CBOE Volatility Index (VIX) Time Series Forecasting

Forecasting VIX Using Time Series Methods in R

Reynaldo Perez

University Of California, Santa Barbara - Fall 2022

Abstract

The purpose of this project is to create the most accurate forecast with my current capabilities in order to predict VIX index values. VIX is a crucial market index for investors, as it anticipates future market volatility, and is a reflection of investor uncertainty and expected future price fluctuations in the U.S stock market. The values of the VIX index attempts to measure how much volatility the S&P 500 index will experience. It essentially uses the weighted prices of the S&P 500 index, which is where its values are derived from.

In order to effectively develop a model that accurately forecasts VIX index values, I split the data into training and testing sets so that we can train our data and test our predictions using the testing set. Next, I observed the training data and decided to perform a box-cox transformation in order to make the data stationary. However, I also observed that the data had trend a seasonality, which must be removed in order to perform proper forecasting. Therefore, I removed these components by differencing the data twice. Afterwards, I ensured my data was stationary and contained no trend nor seasonality by plotting the ACF and PACF of the data and checking for normality. After applying the described transformations, it was time to select a proper model to perform forecasting on. By analyzing the ACF and PACF of the differenced data, checking for invertibility, and performing diagnostic checking, I concluded that a SARIMA$(1,1,0)$X$(1,1,1)_8$ model would be suitable for forecasting. Lastly, I forecasted the next values and compared them to the true values of the full data to ensure that our model was accurate. Based off my forecasts, I concluded that my model semi-accurately predicted the true VIX values.

Introduction

The aim of this project is to perform time series forecasting on the CBOE Volatility Index, otherwise known as VIX. To put it briefly, the Chicago Board Options Exchange Volatility Index, or VIX, operates as an indicator of economic risk. Specifically, VIX is an index that reflects the market's expectations for volatility, which is the measure of an asset's variation. It indicates how much volatility is in the overall market. This is why the VIX is also known as the "fear index", because it represents the market's expectations for volatility. The VIX is derived from the price movements of the S&P 500 index, which is another stock market index that tracks large-cap stocks, and is overall the essential benchmark index for the U.S stock market. In fact, the S&P 500 index and VIX values often show inverse price action; when the S&P rises, the VIX falls, and vice versa.

Given this description, we can see the importance of the VIX. VIX values can foreshadow a failing market if those values are increasing. For example, if the VIX values were to be 30 or more, this would reflect large volatility linked with increased uncertainty, risk, and investor fear. Lower VIX values, say below 20, correspond to a more stable market and thus, lower fears. By predicting this index, we can predict market volatility, which is useful when traders are selecting a security investment. Traders will be able to understand risk more, and be able to determine relative risk of a potential trade.

In my time series analysis and forecasting, I will be using historical VIX data in order to conduct our

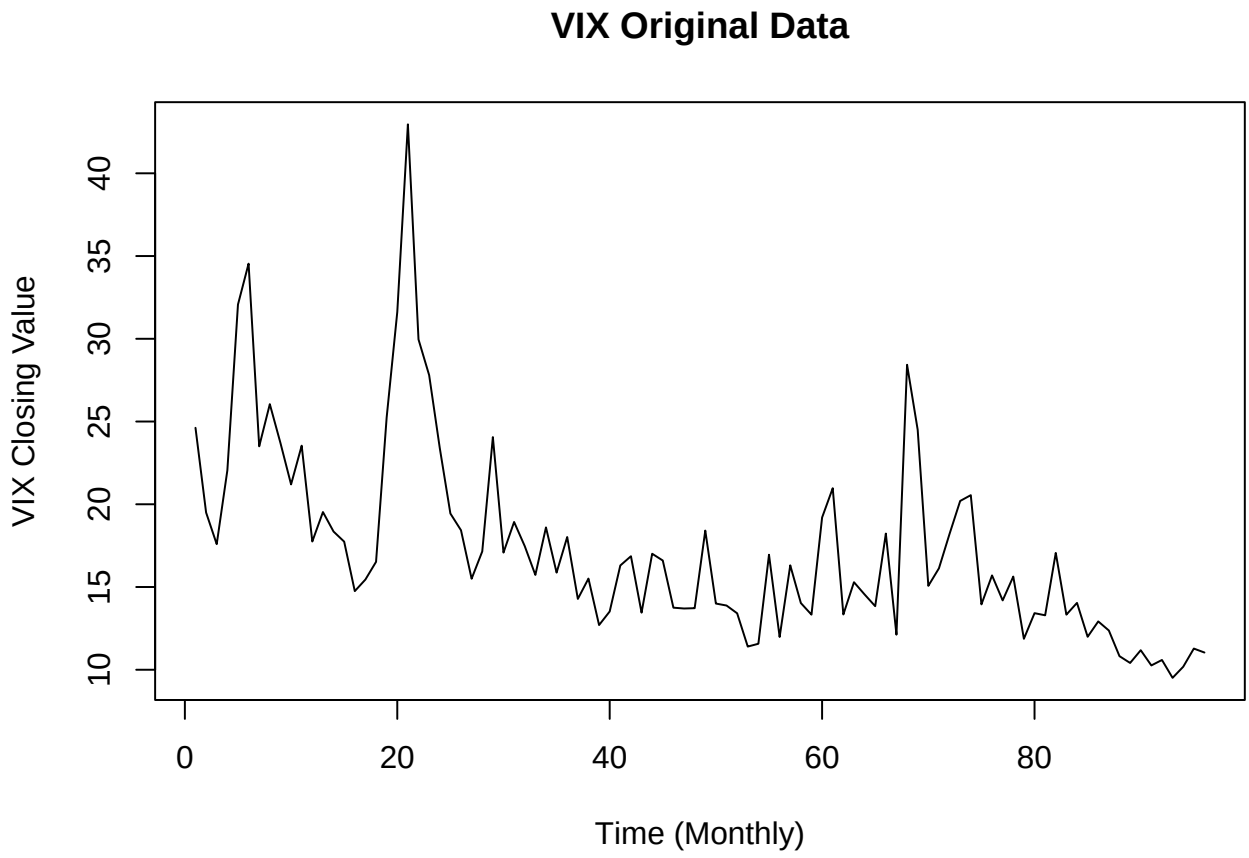
forecasting. This data contains monthly values from January 1st, 2010, to December 1st, 2017. I conveniently obtained this data from the **Federal Reserve Economic Data (FRED) API**, which updates the daily, monthly, and yearly closing values of VIX.

To start off, I split the data into training and testing sets. I cut off the last 12 observations of the data and used them as the testing set. Afterwards, I decided to apply a box-cox transformation to the training data after observing a skewed distribution. I then successfully removed the trend and seasonality that I observed in the decomposed data. I ensured this was successful by observing the data's distribution and plotting the ACF and PACF. Next, I performed model selection by analyzing the ACF and PACF of the differenced data, and concluded that a SARIMA$(1,1,0)$X$(1,1,1)_8$ was the best model to use for forecasting, as it had the lowest Aikaike's Information Corrected Criterion (AICc), and passed all diagnostic checks such as the the Shapiro-Wilk test and Box-Ljung test. With this, I proceeded to perform forecasting with the chosen model, and observed that it semi-accurately predicted the true data. For a volatility index, the forecasts were much more accurate than I expected, as all prediction points lied within the confidence intervals, and also did a moderately well job at predicting trend over the next months.

Time Series Analysis

Plotting Raw Data

To begin our time series analysis, we will plot our raw data in order to get an idea of the shape of the graph.



Before we perform any type of analysis, we need to split our data so that we have a better understanding of our model.

Data Split and Examination of Training Set

As mentioned, we'll be splitting our data into training and testing sets so that we can perform model training and validation. We will partition the last 12 observations as our testing set, as our goal is to forecast the VIX values for the next year.

Let's check the dimensions of the training and testing sets to ensure we have a proper split.

```
# We need to ensure with have a proper split
length(vix_train)
```

```
## [1] 84
```

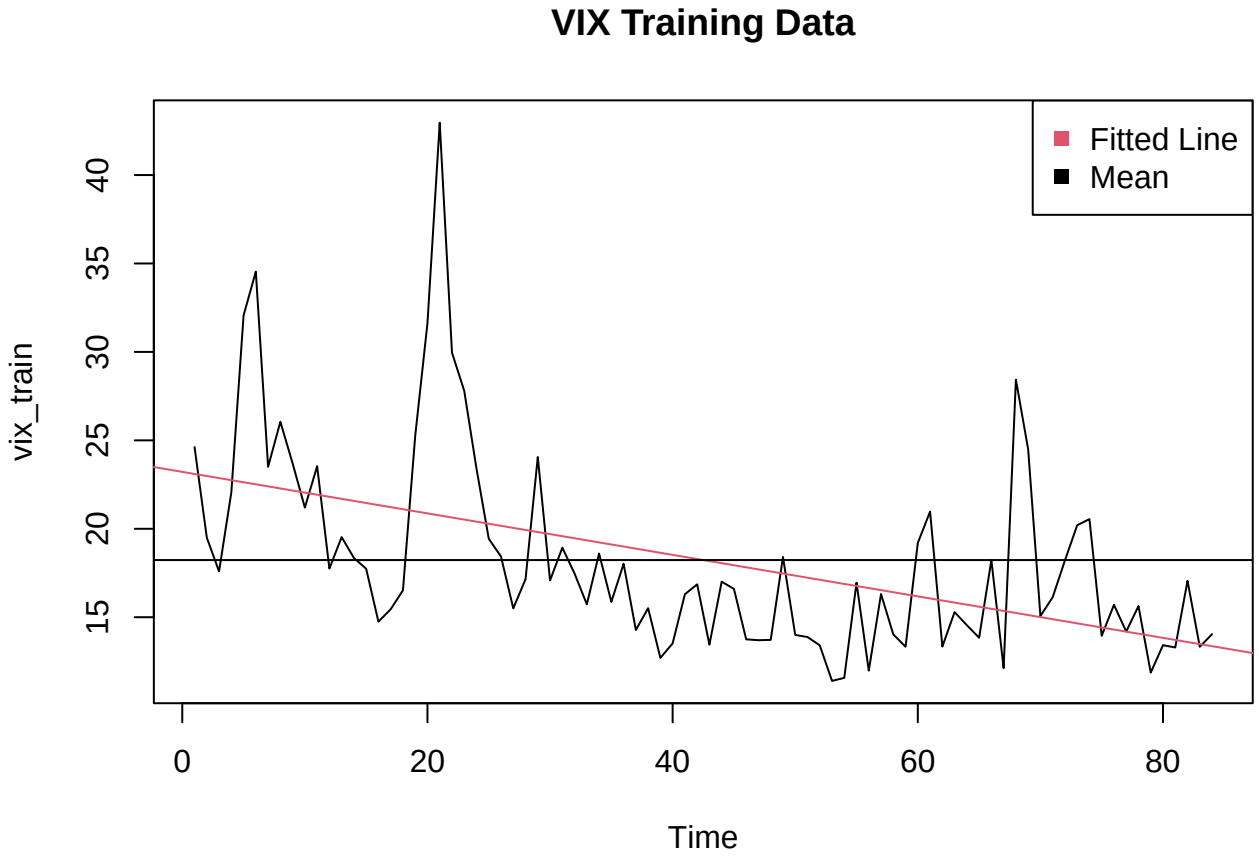
```
length(vix_test)
```

```
## [1] 12
```

Out of a total of 96 observations in the original data, 84 is in the training set, and 12 is in the testing set.

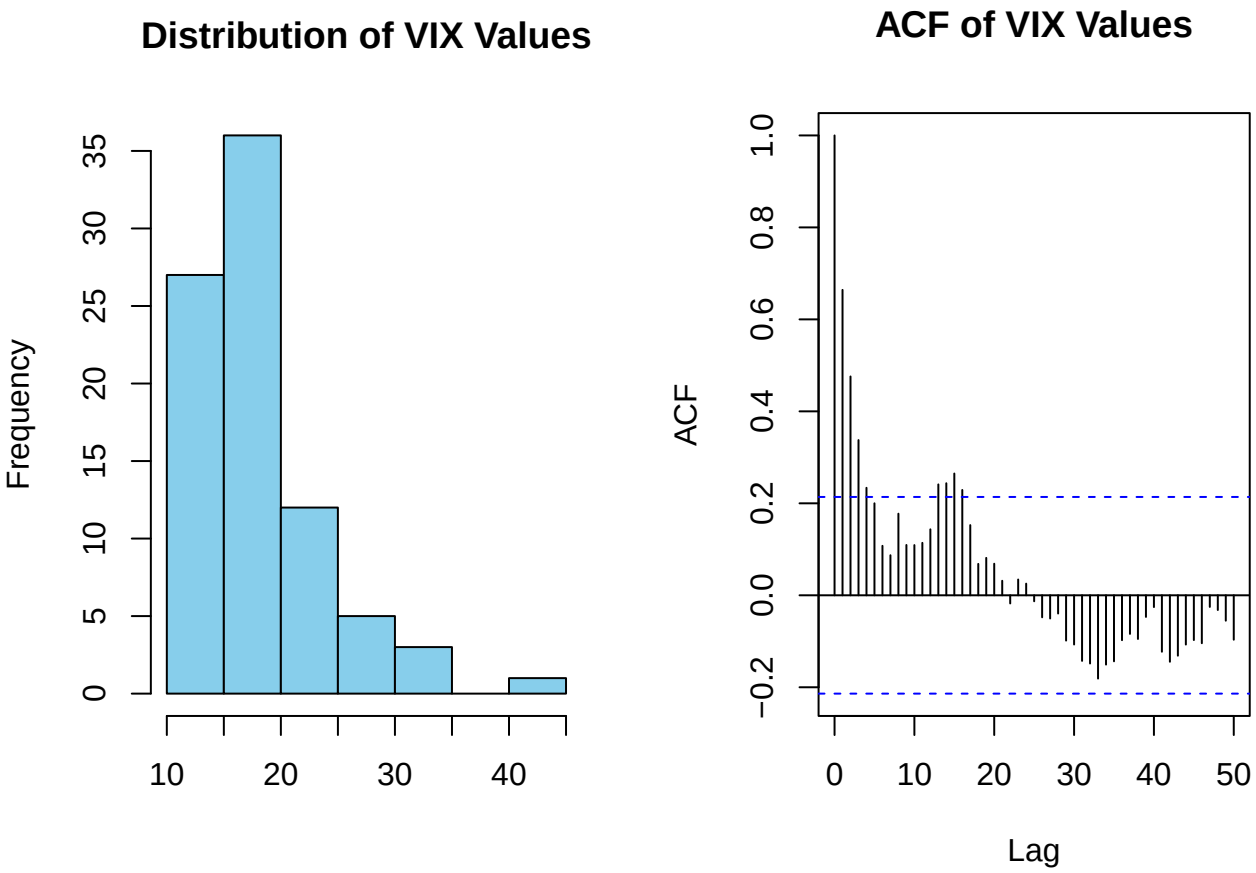
Data Exploration of the Training Set

Now that we've split our data, we can finally perform our time series analysis. To start off, we'll graph our training set.



