DATA 624: PREDICTIVE ANALYTICS Project 1

Gabriel Campos

Last edited March 23, 2024

```
library(fpp3)
library(dplyr)
library(ggplot2)
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 4.3.3
```

```
library(tsibble)
library(psych)
library(tidyr)
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.3.3
```

Description

This project consists of 3 parts - two required and one bonus and is worth 15% of your grade. The project is due at 11:59 PM on Sunday Apr 11. I will accept late submissions with a penalty until the meetup after that when we review some projects.

Part A

ATM Forecast ATM624Data.xlsx

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.

Part B

Forecasting Power ResidentialCustomerForecastLoad-624.xlsx

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.

Part C

BONUS, optional (part or all), Waterflow_Pipe1.xlsx and Waterflow_Pipe2.xlsx

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 Load

https://github.com/GitableGabe/Data624_Data/raw/main/ATM624Data.xlsx

```
atm_coltype<-c("date","text","numeric")

atm_import<-read_xlsx('ATM624Data.xlsx', col_types = atm_coltype)
# Ommitting Extra Credit as I won't be working on it
# WP1_df<-read_xlsx('Waterflow_Pipe1.xlsx')
# WP2_df<-read_xlsx('Waterflow_Pipe2.xlsx')
```

```
power_raw<-read_xlsx('ResidentialCustomerForecastLoad-624.xlsx')
```

Part A

EDA & Cleanup

```
head(atm_import%>%
       filter(ATM=="ATM4"))
```

```
## # A tibble: 6 x 3
##   DATE                ATM    Cash
##   <dttm>              <chr> <dbl>
## 1 2009-05-01 00:00:00 ATM4  777.
## 2 2009-05-02 00:00:00 ATM4  524.
## 3 2009-05-03 00:00:00 ATM4  793.
## 4 2009-05-04 00:00:00 ATM4  908.
## 5 2009-05-05 00:00:00 ATM4   52.8
## 6 2009-05-06 00:00:00 ATM4   52.2
```

```
atm_range<-range(atm_import$DATE)
atm_range[1]
```

```
## [1] "2009-05-01 UTC"
```

```
atm_range[2]
```

```
## [1] "2010-05-14 UTC"
```

```r
sapply(atm_import, function(x) sum(is.na(x)))
```

```
## DATE  ATM Cash
##    0   14   19
```

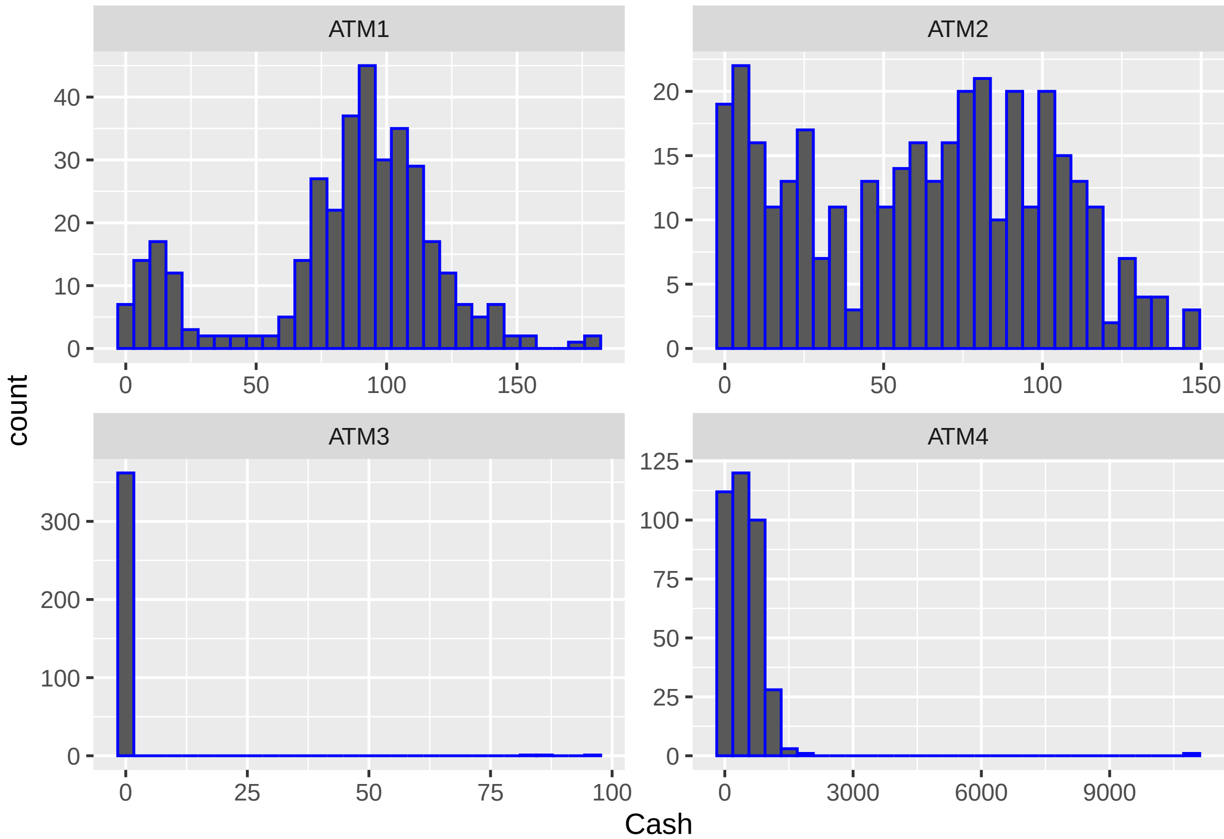
```r
data.frame(atm_import$DATE[atm_import$Cash %in% NA])
```

```
##    atm_import.DATE.atm_import.Cash..in..NA.
## 1                                2009-06-13
## 2                                2009-06-16
## 3                                2009-06-18
## 4                                2009-06-22
## 5                                2009-06-24
## 6                                2010-05-01
## 7                                2010-05-02
## 8                                2010-05-03
## 9                                2010-05-04
## 10                               2010-05-05
## 11                               2010-05-06
## 12                               2010-05-07
## 13                               2010-05-08
## 14                               2010-05-09
## 15                               2010-05-10
## 16                               2010-05-11
## 17                               2010-05-12
## 18                               2010-05-13
## 19                               2010-05-14
```

- ATM624Data had attribute type mismatches, and was converted on import.
- Date conversion somehow kept date time as POSIXct
- ATM4 shows values in greater decimals any country, with Dinars being the only Country that uses more than 2 decimals when using its currency, but even the dinar stops at the 100th decimal.
- Date range is 05-01-2009 to 05-14-2010
- we see the count of NAs in ATM is 14 and Cash column is 19
- The NA dates vary and are not exclusive to a specific sequential time period that we can just filter out.
- I am curious about the distribution of cash considering the forecast ask for this project.

```r
atm_import %>%
  filter(DATE < "2010-05-01", !is.na(ATM)) %>%
  ggplot(aes(x = Cash)) +
    geom_histogram(bins = 30, color= "blue") +
    facet_wrap(~ ATM, ncol = 2, scales = "free")
```

```
## Warning: Removed 5 rows containing non-finite values (`stat_bin()`).
```




```
(atm_df <- atm_import %>%
  mutate(DATE = as.Date(DATE)) %>%
   filter(DATE<"2010-05-01")%>%
  pivot_wider(names_from=ATM, values_from = Cash))
```

```
## # A tibble: 365 x 5
##    DATE        ATM1  ATM2  ATM3  ATM4
##    <date>     <dbl> <dbl> <dbl> <dbl>
##  1 2009-05-01    96   107     0 777.
##  2 2009-05-02    82    89     0 524.
##  3 2009-05-03    85    90     0 793.
##  4 2009-05-04    90    55     0 908.
##  5 2009-05-05    99    79     0  52.8
##  6 2009-05-06    88    19     0  52.2
##  7 2009-05-07     8     2     0  55.5
##  8 2009-05-08   104   103     0 559.
##  9 2009-05-09    87   107     0 904.
## 10 2009-05-10    93   118     0 879.
## # i 355 more rows
```

```
atm_df<-atm_df%>%
  as_tsibble(index=DATE)
head(atm_df)
```

```
## # A tsibble: 6 x 5 [1D]
```

```
##   DATE        ATM1  ATM2  ATM3  ATM4
##   <date>     <dbl> <dbl> <dbl> <dbl>
## 1 2009-05-01    96   107     0 777.
## 2 2009-05-02    82    89     0 524.
## 3 2009-05-03    85    90     0 793.
## 4 2009-05-04    90    55     0 908.
## 5 2009-05-05    99    79     0  52.8
## 6 2009-05-06    88    19     0  52.2
```

```r
summary(atm_df)
```

```
##       DATE                 ATM1             ATM2             ATM3        
##  Min.   :2009-05-01   Min.   :  1.00   Min.   :  0.00   Min.   : 0.0000  
##  1st Qu.:2009-07-31   1st Qu.: 73.00   1st Qu.: 25.50   1st Qu.: 0.0000  
##  Median :2009-10-30   Median : 91.00   Median : 67.00   Median : 0.0000  
##  Mean   :2009-10-30   Mean   : 83.89   Mean   : 62.58   Mean   : 0.7206  
##  3rd Qu.:2010-01-29   3rd Qu.:108.00   3rd Qu.: 93.00   3rd Qu.: 0.0000  
##  Max.   :2010-04-30   Max.   :180.00   Max.   :147.00   Max.   :96.0000  
##                       NA's   :3        NA's   :2                         
##       ATM4          
##  Min.   :    1.563  
##  1st Qu.:  124.334  
##  Median :  403.839  
##  Mean   :  474.043  
##  3rd Qu.:  704.507  
##  Max.   :10919.762  
## 
```

```r
atm_df[!complete.cases(atm_df), ]
```

```
## # A tsibble: 5 x 5 [1D]
##   DATE        ATM1  ATM2  ATM3  ATM4
##   <date>     <dbl> <dbl> <dbl> <dbl>
## 1 2009-06-13    NA    91     0 746.
## 2 2009-06-16    NA    82     0 373.
## 3 2009-06-18    21    NA     0  92.5
## 4 2009-06-22    NA    90     0  80.6
## 5 2009-06-24    66    NA     0  90.6
```

```r
atm_df%>%
  select(DATE,ATM3)%>%
  filter(ATM3>0)
```

```
## # A tsibble: 3 x 2 [1D]
##   DATE        ATM3
##   <date>     <dbl>
## 1 2010-04-28    96
## 2 2010-04-29    82
## 3 2010-04-30    85
```

- Converting `DATE` into a date value made senses type POSIXct may cause future issues.

- Pivoting allowed us to separate the ATM's categorically and isolate the NAs for removal.
- We are able to see that five entries contain NAs and the dates all reside in June
- ATM3 only has 3 dates with withdrawals 4-28 through 4-30 or 2010, and the distribution plot is arguably a reason to omit this column
- These results also brings to question whether there may be some seasonality that will impact May's forecasting
- Considering the distribution, I chose to replace the missing values with the median, as the skewed values in ATM 3 & 4 I believe with negatively impact the mean

```
# seasonality
atm_import %>%
  filter(DATE < "2010-05-01", !is.na(ATM)) %>%
  ggplot(aes(x = DATE, y = Cash, col = ATM)) +
    geom_line(color="blue") +
    facet_wrap(~ ATM, ncol = 2, scales = "free_y")+
  labs(title = "Seasonality Plot", x = "Date", y = "Cash") +
    theme_minimal()
```


```
median_value <- median(atm_df[["ATM1"]], na.rm = TRUE)
atm_df[["ATM1"]][is.na(atm_df[["ATM1"]])] <- median_value
median_value <- median(atm_df[["ATM2"]], na.rm = TRUE)
atm_df[["ATM2"]][is.na(atm_df[["ATM2"]])] <- median_value
```

```r
atm_df[!complete.cases(atm_df), ]
```

```
## # A tsibble: 0 x 5 [?]
## # i 5 variables: DATE <date>, ATM1 <dbl>, ATM2 <dbl>, ATM3 <dbl>, ATM4 <dbl>
```

Forecasts

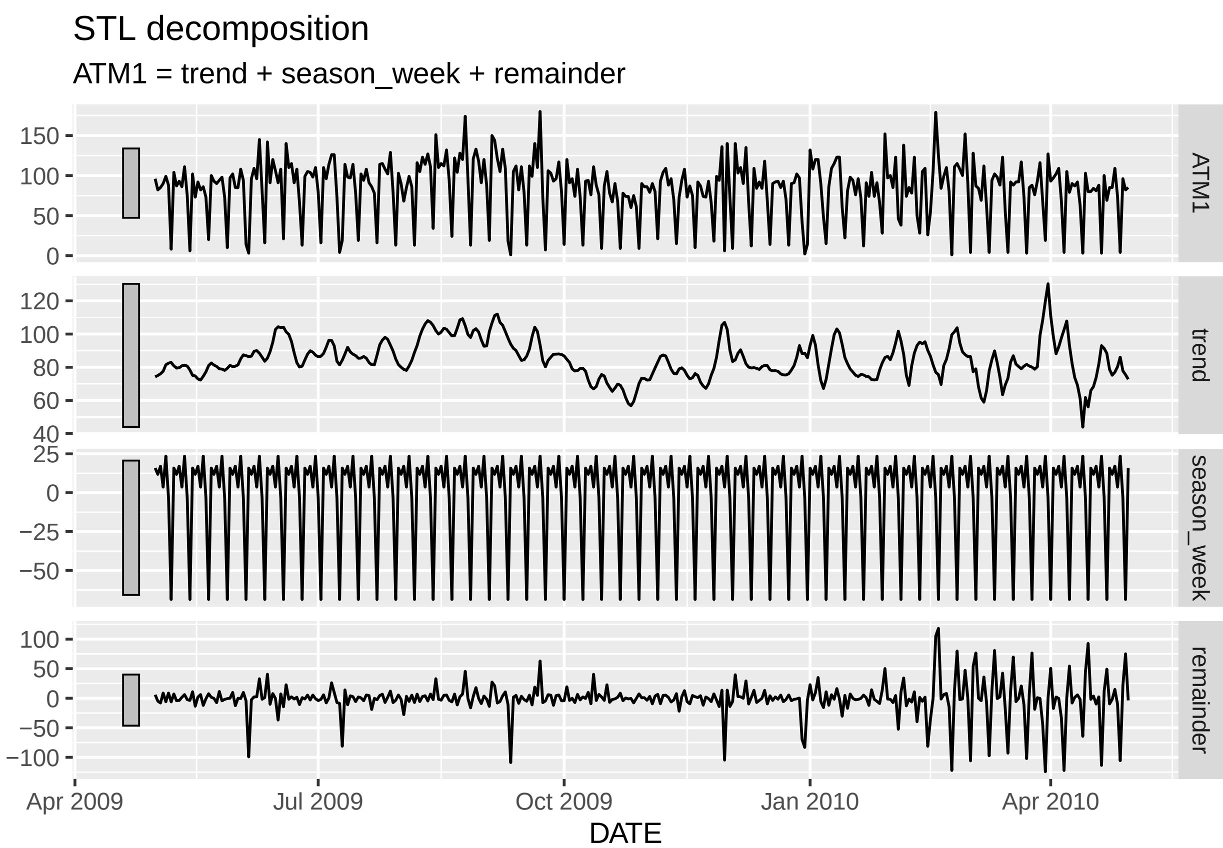
ATM1

STL Decomposition The seasonality plot did not show a trend in the long term but a better assessment in weekly interval is likely needed, using resources from Rob J Hyndman and George Athanasopoulos, Forecasting: Principles and Practice (3rd ed) section 3.6 STL decomposition I will perform a STL "Seasonal and Trend decomposition using Loess" decomposition of the series. To make it weekly I'll set the parameter `trend(window = 7)` and the `season(window='periodic')` to impose seasonality element across days of the week.

