Ground-Level Ozone Time Series

yuqingxia

3/1/2022

Abstract:

This report observes the monthly average of ground-level ozone in Los Angeles, California from 2000 - 2020. Using data transformation and differencing, I find out that the original time series does not need transformation, and it has a seasonal pattern but no significant trend. I identified some models to fit by looking at the ACF anf PACF of differentiated time series, the best model with the lowest AICc for the time series is a seasonal ARIMA model with $(p, d, q)\times(P, D, Q) = (1, 0, 1)\times(1, 1, 2)$.The best model past all the diagnostic checking so it is my final model for the time series. Finally, I forecast my model and compare my forecast with the testing set. The test data points shows that my model is good for predicting at least five time units, we will need to constantly update the data to make accurate predictions.

Introduction:

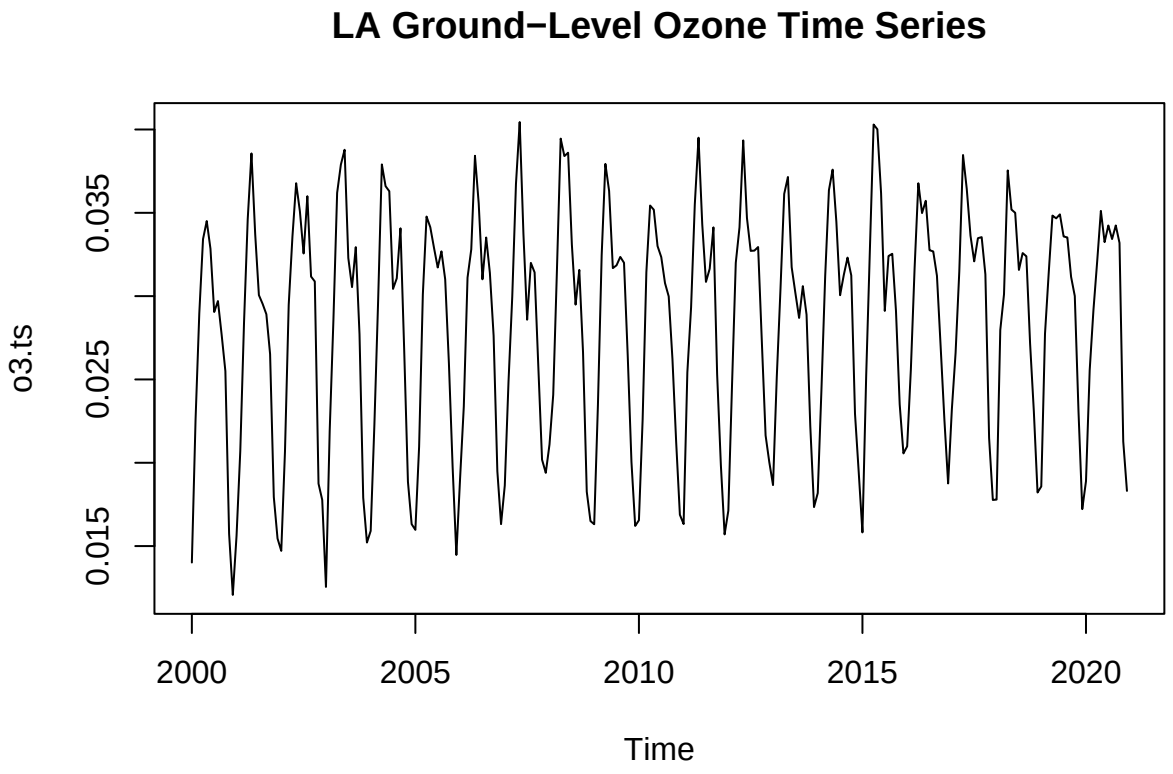
Air quality has been a major concern, especially for people in LA. According to EPA website, ground-level zone is listed as one of the major air pollutant. Ground-level ozone is created when pollutants emitted by cars, power plants, industrial boilers, refineries, chemical plants, and other sources chemically react in the presence of sunlight. And it can cause harm to our health and to the ecosystem. That's why I am interested in this data set which includes all major air pollutants in major American cities from January 2000 to September 2021. The data set comes from Kaggle and it was extracted from EPA database. [O3(ground-level ozone) in the data set is measured in ppm.]

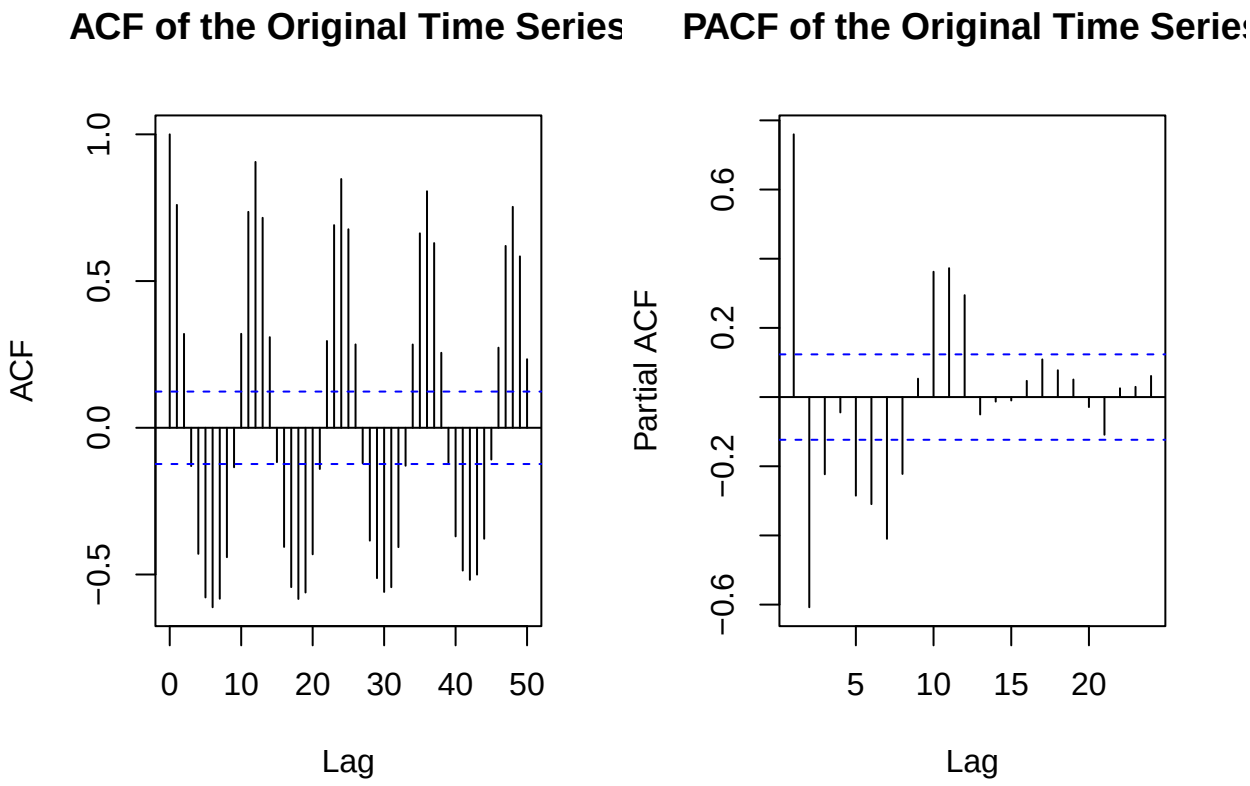
I planned to forecast future values (from Janurary 2021 to March 2022) so that we can potentially continue to make accurate predictions about the of ground-level ozone in specific locations. In order to make accurate forecast, I planned to try data transformation and differencing on the original time series, then use ACF/PACF to identify potential models, use diagnostic checking to see if the model fit well, and finally, forecast and check the test data set. I encountered problems when using the box-cox transformed data to fit the model, so I went back and used the original time series instead, and it worked. The forecast looks natural with the original time series, but the confidence interval of predictions fails include some points in the test data set, which indicates the limit of the model's predictive power. Overall the project achieves its goal of forecasting ground-level ozone with fair accuracy.

Main Sections: Time Series Analysis

Before I start the time series process, I first cleaned and modified my data so it's easier to use. I extract the daily average of ground ozone level in Los Angeles and then transformed them to monthly average. Then I select data from January 2000 to December 2020, save the rest(data from January 2021 to September 2021) as the test data set.

The time series plot of ground-level ozone:

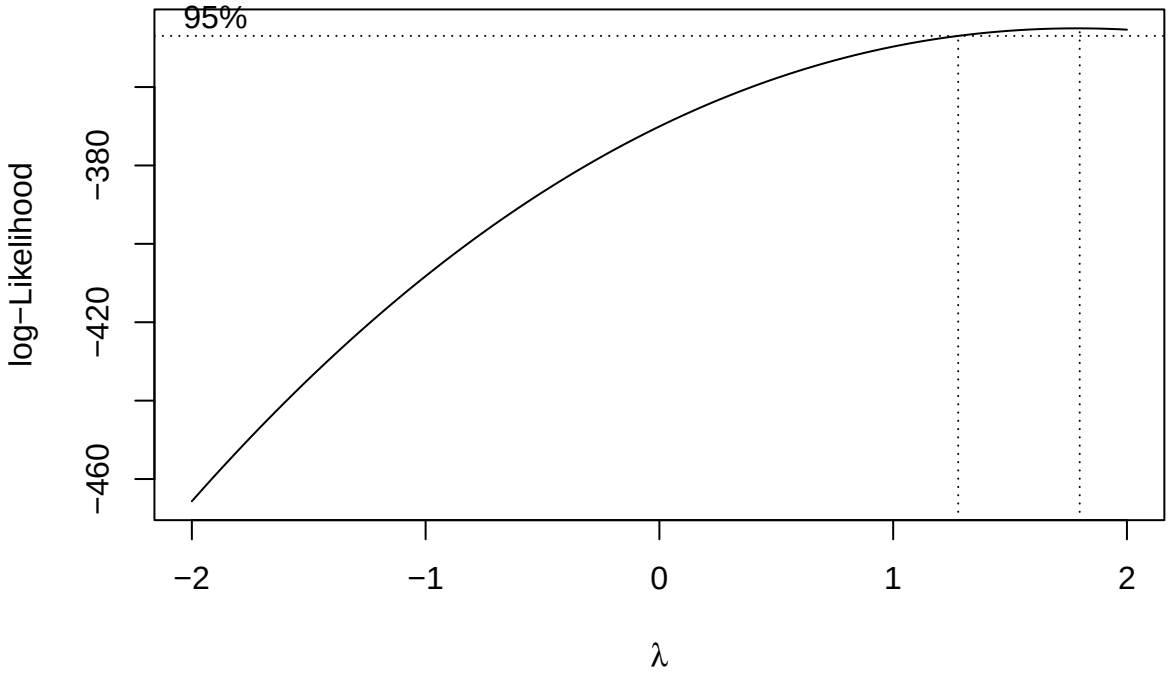




From the time series plot we can see a clear seasonal pattern, but there doesn't appear to be a significant trend. There doesn't seem to be a sharp change in behavior in the time series, and overall the graph seems pretty reasonable. From ACF and PACF of the original time series we can also see the seseaonal pattern since the ACF values are large at lag 12, 24, 36...

