Faculty of Engineering

Department of Systems Design Engineering

SYDE 631: Time-Series Modelling

Course Project

Modelling Electricity Demand using SARIMA and HOLT-WINTERS

Prepared by

Rahij Imran Gillani

rgillani@uwaterloo.ca

University of Waterloo

200 University Avenue West

Waterloo, Ontario, Canada

Table of Contents

Modelling Electricity Demand using SARIMA and Holt-Winters

Abstract

Electricity demand prediction in today's world plays a very crucial role as it helps plan load allocations and also helps in making decisions beforehand regarding the upgrading of the infrastructure for the generation and transmission of electricity. Moreover, forecasting electricty's demand greatly helps cut down costs as well, because it helps with better planning and decision making. Generally energy consumption time series predictions are difficult to model because of the presence of complex linear and non-linear patterns. In this project we will be using the Autoregressive Integrated Moving Average (ARIMA), seasonal ARIMA (SARIMA) and Holt-Winters method to forecast the future demand of electricity in Toronto.

1. Introduction

1.1 Dataset

The dataset used for this project can be downloaded from the link:

http://reports.ieso.ca/public/DemandZonal/.

The data is the hourly consumption of electricity for a number of zones in Ontario, from the period 2003 to 2019. For this project we are only using the data for Toronto's consumption.

Units: MW (Mega Watts)

Dataset metrics: 6045 readings

Time granularity: Hour

Time range: 2003-05-01 – 2019-11-17

1.2 Types of Models

For this project we will be using the ARIMA model mainly, and will be comparing its results with the Holt-Winters model.

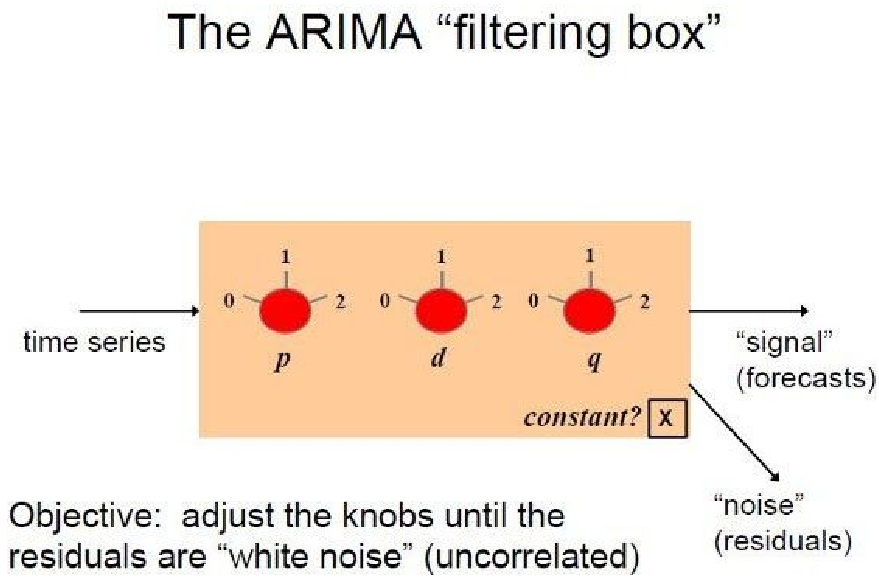
1.2.1 Arima

ARIMA, short for 'Auto Regressive Integrated Moving Average' is a class of models that explains a given time series based on its past values, i.e its own lags and the lagged forecast errors, so that equation can be used to forecast future values. The ARIMA model takes in the three parameters "p, d, q" as input.

"p" is the number of autoregressive terms "AR( )". It allows us to include the effect of past values into our model.

"d" is the number of nonseasonal differences needed for stationarity.

"q" is the number of lagged forecast errors in the prediction equation "MA( )". This allows us to set the error of our model as a linear combination of the error values observed at previous time points. [1]



1.2.2 Holt-Winters

Holt-Winters method also known as Triple Exponential Smoothing, uses exponential smoothing to encode lots of values from the past and use them to predict "typical" values for the present and future. The Holt-Winters seasonal method comprises the forecast equation and three smoothing equations — one for the level ℓt, one for the trend bt, and one for the seasonal component s_t, with corresponding smoothing parameters α, $\beta*$ and γ.

There are two variations to this method that differ in the nature of the seasonal component. The additive method is preferred when the seasonal variations are roughly constant throughout the series, while the multiplicative method is preferred when the seasonal variations are changing proportional to the level of the series. With the additive method, the seasonal component is expressed in absolute terms in the scale of the observed series, and in the level equation the series is seasonally adjusted by subtracting the seasonal component. Within each year, the seasonal component will add up to approximately zero. With the multiplicative method, the seasonal component is expressed in relative terms (percentages), and the series is seasonally adjusted by dividing through by the seasonal component. Within each year, the seasonal component will sum up to approximately m. [2]

1.2.2.1 Additive method

The component form for the additive method is:

$$
\begin{aligned}
\hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t+h-m(k+1)} \\
\ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}
\end{aligned}
$$

where k is the integer part of (h−1)/m, which ensures that the estimates of the seasonal indices used for forecasting come from the final year of the sample. The level equation shows a weighted average between the seasonally adjusted observation (yt−st−m) and the non-seasonal forecast (ℓt−1+bt−1) for time t. The trend equation is identical to Holt's linear method. The seasonal equation shows a weighted average between the current seasonal index, (yt−ℓt−1−bt−1), and the seasonal index of the same season last year (i.e. m time periods ago).

The equation for the seasonal component is often expressed as

$$s_t = \gamma^*(y_t - \ell_t) + (1-\gamma^*)s_{t-m}.$$

1.2.2.2 Multiplicative method

The component form for the multiplicative method is:

$$
\begin{aligned}
\hat{y}_{t+h|t} &= (\ell_t + hb_t)s_{t+h-m(k+1)} \\
\ell_t &= \alpha \frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma \frac{y_t}{(\ell_{t-1} + b_{t-1})} + (1-\gamma)s_{t-m}
\end{aligned}
$$

2. Exploratory Data Analysis

2.1 Data Processing

This step included extracting only those columns from the dataset that were required, and converting the hourly data into daily and monthly data. The maximum values from each day and month were chosen respectively to build two time series (daily and monthly) on which we performed our analysis.

```
time_index <- seq(from = as.Date("2003-05-01"),
                  to = as.Date("2019-11-17"), by = "day")
timeSeries = xts(myList[(1:nrow(myList)),1], order.by = time_index)
attr(timeSeries, 'frequency') <- 365
```

Creating a Daily Time Series

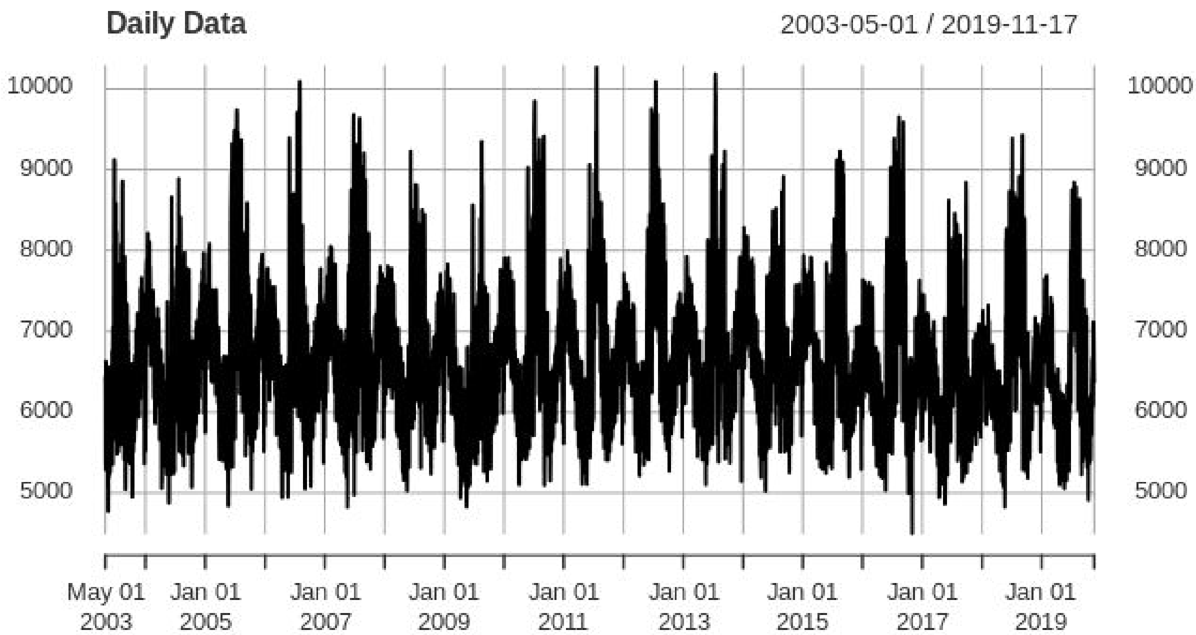
```
2003-05-01 6447
2003-05-02 6318
2003-05-03 5435
2003-05-04 5291
.
.
.
2019-11-17 6364
```