My reference come directly from the chapter.

```
us_retail_employment |>
  model(
    STL(Employed ~ trend(window = 7) +
                   season(window = "periodic"),
    robust = TRUE)) |>
  components() |>
  autoplot()
```

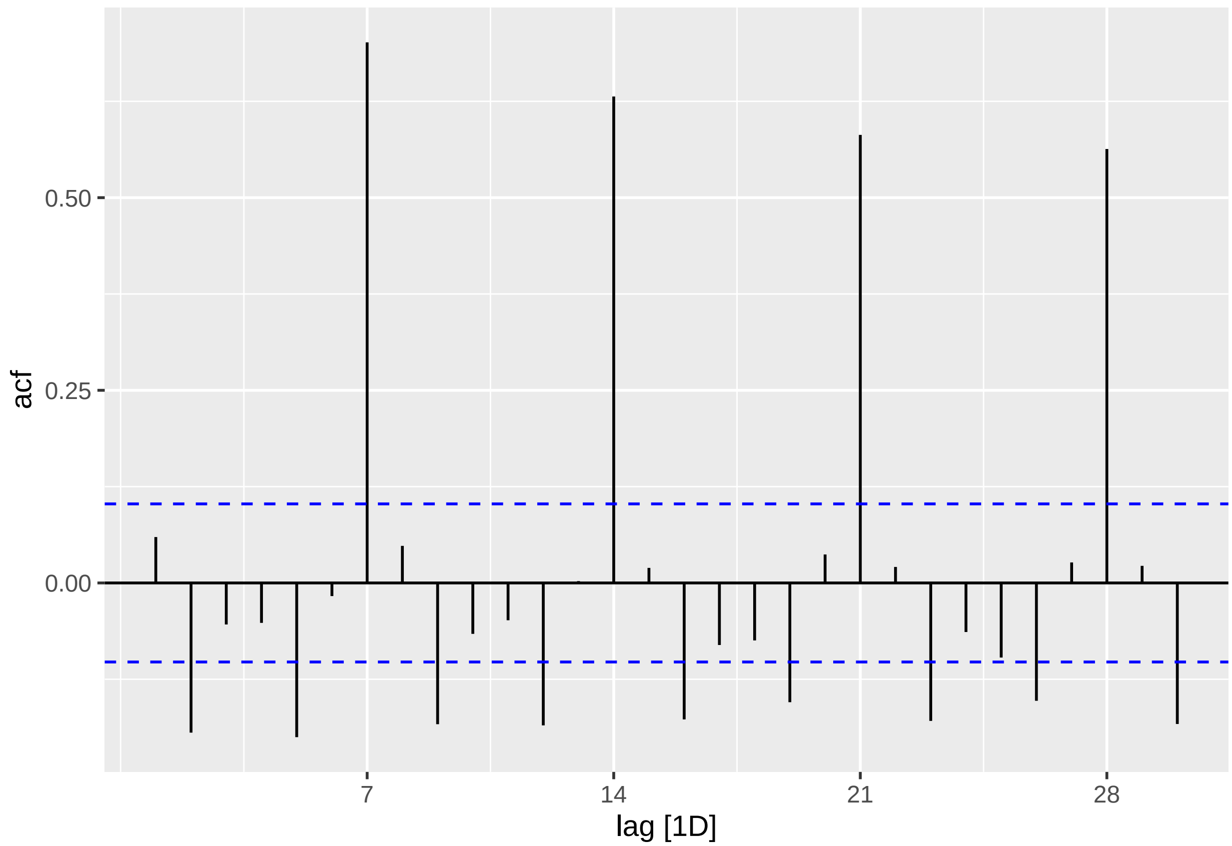
```r
atm1_df <- atm_df %>%
  dplyr::select(DATE, ATM1)

atm1_df %>%
  model(
    STL(ATM1 ~ trend(window = 7) +
                 season(window = "periodic"),
    robust = TRUE)) %>%
  components() %>%
  autoplot()
```




```r
ndiffs(atm1_df$ATM1)
```

```
## [1] 0
```

```r
atm1_df %>%
  ACF(ATM1, lag_max = 30) %>%
  autoplot()
```

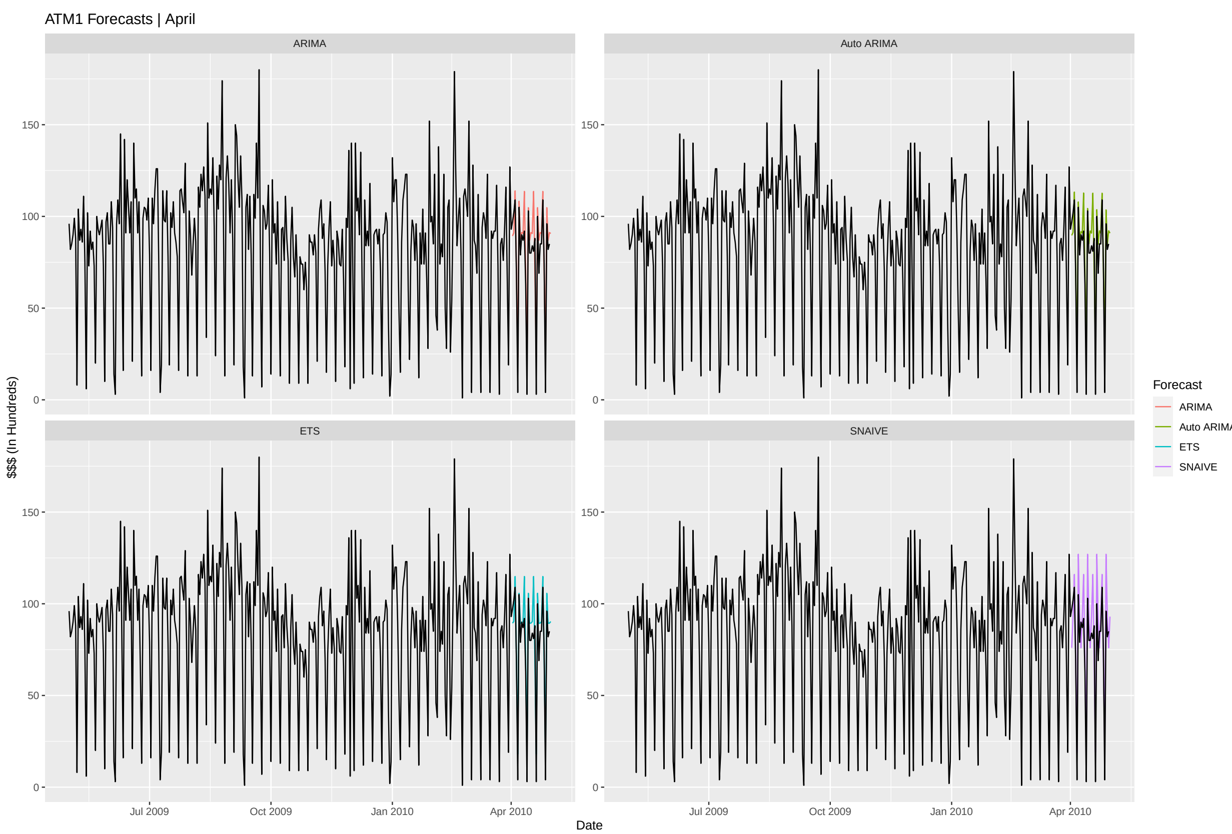


The STL decomposition wasn't as telling as I would have liked, however the ACF plot presents lags at 2, 5, and 7. I believe, given the week starts on Sunday, that this represents Monday, Thursday and Saturday as the days with the most lag. 7 has shown the value with the most significant lag. There is a decreasing trend with the ACF plot, and supports that the data is non-stationary would require differencing however $r_1's$ small value and the results of the `ndiff()` function, showing the first number of differences as 0, negates that suspicion.

ARIMA Seasonal naive method was my preferred choice considering the seasonality, and so we can use the prior time period's withdrawals to conduct our forecast, but I also like to default to `Auto ARIMA` for the optimized selection. I assume ETS and ARIMA wont perform as well but will await for the comparisons. Below we filter out the data residing in May, the month we are forecasting.

```
# train
atm1_train <- atm1_df %>%
  filter(DATE <= "2010-04-01")


atm1_fit <- atm1_train %>%
  model(
    SNAIVE = SNAIVE(ATM1),
    ETS = ETS(ATM1),
    ARIMA = ARIMA(ATM1),
    `Auto ARIMA` = ARIMA(ATM1, stepwise = FALSE, approx = FALSE)
  )

# forecast April
```

```r
atm1_forecast <- atm1_fit %>%
  forecast(h = 30)

#plot
atm1_forecast %>%
  autoplot(atm1_df, level = NULL)+
  facet_wrap( ~ .model, scales = "free_y") +
  guides(colour = guide_legend(title = "Forecast"))+
  labs(title= "ATM1 Forecasts | April") +
  xlab("Date") +
  ylab("$$$ (In Hundreds)")
```




```r
# RMSE
accuracy(atm1_forecast, atm1_df) %>%
  select(.model, RMSE:MAPE)
```

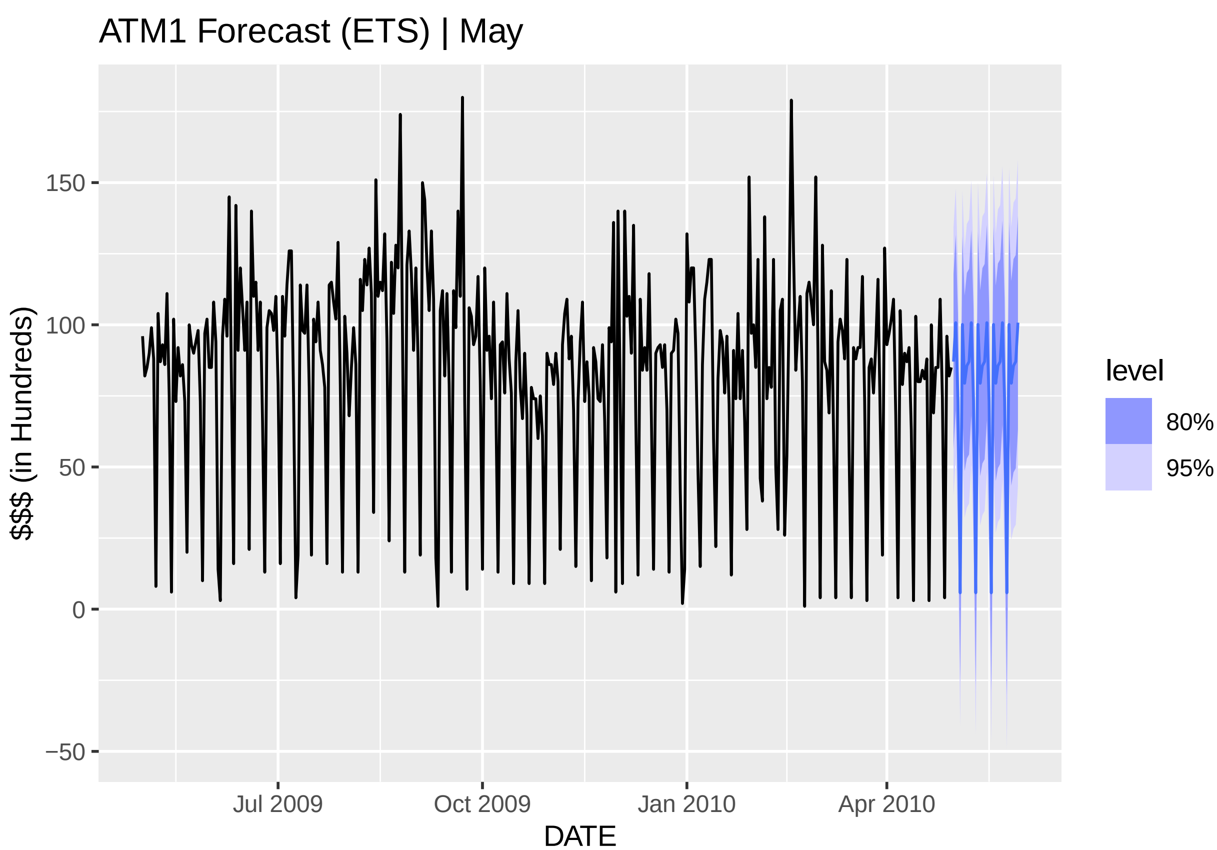
```
## # A tibble: 4 x 5
##   .model       RMSE   MAE   MPE  MAPE
##   <chr>       <dbl> <dbl> <dbl> <dbl>
## 1 ARIMA        12.6 10.0  -85.8  88.9
## 2 Auto ARIMA   13.0 10.1  -98.9 102.
## 3 ETS          12.1  9.55 -64.5  67.8
## 4 SNAIVE       16.8 14.5  -69.5  76.6
```

When interpreting the results, the model with the lowest RMSE and MAE value and the MPE and MAPE values closes to zero the best performing. This is true in all cases for ETS indicating it is the best performing.

