Home Sales in California: (Median) Time on the Housing Market

Author: Chris Chiang

ABSTRACT/EXECUTIVE SUMMARY

In the housing market, various factors are taken into account when it comes to determining and predicting the sold price of a home, popularity or demand among buyers, and the strength of the real estate market in general. One variable that is extremely crucial is the time that a home sits on the market before it gets an offer and is finally sold. Generally speaking, the prices of homes will increase, but understanding the time series of the time on market will allow us to predict the direction that the housing market is going towards. This project focuses on detached home sales in California, where real estate comes at a premium. Although the complete dataset contains data for every county, I will specifically focus on my aggregation of data for the overall state as a whole. The training dataset that will be used is data from January 1990 - December 2015 to forecast the median time on market for the year 2016 (Jan to Dec) by using a SARIMA model. This will then be compared to the true available data to analyze our models. At the end of this report, the results indicate that the SARIMA $(2,1,0) \times (0,1,1)_{12}$ model is an accurate model to model this time series data.

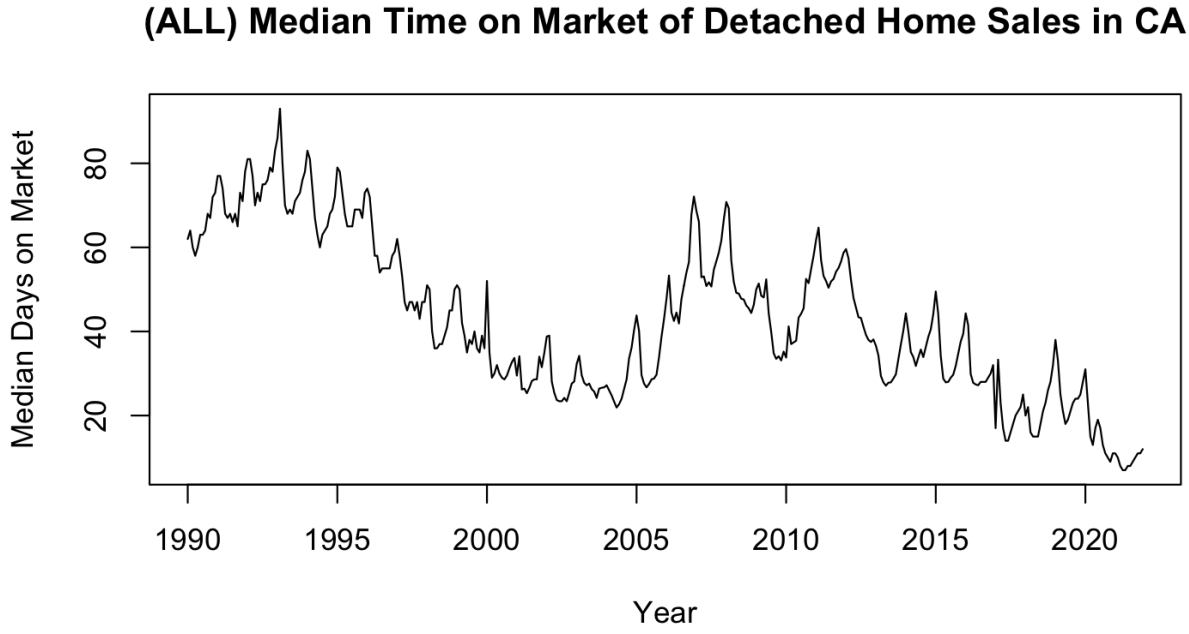
INTRODUCTION

The purpose of this project is to create a SARIMA model to predict future sales for the housing market in California, in regards to the median time that a detached home sits on the market before being sold. I used the R programming language and RStudio to analyze a time series dataset called “Median Time on Market of Existing Detached Homes Historical Data” This data collected was between January 1990 through January 2022 and was created by the California Association of Realtors in a data tab named “Historical Housing Data”. The dataset contains 384 (monthly) data values for each county, as well as for CA, which was the specific analysis that this project is based on. I did not conduct research through 2022 because data from the last few years have been unusually affected by economics from COVID-19. Therefore, deciding to focus on data several years prior will entail a more streamlined and accurate picture for future data, assuming no worldwide event or catastrophe comes into play. California is the most populous state, and real estate is arguably the priciest and in the most demand within the country. Looking at the time it takes for a home to finally receive an offer and sell, helps in forecasting future demand and direction without directly looking at actual home prices since it will certainly increase.

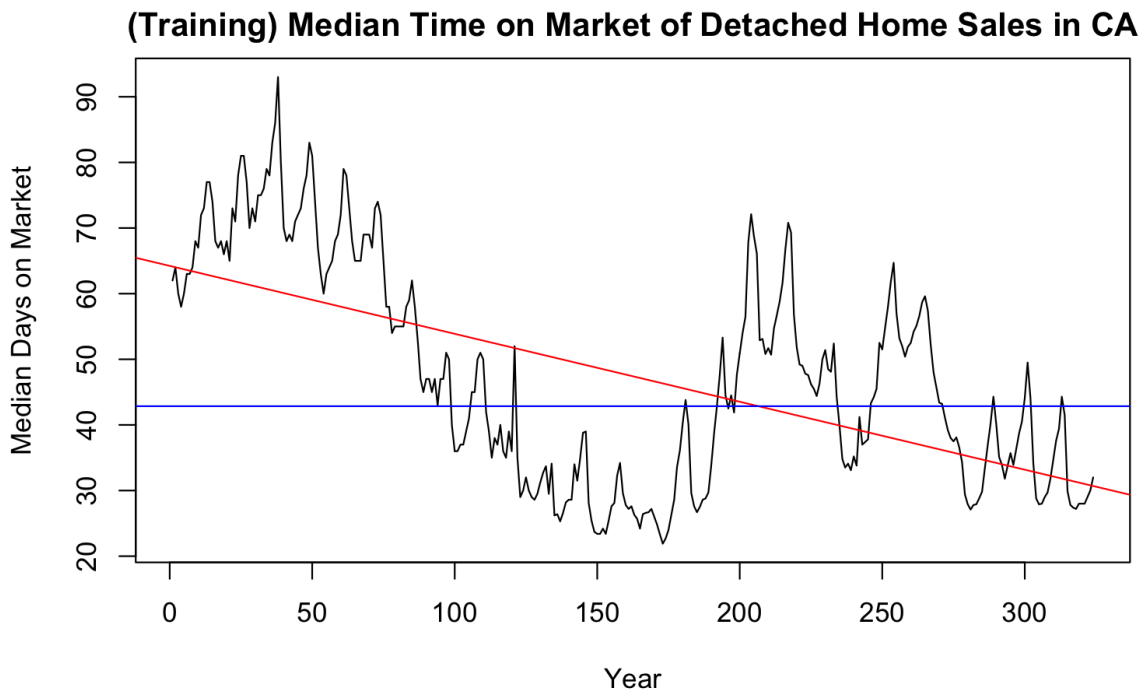
To begin with, the data was partitioned from the monthly sales data into a training set (Jan 1990 - Dec 2015) and a test set (Jan 2016- Dec 2016) to compare forecasted values with actual values to gauge model accuracy. Through the visualizations and analysis of time series plots, ACF and PACF plots, histograms, box-cox and logarithmic transformations of the data, a decomposition series and finally, differencing to remove any trends or seasonality, several SARIMA models were subsequently tested. Following this, a variety of methods to perform diagnostic tests were conducted for the model, as well as residual analysis to check for model accuracy. In deciding on, constructing and testing the final fitted model, the model will then forecast values for January 2016 - December 2016 in the test set. Analyzing the forecasted values in conjunction with the actual observed data from January 2016 -December 2016, the results indicate that SARIMA $(2,1,0) \times (0,1,1)_{12}$ is an accurate model that is suitable to use with forecasting time on market of detached home sales in California.

