WELCOME TO PSY 653 LAB!

MODULE 07: TIME SERIES AND THE ANALYSIS OF LONGITUDINAL DATA – ARIMA MODELING

OBJECTIVES

- Explanation of ARIMA models
- Seasonality and stationarity in ARIMA models (The "I" in ARIMA)
- Differencing
- Explanation of AR & MA terms in ARIMA
- Coding tutorial

ARIMA MODELING

- ARIMA stands for: **Auto Regressive Integrated Moving Average**
  - **Auto Regressive (p) =** refers to the lag function of the differenced series. An auto regressive model makes use of a previous time step to make a prediction about a future time step.
  - **Integrated (d) =** Is the number of differences used to make the time series *stationary*. This will be explained in more detail later.
  - **Moving Average (q) =** How much the average changes from one time point to the next.

- **In short, an ARIMA models use previous time periods to predict outcomes in the future.**

OUR GOAL

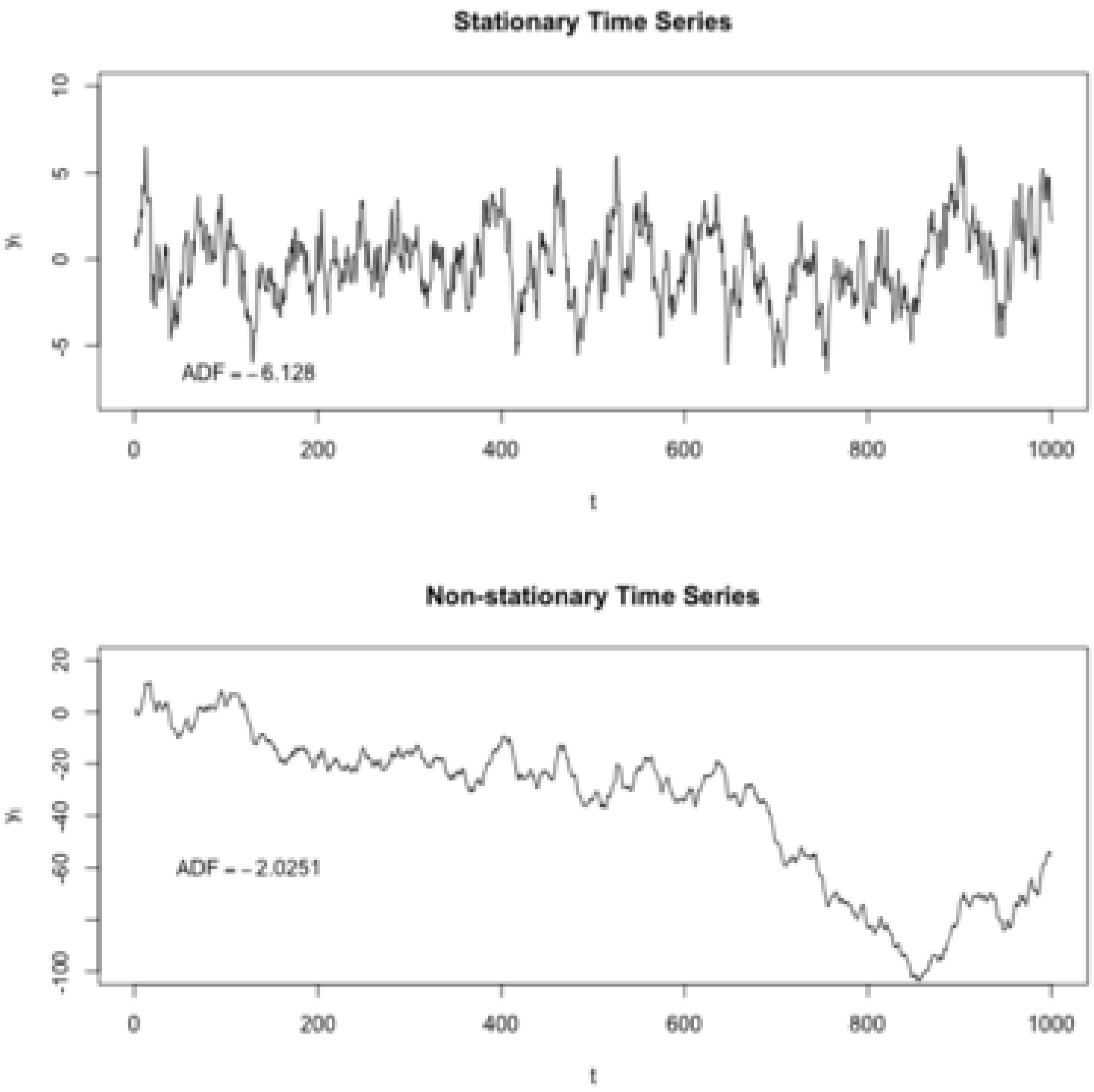
- Today we will be running a series of statistics to fit the best ARIMA model given our data.
  - By fitting the model, we will be able to forecast what data in the future will look like.

- We will be calculating three components to the ARIMA model
  - The number of lags needed in the stationary series (The number of autoregressive terms to fit our model), our ***AR (AKA "p")***
  - The number of differences needed to make the data stationary (The number of nonseasonal differences), our ***I (AKA "d")***
  - The number of lags of the forecast error (The number of moving average terms), our ***MA (AKA "q")***

- With these components, we are able to fit a model that can properly forecast future events. (Or at least we hope so…)

THREE COMPONENTS TO LEARN:

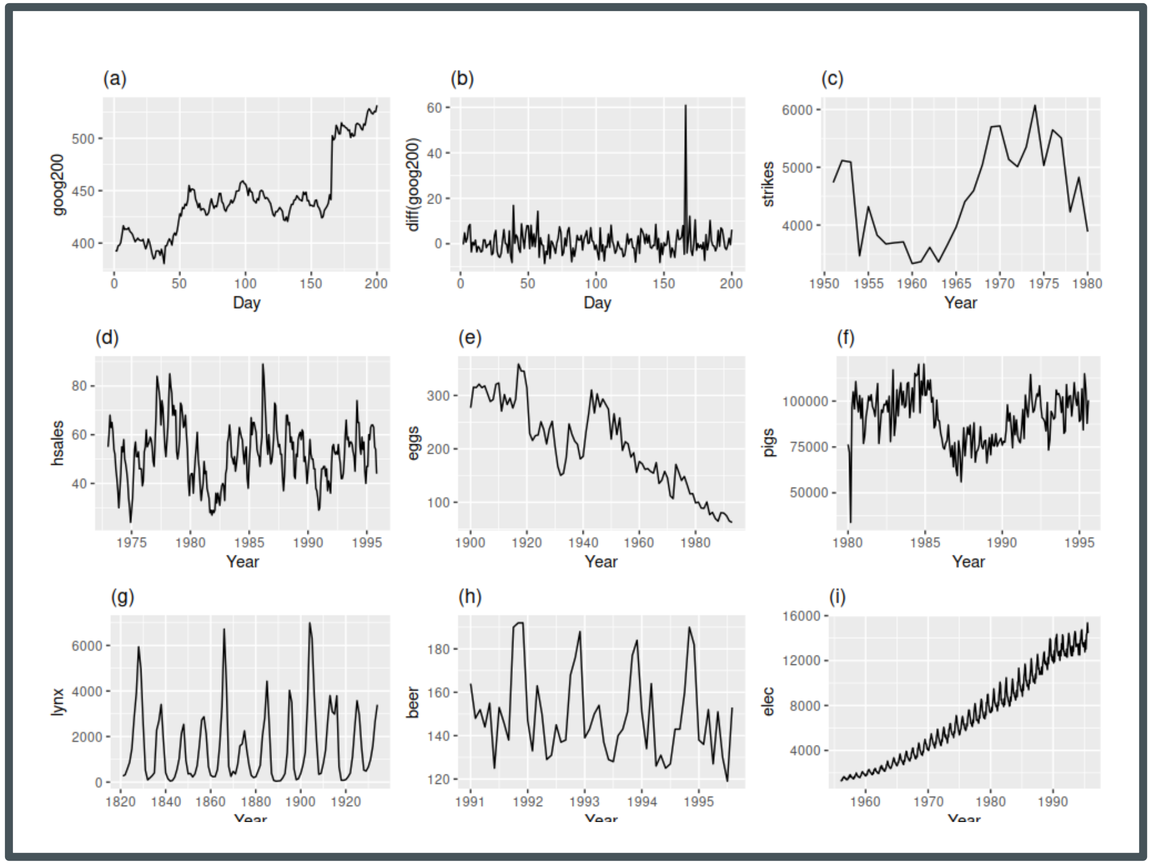
- **Seasonal component** refers to fluctuations in the data related to calendar cycles. Usually, seasonality is fixed at some number; for instance, quarter or month of the year.
- **Trend component** is the overall pattern of the series
- **Cycle component** consists of decreasing or increasing patterns that are not seasonal. Usually, trend and cycle components are grouped together. Trend-cycle component is estimated using moving averages.



STATIONARITY

- A stationary model is one that does not have a seasonality component to it. In other words, the day/month/year/decade does not influence what the datapoint will be.