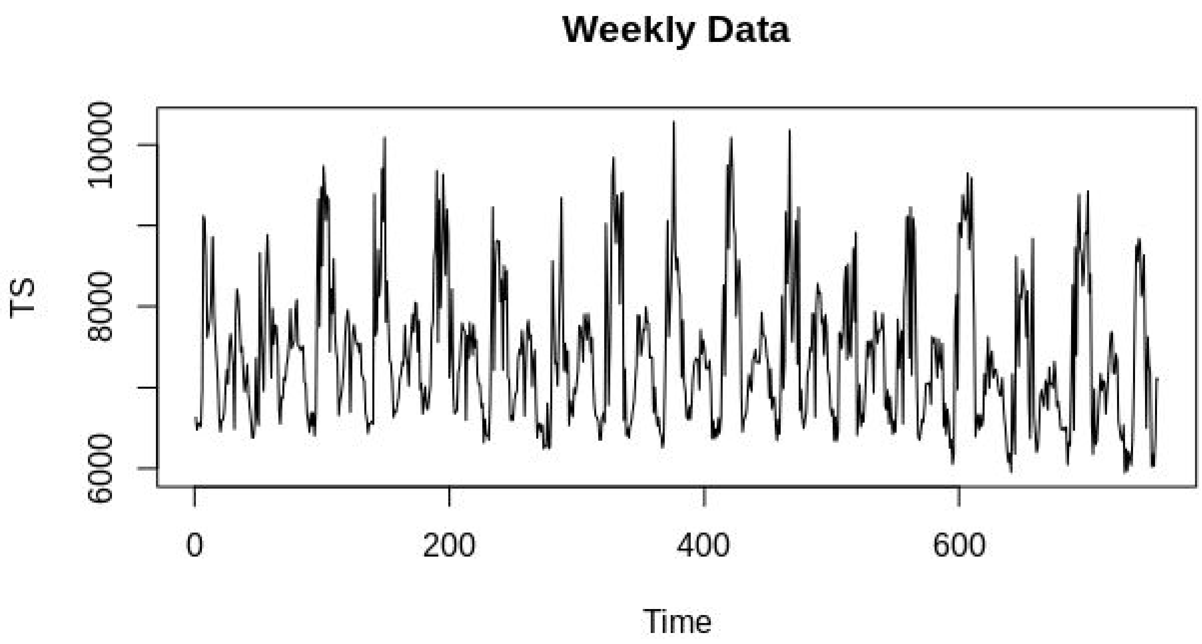
Creating a Monthly Time Series

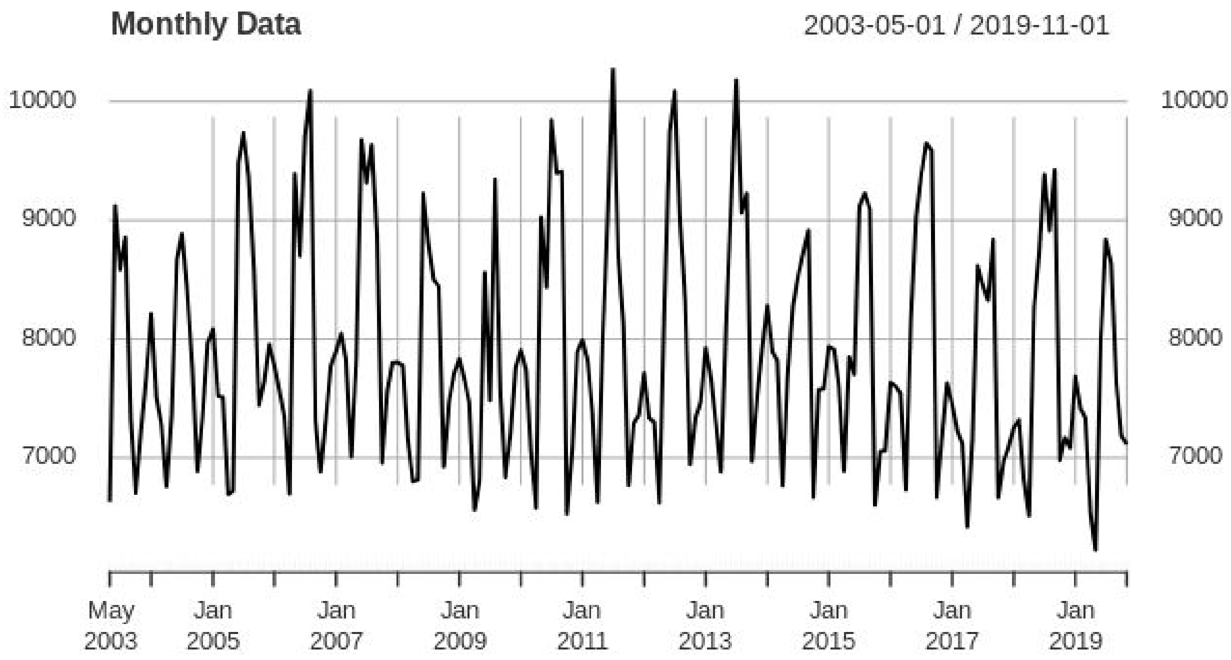
```
i=1
len = length(timeSeries)
while(i < len){
  check = as.integer(days_in_month(time(timeSeries[i])))
  if((i+check-1)>len){
    TS <- c(TS, max(timeSeries[i:len]))
  }
  else{
    TS <- c(TS, max(timeSeries[i:(i+check-1)]))
  }
  i = i+check
}
TS = data.frame(TS)
time_index1 <- seq(from = as.Date("2003-05-01"),
                   to = as.Date("2019-11-17"), by = "month")
TS = xts(TS[(1:nrow(TS)),1], order.by = time_index1)
attr(TS, 'frequency') <- 12
```

```
2003-05-01  6629
2003-06-01  9121
2003-07-01  8582
2003-08-01  8857
.
.
.
2019-11-01  7115
```

2.2 Time-Series Plots







Without transformation, time series datasets rarely satisfy all assumptions of linear regression such as normality, homoscedasticity, and independence of residuals, especially if the dataset is very granular. Hence, after comparing the datasets above we will be choosing the monthly dataset for modeling in this project.

2.3 Visual Interpretation and Data Characteristics

On observing the three time series plots shown above of the dataset, we notice that the data has the following characteristics:

- **Constant Mean**
  - The mean across the series of plots remain constant.
- **Constant Variance**
  - The variance across the series also remains constant.
- **No Trend**
  - The dataset follows no particular trend throughout the years.
- **No Intervention**
  - There appears to be no intervention at any point in the time series.
- **Seasonal**
  - The data appears to have strong seasonality.
- **Non-Stationary**
  - A stationary time series is one whose properties do not depend on the time at which the series is observed. Thus, a time series with seasonality is not stationary.

