M7: Seasonal ARIMA Models in R

Luana Lima

02/25/2021

Setting R code chunk options

First R code chunk is used for setting the options for all R code chunks. The choice echo=TRUE means both code and output will appear on report, include = FALSE neither code nor output is printed.

Loading packages and initializing

Second R code chunk is for loading packages. By setting message = FALSE, the code will appear but not the output.

```
library(lubridate)
library(ggplot2)
library(forecast)
library(Kendall)
library(tseries)
library(outliers)
library(tidyverse)
```

Importing data

For this module we will work with monthly average for electricity retail price in US. The data is from the U.S. Energy Information Administration and can be download [here][https://www.eia.gov/electricity/data/browser/#/topic/7?agg=2,0,1&geo=g&freq=M%2013:41:41%20GMT-0500%20(EST)].

```
#Importing time series data from text file#
electricity_price <- read.csv(file="./Data/Average_retail_price_of_electricity_United_States_monthly.csv

#Inspect data
head(electricity_price)
```

```
##      Month all.sectors.cents.per.kilowatthour
## 1 Nov 2020                              10.45
## 2 Oct 2020                              10.64
## 3 Sep 2020                              11.07
## 4 Aug 2020                              11.11
## 5 Jul 2020                              11.14
## 6 Jun 2020                              10.96
##   residential.cents.per.kilowatthour commercial.cents.per.kilowatthour
## 1                              13.35                             10.59
## 2                              13.60                             10.73
## 3                              13.55                             11.07
## 4                              13.31                             10.95
## 5                              13.26                             10.90
## 6                              13.28                             10.95
```

```
##   industrial.cents.per.kilowatthour
## 1                              6.48
## 2                              6.72
## 3                              7.01
## 4                              7.09
## 5                              7.17
## 6                              6.94
```

```r
nvar <- ncol(electricity_price) - 1
nobs <- nrow(electricity_price)

#Preparing the data - create date object and rename columns
electricity_price_processed <-
  electricity_price %>%
  mutate( Month = my(Month) ) %>%
  rename( All.sectors = all.sectors.cents.per.kilowatthour ) %>%
  rename( Residential = residential.cents.per.kilowatthour ) %>%
  rename( Commercial = commercial.cents.per.kilowatthour ) %>%
  rename( Industrial = industrial.cents.per.kilowatthour ) %>%
  arrange( Month )

head(electricity_price_processed)
```

```
##        Month All.sectors Residential Commercial Industrial
## 1 2001-01-01        6.75        7.73       7.25       4.73
## 2 2001-02-01        6.87        8.04       7.51       4.80
## 3 2001-03-01        7.01        8.32       7.70       4.86
## 4 2001-04-01        7.02        8.46       7.73       4.87
## 5 2001-05-01        7.17        8.83       7.77       5.00
## 6 2001-06-01        7.58        9.07       8.13       5.23
```

```r
summary(electricity_price_processed)
```

```
##      Month             All.sectors      Residential      Commercial
##  Min.   :2001-01-01   Min.   : 6.750   Min.   : 7.73   Min.   : 7.250
##  1st Qu.:2005-12-16   1st Qu.: 8.520   1st Qu.: 9.82   1st Qu.: 9.070
##  Median :2010-12-01   Median : 9.720   Median :11.77   Median :10.080
##  Mean   :2010-11-30   Mean   : 9.381   Mean   :11.23   Mean   : 9.746
##  3rd Qu.:2015-11-16   3rd Qu.:10.305   3rd Qu.:12.64   3rd Qu.:10.540
##  Max.   :2020-11-01   Max.   :11.140   Max.   :13.60   Max.   :11.170
##    Industrial
##  Min.   :4.71
##  1st Qu.:5.99
##  Median :6.58
##  Mean   :6.37
##  3rd Qu.:6.89
##  Max.   :7.72
```

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

Transforming data into time series object

Many of the functions we will use require a time series object. You can transform your data in a time series using the function *ts()*.

```r
ts_electricity_price <- ts(electricity_price_processed[,2:(nvar+1)],
                           start=c(year(electricity_price_processed$Month[1]),month(electricity_price_p
                           frequency=12)
#note that we are only transforming columns with electricity price, not the date columns
head(ts_electricity_price,15)
```

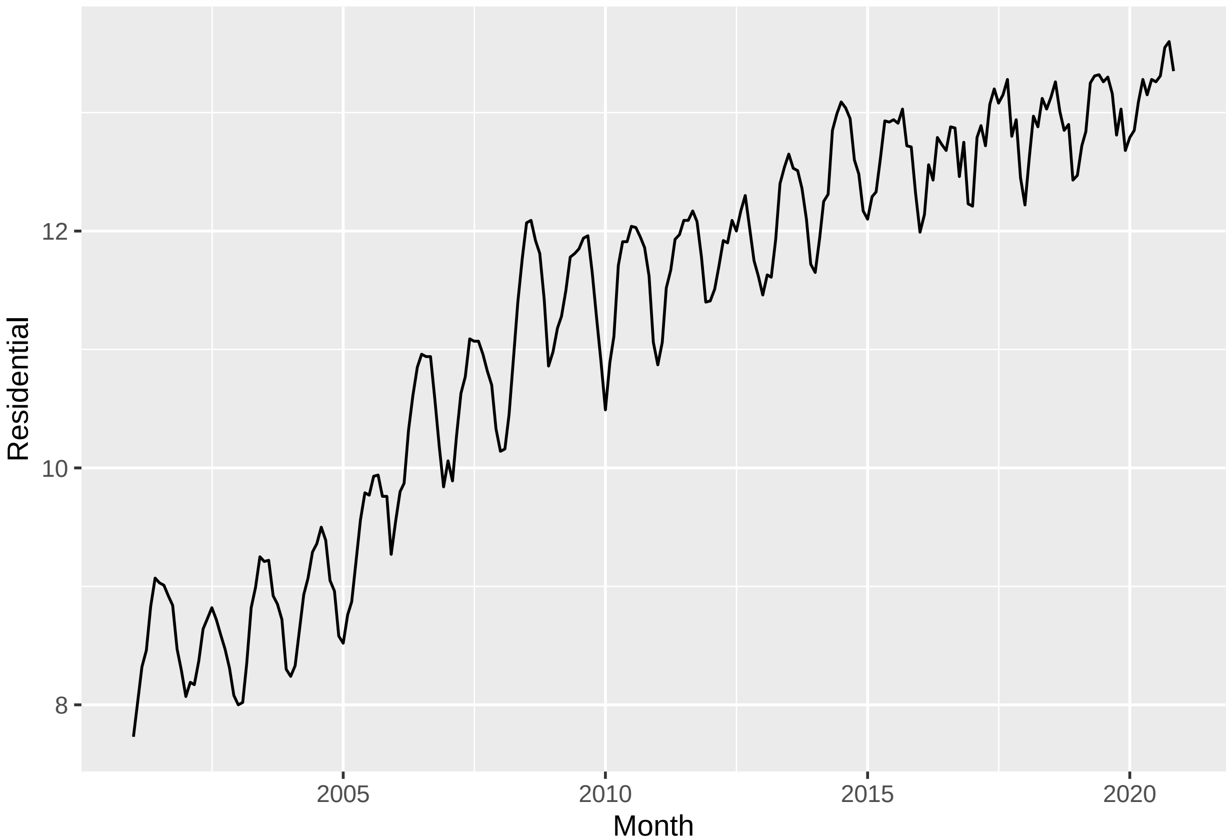
```
##          All.sectors Residential Commercial Industrial
## Jan 2001        6.75        7.73       7.25       4.73
## Feb 2001        6.87        8.04       7.51       4.80
## Mar 2001        7.01        8.32       7.70       4.86
## Apr 2001        7.02        8.46       7.73       4.87
## May 2001        7.17        8.83       7.77       5.00
## Jun 2001        7.58        9.07       8.13       5.23
## Jul 2001        7.88        9.03       8.41       5.57
## Aug 2001        7.84        9.01       8.35       5.50
## Sep 2001        7.62        8.92       8.22       5.31
## Oct 2001        7.43        8.84       8.27       5.07
## Nov 2001        7.02        8.47       7.73       4.78
## Dec 2001        7.03        8.29       7.66       4.78
## Jan 2002        6.95        8.07       7.49       4.73
## Feb 2002        6.97        8.19       7.68       4.76
## Mar 2002        6.95        8.17       7.72       4.73
```

```r
tail(ts_electricity_price,15)
```

