R Notebook

```
library(readxl)
Tng_Ctr_Hour <- read_excel("C:/RBS/Business Forecasting/Group Project/Tng_Ctr_Hour.xlsx")
View(Tng_Ctr_Hour)
```

```
library(data.table)
library(ggplot2)
library(TTR)
library(fpp)
```

```
## Loading required package: forecast

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Loading required package: fma

## Loading required package: expsmooth

## Loading required package: lmtest

## Loading required package: zoo

##
## Attaching package: 'zoo'

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

## Loading required package: tseries
```

```
library(fpp2)
```

```
##
## Attaching package: 'fpp2'

## The following objects are masked from 'package:fpp':
##
##     ausair, ausbeer, austa, austourists, debitcards, departures,
##     elecequip, euretail, guinearice, oil, sunspotarea, usmelec
```

```r
library(ggplot2)
library(stats)
library(dplyr)
```

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

## The following objects are masked from 'package:data.table':
##
##     between, first, last

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

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

```r
library(graphics)
library(ggfortify)
```

```
## Registered S3 methods overwritten by 'ggfortify':
##   method                 from
##   autoplot.Arima         forecast
##   autoplot.acf           forecast
##   autoplot.ar            forecast
##   autoplot.bats          forecast
##   autoplot.decomposed.ts forecast
##   autoplot.ets           forecast
##   autoplot.forecast      forecast
##   autoplot.stl           forecast
##   autoplot.ts            forecast
##   fitted.ar              forecast
##   fortify.ts             forecast
##   residuals.ar           forecast
```

```r
summary(Tng_Ctr_Hour)
```

```
##      Year              Quarter              Month              Device_Hrs
##  Length:81          Length:81          Length:81          Min.   : 222.8
##  Class :character   Class :character   Class :character   1st Qu.: 899.0
##  Mode  :character   Mode  :character   Mode  :character   Median :1008.0
##                                                           Mean   : 990.1
##                                                           3rd Qu.:1101.7
##                                                           Max.   :1519.9
##  DH_Prev_Year       DH_YoY_Change      DH_YoY_Ch_Per      Total_Inst_Hrs
##  Length:81          Length:81          Length:81          Min.   : 504.6
##  Class :character   Class :character   Class :character   1st Qu.:1937.3
##  Mode  :character   Mode  :character   Mode  :character   Median :2203.2
##                                                           Mean   :2165.7
```

```
##                                                           3rd Qu.:2446.8
##                                                           Max.   :3084.1
##  Total_Inst_Hrs_Prev_Year Inst_Hrs_YoY_Change Total_Inst_Hrs_YoY_Change_Per2
##  Length:81                Length:81           Length:81
##  Class :character         Class :character    Class :character
##  Mode  :character         Mode  :character    Mode  :character
##
##
##
```

Create a Factor for the Dataset

```
setDT(Tng_Ctr_Hour)
#changing the character values into factors
Tng_Ctr_Hour[,Quarter:=factor(Quarter)]
Tng_Ctr_Hour[,Month:=factor(Month)]
Tng_Ctr_Hour[,Year:=factor(Year)]
```

Create a subset for the tested data

```
FD = select(Tng_Ctr_Hour, Year, Quarter, Month, Device_Hrs, Total_Inst_Hrs )
```

```
FD
```

```
##        Year Quarter Month Device_Hrs Total_Inst_Hrs
##  1: 2015-01      Q1   Jan     960.42        1700.67
##  2: 2015-02      Q1   Feb     944.08        1614.00
##  3: 2015-03      Q1   Mar    1429.12        2532.90
##  4: 2015-04      Q2   Apr    1097.00        2152.25
##  5: 2015-05      Q2   May     915.85        1695.43
##  6: 2015-06      Q2   Jun     783.45        1675.91
##  7: 2015-07      Q3   Jul    1034.52        2095.00
##  8: 2015-08      Q3   Aug    1169.50        2459.83
##  9: 2015-09      Q3   Sep    1027.08        2219.00
## 10: 2015-10      Q4   Oct    1262.32        2765.47
## 11: 2015-11      Q4   Nov     999.25        2239.33
## 12: 2015-12      Q4   Dec     929.42        2054.59
## 13: 2016-01      Q1   Jan     796.42        1935.51
## 14: 2016-02      Q1   Feb     874.55        2017.40
## 15: 2016-03      Q1   Mar    1091.55        2235.33
## 16: 2016-04      Q2   Apr    1141.84        2409.30
## 17: 2016-05      Q2   May     871.36        1937.34
## 18: 2016-06      Q2   Jun    1181.21        2606.56
## 19: 2016-07      Q3   Jul     757.59        1791.01
## 20: 2016-08      Q3   Aug     972.73        2216.60
## 21: 2016-09      Q3   Sep     807.02        1934.39
## 22: 2016-10      Q4   Oct    1519.92        3084.09
## 23: 2016-11      Q4   Nov    1101.67        2361.81
```

```
## 24: 2016-12      Q4   Dec    801.83        1853.99
## 25: 2017-01      Q1   Jan    995.09        2446.80
## 26: 2017-02      Q1   Feb    962.00        2169.17
## 27: 2017-03      Q1   Mar   1130.24        2768.35
## 28: 2017-04      Q2   Apr   1054.71        2291.76
## 29: 2017-05      Q2   May   1044.95        2172.54
## 30: 2017-06      Q2   Jun   1013.73        2366.74
## 31: 2017-07      Q3   Jul    693.33        1739.90
## 32: 2017-08      Q3   Aug    983.25        2304.53
## 33: 2017-09      Q3   Sep    987.64        2302.29
## 34: 2017-10      Q4   Oct   1252.69        2810.70
## 35: 2017-11      Q4   Nov    969.31        2249.47
## 36: 2017-12      Q4   Dec    806.10        1800.08
## 37: 2018-01      Q1   Jan   1060.57        2466.01
## 38: 2018-02      Q1   Feb   1200.25        2414.06
## 39: 2018-03      Q1   Mar   1262.25        2666.14
## 40: 2018-04      Q2   Apr   1184.45        2625.94
## 41: 2018-05      Q2   May   1059.92        2455.24
## 42: 2018-06      Q2   Jun    993.55        2098.89
## 43: 2018-07      Q3   Jul    908.37        1973.29
## 44: 2018-08      Q3   Aug   1096.93        2403.06
## 45: 2018-09      Q3   Sep   1121.75        2368.10
## 46: 2018-10      Q4   Oct   1412.47        2955.81
## 47: 2018-11      Q4   Nov   1010.25        2203.17
## 48: 2018-12      Q4   Dec    970.12        1991.45
## 49: 2019-01      Q1   Jan   1063.13        2542.16
## 50: 2019-02      Q1   Feb   1036.95        2441.90
## 51: 2019-03      Q1   Mar   1130.87        2456.02
## 52: 2019-04      Q2   Apr    903.97        2286.02
## 53: 2019-05      Q2   May   1284.95        2734.56
## 54: 2019-06      Q2   Jun   1265.56        2571.35
## 55: 2019-07      Q3   Jul    848.64        2075.30
## 56: 2019-08      Q3   Aug   1247.40        2767.26
## 57: 2019-09      Q3   Sep   1106.84        2441.50
## 58: 2019-10      Q4   Oct   1217.08        2626.36
## 59: 2019-11      Q4   Nov   1091.84        2377.05
## 60: 2019-12      Q4   Dec   1024.67        2085.33
## 61: 2020-01      Q1   Jan   1094.62        2523.89
## 62: 2020-02      Q1   Feb   1050.98        2137.86
## 63: 2020-03      Q1   Mar    726.19        1556.44
## 64: 2020-04      Q2   Apr    222.80         504.57
## 65: 2020-05      Q2   May    556.92        1181.00
## 66: 2020-06      Q2   Jun    899.00        1831.79
## 67: 2020-07      Q3   Jul    585.58        1427.42
## 68: 2020-08      Q3   Aug    811.74        1982.89
## 69: 2020-09      Q3   Sep   1047.41        2283.34
## 70: 2020-10      Q4   Oct   1239.26        2568.26
## 71: 2020-11      Q4   Nov    911.93        1968.93
## 72: 2020-12      Q4   Dec    569.75        1303.50
## 73: 2021-01      Q1   Jan    685.91        1685.08
## 74: 2021-02      Q1   Feb    692.88        1605.12
## 75: 2021-03      Q1   Mar    805.42        1810.00
## 76: 2021-04      Q2   Apr    904.00        2178.17
## 77: 2021-05      Q2   May    937.62        1977.58
```

```
## 78: 2021-06      Q2   Jun     954.00         2056.29
## 79: 2021-07      Q3   Jul     605.00         1457.42
## 80: 2021-08      Q3   Aug    1027.23         2175.39
## 81: 2021-09      Q3   Sep    1008.00         2173.00
##        Year Quarter Month Device_Hrs Total_Inst_Hrs
```

