174 Final Project

Laila Elgamiel

Abstract:

In this report, I looked at mean housing costs in the city of London from years 1995-2020, using time series techniques such as transformation, differencing, model selection using ACF and PACFs, AICC comparison, residual diagnostic checking, forecasting, and more. My goal was to determine whether or not mean housing costs in London would continue to steadily increase as they have been for the last 25 years. Results confirmed that London housing prices can be successfully modeled and forecasted by a SARIMA, or Seasonal Autoregressive Integrated Moving Average model, as discussed later in the report.

Introduction:

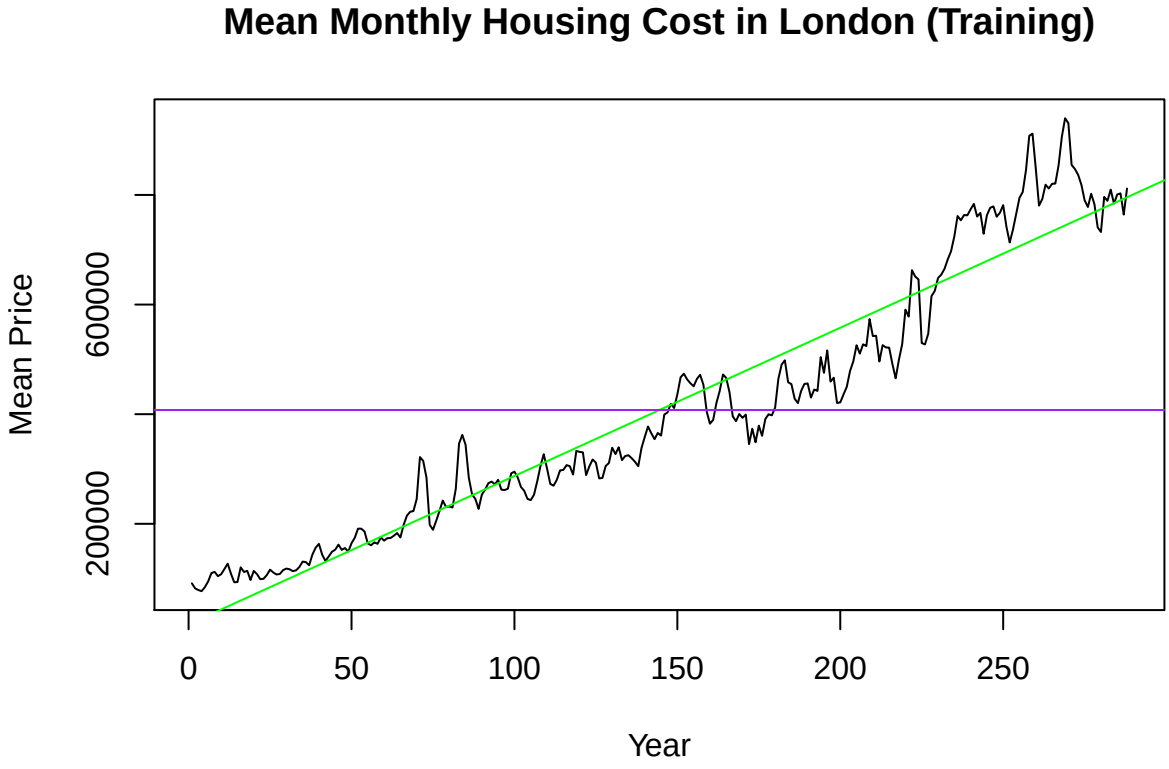
The main problem addressed in this report is finding a time series model to fit to data about the mean housing prices in London from years 1995-2020, in order to predict future trends in housing data and to examine patterns in the housing market of London over the past 25 years, and possibly extrapolate those patterns to the UK. Housing prices are a good indicator of a country's economic health, as well as provide information about quality of life and level of accessibility of affordable housing. While housing costs have always followed a linear growth due to inflation, it is possible that in times of economic depression, the housing market may crash or balloon. Thus, I wanted to model this data in order to allow for forecasting, or prediction, of future housing costs as an indicator of future economic health of the UK, and possibly other countries.

I planned to address the problem of fitting a model to the data by first examining the data graphed as a time series and determining any possible transformations or differencing that must occur. I split the data into a test and training set to test my final model against actual data points after it had been determined. After some comparison between transformation methods, I found that in order to make the data stationary to allow for a model to be properly fit to it, I needed to stabilize the variance of the time series using a box-cox transformation. Differencing at lags 12 and 1 was also required in order to remove the trend and seasonality detected in the original dataset. After making sure the data was in fact stationary, I moved onto examination of autocorrelation functions and partial autocorrelation functions to come up with some possible parameters for both the moving average and autoregressive portions, as well as seasonal portions, of the model. After coming up with a list of possible model parameters, I moved onto AICC calculation and comparison of every single possible model determined by ACF and PACF plots, ensuring that any models with parameter coefficients containing 0 in their respective confidence intervals had their AICCs reexamined after fixing those coefficients at 0. After the best fit model of SARIMA$(1,1,3)(0,1,1)_{12}$ had been determined, I confirmed stationarity and invertibility of both AR and MA portions of the model, then moved onto diagnostics of the residuals of said model, ensuring that they were independent and normally distributed via examination of histograms, QQ-plots, and portmanteau testing. After my chosen model was confirmed to be a good fit with proper residuals, I conducted forecasting on both the transformed and original datasets, which proved to be successful in both cases.

The dataset was sourced from Kaggle; https://www.kaggle.com/justinas/housing-in-london
All analysis was conducted in R, Version 1.3.1093.

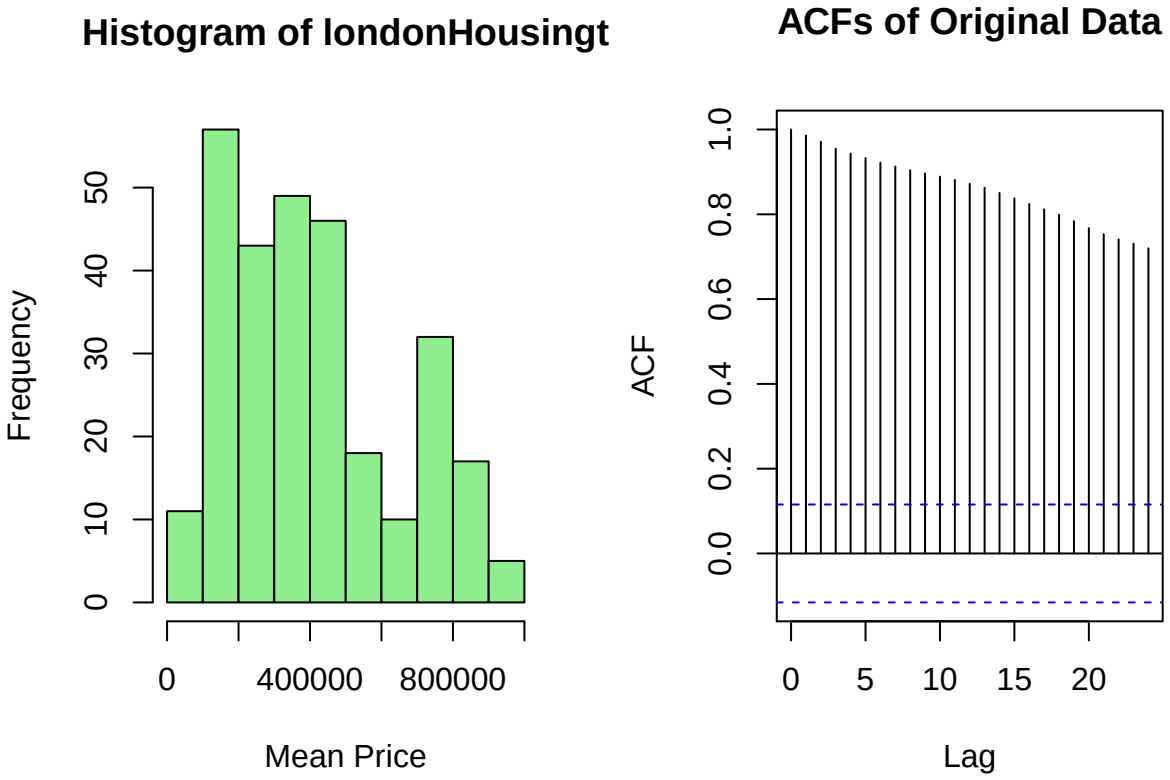
Mean Monthly Housing Cost in London: Examined

In order to test the model I will eventually chose to represent the data, I will separate the dataset into a training portion, and save the last 12 months of data to be a test portion, the latter of which I will use to compare against my eventual predictions on the data. Time to examine the main features of the graph:

