DATA 624: Project 1 - Part B

Sang Yoon (Andy) Hwang & Vinicio Haro

October 22, 2019

Contents

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.

Data Exploration and Processing

Explore data. Process as needed.

```
library(tidyverse)
library(scales)
library(readxl)
library(forecast)
library(lubridate)
library(fpp2)
library(ggplot2)
library(forecast)
library(tseries)
library(imputeTS)
library(tsoutliers)
#install.packages('tsoutliers')
```

```
#power_data <- read_excel("data/ResidentialCustomerForecastLoad-624.xlsx")
library (readr)

power="https://raw.githubusercontent.com/vindication09/DATA-624/master/ResidentialCustomerForecastLoad-

partb_data<-read_csv(url(power))

head(partb_data)
```

```
# A tibble: 6 x 3
  CaseSequence `YYYY-MMM`     KWH
         <dbl> <chr>        <dbl>
1          733 1998-Jan   6862583
2          734 1998-Feb   5838198
3          735 1998-Mar   5420658
4          736 1998-Apr   5010364
5          737 1998-May   4665377
6          738 1998-Jun   6467147
```

Transformed data into time-series with freq - 12. We are missing 2008 Sep data point.

```
ts_data <- ts(partb_data$KWH, frequency = 12, start = c(1998,1))
ts_data
```

```
          Jan      Feb      Mar      Apr      May      Jun      Jul
1998  6862583  5838198  5420658  5010364  4665377  6467147  8914755
```

```
FALSE 1999  7183759  5759262  4847656  5306592  4426794  5500901  7444416
FALSE 2000  7068296  5876083  4807961  4873080  5050891  7092865  6862662
FALSE 2001  7538529  6602448  5779180  4835210  4787904  6283324  7855129
FALSE 2002  7099063  6413429  5839514  5371604  5439166  5850383  7039702
FALSE 2003  7256079  6190517  6120626  4885643  5296096  6051571  6900676
FALSE 2004  7584596  6560742  6526586  4831688  4878262  6421614  7307931
FALSE 2005  8225477  6564338  5581725  5563071  4453983  5900212  8337998
FALSE 2006  7793358  5914945  5819734  5255988  4740588  7052275  7945564
FALSE 2007  8031295  7928337  6443170  4841979  4862847  5022647  6426220
FALSE 2008  7964293  7597060  6085644  5352359  4608528  6548439  7643987
FALSE 2009  8072330  6976800  5691452  5531616  5264439  5804433  7713260
FALSE 2010  9397357  8390677  7347915  5776131  4919289  6696292   770523
FALSE 2011  8394747  8898062  6356903  5685227  5506308  8037779 10093343
FALSE 2012  8991267  7952204  6356961  5569828  5783598  7926956  8886851
FALSE 2013 10655730  7681798  6517514  6105359  5940475  7920627  8415321
FALSE           Aug      Sep      Oct      Nov      Dec
FALSE 1998  8607428  6989888  6345620  4640410  4693479
FALSE 1999  7564391  7899368  5358314  4436269  4419229
FALSE 2000  7517830  8912169  5844352  5041769  6220334
FALSE 2001  8450717  7112069  5242535  4461979  5240995
FALSE 2002  8058748  8245227  5865014  4908979  5779958
FALSE 2003  8476499  7791791  5344613  4913707  5756193
FALSE 2004  7309774  6690366  5444948  4824940  5791208
FALSE 2005  7786659  7057213  6694523  4313019  6181548
FALSE 2006  8241110  7296355  5104799  4458429  6226214
FALSE 2007  7447146  7666970  5785964  4907057  6047292
FALSE 2008  8037137       NA  5101803  4555602  6442746
FALSE 2009  8350517  7583146  5566075  5339890  7089880
FALSE 2010  7922701  7819472  5875917  4800733  6152583
FALSE 2011 10308076  8943599  5603920  6154138  8273142
FALSE 2012  9612423  7559148  5576852  5731899  6609694
FALSE 2013  9080226  7968220  5759367  5769083  9606304
```

We impute missing data using TSImpute's interpolation method.

```
ts_data<-na.interpolation(ts_data)
```

Review the cycle of the time series to get an idea of the positions within the cycle.

```
cycle(ts_data)
```

```
FALSE      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
FALSE 1998   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 1999   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2000   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2001   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2002   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2003   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2004   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2005   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2006   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2007   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2008   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2009   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2010   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2011   1   2   3   4   5   6   7   8   9  10  11  12
```

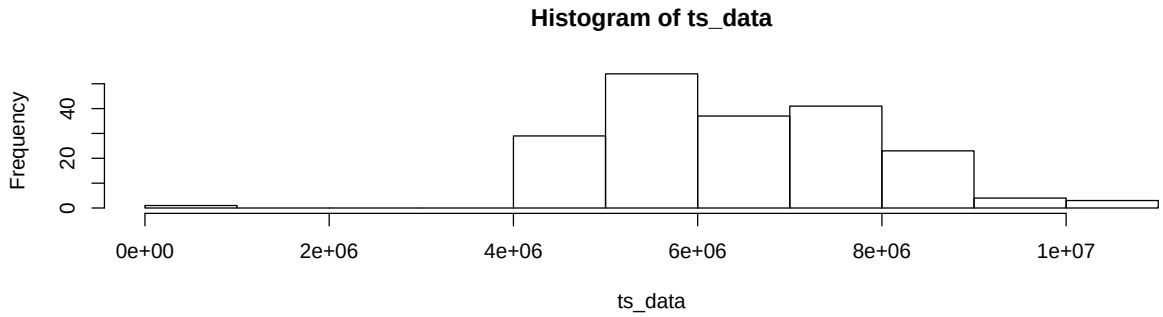
```
FALSE 2012   1   2   3   4   5   6   7   8   9  10  11  12
FALSE 2013   1   2   3   4   5   6   7   8   9  10  11  12
```