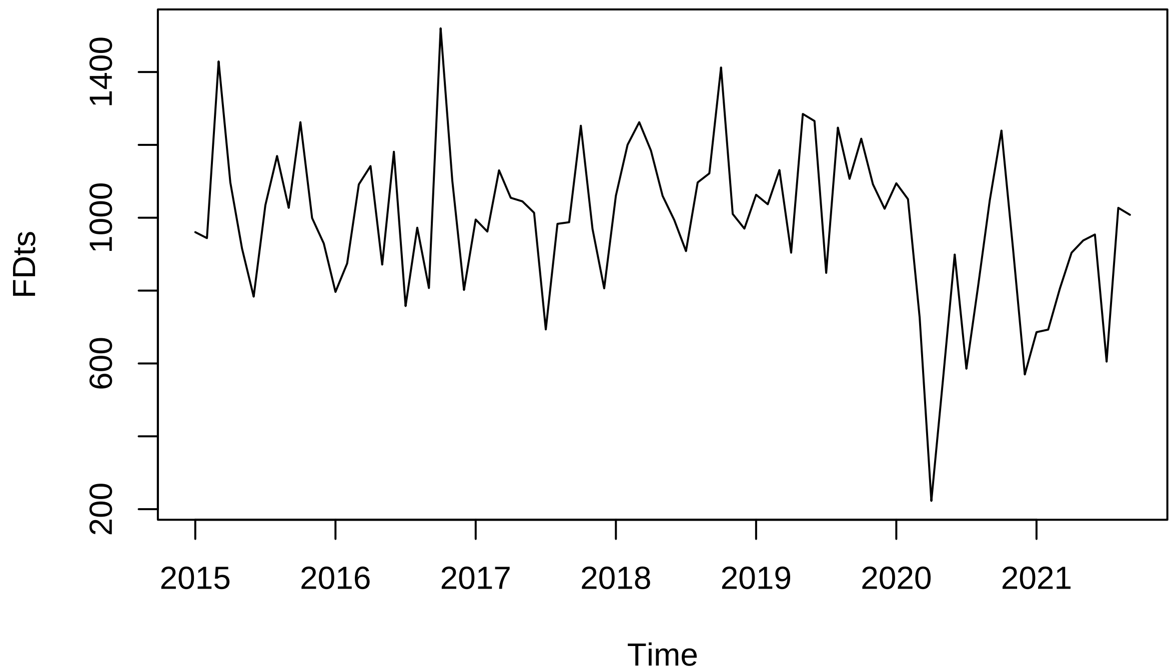
Convert to Time Series Data

```r
FDts = ts(FD$Device_Hrs, frequency = 12, start = c(2015,1))
FDts
```

```
##          Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep
## 2015  960.42  944.08 1429.12 1097.00  915.85  783.45 1034.52 1169.50 1027.08
## 2016  796.42  874.55 1091.55 1141.84  871.36 1181.21  757.59  972.73  807.02
## 2017  995.09  962.00 1130.24 1054.71 1044.95 1013.73  693.33  983.25  987.64
## 2018 1060.57 1200.25 1262.25 1184.45 1059.92  993.55  908.37 1096.93 1121.75
## 2019 1063.13 1036.95 1130.87  903.97 1284.95 1265.56  848.64 1247.40 1106.84
## 2020 1094.62 1050.98  726.19  222.80  556.92  899.00  585.58  811.74 1047.41
## 2021  685.91  692.88  805.42  904.00  937.62  954.00  605.00 1027.23 1008.00
##          Oct     Nov     Dec
## 2015 1262.32  999.25  929.42
## 2016 1519.92 1101.67  801.83
## 2017 1252.69  969.31  806.10
## 2018 1412.47 1010.25  970.12
## 2019 1217.08 1091.84 1024.67
## 2020 1239.26  911.93  569.75
## 2021
```

Create a plot of the time series

```r
plot(FDts)
```



The plot shows a series of peaks and valleys, which suggests seasonality. Hours seemed to flow between ~700 and ~1500 consistently until 2020 when COVID struck. This caused a drop to 222 hrs in April. October is the most busy month, never falling below 1200 hrs. July and December appear to be the lightest months, only surpassing 1000 hours once each during the observed history.

Because of the pandemic, we will remove the data beginning in 2020 as it was affected by unforecastable forces.