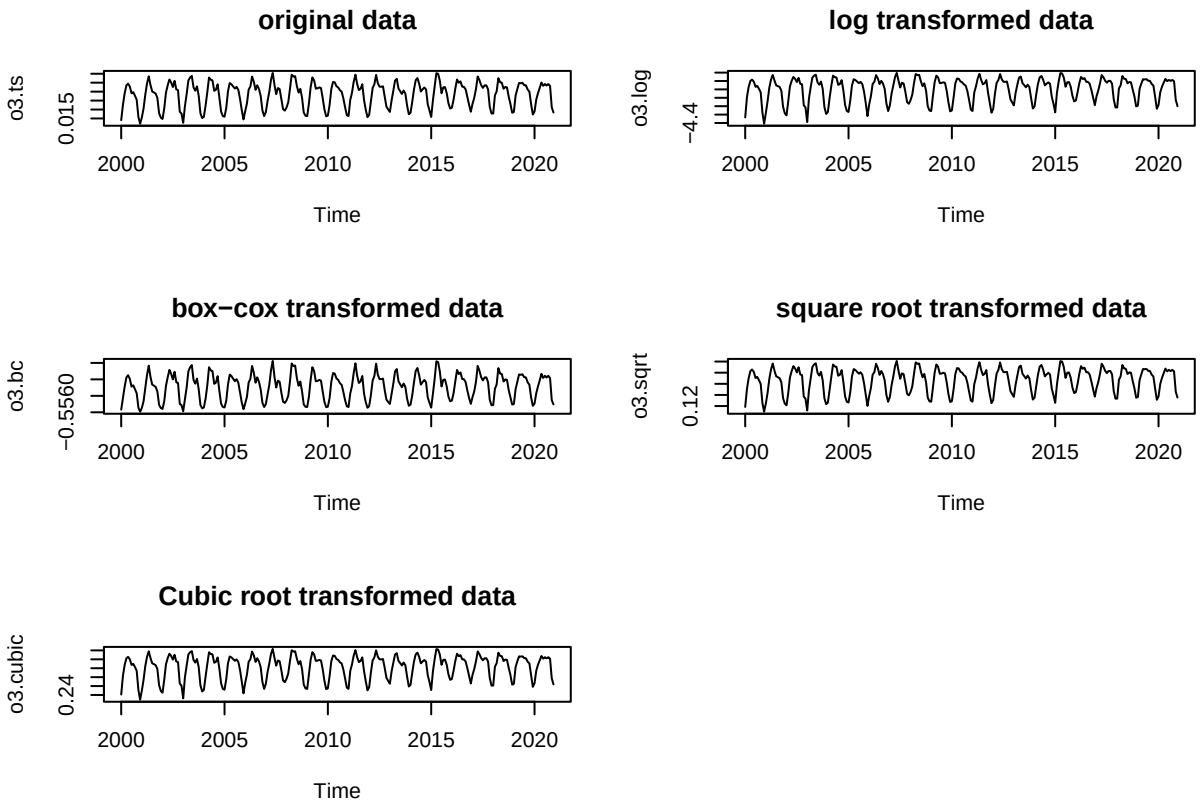
Transformations:

Firstly, I use Box-cox fitting to determine the best lambda value and the confidence interval of lambda.



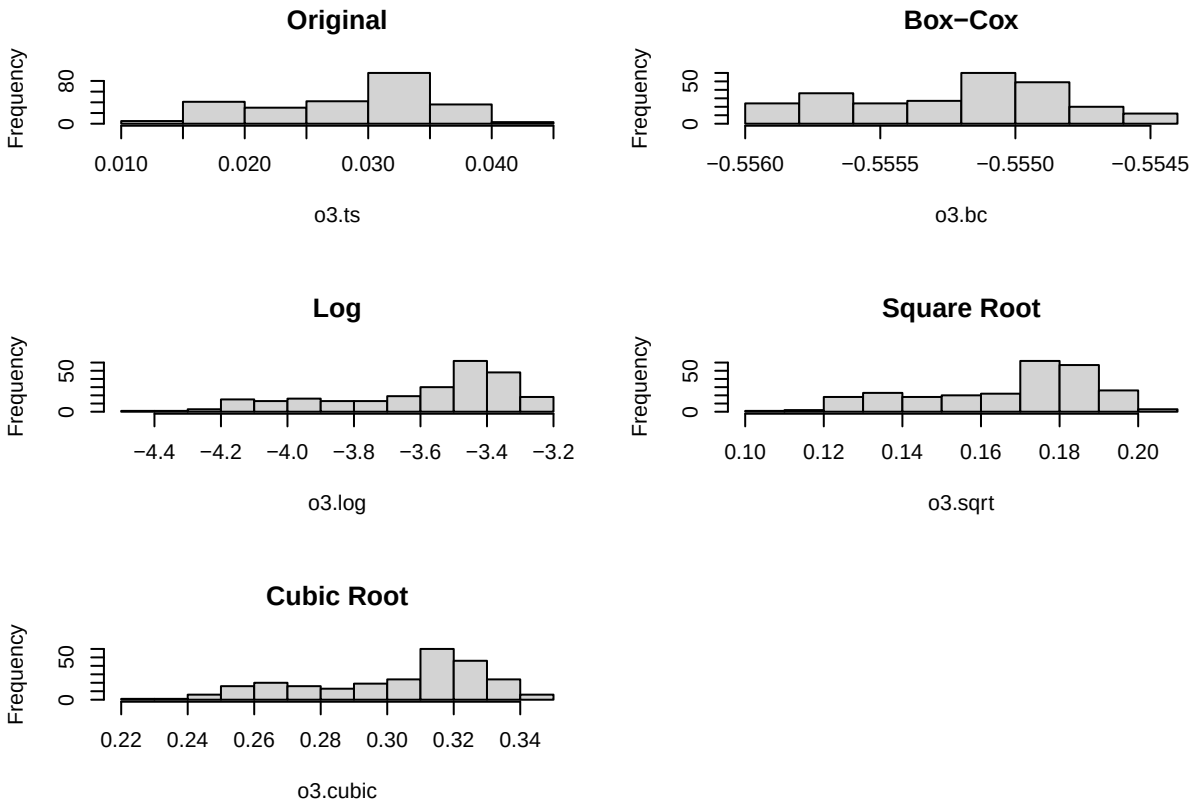

```
## [1] "lambda =  1.7979797979798"
```

Since $\lambda \neq 0$ and the 95% confidence interval does not include $1, \frac{1}{2}, \frac{1}{3}$, so we consider Box-Cox transformation over other transformations.




```
## [1] "original variance:  4.85016666573472e-05"

## [1] "box-cox transformed variance:  1.49528193143123e-07"

## [1] "log transformed variance:  0.076393386157384"

## [1] "square root transformed variance:  0.000473117742317962"

## [1] "cubic root transformed variance:  0.000718541617750946"
```

Unsurprisingly, the Box-Cox transformation yields the smallest variance.



However, we can observe that the histogram of the Box-cox transformed data has high frequency at the two ends, which might suggests that it does not make the data more "normal". Recall that the Box-Cox method does not guarantee normality, instead, it guarantees the smallest variance which leads to a better chance of normality. Also considering the fact that the time series of the original data seems to have a stable variance, the original data probably doesn't need transformations. Therefore, I will not use data transformation in this project.

Differentiate

Since the data was collected on a monthly basis, it was clear that there existed seasonality s = 12 (it could also be seen from PACF and ACF of the original data set). I deseasonalized the data by differentiating it once at lag 12.

```
## [1] "Variance of undifferentiated data: 4.85016666573472e-05"
```

```
## [1] "Variance of data differentiated once: 4.49311787012243e-06"
```

```
## [1] "Variance of data differentiated twice: 1.2191680370353e-05"
```

The data that was differentiated twice had larger variance than the data that was differentiated once, which meant that I over differentiated the data. Thus I should choose to differentiate the data once at lag 12.

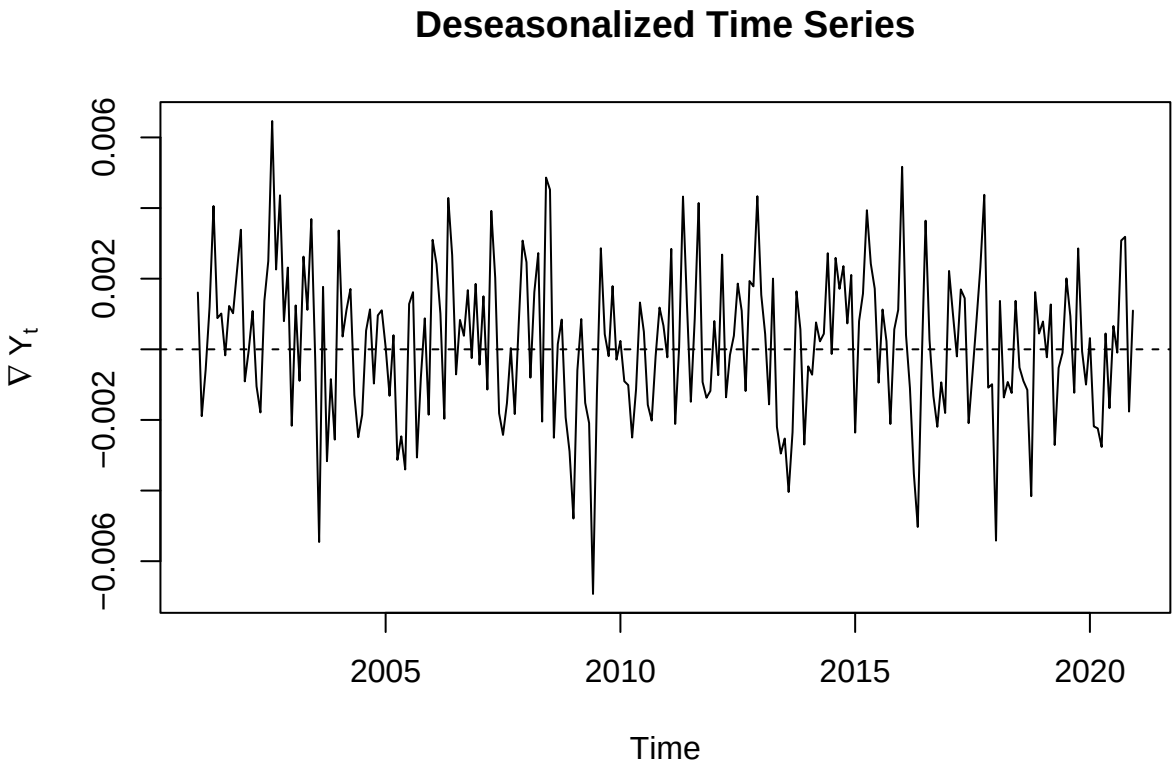
Next, I tried to check if there existed a trend in the data set. I differentiated the previous data at lag 1 once and twice, then I compare them with the previous data variance.

```
## [1] "Variance of undifferentiated data: 4.49311787012243e-06"
```

```
## [1] "Variance of data differentiated once: 7.2710609449437e-06"
```

