Step 1: Loading in the Data

```
# Packages used in tutorials
library(MASS)      # boxcox
library(car)       # qqPlot
```

```
## Loading required package: carData
```

```
library(randtests) # runs.test
```

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

```
# library(forecast) # OPTIONAL if you want auto.arima, not required

bike <- read.csv("trips_per_day.csv")
bike$trip_date <- as.Date(bike$trip_date)

str(bike)
```

```
## 'data.frame':    2969 obs. of  2 variables:
##  $ trip_date: Date, format: "2016-01-10" "2016-01-11" ...
##  $ n_trips  : int  2273 3623 2535 2966 2970 2636 4122 3104 1642 4834 ...
```

```
head(bike)
```

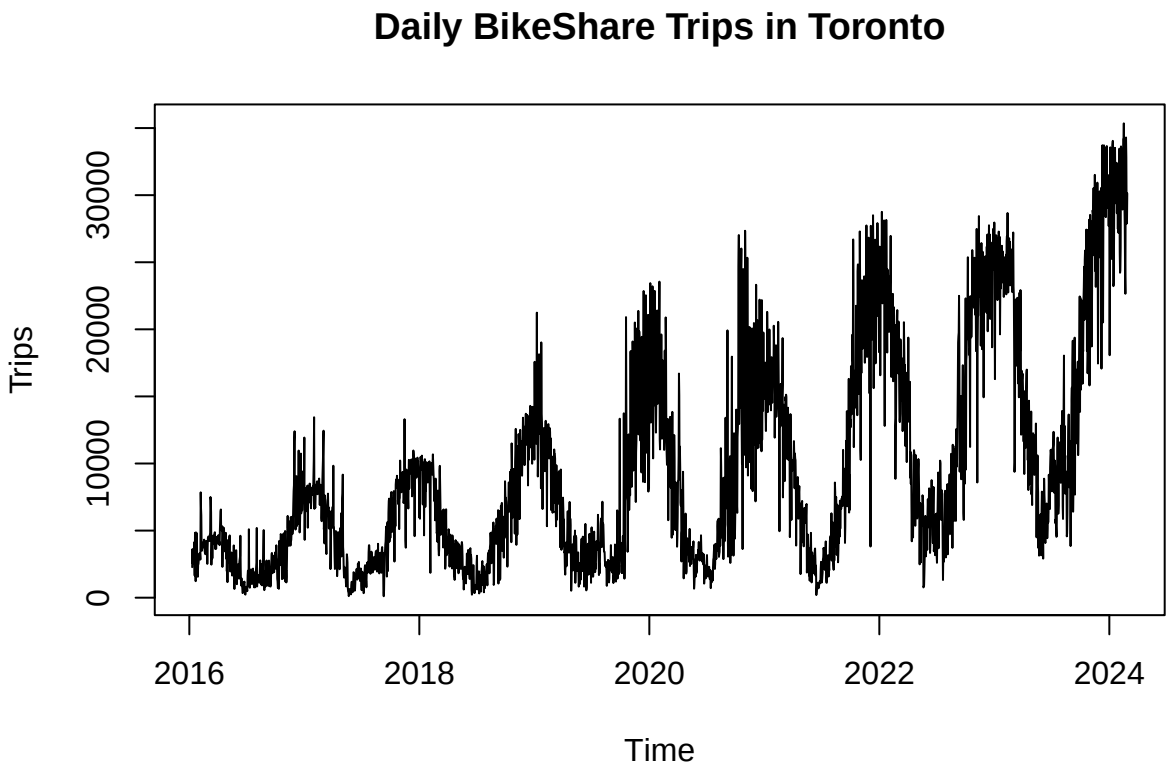
```
##    trip_date n_trips
## 1 2016-01-10    2273
## 2 2016-01-11    3623
## 3 2016-01-12    2535
## 4 2016-02-10    2966
## 5 2016-02-11    2970
## 6 2016-02-12    2636
```

```
range(bike$trip_date)
```

```
## [1] "2016-01-10" "2024-09-30"
```

