DATA 624 - PROJECT 1

OMER OZEREN

Table of Contents

Project 1

```
library(knitr)
library(ggplot2)
```

```r
library(tidyr)
library(dplyr)
library(tseries)
library(forecast)
library(lubridate)
library(tidyverse)
library(gridExtra)
library(kableExtra)
```

Load Data

Load all three data for the project.

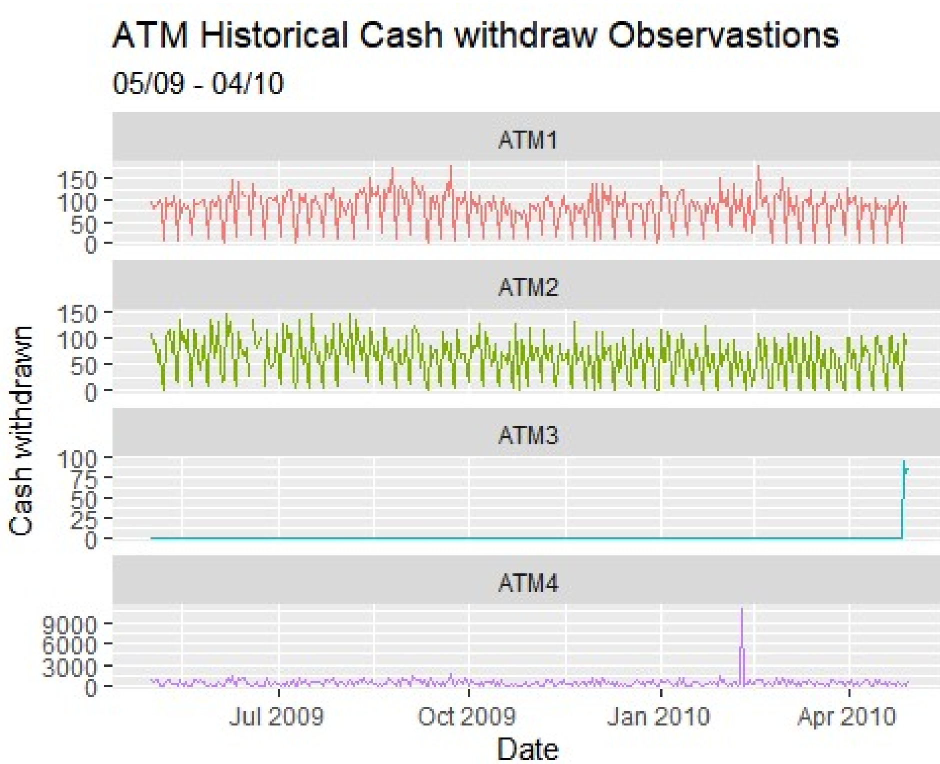
```r
temp_file <- tempfile(fileext = ".xlsx")
download.file(url =
"https://github.com/omerozeren/DATA624/blob/master/Project1/ATM624Data.xlsx?r
aw=true",
              destfile = temp_file,
              mode = "wb",
              quiet = TRUE)
atm_data <- readxl::read_excel(temp_file,skip=0,col_types =
c("date","text","numeric"))
download.file(url =
"https://github.com/omerozeren/DATA624/blob/master/Project1/ResidentialCustom
erForecastLoad-624.xlsx?raw=true",
              destfile = temp_file,
              mode = "wb",
              quiet = TRUE)
power_data <- readxl::read_excel(temp_file,skip=0,col_types =
c("numeric","text","numeric"))
download.file(url =
"https://github.com/omerozeren/DATA624/blob/master/Project1/Waterflow_Pipe1.x
lsx?raw=true",
              destfile = temp_file,
              mode = "wb",
              quiet = TRUE)
water1_data <- readxl::read_excel(temp_file,skip=0,col_types =
c("date","numeric"))
download.file(url =
"https://github.com/omerozeren/DATA624/blob/master/Project1/Waterflow_Pipe2.x
lsx?raw=true",
              destfile = temp_file,
              mode = "wb",
              quiet = TRUE)
water2_data <- readxl::read_excel(temp_file,skip=0,col_types =
c("date","numeric"))
```

Part A - ATM Forecast

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 to make this have a little more business feeling. Explain and demonstrate your process, techniques used and not used, and your actual forecast. I am giving you data via an excel file, please provide your written report on your findings, visuals, discussion and your R code via an RPubs link along with the actual.rmd file Also please submit the forecast which you will put in an Excel readable file.

ATM Data

```
df<-atm_data %>%
  drop_na() %>%
  spread(ATM, Cash) %>%
  mutate(DATE = as.Date(DATE, origin = "1899-12-30"))
atm_ts<-ts(df %>% select(-DATE))

df %>% gather(atm_data, Cash, -DATE) %>%
   ggplot(aes(x = DATE, y = Cash, col = atm_data)) +
   geom_line(show.legend = FALSE) +
   facet_wrap(~ atm_data, ncol = 1, scales = "free_y") +
   labs(title = "ATM Historical Cash withdraw Observastions", subtitle =
"05/09 - 04/10", x = "Date") +
   scale_y_continuous("Cash withdrawn ")
```



The plot shows each ATM's historically cash withdraws.The ATM1 and ATM2 look solid Time series data where they have their own high and lows values. However, ATM3 data has most values zero, that is going to be challenging for building a model.I might need to take just mean valuers of data instead of building time series model.The ATM4 also has some challenges because it cointains some extreme values.

ATM #1

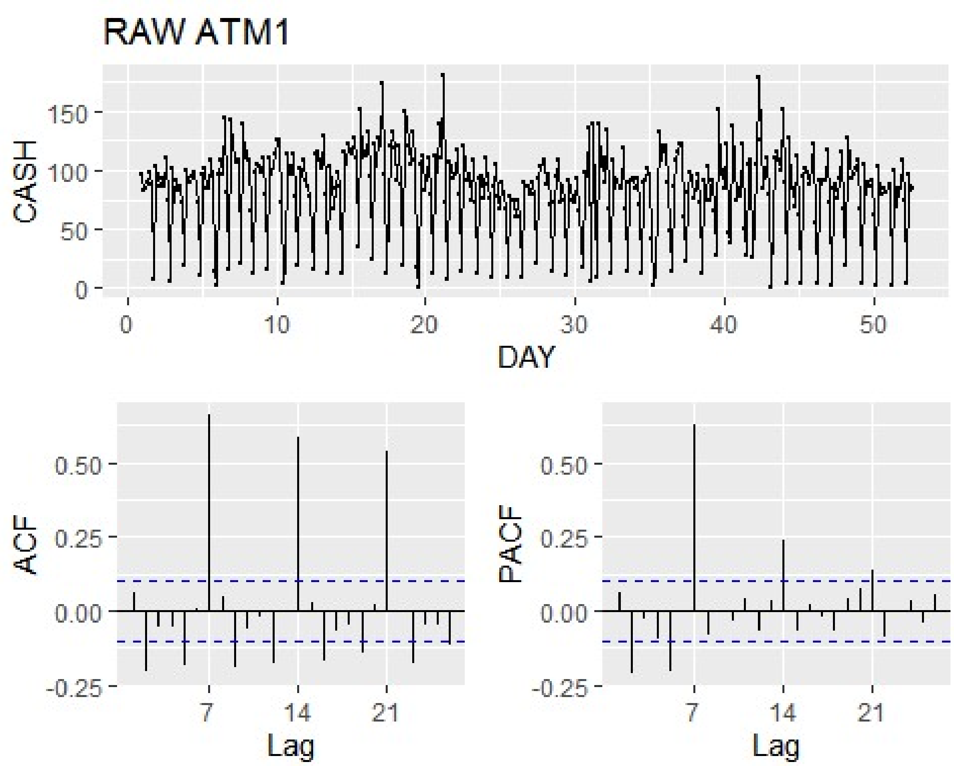
Data Cleanup

There are only three missing values in atm_1 data. I will impute the missing values in order to create a timeseries data.

```
atm1 <- atm_data %>%
  filter(ATM == "ATM1")
```

Here, I'd like to review the "atm1" timeseries data to determine whether there is a seosanality and ACF and PACF plots.

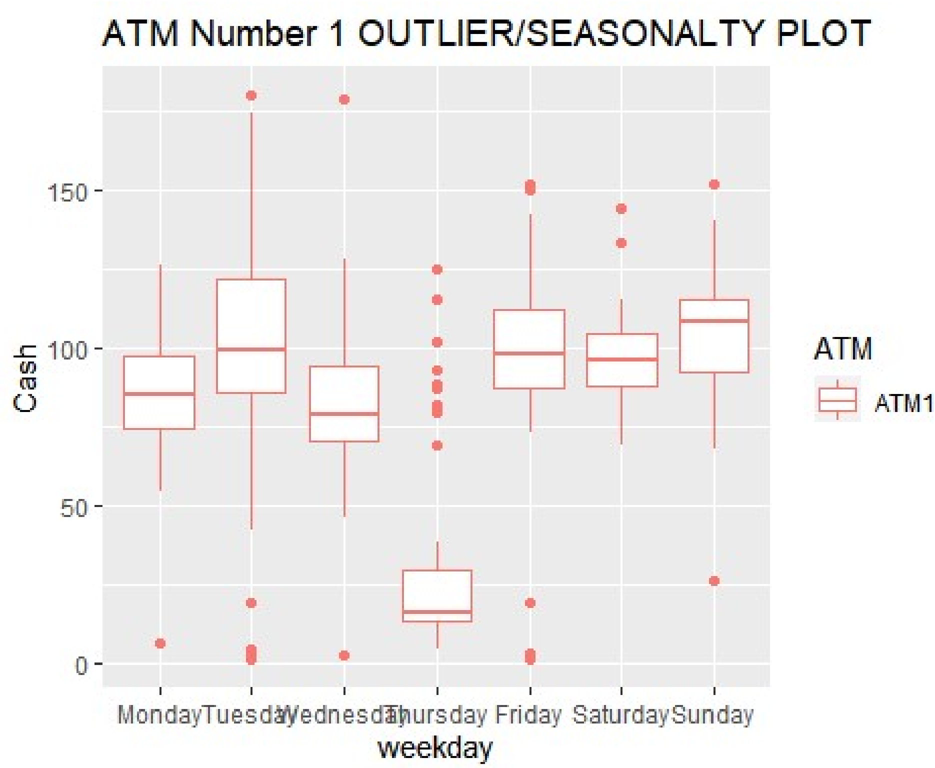
```
ATM1 <- atm_data[atm_data$ATM == "ATM1",]
ATM1 <- ATM1[complete.cases(ATM1),]
ATM1 <- ts(ATM1[c("Cash")],frequency = 7)
ggtsdisplay(ATM1,
            main = "RAW ATM1",
            xlab = "DAY",
            ylab = "CASH")
```



The ACF plot indicates that observations peak on every 7th lag.I will check Seasonality and Outlier for atm1 data.

Outlier and Seasonality

```
atm1$weekday <- factor(weekdays(as.Date(atm1$DATE)))
atm1$weekday <- ordered(atm1$weekday,levels =
c("Monday","Tuesday","Wednesday","Thursday","Friday","Saturday","Sunday"))
#drop NaN values
atm1 <- atm1[complete.cases(atm1),]
ggplot(atm1[complete.cases(atm1),],aes(x=weekday,y=Cash,color=ATM))+
  geom_boxplot()+
  ggtitle("ATM Number 1 OUTLIER/SEASONALTY PLOT")
```



Box plots of the amount of cash is taken by customer in each of days from ATM1.As we can see on plot above, the Sunday has higher mean amount of cash is taken than rest of days.The thursday has the minumum mean of cash amount is taken.The weekdays Tuesdays and Wendsday hold some extreme values (greater than 150).The analysis above drops missing values.

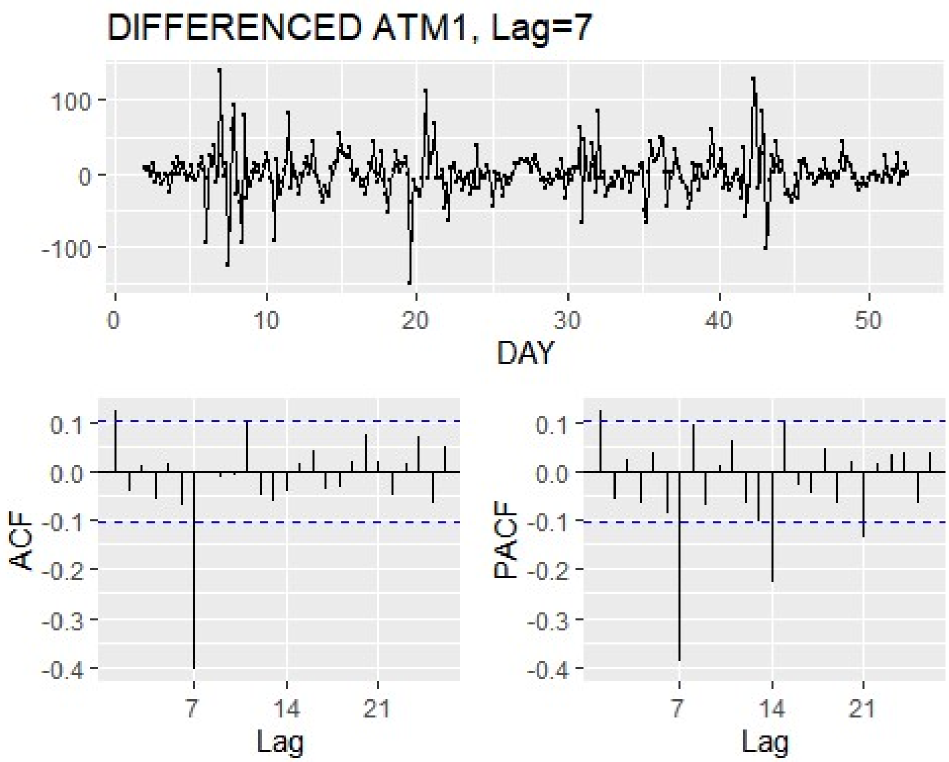
Model Creation

I will use following forecating models on this time series and determine which one is better by estimatin error metric RMSE. I will use time series cross validation function to estimate RMSE for timeseries data.

- Seasonal and Trend decomposition (STL)

- Seasonal and Trend decomposition (STL) with ARIMA
- Holt-Winters
- Holt-Winters with Box Cox Adjustment
- ARIMA with Box Cox Adjustment

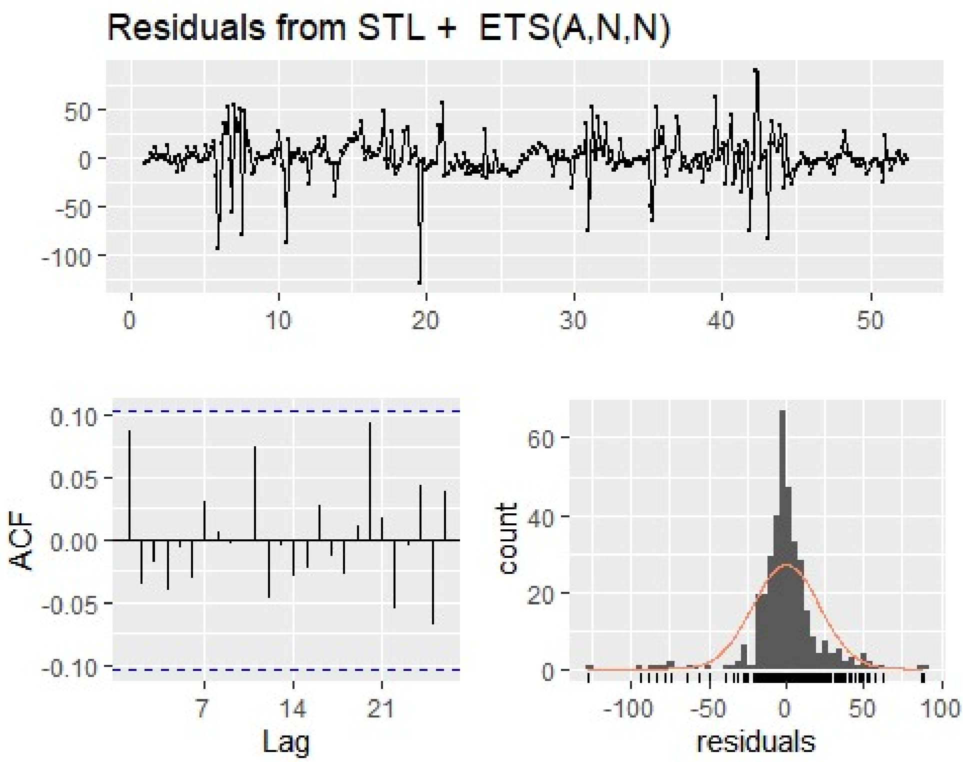
```
ATM1 <- atm_data[atm_data$ATM == "ATM1",]
ATM1 <- ATM1[complete.cases(ATM1),]
ATM1 <- ts(ATM1[c("Cash")],frequency = 7)
Box.test(diff(ATM1,lag=7), type = "Ljung-Box")

##
##  Box-Ljung test
##
## data:  diff(ATM1, lag = 7)
## X-squared = 5.4172, df = 1, p-value = 0.01994

kpss.test(diff(ATM1,lag=7))

## Warning in kpss.test(diff(ATM1, lag = 7)): p-value greater than printed p-
value

##
##  KPSS Test for Level Stationarity
##
## data:  diff(ATM1, lag = 7)
## KPSS Level = 0.021687, Truncation lag parameter = 5, p-value = 0.1

ggtsdisplay(diff(ATM1,lag=7),
            main = "DIFFERENCED ATM1, Lag=7",
            xlab = "DAY",
            ylab = "")
```



The result of KPSS and Box-Cox test indicates that atm1 timeseries is statioanry.

Seasonal and Trend decomposition (STL)

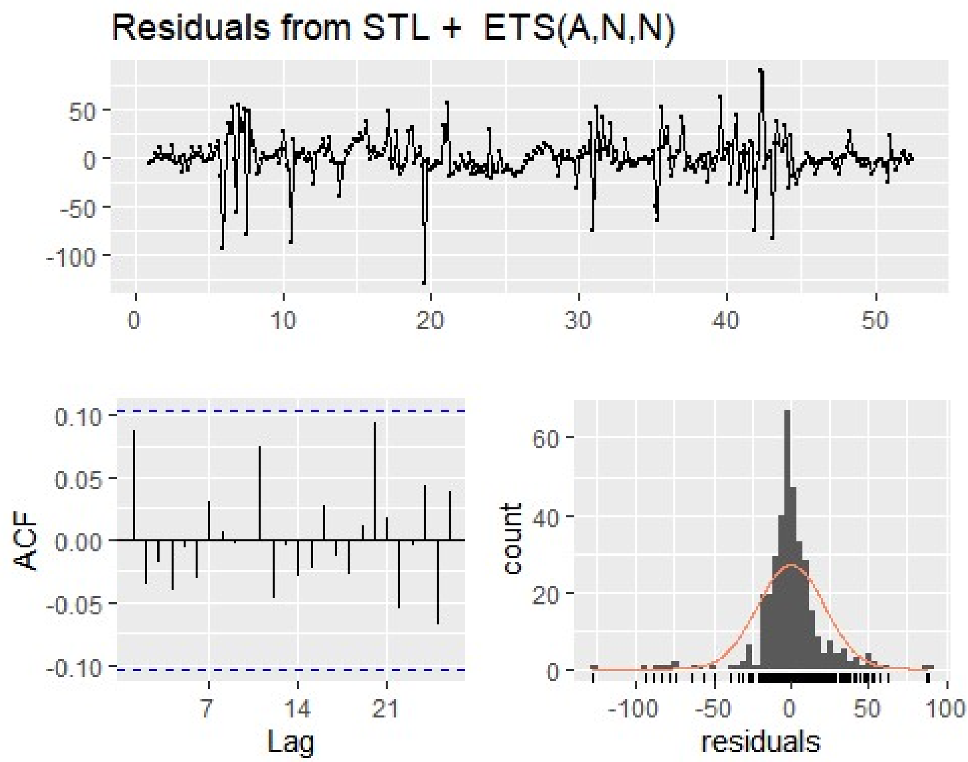
```
atm1_stl_fit <- ATM1 %>%
  stlf(h = 31, s.window = 7, robust = TRUE)
checkresiduals(atm1_stl_fit)

## Warning in checkresiduals(atm1_stl_fit): The fitted degrees of freedom is
based
## on the model used for the seasonally adjusted data.
```




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 7.9258, df = 12, p-value = 0.7909
##
## Model df: 2.   Total lags used: 14
```

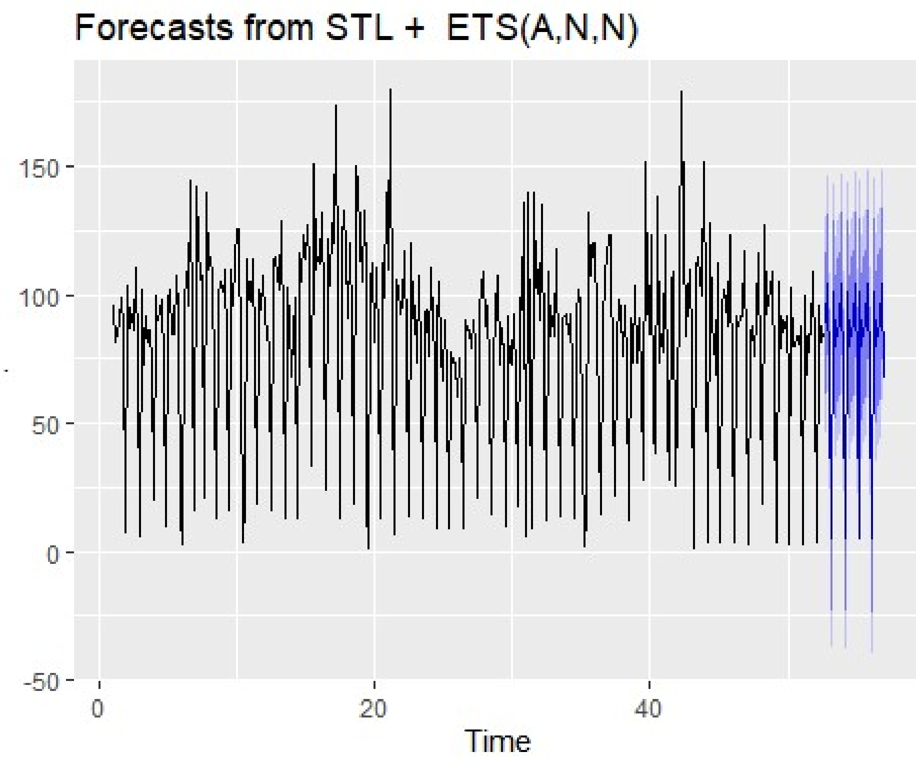
```
checkresiduals(atm1_stl_fit)
```

```
## Warning in checkresiduals(atm1_stl_fit): The fitted degrees of freedom is
based
## on the model used for the seasonally adjusted data.
```



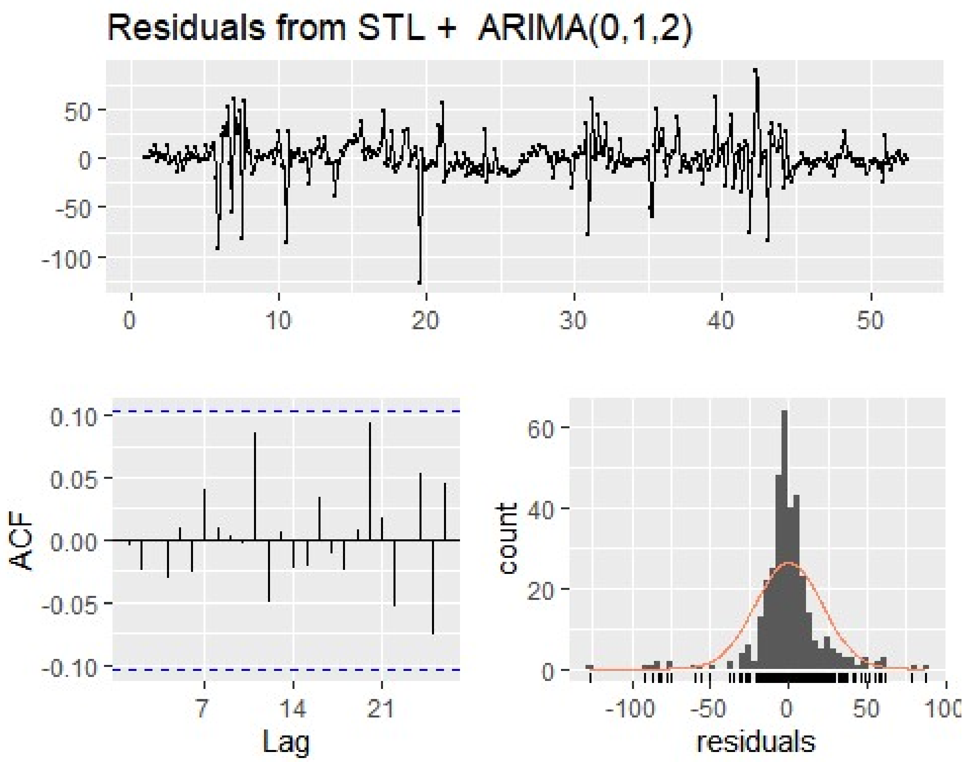

```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 7.9258, df = 12, p-value = 0.7909
##
## Model df: 2.   Total lags used: 14
```

