Northeastern University
College of Professional Studies

Project 5

Time Series Analysis in R

Dickson Wanjau

ALY6015 81058 Intermediate Analytics SPRING 2019 CPS

Instructor: Dr. Roseanna Hopper

Due Date: May 14, 2019.

Introduction

Time series data are data points that are collected sequentially at a regular interval with association over a time period. A time series can be made up of these key components.

- Trend - a long-term increase or decrease in data over time
- Seasonality - an effect of seasonal factors for a fixed or known period e.g. month, quarter of the year etc.
- Cycle - These are the longer ups and down that are not of a fixed or known periods caused by external factors
- Periodicity - an exact repetition in regular pattern of the data
- Residual: This is the remaining signal after removing the seasonality and trend signals. It can be further decomposed to remove the noise component as well.

There are two main goals of time series analysis:

(a) identifying the nature of the phenomenon represented by the sequence of observations

(b) forecasting (predicting future values of the time series variable).

Both of these goals require that the pattern of observed time series data is identified and described. One of the best time series analysis methods is called ARIMA or Box Jenkins. ARIMA stands for Autoregressive Integrated Moving Averages. If a time series is stationary, then it can fit the ARIMA model in a variety of ways. A time series is stationary when mean, variance, autocorrelation are constant over time.

However, if the time series is not stationary, we cannot build a time series model. In the where stationarity is not met, the first step is to make the time series stationary and then try stochastic models to predict this time series. There are several ways to achieve stationarity such as detrending, differencing by etc. Smoothing techniques such as Exponential smoothing and moving averages are widely used

At the basic, a time series can be expressed as either a sum or a product of 3 components, namely, *Seasonality (St)*, *Trend (Tt)* and *Error (et)* (White Noise/ random noise). For each data point Y_t at time t in a time series;

For an additive time series;

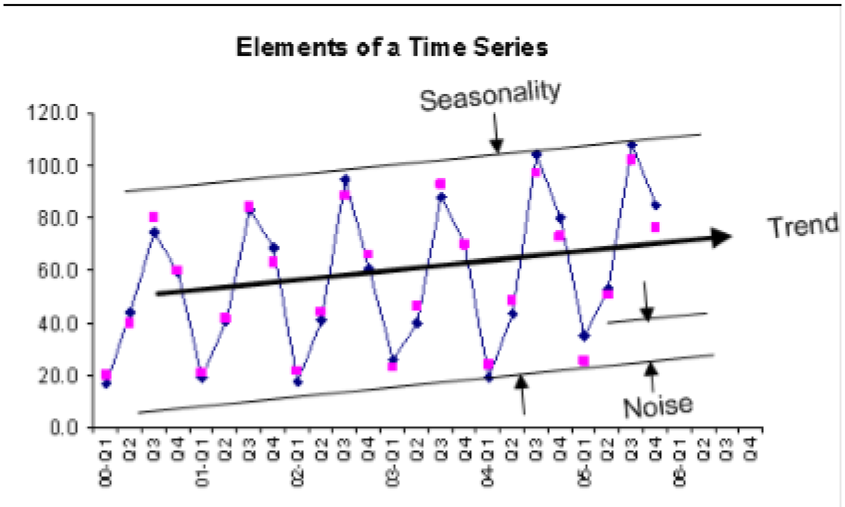
$Y_t = S_t + T_t + \epsilon_t$

For Multiplicative Time Series,

$Y_t = S_t \times T_t \times \epsilon_t$

In general, a multiplicative time series can be converted to additive by taking a log of the time series.

The following diagram shows the elements of a time series:



Analysis

In this analysis, we will use the bike sharing dataset. The data set derived from the “UCI Machine Learning Repository”. The data is related to the capital of Washington D.C. USA ridership with two-year historical log corresponding to years 2011 and 2012.

```
> #reading in data
> bike<-read.csv(file.choose(), header = T)
> bike$dteday<-as.Date(bike$dteday, format="%m/%d/%y" > str(bike)
'data.frame':   731 obs. of  16 variables:
 $ instant   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : Date, format: NA NA NA ...
 $ season    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ yr        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ mnth      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ weekday   : int  6 0 1 2 3 4 5 6 0 1 ...
 $ workingday: int  0 0 1 1 1 1 1 0 0 1 ...
 $ weathersit: int  2 2 1 1 1 1 2 2 1 1 ...
 $ temp      : num  0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num  0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num  0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num  0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : int  331 131 120 108 82 88 148 68 54 41 ...
 $ registered: int  654 670 1229 1454 1518 1518 1362 891 768 1280 ...
```

```
 $ cnt       : int  985 801 1349 1562 1600 1606 1510 959 822 1321 ...
```