ANALYSIS

PART 1: Time Series Plots



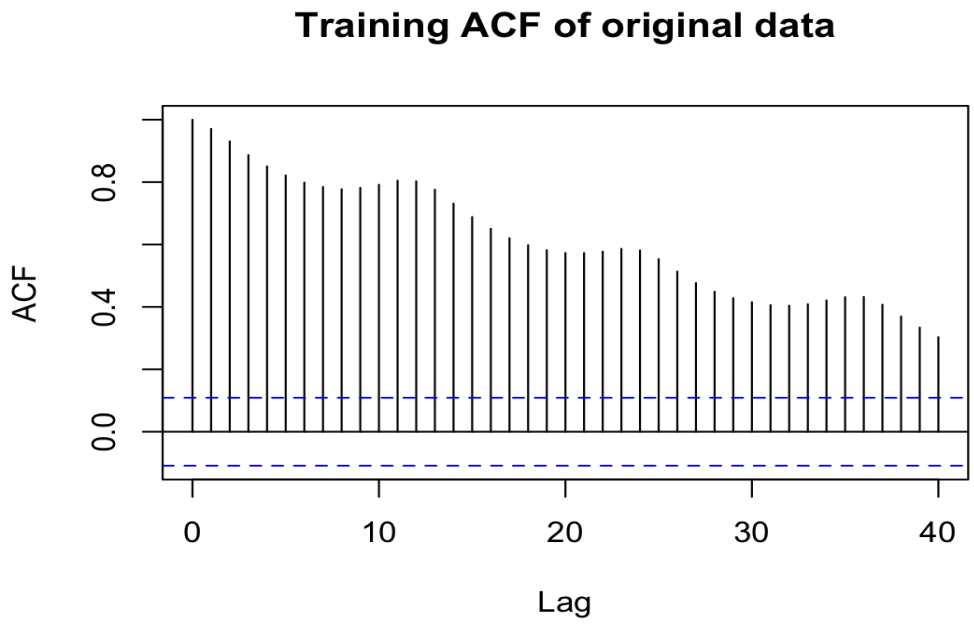
When looking at the complete time series, there are some obvious trends that can be observed. There is seasonality in the months of the year, as well as cycles of increasing and decreasing within. There was a sharp change in the behavior over time: values from about 2000-2005 dropped very low from before and then sharply increased back again after, likely explained by the recession.

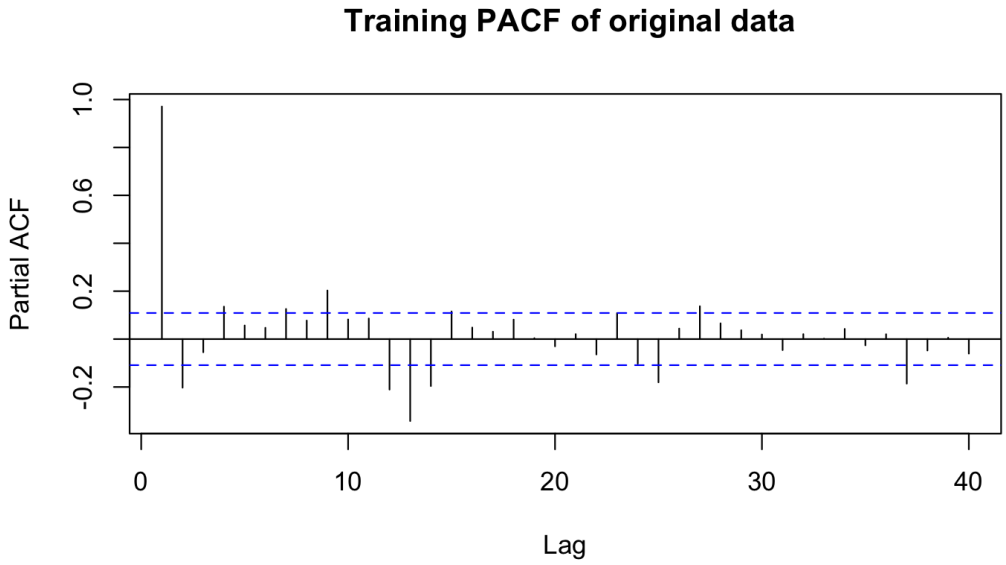


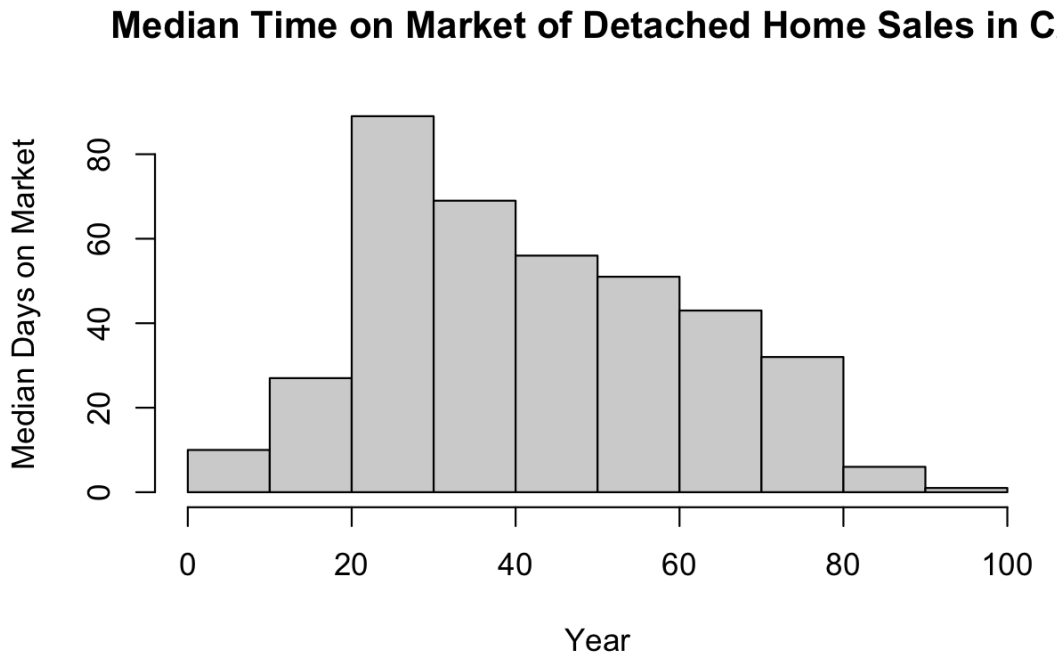
(blue line = mean)
(red line = fit)

This is the time series plot for the training data (1990-2015) which will be used to forecast data from the year 2016. Looking at the fit lines, the median time on market has decreased slightly over time, which suggests an overall increased demand in housing. The data itself does not seem normal or stationary, so performing and comparing several transformations is essential in deriving a model.

PART 2: ACF's and Histogram


Training ACF of original data
ACF
0.0
0.4
0.8
0
10
20
30
40
Lag

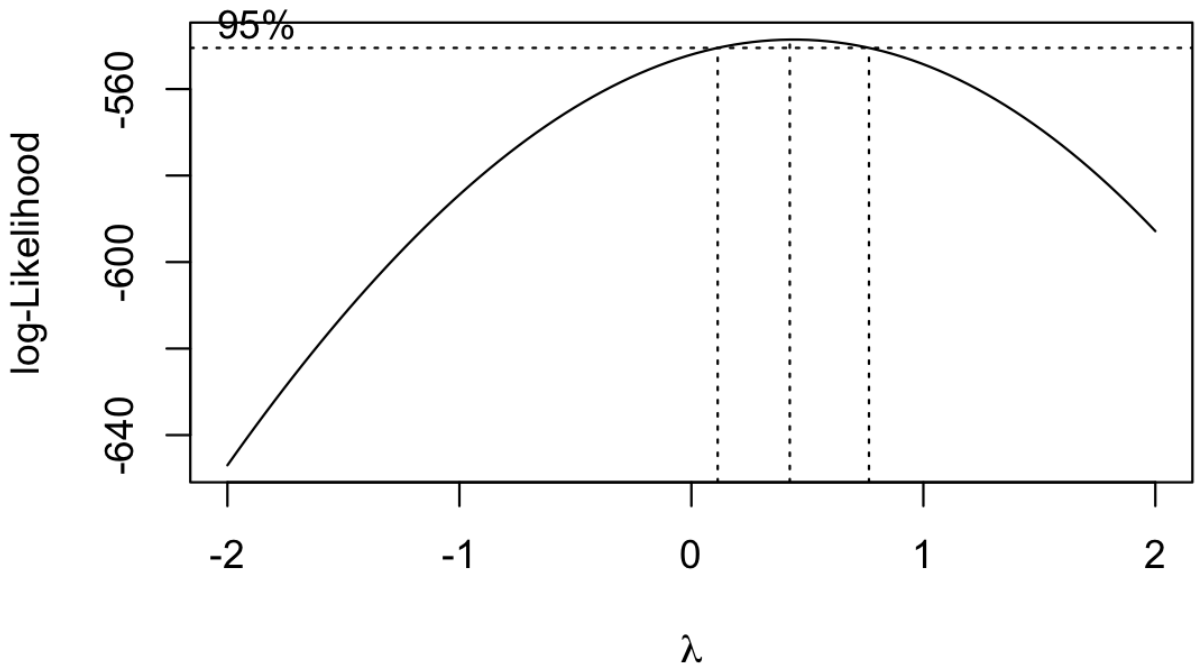

Training PACF of original data
Partial ACF
-0.2
0.2
0.6
1.0
0
10
20
30
40
Lag



The ACF shows clearly that there is seasonality and a decaying trend. Lags at 12, 24, 36, etc. are spiked outside the CI (Confidence Interval, depicted by the blue vertical dashes) shown in the PACF plot. The histogram also shows the data is skewed to the right. With all of this put together, the initial assumptions were right: the data needs to be transformed. Future transformation(s) should be conducted in order to stabilize variance, and differencing should be performed to address the observable trends and seasonality.