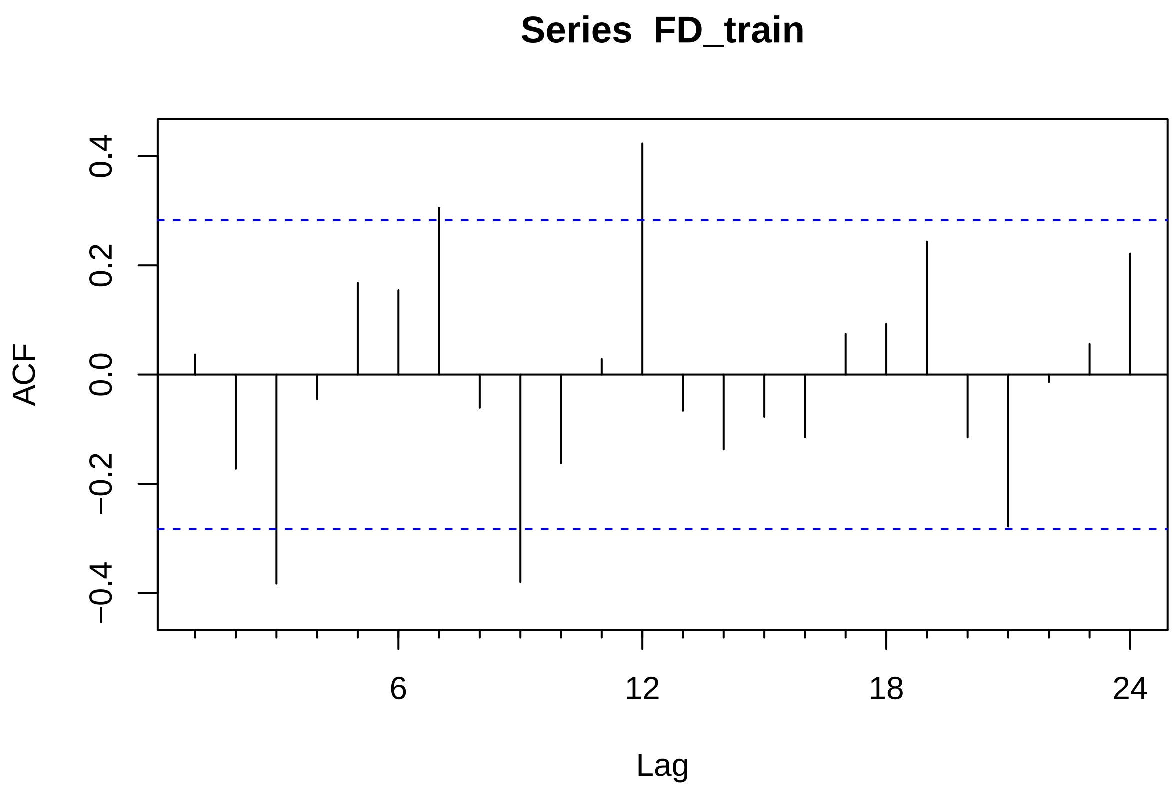
Training the Model

```
FD_train = ts(FD$Device_Hrs, frequency = 12, start = c(2015,1), end = c(2018, 12))
```

```
FD_test = ts(FD$Device_Hrs, frequency = 12, start = c(2019,1), end = c(2019, 12))
```

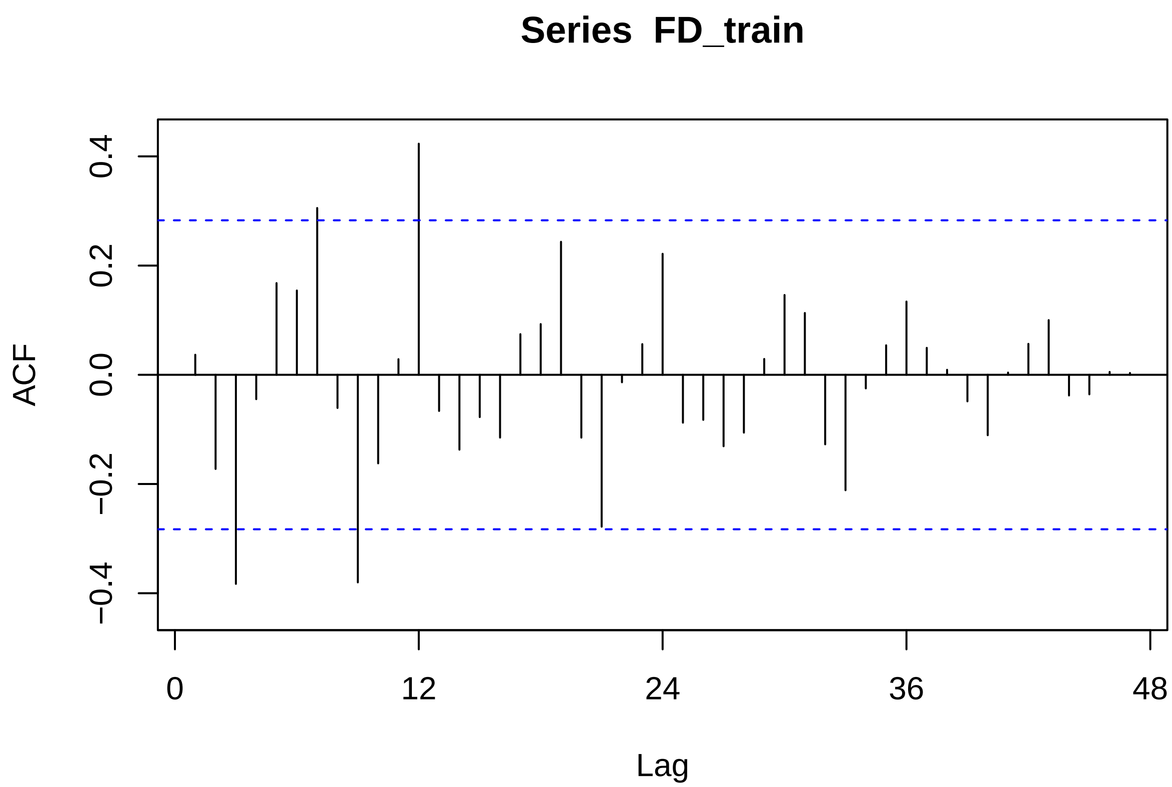
Autocorrlation of the data

```
Acf(FD_train)
```



We observe strong positive and negative autocorrlation, which furthers ours supsicions that there is seasonality

```
Acf(FD_train, lag.max = 48)
```