Let's do quick EDA on ts_data. Outlier is detected - Min is significantly lower than median. We will handle this outlier. In general, data is fairly normally distributed as mean is not far from median.

```
summary(ts_data)
```

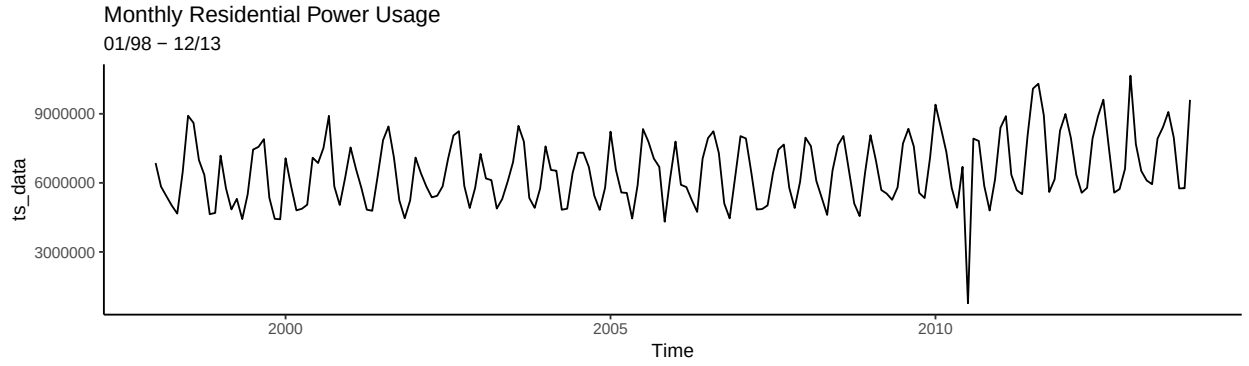
```
FALSE      Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
FALSE    770523  5434539  6314472  6502824  7608792 10655730
```

```
hist(ts_data)
```

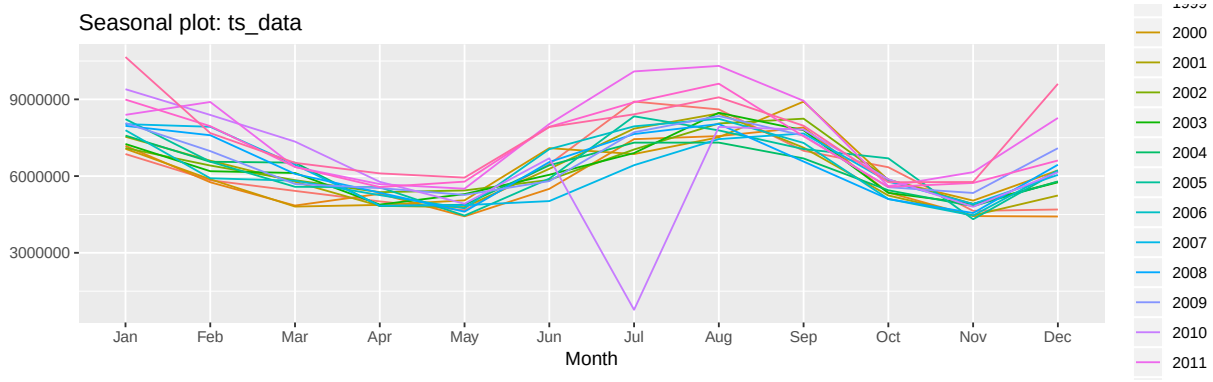


Let's do deeper EDA on ts_data and handle outlier using tsoutliers().

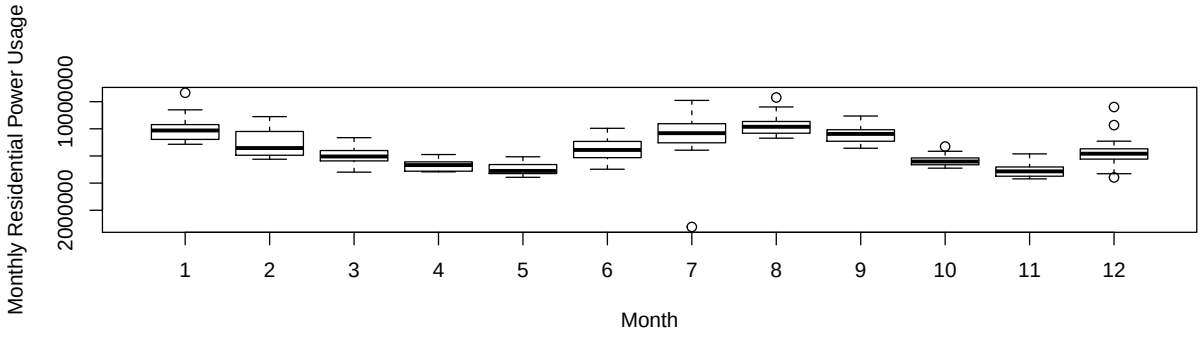
```
#disable scientific notation (ONLY RUN ONCE)
options(scipen = 99999)

autoplot(ts_data) +
labs(title = "Monthly Residential Power Usage", subtitle = "01/98 - 12/13")+
theme_classic();
```