Forecast ** Reference**

```
aus_economy |>
  model(ETS(Population)) |>
  forecast(h = "5 years") |>
  autoplot(aus_economy |> filter(Year >= 2000)) +
  labs(title = "Australian population",
       y = "People (millions)")
```

```
# remade the model from source
atm1_fit_ets <- atm1_df %>%
  model(ETS = ETS(ATM1))

atm1_forecast_ets <- atm1_fit_ets %>%
  forecast(h=30)

atm1_forecast_ets %>%
  autoplot(atm1_df) +
  labs(title = "ATM1 Forecast (ETS) | May",
       y = "$$$ (in Hundreds)")
```

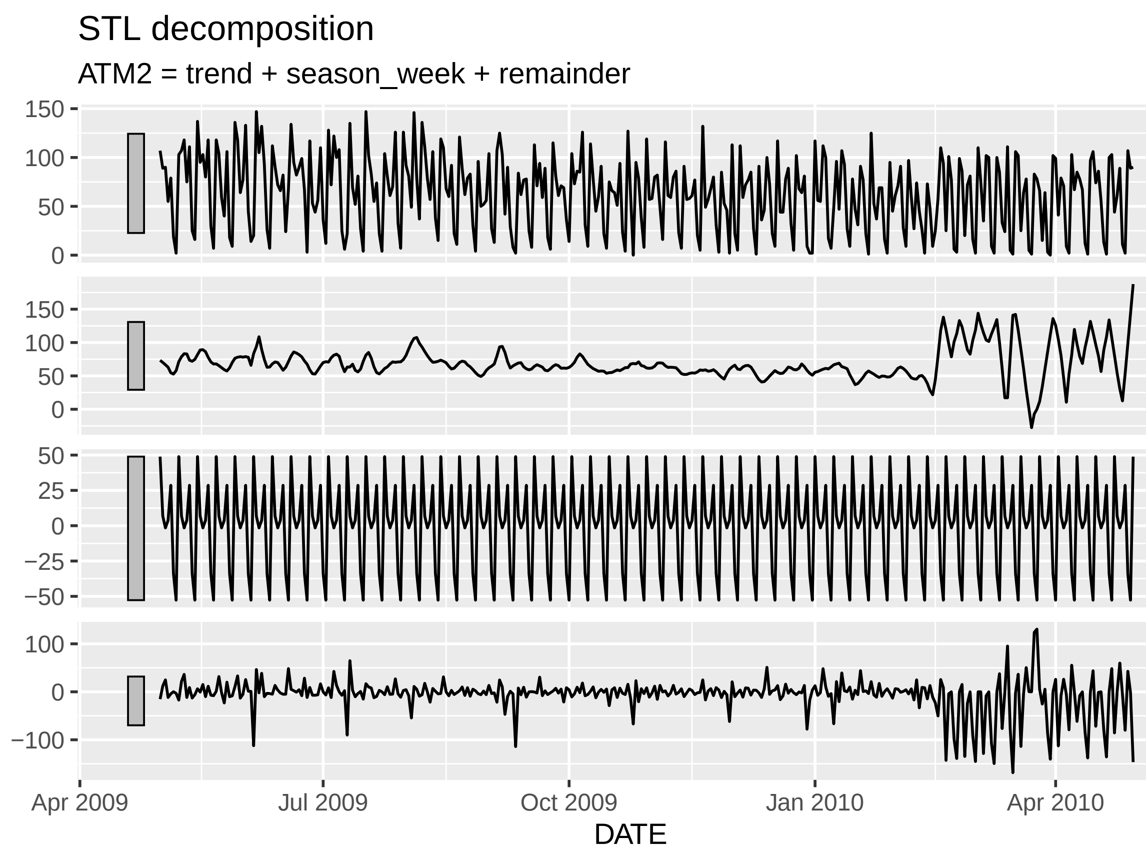



```
(atm1_forecast_results <-
  as.data.frame(atm1_forecast_ets) %>%
    select(DATE, .mean) %>%
      rename(Date = DATE, Cash = .mean)%>%
        mutate(Cash=round(Cash,2)))
```

```
##          Date   Cash
## 1  2010-05-01  87.05
## 2  2010-05-02 100.76
## 3  2010-05-03  73.11
## 4  2010-05-04   5.74
## 5  2010-05-05 100.13
## 6  2010-05-06  79.43
## 7  2010-05-07  85.60
## 8  2010-05-08  87.05
## 9  2010-05-09 100.76
## 10 2010-05-10  73.11
## 11 2010-05-11   5.74
## 12 2010-05-12 100.13
## 13 2010-05-13  79.43
## 14 2010-05-14  85.60
## 15 2010-05-15  87.05
## 16 2010-05-16 100.76
## 17 2010-05-17  73.11
## 18 2010-05-18   5.74
## 19 2010-05-19 100.13
## 20 2010-05-20  79.43
## 21 2010-05-21  85.60
## 22 2010-05-22  87.05
## 23 2010-05-23 100.76
## 24 2010-05-24  73.11
## 25 2010-05-25   5.74
## 26 2010-05-26 100.13
## 27 2010-05-27  79.43
## 28 2010-05-28  85.60
## 29 2010-05-29  87.05
## 30 2010-05-30 100.76
```

ATM2

```r
atm2_df <- atm_df %>%
  dplyr::select(DATE, ATM2)

atm2_df %>%
  model(
    STL(ATM2 ~ trend(window = 7) +
                season(window = "periodic"),
    robust = TRUE)) %>%
  components() %>%
  autoplot()
```



STL Decomposition

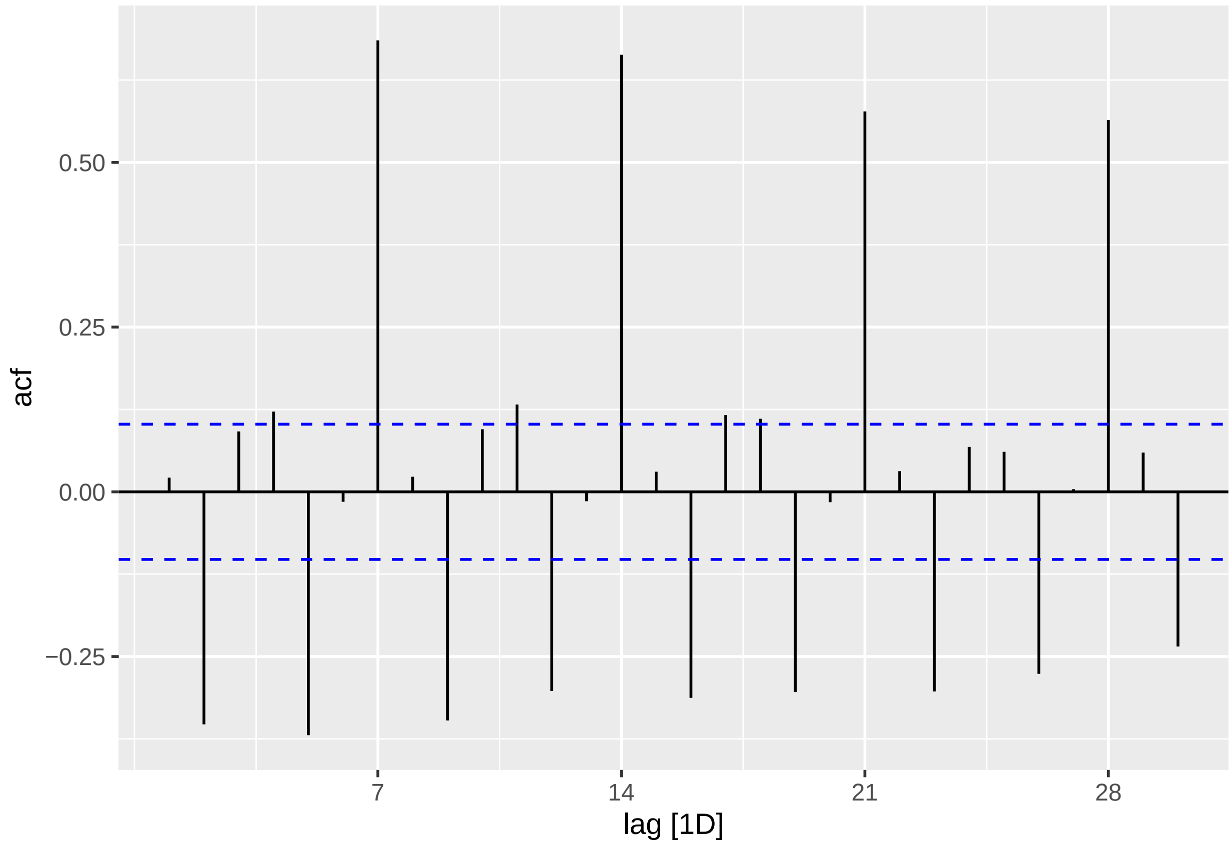
```r
ndiffs(atm2_df$ATM2)
```

```
## [1] 1
```

```r
unitroot_ndiffs(atm2_df$ATM2)
```

```
## ndiffs
##      1
```

```r
atm2_df %>%
  ACF(ATM2, lag_max = 30) %>%
  autoplot()
```



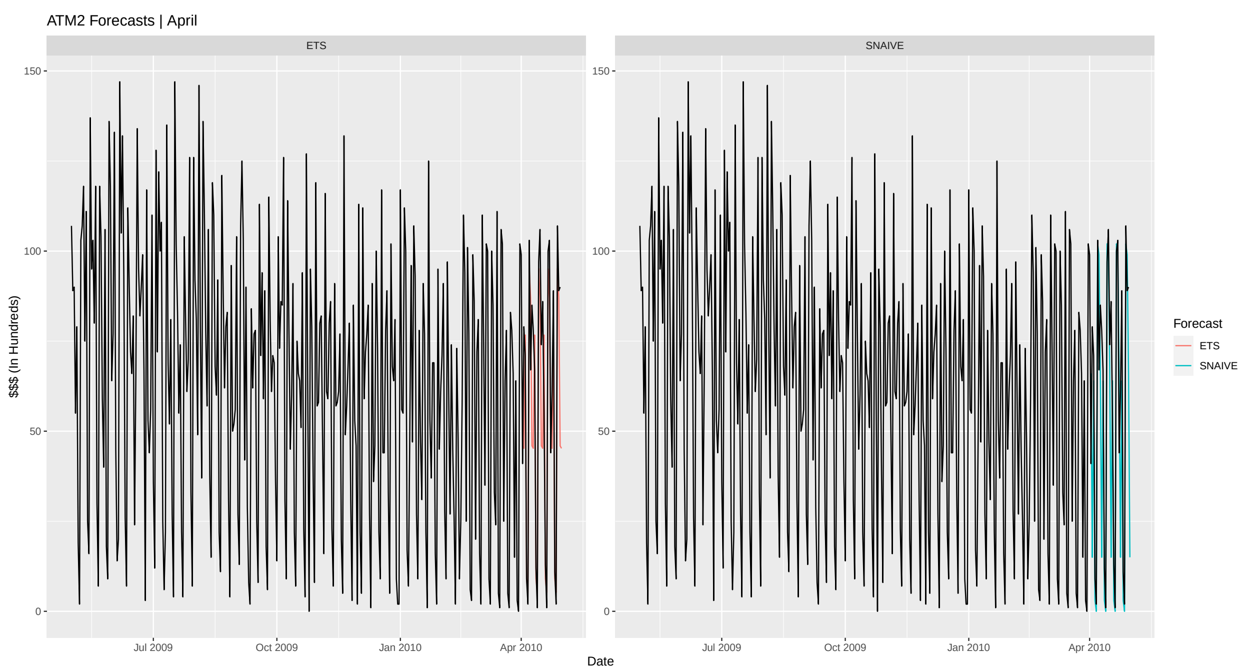
The approach with ATM2 is a rinse and repeat but in this case differencing is needed and achieved with the below code

```
atm2_df <- atm2_df %>%
  mutate(diff_ATM2= difference(ATM2))
```

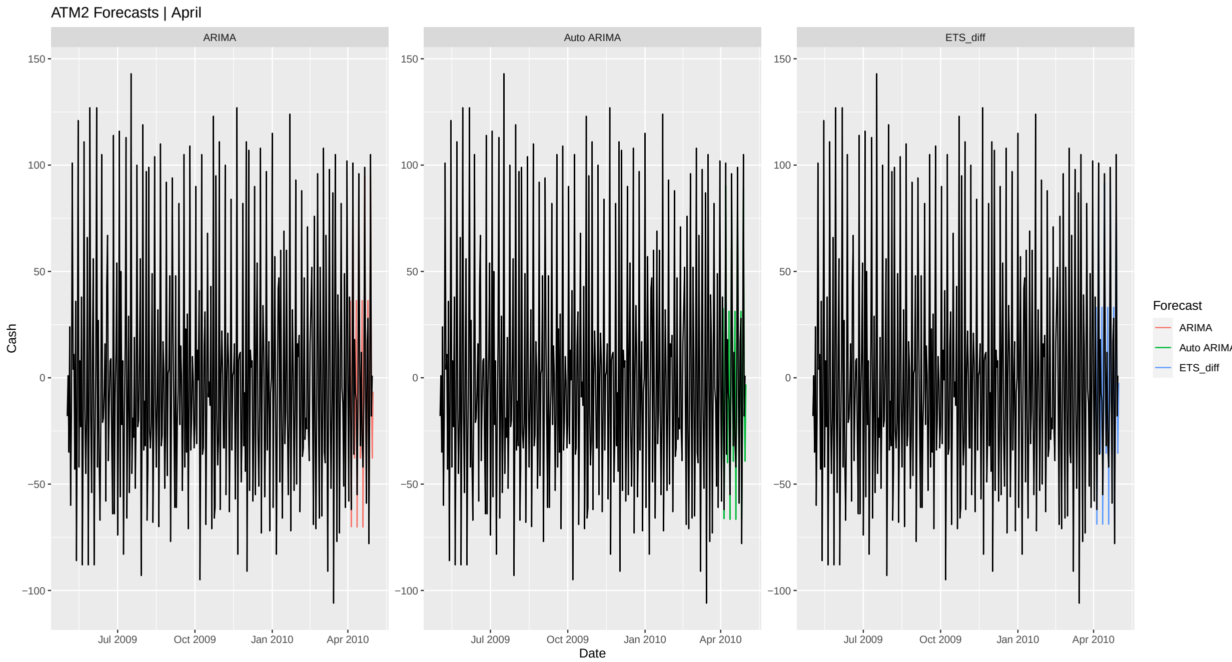
ARIMA Below we again filter out data and identify our best model but include both differenced and non-differenced data.

```
atm2_train <- atm2_df %>%
  filter(DATE <= "2010-04-01")

#run seasonal related models without the differenced data
atm2_fit_nondiff <- atm2_train %>%
  model(
    SNAIVE = SNAIVE(ATM2),
    ETS = ETS(ATM2),
  )

#run models with differenced data
atm2_fit_diff <- atm2_train %>%
  slice(2:336) %>%
  model(
    ETS_diff = ETS(diff_ATM2),
    ARIMA = ARIMA(diff_ATM2),
```

```
  `Auto ARIMA` = ARIMA(diff_ATM2, stepwise = FALSE, approx = FALSE)
 )

#forecast_ATM2 April
atm2_forecast_nondiff <- atm2_fit_nondiff %>%
  forecast(h = 30)

#forecast_ATM2 April
atm2__forecast_diff <- atm2_fit_diff %>%
  forecast(h = 30)

#plot
atm2_forecast_nondiff %>%
  autoplot(atm2_df, level = NULL)+
  facet_wrap( ~ .model, scales = "free_y") +
  guides(colour = guide_legend(title = "Forecast"))+
  labs(title= "ATM2 Forecasts | April") +
  xlab("Date") +
  ylab("$$$ (In Hundreds)")
```




```
#plot 2
atm2__forecast_diff %>%
  autoplot(atm2_df, level = NULL)+
  facet_wrap( ~ .model, scales = "free_y") +
  guides(colour = guide_legend(title = "Forecast"))+
  labs(title= "ATM2 Forecasts | April") +
  xlab("Date") +
  ylab("Cash")
```




```r
accuracy(atm2_forecast_nondiff, atm2_df) %>%
  select(.model, RMSE:MAPE)
```

```
## # A tibble: 2 x 5
##   .model  RMSE   MAE   MPE  MAPE
##   <chr>  <dbl> <dbl> <dbl> <dbl>
## 1 ETS     19.0  13.7 -29.3  59.4
## 2 SNAIVE  26.0  16.9  32.3  45.6
```