We immediately notice that the data is highly non-stationary with a peak at around mid-2011. There also appears to be a downward linear trend in the data. That means the market has been improving overtime in this timeframe (lower VIX values reflect a healthier overall market). Furthermore, there seems to be a strong seasonal component in the data.

Let us now check the distribution of our data with a histogram and plot the ACF.



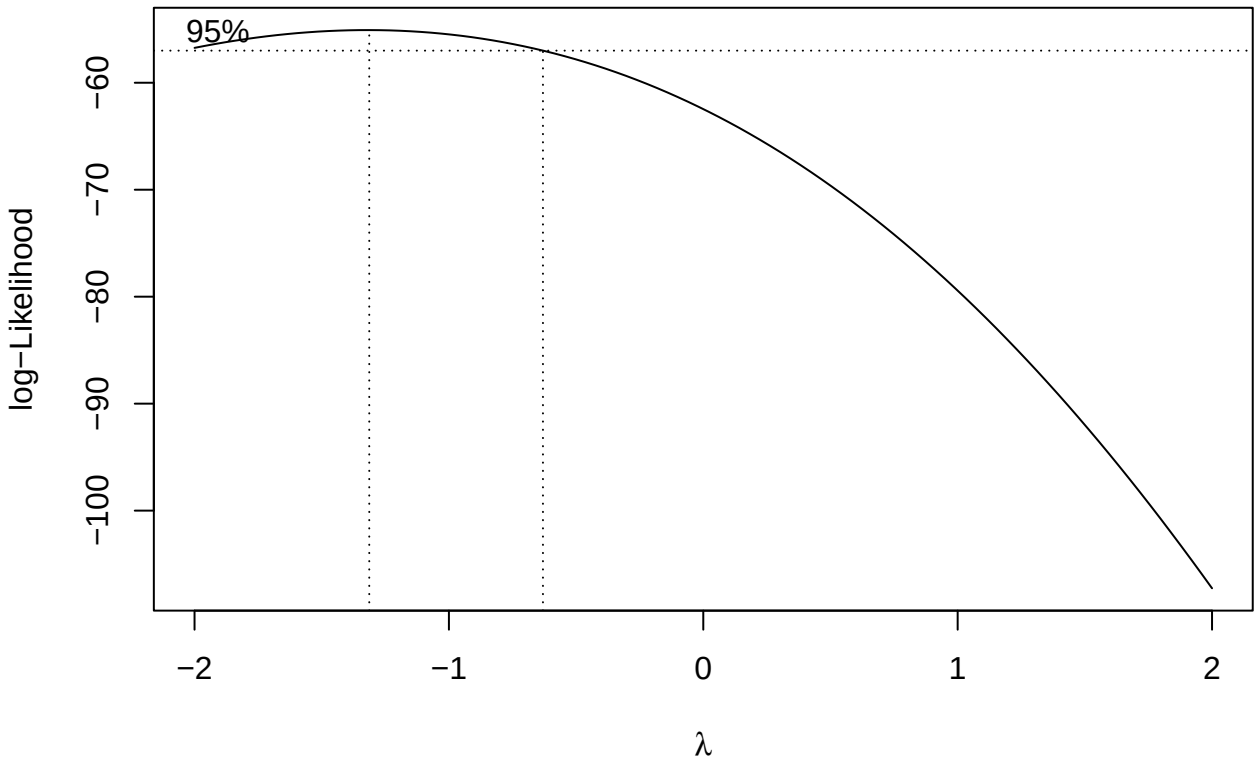
Observe that our histogram has a right-skew, and our ACF is large and periodic. Furthermore, our ACF is slowly decaying which indicates non-stationarity. We'll need to transform our data.

Data Transformation

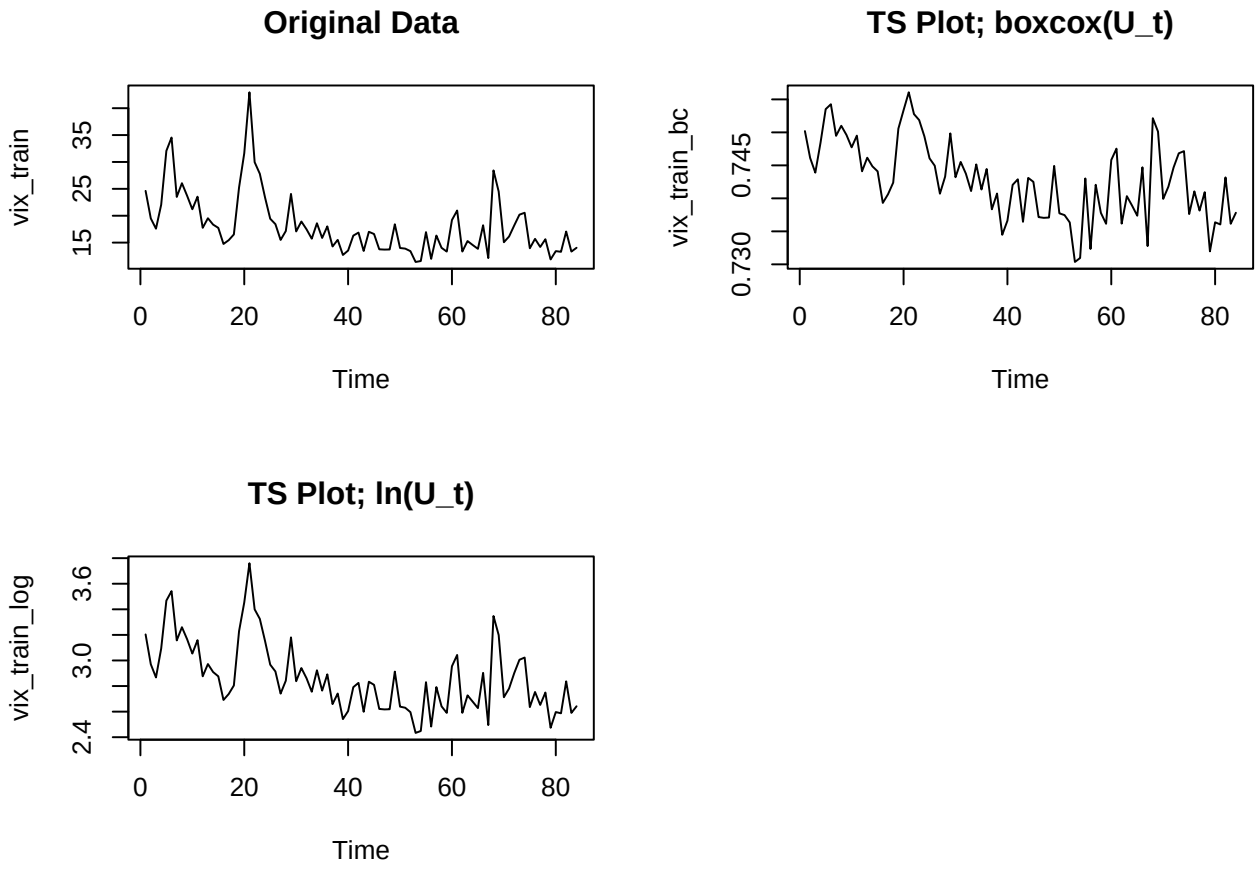
Based off our previous plots, our data is highly nonstationary with trend, seasonality, and nonstable variance. We cannot perform forecasting with these statistical properties, and therefore, we must transform and difference our model. This will allow us to work with a stationary model with no trend or seasonality.

Stationarity

Since the distribution of VIX values are skewed, and the variance is unstable, I believe a Box-Cox transformation is suitable for this situation. The idea behind this is that we are finding some value λ such that the transformed data is as close to normally distributed as possible.

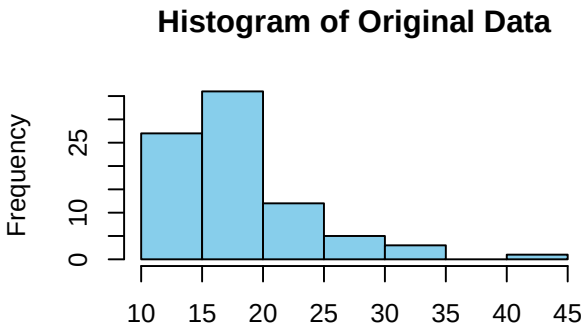


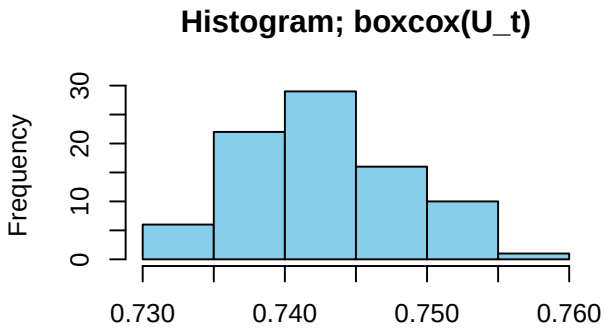
Based on the box-cox transformation and the value $\lambda = -1.313$, a suitable transformation would be $Y_t = \frac{1}{\lambda}(X_t^{\lambda} - 1) \implies Y_t = -\frac{1}{1.313}(X_t^{-1.313} - 1)$. Although it wasn't implied in the box-cox transformation, I also took the log of the data to compare it with the box-cox transformation. However, I expect box-cox to be the better transformation.

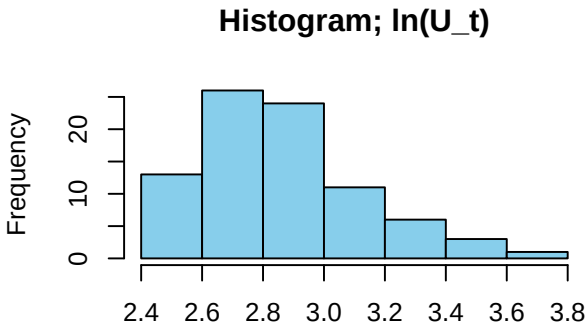


Between the original data and the log-transformed data, there does not seem to be much of a difference. The variance is not much stable, and there's almost no difference. In contrast, the box-cox transformed data seems to be more stabilized, and there are also less sharp dips and peaks compared to the original data.

Let's now have a look at our transformed distributions.





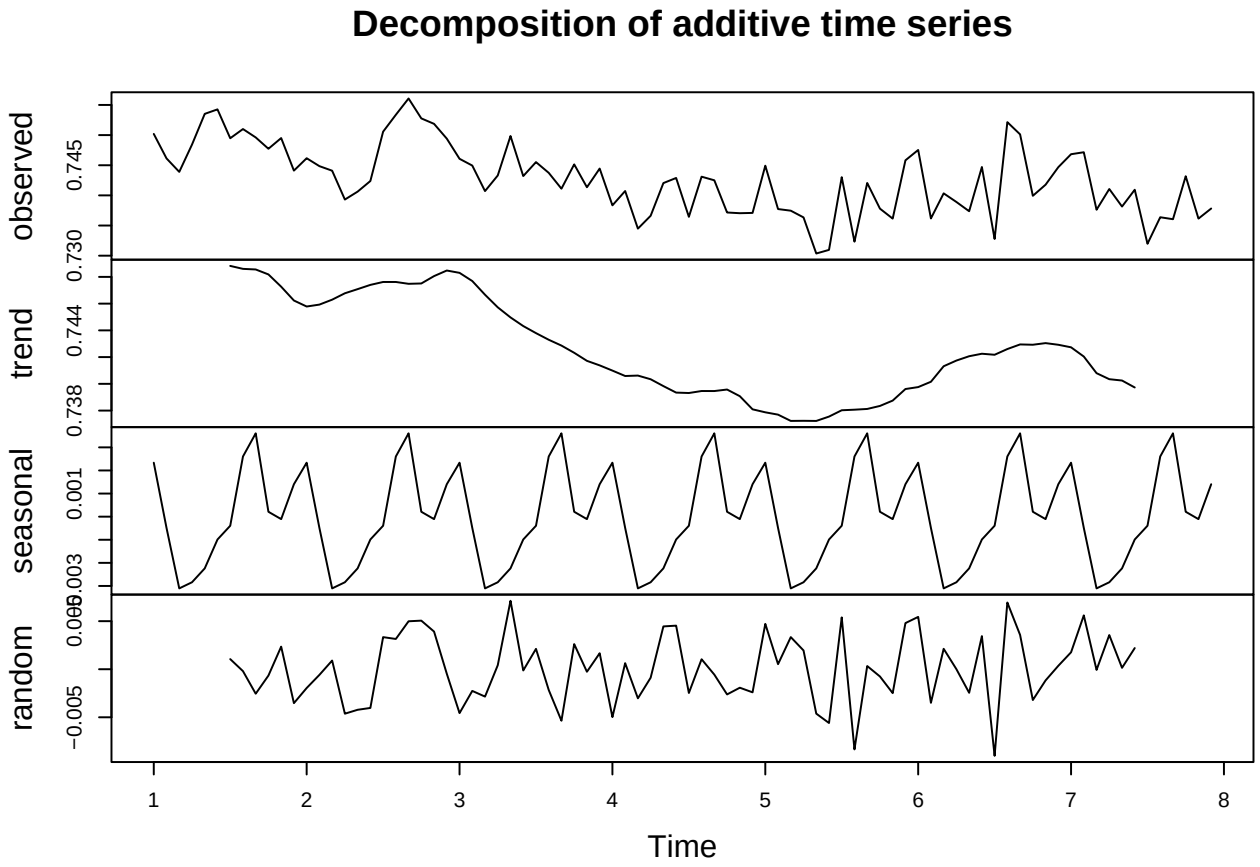


As expected, the box-cox is our candidate transformation. The log-transformed data still has a right-skew like our original data, whereas the box-cox transformation has an approximately normal distribution and is overall more symmetric.