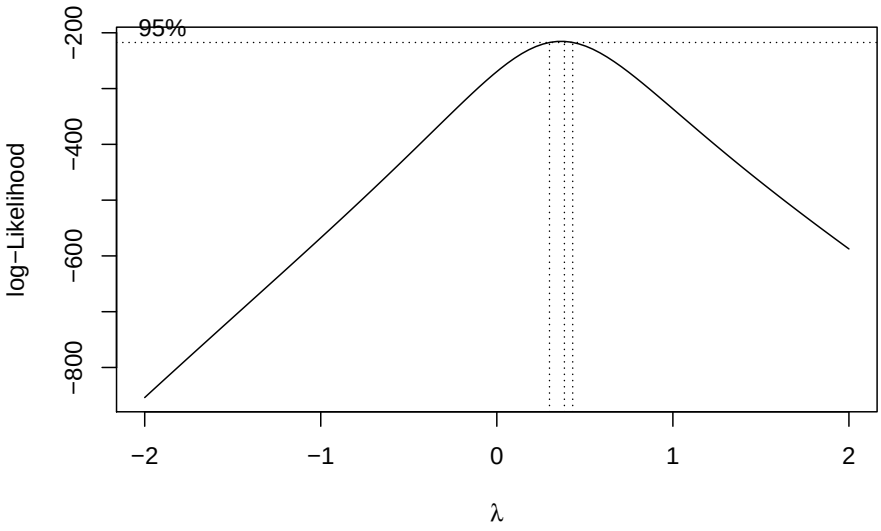


There are three main aspects of this data that I must address before fitting a model, and they are as follows:
i) There is a clear linear trend in the data, represented by the green line. Trend is indicative of non-stationarity, and the goal is to achieve a stationary process, or a process where the mean and variance remain constant throughout and do not depend on time, in order to properly fit the model.
ii) Because the time series is based on data collected about the mean housing price in London at the beginning of that month, I can assume there is seasonality, or recurring changes in the data that take place every 12 months. This cyclical nature of the series also is indicative of on-stationarity. Thus, seasonality, as well as trend, will have to be addressed later on, via differencing.
iii) Just looking at the graph, I can see that the variance seems to increase as time goes on, with greater jumps between data points towards the end of the series when compared to the beginning. Because the variance is not constant, I will have to apply a variance-stabilizing transformation of sorts to address this issue.

The histogram and ACFs of the data should confirm my above ideas:



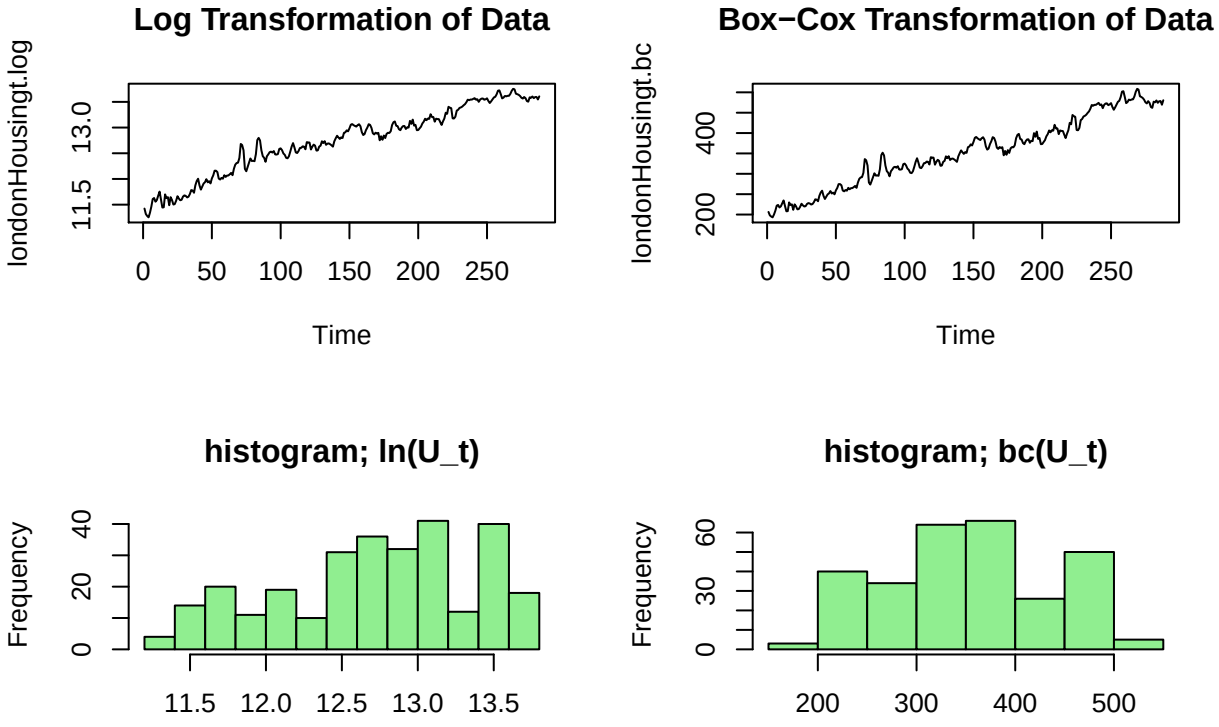
Not only is the histogram somewhat skewed and not Gaussian, or normally distributed, the ACFs decay exponentially, confirming my initial beliefs about the non-stationarity of the data. In order to make the data stationary, I must first begin by applying a transformation to the data. Running the box-cox function in R will give me an idea as to which variance stabilizing transformation I should apply, depending on which lambda, or maximum value, of the transformation is returned:



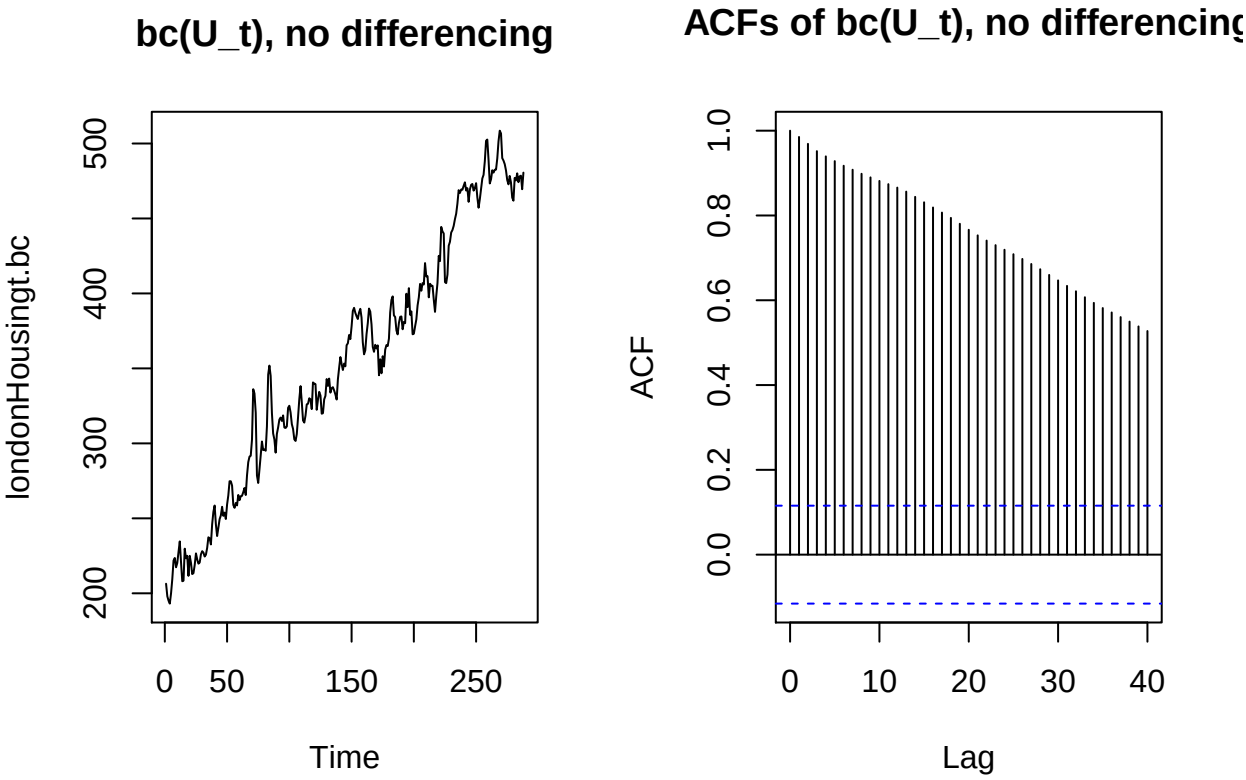

```
## [1] "Lambda:"
```

```
## [1] 0.3838384
```