ARIMA MODEL ASSUMPTIONS

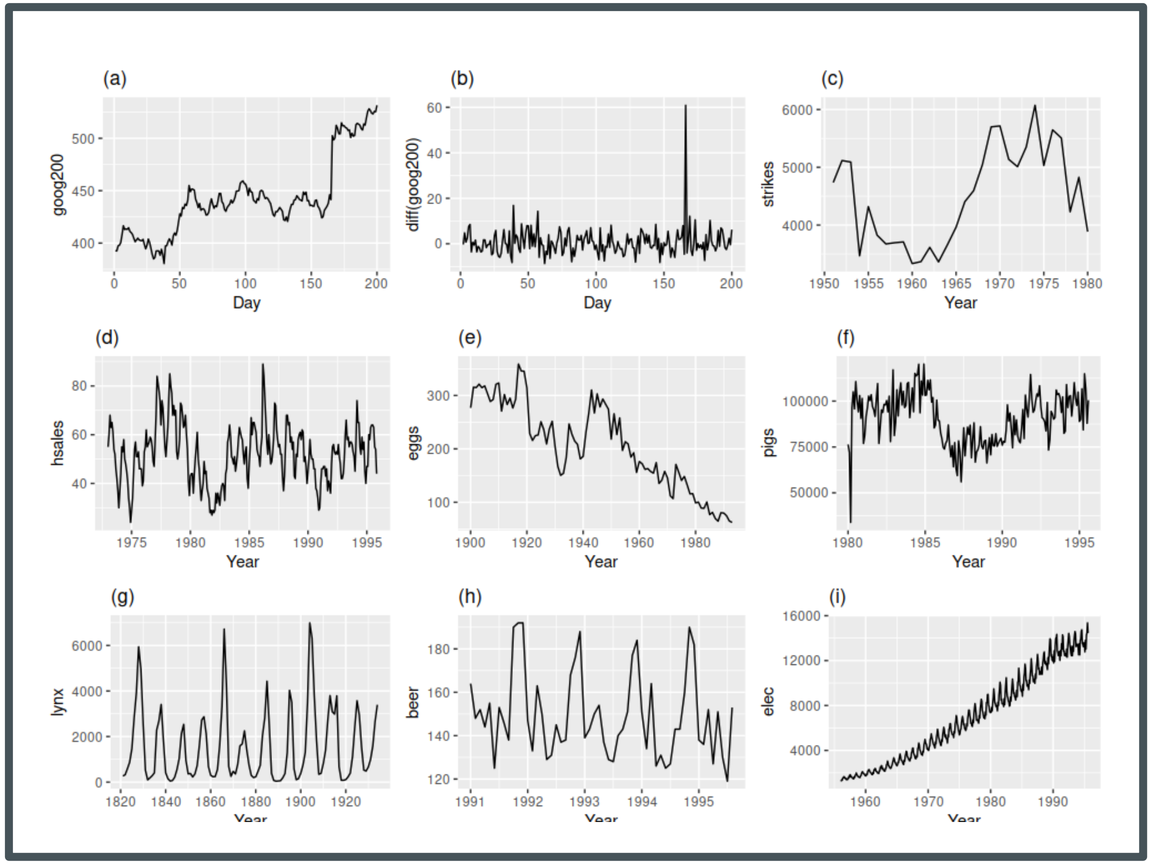


- 1.) **Data should be stationary:** by stationary it means that the properties of the series doesn't depend on the time when it is captured. AKA there is NO seasonality in the trend (The variable being modeled does not depend on the day/month/year you choose it).

- 2.) **Data should be univariate:** ARIMA models only works on a single variable.

https://otexts.com/fpp2/stationarity.html

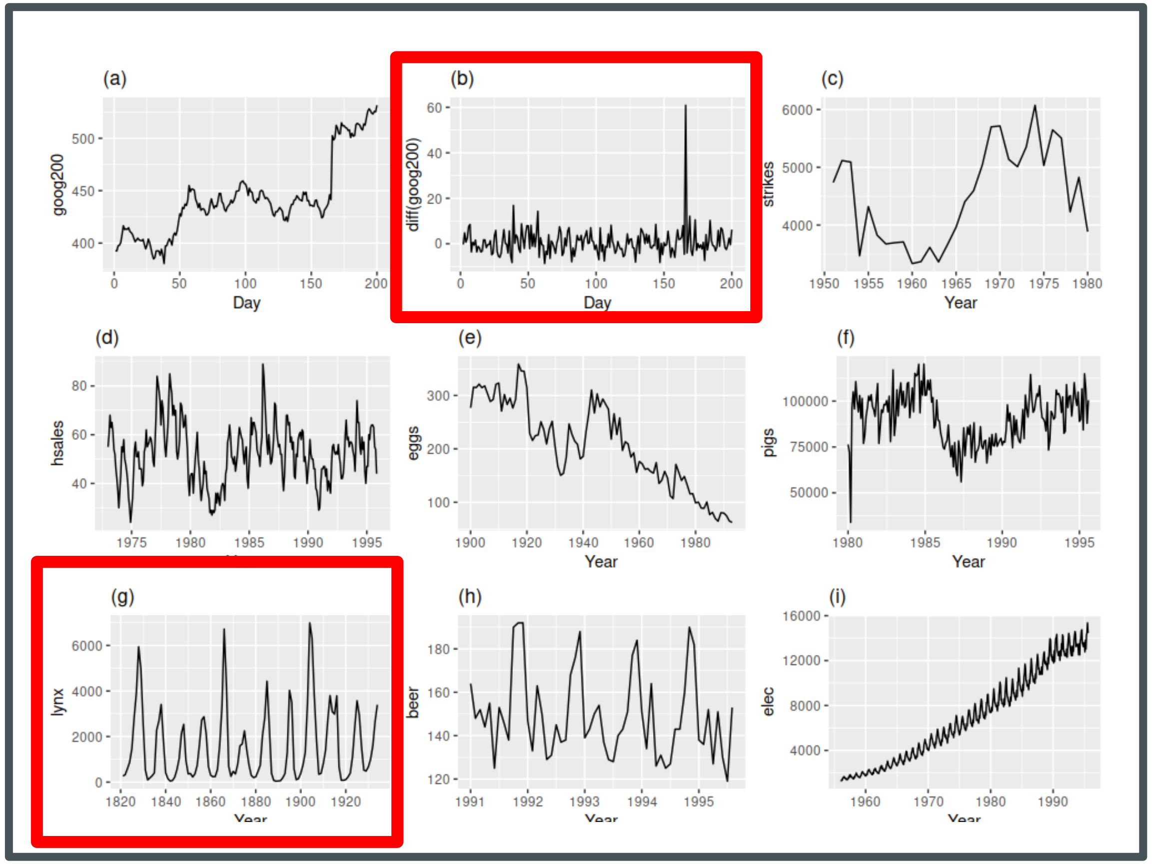
ARIMA MODEL ASSUMPTIONS



https://otexts.com/fpp2/stationarity.html

- 1.) **Data should be stationary:** by stationary it means that the properties of the series doesn't depend on the time when it is captured. AKA there is NO seasonality in the trend (The variable being modeled does not depend on the day/month/year you choose it).
    - Look at the graphs on the left, which models are stationary?

ARIMA MODEL ASSUMPTIONS



- 1.) **Data should be stationary:** by stationary it means that the properties of the series doesn't depend on the time when it is captured. AKA there is NO seasonality in the trend (The variable being modeled does not depend on the day/month/year you choose it).
  - Look at the graphs on the left, which models are stationary?
    - Only graph (b) and (g) show a stationary trend. The rest will need to be transformed before we can run an ARIMA model on it. At first glance, (g) may appear seasonal, but the cycles are aperiodic.

https://otexts.com/fpp2/stationarity.html

HOW DO WE MAKE A DATASET STATIONARY

- Through *differencing!*
  - Differencing = computing the differences between consecutive observations.
  - The more seasonality (non-stationarity) our data has, the more differencing we will need to do.

- Luckily, the R packages we will be using handle all of this for us (*tseries* and *forecast* packages)

- By determining the number of differences you need, you have figured out the "I" (Integrated) in the AR***I***MA model. This is otherwise characterized as a (d).

WHAT ABOUT THE AR (p) AND MA (q) PORTIONS OF THE ARIMA MODEL? WHAT EXACTLY DOES THIS MEAN?

- We will be looking at displays later in the slides to determine what the best AR (p) and MA (q) terms for our model will be.

- But in the end, we will get models that look as so: arima(p = 1, d = 1, q = 1).
  - In short, we are saying that the model requires 1 AR (p) term, 1 differencing (d) terms, and 1 MA (q) term.
    - A model with an AR term specifies that the model tends to return to its mean relatively quickly. If it had to AR terms, then it would indicate that there is an oscillatory pattern associated with it.
    - A model with an MA term shows that the Moving average tends to experience "shocks". As in, it has severe spikes in the data. The more MA terms we have, the more "shocks" the model has.

A COUPLE OF NOTES:

- **The process of fitting an ARIMA model is completely exploratory**
  - We will be using statistical tests to guide our decisions, but different people may come to different conclusions.
- **Some components of this unit will seem like a "black box"**
  - The program will do a lot of the work for us. The underlying equations and processes are beyond the scope of this class.