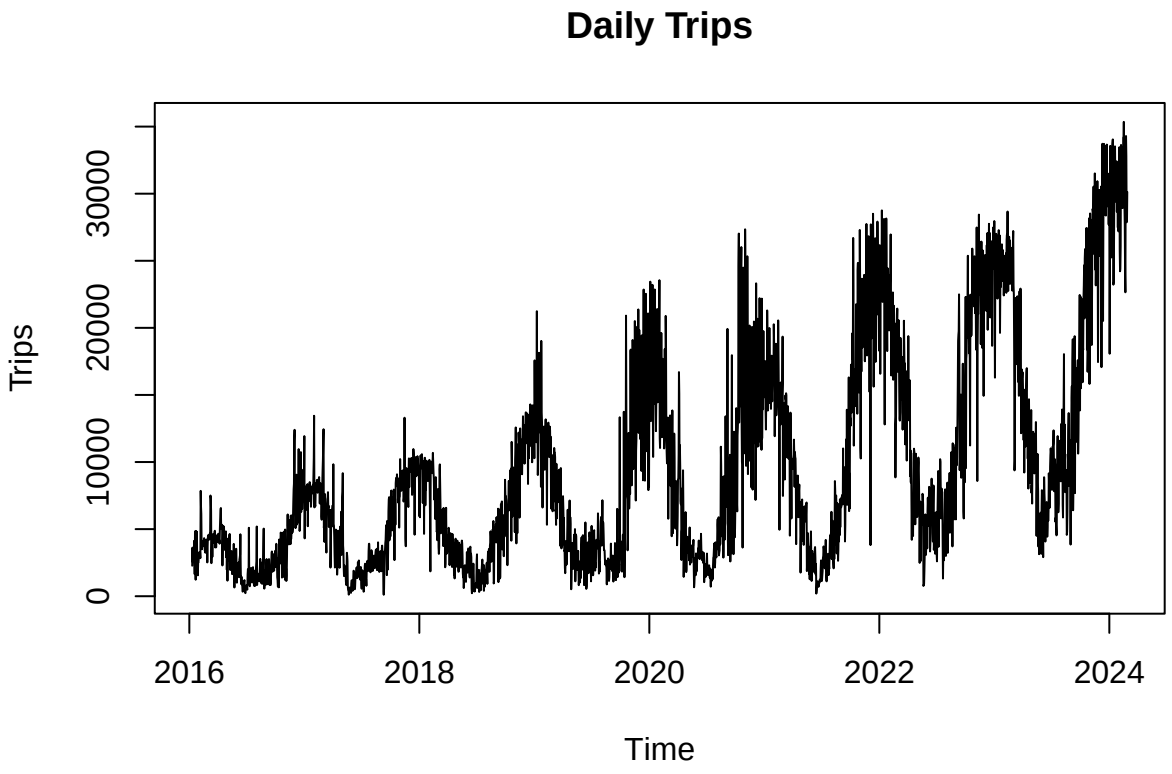
Initial Plotting for Time Series

```
# Sort just in case
bike <- bike[order(bike$trip_date), ]

# Extract response as a vector
y <- bike$n_trips

# Daily frequency with yearly seasonality (approx 365)
bike_ts <- ts(
  y,
  start = c(as.numeric(format(min(bike$trip_date), "%Y")),
            as.numeric(format(min(bike$trip_date), "%j"))),
  frequency = 365
)

plot(bike_ts, main = "Daily BikeShare Trips in Toronto", ylab = "Trips")
```