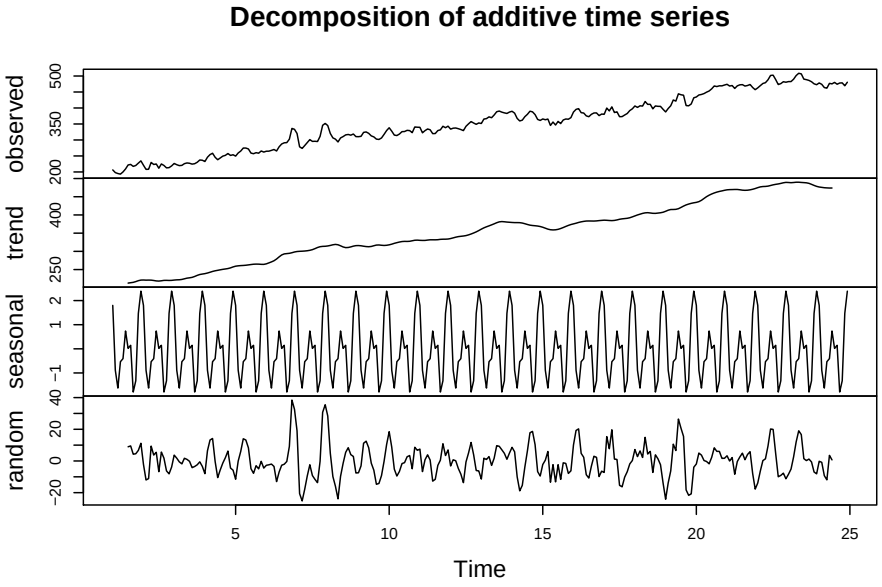
Because the lambda returned by the box-cox has a value = 0.3838384, I can say that a log transformation is not the right choice in this case, as 0 is not contained within the confidence intervals. It is more likely that $\lambda = \frac{1}{3}$ lies within the confidence intervals. Just in case, I will compare the results of a log transformation to that of a box-cox transformation in order to confirm my beliefs:



It seems that the box-cox transformation gave a more symmetric histogram and did a better job at evening out the variance in the time series plot than the log transformation, so I will go ahead and continue with the box-cox transformed data.



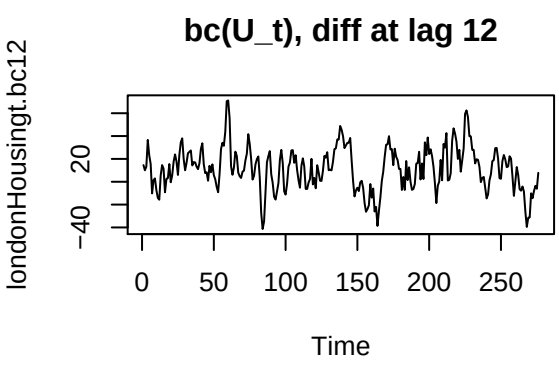
From the time series graph, I see the issue of linear trend in the data, which must be dealt with in order to make the data stationary. From the ACFs, I can determine that there is non-stationarity in the data, as they decay very slowly. I can also see some slight seasonality, and because I know this data is a monthly dataset, I can predict the seasonality is at a lag of 12. To remove both the trend and seasonality, I will need to difference the data, and do so more than once. First, I'll take a closer look at a decomposition graph to confirm my beliefs of trend and stationarity:

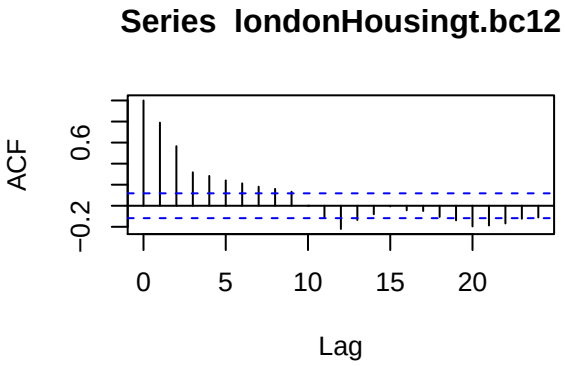


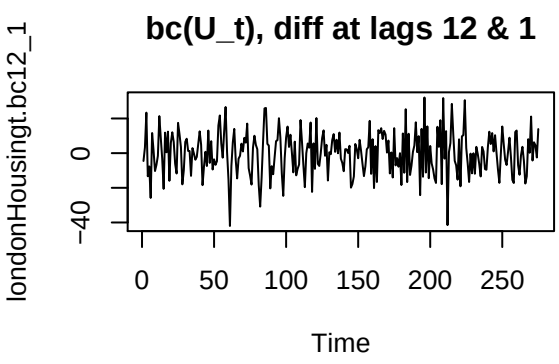
After decomposing the time series, I can confidently detect both a trend and seasonality in the transformed data, indicating that I will need to difference in order to attain stationarity. Differencing first at lag 12 to deal with seasonality, then at lag 1 to deal with the trend:

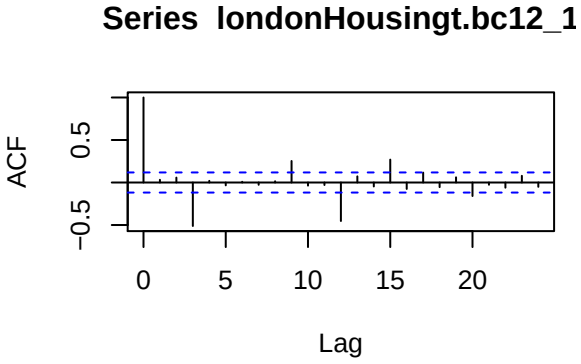
```
## [1] "Variance of transformed, undifferenced data:"
```

```
## [1] 7152.329
```









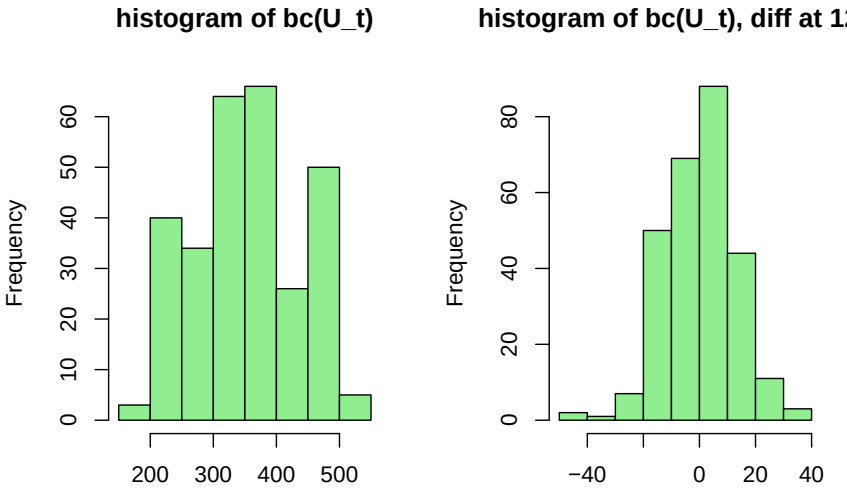

```
## [1] "Variance of transformed data, differenced at lag 12:"
```

```
## [1] 358.2152
```

```
## [1] "Variance of transformed data, differenced at lags 12 and 1:"
```

```
## [1] 151.0222
```

While differencing at lag 12 removed the issue of seasonality and gave a lower variance than un-differenced data, I could still detect non-stationarity due to a trend in the data, because the ACFs decayed slowly. Thus, I decided to difference again at lag 1, which seemed to remove the remaining trend and make the data stationary, as confirmed by the corresponding plot of the time series and the ACFs. The variance of the series differenced at both lags 1 and 12 also confirms this, as it is lower than the variance of the series differenced at just lag 12. An increase in variance would indicate overdifferencing.



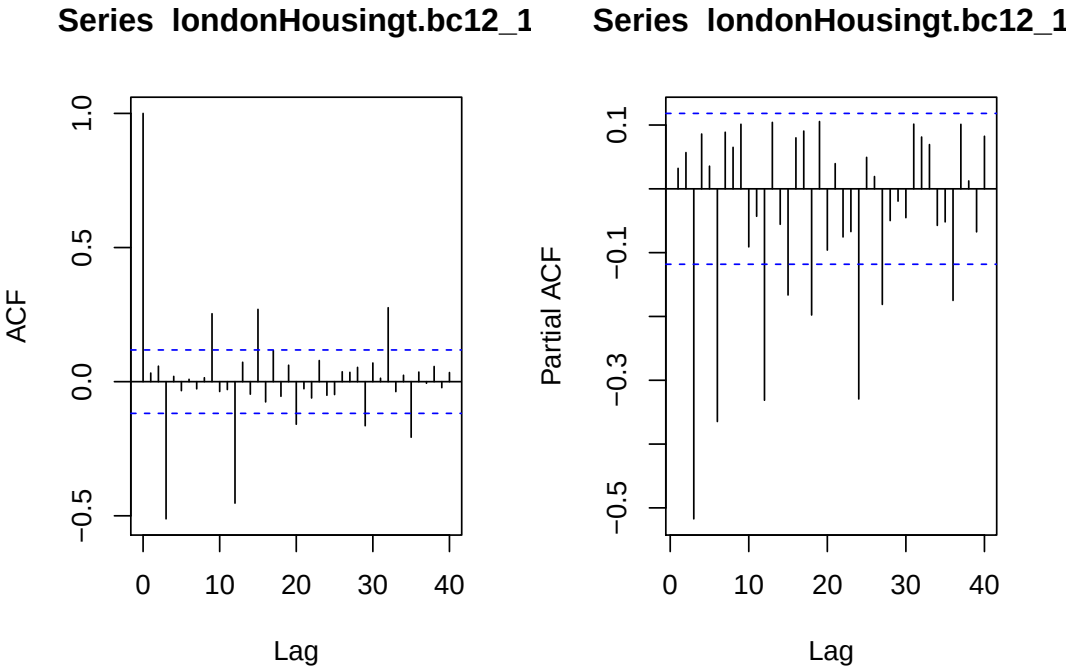
The differencing seems to have produced a more symmetric histogram than that of the solely transformed data, indicating that I have made the correct choice in differencing. Quickly checking whether I should difference once more at lag 1 by comparing variances:

```
## [1] 151.0222
```

```
## [1] 292.6306
```

and because the variance has increased, I can safely say that differencing again at lag 1 (overall differenced at lag 12 and 2) would be overdifferencing the model, so I will stick with my model differenced at lag 12 to remove seasonality, and then again at lag 1 to remove trend. Now, onto the ACFs and PACFs of this model:

