DATA624 Project 1

Bin Lin

2018-4-9

Part A

In part A, I want you to forecast how much cash is taken out of 4 different ATM machines for May 2010. The data is given in a single file. The variable 'Cash' is provided in hundreds of dollars, other than that it is straight forward. I am being somewhat ambiguous on purpose. I am giving you data, please provide your written report on your findings, visuals, discussion and your R code all within a Word readable document, except the forecast which you will put in an Excel readable file. I must be able to cut and paste your R code and run it in R studio or other interpreter.

After I loaded the excel file, I subsetted the dataset into four subsets, because there are four different ATM machines. From the Time Series Plot, ATM4 contains extreme values greater than 10,000 and ATM3 is missing lots of data, it only start having some values in the end of the time series. In the meantime, ATM1 and ATM2 both contain reasonable amount of cash withdrawal. Hypothetically, the oscillation of the cash withdrawal amount might correspond to weekly periods.

```
#install.packages("xlsx")
#install.packages("rJava")

library(rJava)
library(xlsx)
```

```
## Loading required package: xlsxjars
```

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
library(xts)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
##
## Attaching package: 'xts'
```

```
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
library(forecast)
library(fpp)
```

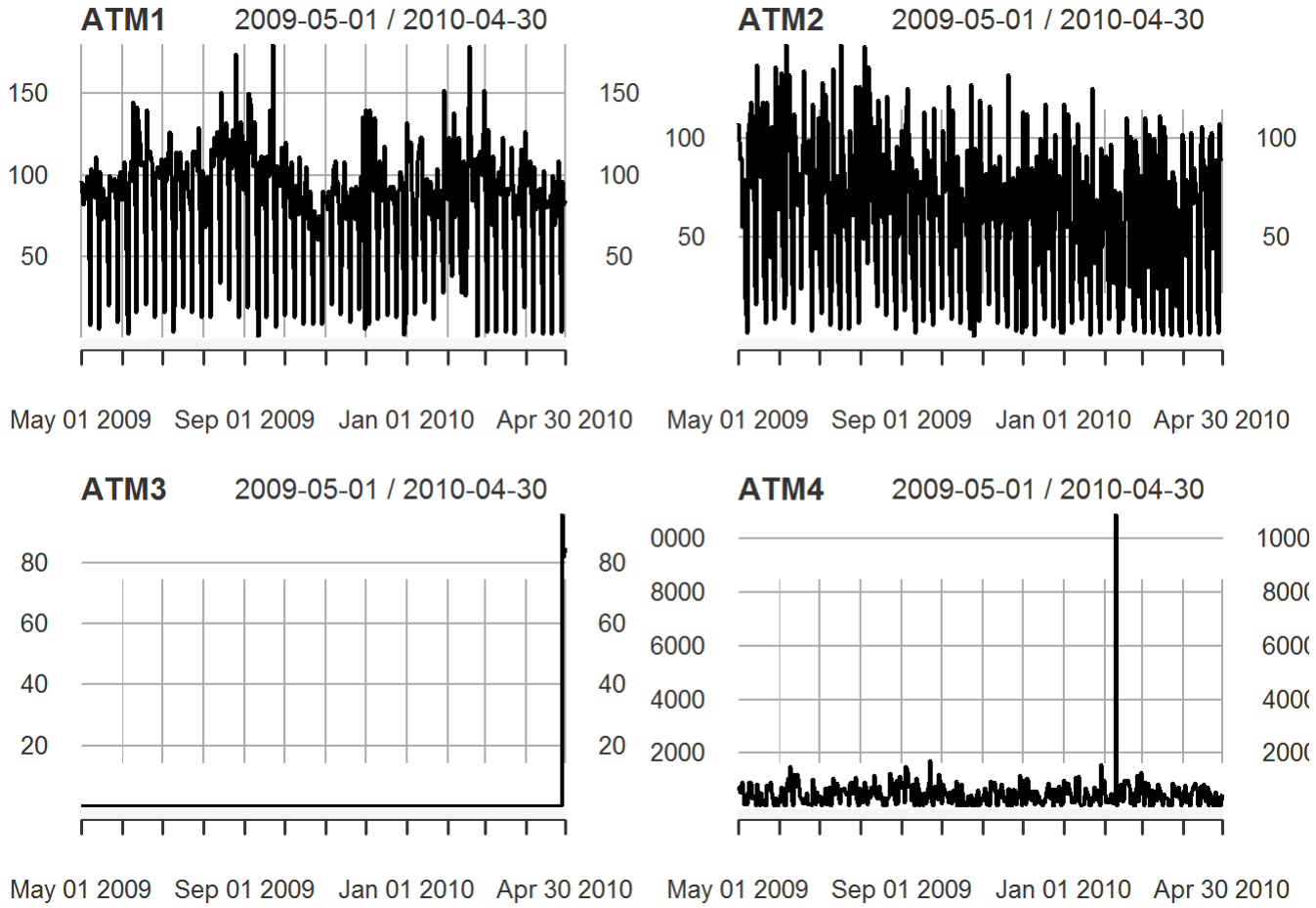
```
## Loading required package: fma
```

```
## Loading required package: expsmooth
```

```
## Loading required package: lmtest
```

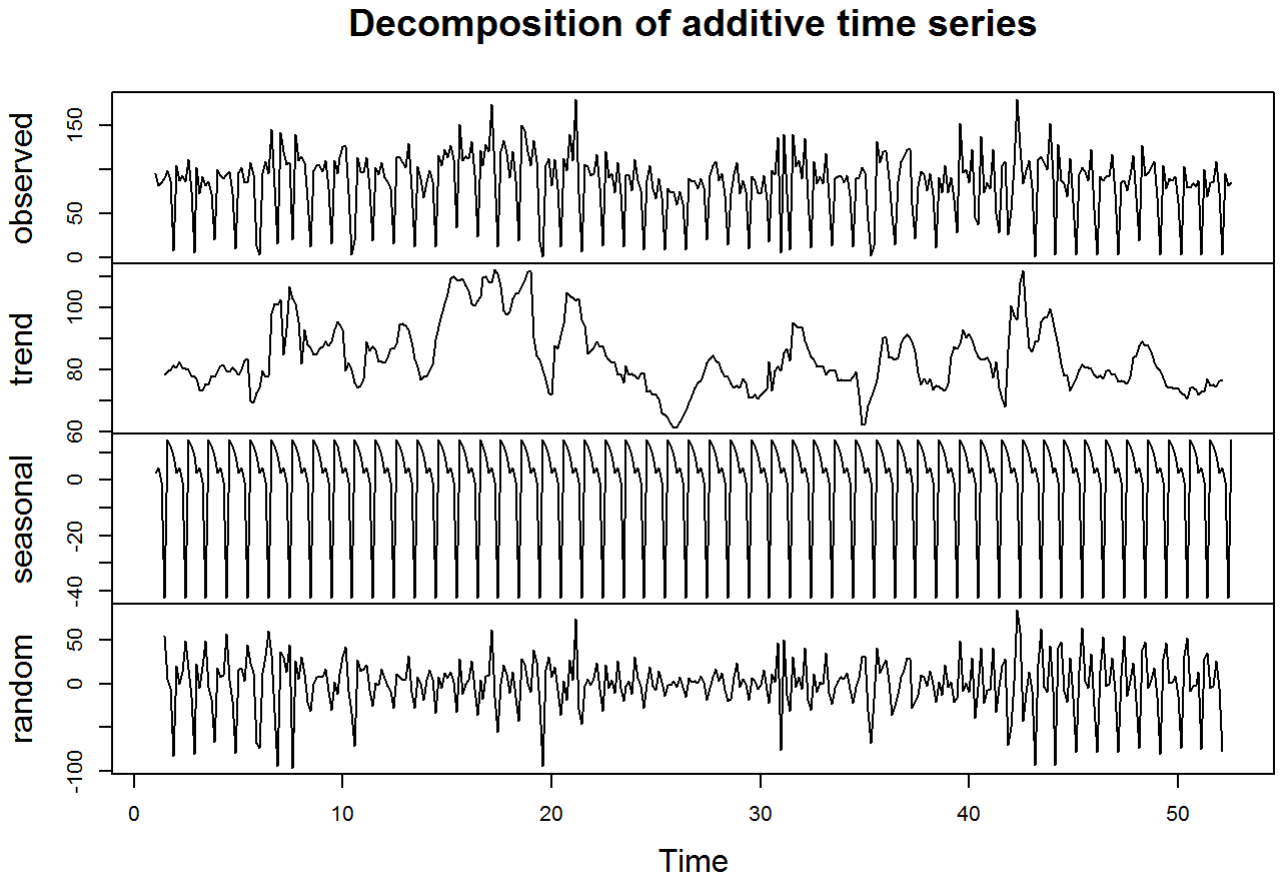
```
## Loading required package: tseries
```

```
raw_data <- read.xlsx("ATM624Data.xlsx", sheetIndex = 1, header = T)
a <- raw_data[complete.cases(raw_data),]

ATM1 <- subset(a, ATM == "ATM1")
ATM1 <- xts(ATM1$Cash, order.by = ATM1$DATE)

ATM2 <- subset(a, ATM == "ATM2")
ATM2 <- xts(ATM2$Cash, order.by = ATM2$DATE)

ATM3 <- subset(a, ATM == "ATM3")
ATM3 <- xts(ATM3$Cash, order.by = ATM3$DATE)

ATM4 <- subset(a, ATM == "ATM4")
ATM4 <- xts(ATM4$Cash, order.by = ATM4$DATE)