PART 3: Transformations

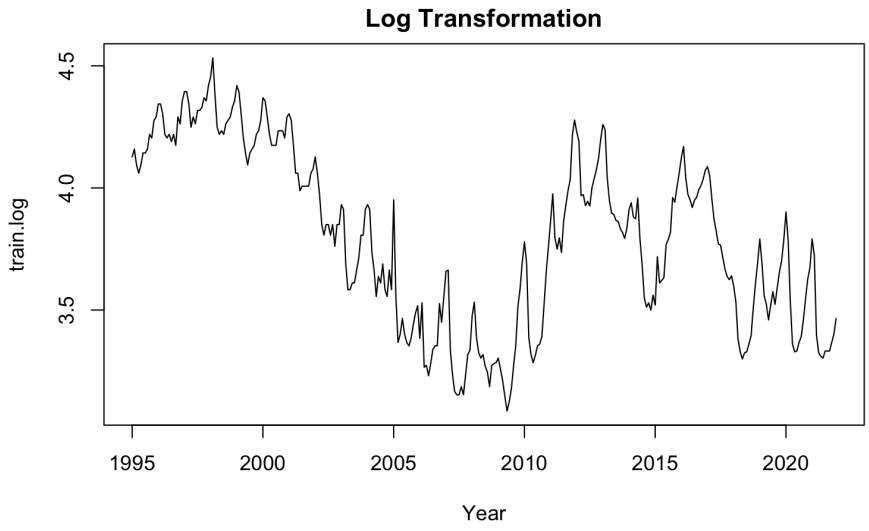
a) Box-Cox Transformation

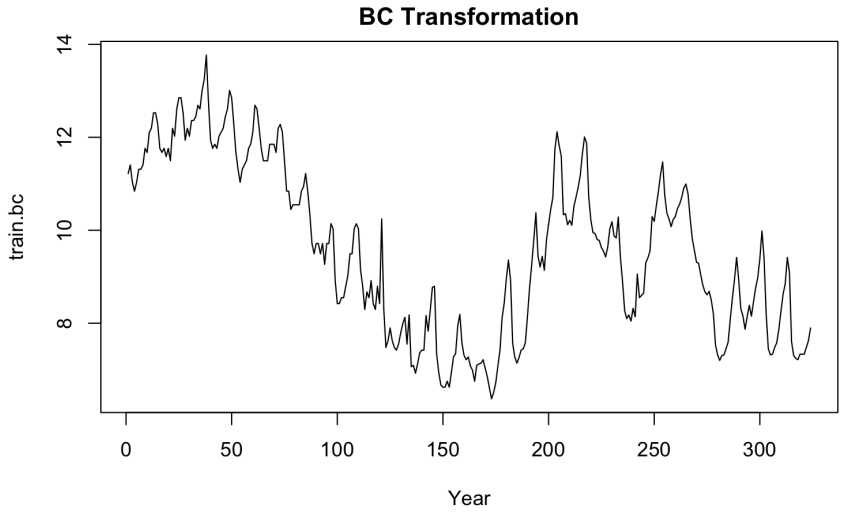


bcTransform$x[which(bcTransform$y == max(bcTransform$y))]

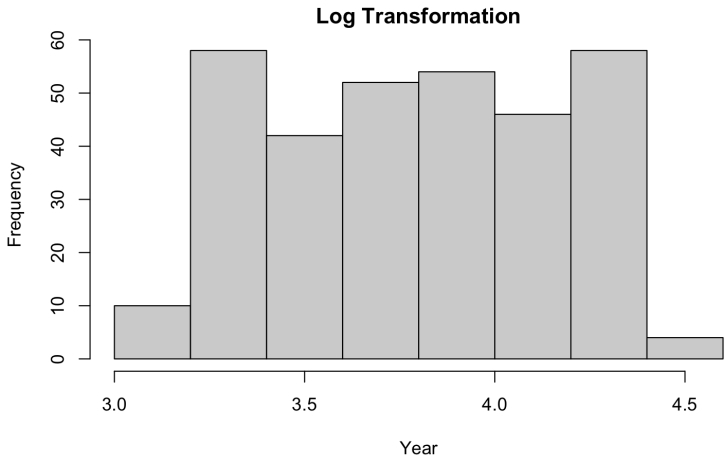
First, a box-cox transformation is performed and the log-likelihood is plotted. Using this function, I obtained a lambda value of 0.424, which is somewhat close to 0. Based on these results, I have decided to compare a log transformation versus a box-cox transformation.

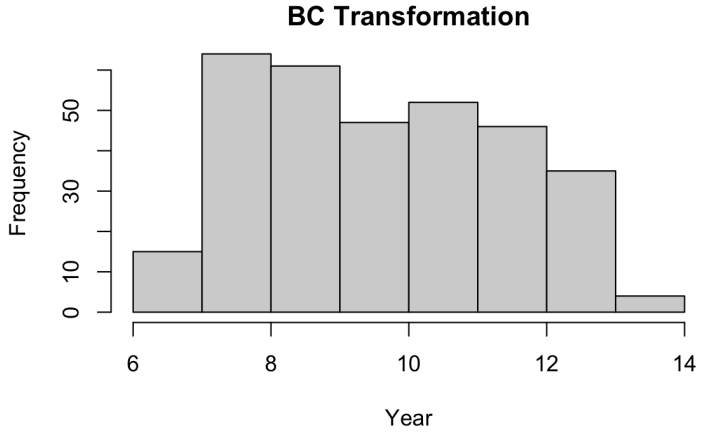
b) Logarithmic versus Box-Cox transformations





Comparing the two time series plots, the Log transformation has a much smaller variance than the Box-Cox transformed data, when looking at their y-axis values. The mean also appears to be smaller in the Log transformed data. Although the log transformation looks like the best transformation to use, additionally comparing their histograms will achieve a more thorough analysis.





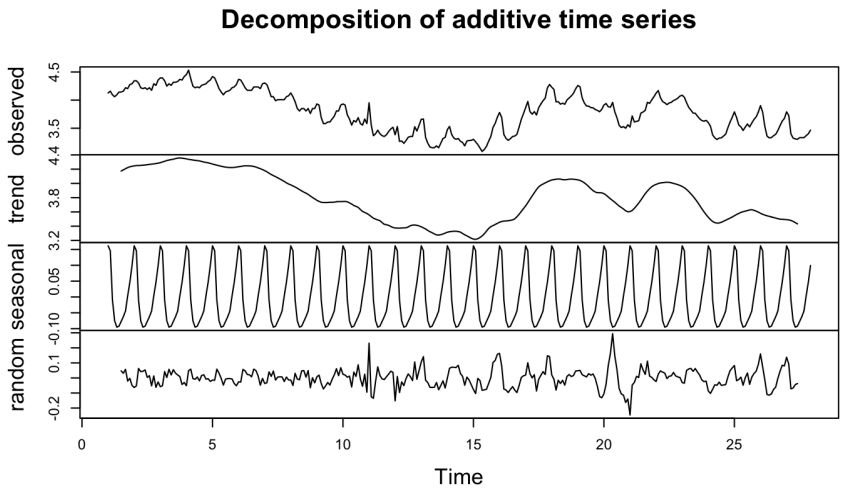
Comparing their histograms, it is hard to visualize their differences at first, but the log transformation seems to be the most consistent. Looking at their variances:

var(train.log)
0.1295526

var(train.bc)
3.291401

Quantitatively, we now know that the log transformation is the best transformation to use since it has the smaller variance. This decision is also previously supported by comparing each of the transformations' time series plots as well as their histograms.