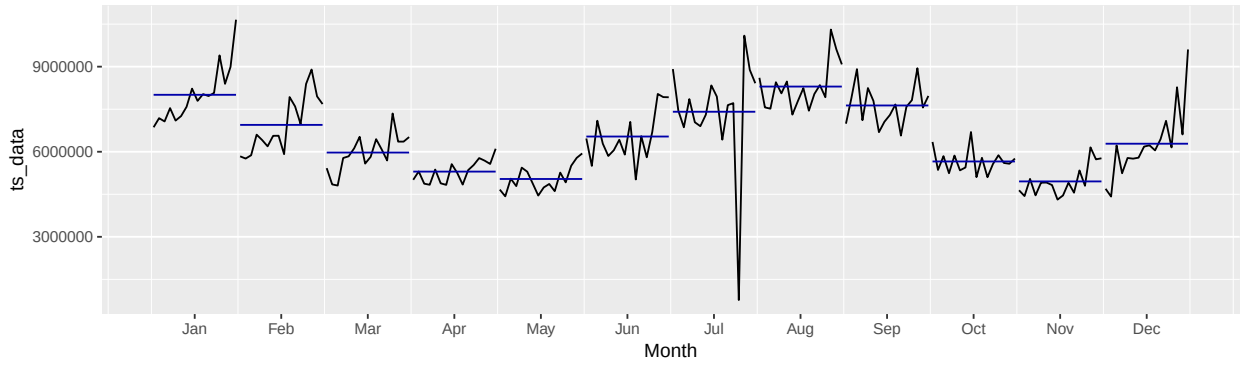

```
ggseasonplot(ts_data);
```



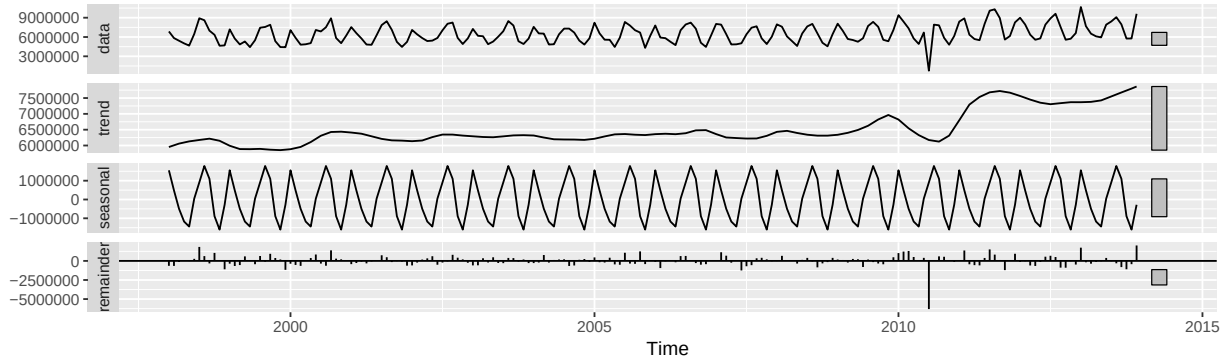

```
boxplot(ts_data~cycle(ts_data),xlab="Month", ylab = "Monthly Residential Power Usage");
```



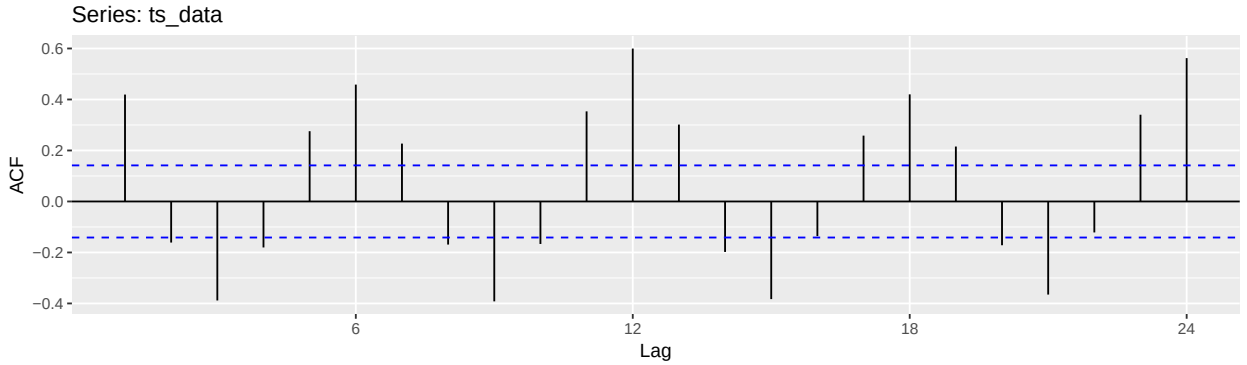

```
ggsubseriesplot(ts_data);
```




```
stl(ts_data, s.window = 'periodic') %>% autoplot();
```



```
ggAcf(ts_data);
```




```
#Box.test(ts_data, type = c("Ljung-Box"))


# handling outlier
#fit <- nnetar(tsclean(ts_data))
outlier_func <- tsoutliers(ts_data, iterate = 2, lambda = "auto")
ts_data[outlier_func$index] <- outlier_func$replacements
```