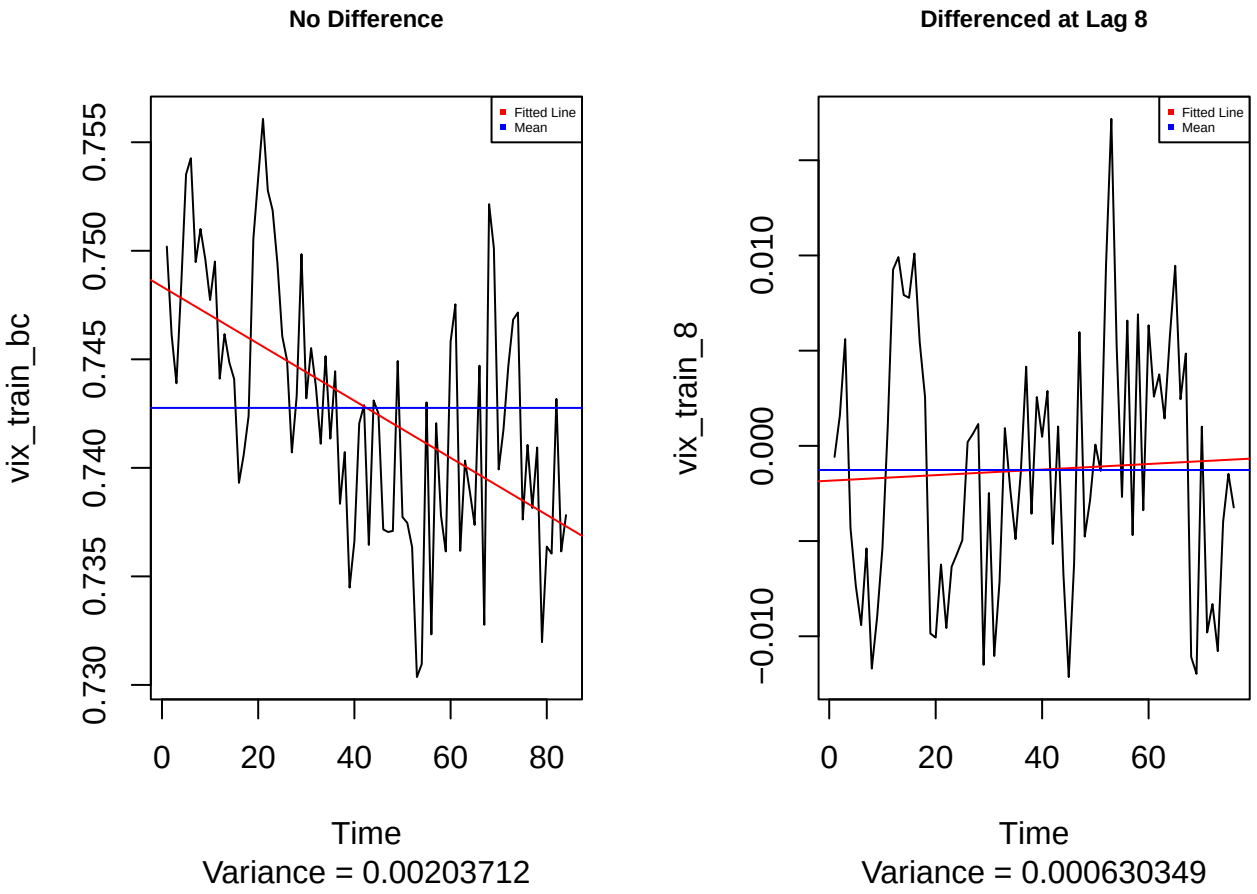
Now that we have made our model stationary, it is time we remove the trends and seasonal components of our data.

Differencing

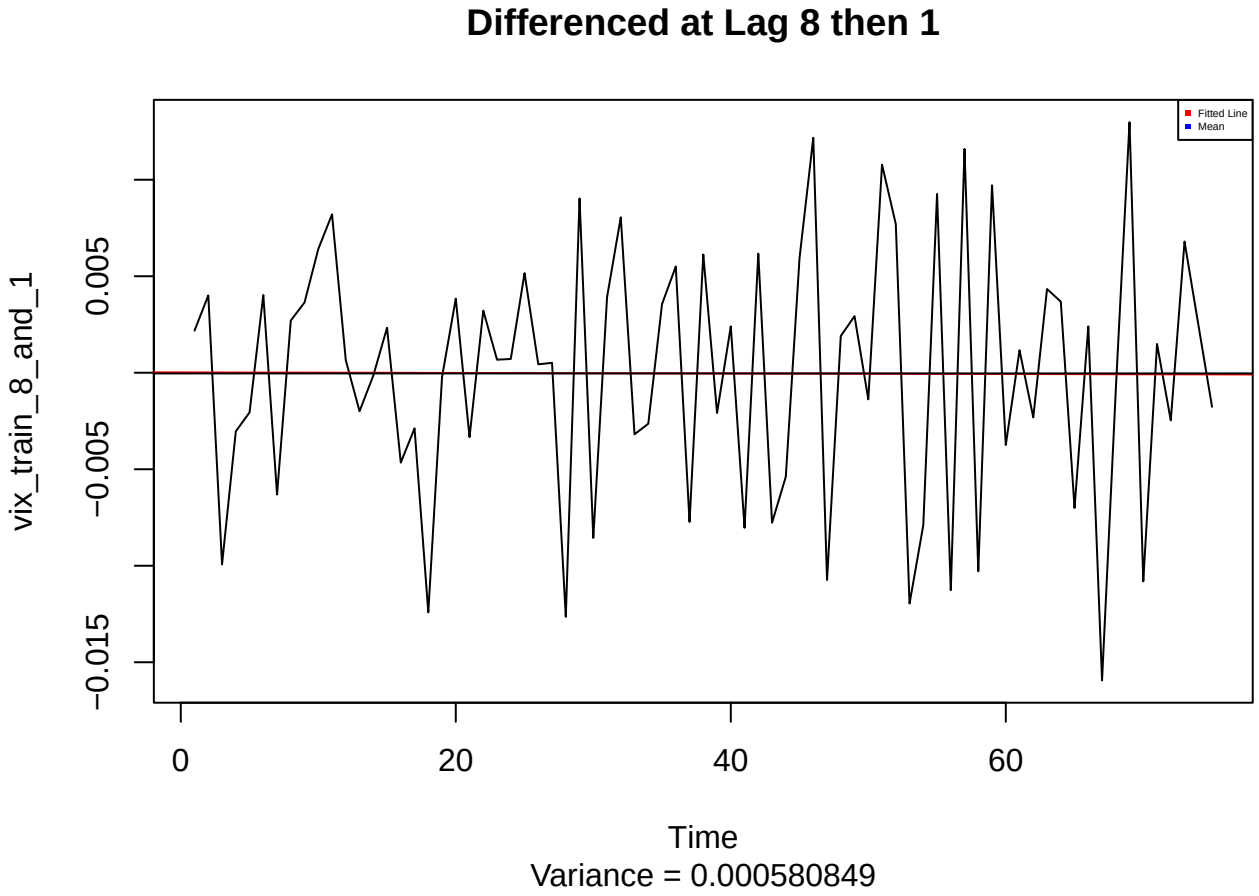
Let's produce a decomposition of our box-cox transformation, boxcox(U_t), to check for trend and seasonality.



Upon observation of the decomposition, there seems to be a non-linear trend and clear seasonality. The `random` decomposition also has a large interval, meaning our data will be difficult to deal with, but this is to be expected with stock data. In either case, we will need to remove the trends and seasonality within our model before we perform forecasting.



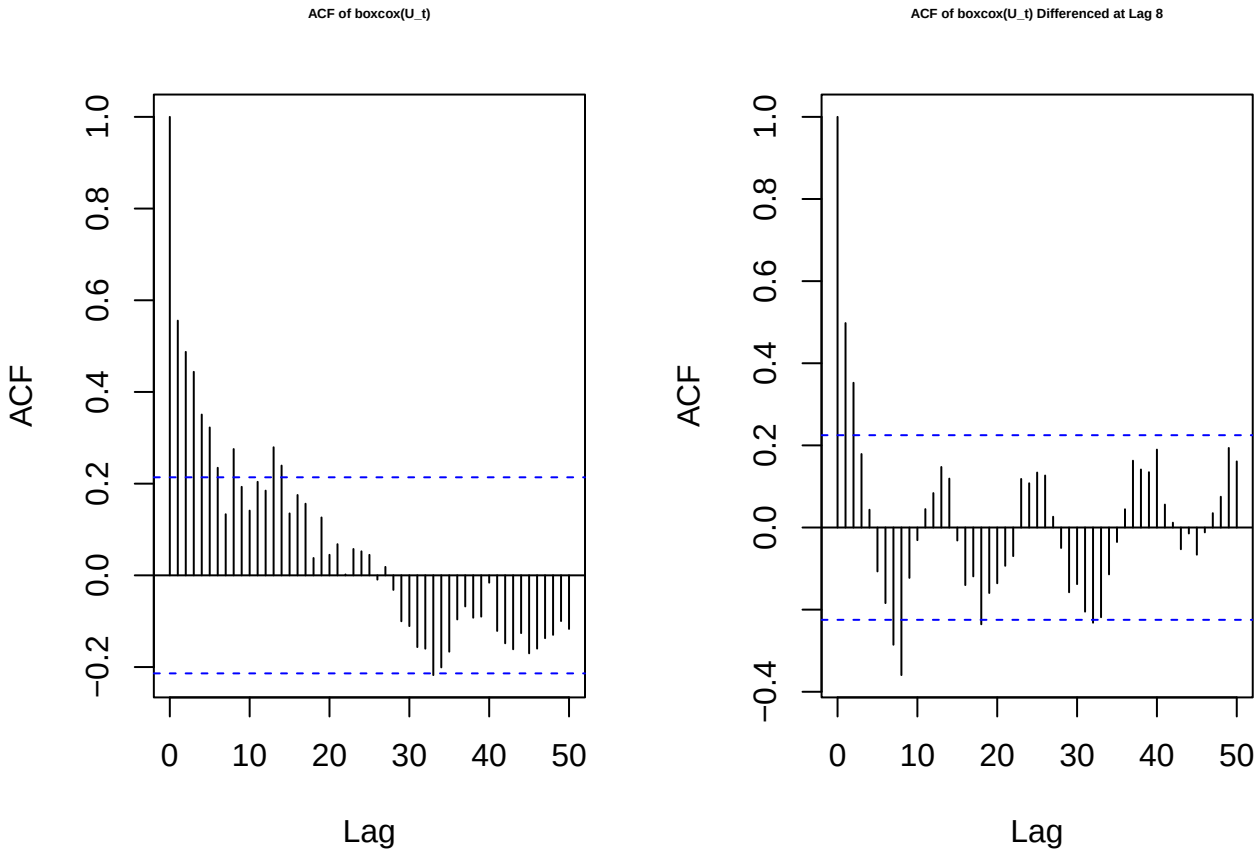
By differencing at lag 8, we see that we no longer have seasonality, and our variance is lower. However, there is still a slightly apparent trend line. To get rid of this, we will be differencing again, but at lag 1.



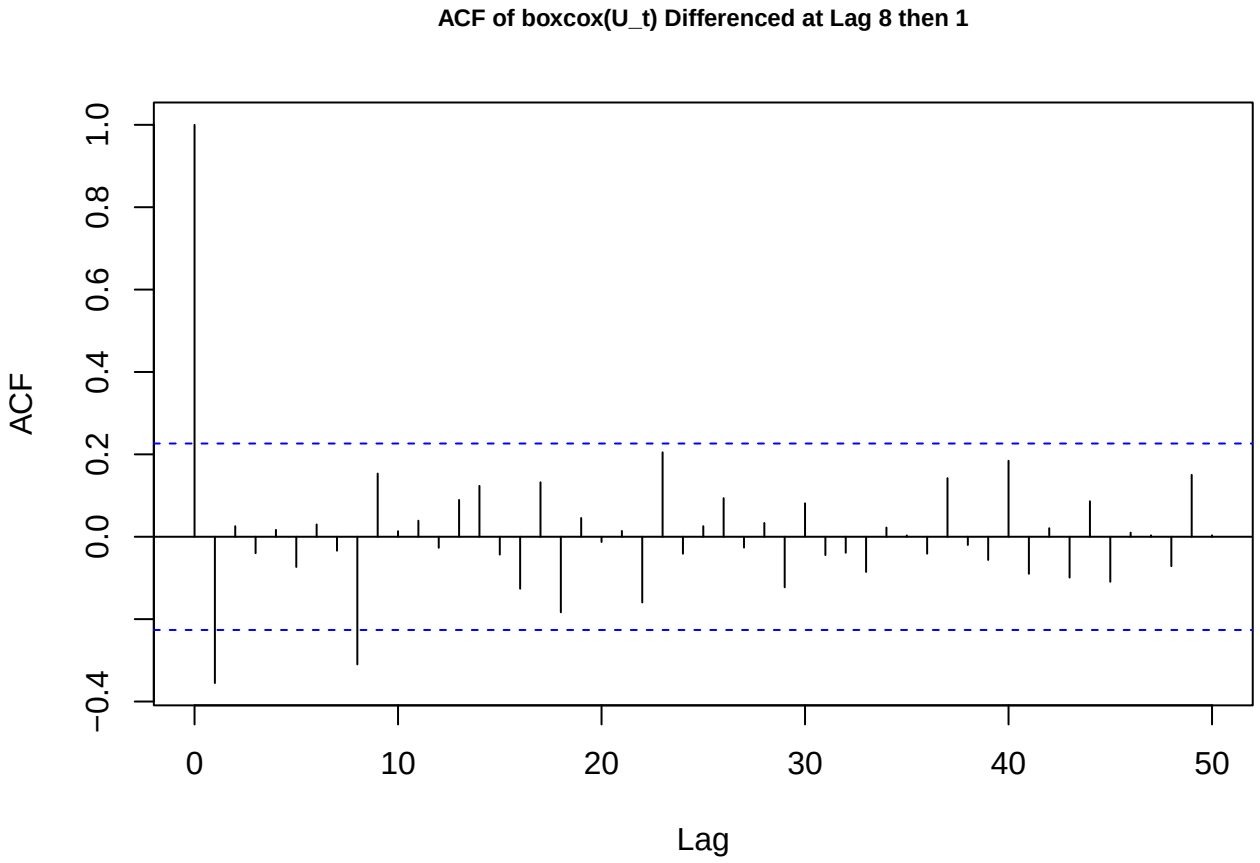
By differencing twice, both by 8 then 1 respectively, we receive a time series plot that has no seasonality, no trend, and much lower variance than the original data. Therefore, differencing at lags 8 and 1 unveiled the lowest variance for the time series while also completely removing trend and seasonality, which is why I chose those lags to difference by.

Now that we have made our time series stationary with no apparent trend or seasonality, let's plot its ACF and PACF to ensure our model is appropriate for proper fitting and forecasting.

