DATA 624: Project 1

Vinicio Haro
Sang Yoon (Andy) Hwang
Julian McEachern
Jeremy O'Brien
Bethany Poulin

October 22, 2019

Contents

Overview

We split the work into three sections for Project 1. Individual team members each took lead on individual problem. Jermey and Julian focused on Part A, Sang Yoon (Andy) and Vinicio worked on Part B, and Bethany took lead on Part C. Juliann created an overall format for the assignment to be used and all team members collectively worked together on reviewing and merging our finished product.

Dependencies

The following R libraries were used to complete this assignment:

```
library(easypackages)

libraries('knitr', 'kableExtra', 'default')

# Processing
libraries('readxl', 'tidyverse', 'janitor', 'imputeTS', 'tsoutliers', 'lubridate')

### UNABLE TO GET JAVA TO CONFIGURE FOR >>> 'xlsx'

# Timeseries
libraries('psych', 'urca', 'forecast', 'timetk', 'fpp2')

# Graphing
libraries('ggplot2', 'grid', 'gridExtra', 'ggfortify','ggpubr', 'scales')
```

Data

Data was stored within our group repository and imported below using the `readxl` package. Each individual question was solved within an R script and the data was sourced into our main report. For replication purposes, we also made our R scripts available within our appendix. All forecasts were exported and saved a `.csv` file in our github repository folder named forecasts.

```
# Data Aquisition
atm_data <- read_excel("data/ATM624Data.xlsx")
power_data <- read_excel("data/ResidentialCustomerForecastLoad-624.xlsx")
pipe1_data <- read_excel("data/Waterflow_Pipe1.xlsx")
pipe2_data <- read_excel("data/Waterflow_Pipe2.xlsx")

# Source Code
source('~/GitHub/CUNY_DATA_624/Project One/scripts/Part-A.R')
source('~/GitHub/CUNY_DATA_624/Project One/scripts/Part-B.R')
source('~/GitHub/CUNY_DATA_624/Project One/scripts/Part-C.R')
```

1 Part A: ATMs

Instructions: 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. Your report must be professional - most of all - readable, EASY to follow. Let me know what you are thinking, assumptions you are making! Your forecast is a simple CSV or Excel file that MATCHES the format of the data I provide.

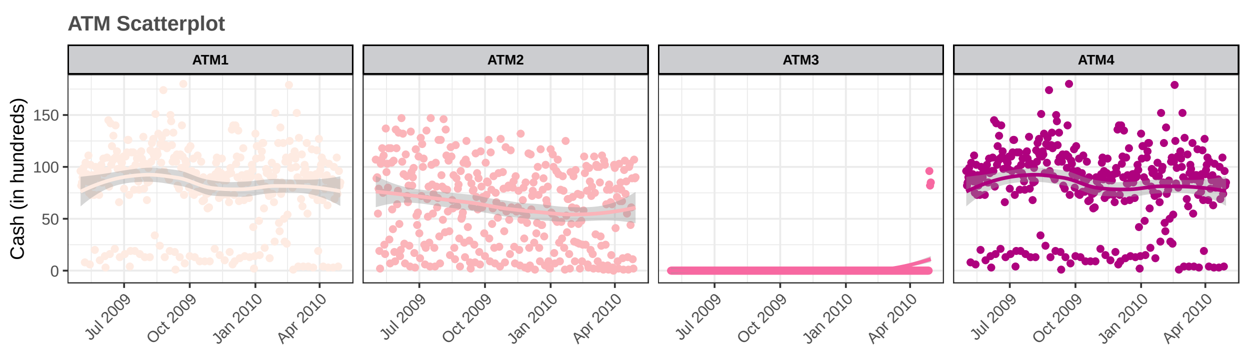
1.1 Exploration

The data covers a period of Friday May 1, 2010 through Saturday April 30, 2010. While reviewing the data, we identified that the original data file contained `NA` values in our `ATM` and `Cash` columns for 14 observations between May 1 and 14, 2010. As these contain no information, we removed these missing values and transformed the dataset into a wide format.

Our initial review also revealed that ATM2 contained one missing value on 2009-10-25 and that ATM4 contained a potential outlier of $1,123 on 2010-02-09. We replaced both values with the corresponding mean value of each machine.

We examined summary statistics for each ATM time series:

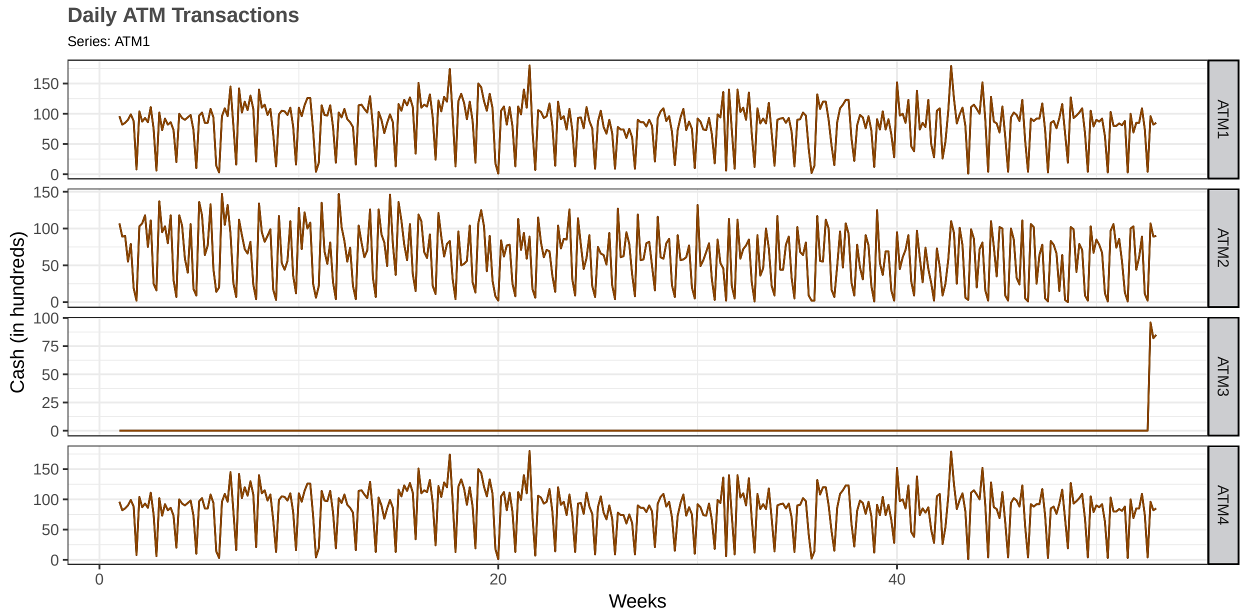
- ATM1 and ATM2 have pretty normal distributions; ATM1's daily mean cash dispensed is $84, and ATM2's is $62.
- ATM3 only dispensed cash on the last three days of the time series - as this provides few data points on which to forecast, we'll need to treat it specially.
- ATM4 has a similar mean to ATM1, but skew and kurtosis suggest the impact of an outlier Wednesday, February 10, 2010. If this ATM is located in the Northeastern United States, this may have a relationship to a blizzard which struck on that day.



Last, we used a scatterplot to examine the correlation between cash withdrawals and dates for each machine. We identified similiar patterns between ATM1 and ATM4, which show non-linear fluctuations that suggest a potential trend component in these timeseries. ATM2 follows a relatively linear path and decreases overtime. This changes in the last few observations, where withdrawals begin to increase. As mentioned, there are only 3 observed transactions for ATM3 that appear at the end of the captured time period.

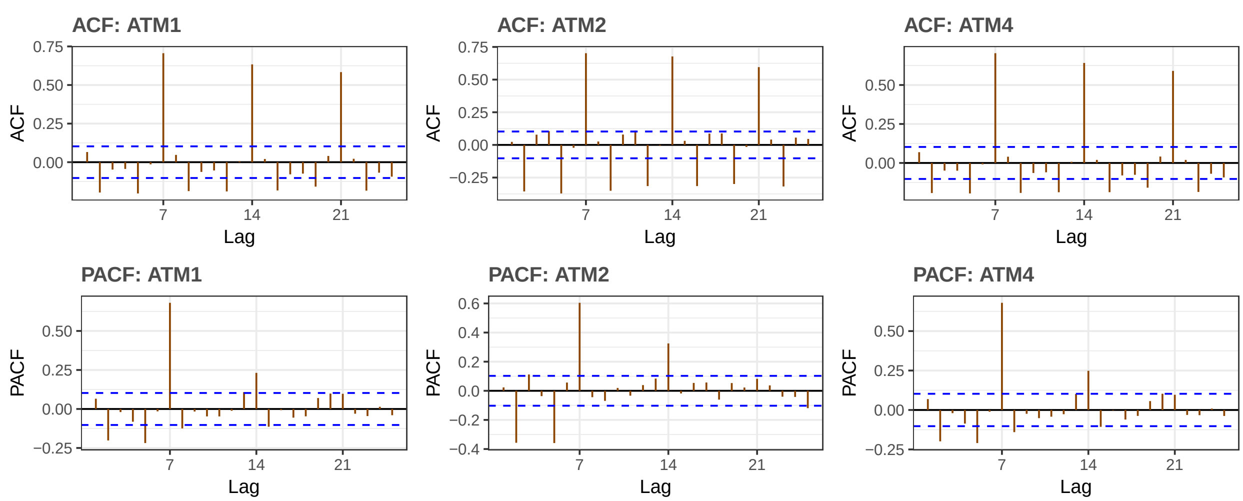
Our cleaned dataframe was then converted into a timeseries format. The time series plots show high weekly variance, for ATM1, ATM2, and ATM4 - consistent with our takeaway from the scatterplots.

These plots also remind us that ATM3 only dispensed cash on 3 days at the end of the timespan, with a daily range between $82 and $96. Given the paucity of observations in the training data, the simplest possible approach to forecasting ATM3, averaging, is likely best. Given that ATM3 distributed no cash until April 28, 2010, we'll assume that it was not operating until then and only include the three day window of non-zero observations in the forecast.



1.2 Evaluation

We constructed our initial timeseries for ATM1, ATM2, and ATM4 using a weekly frequency. Our ACF plots for each ATM showcases large, decreasing lags starting at 7. This pattern continues in a multiple of seven, which confirms our assumption about seasonality within the observed data. These lags are indicative of a weekly pattern.



Our plots further suggest that the ATM data is non-stationary. We performed a unit root test using the `ur.kpss()` function to confirm this observation. The test results below show that differencing is required on all ATM2 and ATM4 series. ATM1 falls just below the cut-off critical value, but could still benefit from differencing due to the observed seasonal pattern.