3. Confirmatory Data Analysis

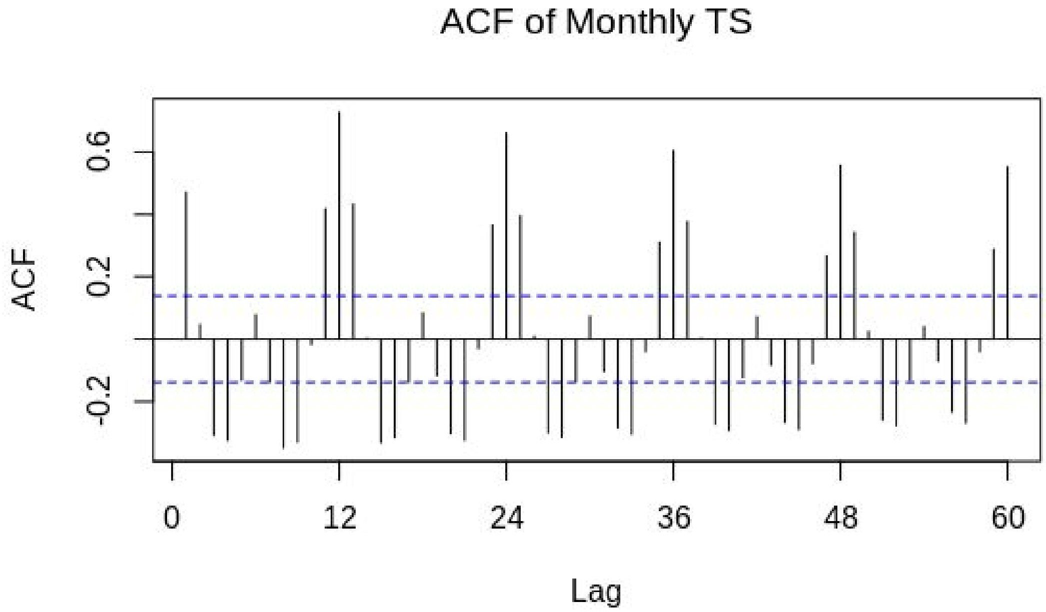
3.1 3 Stages of Model Construction

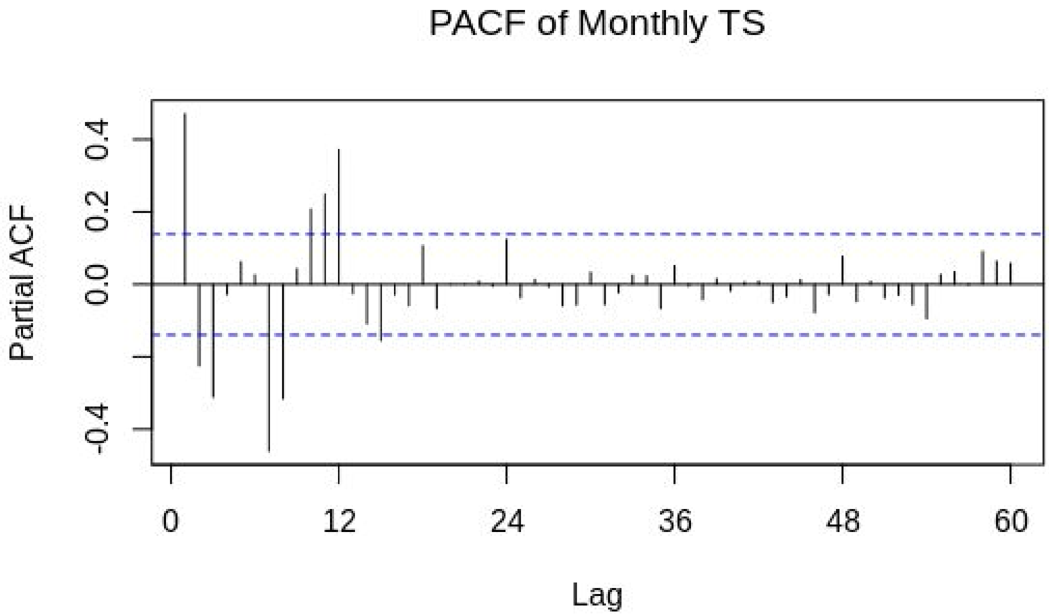
3.1.1. Identification

As our data is non stationary, it will require seasonal differencing, hence we will be using the ARIMA model.

Before Differencing

We can see by looking at the ACF plot of the time series before differencing, that even upto 60 lags the ACF plot does seem to converge to zero. Hence, the series is not stationary and seasonal differencing is required.





After Seasonal Differencing

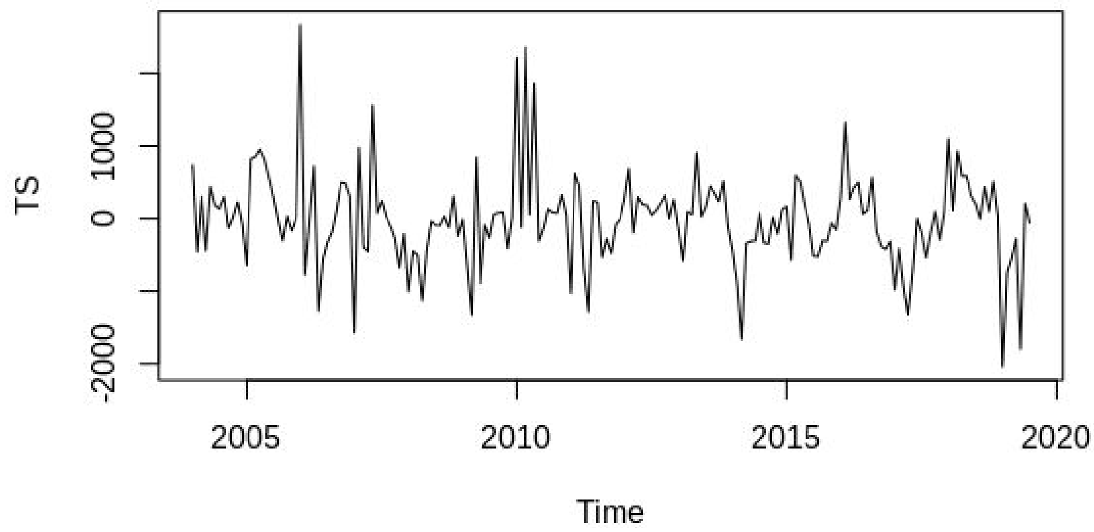
After Seasonal differencing of our time series we can observe that the seasonality of our time series has been removed, and by further observing ACF and PACF plots respectively, we see that the seasonal ACF cuts off at lag 12, and the seasonal PACF can be seen decaying at lags 12, 24, 36, 48 and then it cuts off and does not cross the confidence interval. Looking at the ACF and PACF graphs we can estimate the order of "p,q" terms for our SARIMA model.

The following two rules of thumb can help us in estimating the p,q termsl.

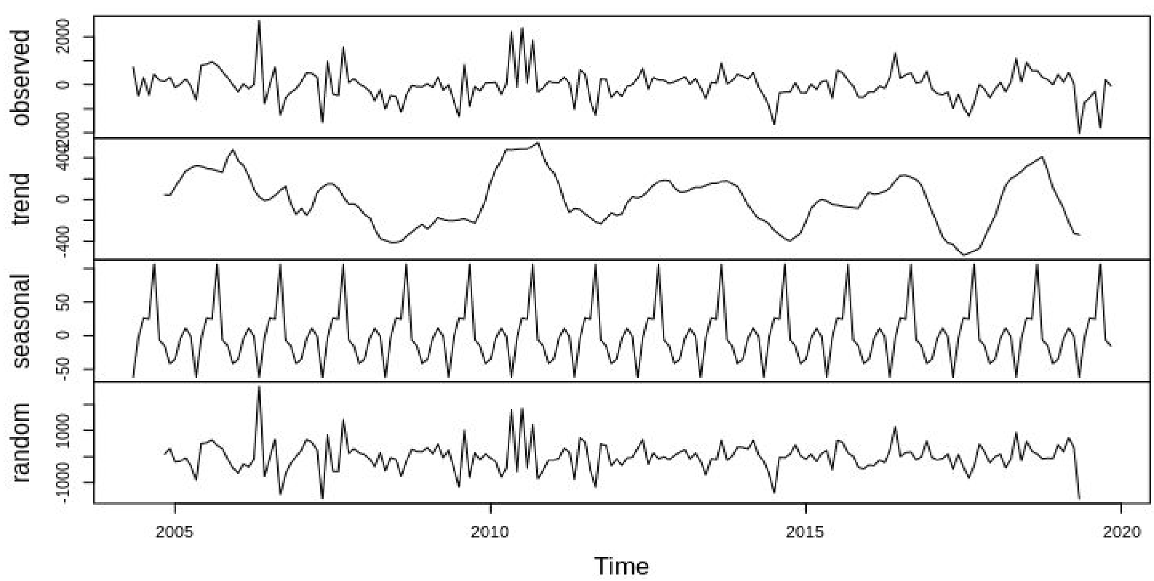
1. *If the PACF of the differenced series displays a sharp cutoff and/or the lag-1 autocorrelation is positive then consider adding one or more AR terms to the model.*
2. *If the ACF of the differenced series displays a sharp cutoff and/or the lag-1 autocorrelation is negative then consider adding an MA term to the model.*