```
atm1_stl_fit%>% forecast(h=31) %>% autoplot()
```



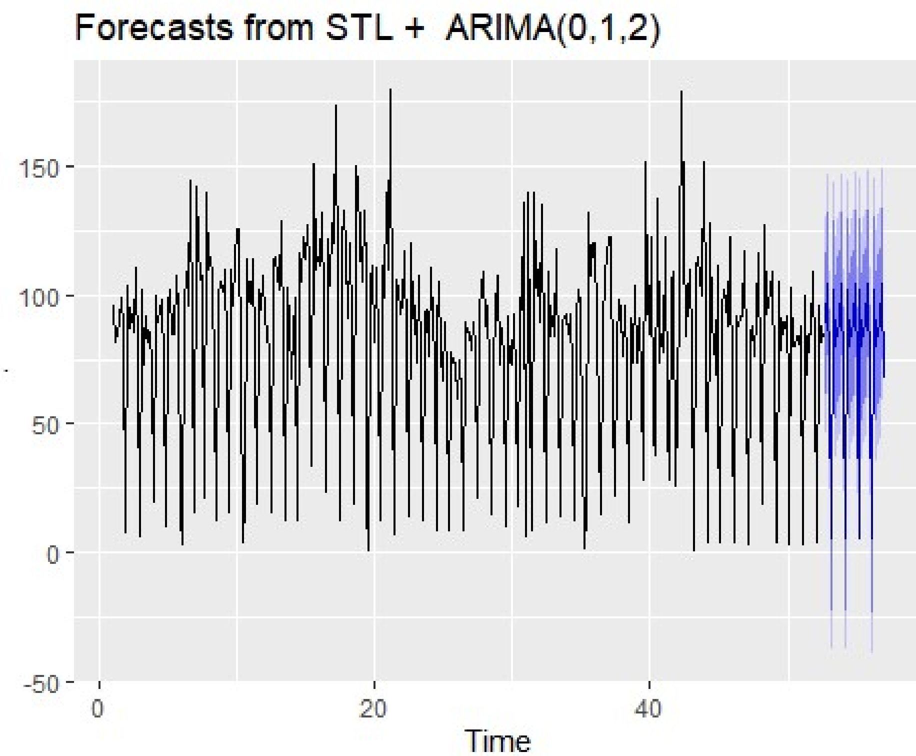
Seasonal and Trend decomposition (STL) with ARIMA

```
atm1_stl_arima_fit <- ATM1 %>%
  stlf(h = 31, s.window = 7, robust = TRUE, method = "arima")
checkresiduals(atm1_stl_arima_fit)

## Warning in checkresiduals(atm1_stl_arima_fit): The fitted degrees of
freedom is
## based on the model used for the seasonally adjusted data.
```



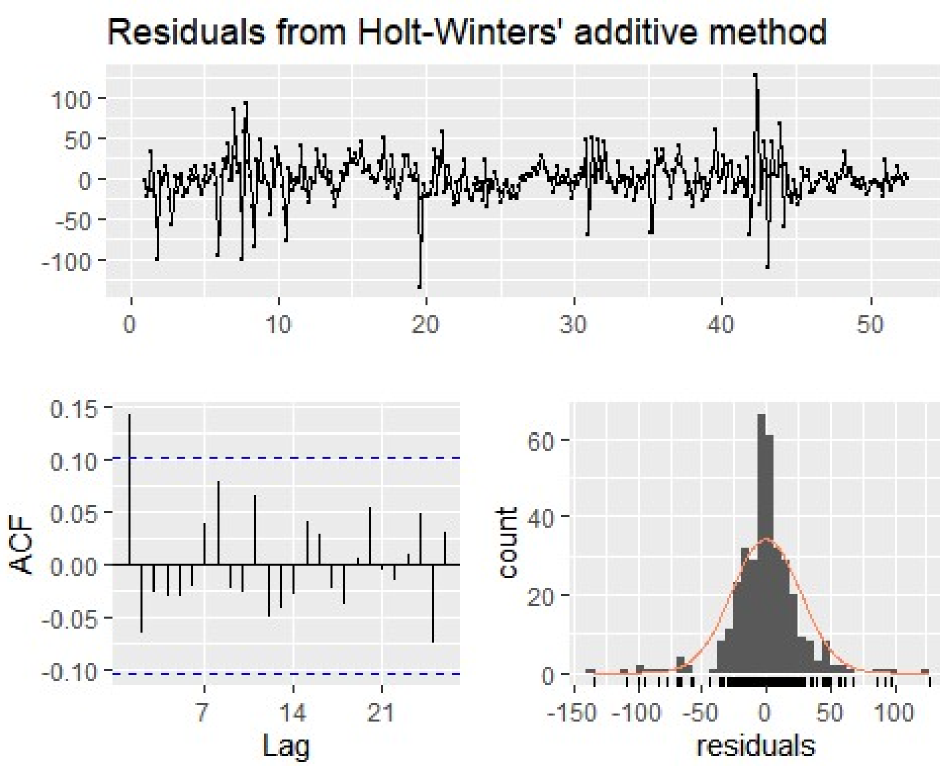

```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ARIMA(0,1,2)
## Q* = 5.4248, df = 12, p-value = 0.9423
##
## Model df: 2.   Total lags used: 14
```

```
atm1_stl_arima_fit%>% forecast(h=31) %>% autoplot()
```



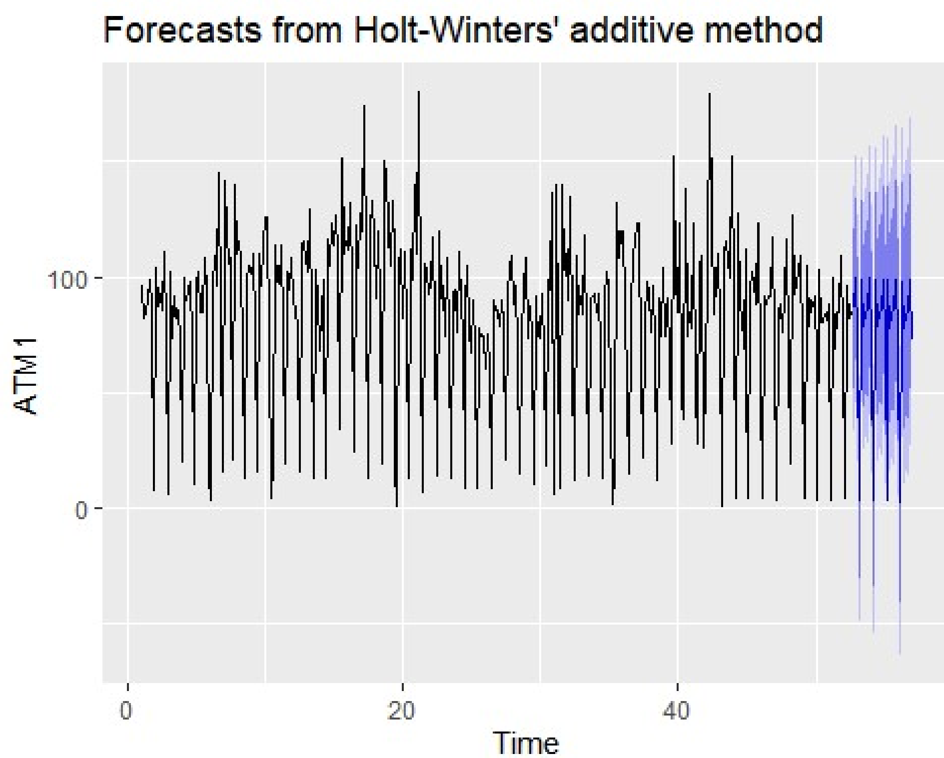
Holt-Winters

```
atm1_hw_fit <- hw(ATM1, h = 31)
checkresiduals(atm1_hw_fit)
```



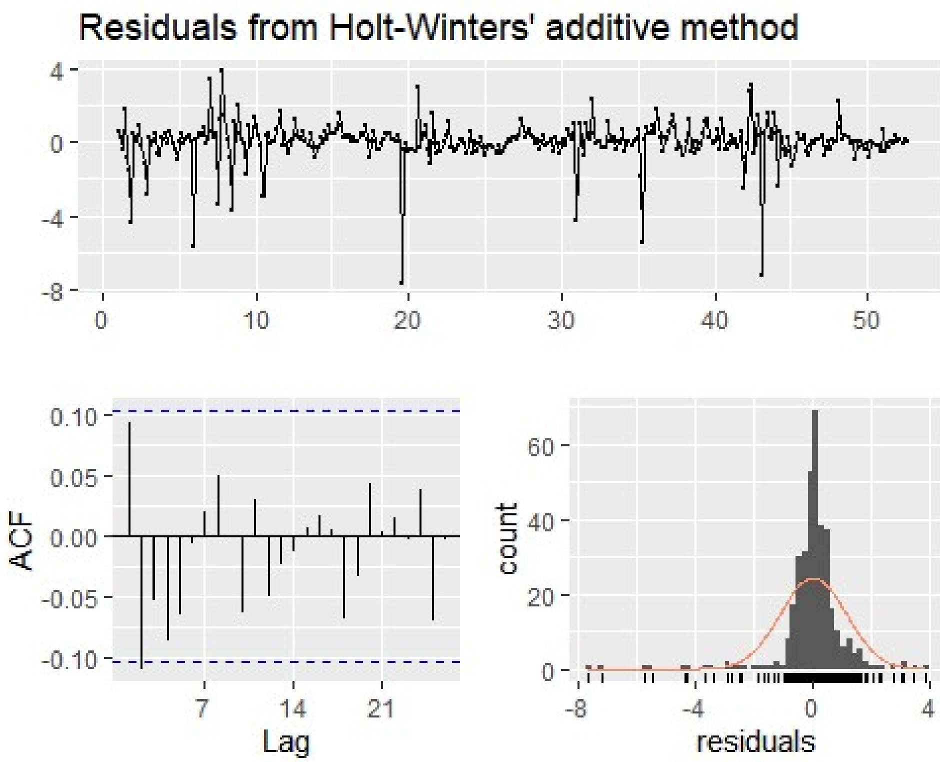
```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 16.871, df = 3, p-value = 0.0007513
##
## Model df: 11.   Total lags used: 14
```

```
atm1_hw_fit%>% forecast(h=31) %>% autoplot()
```



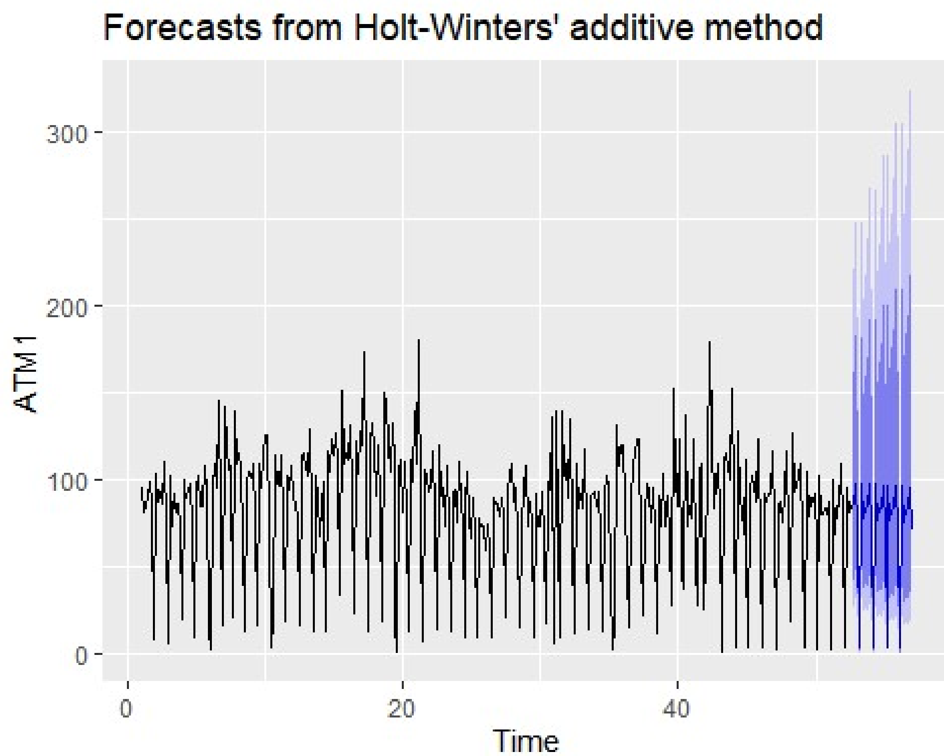
Holt-Winters with Box Cox Adjustment

```
atm1_lambda <- BoxCox.lambda(ATM1)
atm1_adj_hw_fit <- hw(ATM1, h = 31, lambda = atm1_lambda)
checkresiduals(atm1_adj_hw_fit)
```



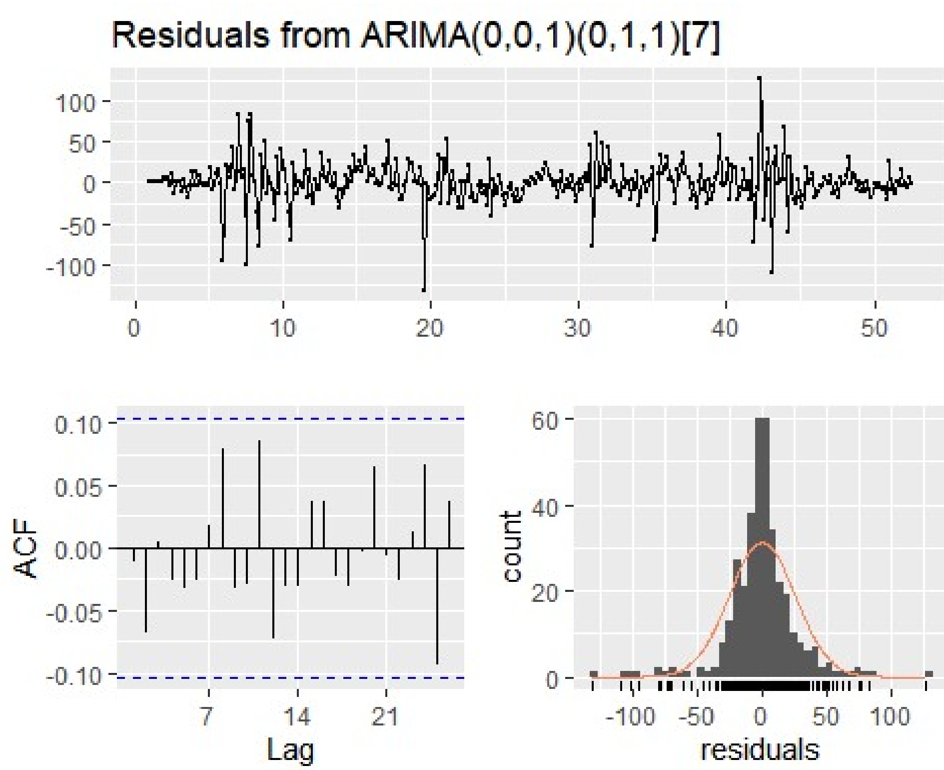

```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 16.921, df = 3, p-value = 0.0007336
##
## Model df: 11.   Total lags used: 14
```

```
atm1_adj_hw_fit%>% forecast(h=31) %>% autoplot()
```



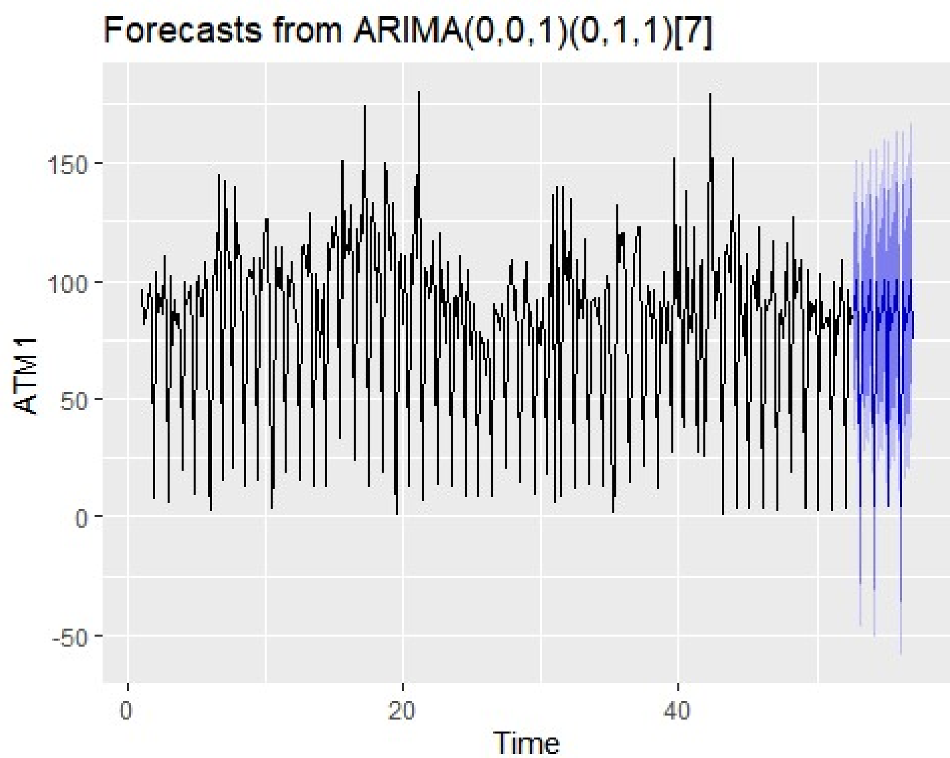
ARIMA

```
atm1_arima_fit <- auto.arima(ATM1)
checkresiduals(atm1_arima_fit)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,0,1)(0,1,1)[7]
## Q* = 11.267, df = 12, p-value = 0.5062
##
## Model df: 2.   Total lags used: 14
```

```
atm1_arima_fit%>% forecast(h=31) %>% autoplot()
```



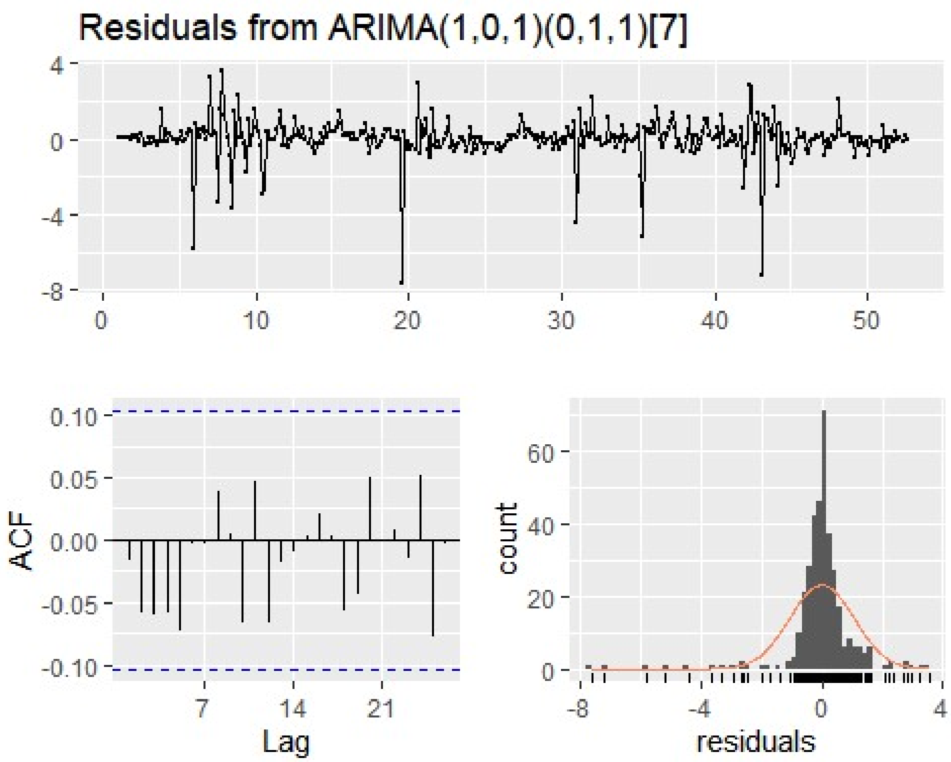

```
kpss.test(resid(atm1_arima_fit))
```

```
## Warning in kpss.test(resid(atm1_arima_fit)): p-value greater than printed
p-
## value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  resid(atm1_arima_fit)
## KPSS Level = 0.098551, Truncation lag parameter = 5, p-value = 0.1
```

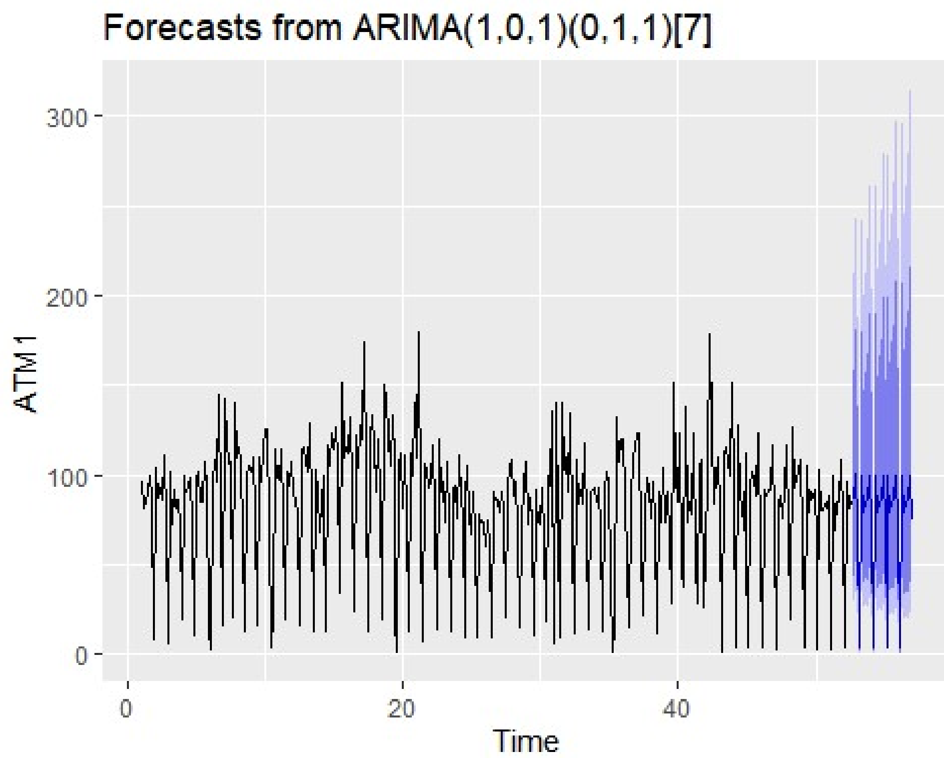
ARIMA with Box Cox Adjustment

```
atm1_lambda = BoxCox.lambda(ATM1)
atm1_box_arima_fit <- Arima(ATM1, order = c(1, 0, 1), seasonal = c(0, 1, 1),
lambda = atm1_lambda)
checkresiduals(atm1_box_arima_fit)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,1)(0,1,1)[7]
## Q* = 10.658, df = 11, p-value = 0.4724
##
## Model df: 3.   Total lags used: 14
```

```
atm1_box_arima_fit%>% forecast(h=31) %>% autoplot()
```




```
kpss.test(resid(atm1_box_arima_fit))

## Warning in kpss.test(resid(atm1_box_arima_fit)): p-value greater than
printed p-
## value

##
##  KPSS Test for Level Stationarity
##
## data:  resid(atm1_box_arima_fit)
## KPSS Level = 0.062513, Truncation lag parameter = 5, p-value = 0.1
```

MODEL EVALUATION

I will use the tsCV function and evaluate the models.My goal is to find the model that produces minumum RMSE.

