PSTAT 174 Final Project
Median Sales Price of Houses Sold for the United States

Yanjie Qi

03/06/2019

Abstract

The House Price is usually one of the significant factor that directly influence people's life, and predicting a approriate time to invest or sell would be also crucial. The Dataset used in this project is the Median Sales Price of House Sold for the United States from the first quarter of 1963 to the last quarter of 2009. I am interested in looking for a time series model to fit the data so that I am able to predict the median House Price in the future. By using the techniques like Box-cox transformation, differencing, ACF, PACF, AICc,etc, I locked the final model on two candidate model: SARIMA(3,1,0)x(0,1,1)_4 (Model 1) and SARIMA(1,1,3)x(0,1,1)_4 (Model 2). After dignostic check and residual tests, both models work, but I chose the Model 1 as my final model. Although in the end, only about 1/3 test datasets is consistent with the model, it is still considered the right model based on the traning datasets. Therefore, SARIMA(3,1,0)x(0,1,1)_4 is the final model for the Median Sales Price of House in United States.

Introduction and Data Sources

Being interested in looking for a time series model to fit the data so that we would be able to forecast the median House Price in the future, I hope the model would help more people make better decision for the house they are likelyn to invest or their house for sell. The final result for the model was reasonable and sucessful but not exactly the consistent with the actual test data because of the financial crisis happened in 2008. Therefore, the model might not be employed to forecast the price in finanicial crisis time but would be most likely depict the overall trend without the financial crisis.

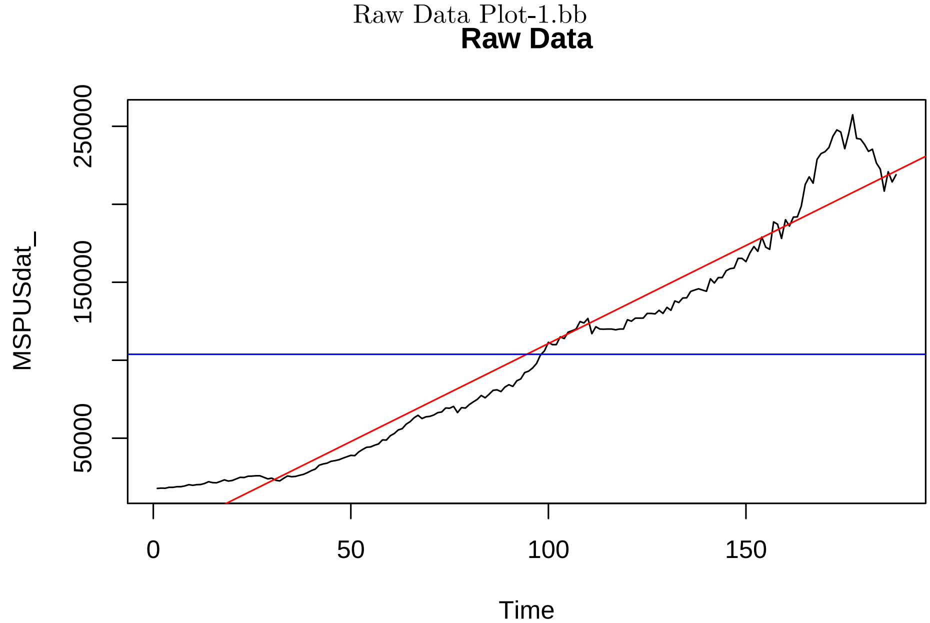
The Data used is from U.S. Census Bureau and U.S. Department of Housing and Urban Development, retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/MSPUS, March 06, 2020.

- 'DATE': The time interval that reflects the median house price. There are four quarters recorded for each year, for example, "1963-01-01" stands for the first quarter of year 1964.
- 'MSPUS': Median Sales Price of Houses Sold for the United States for the given Date, in dollars.

The Software used is the newest version of R studio until 03/14/2020.

Data Exploration

Raw Data Plot-1.bb



We could immediately notice that it is highly non-stationary and has obvious linear trend. Moreover, the variance and mean are non-constant. There is sharp change of behavior around t = 180 and possiblily seasonality.

For testing the feasibility of the model, I chose the data from 1963 to 2006 as the training dataset and data from 2007 to 2010 as the test dataset.

Plot of traning dataset time series:

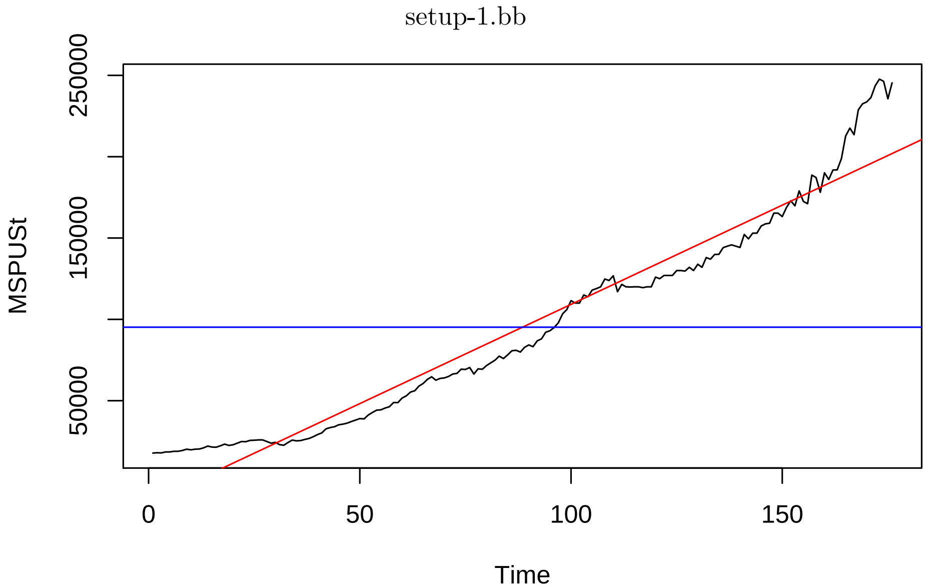
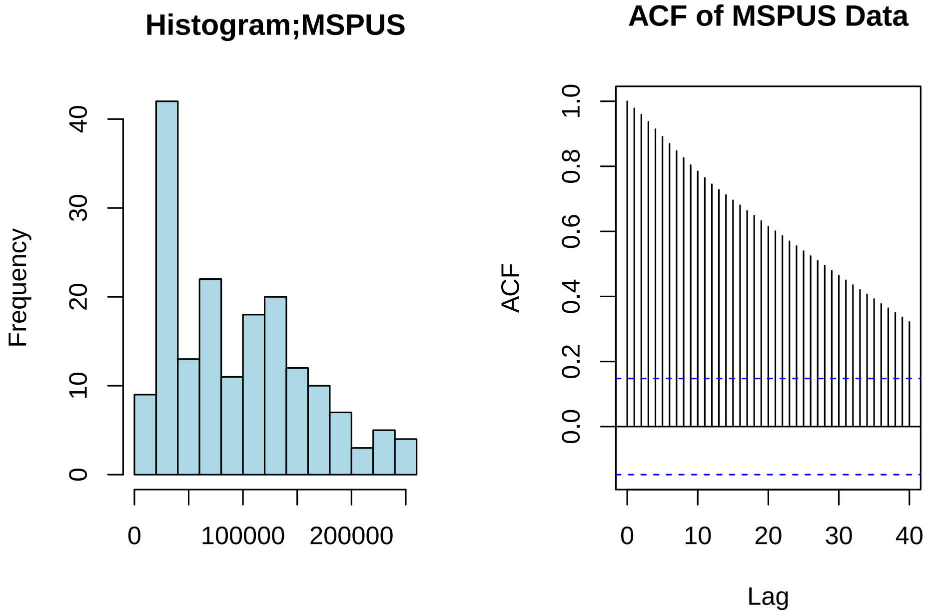


Figure 1: Selected Training Median Sales of House Sold

To understand better on the data, I plotted the histogram of training Dataset and found that it is badly skewed and variance is not constant. For the ACFs of the dataset, it remains large.