Table 1.1: KPSS unit root test

ATM	No-Diff	Diff-1
ATM1	0.4967	0.0219
ATM2	2.0006	0.016
ATM4	0.5182	0.0211

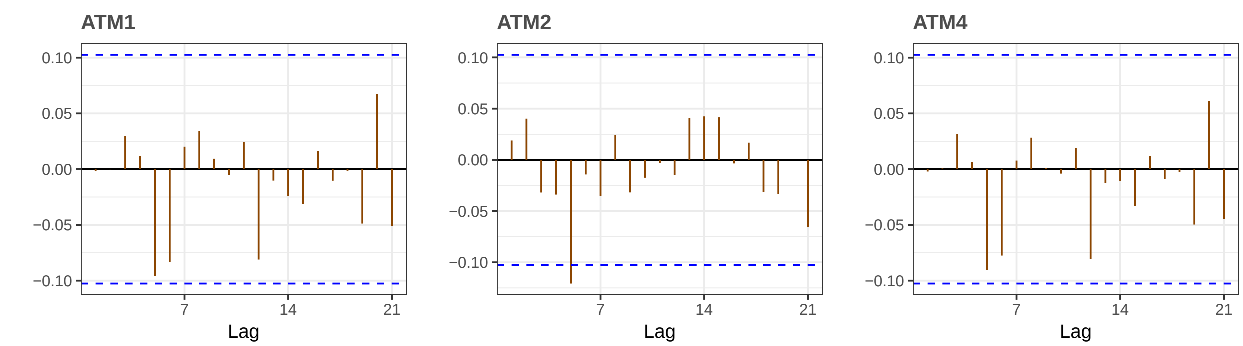
1.3 Modeling

We used `auto.arima()` and set `D=1` to account for seasonal differencing of our data to select the best ARIMA models for ATM1, ATM2, and ATM4. The full models and accuracy statistics for each series can be viewed in the appendix.

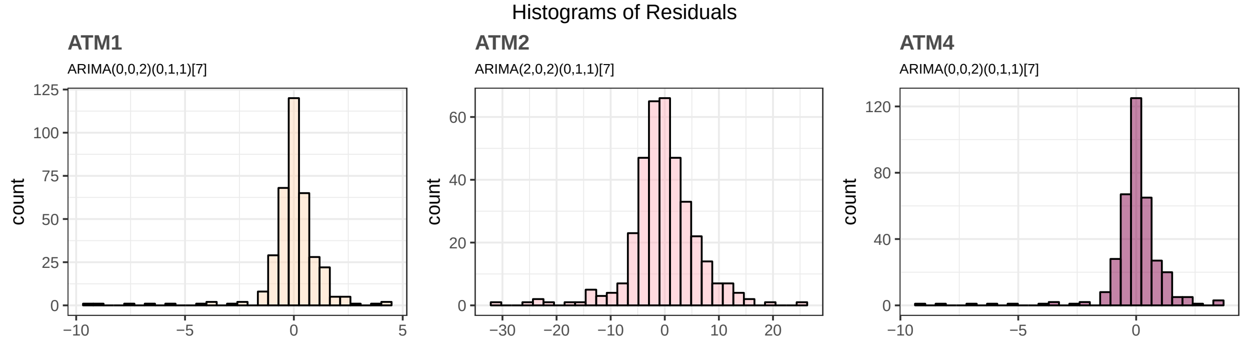
- **ATM1**: ARIMA$(0, 0, 2)(0, 1, 1)_7$
- **ATM2**: ARIMA$(2, 0, 2)(0, 1, 1)_7$
- **ATM3**: MEAN
- **ATM4**: ARIMA$(0, 0, 2)(0, 1, 1)_7$

The residual ACF plots contain no pattern and the lags fall within the critical value, which suggest they are white noise and not autocorrelated. The residual histograms follow a relatively normal distribution that is centered around zero. The p-value from the Ljung-Box test for ATM1, ATM2, and ATM4 all exceeds 0.05, which futher supports that residuals happen by chance and the models adequately fit the observed data.

ACF Plots of Residuals

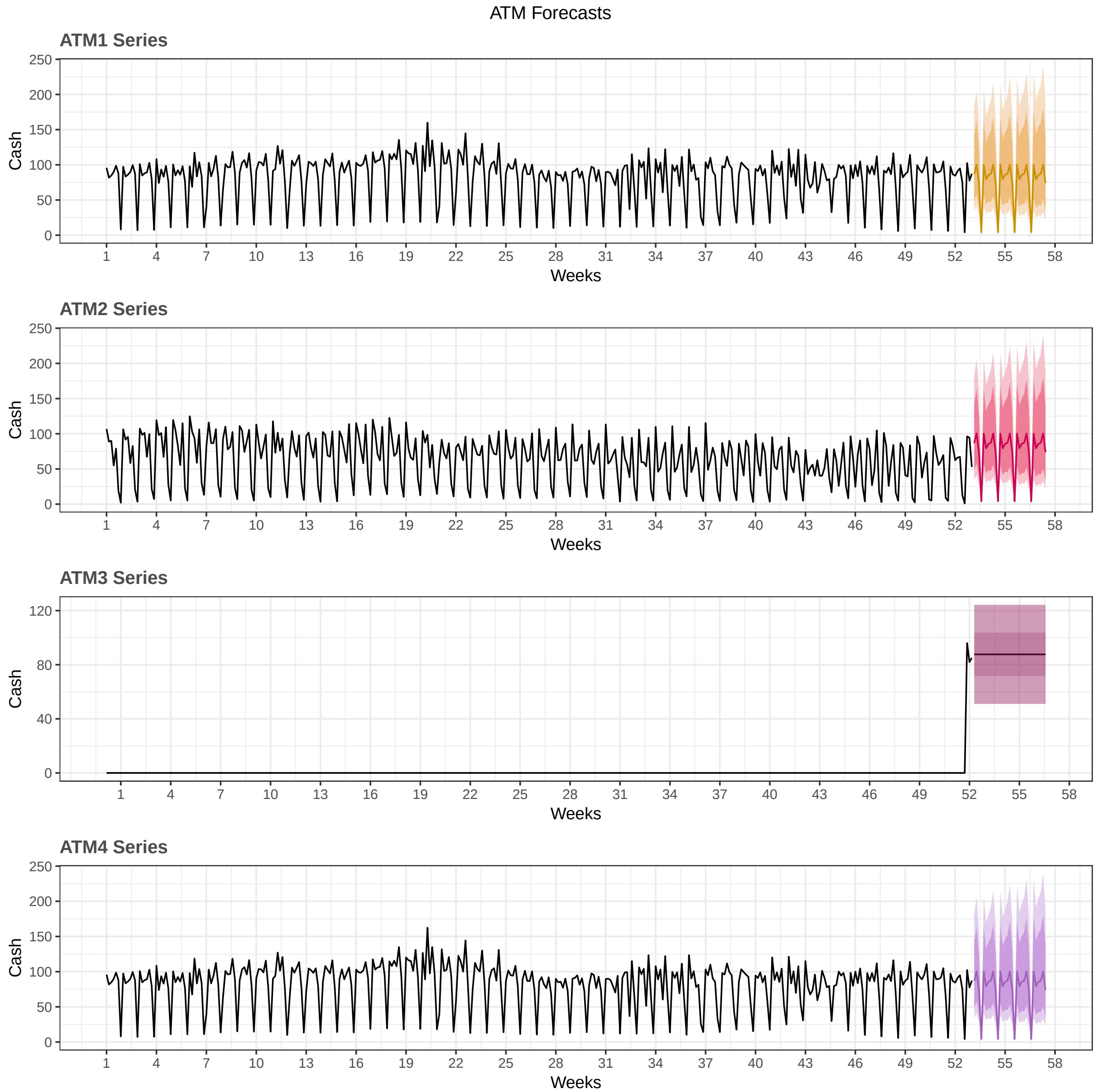


Histograms of Residuals



1.4 Forecast

A forecast for the month of May will be 31 days in length. We applied a forecast to each series, which spanned across 5 weeks. The numeric forecasts can be viewed in a table output in the appendix section and are also located within our data output folder.



1.5 Summary

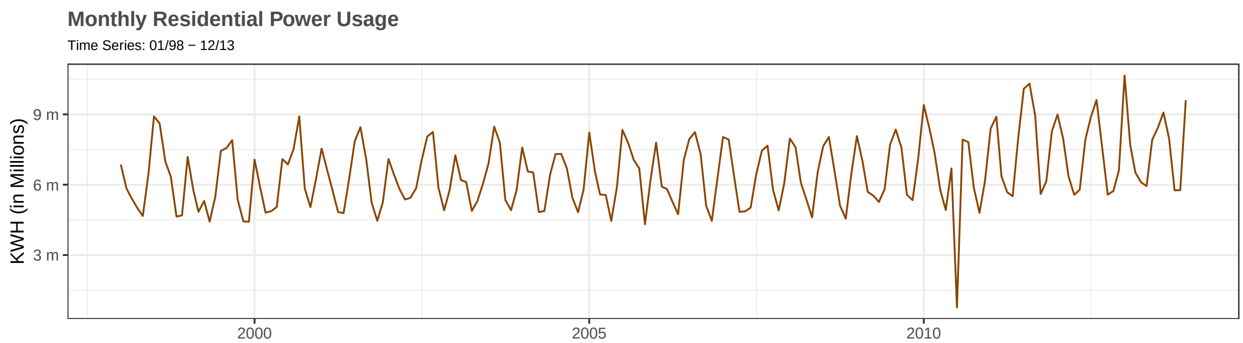
Jeremy & Juliann - synthesize findings

2 Part B: Forecasting Power

Instructions: 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 these to your existing files above - clearly labeled.

2.1 Exploration

We observed there was a missing value in September 2008. We used imputation method called na.interpolation which performs a technique in numerical analysis which estimates a value from known data points. For our case, linear method using first order Taylor polynomial is used.



Our initial time series plot reveal annual seasonality within this time series. The box plot/seasonality plot actually reveals where power consumption fluctuations occur within each of the cycle positions. We can speculate that this could be due to there being no major Holidays that require power draining decor plus we assume minimal AC usage during the cold months.