CREATE A NEW R-PROJECT AND R-NOTEBOOK!

Download the “day.csv” file from Canvas and save it into your R-project file

DATASET EXPLANATION

- This dataset contains the daily count of rental bikes between years 2011 and 2012 in Capital bikeshare system with the corresponding weather and seasonal information.
- Variables of interest:
  - **Date**: Date
  - **season**: 1 = Winter, 2 = Spring, 3 = Summer, 4 = Fall
  - **cnt** = # of bikes rented on that day.

Data obtained from: https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset

Load libraries
````
```{r}
install.packages("forecast")
install.packages("tseries")



library(forecast)
library(tseries)
library(psych)
library(tidyverse)
```
````

read_csv()… A LITTLE DIFFERENTLY

- We need to specify that the *Date* variable is indeed a date. Additionally, we need to specify the date format, "%m/%d/%Y" is telling R that the format of the date is in the American month/day/year format.

```
```{r}
daily_data <- read_csv("day.csv",
                       col_types = cols(Date = col_date("%m/%d/%Y")))
```
```

select() VARIABLES OF INTEREST

```{r}
daily_data <- select(daily_data, Date, season, cnt)
```

```{r}
describe(daily_data)
```

R Console

data.frame
3 x 13

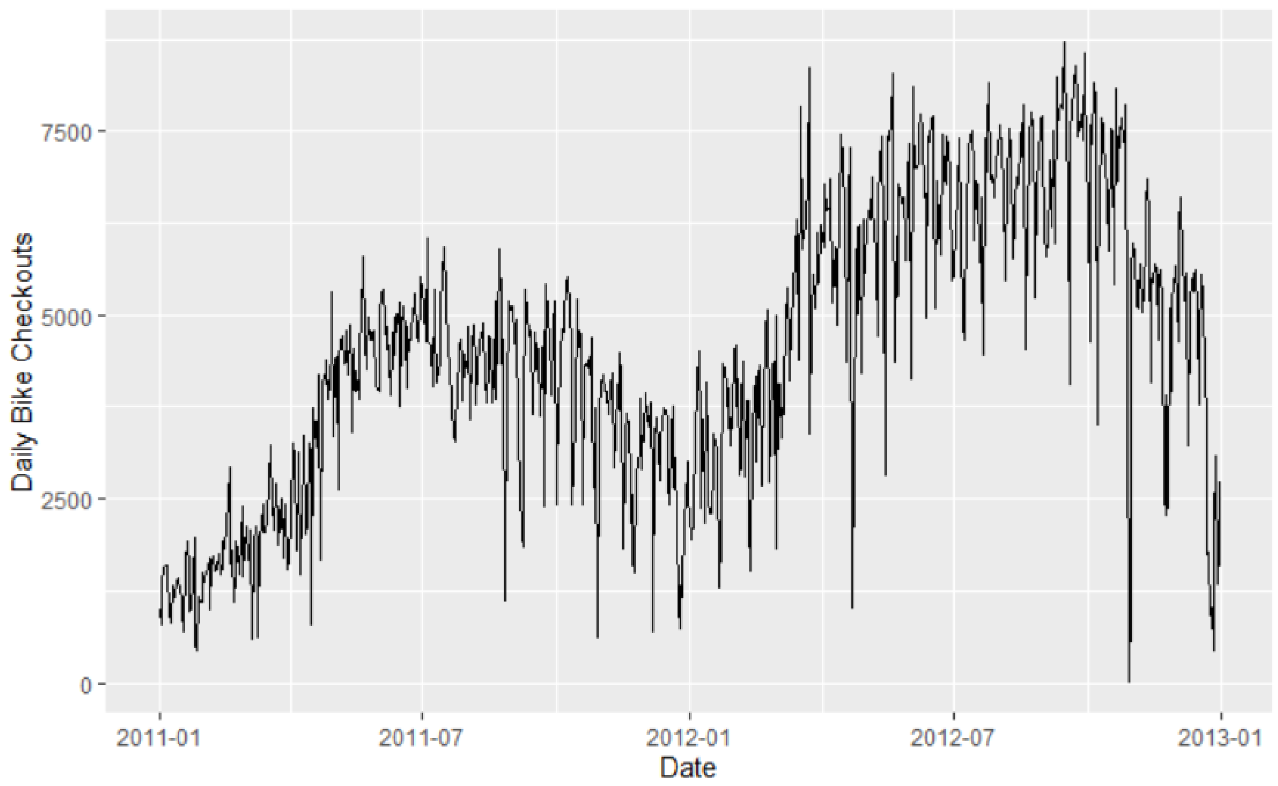
	vars <dbl>	n <dbl>	mean <dbl>	sd <dbl>	median <dbl>	trimmed <dbl>	mad <dbl>	min <dbl>	max <dbl>
Date	1	731	NaN	NA	NA	NaN	NA	Inf	-Inf
season	2	731	2.50	1.11	3	2.50	1.48	1	4
cnt	3	731	4504.35	1937.21	4548	4517.19	2086.02	22	8714

3 rows | 1-10 of 13 columns

describe() DATA

PLOT THE DATA

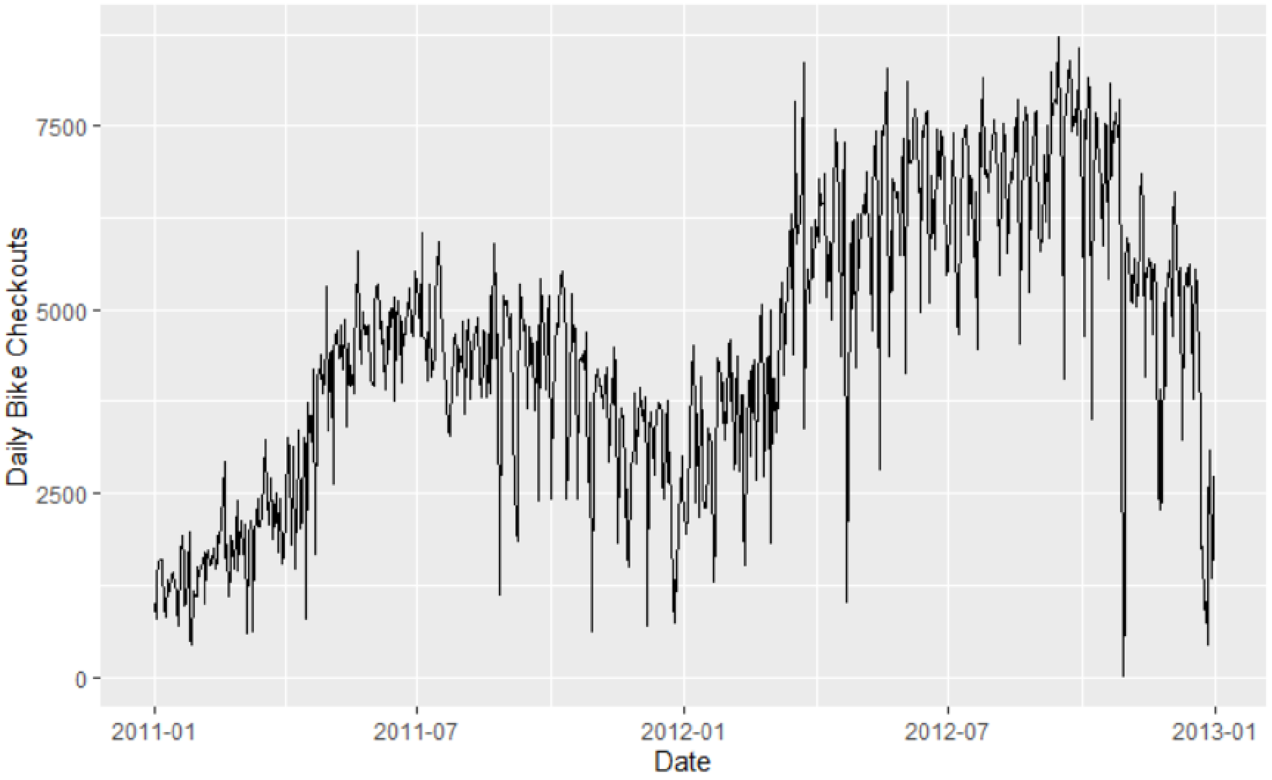
```{r}
ggplot(daily_data, aes(x = Date, y = cnt)) +
  geom_line() +
  ylab("Daily Bike Checkouts")
```



IS THERE SEASONALITY?

- Looks like it! More bikes seem to be rented in the summer months. Therefore, the data does NOT look stationary.
- …We'll statistically check next

```{r}
ggplot(daily_data, aes(x = Date, y = cnt)) +
  geom_line() +
  ylab("Daily Bike Checkouts")
```



STATISTICALLY CHECKING FOR STATIONARITY

- 1. Convert data into a time series object (using ts())
- 2. Use the adf.test() function to test for stationarity
  - ADF test stands for "Augmented Dickey-Fuller" test. It assesses if the data is stationary or not.