Our initial plots reveal annual seasonality within this time series. The box plot/seasonality plot actually reveals where power consumption fluctuations occur within each of the cycke 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.

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.

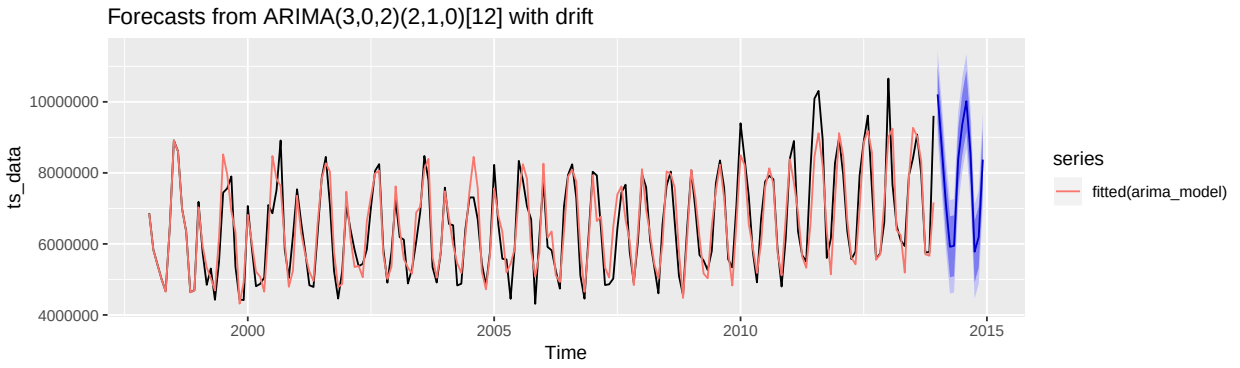
Within the overall TS plot, we see a dip in July 2010. This could be due to a power outtage during a hot summer month. This can certainly be considered to be an outlier within this TS. Using TSOutliers, we can actually identify the index where our outliers may be. TSoutliers also replaces the outlier 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.

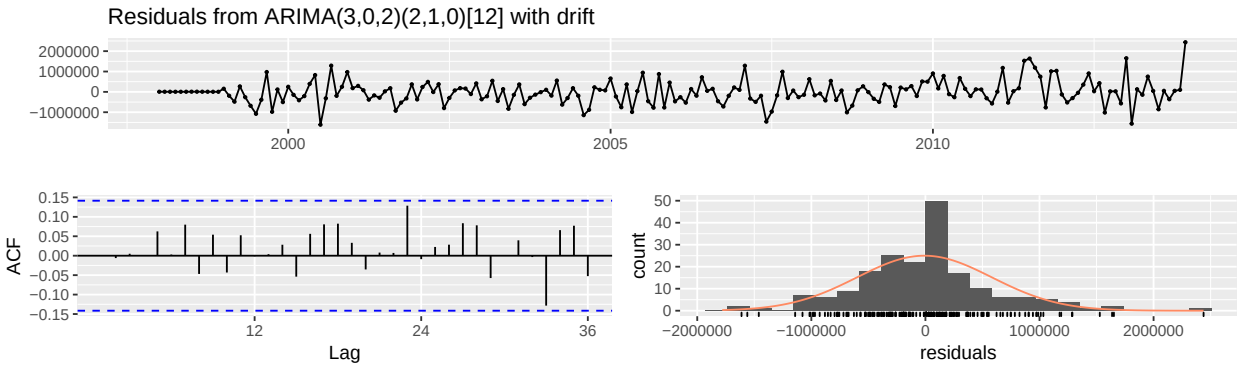
Data Model

0.0.1 Model #1: ARIMA

```
arima_model <- auto.arima(ts_data)

arima_model <- forecast(arima_model, h=12)

autoplot(arima_model) + autolayer(fitted(arima_model))
```