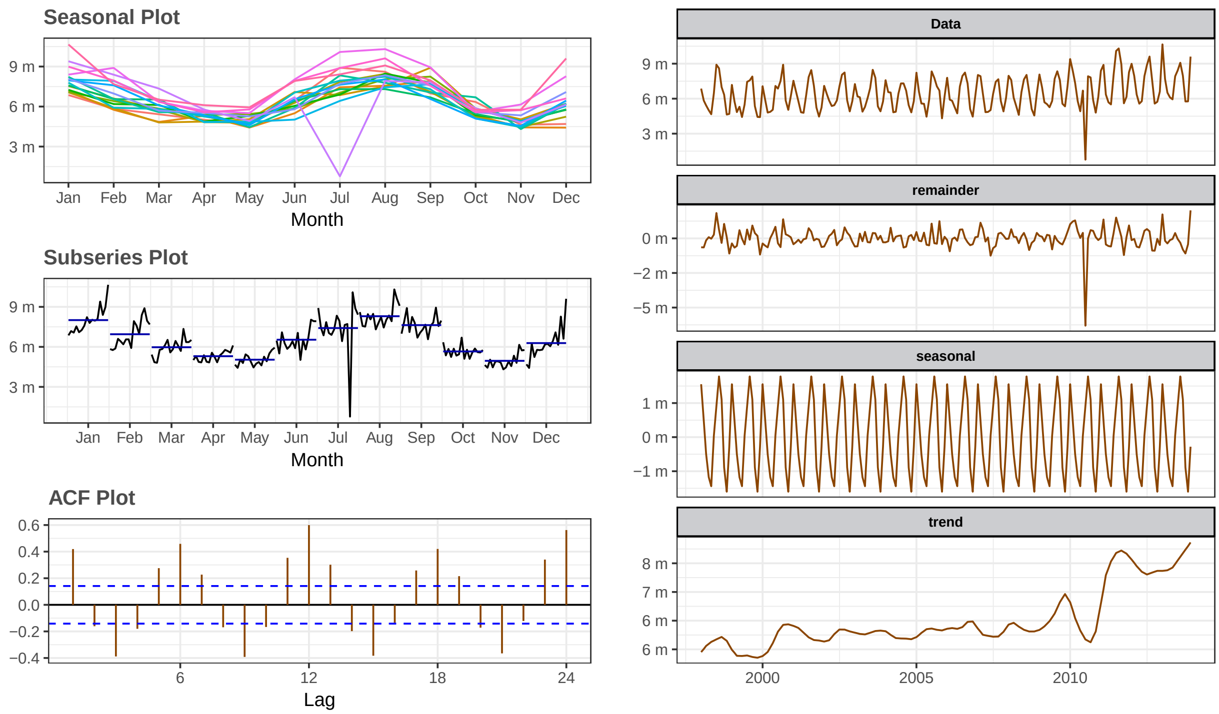
2.2 Evaluation

We see power consumption increase between the months of June and August. This must be tied to AC usage during the warmer months of a year and finally power usage dips from September to Novemeber with a small spike in December. We speculate that thisis due to transitioning out of summer. The spike in December could be connected to the usage or Holiday lights being kept on.

Within the overall TS plot, we see a dip in July 2010. This could be due to a power outtage during a hot summer month. This can certainly be considered to be an outlier within this TS. Using TSOutliers, we can actually identify the index where our outliers may be. TSoutliers also replaces the outlier using Box-Cox. If set lambda=auto, then TSoutliers will automatically perform Box-Cox transformation.

The ACF plot shows that autocorrelations are well outside the significant space indicating the series is not white noise, non-stationary.

Diagnostic Plots

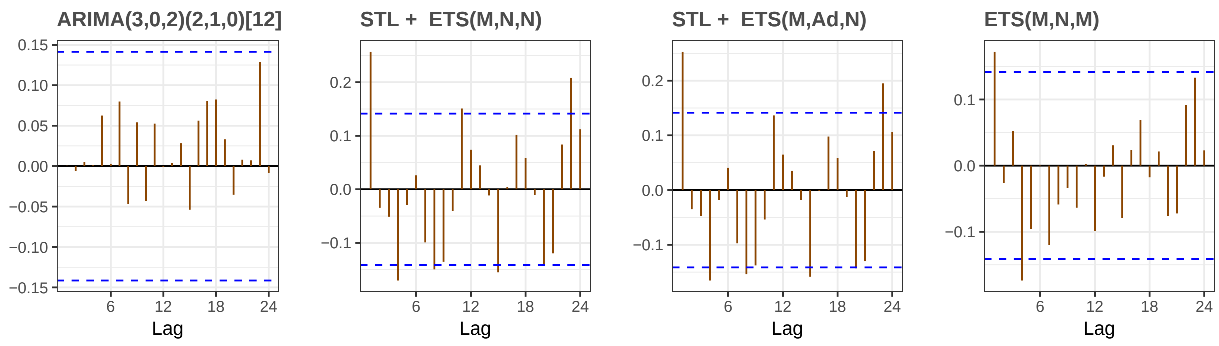


2.3 Modeling

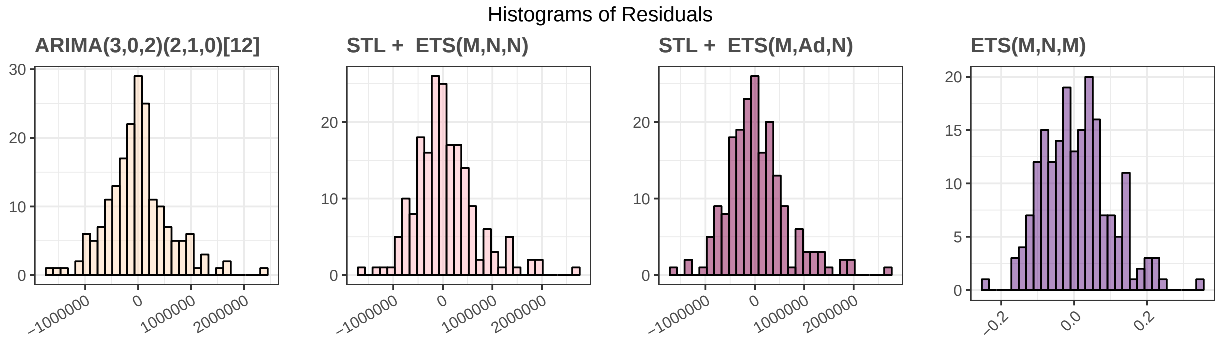
We built four different models on the power data. Methods included ARIMA, STL (with and without dampening), and ETS. We can make some preliminary observations on the reliability of these models from checking the residuals.

The residual ACF plots show residual autocorrelations for each of our models. Model #1: ARIMA has less autocorrelation than the other three models. Model 1 is well within the 95% limits indicated by the dotted blue lines.

ACF Plots of Residuals

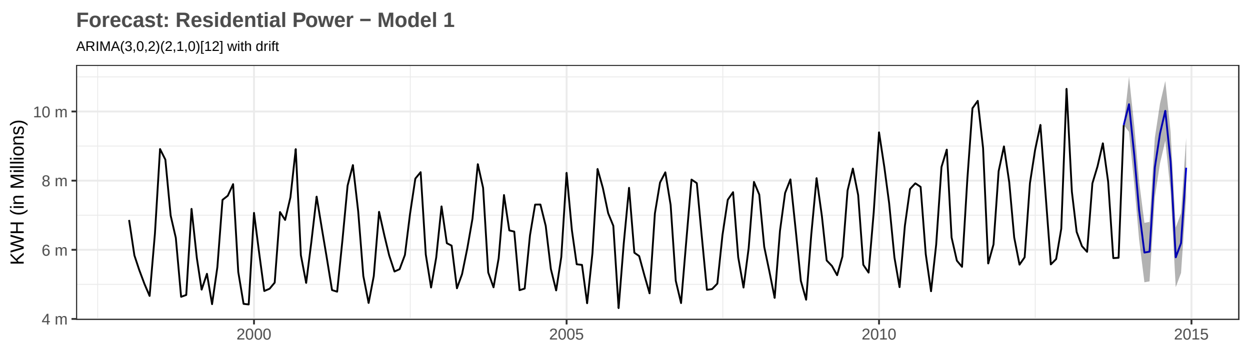


The residuals for each of our models does not have a major deviance from normality, however residuals of Model #1: ARIMA do not have an extended number of bins distorting the normality proximity but we can say it is still fairly normally distributed.



Our last residual check, the Ljung-Box test, showed that only model had a p-value > 0.05: Model #1: ARIMA. This implies that the residuals from other models are not independent, hence not white noise. As a result, we will continue with this model for forcasting. We have included the full model summary for ARIMA and our other attempts in the appendix.

2.4 Forecast



The `auto.arima()` function performs cross validation on hyperparameter tuning to find the best model with parameters of `order` and `seasonal` that minimize `AIC`. This gave us **arima_model**: ARIMA$(3, 0, 2)(2, 1, 0)12$ with drift resulting `AIC` = 5332.24. Since ARIMA is the only reliable model, as other models failed Ljung test, we will plot forecasts of ARIMA only. The forecasted values can be viewed in the appendix.

2.5 Summary

We implemented a cross validation method of testing for `h=12`. The process randomly chooses 12 points to measure and take the average of RMSEs. By definition, a lower RMSE on test set is attributed with a better forecast on unseen data.

Using Time series cross-validation, we compute RMSE on testset (`h=12`). We would have to pick the model with the lowest RMSE on test set as our final model if we had more than 1 model to compare. In our case, since we only have 1 model left after Ljung test, we have no choice but to pick seasonal ARIMA model as our final choice. Cross-validation test shows that RMSE on test is around 720k when RMSE on training is around 589k. We can conclude the model is not necessarily overfitted. Given that MAPE on training is less than 7, it is not a suprising result.

```
FALSE [1] "RMSE - Train: 589381.7 ; RMSE - Test: 725175"
```

3 Part C: Waterflow

Instructions: 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 test appropriate assumptions and forecast a week forward with confidence bands (80 and 95%). Add these to your existing files above - clearly labeled.

3.1 Exploration

Because of the disparities in the data some grooming was necessary:

Pipe one:
* 1000 observations * No missing values * Multiple reading within each hour
* 9-days of data

Pipe Two
* 100 Observations
* No missing values * Single reading on the hour * 41-days of data

For Pipe One, representing 9-days of water flow rate measurements multiple samples per hour, a mean of all rates in the hour was taken and labeled with the whole-hour at the beggining of the period (floor hour) to align with the hourly readings from Pipe Two. After aggregating, there were only 236 observations (spanning 9-days) of pipe one and still 1000 observations (spanning 41-days) from Pipe Two.