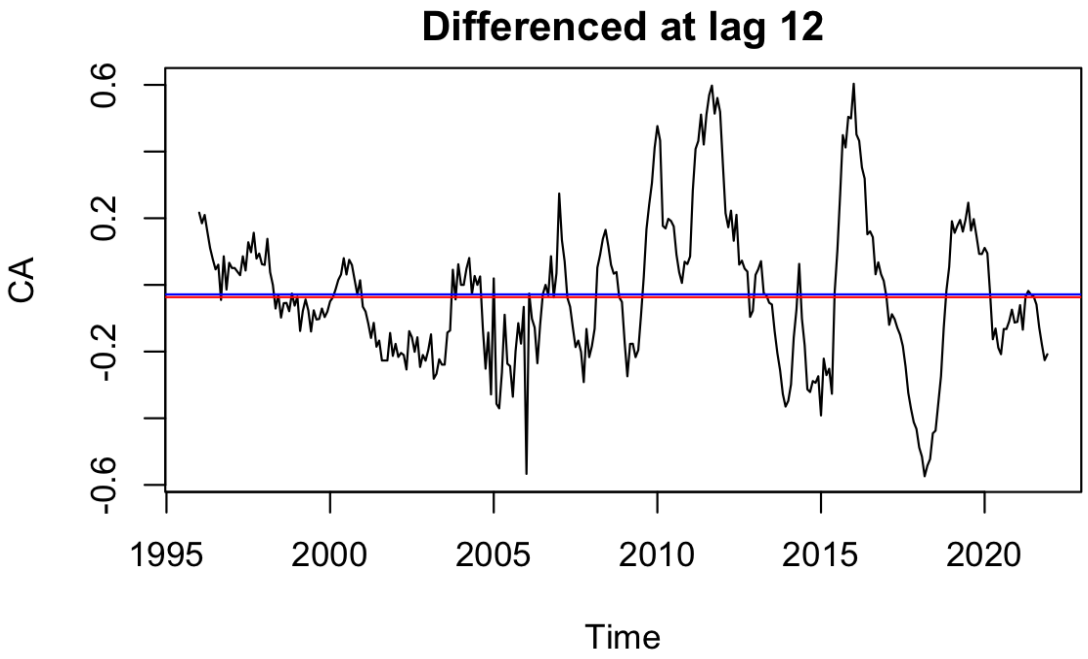
PART 4: Decomposition



Graphing the decomposition, I can further analyze what needs to be done in terms of making the series more stationary. The series looks random. However, there is clear seasonality, for every 1-year period (12 months) so it has to be addressed.

PART 5: Differencing and Lags

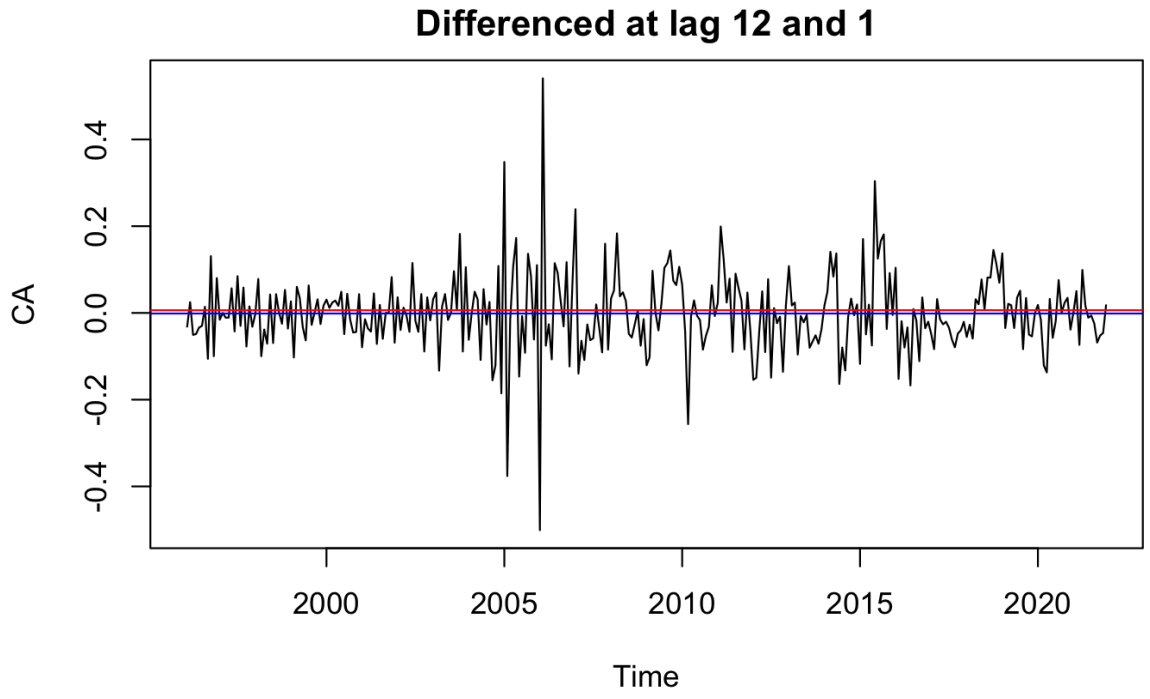
Based on the decomposition analysis, I observed that there is seasonality. I have decided to, then, use differencing at lag 12 to remove it.



Variance after differencing at lag 12:
0.04697876

Mean after differencing at lag 12:
-0.02848543

After differencing at lag 12, it looks like there isn't any more seasonality. I also calculated the variance after this (0.04697876), which is much smaller than the variance after simply doing the log transformation (0.1295526). The mean is also very close to 0 (-0.02848543).



Variance after differencing at lag 12 and 1:
0.008587457

Mean after differencing at lag 12 and 1:
-0.001365598

After differencing at lag 12 AND lag 1, it looks like there isn't any more seasonality or trends. I also calculated the variance after this (0.008587457), which is much smaller than the variance after only differencing at lag 12 (0.04697876). The mean is also closer to 0 (-0.001365598) than only differencing at lag 12 (-0.02848543). No more differencing needs to be done.

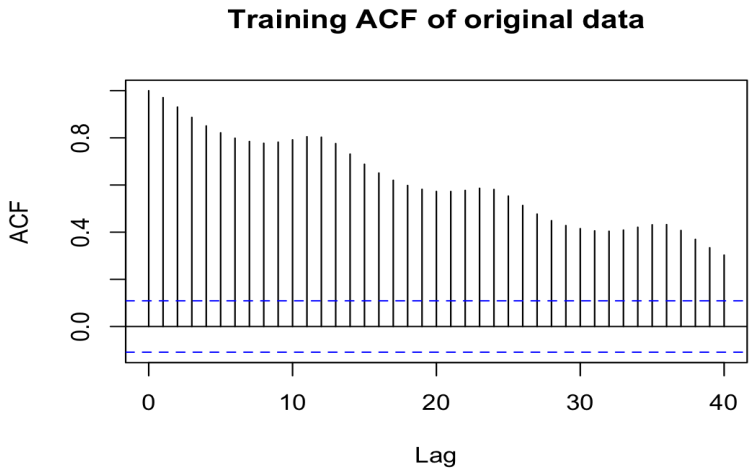
Now, after using the log transformation and differencing at lag 12 and lag 1, the data looks stationary!

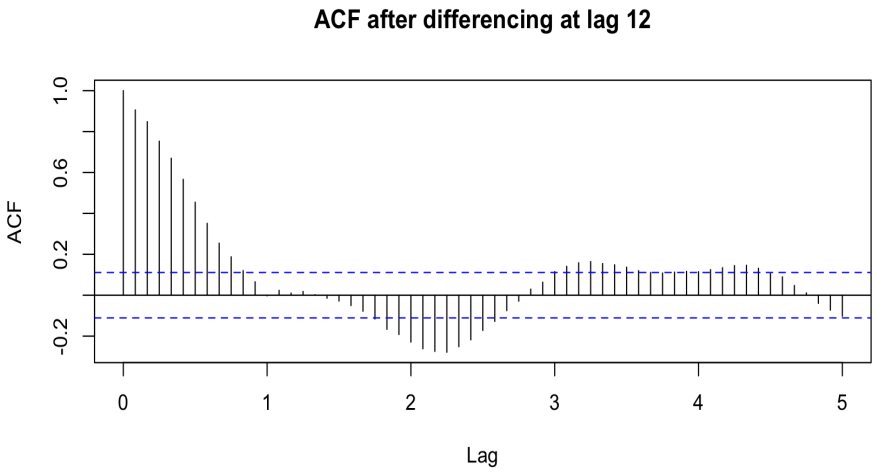
Modeling

PART 1: ACF/PACF and Model Estimations

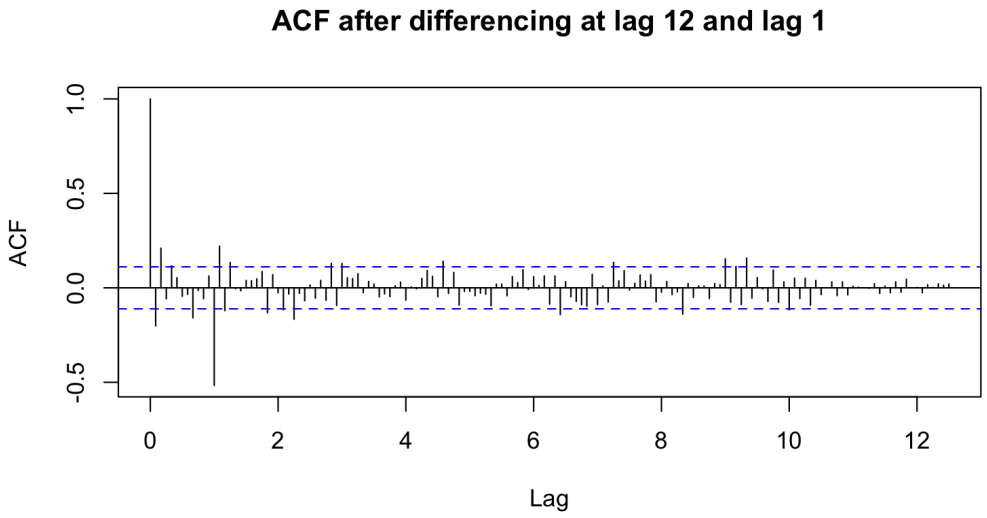
For model estimation variables, I'm using D=1 for differencing at lag=12 for seasonality, and d=1 for differencing at lag 1. s=12 is given, based on the dataset. Looking at the ACF and PACF can help determine the other variables to be used.