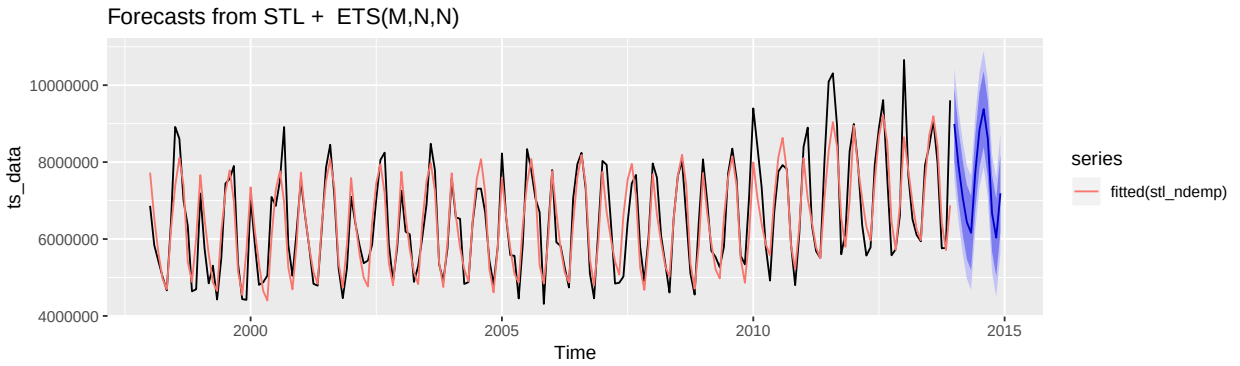

```
checkresiduals(arima_model)
```



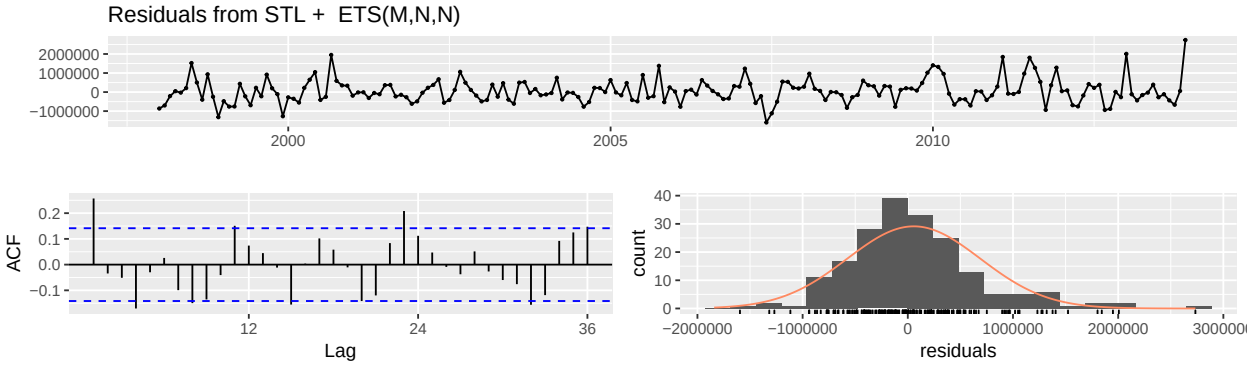

```
FALSE
FALSE   Ljung-Box test
FALSE
FALSE data:  Residuals from ARIMA(3,0,2)(2,1,0)[12] with drift
FALSE Q* = 12.555, df = 16, p-value = 0.705
FALSE
FALSE Model df: 8.   Total lags used: 24
```

0.0.2 Model #2: STL (no-demped) - MNN

```
#stlf - etsmodel estimation --- M,N,N is chosen.
stl_ndemp <- stlf(ts_data, s.window = "periodic", robust=TRUE, h = 12)

# forecast plot
autoplot(stl_ndemp) + autolayer(fitted(stl_ndemp))
```