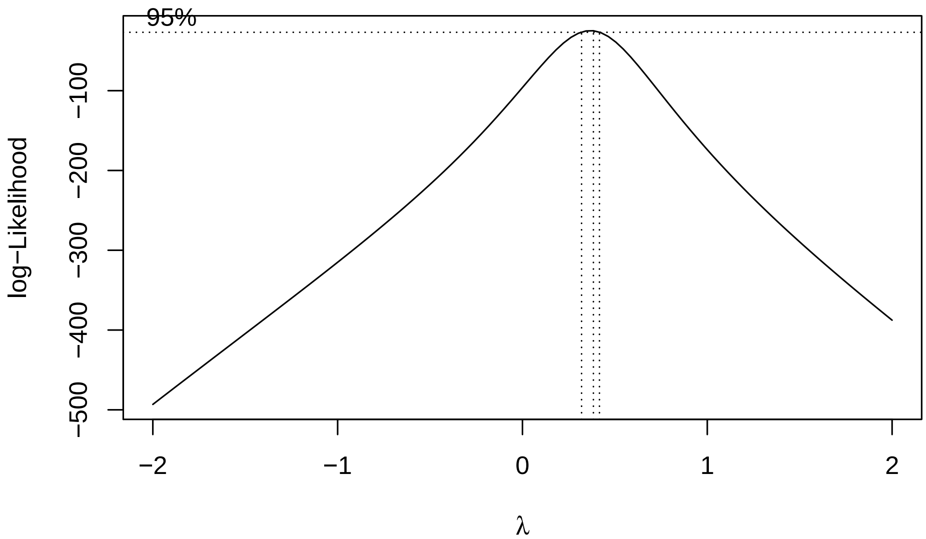
dataset analysis-1.bb



To stablize the variance, I used Box-Cox transformation and got the following value of lambda from the plot.

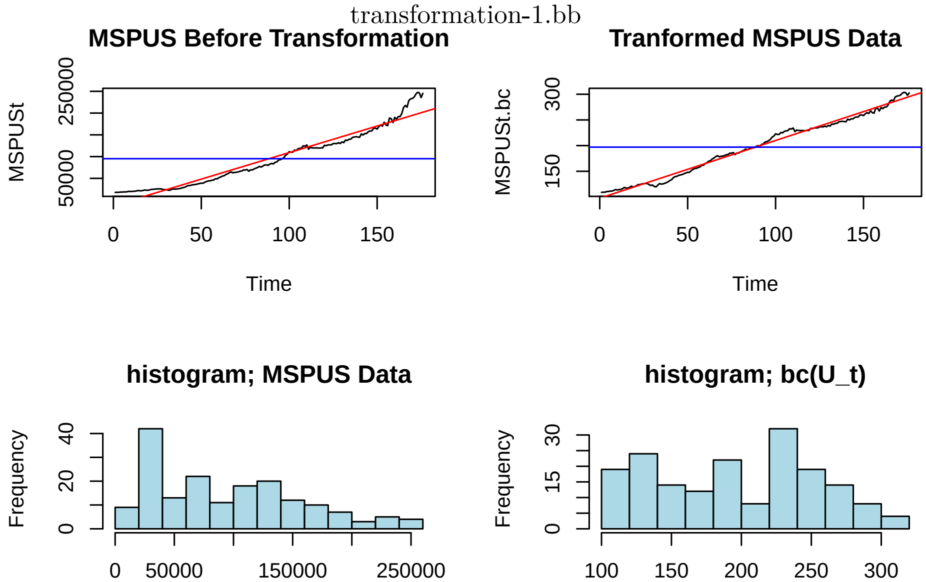
```
## [1] "lambda="           "0.383838383838384"
```

transformation-1.bb

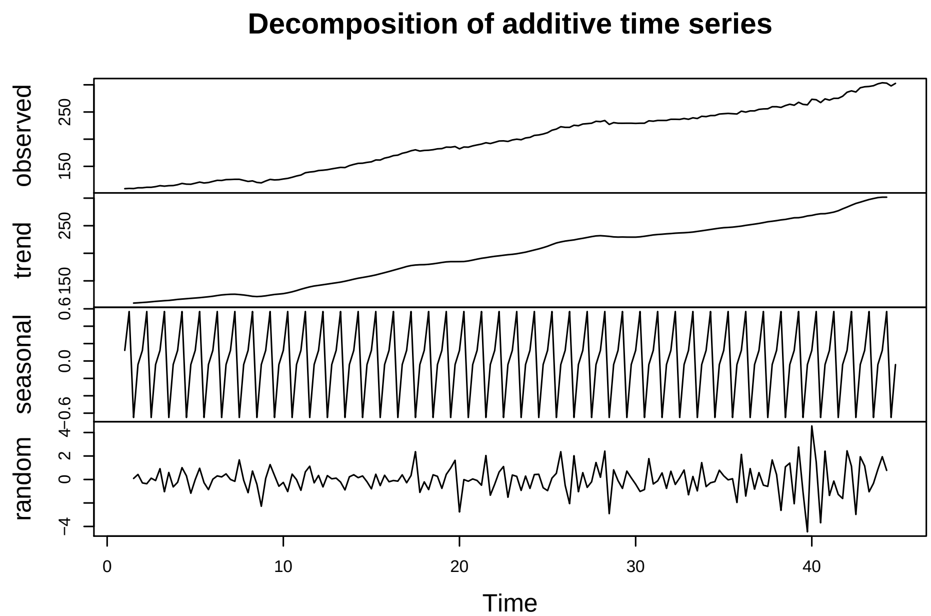


After getting the lambda, I did the Box-cox transformation and plotted the transformed data and its histogram. From the plot, variance is more stable after transformation.

transformation-1.bb



I plotted the Decomposition of bc(Ut) and it shows that there is seasonality and almost linear trend, so it needs to be differenced.



To remove seasonality shown in the decomposition plot, I differenced at lag 4 first since it is quarterly distributed data so it should have period = 4, and then the variance went down. Then I differenced at lag 1 to remove its trend and the variance still goes down, meaning that the differencings were successful.

```
## [1] "variance of bc(U_t) = " "3317.31268264939"
```

```
## [1] "variance of bc(U_t) after differencing at lag 4 ="
## [2] "13.4347936400725"
```

```
## [1] "variance of bc(U_t) after differencing at lag 4 and lag 1 = "
## [2] "9.46400469552432"
```

After transformation and differencing at lag4 and lag1, the plot is stationary now.