Both data sets posed an interesting conundrum. With two possible ways of handling it:

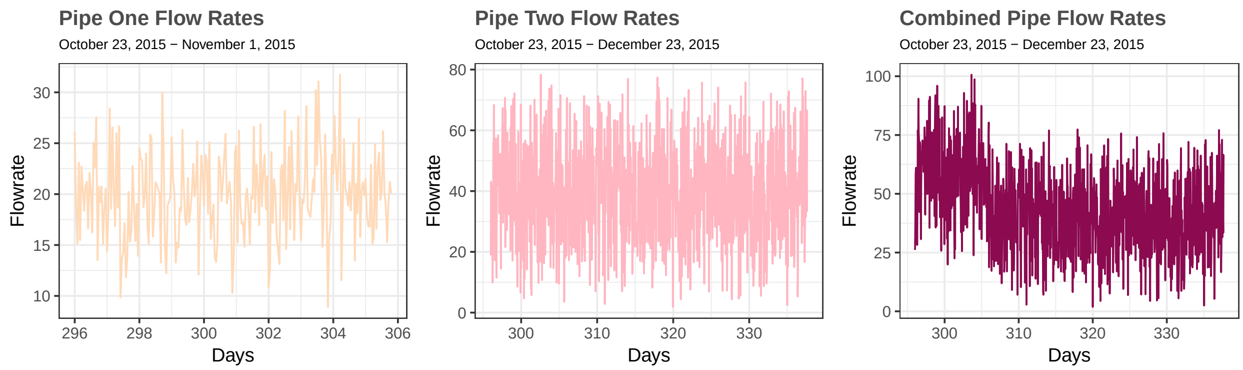
1. Merge the files, and use only 236 observations.

- All forecasts would be based on the combined data.
- This would mean making 168 forecasts with only 236 data-points prior.
- All forecasts would be starting November 1, instead of from the end of data: December 3.

2. Merge the files and use the whole set to make predictions.

- We would have 100 observations to model prior to forecasts.
- 236 of the observations would be be different from the remaining 764, which could both alter the model type and forecast.
- We would be forecasting from the natural ending of the Pipe Two readings

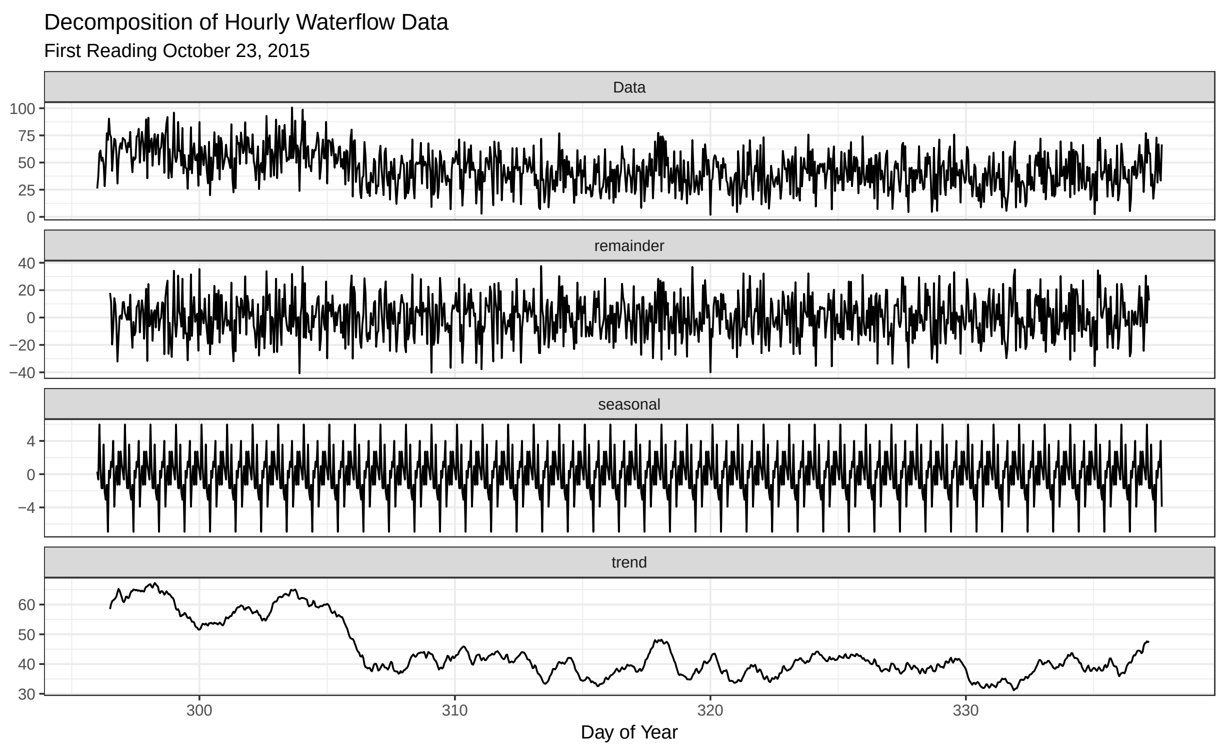
Because it was concievable that there might be a daily periodicity, it was important to have a frequency of 24, which made numbering by day of year and grooming the time series to start on the 7081 hour aligning with October 23 01:00 AM.



3.2 Evaluation

3.2.1 Decomposition

It is clear from the combined plot that there is a pretty notable change in the trend when the readings from Pipe One wane. Let's look at the decomposed seriesand see if it gives us some insight into a good model.



From the decomposition, the appears to be a seasonal component in agreement with the assessment that there might be a daily flowrate periodicity. Also, as expected, around day 306 where Pipe One flow rates go silent there is a trend down and then relatively flat trend thereafter.

3.2.2 Estimating Stationarity

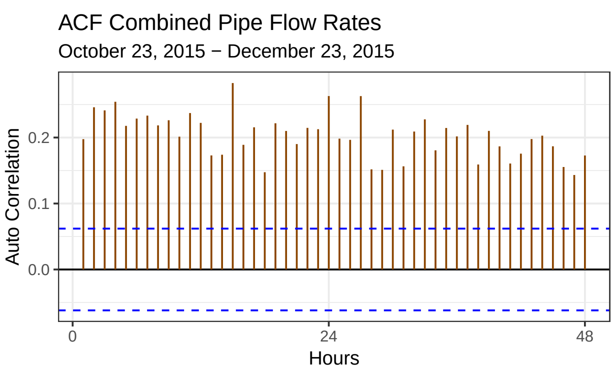
Number of Estimated Differences: 1

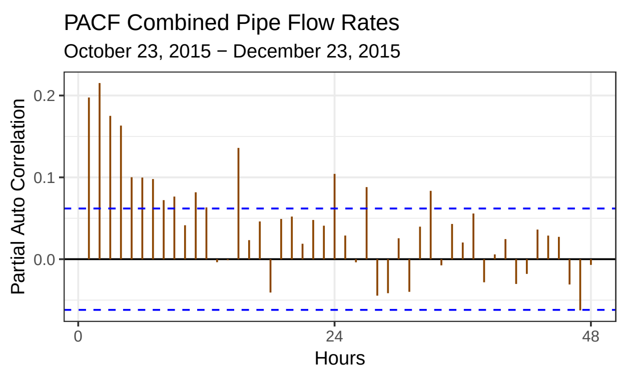
```
FALSE
FALSE   Augmented Dickey-Fuller Test
FALSE
FALSE data:  ws
FALSE Dickey-Fuller = -6.4409, Lag order = 9, p-value = 0.01
FALSE alternative hypothesis: stationary
```

Here we have contradictory esitmates, `ndiffs()` suggests a difference of 1, and the augmented dicky fuller test suggests that we are stationary as-is. An `auto.arima()` may give us a reasonable starting place.

3.2.3 Estimating Orders for ARIMA

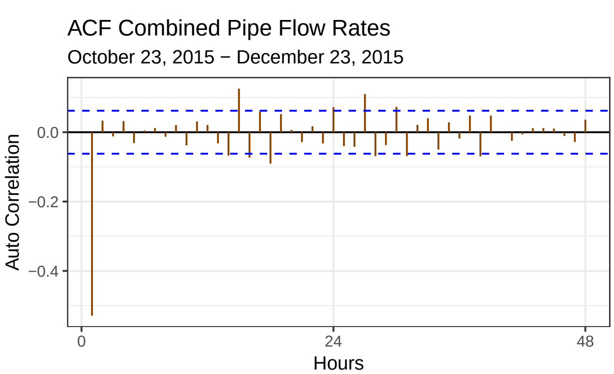
3.2.3.1 Interpreting the ACF and PACF

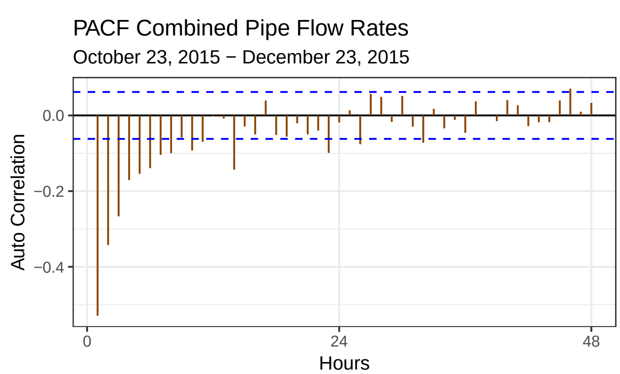




The ACF remain wholly above the critical threshold, so will likely require differencing as suggested by the `ndiffs()`, in looking at the PACF, there is some abiguity caused by the needed differencing, but after the intial trend down below the critical threshold, there is definitely a slight spike at 24, which would suggest there may indeed by a daily period or season we need to account for in our forecast.

3.2.3.2 Differenced ACF





A final ACF of the differenced data was done to ensure that a seconf first-order difference was not needed; thus we assume $d = 1$, a but it was not so clear about the appropriate value of $q4 should it be 5? , so `auto.arima()` is in order to help iterate up on the likely best starting place

3.3 Modeling

The `auto.arima()` function was used in model selection. Using a Box-Cox lambda value to normalize the data may make $\lambda = .931552$. Because models can vary a lot based on the selection criterion, both BIC and AIC models were run, using lambda, to estimate a good starting place. We included the transformations in the model (instead of doing it outside the model), because we are using the ARIMA function to difference the data automatically allow more constiency and flexibility in testing other model orders.