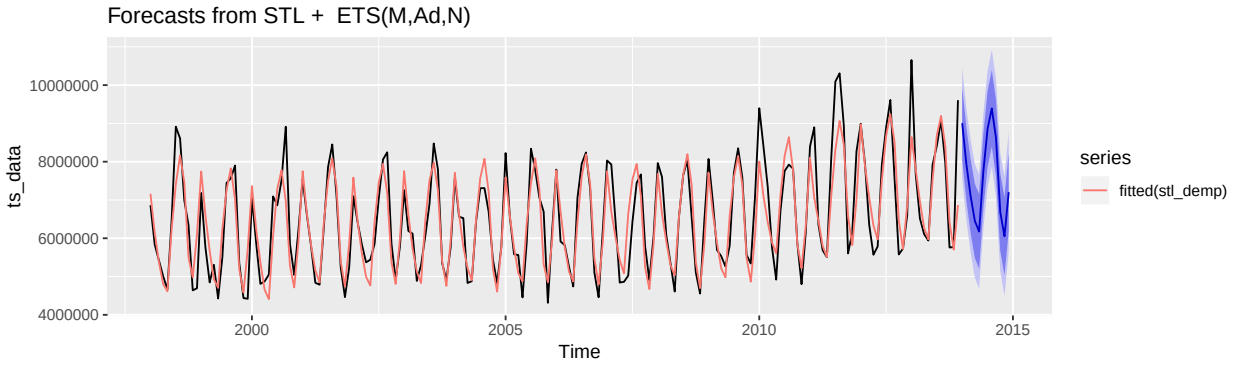
```
checkresiduals(stl_ndemp)
```



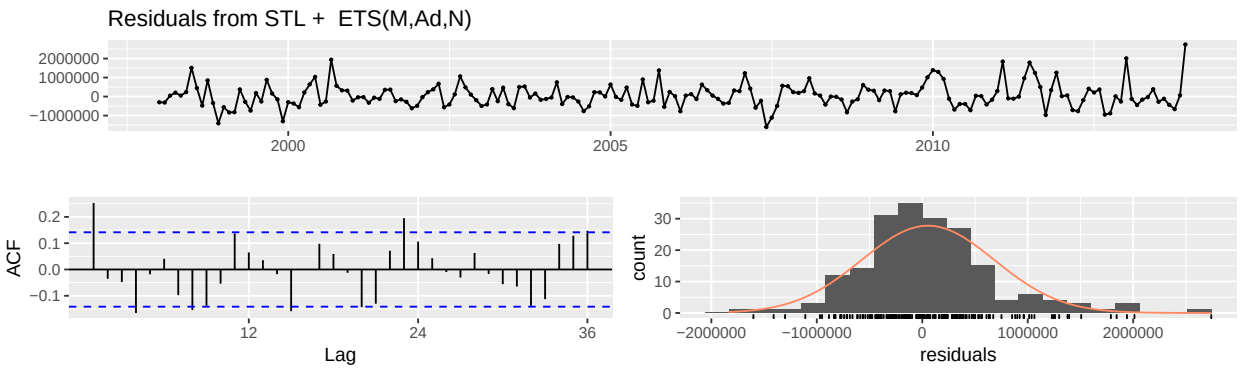

```
FALSE
FALSE   Ljung-Box test
FALSE
FALSE data:  Residuals from STL +  ETS(M,N,N)
FALSE Q* = 65.934, df = 22, p-value = 0.00000284
FALSE
FALSE Model df: 2.   Total lags used: 24
```

0.0.3 Model #2-2: STL (demped) - MAdN

```
#stlf - etsmodel estimation --- M, Ad, N is chosen.
stl_demp <- stlf(ts_data, damped=TRUE, s.window = "periodic", robust=TRUE, h = 12)

# forecast plot
autoplot(stl_demp) + autolayer(fitted(stl_demp))
```



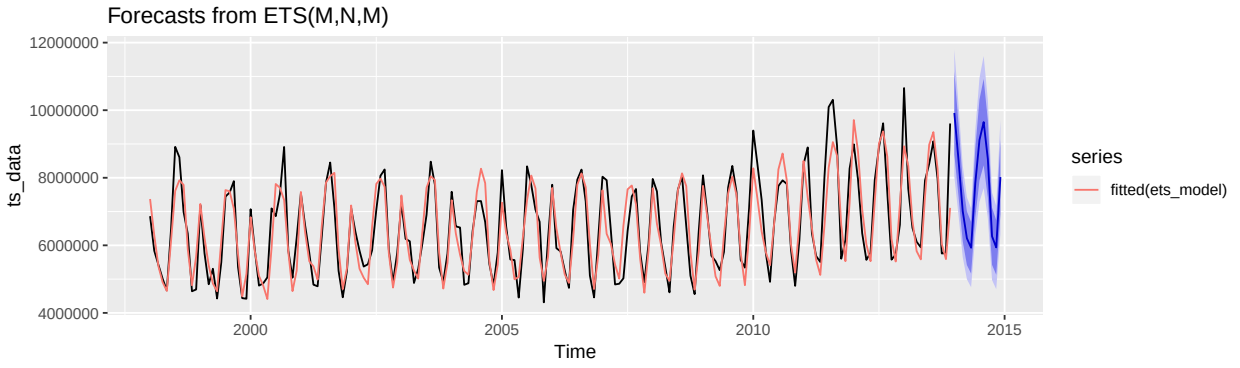

```
checkresiduals(stl_demp)
```



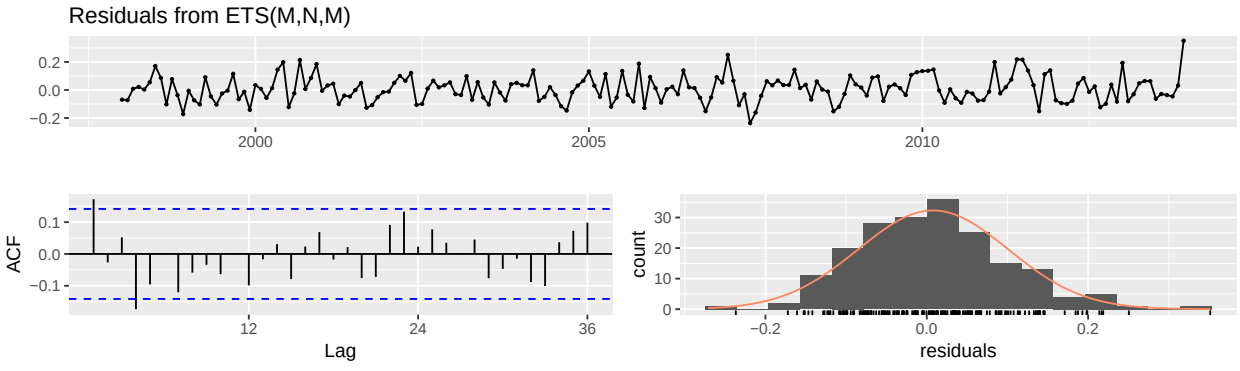

```
FALSE
FALSE   Ljung-Box test
FALSE
FALSE data:  Residuals from STL +  ETS(M,Ad,N)
FALSE Q* = 63.375, df = 19, p-value = 0.000001119
FALSE
FALSE Model df: 5.   Total lags used: 24
```