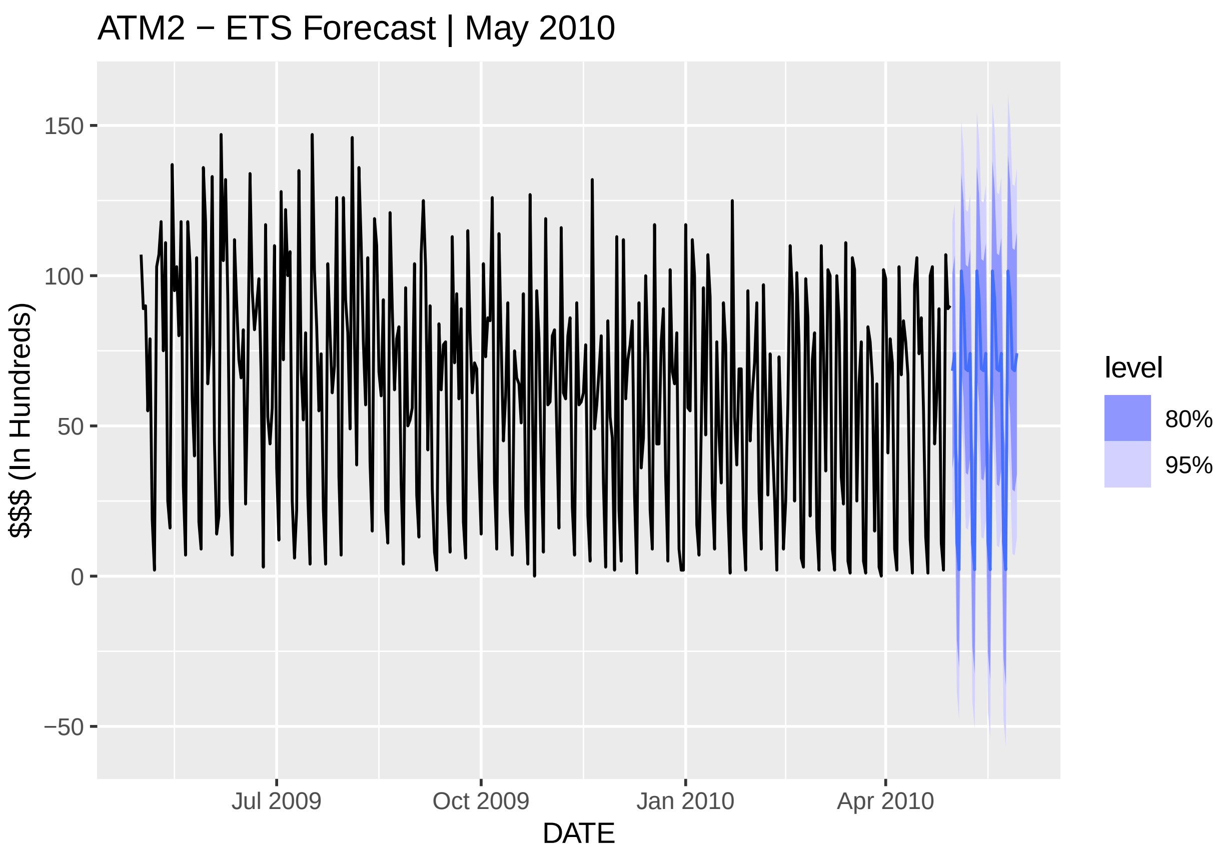
```r
accuracy(atm2__forecast_diff, atm2_df) %>%
  select(.model, RMSE:MAPE)
```

```
## # A tibble: 3 x 5
##   .model      RMSE   MAE   MPE  MAPE
##   <chr>      <dbl> <dbl> <dbl> <dbl>
## 1 ARIMA       26.2  19.5  228.  239.
## 2 Auto ARIMA  25.2  19.1  234.  242.
## 3 ETS_diff    25.0  19.1  220.  229.
```

Among the reuslts, the non-difference ETS model had the lowest RMSE & MAE, and MPE & MAPE closest to zero, making it the optimal choice.

```r
atm2_fit_ets <- atm2_df %>%
  model(
    ETS = ETS(ATM2))

#generate the values
atm2_forecast_ets <- atm2_fit_ets %>%
```

```r
  forecast(h=30)

#plot
atm2_forecast_ets %>%
  autoplot(atm2_df) +
  labs(title = "ATM2 - ETS Forecast | May 2010",
       y = "$$$ (In Hundreds)")
```



Forecast

```r
(atm2_forecast_results <-
  as.data.frame(atm2_forecast_ets) %>%
    select(DATE, .mean) %>%
      rename(Date = DATE, Cash = .mean)%>%
        mutate(Cash=round(Cash,2)))
```

```
##          Date   Cash
## 1  2010-05-01  68.35
## 2  2010-05-02  74.19
## 3  2010-05-03  11.09
## 4  2010-05-04   2.14
## 5  2010-05-05 101.60
## 6  2010-05-06  92.38
## 7  2010-05-07  68.98
## 8  2010-05-08  68.35
## 9  2010-05-09  74.19
## 10 2010-05-10  11.09
```

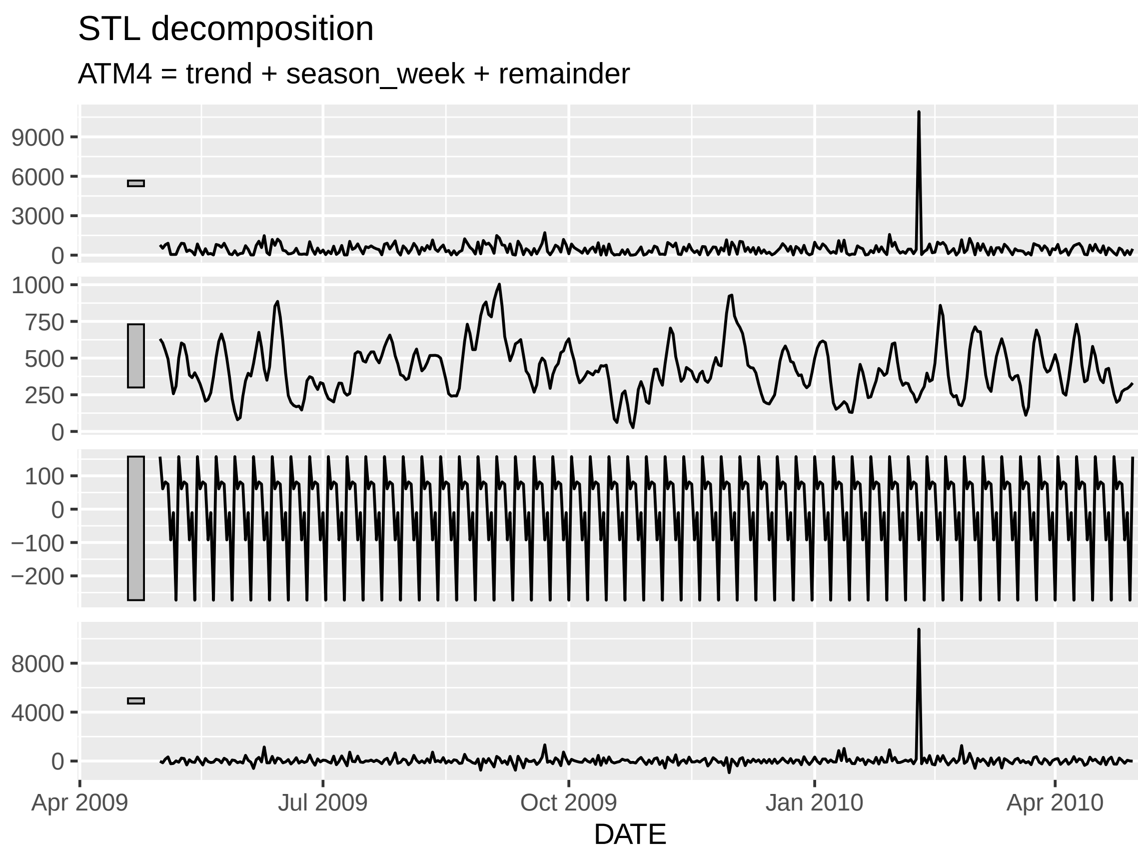
```
## 11 2010-05-11   2.14
## 12 2010-05-12 101.60
## 13 2010-05-13  92.38
## 14 2010-05-14  68.98
## 15 2010-05-15  68.35
## 16 2010-05-16  74.19
## 17 2010-05-17  11.09
## 18 2010-05-18   2.14
## 19 2010-05-19 101.60
## 20 2010-05-20  92.38
## 21 2010-05-21  68.98
## 22 2010-05-22  68.35
## 23 2010-05-23  74.19
## 24 2010-05-24  11.09
## 25 2010-05-25   2.14
## 26 2010-05-26 101.60
## 27 2010-05-27  92.38
## 28 2010-05-28  68.98
## 29 2010-05-29  68.35
## 30 2010-05-30  74.19
```

ATM3

ATM3 was ultimately omitted, considering the limited date range and skewed distributions. It can be considered when more data is provided.

ATM4

```
atm4_df <- atm_df %>%
  select(DATE, ATM4)

atm4_df %>%
  model(
    STL(ATM4 ~ trend(window = 7) +
                 season(window = "periodic"),
    robust = TRUE)) %>%
  components() %>%
  autoplot()
```



STL Decomposition

Considering the variance from the time series, I decided to tranform the data before forecasting using box-cox transformation

Box-Cox Reference

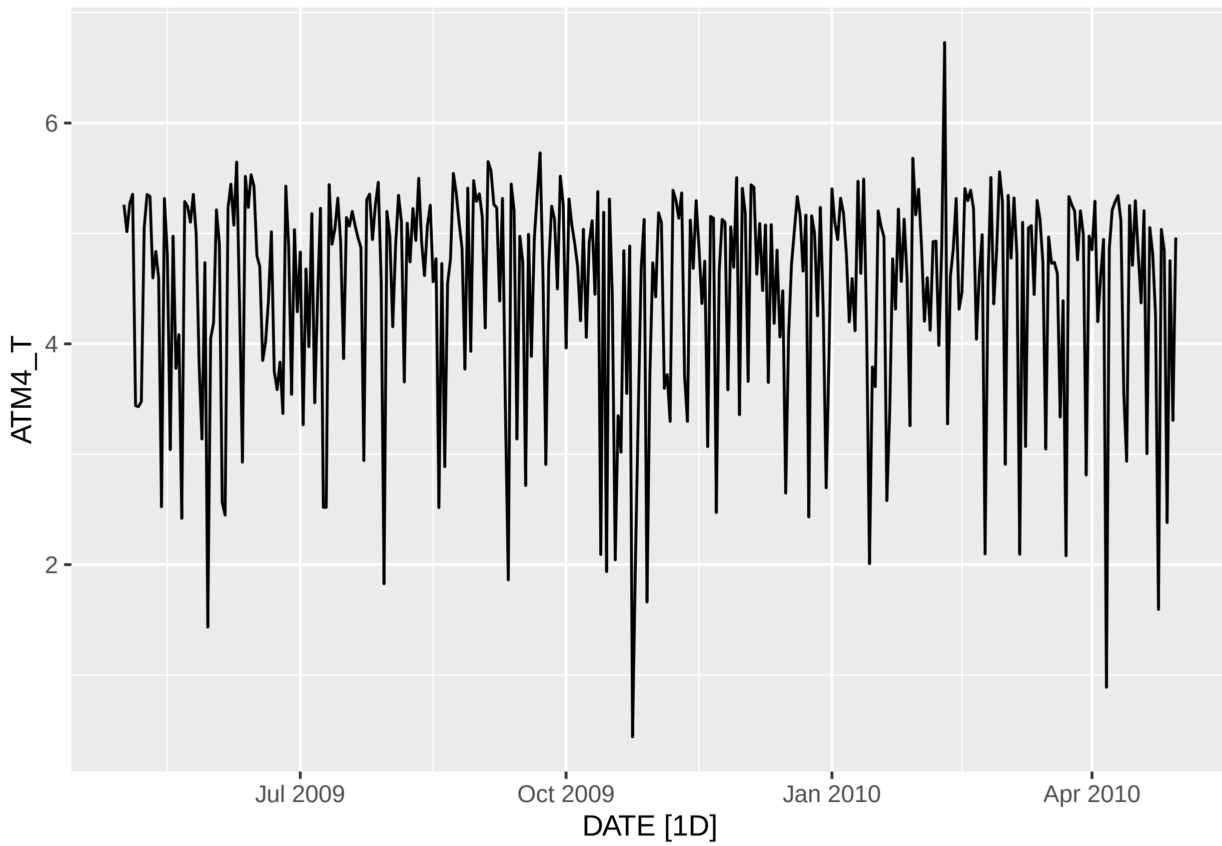
Forecasting Principles and Practice

```
lambda <- aus_production |>
  features(Gas, features = guerrero) |>
  pull(lambda_guerrero)
aus_production |>
  autoplot(box_cox(Gas, lambda)) +
  labs(y = "",
       title = latex2exp::TeX(paste0(
         "Transformed gas production with $\\lambda$ = ",
         round(lambda,2))))
```

```
(atm4_lambda <- atm4_df %>%
  features(ATM4, features = guerrero) %>%
  pull(lambda_guerrero))
```

```
## [1] -0.0737252
```

```r
atm4_transformed <- BoxCox(atm4_df$ATM4, lambda = atm4_lambda)

# Extract the transformed data

atm4_df$ATM4_T<-atm4_transformed

#plot
atm4_df%>%
  autoplot(ATM4_T)
```



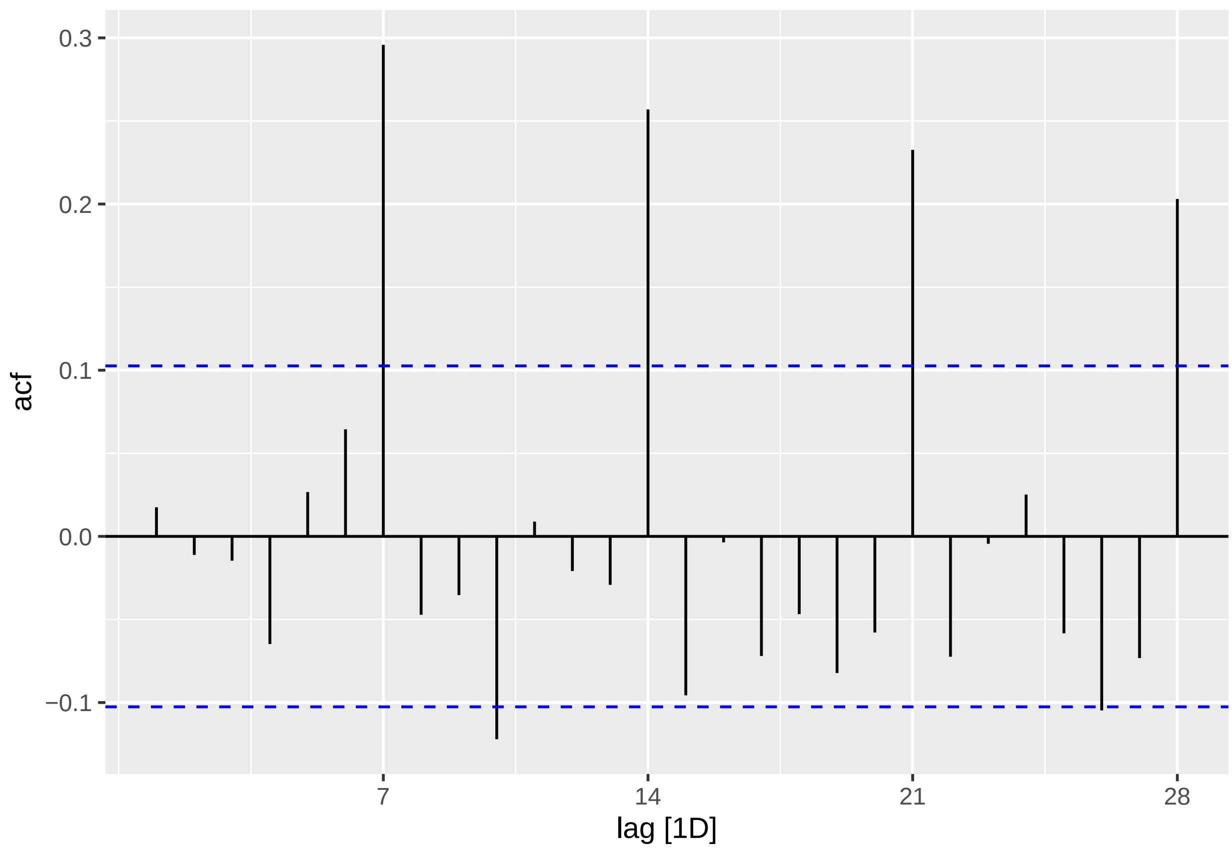

```r
ndiffs(atm4_df$ATM4)
```

```
## [1] 0
```

```r
ndiffs(atm4_df$ATM4_T)
```

```
## [1] 0
```

```r
atm4_df %>%
  ACF(ATM4_T, lag_max = 28) %>%
  autoplot()
```