Possible Model Selection



Ignoring significant acfs and pacfs at any lags within 3 lags of seasonality ($12 \pm 1,\ 12 \pm 2,\ 12 \pm 3$), as they may appear outside of the confidence intervals, I have come up with the following possible parameters for a SARIMA(p,d,q)(P,D,Q)$_s$ model:

s=12, because this is monthly data, so it clearly has a seasonality of 12.
d=1, as I chose to difference at lag 1 to remove the trend.
D=1, as I differenced at a lag of 12 to remove seasonality.
p= 3 or 6, because these are the only significant pacfs not within 3 lags from lag 12, 24, or 36, and not larger than around 11 lags. I will compare AICCs for all lags 1:6.
q= 3, because it is the only significant autocorrelation not within 3 lags from lag 12, 24, or 36, and not larger than around 11 lags. While lag 20 technically lies outside of the confidence intervals, I chose to exclude that possibility because it is too close to the edge. I will compare AICCs for lags 1:3 as well just in case.
P= 1 or 2, as both pacfs at lags 12 and 24 are outside of the confidence intervals. Pacf at lag 36 is not significantly large enough for me to consider it.
Q= 1 or 3, as both acfs lags 12 and 35 (very close to 36) are outside of the confidence intervals.

AICCs: Comparing and Contrasting

I will begin by building and comparing several SMA models to find which q and Q values give the lowest AICCs:

```
## [1] 2024.911 1991.864 1891.479
```

```
## [1] 2028.923 1995.349 1894.375
```

Of these, the lowest AICC (1891.479) is given by a SMA model with parameters q=3, Q=1. The second lowest AICC value (1894.375) belongs to a model where q=3, Q=3. I will examine both of these models more closely to determine if I should fix any parameters at 0:

```
##
## Call:
## arima(x = londonHousingt.bc, order = c(0, 1, 3), seasonal = list(order = c(0,
##     1, 1), period = 12), method = "ML")
##
## Coefficients:
##          ma1     ma2      ma3     sma1
##       0.1889  0.1528  -0.7413  -0.8727
## s.e.  0.0457  0.0533   0.0488   0.0460
##
## sigma^2 estimated as 50.13:  log likelihood = -940.63,  aic = 1891.26
```

```
##
## Call:
## arima(x = londonHousingt.bc, order = c(0, 1, 3), seasonal = list(order = c(0,
##     1, 3), period = 12), method = "ML")
##
## Coefficients:
##          ma1     ma2      ma3     sma1     sma2    sma3
##       0.1840  0.1578  -0.7547  -0.8151  -0.0826  0.0073
## s.e.  0.0448  0.0547   0.0537   0.0684   0.0829  0.0648
##
## sigma^2 estimated as 49.59:  log likelihood = -939.98,  aic = 1893.96
```

In the first model with Q=1, I can see that no coefficients have 0 contained within the confidence interval (coefficient ± 2*s.e.), but looking at the second model with Q=3, I can see that both sma2 and sma3 can be fixed at 0, leaving only one seasonal MA coefficient, such as in the first model. For this reason, I will proceed with the first model (q=3, Q=1) and introduce an AR component:

```
## [1] 1892.976 1893.862 1895.587 1897.677 1899.499 1900.916
```

```
## [1] 1895.083 1895.981 1897.699    0.000 1901.653 1903.052
```

Because of an R error in calculating the AICC of the P=2, p=4 model, I was unable to determine the exact value, but I can be certain that it would be greater than the two lowest AICCs that were found. Introducing the AR component did increase the AICC slightly. I found the lowest AICC value (1892.976) to belong to model P=1, p=1, then the second lowest (1893.862) to belong to model P=1, p=2. Because AICC tends to overestimate p, I can be fairly certain that the first model is the better fit, but I will continue with checking the coefficients of both of the models to be sure:

```
##
## Call:
## arima(x = londonHousingt.bc, order = c(1, 1, 3), seasonal = list(order = c(1,
##     1, 1), period = 12), method = "ML")
##
## Coefficients:
##          ar1     ma1     ma2      ma3    sar1     sma1
##       0.0983  0.1382  0.1248  -0.7893  0.0798  -0.9061
## s.e.  0.0816  0.0535  0.0524   0.0548  0.0780   0.0538
##
## sigma^2 estimated as 49.18:  log likelihood = -939.28,  aic = 1892.56
```

```
##
## Call:
## arima(x = londonHousingt.bc, order = c(2, 1, 3), seasonal = list(order = c(1,
##     1, 1), period = 12), method = "ML")
##
## Coefficients:
##          ar1     ar2     ma1     ma2      ma3    sar1     sma1
##       0.1181  0.0914  0.1111  0.0754  -0.8077  0.0780  -0.9041
## s.e.  0.0809  0.0834  0.0532  0.0650   0.0502  0.0772   0.0532
##
## sigma^2 estimated as 49.13:  log likelihood = -938.66,  aic = 1893.32
```

Looking at the first model with AICC=1892.976, I can see the possibility of fixing both ar1 and sar1 at 0, as these coefficients contain 0 within their confidence intervals. I will check if the AICC is lowered by fixing those coefficients at 0:

```
## [1] 1892.303
```

```
## [1] 1891.935
```

```
## [1] 1891.479
```

It seems that fixing sar1 at 0 (1891.935), and both ar1 and sar1 (1891.479) at 0 lowered the variance. However, fixing only ar1 at 0 did not lower the AICC significantly enough for me to remove it from my model entirely. Because of this, I will choose to fix only sar1 at 0, thus changing this model to have parameter P=0.

Now looking at my second best model with AICC=1893.862, I see that coefficients ar1, ar2, ma2, and sar1 can all be fixed at 0. Once again I will check and compare AICCs with the original AICC:

```
## [1] 1893.777
```

```
## [1] 1892.976
```

```
## [1] 1896.554
```

```
## [1] 1892.778
```

```
## [1] 1891.935
```

Here, I see that fixing ar2 (1892.976), sar1 (1892.778), and both (1891.935) all lowered the AICCs, so I am inclined to say that my first model with p=1 is the better fitting of the 2. Because fixing both ar2 and sar1 at 0 would give me the same fit as my first choice model, I will check the possibility of fixing both ar1 and sar1 at 0:

```
## [1] 1892.815
```

Fixing both ar1 and sar1 at 0 lowered the AICC as well, and almost as much as fixing sar1 at 0 alone. Thus, for this p=2 model, I will change the P parameter to be P=0, and fix ar1 at 0.

With that, I have determined the two best models for this particular time series data to be:

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

and

$$\text{SARIMA}(2,1,3)(0,1,1)_{12} \text{ with ar1 fixed at 0}$$

The first model matches a possible model suggested by looking at the ACF and PACFs of the transformed and differenced data, while the second model has parameter p=2, where a PACF at lag 2 was not visibly outside of the confidence intervals in the plot. I will fit both models and conduct further tests to determine which model gives a better fit.

Final Two Models: Which is Best?