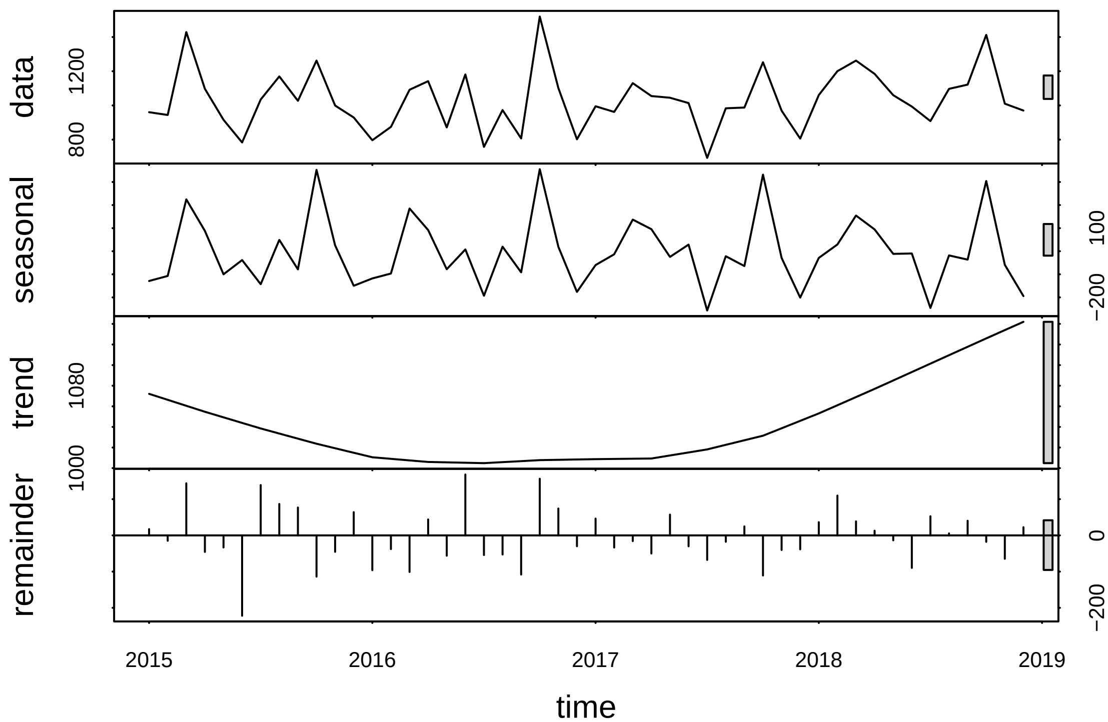
Checking For Seasonality and Trends

```
fit= stl(FD_train, s.window = 5)
fit
```

```
##  Call:
##  stl(x = FD_train, s.window = 5)
##
## Components
##              seasonal    trend   remainder
## Jan 2015 -129.045449 1072.110   17.355668
## Feb 2015 -107.262914 1066.307  -14.963741
## Mar 2015  224.832060 1060.504  143.784413
## Apr 2015   88.044656 1054.700  -45.745057
## May 2015 -100.010890 1049.299  -33.438306
## Jun 2015  -38.452890 1043.898 -221.995102
## Jul 2015 -142.974820 1038.497  138.998032
## Aug 2015   49.132423 1033.557   86.811014
## Sep 2015  -78.637347 1028.616   77.101009
## Oct 2015  352.636565 1023.676 -113.992678
## Nov 2015   25.472369 1019.294  -45.516306
## Dec 2015 -149.739347 1014.912   64.247586
## Jan 2016 -117.920199 1010.530  -96.189386
## Feb 2016  -96.644317 1009.011  -37.816415
## Mar 2016  185.174945 1007.492 -101.116824
## Apr 2016   91.762657 1005.973   44.104317
## May 2016  -78.120110 1005.592  -56.111737
```

```
## Jun 2016    7.703243 1005.211  168.296088
## Jul 2016 -192.946834 1004.829  -54.292656
## Aug 2016   19.761423 1005.800  -52.831165
## Sep 2016  -91.470378 1006.770 -108.279615
## Oct 2016  355.701025 1007.740  156.478730
## Nov 2016   19.348695 1008.064   74.257790
## Dec 2016 -176.476142 1008.387  -30.080643
## Jan 2017  -60.060469 1008.710   46.440415
## Feb 2017  -13.295386 1008.911  -33.615230
## Mar 2017  137.033245 1009.111  -15.904423
## Apr 2017   95.565533 1009.312  -50.167271
## May 2017  -24.715705 1012.262   57.403293
## Jun 2017   28.641174 1015.213  -30.124259
## Jul 2017 -256.977073 1018.164  -67.856686
## Aug 2017  -21.654380 1022.593  -17.688292
## Sep 2017  -64.283047 1027.022   24.901463
## Oct 2017  332.006351 1031.450 -110.766847
## Nov 2017  -28.975050 1038.667  -40.382206
## Dec 2017 -201.159951 1045.884  -38.624064
## Jan 2018  -29.025710 1053.101   36.494935
## Feb 2018   29.202447 1061.037  110.010983
## Mar 2018  154.571226 1068.972   38.706409
## Apr 2018   94.558383 1076.908   12.983457
## May 2018  -11.493527 1085.086  -13.672874
## Jun 2018   -9.889678 1093.265  -89.824965
## Jul 2018 -245.915270 1101.443   52.842385
## Aug 2018  -18.182053 1109.607    5.505051
## Sep 2018  -36.462252 1117.771   40.441132
## Oct 2018  304.327121 1125.935  -17.792358
## Nov 2018  -59.028409 1133.988  -64.710038
## Dec 2018 -194.556831 1142.042   22.635174
```

```
plot(fit)
```

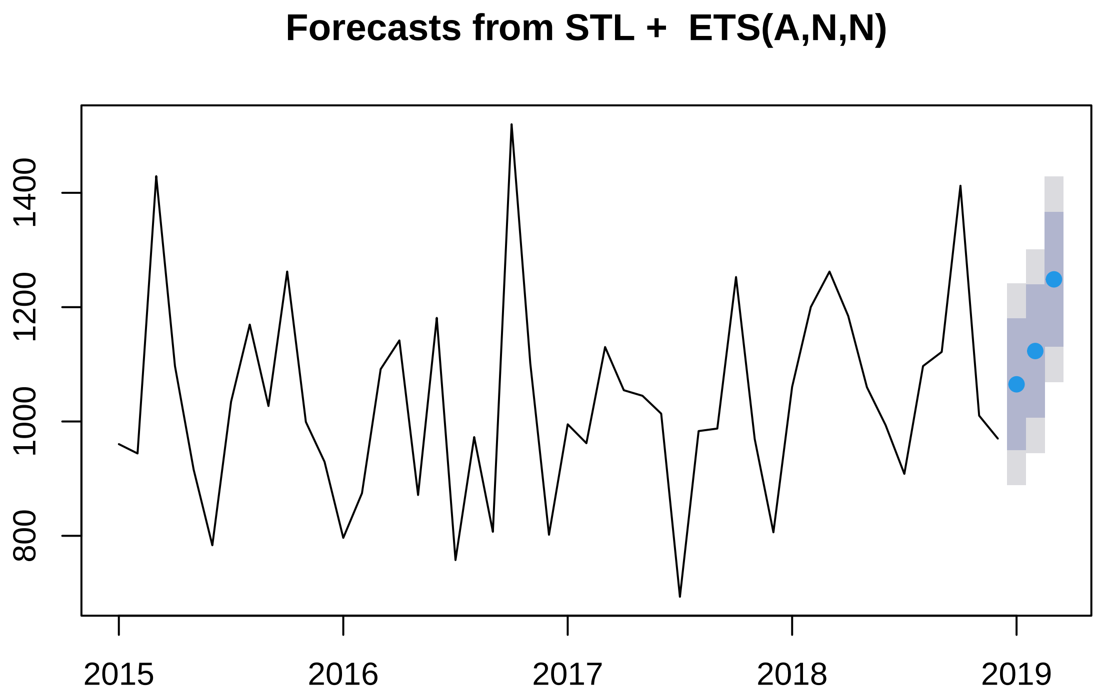


We can see some seasonality though it is not perfect. The trend is interesting as it appears that training hours were trending down, only to rebound.