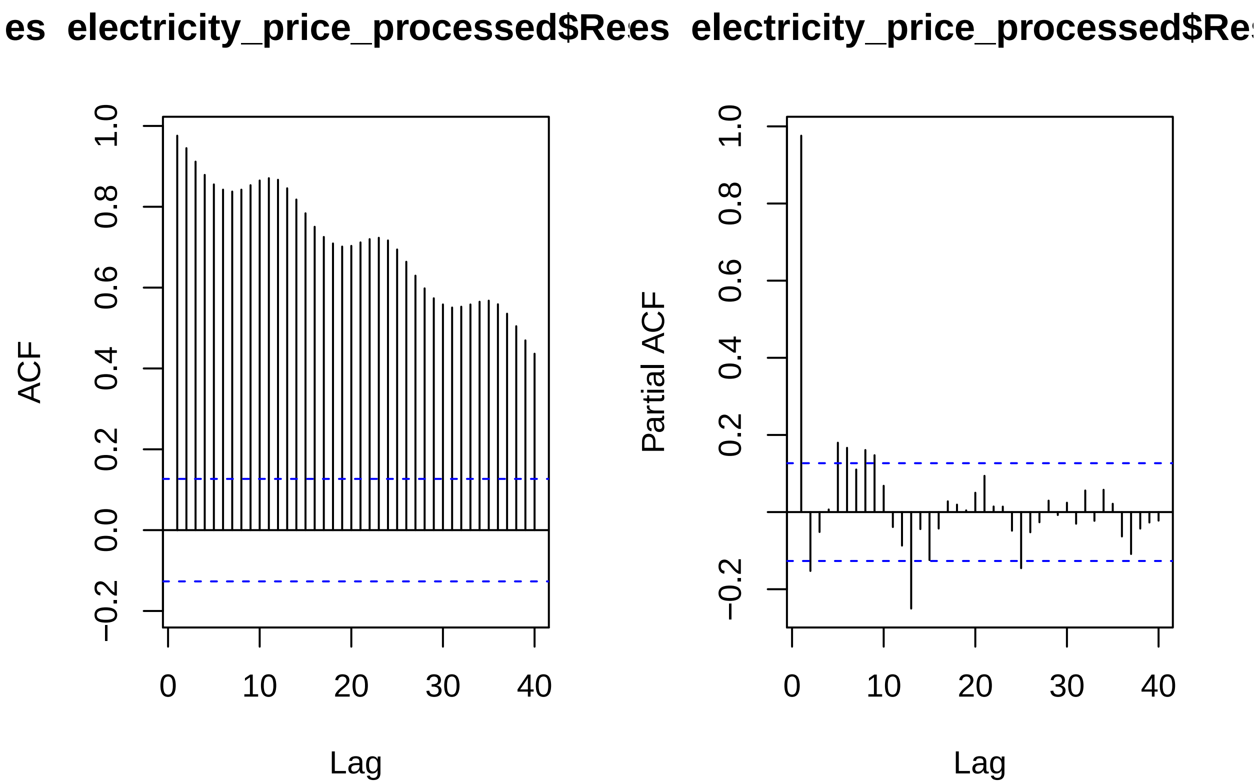
```
##          All.sectors Residential Commercial Industrial
## Sep 2019       10.82       13.16      10.96       7.06
## Oct 2019       10.39       12.81      10.74       6.84
## Nov 2019       10.38       13.03      10.57       6.72
## Dec 2019       10.22       12.68      10.32       6.38
## Jan 2020       10.28       12.79      10.24       6.33
## Feb 2020       10.29       12.85      10.36       6.41
## Mar 2020       10.29       13.09      10.41       6.38
## Apr 2020       10.42       13.28      10.42       6.40
## May 2020       10.47       13.15      10.46       6.53
## Jun 2020       10.96       13.28      10.95       6.94
## Jul 2020       11.14       13.26      10.90       7.17
## Aug 2020       11.11       13.31      10.95       7.09
## Sep 2020       11.07       13.55      11.07       7.01
## Oct 2020       10.64       13.60      10.73       6.72
## Nov 2020       10.45       13.35      10.59       6.48
```