par(mfrow = c(2, 2))
plot(ATM1, main = "ATM1")
plot(ATM2, main = "ATM2")
plot(ATM3, main = "ATM3")
plot(ATM4, main = "ATM4")
```

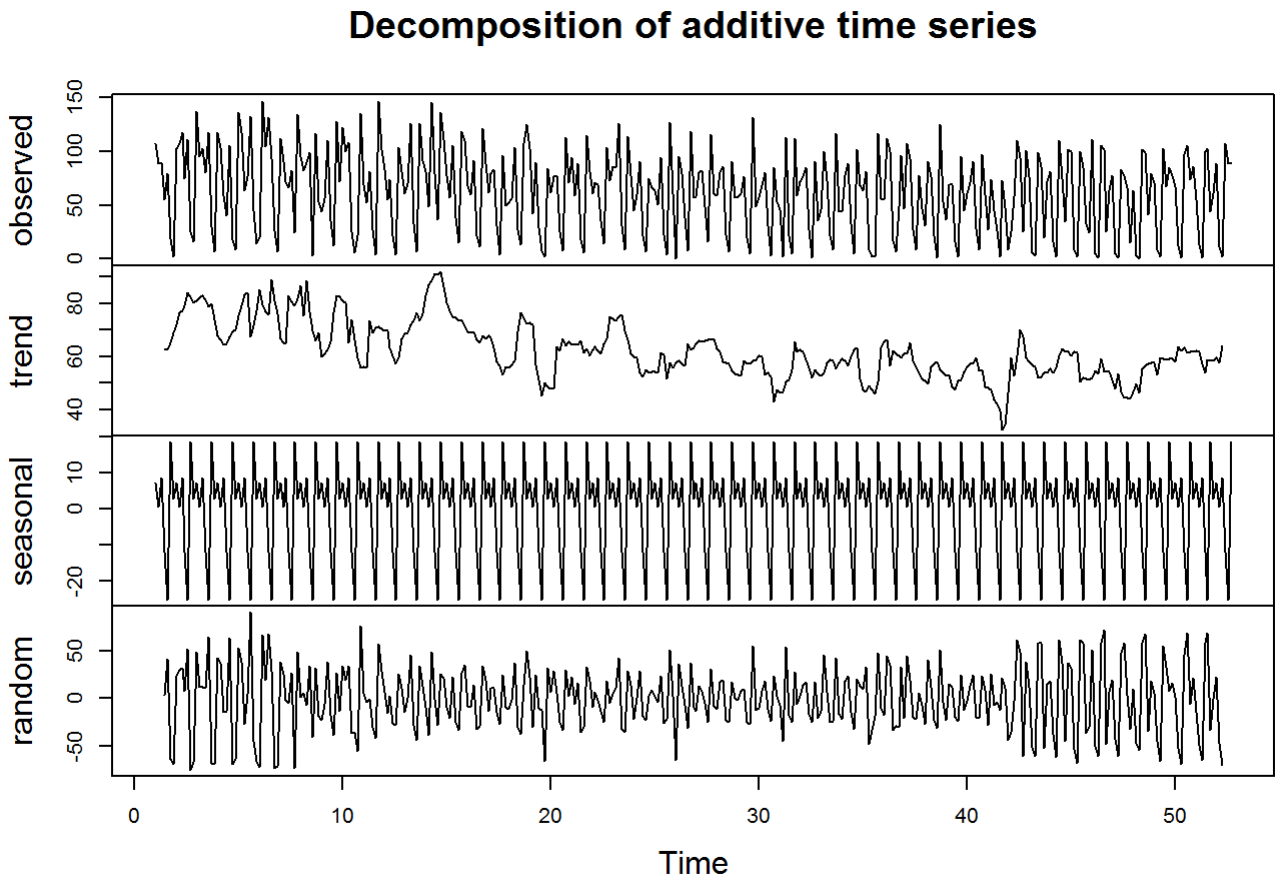


I am using classical decomposition to investigate the trend, seasonal, and random components. I will also set the time frequency into 7 to search for any weekly seasonality. The resulting graph tells me that all four ATMs have strong seasonal components.

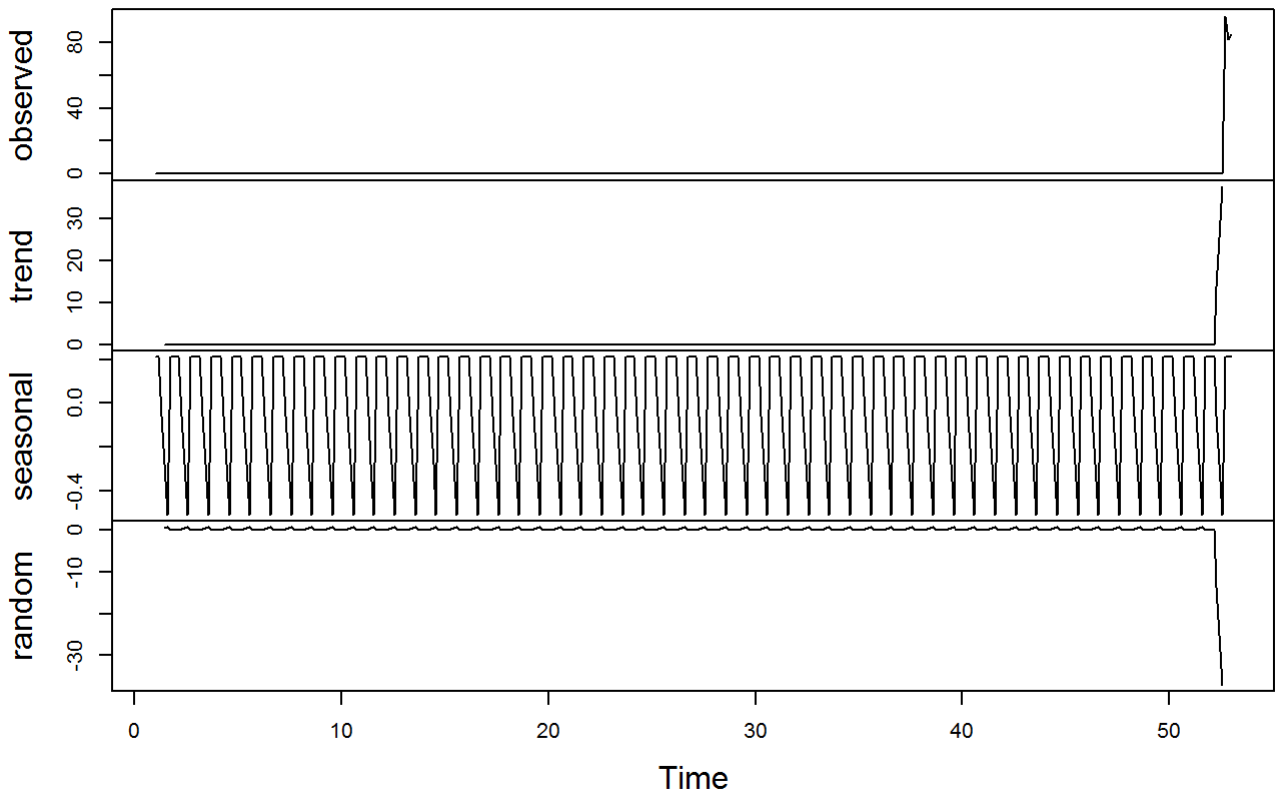
```
atm1 <- ts(ATM1, frequency =  7)
atm2 <- ts(ATM2, frequency =  7)
atm3 <- ts(ATM3, frequency =  7)
atm4 <- ts(ATM4, frequency =  7)