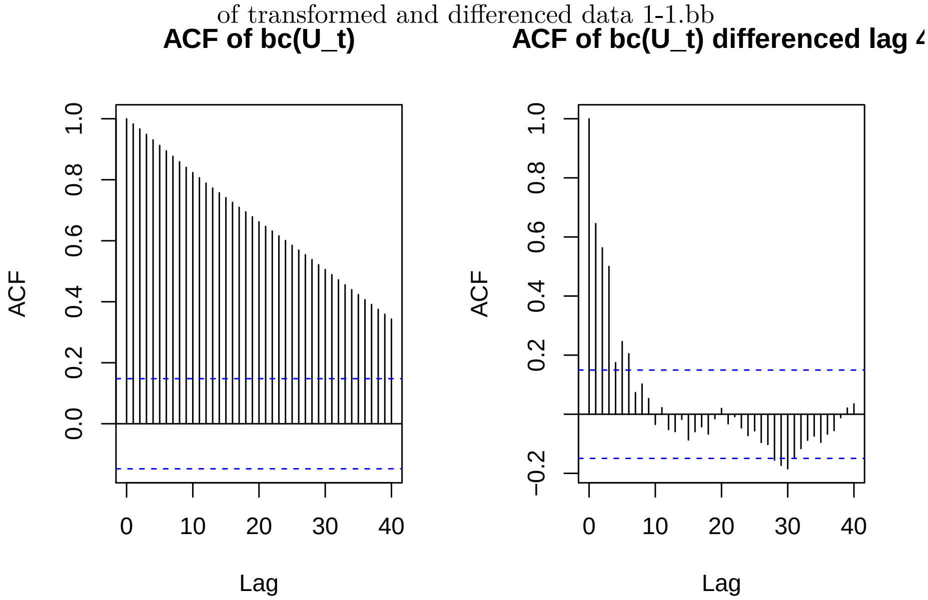
- To see if I need to difference again to remove the trend, I took the difference again at lag 1 and checked its variance. Since the variance went up, I did not choose to difference again the dataset to be the final data.

```
## [1] "variance of bc(U_t) after differencing at lag 4 and lag 1 twice = "
## [2] "26.4748423048931"
```

Model Selection

To see the effect of differencing and also get to know the MA part of the model, I plotted acf of data in the three phases: 1) just after transformation 2) after transformation and take the difference at lag 4

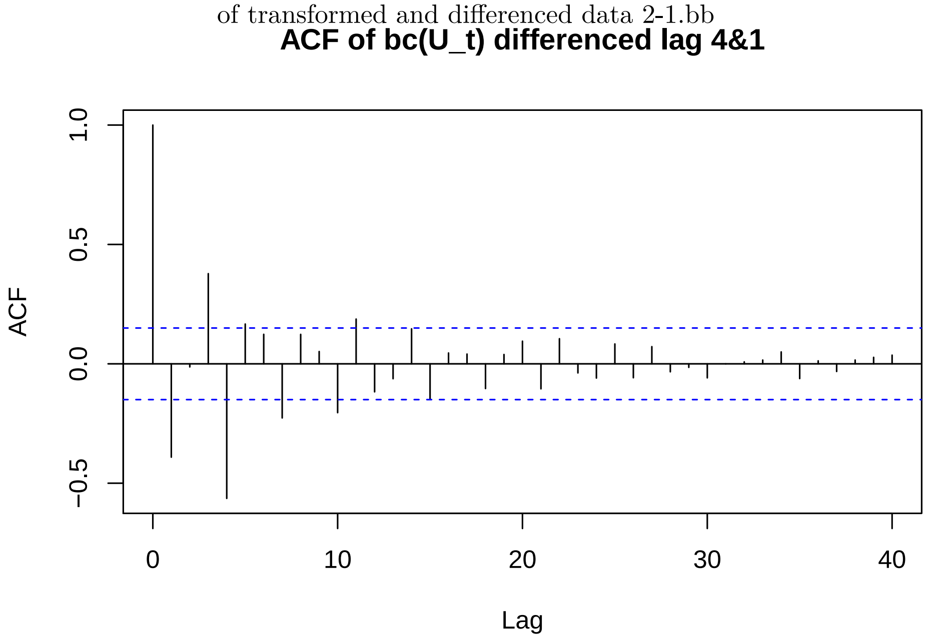
of transformed and differenced data 1-1.bb



3) after transformation, difference at lag 4 and difference at lag 1.

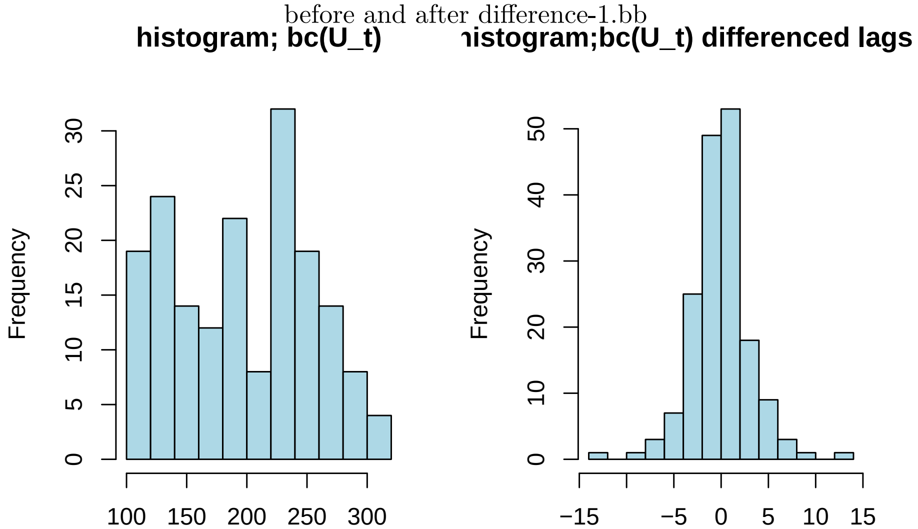
Notice that in the ACF plot of transformed and differenced data, Lags 1,3,4,7,10,11 are ACFs outside of the confidence interval.

of transformed and differenced data 2-1.bb



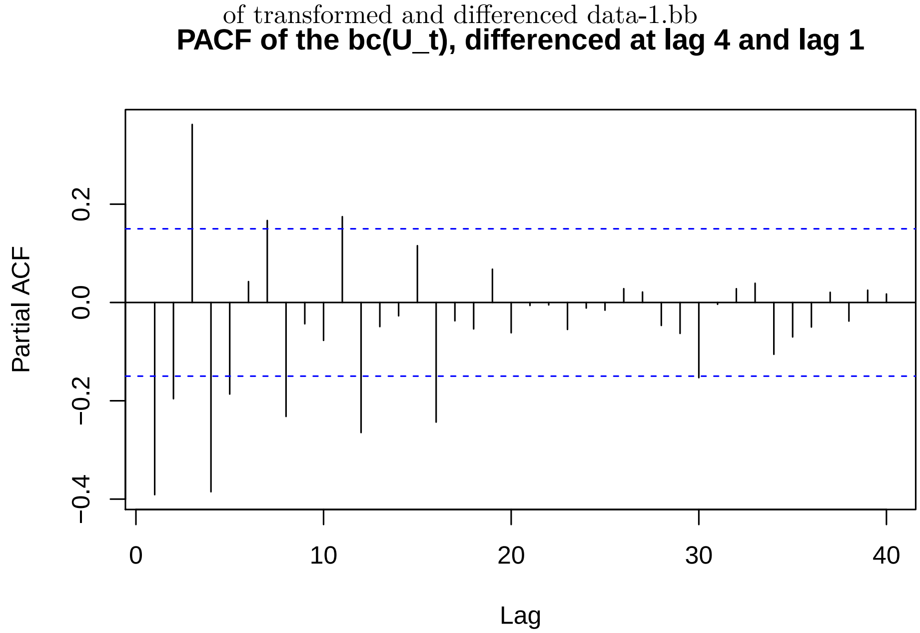
To see if it is more Gaussian and symmetric, I plotted histogram of the transformed and differenced data and the transformed data to compare:

before and after difference-1.bb



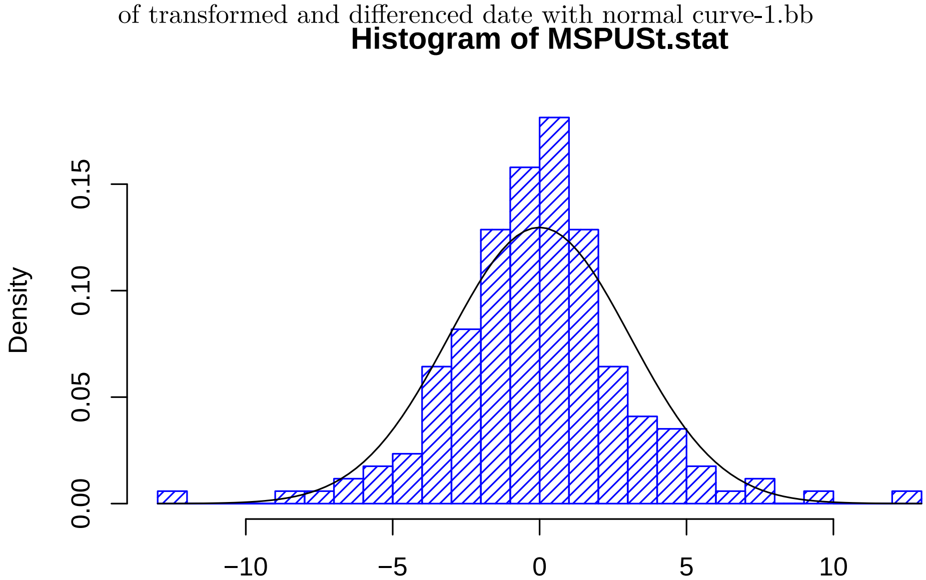
Also, I plotted the PACF plot of transformed and differenced data. From the PACF, lags 1,2,3,4,5,7,8,11,12,16 are outside of the confidence intervals.

of transformed and differenced data-1.bb



To see the normality, plotted the histogram of transformed and differenced data with normal curve. The plot shows nearly normal feature and acceptable difference.

of transformed and differenced date with normal curve-1.bb



According to the ACF and PACF plot of transformed and differenced data and the analysis of lags in ACF and PACF, I had the some candidate model with following potential suppositions:

SARIMA for bc(U_t): s=4, D=1, d=1; (Based on what I did to the dataset)

Q= 1; (Since in ACF, only lag 4, which is multiple of 4, is outside of confidence interval)

P=1,2,3,4; (Since in PACF, lag 4,8,12,16, which are multiple of 4, are all outside of the confidence interval. Then for the seasonal AR part, 1,2,3,4 are all possible for the model)

q=1,3,4,7; (Among the lags outside of confidence interval in ACF, choose the order that smaller than 10 to test its AICc.)

p=1,2,3,4,5,6,7,8; (Among the lags outside of confidence interval in PACF, choose the order that smaller than 10 to test its AICc.)

Possibly AR(16) and MA(11). (SInce lag 11 and lag 16 the largest lags outside of confidence interval in ACF and PACF)

After checking the AICc of AR(16) and MA(11), I got AICc of MA(11) = 747.791 and AICc of of AR(16) = 762.6107, in which MA(11) could be potential model compared to AR(16).

To check the AICc of candidate model, I sat up the for loop to show AICc and compare them to each other to get the most likely model.

Throughout the calculation of AICC by using for loop, I had two condidate model: SARIMA(3,1,0)x(0,1,1)_4 (Model 1) and SARIMA(1,1,3)x(0,1,1)_4 (Model 2).

```
##
## Call:
## arima(x = MSPUSt.bc, order = c(3, 1, 0), seasonal = list(order = c(0, 1, 1),
##     period = 4), method = "ML")
##
## Coefficients:
##           ar1     ar2     ar3     sma1
##       -0.2649  0.0501  0.2899  -0.8528
## s.e.   0.0739  0.0778  0.0749   0.0488
##
## sigma^2 estimated as 4.009:  log likelihood = -364.33,  aic = 738.67
```

```
## [1] "AICc for Model 1 =" "738.900816668061"
```

For Model 1, I fixed the insiginificant coeffient and set it to 0 and got the AICc for the Revised Model 1 = 737.3149.

```
##
## Call:
```

```
## arima(x = MSPUSt.bc, order = c(1, 1, 3), seasonal = list(order = c(0, 1, 1),
##     period = 4), method = "ML")
##
## Coefficients:
##           ar1     ma1      ma2     ma3     sma1
##       -0.7576  0.5171  -0.1056  0.3082  -0.8244
## s.e.   0.0821  0.1019   0.0850  0.0781   0.0571
##
## sigma^2 estimated as 3.963:  log likelihood = -363.98,  aic = 739.95

## [1] "AICc for Model 1 =" "740.305646272151"
```

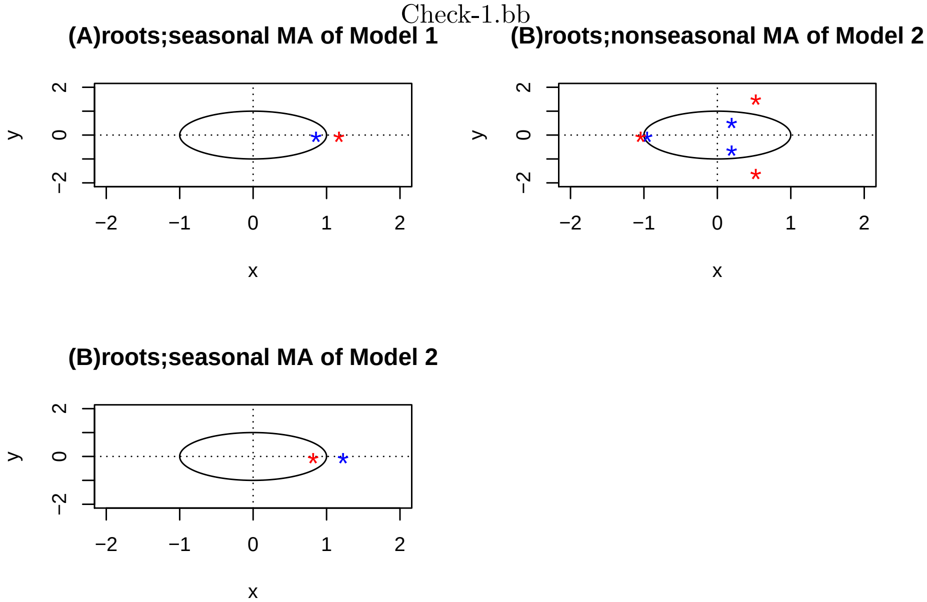
For Model 2, I also fixed the insiginificant coeffient and set it to 0 and got the AICc for the Revised Model 1 = 739.7557