```
h <- 31
get_rmse <- function(error) {
  sqrt(mean(error^2, na.rm = TRUE))
}
atm1_arima_forecast <- function(x, h) {
  forecast(Arima(x, order = c(0, 0, 1), seasonal = c(0, 1, 1)), h = h)
}
atm1_arima_box_forecast <- function(x, h) {
  forecast(Arima(x, order = c(1, 0, 1), seasonal = c(0, 1, 1),lambda =
atm1_lambda), h = h)
```

```
}

residuals_stl <- tsCV(ATM1, stlf, h = h, s.window = 7, robust = TRUE)
residuals_stl_arima <- tsCV(ATM1, stlf, h = h, s.window = 7, robust = TRUE,
method = "arima")
residuals_hw <- tsCV(ATM1, hw, h = h)
residuals_arima <- tsCV(ATM1, atm1_arima_forecast, h = h)
residuals_arima_box <- tsCV(ATM1, atm1_arima_box_forecast, h = h)
data.frame(Model_Name = c("STL", "STL & ARIMA", "Holt-Winters",
"ARIMA","ARIMA-BOX_COX"),
           RMSE = c(get_rmse(residuals_stl[, h]),
get_rmse(residuals_stl_arima[, h]), get_rmse(residuals_hw[, h]),
get_rmse(residuals_arima[, h]),

get_rmse(residuals_arima_box[, h]))) %>%
  arrange(RMSE) %>%
  kable() %>%
  kable_styling()
```

Model_Name

RMSE

ARIMA

35.22610

ARIMA-BOX_COX

36.14111

STL & ARIMA

37.99406

STL

38.75113

Holt-Winters

46.03976

The ARIMA(0,0,1)(0,1,1) model gives the minumum RMSE among the other models.

ATM #2

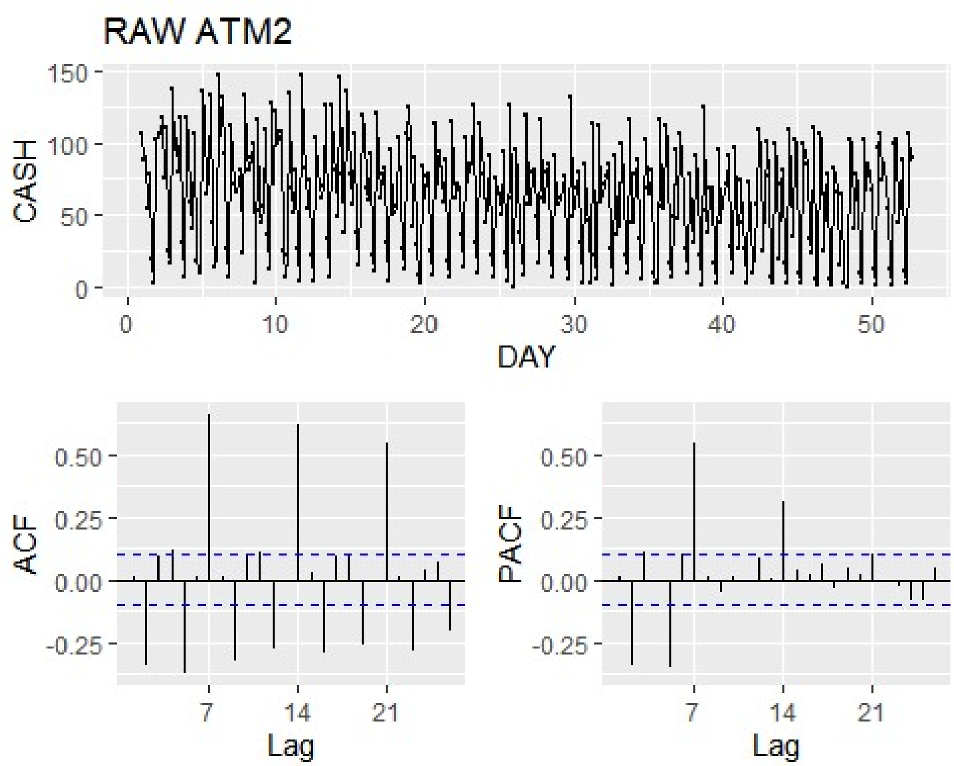
Data Cleanup

```
atm2 <- atm_data %>%
  filter(ATM == "ATM2")
```

Here, I'd like to review the "atm2" timeseries data to determine whether there is a seosanality and ACF and PACF plots.

```
ATM2 <- atm_data[atm_data$ATM == "ATM2",]
ATM2 <- ATM2[complete.cases(ATM2),]
ATM2 <- ts(ATM2[c("Cash")],frequency = 7)
ggtsdisplay(ATM2,
            main = "RAW ATM2",
```

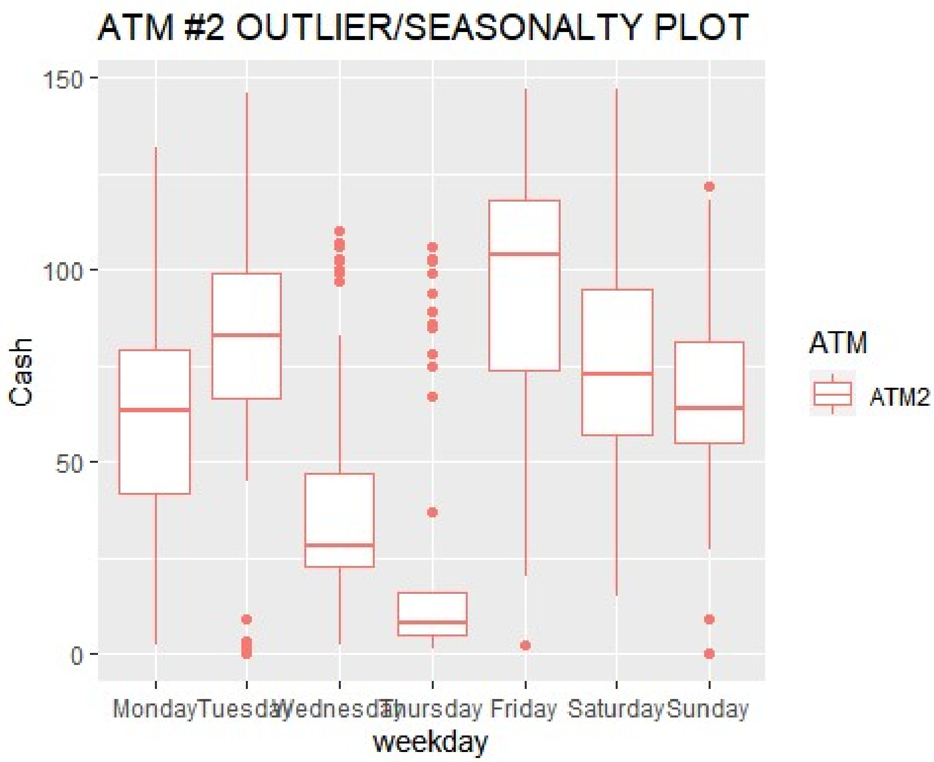
```
            xlab = "DAY",
            ylab = "CASH")
```



As above plots, the large spike at lag=2,5,7 suggests I=1,I will check Seasonality and Outlier for atm2 data.

Outlier and Seasonality

```
atm2$weekday <- factor(weekdays(as.Date(atm2$DATE)))
atm2$weekday <- ordered(atm2$weekday,levels =
c("Monday","Tuesday","Wednesday","Thursday","Friday","Saturday","Sunday"))
#drop NaN values
atm2 <- atm2[complete.cases(atm2),]
ggplot(atm2[complete.cases(atm2),],aes(x=weekday,y=Cash,color=ATM))+
  geom_boxplot()+
  ggtitle("ATM #2 OUTLIER/SEASONALTY PLOT")
```



Box plots of the amount of cash is taken by customer in each of days from ATM2.As we can see on plot above, the Friday has higher mean amount of cash is taken than rest of days.The thursday has the minumum mean of cash amount is taken.

Model Creation

I will use following forecating models on this time series and determine which one is better by estimatin error metric RMSE. I will use time series cross validation function to estimate RMSE for timeseries data.

- Seasonal and Trend decomposition (STL)
- Seasonal and Trend decomposition (STL) with ARIMA
- Holt-Winters
- Holt-Winters with Box Cox Adjustment
- ARIMA
- ARIMA with Box Cox Adjustment

```
ATM2 <- atm_data[atm_data$ATM == "ATM2",]
ATM2 <- ATM2[complete.cases(ATM2),]
ATM2 <- ts(ATM2[c("Cash")],frequency = 7)
Box.test(diff(ATM2,lag=7), type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
```

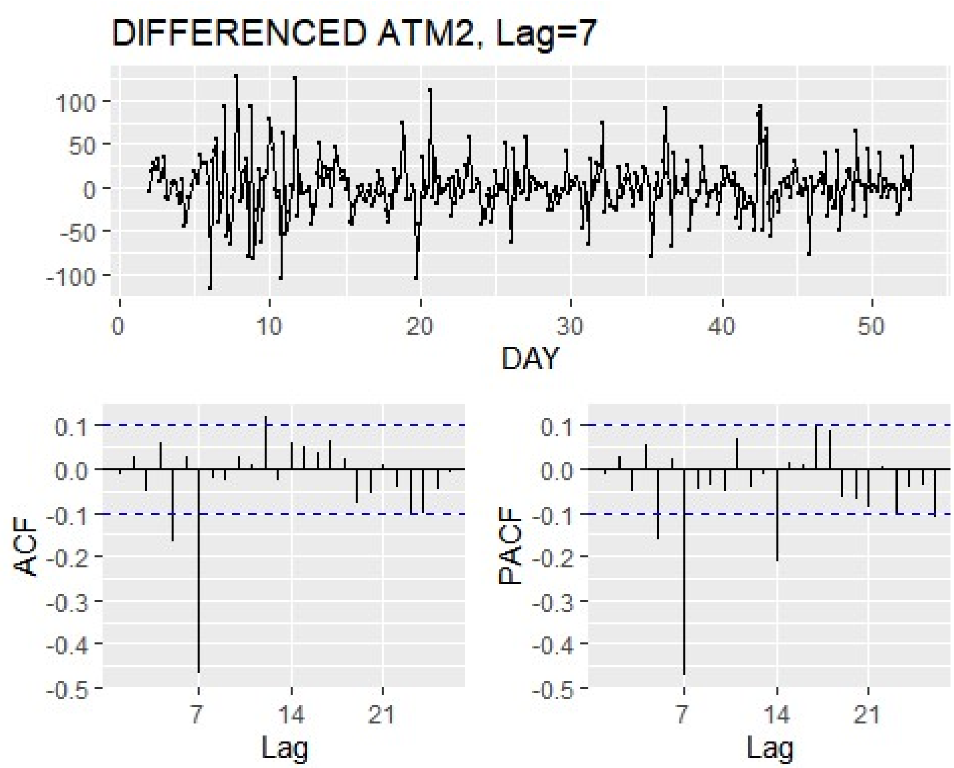
```
## data:  diff(ATM2, lag = 7)
## X-squared = 0.080312, df = 1, p-value = 0.7769
```

```
kpss.test(diff(ATM2,lag=7))
```

```
## Warning in kpss.test(diff(ATM2, lag = 7)): p-value greater than printed p-
value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  diff(ATM2, lag = 7)
## KPSS Level = 0.017341, Truncation lag parameter = 5, p-value = 0.1
```

```
ggtsdisplay(diff(ATM2,lag=7),
            main = "DIFFERENCED ATM2, Lag=7",
            xlab = "DAY",
            ylab = "")
```

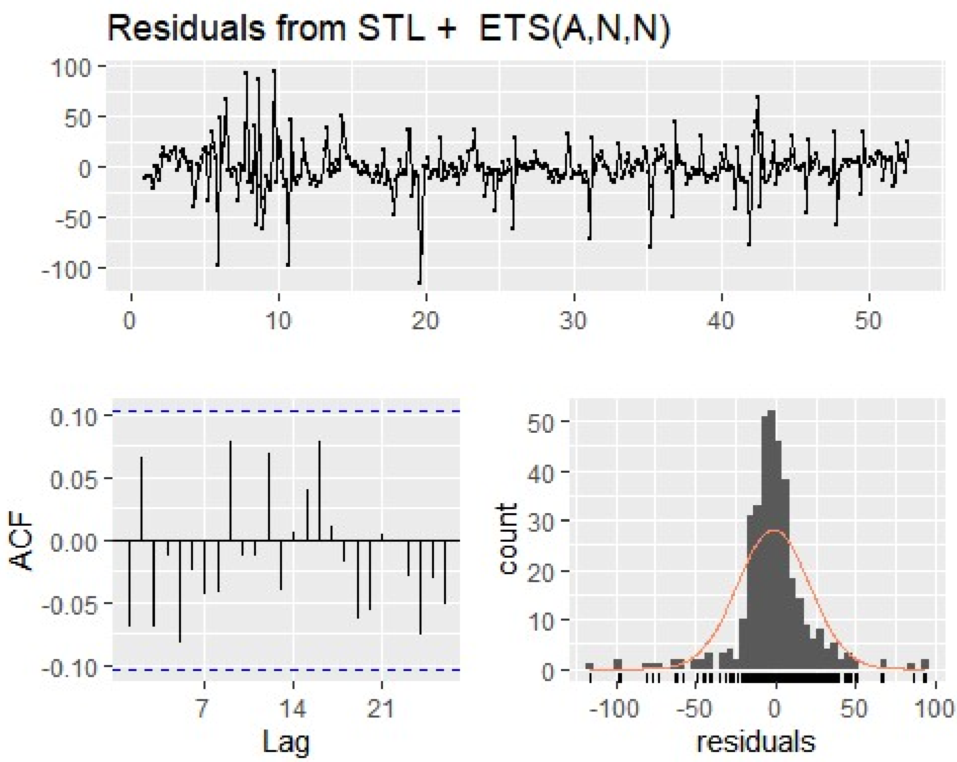


The result of KPSS and Box-Cox test indicates that atm1 timeseries is statioanry.

Seasonal and Trend decomposition (STL)

```
atm2_stl_fit <- ATM2 %>%
  stlf(h = 31, s.window = 7, robust = TRUE)
checkresiduals(atm2_stl_fit)
```

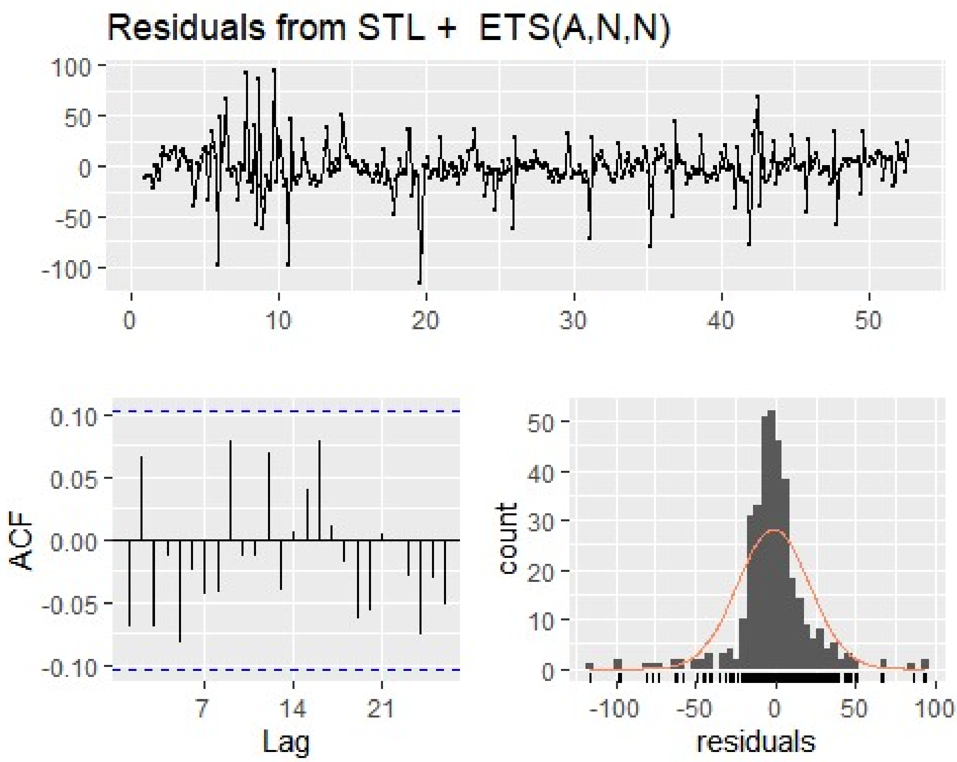
```
## Warning in checkresiduals(atm2_stl_fit): The fitted degrees of freedom is
based
## on the model used for the seasonally adjusted data.
```




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 14.34, df = 12, p-value = 0.2795
##
## Model df: 2.   Total lags used: 14
```

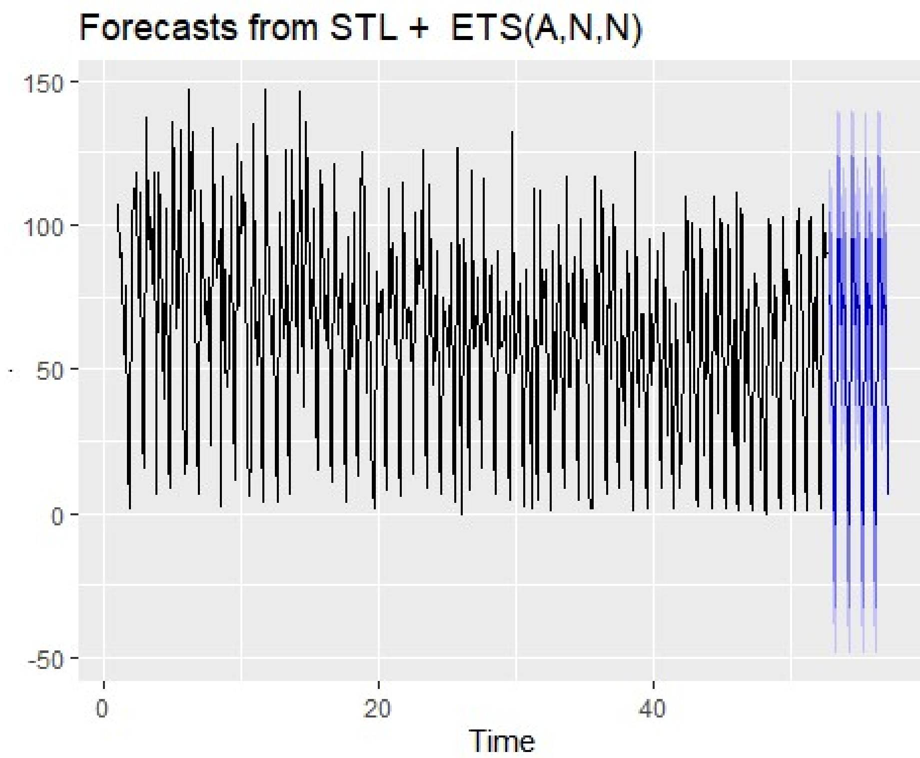
```
checkresiduals(atm2_stl_fit)
```

```
## Warning in checkresiduals(atm2_stl_fit): The fitted degrees of freedom is
based
## on the model used for the seasonally adjusted data.
```




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 14.34, df = 12, p-value = 0.2795
##
## Model df: 2.   Total lags used: 14
```

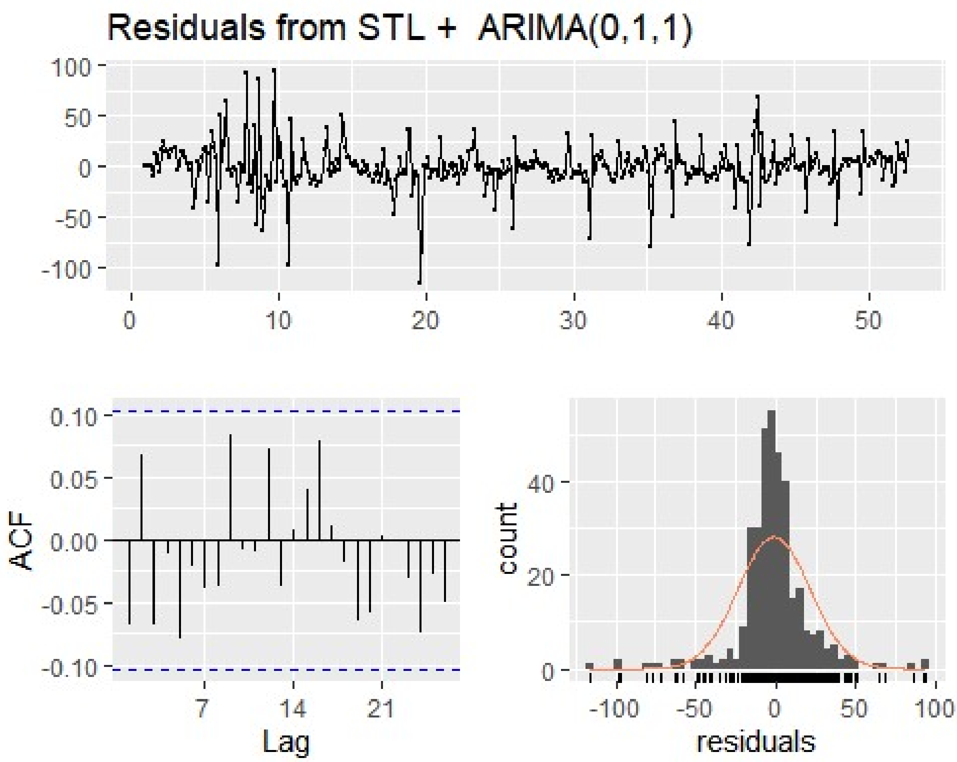
```
atm2_stl_fit%>% forecast(h=31) %>% autoplot()
```



Seasonal and Trend decomposition (STL) with ARIMA

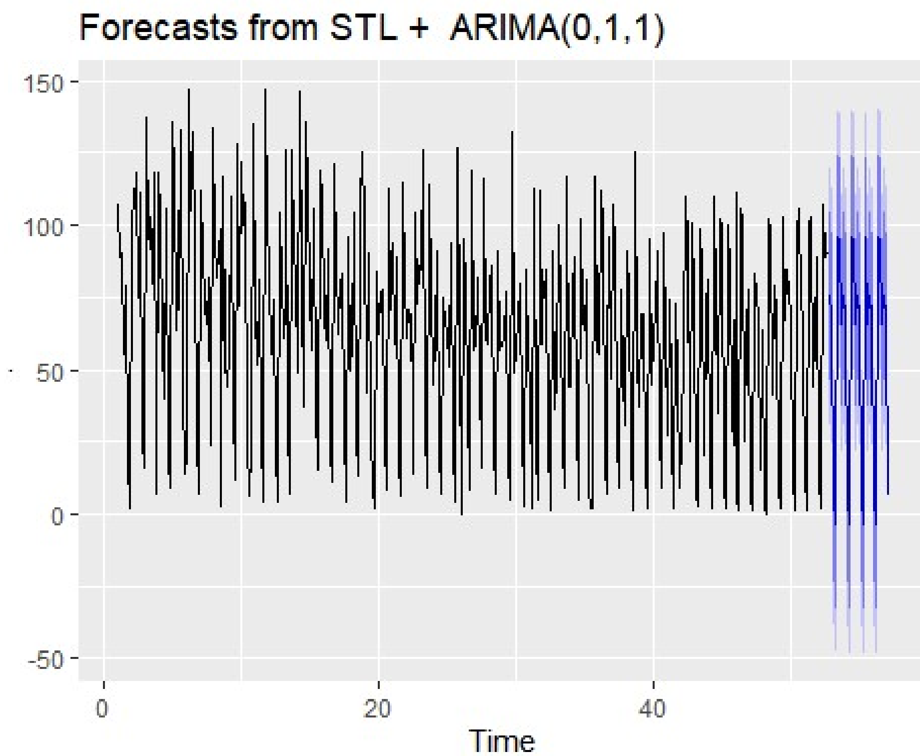
```
atm2_stl_arima_fit <- ATM2 %>%
  stlf(h = 31, s.window = 7, robust = TRUE, method = "arima")
checkresiduals(atm2_stl_arima_fit)
```

```
## Warning in checkresiduals(atm2_stl_arima_fit): The fitted degrees of
freedom is
## based on the model used for the seasonally adjusted data.
```



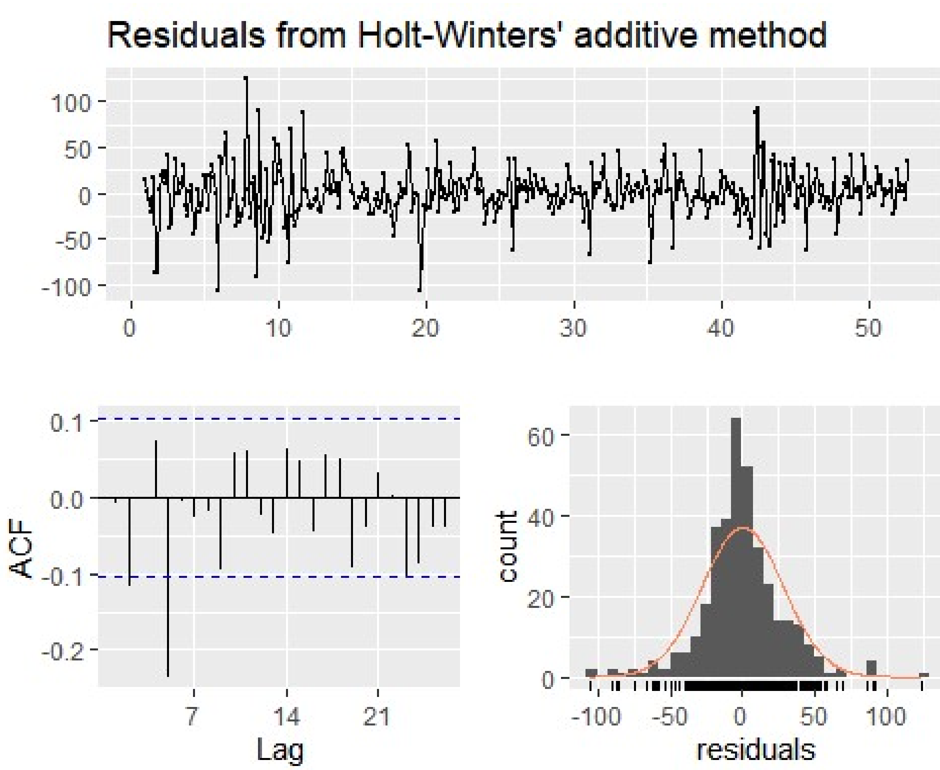

```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ARIMA(0,1,1)
## Q* = 13.983, df = 13, p-value = 0.375
##
## Model df: 1.   Total lags used: 14
```