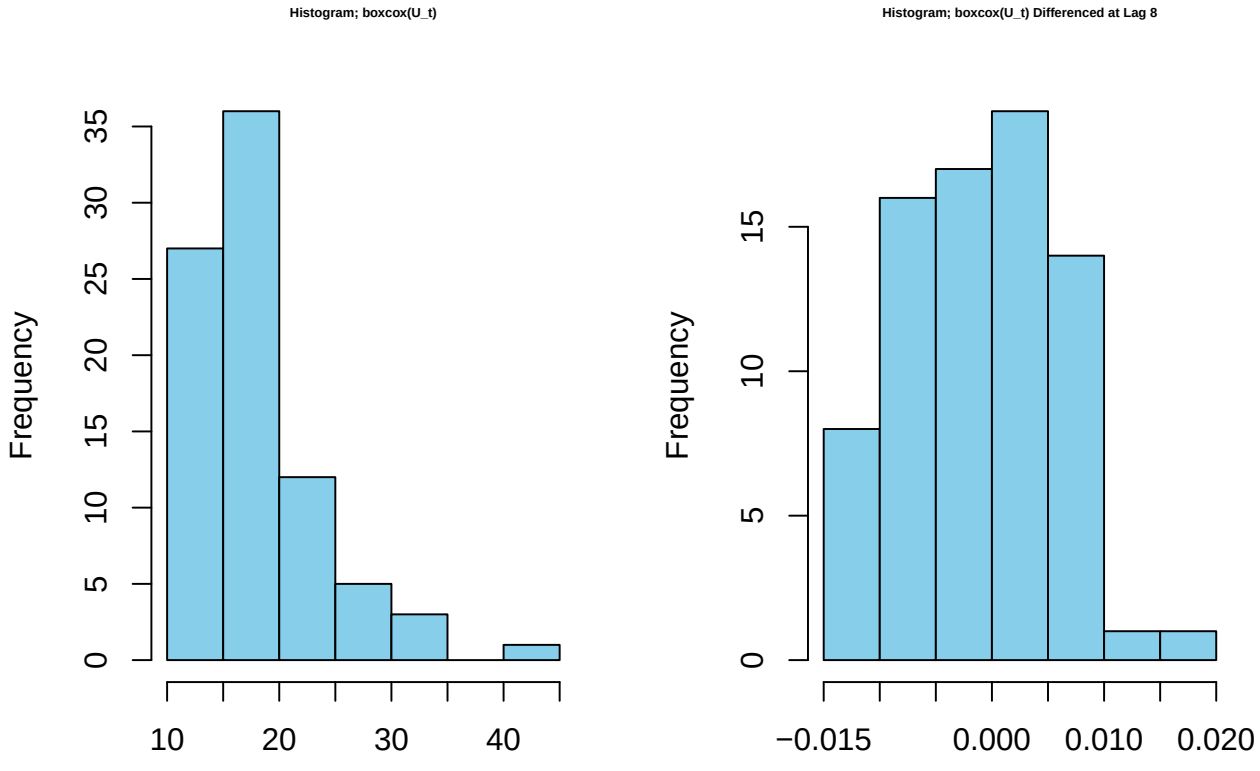
First, let's plot the ACFs. We'll be plotting the ACF of our other models for comparison.

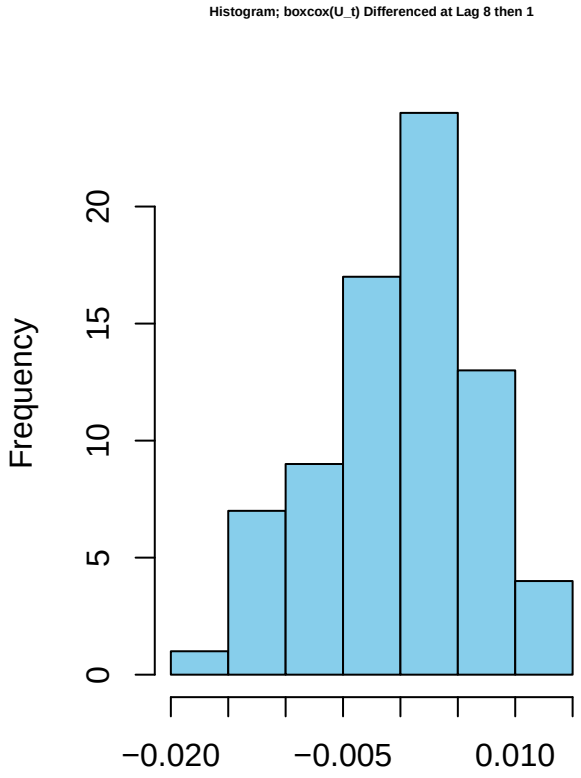


Observe the ACF on the left. Clearly, one sees slow decay as well as seasonality. By differencing once at lag 8, we see reduced seasonality. We will apply differencing once more, this time at lag 1.



The ACF quickly decays to zero, therefore corresponding to a stationary process with no trend or seasonality. Let's now see the distributions of our differenced data.

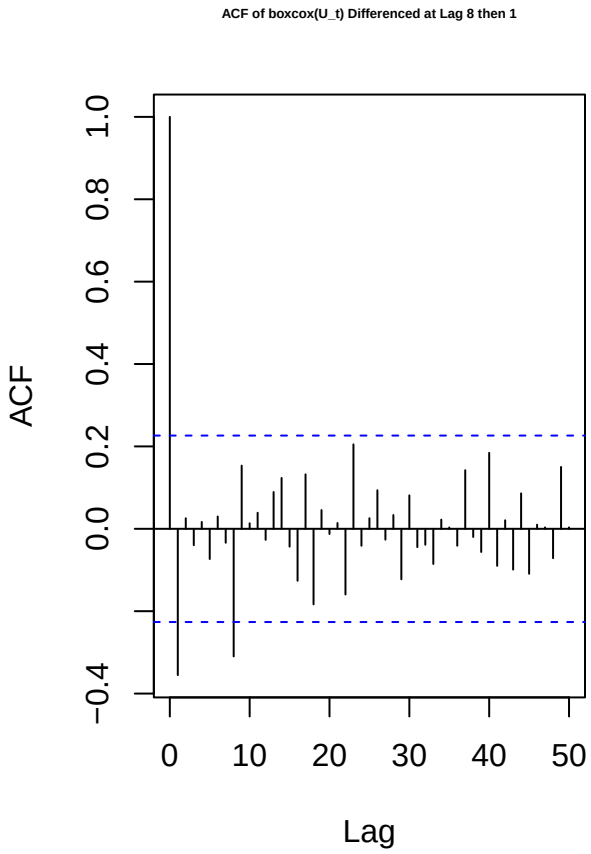


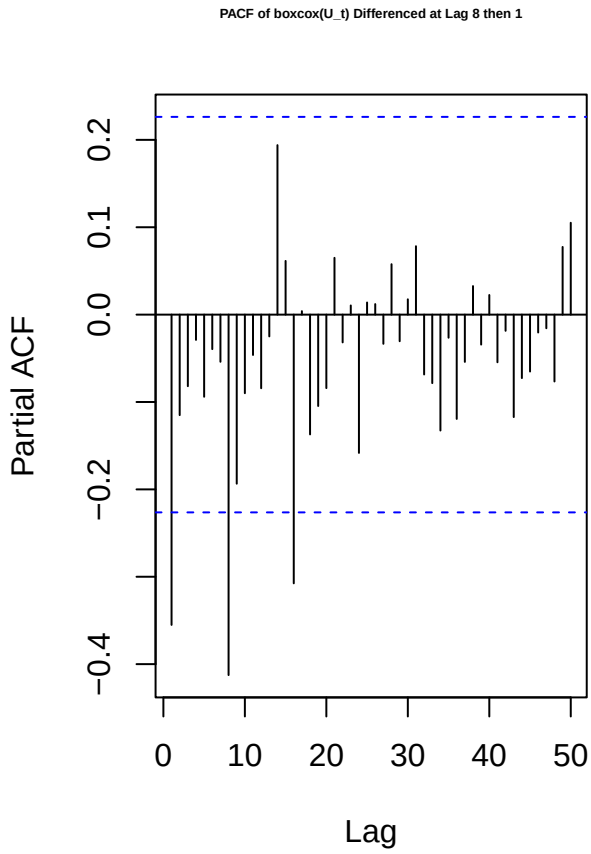


Comparing these distributions, we see that the distribution of the twice-differenced data follows a more Gaussian distribution compared to the other histograms. In addition, its ACF follows a more stationary process as mentioned. Hence, we will be using the twice-differential data to build our model off of, that is, $\nabla_1 \nabla_8 boxcox(U_t)$.

Model Selection

Now that we've transformed and differenced our data, let's try selecting a suitable model for forecasting. This can be obtained by analyzing the ACF and PACF of the new data.





Upon close inspection, the ACF seems to go outside its confidence interval at lags 0, 1, and 8.

On the other hand, the PACF is outside the confidence intervals at lags 1, 8, and 16.

From this, let's try to come up with candidate models. We'll be building a SARIMA model for boxcox(U_t). We'll start off with modeling the seasonal part of our models, (P, D, Q).

- We applied one seasonal differencing, so $D = 1$ at lag $= 8$.
- The ACF shows a large peak at lag $= 0$, followed by smaller peaks at lag $= 1$ and lag $= 8$. A good choice for the moving average (MA) part of our model is $Q = 0$ or $Q = 1$.
- The PACF shows strong peaks at lags 1, 8. Perhaps $P = 1$ would be suitable for the autoregressive (AR) part of the model.

Now, let's set up the non-seasonal part of our model, (p, d, q).

- We also differenced to remove trend, so $d = 1$.
- ACF trails off at lag 0 and cuts off at lag 1. Suggest $q = 0$ or $q = 1$.
- PACF seems to cut off at lag $= 1$ and lag $= 8$. Suggest $p = 1$.

Now, we test these models and choose one with the lowest Second-order Akaike Information Criterion (AICc).

We'll be testing $P = 1$, $D = 1$, $Q = 0, 1$, $p = 1$, $d = 1$, $q = 0, 1$, with $s = 8$, where s is the period.

```
##
## Call:
## arima(x = vix_train_bc, order = c(1, 1, 0), seasonal = list(order = c(1, 1,
##     0), period = 8), method = "ML")
##
## Coefficients:
##          ar1    sar1
##       -0.405  -0.388
```

```
## s.e.   0.107   0.110
##
## sigma^2 estimated as 3.24e-05:  log likelihood = 280.5,  aic = -555
```

```
## [1] -554.7
```

```
## [1] Highest AICc
```

```
##
## Call:
## arima(x = vix_train_bc, order = c(1, 1, 0), seasonal = list(order = c(1, 1,
##     1), period = 8), method = "ML")
##
## Coefficients:
##          ar1   sar1    sma1
##       -0.425  0.136  -0.939
## s.e.   0.106  0.186   0.482
##
## sigma^2 estimated as 2.27e-05:  log likelihood = 288.1,  aic = -568.3
```

```
## [1] -567.7
```

```
## [1] Second lowest AICc
```

```
##
## Call:
## arima(x = vix_train_bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1,
##     0), period = 8), method = "ML")
##
## Coefficients:
##         ar1     ma1    sar1
##       0.489  -1.000  -0.337
## s.e.  0.104   0.057   0.111
##
## sigma^2 estimated as 2.87e-05:  log likelihood = 283.3,  aic = -558.6
```

```
## [1] -558.1
```

```
## [1] Third lowest AICc
```

```
##
## Call:
## arima(x = vix_train_bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1,
##     1), period = 8), method = "ML")
##
## Coefficients:
##         ar1     ma1   sar1    sma1
##       0.370  -0.884  0.158  -1.000
## s.e.  0.198   0.141  0.153   0.914
##
## sigma^2 estimated as 2.01e-05:  log likelihood = 290.2,  aic = -570.5
```

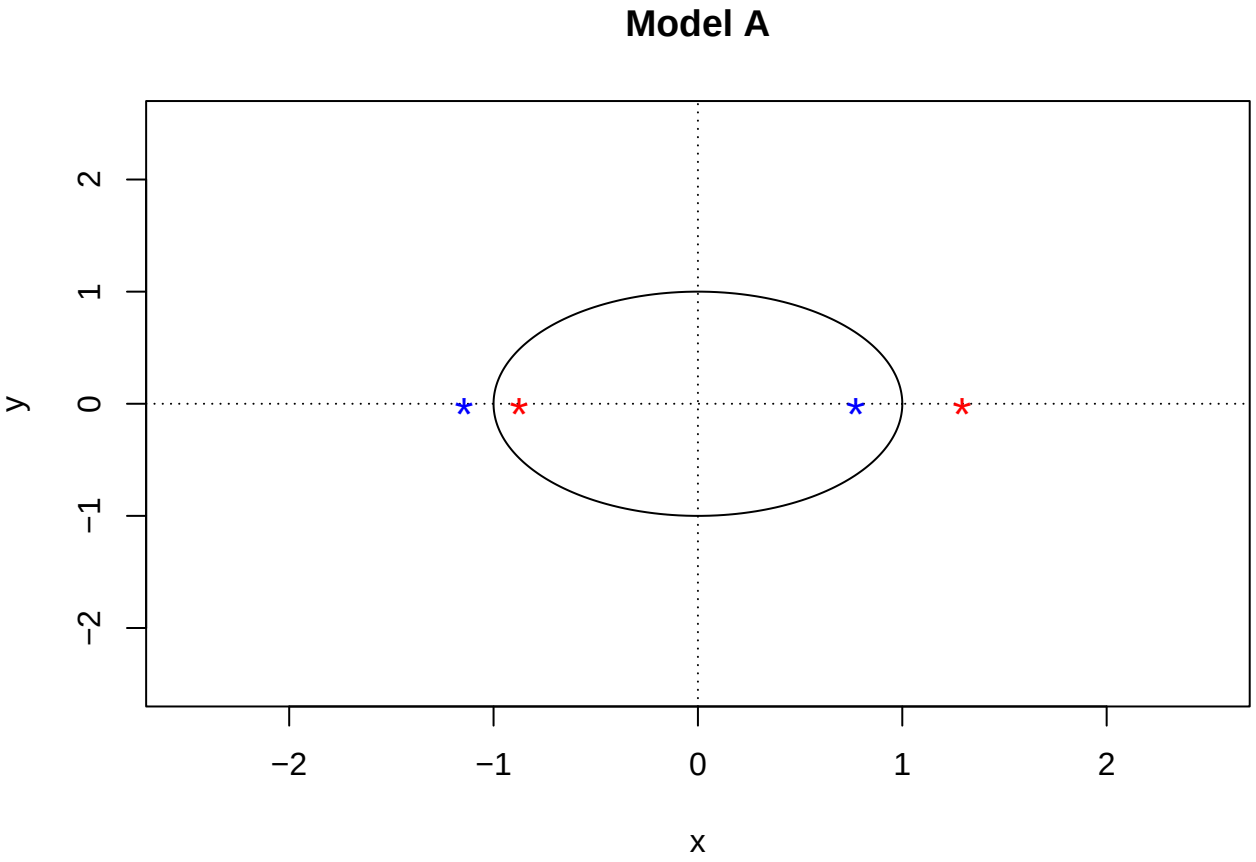
```
## [1] -569.6
```

```
## [1] Lowest AICc
```

We'll be choosing the two models with the lowest AICc's, that is:

- Model A - SARIMA$(1,1,1)$X$(1,1,1)_8$ with AICc $= -569.6$.
- Model B - SARIMA$(1,1,0)$X$(1,1,1)_8$ with AICc $= -567.7$.