Initial Plots

```r
#Generating a box plot by factor where factor is month of the year

TS_Plot <-
  ggplot(electricity_price_processed, aes(x=Month, y=Residential)) +
      geom_line()
plot(TS_Plot)
```



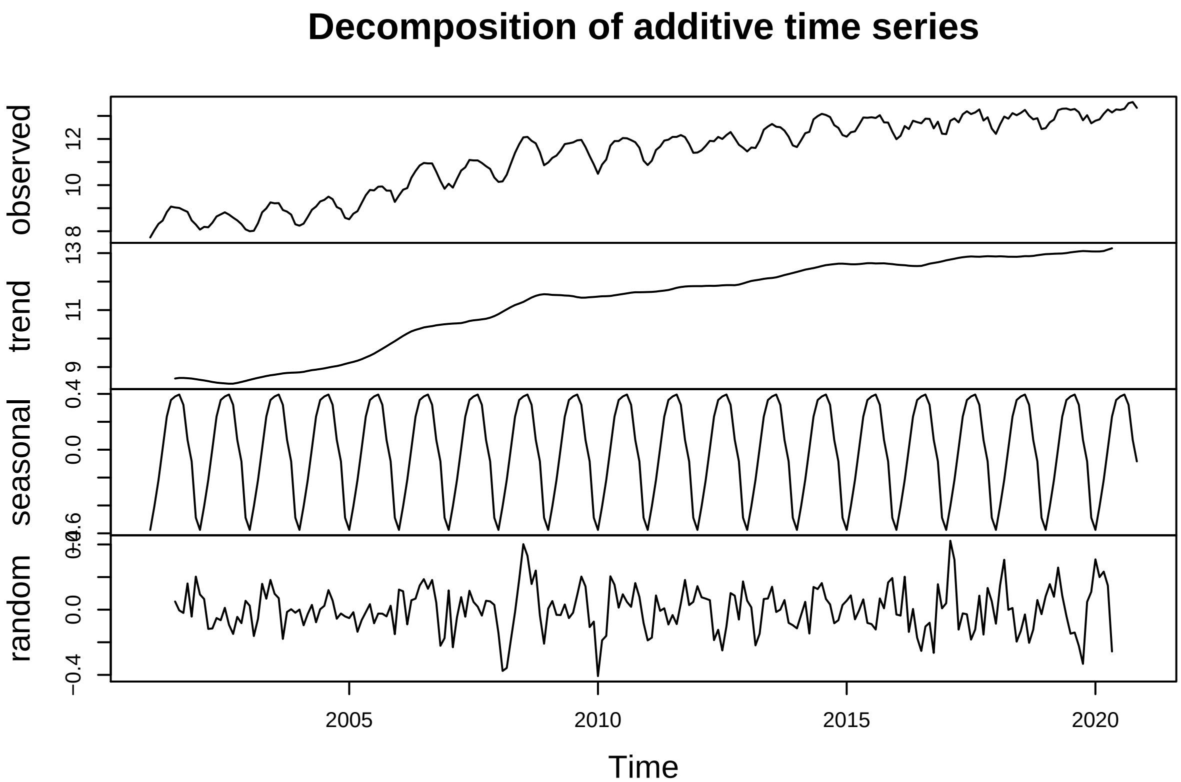

```
#Note that although the date is reversed on the data frame, since we are using the ggplot and a date ob

#ACF and PACF plots
par(mfrow=c(1,2))

ACF_Plot <- Acf(electricity_price_processed$Residential, lag = 40, plot = TRUE)
PACF_Plot <- Pacf(electricity_price_processed$Residential, lag = 40)
```



Decomposing the time series

The plots from the previous section show the data has a seasonal component. Since we are working with non-seasonal ARIMA, we need to decompose the series and eliminate the seasonality.

```
#Using R decompose function
decompose_residential_price <- decompose(ts_electricity_price[,"Residential"],"additive")
plot(decompose_residential_price)
```




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

This time we will not remove seasonality to enter the Arima(). But we still need to remove seasonal component to run stationarity test and find the order of the non-seasonal part of the ARIMA, i.e., (p,d,q).

Modeling the non-seasonal part

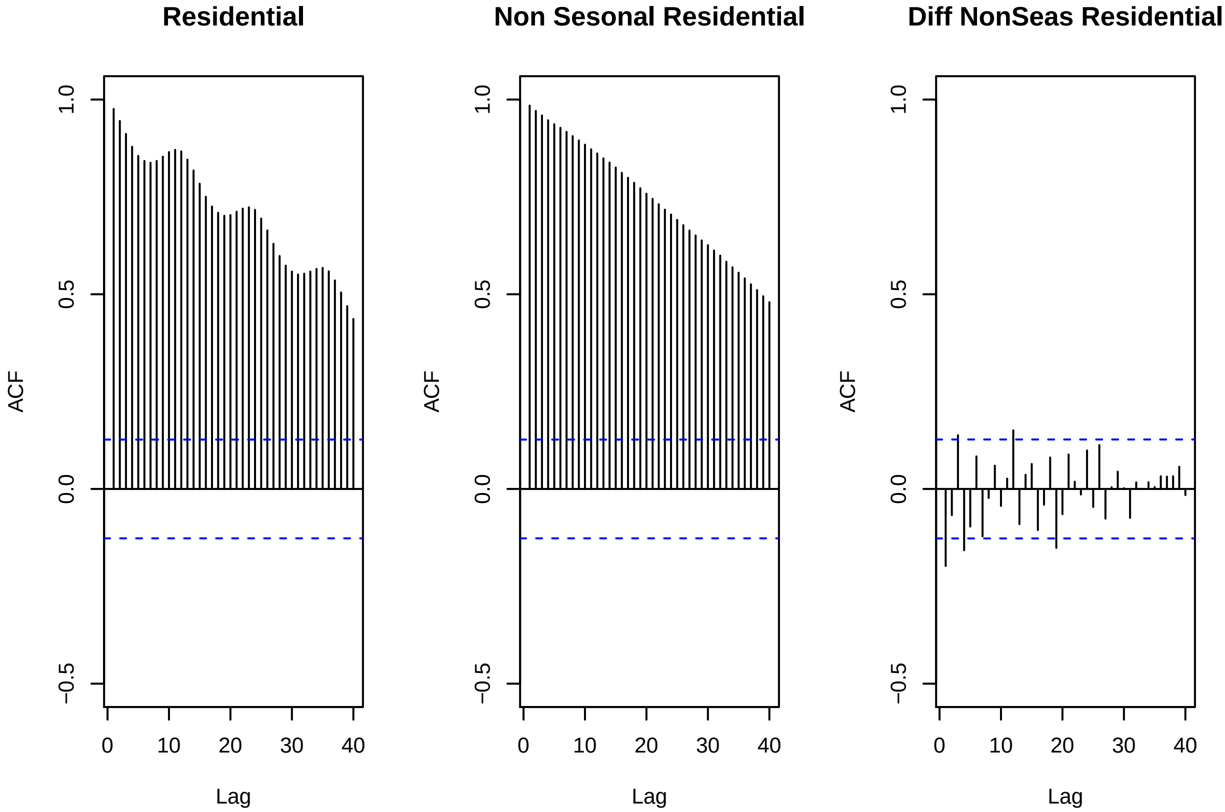
Remember from previous scripts that the electricity price series has a stochastic trend. A useful function to help determine how many times you should difference your series is the ndiffs() from package 'forecast'.

```
#Creating non-seasonal residential price time series
deseasonal_residential_price <- seasadj(decompose_residential_price)

# Find out how many time we need to difference
n_diff <- ndiffs(deseasonal_residential_price)
cat("Number of differencing needed: ",n_diff)
```