After fitting both my models, I will examine the summary of each model, with the final estimated model parameters:

```
## [1] "fit1:"
```

```
##
## Call:
## arima(x = londonHousingt.bc, order = c(1, 1, 3), seasonal = list(order = c(0,
##     1, 1), period = 12), method = "ML")
##
## Coefficients:
##          ar1     ma1     ma2      ma3    sma1
##       0.1063  0.1368  0.1186  -0.7773  -0.879
## s.e.  0.0823  0.0549  0.0501   0.0501   0.046
##
## sigma^2 estimated as 49.7:  log likelihood = -939.81,  aic = 1891.62
```

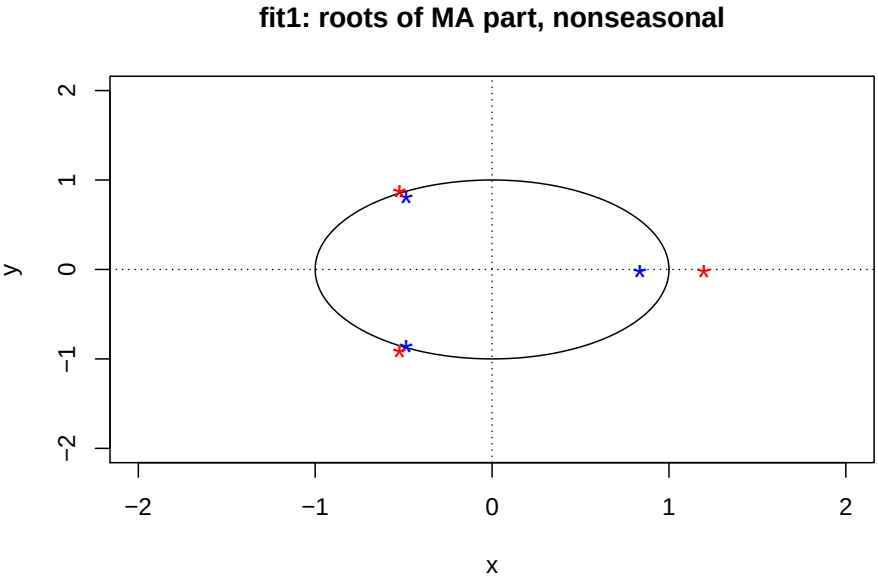
```
## [1] "fit2:"

##
## Call:
## arima(x = londonHousingt.bc, order = c(2, 1, 3), seasonal = list(order = c(0,
##     1, 1), period = 12), transform.pars = FALSE, fixed = c(0, NA, NA, NA, NA,
##     NA), method = "ML")
##
## Coefficients:
##       ar1     ar2     ma1     ma2      ma3     sma1
##         0  0.0786  0.2597  0.1958  -0.8926  -1.1461
## s.e.    0  0.0904  0.0553  0.0729   0.0654   0.0600
##
## sigma^2 estimated as 32.39:  log likelihood = -940.25,  aic = 1892.5
```

Now, I must check the stationarity and invertibility of both these models.

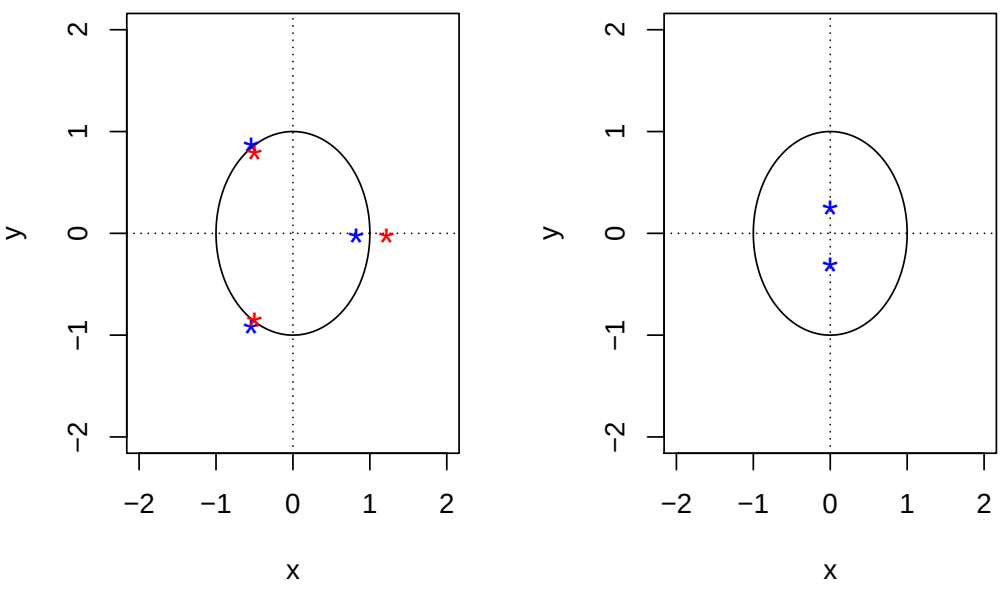
Stationarity and Invertibility of fit1, fit2

The first model is stationary, as the coefficient ar1, $|\phi_1| < 1$. Checking invertibility:



Because the roots (represented by red stars) of the equation $\theta(z)$ lie outside of the unit circle, and the seasonal coefficient sma1 meets the condition $|\Theta(z)| = 0.879 < 1$, the fit1 model is stationary, as well as invertible.

Looking at the roots of the $\phi(z)$ and $\theta(z)$ equations of fit2:




```
## [1] "fit2; roots of AR part, nonseasonal:"
```

```
## [1] 0+3.566882i 0-3.566882i
```

Here, I can see that the roots (represented by red stars) of $\theta(z)$ equation are do not lie outside the unit circle, indicating that the model is not invertible. For this reason, I can be certain that the fit2 model is not the correct choice for modeling this data. Having ruled out fit2, I can conclude that fit1, SARIMA$(1,1,3)(0,1,1)_{12}$, is the right model to fit this data. The model equation for fit1 can be written:

$$(1-B^{12})(1-B)(1-0.1063_{(0.0823)}B)X_t = (1+0.1368_{(0.0549)}B+0.1186_{(0.0501)}B^2-0.7773_{(0.0501)}B^3)(1-0.879_{(0.046)}B^{12})Z_t,$$

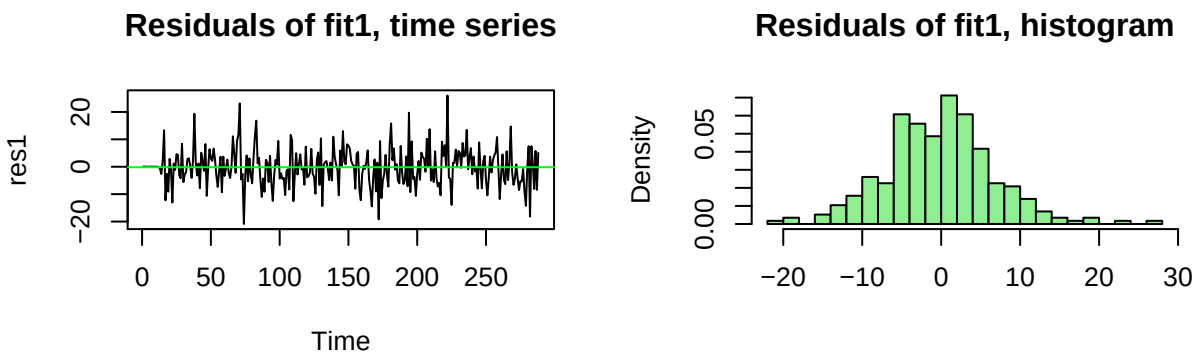
$$\hat{\sigma_z}^2 = 49.7$$

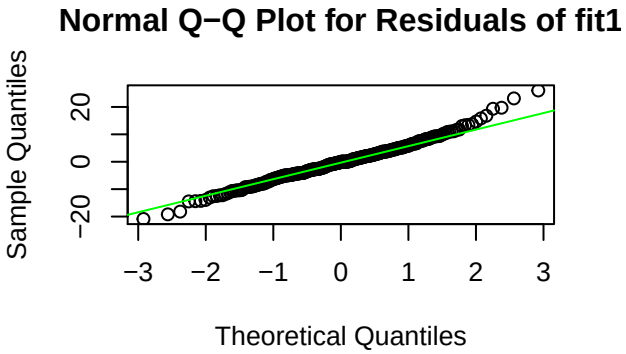
I will continue by conducting diagnostic checking for the residuals of the fit1 model, to confirm the goodness-of-fit of the model.

Diagnostic Checking of Residuals

```
## [1] "Mean of residuals of fit1:"
```

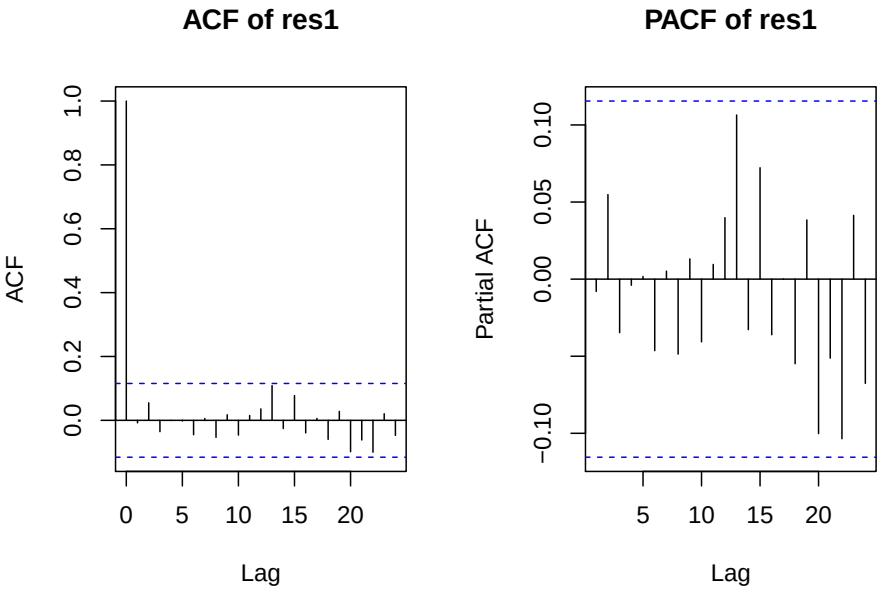
```
## [1] -0.1676825
```





From these plots, I can see that the residuals of fit1 follow a normal distribution, as the mean=-0.1676825, which is very close to 0, the time series seems to have constant variance with no sign of visible trend or seasonality, and the histogram and QQ-plot look good (histogram takes Gaussian distribution about 0 with slight heavy tail to left, QQ-plot is linear).