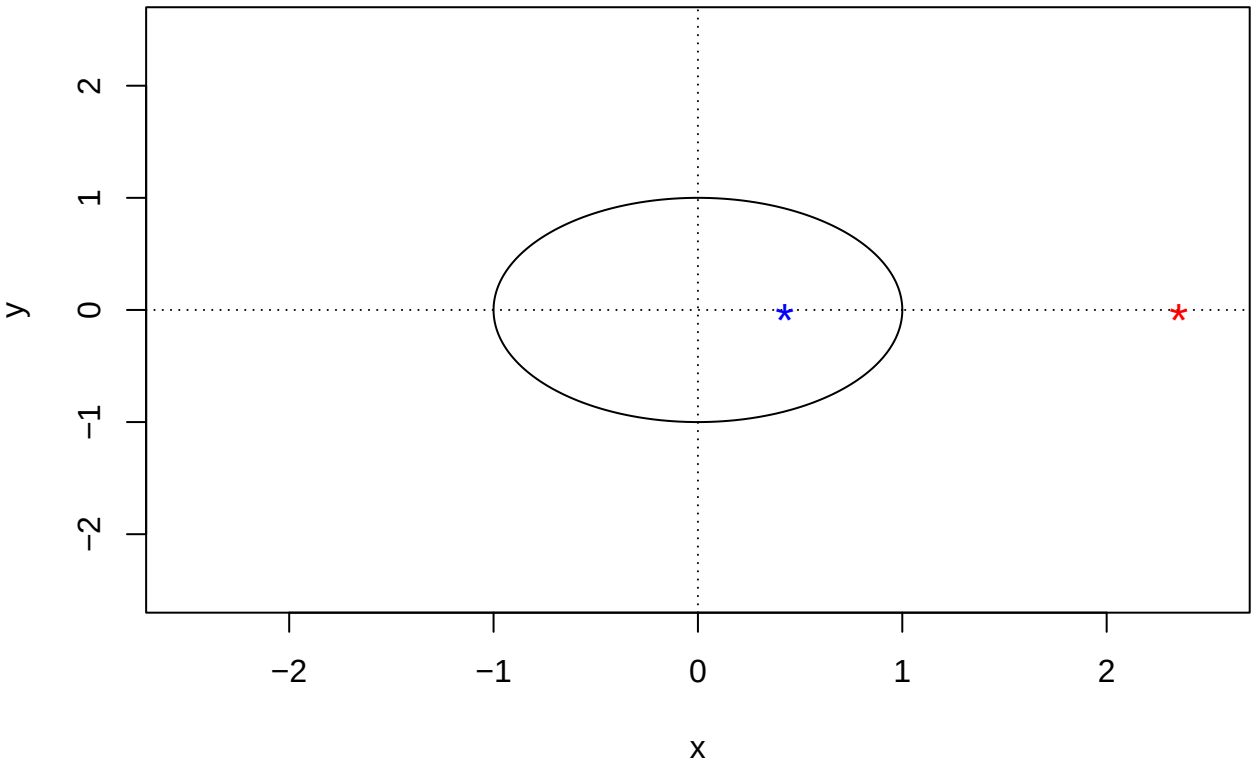
We'll be conducting unit root tests for these models to ensure invertibility. Note that the red asterisks denote the roots, and the blue asterisks denote the inverse roots. The roots should be outside of the unit circle. In contrast, the inverse roots should be less than one, or in other words, inside the unit circle, and neither roots cannot be on the unit circle.




```
## [1]  1.2933+0i -0.8747-0i
```

```
## [1] One root is inside the unit circle; model is not invertible!
```

Unfortunately, Model A is not invertible because $|\theta_1| > 1$. Furthermore, one of the roots is inside of the unit circle. We cannot perform forecasting with this model. Let's see if Model B passes the unit root test.




```
## [1] 2.353+0i
```

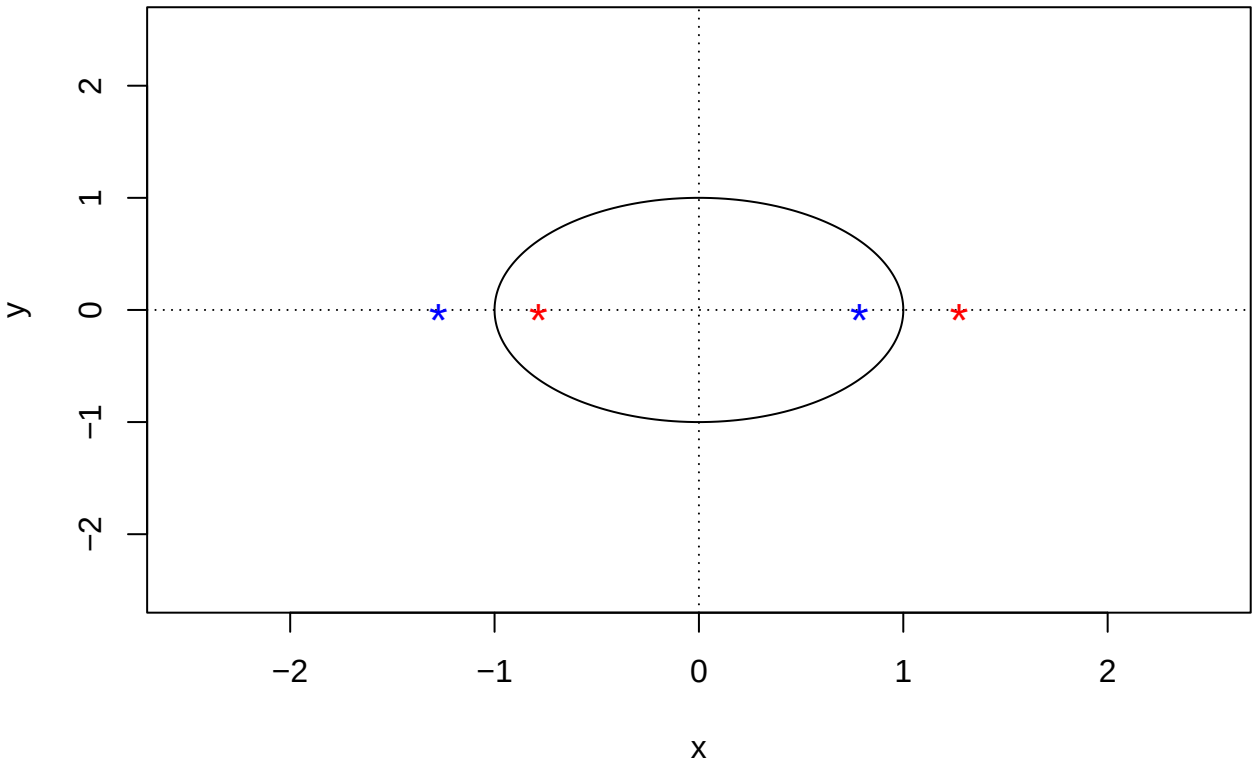
```
## [1] Root is outside; model is invertible
```

Thankfully, Model B passed the unit root test. Observe that $|\theta_1| < 0$. The root is outside the unit circle and the inverse root is within the unit circle. Thus, Model B is causal and invertible.

We saw that Model A didn't pass the unit test, hence we will not perform diagnostic testing on it. Since only one model will be moving on to diagnostic testing, let's check for invertibility in our other plausible models, that is,

- Model C - SARIMA$(1,1,1)$X$(1,1,0)_8$ with AICc $= -558.1$.
- Model D - SARIMA$(1,1,0)$X$(1,1,0)_8$ with AICc $= -554.7$.

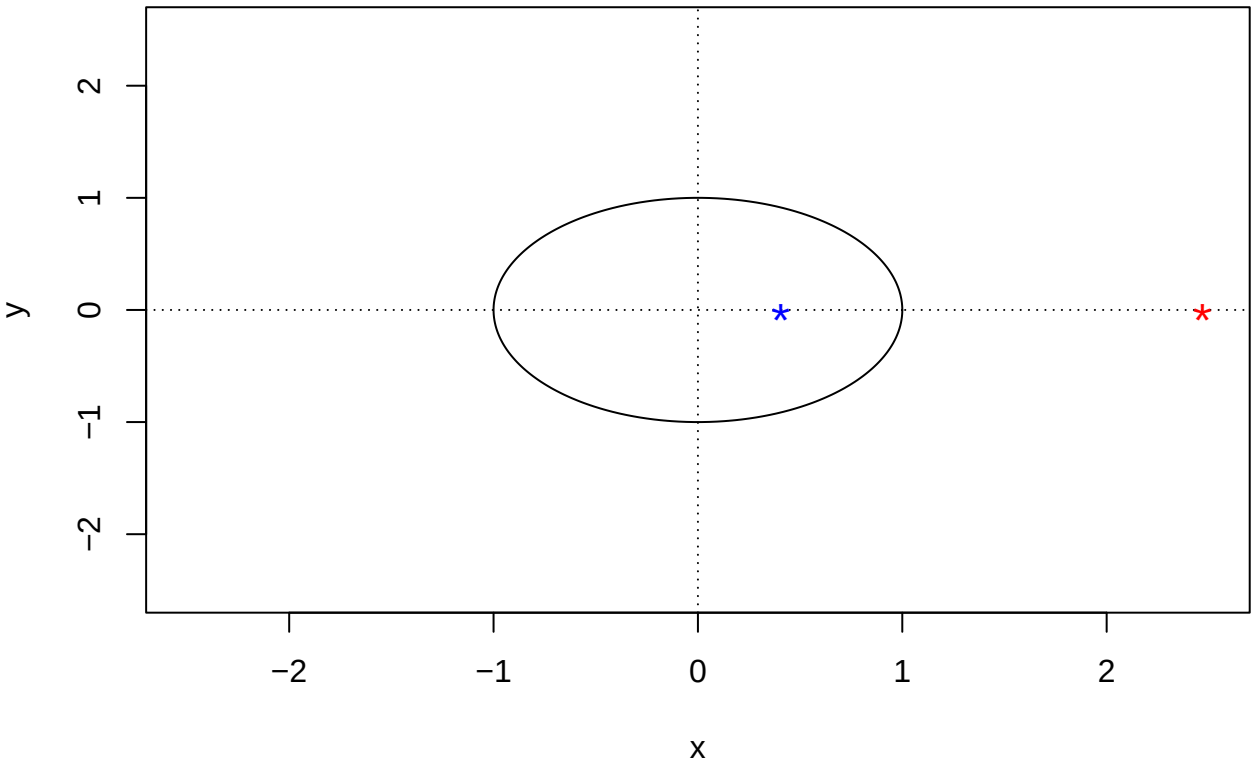
Let's check for invertbility on these models.




```
## [1]  1.274-0i -0.785+0i
```

```
## [1] One root is inside the unit circle, model is not invertible!
```

Here, there is one root that is within the unit circle. Therefore, this model is not invertible and cannot be used for proper forecasting.




```
## [1] 2.469+0i
```

```
## [1] Root is outside; model is invertible
```

We see that the root is outside the unit circle, and the inverse root within the unit circle. Therefore, it passes the unit root test and can move on for diagnostic checking.

Diagnostic Checking for Model B

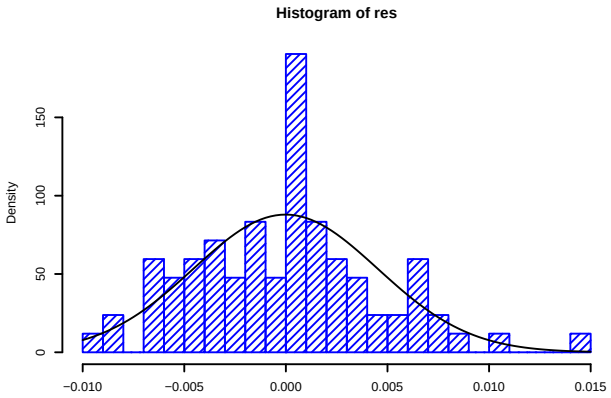
Models B and D were the only suitable models to proceed to diagnostic checking. We saw that models A and C did not pass the unit root test, as the models were not invertible. Let's try performing diagnostic checking on Model B first.

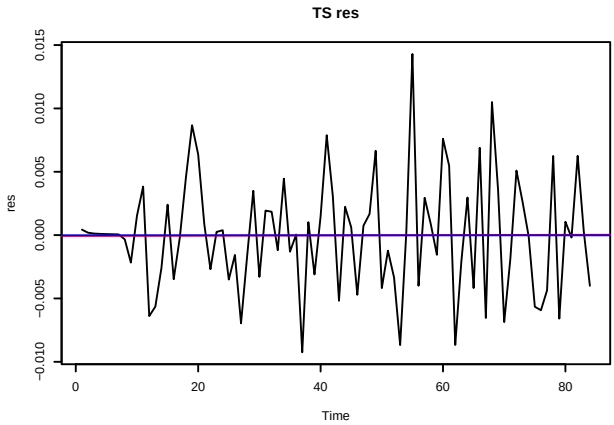
Recall that Model B is a SARIMA$(1,1,0)$X$(1,1,1)_8$ model with AICc $= -567.7$. We can write this model as

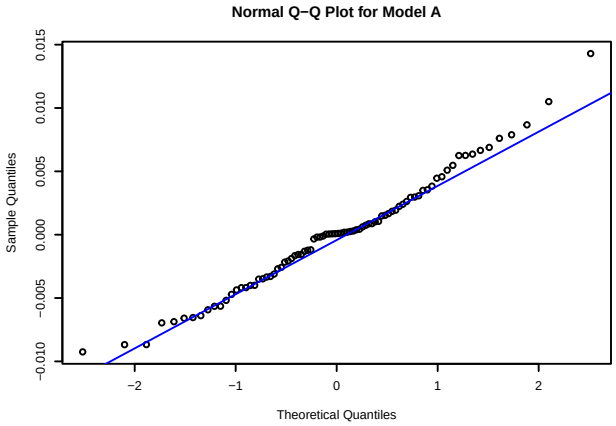
$\nabla_1\nabla_8 boxcox(U_t) = (1 - 0.405_{(0.107)}B)(1 + 0.136_{(0.186)}B^8)(1 - 0.939_{(0.482)}B^8)Z_t,$

where B is the backshift operator.