Forecasting Data

Simple forecast of three periods

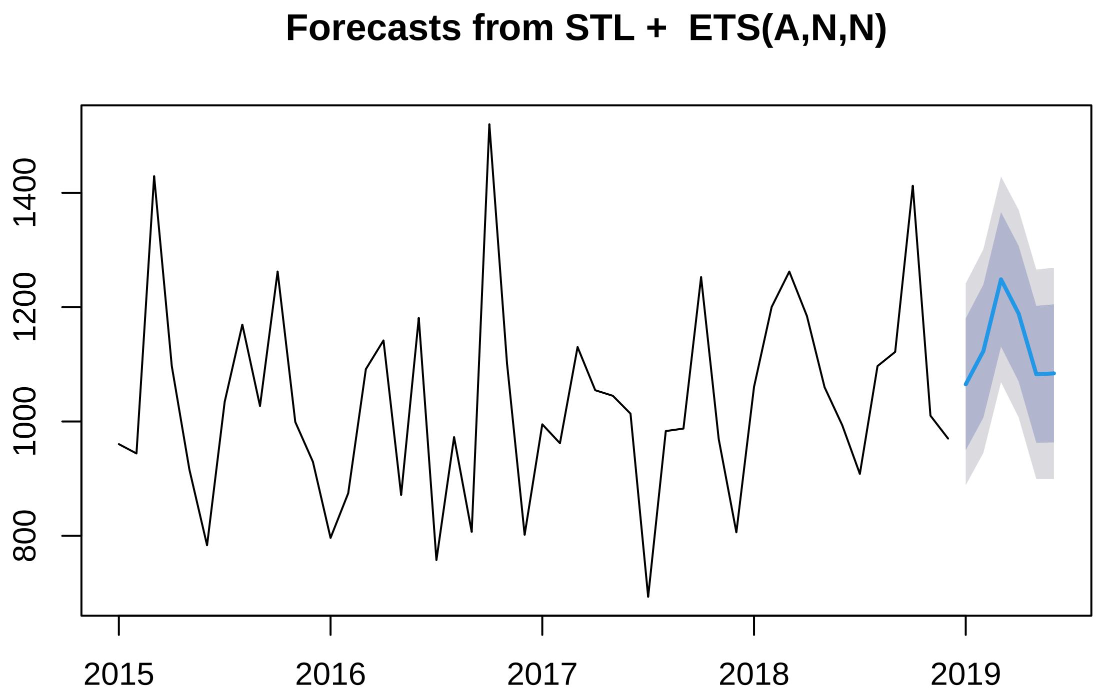
```
FitFore3 = forecast(fit, h=3)
plot(FitFore3)
```



Over the next three periods, we expect the number of hours to increase

What about 6 periods?

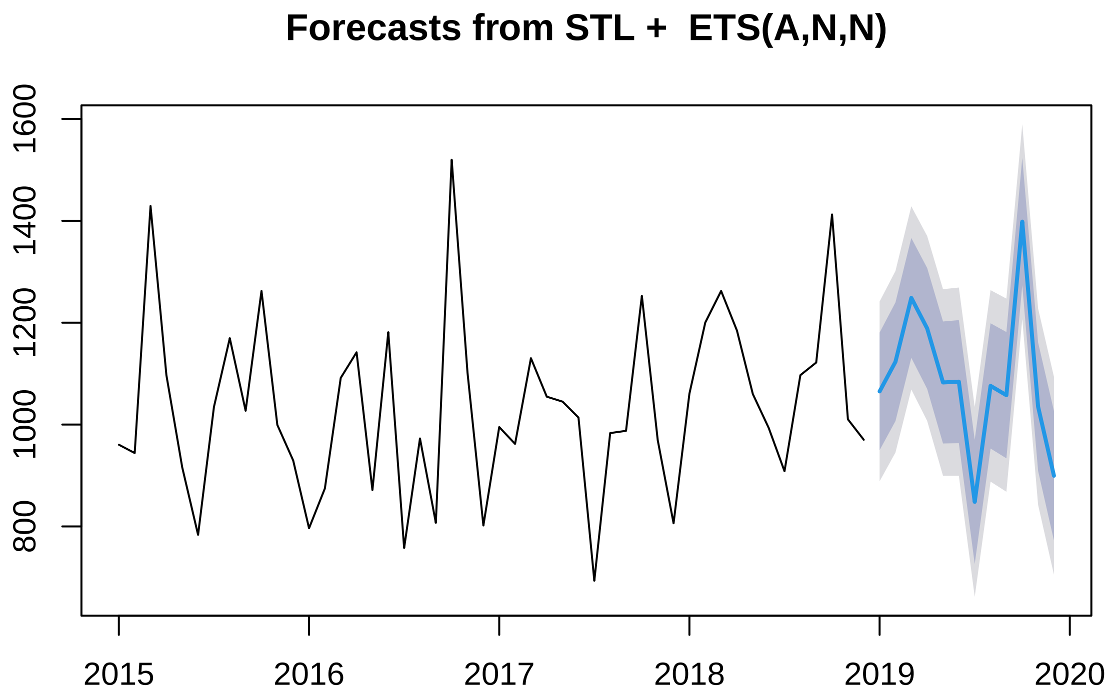
```
FitFore6 = forecast(fit, h=6)
plot(FitFore6)
```



We expect the data to increase, then decrease after three periods, consistent with our history

What about 12 Periods?!?!

```
FitFore12 = forecast(fit, h=12)
plot(FitFore12)
```



This is interesting and shows the seasonality of the data. It doesn't appear to be an exact duplication of the previous 12 months.

```
accuracy(FitFore3)
```

Accuracy Test of Forecast

```
##                    ME     RMSE      MAE         MPE     MAPE      MASE
## Training set 7.149583 88.13674 72.24431 -0.02977668 7.049975 0.5119645
##                     ACF1
## Training set -0.04022578
```

```
accuracy(FitFore6)
```

```
##                    ME     RMSE      MAE         MPE     MAPE      MASE
## Training set 7.149583 88.13674 72.24431 -0.02977668 7.049975 0.5119645
##                     ACF1
## Training set -0.04022578
```

```
accuracy(FitFore12)
```

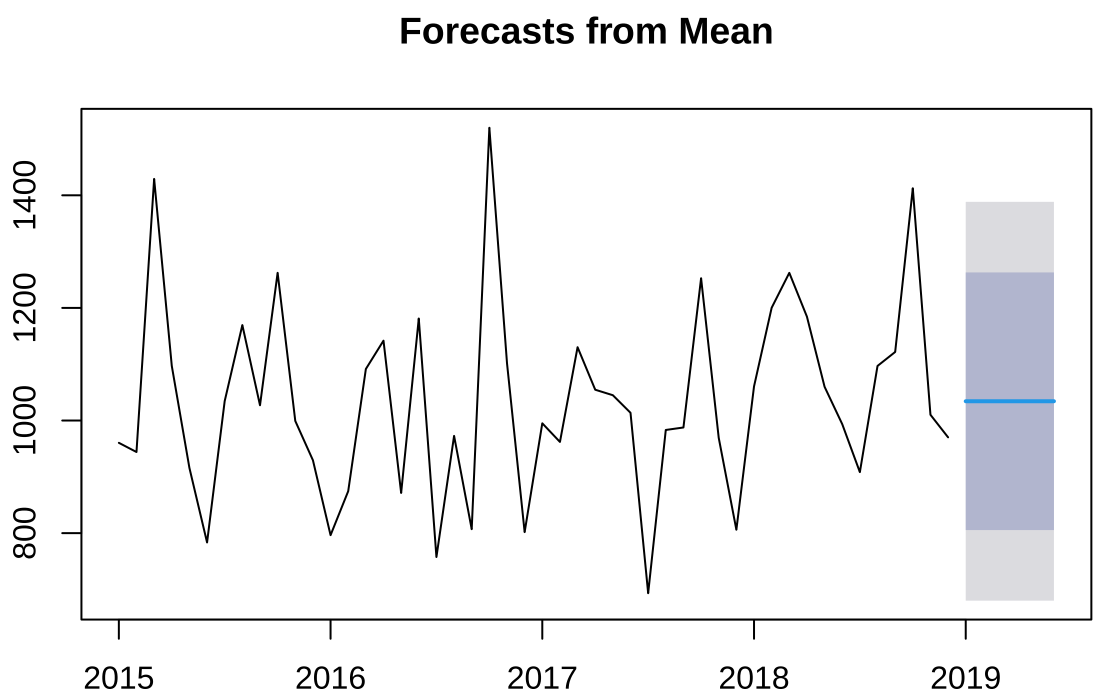
```
##                    ME     RMSE      MAE         MPE     MAPE      MASE
```

```
## Training set 7.149583 88.13674 72.24431 -0.02977668 7.049975 0.5119645
##                      ACF1
## Training set -0.04022578
```

All forecasts have the same accuracy measures. Length of the forecast does not impact the accuracy. Interesting.