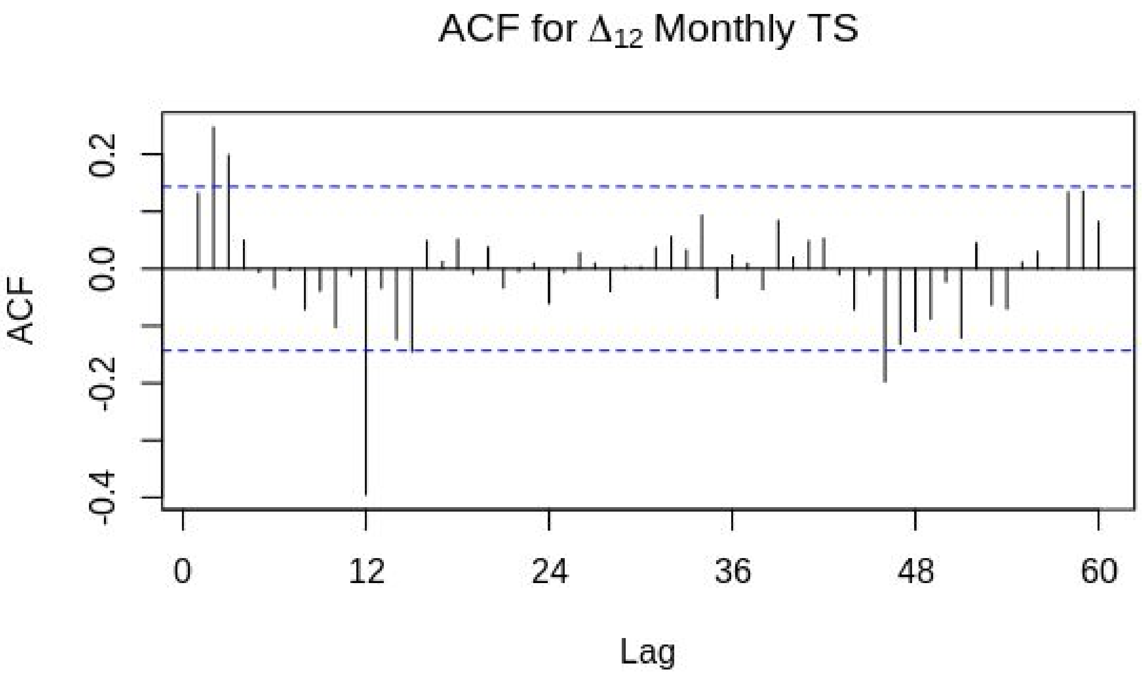
Hence, we can estimate that the model should be **ARIMA(2,0,0)(0,1,1)$_{12}$** or **ARIMA(2,0,0)(4,1,1)$_{12}$.**

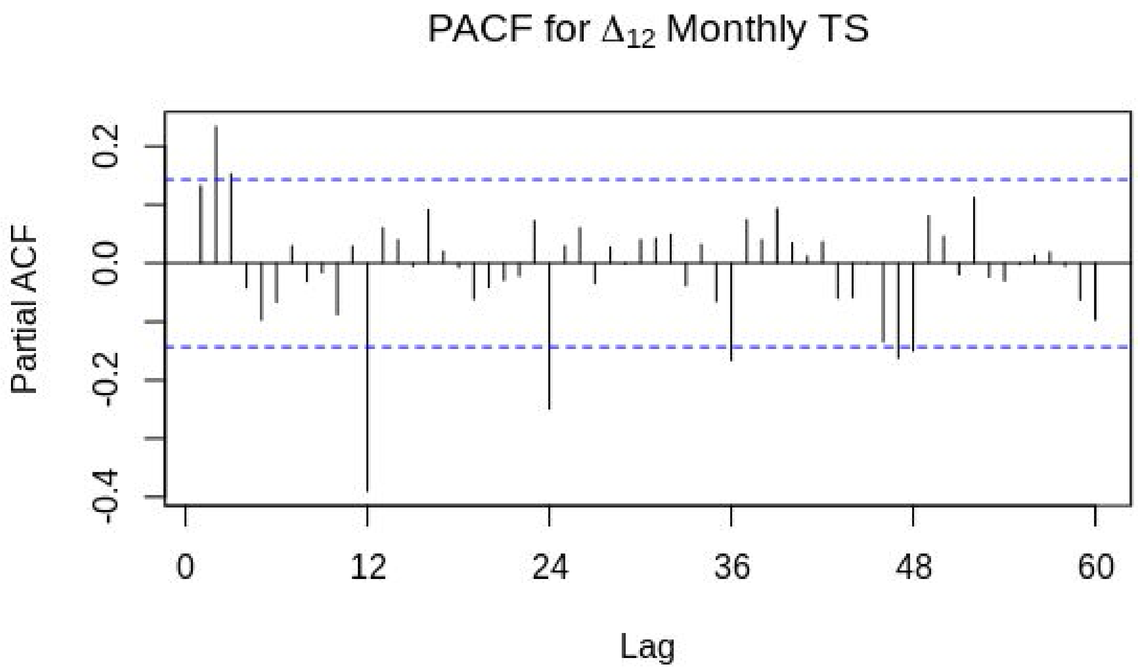
Time Series after Δ_{12}Differencing



Decomposition of additive time series







ADF Test

The Augmented Dickey Fuller Test (ADF) is the unit root test for stationarity. Unit roots can cause unpredictable results in your time series analysis. The Augmented Dickey-Fuller test can be used with serial correlation. The ADF test can handle more complex models than the Dickey-Fuller test, and it is also more powerful. [5]

The Null hypothesis for this test there is a unit root.

The alternate hypothesis is that the time series is stationary.

The adf test result for our time series indicates towards the stationarity of the data.

```
Augmented Dickey-Fuller Test

data:  diff(TS, 12)

Dickey-Fuller = -5.2397, Lag order = 5, p-value = 0.01

alternative hypothesis: stationary
```

The more negative the ADF value is, the more likely we are to reject the null hypothesis. Hence, it can be concluded from the ADF and p-values that our dataset is now stationary.

3.1.2. Parameter Estimation

Now we further test our ARIMA model and calculate its parameters. We divide our monthly dataset into 2 parts, training and testing.

```
train_date <- nrow(TS) *0.8
train <- TS[1:train_date,]

test <- TS[-c(1:train_date),]
```

Using the auto.arima( ) function in R we get the model **ARIMA(2,0,0)(0,1,1)$_{12}$**, which is the same as one of the models that we estimated earlier in the identification stage. The method used in estimating the parameters is the **conditional-sum-of-squares** method to find starting values, and then the **maximum likelihood method.**

3.1.2.1 Candidate models:

```
Series: train
ARIMA(2,0,0)(0,1,1)[12]

Coefficients:

         ar1     ar2     sma1

      0.0837  0.2427  -0.8399

s.e.  0.0819  0.0814   0.0815


sigma^2 estimated as 241525:  log likelihood=-1125.42

AIC=2258.84   AICc=2259.12   BIC=2270.8
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	44.81665	467.6962	302.9963	0.2786995	3.664791	0.701513	-0.01975553	NA
Test set	-361.26710	1382.9783	1163.9701	-6.3253170	15.665579	2.694884	0.48460725	1.612009

```
Series: train
ARIMA(2,0,0)(4,1,1)[12]
Coefficients:
          ar1     ar2     sar1     sar2     sar3     sar4     sma1
      -0.0228  0.2122  -0.3322  -0.4678  -0.3660  -0.2948  -0.5069
s.e.   0.0909  0.0836   0.1369   0.1136   0.1061   0.1119   0.1373