Let's check the distribution of our residuals.

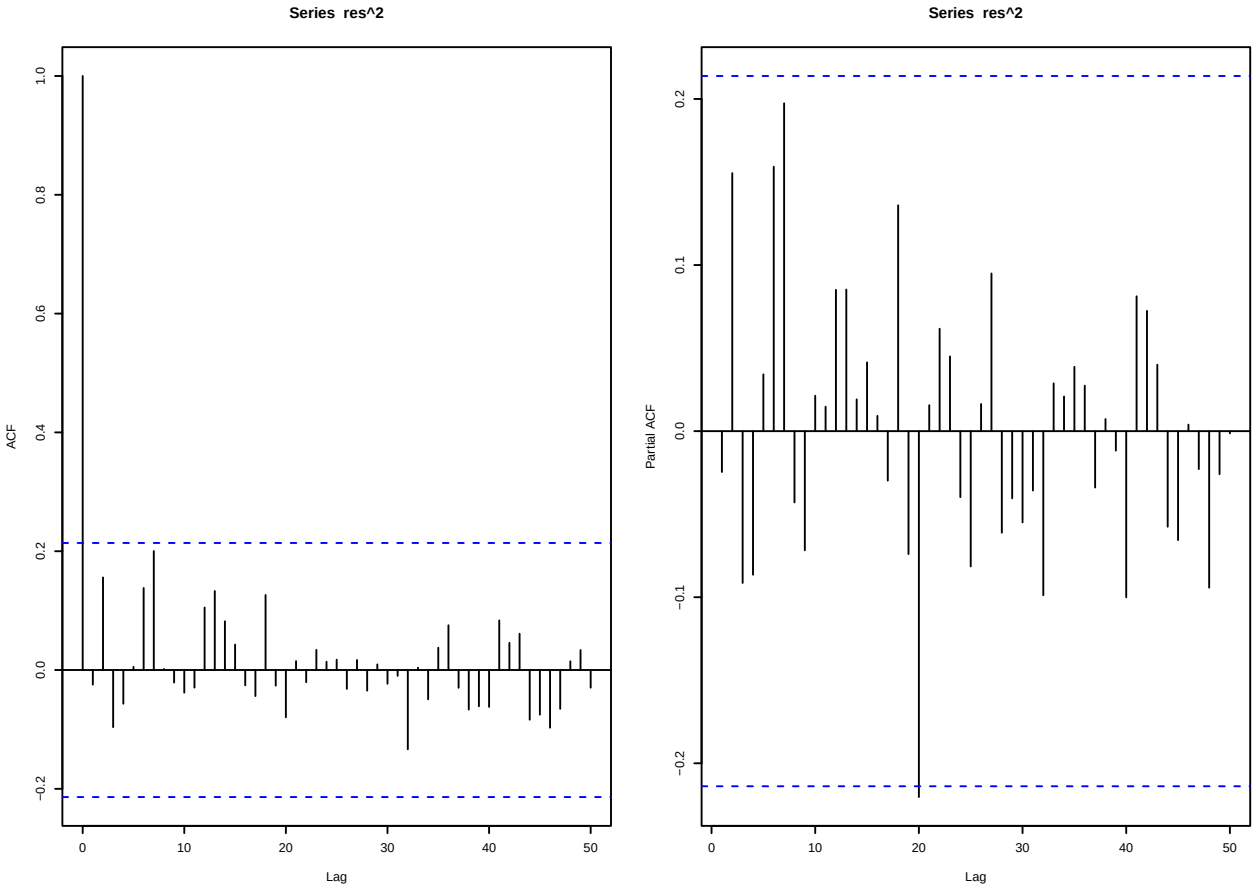






Notice there is no trend nor seasonality, and constant variance. The distribution is roughly normal, and the Q-Q plot seems well, as most of the points are aligned.

Let's check the ACF and PACF of the residuals.



All of the ACF and PACF residuals are within confidence intervals and can be counted as zeros.

We will now run the following tests:

- Shapiro-Wilk normality test
- Box-Pierce Test
- Box-Ljung

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.98, p-value = 0.4
```

```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 7.7, df = 6, p-value = 0.3
```

```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 8.4, df = 6, p-value = 0.2
```

```
##
##  Box-Ljung test
##
```

```
## data:  res^2
## X-squared = 8.9, df = 9, p-value = 0.4
```

All our tests reveal a p-value higher than 0.05, meaning all our tests pass, and we can forecast future values with this model.

Finally, let's check if our residuals follow Gaussian White Noise by using yule-walker estimation.

```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  2.06e-05
```

Our fitted residuals correspond to AR(0), which is a white noise process.

With these diagnostics, Model B is prepared to be used for forecasting. However, let's see if Model D can also be used for forecasting.

Diagnostic Checking for Model D

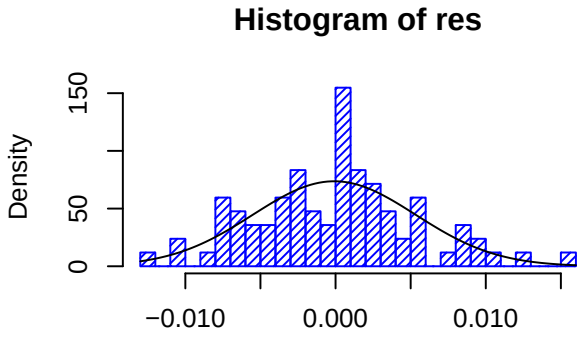
We also so that Model D proceeded for diagnostic checking, since the model is invertible.

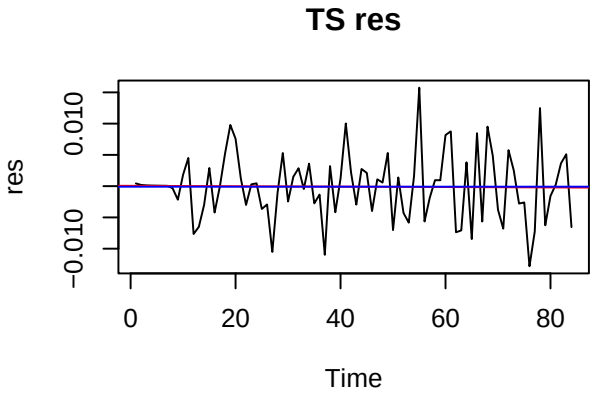
Recall that Model D is a SARIMA$(1,1,0)$X$(1,1,0)_8$ model with AICc $= -554.7$. We can write this model as

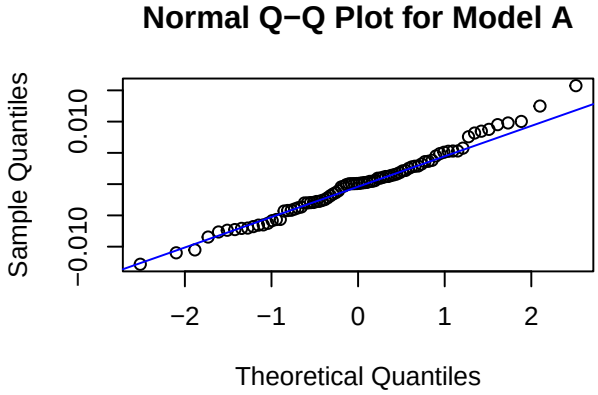
$\nabla_1\nabla_8 boxcox(U_t) = (1 - 0.405_{(0.107)}B)(1 - 0.388_{(0.110)}B^8)Z_t,$

where B is the backshift operator.

Let's check the distribution of the residuals.

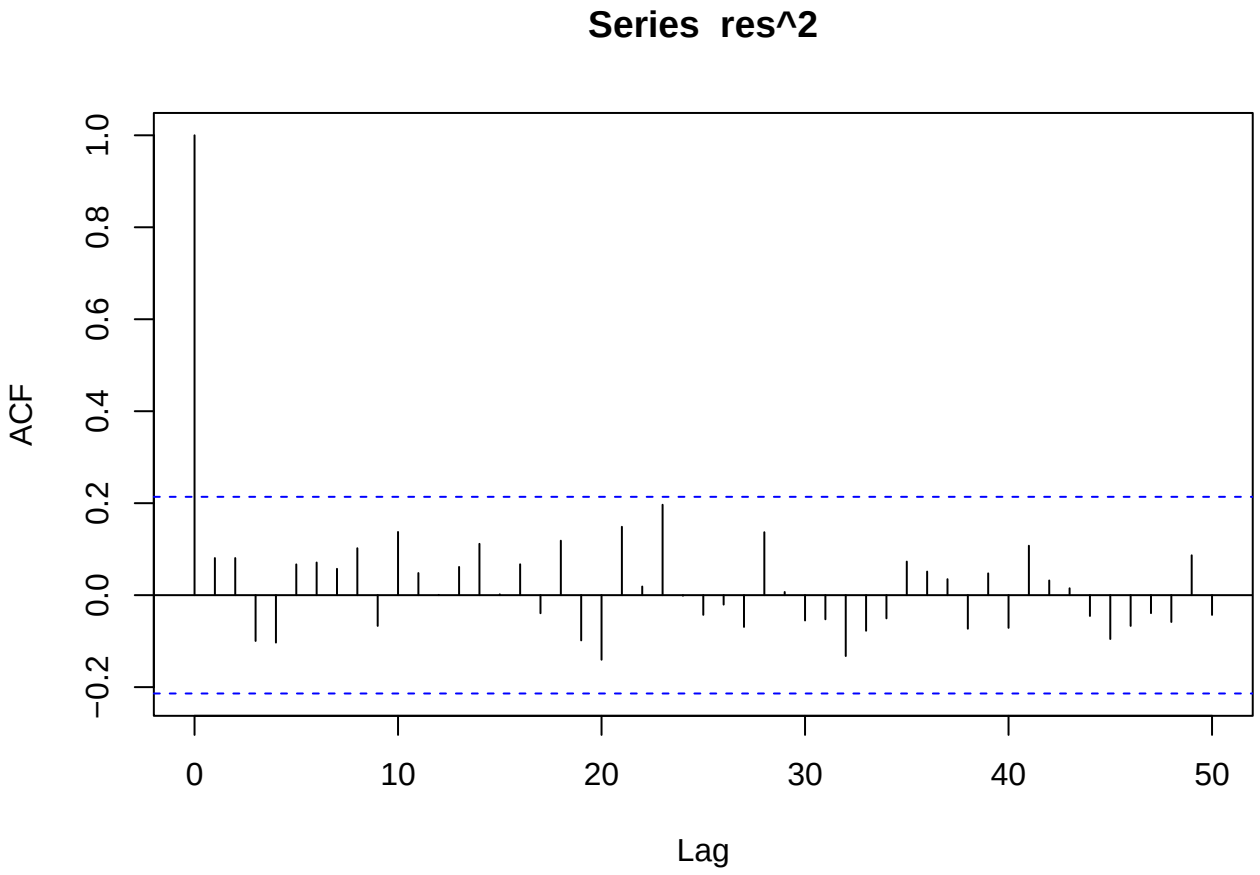


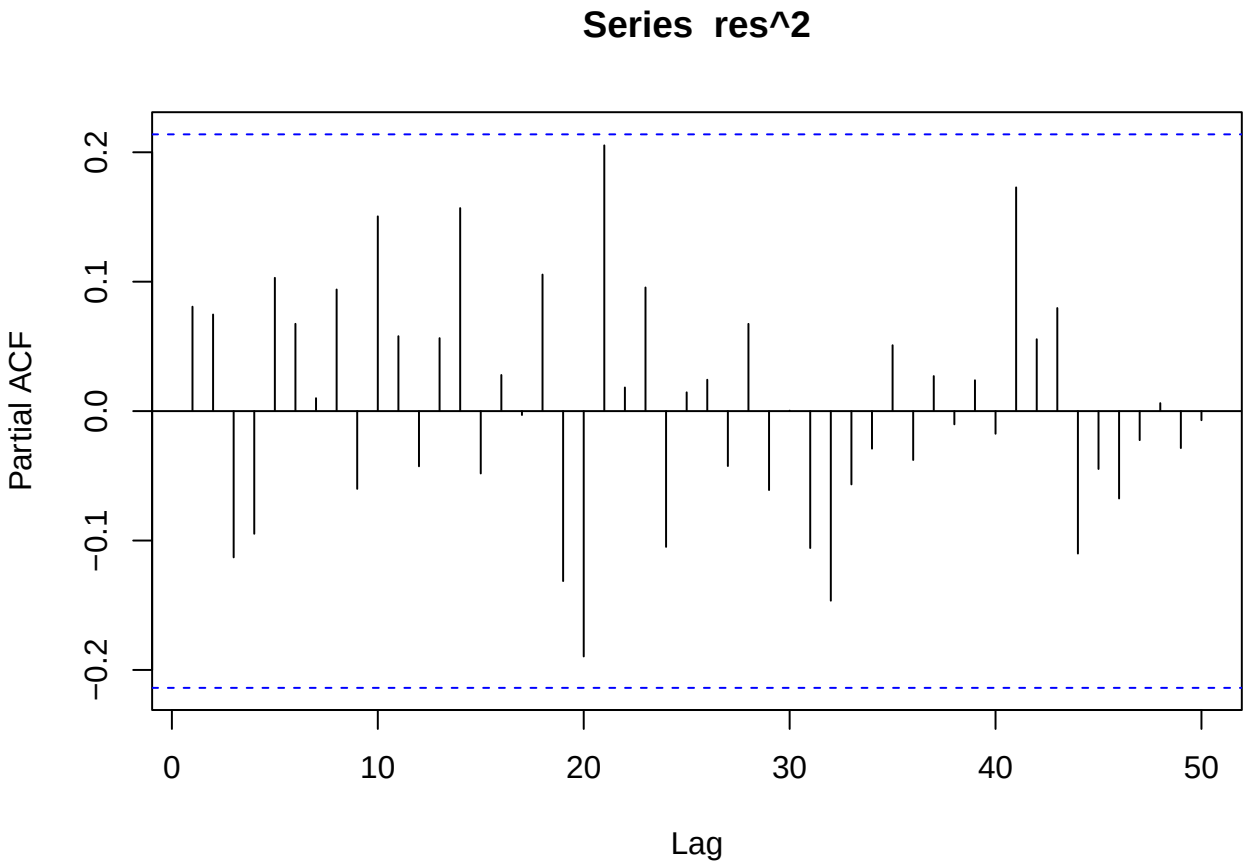




Observe that there is no trend or seasonality, and there is constant variance. The distribution is also roughly normal and Q-Q plot shows normality.

Let's check the ACF and PACF of the residuals.





Residuals on the ACF and PACF are within confidence intervals and can also be counted as zeros.

Let's run the same tests we ran for Model B.

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.99, p-value = 0.7
```

```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 9.1, df = 7, p-value = 0.2
```

```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 10, df = 7, p-value = 0.2
```

```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 5.6, df = 9, p-value = 0.8
```

All p-values are greater than 0.05, meaning our tests pass, and Model D can be used for forecasting.