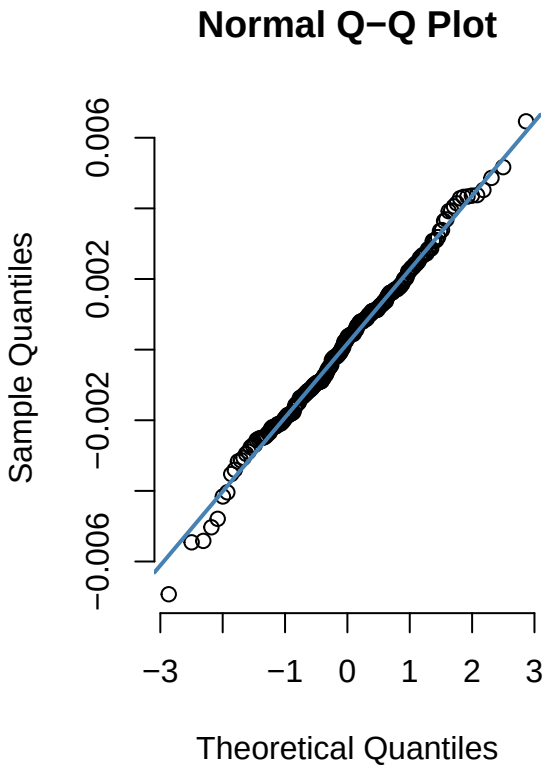
```
## [1] "Variance of data differentiated twice: 2.0339620918806e-05"
```

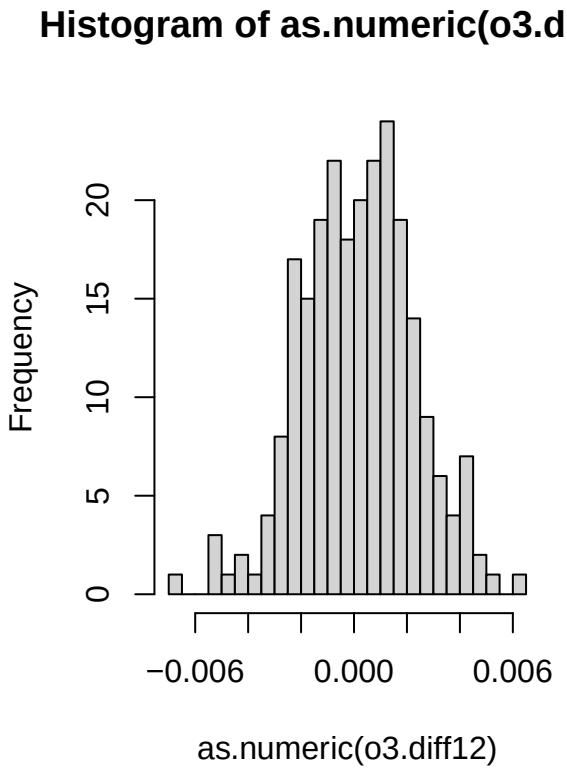
The variance of undifferentiated data was actually the smallest, which indicated that there deosn't exist a significant trend in the time series. So I would be using the deseasonalized original data to analyze the time series.



Check Stationarity

From the graph of the de-trended/seasonalized data we can see that it is similar to white noise. To further show the stationarity, we use Q-Q plot, histogram and the Shapiro-Wilk test.

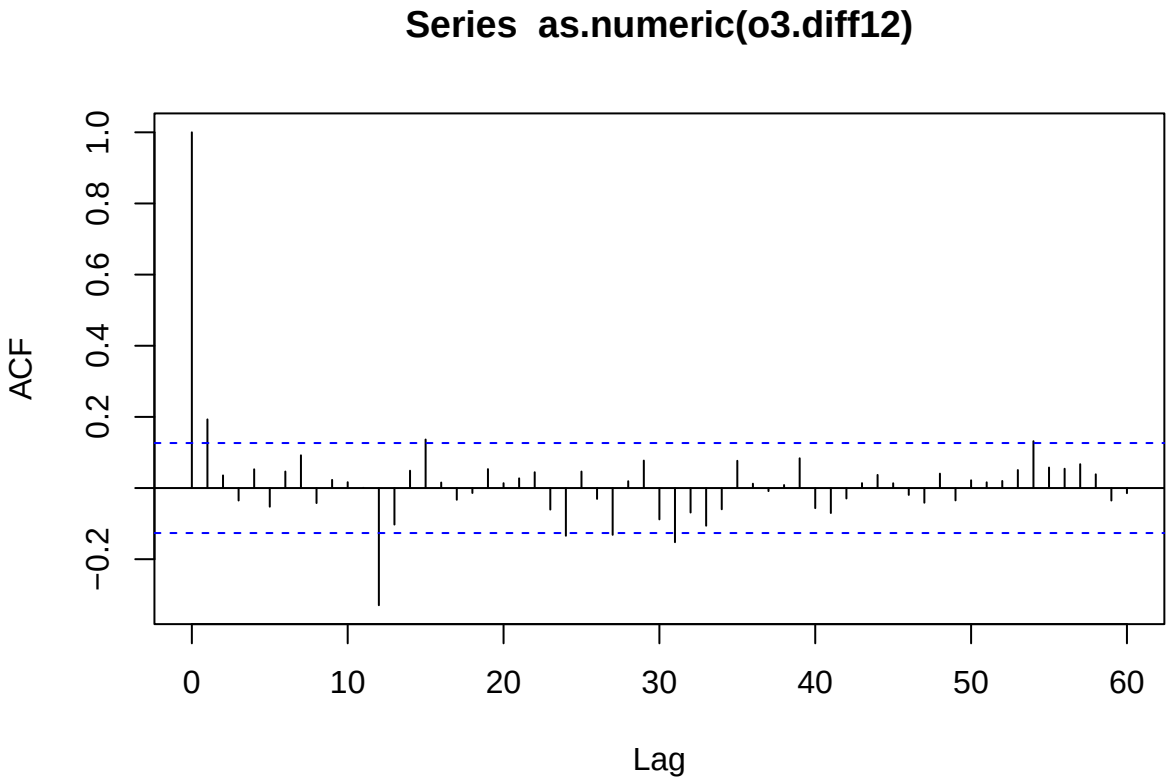


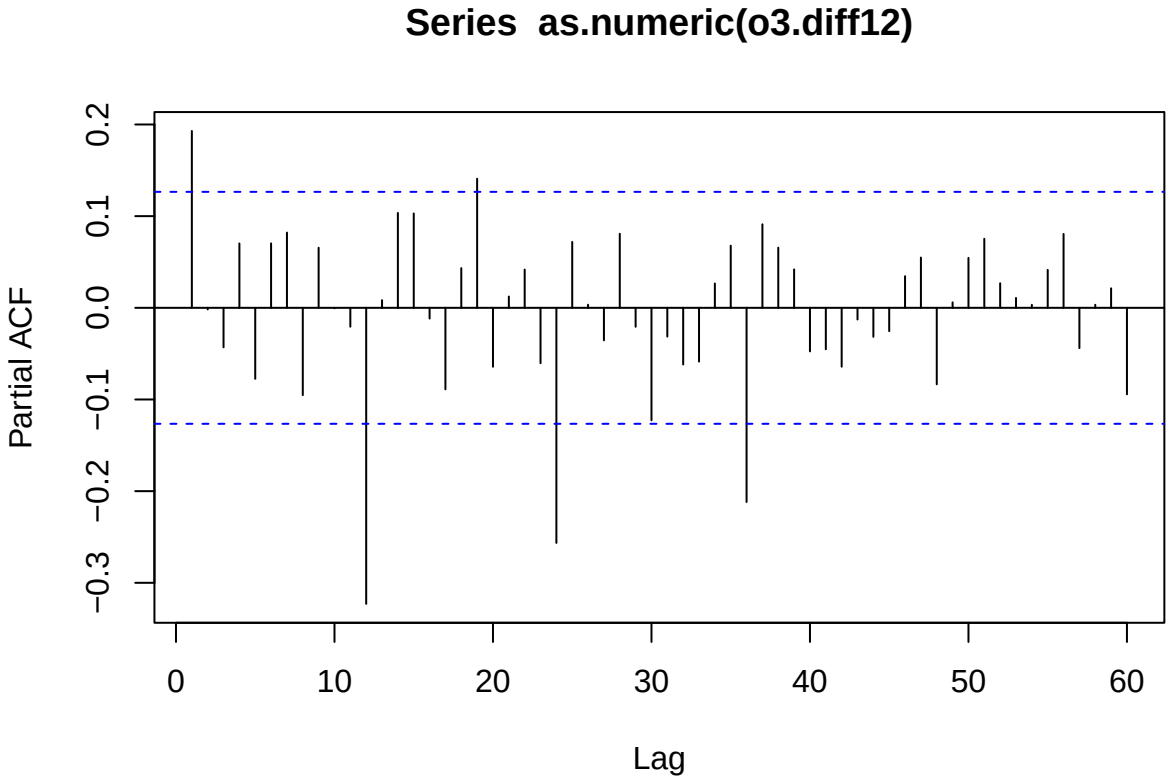



```
##
##  Shapiro-Wilk normality test
##
## data:  o3.diff12
## W = 0.99476, p-value = 0.58
```

The Q-Q is approximately a straight line, and the histogram is approximately a Gaussian distribution. Furthermore, the Shaprio-Wilk test returns a p-value that is significantly bigger than significance level $\alpha = 0.05$. Therefore we can conclude that our data is stationary after differencing and move on to model identification.