Preliminary Validation of Model Rationality

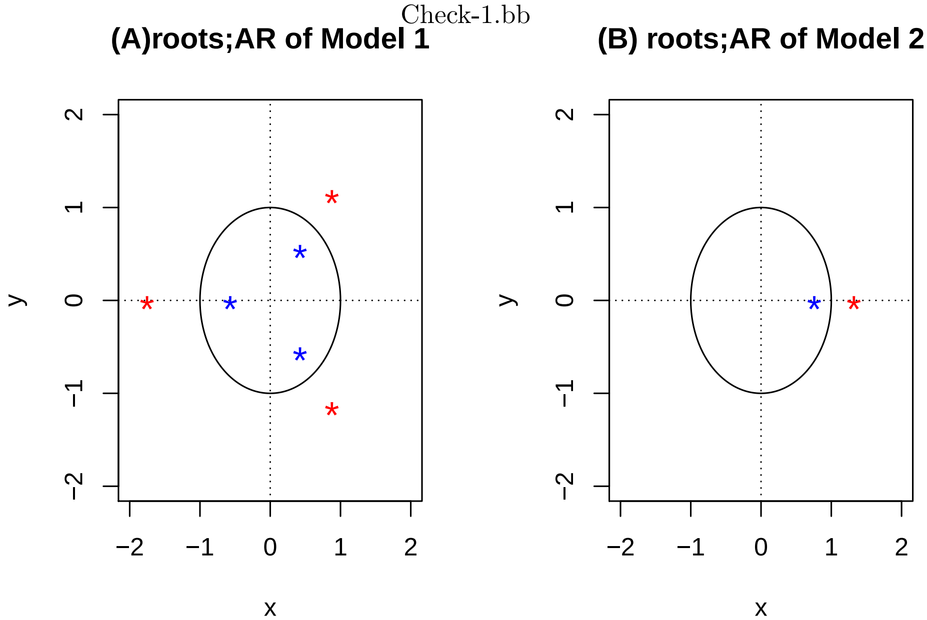
–Invertibility and Causality Checking

To check whether the model is invetible and causal, I plotted the roots of different part to see if it is in the unit circle.

Check-1.bb



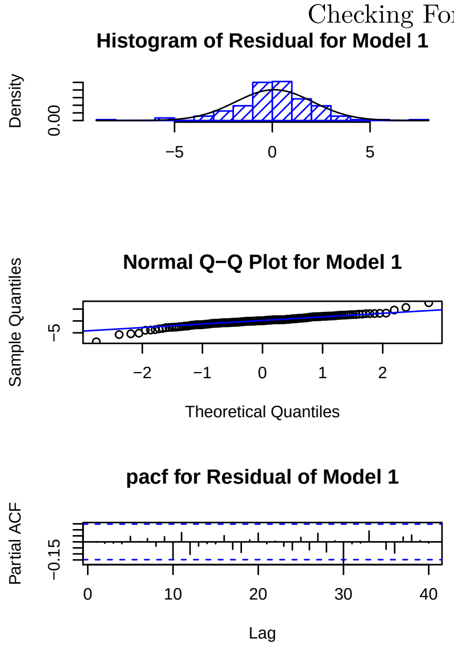
Check-1.bb



Based on the plots above, both Revised Model 1 is invertible and causal since all roots are outside the unit circles, but Model 2 is not invertible since the its seasonal MA root is in the unit circle.

Dignostic Checking

Checking For Model 1-1.bb



From the plots for the Model 1 above, there is no trend, no visible change of variance, no seasonality. Sample mean is almost zero. Histograms and Q-Q plots look approriate. All acf and pacf of residuals are within confidence intervals or can be counted as zeros.

Then, move to the test part where I used Shapiro-Wilk normality test, Box-Pierce test, Box-Ljung test, and Mc-Leod Li test:

Model 1:

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.95512, p-value = 2.13e-05
```

```
##
##  Box-Pierce test
##
## data:  res
```

```
## X-squared = 7.1401, df = 8, p-value = 0.5216
```

```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 7.6664, df = 8, p-value = 0.4667
```

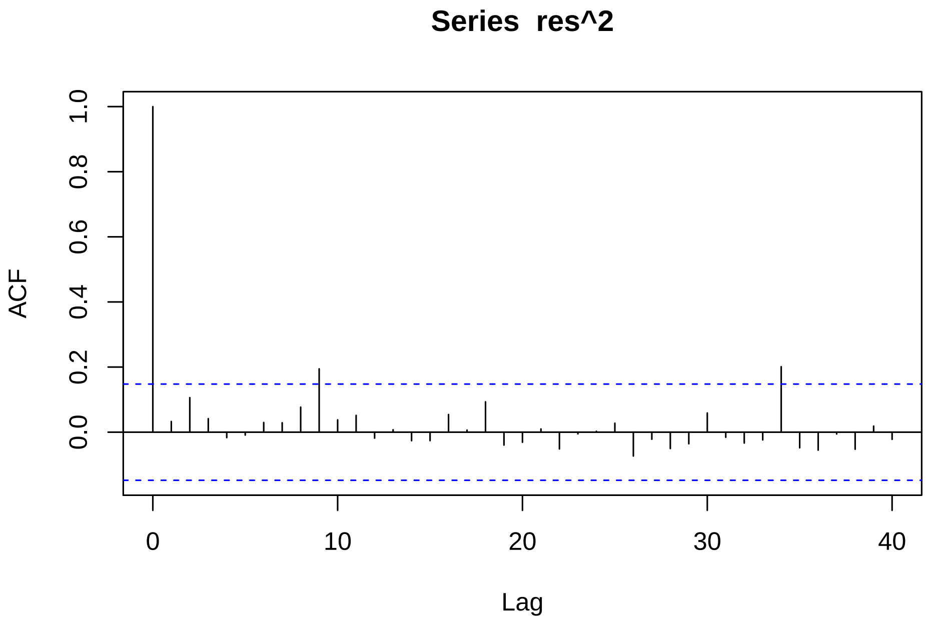
```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 11.993, df = 13, p-value = 0.5282
```

From the tests for Model 1, only p-value for shapiro-test is < 0.05, which means that the model has enough data to confidently see that the residuals are not sampled from a normal distribution; however, it is tolerable and does not prevent us from using the model.

Run the AR Parameter estimation for the residual of Model 1 and found that the it fitted residuals of Model 1 to AR(0), i.e. WN.