#Step 1: COnvert to a time series object

```{r}
# Convert to a timeseries
count_ts <-  ts(daily_data$cnt)
```

Step 2, check stationairity of the data

```{r}
# Data is not stationary we need to do a difference count
adf.test(count_ts)

```

```
        Augmented Dickey-Fuller Test

data:  count_ts
Dickey-Fuller = -1.6351, Lag order = 9, p-value = 0.7327
alternative hypothesis: stationary
```

#Step 1: COnvert to a time series object

```{r}
# Convert to a timeseries
count_ts <-  ts(daily_data$cnt)
```

Step 2, check stationairity of the data

```{r}
# Data is not stationary we need to do a difference count
adf.test(count_ts)

```

```
	Augmented Dickey-Fuller Test

data:  count_ts
Dickey-Fuller = -1.6351, Lag order = 9, p-value = 0.7327
alternative hypothesis: stationary
```

A nonsignificant p-value indicates the data is NOT stationary

REMOVE SEASONALITY FROM THE DATA
STEP 1: USE stl() TO SMOOTH OUR DATA

````
```{r}
# Set to 30 cases per month
count_ma <- ts(daily_data$cnt, frequency = 30)
decomp <- stl(count_ma, s.window="periodic")

```
````

REMOVE SEASONALITY FROM THE DATA STEP 1: USE stl() TO SMOOTH OUR DATA

Sets us up to smooth our data by month

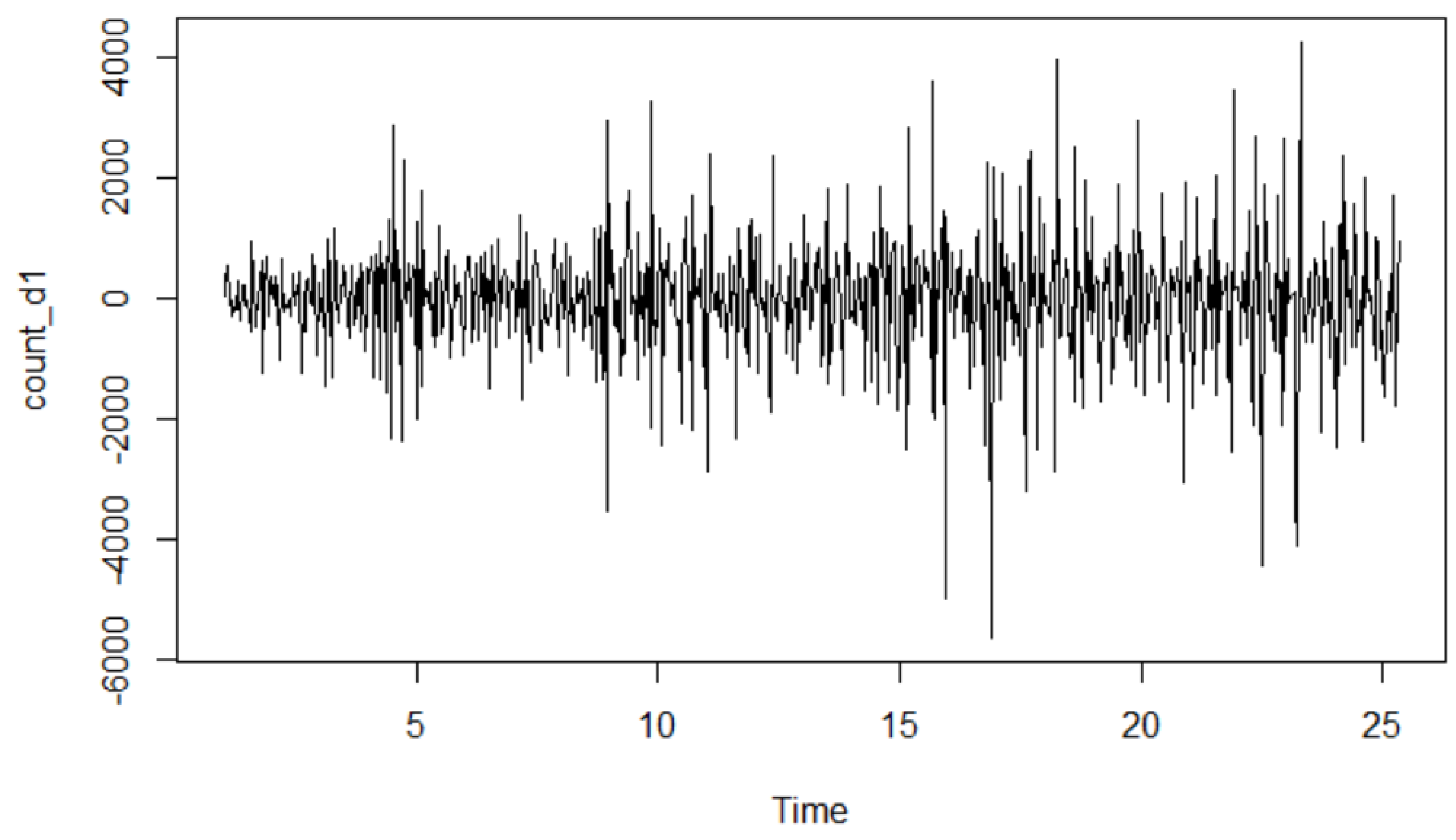
```{r}
# Set to 30 cases per month
count_ma <- ts(daily_data$cnt, frequency = 30)
decomp <- stl(count_ma, s.window="periodic")

```

Data smoothing

REMOVE SEASONALITY FROM THE DATA

STEP 2: REMOVE SEASONALITY AND PLOT THE RESULTS

```{r}
deseasonal_cnt <- seasadj(decomp)
count_d1 <-  diff(deseasonal_cnt, differences = 1)
plot(count_d1)
```



REMOVE SEASONALITY FROM THE DATA

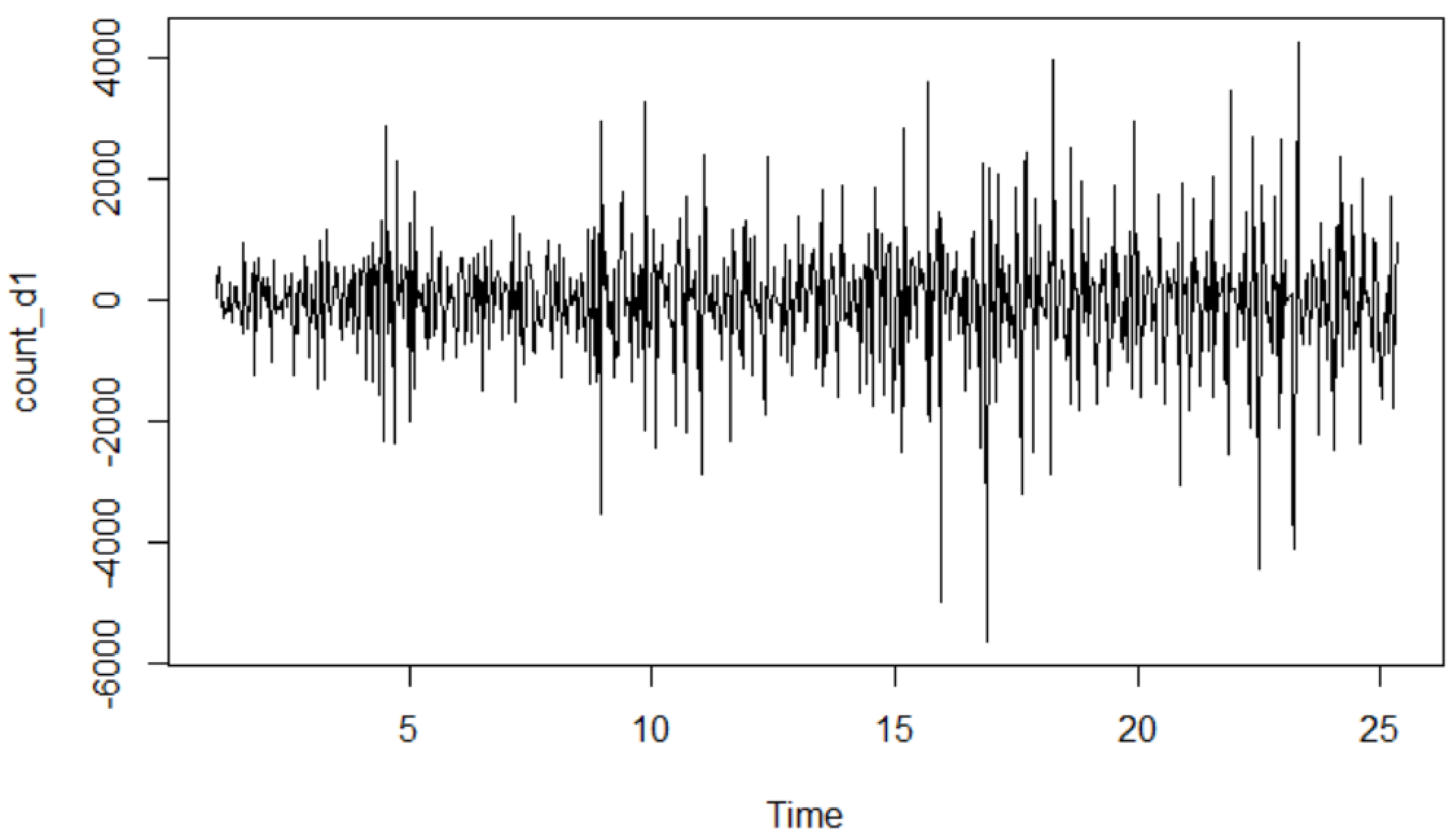
STEP 2: REMOVE SEASONALITY AND PLOT THE RESULTS

```{r}
deseasonal_cnt <- seasadj(decomp)
count_d1 <-  diff(deseasonal_cnt, differences = 1)
plot(count_d1)
```