a) ACF's





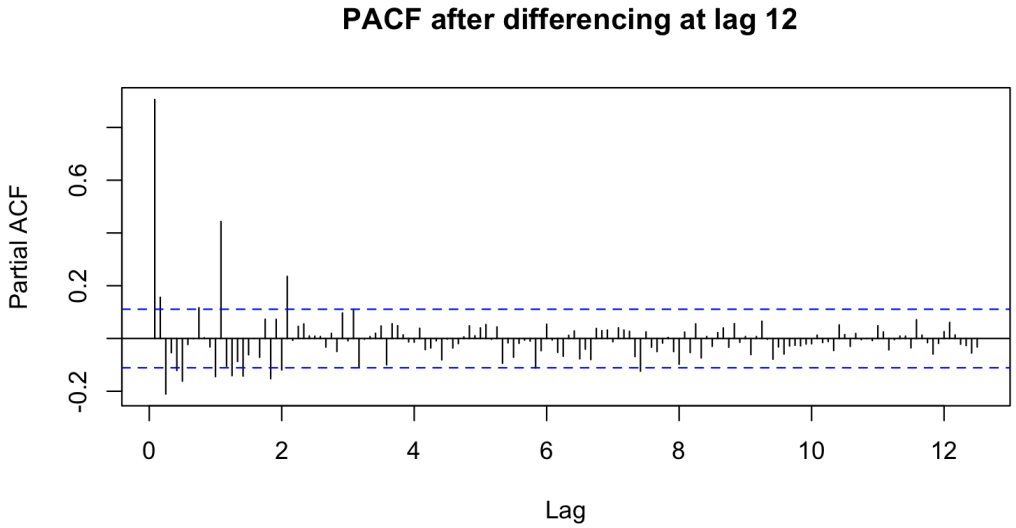
Looking closely at the ACF after differencing at lag 12, there are fewer lines going outside of the CI but there is still clearly seasonality and a trend. We should now perform differencing at lags 12 and 1 to address these retaining issues.

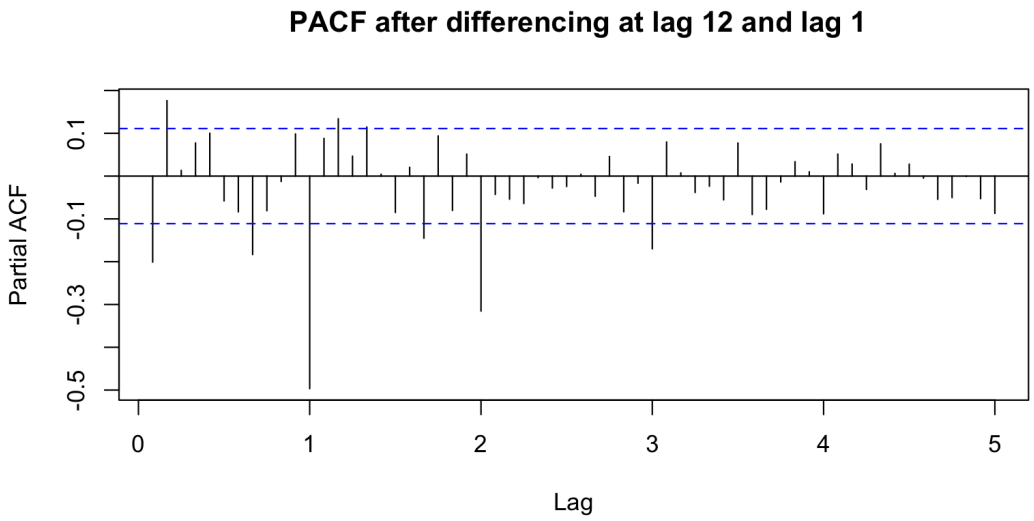


** Each 1 lag is representative of 12 (months)**

Looking closely at the ACF, I have determined the use of Q=1 since the spikes cut off after lag 1.

b) PACF's

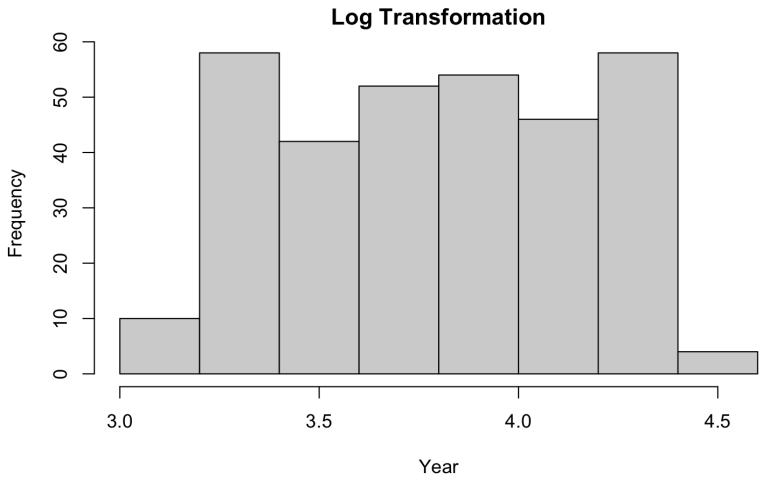


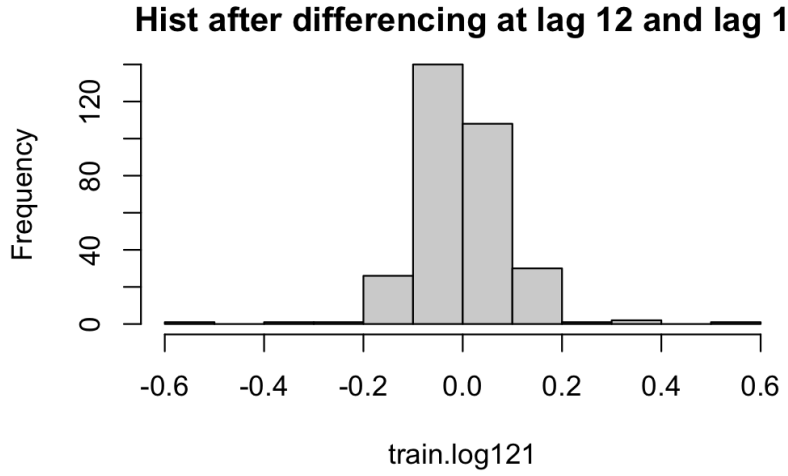


** Each 1 lag is representative of 12 (months)**

Looking at the PACF after differencing for lag 12 and lag 1, there is no evidence of seasonality or trend. There are also fewer spikes outside of the CI, compared with the original. I will decide between The PACF suggests AR(3). P=0, 1, 2, 3 since the lags stop after lag 3, meaning it should be 3 or less. Because there is a first spike at lag 0, q=0 will be used.

c) Histograms





After differencing at lags 12 and 1, the histogram visually depicts the mean being significantly closer to 0 and with very small variance.