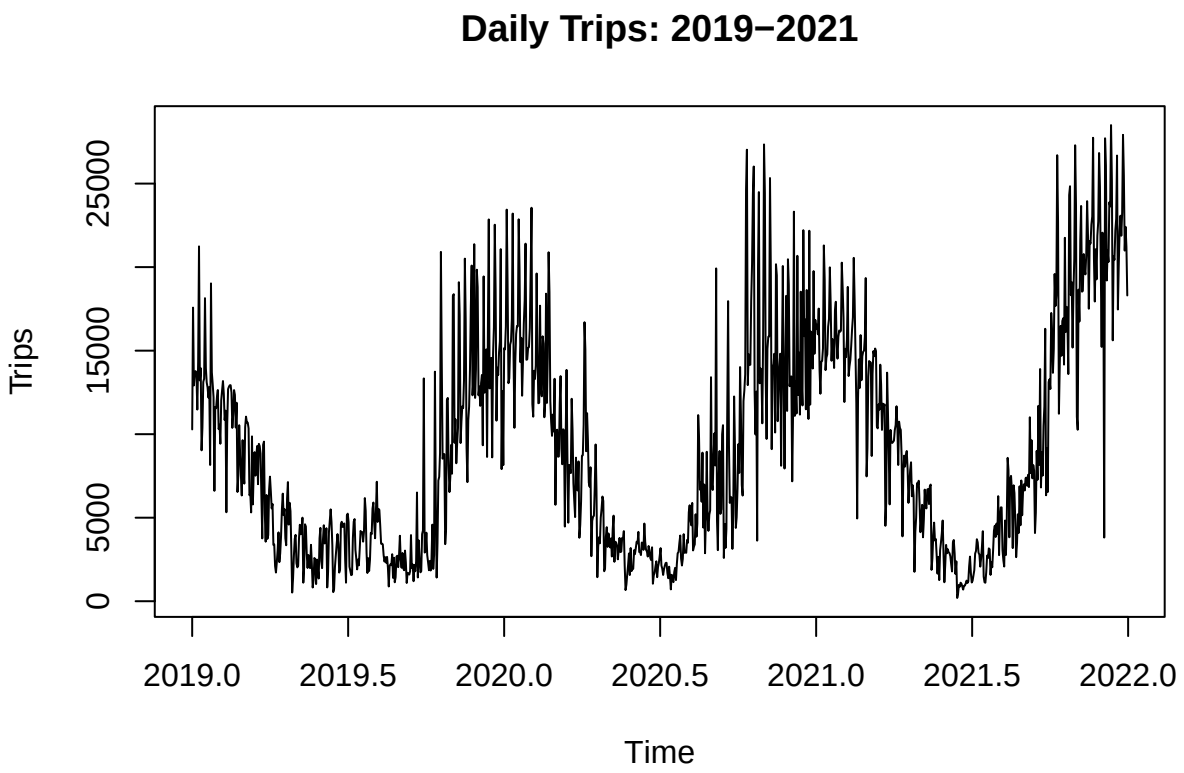


Step 2: EDA

```
par(mfrow = c(1, 1))
plot(bike_ts, main = "Daily Trips", ylab = "Trips")
```