plot(decompose(atm1))
```



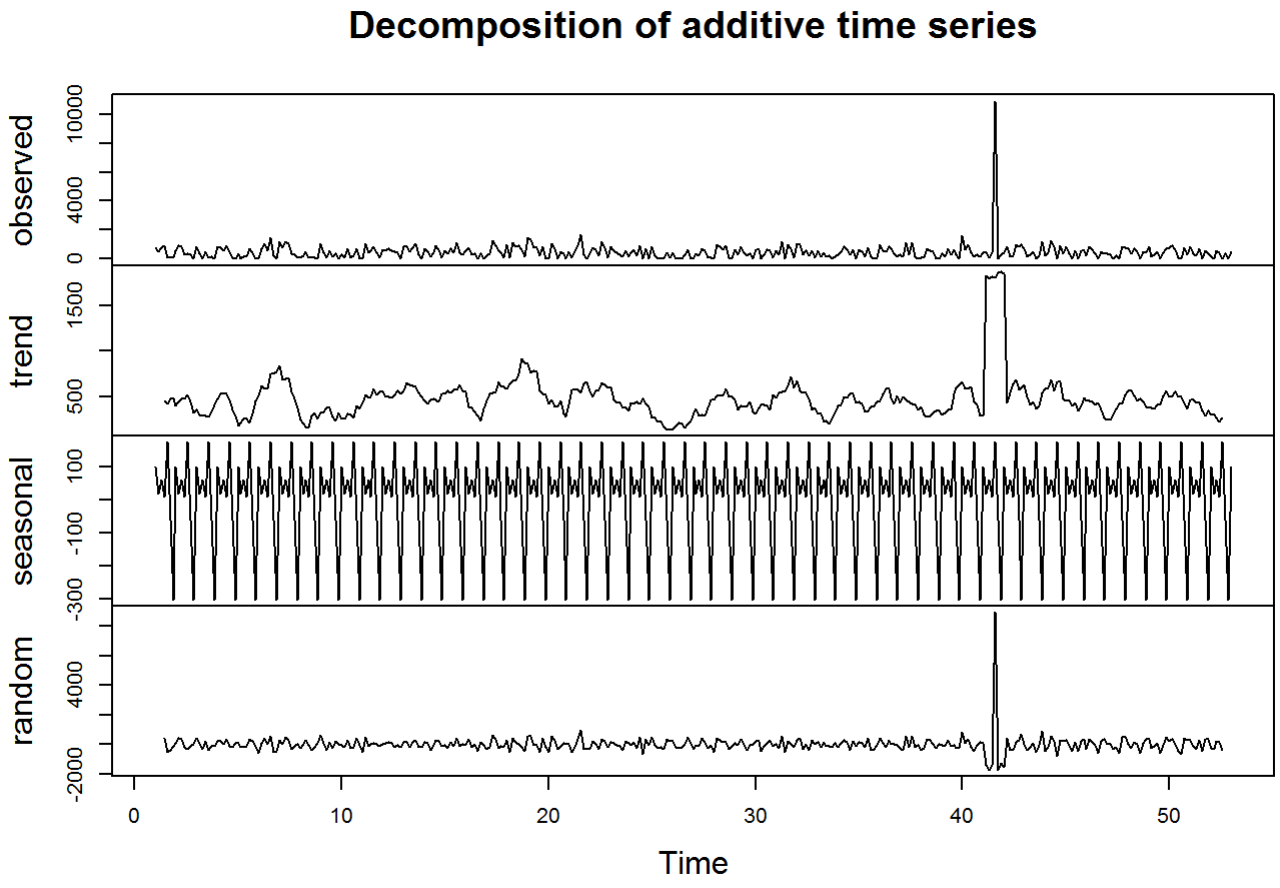

```
plot(decompose(atm2))
```




```
plot(decompose(atm3))
```

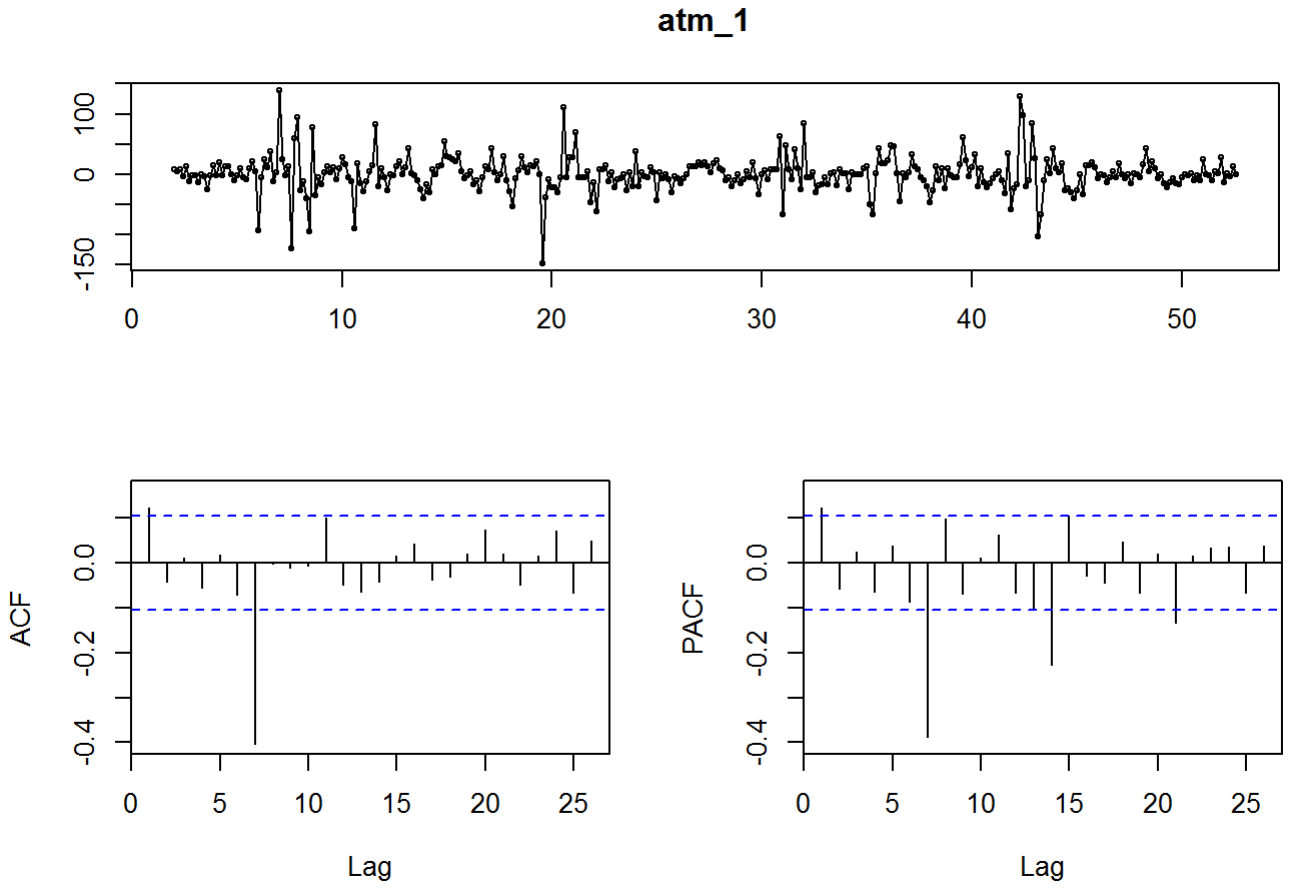



```
plot(decompose(atm4))
```

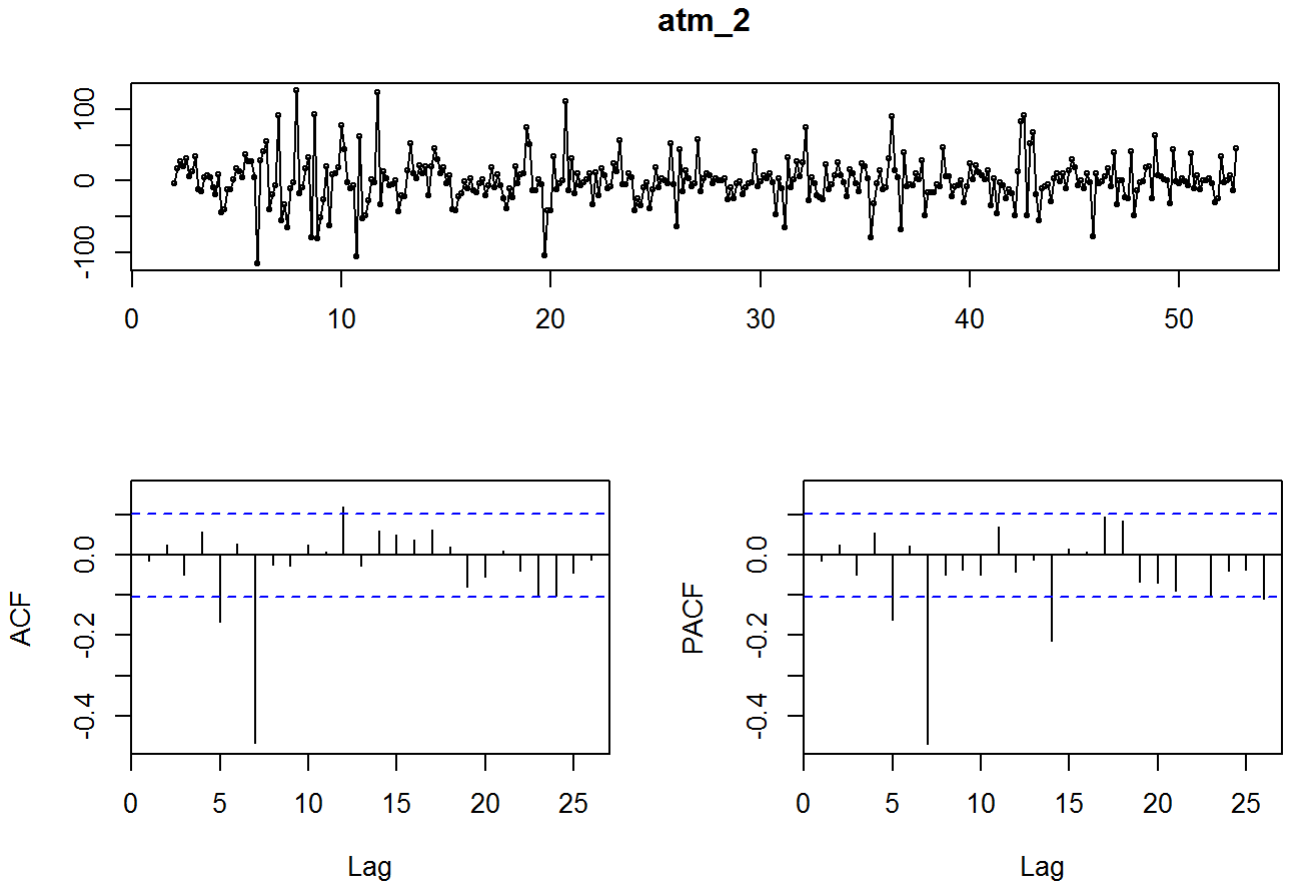


Any time series with trends, or with seasonality, are not stationary. Therefore, I will apply seasonal differencing with lag of 7 first try to make the data stationary. The first difference graph looks stationary. However, both the ACF and PACF figure has some spikes that is going out of the bound. I will perform Unit Root test and KPSS test to check if more differencing is necessary.

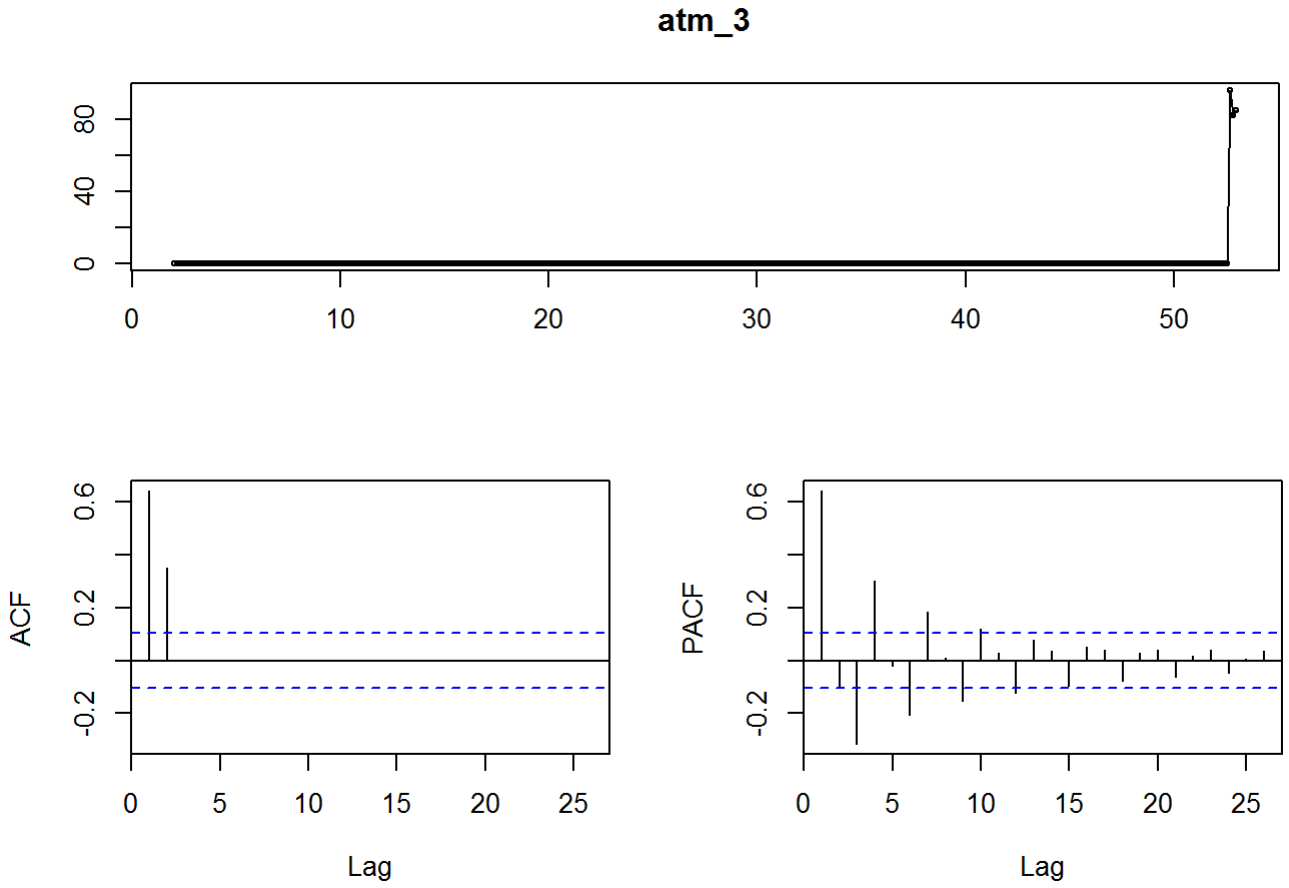
```
atm_1 <- diff(atm1, 7)
atm_2 <- diff(atm2, 7)
atm_3 <- diff(atm3, 7)
atm_4 <- diff(atm4, 7)

tsdisplay(atm_1)
```



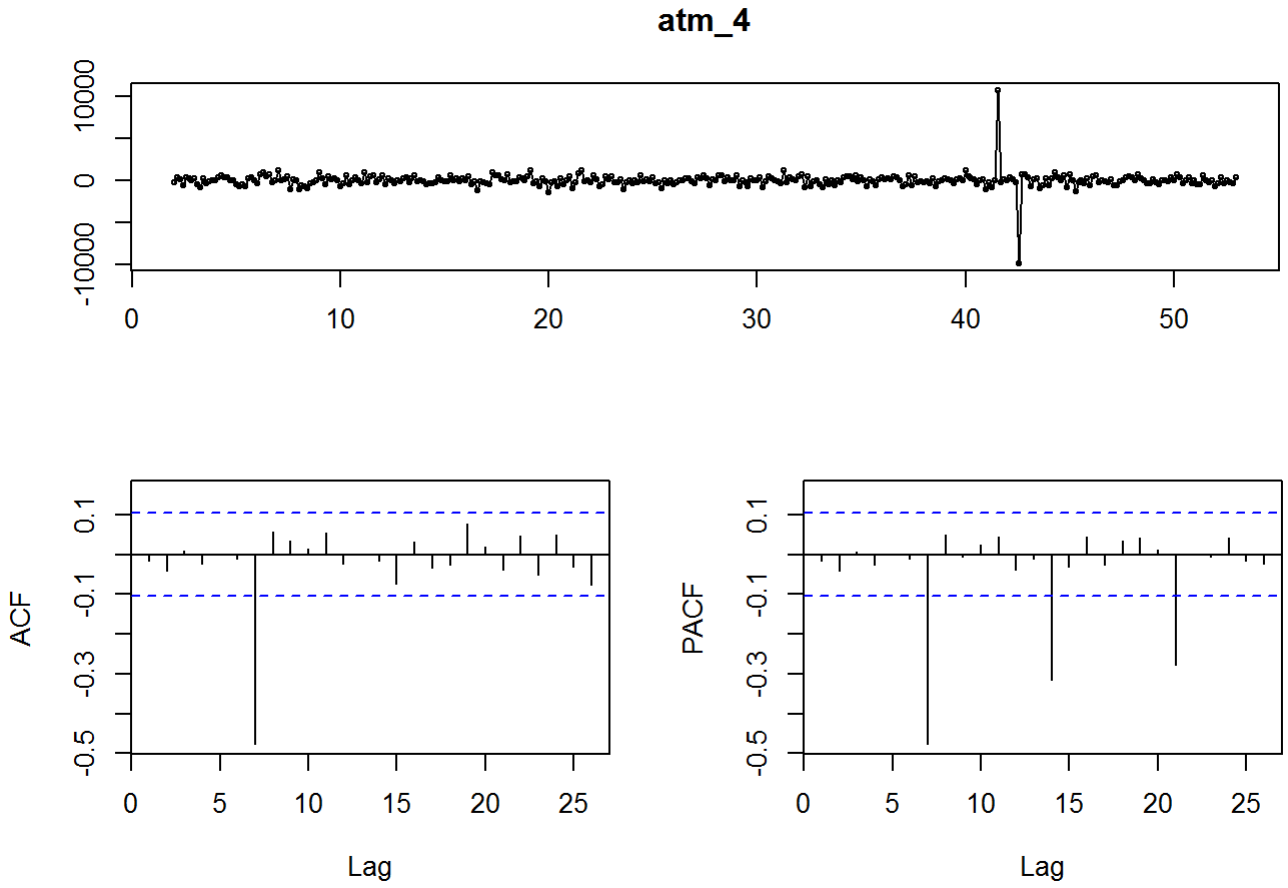

```
tsdisplay(atm_2)
```




```
tsdisplay(atm_3)
```




```
tsdisplay(atm_4)
```



If p-value from Unit Root Test is less than 5% or p-value from KPSS test is greater than 5%, it indicates that the data is stationary. From the following results, only ATM3 does not meet the requirements, which is probably due to its missing values.