```
atm2_stl_arima_fit%>% forecast(h=31) %>% autoplot()
```



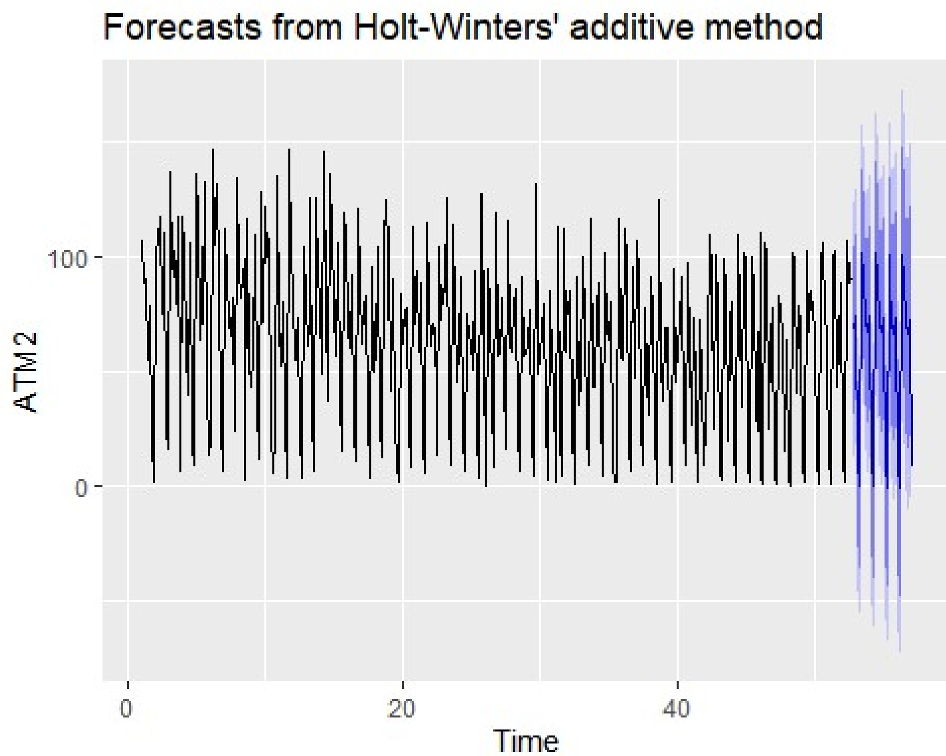
Holt-Winters

```
atm2_hw_fit <- hw(ATM2, h = 31)
checkresiduals(atm2_hw_fit)
```



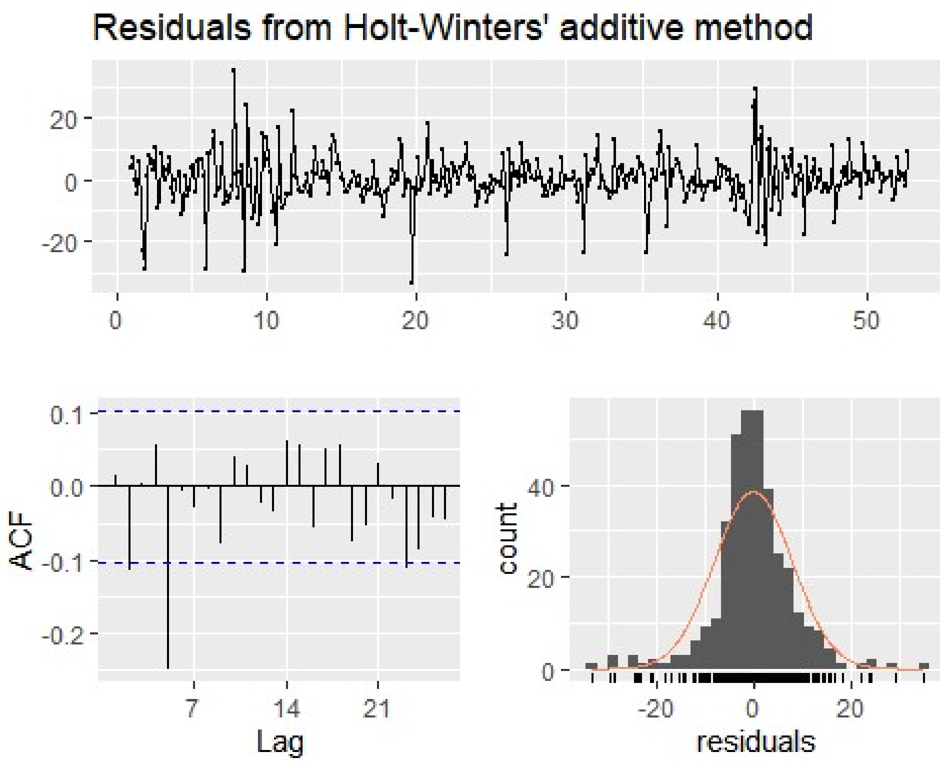
```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 36.563, df = 3, p-value = 5.693e-08
##
## Model df: 11.   Total lags used: 14
```

```
atm2_hw_fit%>% forecast(h=31) %>% autoplot()
```



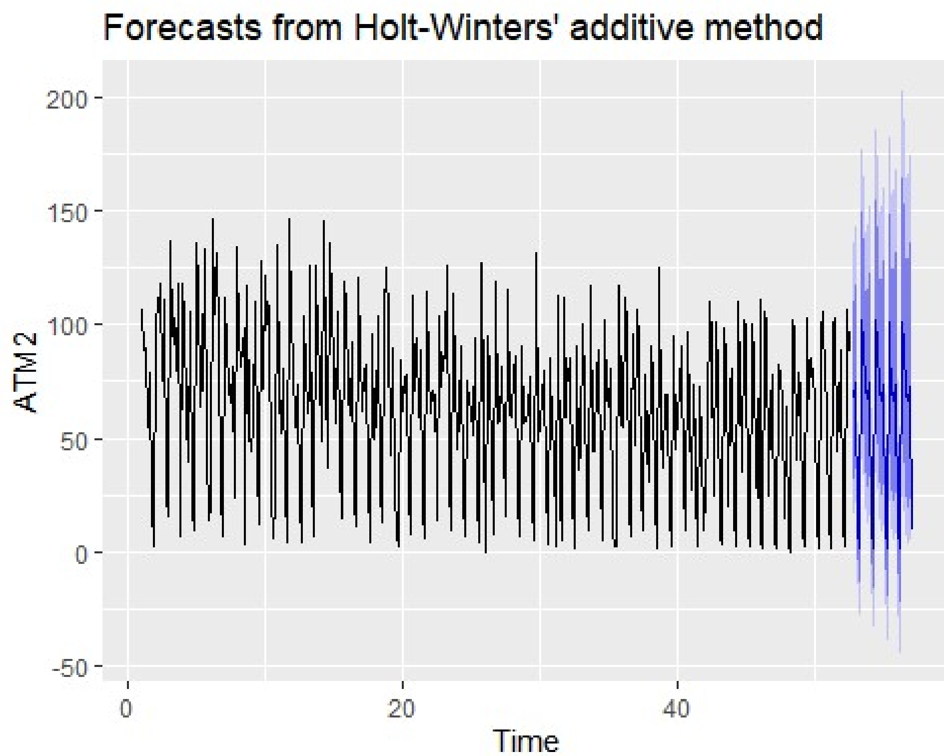
Holt-Winters with Box Cox Adjustment

```
atm2_lambda <- BoxCox.lambda(ATM2)
atm2_adj_hw_fit <- hw(ATM2, h = 31, lambda = atm2_lambda)
checkresiduals(atm2_adj_hw_fit)
```



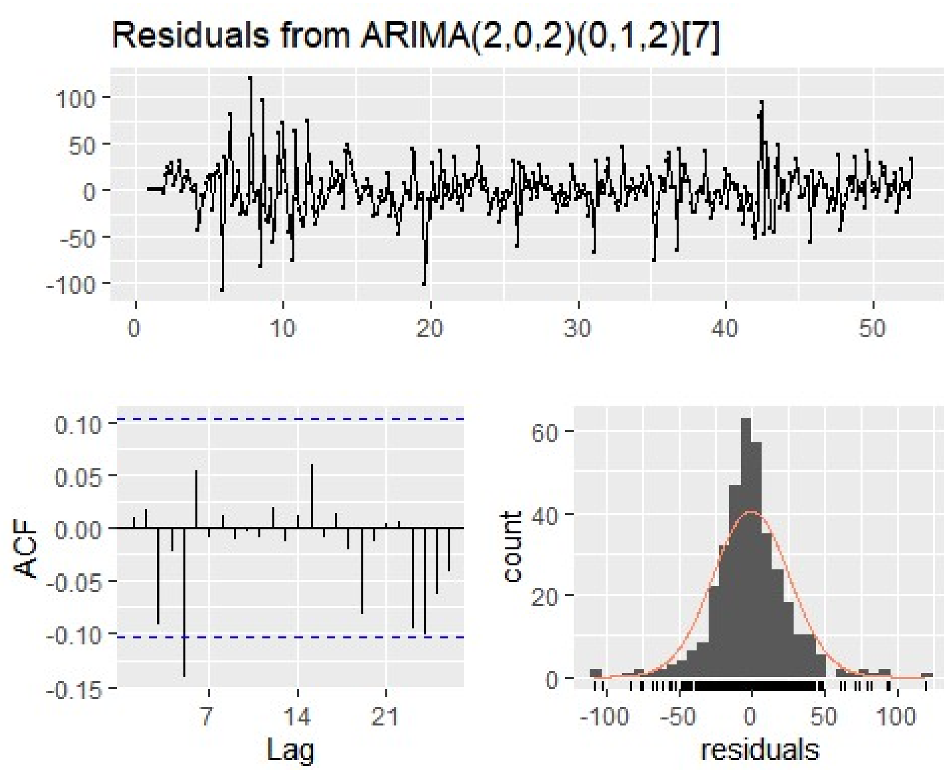

```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 34.635, df = 3, p-value = 1.455e-07
##
## Model df: 11.   Total lags used: 14
```

```
atm2_adj_hw_fit%>% forecast(h=31) %>% autoplot()
```



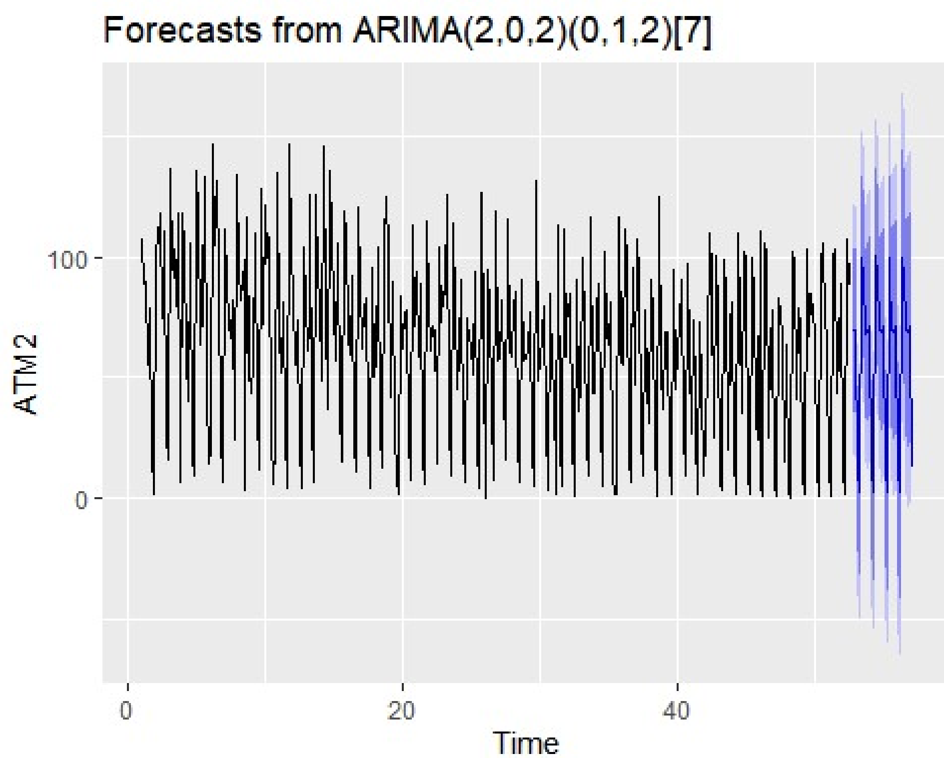
ARIMA

```
atm2_arima_fit <- auto.arima(ATM2)
checkresiduals(atm2_arima_fit)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,2)(0,1,2)[7]
## Q* = 12.12, df = 8, p-value = 0.1459
##
## Model df: 6.   Total lags used: 14
```

```
atm2_arima_fit%>% forecast(h=31) %>% autoplot()
```



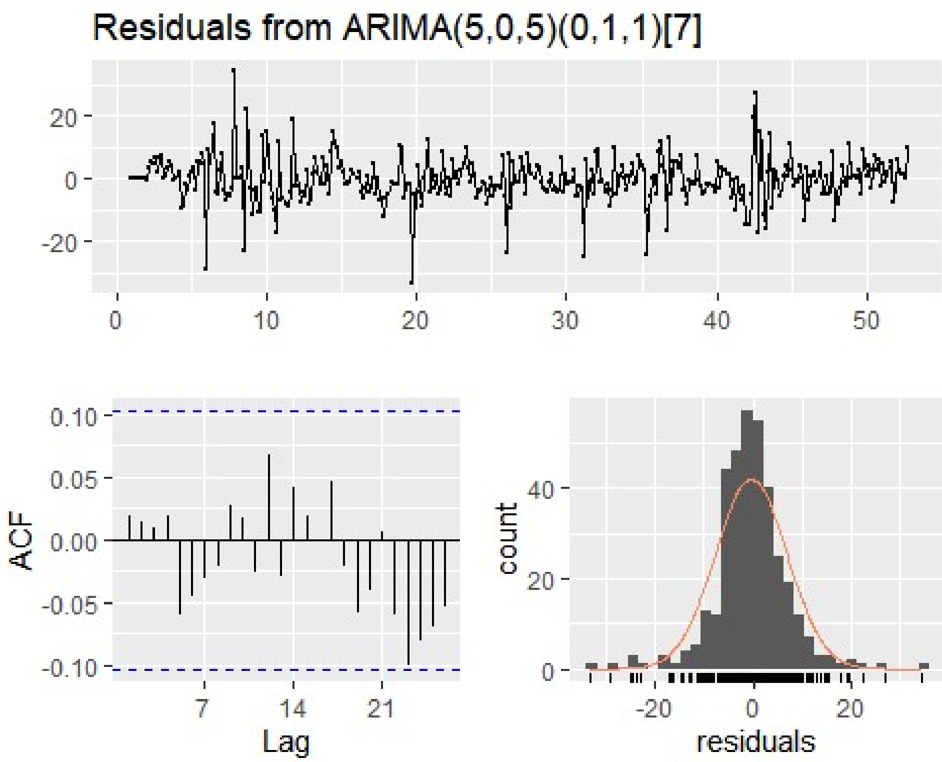

```
kpss.test(resid(atm2_arima_fit))
```

```
## Warning in kpss.test(resid(atm2_arima_fit)): p-value greater than printed
p-
## value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  resid(atm2_arima_fit)
## KPSS Level = 0.078093, Truncation lag parameter = 5, p-value = 0.1
```

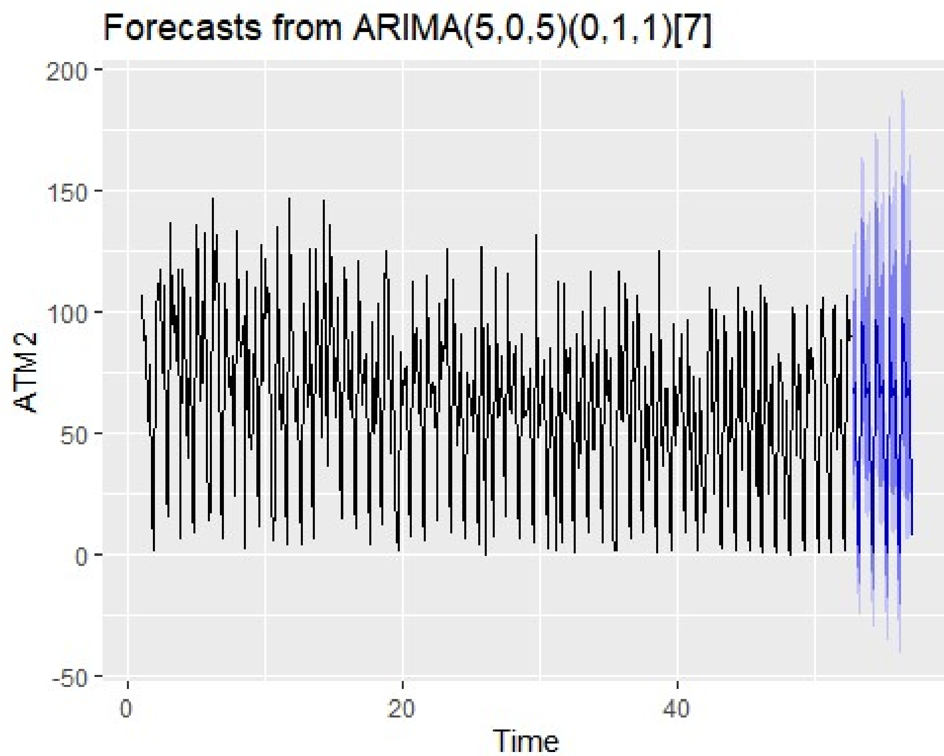
ARIMA with Box Cox Adjustment

```
atm2_lambda = BoxCox.lambda(ATM2)
atm2_box_arima_fit <-  Arima(ATM2, order = c(5, 0, 5), seasonal = c(0, 1, 1),
lambda = atm2_lambda)
checkresiduals(atm2_box_arima_fit)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(5,0,5)(0,1,1)[7]
## Q* = 6.2679, df = 3, p-value = 0.09928
##
## Model df: 11.   Total lags used: 14
```

```
atm2_box_arima_fit%>% forecast(h=31) %>% autoplot()
```




```
kpss.test(resid(atm2_box_arima_fit))

## Warning in kpss.test(resid(atm2_box_arima_fit)): p-value greater than
printed p-
## value

##
##  KPSS Test for Level Stationarity
##
## data:  resid(atm2_box_arima_fit)
## KPSS Level = 0.11532, Truncation lag parameter = 5, p-value = 0.1
```

MODEL EVALUATION

I will use the `tsCV` function and evaluate the models.My goal is to find the model that produces minumum RMSE.

```
h <- 31
get_rmse <- function(error) {
  sqrt(mean(error^2, na.rm = TRUE))
}
atm2_arima_forecast <- function(x, h) {
  forecast(Arima(x, order = c(2, 0, 2), seasonal = c(0, 1, 2)), h = h)
}
atm2_arima_box_forecast <- function(x, h) {
  forecast(Arima(x, order = c(5, 0, 5), seasonal = c(0, 1, 1),lambda =
atm2_lambda), h = h)
```

```
}

residuals_stl <- tsCV(ATM2, stlf, h = h, s.window = 7, robust = TRUE)
residuals_stl_arima <- tsCV(ATM2, stlf, h = h, s.window = 7, robust = TRUE,
method = "arima")
residuals_hw <- tsCV(ATM2, hw, h = h)
residuals_arima <- tsCV(ATM2, atm2_arima_forecast, h = h)
residuals_arima_box <- tsCV(ATM2, atm2_arima_box_forecast, h = h)
data.frame(Model_Name = c("STL", "STL & ARIMA", "Holt-Winters",
"ARIMA","ARIMA-BOX_COX"),
           RMSE = c(get_rmse(residuals_stl[, h]),
get_rmse(residuals_stl_arima[, h]), get_rmse(residuals_hw[, h]),
get_rmse(residuals_arima[, h]),

get_rmse(residuals_arima_box[, h]))) %>%
  arrange(RMSE) %>%
  kable() %>%
  kable_styling()
```

Model_Name	RMSE
ARIMA	39.69270
ARIMA-BOX_COX	41.24307
STL & ARIMA	44.05280
STL	44.59259
Holt-Winters	58.72761

The ARIMA(2, 0, 2)(0, 1, 2) model gives the minumum RMSE among the other models.The residuals appear to be approximately normally distributed with a mean around zero.

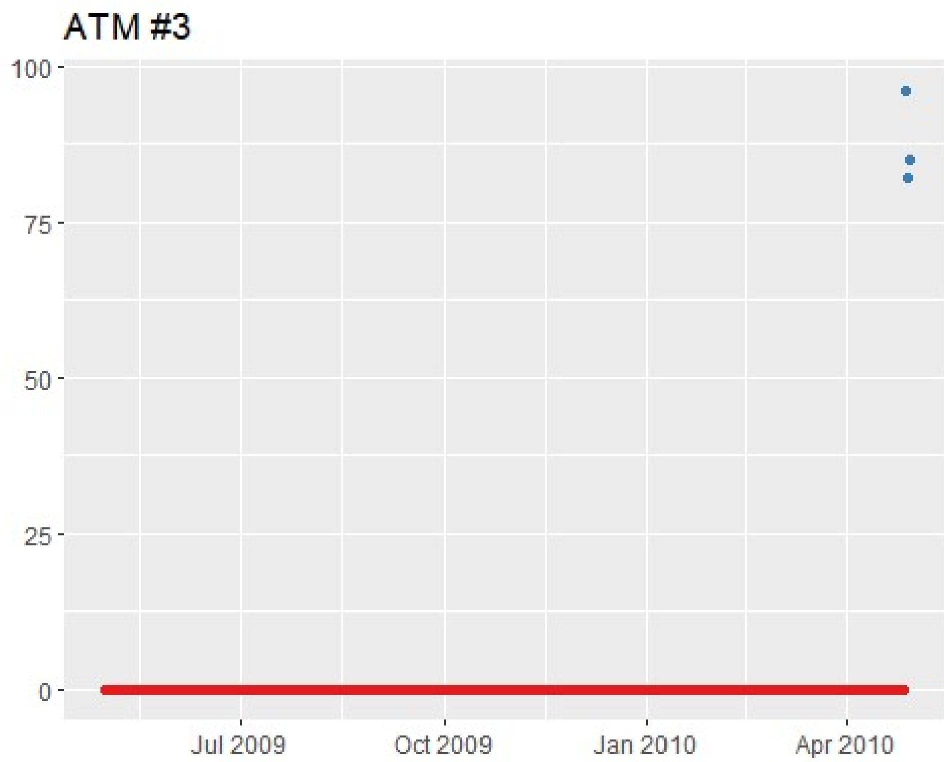
ATM #3

Data Cleanup

The ATM#3 data is quite challenging data as seen below since most of values are zero.

```
atm_data %>%
  filter(ATM == "ATM3") %>%
  mutate(nonzero = if_else(Cash == 0, "No", "Yes")) %>%
  ggplot(aes(DATE, Cash, color = nonzero)) +
  geom_point() +
  ggtitle("ATM #3") +
```

```
  scale_color_brewer(palette = "Set1") +
  theme(axis.title = element_blank(), legend.position = "none")
```



Model Creation

I will be using mean value for ATM3 data since we only have three observations.

Mean of Observations

```
atm3 <- atm_data %>%
  filter(ATM == "ATM3", Cash > 0)
atm3_mean <- mean(atm3$Cash)
```

The ARIMA model gives the minumum RMSE among the other models.

ATM #4

Data Cleanup

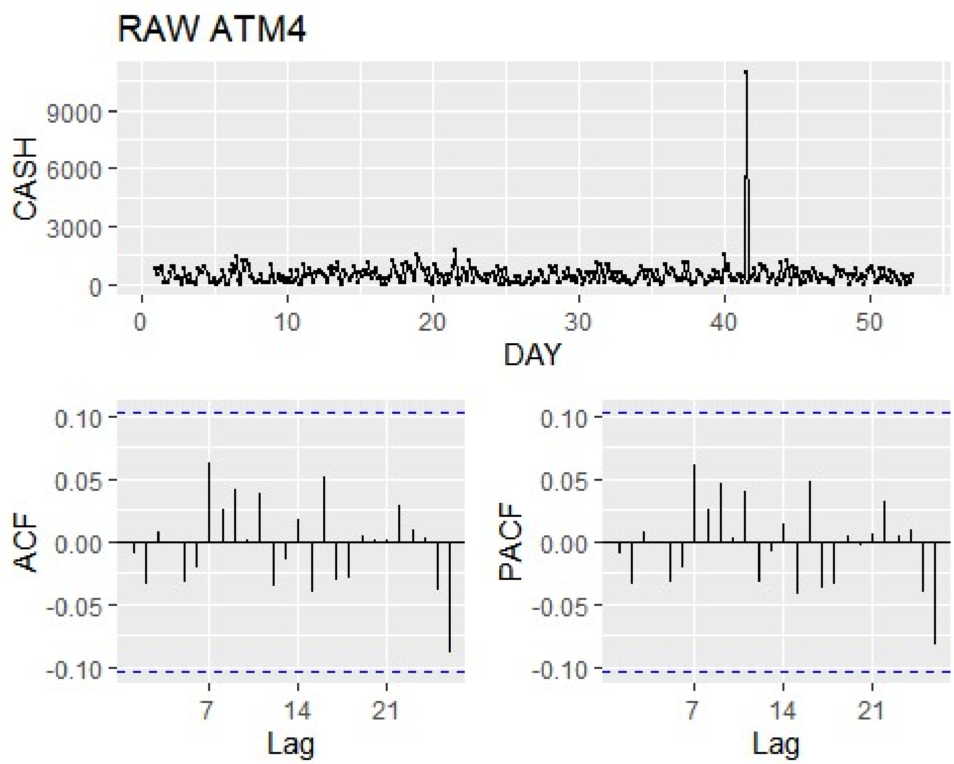
The ATM#4 also challenging, however it is not as bad as ATM #3 data.

```
atm4 <- atm_data %>%
  filter(ATM == "ATM4")
```

Here, I'd like to review the "atm4" timeseries data to determine whether there is a seosanality and ACF and PACF plots.