Finally, we will check if the residuals follow Gaussian White Noise.

```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  2.93e-05
```

We see that our fitted residuals follow white noise.

Time Series Analysis Summary

After transforming our data, differencing, conducting unit root tests for invertibility, and performing diagnostics on our data, I have concluded that Model B and Model D are suitable for forecasting. However, our choice of model we will use for forecasting will be Model B because it has a lower AICc value, and also seemed to more closely resemble a normal distribution compared to Model D. Hence, we will be working with Model B. As a reminder, Model B is modeled as SARIMA$(1,1,0)$X$(1,1,1)_8$, and can be written as

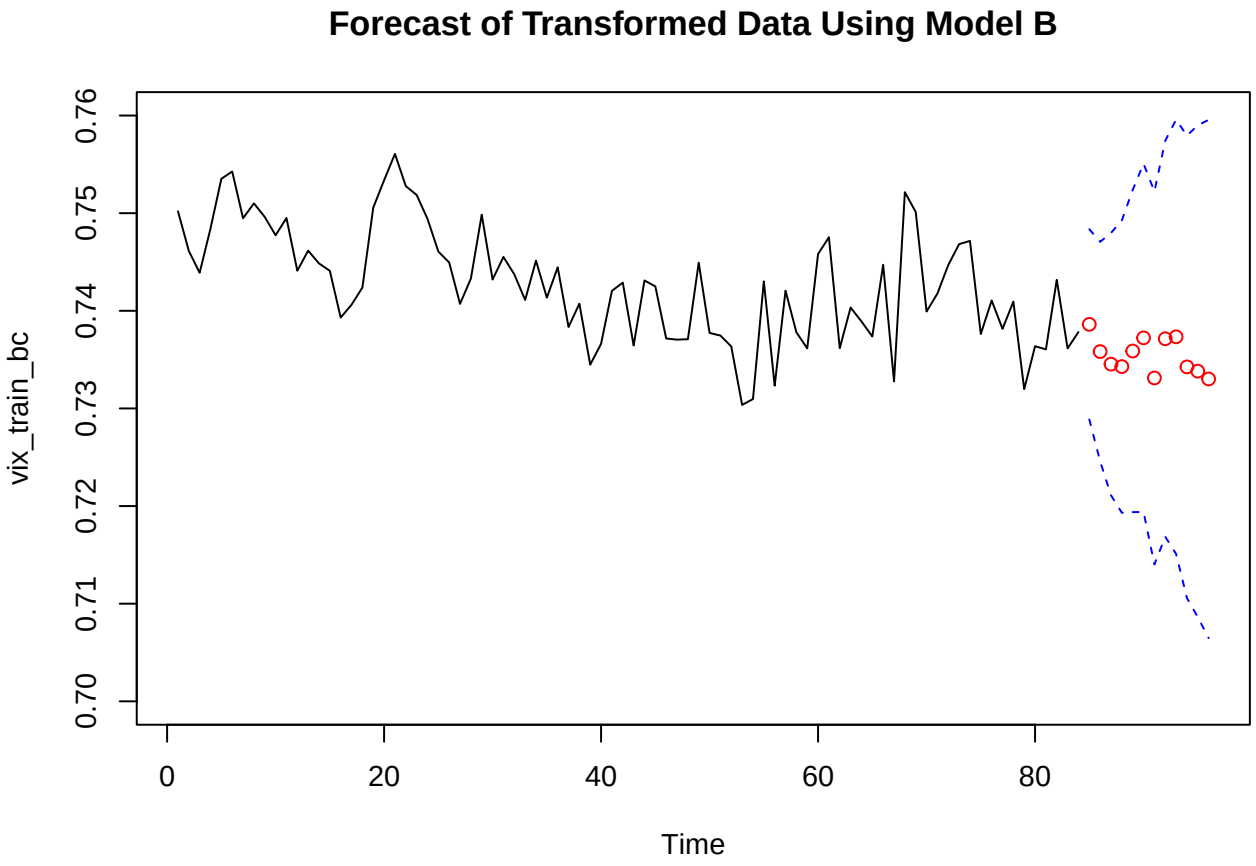
$\nabla_1\nabla_8 boxcox(U_t) = (1 - 0.405_{(0.107)}B)(1 + 0.136_{(0.186)}B^8)(1 - 0.939_{(0.482)}B^8)Z_t,$

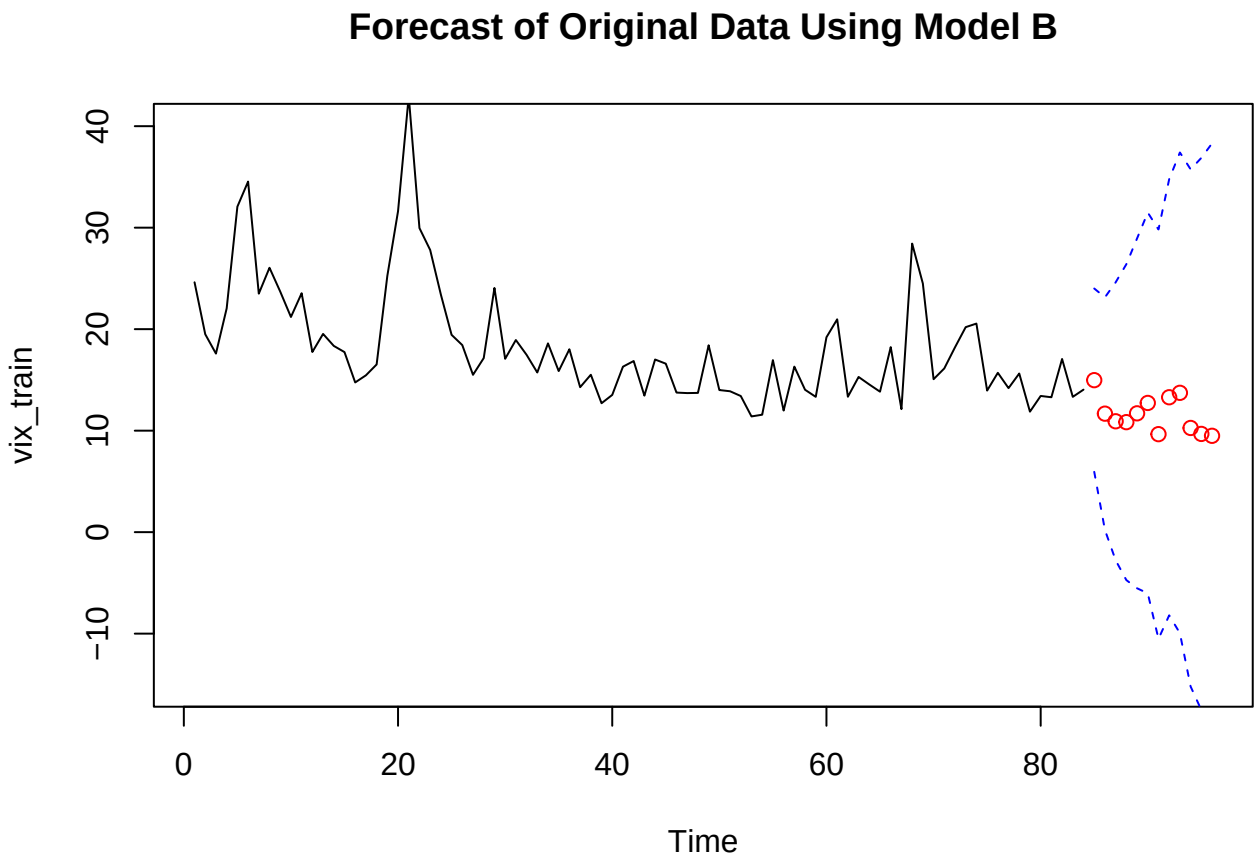
where B is the backshift operator.

Forecasting

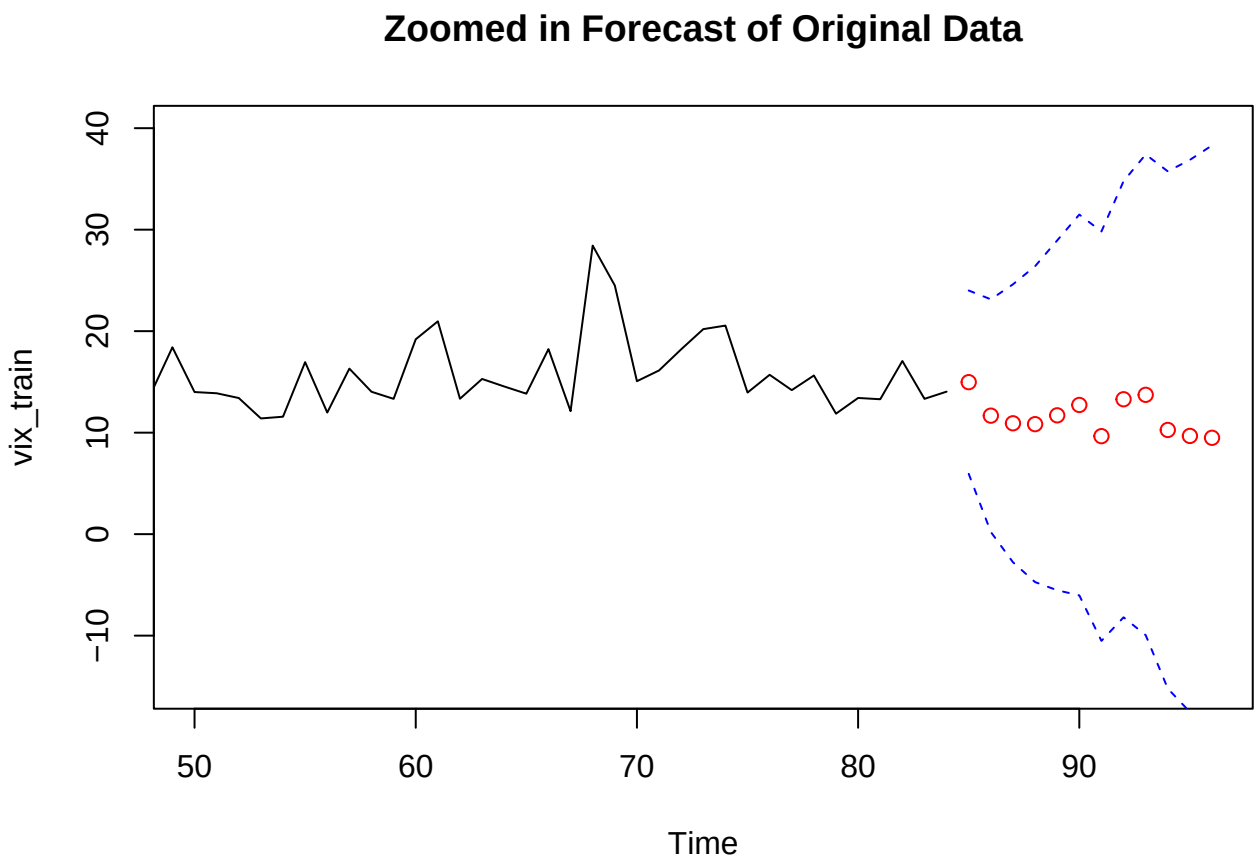
We will be forecasting the next 12 observations, which are months.

First, we'll start off forecasting values for the box-cox transformed data. Then, we will forecast values for the training data, and see how these values performed using our testing data.

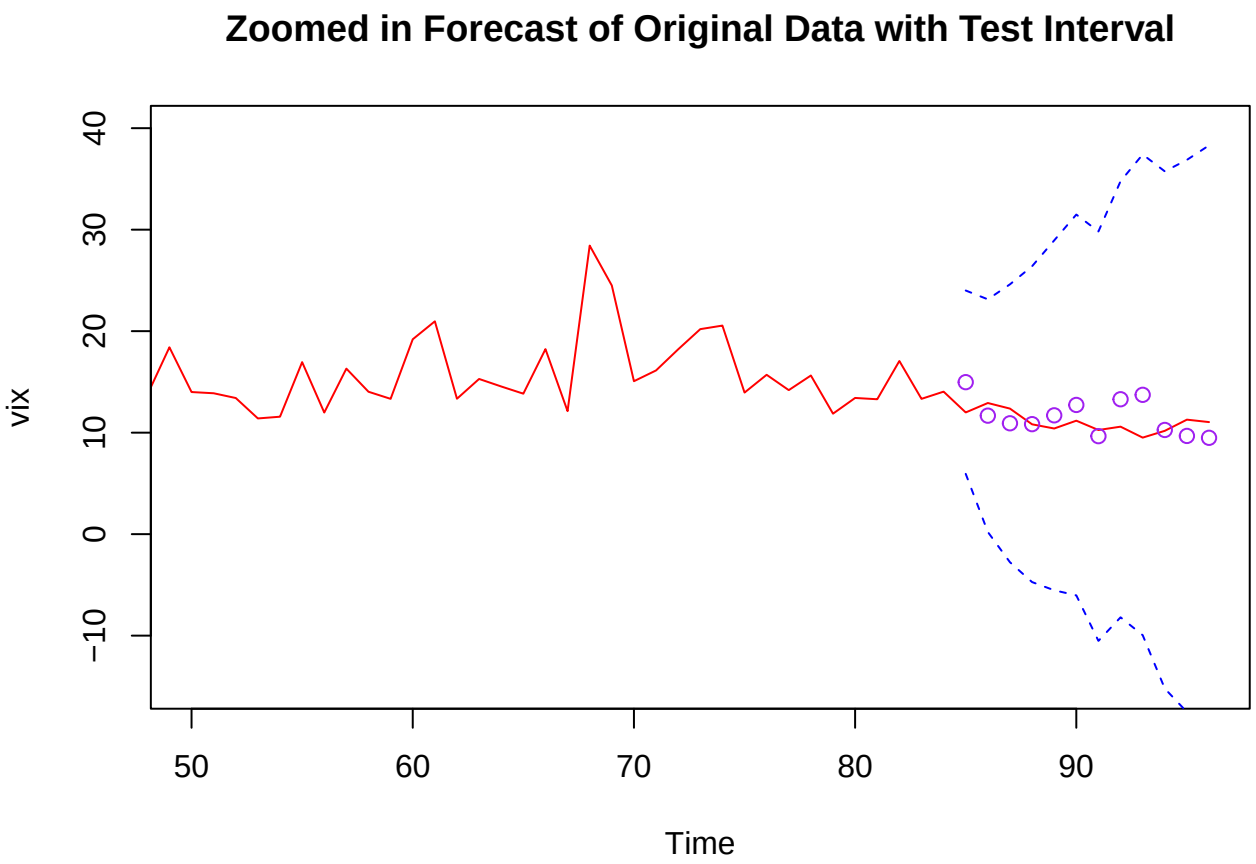




We will zoom in on the forecasts for better readability.