Model Identification





△

Modeling the seasonal part(P, D, Q): For this part, focus on the seasonal lags $h = ks, k = 1, 2, 3...$

•We applied one seasonal differencing at lag 12 so $D = 1, s = 12$.

•The ACF shows a strong peak at h = 1s and smaller peak at h = 2s. A good choice for the MA part could be $Q = 0, 1, 2$.

•The PACF shows a strong peak at $h = 1s, 2s, 3s$. A good choice for the AR part could be $P = 0, 1, 2, 3$.

△

Modeling the non-seasonal part(p, d, q):In this case focus in the within seasonal lags, $h = 1, ...11$.

•We did not apply differencing to remove the trend so d = 0.

•The ACF seems to be cut off after 1 so q = 1 or 0.

•The PACF seems to be cut off after 1 so p = 1 or 0.

I use a for loop to select five models with the lowest AICcs:

```
## [1] "1 0 1 1 1 2"
```

```
## [1] "1 0 1 2 1 1"
```

```
## [1] "1 0 1 2 1 2"
```

```
## [1] "1 0 1 1 1 1"
```

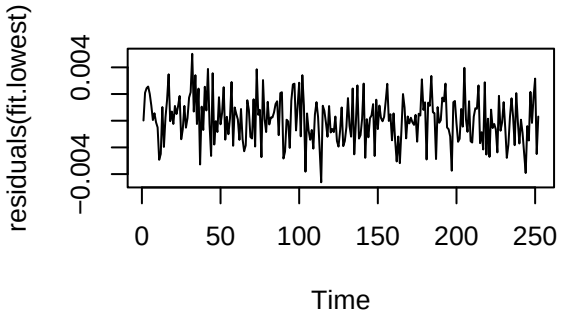
```
## [1] "0 0 1 1 1 2"
```

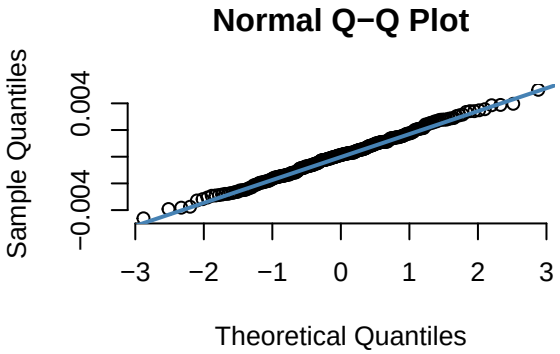
Fit the Models

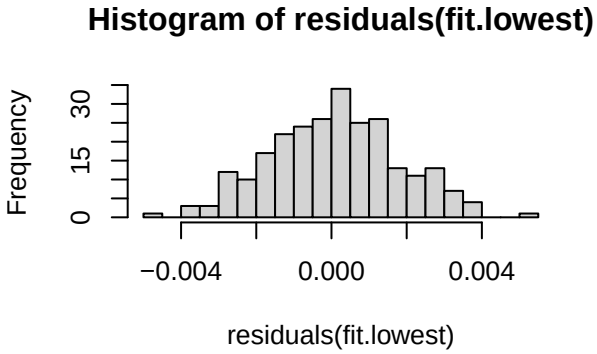
```
fit.lowest <- arima(as.numeric(o3.ts), order = c(1,0,1),seasonal = c(1, 1, 2, period = 12), me
fit.2nd.lowest <- arima(as.numeric(o3.ts), order = c(1,0,1),seasonal = c(2, 1, 1, period = 12)
fit.3rd.lowest <- arima(as.numeric(o3.ts), order = c(1,0,1),seasonal = c(2, 1, 2, period = 12)
fit.4th.lowest <- arima(as.numeric(o3.ts), order = c(1,0,1),seasonal = c(1, 1, 1, period = 12)
fit.5th.lowest <- arima(as.numeric(o3.ts), order = c(0,0,0),seasonal = c(1, 1, 2, period = 12)
```

It if the fit is good, then residuals resemble Gaussian White Noise~ (0, 1). In order to check this, I run Q-Q plot, histogram and shapiro test for each model: # Q-Q plot Histogram Shapiro-Wilk test

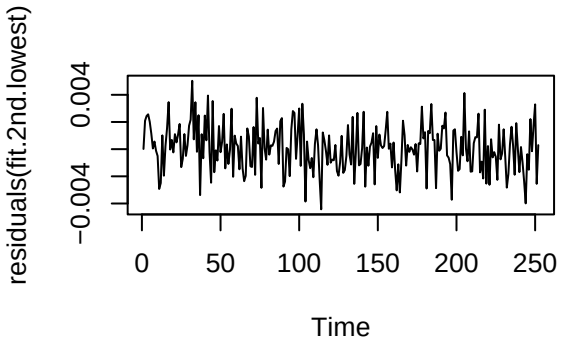
```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(fit.lowest)
## W = 0.99706, p-value = 0.9261
```

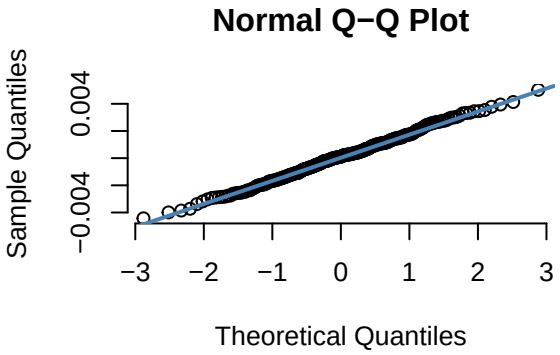


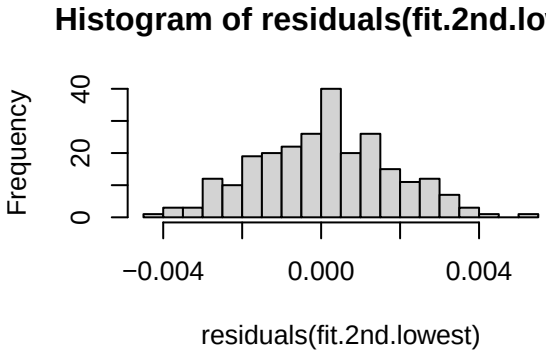




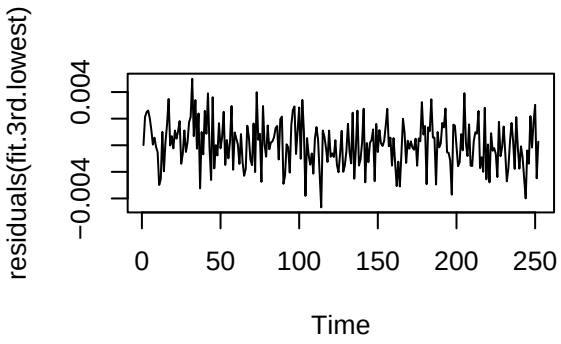

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(fit.2nd.lowest)
## W = 0.99705, p-value = 0.9254
```

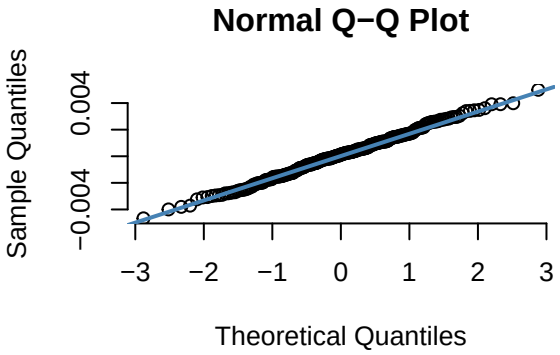


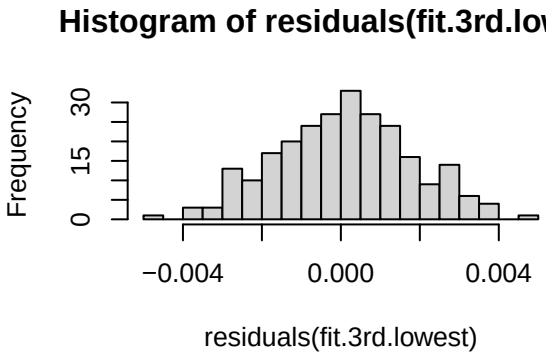




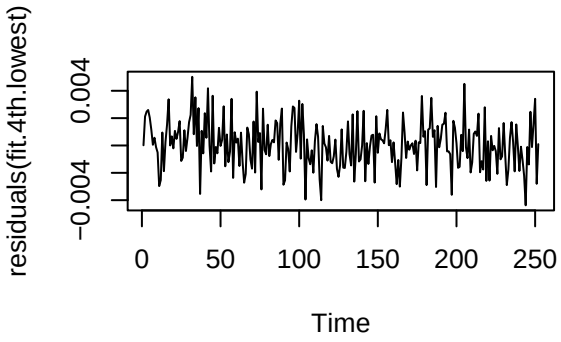

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(fit.3rd.lowest)
## W = 0.99708, p-value = 0.928
```