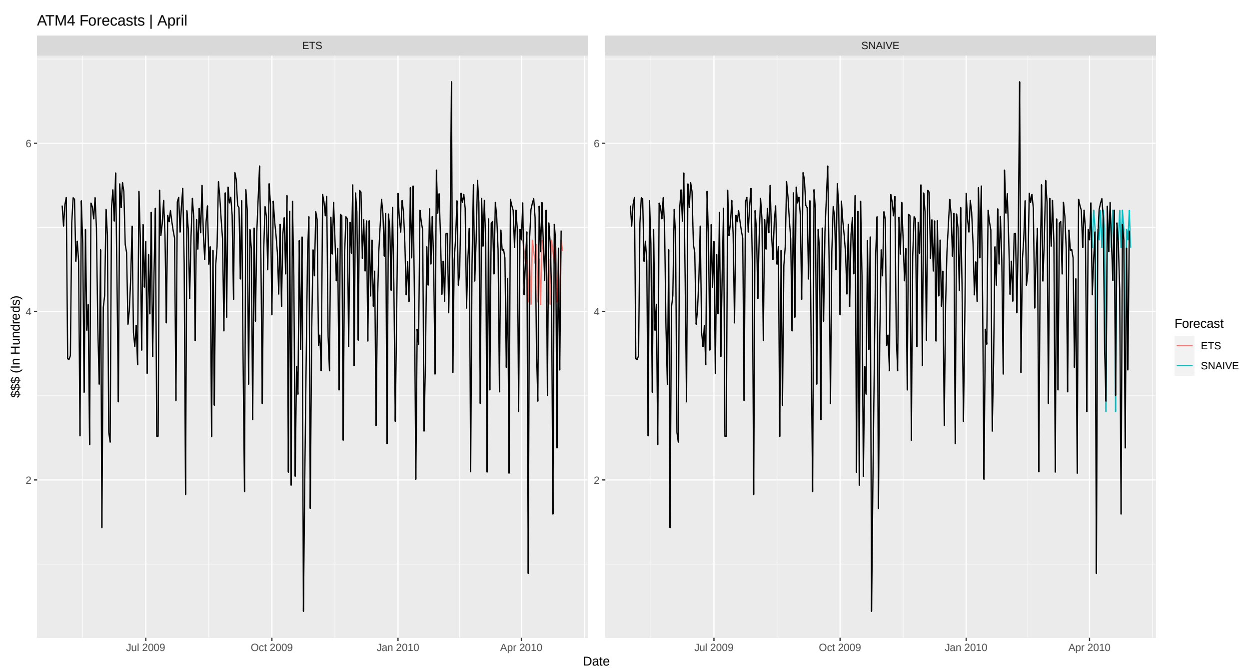
Using ndiff() we identify that theres no need for differencing, and the ACF shows

The ACF plot below suggest lags 7 consistently and on 2 other occasions in different periods. Despite the ndiff() function resulting in 0, if believe this does require differencing using the transformed data.

```
atm4_df <- atm4_df %>%
  mutate(diff_ATM4= difference(ATM4_T))
```

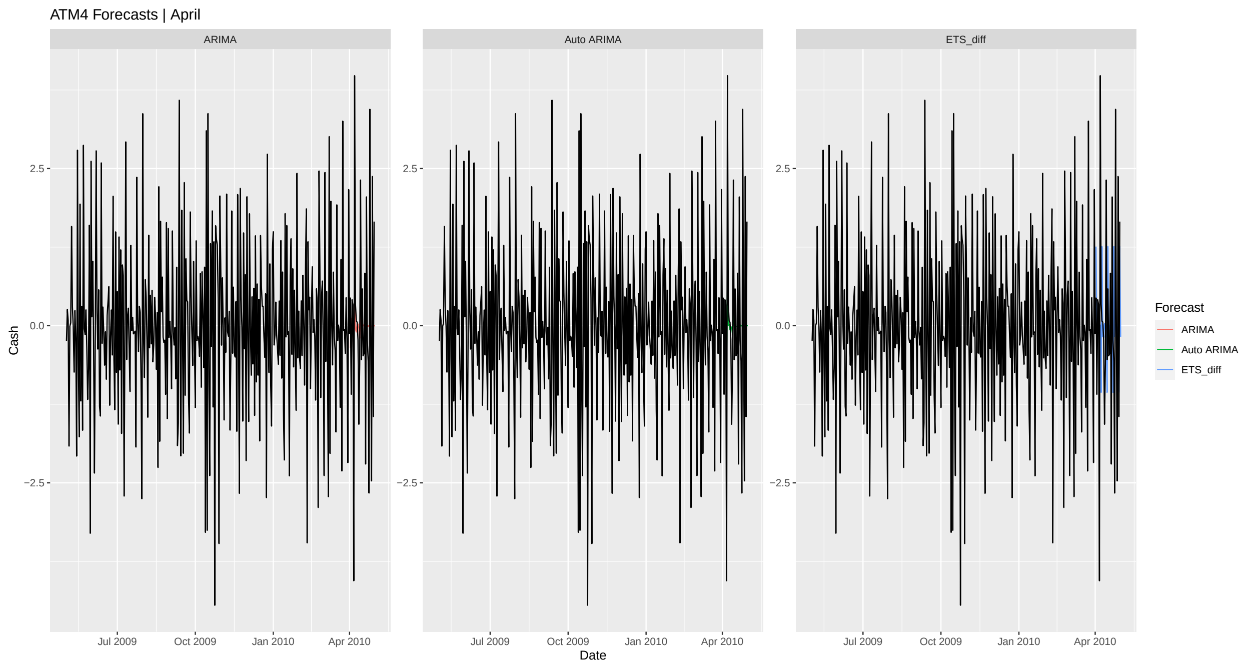
```
atm4_train <- atm4_df %>%
  filter(DATE <= "2010-04-01")

#run seasonal related models without the differenced data
atm4_fit_nondiff <- atm4_train %>%
  model(
    SNAIVE = SNAIVE(ATM4_T),
    ETS = ETS(ATM4_T),
  )

#run models with differenced data
atm4_fit_diff <- atm4_train %>%
  slice(2:336) %>%
  model(
    ETS_diff = ETS(diff_ATM4),
    ARIMA = ARIMA(diff_ATM4),
```

```
  `Auto ARIMA` = ARIMA(diff_ATM4, stepwise = FALSE, approx = FALSE)
  )

#forecast_ATM2 April
atm4_forecast_nondiff <- atm4_fit_nondiff %>%
  forecast(h = 30)

#forecast_ATM2 April
atm4__forecast_diff <- atm4_fit_diff %>%
  forecast(h = 30)

#plot
atm4_forecast_nondiff %>%
  autoplot(atm4_df, level = NULL)+
  facet_wrap( ~ .model, scales = "free_y") +
  guides(colour = guide_legend(title = "Forecast"))+
  labs(title= "ATM4 Forecasts | April") +
  xlab("Date") +
  ylab("$$$ (In Hundreds)")
```



ARIMA

```
#plot 2
atm4__forecast_diff %>%
  autoplot(atm4_df, level = NULL)+
  facet_wrap( ~ .model, scales = "free_y") +
  guides(colour = guide_legend(title = "Forecast"))+
  labs(title= "ATM4 Forecasts | April") +
  xlab("Date") +
  ylab("Cash")
```




```
accuracy(atm4_forecast_nondiff, atm4_df) %>%
  select(.model, RMSE:MAPE)
```

```
## # A tibble: 2 x 5
##   .model  RMSE   MAE   MPE  MAPE
##   <chr>  <dbl> <dbl> <dbl> <dbl>
## 1 ETS     1.07 0.740 -22.3  32.8
## 2 SNAIVE 0.861 0.510 -19.5  22.7
```

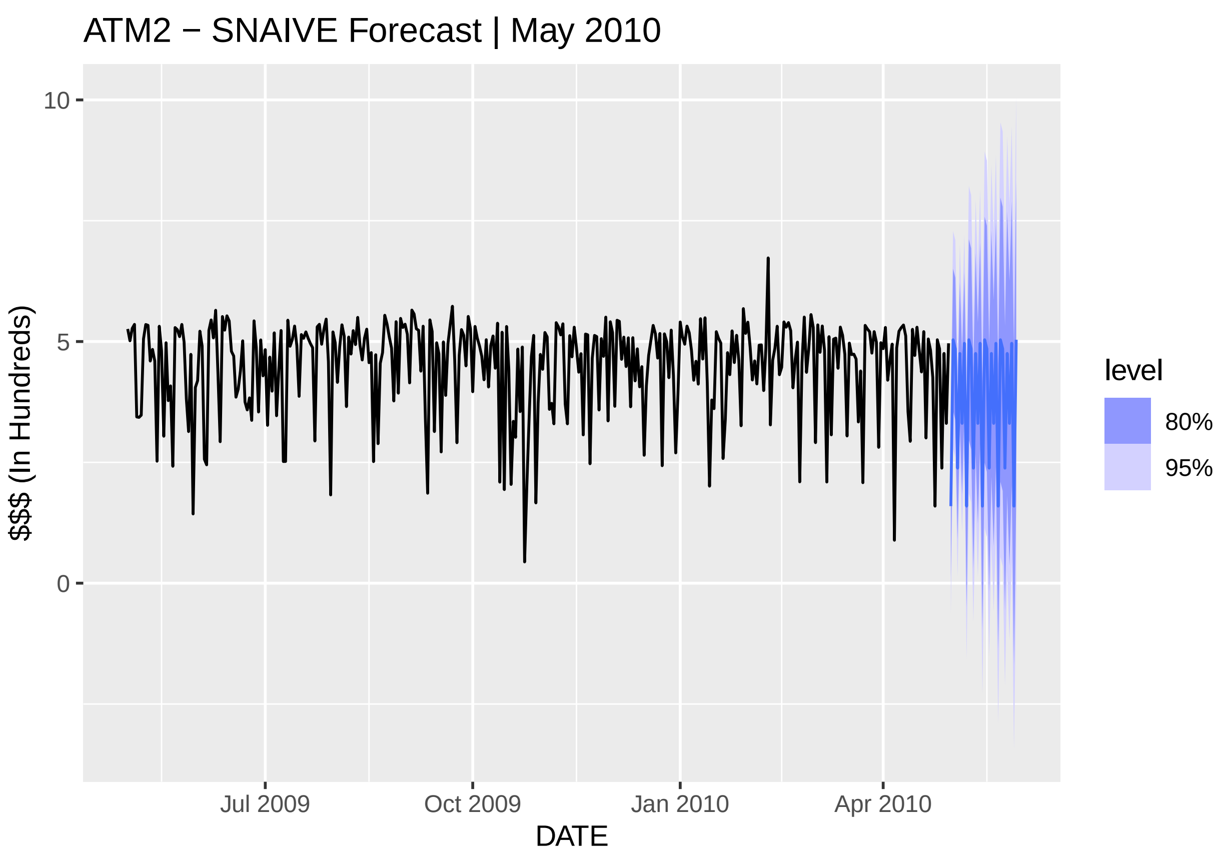
```
accuracy(atm4__forecast_diff, atm4_df) %>%
  select(.model, RMSE:MAPE)
```

```
## # A tibble: 3 x 5
##   .model      RMSE   MAE   MPE  MAPE
##   <chr>      <dbl> <dbl> <dbl> <dbl>
## 1 ARIMA       1.64  1.26 104.   104.
## 2 Auto ARIMA  1.66  1.26 103.   103.
## 3 ETS_diff    1.73  1.33  21.7  184.
```

Of the models, SNAIVE for non differenced data was the most accurate so I will proceed with this.

```
atm4_fit_snaive <- atm4_df %>%
  model(
    SNAIVE = SNAIVE(ATM4_T))

#generate the values
atm4_forecast_snaive <- atm4_fit_snaive %>%
  forecast(h=30)
```

```r
#plot
atm4_forecast_snaive %>%
  autoplot(atm4_df) +
  labs(title = "ATM2 - SNAIVE Forecast | May 2010",
       y = "$$$ (In Hundreds)")
```



Forecast

```r
(atm4_forecast_results <-
  as.data.frame(atm4_forecast_snaive) %>%
    select(DATE, .mean) %>%
      rename(Date = DATE, Cash = .mean)%>%
        mutate(Cash=round(Cash,2)))
```

```
##          Date Cash
## 1  2010-05-01 1.59
## 2  2010-05-02 5.04
## 3  2010-05-03 4.85
## 4  2010-05-04 2.38
## 5  2010-05-05 4.75
## 6  2010-05-06 3.31
## 7  2010-05-07 4.96
## 8  2010-05-08 1.59
## 9  2010-05-09 5.04
## 10 2010-05-10 4.85
## 11 2010-05-11 2.38
```

```
## 12 2010-05-12 4.75
## 13 2010-05-13 3.31
## 14 2010-05-14 4.96
## 15 2010-05-15 1.59
## 16 2010-05-16 5.04
## 17 2010-05-17 4.85
## 18 2010-05-18 2.38
## 19 2010-05-19 4.75
## 20 2010-05-20 3.31
## 21 2010-05-21 4.96
## 22 2010-05-22 1.59
## 23 2010-05-23 5.04
## 24 2010-05-24 4.85
## 25 2010-05-25 2.38
## 26 2010-05-26 4.75
## 27 2010-05-27 3.31
## 28 2010-05-28 4.96
## 29 2010-05-29 1.59
## 30 2010-05-30 5.04
```