The *AICc* chose a seasonal ARIMA of the following order:

$ARIMA(1,1,3)(0,0,1)[24]$ *AIC=7359.84 AICc=7359.9 BIC=7384.38*

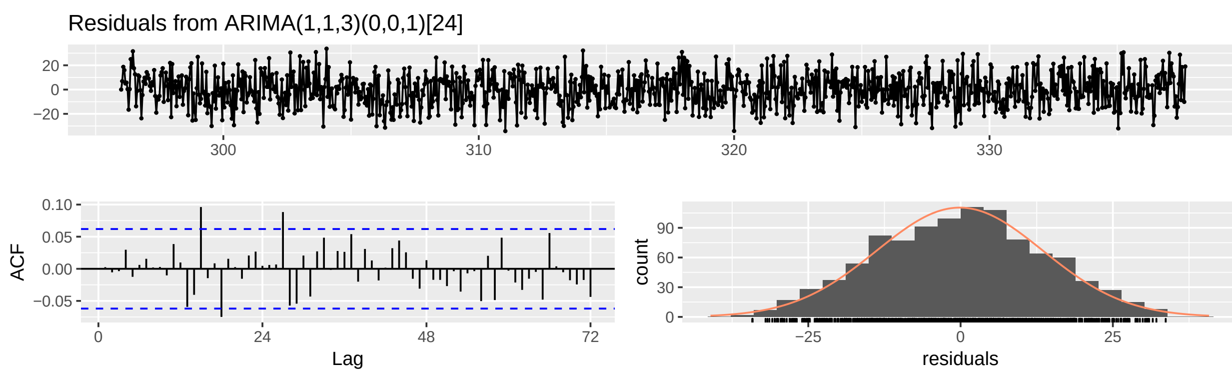
The *BIC* chose a non-seasonal ARIMA model as follows:

$ARIMA(2,1,1)$ *AIC=8082.22 AICc=8082.26 BIC=8101.85*

In both cases, the arima estimated that there needed to be differencing which was supported by `ndiffs()` and our ACF & PACF plots.

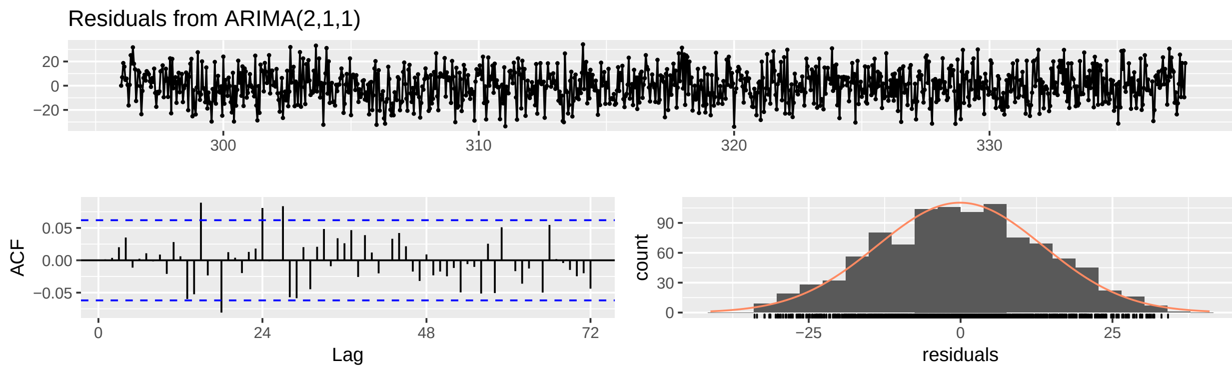
In comparing the two forecasts, for these automated models, they both degrade toward the series mean pretty quickly, however, the AICc model makes forecasts which consider the variation of the model a bit better before it levels out. So we decided to explore this model and see if we could tune it to provide more robust predictions

AIC $ARIMA(1,1,3)(0,0,1)[24]$ Residual Plots




```
FALSE
FALSE   Ljung-Box test
FALSE
FALSE data:  Residuals from ARIMA(1,1,3)(0,0,1)[24]
FALSE Q* = 57.362, df = 43, p-value = 0.07027
FALSE
FALSE Model df: 5.   Total lags used: 48
```

BIC $ARIMA(2,1,1)$ Residual Plots



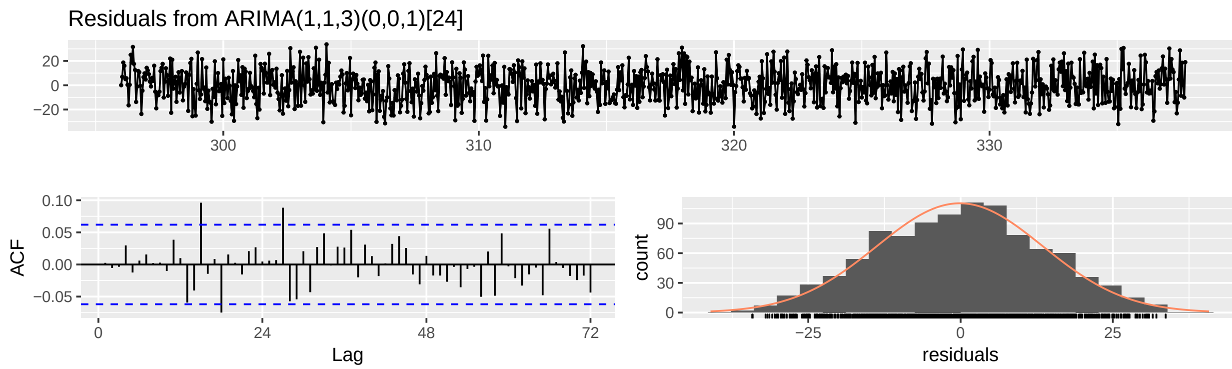

```
FALSE
FALSE   Ljung-Box test
FALSE
FALSE data:  Residuals from ARIMA(2,1,1)
FALSE Q* = 64.403, df = 45, p-value = 0.03029
FALSE
FALSE Model df: 3.   Total lags used: 48
```

3.3.1 Interpreting `auto.arima()`

In looking at the AICc and BIC ARIMA models, the both appear to be relatively white-noisy with no autocorrelation on the first or 24th observations, with relatively normal residuals. However, in looking at the Ljung-Box test for independence, it is clear that the Seasonal $ARIMA(1,1,3)(0,0,1)[24]$ is independent, where the $ARIMA(2,1,1)$ is not, thus reaffirming the lingering suspicion that thee is unaccounted for seasonal variation in the model requiring a seasona MA(1) to rectify. To be sure that the best model has been found, p & q as well as Q will be varied to see if a slight modification improves the performance of the model.

3.3.2 Manual ARIMA testing

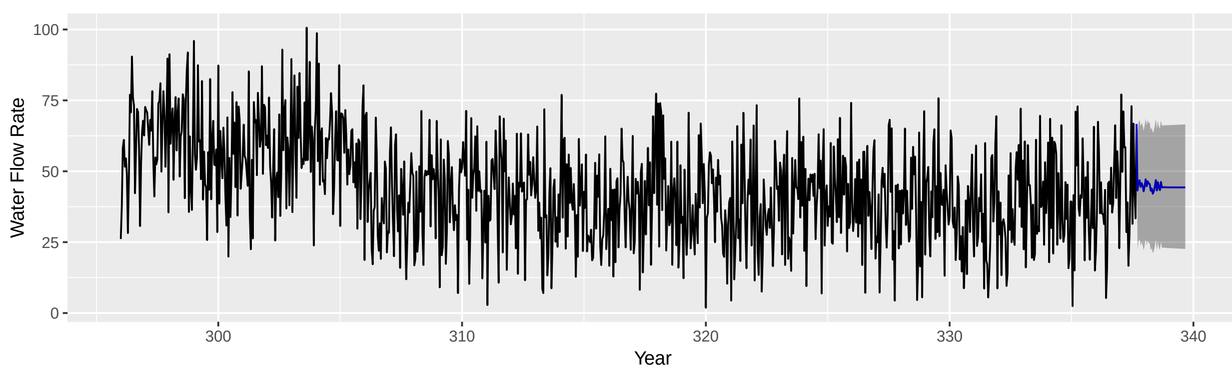
```
FALSE Series: ws
FALSE ARIMA(1,1,3)(0,0,1)[24]
FALSE Box Cox transformation: lambda= 0.9531552
FALSE
FALSE Coefficients:
FALSE          ar1      ma1     ma2      ma3    sma1
FALSE       0.7602  -1.7578  0.8286  -0.0614  0.0833
FALSE s.e.  0.1857   0.1874  0.1886   0.0324  0.0320
FALSE
FALSE sigma^2 estimated as 187:  log likelihood=-4033.28
FALSE AIC=8078.56   AICc=8078.64   BIC=8108
```




```
FALSE
FALSE 	Ljung-Box test
FALSE
FALSE data:  Residuals from ARIMA(1,1,3)(0,0,1)[24]
FALSE Q* = 47.142, df = 31, p-value = 0.03174
FALSE
FALSE Model df: 5.   Total lags used: 36
```