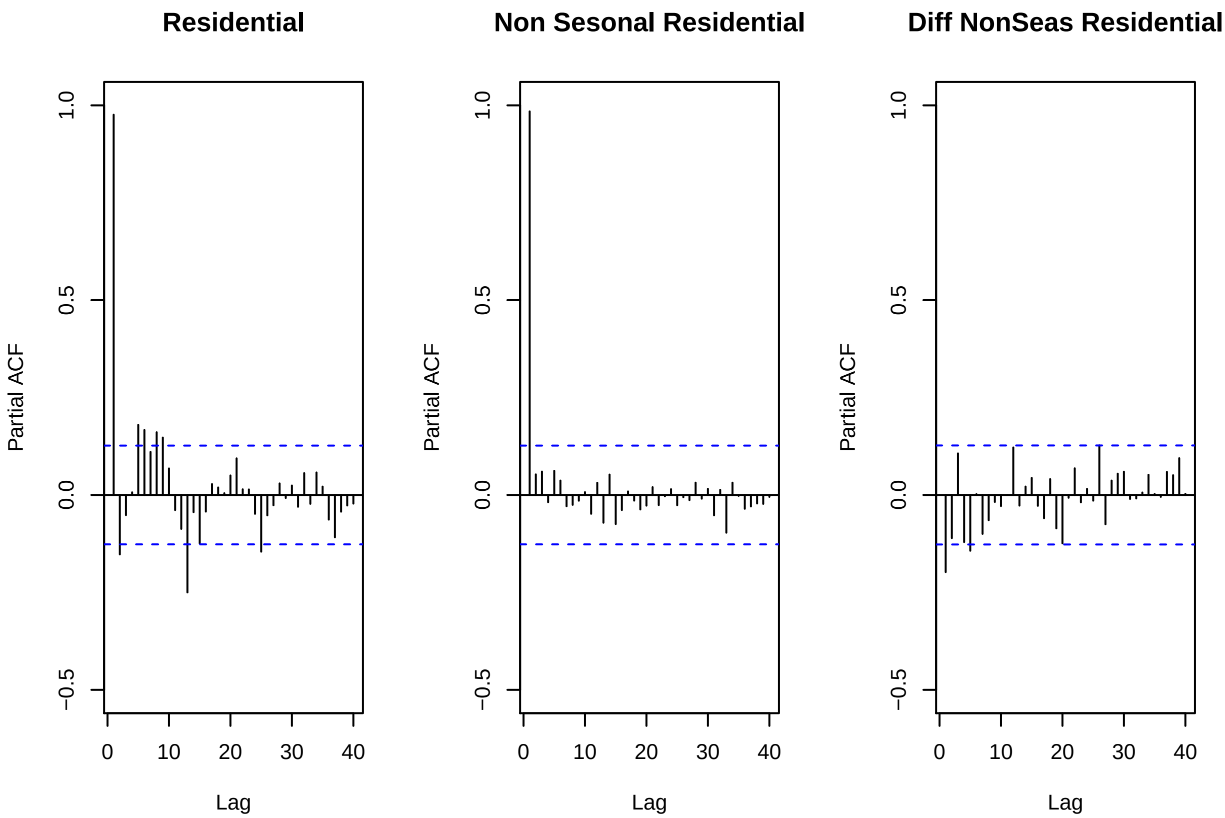
```
## Number of differencing needed:  1
```

```
#Lets difference the series once at lag 1 to remove the trend.
deseasonal_residential_price_diff <- diff(deseasonal_residential_price,differences=1,lag=1)

#Add the new series to our data frame
df_residential_full <- data.frame( Month = electricity_price_processed$Month,
                                   Residential = electricity_price_processed$Residential,
                                   NonSeasonalResidential = as.numeric(deseasonal_residential_price),
                                   ResidentialDiff = c(NA,as.numeric(deseasonal_residential_price_diff)))

#Check autocorrelation plot again
#Comparing ACFs
par(mfrow=c(1,3))
```

```r
Acf(df_residential_full$Residential,lag.max=40,main="Residential",ylim=c(-.5,1))
Acf(df_residential_full$NonSeasonalResidential,lag.max=40,main="Non Sesonal Residential",ylim=c(-.5,1))
Acf(df_residential_full$ResidentialDiff,lag.max=40,main="Diff NonSeas Residential",ylim=c(-.5,1))
```




```r
#Comparing PACFs
par(mfrow=c(1,3))
Pacf(df_residential_full$Residential,lag.max=40,main="Residential",ylim=c(-.5,1))
Pacf(df_residential_full$NonSeasonalResidential,lag.max=40,main="Non Sesonal Residential",ylim=c(-.5,1)
Pacf(df_residential_full$ResidentialDiff,lag.max=40,main="Diff NonSeas Residential",ylim=c(-.5,1))
```