sigma^2 estimated as 216190:  log likelihood=-1118.49
AIC=2252.99   AICc=2254.03   BIC=2276.91
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	46.36732	436.2977	281.4977	0.32702	3.40949	0.6517381	-0.01962522	NA
Test set	-394.84973	1415.7624	1164.5725	-6.75033	15.71139	2.6962791	0.47403306	1.636173

```
Series: train
ARIMA(2,0,2)(0,1,1)[12]
Coefficients:
         ar1      ar2      ma1     ma2     sma1
      0.3991  -0.0662  -0.3239  0.2927  -0.8210
s.e.  0.3647   0.3463   0.3493  0.3154   0.0791

sigma^2 estimated as 245439:  log likelihood=-1125
AIC=2262.01   AICc=2262.61   BIC=2279.95
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	44.32117	468.1847	304.8319	0.2718708	3.691653	0.7057627	-0.008528745	NA
Test set	-362.67605	1385.1487	1165.1419	-6.3427600	15.683366	2.6975975	0.479705400	1.614701

3.1.2.1 Model Selection:

Akaike Information Criterion (AIC):

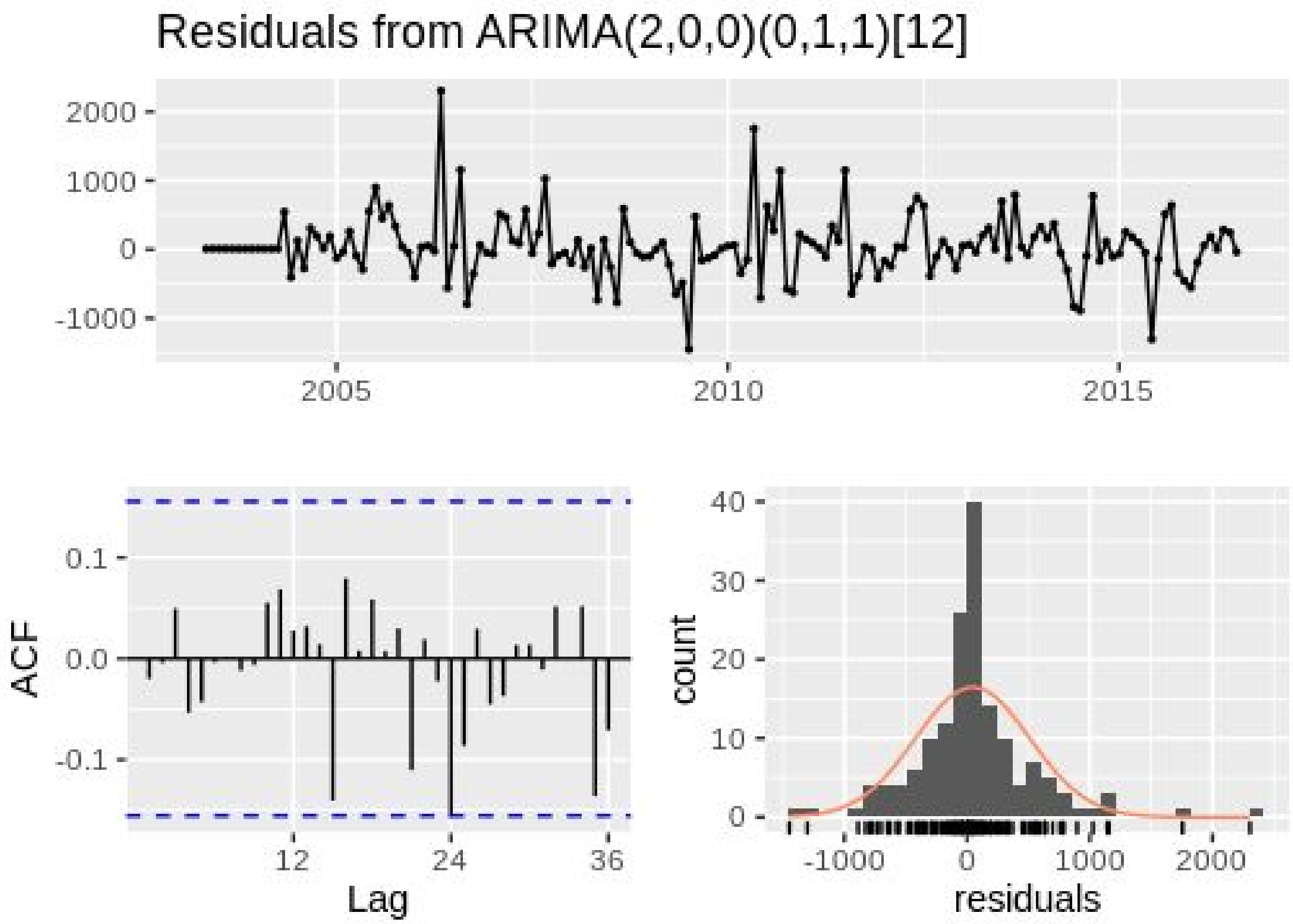
We take the help of the idea of Partial AIC study, i.e determining the AIC for set of models selected at the identification stage. According to this criterion we chose the model with the lowest AIC value. If we compare the two candidate models mentioned above we can see that the **ARIMA(2,0,0)(4,1,1)$_{12}$** model has lower AIC and AICc than **ARIMA(2,0,0)(0,1,1)$_{12}$**, and it also has smaller errors in the prediction values against the training set, but the model **ARIMA(2,0,0)(0,1,1)$_{12}$** fits better to the testing set; moreover, the addition of 4 AR parameters does not make a huge difference, hence we stick to the **ARIMA(2,0,0)(0,1,1)$_{12}$** model.

3.1.3. Diagnostic Checking

Diagnostic check ensures the selected model adequately describes the dynamics of the time series.

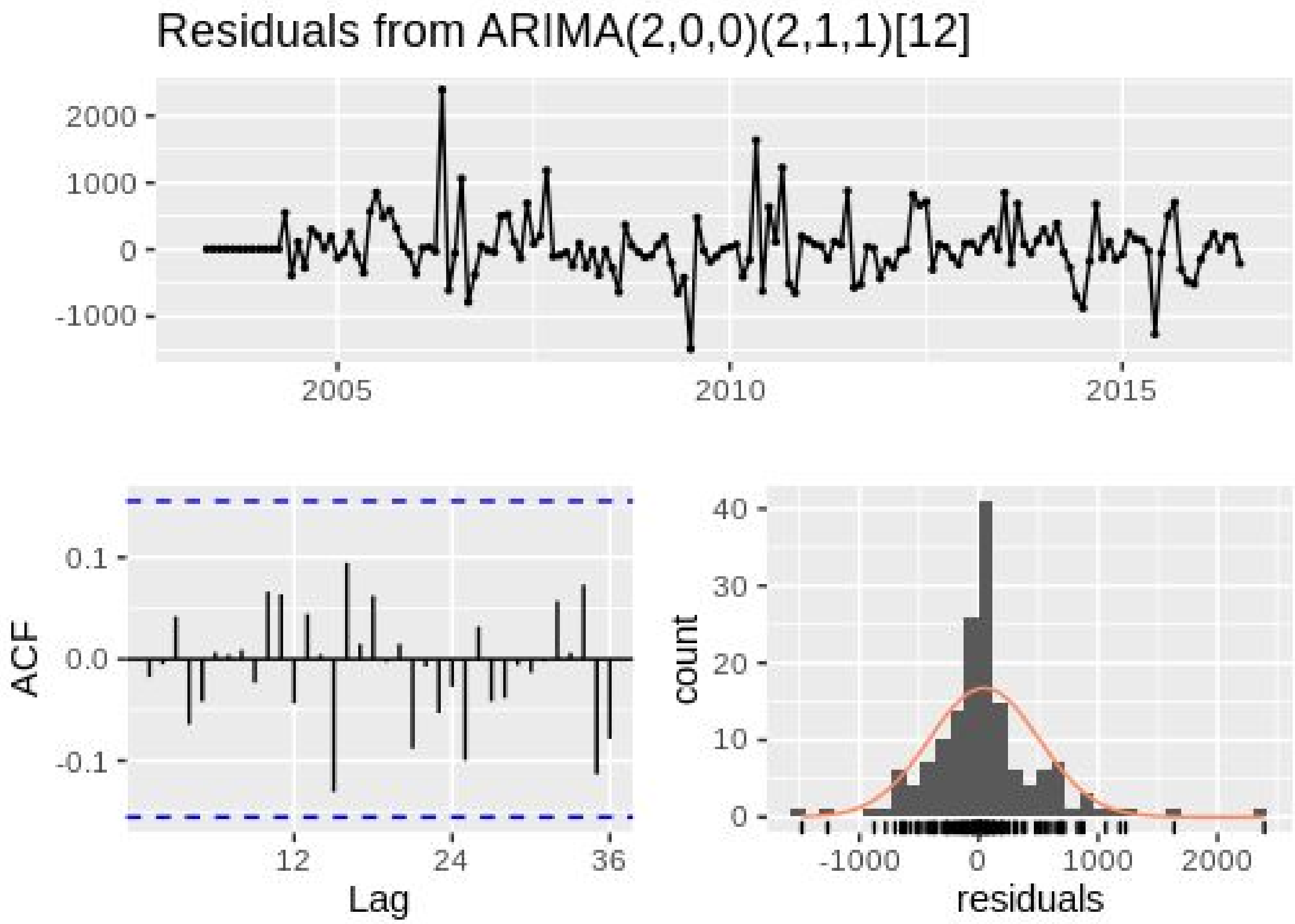
3.1.3.1 Whiteness Test:

White test is a statistical test that establishes whether the variance of the errors in a regression model is constant. Looking at the models residuals below we can see that the variance is constant.



However, in this model although the residuals seem homoscedastic and appear to have constant mean as well, but the ACF of the residuals seems to have correlation at lag 24; hence, we consider changing our model to **ARIMA(2,0,0)(2,1,1)**$_{12}$.