Season (spring, summer, fall, winter) changes the pattern of bookings and bike sharing. Year and month show us a clear picture about the timings of the booking. Holiday and weekday describe the pattern of changes in bookings. Weathersit points out the conditions like fewer clouds or party clouds, etc. Temperature and humidity are the normalized weather condition. Wind speed gives the average speed which gives us a point to check. “Cnt” depicts the average amount of count of the bikes booked.

In the initial part, we load the data in R and extract the trend, seasonality and error by decomposition.

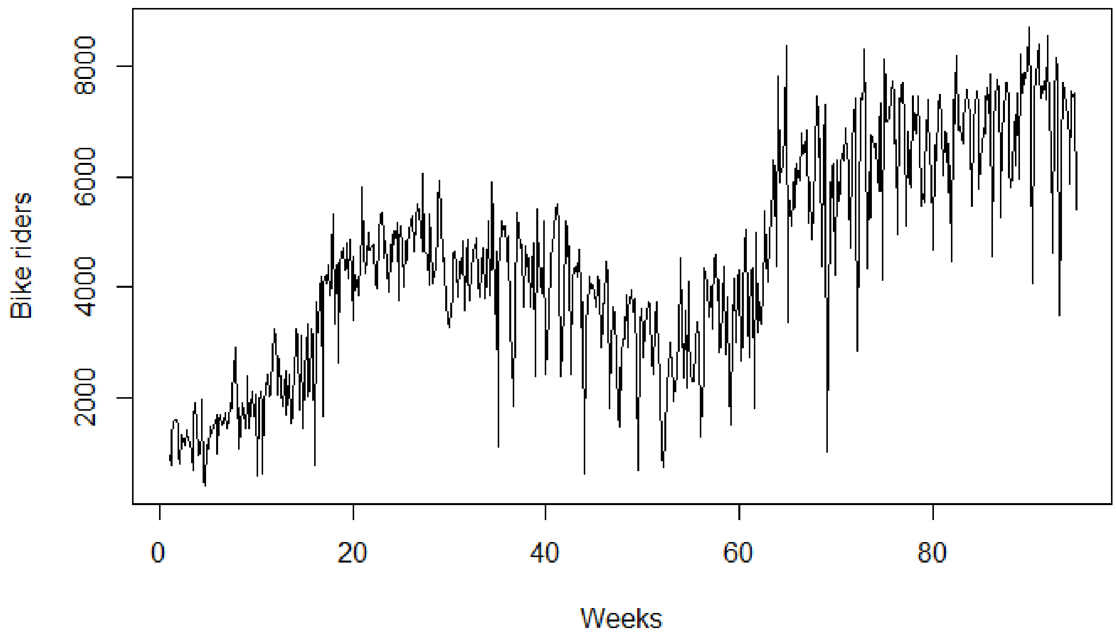
```
> summary(bike)
    instant          dteday         season           yr              mnth      
 Min.   :  1.0   Min.   :NA    Min.   :1.000   Min.   :0.0000   Min.   : 1.00  
 1st Qu.:183.5   1st Qu.:NA    1st Qu.:2.000   1st Qu.:0.0000   1st Qu.: 4.00  
 Median :366.0   Median :NA    Median :3.000   Median :1.0000   Median : 7.00  
 Mean   :366.0   Mean   :NA    Mean   :2.497   Mean   :0.5007   Mean   : 6.52  
 3rd Qu.:548.5   3rd Qu.:NA    3rd Qu.:3.000   3rd Qu.:1.0000   3rd Qu.:10.00  
 Max.   :731.0   Max.   :NA    Max.   :4.000   Max.   :1.0000   Max.   :12.00  
                 NA's   :731                                                   
    holiday           weekday        workingday      weathersit         temp   
 Min.   :0.00000   Min.   :0.000   Min.   :0.000   Min.   :1.000   Min.   :0.
05913  
 1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.000   1st Qu.:1.000   1st Qu.:0.
33708  
 Median :0.00000   Median :3.000   Median :1.000   Median :1.000   Median :0.
49833  
 Mean   :0.02873   Mean   :2.997   Mean   :0.684   Mean   :1.395   Mean   :0.
49538  
 3rd Qu.:0.00000   3rd Qu.:5.000   3rd Qu.:1.000   3rd Qu.:2.000   3rd Qu.:0.
65542  
 Max.   :1.00000   Max.   :6.000   Max.   :1.000   Max.   :3.000   Max.   :0.
86167  

     atemp              hum           windspeed           casual           regi
stered
 Min.   :0.07907   Min.   :0.0000   Min.   :0.02239   Min.   :   2.0   Min.   
:  20  
 1st Qu.:0.33784   1st Qu.:0.5200   1st Qu.:0.13495   1st Qu.: 315.5   1st Qu
.:2497  
 Median :0.48673   Median :0.6267   Median :0.18097   Median : 713.0   Median 
:3662  
 Mean   :0.47435   Mean   :0.6279   Mean   :0.19049   Mean   : 848.2   Mean   
:3656  
 3rd Qu.:0.60860   3rd Qu.:0.7302   3rd Qu.:0.23321   3rd Qu.:1096.0   3rd Qu
.:4776  
 Max.   :0.84090   Max.   :0.9725   Max.   :0.50746   Max.   :3410.0   Max.   
:6946  

      cnt      
 Min.   :  22  
 1st Qu.:3152  
 Median :4548  
 Mean   :4504  
 3rd Qu.:5956  
 Max.   :8714  
```