```
adf.test(atm_1)
```

```
## Warning in adf.test(atm_1): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  atm_1
## Dickey-Fuller = -8.7058, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(atm_1)
```

```
## Warning in kpss.test(atm_1): p-value greater than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  atm_1
## KPSS Level = 0.021606, Truncation lag parameter = 4, p-value = 0.1
```

```
adf.test(atm_2)
```

```
## Warning in adf.test(atm_2): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  atm_2
## Dickey-Fuller = -11.123, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(atm_2)
```

```
## Warning in kpss.test(atm_2): p-value greater than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  atm_2
## KPSS Level = 0.016516, Truncation lag parameter = 4, p-value = 0.1
```

```
adf.test(atm_3)
```

```
## Warning in adf.test(atm_3): p-value greater than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  atm_3
## Dickey-Fuller = -0.059487, Lag order = 7, p-value = 0.99
## alternative hypothesis: stationary
```

```
kpss.test(atm_3)
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  atm_3
## KPSS Level = 0.40334, Truncation lag parameter = 4, p-value =
## 0.07571
```

```
adf.test(atm_4)
```

```
## Warning in adf.test(atm_4): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  atm_4
## Dickey-Fuller = -9.6378, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(atm_4)
```

```
## Warning in kpss.test(atm_4): p-value greater than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  atm_4
## KPSS Level = 0.013862, Truncation lag parameter = 4, p-value = 0.1
```

I am going to use Box-Cox transformation to normalize the data for model building. This method can help reduce the effects of increasing variances. I will mainly use function auto.arima to search for the best model that will minimize the AIC values for ATM 1, 2, and 4.

From the following results, we can tell the model the generate smallest AIC for ATM1 is ARIMA(1,0,2)(1,0,1)[7]. For ATM2 is ARIMA(3,1,0)(1,0,1)[7]. And for ATM4 is ARIMA(0,0,0)(2,0,0)[7].

```
lambda1 <- BoxCox.lambda(atm1)
fit1 <- auto.arima(atm1, seasonal = TRUE, stepwise = FALSE, lambda = lambda1)
summary(fit1)
```

```
## Series: atm1
## ARIMA(1,0,2)(1,0,1)[7] with non-zero mean
## Box Cox transformation: lambda= 0.1714881
##
## Coefficients:
##          ar1      ma1      ma2    sar1     sma1    mean
##       0.8120  -0.7443  -0.1665  0.9289  -0.4895  6.2221
## s.e.  0.1027   0.1147   0.0613  0.0246   0.0616  0.1581
##
## sigma^2 estimated as 1.11:  log likelihood=-532.92
## AIC=1079.83   AICc=1080.15   BIC=1107.07
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 5.444557 27.43557 18.68391 -81.39725 104.1882 0.9927835
##                     ACF1
## Training set 0.05428586
```

```
lambda2 <- BoxCox.lambda(atm2)
fit2 <- auto.arima(atm2, seasonal = TRUE, stepwise = FALSE, lambda = lambda2)
summary(fit2)
```

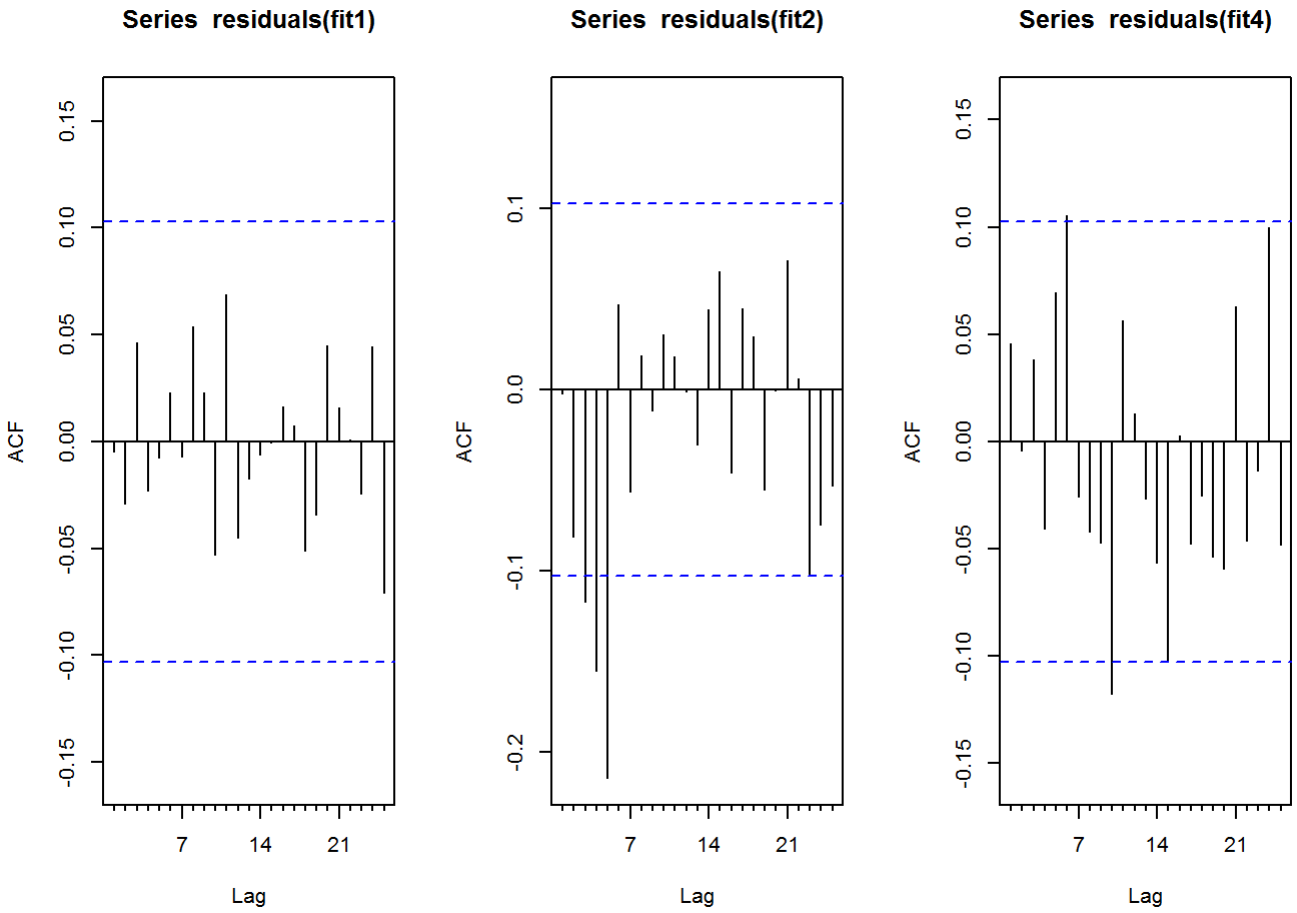
```
## Series: atm2
## ARIMA(3,1,0)(1,0,1)[7]
## Box Cox transformation: lambda= 0.6846687
##
## Coefficients:
##           ar1      ar2      ar3    sar1     sma1
##       -0.7375  -0.5795  -0.3150  0.9282  -0.4828
## s.e.   0.0501   0.0560   0.0499  0.0236   0.0526
##
## sigma^2 estimated as 73.21:  log likelihood=-1292.38
## AIC=2596.75   AICc=2596.99   BIC=2620.1
##
## Training set error measures:
##                     ME     RMSE      MAE  MPE MAPE      MASE        ACF1
## Training set 1.724117 30.10231 20.95502 -Inf  Inf 0.9751615 -0.02069533
```

```
lambda4 <- BoxCox.lambda(atm4)
fit4 <- auto.arima(atm4, seasonal = TRUE, stepwise = FALSE, lambda = lambda4)
summary(fit4)
```

```
## Series: atm4
## ARIMA(0,0,0)(2,0,0)[7] with non-zero mean
## Box Cox transformation: lambda= -0.07372558
##
## Coefficients:
##         sar1    sar2    mean
##       0.2487  0.1947  4.4864
## s.e.  0.0521  0.0525  0.0841
##
## sigma^2 estimated as 0.8418:  log likelihood=-485.59
## AIC=979.18   AICc=979.29   BIC=994.78
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE    MAPE      MASE
## Training set 209.7822 678.9958 317.1943 -239.7643 304.041 0.7892482
##                      ACF1
## Training set -0.005100994
```

The ACF plot of the residuals shows all correlations within the threshold limits indicating that the residuals are behaving like white noise. A portmanteau test returns a p-values for ATM1 and ATM4 are greater than the threshold 0.05. Therefore, the result indicates the residuals are white noise, so that the model is satisfactory. For model 2, even the p-value from Box-Ljung test is smaller than 5%, but majority of the autocorrelation fall within the bounds. Therefore, this model is also considered appropriate for forecast.

```
par(mfrow = c(1, 3))
Acf(residuals(fit1))
Acf(residuals(fit2))
Acf(residuals(fit4))
```




```
Box.test(residuals(fit1), lag = 7, fitdf = 2, type = "Ljung")
```

```
##
##  Box-Ljung test
##
## data:  residuals(fit1)
## X-squared = 1.5365, df = 5, p-value = 0.9088
```

```
Box.test(residuals(fit2), lag = 7, fitdf = 5, type = "Ljung")
```

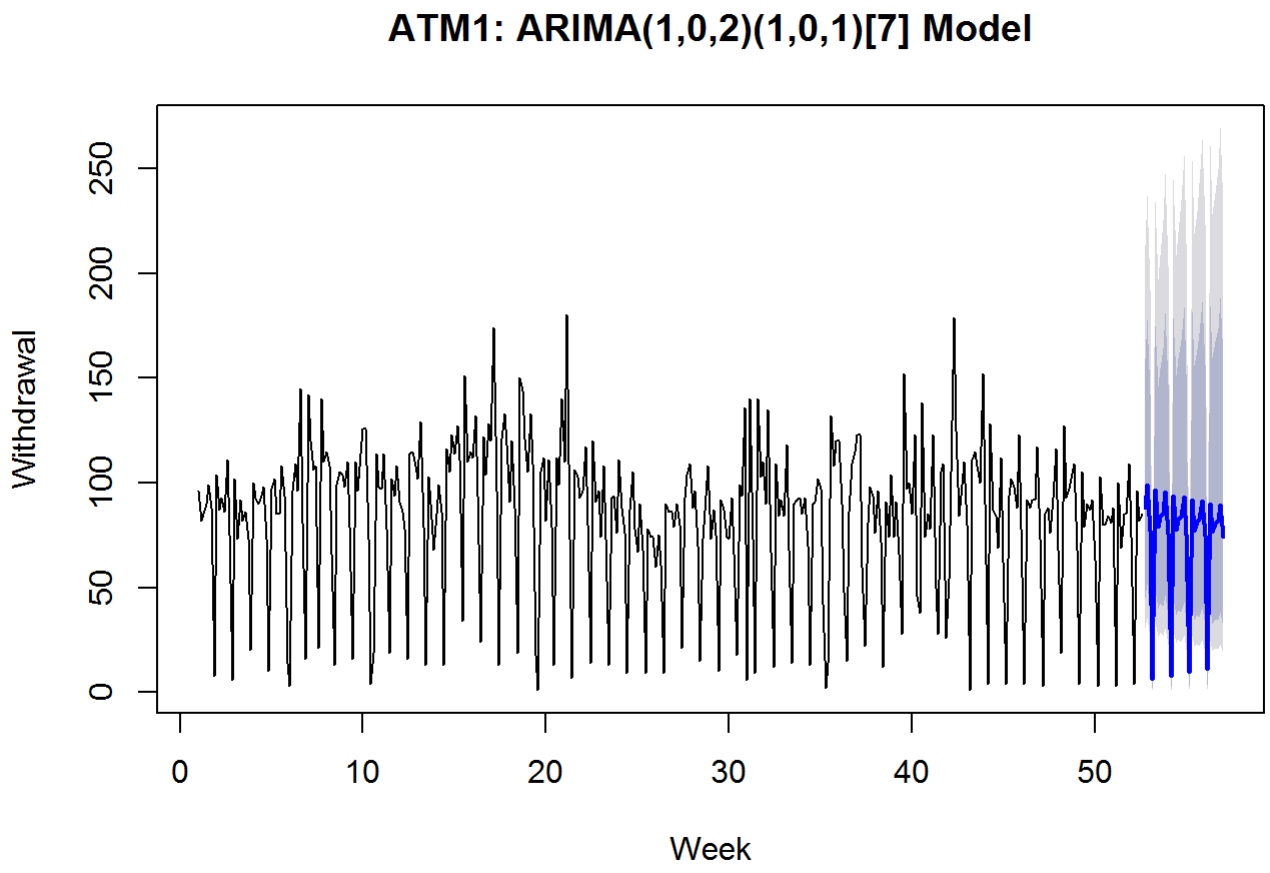
```
##
##  Box-Ljung test
##
## data:  residuals(fit2)
## X-squared = 35.426, df = 2, p-value = 2.029e-08
```

