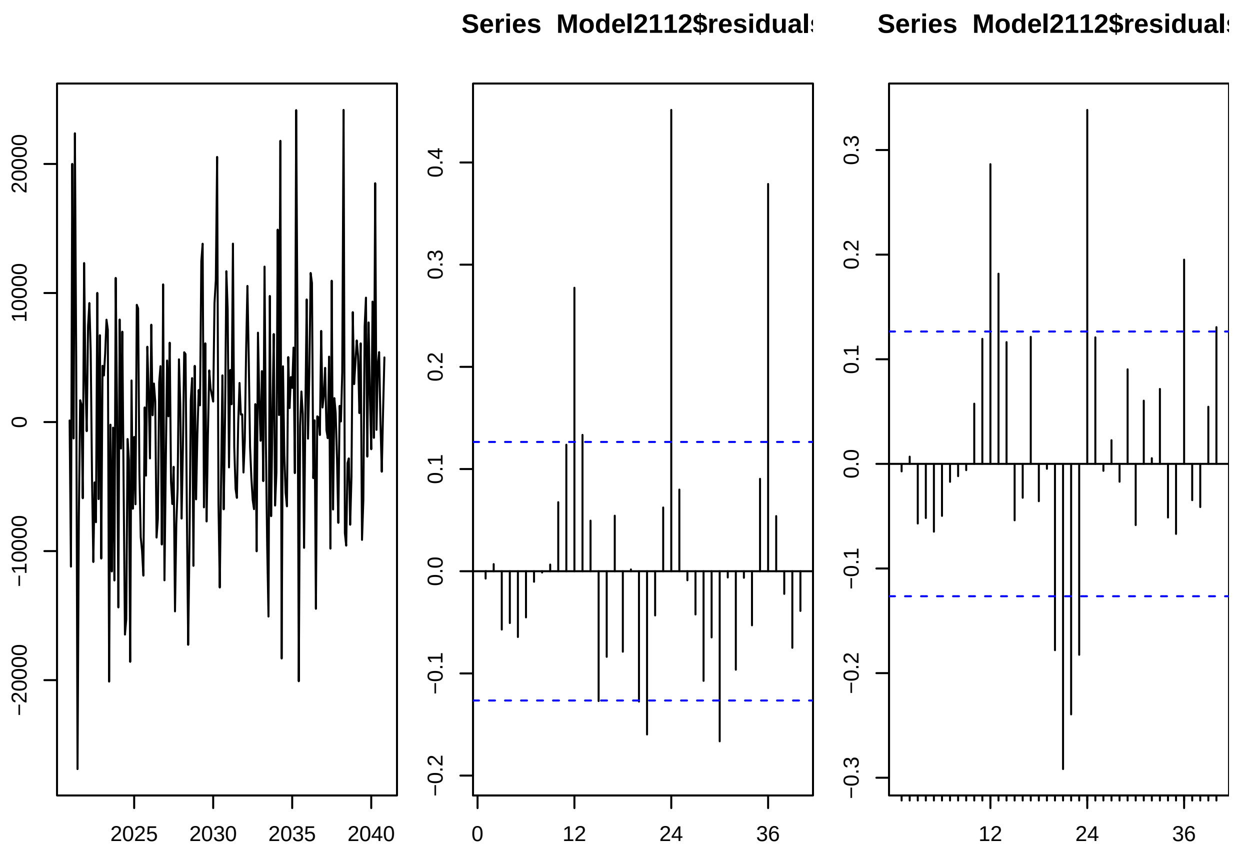
Q7

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: P: number of autoregressive terms -> 2 D: seasonality present, number of differences -> 1 Q: number of moving average terms -> 12 inlcude.mean=FALSE: because d>0

```
Model2112 <- Arima(gas.ts,order=c(2,1,12),include.drift=TRUE)
print(Model2112)
```

```
## Series: gas.ts
## ARIMA(2,1,12) with drift
##
## Coefficients:
##          ar1      ar2      ma1      ma2      ma3      ma4     ma5      ma6
##       0.2107  -0.0734  -0.1676  -0.2503  -0.2708  -0.2366  0.0523  -0.1478
## s.e.  0.1480   0.1300   0.1480   0.1191   0.0817   0.0926  0.0719   0.0642
##           ma7      ma8      ma9     ma10    ma11    ma12      drift
##       -0.0769  -0.2819  -0.3526  -0.0317  0.1720  0.6473  -362.2512
## s.e.   0.0817   0.0952   0.0805   0.0724  0.0871  0.0774    33.0808
##
## sigma^2 = 70753001:  log likelihood = -2500.15
## AIC=5032.3   AICc=5034.76   BIC=5087.93
```