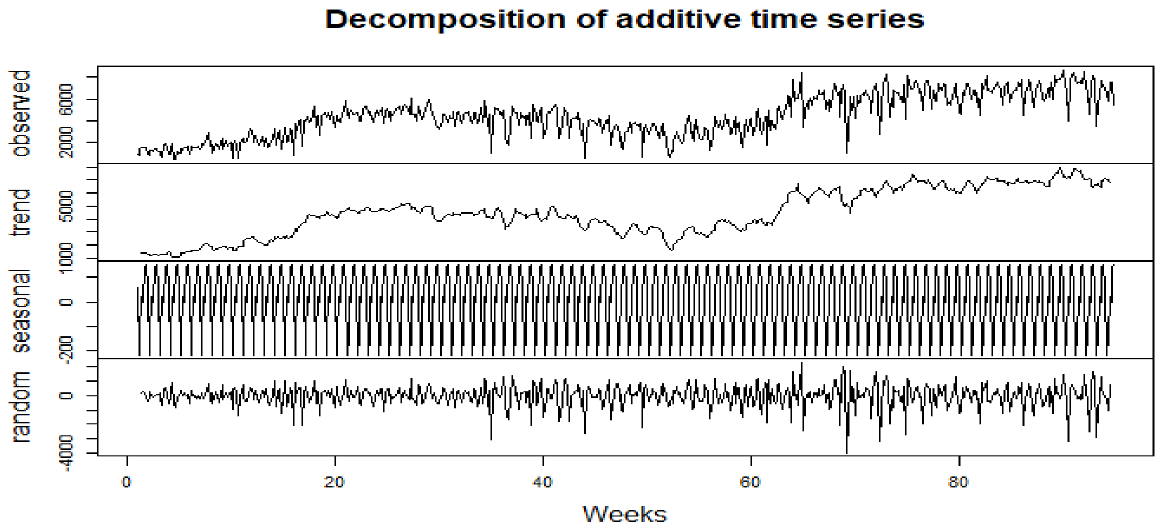
Splitting data into training and testing sets. We take 90% of data for the training set and rest 10% for the testing. The plot below shows the trend of the data. The time series described an additive model since the random fluctuations in the data are roughly constant in size over time.

```
> data_ts <-msts(bike$cnt, seasonal.periods=c(7))
> train_ts <- head(data_ts, round(length(data_ts) * 0.9))
> test_ts <- tail(data_ts, round(length(data_ts) * 0.1))
> plot(train_ts, xlab="Weeks", ylab="Bike riders")
```



The decomposition plot below shows the original time series on top, the estimated trend component, the estimated seasonal component, and the estimated remainder component. We can see, there is a very strong seasonality over the weeks.

```
> plot(decompose(train_ts, type="add"), xlab="Weeks")
```



We see that the estimated trend component shows a gradual increase from about 1000 from the first week to about 6000.