```
ATM4 <- atm_data[atm_data$ATM == "ATM4",]
ATM4 <- ATM4[complete.cases(ATM4),]
```

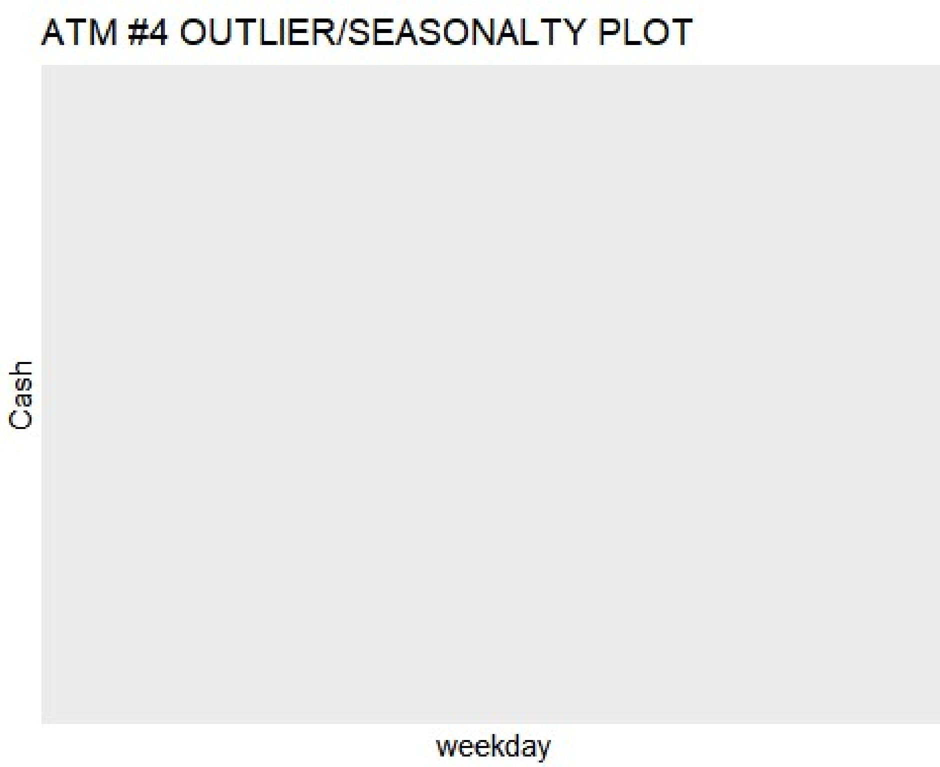
```r
ATM4 <- ts(ATM4[c("Cash")],frequency = 7)
ggtsdisplay(ATM4,
            main = "RAW ATM4",
            xlab = "DAY",
            ylab = "CASH")
```



I will check Seasonality and Outlier for atm4 data.

Outlier and Seasonality

```r
atm4$weekday <- factor(weekdays(as.Date(atm4$DATE)))
atm4$weekday <- ordered(atm4$weekday,levels =
c("Mon","Tues","Wedn","Thurs","Fri","Satur","Sun"))
#drop NaN values
atm4 <- atm4[complete.cases(atm4),]
ggplot(atm4[complete.cases(atm4),],aes(x=weekday,y=Cash,color=ATM))+
  geom_boxplot()+
  ggtitle("ATM #4 OUTLIER/SEASONALTY PLOT")
```



Box plots of the amount of cash is taken by customer in each of days from ATM2.As we can see on plot above, the Friday has higher mean amount of cash is taken than rest of days.The thursday has the minumum mean of cash amount is taken.The Tuesdaya has one extreme value that greater than any day’s mean value.

Model Creation

I will use following forecating models on this time series and determine which one is better by estimatin error metric RMSE. I will use time series cross validation function to estimate RMSE for timeseries data.

- Seasonal and Trend decomposition (STL)
- Seasonal and Trend decomposition (STL) with ARIMA
- Holt-Winters
- Holt-Winters with Box Cox Adjustment
- ARIMA
- ARIMA with Box Cox Adjustment

```
ATM4 <- atm_data[atm_data$ATM == "ATM4",]
ATM4 <- ATM4[complete.cases(ATM4),]
ATM4 <- ts(ATM4[c("Cash")],frequency = 7)
Box.test(diff(ATM4,lag=7), type = "Ljung-Box")

##
##  Box-Ljung test
##
```

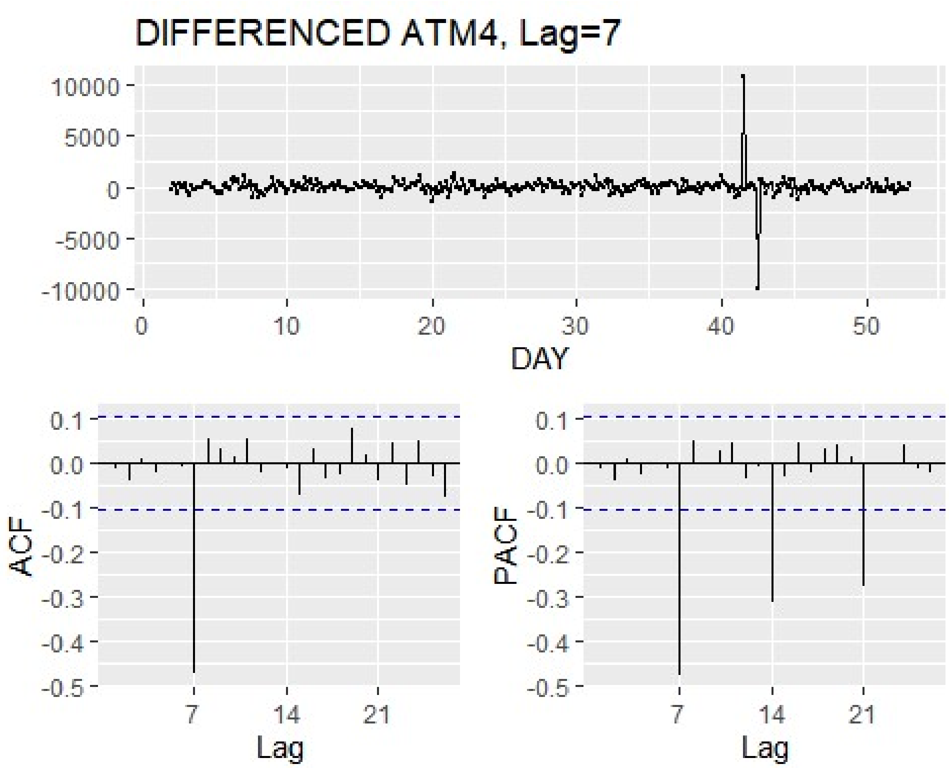
```
## data:  diff(ATM4, lag = 7)
## X-squared = 0.081947, df = 1, p-value = 0.7747
```

```
kpss.test(diff(ATM4,lag=7))
```

```
## Warning in kpss.test(diff(ATM4, lag = 7)): p-value greater than printed p-
value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  diff(ATM4, lag = 7)
## KPSS Level = 0.014025, Truncation lag parameter = 5, p-value = 0.1
```

```
ggtsdisplay(diff(ATM4,lag=7),
            main = "DIFFERENCED ATM4, Lag=7",
            xlab = "DAY",
            ylab = "")
```

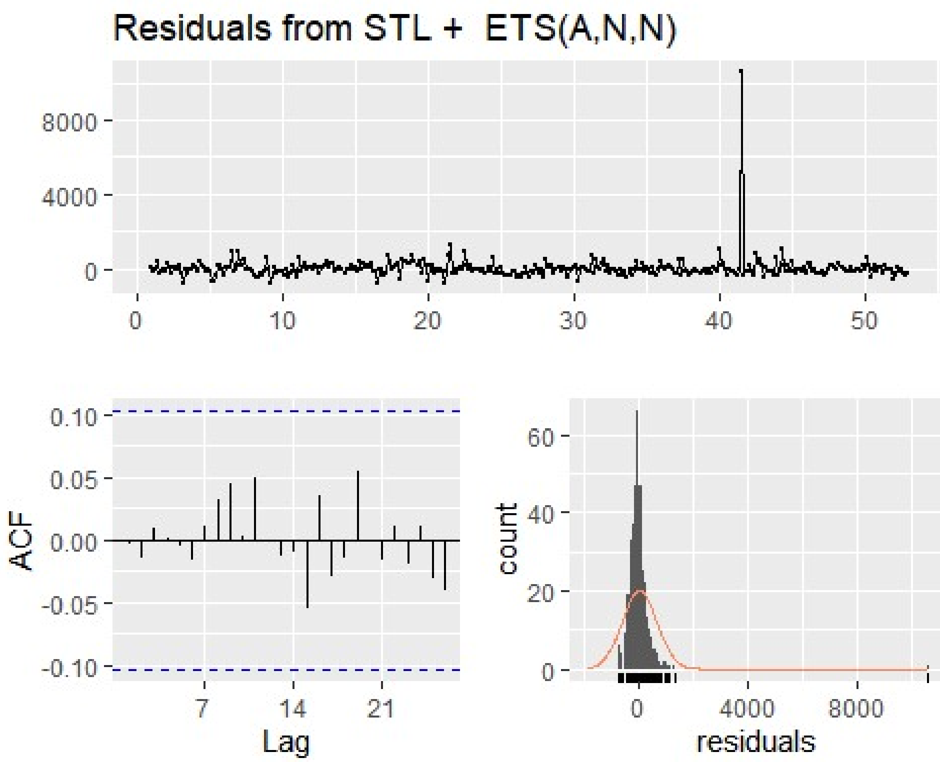


The result of KPSS and Box-Cox test indicates that atm1 timeseries is statioanry eventhou we see some extreme values on lag=7.

Seasonal and Trend decomposition (STL)

```
atm4_stl_fit <- ATM4 %>%
  stlf(h = 31, s.window = 7, robust = TRUE)
checkresiduals(atm4_stl_fit)
```

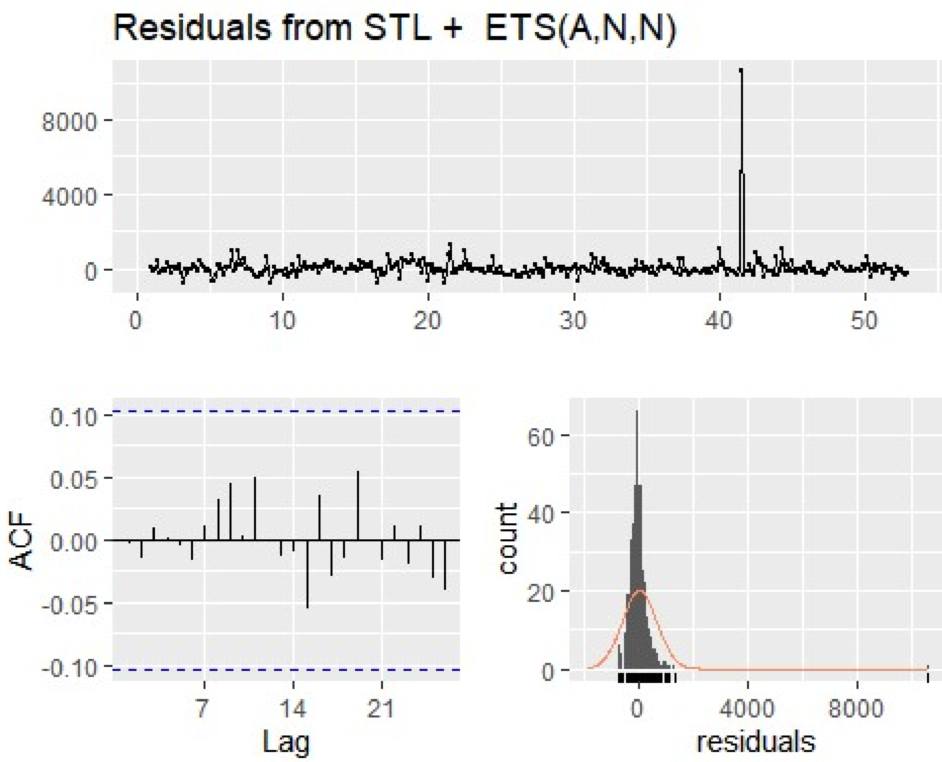
```
## Warning in checkresiduals(atm4_stl_fit): The fitted degrees of freedom is
based
## on the model used for the seasonally adjusted data.
```




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 2.4559, df = 12, p-value = 0.9983
##
## Model df: 2.   Total lags used: 14
```

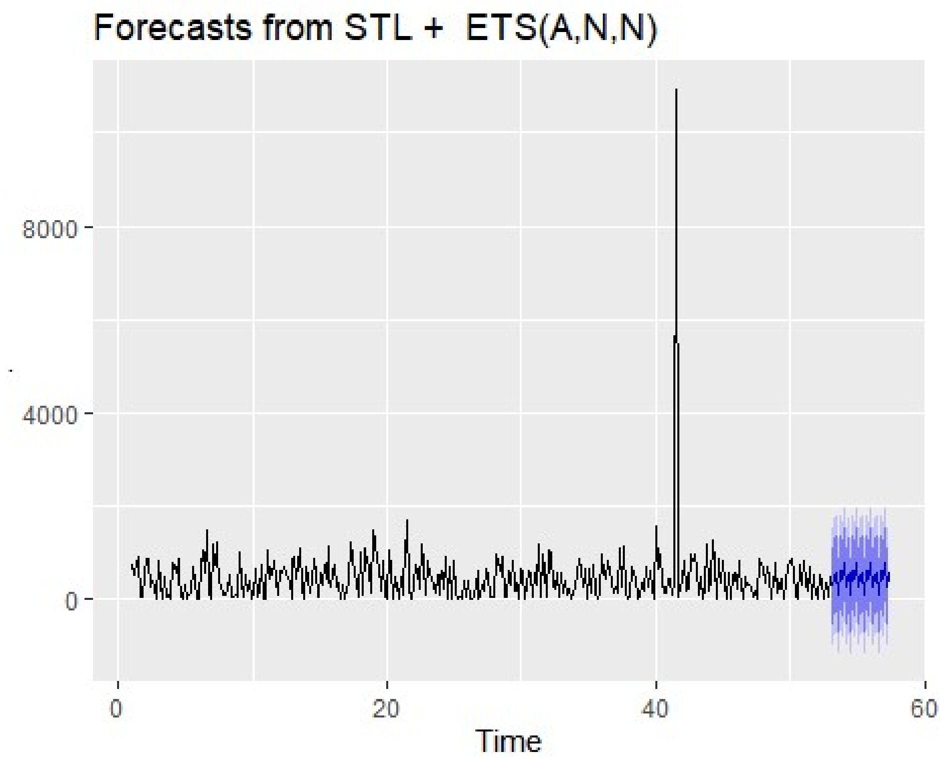
```
checkresiduals(atm4_stl_fit)
```

```
## Warning in checkresiduals(atm4_stl_fit): The fitted degrees of freedom is
based
## on the model used for the seasonally adjusted data.
```



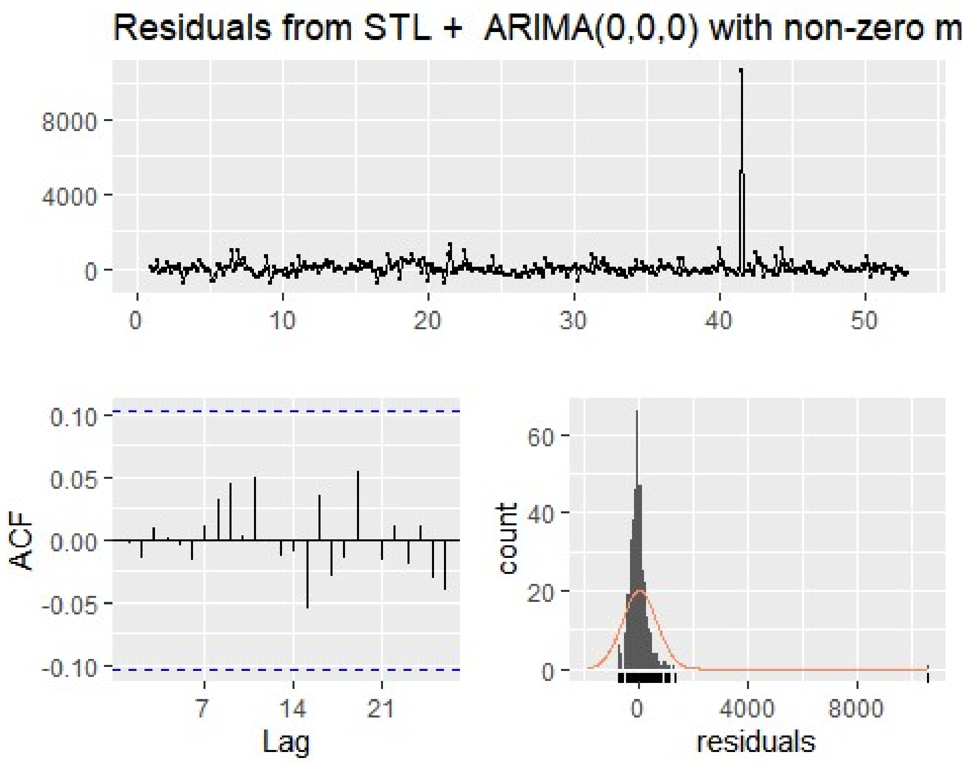

```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ETS(A,N,N)
## Q* = 2.4559, df = 12, p-value = 0.9983
##
## Model df: 2.   Total lags used: 14
```

```
atm4_stl_fit%>% forecast(h=31) %>% autoplot()
```



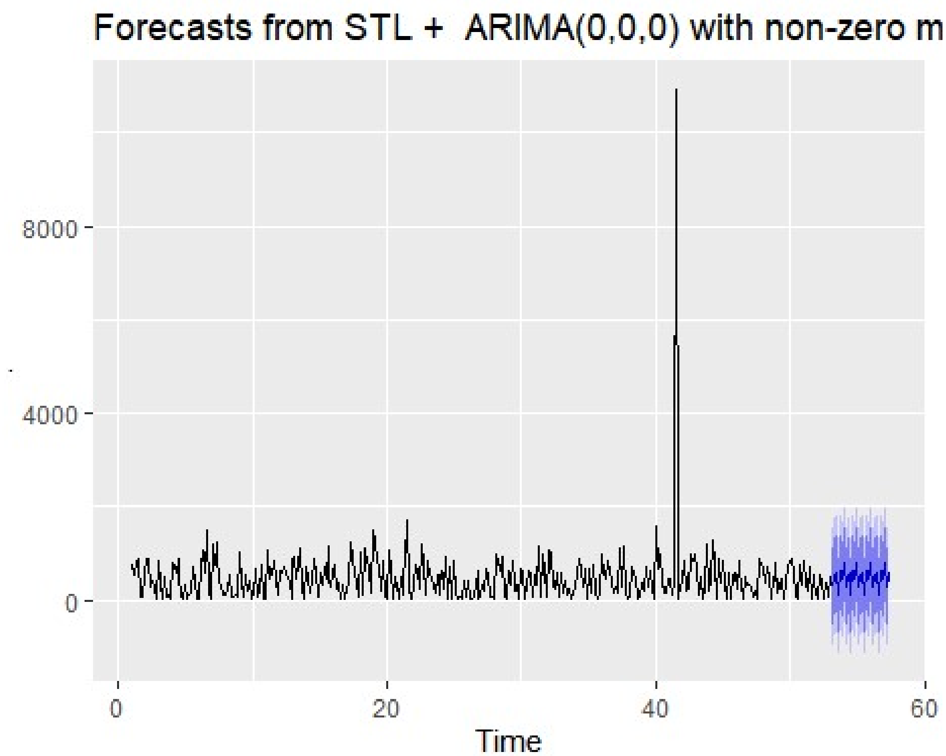
Seasonal and Trend decomposition (STL) with ARIMA

```
atm4_stl_arima_fit <- ATM4 %>%
  stlf(h = 31, s.window = 7, robust = TRUE, method = "arima")
checkresiduals(atm4_stl_arima_fit)

## Warning in checkresiduals(atm4_stl_arima_fit): The fitted degrees of
freedom is
## based on the model used for the seasonally adjusted data.
```



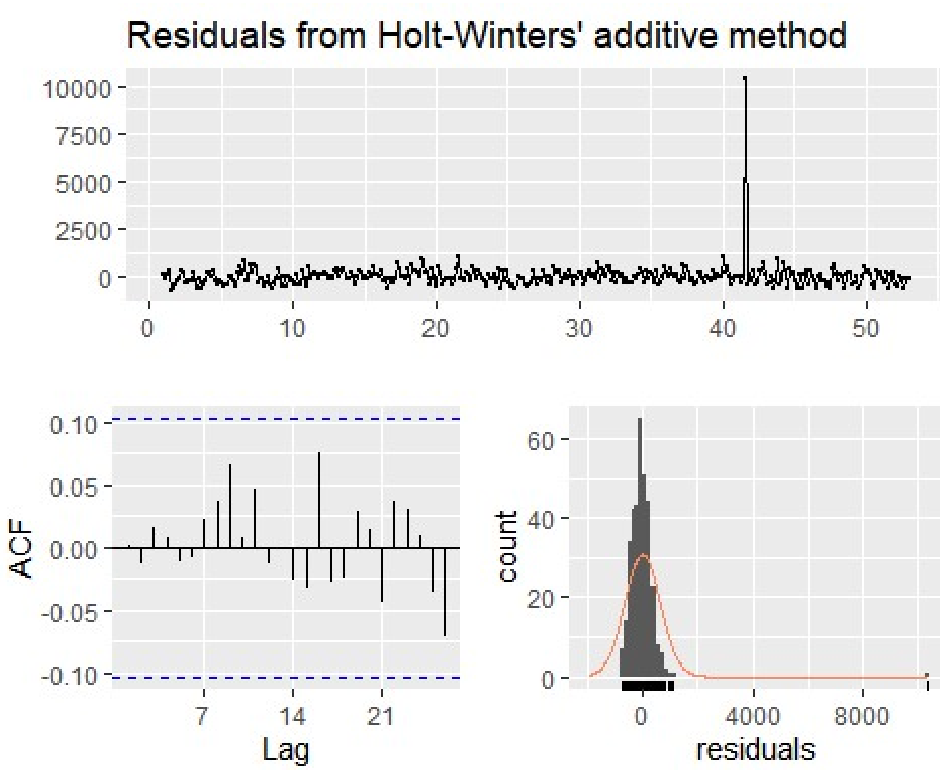

```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  ARIMA(0,0,0) with non-zero mean
## Q* = 2.4561, df = 13, p-value = 0.9993
##
## Model df: 1.   Total lags used: 14
```

```
atm4_stl_arima_fit%>% forecast(h=31) %>% autoplot()
```



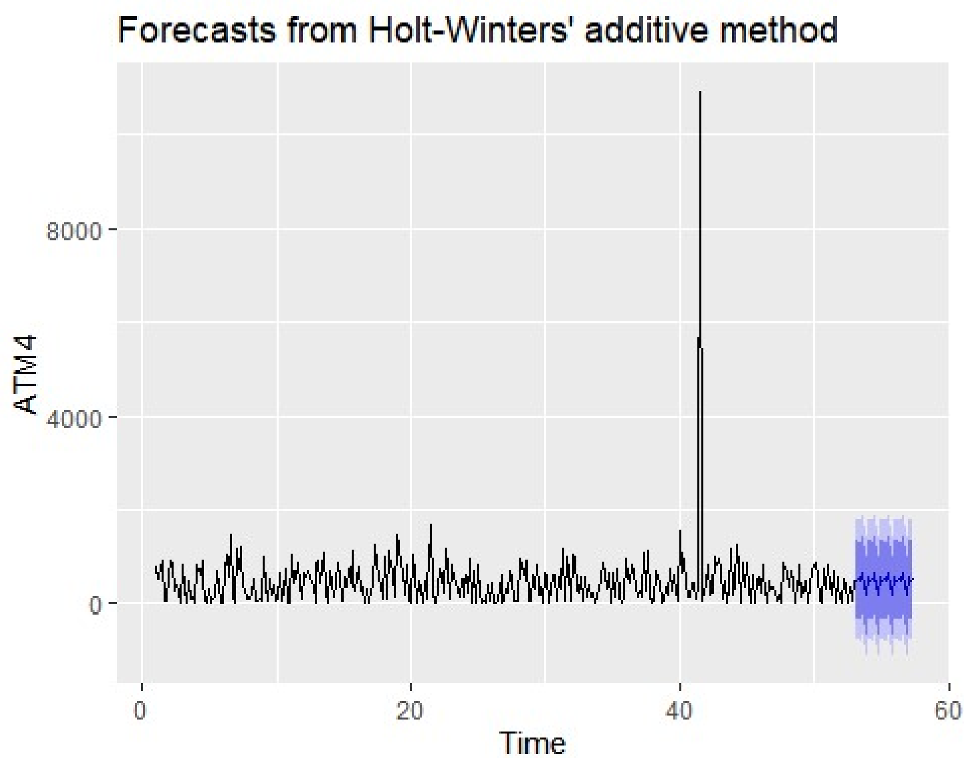
Holt-Winters

```
atm4_hw_fit <- hw(ATM4, h = 31)
checkresiduals(atm4_hw_fit)
```