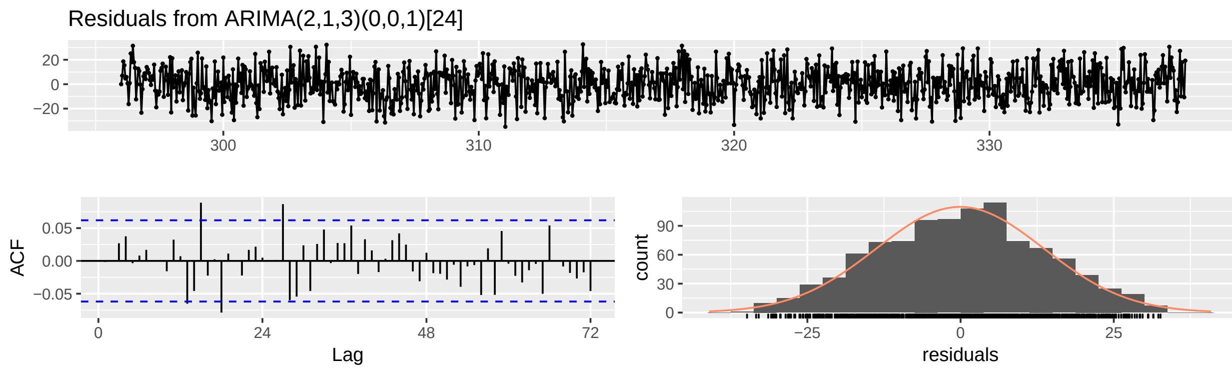
3.4 Forecast

3.4.1 $ARIMA(1,1,3)(0,0,1)[24]$



3.4.2 $ARIMA(2,1,3)(0,0,1)[24]$

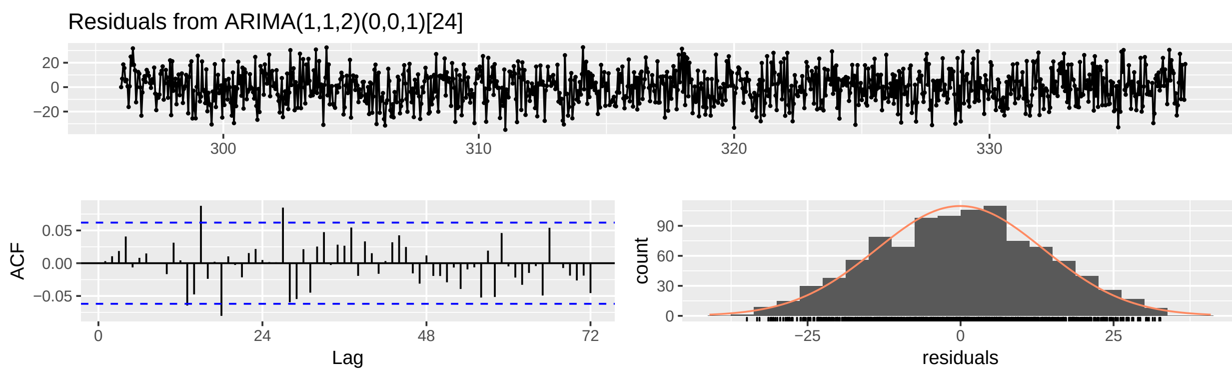
```
FALSE Series: ws
FALSE ARIMA(2,1,3)(0,0,1)[24]
FALSE Box Cox transformation: lambda= 0.9531552
FALSE
FALSE Coefficients:
FALSE           ar1     ar2      ma1      ma2     ma3    sma1
FALSE       -0.1435  0.1884  -0.8478  -0.2709  0.1621  0.0798
FALSE s.e.      NaN  0.5408      NaN   0.6069  0.5320  0.0318
FALSE
FALSE sigma^2 estimated as 187.5:  log likelihood=-4034.02
FALSE AIC=8082.05   AICc=8082.16   BIC=8116.4
```




```
FALSE
FALSE 	Ljung-Box test
FALSE
FALSE data:  Residuals from ARIMA(2,1,3)(0,0,1)[24]
FALSE Q* = 48.506, df = 30, p-value = 0.01764
FALSE
FALSE Model df: 6.   Total lags used: 36
```

This Ljung-Box shows unexplained variances in the residuals indicating that this model is not yet fully realized and inferior to the Seasonal $ARIMA(1,1,3)(0,0,1)[24]$.

```
FALSE Series: ws
FALSE ARIMA(1,1,2)(0,0,1)[24]
FALSE Box Cox transformation: lambda= 0.9531552
FALSE
FALSE Coefficients:
FALSE           ar1      ma1      ma2    sma1
FALSE       -0.2655  -0.7307  -0.2104  0.0790
FALSE s.e.   0.9490   0.9533   0.9121  0.0318
FALSE
FALSE sigma^2 estimated as 187.1:  log likelihood=-4034.08
FALSE AIC=8078.16   AICc=8078.22   BIC=8102.7
```

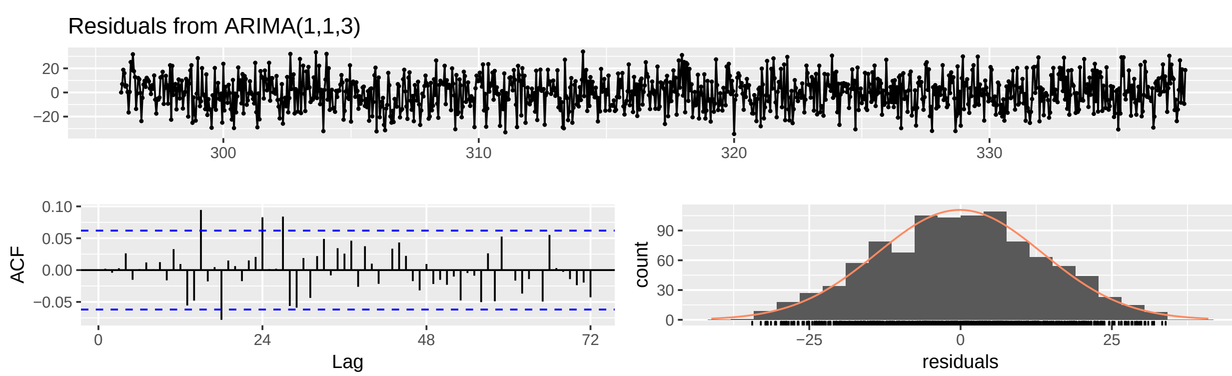



```
FALSE
FALSE 	Ljung-Box test
FALSE
FALSE data:  Residuals from ARIMA(1,1,2)(0,0,1)[24]
```

```
FALSE Q* = 47.963, df = 32, p-value = 0.03467
FALSE
FALSE Model df: 4.   Total lags used: 36
```

This Ljung-Box also shows unexplained variances in the residuals indicating that this model is not yet fully realized and inferior to the Seasonal $ARIMA(1,1,2)(0,0,1)[24]$.

```
FALSE Series: ws
FALSE ARIMA(1,1,3)
FALSE Box Cox transformation: lambda= 0.9531552
FALSE
FALSE Coefficients:
FALSE          ar1      ma1     ma2      ma3
FALSE       0.6792  -1.6742  0.7437  -0.0553
FALSE s.e.  0.2923   0.2930  0.2903   0.0330
FALSE
FALSE sigma^2 estimated as 188.1:  log likelihood=-4036.63
FALSE AIC=8083.27   AICc=8083.33   BIC=8107.81
```

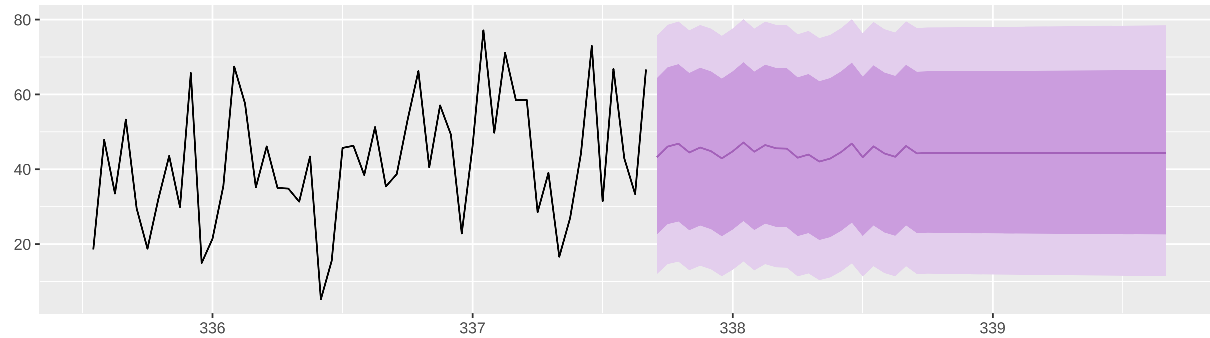



```
FALSE
FALSE   Ljung-Box test
FALSE
FALSE data:  Residuals from ARIMA(1,1,3)
FALSE Q* = 53.61, df = 32, p-value = 0.009708
FALSE
FALSE Model df: 4.   Total lags used: 36
```

This Ljung-Box also shows unexplained variances in the residuals indicating that this model is not yet fully realized and inferior to the Seasonal $ARIMA(1,1,3)$.

3.4.3 Accepting the `auto.arima()`

Given that the other models show unexplained variance in the residuals, the final predictions will be made using the AICc recommended model of $ARIMA(1,1,3)(0,0,1)[24]$.



3.4.4 Forecast Accuracy

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	0.0015679	16.27402	13.23093	-28.76247	50.34448	0.7489308	0.0014339

3.5 Summary

Ultimately this model is marginally useful as seen by the Mean Absolute Percentage of Error which reveals that the average percentage each forecast is off by is around 50%. In looking at the graph of the forecast above, which is the last 150 points in the time series and the forecasted points, you can see this as the predictions lightly modulate around the mean and deteriorate to it pretty quickly.

In looking at the original decomposition, there very little trend, a lot of seasonality, is a pretty substatial amount of random noise, which is not considered in the model, and is responsible for the majority of the error in this model, as white noise is never predictable.

Appendix A

ARIMA Model Summary

ATM1:

```
FALSE Series: ATM1_ts
FALSE ARIMA(0,0,2)(0,1,1)[7]
FALSE Box Cox transformation: lambda= 0.2584338
FALSE
FALSE Coefficients:
FALSE          ma1      ma2     sma1
FALSE       0.1085  -0.1089  -0.6425
FALSE s.e.  0.0524   0.0521   0.0431
FALSE
FALSE sigma^2 estimated as 1.726:  log likelihood=-606.1
FALSE AIC=1220.2   AICc=1220.32   BIC=1235.72
```

ATM2:

```
FALSE Series: ATM2_ts
FALSE ARIMA(2,0,2)(0,1,1)[7]
FALSE Box Cox transformation: lambda= 0.661752
FALSE
FALSE Coefficients:
FALSE           ar1      ar2     ma1     ma2     sma1
FALSE       -0.4238  -0.8978  0.4766  0.7875  -0.7064
FALSE s.e.   0.0592   0.0473  0.0883  0.0608   0.0417
FALSE
FALSE sigma^2 estimated as 38.94:  log likelihood=-1162.96
FALSE AIC=2337.93   AICc=2338.17   BIC=2361.21
```

ATM4:

```
FALSE Series: ATM4_ts
FALSE ARIMA(0,0,2)(0,1,1)[7]
FALSE Box Cox transformation: lambda= 0.2328582
FALSE
FALSE Coefficients:
FALSE          ma1      ma2     sma1
FALSE       0.1095  -0.1088  -0.6474
FALSE s.e.  0.0524   0.0523   0.0420
FALSE
FALSE sigma^2 estimated as 1.439:  log likelihood=-573.5
FALSE AIC=1154.99   AICc=1155.11   BIC=1170.52
```

Point Forecasts

Table 3.1: ATM Mean Point Forecast

Date	ATM1	ATM2	ATM3	ATM4
2010-05-01	87	66	88	87
2010-05-02	101	71	88	101
2010-05-03	74	11	88	74
2010-05-04	4	2	88	4
2010-05-05	100	98	88	100
2010-05-06	79	89	88	79
2010-05-07	86	66	88	86
2010-05-08	87	66	88	87
2010-05-09	100	71	88	100
2010-05-10	74	11	88	74
2010-05-11	4	2	88	4
2010-05-12	100	98	88	100
2010-05-13	79	89	88	79
2010-05-14	86	66	88	86
2010-05-15	87	66	88	87
2010-05-16	100	71	88	100
2010-05-17	74	11	88	74
2010-05-18	4	2	88	4
2010-05-19	100	98	88	100
2010-05-20	79	89	88	79
2010-05-21	86	66	88	86
2010-05-22	87	66	88	87
2010-05-23	100	71	88	100
2010-05-24	74	11	88	74
2010-05-25	4	2	88	4
2010-05-26	100	98	88	100
2010-05-27	79	89	88	79
2010-05-28	86	66	88	86
2010-05-29	87	66	88	87
2010-05-30	100	71	88	100
2010-05-31	74	11	88	74