Remove seasonality component

Set up lag difference of 1

Plot to see that seasonality has been removed



USE OUR adf.test() AGAIN TO CONFIRM SEASONALITY WAS REMOVED

```{r}
adf.test(count_d1)
```

```
p-value smaller than printed p-value
        Augmented Dickey-Fuller Test

data:  count_d1
Dickey-Fuller = -13.859, Lag order = 8, p-value = 0.01
alternative hypothesis: stationary
```

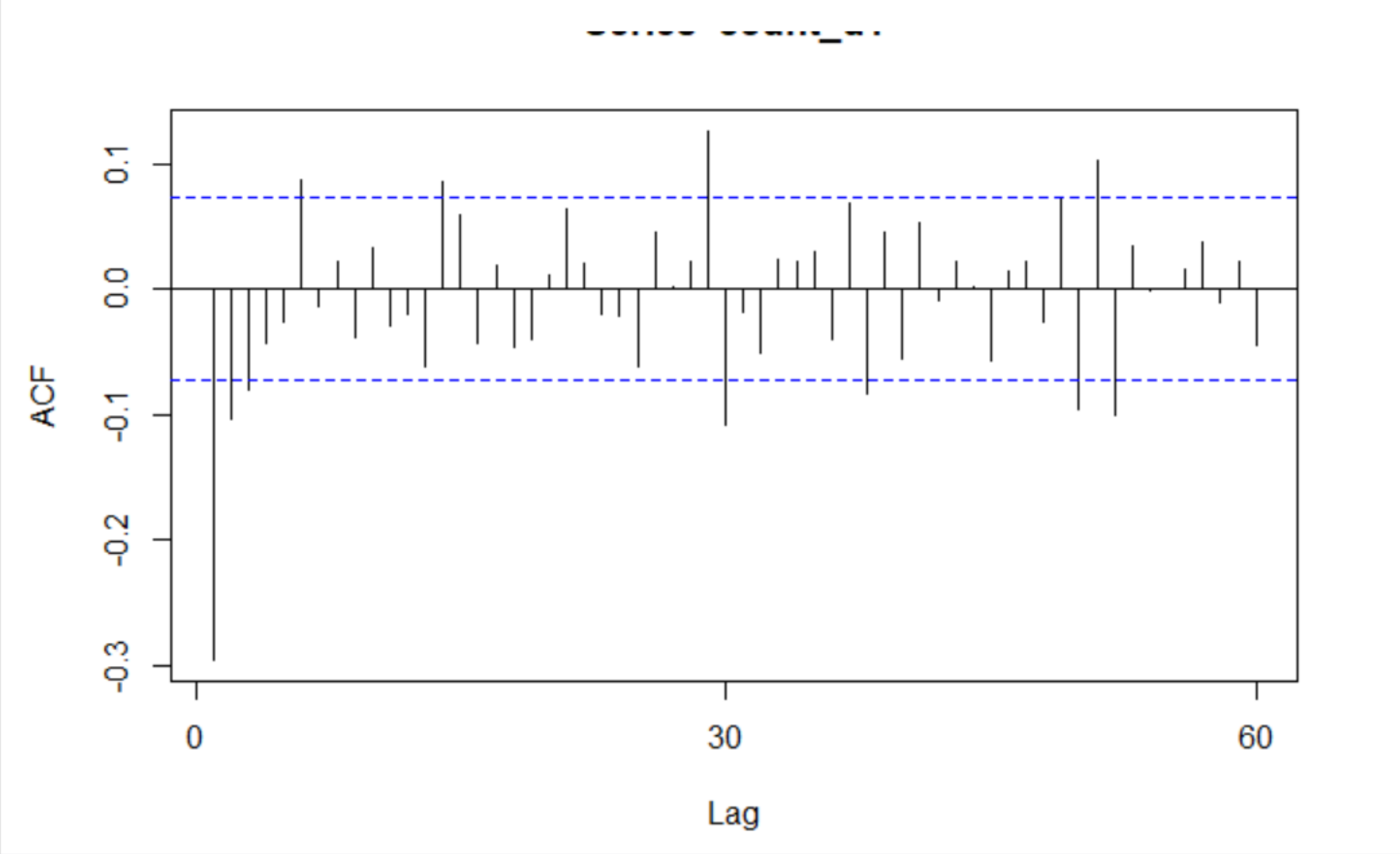
AUTOCORRELATION FUNCTION PLOTS (ACF) & PARTIAL AUTO CORRELATION FUNCTION (PACF) PLOTS TO DETERMINE THE AR (p) & MA (q) POTIONS OF OUR MODEL

- We use autocorrelation functions (ACF) and Partial autocorrelation functions (PACF) to determine what to set our *auto regressive (AR)* and *moving average (MA)* aspects of our ***AR***I***MA*** model.
  - ***ACF is used to determine what our AR should be***
  - ***PACF is used to determine what our MA should be***

- In the next set of slides, we will be outputting ACF and PACF graphs. We will literally be eyeballing these graphs to determine what we will be setting out AR and MA portions of our ARIMA model.

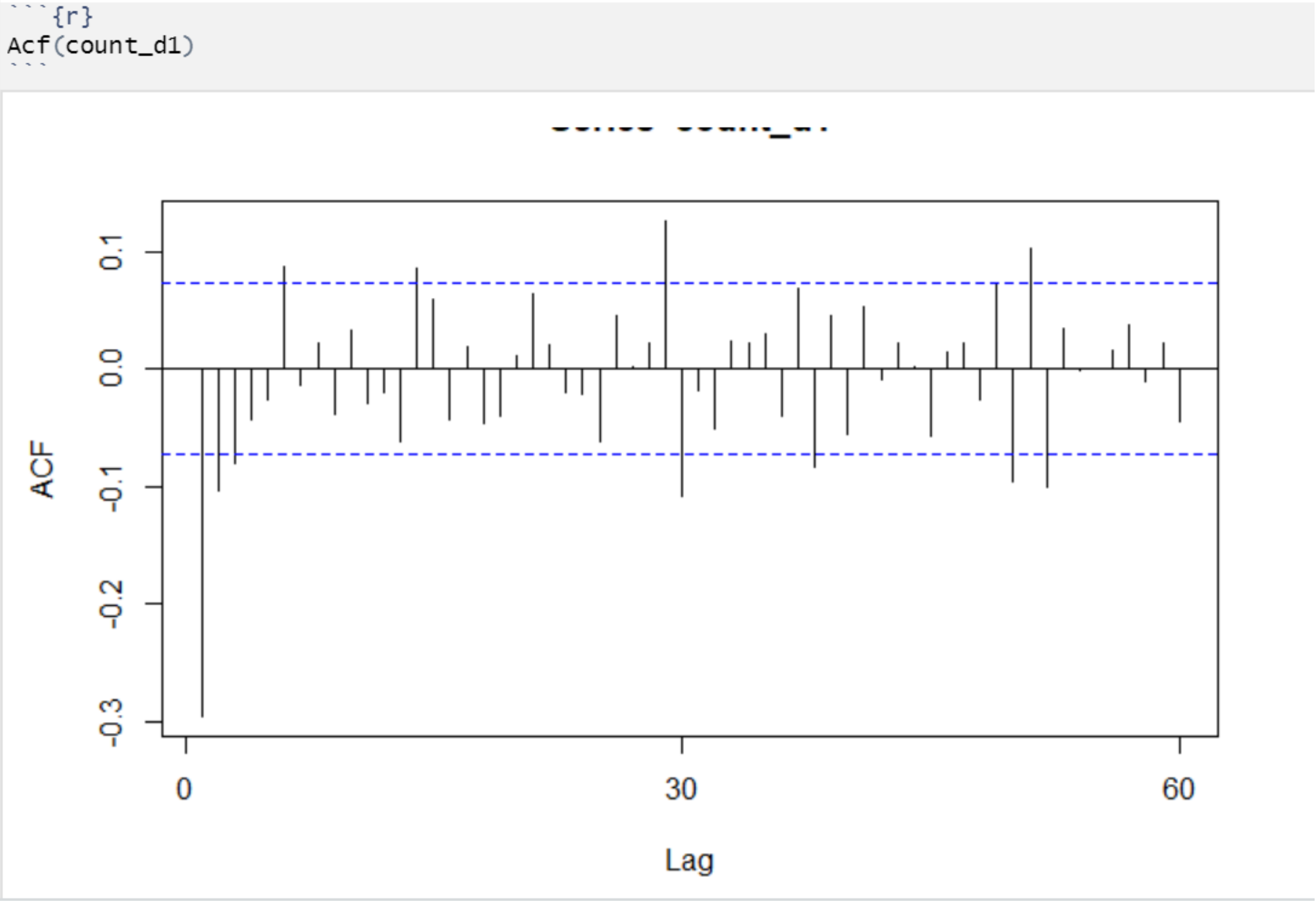
ACF PLOT: USED TO DETERMINE THE **AR (p)** OF OUR MODEL

```{r}
Acf(count_d1)
```