Since we have an additive model, we can seasonally adjust the time series by estimating the seasonal component and subtracting the estimated seasonal component from the original time series.

Fitting an ARIMA model requires the series to be stationary (Dalinina, n.d.). We check for the stationarity of the series by using the ADF test (Augmented Dickey-Fuller). The test shows that the p-value is .19 which is less than the 0.05 threshold. This indicates series is non-stationary.

```
> library(tseries)
> adf_test <- adf.test(train_ts, alternative='stationary')
> print(adf_test)

        Augmented Dickey-Fuller Test

data:  train_ts
Dickey-Fuller = -2.8989, Lag order = 8, p-value = 0.1978
alternative hypothesis: stationary
```

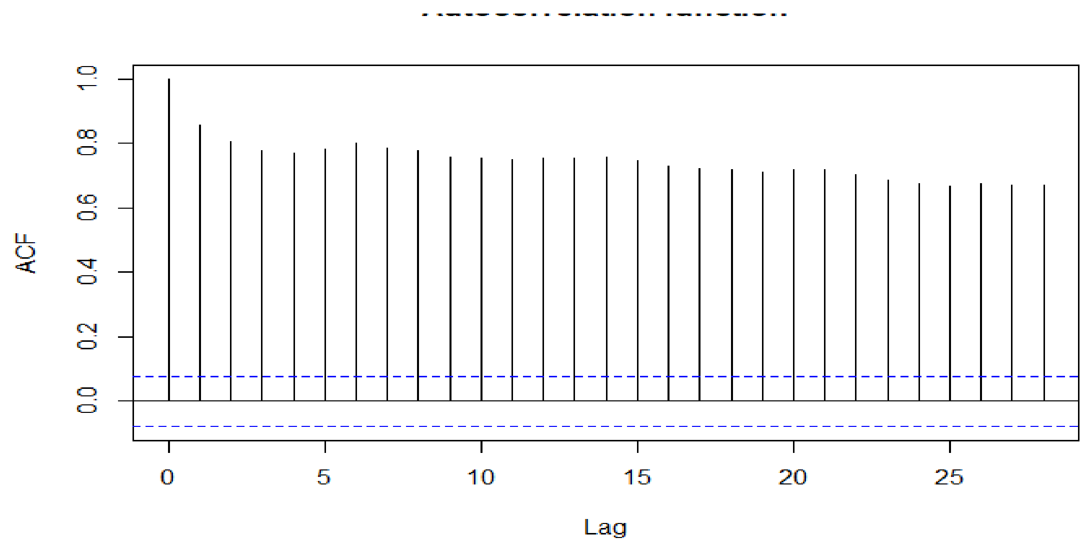
Here are the ACF and PACF plots for the series. ACF is a plot of total correlation between different lag functions (correlation of x(t) with x(t-1) , x(t-2) and so on).

For an MA series (looking at ACF), if the total correlation chart cuts off at nth lag, our lag is nth for MA series. However, if the correlation gradually goes down without a cutoff, we have MA(0).

