Time Series

2024-12-21

Contents

Predict VS Forecast

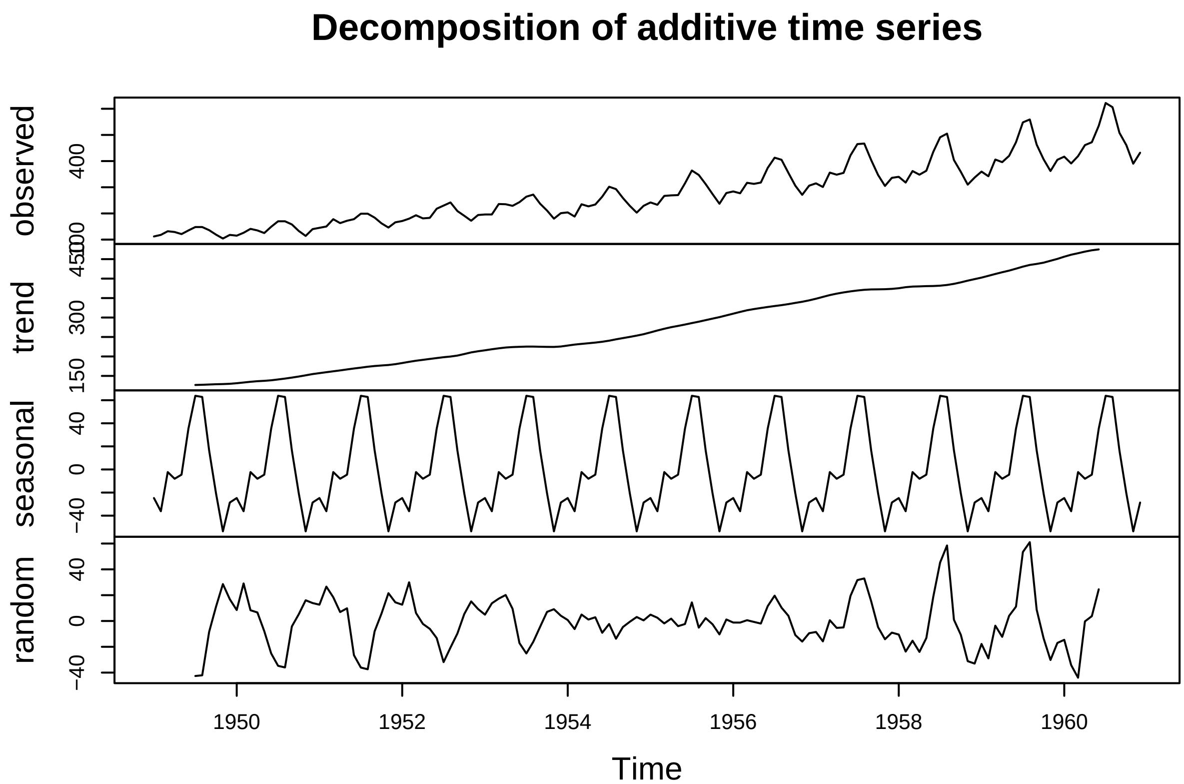
- **Predict** : predict response based on other variables (linear regression)
- **Forecast** : forecast future values based on previous values (time series)

Time series data can't be applied into linear regression model due to the seasonal fluctuations. Thus, the ability for make correct prediction will be poor.

```
library(forecast)
```

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

```
data("AirPassengers")
plot(decompose(AirPassengers))
```



Interpretation :

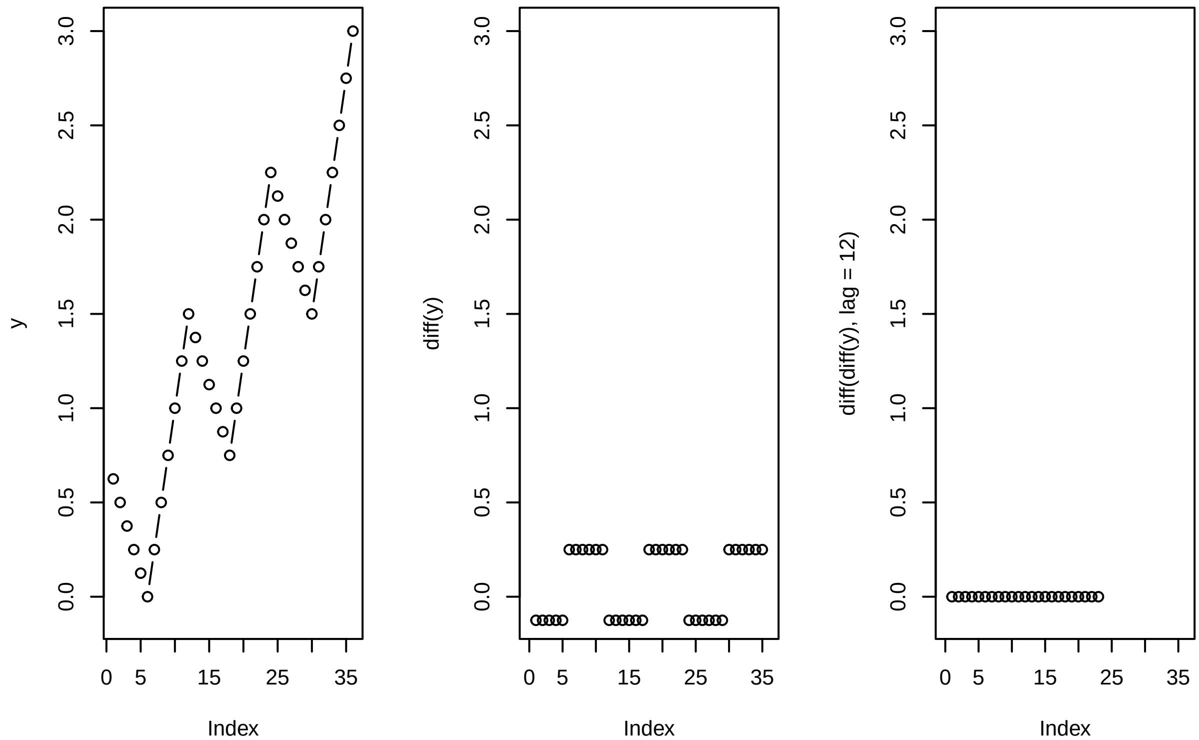
- **Observed** : original data
- **Trend** : data after removing seasonal component
- **Seasonal** : data after removing trend component
- **Random** : data after removing both trend and seasonal component

Model Assumption

Stationary is when the data has a **constant mean and variance.** If there is trend or seasonal component on the original data, that means you data is not stationary and require transformation by using differencing technique.