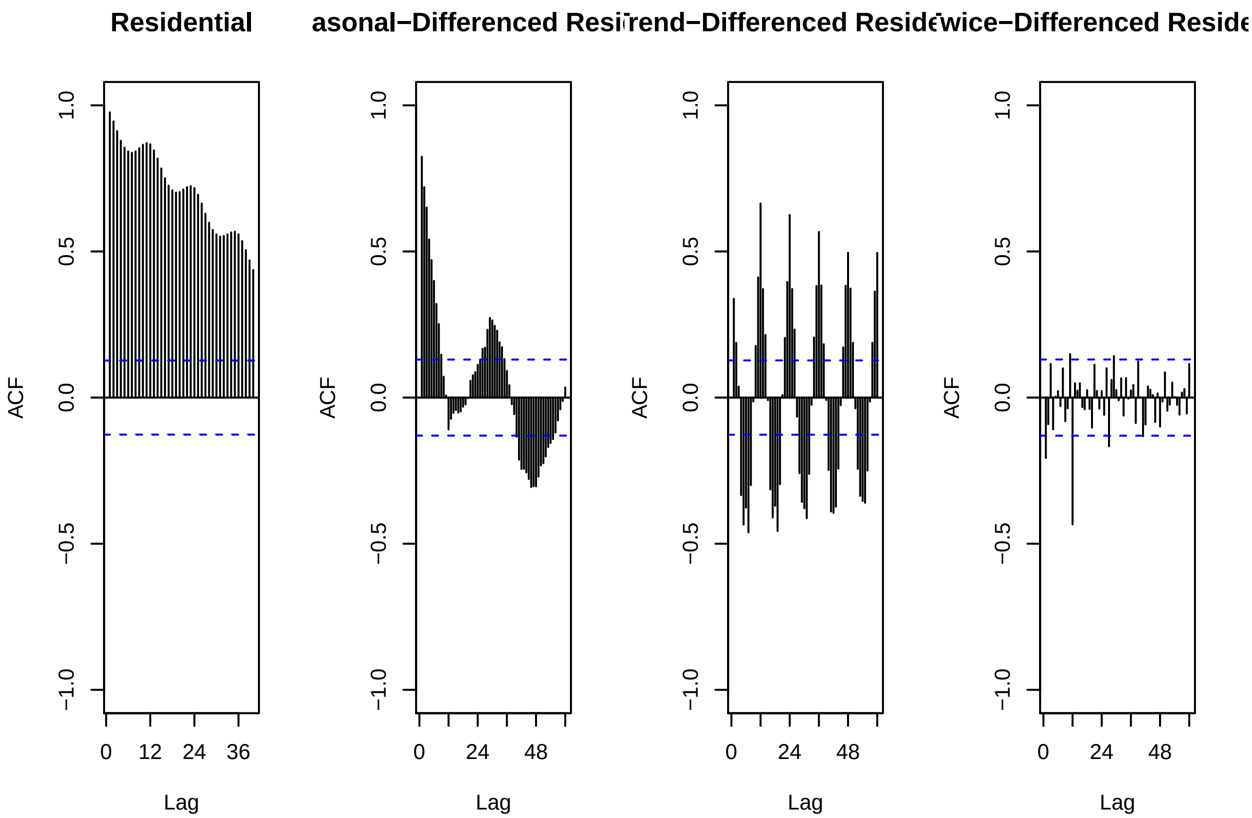
Modeling the seasonal part

I will not cover the hypothesis test associated with deterministic and stochastic seasonal component. We will use the nsdiffs() function to find if our series need differencing at the seasonal lag or not. The function will run the statistical tests internally.

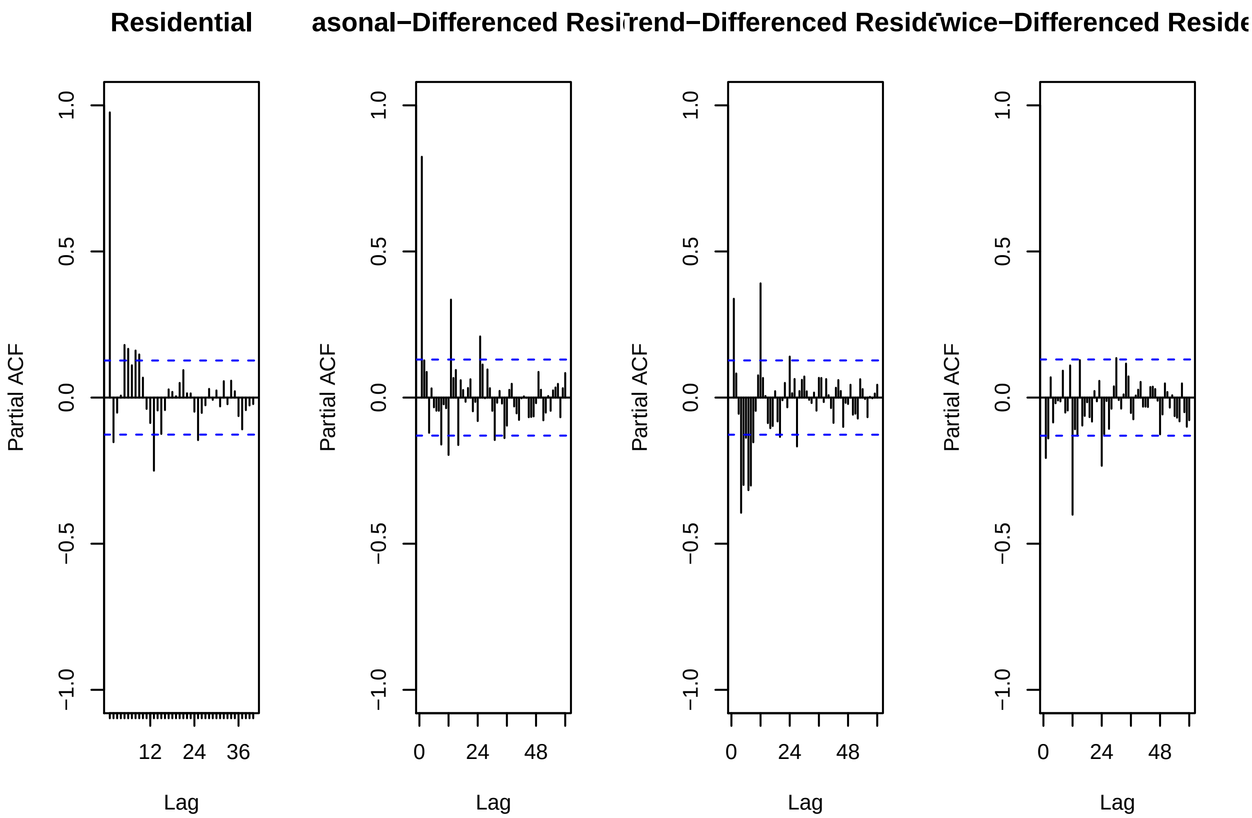
```
# Find out how many time we need to difference
ns_diff <- nsdiffs(ts_electricity_price[,"Residential"])
cat("Number of seasonal differencing needed: ",ns_diff)
```

```
## Number of seasonal differencing needed:  1
```