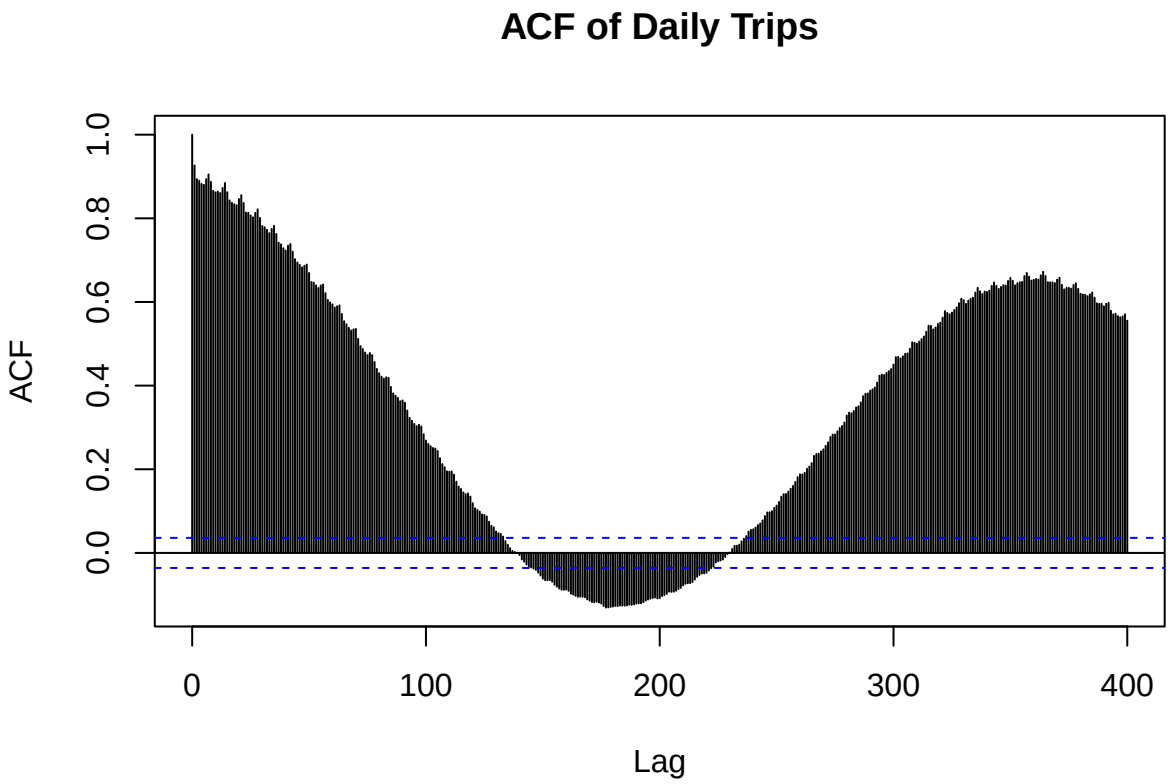


```r
# maybe a zoom on a couple of years
plot(window(bike_ts, start = c(2019, 1), end = c(2021, 365)),
     main = "Daily Trips: 2019-2021", ylab = "Trips")
```