We create a sample data to show how to use `diff()`

```
seq_down <- seq(.625, .125, -0.125)
seq_up <- seq(0, 1.5, 0.25)
y <- c(seq_down, seq_up, seq_down + .75, seq_up + .75, seq_down + 1.5,
       seq_up + 1.5)

par(mfrow = c(1, 3))
plot(y, type = "b", ylim = c(-.1, 3))
plot(diff(y), ylim = c(-.1, 3), xlim = c(0, 36))
plot(diff(diff(y), lag = 12), ylim = c(-.1, 3), xlim = c(0, 36))
```



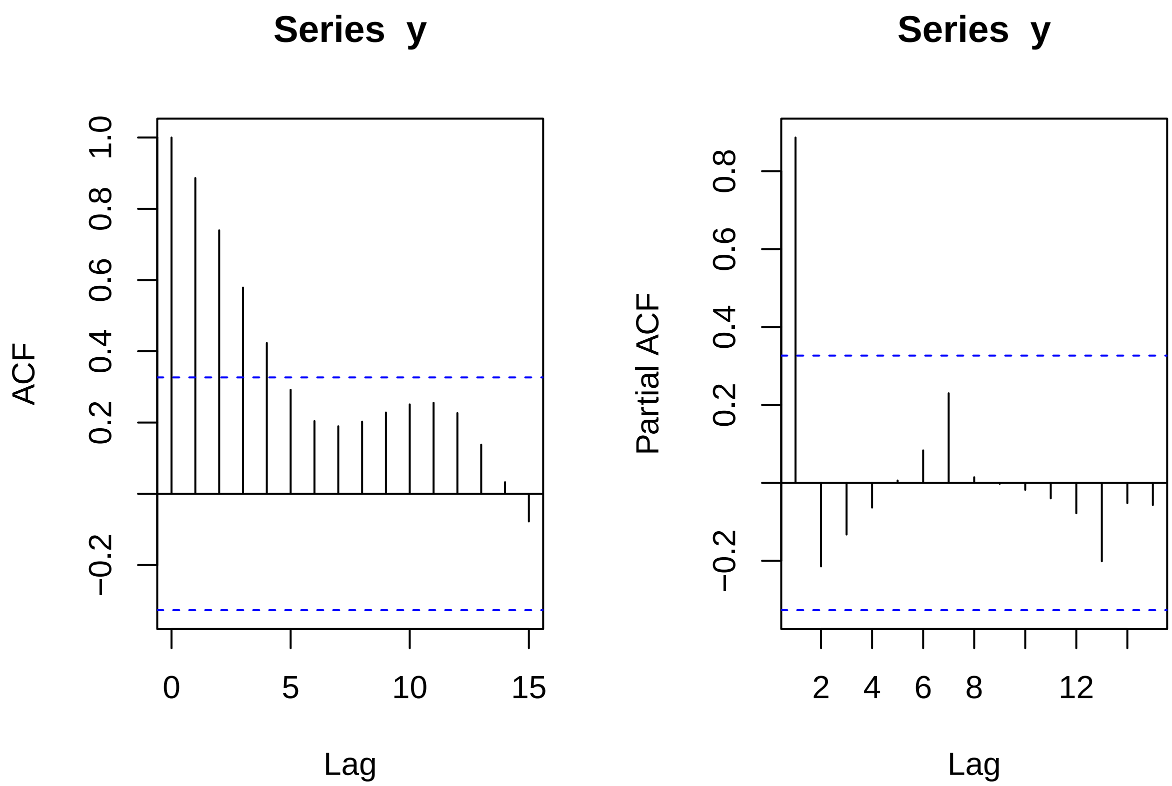

```
par(mfrow = c(1, 1))
```

Model Building

3 elements in ARIMA:

- **AR** : Auto regressive (p) | ACF slowly diminish or cycle and its PACF cut off under a significant line after a certain number of lags
- **I** : Integrated, differencing (d)
- **MA** : Moving average (q) | PACF slowly diminish or cycle. ACF cut off under a significance line after a certain number of lags

```
par(mfrow=c(1,2))
acf(y)
pacf(y)
```




```r
library(lubridate) # year, month
```

```
##
## Attaching package: 'lubridate'

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

```r
library(dplyr) # %>%
```

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

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

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

```r
library(forecast) # auto.arima
```

Case Study

ETL

```r
cycle = read.csv('./Data/Ch6_ridership_data_2011-2012.csv')
str(data)
```

```
## function (..., list = character(), package = NULL, lib.loc = NULL, verbose = getOption("verbose"),
##     envir = .GlobalEnv, overwrite = TRUE)
```

We can see that the data is in hourly data frame and we want to convert it into monthly data frame.

```r
library(dplyr)
library(lubridate)

cycle$datetime = as.Date(cycle$datetime, format = '%Y-%m-%d')

str(cycle)
```

```
## 'data.frame':    17379 obs. of  2 variables:
##  $ datetime: Date, format: "2011-01-01" "2011-01-01" ...
##  $ count   : int  16 40 32 13 1 1 2 3 8 14 ...
```

```r
monthly_ride = cycle %>%
  group_by(year = year(datetime), month = month(datetime)) %>%
  summarise(riders = sum(count))
```

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```r
table(monthly_ride$year, monthly_ride$month)
```

```
##
##         1  2  3  4  5  6  7  8  9 10 11 12
##   2011  1  1  1  1  1  1  1  1  1  1  1  1
##   2012  1  1  1  1  1  1  1  1  1  1  1  1
```