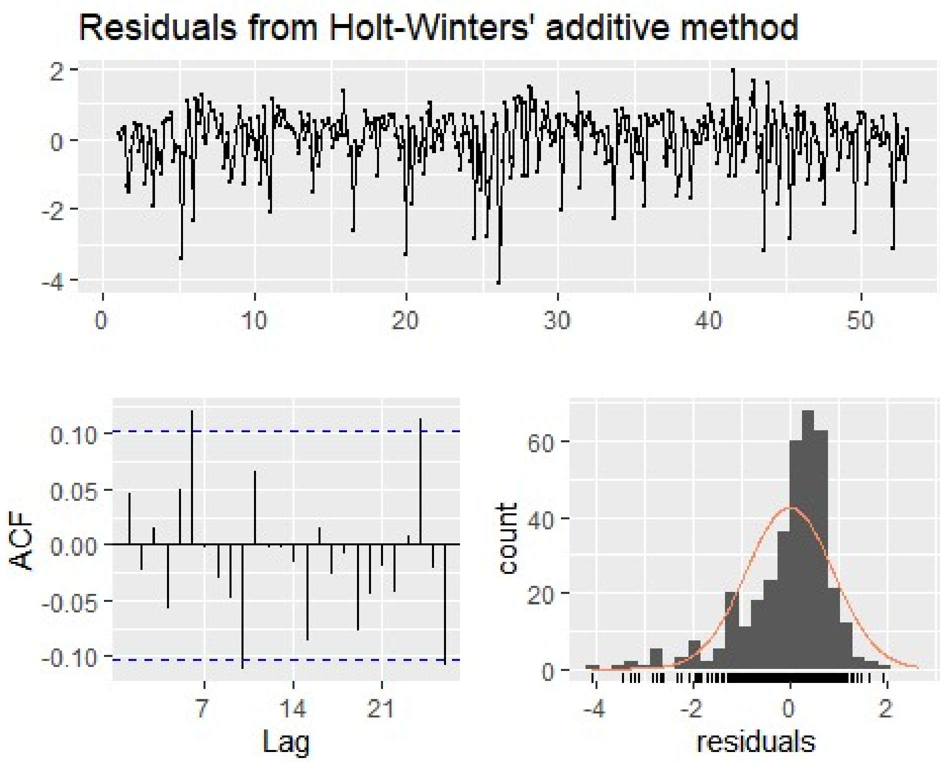
```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 3.7364, df = 3, p-value = 0.2914
##
## Model df: 11.   Total lags used: 14
```

```
atm4_hw_fit%>% forecast(h=31) %>% autoplot()
```



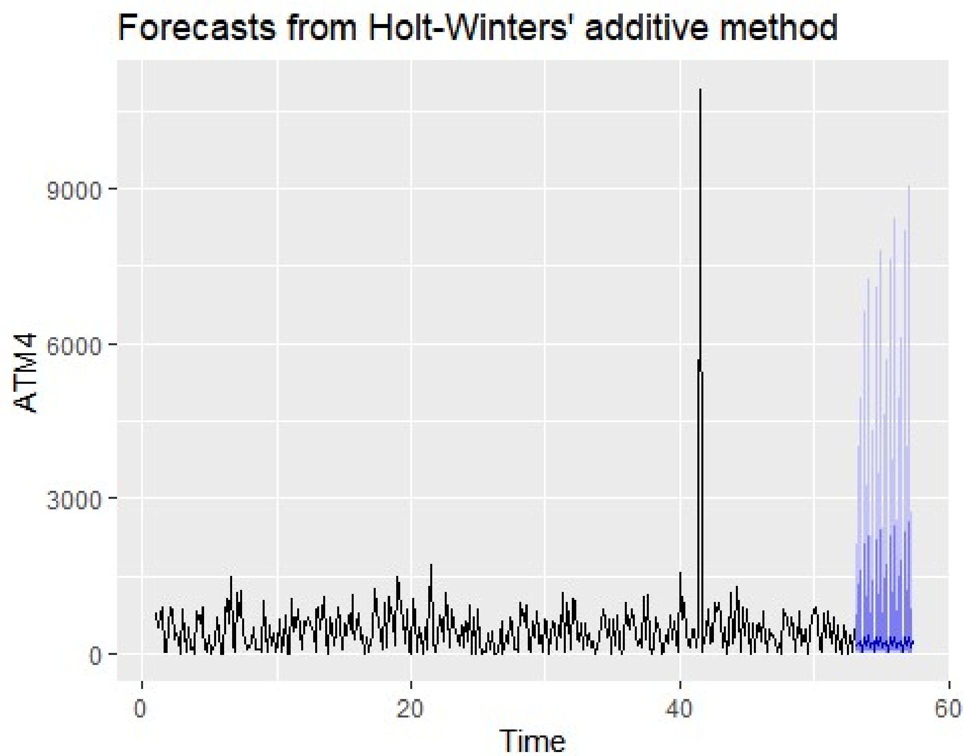
Holt-Winters with Box Cox Adjustment

```
atm4_lambda <- BoxCox.lambda(ATM4)
atm4_adj_hw_fit <- hw(ATM4, h = 31, lambda = atm4_lambda)
checkresiduals(atm4_adj_hw_fit)
```



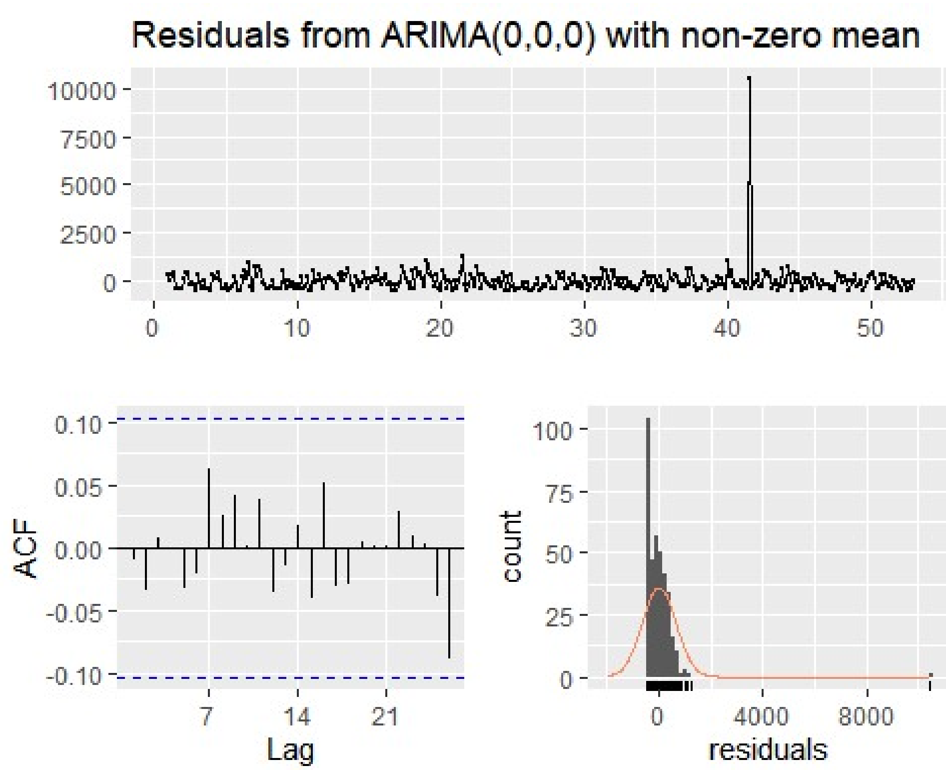

```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 16.33, df = 3, p-value = 0.0009704
##
## Model df: 11.   Total lags used: 14
```

```
atm4_adj_hw_fit%>% forecast(h=31) %>% autoplot()
```



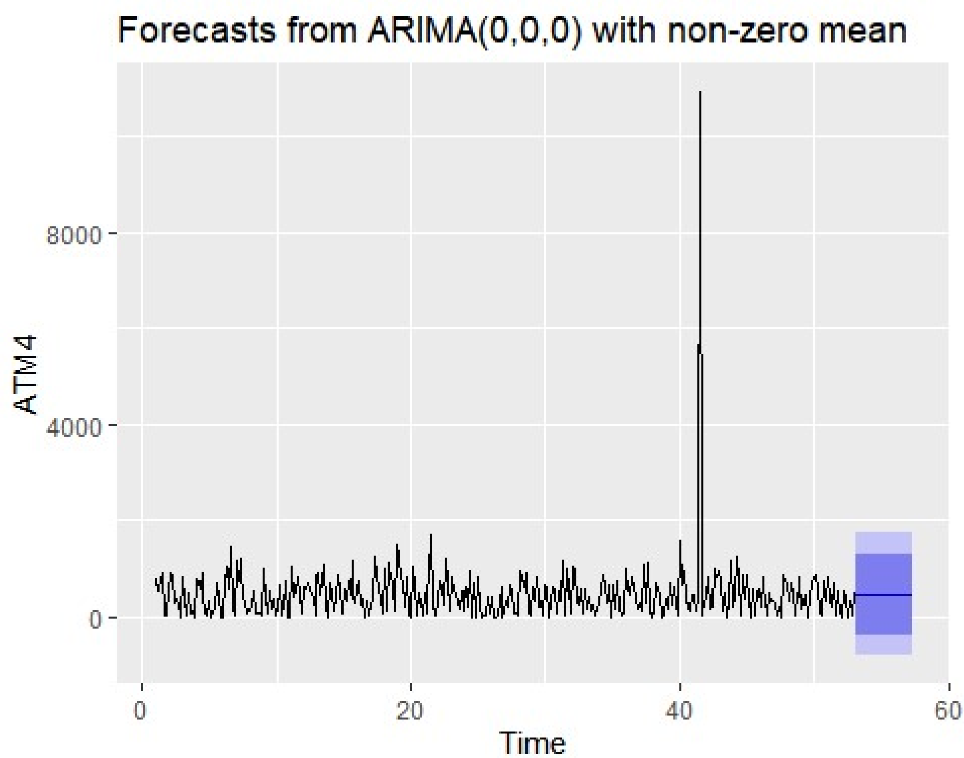
ARIMA

```
atm4_arima_fit <- auto.arima(ATM4)
checkresiduals(atm4_arima_fit)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,0,0) with non-zero mean
## Q* = 4.6668, df = 13, p-value = 0.9818
##
## Model df: 1.   Total lags used: 14
```

```
atm4_arima_fit%>% forecast(h=31) %>% autoplot()
```



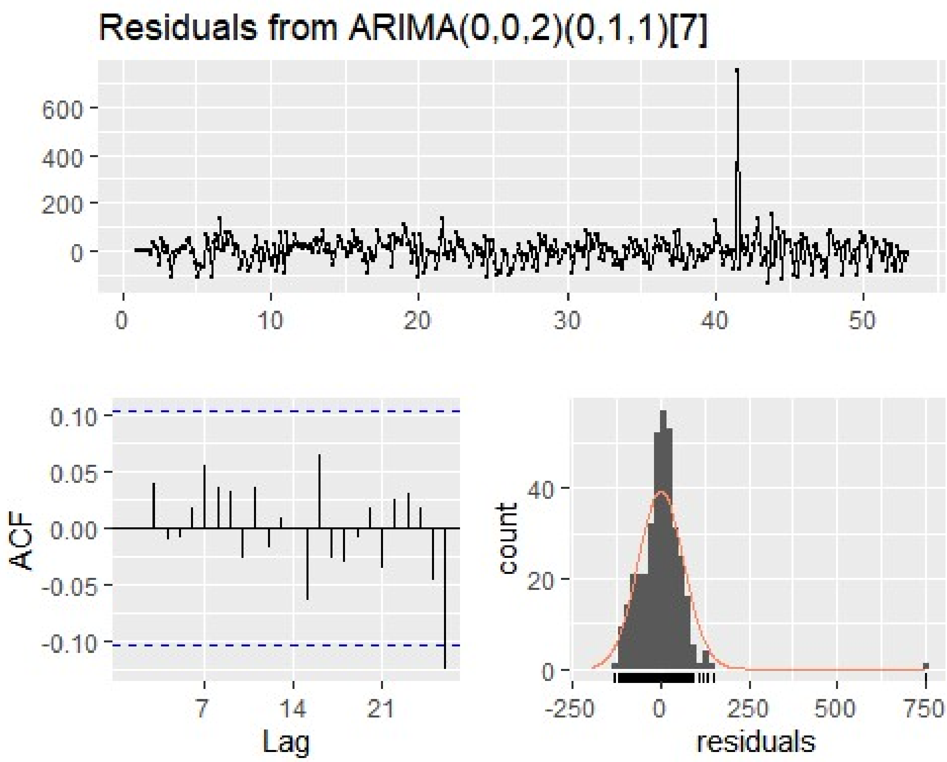

```
kpss.test(resid(atm4_arima_fit))
```

```
## Warning in kpss.test(resid(atm4_arima_fit)): p-value greater than printed
p-
## value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  resid(atm4_arima_fit)
## KPSS Level = 0.079654, Truncation lag parameter = 5, p-value = 0.1
```

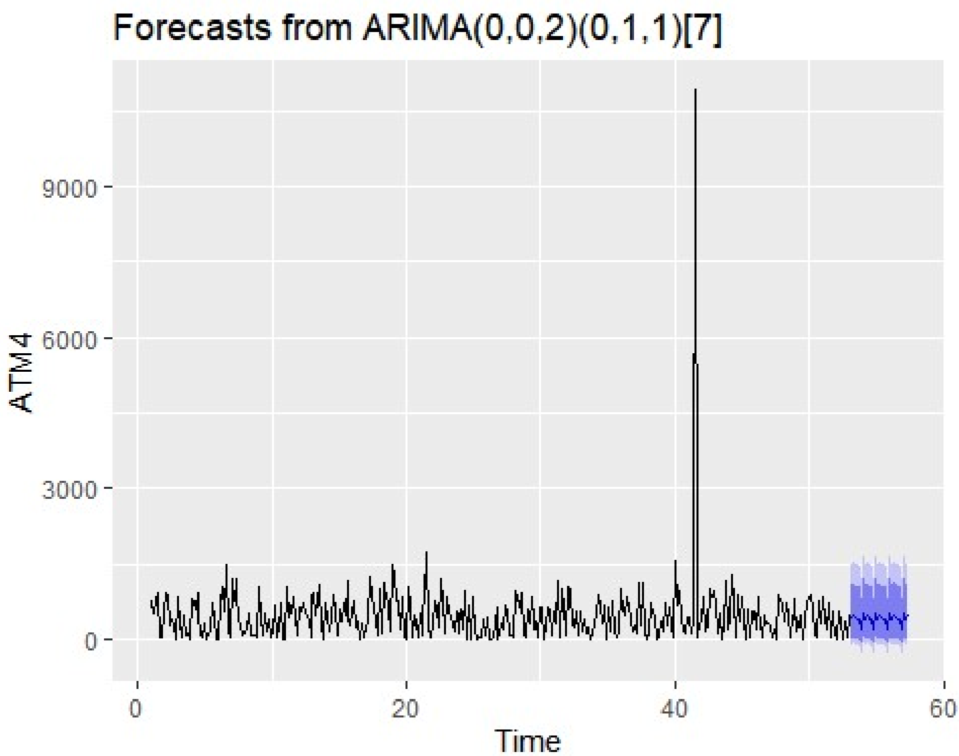
ARIMA with Box Cox Adjustment

```
atm4_lambda = BoxCox.lambda(ATM4)
atm4_box_arima_fit <- Arima(ATM4, order = c(0, 0, 2), seasonal = c(0, 1, 1),
lambda = atm2_lambda)
checkresiduals(atm4_box_arima_fit)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,0,2)(0,1,1)[7]
## Q* = 3.6778, df = 11, p-value = 0.9784
##
## Model df: 3.   Total lags used: 14
```

```
atm4_box_arima_fit%>% forecast(h=31) %>% autoplot()
```




```
kpss.test(resid(atm4_box_arima_fit))

## Warning in kpss.test(resid(atm4_box_arima_fit)): p-value greater than
printed p-
## value

##
##  KPSS Test for Level Stationarity
##
## data:  resid(atm4_box_arima_fit)
## KPSS Level = 0.074774, Truncation lag parameter = 5, p-value = 0.1
```

MODEL EVALUATION

I will use the `tsCV` function and evaluate the models.My goal is to find the model that produces minumum RMSE.

```
h <- 31
get_rmse <- function(error) {
  sqrt(mean(error^2, na.rm = TRUE))
}
atm4_arima_forecast <- function(x, h) {
  forecast(Arima(x, order = c(0, 0, 0)), h = h)
}
atm4_arima_box_forecast <- function(x, h) {
  forecast(Arima(x, order = c(0, 0, 2), seasonal = c(0, 1, 1),lambda =
atm4_lambda), h = h)
```

```
}

residuals_stl <- tsCV(ATM4, stlf, h = h, s.window = 7, robust = TRUE)
residuals_stl_arima <- tsCV(ATM4, stlf, h = h, s.window = 7, robust = TRUE,
method = "arima")
residuals_hw <- tsCV(ATM4, hw, h = h)
residuals_arima <- tsCV(ATM4, atm4_arima_forecast, h = h)
residuals_arima_box <- tsCV(ATM4, atm4_arima_box_forecast, h = h)
data.frame(Model_Name = c("STL", "STL & ARIMA", "Holt-Winters",
"ARIMA","ARIMA-BOX_COX"),
           RMSE = c(get_rmse(residuals_stl[, h]),
get_rmse(residuals_stl_arima[, h]), get_rmse(residuals_hw[, h]),
get_rmse(residuals_arima[, h]),

get_rmse(residuals_arima_box[, h]))) %>%
  arrange(RMSE) %>%
  kable() %>%
  kable_styling()
```

Model_Name

RMSE

ARIMA

675.1356

STL & ARIMA

689.7695

ARIMA-BOX_COX

696.7529

STL

697.5947

Holt-Winters

777.7074

The ARIMA(0, 0, 2)(0, 1, 1) model gives the minumum RMSE among the other models withresiduals approximately normally distributed with a mean around zero.

SUMMARY

```
atm1_forecast <- atm1_arima_fit %>% forecast(h=31)
atm2_forecast <- atm2_arima_fit %>% forecast(h=31)
atm3_forecast <- rep(atm3_mean, h=31)
atm4_forecast <- atm4_box_arima_fit %>% forecast(h=31)
atm_forecasts_df <- data.frame("DATE" = seq(ymd("2010-05-01"), ymd("2010-05-
31"), by = "1 day"), "ATM" = c("ATM1"), "Cash" = c(atm1_forecast$mean))
atm_forecasts_df <- data.frame("DATE" = seq(ymd("2010-05-01"), ymd("2010-05-
31"), by = "1 day"), "ATM" = c("ATM2"), "Cash" = c(atm2_forecast$mean)) %>%
  rbind(atm_forecasts_df, .)
atm_forecasts_df <- data.frame("DATE" = seq(ymd("2010-05-01"), ymd("2010-05-
31"), by = "1 day"), "ATM" = c("ATM3"), "Cash" = atm3_forecast) %>%
```

```
  rbind(atm_forecasts_df, .)
atm_forecasts_df <- data.frame("DATE" = seq(ymd("2010-05-01"), ymd("2010-05-
31"), by = "1 day"), "ATM" = c("ATM4"), "Cash" = c(atm4_forecast$mean)) %>%
  rbind(atm_forecasts_df, .)
atm_forecasts_df %>%
  kable() %>%
  kable_styling()
```

DATE

ATM

Cash

2010-05-01

ATM1

86.805607

2010-05-02

ATM1

100.640560

2010-05-03

ATM1

74.714560

2010-05-04

ATM1

4.762634

2010-05-05

ATM1

100.063192

2010-05-06

ATM1

79.356704

2010-05-07

ATM1

85.410706

2010-05-08

ATM1

86.967155

2010-05-09

ATM1

100.640560

2010-05-10

ATM1

74.714560

2010-05-11

ATM1
4.762634
2010-05-12
ATM1
100.063192
2010-05-13
ATM1
79.356704
2010-05-14
ATM1
85.410706
2010-05-15
ATM1
86.967155
2010-05-16
ATM1
100.640560
2010-05-17
ATM1
74.714560
2010-05-18
ATM1
4.762634
2010-05-19
ATM1
100.063192
2010-05-20
ATM1
79.356704
2010-05-21
ATM1
85.410706
2010-05-22
ATM1
86.967155
2010-05-23
ATM1
100.640560
2010-05-24
ATM1

74.714560
2010-05-25
ATM1
4.762634
2010-05-26
ATM1
100.063192
2010-05-27
ATM1
79.356704
2010-05-28
ATM1
85.410706
2010-05-29
ATM1
86.967155
2010-05-30
ATM1
100.640560
2010-05-31
ATM1
74.714560
2010-05-01
ATM2
69.432448
2010-05-02
ATM2
69.881366
2010-05-03
ATM2
12.308924
2010-05-04
ATM2
2.758906
2010-05-05
ATM2
99.519624
2010-05-06
ATM2
93.446892

2010-05-07
ATM2
68.207442
2010-05-08
ATM2
68.692335
2010-05-09
ATM2
71.724476
2010-05-10
ATM2
12.261298
2010-05-11
ATM2
2.746415
2010-05-12
ATM2
100.273631
2010-05-13
ATM2
93.053089
2010-05-14
ATM2
70.543642
2010-05-15
ATM2
68.678272
2010-05-16
ATM2
71.633493
2010-05-17
ATM2
12.311587
2010-05-18
ATM2
2.806723
2010-05-19
ATM2
100.203746
2010-05-20

ATM2
93.028275
2010-05-21
ATM2
70.616283
2010-05-22
ATM2
68.670276
2010-05-23
ATM2
71.572003
2010-05-24
ATM2
12.344230
2010-05-25
ATM2
2.848040
2010-05-26
ATM2
100.157484
2010-05-27
ATM2
93.010606
2010-05-28
ATM2
70.664886
2010-05-29
ATM2
68.665877
2010-05-30
ATM2
71.530466
2010-05-31
ATM2
12.365385
2010-05-01
ATM3
87.666667
2010-05-02
ATM3

87.666667
2010-05-03
ATM3
87.666667
2010-05-04
ATM3
87.666667
2010-05-05
ATM3
87.666667
2010-05-06
ATM3
87.666667
2010-05-07
ATM3
87.666667
2010-05-08
ATM3
87.666667
2010-05-09
ATM3
87.666667
2010-05-10
ATM3
87.666667
2010-05-11
ATM3
87.666667
2010-05-12
ATM3
87.666667
2010-05-13
ATM3
87.666667
2010-05-14
ATM3
87.666667
2010-05-15
ATM3
87.666667

2010-05-16
ATM3
87.666667
2010-05-17
ATM3
87.666667
2010-05-18
ATM3
87.666667
2010-05-19
ATM3
87.666667
2010-05-20
ATM3
87.666667
2010-05-21
ATM3
87.666667
2010-05-22
ATM3
87.666667
2010-05-23
ATM3
87.666667
2010-05-24
ATM3
87.666667
2010-05-25
ATM3
87.666667
2010-05-26
ATM3
87.666667
2010-05-27
ATM3
87.666667
2010-05-28
ATM3
87.666667
2010-05-29

ATM3
87.666667
2010-05-30
ATM3
87.666667
2010-05-31
ATM3
87.666667
2010-05-01
ATM4
416.325318
2010-05-02
ATM4
462.084491
2010-05-03
ATM4
433.476211
2010-05-04
ATM4
396.933789
2010-05-05
ATM4
415.667039
2010-05-06
ATM4
218.504549
2010-05-07
ATM4
532.792032
2010-05-08
ATM4
412.217409
2010-05-09
ATM4
461.255552
2010-05-10
ATM4
433.476211
2010-05-11
ATM4

396.933789
2010-05-12
ATM4
415.667039
2010-05-13
ATM4
218.504549
2010-05-14
ATM4
532.792032
2010-05-15
ATM4
412.217409
2010-05-16
ATM4
461.255552
2010-05-17
ATM4
433.476211
2010-05-18
ATM4
396.933789
2010-05-19
ATM4
415.667039
2010-05-20
ATM4
218.504549
2010-05-21
ATM4
532.792032
2010-05-22
ATM4
412.217409
2010-05-23
ATM4
461.255552
2010-05-24
ATM4
433.476211

2010-05-25
ATM4
396.933789
2010-05-26
ATM4
415.667039
2010-05-27
ATM4
218.504549
2010-05-28
ATM4
532.792032
2010-05-29
ATM4
412.217409
2010-05-30
ATM4
461.255552
2010-05-31
ATM4
433.476211

```
write.csv(atm_forecasts_df,"atm_forecasts_df.csv")
```

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.