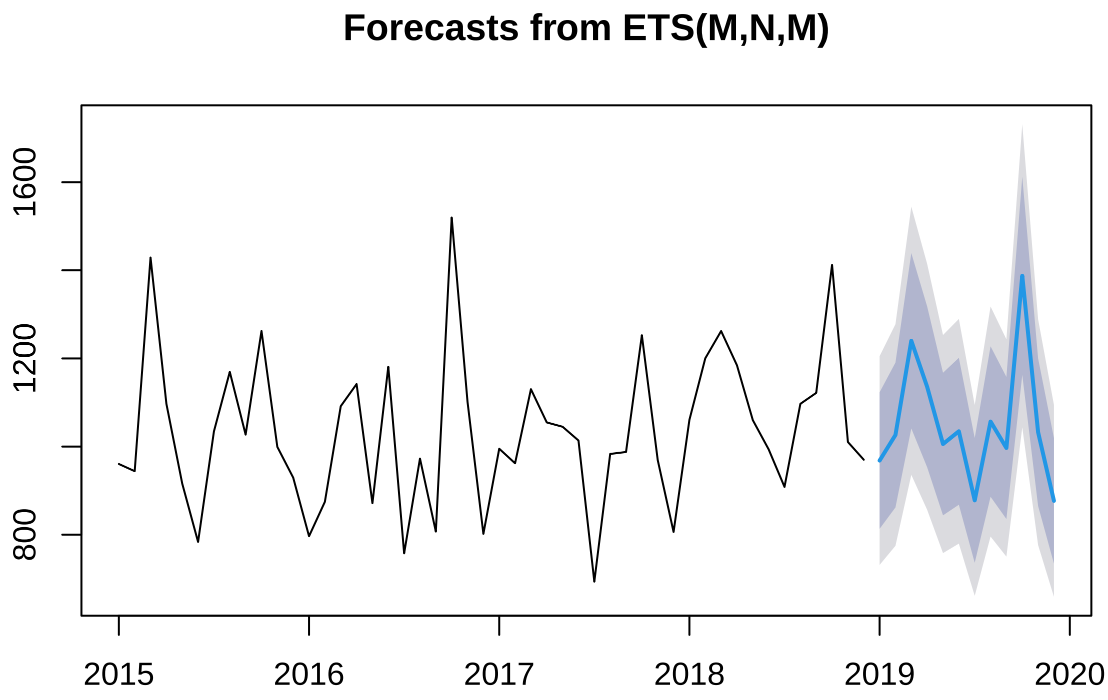
Mean Forecast Method

```
mean_FDT <- meanf(FD_train,6) # 6 is the forecasting period (6 quarters out)
plot(mean_FDT)
```

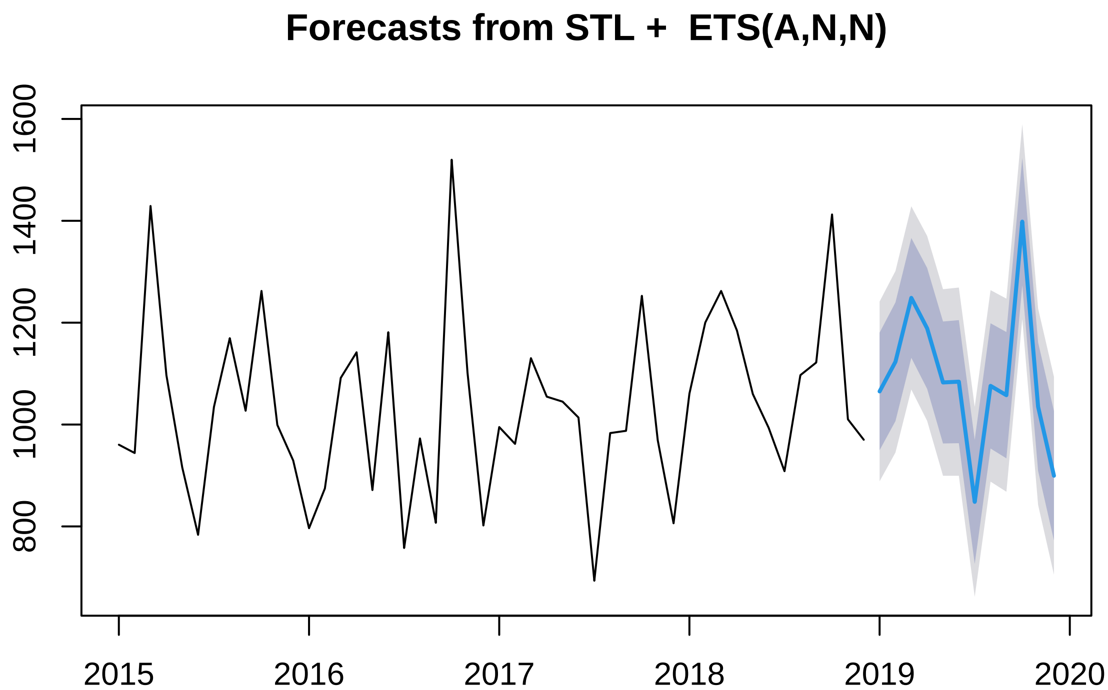


12 Month Forecast of Trained data (raw) vs. FIT data

```
Forecast_Train12 = forecast(FD_train, h=12)
plot(Forecast_Train12)
```




```r
FitFore12 = forecast(fit, h=12)
plot(FitFore12)
```



Fit forecast is steeper. Is it capturing more of the seasonal swings?

```
accuracy(FitFore12)
```

Let's compare the accuracy of the forecasts

```
##                    ME     RMSE      MAE         MPE     MAPE      MASE
## Training set 7.149583 88.13674 72.24431 -0.02977668 7.049975 0.5119645
##                     ACF1
## Training set -0.04022578
```