PART 2: Comparing Fit Models

est=ar(train.log, aic = TRUE, order.max = 3, method = c("yule-walker"))
est$ar

[1] 1.14664329 -0.10528151 -0.08125792

Based on these parameter estimations, the fit suggests the use of P=3. However, we will still test more models. Considering all analysis done and ACF/PACF charts, the variables I decided on for SARIMA (p,d,q) (P,D,Q)s estimation are as follows:

p= 2 or 3	d=1	q=0	P= 0,1,2,3	D=1	Q=1	s=12

I have decided to compare 3 following models:

SARIMA (3,1,0) x $(0,1,1)_{12}$

SARIMA (3,1,0) x $(1,1,1)_{12}$

SARIMA (2,1,0) x $(0,1,1)_{12}$

```
Call:
arima(x = train.log, order = c(3, 1, 0), seasonal = list(order = c(0, 1, 1),
    period = 12), method = "ML")

Coefficients:
          ar1     ar2     ar3     sma1
```

```
     -0.0868  0.1774  0.0227  -1.0000
s.e.   0.0566  0.0561  0.0567   0.1046

sigma^2 estimated as 0.003986:  log likelihood = 398.06,  aic = -786.11
```

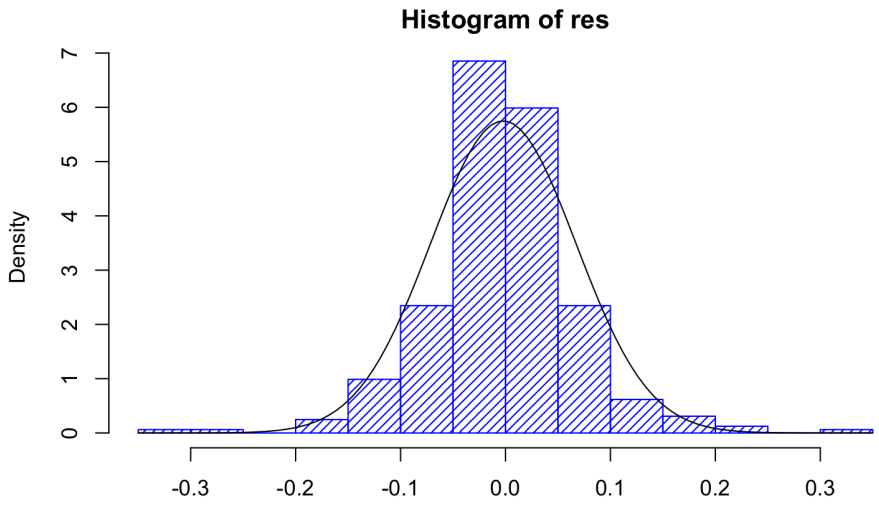
```
Call:
arima(x = train.log, order = c(3, 1, 0), seasonal = list(order = c(1, 1, 1),
    period = 12), method = "ML")

Coefficients:
         ar1     ar2     ar3    sar1     sma1
     -0.0877  0.1774  0.0225  0.0054  -0.9999
s.e.   0.0574  0.0561  0.0567  0.0592   0.0986

sigma^2 estimated as 0.003988:  log likelihood = 398.06,  aic = -787.12
```

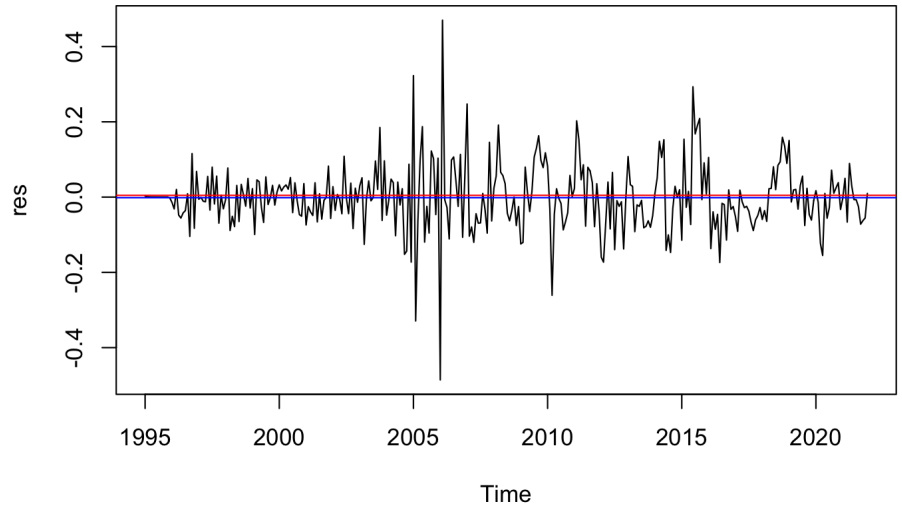
```
Call:
arima(x = train.log, order = c(2, 1, 0), seasonal = list(order = c(0, 1, 1),
    period = 12), method = "ML")

Coefficients:
         ar1     ar2     sma1
     -0.0828  0.1753  -0.9999
s.e.   0.0558  0.0559   0.1044

sigma^2 estimated as 0.003988:  log likelihood = 397.98,  aic = -784.95
```

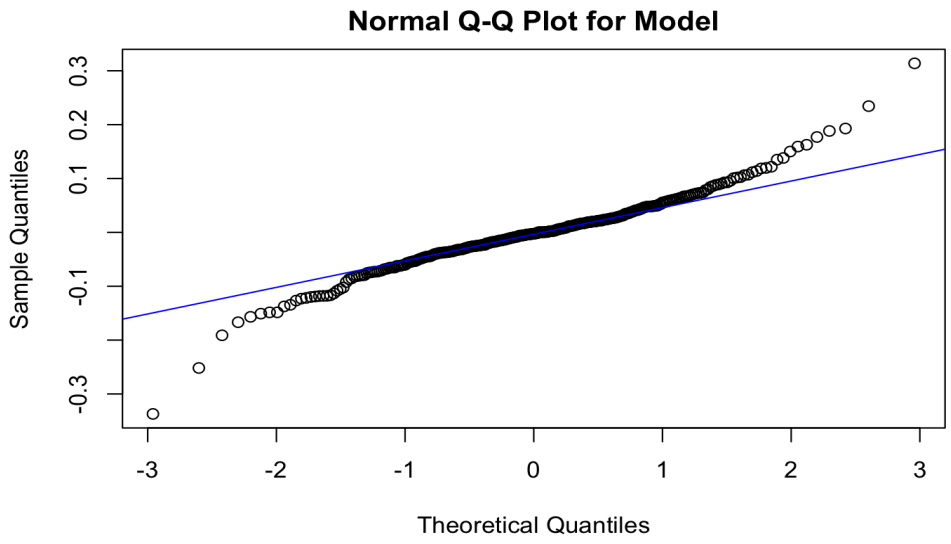
The AIC for SARIMA (2,1,0) x $(1,1,1)_{12}$ has the lowest value out of the three models tested and I will proceed with this model. Additionally, |ar2| & |sma1| shown above are less than 1, meaning the model is both invertible and causal.

PART 3: Diagnostic Checking for Selected Model

a) Histograms and Normal Distribution


Histogram of res
Density
0
1
2
3
4
5
6
7
-0.3
-0.2
-0.1
0.0
0.1
0.2
0.3


res
0.4
0.2
0.0
-0.2
-0.4
1995
2000
2005
2010
2015
2020
Time



The histogram shows the data represents a normal distribution. The residual plot also shows no visible trends, seasonality or any changes in the mean (which is nearly 0: -0.00197) or variance. The QQ plot also looks solid and follows a normal distribution.