```
Box.test(residuals(fit4), lag = 7, fitdf = 1, type = "Ljung")
```

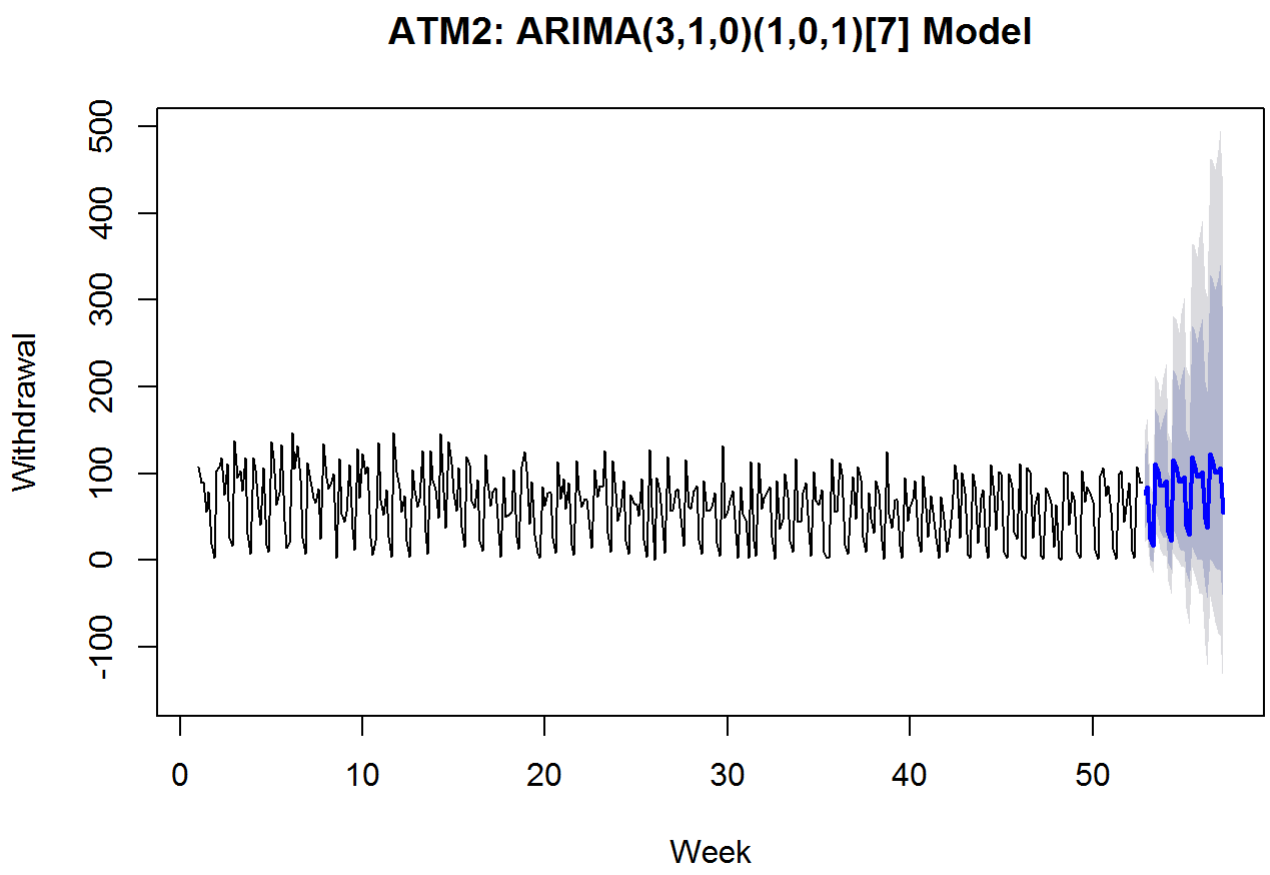
```
##
##  Box-Ljung test
##
## data:  residuals(fit4)
## X-squared = 8.1244, df = 6, p-value = 0.2291
```

The month of May, 2010 has 31 days. That is why I am predicting 31 values. For ATM3, I feel the best way to forecast is using the naive method. Since it only has three values while all other values are zero, therefore, using the mean method will be inappropriate.

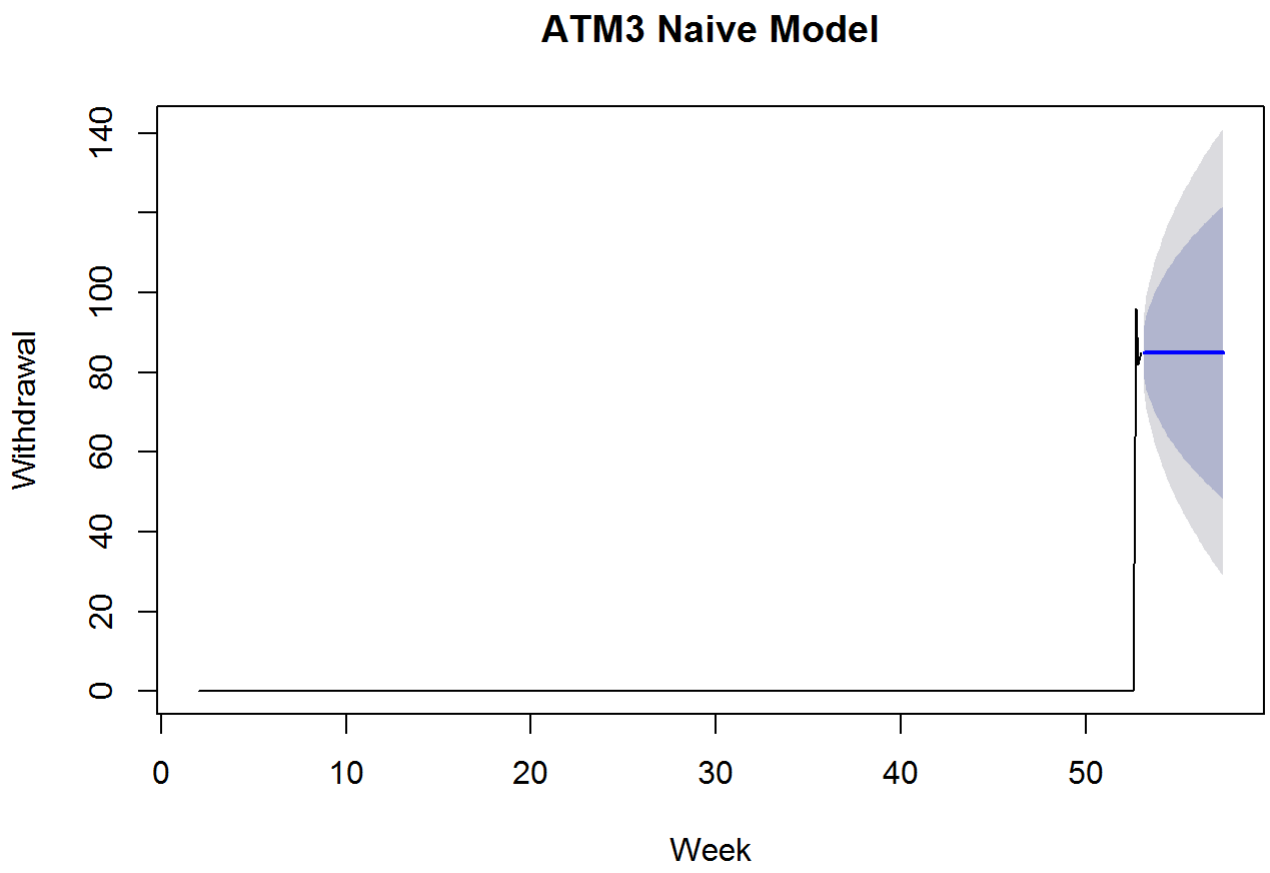
```
forecast1 <- forecast(fit1, h = 31)
forecast2 <- forecast(fit2, h = 31)
forecast3 <- naive(atm_3, h = 31)
forecast4 <- forecast(fit4, h = 31)

plot(forecast1, main = "ATM1: ARIMA(1,0,2)(1,0,1)[7] Model", xlab = "Week", ylab = "Withdrawal")
```



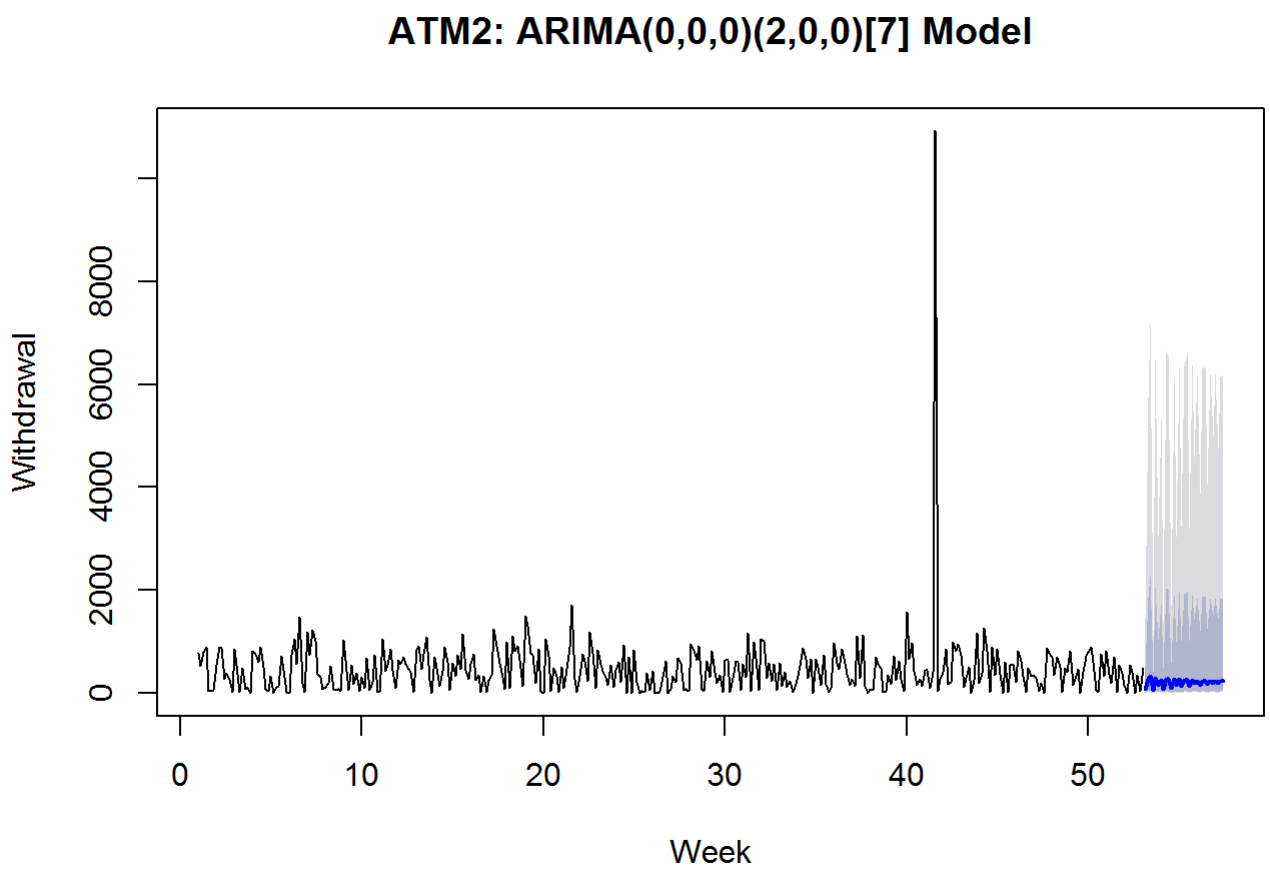