Plotting ACFs and PACFs of the residuals to see if the model can be improved in any way:



Here, residual acfs and pacfs for all lags are contained within the confidence intervals and can be counted as 0, confirming that no necessary changes need to be made in the parameters (p, q) of the current fit model.

Now I will conduct various portmanteau tests to confirm the normality and independence of the residuals, with lag=17, as $\sqrt{288} = 16.97 \approx 17$:

```
##
##  Shapiro-Wilk normality test
##
## data:  res1
## W = 0.98835, p-value = 0.0204
```

```
##
##  Box-Pierce test
##
## data:  res1
## X-squared = 9.5997, df = 12, p-value = 0.651
```

```
##
##  Box-Ljung test
##
## data:  res1
## X-squared = 10.052, df = 12, p-value = 0.6114
```

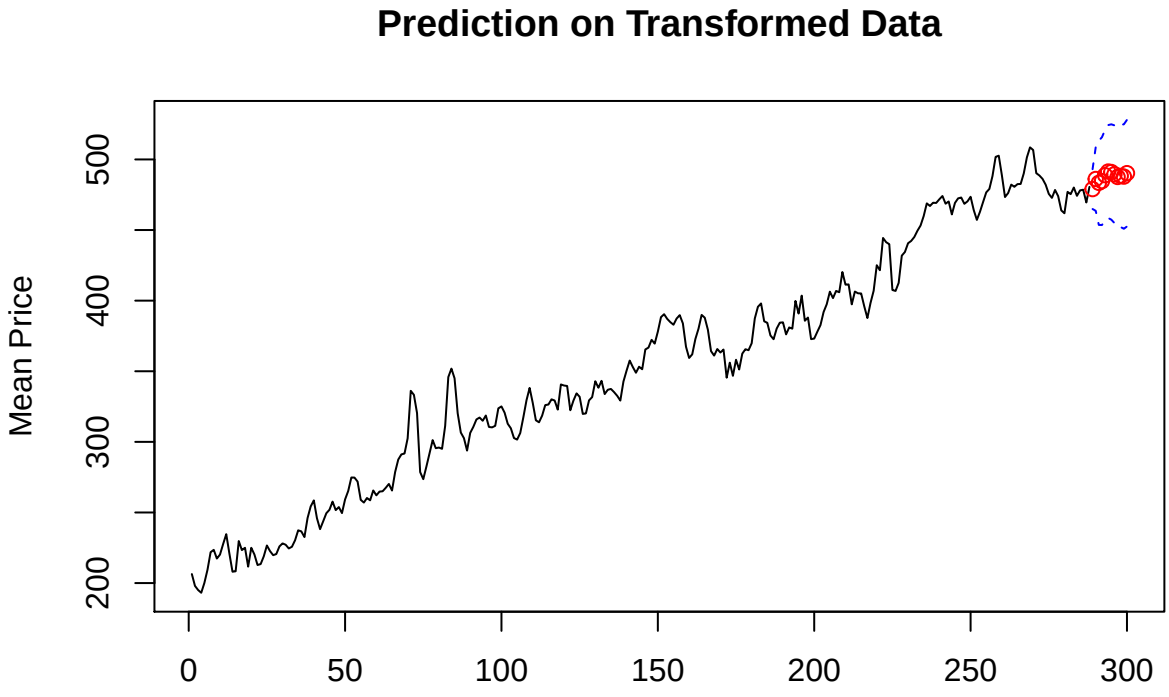
```
##
##  Box-Ljung test
##
## data:  (res1)^2
## X-squared = 19.879, df = 17, p-value = 0.2805
```

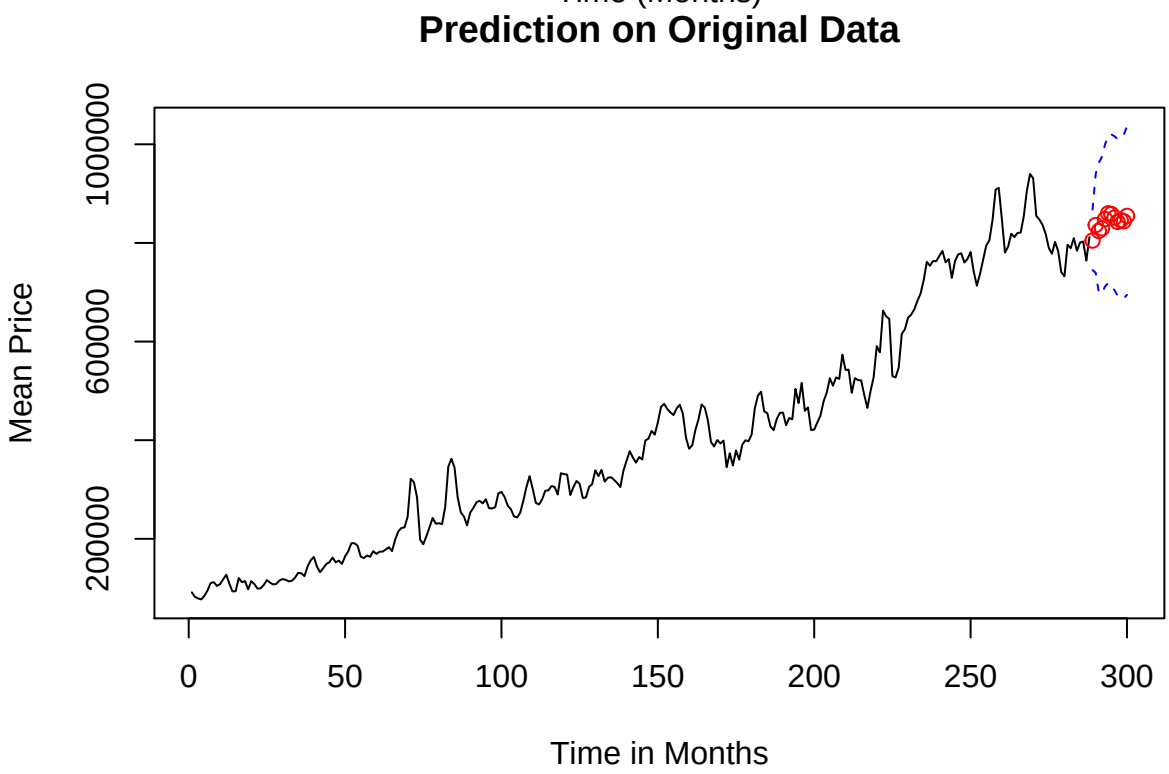
The Box-Pierce test was passed, indicating that the residuals are uncorrelated, the Box-Ljung test was passed, disproving linear dependence amongst the residuals, and the McLeod-Li test was passed, indicating no non-linear dependence in the residuals. However, the Shapiro-Wilk test was not passed (p-value less than 0.05). Upon further examination, I can determine that this is due to the fact that the histograms of both the original dataset and the residuals are slightly heavy tailed. This can lead the Shapiro-Wilk test to return non-normality. I will confirm that the residuals can be fit to an AR model of order 0, indicating that they meet the definition of white noise:

```
##
## Call:
## ar(x = res1, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  47.59
```

and because order 0 was selected, I can confirm that the residuals meet the requirements for a well-fitted model. Now, onto forecasting:

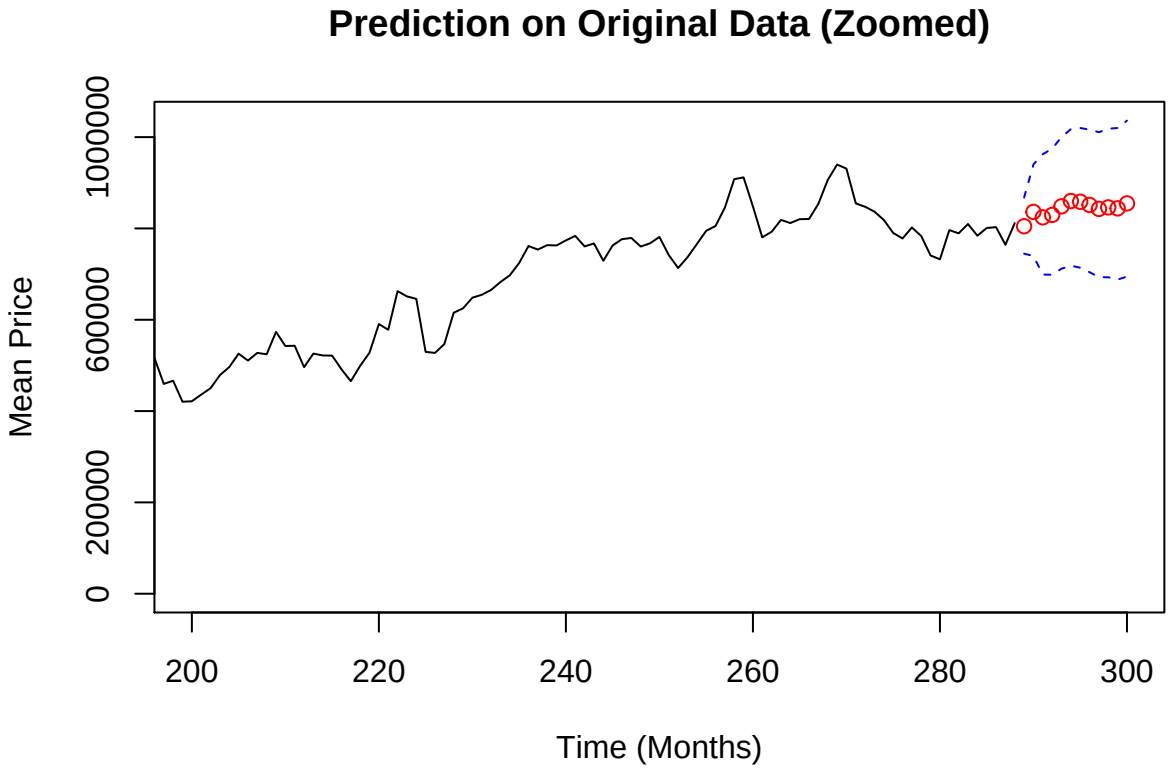
Forecasting

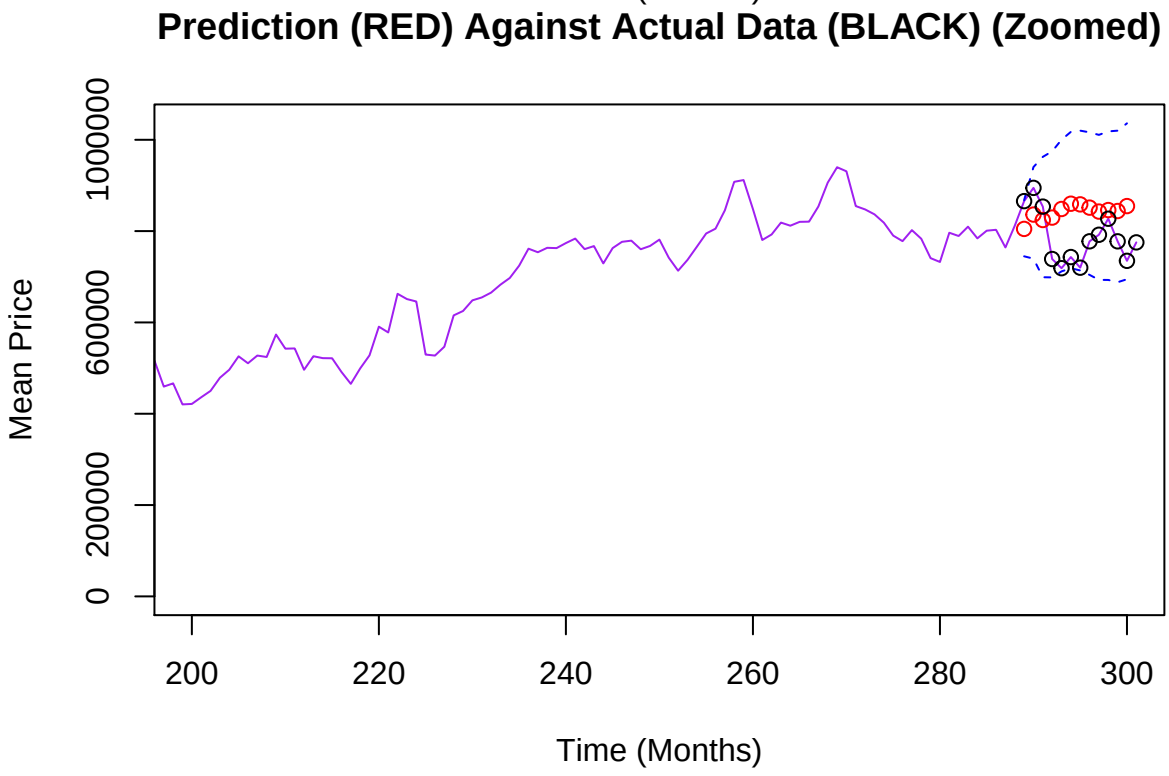




The points predicted from my chosen model (in red throughout) lie within the prediction intervals in both the transformed data plot, as well as the original data plot (no box-cox transformation). Thus, I can determine that my forecasting was successful, and the model that I chose to fit the data was a good choice. Zooming

in a bit on the above graph of non-transformed data for a better view:





and I can see that the predicted test points and actual test points both lie within the prediction intervals I calculated from the original, non-transformed data. Thus, I can conclude that my chosen model provides a good fit for the data, and accurately forecasts future data points in the time series.

Conclusion:

In conclusion, I was able to successfully model the time series data and forecast on the dataset, using the model:

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

with mathematical formula:

$$(1-B^{12})(1-B)(1-0.1063_{(0.0823)}B)X_t = (1+0.1368_{(0.0549)}B+0.1186_{(0.0501)}B^2-0.7773_{(0.0501)}B^3)(1-0.879_{(0.046)}B^{12})Z_t,$$

$$\hat{\sigma_z}^2 = 49.7$$

I want to thank TAs Sunpeng Duan and Jasmine Li for all their help in both section and office hours on my project.

References

[link] (https://www.kaggle.com/justinas/housing-in-london) -For dataset
[link] (https://stat.ethz.ch/pipermail/r-help/2007-June/134480.html) -For function used for undoing box-cox transformation
[link] (https://www.stat.berkeley.edu/~bartlett/courses/153-fall2010/lectures/6.pdf) -For more information on invertibility and stationarity

Lectures and Class Materials:
Week 3, slides 66, 68
Week 15, all slides
Check your Understanding, week 3-4
Check your Understanding, week 5-6