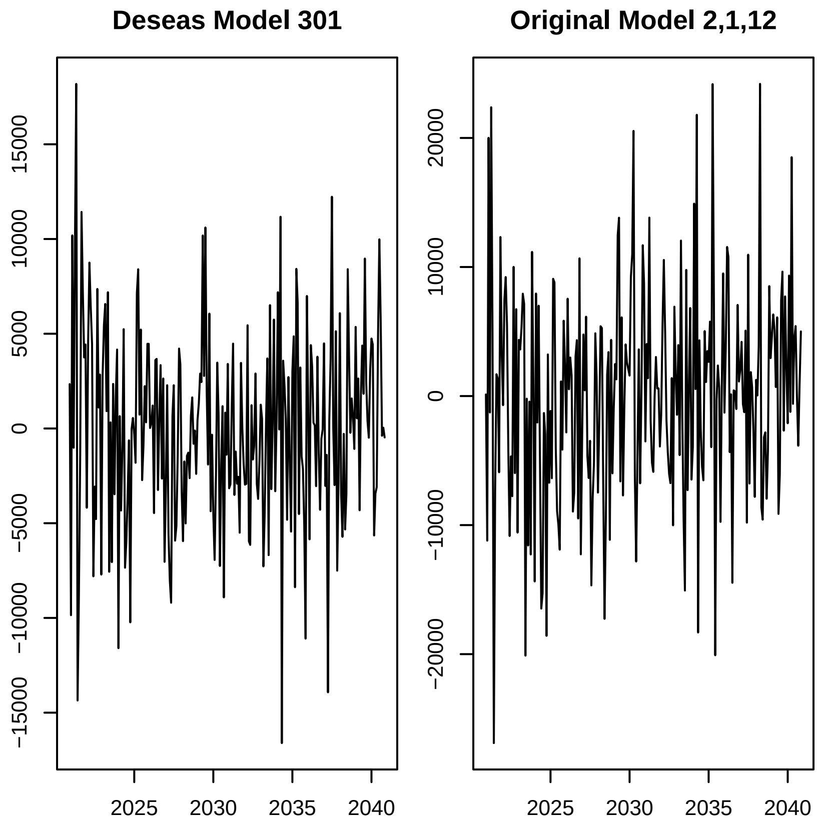
```
compare_aic <- data.frame(Model2112$aic)
par(mar=c(3,3,3,0));par(mfrow=c(1,3))
ts.plot(Model2112$residuals)
Acf(Model2112$residuals,lag.max=40)
Pacf(Model2112$residuals,lag.max=40)
```



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
par(mar=c(3,3,3,0));par(mfrow=c(1,3))
ts.plot(Model301$residuals, main="Deseas Model 301")
ts.plot(Model2112$residuals, main="Original Model 2,1,12")
```



> Answer: The first model with the deseasoned data (Model (3,0,1)), looks the best representing ARIMA because its autoregressive terms are closer to 0 than in the seasonal model.

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 correct orders. The intention of the assignment is to walk you to the process and help you figure out what you did wrong (if you did anything wrong!).

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?

> Answer: The order of the best ARIMA model is 3,1,0, wheras I predicted a 3,0,1 model. It does not match my specification in Q4.

```
auto.arima(deseason.gas)
```

```
## Series: deseason.gas
## ARIMA(3,1,0)(1,0,1)[12] with drift
##
## Coefficients:
```

```
##           ar1      ar2      ar3    sar1     sma1      drift
##       -0.2028  -0.1851  -0.1378  0.6609  -0.4698  -331.8138
## s.e.   0.0645   0.0655   0.0682  0.1918   0.2120   328.8253
##
## sigma^2 = 27791547:  log likelihood = -2384.89
## AIC=4783.79   AICc=4784.27   BIC=4808.12
```

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?

Answer: The parameters specified by R do not match my prediction in Q7.

```
auto.arima(gas.ts, include.mean=TRUE)
```

```
## Series: gas.ts
## ARIMA(2,0,1)(2,1,2)[12] with drift
##
## Coefficients:
##          ar1      ar2      ma1     sar1     sar2     sma1    sma2      drift
##       1.1650  -0.2834  -0.4837  -0.0667  -0.0785  -0.6371  0.0072  -357.8435
## s.e.  0.4164   0.3180   0.3993   1.3218   0.1014   1.3211  0.9212    44.1285
##
## sigma^2 = 27958165:  log likelihood = -2278.46
## AIC=4574.91   AICc=4575.74   BIC=4605.77
```