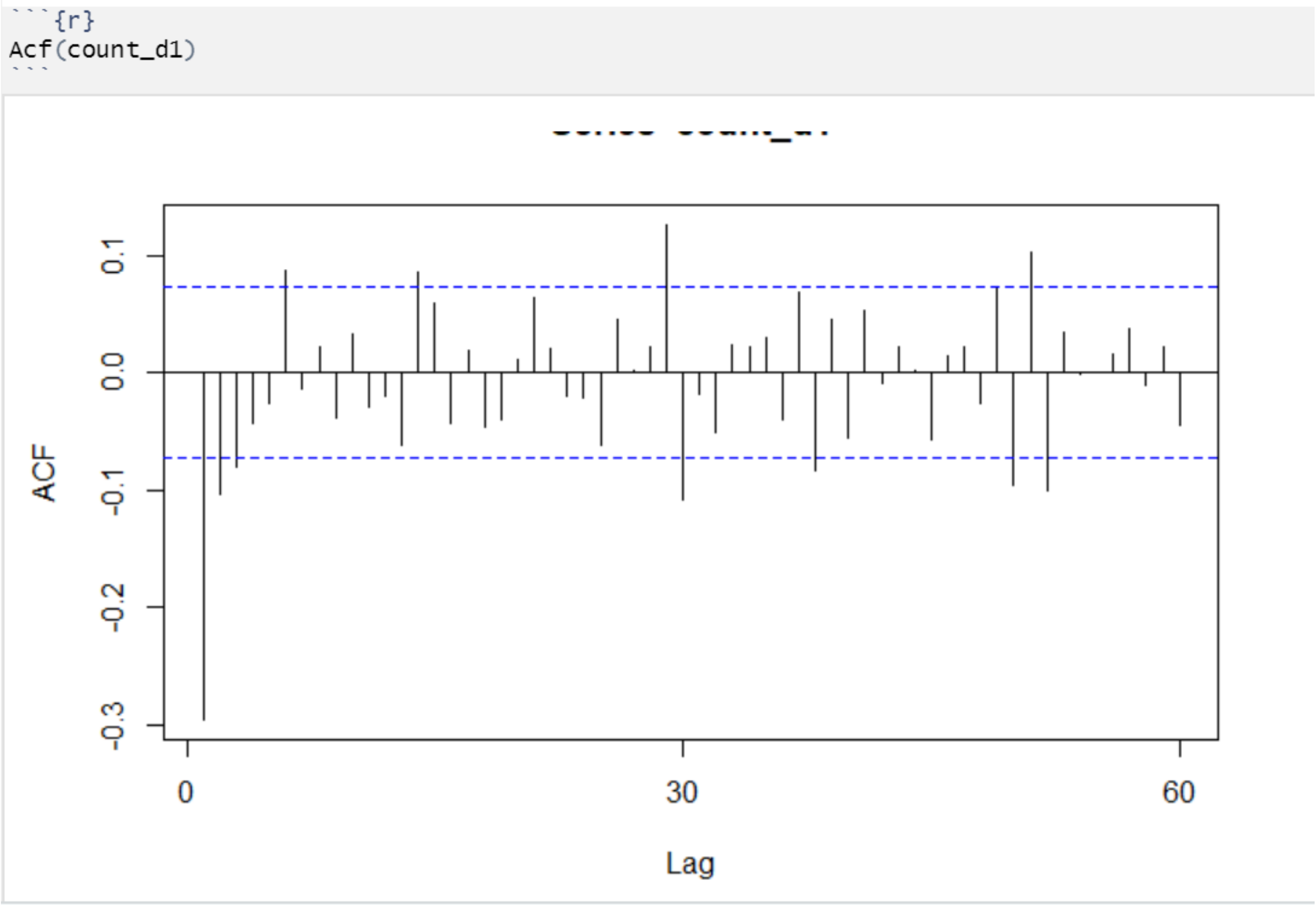
This plot shows the correlation of the series with ***itself*** with different lag functions applied to it.

ACF PLOT: USED TO DETERMINE THE **AR (p)** OF OUR MODEL



To read this plot, we need to see how many of the lines fall outside the range of our 95% confidence intervals (The blue dashed lines).

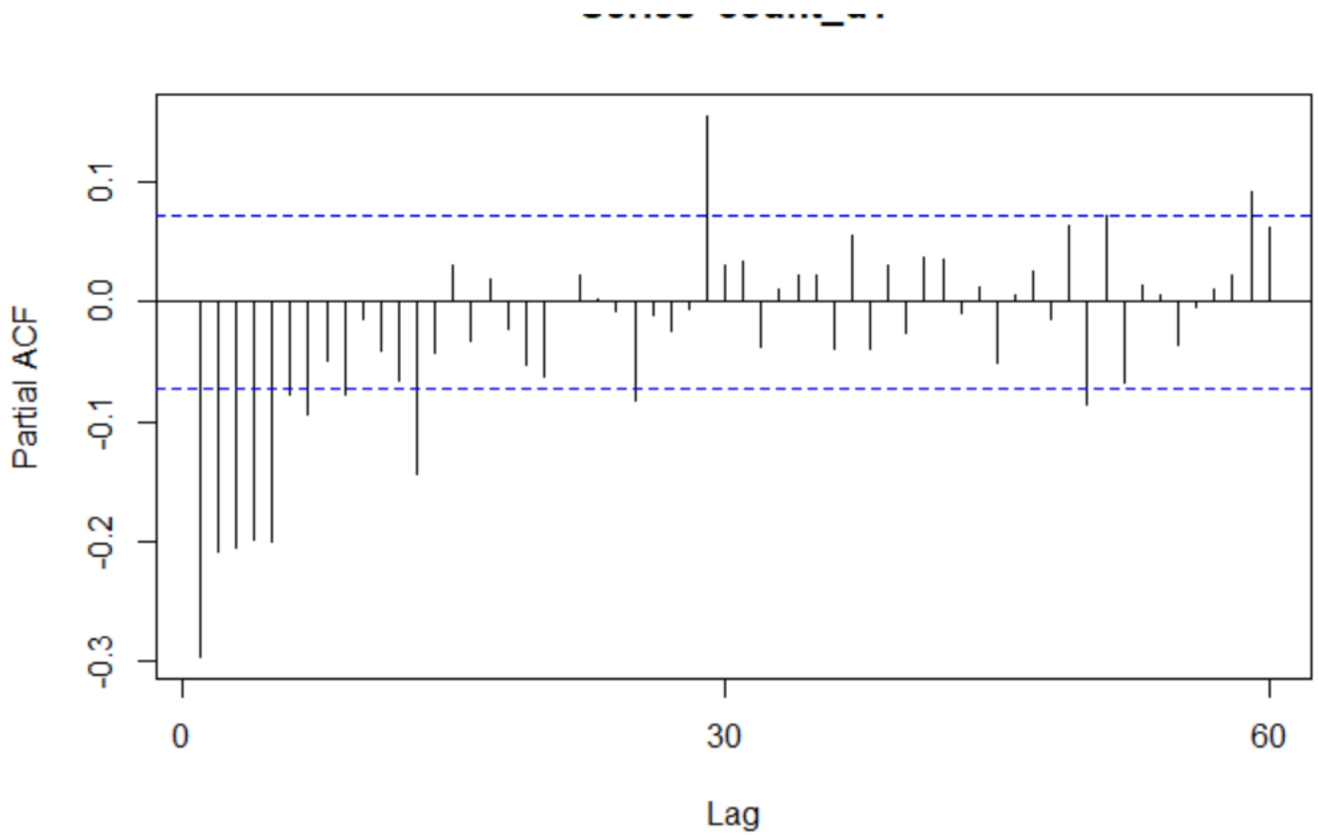
ACF PLOT: USED TO DETERMINE THE **AR (p)** OF OUR MODEL



Lines 1, 2, & 3 are the first immediate lines that fall out of our confidence intervals. However, line 1 seems to be the most significant and it is a sharp decay after line 1. Therefore, we are going to initially set our AR to ***1***. We want parsimony in our model. The less AR terms we set, the better.