Part B

EDA & Cleanup

```
str(power_raw)
```

```
## tibble [192 x 3] (S3: tbl_df/tbl/data.frame)
##  $ CaseSequence: num [1:192] 733 734 735 736 737 738 739 740 741 742 ...
##  $ YYYY-MMM    : chr [1:192] "1998-Jan" "1998-Feb" "1998-Mar" "1998-Apr" ...
##  $ KWH         : num [1:192] 6862583 5838198 5420658 5010364 4665377 ...
```

```
describe(power_raw)
```

```
##              vars   n      mean         sd    median   trimmed        mad
## CaseSequence    1 192     828.5      55.57     828.5     828.5      71.16
## YYYY-MMM*       2 192      96.5      55.57      96.5      96.5      71.16
## KWH             3 191 6502474.6 1447570.89 6283324.0 6439474.9 1543073.77
##                 min      max   range skew kurtosis        se
## CaseSequence    733      924     191 0.00    -1.22      4.01
## YYYY-MMM*         1      192     191 0.00    -1.22      4.01
## KWH          770523 10655730 9885207 0.17     0.45 104742.55
```

```
data.frame(power_raw$`YYYY-MMM`[power_raw$KWH %in% NA])
```

```
##   power_raw..YYYY.MMM..power_raw.KWH..in..NA.
## 1                                    2008-Sep
```

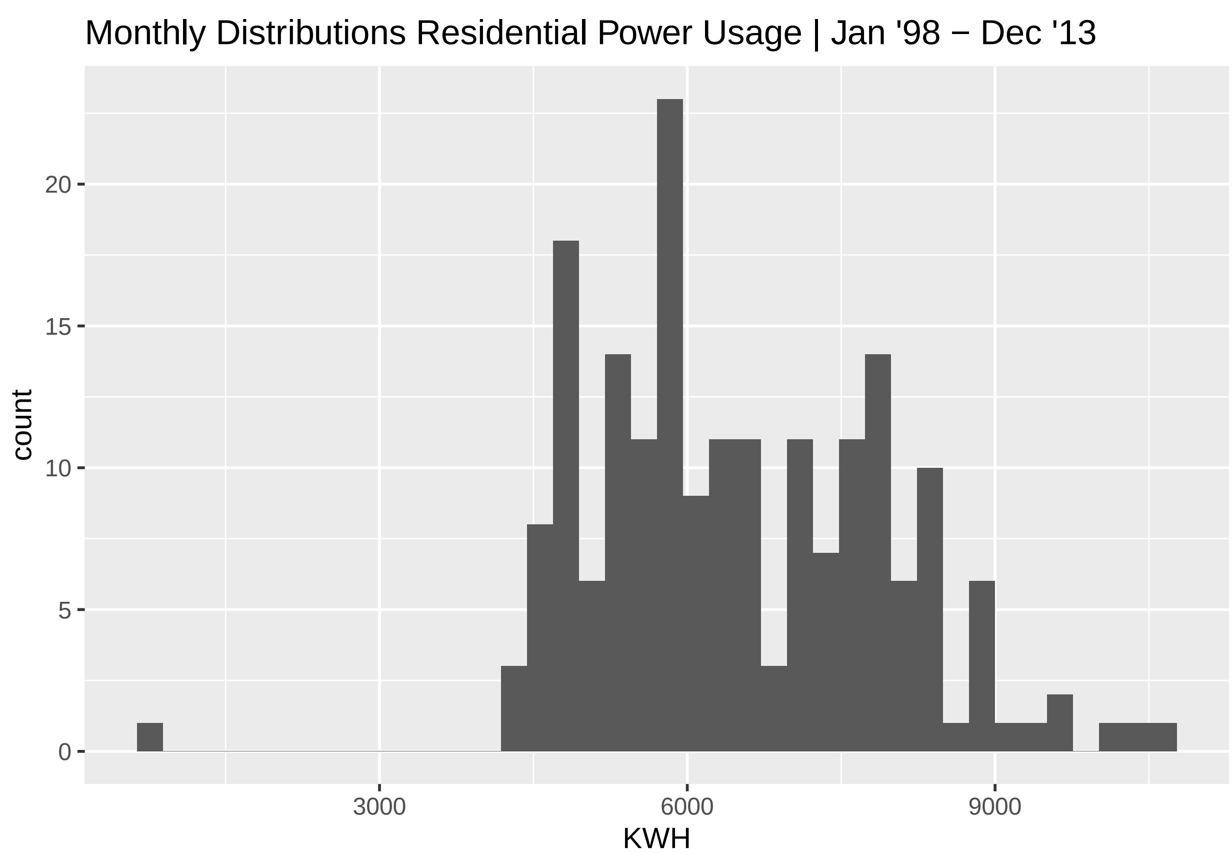
I renamed the `YYYY-MMM` for preference to `DATE`. The cleanup is a change of type for `DATE`, removal of `CaseSequence` as it does not help our model, and reducing our model to values in the thousands for ease of analysis. Like before we'll also be indexing by DATE

```r
#change variable type
power_df <- power_raw %>%
  mutate(DATE = yearmonth(`YYYY-MMM`), KWH = KWH/1000) %>%
  select(-CaseSequence, -'YYYY-MMM') %>%
  tsibble(index= DATE)
```

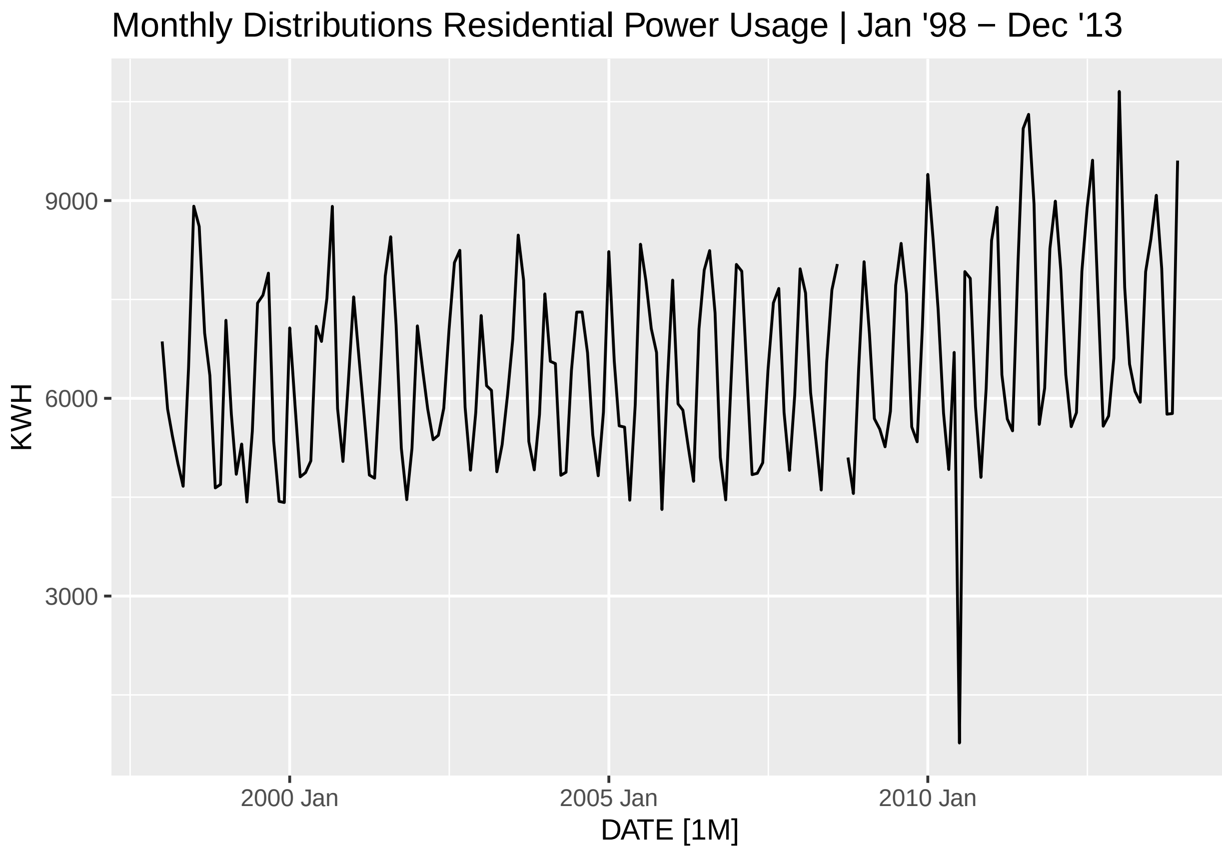
```r
head(power_df)
```

```
## # A tsibble: 6 x 2 [1M]
##      KWH     DATE
##    <dbl>    <mth>
## 1 6863. 1998 Jan
## 2 5838. 1998 Feb
## 3 5421. 1998 Mar
## 4 5010. 1998 Apr
## 5 4665. 1998 May
## 6 6467. 1998 Jun
```

```r
ggplot(power_df, aes(x=KWH))+
  geom_histogram(bins=40)+
  labs(title = "Monthly Distributions Residential Power Usage | Jan '98 - Dec '13")
```



```r
power_df %>%
  autoplot(KWH) +
  labs(title = "Monthly Distributions Residential Power Usage | Jan '98 - Dec '13")
```

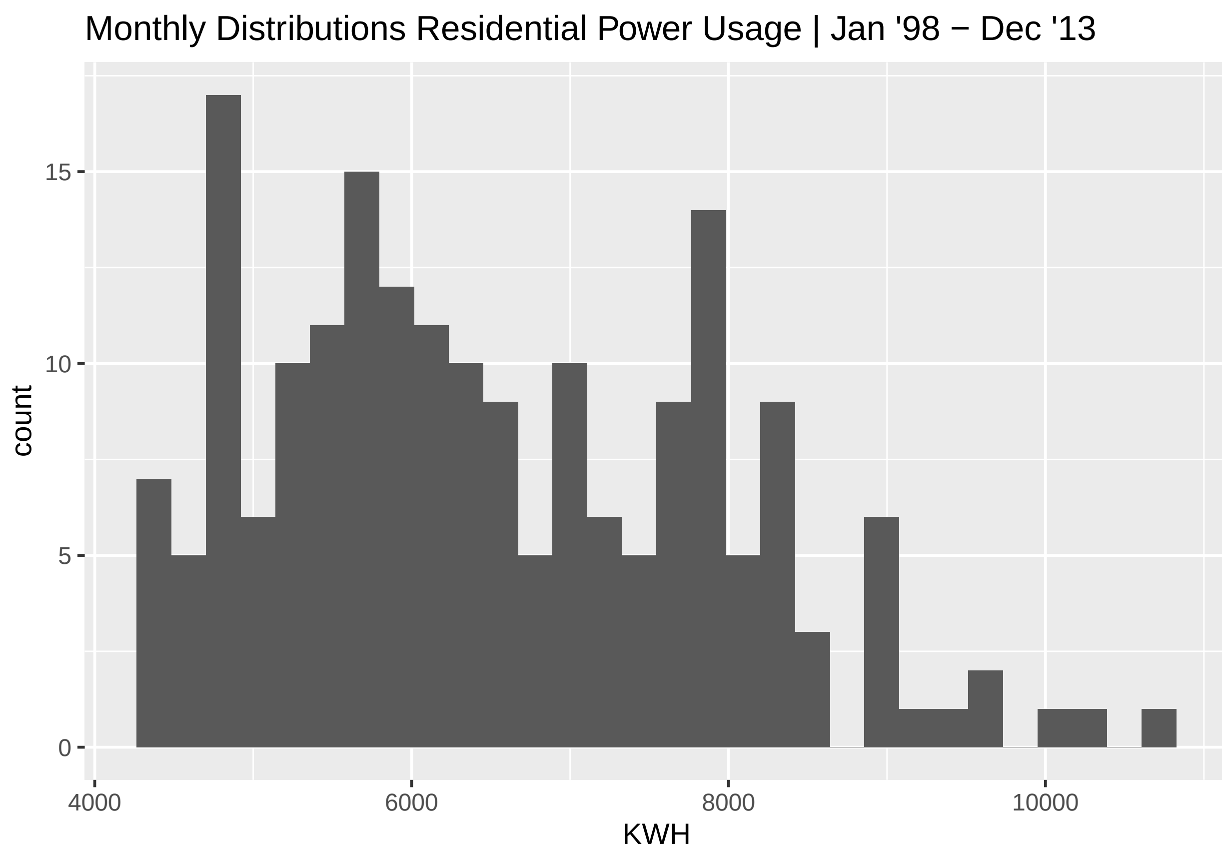


The data has an apparent outlier and is right skewed that appears in both plots, and resides sometime after January of 2010.

```r
# made a copy of the data first
power_df2<-power_df
power_df2$KWH <- na.interp(power_df2$KWH)
power_df2$KWH <- replace(power_df2$KWH, power_df2$KWH == min(power_df2$KWH),
                         median(power_df2$KWH))
```

Considering the distribution, I again thought it best to replace the missing value with the median, but considering I will be using that method to address the outlier, I decided to use `na.interp` since its a tool used by the author of our textbook Rob J Hydman's github repo. Regardless, the transformation below shows its still right skewed but shows seasonality with an upward trend.