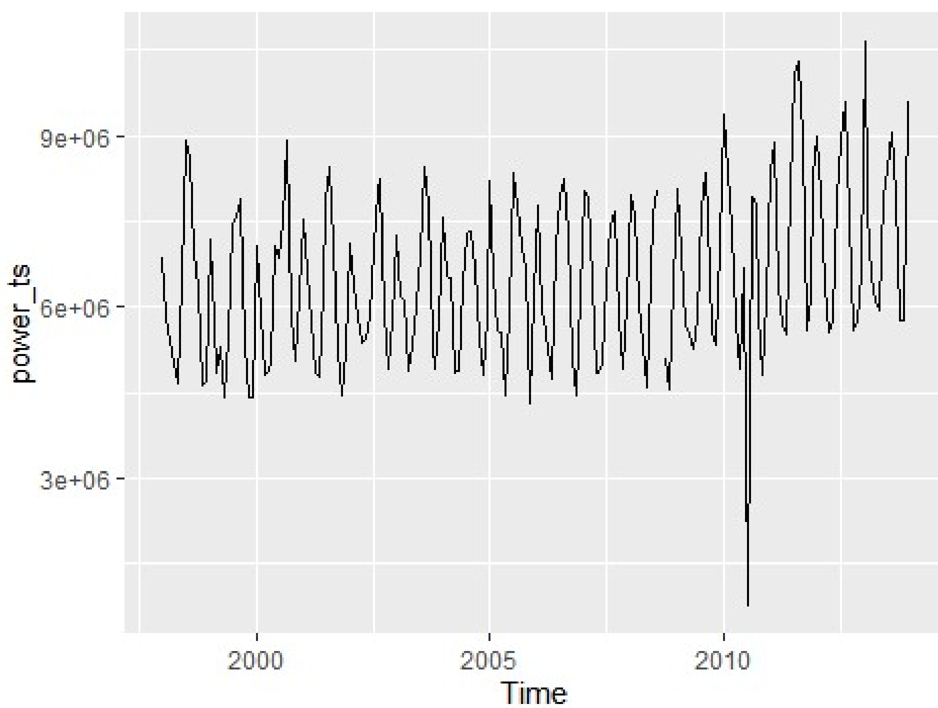
Data Review-Missing values

```
kable(tail(power_data))
```

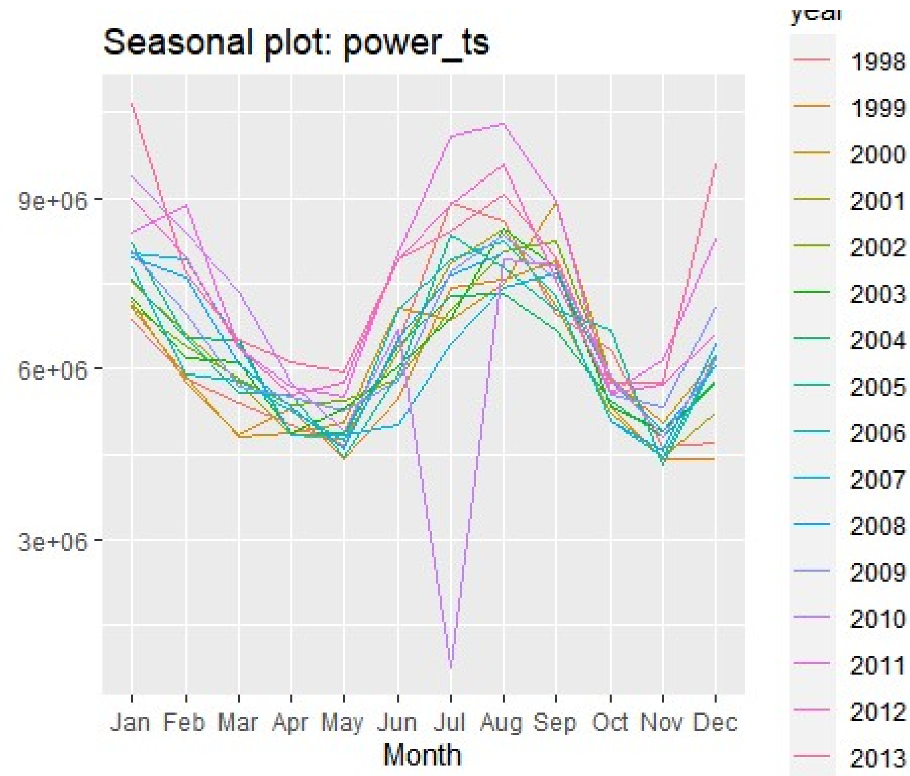
CaseSequence
YYYY-MMM
KWH
919
2013-Jul
8415321
920
2013-Aug
9080226
921

2013-Sep
7968220
922
2013-Oct
5759367
923
2013-Nov
5769083
924
2013-Dec
9606304

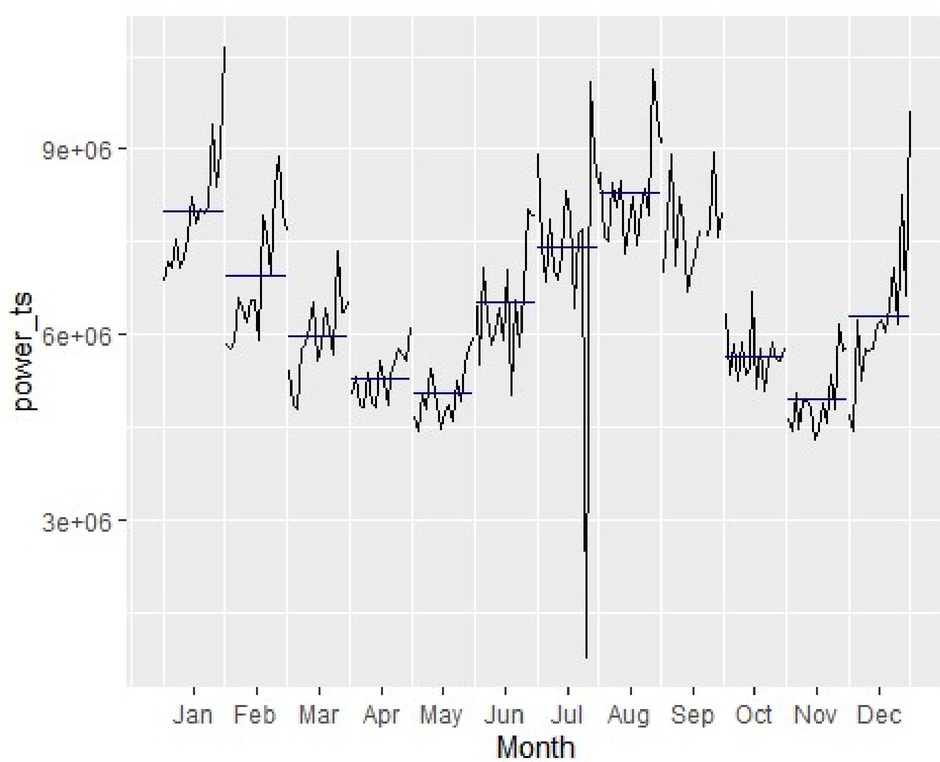
```
power_ts <-power_data %>%
  select(KWH) %>%
  ts(start = decimal_date(date("1998-01-01")), frequency = 12)
autoplot(power_ts)
```



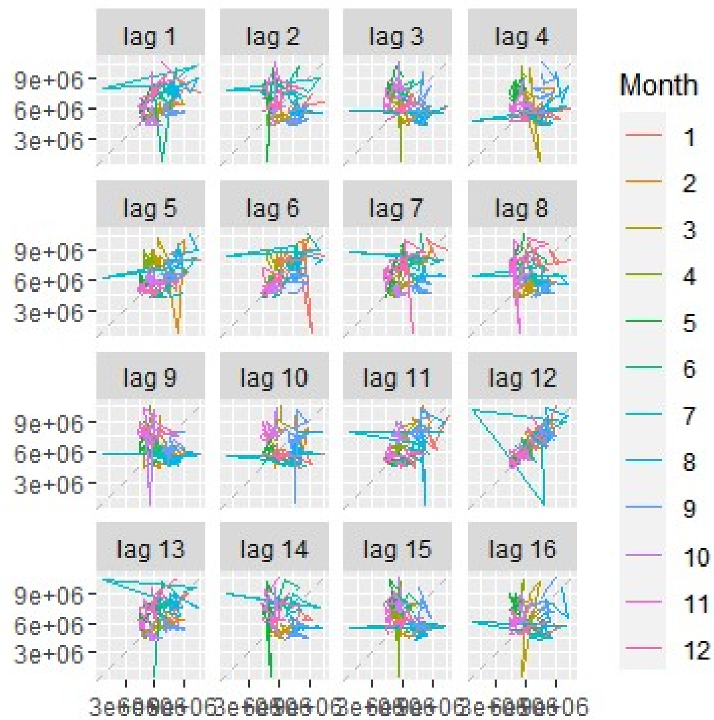

```
ggseasonplot(power_ts)
```



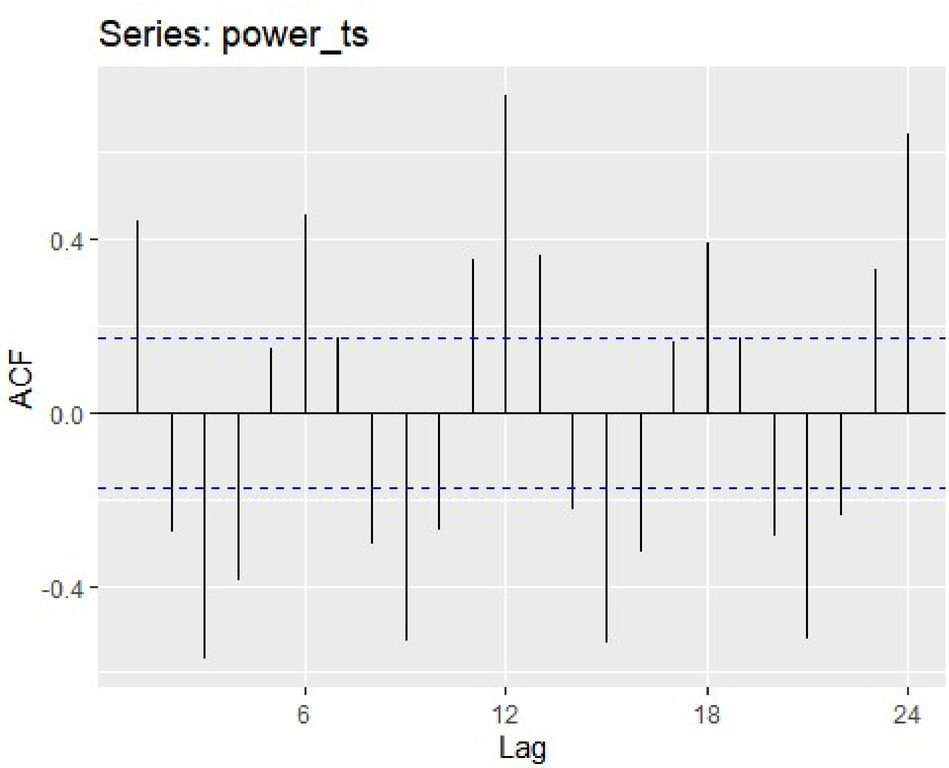

```r
ggsubseriesplot(power_ts)
## Warning: Removed 16 row(s) containing missing values (geom_path).
```



```r
gglagplot(power_ts)
```



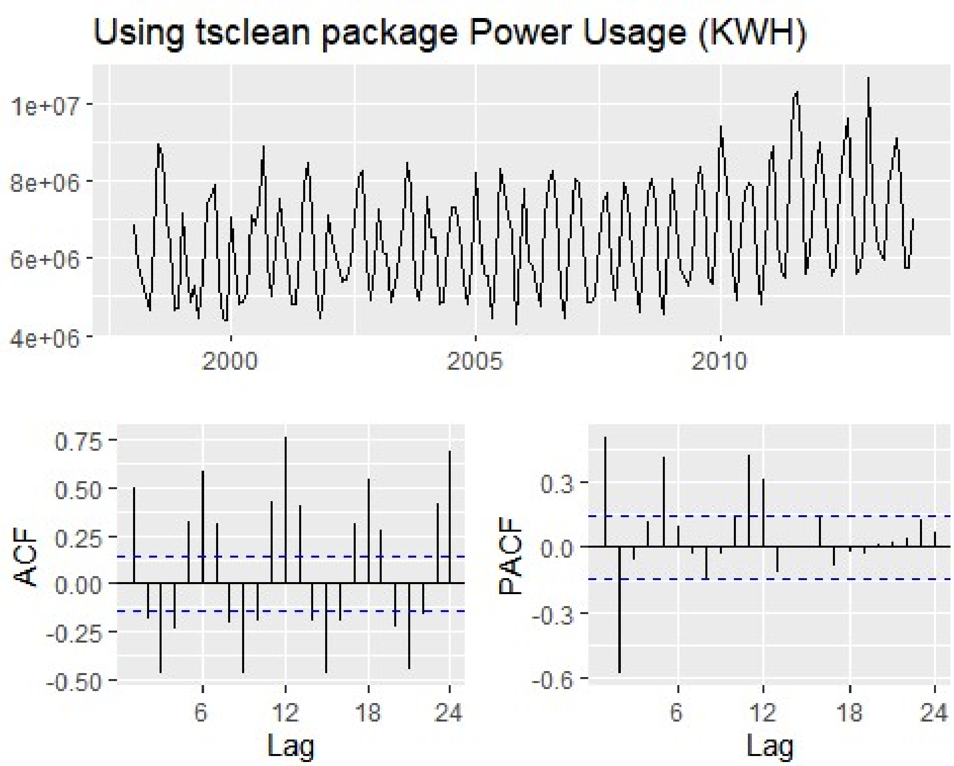

```r
ggAcf(power_ts)
```



The power_data contains 1 missing values on Sept 2008.The data also contains an extreme value in july 2010.ACF plot suggest that there is a sesonality in power_data.I possoble need to use lag=12 sesonality adjustment.

I will use `tsclean` package to clean the missing values and winsorize extreme value and replot the differents graphs.

```
power_ts <- tsclean(power_ts)
ggtsplot <- function(ts, title) {
  grid.arrange(
    autoplot(ts) +
      scale_y_continuous() +
      ggtitle(title) +
      theme(axis.title = element_blank()),
    grid.arrange(
      ggAcf(ts) + ggtitle(element_blank()),
      ggPacf(ts) + ggtitle(element_blank()), ncol = 2)
    , nrow = 2)
}
ggtsplot(power_ts, "Using tsclean package Power Usage (KWH)")
```



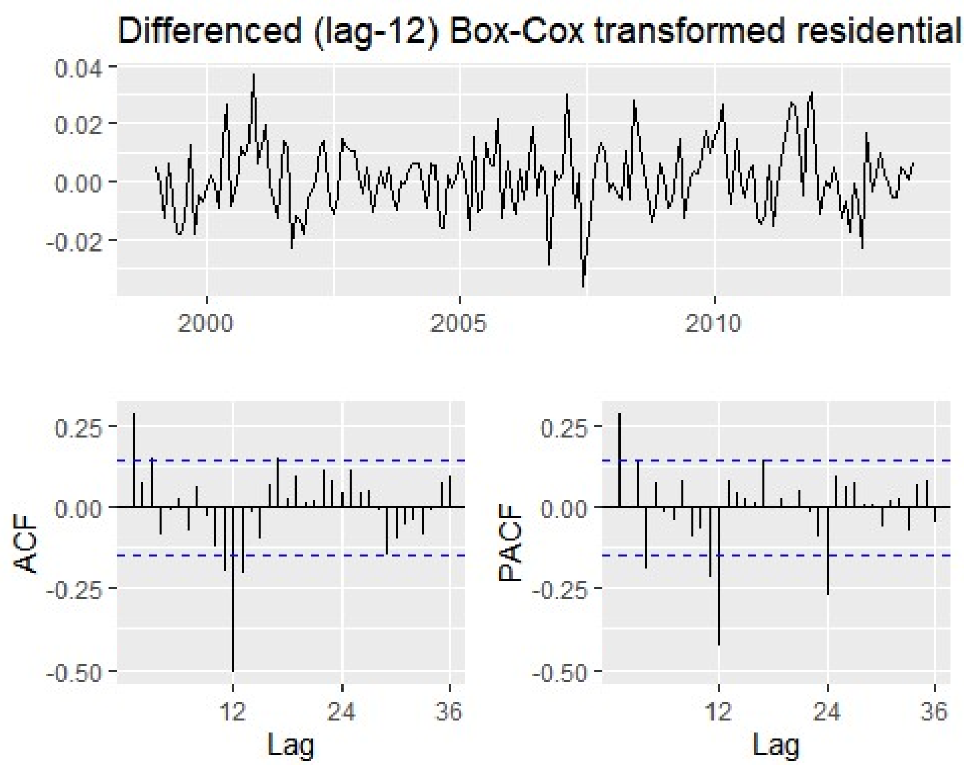
It is clearn that there is a seasonality in this data.

Model Creation

I will try to build a model that captures seasonality.

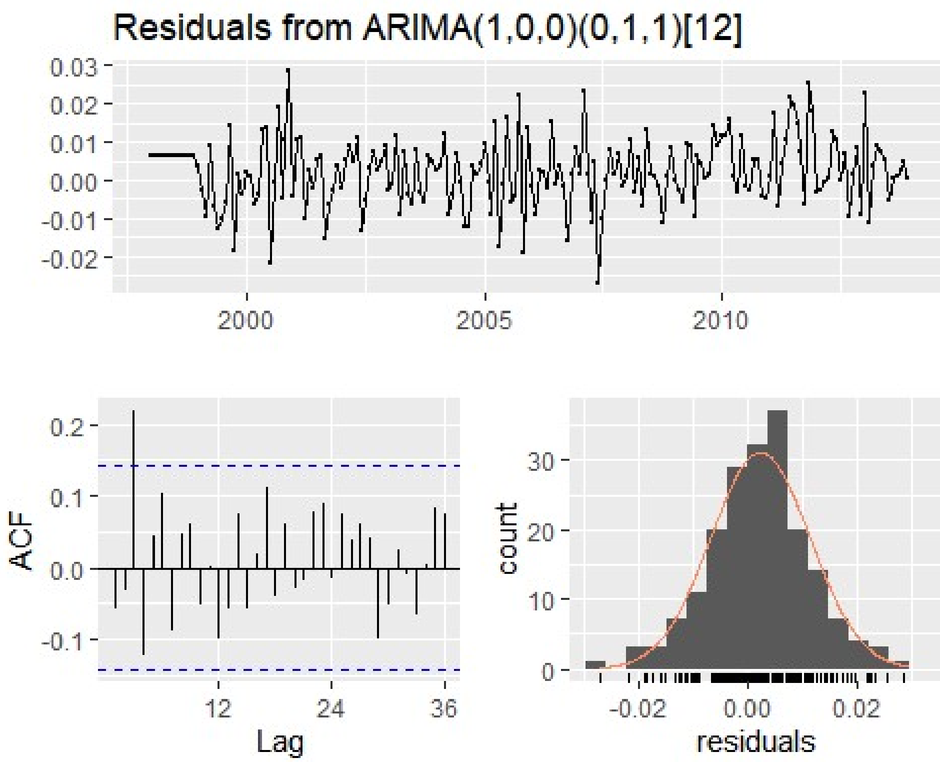
ARIMA with Box Cox Adjustment

```
power_ts <- tsclean(power_ts)
power_ts_lambda = BoxCox.lambda(power_ts)
power_ts_diff<-BoxCox(power_ts, power_ts_lambda)
ggtsdisplay(diff(power_ts_diff, 12), points = FALSE, main = "Differenced
(lag-12) Box-Cox transformed residential power usage")
```



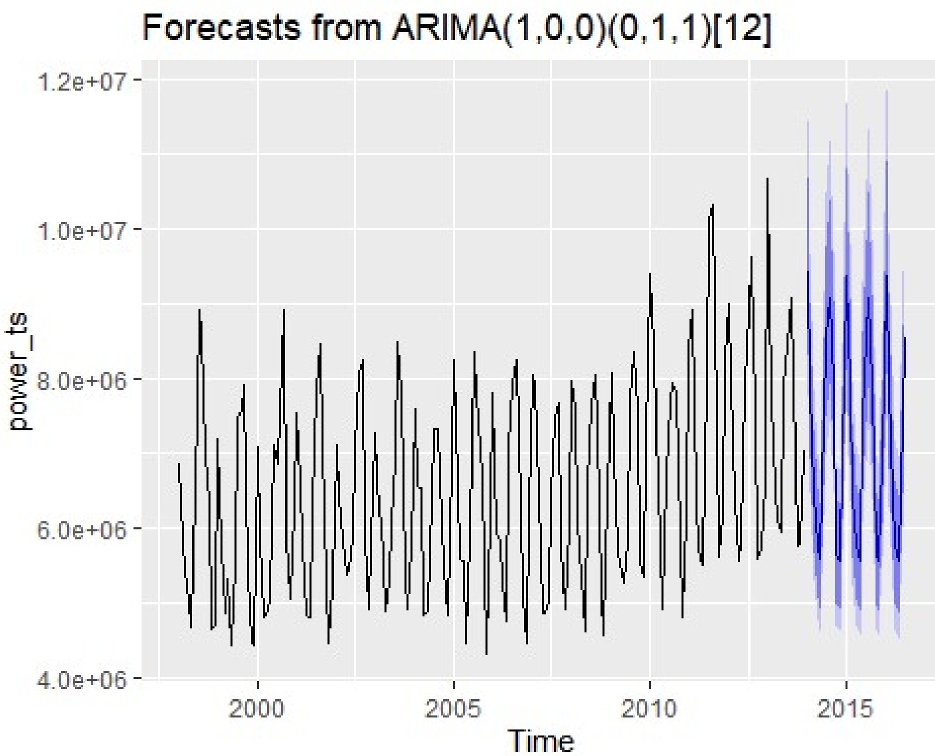
The timeseries data looks statioanary, I wont use seosanaly differencing.I also can see that extreme value in the PACF and ACF at lag=1 and lag=4.

```
power_ts_arima_fit <- Arima(power_ts, order = c(1, 0, 0), seasonal = c(0, 1,
1), lambda = power_ts_lambda)
checkresiduals(power_ts_arima_fit)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0)(0,1,1)[12]
## Q* = 31.099, df = 22, p-value = 0.0941
##
## Model df: 2.   Total lags used: 24
```

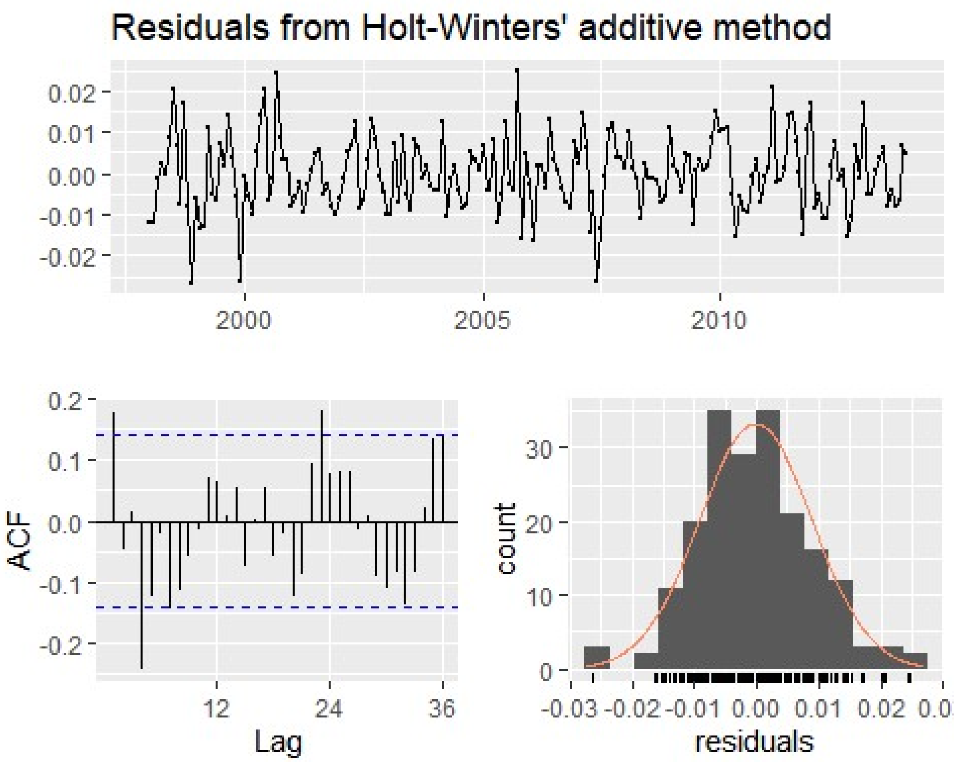
```
power_ts_arima_fit%>% forecast(h=31) %>% autoplot()
```




```
kpss.test(resid(power_ts_arima_fit))

## Warning in kpss.test(resid(power_ts_arima_fit)): p-value greater than
printed p-
## value

##
##  KPSS Test for Level Stationarity
##
## data:  resid(power_ts_arima_fit)
## KPSS Level = 0.32902, Truncation lag parameter = 4, p-value = 0.1
```

Box-Cox Holt-Winters Model

```
power_ts <- tsclean(power_ts)
adj_hw_fit <- hw(power_ts, h = 12, lambda = power_ts_lambda)
checkresiduals(adj_hw_fit)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 48.973, df = 8, p-value = 6.433e-08
##
## Model df: 16.   Total lags used: 24
```

MODEL EVALUATION

I will use the tsCV function and evaluate the models as I used for PART A.My goal is to find the model that produces minumum RMSE.

```
h <- 12
get_rmse <- function(error) {
  sqrt(mean(error^2, na.rm = TRUE))
}

power_ts_arima_fit_forecast <- function(x, h) {
  forecast(Arima(x, order = c(1, 0, 0), seasonal = c(0, 1, 1), lambda =
power_ts_lambda), h = h)
}

residuals_hw <- tsCV(power_ts, hw, h = h)
residuals_arima <- tsCV(power_ts, power_ts_arima_fit_forecast, h = h)
data.frame(Model_Name = c("Holt-Winters", "ARIMA"),
           RMSE = c(get_rmse(residuals_hw[, h]), get_rmse(residuals_arima[,
```

```
h]))) %>%
  arrange(RMSE) %>%
  kable() %>%
  kable_styling()
```

Model_Name
RMSE
ARIMA
703989.5
Holt-Winters
1010721.9

The ARIMA(1, 0, 0)(0, 1, 1) model gives the minumum RMSE among the other model.