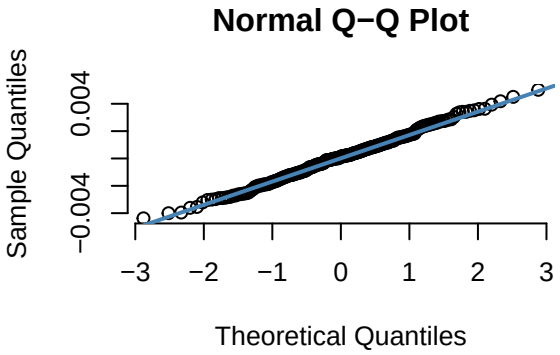


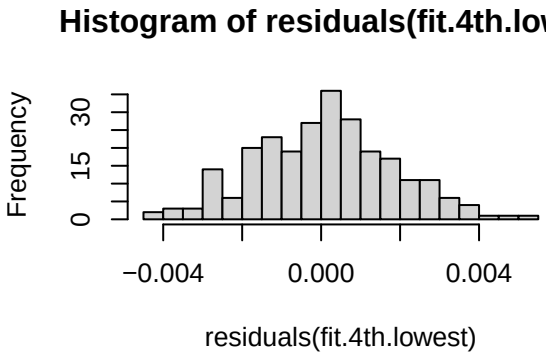




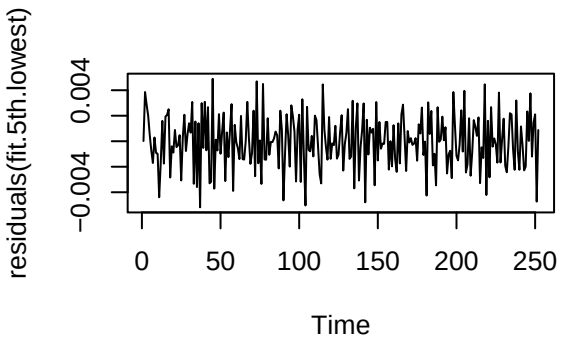

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(fit.4th.lowest)
## W = 0.99644, p-value = 0.8422
```

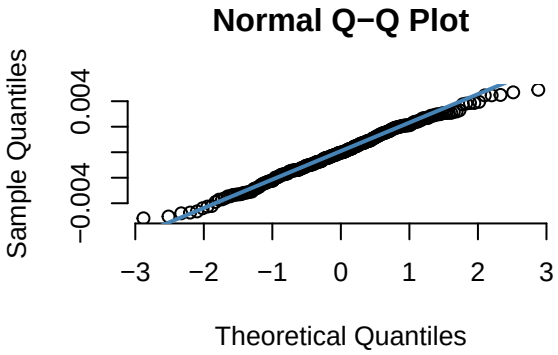


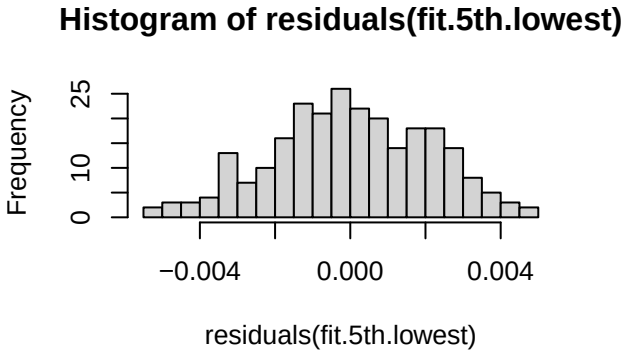





```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(fit.5th.lowest)
## W = 0.99316, p-value = 0.302
```







All of the data have near-straight line Q-Q plot and approximately Gaussian histograms. Since all five models have p-values bigger than $\alpha = 0.05$, we conclude that they all passed the Shapiro-Wilk normality test. Now let's see if they can pass the Box tests.

Box Tests

For box tests I choose $lag = \sqrt{n} = 16$ and $fitdf = p + q$ Box-Pierce Test:

```
##
##  Box-Pierce test
##
## data:  residuals(fit.lowest)
## X-squared = 13.614, df = 14, p-value = 0.4789

##
##  Box-Pierce test
##
## data:  residuals(fit.2nd.lowest)
## X-squared = 13.069, df = 14, p-value = 0.5211

##
##  Box-Pierce test
```

```
##
## data:  residuals(fit.3rd.lowest)
## X-squared = 13.469, df = 14, p-value = 0.4899

##
##  Box-Pierce test
##
## data:  residuals(fit.4th.lowest)
## X-squared = 14.121, df = 14, p-value = 0.4408

##
##  Box-Pierce test
##
## data:  residuals(fit.5th.lowest)
## X-squared = 44.551, df = 15, p-value = 9.011e-05
```

Ljung-Box Test:

```
##
##  Box-Ljung test
##
## data:  residuals(fit.lowest)
## X-squared = 14.265, df = 14, p-value = 0.4302

##
##  Box-Ljung test
##
## data:  residuals(fit.2nd.lowest)
## X-squared = 13.684, df = 14, p-value = 0.4735

##
##  Box-Ljung test
##
## data:  residuals(fit.3rd.lowest)
## X-squared = 14.115, df = 14, p-value = 0.4412

##
##  Box-Ljung test
##
## data:  residuals(fit.4th.lowest)
## X-squared = 14.798, df = 14, p-value = 0.3921

##
##  Box-Ljung test
##
## data:  residuals(fit.5th.lowest)
## X-squared = 45.489, df = 15, p-value = 6.408e-05
```

McLeod-Li test: tests residuals for non-linear dependence(If residuals are independent, then the squares should be uncorrelated)

```
##
##  Box-Ljung test
##
## data:  (residuals(fit.lowest))^2
## X-squared = 11.259, df = 16, p-value = 0.7932
```

```
##
##  Box-Ljung test
##
## data:  (residuals(fit.2nd.lowest))^2
## X-squared = 6.8504, df = 12, p-value = 0.8673
```

```
##
##  Box-Ljung test
##
## data:  (residuals(fit.3rd.lowest))^2
## X-squared = 10.665, df = 16, p-value = 0.8297
```

```
##
##  Box-Ljung test
##
## data:  (residuals(fit.4th.lowest))^2
## X-squared = 6.0629, df = 12, p-value = 0.9129
```

```
##
##  Box-Ljung test
##
## data:  (residuals(fit.5th.lowest))^2
## X-squared = 11.269, df = 16, p-value = 0.7926
```

Note that all models besides the fifth model have p-values that are bigger than significance level $\alpha = 0.05$. So only the fifth model doesn't pass Box-pierce and Ljung-box tests. For other models, since their p-values are greater than 0.05, we fail to reject the null hypothesis and conclude that their residuals are normal. So I will give up on the fifth model and move on to other diagostic checking without it.

Yule-Walker Estimation

```
ar(residuals(fit.lowest), aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

```
##
## Call:
## ar(x = residuals(fit.lowest), aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  2.953e-06
```

```
ar(residuals(fit.2nd.lowest), aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

```
##
## Call:
## ar(x = residuals(fit.2nd.lowest), aic = TRUE, order.max = NULL,     method = c("yule-walker"
##
##
## Order selected 0  sigma^2 estimated as  2.968e-06
```

```
ar(residuals(fit.3rd.lowest), aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

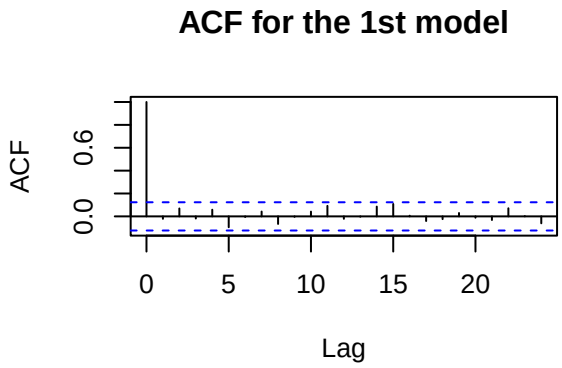
```
##
## Call:
## ar(x = residuals(fit.3rd.lowest), aic = TRUE, order.max = NULL,     method = c("yule-walker"
##
##
## Order selected 0  sigma^2 estimated as  2.96e-06
```