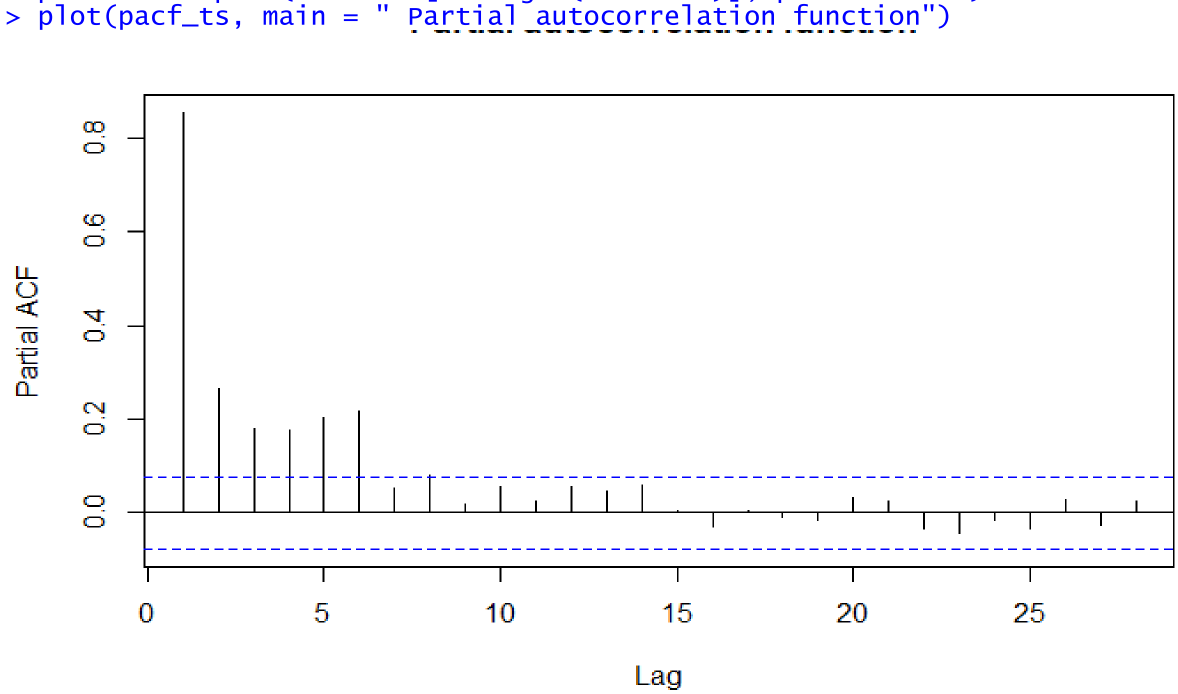
We can estimate the seasonal MA from ACF and AR from PACF for this series. Looking at the plots we can say that this is an 'Auto Regressive' (AR) type of series. Thus, the order will be AR(6) and MA(0).

```
> acf_ts <- acf(train_ts[1:length(train_ts)], plot = FALSE)
```

```
> plot(acf_ts, main = "Autocorrelation function", cex=0.5)
```




```
> pacf_ts <- pacf(train_ts[1:length(train_ts)], plot = FALSE)
> plot(pacf_ts, main = " Partial autocorrelation function")
```