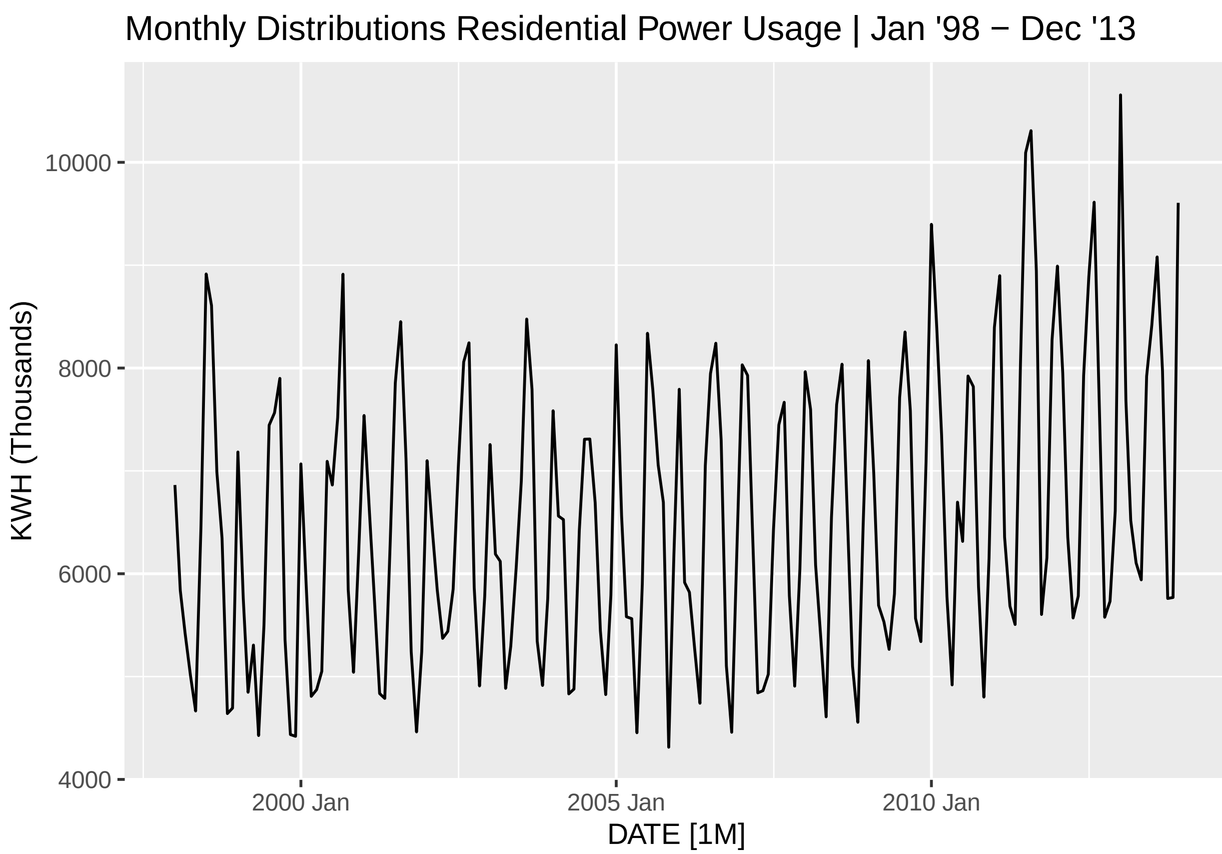
```r
ggplot(power_df2, aes(x=KWH))+
  geom_histogram()+
  labs(title = "Monthly Distributions Residential Power Usage | Jan '98 - Dec '13")
```




```r
#summary
summary(power_df2$KWH)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4313    5444    6330    6532    7609   10656
```

```r
#ts plot
power_df2 %>%
  autoplot(KWH) +
  labs(title = "Monthly Distributions Residential Power Usage | Jan '98 - Dec '13")+
  ylab(label= "KWH (Thousands)")
```



Before forecasting I will transform the data using a Box-Cox transformation.

```
#get lambda
(power_lambda <- power_df2 %>%
  features(KWH, features = guerrero) %>%
  pull(lambda_guerrero))
```

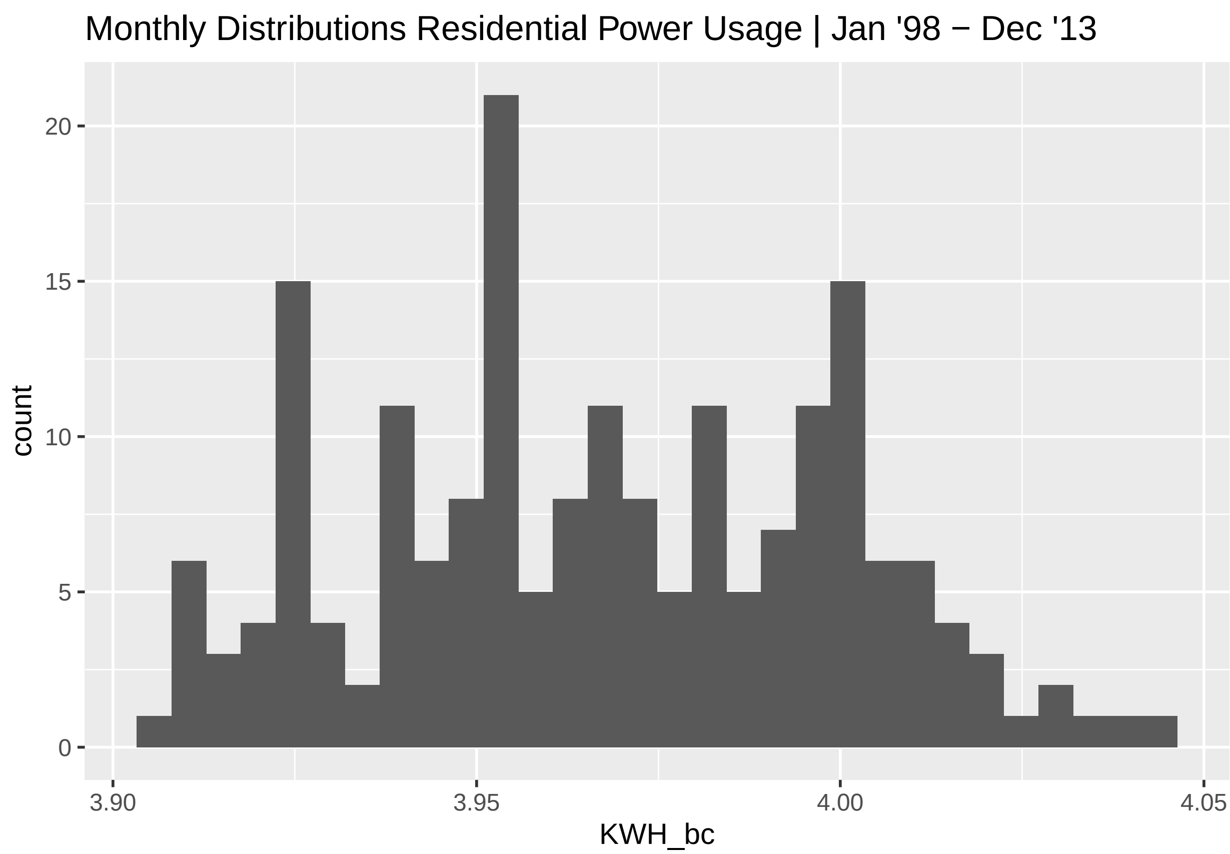
```
## [1] -0.2130548
```

```
power_df2 <- power_df2 %>%
    mutate(KWH_bc = box_cox(KWH, power_lambda))


summary(power_df2$KWH_bc)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.905   3.943   3.967   3.967   3.994   4.043
```

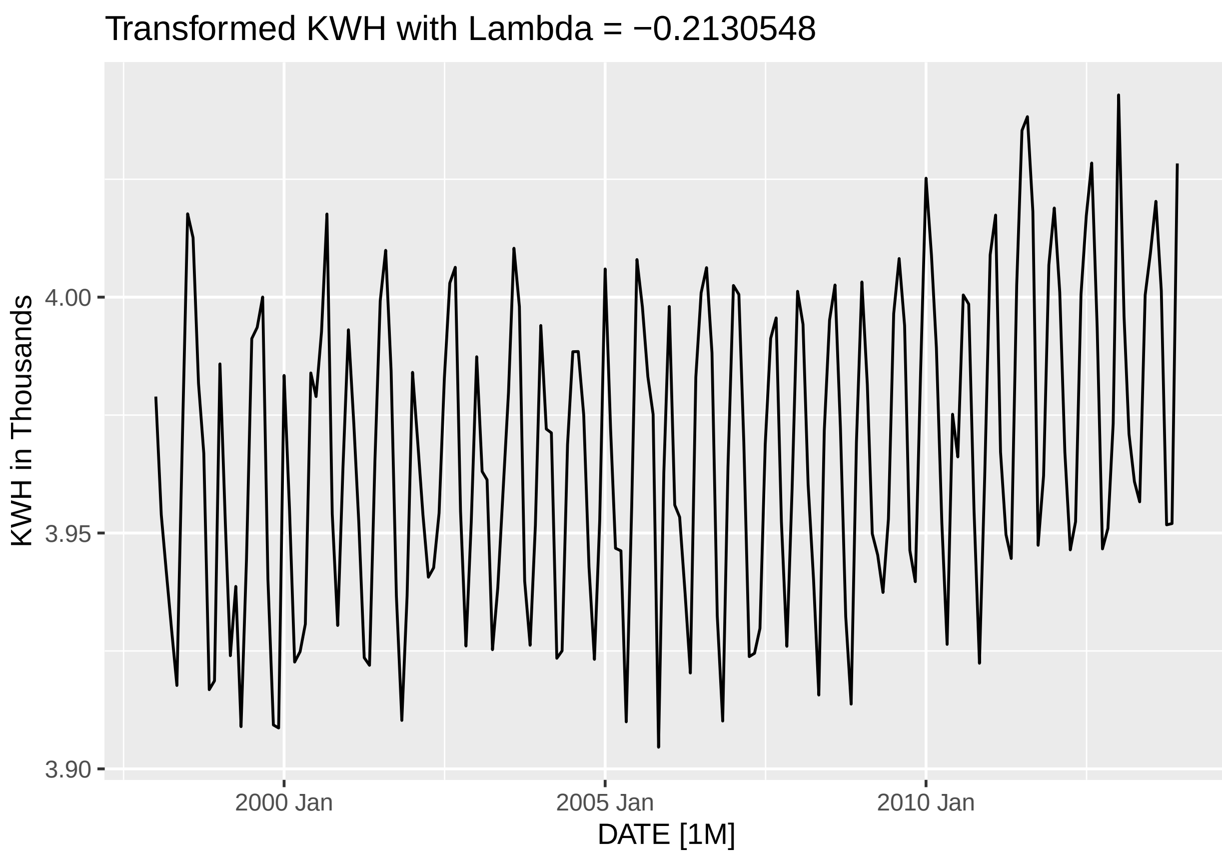
```
ggplot(power_df2, aes(x=KWH_bc))+
  geom_histogram()+
  labs(title = "Monthly Distributions Residential Power Usage | Jan '98 - Dec '13")
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
power_df2 %>%
  autoplot(KWH_bc) +
  labs(y = "KWH in Thousands",
       title = "Transformed KWH with Lambda = -0.2130548")
```

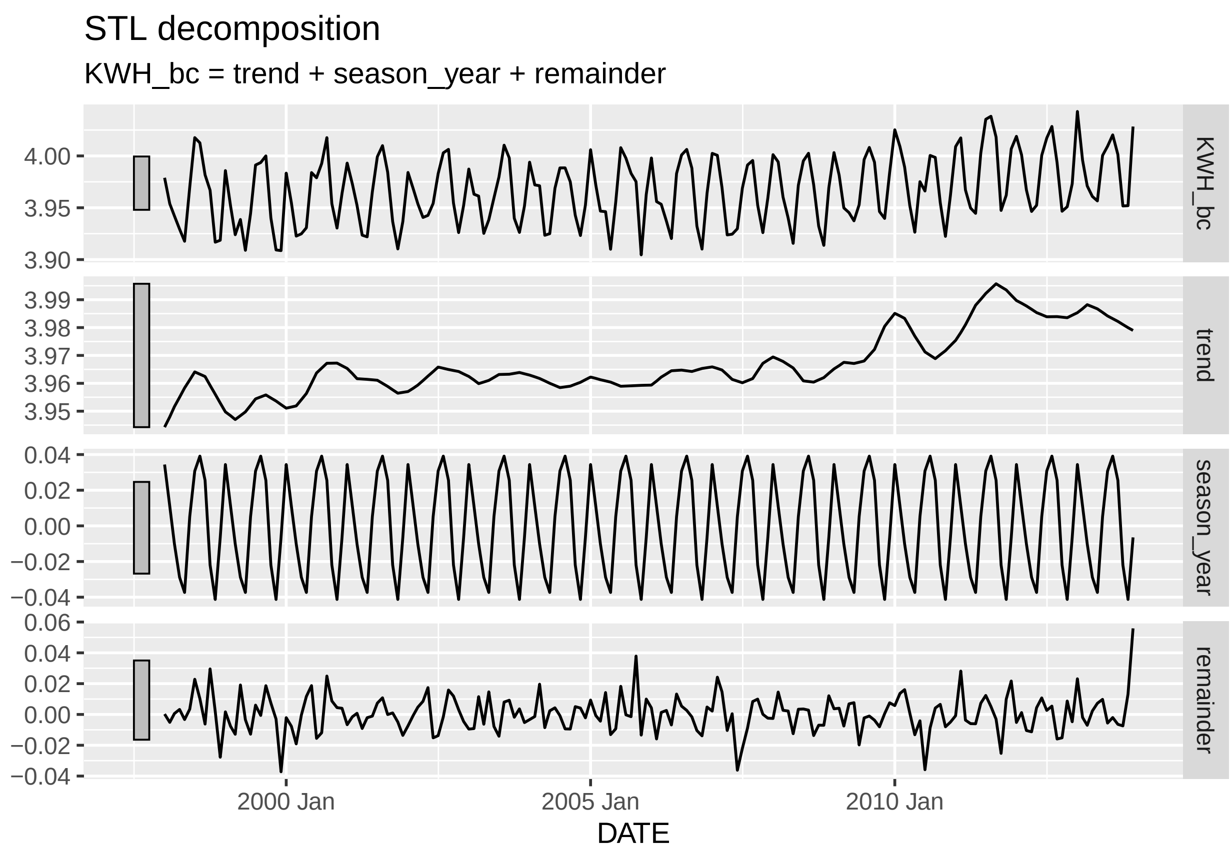
```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```



STL Decomposition

- STL decomposition again used to identify seasonality, variance, etc.
- ndiff() and ACF will identify if differencing is needed.

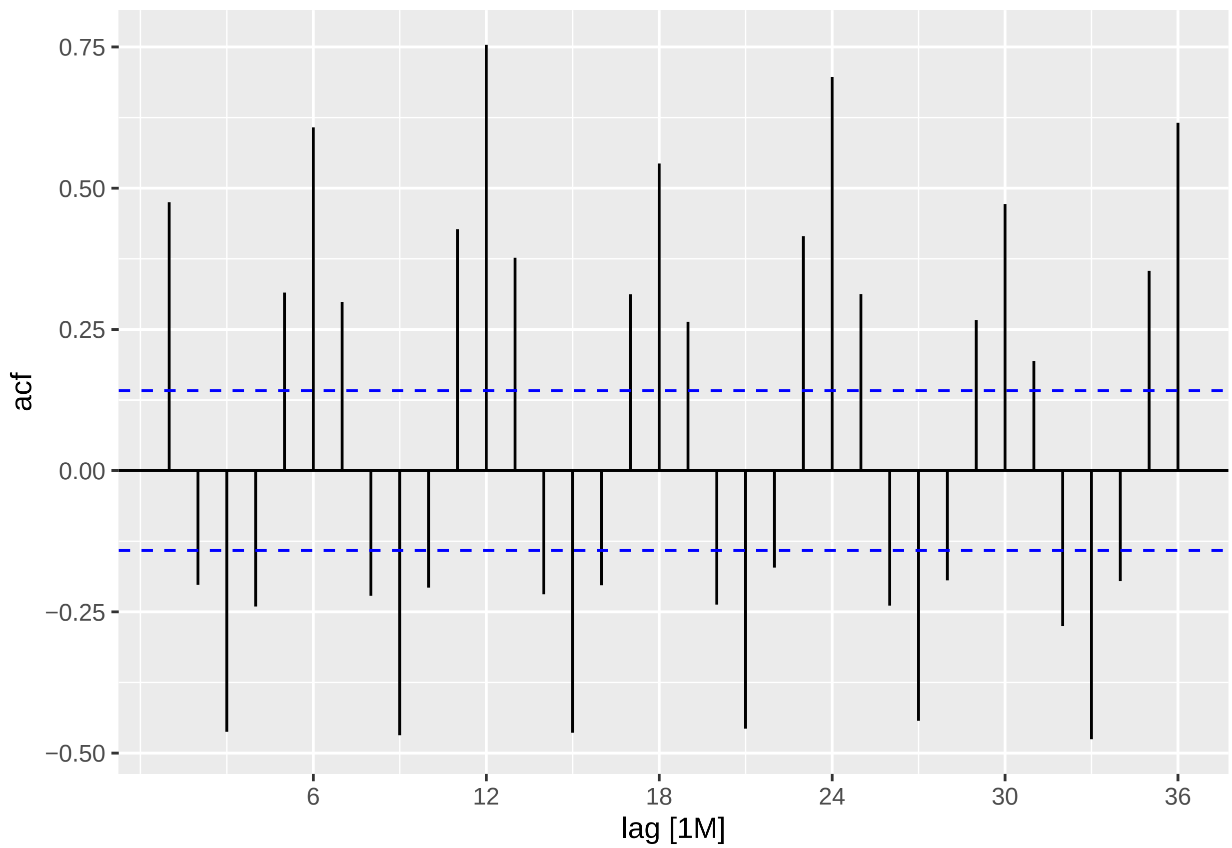
```
power_df2 %>%
  model(
    STL(KWH_bc ~ trend(window = 13) +
                   season(window = "periodic"),
                      robust = TRUE)) %>%
                        components() %>%
                            autoplot()
```




```r
ndiffs(power_df2$KWH_bc)
```

```
## [1] 1
```

```r
power_df2 %>%
  ACF(KWH_bc, lag_max = 36) %>%
  autoplot()
```



Differencing is needed.