Shapiro-Wilk normality test

data: res
W = 0.95383, p-value = 0.0841

Box-Pierce test

data: res
X-squared = 19.153, df = 17, p-value = 0.3198

Box-Ljung test

data: res
X-squared = 19.931, df = 17, p-value = 0.2778

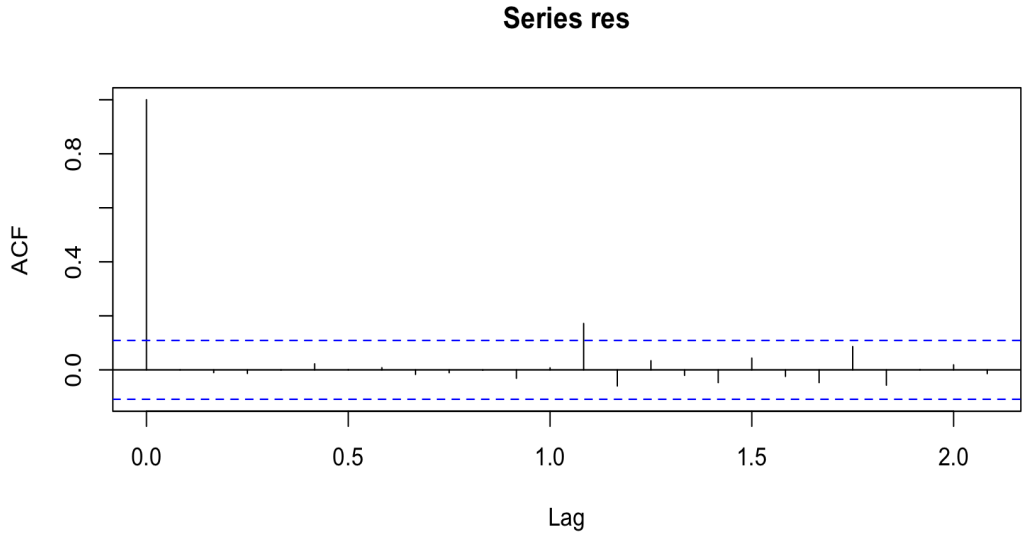
Box-Ljung test

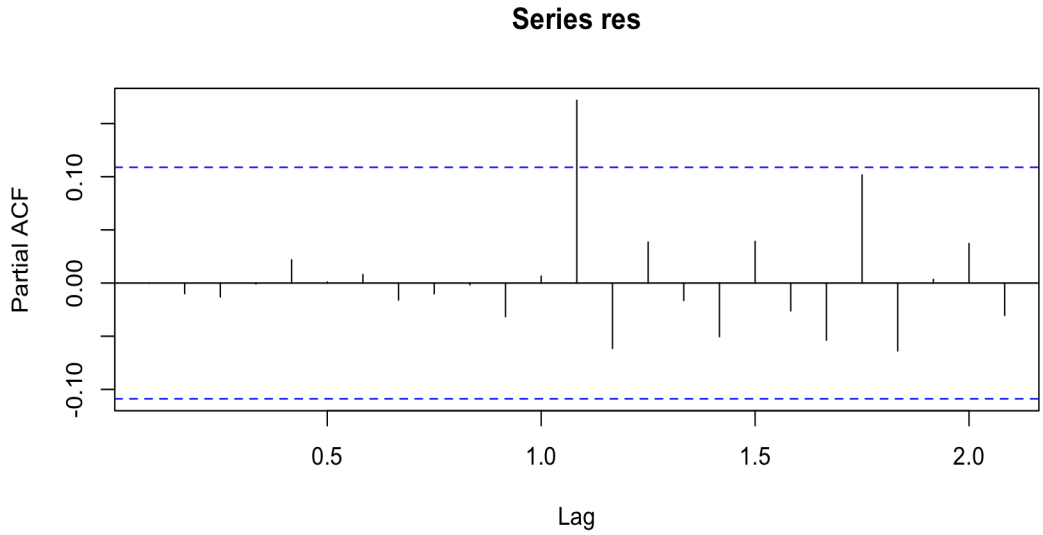
data: res
X-squared = 19.931, df = 18, p-value = 0.0687

These are the Shapiro-Wilks, Box-Pierce and Box-Ljung test results, and it looks like all of the

P-values are over 0.05 so it passes!

b) _ACF and PACF



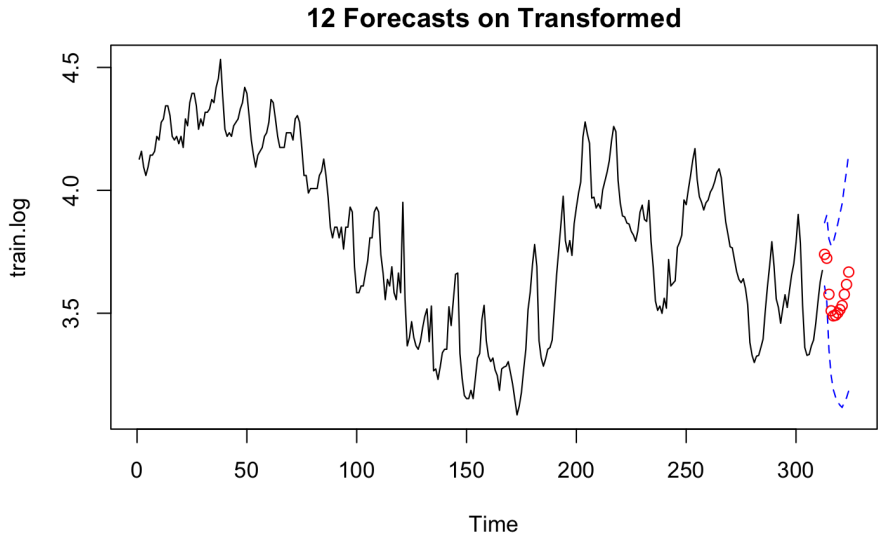


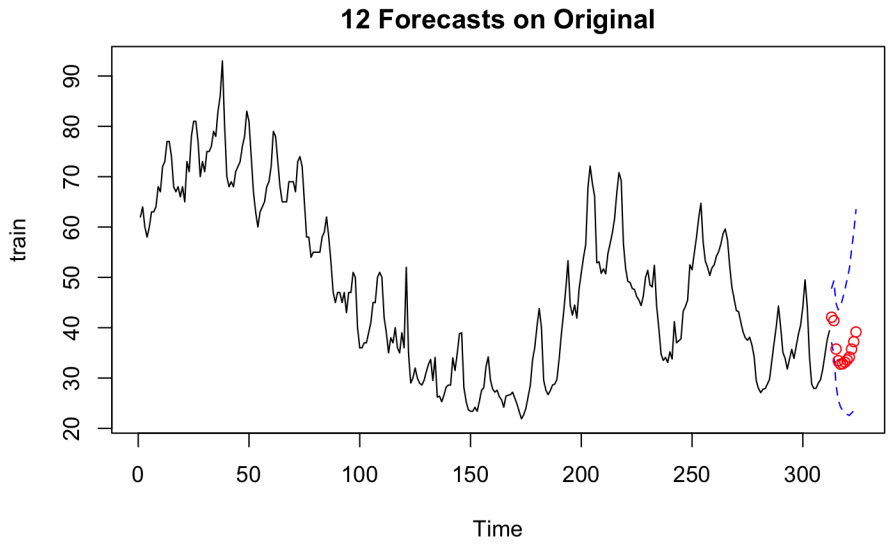
Both plots show no seasonality, trend or changes. Almost all residuals in ACF and PACF appear within the CI except for one lag, which might be a little concerning but otherwise, can be considered white noise. At the end of the day, $(2,1,0) \times (0,1,1)_{12}$ will be the final model, and the algebraic form is:

$$(1+0.0828_{(0.0574)}B)(1-B)(1-B^{12})X_t=(1+0.99_{(0.1044)}B^{12})Z_t$$

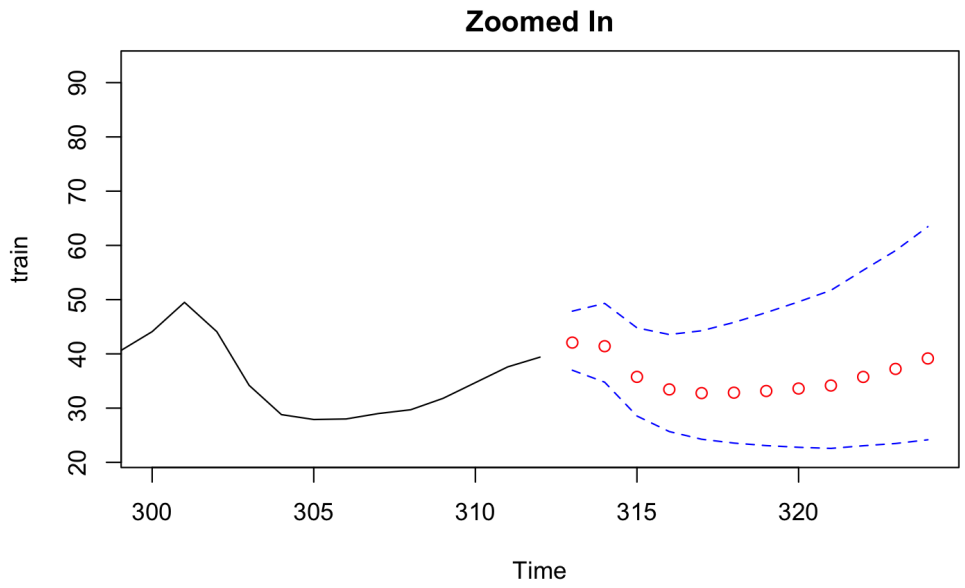
PART 4: Forecasting

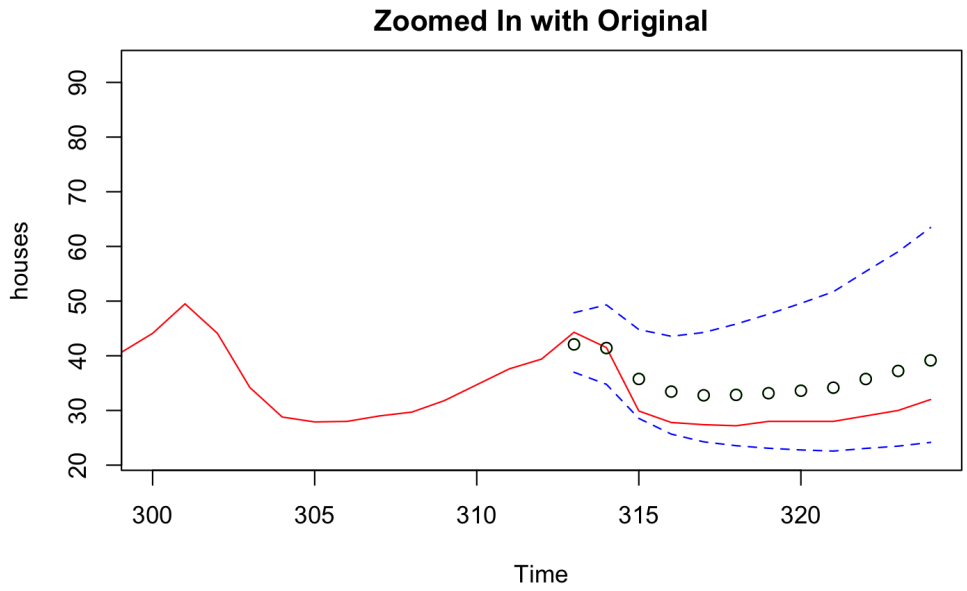
The following forecasting plots are for training data of Jan 1990-Dec 2015 and the forecasted test data are for the 12 months of Jan 2016-Dec 2016.