```
Series: train
ARIMA(2,0,0)(2,1,1)[12]
Coefficients:
         ar1     ar2     sar1      sar2      sma1
      0.0544  0.2450  0.0228   -0.1833   -0.8017
s.e.  0.0856  0.0822  0.1039    0.0912    0.0966
sigma^2 estimated as 236410:  log likelihood=-1123.24
AIC=2258.49   AICc=2259.09   BIC=2276.43
```



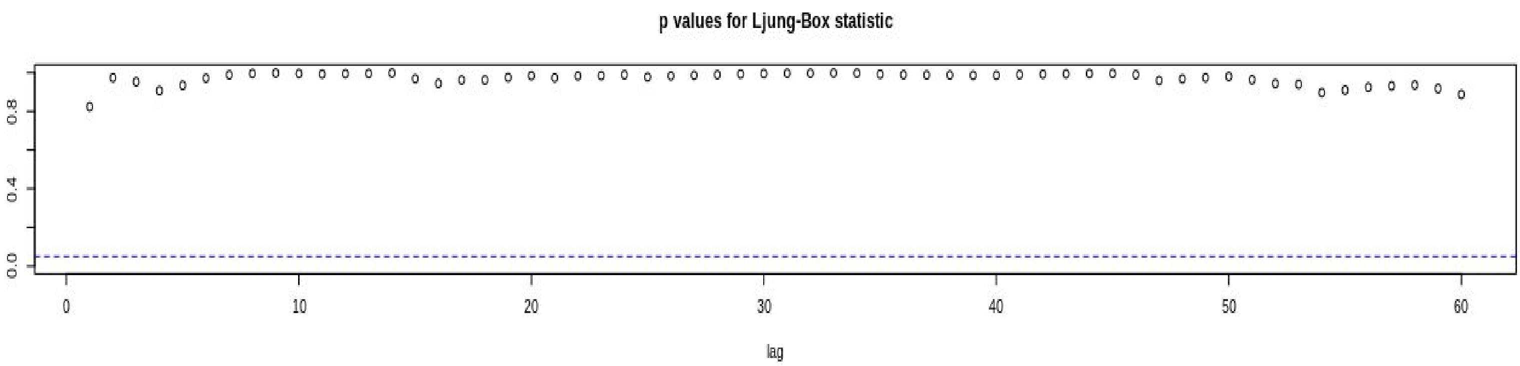
Ljung-Box test: Tests the lack of fit of a model

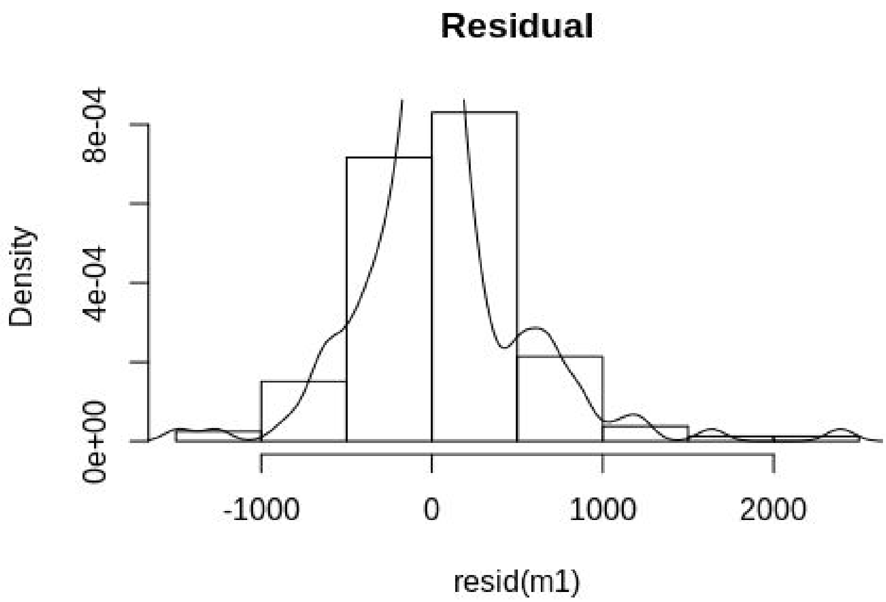
data: Residuals from ARIMA(2,0,0)(2,1,1)[12]

Q* = 11.085, df = 19, **p-value = 0.921**

Model df: 5. Total lags used: 24

A significant p - value means that the residuals are uncorrelated (i.e the model is fine).





Now our residuals for the model **ARIMA(2,0,0)(2,1,1)$_{12}$**, are uncorrelated and have a decent normal distribution.

3.1.3.1 Overfitting

Now we check if by including one or more parameters, the resulting model fits better to the time series than our current model or not. The **Likelihood Ratio** test is performed to see if the overfitted model is better than the current one.

The following models performed better according to the Likelihood Ratio test against our current **ARIMA(2,0,0)(2,1,1)$_{12}$** model. The parameters were increased one step at a time and the magnitudes of the new coefficients were also taken into account.

```
Arima(train, order=c(2,0,0),seasonal=list(order=c(3,1,1), period=12))

Arima(train, order=c(2,0,0),seasonal=list(order=c(4,1,1), period=12))

Arima(train, order=c(2,0,0),seasonal=list(order=c(4,1,2), period=12))
```

FOR EXAMPLE:

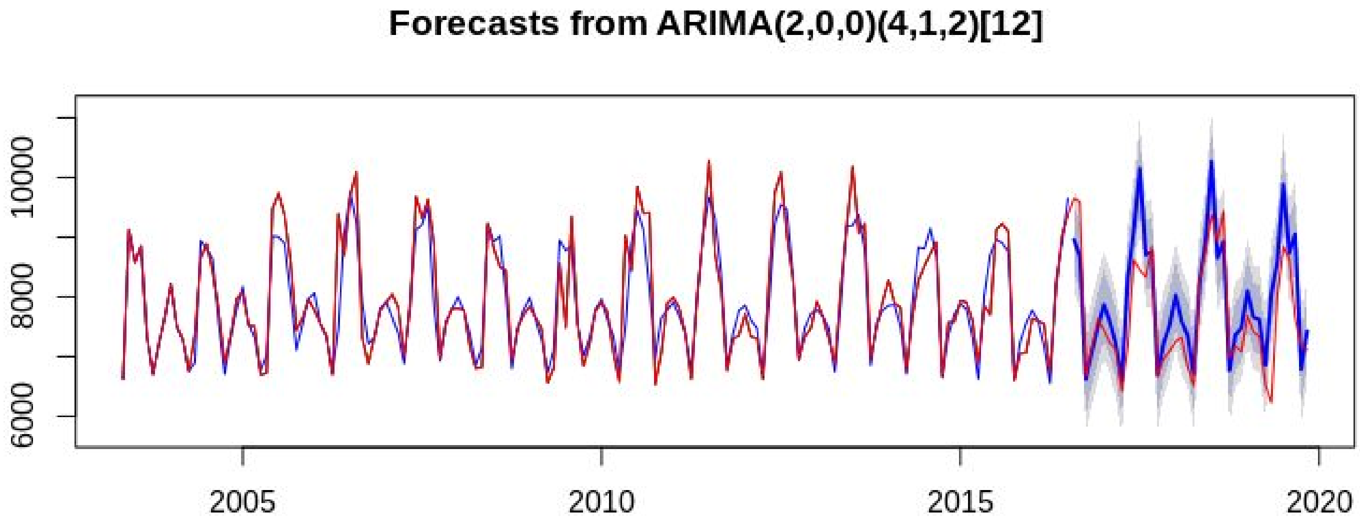
```
Likelihood ratio test