0.0.4 Model #3: ets - MNM

```
# ETS models - MNM
ets_model <- ets(ts_data)

# forecast plot
autoplot(forecast(ets_model, h=12)) + autolayer(fitted(ets_model))
```



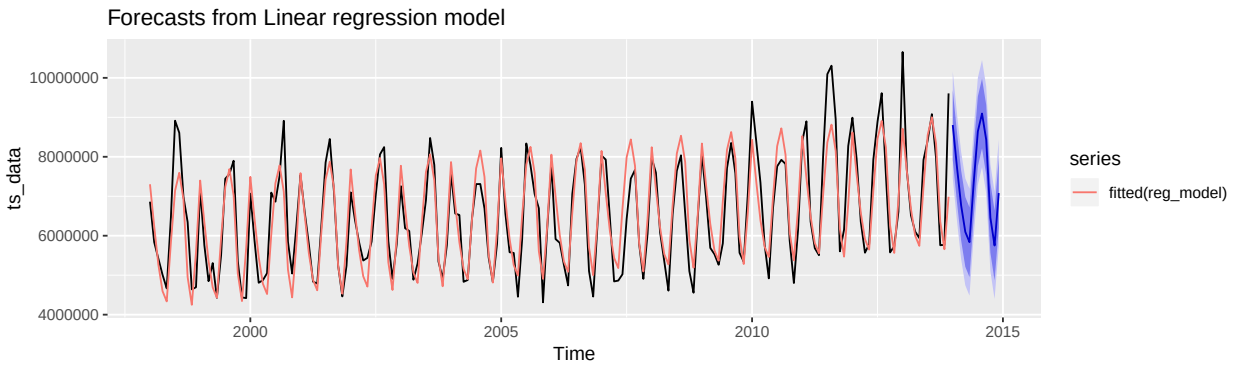

```
checkresiduals(ets_model)
```



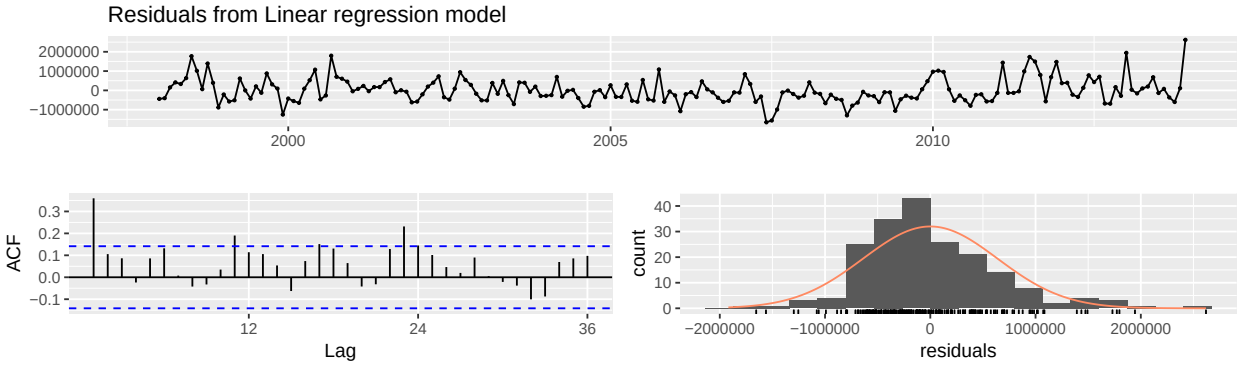
```
FALSE
FALSE   Ljung-Box test
FALSE
FALSE data:  Residuals from ETS(M,N,M)
FALSE Q* = 32.042, df = 10, p-value = 0.000394
FALSE
FALSE Model df: 14.   Total lags used: 24
```

0.0.5 Model #4: Regression

```
fit.reg <- tslm(ts_data ~ trend + season)
reg_model <- forecast(fit.reg, h = 12)

# forecast plot
autoplot(reg_model) + autolayer(fitted(reg_model))
```