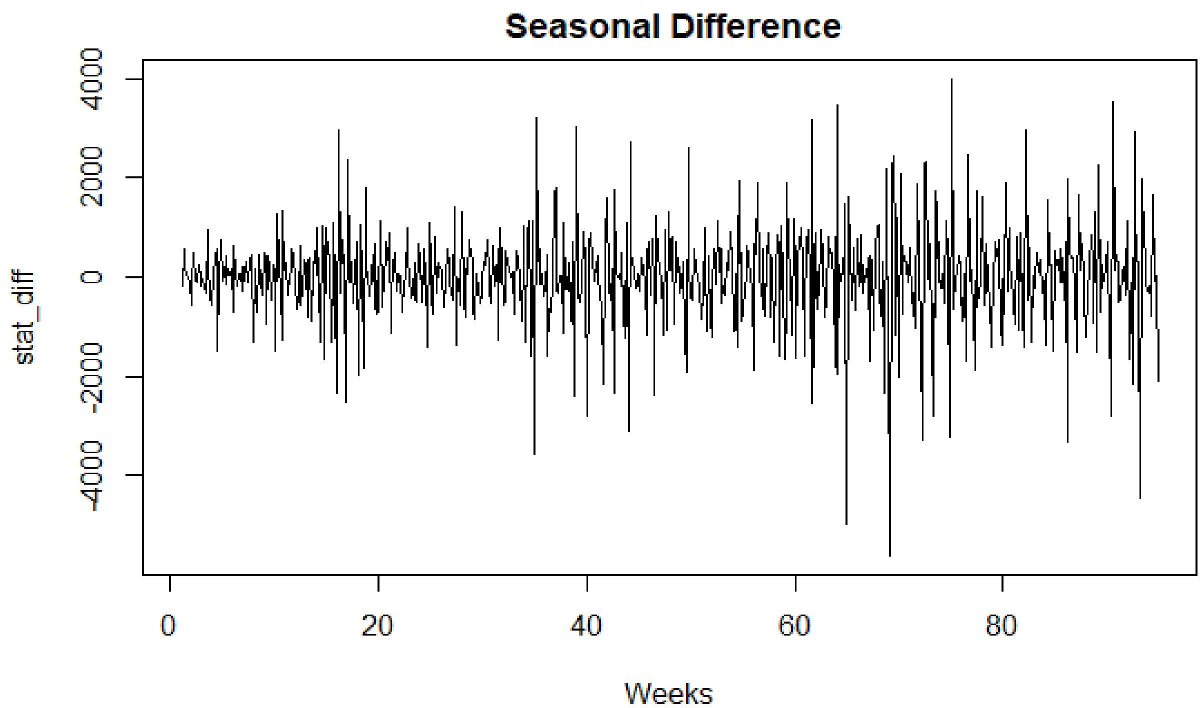


As we can see, the decay of ACF chart is very slow, which means that the population is not stationary.

Differencing method is one of the methods to convert non-stationary series to the stationary one. Usually, non-stationary series can be corrected by a simple transformation; differencing. Differencing the series can help in removing its trend or cycles.

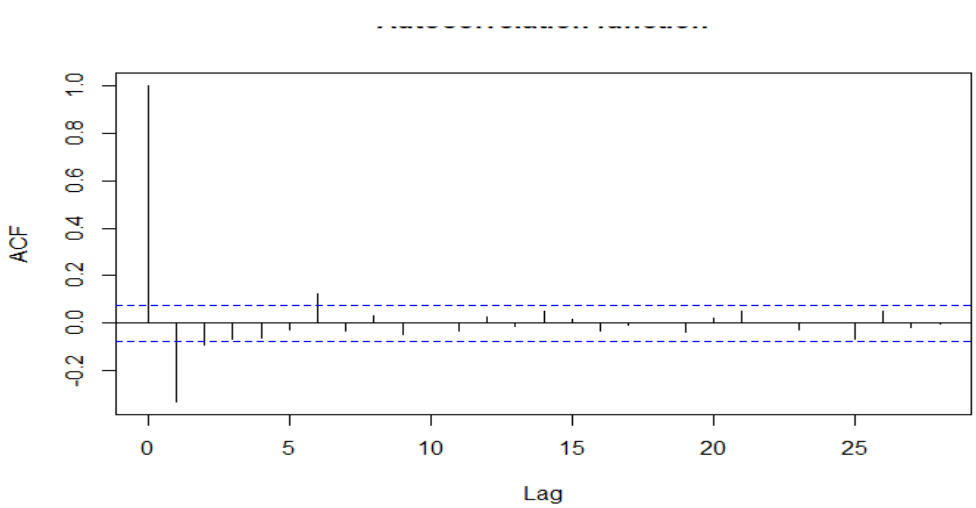
The idea behind differencing is that, if the original data series does not have constant properties over time, then the change from one period to another might (Dalinina, n.d.). After plotting the differenced series, we see an oscillating pattern and no visible strong trend. This suggests at differencing of order 1 terms is sufficient and should be included in the model.

```
> stat_diff <- diff(train_ts, differences = 1)
> plot(stat_diff, main = " Seasonal Difference", xlab= "Weeks")
```