SUMMARY

Since Arima Model gives better RMSE result, I will use ARIMA model for forecasting purposes.

```
power_ts_forecast <-power_ts_arima_fit %>% forecast(h=12)
power_forecast_df <-data_frame(DATE = paste0(2014, "-", month.abb), POWER_KWH
= power_ts_forecast$mean)

## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.

power_forecast_df %>%
  kable() %>%
  kable_styling()
```

DATE
POWER_KWH
2014-Jan
9417011
2014-Feb
7893327
2014-Mar
6436805
2014-Apr
5756614
2014-May
5570785
2014-Jun
7479484
2014-Jul
8522770

2014-Aug
9080593
2014-Sep
7909745
2014-Oct
5661778
2014-Nov
5568654
2014-Dec
6919608

```
write_csv(power_forecast_df,"power_forecast_df.csv")
```

Part C - BONUS, optional (part or all)

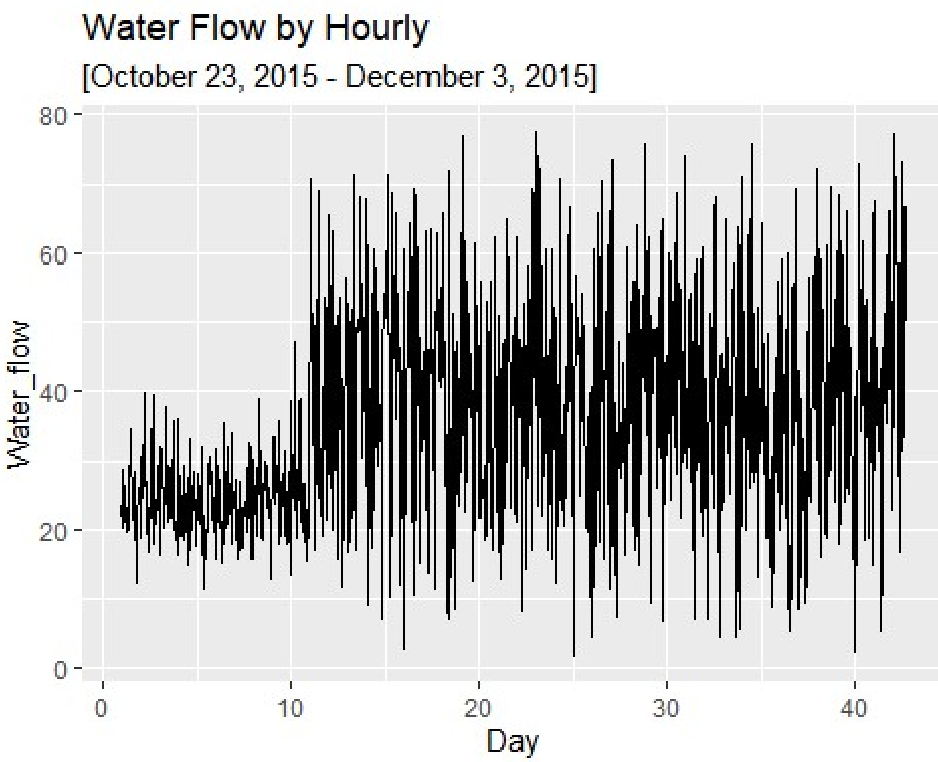
Part C consists of two data sets. These are simple 2 columns sets, however they have different time stamps. Your optional assignment is to time-base sequence the data and aggregate based on hour (example of what this looks like, follows). Note for multiple recordings within an hour, take the mean. Then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

Data Engineering

```
colnames(water1_data)= c("w1_date_time","WaterFlow")
colnames(water2_data)= c("w2_date_Time","WaterFlow")
water_df= water1_data %>% mutate(w2_date_Time =
lubridate::round_date(w1_date_time,"hour") ) %>%
select(w2_date_Time,WaterFlow) %>% bind_rows(water2_data) %>%
group_by(w2_date_Time) %>% summarize(WaterFlowF = mean(WaterFlow, na.rm = T))
colnames(water_df)= c("Date_Time","WaterFlow")
water_ts = ts(water_df$WaterFlow,frequency = 24)
```

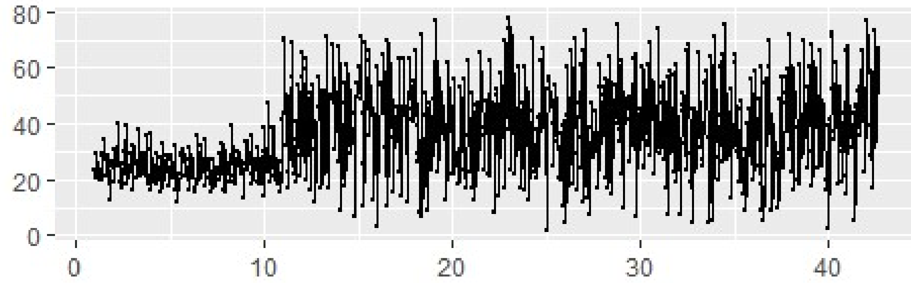
Data Graphs

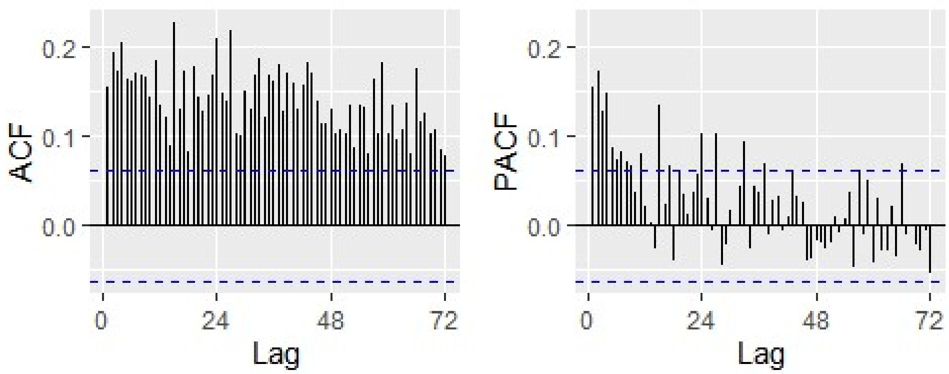
```
autoplot(water_ts) +
  labs(title = "Water Flow by Hourly", subtitle = "[October 23, 2015 -
December 3, 2015]", x = "Day", y = "Water_flow")
```



It is clear that data shows a lot of high and low as outliers or shocks.The variance of data also seems like not constant.

```
ggtsdisplay(water_ts)
```





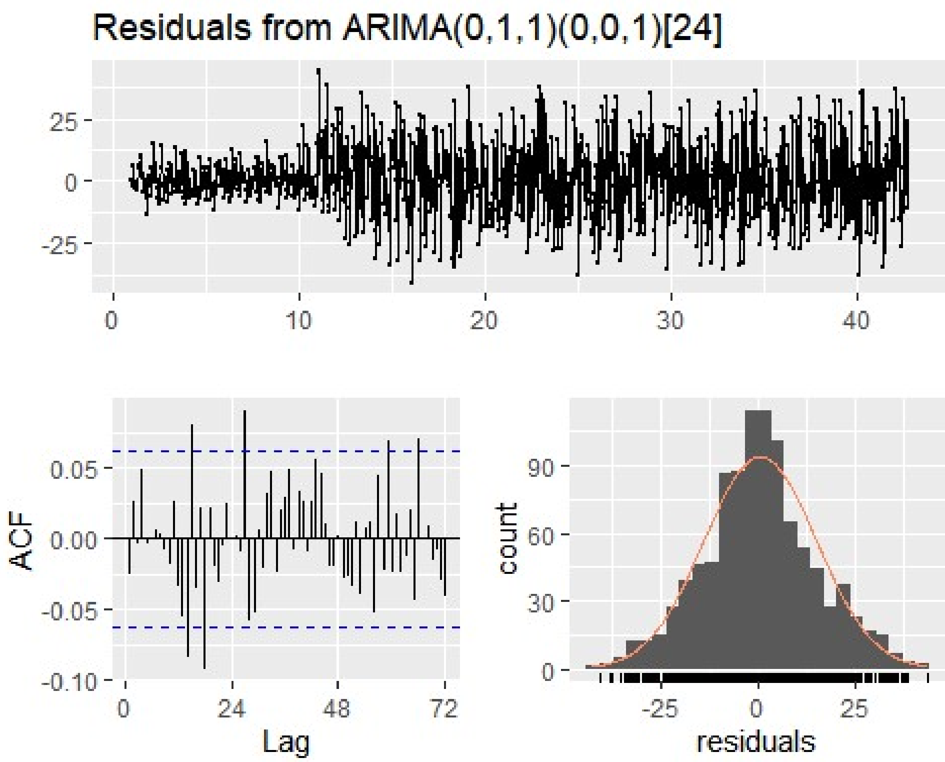
The ACF and PACF plots above indivates that AR(5) autoregression of 5, and the ACF a MA(2) moving average of order 2

Model Creation

I will use following forecating models on this time series and determine which one is better by estimatin error metric RMSE. I will use time series cross validation function to estimate RMSE for timeseries data.

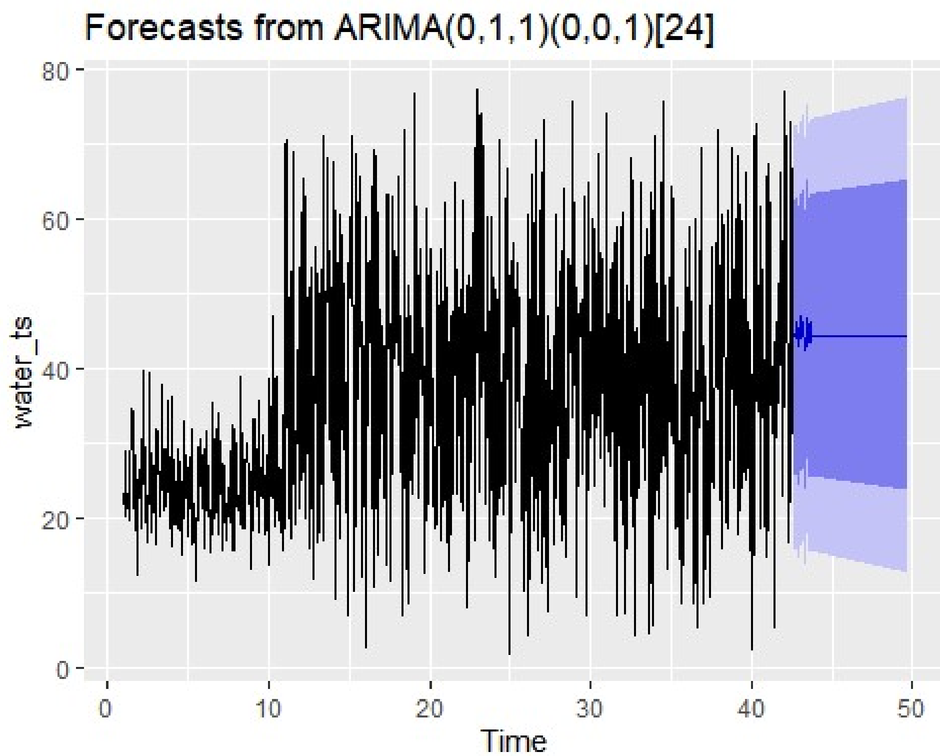
ARIMA

```
water_ts_arima_fit <- auto.arima(water_ts)
checkresiduals(water_ts_arima_fit)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)(0,0,1)[24]
## Q* = 67.6, df = 46, p-value = 0.02068
##
## Model df: 2.   Total lags used: 48
```

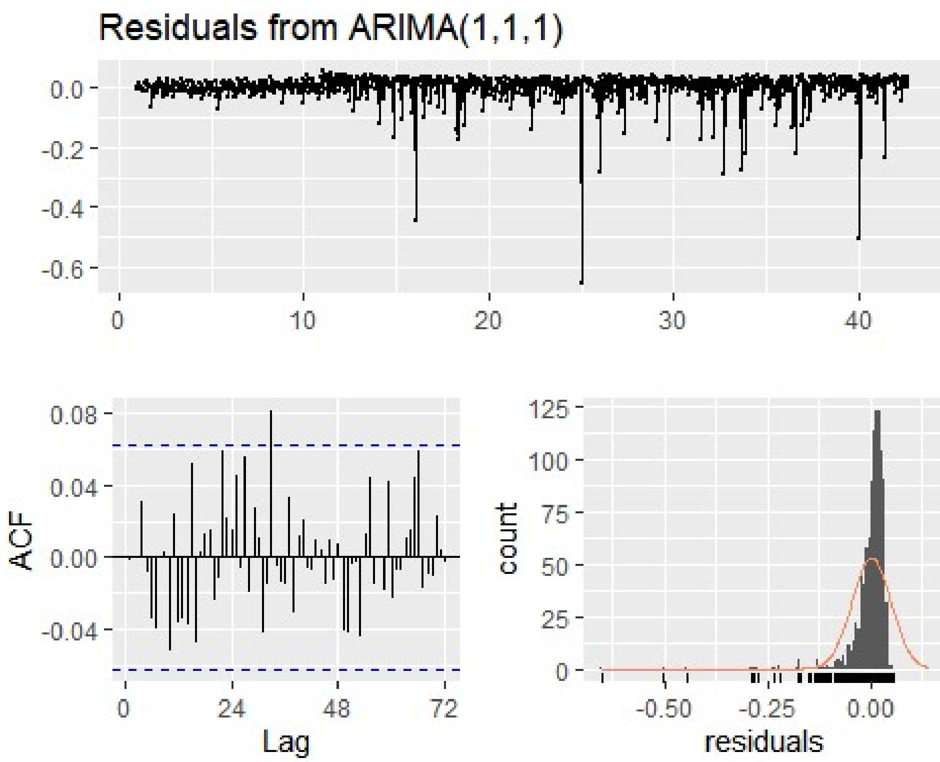
```
water_ts_arima_fit%>% forecast(h=168) %>% autoplot()
```




```
kpss.test(resid(water_ts_arima_fit))

## Warning in kpss.test(resid(water_ts_arima_fit)): p-value greater than
printed p-
## value

##
##  KPSS Test for Level Stationarity
##
## data:  resid(water_ts_arima_fit)
## KPSS Level = 0.076651, Truncation lag parameter = 7, p-value = 0.1
```

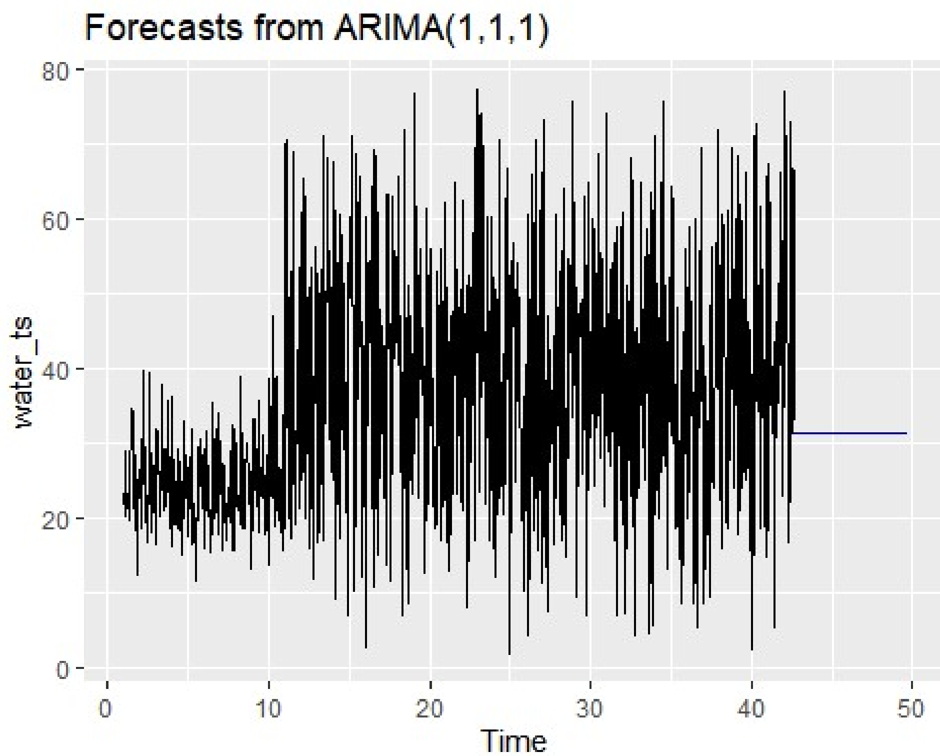
ARIMA with Box Cox Adjustment

```
water_ts_lambda = BoxCox.lambda(water_ts)
water_ts_box_arima_fit <- Arima(water_ts, order=c(1, 1, 1),lambda =
water_ts_lambda)
checkresiduals(water_ts_box_arima_fit)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,1)
## Q* = 41.128, df = 46, p-value = 0.6761
##
## Model df: 2.   Total lags used: 48
```

```
water_ts_box_arima_fit%>% forecast(h=168) %>% autoplot()
```




```
kpss.test(resid(water_ts_box_arima_fit))

## Warning in kpss.test(resid(water_ts_box_arima_fit)): p-value greater than
## printed p-value

##
##  KPSS Test for Level Stationarity
##
## data:  resid(water_ts_box_arima_fit)
## KPSS Level = 0.087414, Truncation lag parameter = 7, p-value = 0.1
```

MODEL EVALUATION

I will use the `tsCV` function and evaluate the models as I used for PART A and PART B.My goal is to find the model that produces minumum RMSE.

```
h <- 168
get_rmse <- function(error) {
  sqrt(mean(error^2, na.rm = TRUE))
}

water_ts_arima_fit_forecast <- function(x, h) {
  forecast(Arima(x, order = c(0, 1, 1),seasonal = c(0, 0, 1)), h = h)
}

water_ts_arima_box_fit_forecast <- function(x, h) {
  forecast(Arima(x, order=c(1, 1, 1), lambda = water_ts_lambda), h = h)
```

```
}


''

## [1] ""

residuals_arima<- tsCV(water_ts,water_ts_arima_fit_forecast, h = h)
residuals_arima_box <- tsCV(water_ts, water_ts_arima_box_fit_forecast, h = h)
data.frame(Model_Name = c("ARIMA", "ARIMA_BOX_COX"),
           RMSE = c(get_rmse(residuals_arima[, h]),
get_rmse(residuals_arima_box[, h]))) %>%
  arrange(RMSE) %>%
  kable() %>%
  kable_styling()
```

Model_Name	RMSE
ARIMA	17.36650
ARIMA_BOX_COX	18.82279

SUMMARY

The ARIMA and ARIMA with Box_COX transformation model results are pretty close.However, Since Arima gives minumum value in RMSE result, I will use ARIMA(0,1,1)(0,0,1) model for forecasting purposes.

```
water_forecast<-water_ts_arima_fit %>% forecast(h=168)
water_forecast_df <- data.frame(water_forecast$mean)
colnames(water_forecast_df) <- "water_flow"
row.names(water_forecast_df) <- seq(ymd_hm("2015-12-3 17:00"), ymd_hm("2015-
12-10 16:00"), by = "hour")
kable(water_forecast_df)
```

	water_flow
2015-12-03 17:00:00	44.10568
2015-12-03 18:00:00	45.27422
2015-12-03 19:00:00	46.18329
2015-12-03 20:00:00	44.17254
2015-12-03 21:00:00	45.49246

2015-12-03 22:00:00
44.69631
2015-12-03 23:00:00
42.99370
2015-12-04 00:00:00
44.72864
2015-12-04 01:00:00
46.97255
2015-12-04 02:00:00
44.73431
2015-12-04 03:00:00
46.39813
2015-12-04 04:00:00
45.59040
2015-12-04 05:00:00
45.52474
2015-12-04 06:00:00
43.32449
2015-12-04 07:00:00
44.10577
2015-12-04 08:00:00
42.42523
2015-12-04 09:00:00
43.14329
2015-12-04 10:00:00
44.67196
2015-12-04 11:00:00
46.79987
2015-12-04 12:00:00
43.47161
2015-12-04 13:00:00
46.13864
2015-12-04 14:00:00
44.38329
2015-12-04 15:00:00
43.58102
2015-12-04 16:00:00
46.20608
2015-12-04 17:00:00
44.46400

2015-12-04 18:00:00
44.46400
2015-12-04 19:00:00
44.46400
2015-12-04 20:00:00
44.46400
2015-12-04 21:00:00
44.46400
2015-12-04 22:00:00
44.46400
2015-12-04 23:00:00
44.46400
2015-12-05 00:00:00
44.46400
2015-12-05 01:00:00
44.46400
2015-12-05 02:00:00
44.46400
2015-12-05 03:00:00
44.46400
2015-12-05 04:00:00
44.46400
2015-12-05 05:00:00
44.46400
2015-12-05 06:00:00
44.46400
2015-12-05 07:00:00
44.46400
2015-12-05 08:00:00
44.46400
2015-12-05 09:00:00
44.46400
2015-12-05 10:00:00
44.46400
2015-12-05 11:00:00
44.46400
2015-12-05 12:00:00
44.46400
2015-12-05 13:00:00
44.46400

2015-12-05 14:00:00
44.46400
2015-12-05 15:00:00
44.46400
2015-12-05 16:00:00
44.46400
2015-12-05 17:00:00
44.46400
2015-12-05 18:00:00
44.46400
2015-12-05 19:00:00
44.46400
2015-12-05 20:00:00
44.46400
2015-12-05 21:00:00
44.46400
2015-12-05 22:00:00
44.46400
2015-12-05 23:00:00
44.46400
2015-12-06 00:00:00
44.46400
2015-12-06 01:00:00
44.46400
2015-12-06 02:00:00
44.46400
2015-12-06 03:00:00
44.46400
2015-12-06 04:00:00
44.46400
2015-12-06 05:00:00
44.46400
2015-12-06 06:00:00
44.46400
2015-12-06 07:00:00
44.46400
2015-12-06 08:00:00
44.46400
2015-12-06 09:00:00
44.46400

2015-12-06 10:00:00
44.46400
2015-12-06 11:00:00
44.46400
2015-12-06 12:00:00
44.46400
2015-12-06 13:00:00
44.46400
2015-12-06 14:00:00
44.46400
2015-12-06 15:00:00
44.46400
2015-12-06 16:00:00
44.46400
2015-12-06 17:00:00
44.46400
2015-12-06 18:00:00
44.46400
2015-12-06 19:00:00
44.46400
2015-12-06 20:00:00
44.46400
2015-12-06 21:00:00
44.46400
2015-12-06 22:00:00
44.46400
2015-12-06 23:00:00
44.46400
2015-12-07 00:00:00
44.46400
2015-12-07 01:00:00
44.46400
2015-12-07 02:00:00
44.46400
2015-12-07 03:00:00
44.46400
2015-12-07 04:00:00
44.46400
2015-12-07 05:00:00
44.46400

2015-12-07 06:00:00
44.46400
2015-12-07 07:00:00
44.46400
2015-12-07 08:00:00
44.46400
2015-12-07 09:00:00
44.46400
2015-12-07 10:00:00
44.46400
2015-12-07 11:00:00
44.46400
2015-12-07 12:00:00
44.46400
2015-12-07 13:00:00
44.46400
2015-12-07 14:00:00
44.46400
2015-12-07 15:00:00
44.46400
2015-12-07 16:00:00
44.46400
2015-12-07 17:00:00
44.46400
2015-12-07 18:00:00
44.46400
2015-12-07 19:00:00
44.46400
2015-12-07 20:00:00
44.46400
2015-12-07 21:00:00
44.46400
2015-12-07 22:00:00
44.46400
2015-12-07 23:00:00
44.46400
2015-12-08 00:00:00
44.46400
2015-12-08 01:00:00
44.46400

2015-12-08 02:00:00
44.46400
2015-12-08 03:00:00
44.46400
2015-12-08 04:00:00
44.46400
2015-12-08 05:00:00
44.46400
2015-12-08 06:00:00
44.46400
2015-12-08 07:00:00
44.46400
2015-12-08 08:00:00
44.46400
2015-12-08 09:00:00
44.46400
2015-12-08 10:00:00
44.46400
2015-12-08 11:00:00
44.46400
2015-12-08 12:00:00
44.46400
2015-12-08 13:00:00
44.46400
2015-12-08 14:00:00
44.46400
2015-12-08 15:00:00
44.46400
2015-12-08 16:00:00
44.46400
2015-12-08 17:00:00
44.46400
2015-12-08 18:00:00
44.46400
2015-12-08 19:00:00
44.46400
2015-12-08 20:00:00
44.46400
2015-12-08 21:00:00
44.46400

2015-12-08 22:00:00
44.46400
2015-12-08 23:00:00
44.46400
2015-12-09 00:00:00
44.46400
2015-12-09 01:00:00
44.46400
2015-12-09 02:00:00
44.46400
2015-12-09 03:00:00
44.46400
2015-12-09 04:00:00
44.46400
2015-12-09 05:00:00
44.46400
2015-12-09 06:00:00
44.46400
2015-12-09 07:00:00
44.46400
2015-12-09 08:00:00
44.46400
2015-12-09 09:00:00
44.46400
2015-12-09 10:00:00
44.46400
2015-12-09 11:00:00
44.46400
2015-12-09 12:00:00
44.46400
2015-12-09 13:00:00
44.46400
2015-12-09 14:00:00
44.46400
2015-12-09 15:00:00
44.46400
2015-12-09 16:00:00
44.46400
2015-12-09 17:00:00
44.46400

2015-12-09 18:00:00
44.46400
2015-12-09 19:00:00
44.46400
2015-12-09 20:00:00
44.46400
2015-12-09 21:00:00
44.46400
2015-12-09 22:00:00
44.46400
2015-12-09 23:00:00
44.46400
2015-12-10 00:00:00
44.46400
2015-12-10 01:00:00
44.46400
2015-12-10 02:00:00
44.46400
2015-12-10 03:00:00
44.46400
2015-12-10 04:00:00
44.46400
2015-12-10 05:00:00
44.46400
2015-12-10 06:00:00
44.46400
2015-12-10 07:00:00
44.46400
2015-12-10 08:00:00
44.46400
2015-12-10 09:00:00
44.46400
2015-12-10 10:00:00
44.46400
2015-12-10 11:00:00
44.46400
2015-12-10 12:00:00
44.46400
2015-12-10 13:00:00
44.46400

2015-12-10 14:00:00
44.46400
2015-12-10 15:00:00
44.46400
2015-12-10 16:00:00
44.46400

```
write.csv(water_forecast_df,"water_flow_forecast.csv")
```