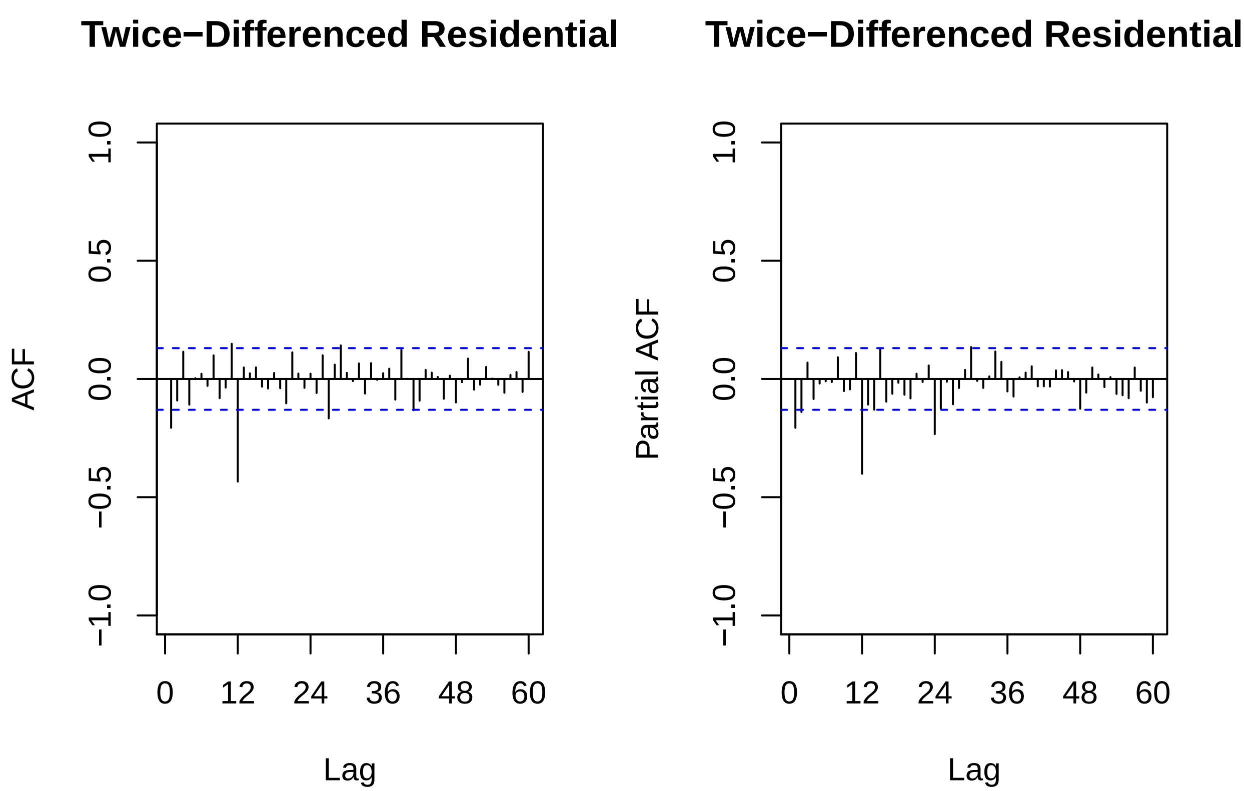
```
#Lets difference the series once at lag 12 to remove the seasonal trend.
residential_price_seas_diff <- diff(ts_electricity_price[,"Residential"],lag=12, differences=1)
residential_price_trend_diff <- diff(ts_electricity_price[,"Residential"],lag =1, differences=1) #diff
residential_price_both_diff <- diff(residential_price_trend_diff,lag =12, differences=1)

#Check autocorrelation plots for differenced series
#Comparing ACFs
par(mfrow=c(1,4))
Acf(ts_electricity_price[,"Residential"],lag.max=40,main="Residential",ylim=c(-1,1))
Acf(residential_price_seas_diff,lag.max=60,main="Seasonal-Differenced Residential",ylim=c(-1,1))
Acf(residential_price_trend_diff,lag.max=60,main="Trend-Differenced Residential",ylim=c(-1,1))
Acf(residential_price_both_diff,lag.max=60,main="Twice-Differenced Residential",ylim=c(-1,1))
```




```
#Comparing PACFs
par(mfrow=c(1,4))
Pacf(ts_electricity_price[,"Residential"],lag.max=40,main="Residential",ylim=c(-1,1))
Pacf(residential_price_seas_diff,lag.max=60,main="Seasonal-Differenced Residential",ylim=c(-1,1))
Pacf(residential_price_trend_diff,lag.max=60,main="Trend-Differenced Residential",ylim=c(-1,1))
Pacf(residential_price_both_diff,lag.max=60,main="Twice-Differenced Residential",ylim=c(-1,1))
```




```
#Plot ACF and PACF for twice-differenced series - Steps 3 (order of non-seasonal) and 5 ) order of seas
par(mfrow=c(1,2))
Acf(residential_price_both_diff,lag.max=60,main="Twice-Differenced Residential",ylim=c(-1,1))
Pacf(residential_price_both_diff,lag.max=60,main="Twice-Differenced Residential",ylim=c(-1,1))
```



Look at the twice differenced series to identify model order.

Note that we look at the ACF and PACF plot of the differenced series to try to find the order of the model. Here when we look at the first 12 lags for ACF & PACF we don't see slow decays but it looks like we have cut offs at lag 1 on both plots indicating an ARMA(p=1,q=1), and we know from ndiffs that d=1.