```
diff_power <- power_df2 %>%
  mutate(diff_KWH= difference(KWH), diff_KWH_bc = difference(KWH_bc))
```

- Differencing created
- NA and some columns need removal

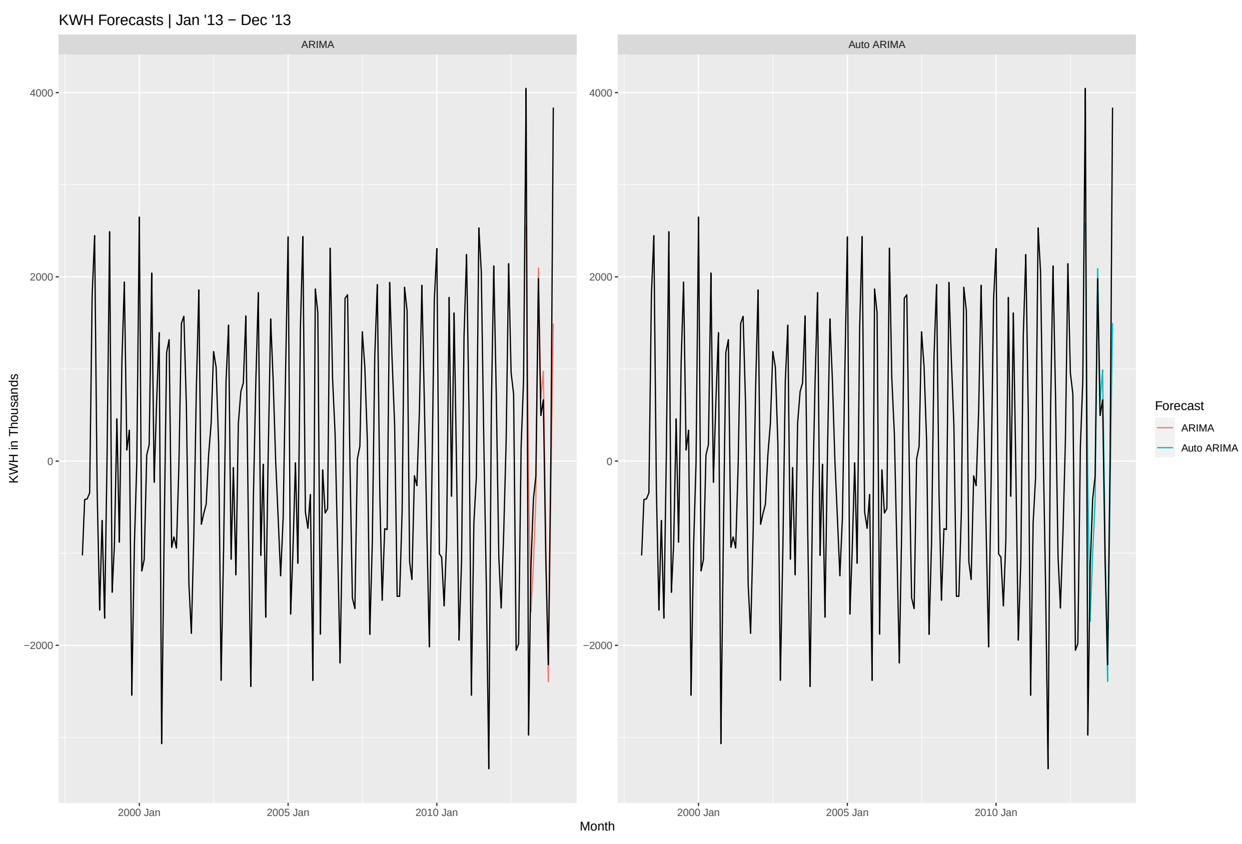
```
diff_power<-diff_power%>%
  select(-KWH, -KWH_bc)%>%
                              slice(-1)

ndiffs(diff_power$diff_KWH_bc)
```

```
## [1] 0
```

Forecast

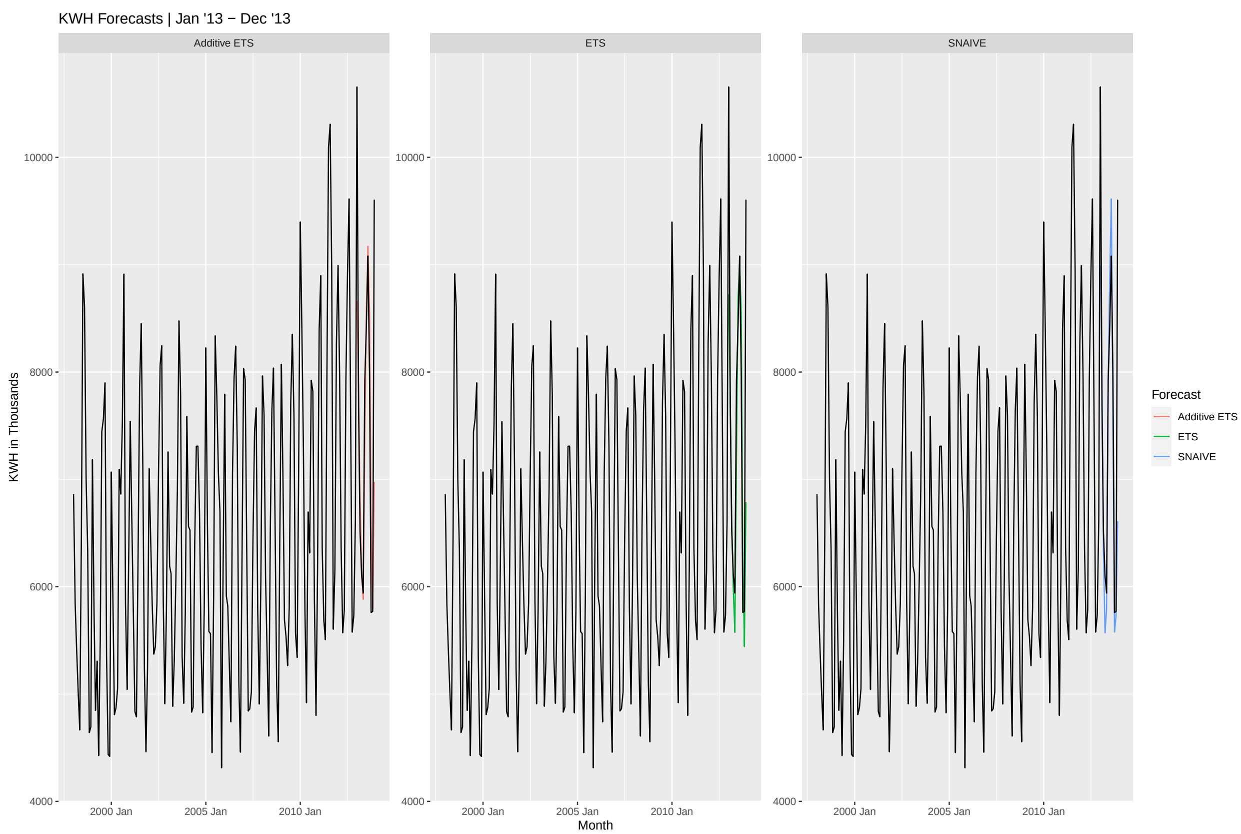
```
#Differenced data for arima

#split
power_train_diff <- diff_power %>%
  filter(year(DATE) < 2013)
```

```
#models
power_fit_diff <- power_train_diff %>%
    model(
    ARIMA = ARIMA(diff_KWH),
    `Auto ARIMA` = ARIMA(diff_KWH, stepwise = FALSE, approx = FALSE)
  )

#forecast of 2013
power_forecast_diff <- power_fit_diff %>%
  forecast(h = "1 year")

#plot
power_forecast_diff %>%
  autoplot(diff_power, level = NULL)+
  facet_wrap( ~ .model, scales = "free_y") +
  guides(colour = guide_legend(title = "Forecast"))+
  labs(title= "KWH Forecasts | Jan '13 - Dec '13")+
  xlab("Month") +
  ylab("KWH in Thousands")
```




```
#split
power_train <- power_df2 %>%
  filter(year(DATE) < 2013)

#models
power_fit <- power_train %>%
```

```
    model(
    ETS = ETS(KWH),
    `Additive ETS` = ETS(KWH ~ error("A") + trend("A") + season("A")),
    SNAIVE = SNAIVE(KWH)
  )

#forecast of 2013
power_forecast <- power_fit %>%
  forecast(h = "1 year")

#plot
power_forecast %>%
  autoplot(power_df2, level = NULL)+
  facet_wrap( ~ .model, scales = "free_y") +
  guides(colour = guide_legend(title = "Forecast"))+
  labs(title= "KWH Forecasts | Jan '13 - Dec '13")+
  xlab("Month") +
  ylab("KWH in Thousands")
```




```
#find ARIMA RMSE, MAE
accuracy(power_forecast_diff, diff_power) %>%
  select(.model, RMSE:MAE)
```

```
## # A tibble: 2 x 3
##   .model       RMSE   MAE
##   <chr>       <dbl> <dbl>
```

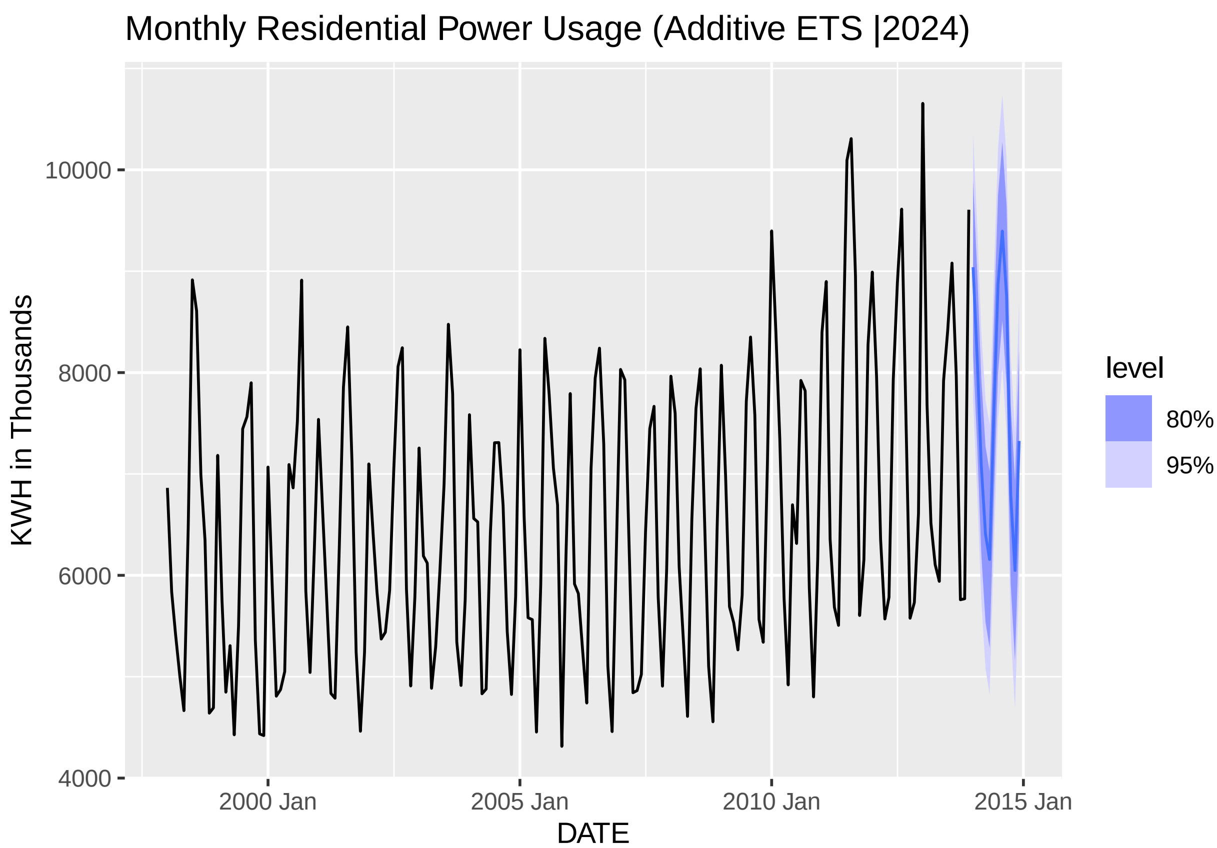
```
## 1 ARIMA      1168.  774.
## 2 Auto ARIMA 1154.  771.
```

```
#find other RMSE, MAE
accuracy(power_forecast, power_df2) %>%
  select(.model, RMSE:MAE)
```

```
## # A tibble: 3 x 3
##   .model        RMSE   MAE
##   <chr>        <dbl> <dbl>
## 1 Additive ETS 1020.  626.
## 2 ETS          1050.  664.
## 3 SNAIVE       1036.  619.
```

Additive ETS is the best model based on RMSE and MAE

```
#reproduce the mode using the original dataset
power_ETS_fit <- power_df2 %>%
  model(`Additive ETS` = ETS(KWH ~ error("A") + trend("A") + season("A")))

#generate the values
power_ETS_forecast <- power_ETS_fit %>%
  forecast(h=12)

#plot
power_ETS_forecast %>%
  autoplot(power_df2) +
  labs(title = "Monthly Residential Power Usage (Additive ETS |2024)",
       y = "KWH in Thousands")
```




```
(power_forecast_results <-
  as.data.frame(power_ETS_forecast) %>%
    select(DATE, .mean) %>%
      rename('KWH Forecast' = .mean))
```

```
##        DATE KWH Forecast
## 1  2014 Jan     9039.733
## 2  2014 Feb     8098.821
## 3  2014 Mar     7089.265
## 4  2014 Apr     6405.892
## 5  2014 May     6155.882
## 6  2014 Jun     7622.841
## 7  2014 Jul     8871.033
## 8  2014 Aug     9395.556
## 9  2014 Sep     8757.208
## 10 2014 Oct     6798.997
## 11 2014 Nov     6048.011
## 12 2014 Dec     7325.489
```