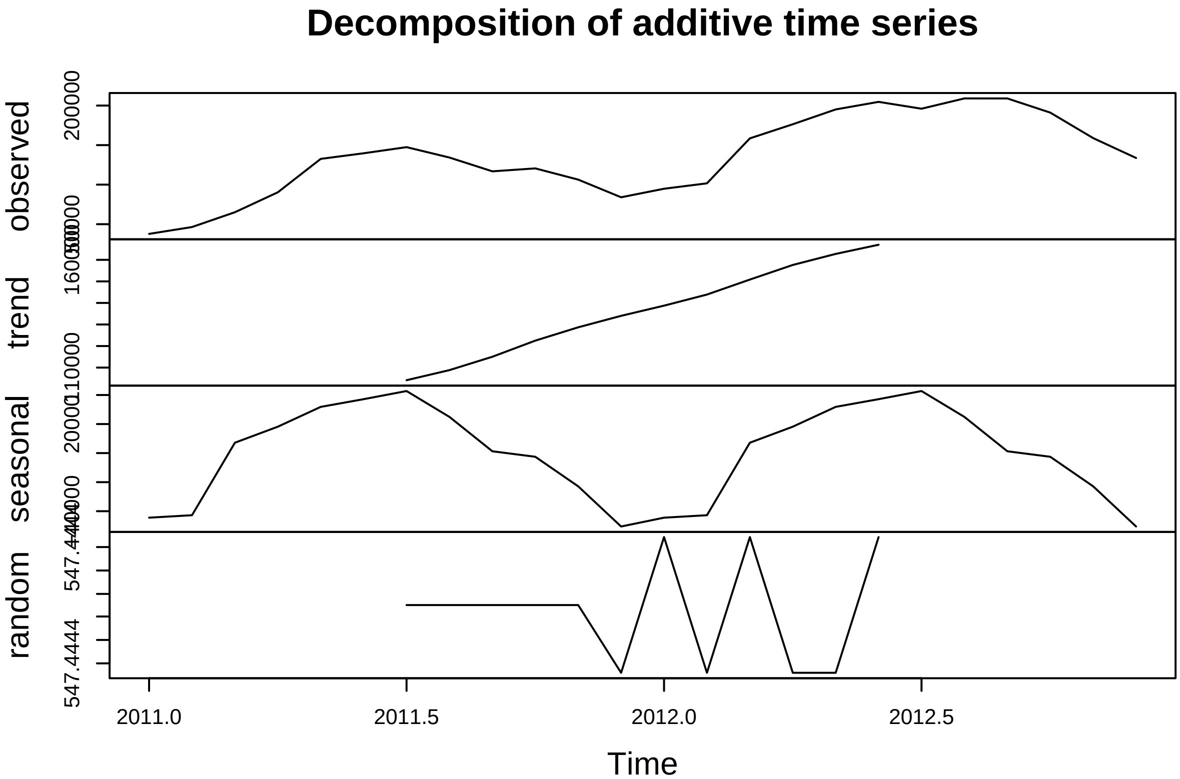
```r
riders = monthly_ride[,3]
monthly = ts(riders, frequency = 12, start = c(2011,1))
class(monthly)
```

```
## [1] "ts"
```

```r
monthly
```

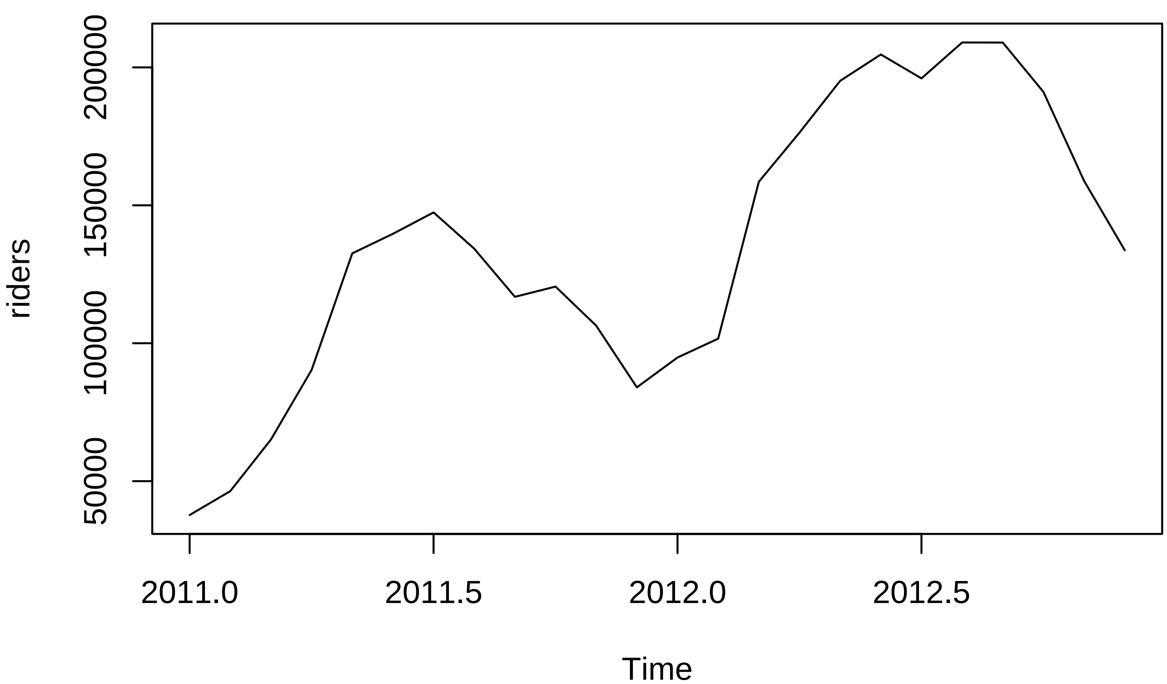
```
##         Jan    Feb    Mar    Apr    May    Jun    Jul    Aug    Sep    Oct
## 2011  37727  46396  65109  90332 132580 139674 147426 134280 116825 120535
## 2012  94832 101668 158535 176349 195114 204683 196014 209024 208995 191108
##         Nov    Dec
## 2011 106361  84025
## 2012 158855 133735
```

```r
plot(decompose(monthly))
```