PACF PLOT: USED TO DETERMINE THE **MA (q)** OF OUR MODEL

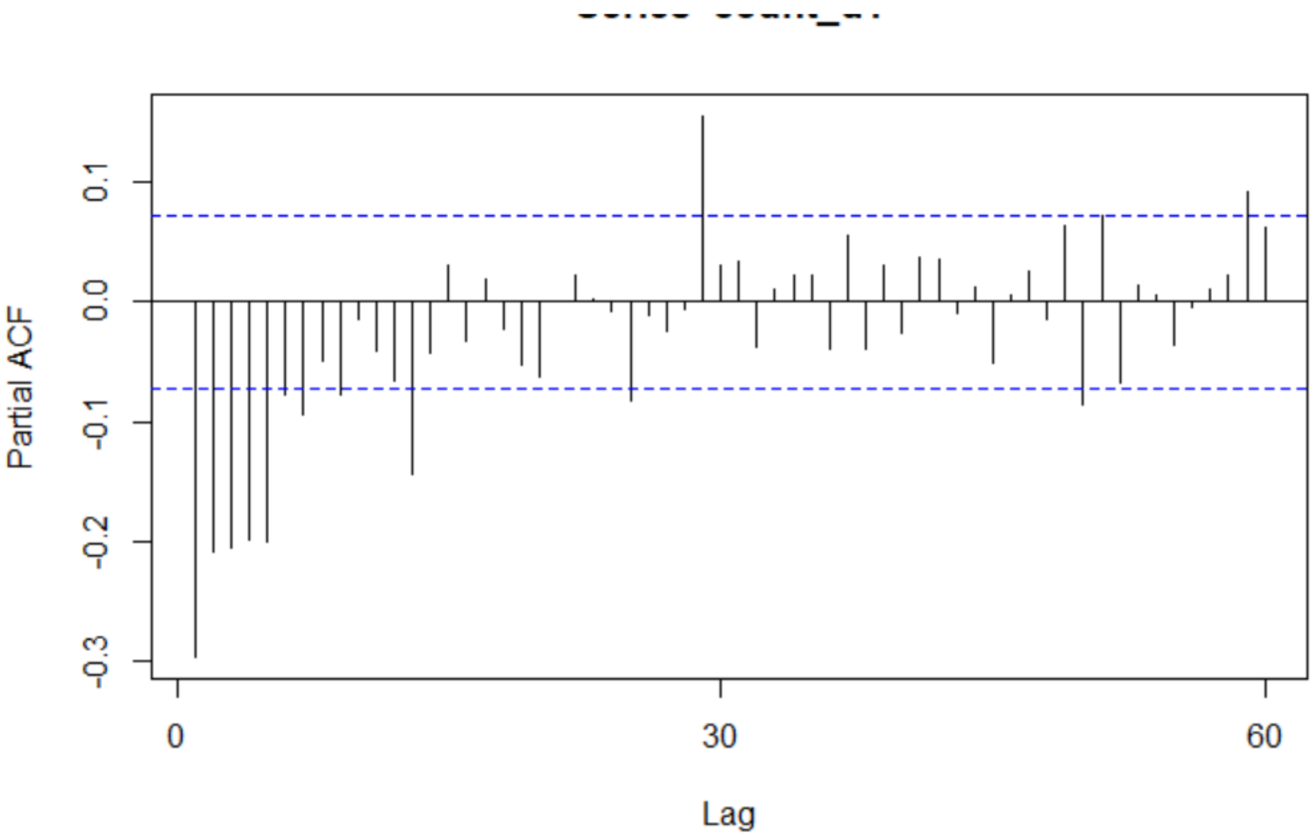
```{r}
Pacf(count_d1)
```



This plot shows the amount of autocorrelation at corresponding "lags" that is not explained by lower-order autocorrelations.

PACF PLOT: USED TO DETERMINE THE **MA (q)** OF OUR MODEL

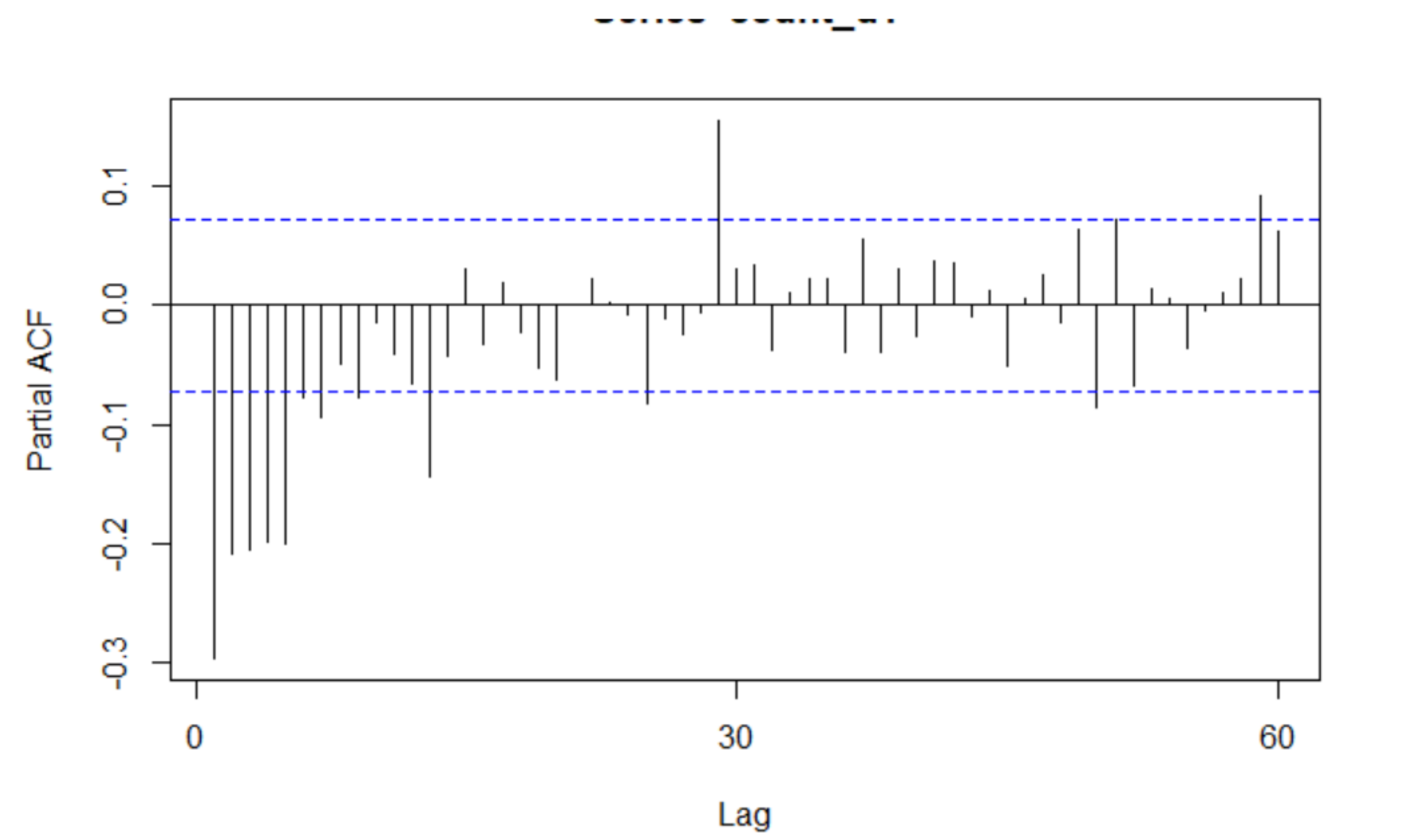
```{r}
Pacf(count_d1)
```



To read this plot, we need to see how many of the lines fall outside the range of our 95% confidence intervals (The blue dashed lines). It is similar to how we read the ACF plot.

PACF PLOT: USED TO DETERMINE THE **MA (q)** OF OUR MODEL

```{r}
Pacf(count_d1)
```



Lines 1, 2, 3, 4, & 5 are the first immediate lines that fall out of our confidence intervals. However, line 1 seems to be the most significant and it is a sharp decay after line 1. Therefore, we are going to initially set our MA to ***1***. We want parsimony in our model. The less MA terms we set, the better.

LET'S MODEL!

```{r}
arima1 <- arima(deseasonal_cnt, c(1,1,1))
summary(arima1)
confint(arima1)
# All confidence intervals are outside the bounds of 0, therefore, this is a good model.
```

```
Call:
arima(x = deseasonal_cnt, order = c(1, 1, 1))

Coefficients:
         ar1      ma1
      0.3527  -0.8887
s.e.  0.0421   0.0189

sigma^2 estimated as 829488:  log likelihood = -6010.72,  aic = 12027.44

Training set error measures:
                   ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set 13.3298 910.1394 648.3641 -9.087824 23.12503 0.8899097 0.01020787
         2.5 %     97.5 %
ar1  0.2700530  0.4352527
ma1 -0.9258615 -0.8515855
```

LET'S MODEL!

p,d,q

```{r}
arima1 <- arima(deseasonal_cnt, c(1,1,1))
summary(arima1)
confint(arima1)
# All confidence intervals are outside the bounds of 0, therefore, th
```

```
Call:
arima(x = deseasonal_cnt, order = c(1, 1, 1))

Coefficients:
         ar1      ma1
      0.3527  -0.8887
s.e.  0.0421   0.0189

sigma^2 estimated as 829488:  log likelihood = -6010.72,  aic = 1202

Training set error measures:
                  ME     RMSE      MAE       MPE     MAPE      MASE
Training set 13.3298 910.1394 648.3641 -9.087824 23.12503 0.8899097 (
          2.5 %     97.5 %
ar1  0.2700530  0.4352527
ma1 -0.9258615 -0.8515855
```

ar1: This is a coefficient to show how quickly the time series model tends to pull back towards the mean. The closer it is to 0, the more it tends to pull towards the mean. It is significant because the confidence intervals do NOT contain zero.

ma1: An indication of the moving average behavior. The closer to zero it is, the less the average moves. It is significant because the confidence intervals do NOT contain zero.