Appendix.

```
knitr::opts_chunk$set(echo = TRUE)
#install.packages("ggplot2")
#install.packages("ggfortify")
#install.packages("MuMIn")
#install.packages("forecast")
library(ggplot2)
library(ggfortify)
library(tidyverse)
library(MASS)
library(MuMIn)
library(forecast)
options(scipen=999)

housing<-read.csv("housing_in_london_monthly_variables.csv")
londonH <- housing %>% filter(area=="city of london")
londonHousing<-ts(londonH$average_price, frequency=12, start=1995)
plot.ts(londonHousing, xlab="Year", ylab="Mean Price", main="Mean Monthly Housing Cost in London from 19
londonHousingt<-londonH$average_price[c(1:288)]
```

```r
mean_training=mean(londonHousingt)
londonHousing_test = londonH$average_price[c(289:301)] #last 12 data points to test
plot.ts(londonHousingt, xlab="Year", ylab="Mean Price", main="Mean Monthly Housing Cost in London (Trai
fit <- lm(londonHousingt~ as.numeric(1:length(londonHousingt))); abline(fit, col="green")
abline(h=mean_training, col="purple")
par(mfrow=c(1,2))
hist(londonHousingt, xlab="Mean Price", col="light green")
acf(londonHousingt, main="ACFs of Original Data")
bcTransform <- boxcox(londonHousingt~ as.numeric(1:length(londonHousingt)))
print("Lambda:")
bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
lambda=bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
londonHousingt.bc = (1/lambda)*(londonHousingt^lambda-1)
londonHousingt.log <- log(londonHousingt)
par(mfrow=c(2,2))
plot.ts(londonHousingt.log, main="Log Transformation of Data")
plot.ts(londonHousingt.bc, main="Box-Cox Transformation of Data")
hist(londonHousingt.log, col="light green", xlab="", main="histogram; ln(U_t)")
hist(londonHousingt.bc, col="light green", xlab="", main="histogram; bc(U_t)")
par(mfrow=c(1,2))
plot.ts(londonHousingt.bc, main="bc(U_t), no differencing")
acf(londonHousingt.bc, lag.max=40, main= "ACFs of bc(U_t), no differencing")
y <- ts(as.ts(londonHousingt.bc), frequency = 12)
decomp <- decompose(y)
plot(decomp)
print("Variance of transformed, undifferenced data:")
var(londonHousingt.bc)
londonHousingt.bc12 <- diff(londonHousingt.bc, lag=12)
londonHousingt.bc12_1<- diff(londonHousingt.bc12, lag=1)
par(mfrow=c(2,2))
plot.ts(londonHousingt.bc12, main="bc(U_t), diff at lag 12")
acf(londonHousingt.bc12)
plot.ts(londonHousingt.bc12_1, main="bc(U_t), diff at lags 12 & 1")
acf(londonHousingt.bc12_1)
print("Variance of transformed data, differenced at lag 12:")
var(londonHousingt.bc12)
print("Variance of transformed data, differenced at lags 12 and 1:")
var(londonHousingt.bc12_1)
par(mfrow=c(1,2))
hist(londonHousingt.bc, col="light green", xlab="", main="histogram of bc(U_t)")
hist(londonHousingt.bc12_1, col="light green", xlab="", main="histogram of bc(U_t), diff at 12&1")
var(londonHousingt.bc12_1) #variance of bc(U_t), diff at 12 and 1
londonHousingt.bc12_2<- diff(londonHousingt.bc12_1, lag=1) #diffferencing again
var(londonHousingt.bc12_2) #variance of bc(U_t), diff at 12 and 2
par(mfrow=c(1,2))
acf(londonHousingt.bc12_1, lag.max = 40) #acfs of bc(U_t), diff at 12 and 1
pacf(londonHousingt.bc12_1, lag.max = 40) #pacfs of bc(U_t), diff at 12 and 1
#creating vectors of SMA aiccs with Q=1 and Q=3
SMAaiccsQ1<-c(0,0,0)
SMAaiccsQ3<-c(0,0,0)
for (q in 1:3) #loop to fill vector; Q=1, q=1, 2, 3 AICCs
{
  SMAaiccsQ1[q] <- AICc(arima(londonHousingt.bc, order=c(0,1,q), seasonal = list(order =c(0,1,1), perio
```

```
}
for (q in 1:3) #loop to fill vector; Q=3, q=1, 2, 3 AICCs
{
  SMAaiccsQ3[q] <- AICc(arima(londonHousingt.bc, order=c(0,1,q), seasonal = list(order =c(0,1,3), perioc
}

SMAaiccsQ1 #outputting vectors of AICCs
SMAaiccsQ3

#checking confidence intervals of model coefficients to determine if any can be fixed at 0
arima(londonHousingt.bc, order=c(0,1,3), seasonal = list(order =c(0,1,1), period = 12), method="ML")
arima(londonHousingt.bc, order=c(0,1,3), seasonal = list(order =c(0,1,3), period = 12), method="ML")
#creating vectors of SAR aiccs with P=1 and P=2
SARaiccsP1<-c(0,0,0,0,0,0)
SARaiccsP2<-c(0,0,0,0,0,0)
#loop to fill vectors with P=1, p=1,2,3,4,5,6 AICCs
for (p in 1:6)
{
  SARaiccsP1[p] <- AICc(arima(londonHousingt.bc, order=c(p,1,3), seasonal = list(order =c(1,1,1), perioc
}


##calculating AICCs for P=2, p=1,2,3,4,5, and 6 AICCs, by hand as error encountered in the calculation
SARaiccsP2[1] <- AICc(arima(londonHousingt.bc, order=c(1,1,3), seasonal = list(order =c(2,1,1), period =
SARaiccsP2[2] <- AICc(arima(londonHousingt.bc, order=c(2,1,3), seasonal = list(order =c(2,1,1), period =
SARaiccsP2[3] <- AICc(arima(londonHousingt.bc, order=c(3,1,3), seasonal = list(order =c(2,1,1), period =
SARaiccsP2[5] <- AICc(arima(londonHousingt.bc, order=c(5,1,3), seasonal = list(order =c(2,1,1), period =
SARaiccsP2[6] <- AICc(arima(londonHousingt.bc, order=c(6,1,3), seasonal = list(order =c(2,1,1), period =


SARaiccsP1
SARaiccsP2

#checking confidence intervals of model coefficients to determine if any can be fixed at 0
arima(londonHousingt.bc, order=c(1,1,3), seasonal = list(order =c(1,1,1), period = 12), method="ML")
arima(londonHousingt.bc, order=c(2,1,3), seasonal = list(order =c(1,1,1), period = 12), method="ML")
AICc(arima(londonHousingt.bc, order=c(1,1,3), seasonal = list(order =c(1,1,1), period = 12), transform.p
AICc(arima(londonHousingt.bc, order=c(1,1,3), seasonal = list(order =c(1,1,1), period = 12), transform.p
AICc(arima(londonHousingt.bc, order=c(1,1,3), seasonal = list(order =c(1,1,1), period = 12), transform.p
AICc(arima(londonHousingt.bc, order=c(2,1,3), seasonal = list(order =c(1,1,1), period = 12), transform.p
AICc(arima(londonHousingt.bc, order=c(2,1,3), seasonal = list(order =c(1,1,1), period = 12), transform.p
AICc(arima(londonHousingt.bc, order=c(2,1,3), seasonal = list(order =c(1,1,1), period = 12), transform.p
AICc(arima(londonHousingt.bc, order=c(2,1,3), seasonal = list(order =c(1,1,1), period = 12), transform.p
AICc(arima(londonHousingt.bc, order=c(2,1,3), seasonal = list(order =c(1,1,1), period = 12), transform.p
AICc(arima(londonHousingt.bc, order=c(2,1,3), seasonal = list(order =c(1,1,1), period = 12), transform.p
fit1<-arima(londonHousingt.bc, order=c(1,1,3), seasonal = list(order =c(0,1,1), period = 12), method="MI
fit2<-arima(londonHousingt.bc, order=c(2,1,3), seasonal = list(order =c(0,1,1), period = 12), transform
print("fit1:")
fit1
print("fit2:")
fit2
source("plot.roots.R")
plot.roots(NULL, polyroot(c(1, 0.1368, 0.1186, -0.7773)), main="fit1: roots of MA part, nonseasonal")
```

```r
source("plot.roots.R")
par(mfrow=c(1,2))
plot.roots(NULL,polyroot(c(1, 0.2597, 0.1958, -0.8926)), main="fit2: roots of MA part, nonseasonal")
plot.roots(NULL,polyroot(c(1, 0, 0.0786)), main="fit2: roots of AR part, nonseasonal")
print("fit2; roots of AR part, nonseasonal:")
polyroot(c(1, 0, 0.0786))
res1<-residuals(fit1)
m <- mean(res1)
par(mfrow=c(2,2))
plot.ts(res1, main="Residuals of fit1, time series")
abline(h=m, col="green")
hist(res1, breaks=20, col="light green", xlab="", prob=TRUE, main="Residuals of fit1, histogram")
qqnorm(res1, main= "Normal Q-Q Plot for Residuals of fit1")
qqline(res1, col="green")
print("Mean of residuals of fit1:")
m
par(mfrow=c(1,2))
acf(res1, main="ACF of res1")
pacf(res1, main="PACF of res1")
shapiro.test(res1)
Box.test(res1, lag=17, type = c("Box-Pierce"), fitdf = 5)
Box.test(res1, lag=17, type = c("Ljung-Box"), fitdf = 5)
Box.test((res1)^2, lag=17, type = c("Ljung-Box"), fitdf = 0)
ar(res1, aic = TRUE, order.max = NULL, method = c("yule-walker"))

#transformed data prediction
pred.tr <- predict(fit1, n.ahead = 12)
U.tr= pred.tr$pred + 2*pred.tr$se
L.tr= pred.tr$pred - 2*pred.tr$se
ts.plot(londonHousingt.bc, xlim=c(1,length(londonHousingt.bc)+12), ylim = c(min(londonHousingt.bc), max
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(londonHousingt.bc)+1):(length(londonHousingt.bc)+12), pred.tr$pred, col="red")

#defining inverse box-cox to undo box-cox transformation
invBoxCox <- function(x, lambda)
            if (lambda == 0) exp(x) else (lambda*x + 1)^(1/lambda)

#original data prediction
pred.orig <- invBoxCox(pred.tr$pred, lambda)
U= invBoxCox(U.tr, lambda)
L= invBoxCox(L.tr, lambda)
ts.plot(londonHousingt, xlim=c(1,length(londonHousingt)+12), ylim = c(min(londonHousingt),max(U)), xlab=
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(londonHousingt)+1):(length(londonHousingt)+12), pred.orig, col="red")

#zoomed in on original data prediction
ts.plot(londonHousingt, xlim = c(200,length(londonHousingt)+12), ylim = c(288,max(U)), xlab="Time (Montl
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(londonHousingt)+1):(length(londonHousingt)+12), pred.orig, col="red")
```

```
#zoomed forecasts with test data
ts.plot(londonH$average_price, xlim = c(200,length(londonHousingt)+12), ylim = c(288,max(U)), col="purpl
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(londonHousingt)+1):(length(londonHousingt)+12), pred.orig, col="red")
points(289:301, londonHousing_test, col="black")
```