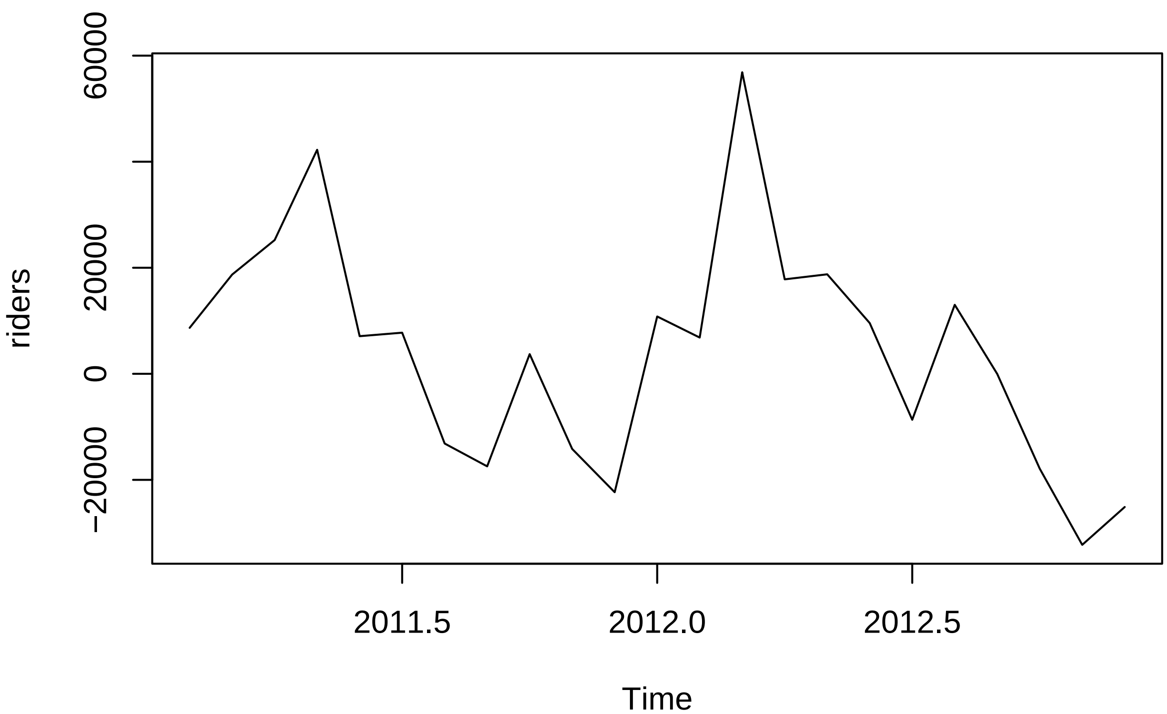
Analyze time series manually

```
plot(monthly)
```

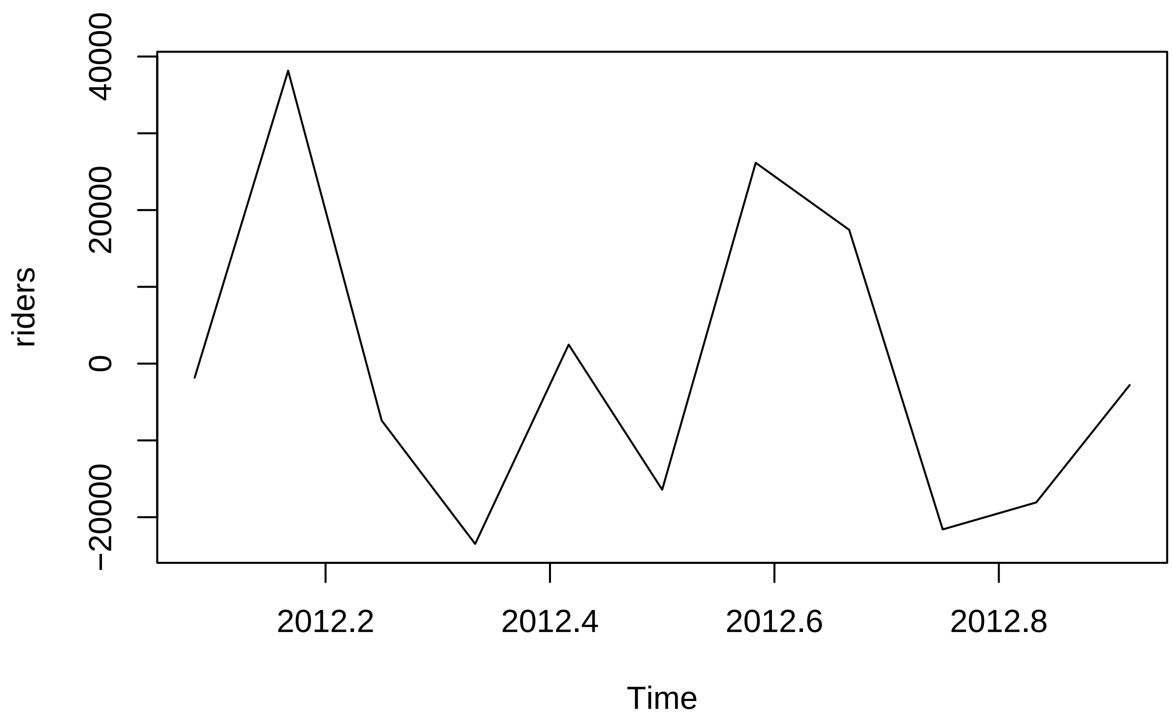


By looking at the plot, we can see that there's a trend, so we need to differentiate 1 time, and there's also seasonality. So, we need to differentiate the second time but this time with a lag.

```
y=diff(monthly)
plot(y)
```