```
plot(forecast2, main = "ATM2: ARIMA(3,1,0)(1,0,1)[7] Model", xlab = "Week", ylab = "Withdrawal")
```




```
plot(forecast3, main = "ATM3 Naive Model", xlab = "Week", ylab = "Withdrawal")
```




```
plot(forecast4, main = "ATM2: ARIMA(0,0,0)(2,0,0)[7] Model", xlab = "Week", ylab = "Withdrawal")
```



Create a data frame with final results and save it into an excel file. Using append = TRUE can create a worksheet, and allow me to add more worksheets to the same file in the end.

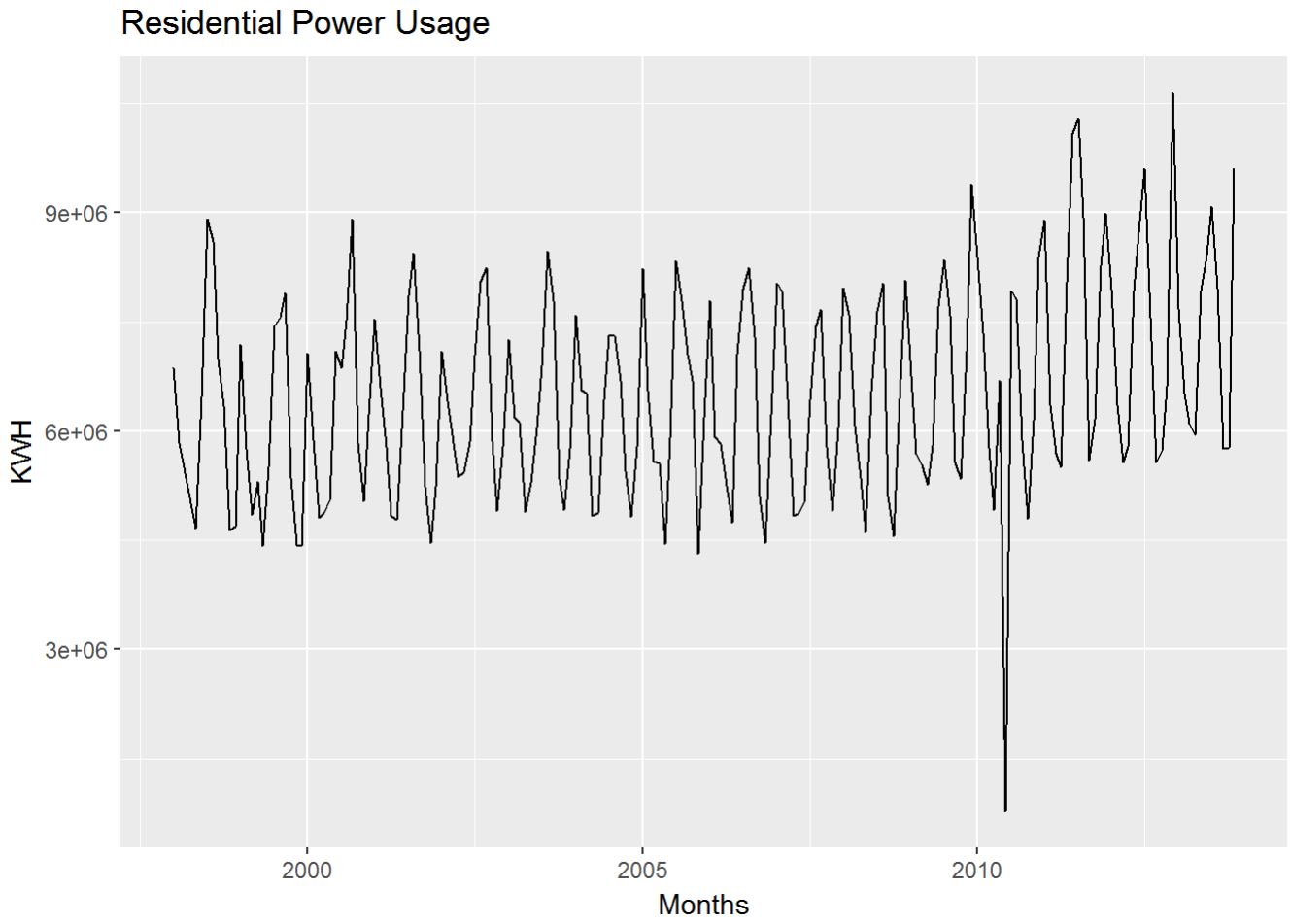
```
result1 <- data.frame(ATM1 = forecast1$mean, ATM2 = forecast2$mean, ATM3 = forecast3$mean, ATM4
= forecast4$mean)

write.xlsx(result1, file = "Project1_Output.xlsx", sheetName="ATM_Forecast(May2010)", append=TRU
E)
```

Part B - Forecasting Power

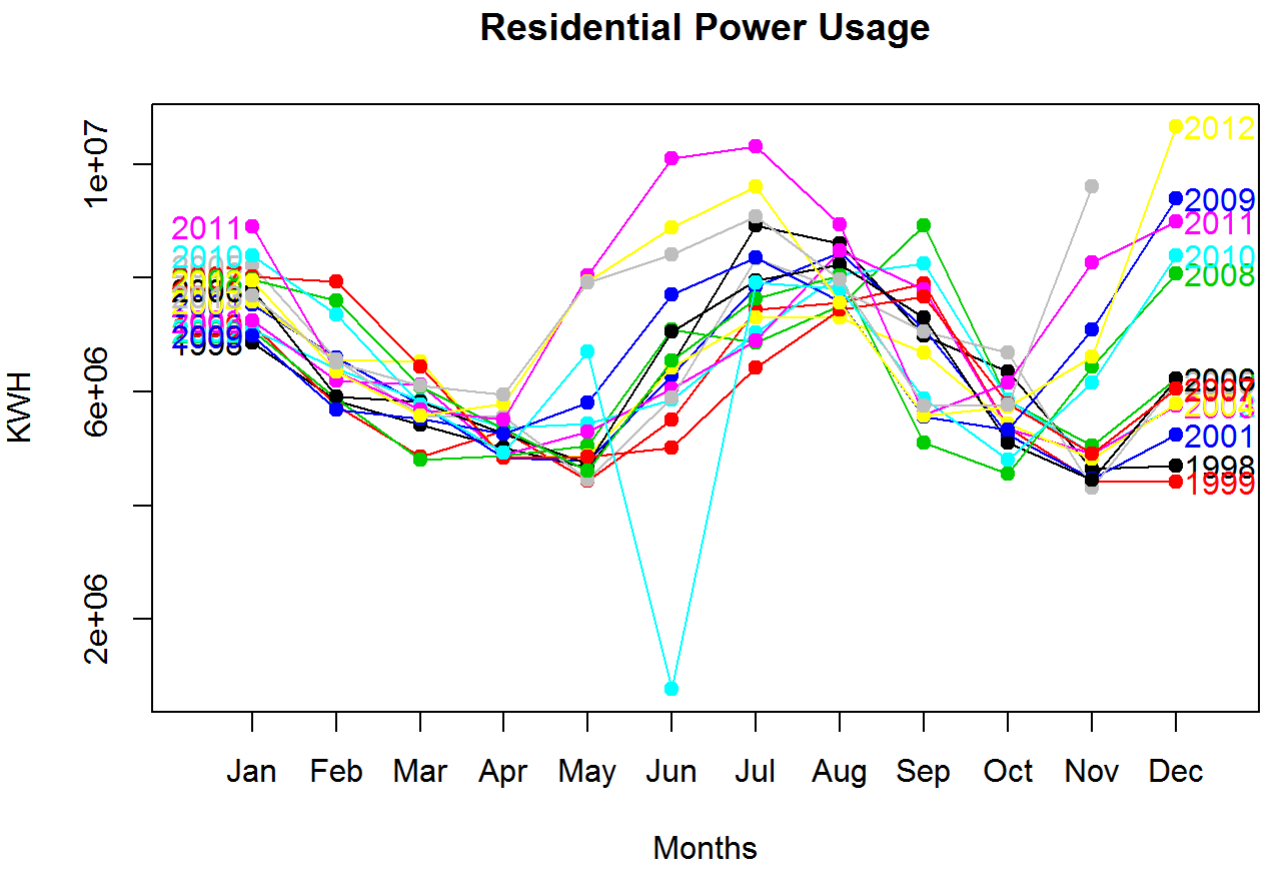
Part B consists of a simple dataset of residential power usage for January 1998 until December 2013. Your assignment is to model these data and a monthly forecast for 2014. The data is given in a single file. The variable 'KWH' is power consumption in Kilowatt hours, the rest is straight forward. Add this to your existing files above.

The Time Plot shows the observations oscillate over the years. There are peaks and troughs periodically, which is clear sign for seasonality. Both the Seasonal Plot and Monthly plot shows that during each year, the peak usually happen in the summer and winter. In addition, we also noticed that the variance keep increasing over the years, which justify the usage of Box-Cox transformation to stabilize the variances. Right now, there is no clear explanation for the dip of the residential power usage in the begining of 2010.

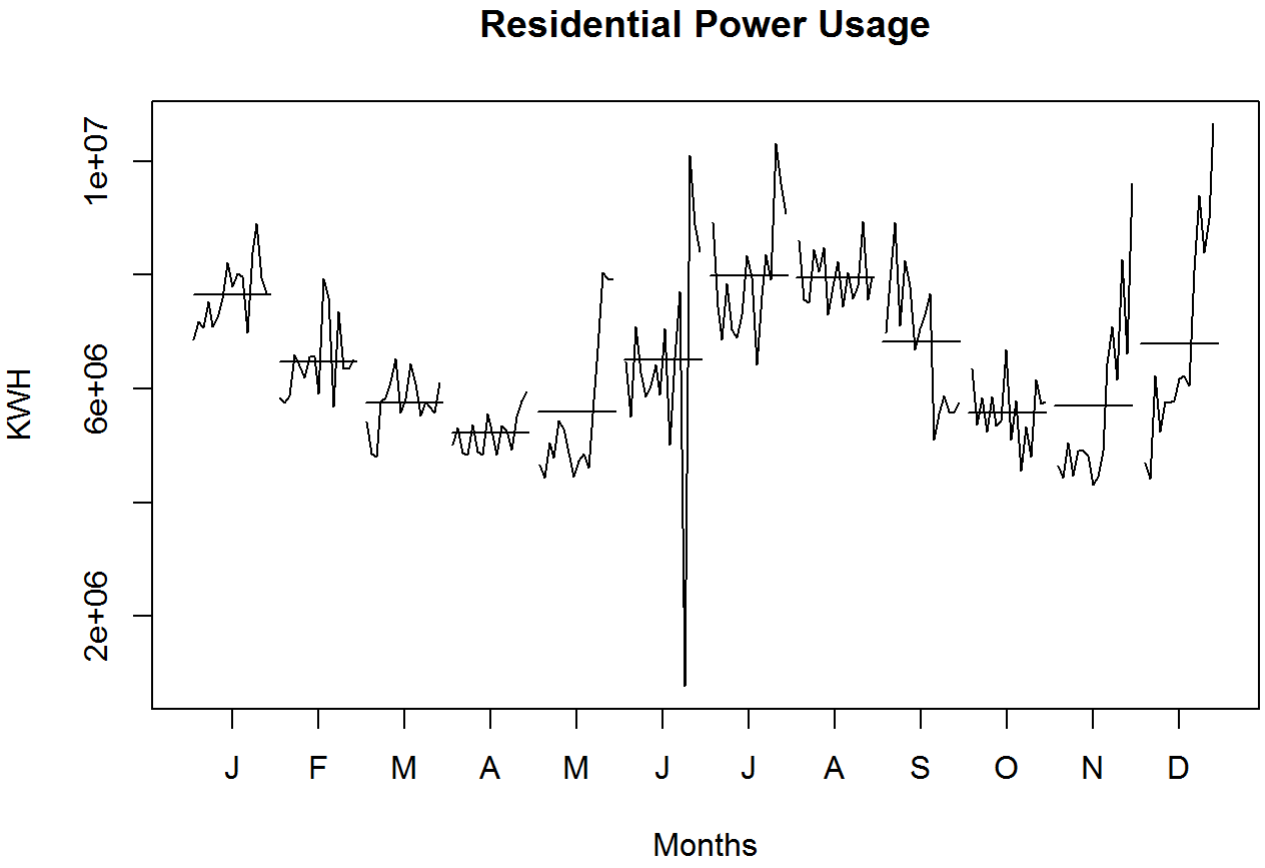
```
raw_data2 <- read.xlsx("ResidentialCustomerForecastLoad-624.xlsx", sheetIndex = 1, header = T)
b <- raw_data2[complete.cases(raw_data2),]
res_power <- ts(b[, "KWH"], start = c(1998, 1), frequency = 12)