R Script

```
# Load data
atm_data <- read_excel("data/ATM624Data.xlsx")

# clean dataframe
atm <- atm_data %>%
  # create wide dataframe
  spread(ATM, Cash) %>%
  # remove NA column using function from janitor package
  remove_empty(which = "cols") %>%
  # filter unobserved values from May 2010
  filter(DATE < as.Date("2010-05-01")) %>% arrange(DATE)

atm$ATM2[is.na(atm$ATM2)] <- mean(atm$ATM2, na.rm = TRUE) ## remove NA
atm$ATM4[which.max(atm$ATM4)] <- mean(atm$ATM4, na.rm = TRUE) ## remove outlier

# create TS with weekly frequency & subset data
atm_ts <- atm %>% select(-DATE) %>% ts(start=1,  frequency = 7)
ATM1_ts <- atm_ts[,1]; ATM2_ts <- atm_ts[,2]; ATM3_ts <- atm_ts[,3]; ATM4_ts <- atm_ts[,4]

#unit root test:
ATM1_ur <-ur.kpss(ATM1_ts); ATM2_ur <-ur.kpss(ATM2_ts); ATM4_ur <-ur.kpss(ATM4_ts)
ATM1d_ur <-ur.kpss(diff(ATM1_ts, lag=7)); ATM2d_ur <-ur.kpss(diff(ATM2_ts, lag=7))
ATM4d_ur <-ur.kpss(diff(ATM4_ts, lag=7))

# AUTO.ARIMA function; set D=1 for seasonal differencing
ATM1_AA <-auto.arima(ATM1_ts, D = 1, lambda = "auto", approximation = F, stepwise = T)
ATM2_AA <-auto.arima(ATM2_ts, D = 1, lambda = "auto", approximation = F, stepwise = T)
ATM4_AA <-auto.arima(ATM4_ts, D = 1, lambda = "auto", approximation = F, stepwise = T)

# Forecast Results
ATM1_fc <- forecast(ATM1_AA,h=31); ATM2_fc <- forecast(ATM2_AA,h=31)
ATM3_fc <- meanf(ATM3_ts[ATM3_ts > 0], h=31); ATM4_fc <- forecast(ATM4_AA,h=31)

# Prepare dataframe for ATM3 mean forcast plotting
ATM3_plotdata_fc <- cbind(seq(from = 366, to = 396), ATM3_fc[[5]], ATM3_fc[[6]],
                          ATM3_fc[[7]]) %>% as.data.frame()

colnames(ATM3_plotdata_fc) <- c('Date', 'Point Forecast',
                                'Lo 80', 'Lo 95', 'Hi 80', 'Hi 95')
ATM3_plotdata <- ATM3_ts %>% fortify() %>% select(-Index) %>% rename(Cash = Data) %>%
  mutate(Date = as.numeric(row.names(.))) %>% select(Date, Cash) %>%
  full_join(ATM3_plotdata_fc, by = 'Date')

#Revert results back into original form
date <- as.character(seq(as.Date('2010-05-01'), length.out=31, by=1))
ATM_FC <-  cbind("Date"=date, "ATM1"=ATM1_fc$mean, "ATM2"=ATM2_fc$mean,
                 "ATM3"=ATM3_fc$mean, "ATM4"=ATM4_fc$mean) %>%
  as.data.frame() %>% gather("ATM", "cash", -Date) %>%
  mutate(Date = as.Date(as.character(Date)), Cash = round(as.numeric(cash))) %>%
  select(-cash)
```

```r
write_csv(ATM_FC, path = "forecasts/ATM_all_forecast.csv")
```

Appendix B

Model Summary

ARIMA:

```
FALSE Series: ts_data_o
FALSE ARIMA(3,0,2)(2,1,0)[12] with drift
FALSE
FALSE Coefficients:
FALSE           ar1      ar2     ar3     ma1     ma2     sar1     sar2     drift
FALSE       -0.5606  -0.2216  0.3284  0.8902  0.4827  -0.7249  -0.4152  9018.405
FALSE s.e.   0.3992   0.3382  0.0960  0.4120  0.4551   0.0797   0.0841  3027.685
FALSE
FALSE sigma^2 estimated as 387762785879:  log likelihood=-2657.12
FALSE AIC=5332.24   AICc=5333.3   BIC=5360.97
FALSE
FALSE Training set error measures:
FALSE                     ME     RMSE      MAE        MPE     MAPE      MASE
FALSE Training set -8455.077 589381.7 427752.5 -0.7944782 6.475365 0.6904053
FALSE                      ACF1
FALSE Training set 0.0006090194
```

STL - MNN:

```
FALSE
FALSE Forecast method: STL +  ETS(M,N,N)
FALSE
FALSE Model Information:
FALSE ETS(M,N,N)
FALSE
FALSE Call:
FALSE  ets(y = x, model = etsmodel, allow.multiplicative.trend = allow.multiplicative.trend)
FALSE
FALSE   Smoothing parameters:
FALSE     alpha = 0.1159
FALSE
FALSE   Initial states:
FALSE     l = 6317745.8917
FALSE
FALSE   sigma:  0.097
FALSE
FALSE      AIC     AICc      BIC
FALSE 6139.631 6139.758 6149.403
FALSE
FALSE Error measures:
FALSE                    ME     RMSE      MAE         MPE     MAPE      MASE
FALSE Training set 56926.03 633571.7 460713.4 -0.03288687 6.945185 0.7436052
FALSE                    ACF1
```

```
FALSE Training set 0.2570241
FALSE
FALSE Forecasts:
FALSE          Point Forecast   Lo 80    Hi 80   Lo 95    Hi 95
FALSE Jan 2014        8992609 8049591  9935628 7550387 10434831
FALSE Feb 2014        7908116 6958724  8857508 6456146  9360086
FALSE Mar 2014        7079434 6123709  8035158 5617779  8541088
FALSE Apr 2014        6435209 5473193  7397225 4963933  7906486
FALSE May 2014        6161593 5193326  7129860 4680756  7642430
FALSE Jun 2014        7728705 6754226  8703185 6238368  9219043
FALSE Jul 2014        8837980 7857327  9818633 7338201 10337759
FALSE Aug 2014        9376841 8390053 10363630 7867678 10886004
FALSE Sep 2014        8601001 7608114  9593888 7082511 10119490
FALSE Oct 2014        6670419 5671470  7669368 5142658  8198180
FALSE Nov 2014        6035845 5030870  7040821 4498868  7572822
FALSE Dec 2014        7189087 6178120  8200053 5642947  8735226
```

STL - MAdN:

```
FALSE
FALSE Forecast method: STL +  ETS(M,Ad,N)
FALSE
FALSE Model Information:
FALSE ETS(M,Ad,N)
FALSE
FALSE Call:
FALSE  ets(y = x, model = etsmodel, damped = TRUE, allow.multiplicative.trend = allow.multiplicative.tr
FALSE
FALSE   Smoothing parameters:
FALSE     alpha = 0.1233
FALSE     beta  = 0.0001
FALSE     phi   = 0.8
FALSE
FALSE   Initial states:
FALSE     l = 5615471.7851
FALSE     b = 173606.4508
FALSE
FALSE   sigma:  0.0972
FALSE
FALSE      AIC     AICc      BIC
FALSE 6143.452 6143.906 6162.997
FALSE
FALSE Error measures:
FALSE                    ME     RMSE      MAE         MPE     MAPE      MASE
FALSE Training set 54337.68 631081.9 458777.5 -0.07364717 6.937249 0.7404807
FALSE                   ACF1
FALSE Training set 0.2528558
FALSE
FALSE Forecasts:
FALSE          Point Forecast   Lo 80    Hi 80   Lo 95    Hi 95
FALSE Jan 2014        9007707 8060947  9954467 7559763 10455651
FALSE Feb 2014        7923348 6969325  8877372 6464295  9382401
FALSE Mar 2014        7094774 6133536  8056011 5624687  8564860
FALSE Apr 2014        6450635 5482232  7419038 4969591  7931680
```

```
FALSE May 2014        6177088 5201569  7152607 4685160  7669016
FALSE Jun 2014        7744256 6761668  8726843 6241518  9246993
FALSE Jul 2014        8853574 7863967  9843182 7340100 10367048
FALSE Aug 2014        9392471 8395890 10389052 7868332 10916609
FALSE Sep 2014        8616658 7613151  9620166 7081926 10151391
FALSE Oct 2014        6686100 5675711  7696488 5140843  8231356
FALSE Nov 2014        6051544 5034319  7068769 4495832  7607255
FALSE Dec 2014        7204799 6180782  8228817 5638700  8770899
```

ets - MNM:

```
FALSE
FALSE Forecast method: ETS(M,N,M)
FALSE
FALSE Model Information:
FALSE ETS(M,N,M)
FALSE
FALSE Call:
FALSE  ets(y = ts_data_o)
FALSE
FALSE   Smoothing parameters:
FALSE     alpha = 0.1428
FALSE     gamma = 0.2119
FALSE
FALSE   Initial states:
FALSE     l = 6189149.8743
FALSE     s = 0.8984 0.7596 0.938 1.2229 1.2597 1.2396
FALSE            1.0059 0.7638 0.8078 0.8864 1.0269 1.191
FALSE
FALSE   sigma:  0.0967
FALSE
FALSE      AIC     AICc      BIC
FALSE 6144.033 6146.760 6192.895
FALSE
FALSE Error measures:
FALSE                    ME     RMSE      MAE         MPE     MAPE      MASE
FALSE Training set 45241.77 628252.5 481520.9 -0.04000239 7.277118 0.7771892
FALSE                  ACF1
FALSE Training set 0.1927438
FALSE
FALSE Forecasts:
FALSE          Point Forecast   Lo 80    Hi 80   Lo 95    Hi 95
FALSE Jan 2014        9917654 8689211 11146096 8038913 11796394
FALSE Feb 2014        8522973 7456477  9589469 6891908 10154038
FALSE Mar 2014        7012478 6126191  7898765 5657019  8367937
FALSE Apr 2014        6208601 5416196  7001006 4996722  7420480
FALSE May 2014        5928833 5164834  6692832 4760398  7097269
FALSE Jun 2014        7840532 6820624  8860440 6280717  9400347
FALSE Jul 2014        9115823 7919004 10312642 7285446 10946200
FALSE Aug 2014        9648549 8370229 10926869 7693527 11603571
FALSE Sep 2014        8553364 7409986  9696742 6804718 10302010
FALSE Oct 2014        6266745 5421655  7111835 4974291  7559199
FALSE Nov 2014        5938289 5130560  6746017 4702975  7173603
FALSE Dec 2014        8020901 6920610  9121192 6338151  9703651
```