Model 1: Arima(y = train, order = c(2, 0, 0), seasonal = list(order = c(3,
    1, 1), period = 12))
Model 2: Arima(y = train, order = c(2, 0, 0), seasonal = list(order = c(4,
    1, 1), period = 12))
  #Df  LogLik Df  Chisq Pr(>Chisq)
1   7 -1121.2
2   8 -1118.5  1 5.4443    0.01963 *
```

A lower P value (<0.5) means that overfitting is **required.**

4. Forecasting

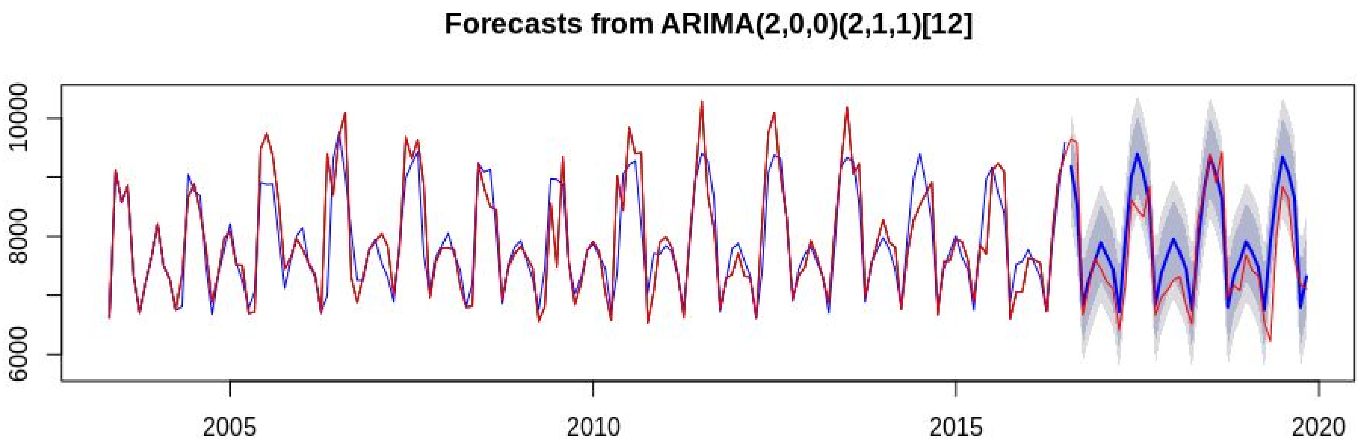
Now we use our selected models to make predictions for the future and compare them to our test dataset.

As we saw in the Diagnostic stage that according to the Likelihood Ratio Test overfitting was required; hence, we first use the new overfitted model to do forecasting and compare it with our initial model.



BLUE: Fitted Values **RED: Original Series**

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	37.66074	384.7334	248.851	0.2583434	3.012718	0.5761528	-0.02307863	NA
Test set	-271.36895	1456.2247	1243.410	-5.1933189	16.528261	2.8788081	0.55587964	1.680447



BLUE: Fitted Values **RED: Original Series**

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	44.74501	459.4922	293.689	0.288841	3.54803	0.6799641	-0.01753239	NA
Test set	-243.21261	1323.4889	1148.138	-4.746073	15.27151	2.6582297	0.59231416	1.502731

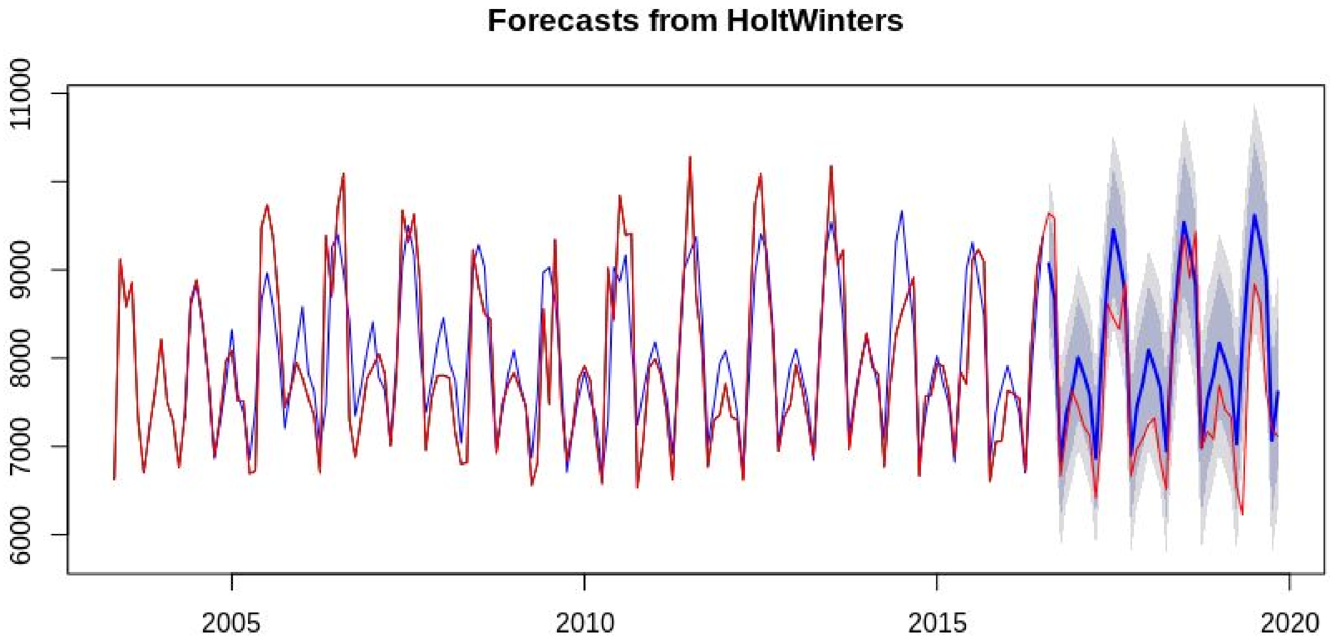
If we compare the two forecasts above we can see that the overfitted model **ARIMA(2,0,0)(4,1,2)$_{12}$** fits the training data better than the other **ARIMA(2,0,0)(2,1,1)$_{12}$**. However, our original model fits the test data better as the error between the original data and forecasted data is lesser than the overfitted model.

Same is the case with other overfitted models as shown below:

```
> model_overfitt=Arima(train, order=c(2,0,0),seasonal=list(order=c(4,1,1), period=12))
> model_overfitt %>% forecast(h = length(test) ) %>% accuracy(test)
                    ME      RMSE       MAE       MPE     MAPE      MASE        ACF1 Theil's U
Training set  46.36732  436.2977  281.4977  0.327020  3.40949 0.6517381 -0.01962522        NA
Test set    -278.27196 1390.9018 1205.6817 -5.239101 16.02966 2.7914574  0.61425715  1.589212
> model_overfitt=Arima(train, order=c(2,0,0),seasonal=list(order=c(3,1,1), period=12))
> model_overfitt %>% forecast(h = length(test) ) %>% accuracy(test)
                    ME      RMSE       MAE        MPE      MAPE      MASE        ACF1 Theil's U
Training set  43.81285  450.2834  290.8351  0.2827969  3.516456 0.6733565 -0.02078775        NA
Test set    -264.29692 1349.7340 1170.5373 -5.0308611 15.571762 2.7100891  0.60522093  1.539337
> model_overfitt=Arima(train, order=c(2,0,0),seasonal=list(order=c(3,1,2), period=12))
> model_overfitt %>% forecast(h = length(test) ) %>% accuracy(test)
                    ME      RMSE       MAE        MPE      MAPE      MASE       ACF1 Theil's U
Training set  36.27846  388.3997  252.1589  0.2383938  3.054382 0.5838114 -0.0243656        NA
Test set    -274.93649 1461.6554 1250.5776 -5.2478386 16.633332 2.8954028  0.5613741  1.687049
```

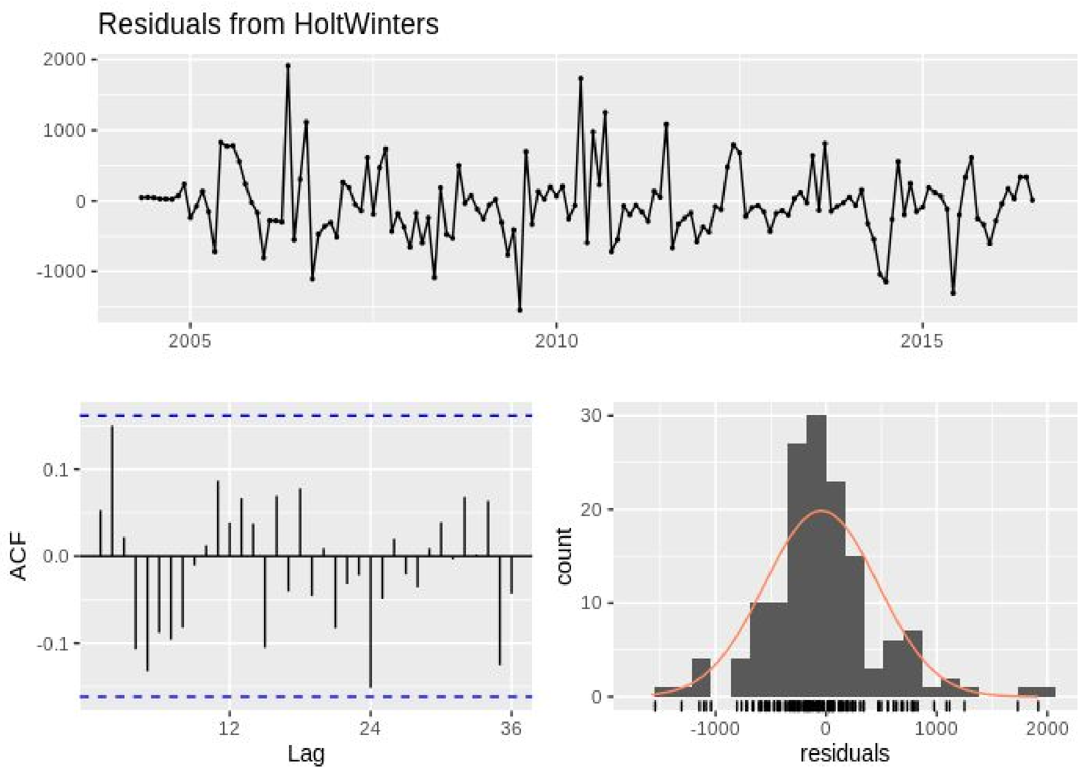
5. Seasonal ARIMA vs Holt-Winters

We now use the Holt-Winters model used for seasonal data to compare its results with the ARIMA model. The Holt-Winters model is uses exponential smoothing to encode lots of values from the past and use them to predict "typical" values for the present and future.



Holt-Winters exponential smoothing with trend and additive seasonal component.