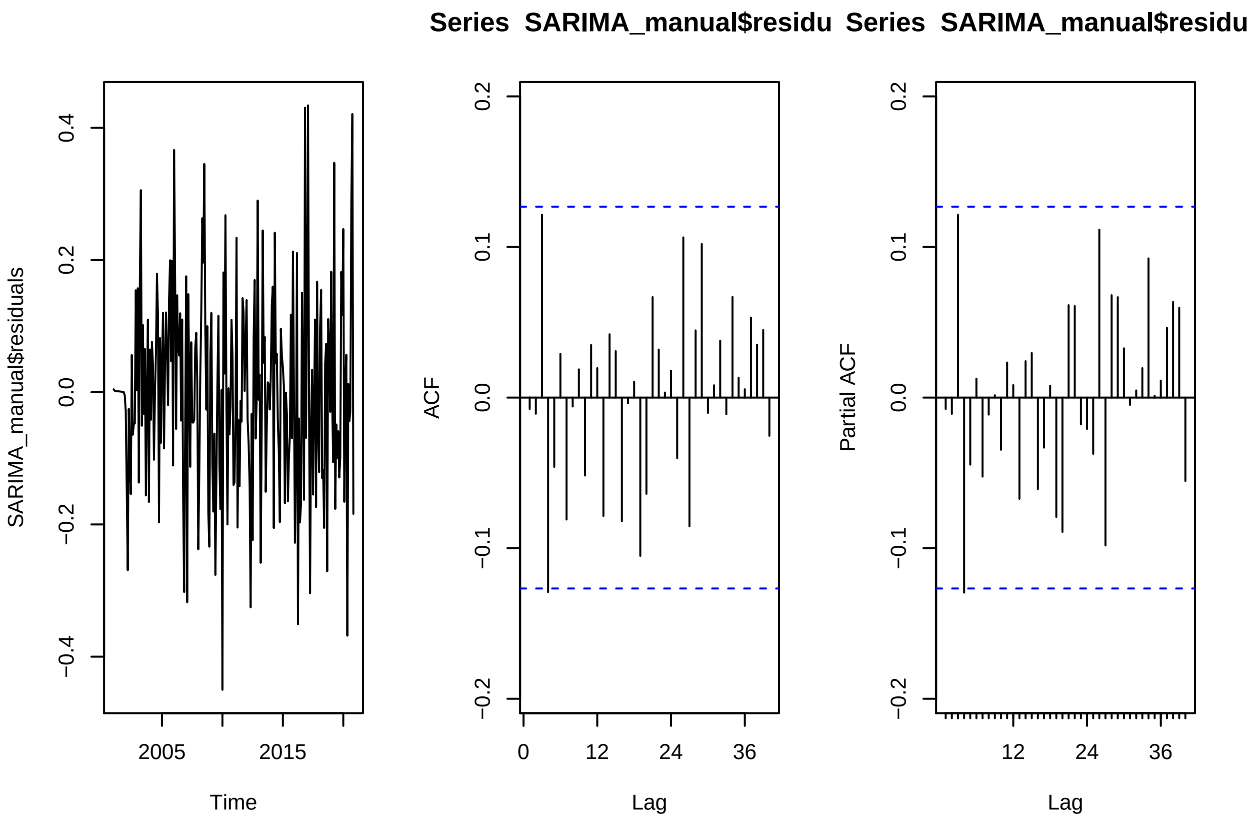
Now let's look at seasonal lags only (12,24,36,48). ACF has one spike at 12 and PACF has 2 spikes one at 12 and one at 24. This is an indication of a seasonal moving average (SMA). Therefore the order of seasonal component is P=0 and Q=1. We know from nsdiffs that D=1.

Manually fitting seasonal ARIMA to original series

```
SARIMA_manual <- Arima(ts_electricity_price[,"Residential"],order=c(1,1,1),seasonal=c(0,1,1),include.dr
print(SARIMA_manual)
```

```
## Series: ts_electricity_price[, "Residential"]
## ARIMA(1,1,1)(0,1,1)[12]
##
## Coefficients:
##          ar1      ma1     sma1
##       0.2957  -0.5393  -0.7534
## s.e.  0.3612   0.3266   0.0620
##
## sigma^2 estimated as 0.02352:  log likelihood=99.49
## AIC=-190.98   AICc=-190.8   BIC=-177.3
```

```
par(mfrow=c(1,3))
ts.plot(SARIMA_manual$residuals)
Acf(SARIMA_manual$residuals,lag.max=40)
Pacf(SARIMA_manual$residuals,lag.max=40)
```