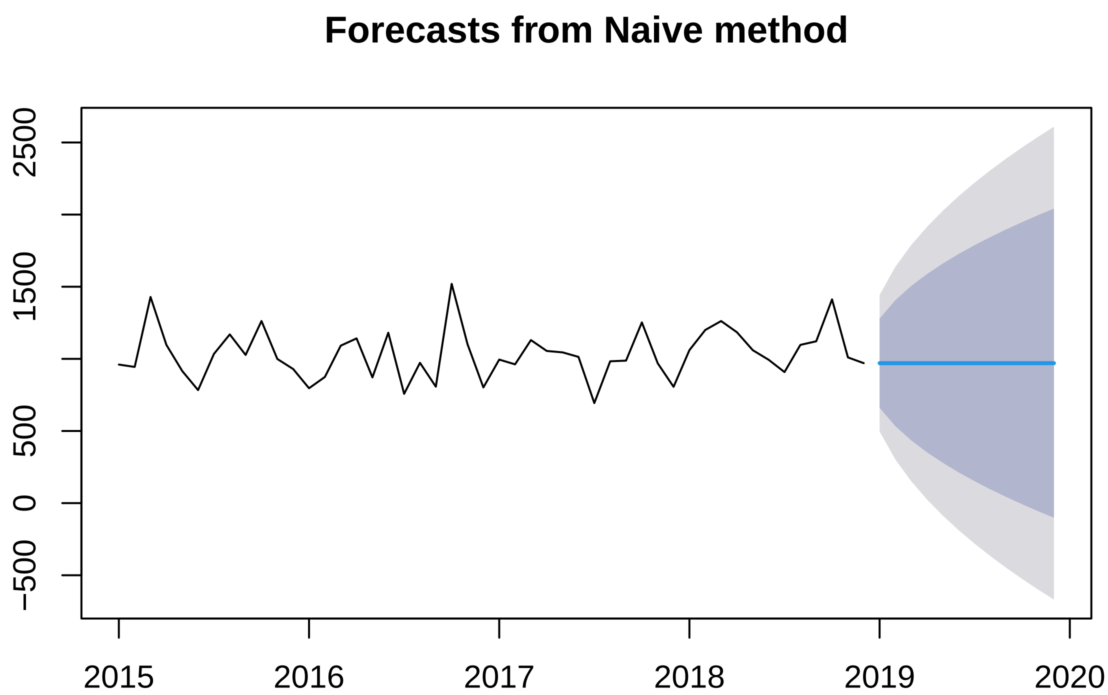
```
accuracy(Forecast_Train12)
```

```
##                    ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 8.442611 105.9033 87.19015 -0.2842957 8.633811 0.6178793
##                    ACF1
## Training set 0.02967374
```

Fit forecast appears more accurate. Has lower RMSE and MAPE.

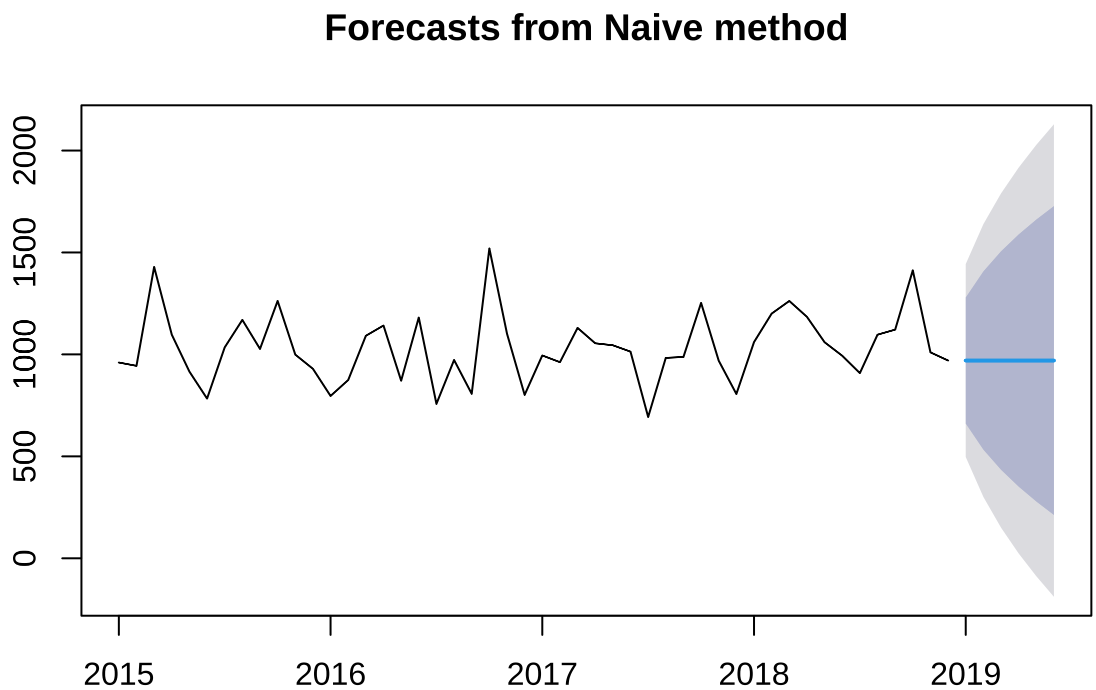
Naive Forecast

```
naive_forecast <- naive(FD_train,12)
plot(naive_forecast)
```



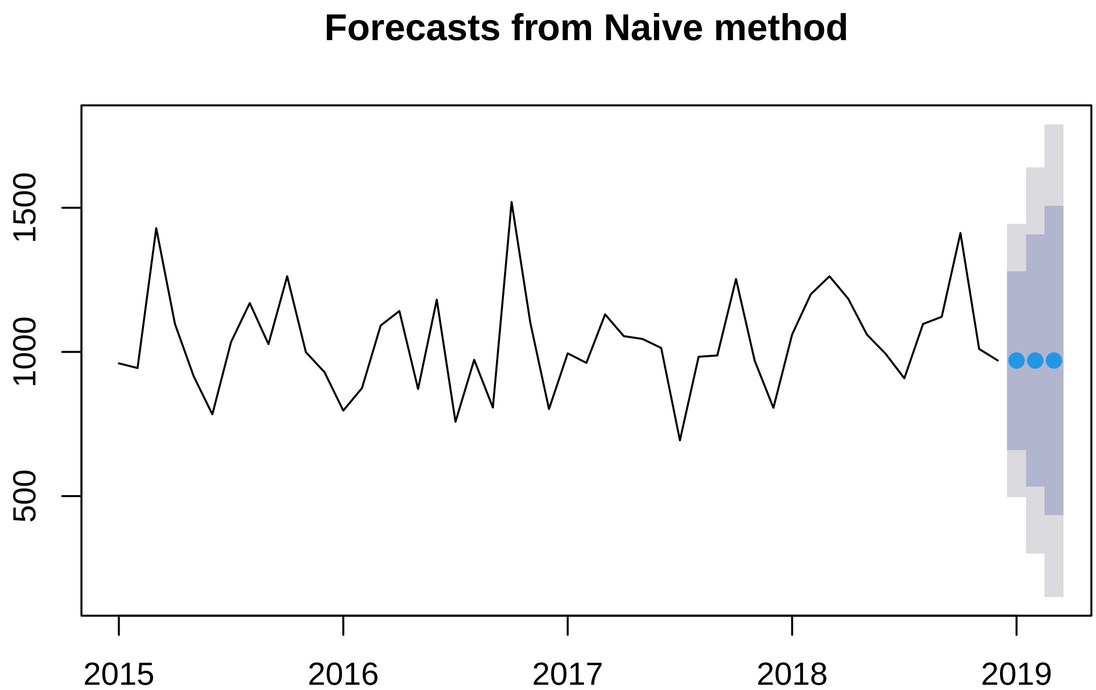
Not particularly useful because the the scope stretches into negative territory. Lets trim down the forecast to something shorter

```
naive_forecast6 <- naive(FD_train,6)
plot(naive_forecast6)
```



Still not good. 3 Months?