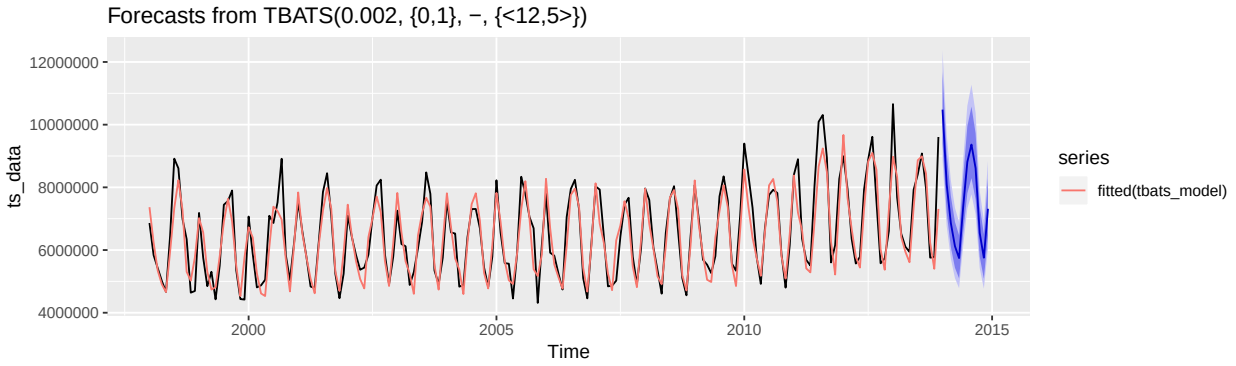

```
checkresiduals(reg_model)
```



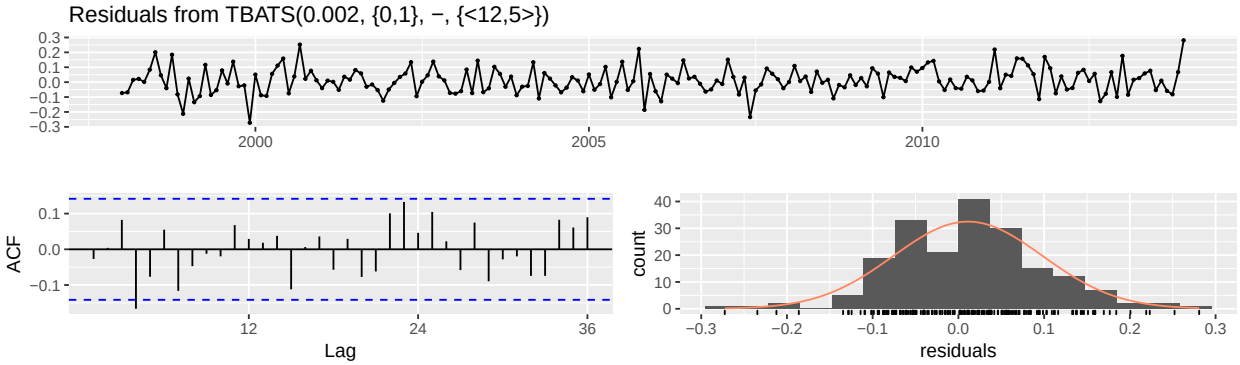

```
FALSE
FALSE   Ljung-Box test
FALSE
FALSE data:  Residuals from Linear regression model
FALSE Q* = 80.001, df = 11, p-value = 0.000000000001475
FALSE
FALSE Model df: 13.   Total lags used: 24
```

0.0.6 Model #5: TBATS

```
fit.tbats <- tbats(ts_data)
tbats_model <- forecast(fit.tbats, h = 12)

# forecast plot
autoplot(tbats_model) + autolayer(fitted(tbats_model))
```




```
checkresiduals(tbats_model)
```




```
FALSE
FALSE   Ljung-Box test
FALSE
FALSE data:  Residuals from TBATS(0.002, {0,1}, -, {<12,5>})
FALSE Q* = 26.054, df = 7, p-value = 0.0004925
FALSE
FALSE Model df: 17.   Total lags used: 24
```

0.0.7 Accuracy of Models

```
accuracy(arima_model);
```

```
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
```

```
accuracy(stl_ndemp);
```

```
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
```

```
accuracy(stl_demp);
```

```
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
```

```
accuracy(ets_model);
```

```
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
```

```
accuracy(reg_model);
```

```
FALSE                                 ME     RMSE      MAE        MPE     MAPE
FALSE Training set -0.00000000001455192 637832.8 480849.1 -0.8253442 7.286874
FALSE                    MASE      ACF1
FALSE Training set 0.7761049 0.3597939
```

```
accuracy(tbats_model)
```

```
FALSE                     ME     RMSE      MAE       MPE    MAPE      MASE
FALSE Training set 97092.89 577485.8 433659.7 0.7290721 6.54268 0.6999398
FALSE                     ACF1
FALSE Training set 0.01196249
```

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

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.

If we examine the Ljung-Box test results for our models, the only model with a p-value > 0.05 is Model #1: ARIMA. This implies that the residuals from other models are not independent, hence not white noise.

In contrast, when we first attempted the analysis by building models without handling outlier (2010 Jul), the only model with a p-value < 0.05 was Model #3: ets - MNN and hence we accepted all the other models except for #3.

Handling 1 outlier dramatically changed the outcome of Ljung-Box test.

Forecast

We will implement 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. We only accepted Model #1 from Ljung-Box test and hence it is our final choice.

0.0.8 Model #1: ARIMA

```
arima_cv <- function(x, h){forecast(Arima(x, order = c(3, 0, 2), seasonal = c(2, 1, 0), include.drift =

e <- tsCV(ts_data, arima_cv, h=12)
```

```r
sqrt(mean(e^2, na.rm=TRUE))
```

```
FALSE [1] 725175
```

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.

Discussion

In our first phased implicit analysis, which is not recorded on this Rmd, where outlier (2010 - Jul) was not handled, STL - ANN was the best model in terms of RMSE on test set. As outlier was handled, on the other hand, STL models became invalid predictors as residuals were autocorrelated. We learned that handling outlier, even if it is just one data point, makes the result of modelling outcome completely.