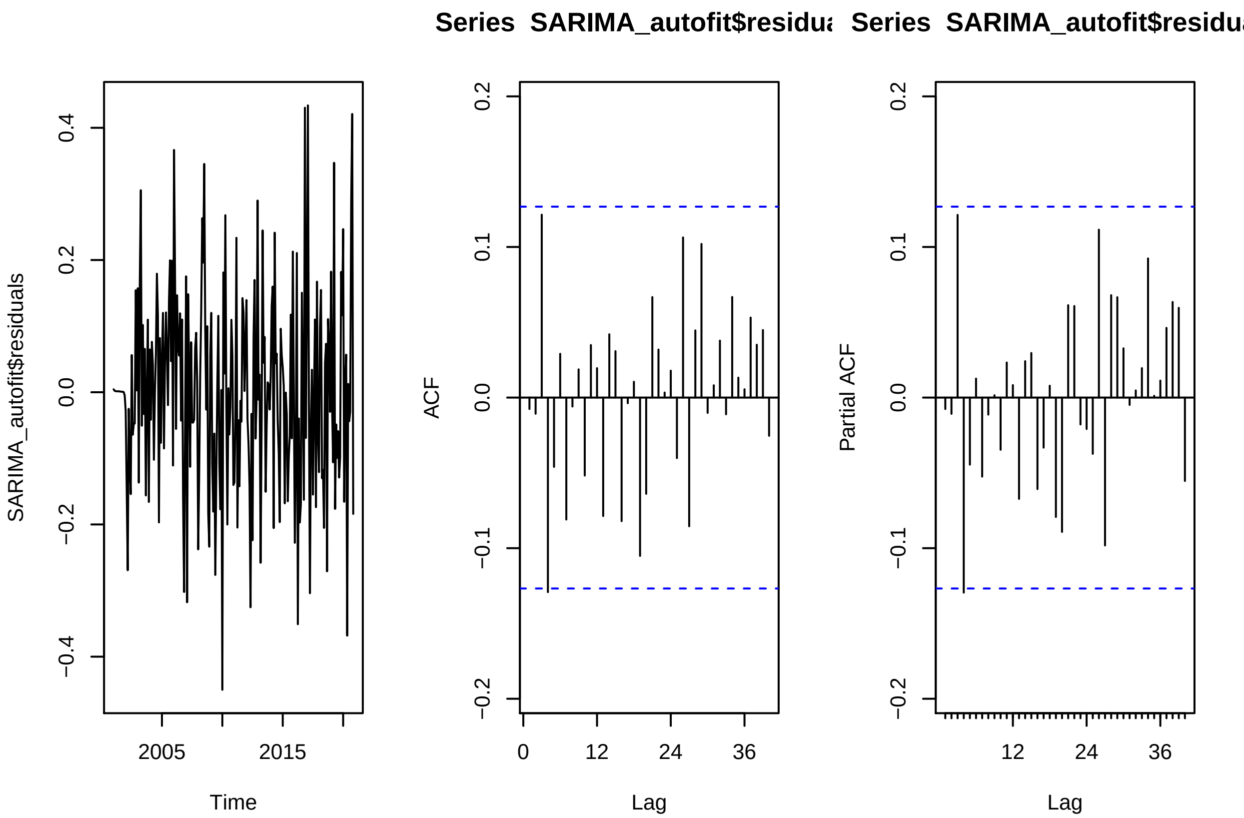


Automatically fitting seasonal ARIMA to original series

```
SARIMA_autofit <- auto.arima(ts_electricity_price[,"Residential"])
print(SARIMA_autofit)
```

```
## Series: ts_electricity_price[, "Residential"]
## ARIMA(1,1,1)(0,1,1)[12]
##
## Coefficients:
##          ar1      ma1     sma1
##       0.2957  -0.5393  -0.7534
## s.e.  0.3612   0.3266   0.0620
##
## sigma^2 estimated as 0.02352:  log likelihood=99.49
## AIC=-190.98   AICc=-190.8   BIC=-177.3
```

```
par(mfrow=c(1,3))
ts.plot(SARIMA_autofit$residuals)
Acf(SARIMA_autofit$residuals,lag.max=40)
Pacf(SARIMA_autofit$residuals,lag.max=40)
```



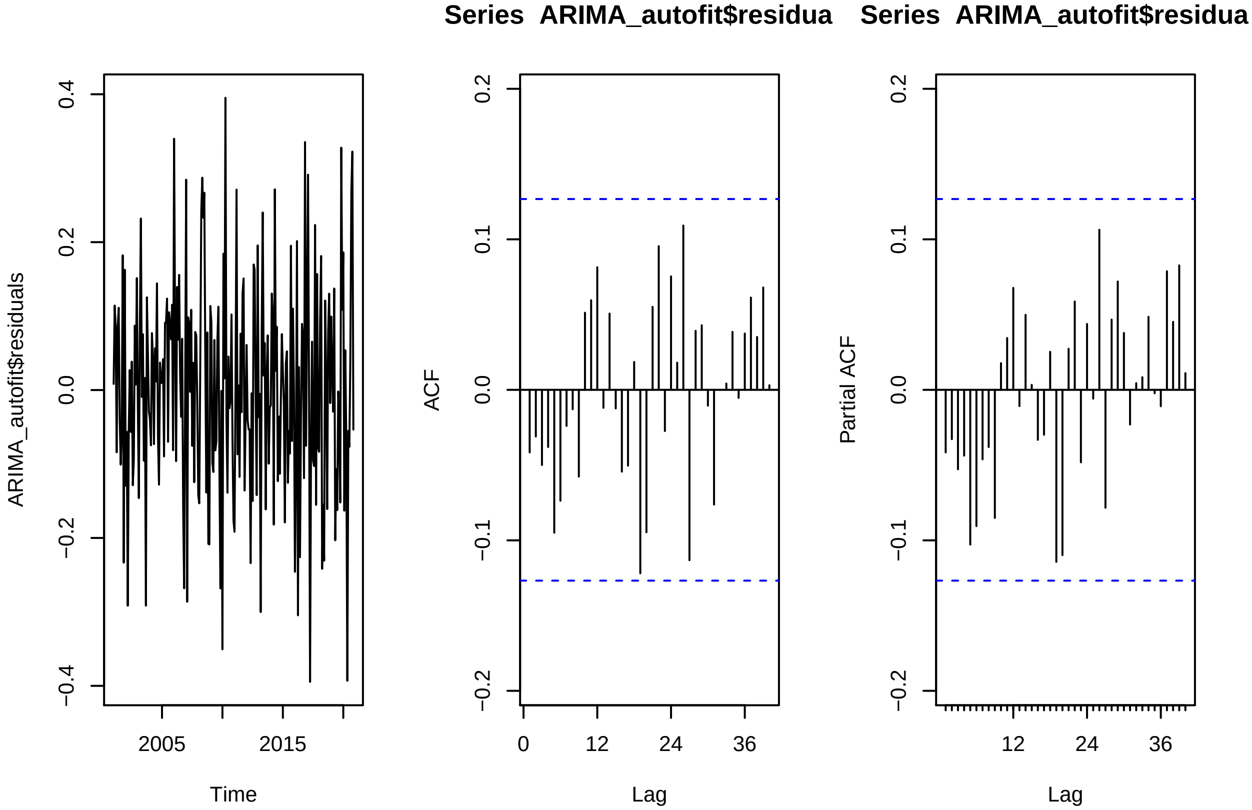
Automatically fitting ARIMA to deseasonal series

Recall from M6 that the best fit for the non-seasonal time series is a ARIMA(2,1,2) with drift.

```
ARIMA_autofit <- auto.arima(deseasonal_residential_price,max.D=0,max.P = 0,max.Q=0)
print(ARIMA_autofit)
```

```
## Series: deseasonal_residential_price
## ARIMA(2,1,2) with drift
##
## Coefficients:
##          ar1      ar2     ma1     ma2   drift
##      -0.9488  -0.8484  0.8040  0.7078  0.0217
## s.e.  0.0867   0.1001  0.1206  0.1391  0.0083
##
## sigma^2 estimated as 0.02052:  log likelihood=127.14
## AIC=-242.29   AICc=-241.93   BIC=-221.46
```

```
par(mfrow=c(1,3))
ts.plot(ARIMA_autofit$residuals)
Acf(ARIMA_autofit$residuals,lag.max=40)
Pacf(ARIMA_autofit$residuals,lag.max=40)
```



Discussion: Which one to do? ARIMA or SARIMA?