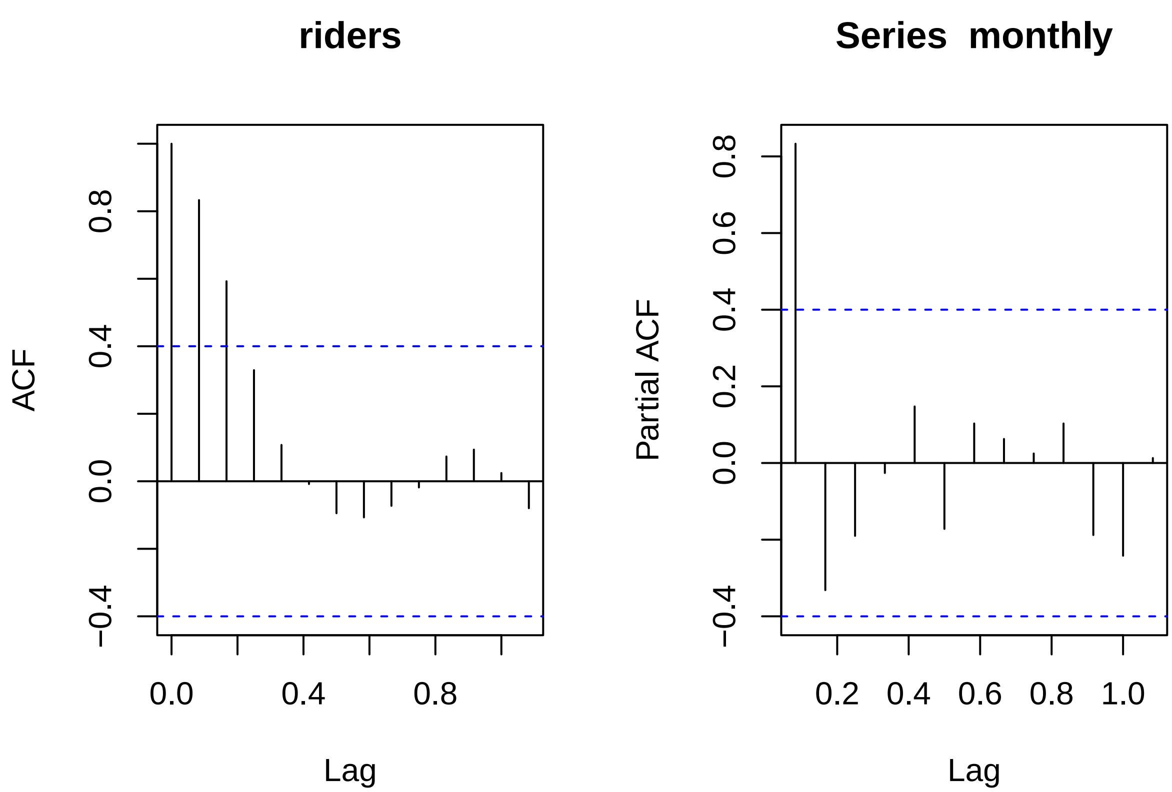



```
z=diff(diff(monthly), lag =12)
plot(z)
```



Now, we can see that the data has become stationary (constant mean and variance). However, the second differencing does not appear to make the data more stationary. Thus the first differencing is already sufficient for this case. Next, we'll see the ACF and PACF plot.

```
par(mfrow=c(1,2))
acf(monthly)
pacf(monthly)
```



Interpretation :

- **ACF** : Diminish slowly with a cycle. This suggests that the data is AR
- **PACF** : Cut off after the first lag. This suggest that the data is AR(1) with no seasonal component.

Next step is to compare the results from a number of models. We can use **Akaike Information Criterion (AIC)** the lower the better

from the plot, we can infer that pacf has AR(1) and MA(1) & MA(2) are potential.

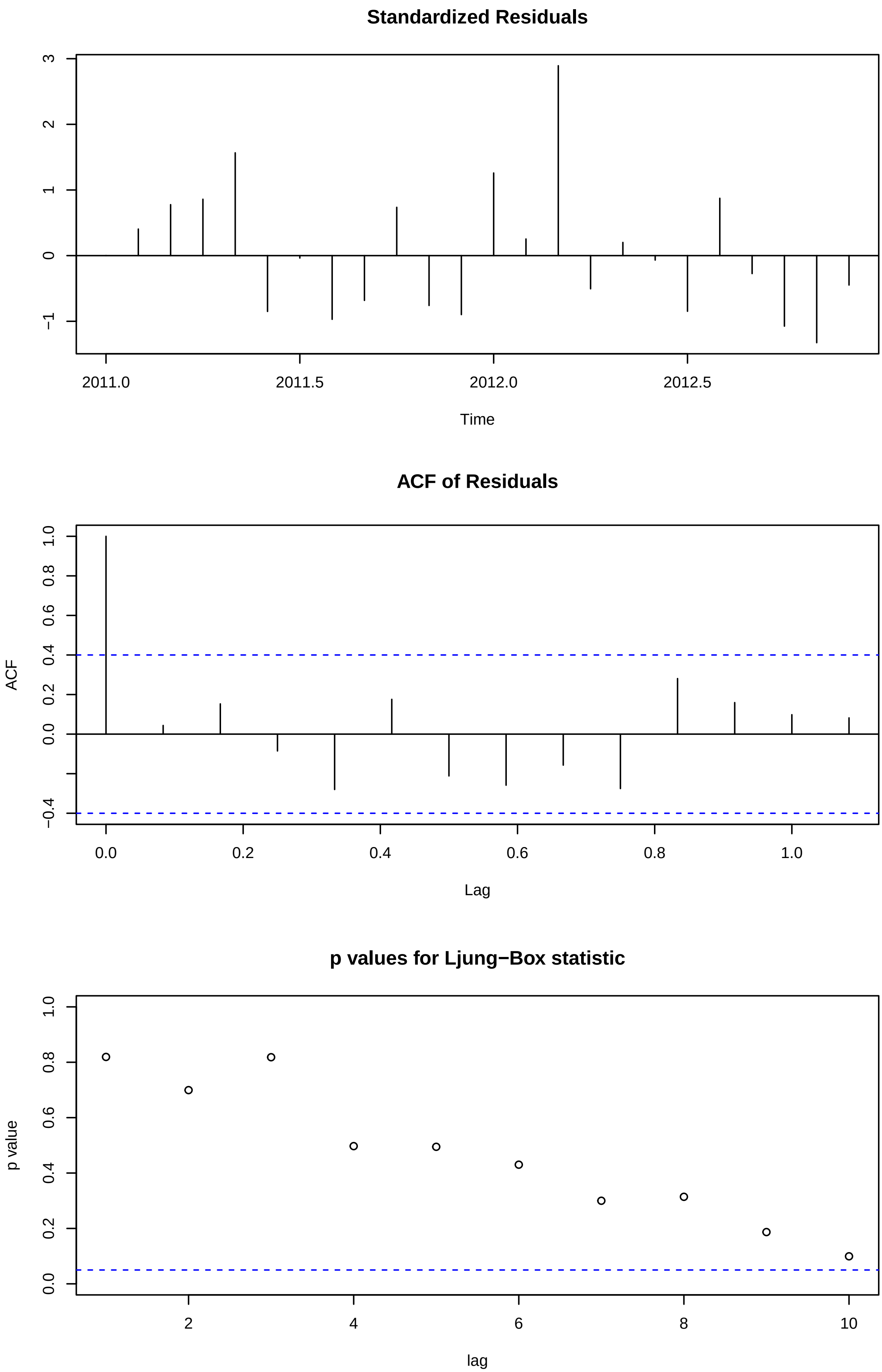
```
model1 = arima(monthly, c(1,0,0), seasonal=list(order=c(0,0,0)))
model2 = arima(monthly, c(1,1,0), seasonal=list(order=c(0,0,0)))
model3 = arima(monthly, c(2,1,0), seasonal=list(order=c(0,0,0)))
model4 = arima(monthly, c(1,1,0), seasonal=list(order=c(0,1,0)))
model5 = arima(monthly, c(0,1,1), seasonal=list(order=c(0,0,0)))

aic = c(model1$aic, model2$aic, model3$aic, model4$aic, model5$aic)
model = seq(1:5)
df = data.frame(model, aic)
df
```

```
##   model      aic
## 1     1 553.7413
## 2     2 520.9564
## 3     3 522.6320
## 4     4 252.3762
## 5     5 523.2984
```

```
str(model1)
```

```
## List of 14
##  $ coef     : Named num [1:2] 9.25e-01 1.10e+05
##   ..- attr(*, "names")= chr [1:2] "ar1" "intercept"
##  $ sigma2   : num 4.41e+08
##  $ var.coef : num [1:2, 1:2] 4.45e-03 -7.93e+02 -7.93e+02 1.76e+09
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:2] "ar1" "intercept"
##   .. ..$ : chr [1:2] "ar1" "intercept"
##  $ mask     : logi [1:2] TRUE TRUE
##  $ loglik   : num -274
##  $ aic      : num 554
##  $ arma     : int [1:7] 1 0 0 0 12 0 0
##  $ residuals: Time-Series [1:24] from 2011 to 2013: -27538 3240 13932 21843 40755 ...
##  $ call     : language arima(x = monthly, order = c(1, 0, 0), seasonal = list(order = c(0, 0,      0)))
##  $ series   : chr "monthly"
##  $ code     : int 0
##  $ n.cond   : int 0
##  $ nobs     : int 24
##  $ model    :List of 10
##   ..$ phi  : num 0.925
##   ..$ theta: num(0)
##   ..$ Delta: num(0)
##   ..$ Z    : num 1
##   ..$ a    : num 23458
##   ..$ P    : num [1, 1] 0
##   ..$ T    : num [1, 1] 0.925
##   ..$ V    : num [1, 1] 1
##   ..$ h    : num 0
##   ..$ Pn   : num [1, 1] 1
##  - attr(*, "class")= chr "Arima"
```