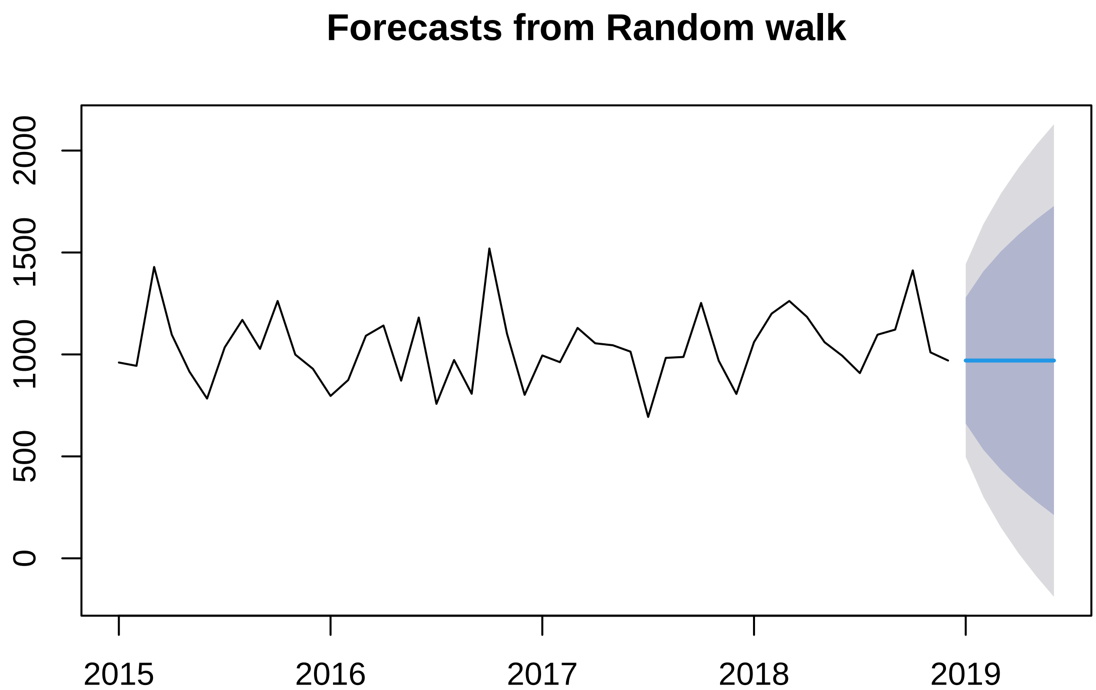
```
naive_forecast3 <- naive(FD_train,3)
plot(naive_forecast3)
```



The range of outcomes still appears too large. Naive doesn't look like the best model.

Random Walk Forecast

```
rwf_forecast <- rwf(FD_train,6)
plot(rwf_forecast)
```



This graph also shows little by way of seasonality.

Lets compare the accuracy of all the forecasts:

```
accuracy(Forecast_Train12)
```

```
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 8.442611 105.9033 87.19015 -0.2842957 8.633811 0.6178793
##                     ACF1
## Training set 0.02967374
```

```
accuracy(FitFore12)
```

```
##                     ME     RMSE      MAE         MPE     MAPE      MASE
## Training set 7.149583 88.13674 72.24431 -0.02977668 7.049975 0.5119645
##                      ACF1
## Training set -0.04022578
```

```
accuracy(rwf_forecast)
```

```
##                     ME     RMSE      MAE       MPE     MAPE     MASE       ACF1
## Training set 0.206383 241.3982 194.2936 -2.591405 18.87992 1.376876 -0.3923039
```

```r
accuracy(naive_forecast)
```

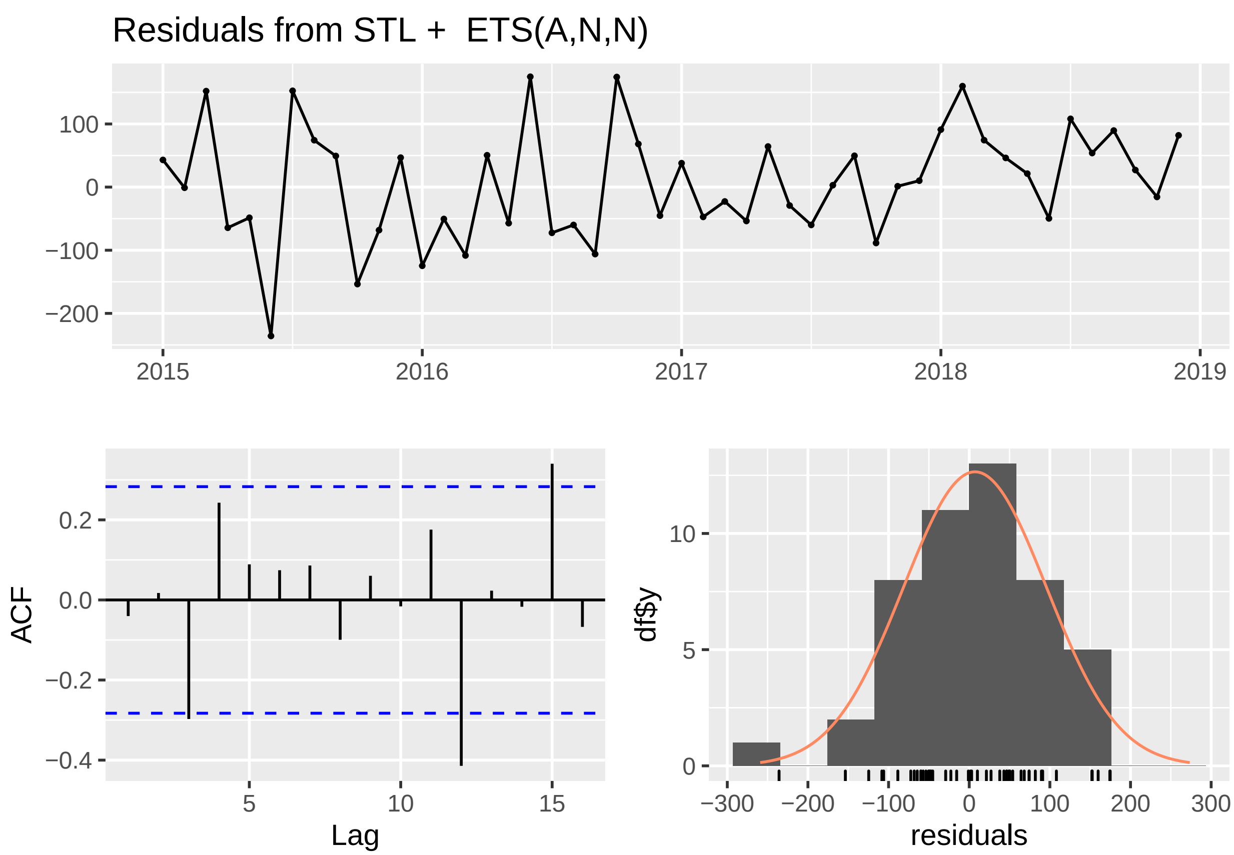
```
##                    ME     RMSE      MAE       MPE     MAPE     MASE       ACF1
## Training set 0.206383 241.3982 194.2936 -2.591405 18.87992 1.376876 -0.3923039
```

Fit forecast appears the strongest based on these metrics. It also captures the seasonality well.

Check Residuals for Fit Forecast

```r
checkresiduals(FitFore12)
```

```
## Warning in checkresiduals(FitFore12): 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* = 10.05, df = 8, p-value = 0.2615
##
## Model df: 2.   Total lags used: 10
```

Residuals appear to be normally distributed with some slight skewing to the left. P-values are above .05 meaning

what? ^^^

Seasonal Naive