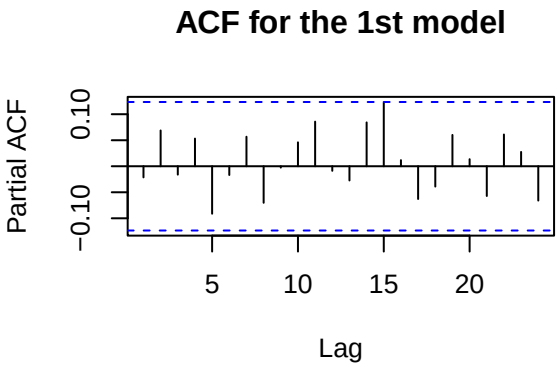
```
ar(residuals(fit.4th.lowest), aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

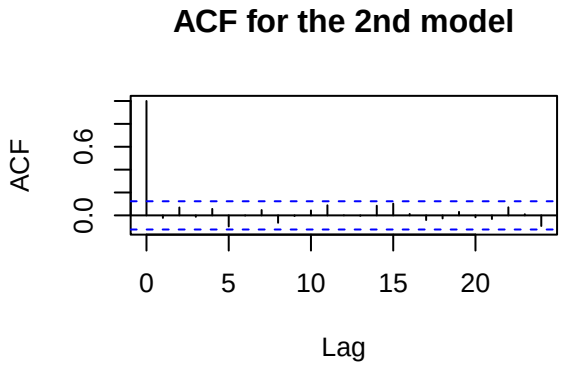
```
##
## Call:
## ar(x = residuals(fit.4th.lowest), aic = TRUE, order.max = NULL,     method = c("yule-walker"
##
##
## Order selected 0  sigma^2 estimated as  3.049e-06
```

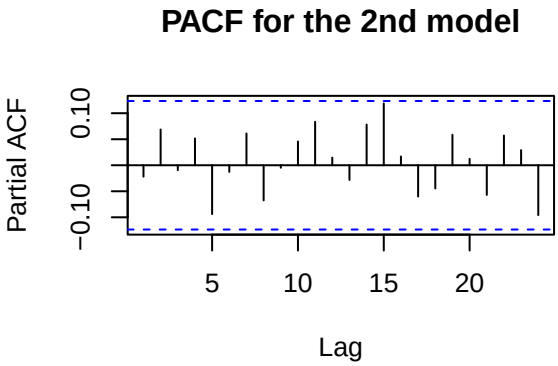
Since the residuals fit into AR(0) model, the residuals of our four models are white noise.

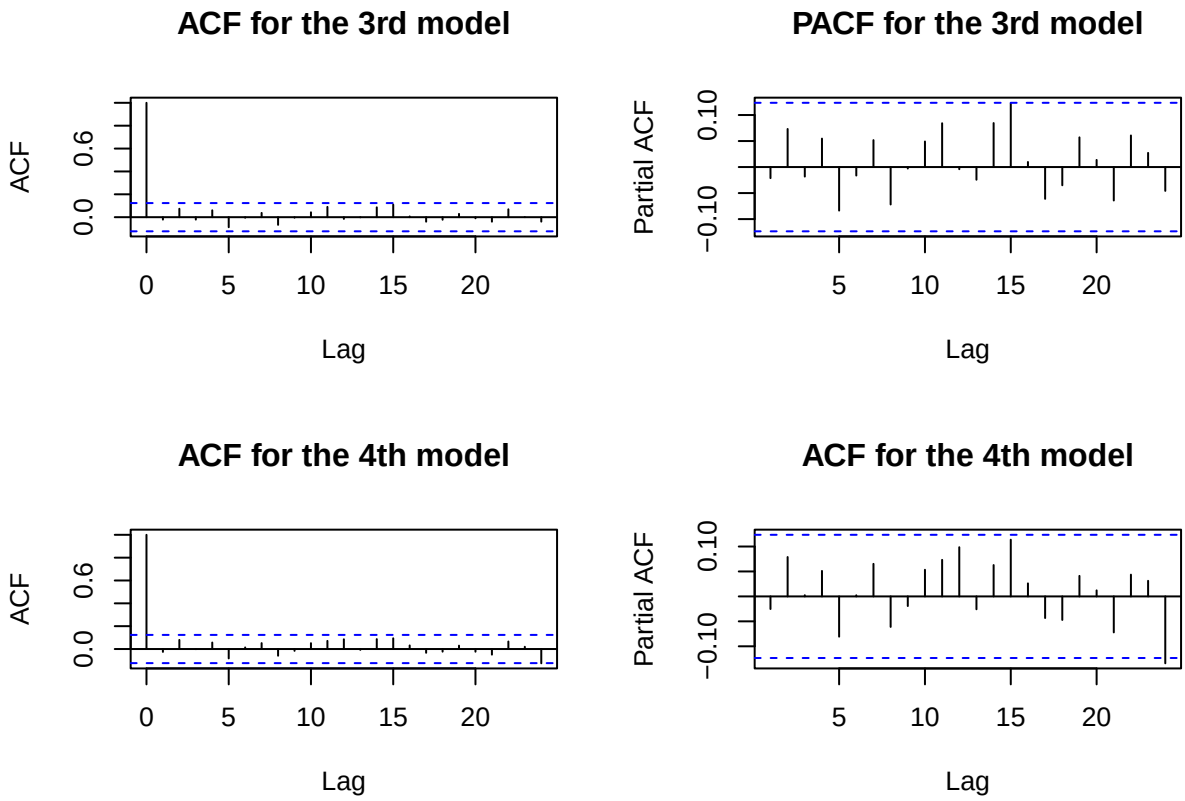
ACF/PACF of Residuals











All of the models beside the 4th model have ACF and PACF that resemble white noise.Observe that only the the 4th model's residuals PACF have value outside of confidence interval at lag 25, It may due to some hidden correlations within the residuals or due to the variance of residuals.

Since the first model has the smallest AICc and passed all the diagnostic checking (of residuals) so far, we will continue diagnostic checking with the first model and ignore the other models for now. The coefficients of the first model are:

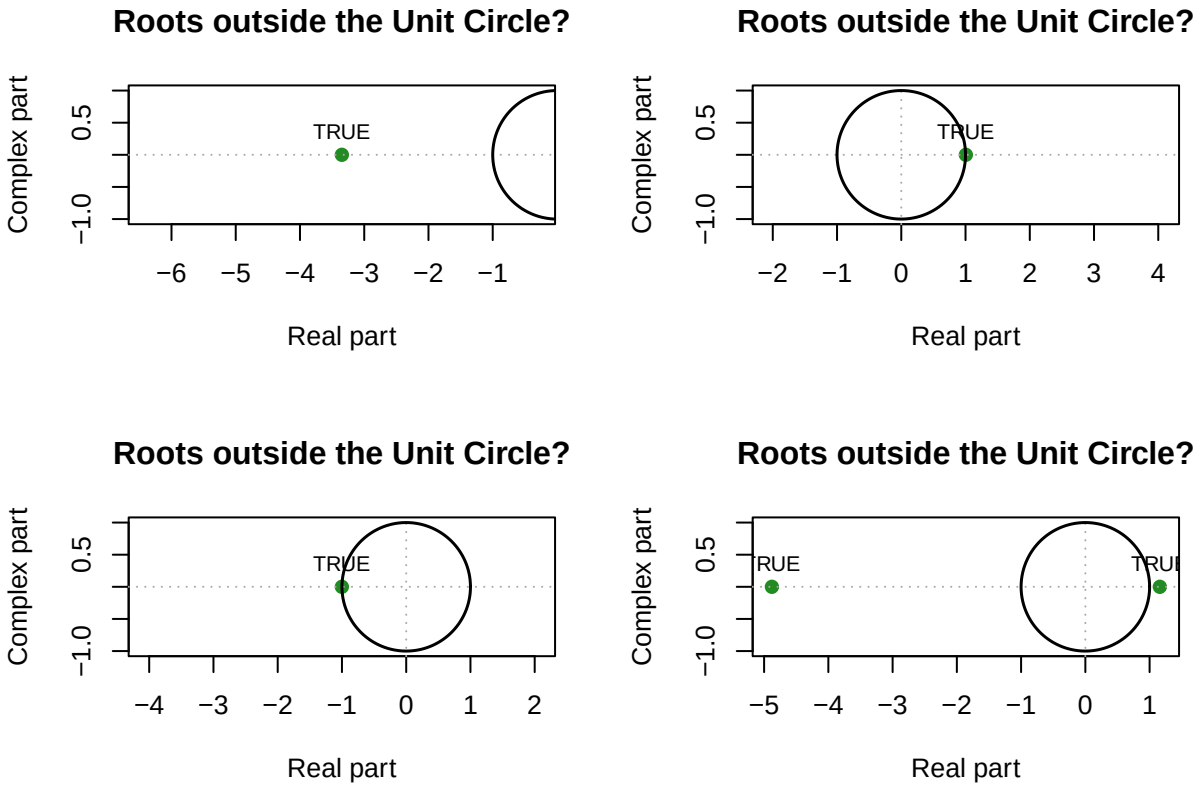
```
##         ar1         ma1        sar1        sma1        sma2
##   0.2986805 -0.9947257   0.9986387 -0.6575869 -0.1768040
```

Unit Circle Check

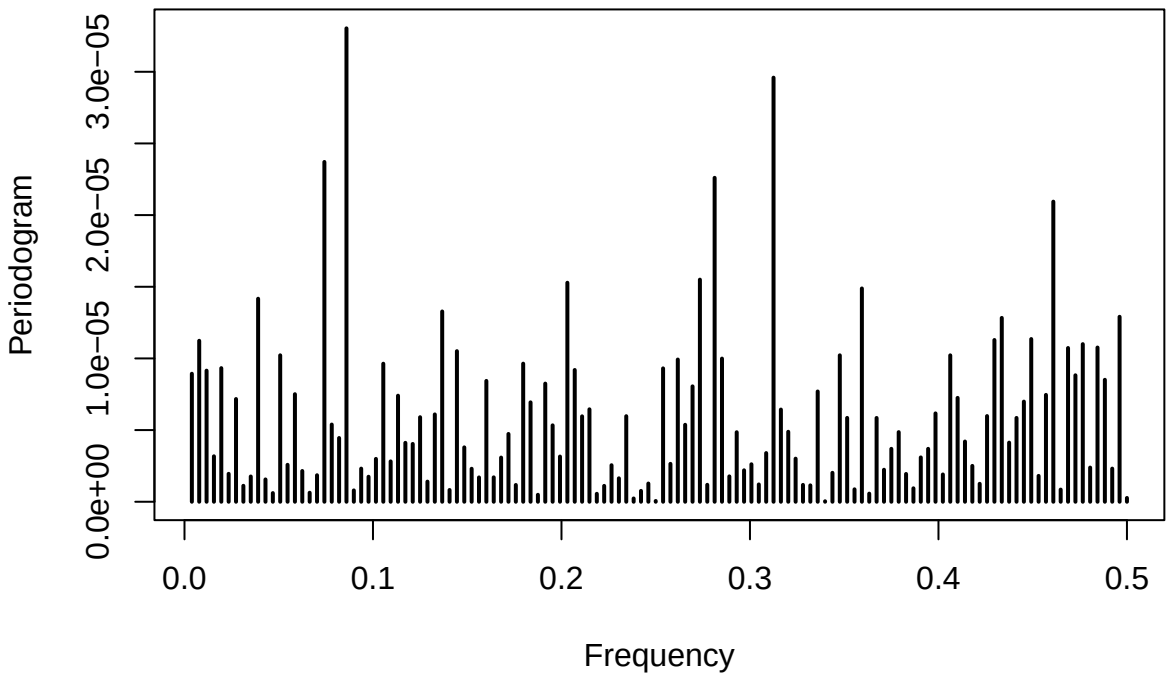
Now do the unit circle check to make sure the model is invertible and stationary in theory.