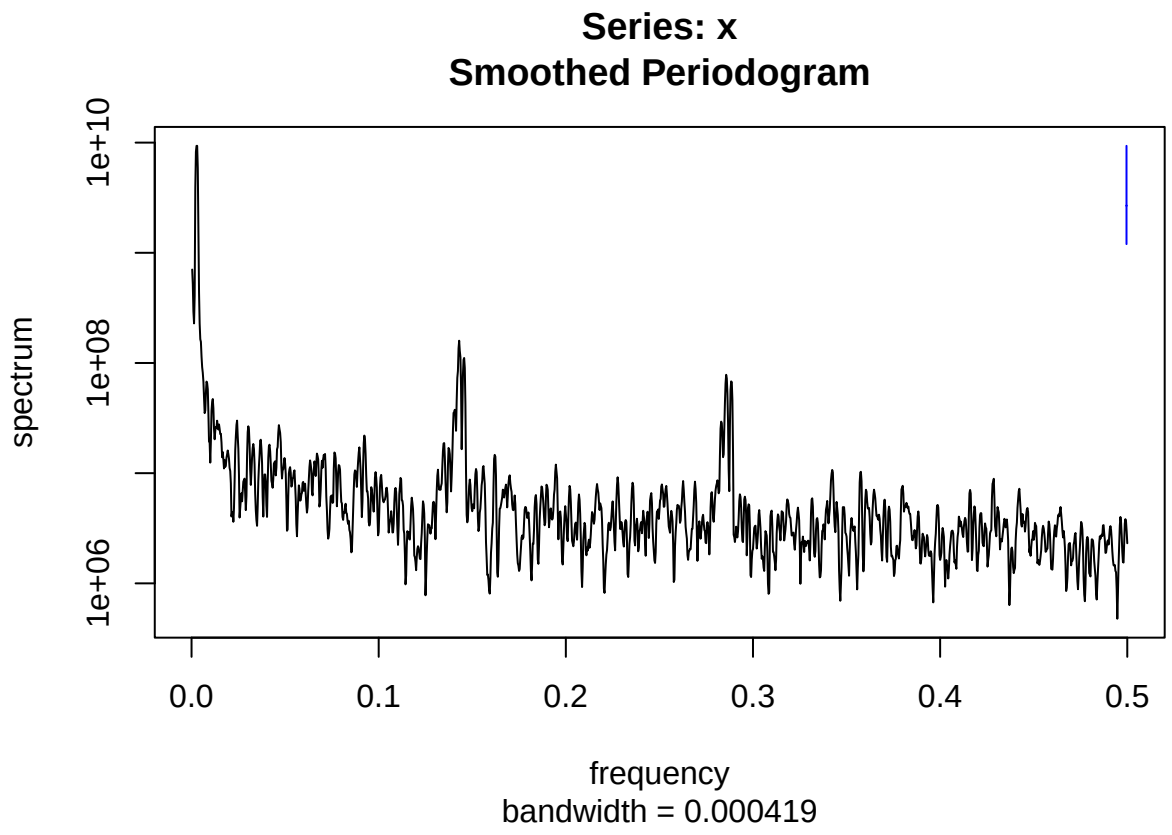


Confirming Seasonality with ACF and Spectrum

```r
acf(as.vector(bike_ts), lag.max = 400,
    main = "ACF of Daily Trips")
```




```r
spec_bike <- spectrum(as.vector(bike_ts), spans = 5)
```

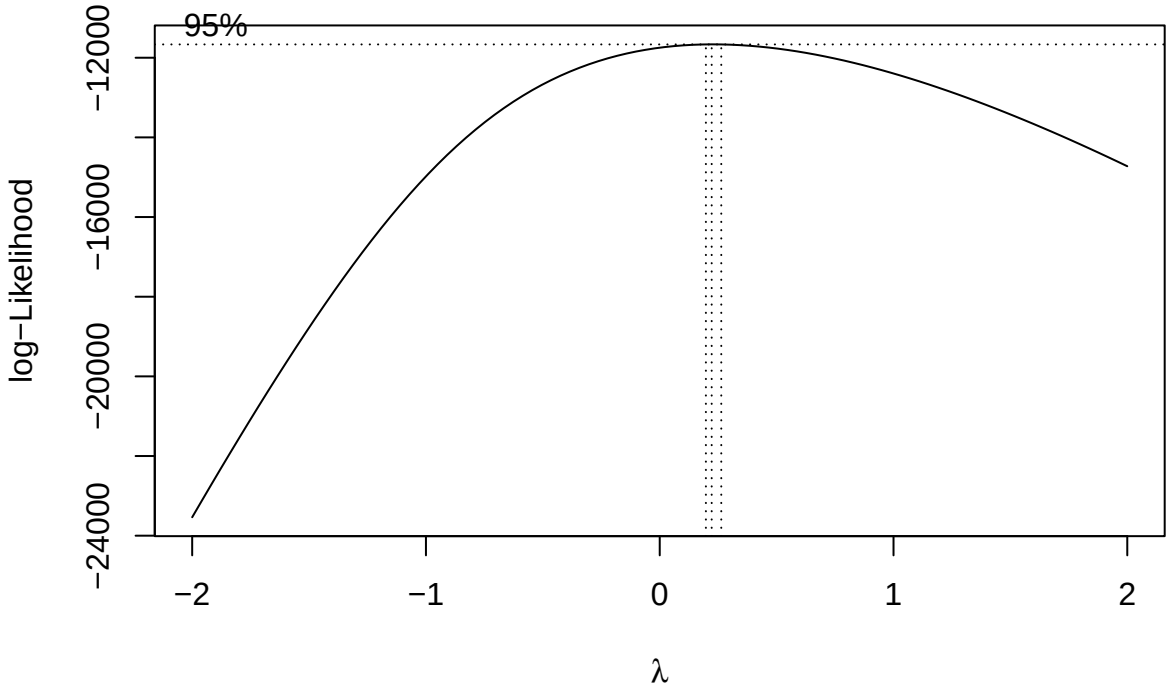


```r
1 / spec_bike$freq[which.max(spec_bike$spec)]  # estimated period
```

```
## [1] 333.3333
```