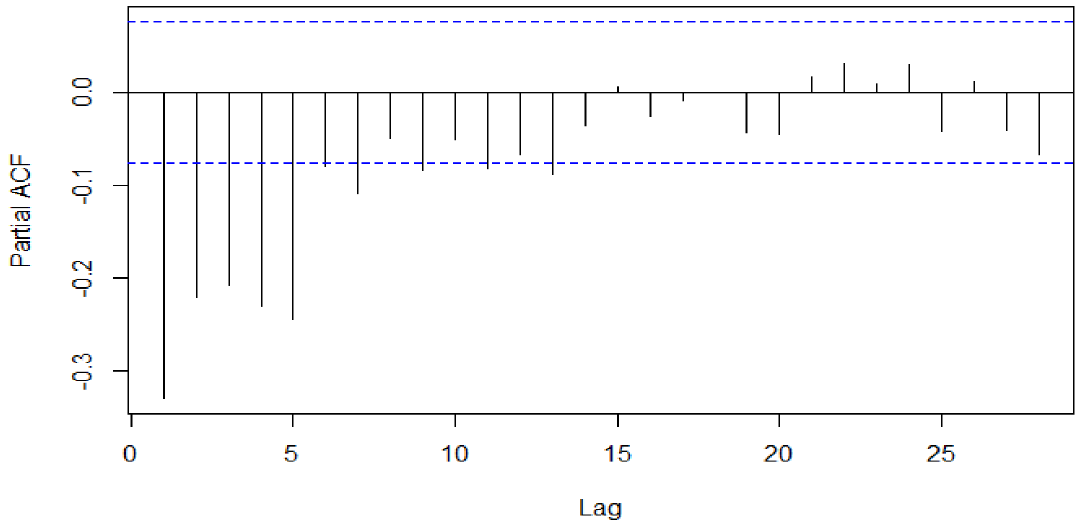
We can estimate the non-seasonal MA from ACF and AR from PACF for this series. Looking at the plots we can say that this is an 'Auto Regressive'(AR) type of series. Thus, the order will be AR (7) and MA (0).

```
> acf_ts <- acf(stat_diff[1:length(stat_diff)], plot = FALSE)
> plot(acf_ts, main = "Autocorrelation function")
```




```
> pacf_ts <- pacf(stat_diff[1:length(stat_diff)], plot = FALSE)
```

```
> plot(pacf_ts, main = " Partial autocorrelation function")
```

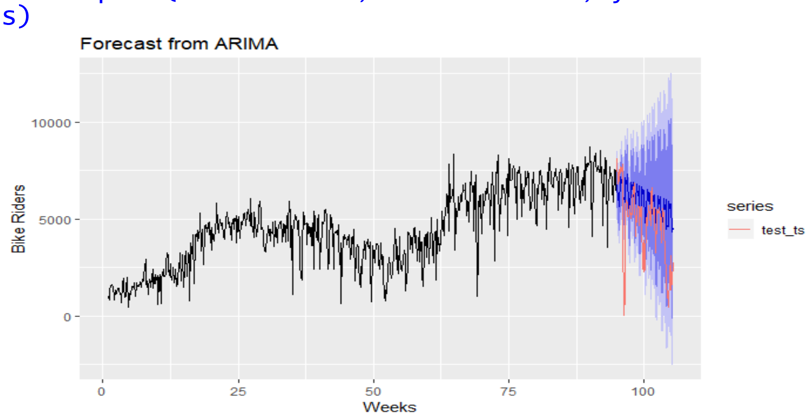


We fitted a model with the order (7,1,0) and seasonal (6,1,0) with period 7 weeks. Forecasts from the model for the next 10 weeks are also shown in the plot below. The forecasts follow the recent trend in the data, because of the differencing. We can see that the forecast is not very close to the actual data (blue line).

We can see the model captures the general trend well but is not able to capture sharp ups and downs. It is probably not the best model for decision making on prediction about the number of bikes sharing users.

The few key reasons this model doesn't perform well are that the model considers only time as a factor, and no other crucial features like whether it was working day or holiday, sunny day or rainy day etc. weather-related parameters into account.

```
> library(forecast)
> fit1 <- Arima(train_ts, order=c(7,1,0),seasonal=c(6,1,0), method = "CSS", o
ptim.method = "BFGS")
> forecast_ts <- forecast(fit1,h= length(test_ts))
> autoplot(forecast_ts, xlab="Weeks", ylab= "Bike Riders") + autolayer(test_t
s)
```



Conclusion

In conclusion, the resultant time series forecasting model was not very close to the actual data due to the shortcomings we highlighted. A better model would some more predictor of bike usage characteristics. Ensemble methods such the Random Forest would prove to be a better predictive model.

References

1. Brett, L. (2015). *Machine Learning with R*. Birmingham, UK. Packt Publishing Ltd.
2. Peter, B. (2017). Practical Statistics for Data Scientists. Boston, MA. O'Reilly Media, Inc.