autoplot(res_power, main = "Residential Power Usage", xlab = "Months", ylab = "KWH")
```




```
seasonplot(res_power, main = "Residential Power Usage", xlab = "Months", ylab = "KWH", year.labe
ls = TRUE, year.labels.left = TRUE, col = 1:20, pch = 19)
```

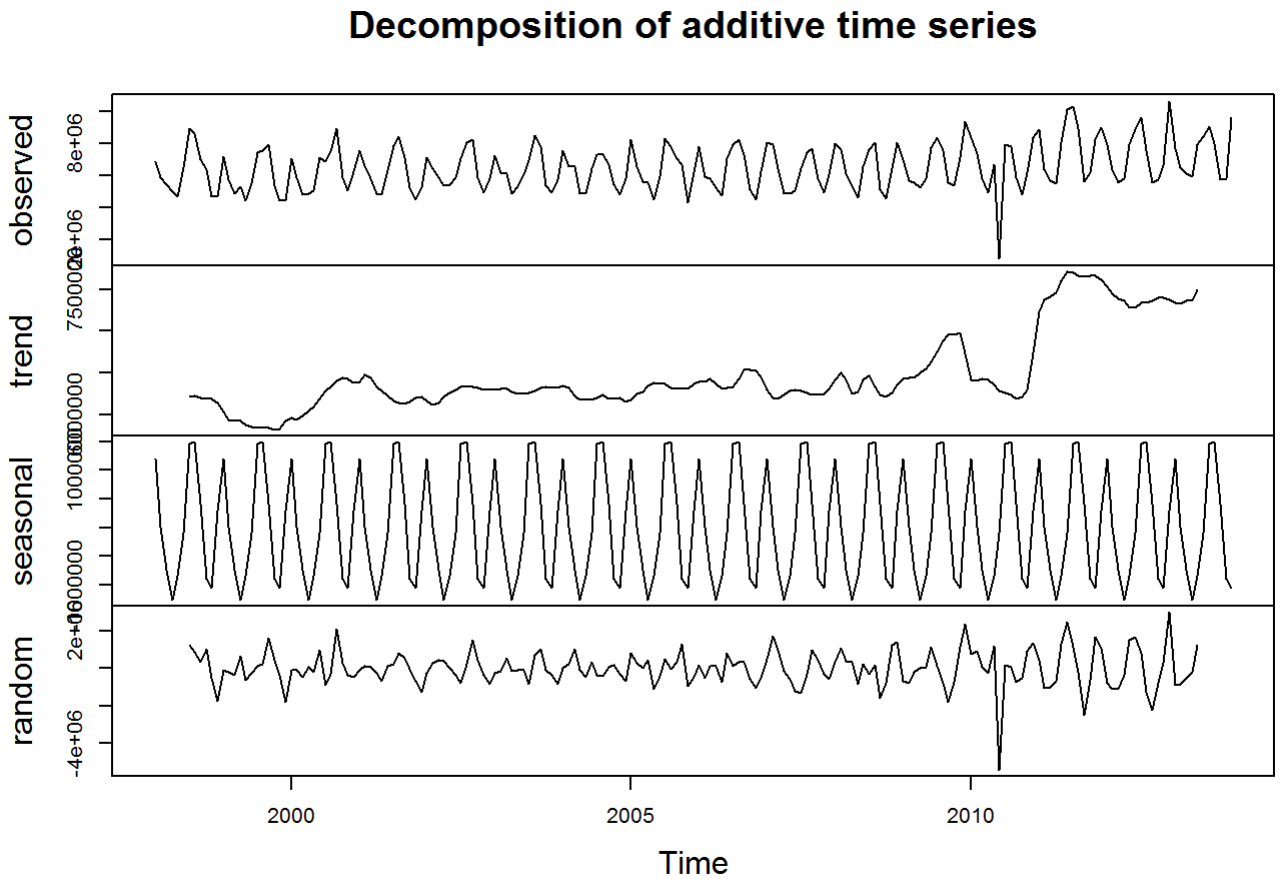



```
monthplot(res_power, main = "Residential Power Usage", xlab = "Months", ylab = "KWH")
```



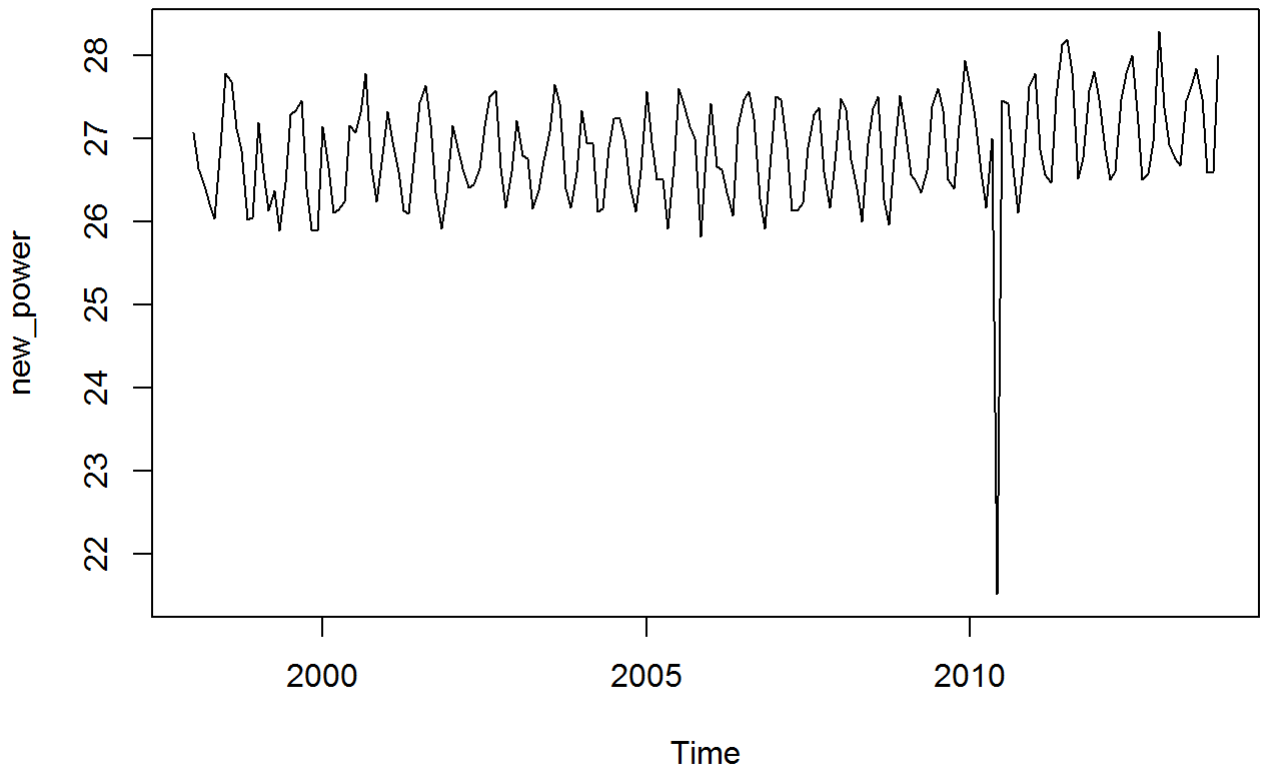
The classical decomposition graph tells us more information, including trend, seasonal, and random components. Not only there is seasonality components involved. In the year of 2010, there is upward trend on the residential power usage.

```
plot(decompose(res_power))
```

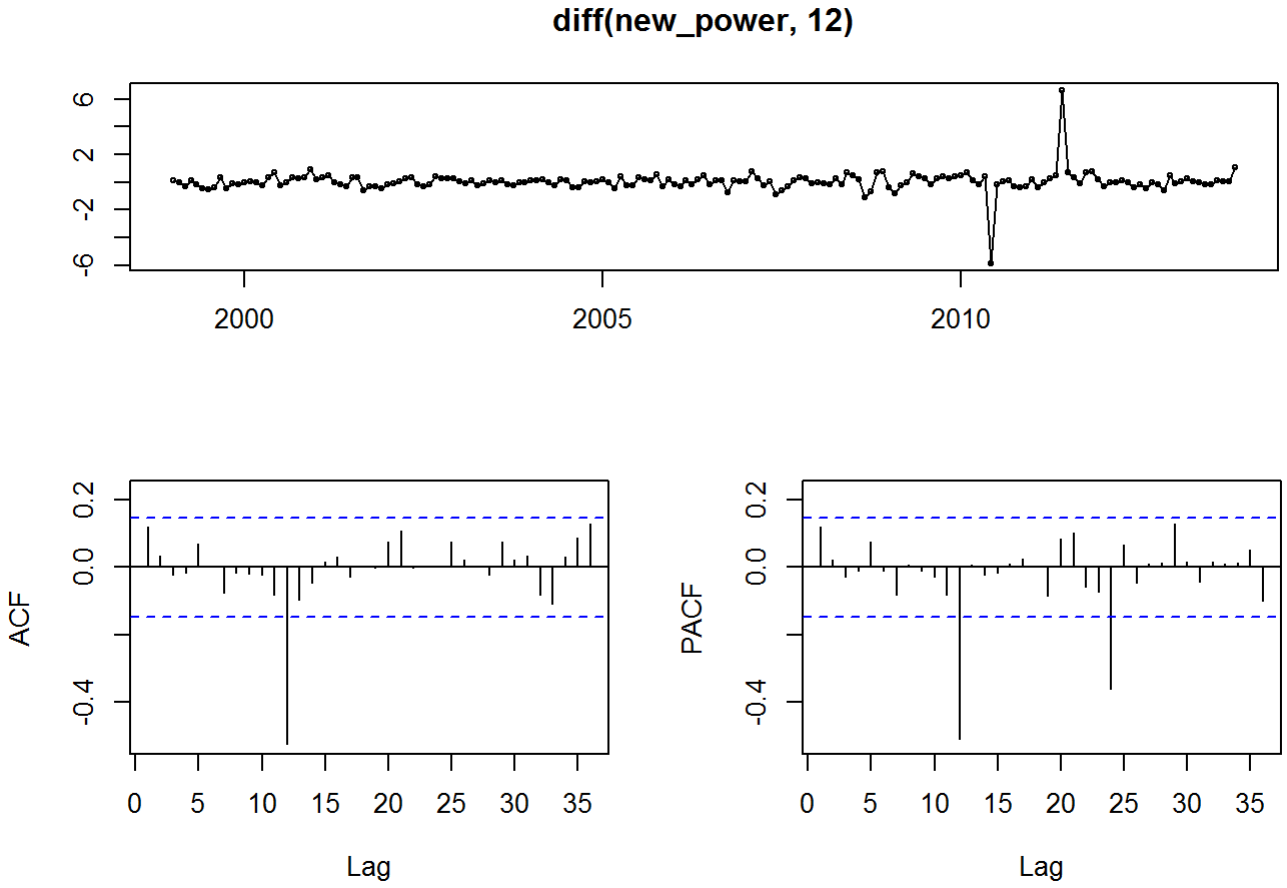


After Box-Cox transformation and seasonal differencing, the ACF and PACF figures show very few spikes that are going out of bound (less than 5%). The results from Unit Root Test and KPSS test are also very promising since the p-value is less than 5% for the first test and greater than 5% for the second test. All of these prove that the data are stationary and behave like white noise. Or in other words, there is no autocorrelation or partial correlation exist in the transformed data.