```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  3.908
```

1 Yuler-walker-1.bb



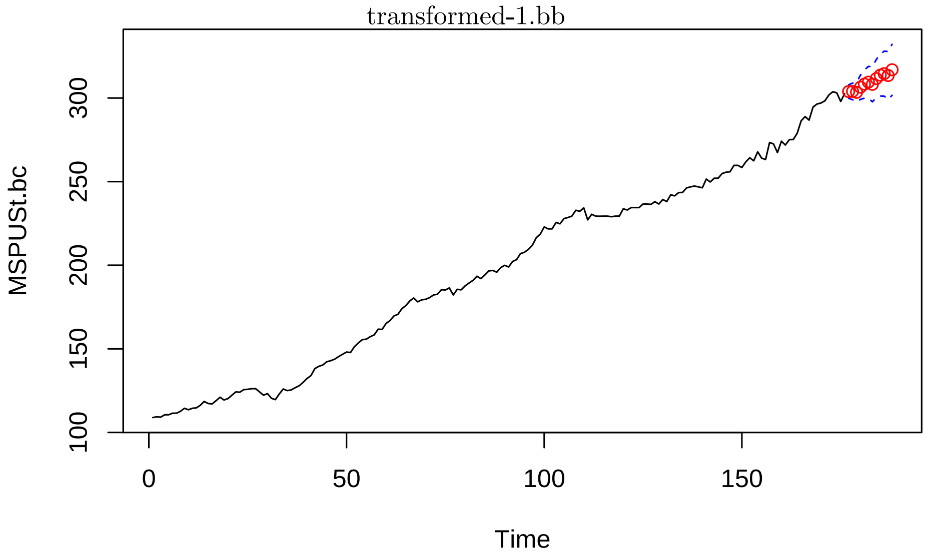
Therefore, Model 1 passed the disnostic checking. Since the Model 2 is not invertible, only Model 2 can be my final Model. The test above proved that it is ready to be used to forecast.

Forecast by the Final Model

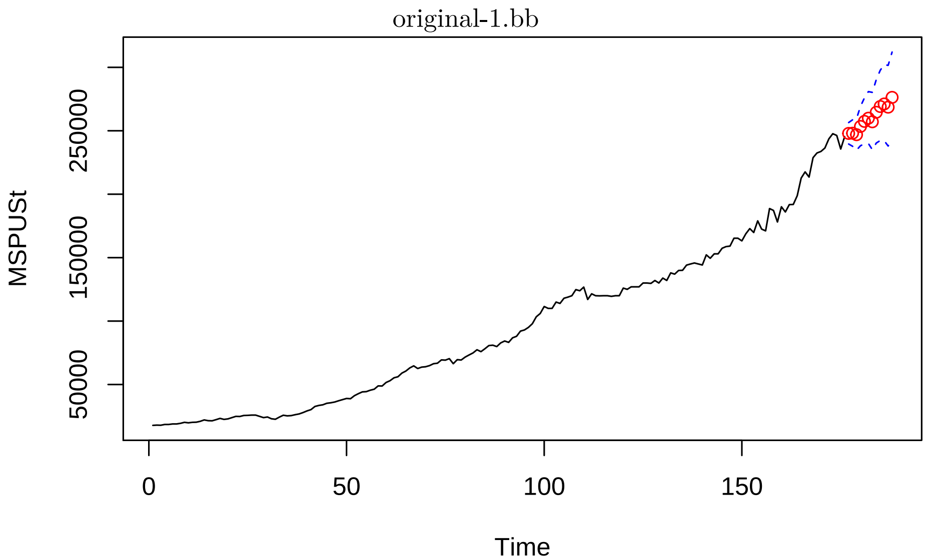
The Final Model for the Box-Cox transform of the original Data: bc(U_t) follows SARIMA(3,1,0)x(0,1,1)_4 model

1 4(1-0.2766*B+0.2753B^3)bc(U_t) = (1-0.8557B$^{12)Zt\ sigma}$2 = 4.016

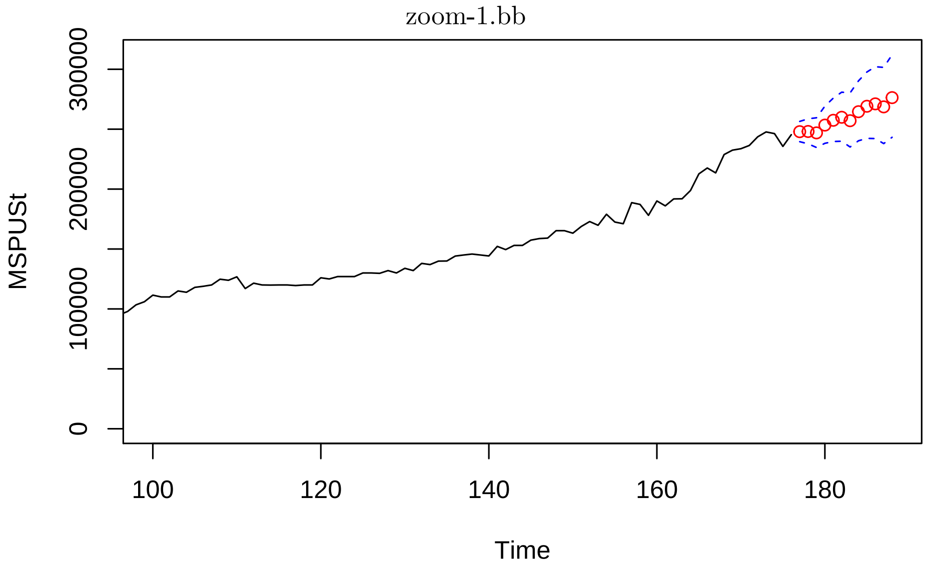
Then we have the forecast of transformed data using the Final Model:



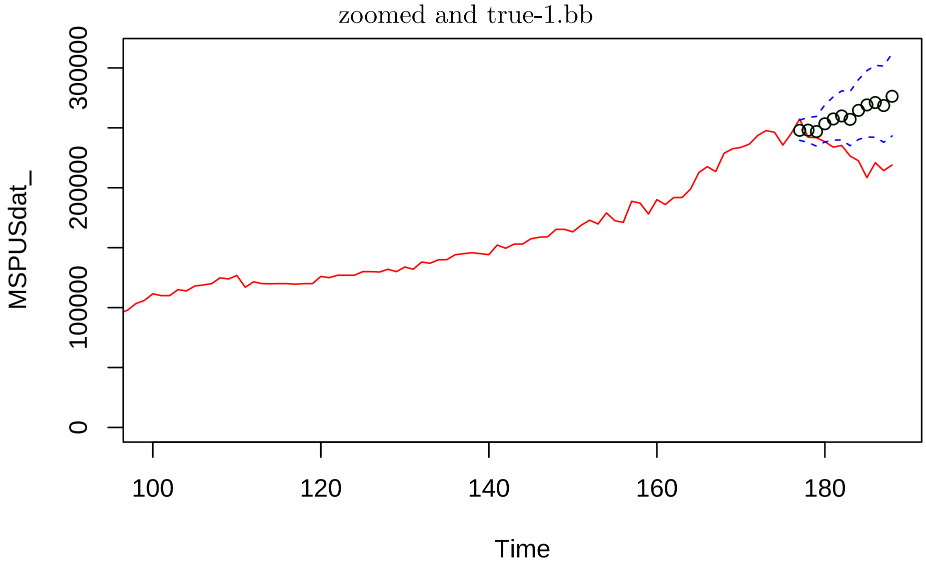
Forecast of original data using the Final Model:



The following is the zoomed forecast of original Data using the Final Model:



Forecast of original Data and the test set:



Conclusion

From the plot, about 1/3 of the test set are in the prediction intervals. Considering the fact that there was a great recession in 2008 and it is reasonable that there was a sudden decreasing of the median house price in United States. However, the prediction values are approriate and reasonable based on the given training datasets. Thus, SARIMA(3,1,0)x(0,1,1)_4 is the feasible and factual model for the Median Sales Price of Houses Sold for the United States if there was no financial crisis in 2008 but not applied the actual data because of the crisis reason.

Also, Thanks to Dr. Raya Feldman for the instructions and advices in the process of producing this project.

- Math Formula used: MSPUSt.bc = (1/lambda)*(MSPUSt^lambda-1) [Box-Cox Transfomation Formula]

Reference

- U.S. Census Bureau and U.S. Department of Housing and Urban Development, Median Sales Price of Houses Sold for the United States [MSPUS], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/MSPUS, March 06, 2020.
- Winter 2020 PSTAT 174 Lecture Slides, retrieved from Gauchospace, Dr. Raya Feldman, March 06, 2020.

Appendix: R Code