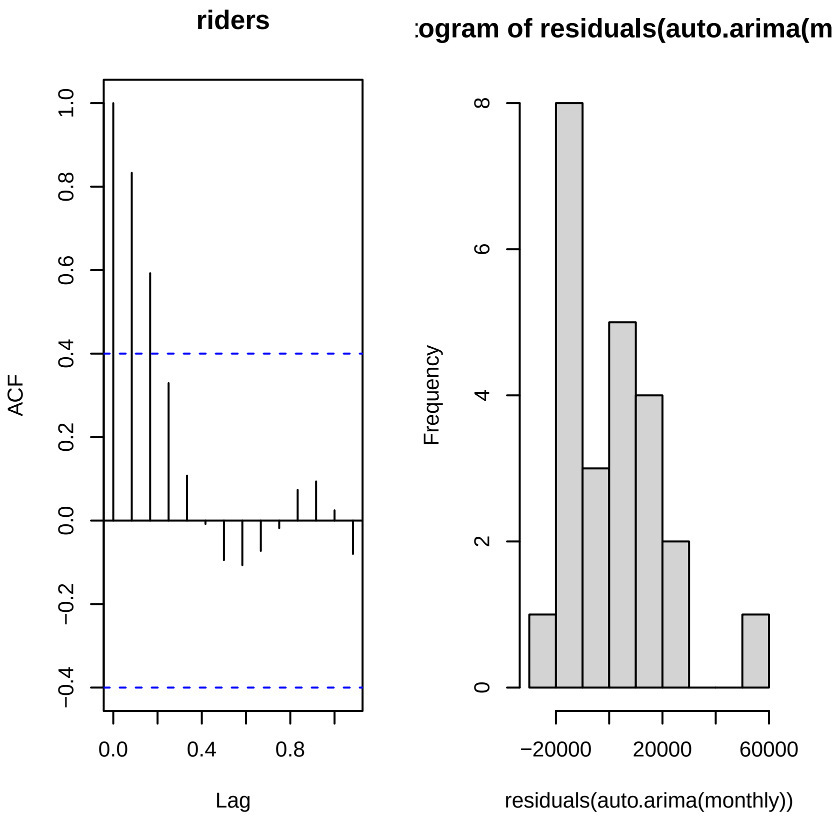
```r
tsdiag(model3)
```



```r
str(auto.arima(monthly))
```

```
## List of 18
##  $ coef     : Named num 0.517
##   ..- attr(*, "names")= chr "ar1"
##  $ sigma2   : num 3.49e+08
##  $ var.coef : num [1, 1] 0.0314
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr "ar1"
##   .. ..$ : chr "ar1"
##  $ mask     : logi TRUE
##  $ loglik   : num -258
##  $ aic      : num 521
##  $ arma     : int [1:7] 1 0 0 0 12 1 0
##  $ residuals: Time-Series [1:24] from 2011 to 2013: 37.7 7420.2 14230.6 15547.3 29206.2 ...
##  $ call     : language auto.arima(y = monthly, x = list(x = c(37727L, 46396L, 65109L, 90332L,  132580L, 139674L
##  $ series   : chr "monthly"
##  $ code     : int 0
##  $ n.cond   : int 0
##  $ nobs     : int 23
##  $ model    :List of 10
##   ..$ phi  : num 0.517
##   ..$ theta: num(0)
##   ..$ Delta: num 1
##   ..$ Z    : num [1:2] 1 1
##   ..$ a    : num [1:2] -25120 158855
##   ..$ P    : num [1:2, 1:2] 0.00 2.91e-21 2.91e-21 -2.91e-21
##   ..$ T    : num [1:2, 1:2] 0.517 1 0 1
##   ..$ V    : num [1:2, 1:2] 1 0 0 0
##   ..$ h    : num 0
##   ..$ Pn   : num [1:2, 1:2] 1.00 -1.50e-21 -1.50e-21 -2.91e-21
##  $ bic      : num 523
##  $ aicc     : num 522
##  $ x        : Time-Series [1:24, 1] from 2011 to 2013: 37727 46396 65109 90332 132580 139674 147426 134280 1168
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr "riders"
##  $ fitted   : Time-Series [1:24, 1] from 2011 to 2013: 37689 38976 50878 74785 103374 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr "x"
##  - attr(*, "class")= chr [1:3] "forecast_ARIMA" "ARIMA" "Arima"
```

```r
par(mfrow=c(1,3))
acf(monthly)
hist(residuals(auto.arima(monthly)))
```