EXPLORATORY, LET'S TRY ANOTHER MODEL: SET THE AR TO TWO TERMS

```
```{r}
arima3 <- arima(deseasonal_cnt, c(2,1,1))
summary(arima3)
confint(arima3)
# The confindence intervals for ar2 contains 0, therefore, the previous model was better. No need to continue further.
```
```

```
Call:
arima(x = deseasonal_cnt, order = c(2, 1, 1))

Coefficients:
         ar1      ar2      ma1
      0.3555  -0.0376  -0.8806
s.e.  0.0428   0.0404   0.0222

sigma^2 estimated as 828497:  log likelihood = -6010.29,  aic = 12028.58

Training set error measures:
                   ME     RMSE      MAE       MPE     MAPE      MASE        ACF1
Training set 12.57936 909.5952 647.0396 -9.105965 23.08264 0.8880917 -0.00303815
         2.5 %      97.5 %
ar1  0.2716133  0.43936971
ar2 -0.1168870  0.04167283
ma1 -0.9240361 -0.83710146
```

The **ar2** 95% CI contains zero. Therefore, it is NOT worth it to keep it in the model.

EXPLORATORY, LET'S TRY ANOTHER MODEL: SET THE MA TO TWO TERMS

```{r}
arima2 <- arima(deseasonal_cnt, c(1,1,2))
summary(arima2)
confint(arima2)
# The confindence intervals for ma2 contains 0, therefore, the previous model was better. No need to continue further.
```

```
Call:
arima(x = deseasonal_cnt, order = c(1, 1, 2))

Coefficients:
         ar1      ma1      ma2
      0.2804  -0.8083  -0.0667
s.e.  0.1058   0.1067   0.0861

sigma^2 estimated as 828781:  log likelihood = -6010.41,  aic = 12028.83

Training set error measures:
                   ME    RMSE      MAE       MPE     MAPE      MASE         ACF1
Training set 12.86361 909.751 647.4526 -9.096764 23.09202 0.8886586 0.0004521082
           2.5 %     97.5 %
ar1   0.07297229  0.4878803
ma1  -1.01738352 -0.5991233
ma2  -0.23541903  0.1020262
```

The **ma2** 95% CI contains zero. Therefore, it is NOT worth it to keep it in the model.

OUR BEST MODEL IS THE FIRST ONE WE RAN!

```{r}
arima1 <- arima(deseasonal_cnt, c(1,1,1))
summary(arima1)
confint(arima1)
# All confidence intervals are outside the bounds of 0, therefore, this is a good model.
```

```
Call:
arima(x = deseasonal_cnt, order = c(1, 1, 1))

Coefficients:
         ar1      ma1
      0.3527  -0.8887
s.e.  0.0421   0.0189

sigma^2 estimated as 829488:  log likelihood = -6010.72,  aic = 12027.44

Training set error measures:
                   ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set 13.3298 910.1394 648.3641 -9.087824 23.12503 0.8899097 0.01020787
         2.5 %     97.5 %
ar1  0.2700530  0.4352527
ma1 -0.9258615 -0.8515855
```

FINAL STEP: USE auto.arima() TO HAVE R CREATE THE MODEL FOR US

```{r}
auto_arima <- auto.arima(deseasonal_cnt, seasonal= FALSE)
summary(auto_arima)
```

```
Series: deseasonal_cnt
ARIMA(1,1,1)

Coefficients:
         ar1      ma1
      0.3527  -0.8887
s.e.  0.0421   0.0189

sigma^2 estimated as 831767:  log likelihood=-6010.72
AIC=12027.44   AICc=12027.48   BIC=12041.22

Training set error measures:
                  ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set 13.3298 910.1394 648.3641 -9.087824 23.12503 0.5589992 0.01020787
```

FINAL STEP: USE auto.arima() TO HAVE R CREATE THE MODEL FOR US

```
```{r}
auto_arima <- auto.arima(deseasonal_cnt, seasonal= FALSE)
summary(auto_arima)
```
```

```
Series: deseasonal_cnt
ARIMA(1,1,1)

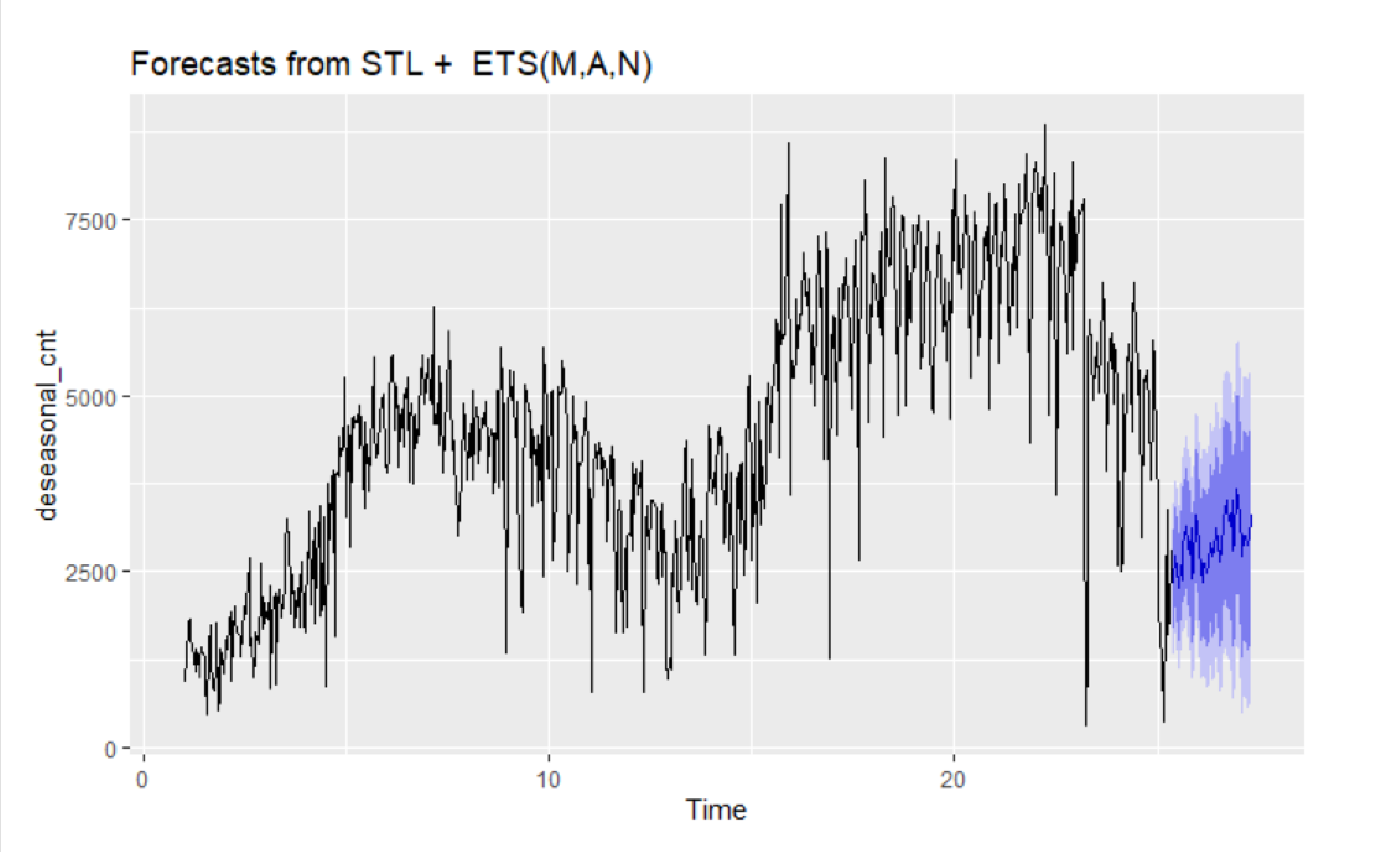
Coefficients:
         ar1      ma1
      0.3527  -0.8887
s.e.  0.0421   0.0189

sigma^2 estimated as 831767:  log likelihood=-6010.72
AIC=12027.44   AICc=12027.48   BIC=12041.22

Training set error measures:
                  ME     RMSE      MAE       MPE     MAPE      MASE       ACF1
Training set 13.3298 910.1394 648.3641 -9.087824 23.12503 0.5589992 0.01020787
```

The auto.arima() function chose the same model we chose (arima(1,1,1)). Therefore, we can have more assurance that our model is the correct one.

```{r}
autoplot(forecast(deseasonal_cnt))
```



PLOT THE MODEL WITH FORECASTING

EXTRA RESOURCES

- https://people.duke.edu/~rnau/Slides_on_ARIMA_models--Robert_Nau.pdf
- https://blogs.oracle.com/datascience/introduction-to-forecasting-with-arima-in-r
- https://datascienceplus.com/time-series-analysis-using-arima-model-in-r/