```
knitr::opts_chunk$set(echo = TRUE,fig.pos = 'H')
# install.packages("ggplot2")
# install.packages("ggfortify")
library(ggplot2)
library(ggfortify)
# install.packages("qpcR")
library(qpcR)
library(forecast)

# r data_organizing
## Read Data
MSPUS <- read.csv("~/Desktop/Winter 2020/PSTAT 174/Final Project/MSPUS.csv")
## Change to ts data type
MSPUSdat <- ts(data = MSPUS$MSPUS, start = 1963, end = c(2019,04), frequency = 4)
## Extract the object data
MSPUSdat_ <- MSPUSdat[c(1:188)]
```

```
# r raw data presentation
# raw data plot
plot.ts(MSPUS$MSPUS)
nt <- length(MSPUS$MSPUS)
# add trend to the data plot
fit <- lm(MSPUS$MSPUS ~ as.numeric((1:nt))); abline(fit,col="red")
# add mean to the data plot
mean(MSPUS$MSPUS) #134892.1
abline(h=mean(MSPUS$MSPUS), col="blue")

# r Object Raw Data Plot
# Object Raw Data Plot
ts.plot(MSPUSdat_, main="Raw Data")
fit <- lm(MSPUSdat_~as.numeric(1:length(MSPUSdat_)));abline(fit,col="red")
abline(h=mean(MSPUSdat_),col="blue")

# r dataset setup
# training dataset
MSPUSt = MSPUS$MSPUS[c(1:176)]
# test dataset
MSPUS.test = MSPUS$MSPUS[c(177:188)]
# plot of training dataset
plot.ts(MSPUSt)
fitt <- lm(MSPUSt~as.numeric(1:length(MSPUSt))); abline(fitt,col="red")
abline(h=mean(MSPUSt),col="blue")

# r tranning dataset analysis
# plots histogram of training datasets
hist(MSPUSt,col="light blue",xlab = "",main = "histogram;Median Sales Price of Houses Sold I
# plots ACF
acf(MSPUSt,lag.max = 40,main="ACF of the Median Sales Price of Houses Sold Data")
# pacf(MSPUSt,lag.max = 40,main="PACF of the Median Sales Price of Houses Sold Data")

# r boxcox transformation
# To choose paramater lambda of the Box-Cox transformation for
# dataset MSPUSt
# PLot the graph
require(MASS)
bcTransform <- boxcox(MSPUSt~as.numeric(1:length(MSPUSt)))
# gives the value of
print(c("lambda=",bcTransform$x[which(bcTransform$y==max(bcTransform$y))]))
lambda=bcTransform$x[which(bcTransform$y == max(bcTransform$y))]

# r data transformation
# plot of training dataset
```

```
plot.ts(MSPUSt, main="Median Sales Price of House Sold Data Before transformation")
fitt <- lm(MSPUSt~as.numeric(1:length(MSPUSt))); abline(fitt,col="red")
abline(h=mean(MSPUSt),col="blue")

# Perform transformations, plot transformed data, histograms
MSPUSt.bc = (1/lambda)*(MSPUSt^lambda-1)
plot.ts(MSPUSt.bc, main="Tranformed Median Sales Price of Houses Sold Data")

# Compare Histogram
# plots histogram of training datasets
hist(MSPUSt,col="light blue",xlab = "",main = "histogram;Median Sales Price of Houses Sold D
# after transformation
hist(MSPUSt.bc,col = "light blue",xlab = "",main = "histogram; bc(U_t)")

# r decomposition
# Produce decomposition of bc(U_t)
y1 <- ts(as.ts(MSPUSt.bc), frequency = 4)
decomp1 <- decompose(y1)
plot(decomp1)

# r differencing
# Differencing at lag 4 to remove its seasonality
print(c("variance of bc(U_t) = ",var(MSPUSt.bc))) # 3317.313
MSPUSt.bc_4 <- diff(MSPUSt.bc, lag = 4)
plot.ts(MSPUSt.bc_4, main ="bc(U_t) differenced at lag 4")
fit_4 <- lm(MSPUSt.bc_4~as.numeric(1:length(MSPUSt.bc_4))); abline(fit_4,col="red")
# mean(MSPUSt.bc_4) # 4.47357
abline(h=mean(MSPUSt.bc_4),col="blue")
print(c("variance of bc(U_t) after differencing at lag 4 =",var(MSPUSt.bc_4))) # 13.43479

# # Differencing again at lag4
# MSPUSt.bc_4_a <- diff(MSPUSt.bc_4, lag = 4)
# plot.ts(MSPUSt.bc_4_a, main ="bc(U_t) differenced twice at lag 4")
# fit_4_a <- lm(MSPUSt.bc_4_a~as.numeric(1:length(MSPUSt.bc_4_a))); abline(fit_4_a,col="red
# mean(MSPUSt.bc_4_a)
# abline(h=mean(MSPUSt.bc_4_a),col="blue")
# var(MSPUSt.bc_4_a)

# Differencing MSPUSt.bc at lag 1
MSPUSt.stat <- diff(MSPUSt.bc_4, lag = 1)
plot.ts(MSPUSt.stat, main="bc(U_t) differenced at lag4 and lag 1")
fit_stats <- lm(MSPUSt.stat~as.numeric(1:length(MSPUSt.stat))); abline(fit_stats, col="red")
# mean(MSPUSt.stat) # -0.004689219
abline(h=mean(MSPUSt.stat),col="blue")
print(c("variance of bc(U_t) after differencing at lag 4 and lag 1 = ",var(MSPUSt.stat))) #
```

```
# r more differencing on trend
# Differencing again to see if needed
MSPUSt.stat_a <- diff(MSPUSt.stat, lag = 1)
plot.ts(MSPUSt.stat_a, main="bc(U_t) differenced at lag 4 and twice at lag 1")
fit_stats_a <- lm(MSPUSt.stat_a~as.numeric(1:length(MSPUSt.stat_a))); abline(fit_stats_a, co
# mean(MSPUSt.stat_a) # 0.004699543
abline(h=mean(MSPUSt.stat_a),col="blue")
print(c("variance of bc(U_t) after differencing at lag 4 and lag 1 twice = ",var(MSPUSt.stat
## Dont need to difference again since the variance increase
## after the second differencing

# r ACF of transformed and differenced data
acf(MSPUSt.bc, lag.max = 40, main="ACF of the bc(U_t)")
acf(MSPUSt.bc_4, lag.max = 40, main="ACF of the bc(U_t), differenced at lag 4")
acf(MSPUSt.stat,lag.max = 40,main="ACF of the bc(U_t), differenced at lag 4 and lag 1")
## ACF outside the confidence intervals:
## Lags 1,3,4,7,10,11

# r histogram before and after difference
hist(MSPUSt.bc,col = "light blue",xlab = "",main = "histogram; bc(U_t)")
hist(MSPUSt.stat, col="light blue", xlab="", main="histogram; bc(U_t) differenced at lags 1'