Step 3: Box-Cox transformation

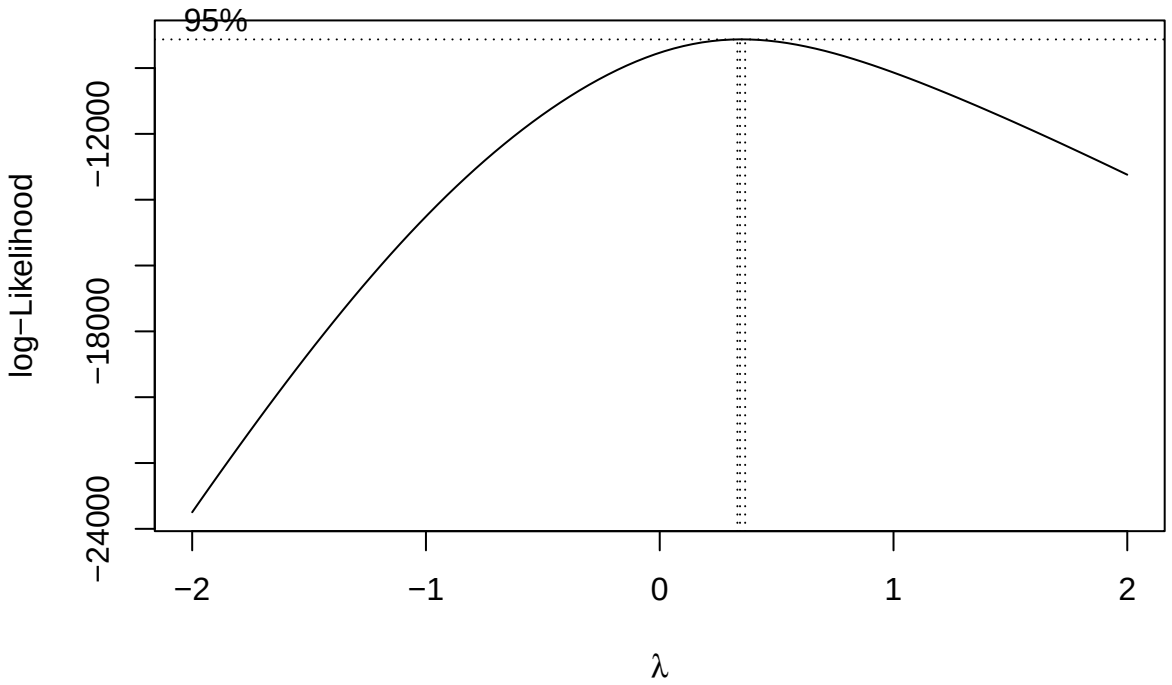
```r
# Simple intercept-only model (like in tutorial)
bc_model_raw <- lm(bike_ts ~ 1)
boxcox_raw   <- MASS::boxcox(bc_model_raw, lambda = seq(-2, 2, 0.1))
```




```r
(lambda_opt_raw <- boxcox_raw$x[which.max(boxcox_raw$y)])
```

```
## [1] 0.2222222
```