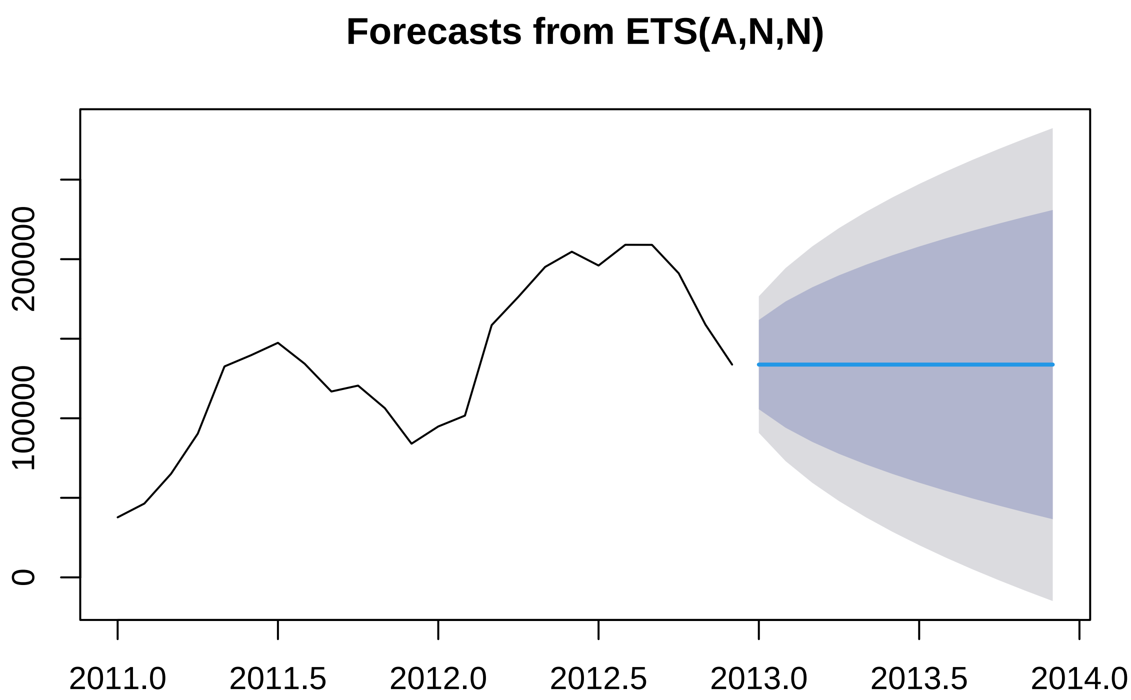
Automatically using auto.arima function

```r
auto.arima(monthly)
```

```
## Series: monthly
## ARIMA(1,1,0)
##
## Coefficients:
##          ar1
##       0.5171
## s.e.  0.1772
##
## sigma^2 = 348592099:  log likelihood = -258.48
## AIC=520.96   AICc=521.56   BIC=523.23
```

Forecasting

```r
yr_forecast = forecast(monthly, h=12)
plot(yr_forecast)
```



Interpretation :

- Blue line : mean forecast
- Dark inner cone : 80% confidence interval
- Light outer cone : 90% confidence interval

However, this is not a pretty forecast since it indicates that anything is possible within the next 12 months

TBAT

This technique is more suitable when :