# r PACF of transformed and differenced data
pacf(MSPUSt.stat,lag.max = 40,main="PACF of the bc(U_t), differenced at lag 1")
## PACF outside the confidence intervals:
## Lags 1,2,3,4,5,7,8,11,12,16

# r Histogram of transformed and differenced date with normal curve

# Histogram of transformed and differenced data with normal
# curve
hist(MSPUSt.stat,density = 20, breaks = 20,
     col = "blue",xlab = "",prob=TRUE)
m <- mean(MSPUSt.stat)
std <- sqrt(var(MSPUSt.stat))
curve(dnorm(x,m,std),add=TRUE)

# r Check AICc of AR(16) and MA(11)
# AICc of MA(11) # 747.791
AICc(arima(MSPUSt.bc, order = c(0,1,11), seasonal=list(order=c(0,1,0),period=4),method = "MI

# AICc of AR(16) # 762.6107
AICc(arima(MSPUSt.bc, order = c(16,1,0), seasonal=list(order=c(0,1,0),period=4),method = "MI

# r for loop to get AICc of possible models
```

```
# y <- c(0,1,3,4,7)
# for (i in 0:8) {
#   for (j in y) {
#     for (m in 0:4){
#     print(c("SARIMA(",i,",1,",j,")*(",m,"1,1)"));print(AICc(arima(MSPUSt.bc, order = c(i,
#     }
#   }
# }

# r Model 1 and AICc
# Model 1
arima(MSPUSt.bc, order = c(3,1,0), seasonal=list(order=c(0,1,1),period=4),method = "ML")
print(c("AICc for Model 1 =",AICc(arima(MSPUSt.bc, order = c(3,1,0), seasonal=list(order=c(
# 738.9008

# r Revised Model 1 and AICc
# Revised Model 1
arima(MSPUSt.bc, order = c(3,1,0), seasonal=list(order=c(0,1,1),period=4),
      fixed = c(NA,0,NA,NA),transform.pars = FALSE, method = "ML")
AICc(arima(MSPUSt.bc, order = c(3,1,0), seasonal=list(order=c(0,1,1),period=4),
      fixed = c(NA,0,NA,NA),transform.pars = FALSE, method = "ML")) # 737.3149

# r Model 2 and AICc
# Model 2
arima(MSPUSt.bc, order = c(1,1,3), seasonal=list(order=c(0,1,1),period=4),method = "ML")
print(c("AICc for Model 1 =",AICc(arima(MSPUSt.bc, order = c(1,1,3), seasonal=list(order=c(
# 740.3056

# r Revised Model 2 and AICc
# Revised Model 2
arima(MSPUSt.bc, order = c(1,1,3), seasonal=list(order=c(0,1,1),period=4),
      fixed=c(NA,NA,0,NA,NA),transform.pars = FALSE, method = "ML")
AICc(arima(MSPUSt.bc, order = c(1,1,3), seasonal=list(order=c(0,1,1),period=4),
      fixed=c(NA,NA,0,NA,NA),transform.pars = FALSE, method = "ML")) #739.7557

# r Invetibility Check
# Check their invertibility

# For Model 1  SARIMA(3,1,0)*(0,1,1)^4
source("plot.roots.R")
plot.roots(NULL,polyroot(c(1,-0.8557)), main="(A) roots of ma part of Model 1, seasonal")

# For Model 2 SARIMA(1,1,3)*(0,1,1)^4
source("plot.roots.R")
plot.roots(NULL,polyroot(c(1,0.5692,0,0.3549)), main="(A) roots of ma part of Model 2, nonse
```

```
plot.roots(NULL,polyroot(c(1,-1.2243)), main="(A) roots of ma part of Model 2, seasonal")

# r Causaility Check
# Chech their causaility

# For Model 1  SARIMA(3,1,0)*(0,1,1)^4
source("plot.roots.R")
plot.roots(NULL,polyroot(c(1,-0.2766,0,0.2753)), main="(A) roots of ar part of Model 1, nons


# For Model 2 SARIMA(1,1,3)*(0,1,1)^4
plot.roots(NULL,polyroot(c(1,-0.7570)), main="(A) roots of ma part of Model 2, nonseasonal")

# r Dignostic Checking For Model 1
# Dignostic Checking for Model 1
# Histogram
fit_dig <- arima(MSPUSt.bc, order = c(3,1,0), seasonal = list(order=c(0,1,1), period=4), met
res <- residuals(fit_dig)
hist(res,density = 20, breaks = 20,col = "blue",xlab = "",prob=TRUE, main = "Histogram of Re
m_res <- mean(res)
std_res <- sqrt(var(res))
curve(dnorm(x,m_res,std_res), add = TRUE)
# Model 1 Plot
plot.ts(res, main="Residual for Model 1")
fit_res <- lm(res~as.numeric(1:length(res))); abline(fit_res, col="red")
abline(h=mean(res),col="blue")
# Model 1 QQ-plot
qqnorm(res,main = "Normal Q-Q Plot for Model 1")
qqline(res,col="blue")
# model 1 ACF and PACF
acf(res,lag.max = 40, main="acf for Residual of Model 1")
pacf(res,lag.max = 40, main="pacf for Residual of Model 1")

# r test for Model 1
# Model 1
shapiro.test(res)
Box.test(res, lag = 13, type = c("Box-Pierce"), fitdf = 5)
Box.test(res, lag = 13, type = c("Ljung-Box"), fitdf = 5 )
Box.test(res^2, lag = 13, type = c("Ljung-Box"), fitdf = 0)

# r Model 1 Yuler-walker
# Model 1 Yuler-walker
acf(res^2,lag.max = 40)
ar(res,aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

```r
# r forecast setup
fit.A <- arima(MSPUSt.bc, order = c(3,1,0), seasonal=list(order=c(0,1,1),period=4),
      fixed = c(NA,0,NA,NA),transform.pars = FALSE, method = "ML")
forecast(fit.A)

# r forecast transformed
# 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
L.tr = pred.tr$pred - 2*pred.tr$se
ts.plot(MSPUSt.bc,xlim=c(1,length(MSPUSt.bc)+12), ylim=c(min(MSPUSt.bc),max(U.tr)))
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(MSPUSt.bc)+1):(length(MSPUSt.bc)+12),pred.tr$pred,col="red")

# r forecast original
# To produce graph with forecasts on original data
pred.orig <- (lambda*pred.tr$pred+1)^(1/lambda) #MSPUSt.bc = (1/lambda)*(MSPUSt^lambda-1)
U = (lambda*U.tr+1)^(1/lambda)
L = (lambda*L.tr+1)^(1/lambda)
ts.plot(MSPUSt,xlim=c(1,length(MSPUSt)+12),ylim=c(min(MSPUSt),max(U)))
lines(U,col="blue",lty="dashed")
lines(L,col="blue",lty="dashed")
points((length(MSPUSt)+1):(length(MSPUSt)+12),pred.orig,col="red")

# r forecast zoomed and true
# To plot zoomed forecasts and true values:
ts.plot(MSPUSdat_,xlim=c(100,length(MSPUSt)+12),ylim=c(250,max(U)),col="red")
lines(U,col="blue",lty="dashed")
lines(L,col="blue",lty="dashed")
points((length(MSPUSt)+1):(length(MSPUSt)+12),pred.orig,col="green")
points((length(MSPUSt)+1):(length(MSPUSt)+12),pred.orig,col="black")
```