```
HoltWinters(x=train)

Smoothing parameters:
 alpha: 0.1102925
 beta : 0.001315384
 gamma: 0.2011258

Coefficients:
            [,1]

a    8065.491326
b       6.830823
s1    997.830676
s2    590.168875
s3  -1263.410658
s4   -722.437507
s5   -464.532345
s6    -98.114455
s7   -304.174939
s8   -540.070254
s9  -1260.511827
s10  -109.366041
s11   727.621948
s12  1309.297226
```



As we can see that the Holt-Winters model's residuals are homoscedastic, are not correlated and are also aptly randomly distributed.

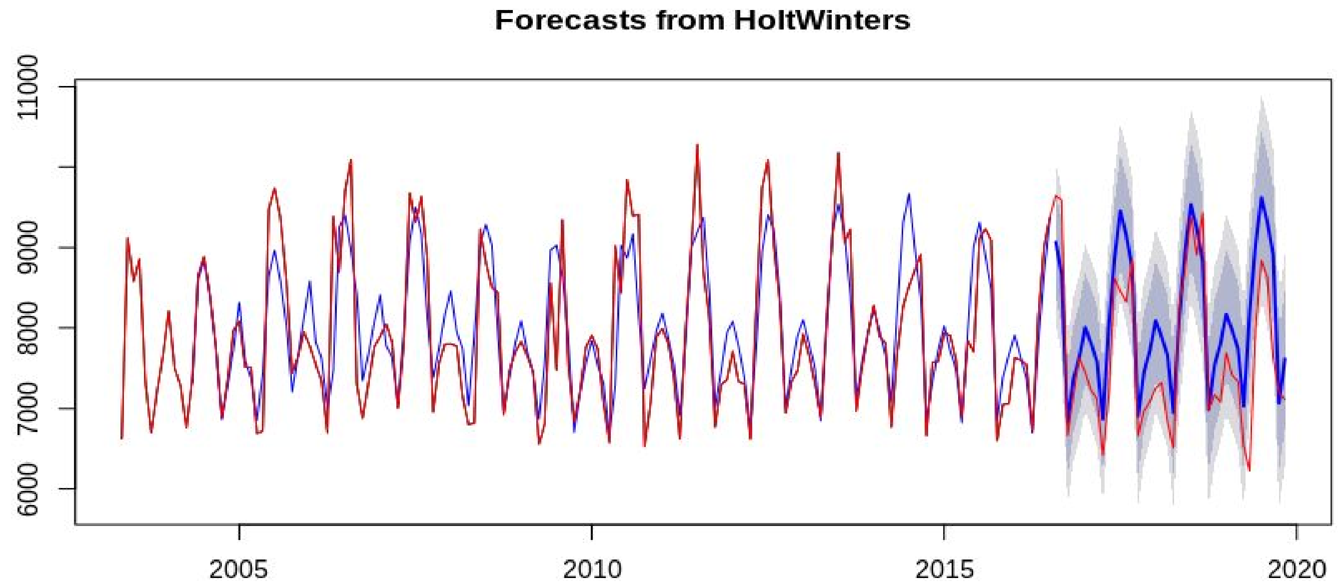
Holt-Winters Errors

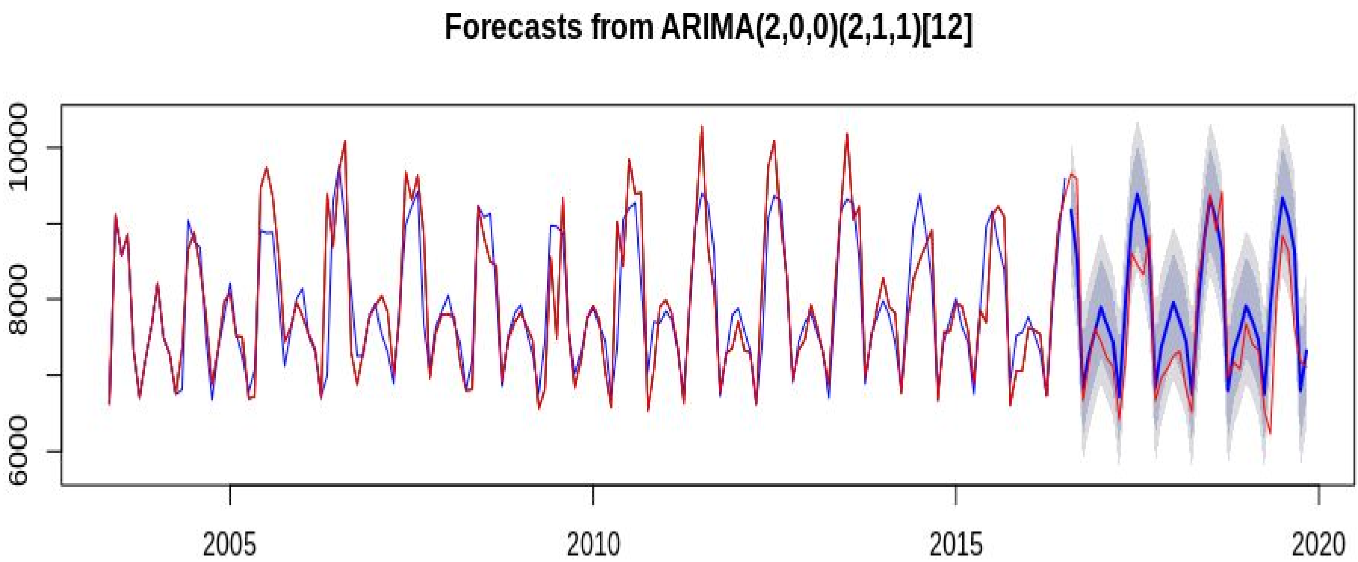
```
> HoltWinters(train) %>% forecast(h = length(test) ) %>% accuracy(test)
```

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	-43.25089	511.6597	366.8137	-0.9445038	4.500464	0.8492664	0.05299794	NA
Test set	-380.11074	1349.0604	1154.7219	-6.5725381	15.500413	2.6734726	0.58234677	1.547383

ARIMA Errors

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1	Theil's U
Training set	44.74501	459.4922	293.689	0.288841	3.54803	0.6799641	-0.01753239	NA
Test set	-243.21261	1323.4889	1148.138	-4.746073	15.27151	2.6582297	0.59231416	1.502731





Comparing the two models above we can see that Arima performs better than Holt-Winters as the errors in forecasting are smaller for the Arima model.

6. Conclusion

The aim of this project was to fit a good model to our electricity demand data, so that accurate predictions can be made for the future, which would help in better planning and allocation of resources. For this project we used the SARIMA model to fit onto our time series and the results have been pretty decent. The model **ARIMA(2,0,0)(2,1,1)$_{12}$** nicely fits the training data, and performs better on our test data than all the other candidate ARIMA models we tested out. Our Arima model also fitts the test data better than the Holt-Winters model if we take the forecast errors from the test data into account. If in the future we would like to make short term predictions only, we might consider using the hourly or daily data, but for long term predictions like we did in this project, monthly data was the best choice.

REFERENCES

1. Mishra, Madhav. “Unboxing ARIMA Models.” Medium, Towards Data Science, 11 June 2018, https://towardsdatascience.com/unboxing-arima-models-1dc09d2746f8.

2. 7.3 Holt-Winters’ Seasonal Method | Forecasting: Principles and Practice. https://Otexts.com/fpp2/. Accessed 3 Dec. 2019.

3. Jinka, Preetam. “Holt-Winters Forecasting Simplified.” *Database Monitoring Tools from VividCortex*, www.vividcortex.com/blog/holt-winters-forecasting-simplified.

4. “ARIMA Model - Complete Guide to Time Series Forecasting in Python: ML+.” *Machine Learning Plus*, 13 Oct. 2019, www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/

5. Stephanie. “ADF - Augmented Dickey Fuller Test.” *Statistics How To*, 8 Aug. 2019, www.statisticshowto.datasciencecentral.com/adf-augmented-dickey-fuller-test/.