R Script

```r
library(readxl)
library(tidyverse)
library(forecast)
library(imputeTS)
library(tsoutliers)

# load data
power_data <- read_excel("data/ResidentialCustomerForecastLoad-624.xlsx")
# Time Series
ts_data <- ts(power_data$KWH, frequency = 12, start = c(1998,1))
# Missing value imputation
ts_data <- na_interpolation(ts_data)
# STL decomposition
stl1 <- stl(ts_data, s.window = 'periodic')
# Handling outlier
outlier_func <- tsoutliers(ts_data, iterate = 2, lambda = "auto")

# Time Series - After outlier and imputation handeled
ts_data_o <- ts_data  # Let's treate outlier handled data seperatly for Modelling part.
ts_data_o[outlier_func$index] <- outlier_func$replacements

# Model#1: ARIMA
arima_auto <- auto.arima(ts_data_o)
arima_fc <- forecast(arima_auto, h=12)

# Model #2: STL (no-demped) - MNN
stl_ndemp <- stlf(ts_data_o, s.window = "periodic", robust=TRUE, h = 12)

# Model #2-2: STL (demped) - MAdN
stl_demp <- stlf(ts_data_o, damped=TRUE, s.window = "periodic", robust=TRUE, h = 12)

# Model #3: ets - MNM
ets_auto <- ets(ts_data_o)
ets_model <- forecast(ets_auto, h=12)

# tsCv - ARIMA -> it takes so much time. I got the results and saved them
##arima_cv <- function(x, h){forecast(Arima(x, order = c(3, 0, 2),
## seasonal = c(2, 1, 0), include.drift = TRUE), h=h)}
##e <- tsCV(ts_data_o, arima_cv, h=12)

# RMSEs -> tsCV takes lot of time to process so just saved the output
#rmse_train_arima <- arima_auto[2]
#rmse_test_arima <- sqrt(mean(e^2, na.rm=TRUE))
rmse_train_arima <- 589381.7
rmse_test_arima <- 725175

# Save output
write.csv(arima_fc, file="forecasts/POWER_ARIMA_FC.csv")
```

Appendix C

Sample Forecasts

Table 3.2: First few predictions in the set

DateTime	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
2015-12-03 17:00:00	43.21837	22.59441	64.33311	12.00034	75.65243
2015-12-03 18:00:00	46.07958	25.37341	67.24682	14.70394	78.58795
2015-12-03 19:00:00	46.85016	26.06919	68.08732	15.35468	79.46457
2015-12-03 20:00:00	44.49638	23.73897	65.73546	13.06315	77.11903
2015-12-03 21:00:00	45.83029	25.00018	67.13008	14.27275	78.54342
2015-12-03 22:00:00	44.85032	24.01864	66.16308	13.30217	77.58566
2015-12-03 23:00:00	42.92705	22.12687	64.23068	11.45169	75.65293
2015-12-04 00:00:00	44.79836	23.91958	66.16114	13.18081	77.61089
2015-12-04 01:00:00	47.17329	26.20684	68.60103	15.39770	80.08059
2015-12-04 02:00:00	44.70609	23.78935	66.10979	13.03325	77.58190
2015-12-04 03:00:00	46.48881	25.50281	67.94446	14.69153	79.44061
2015-12-04 04:00:00	45.62158	24.64210	67.08023	13.84406	78.57995
2015-12-04 05:00:00	45.52709	24.53307	67.00208	13.72907	78.51085
2015-12-04 06:00:00	43.10639	22.16724	64.55376	11.42233	76.05335
2015-12-04 07:00:00	43.96360	22.98208	65.44444	12.20441	76.96005
2015-12-04 08:00:00	42.07391	21.13451	63.53552	10.40558	75.04543
2015-12-04 09:00:00	42.87840	21.89785	64.37233	11.13642	75.89768
2015-12-04 10:00:00	44.60108	23.55269	66.14421	12.73392	77.69198
2015-12-04 11:00:00	46.89847	25.76875	68.50006	14.88240	80.07419
2015-12-04 12:00:00	43.23698	22.19835	64.78723	11.40350	76.34217
2015-12-04 13:00:00	46.15105	25.01095	67.77192	14.12809	79.35815
2015-12-04 14:00:00	44.24754	23.14728	65.84951	12.30812	77.42997
2015-12-04 15:00:00	43.35173	22.26320	64.95292	11.44256	76.53515
2015-12-04 16:00:00	46.23353	25.04461	67.90461	14.13691	79.51781
2015-12-04 17:00:00	44.25878	22.97866	66.04954	12.05224	77.73211
2015-12-04 18:00:00	44.38901	23.08956	66.19841	12.15194	77.89075
2015-12-04 19:00:00	44.37188	23.05269	66.20224	12.10576	77.90597
2015-12-04 20:00:00	44.35886	23.02043	66.20961	12.06437	77.92439
2015-12-04 21:00:00	44.34896	22.99166	66.21966	12.02661	77.94527
2015-12-04 22:00:00	44.34144	22.96555	66.23177	11.99162	77.96801

R-Script

```
library(tidyverse)
library(readxl)
library(fpp2)
library(forecast)
library(lubridate)
library(psych)
#library(xlsx)
options(scipen = 999)

# Reading Data
waterflow_1 <- read_excel("data/Waterflow_Pipe1.xlsx")
waterflow_2 <- read_excel("data/Waterflow_Pipe2.xlsx")

# Writing original data to submission file
#file ='forecasts/water-pipes.xlsx'
#write.xlsx(waterflow_1, file =  file , sheetName ="Waterflow Pipe 1", col.names = TRUE, row.names = TR
#write.xlsx(waterflow_2, file=file, sheetName = "Waterflow Pipe 2", col.names = TRUE, row.names = TRUE,

# Grooming, aligning dates and aggregating Data
waterflow_1<-waterflow_1 %>%
    mutate(DateTime = as.POSIXct(DateTime))%>%
    group_by(hour=floor_date(DateTime, "hour")) %>%
    summarize(WaterFlow=mean(WaterFlow))


waterflow_2<-waterflow_2 %>%
    mutate(DateTime = as.POSIXct(DateTime))%>%
    group_by(hour=floor_date(DateTime, "hour")) %>%
    summarize(WaterFlow=mean(WaterFlow))

# Creating a combined data set
waterflow_all <-merge(waterflow_1, waterflow_2, by = 'hour', all = TRUE)%>%
    mutate(waterflow = rowSums(.[c("WaterFlow.y", "WaterFlow.x")], na.rm = TRUE))%>%
    select(hour, waterflow)

# Converting all Three Data Sets to Time Series
w1<-ts(waterflow_1$WaterFlow ,start=c(1,7081),frequency=24)
w2<-ts(waterflow_2$WaterFlow ,start=c(1,7081),frequency=24)
ws <- ts(waterflow_all$waterflow ,start=c(1,7081),frequency=24)


#Decomposition of Time Series
ws_decomp<- ws%>%
    decompose()%>%
    autoplot()+
    labs(title = "Decomposition of Hourly Waterflow Data",
         subtitle = 'First Reading October 23, 2015',
         x = 'Day of Year')+
    theme_bw()
```

```r
# Checking Differences
ws_diffs<- ws%>%
    ndiffs() #1


# Testing Stationarity
dickie<-tseries::adf.test(ws)

# ACF & PACF

ws_acf <- ggAcf(ws, color = 'darkorange4')+
    labs(title = "ACF Combined Pipe Flow Rates",
         subtitle = 'October 23, 2015 - December 23, 2015',
         y="Auto Correlation", x="Hours")+
    theme_bw()+ theme()

ws_pacf <- ggPacf(ws, color = 'darkorange4')+
    labs(title = "PACF Combined Pipe Flow Rates",
         subtitle = 'October 23, 2015 - December 23, 2015',
         y="Partial Auto Correlation", x="Hours")+
    theme_bw()+ theme()

# Differencesd ACF & PACF

ws_acf_diff <-ggAcf(diff(ws,lag = 1), color = 'darkorange4')+
    labs(title = "ACF Combined Pipe Flow Rates",
         subtitle = 'October 23, 2015 - December 23, 2015',
         y="Auto Correlation", x="Hours")+
    theme_bw()+ theme()

ws_pacf_diff <-ggPacf(diff(ws,lag = 1), color = 'darkorange4')+
    labs(title = "PACF Combined Pipe Flow Rates",
         subtitle = 'October 23, 2015 - December 23, 2015',
         y="Auto Correlation", x="Hours")+
    theme_bw()+ theme()

#Establishing a lambda value for ARIMA transformations
lambda <-  BoxCox.lambda(ws)
#Lambda = 0.9531552


# Auto arima's including season components for AICc and BIC
aic<- auto.arima(ws, seasonal = TRUE, ic = 'aicc', lambda = lambda)

bic<-auto.arima(ws, seasonal = TRUE, ic = 'bic', lambda = lambda )


# Plots of auto.arimas
aic_plot <- auto.arima(ws, seasonal = TRUE, ic = 'aicc', lambda = lambda)%>%
    forecast(h=24*7)%>%
    autoplot() +
    labs(title = "AIC selected ARIMA(1,1,3)(0,0,1)[24] ",
                 subtitle = 'October 23, 2015 - December 23, 2015',
```

```
                y="Flowrate", x="Days")+
    theme_bw()+ theme()


bic_plot<-auto.arima(ws, seasonal = TRUE, ic = 'bic', lambda = lambda )%>%
    forecast(h=24*7)%>%
    autoplot()+
    labs(title = "BIC selected ARIMA(2,1,1)  ",
         subtitle = 'October 23, 2015 - December 23, 2015',
         y="Flowrate", x="Days")+
    theme_bw()+ theme()


# Final AIC from AICc and predictions
final_ws <- Arima(ws, order=c(1,1,3), seasonal=c(0,0,1),lambda=lambda)

preds_ws <-as.data.frame(forecast(final_ws, h = 168))


#Renaming fields for output data
waterflow_all <-waterflow_all%>%
    rename( DateTime = hour,
            WaterFlow = waterflow)
# Formatting forecasts for output data
preds_ws<-preds_ws%>%
    mutate(DateTime = seq(from=as.POSIXct("2015-12-3 17:00", tz="UTC"),
                          to=as.POSIXct("2015-12-10 16:00", tz="UTC"),
                          by="hour") )%>%
    select(DateTime, `Point Forecast`, `Lo 80`,`Hi 80`, `Lo 95`, `Hi 95`)

# Writing forecasts and final data to the 'XLSX' file
#write.xlsx(waterflow_all, file = file, sheetName = "Combined Waterflow", col.names = TRUE, row.names =
#write.xlsx(preds_ws, file =  file , sheetName = "Forecasts", col.names = TRUE, row.names = FALSE, appe
```