The following plot is the zoomed-in plots of the forecasted data, starting from entry point 300. Zooming into the plot allows a better visual to be seen.





Zooming in on the data that we forecasted, there is pretty good accuracy, as all of the black dots (forecast) are within the blue lines (95% CI). Although all of the points are not exactly accurate, the forecasted lines (in red) lie well in the middle of the CI points (blue lines). The model SARIMA (2,1,0) x $(0,1,1)_{12}$ appears to be a good model to predict the data.

CONCLUSION

The purpose of the project is to find a SARIMA model to predict median time on market of detached home sales in California. The final model, from a series of analysis, is SARIMA (2,1,0) x $(0,1,1)_{12}$,with the algebraic form of $(1+0.0828_{(0.0574)}B)(1-B)(1-B^{12})X_t=(1+0.99_{(0.1044)}B^{12})Z_t$.

Based on the forecasting data analysis, the SARIMA proved to be pretty generally accurate to describe the data, as the forecasted values are close to the original.

I would like to acknowledge and give a big thanks to Professor Raya Feldman, as well as TA's Sundeep and Youhong, for teaching the material to make all analysis in the project possible!

REFERENCES

NA. California Association of Realtors. “Historical Housing Data”. 07 March 2022. 15 March 2022. https://www.car.org/en/marketdata/data/housingdata.

Feldman, Raya. PSTAT174 Lectures 1-17. PSTAT174 Time Series Analysis. N.p., Winter 2021. Web.

APPENDIX

```
rm(list=ls())
library(reshape2)
library(forecast)
library(ROCR)
library(ggplot2)
library(ggfortify)

# importing dataset, selecting overall CA values only
data<-subset(UpdatedMedianTimeonMarketofExistingDetachedHomesHistoricalData)

# We are selecting (training) the years of 1990-2015 to forecast (testing) 2016
houses<-data[1:324, c(2)] #Jan 1990-Jan 2016
train<-data[1:312, c(2)] #Jan 1990-Dec 2015
test<-data[313:324, c(2)] #Jan 2016-Dec 2016

ts_houses=ts(houses)
plot.ts(ts_houses, main="(ALL) Median Time on Market of Detached Home all in CA",
        xlab= "Months", ylab="Median Days on Market")

# creating time series plots with training
tstrain=ts(train)
plot.ts(train, main="(Training) Median Time on Market of Detached Home all in CA",
        xlab= "Months", ylab="Median Days on Market")

# seeing a trend, looking at fit
length<-length(tstrain)
fit <- lm(tstrain ~ as.numeric(1:length(tstrain)))
abline(fit, col="red")
abline(h=mean(ts_houses), col="blue")

# histogram
hist(ts_houses, main="Median Time on Market of Detached Home all in CA",
     xlab= "Months", ylab="Median Days on Market")

# PACF and ACF
acf(train, main="Training ACF of original data", lag.max=40)
pacf(train, main =" Training PACF of original data", lag.max=40)

# Box-Cox test for data transformation
library(MASS)
t=1:length(tstrain)
fit=lm(tstrain~t)
bcTransform<-boxcox(tstrain~t, plotit=TRUE)

#lambda values
bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
lambda=bcTransform$x[which(bcTransform$y == max(bcTransform$y))]

# Perform transformations:
train.bc=(1/lambda)*(tstrain^lambda-1)
train.log = log(tstrain)

#compare TS PLOT for log versus boxcox
```

```r
train.log<-log(tstrain)
ts.plot(train.log, main= "Log Transformation", xlab="Year")
ts.plot(train.bc, main= "BC Transformation", xlab= "Year")

#compare HISTOGRAMS, log versus boxcox
hist(train.log, main= "Log Transformation", xlab="Year")
hist(train.bc, main= "BC Transformation", xlab="Year")

#compare VARIANCE, log versus boxcox
var(train.log)
var(train.bc)

# decomposition after choosing log
y <- ts(as.ts(train.log), frequency = 12)
decom <- decompose(y)
plot(decom)

#differencing at lag 12
train.log12 <- diff(train.log, lag=12)
plot.ts(train.log12, main="Differenced at lag 12")
fit <- lm(train.log12 ~ as.numeric(1:length(train.log12)))
abline(fit, col="red")
abline(h=mean(train.log12), col="blue")
# mean and variance after differencing at lag 12
var(train.log12)
mean (train.log12)

#differencing at lag 12 and 1
train.log121 <- diff(train.log12, lag=1)
plot.ts(train.log121, main="Differenced at lag 12 and 1")
fit <- lm(train.log121 ~ as.numeric(1:length(train.log121)))
abline(fit, col="red")
abline(h=mean(train.log121), col="blue")
# mean and variance after differencing at lag 12 and 1
var(train.log121)
mean (train.log121)

#acf and pcf after differencing at lag 12
acf(train.log12, main="ACF after differencing at lag 12", lag.max= 150 )
pacf(train.log12, main =" PACF after differencing at lag 12", lag.max=150)

#acf and pcf after differencing at lag 12 and lag 1
acf(train.log121, main="ACF after differencing at lag 12 and lag 1", lag.max=40)
pacf(train.log121, main =" PACF after differencing at lag 12 and lag 1", lag.max=40)

#histograms after differencing
hist(train.log12, main="Hist after differencing at lag 12")
hist(train.log121, main="Hist after differencing at lag 12 and lag 1")

#testing models
est=ar(train.log, aic = TRUE, order.max = 3, method = c("yule-walker"))
est$ar
```

```r
M1 <- arima(train.log, order=c(3,1,0), seasonal = list(order = c(0,1,1),
period = 12), method="ML")
M1

M2 <- arima(train.log, order=c(3,1,0), seasonal = list(order = c(1,1,1),
period = 12), method="ML")
M2

M3 <- arima(train.log, order=c(2,1,0), seasonal = list(order = c(0,1,1),
period = 12), method="ML")
M3

# Check histogram, residuals
fit <- arima(train.bc, order=c(2,1,0), seasonal = list(order = c(0,1,1),
                                                       period = 12), method="ML")
res = residuals(fit)
hist(res, density=20,breaks=20, col="blue", xlab="", prob=TRUE)

m <- mean(res)
std <- sqrt(var(res))
curve(dnorm(x,m,std), add=TRUE)
plot.ts(res)
fitt <- lm(res ~ as.numeric(1:length(res)));abline(fitt, col="red")
abline(h=mean(res), col="blue")

# Checking for normal distribution, ACF, PACF
qqnorm(res,main= "Normal Q-Q Plot for Model")
qqline(res,col="blue")

#tests,lag ~18 b/c sqrt(n=324)
shapiro.test(res)
Box.test(res, lag = 18, type = c("Box-Pierce"), fitdf = 1)
Box.test(res, lag = 18, type = c("Ljung-Box"), fitdf = 1)
Box.test(res^2, lag = 18, type = c("Ljung-Box"))

#acf and pacf
acf(res, main="Series res")
pacf(res, main="Series res")

fit.A <- arima(train.log, order=c(2,1,0), seasonal = list(order = c(0,1,1),
                                                          period = 12), method="ML")
forecast(fit.A) # prints forecasts with prediction bounds in a table

# To produce graph with 12 forecasts on transformed data:
pred.tr <- predict(fit.A, n.ahead = 12)
U.tr= pred.tr$pred + 2*pred.tr$se # upper bound of prediction interval
L.tr= pred.tr$pred - 2*pred.tr$se # lower bound
ts.plot(train.log,  main= "12 Forecasts on Transformed", xlim=c(1,length(train.log)+12))
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(train.log)+1):(length(train.log)+12),pred.tr$pred, col="red")

# To produce graph with forecasts on original data
```

```
pred.orig <- exp(pred.tr$pred)
U= exp(U.tr)
L= exp(L.tr)
ts.plot(train, main= "12 Forecasts on Original",xlim=c(1,length(train)+12))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(train)+1):(length(train)+12), pred.orig, col="red",
       main= "12 Forecasts Original")

# To zoom the graph, starting from entry 300:
ts.plot(train, main= "Zoomed In", xlim = c(300,length(train)+12))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(train)+1):(length(train)+12), pred.orig, col="red")

# To plot zoomed forecasts and true values (in houses):
plot.ts(houses, main= "Zoomed In with Original", xlim = c(300,length(train)+12), col="red")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(train)+1):(length(train)+12), pred.orig, col="green")
points((length(train)+1):(length(train)+12), pred.orig, col="black")
```