```
library(UnitCircle)
op <- par(mfrow = c(2,2))
# non-seasonal ar unit circle check
uc.check(pol_ = c(1,0.2986805), plot_output = TRUE, print_output = FALSE)
# non-seasonal ma unit circle check
uc.check(pol_ = c(1, -0.9947257), plot_output = TRUE, print_output = FALSE)
# seasonal ar unit circle check
uc.check(pol_ = c(1,0.9986387), plot_output = TRUE, print_output = FALSE)
```

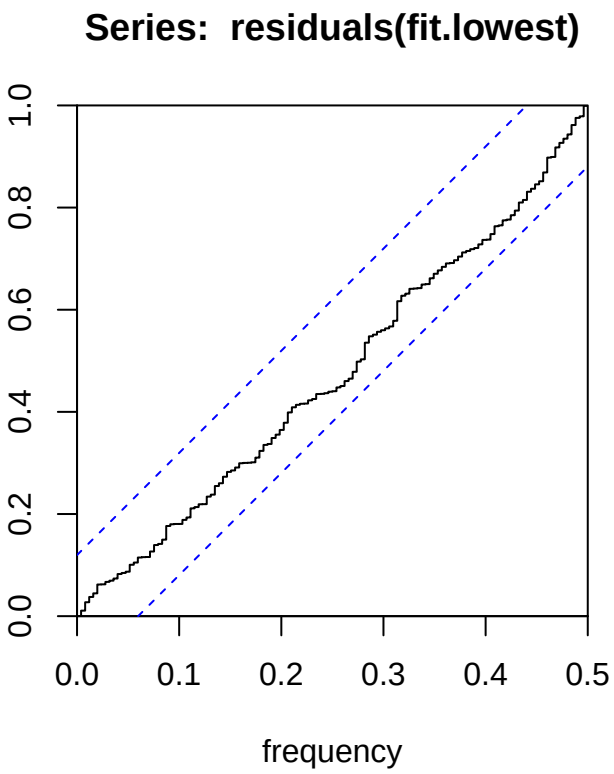
```r
# seasonal ma unit circle check
uc.check(pol_ = c(1,-0.6575869,-0.1768040 ), plot_output = TRUE,print_output = FALSE)
```



The model passes the unit circle test(all of the roots are outside of the unit circle, therefore we can conclude that our model is stationary and invertible. # McLeod-Li test # Spectral Analysis Before forecasting, I perform spectral analysis on my model to make sure that the residuals of the model are indeed white noise without hidden patterns.



From the periodgram, we can see that there's no dominated frequency in residuals




```
## [1] "Fisher test p-value: 0.810120361496981"
```

The line is inside the 95% interval so the residuals of our model pass the Kolmogorov-Smirnov Test for cumulative periodogram of residuals so we fail to reject the null hypothesis and conclude that the residuals of the model is Gaussian white noise.

Fisher's test yields value 0.81 which is larger than $\alpha = 0.05$ so we fail to reject the null hypothesis and conclude that the residuals of the model is Gaussian white noise.

Fitted Model in Algebraic Form

Since we choose $SARIMA(p,d,q)\times(P,D,Q)_s$ where $(p,d,q) = (1,0,1), (P,D,Q) = (1,1,2), s = 12$ SARIMA process:

$\phi(B)\Phi(B^s)Y_t = \theta(B)\Theta(B^s)Z_t$ where $Y_t = (1-B)^d(1-B^s)^D X_t$

$Y_t = (1-B)^0(1-B^{12})^1 X_t = (1-B^{12})X_t$

$\phi(B) = 1 - \phi_1 B$

$\Phi(B^s) = 1 - \Phi_1 B^{12}$

$\theta(B) = 1 + \theta_1 B$

$\Theta(B^s) = 1 + \Theta_1 B^{12} + \Theta_2 B^{24}$

$\phi(B)\Phi(B^s)Y_t = \theta(B)\Theta(B^s)Z_t \Longrightarrow$

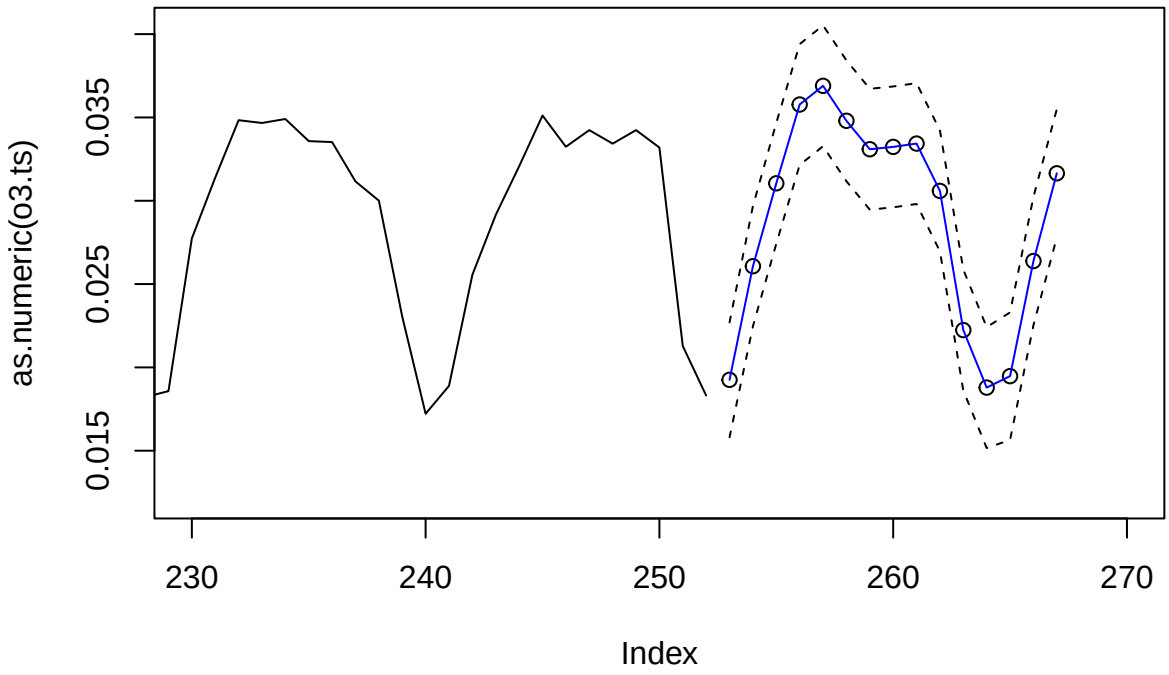
$(1-\phi_1 B)(1-\Phi_1 B^{12})(1-B^{12})X_t = (1+\theta_1 B)(1+\Theta_1 B^{12}+\Theta_2 B^{24})Z_t \Longrightarrow$

$(1-0.2987B)(1-0.9986B^{12})(1-B^{12})X_t = (1-0.9947B)(1-0.6576B^{12}-0.1768B^{24})Z_t \Longrightarrow$

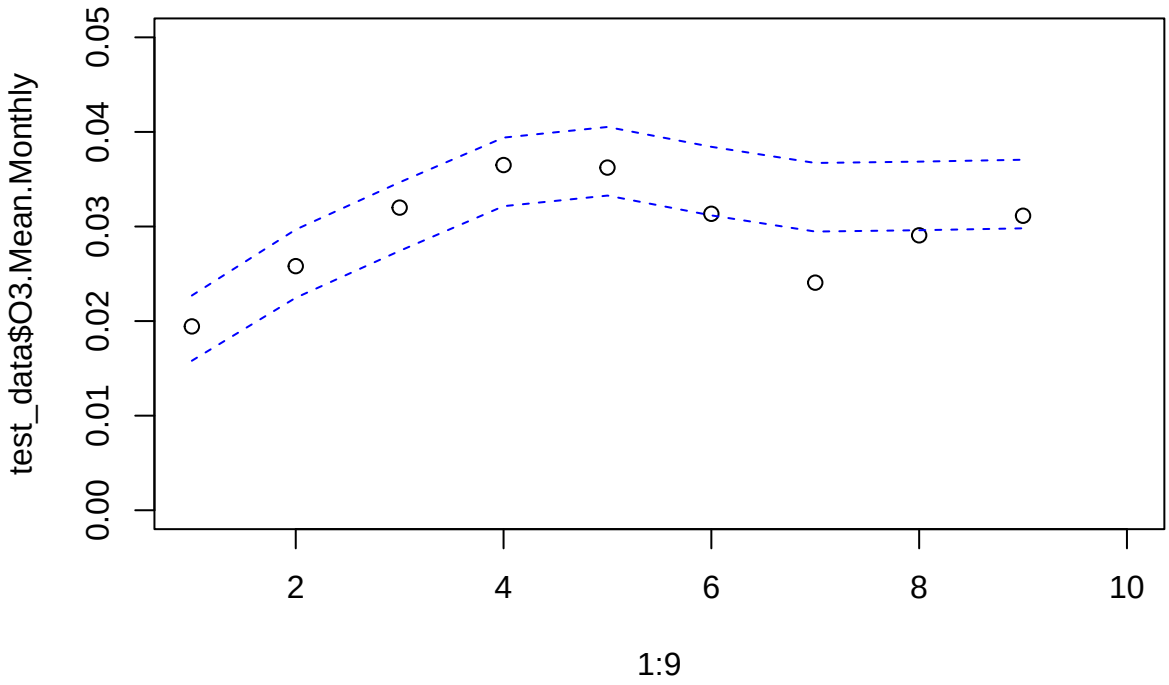
$(X_t - 0.2987X_{t-1})(X_t + 0.9986X_{t-12})(X_t - X_{t-12}) = (Z_t - 0.9947Z_{t-1})(Z_t - 0.6576Z_{t-12} - 0.1768Z_{t-24})$

Forecast

Below is the ozone level forecast from January 2021 to March 2022 with 95% confidence interval. According to our forecast, the monthly average ground ozone observed in LA will be 0.03164814ppm.



Fit the test data:



Observe that the first five points fall within the confidence interval but the interval doesn't do a well job covering data points after the fifth one. This shows that the model's predictive ability decreases after the first 5 forecasts. One way of fixing it is to frequently update the data.

Conclusion:

In this project, I achieved my goal of identifying a model for the ground-level ozone time series and forecasting the ground-level ozone from January 2021 to present. My model for the ground-level ozone time series:

$$(X_t - 0.2987X_{t-1})(X_t + 0.9986X_{t-12})(X_t - X_{t-12}) = (Z_t - 0.9947Z_{t-1})(Z_t - 0.6576Z_{t-12} - 0.1768Z_{t-24})$$

However, to increase the predictive power of this model, one does need to keep updating the data.

#Thanks to Professor Feldman and Sunpeng for answering my questions about Box-Cox transformation and spectral analysis.

References:

Source of data set:

https://www.epa.gov/ground-level-ozone-pollution/health-effects-ozone-pollution

https://www.kaggle.com/alpacanonymous/us-pollution-20002021

Appendix:

```
library(dplyr)
library(lubridate)
library(forecast)

# read in data from csv file
US_pollution <- read.csv("pollution_2000_2021.csv")
US_pollution <- as_tibble(US_pollution)

# select relevant columns
Cal_pollution <- US_pollution %>%
  filter(State == "California")
Cal_pollution <- Cal_pollution[,c("Date","O3.Mean")]
Cal_pollution$Date <- as.Date(Cal_pollution$Date)

# change the daily average data to monthly average data
Cal_pollution$Date <- floor_date(Cal_pollution$Date, "month")
Cal_pollution <- Cal_pollution %>%
  group_by(Date) %>%
  summarise(mean = mean(O3.Mean)) %>%
  mutate(O3.Mean.Monthly = mean) %>%
  mutate(Date = format(Date,"%Y-%m"))
Cal_pollution <- Cal_pollution[,c("Date","O3.Mean.Monthly")]
test_data <- Cal_pollution[Cal_pollution$Date > "2020-12",]
Cal_pollution <- Cal_pollution[Cal_pollution$Date <="2020-12",]
# Cal_pollution is the the final data set that we are going to use
# It contains monthly O3 mean from Jan 2000 to Dec 2020