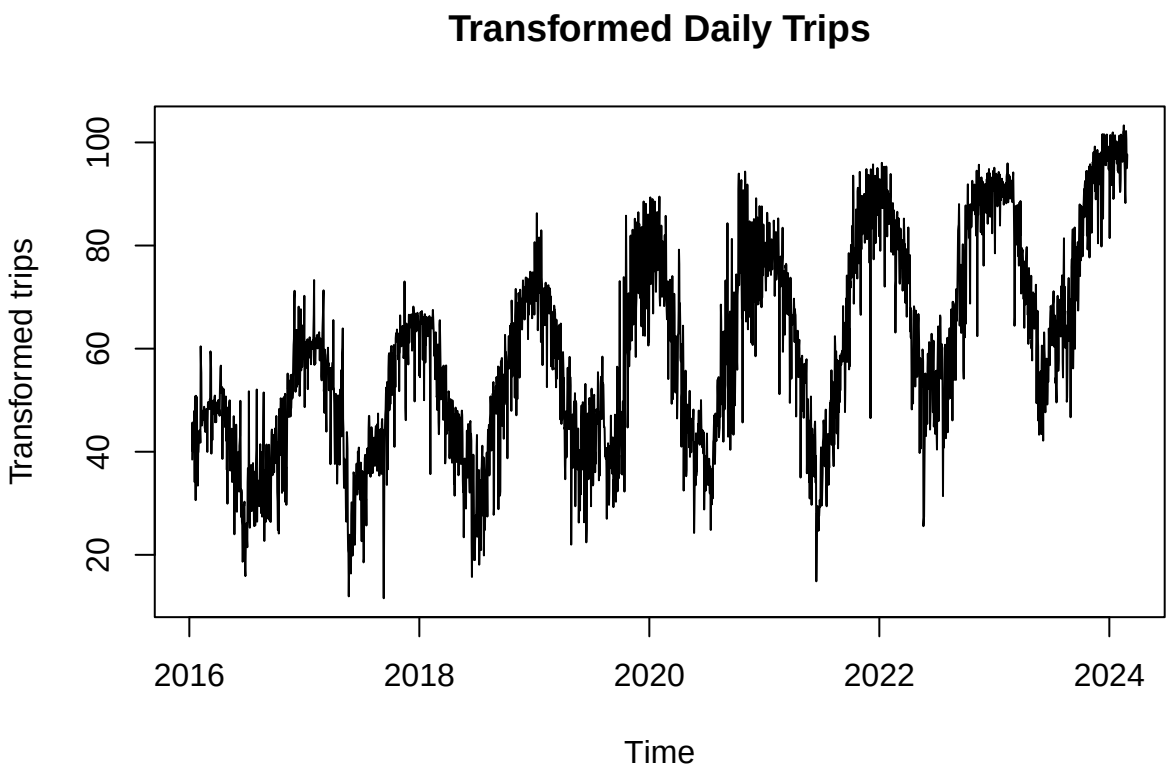
```r
tim <- time(bike_ts)  # continuous time index

# Season: year and day-of-year or month; simplest is month
# Build a monthly factor from dates (instead of cycle, since this is daily)
month <- factor(format(bike$trip_date, "%m"))

reg_for_bc <- lm(bike_ts ~ tim + month)
boxcox_mod <- MASS::boxcox(reg_for_bc, lambda = seq(-2, 2, 0.1))
```




```
(lambda_opt_mod <- boxcox_mod$x[which.max(boxcox_mod$y)])
```

```
## [1] 0.3434343
```

```
lam <- lambda_opt_mod   # keep this for later

if (lam == 0) {
  y_trans <- log(bike_ts)
} else if (lam > 0) {
  y_trans <- (bike_ts^lam - 1) / lam
} else {
  # negative lambda → use minus sign trick like in lectures
  y_trans <- -(bike_ts^lam)
}

plot(y_trans, main = "Transformed Daily Trips", ylab = "Transformed trips")
```


Transformed Daily Trips
Transformed trips
20
40
60
80
100
2016
2018
2020
2022
2024
Time