```
lambda <- BoxCox.lambda(res_power)
new_power <- BoxCox(res_power, lambda = lambda)


plot(new_power)
```




```
tsdisplay(diff(new_power, 12))
```




```
adf.test(diff(new_power, 12))
```

```
## Warning in adf.test(diff(new_power, 12)): p-value smaller than printed p-
## value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(new_power, 12)
## Dickey-Fuller = -4.9808, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(diff(new_power, 12))
```

```
## Warning in kpss.test(diff(new_power, 12)): p-value greater than printed p-
## value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  diff(new_power, 12)
## KPSS Level = 0.068957, Truncation lag parameter = 3, p-value = 0.1
```

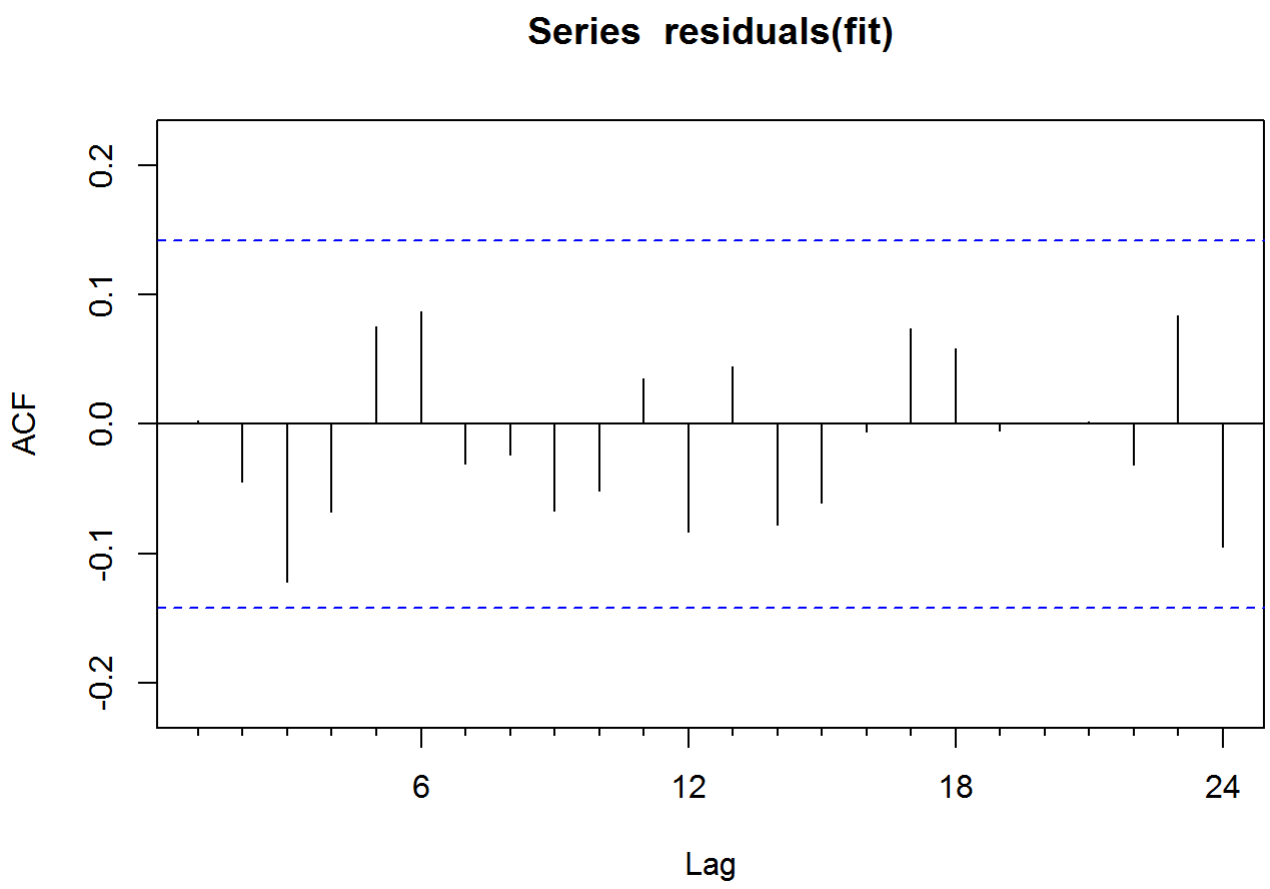
The method of auto.arima generate ARIMA(1,1,1)(2,0,0)[12] for modeling of residential power usage. This model technically is the one with the lowest AIC value.

```
fit <- auto.arima(res_power, seasonal = TRUE, stepwise = FALSE, lambda = lambda)
summary(fit)
```

```
## Series: res_power
## ARIMA(1,1,1)(2,0,0)[12]
## Box Cox transformation: lambda= 0.06361748
##
## Coefficients:
##          ar1      ma1    sar1    sar2
##       0.1523  -0.9833  0.2802  0.2827
## s.e.  0.0737   0.0149  0.0683  0.0693
##
## sigma^2 estimated as 0.3558:  log likelihood=-172.41
## AIC=354.83   AICc=355.15   BIC=371.06
##
## Training set error measures:
##                     ME    RMSE      MAE      MPE     MAPE     MASE
## Training set 221729.5 1140576 768086.4 -2.74024 15.03434 1.032238
##                    ACF1
## Training set 0.1410837
```

To investigate if ARIMA(1,1,1)(2,0,0)[12] model is appropriate for forecast, we have to examine the autocorrelation graph of the residuals and the portmanteau test. If the residual appear to be white noise and p-value is greater than the threshold 5%, then the model is satisfactory. The results meet both criteria.

```
Acf(residuals(fit))
```

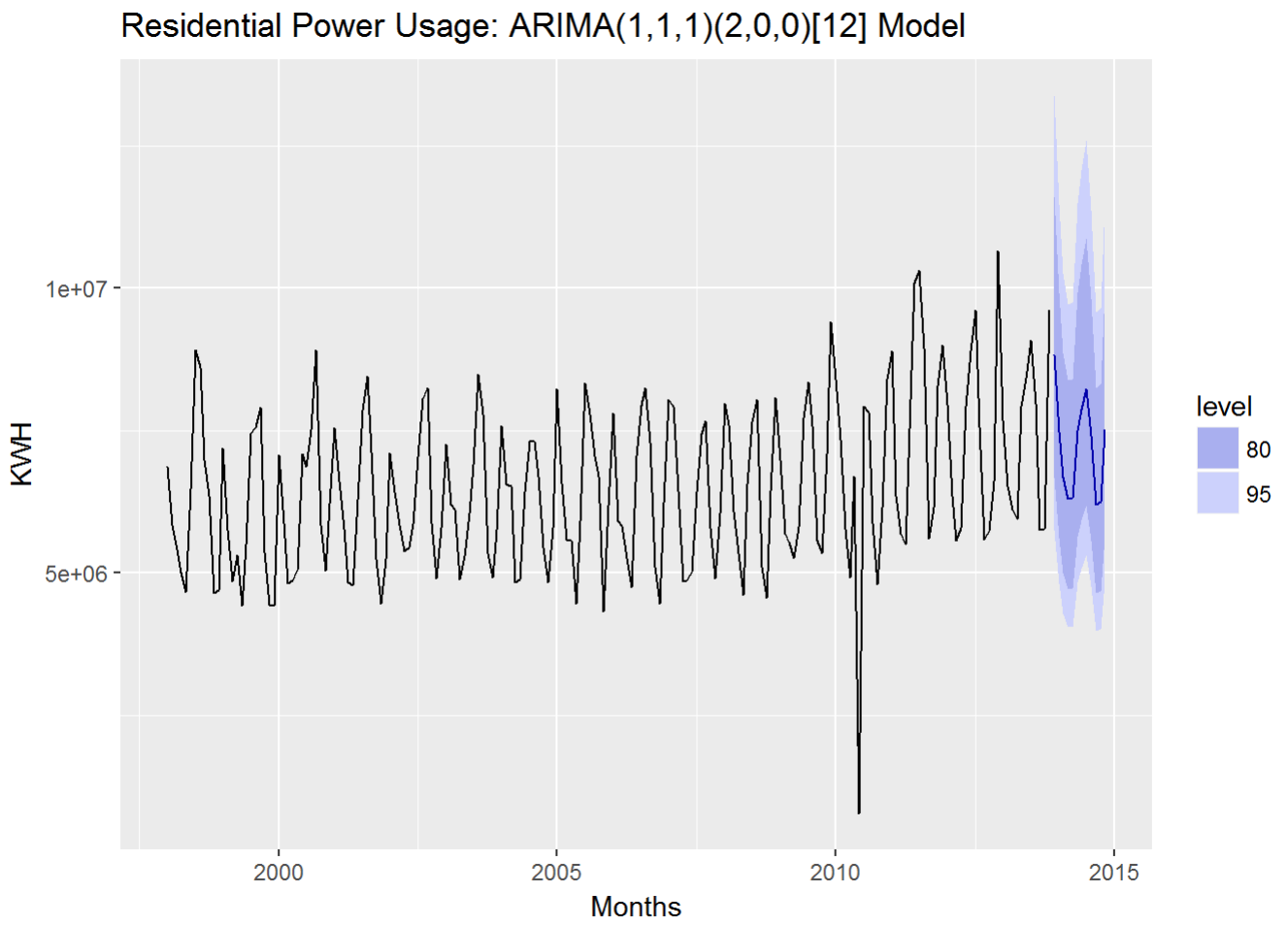



```
Box.test(residuals(fit), lag = 12, fitdf = 4, type = "Ljung")
```

```
##
##  Box-Ljung test
##
## data:  residuals(fit)
## X-squared = 10.23, df = 8, p-value = 0.2492
```

Results with 80% confidence interval and 95% confidence interval are plotted. The output is save into the previous excel file by making append = TRUE.

```
forecast <- forecast(fit, h = 12)
autoplot(forecast, main = "Residential Power Usage: ARIMA(1,1,1)(2,0,0)[12] Model", xlab = "Mont
hs", ylab = "KWH")
```




```
result2 <- data.frame(KWH = forecast$mean)
write.xlsx(result2, file = "Project1_Output.xlsx", sheetName="Residential Power Usage", append=T
RUE)
```

Part C - BONUS, optional (part or all)

Part C consists of two data sets. These are simple 2 columns sets, however they have different time stamps. Your optional assignment is to time-base sequence the data and aggregate based on hour (example of what this looks like, follows). Note for multiple recordings within an hour, take the mean. Then to determine if the data is stationary and can it be forecast. If so, provide a week forward forecast and present results as above in a Word readable file and the forecast in an Excel readable file.