# create a time series based on the monthly o3 data
o3.ts = ts(Cal_pollution$O3.Mean.Monthly, start = c(2000,1), end = c(2020,12), frequency = 12)
ts.plot(o3.ts, main  =  "California Ground Ozone Time Series")


# log transformation
o3.log <- log(o3.ts)
# box cox transformation
require(MASS)
t <- 1:length(o3.ts)
fit <- lm(o3.ts~t)
bcTransform <- boxcox(o3.ts~t, plotit = TRUE)
# find the best lambda
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
print(paste("lambda = ", lambda))
o3.bc <- (1/lambda)*(o3.ts^lambda-1)
#co3.bc <- o3.ts^1.5
```

```r
# power transformation
o3.sqrt <- sqrt(o3.ts)
o3.cubic <- (o3.ts)^(1/3)

# time series comparisons between different transformations
op <- par(mfrow = c(2,2))
ts.plot(o3.ts, main = "original data")
ts.plot(o3.log, main = "log transformed data")
ts.plot(o3.bc, main = "box-cox transformed data")
ts.plot(o3.sqrt, main = "square root transformed data")
ts.plot(o3.cubic, main = "Cubic root transformed data")

# variance comparison between different time series
print(paste("original variance: ", var(o3.ts)))
print(paste("box-cox transformed variance: ",var(o3.bc)))
print(paste("log transformed variance: ",var(o3.log)))
print(paste("square root transformed variance: ",var(o3.sqrt)))
print(paste("cubic root transformed variance: ",var(o3.cubic)))

# compare histograms
op <- par(mfrow = c(2,2))
hist(o3.ts, main = "Original")
hist(o3.bc, main = "Box-Cox")
hist(o3.log, main = "Log")
hist(o3.sqrt, main = "Square Root")
hist(o3.cubic, main = "Cubic Root")

# differentiate at lag 12 / compare variances
o3.diff12 <- diff(o3.ts, lag = 12, differences = 1)
o3.diff122 <- diff(o3.ts, lag = 12, differences = 2)
print(paste("Variance of undifferentiated data:", var(o3.bc)))
print(paste("Variance of data differentiated once:", var(o3.diff12)))
print(paste("Variance of data differentiated twice:", var(o3.diff122)))

# differentiate at lag 1 / compare variances
o3.diff1 <- diff(o3.diff12, lag = 1, differences = 1)
o3.diff2 <- diff(o3.diff12, lag = 1, differences = 2)
print(paste("Variance of undifferentiated data:", var(o3.diff12)))
print(paste("Variance of data differentiated once:", var(o3.diff1)))
print(paste("Variance of data differentiated twice:", var(o3.diff2)))

# plot deseasonalized time series
plot(o3.diff12, main = "Deseasonalized Time Series", ylab = expression(nabla~Y[t]))
abline(h =0, lty = 2)

# test stationarity by Q-Q plot, histogram, and shapiro-wilk test
op <- par(mfrow = c(1,2))
```

```
qqnorm(as.numeric(o3.diff12), pch = 1, frame = FALSE)
qqline(as.numeric(o3.diff12), col = "steelblue", lwd = 2)
hist(as.numeric(o3.diff12), 25)
shapiro.test(o3.diff12)
# same code for all 5 models

# acf/pacf of deseasonalized time series
# for identify models
acf(as.numeric(o3.diff12),60)
pacf(as.numeric(o3.diff12),60)

library(astsa)
library(MuMIn)
is_try_error <- function(x) inherits(x, "try-error")
# creates a function to determine whether the code produces a try error

# for loop
v <- c() #initialize empty vector for p, q, P, Q combinations
m <- c() #initialize empty vector for AICc's
i = 1 #initialize index
for (P in c(0,1, 2, 3)){
  for(p in c(0,1)){
    for (q in c(0, 1)){
      for (Q in c(0, 1, 2)){
        #assigns try function to variable x
        x = try(AICc(arima(as.numeric(o3.ts), order = c(p,0,q),
                           seasonal = c(P, 1, Q, period = 12), method="ML")))
        #if the try function produces an error, then assign NA to the index in the AICc vecto
        if (is_try_error(x)){
          m[i] = NA
          v[i] = paste(p, 0, q, P,1, Q, sep = " ")
          i = i + 1
          }else{
            #if no error, then assign the AICc to the proper index in the AICc vector
            m[i] = AICc(arima(as.numeric(o3.ts), order = c(p,0,q),
                              seasonal = c(P, 1, Q, period = 12), method="ML"))
            v[i] = paste(p, 0, q, P,1, Q, sep = " ")
            i = i +1
          }
      }
    }
 }
}

# get five models with lowest AICcs
m1<- m[-which(is.na(m))] # remove the ones that produce error
v1 <- v[-which(is.na(m))]
```

```r
first <- v1[which(m1 == min(m1))]

m2 <- m1[-which(m1 == min(m1))]
v2 <- v1[-which(m1 == min(m1))]
second <- v2[which(m2 == min(m2))]

m3 <-m2[-which(m2 == min(m2))]
v3 <-v2[-which(m2 == min(m2))]
third <- v3[which(m3 == min(m3))]

m4 <-m3[-which(m3 == min(m3))]
v4 <-v3[-which(m3 == min(m3))]
fourth <- v4[which(m4 == min(m4))]

m5 <-m4[-which(m4 == min(m4))]
v5 <-v4[-which(m4 == min(m4))]
fifth <- v4[which(m5 == min(m5))]

# fit the models
fit.lowest <- arima(as.numeric(o3.ts), order = c(1,0,1),seasonal = c(1, 1, 2, period = 12), me
fit.2nd.lowest <- arima(as.numeric(o3.ts), order = c(1,0,1),seasonal = c(2, 1, 1, period = 12)
fit.3rd.lowest <- arima(as.numeric(o3.ts), order = c(1,0,1),seasonal = c(2, 1, 2, period = 12)
fit.4th.lowest <- arima(as.numeric(o3.ts), order = c(1,0,1),seasonal = c(1, 1, 1, period = 12)
fit.5th.lowest <- arima(o3.ts, order = c(0,0,0),seasonal = c(1, 1, 2, period = 12), method="ML

# test the residuals of first model
op <- par(mfrow = c(2,2))
# plot residuals
plot(residuals(fit.lowest))
# Q-Q plot of residuals
qqnorm(residuals(fit.lowest), pch = 1, frame = FALSE)
qqline(residuals(fit.lowest), col = "steelblue", lwd = 2)
# histogram of residuals
hist(residuals(fit.lowest), 25)
# shapiro test
shapiro.test(residuals(fit.lowest))
## same for other models

# box-pierce test for the three models
Box.test(residuals(fit.lowest),lag = 16, type = "Box-Pierce", fitdf = 2)
Box.test(residuals(fit.2nd.lowest),lag = 16,type = "Box-Pierce", fitdf = 2)
Box.test(residuals(fit.3rd.lowest),lag = 16, type = "Box-Pierce", fitdf = 2)
Box.test(residuals(fit.4th.lowest),lag = 16, type = "Box-Pierce", fitdf = 2)
Box.test(residuals(fit.5th.lowest),lag = 16, type = "Box-Pierce", fitdf = 1)

# box-ljung test for the three models
Box.test(residuals(fit.lowest), lag = 16, type = "Ljung", fitdf = 2)
```

```r
Box.test(residuals(fit.2nd.lowest), lag = 16, type = "Ljung", fitdf = 2)
Box.test(residuals(fit.3rd.lowest), lag = 16, type = "Ljung", fitdf = 2)
Box.test(residuals(fit.4th.lowest), lag = 16, type = "Ljung", fitdf = 2)
Box.test(residuals(fit.5th.lowest), lag = 16, type = "Ljung", fitdf = 1)

# acf/pacf of residuals
acf(residuals(fit.lowest), main = "ACF for the 1st model")
pacf(residuals(fit.lowest), main = "ACF for the 1st model")

acf(residuals(fit.2nd.lowest), main = "ACF for the 2nd model")
pacf(residuals(fit.2nd.lowest), main = "PACF for the 2nd model")

acf(residuals(fit.3rd.lowest), main = "ACF for the 3rd model")
pacf(residuals(fit.3rd.lowest), main = "PACF for the 3rd model")

acf(residuals(fit.4th.lowest), main = "ACF for the 4th model")
pacf(residuals(fit.4th.lowest), main = "ACF for the 4th model")


# unit circle test
library(UnitCircle)
op <- par(mfrow = c(2,2))
# non-seasonal ar unit circle check
uc.check(pol_ = c(1,0.2986805), plot_output = TRUE, print_output = FALSE)
# non-seasonal ma unit circle check
uc.check(pol_ = c(1, -0.9947257), plot_output = TRUE, print_output = FALSE)
# seasonal ar unit circle check
uc.check(pol_ = c(1,0.9986387), plot_output = TRUE, print_output = FALSE)
# seasonal ma unit circle check
uc.check(pol_ = c(1,-0.6575869,-0.1768040 ), plot_output = TRUE,print_output = FALSE)

# spectral analysis
require(TSA)\
# periodogram
TSA::periodogram(residuals(fit.lowest), log = FALSE, plot = TRUE)
library(stats)
# Plots a cumulative periodogram
cpgram(residuals(fit.lowest))
library(GeneCycle)
# fihser's test
fisher.g.test(residuals(fit.lowest))


# Forecast
mypred <- predict(fit.lowest, n.ahead = 15)
plot(as.numeric(o3.ts),xlim = c(230,270),type = "l")
points(253:267, mypred$pred)
```

```r
lines(253:267, mypred$pred, lty = 1, col = "blue")
lines(253:267, mypred$pred+2*mypred$se, lty = 2)
lines(253:267, mypred$pred-2*mypred$se, lty = 2)

# Test Statistics
mypred2 <- predict(fit.lowest, n.ahead = 9)
plot(x= 1:9, y = test_data$O3.Mean.Monthly, xlim = c(1,10), ylim = c(0, 0.05))
lines(1:9, mypred2$pred+2*mypred2$se, col = "blue", lty = 2)
lines(1:9, mypred2$pred-2*mypred2$se, col = "blue", lty = 2)
```