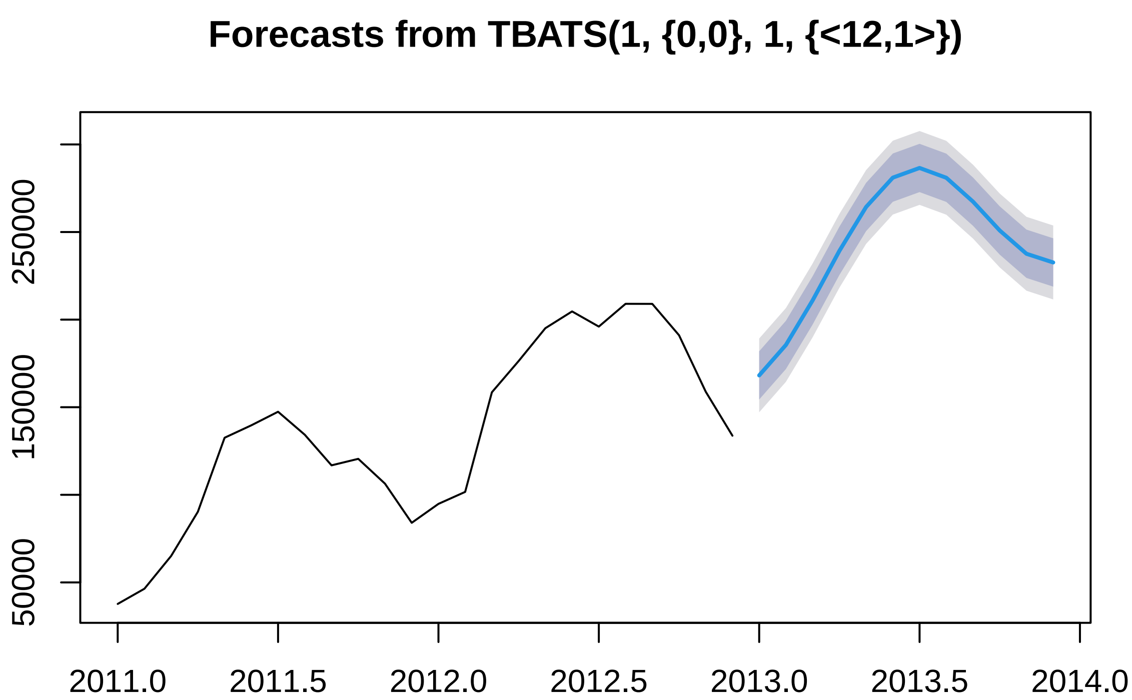
1. Small dataset
2. Frequency > 24 (hourly)

```
tbats(monthly)
```

```
## TBATS(1, {0,0}, 1, {<12,1>})
##
## Call: tbats(y = monthly)
##
## Parameters
##   Alpha: -0.02010182
##   Beta: 0.0001000385
##   Damping Parameter: 1
##   Gamma-1 Values: 3.69966e-05
##   Gamma-2 Values: 3.882865e-05
##
## Seed States:
##             [,1]
## [1,]  62200.582
```

```
## [2,]   5894.686
## [3,] -41521.948
## [4,]  11867.898
##
## Sigma: 10738.16
## AIC: 537.7881
```

```r
j=forecast(tbats(monthly), h=12)
plot(j)
```




```r
summary(j$mean)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  168164  227220  245052  242105  270736  286574
```

```r
summary(j$upper)
```

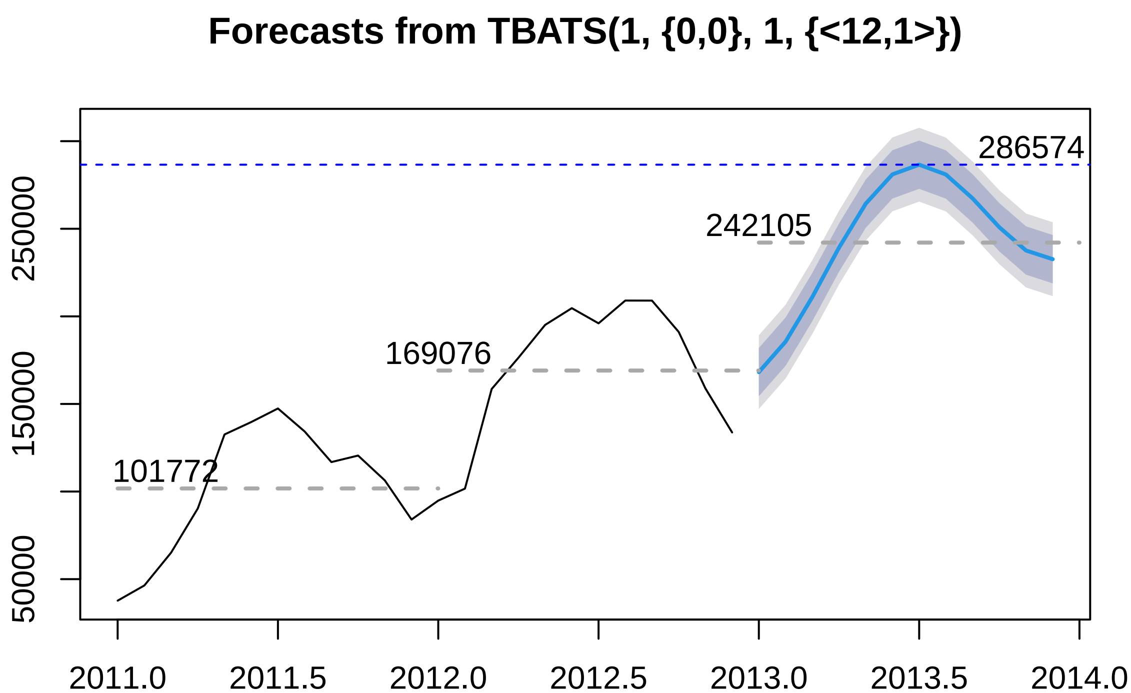
```
##       80%               95%
##  Min.   :181925   Min.   :189210
##  1st Qu.:241006   1st Qu.:248303
##  Median :258830   Median :266123
##  Mean   :255881   Mean   :263174
##  3rd Qu.:284519   3rd Qu.:291815
##  Max.   :300352   Max.   :307646
```

```r
summary(j$lower)
```

```
##       80%               95%
##  Min.   :154402   Min.   :147118
##  1st Qu.:213435   1st Qu.:206138
##  Median :231274   Median :223980
##  Mean   :228328   Mean   :221035
##  3rd Qu.:256953   3rd Qu.:249657
##  Max.   :272796   Max.   :265503
```

```r
mean_2011 = round(as.numeric(filter(monthly_ride, year == 2011) %>%
            summarise(mean = mean(riders))), 0)
mean_2012 = round(as.numeric(filter(monthly_ride, year == 2012) %>%
            summarise(mean = mean(riders))), 0)
mean_2013 = round(mean(j$mean),0)
max_mean_2013 = round(max(j$mean),0)
```

```r
plot(j)
abline(h=max(j$mean), lty=2, col='blue')
segments(2011, mean_2011, x1=2012, y1=mean_2011, col='darkgray', lty=2, lwd=2)
segments(2012, mean_2012, x1=2013, y1=mean_2012, col='darkgray', lty=2, lwd=2)
segments(2013, mean_2013, x1=2014, y1=mean_2013, col='darkgray', lty=2, lwd=2)

text(2011.15, mean_2011 + 10000, mean_2011)
text(2012, mean_2012 + 10000, mean_2012)
text(2013, mean_2013 + 10000, mean_2013)
text(2013.85, max_mean_2013 + 10000, max_mean_2013)
```



The middle parameter of {0,0} indicates that this technique is using AR(0) and MA(0)