Let's now add back the testing set to see how our forecast did.



For my forecast, I forecasted 12 observations using the confidence intervals of the chosen model, which was Model B. We can see that the original line is within the confidence intervals of the forcasted values, and the points seem to do a moderately good job at following the testing set, though there are a few errors. The large gap in confidence intervals is to be expected, since we are forcasting a volatility index.

Conclusion

As expected, predicting the stock market is not an easy job. By transforming our data and applying practical time series methods to our data, we were able to perform proper forecasting for our data, and moderately determine the VIX values in the coming months. Given that we are working with a volatility index, I'm surprised that the predicting values are semi-accurate. Although I believe our model could be better, as our confidence intervals are high. Overall, the best model that helped us achieve our goal was Model B, denoted as the SARIMA$(1,1,0)$X$(1,1,1)_8$ model, which can also be written as:

$\nabla_1\nabla_8 boxcox(U_t) = (1 - 0.405_{(0.107)}B)(1 + 0.136_{(0.186)}B^8)(1 - 0.939_{(0.482)}B^8)Z_t.$

Despite our semi-accurate forecast, individuals will now have a more accurate idea on whether the VIX values will go down or not in the upcoming months, which will allow traders to make important investment decisions.

References

VIX: What you should know about the volatility index. https://www.fidelity.com.sg/beginners/what-is-volatility/volatility-index

Data:

Chicago Board Options Exchange. (2010, January 01). CBOE Volatility Index: VIX. Federal Reserve Economic Data. https://fred.stlouisfed.org/series/VIXCLS#0

Appendix

```r
# Load libraries
library(dplyr)
library(ggplot2)
library(ggfortify)
library(MASS)
library(forecast)
library(UnitCircle)
library(MuMIn)


# Read in file into R
vix <- read.csv("/Users/reynaldoperez/Downloads/VIXCLS.csv")
vix <- vix$VIXCLS

# Plot raw data
ts.plot(vix, main = paste("VIX Original Data"), gpars = list(xlab = "Time (Monthly)", ylab = "VIX Closi

# Split data into training and testing sets
vix_train <- vix[c(1:84)]  # Training set
vix_test <- vix[c(85:96)]  # Testing set

# We need to ensure with have a proper split
length(vix_train)
length(vix_test)

# Plot training data
plot.ts(vix_train, main = "VIX Training Data")
nt = length(vix_train)
fit <- lm(vix_train ~ as.numeric(1:nt)); abline(fit, col=2)
abline(h = mean(vix_train), col = 1)
legend("topright", legend = c("Fitted Line", "Mean"), pch = rep(15, 4), col = 2:1)

par(mfrow = c(1,2))  # Side by side graphs

# Histogram of training set
hist(vix_train, col = "sky blue", xlab = "", main = "Distribution of VIX Values")

# ACF of training set
acf(vix_train, lag.max = 50, main = "ACF of VIX Values")

# Box-Cox graph
bc_transform <- boxcox(vix_train ~ as.numeric(1:length(vix_train)))

# Value of lambda
lambda = bc_transform$x[which(bc_transform$y == max(bc_transform$y))]

# Compare box-cox and log transformations to original data
par(mfrow = c(2,2))  # Side by side for comparison
plot.ts(vix_train, main = "Original Data")

vix_train_bc = (1/lambda)*(vix_train^lambda-1)  # Box-cox transformation
vix_train_log <- log(vix_train)  # Log tranformation
```

```r
plot.ts(vix_train_bc, main = "TS Plot; boxcox(U_t)")  # Plot boxcox transformation for time series plot
plot.ts(vix_train_log, main = "TS Plot; ln(U_t)")  # Plot log transformation for time series plot
# Transformed histograms
par(mfrow = c(2,2))  # Side by side histograms for comparison
hist(vix_train, col = "sky blue", xlab = "", main = "Histogram of Original Data")  # Original data
hist(vix_train_bc, col="sky blue", xlab="", main="Histogram; boxcox(U_t)")  # Plot boxcox transformatio
hist(vix_train_log, col="sky blue", xlab="", main="Histogram; ln(U_t)")  # Plot log transformation for

# Decomposition of boxcox transformation
y <- ts(as.ts((vix_train_bc)), frequency = 12)
decomp <- decompose(y)
plot(decomp)

var(vix_train_bc)  # Variance

vix_train_8 <- diff(vix_train_bc, lag = 8)  # Difference at lag 8
var(vix_train_8)  # Variance of differenced data

vix_train_8_and_1 <- diff(vix_train_8, lag = 1)  # Difference at lag 8 then 1
var(vix_train_8_and_1)  # Variance of differenced data
par(mfrow = c(1,2))  # Side by side
par(cex.main = 0.75)  # Title size
# Plot no-differenced data
plot.ts(vix_train_bc, main = "No Difference", sub = "Variance = 0.00203712")

# Add mean and trend line
nt=length(vix_train_bc)
fit <- lm(vix_train_bc ~ as.numeric(1:nt))
abline(fit, col="red")
abline(h=mean(vix_train_bc), col="blue")
legend("topright", legend = c("Fitted Line", "Mean"), pch = rep(15, 4), col = c("red","blue"), cex = 0.

# Plot differenced data at lag 8
plot.ts(vix_train_8, main = "Differenced at Lag 8", sub = "Variance = 0.000630349")

# Add mean and trend line
nt1 = length(vix_train_8)
fit1 <- lm(vix_train_8 ~ as.numeric(1:nt1))
abline(fit1, col = "red")
abline(h=mean(vix_train_8), col = "blue")
legend("topright", legend = c("Fitted Line", "Mean"), pch = rep(15, 4), col = c("red","blue"), cex = 0.

# Plot second difference
plot.ts(vix_train_8_and_1, main = "Differenced at Lag 8 then 1", sub = "Variance = 0.000580849")
# Add mean and trend line
nt2 <- length(vix_train_8_and_1)
fit2 <- lm(vix_train_8_and_1 ~ as.numeric(1:nt2))
abline(fit2, col = "red")
abline(h=mean(vix_train_8_and_1, col = "blue"))
legend("topright", legend = c("Fitted Line", "Mean"), pch = rep(15, 4), col = c("red","blue"), cex = 0.

par(cex.main = 0.45)  # Fit title
# Plot ACFs
```

```r
par(mfrow = c(1,2))  # Side by side
acf(vix_train_bc, lag.max = 50, main = "ACF of boxcox(U_t)")  # Original boxcox
acf(vix_train_8, lag.max = 50, main = "ACF of boxcox(U_t) Differenced at Lag 8")  # ACF of differenced
par(cex.main = 0.75)  # Fit title
acf(vix_train_8_and_1, lag.max = 50, main = "ACF of boxcox(U_t) Differenced at Lag 8 then 1")  # ACF of


par(mfrow = c(1,2))  # Side by side
par(cex.main = 0.40)  # title
# Check for normality
hist(vix_train, col="sky blue", xlab="", main="Histogram; boxcox(U_t)")  # Plot non-differenced data
hist(vix_train_8, col="sky blue", xlab="", main="Histogram; boxcox(U_t) Differenced at Lag 8")  # Plot
hist(vix_train_8_and_1, col="sky blue", xlab="", main="Histogram; boxcox(U_t) Differenced at Lag 8 then

par(mfrow = c(1,2))  # Side by side
par(cex.main = 0.4)  # Title
acf(vix_train_8_and_1, lag.max = 50, main = "ACF of boxcox(U_t) Differenced at Lag 6 then 1")  # ACF of
pacf(vix_train_8_and_1, lag.max = 50, main = "PACF of boxcox(U_t) Differenced at Lag 6 then 1")  # PACF

## Trying different models

# Highest AICc
arima(vix_train_bc, order = c(1, 1, 0), seasonal = list(order = c(1, 1, 0), period = 8), method = "ML")
AICc(arima(vix_train_bc, order=c(1,1,0), seasonal = list(order = c(1,1,0), period = 8), method="ML"))
print("Highest AICc", quote = FALSE)

# Second lowest AICc
arima(vix_train_bc, order = c(1, 1, 0), seasonal = list(order = c(1, 1, 1), period = 8), method = "ML")
AICc(arima(vix_train_bc, order=c(1,1,0), seasonal = list(order = c(1,1,1), period = 8), method="ML"))
print("Second lowest AICc", quote = FALSE)

# Third lowest AICc
arima(vix_train_bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 0), period = 8), method = "ML")
AICc(arima(vix_train_bc, order=c(1,1,1), seasonal = list(order = c(1,1,0), period = 8), method="ML"))
print("Third lowest AICc", quote = FALSE)

# Lowest AICc
arima(vix_train_bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 1), period = 8), method = "ML")
AICc(arima(vix_train_bc, order=c(1,1,1), seasonal = list(order = c(1,1,1), period = 8), method="ML"))
print("Lowest AICc", quote = FALSE)

# Helper function to plot roots
plot.roots <- function(ar.roots=NULL, ma.roots=NULL, size=2.5, angles=FALSE, special=NULL, sqecial=NULL
{xylims <- c(-size,size)
omegas <- seq(0,2*pi,pi/500)
temp <- exp(complex(real=rep(0,length(omegas)),imag=omegas))
plot(Re(temp),Im(temp),typ="l",xlab="x",ylab="y",xlim=xylims,ylim=xylims,main=main)
abline(v=0,lty="dotted")
abline(h=0,lty="dotted")
if(!is.null(ar.roots))
{
  points(Re(1/ar.roots),Im(1/ar.roots),col=first.col,pch=my.pch)
  points(Re(ar.roots),Im(ar.roots),col=second.col,pch=my.pch)
```

```
}
if(!is.null(ma.roots))
{
  points(Re(1/ma.roots),Im(1/ma.roots),pch="*",cex=1.5,col=first.col)
  points(Re(ma.roots),Im(ma.roots),pch="*",cex=1.5,col=second.col)
}
if(angles)
{
  if(!is.null(ar.roots))
  {
    abline(a=0,b=Im(ar.roots[1])/Re(ar.roots[1]),lty="dotted")
    abline(a=0,b=Im(ar.roots[2])/Re(ar.roots[2]),lty="dotted")
  }
  if(!is.null(ma.roots))
  {
    sapply(1:length(ma.roots), function(j) abline(a=0,b=Im(ma.roots[j])/Re(ma.roots[j]),lty="dotted"))
  }
}
if(!is.null(special))
{
  lines(Re(special),Im(special),lwd=2)
}
if(!is.null(sqecial))
{
  lines(Re(sqecial),Im(sqecial),lwd=2)
}
}

# Roots of Model A
plot.roots(NULL,polyroot(c(1,0.370,-0.884)), main="Model A")
polyroot(c(1, 0.370, -0.884))
print("One root is inside the unit circle; model is not invertible!", quote = FALSE)
# Roots of Model B
plot.roots(NULL,polyroot(c(1,  -0.425)), main="Model B")
polyroot(c(1, -0.425))
print("Root is outside; model is invertible", quote = FALSE)
# Roots of Model C
plot.roots(NULL,polyroot(c(1,  0.489,  -1.000)), main="Model C")
polyroot(c(1, 0.489,  -1.000))
print("One root is inside the unit circle, model is not invertible!", quote = FALSE)
# Roots of Model D
plot.roots(NULL,polyroot(c(1,  -0.405)), main="Model D")
polyroot(c(1, -0.405))
print("Root is outside; model is invertible", quote = FALSE)

# Diagnostic checking for model B
par(mfrow = c(2,2))  # Side by side
par(cex = 0.4)
fit <- arima(vix_train_bc, order=c(1,1,0), seasonal = list(order = c(1,1,1), period = 8), method="ML")
res <- residuals(fit)
hist(res,density=30,breaks=30, col="blue", xlab="", prob=TRUE)
m <- mean(res)
std <- sqrt(var(res))
```

```r
curve(dnorm(x,m,std), add=TRUE)
plot.ts(res)
fitt <- lm(res ~ as.numeric(1:length(res))); abline(fitt, col="red")
abline(h=mean(res), col="blue")
qqnorm(res,main= "Normal Q-Q Plot for Model A")
qqline(res,col="blue")
mean1 <- mean(res)

par(mfrow = c(1,2))
par(cex = 0.4)
acf(res^2, lag.max=50)
pacf(res^2, lag.max = 50)
shapiro.test(res)
Box.test(res, lag = 9, type = c("Box-Pierce"), fitdf = 3)
Box.test(res, lag = 9, type = c("Ljung-Box"), fitdf = 3)
Box.test(res^2, lag = 9, type = c("Ljung-Box"), fitdf = 0)
ar(res, aic = TRUE, order.max = NULL, method = c("yule-walker")) # AR(0)?

# Diagnostic checking for model D
par(mfrow = c(2,2))  # Side by side
par(cex = 0.8)
fit <- arima(vix_train_bc, order=c(1,1,0), seasonal = list(order = c(1,1,0), period = 8), method="ML")
res <- residuals(fit)
hist(res,density=30,breaks=30, col="blue", xlab="", prob=TRUE)
m <- mean(res)
std <- sqrt(var(res))
curve(dnorm(x,m,std), add=TRUE )
plot.ts(res)
fitt <- lm(res ~ as.numeric(1:length(res))); abline(fitt, col="red")
abline(h=mean(res), col="blue")
qqnorm(res,main= "Normal Q-Q Plot for Model A")
qqline(res,col="blue")

acf(res^2, lag.max=50)
pacf(res^2, lag.max = 50)

shapiro.test(res)
Box.test(res, lag = 9, type = c("Box-Pierce"), fitdf = 2)
Box.test(res, lag = 9, type = c("Ljung-Box"), fitdf = 2)
Box.test(res^2, lag = 9, type = c("Ljung-Box"), fitdf = 0)
ar(res, aic = TRUE, order.max = NULL, method = c("yule-walker"))  # AR(0)?

par(cex = 0.9)  # Title size
# Forecast transformed data
fit.A <- arima(vix_train_bc, order=c(1,1,0), seasonal = list(order = c(1,1,1), period = 8), method="ML")
pred.tr <- predict(fit.A, n.ahead = 12)
U.tr= pred.tr$pred + 2*pred.tr$se # upper bound of prediction interval
L.tr= pred.tr$pred - 2*pred.tr$se # lower bound
#ts.plot(vix_train_bc, xlim=c(1,length(vix_train_bc)+12), ylim = c(min(vix_train_bc),max(U.tr)))
ts.plot(vix_train_bc, xlim=c(1, length(vix_train_bc) + 12), ylim = c(0.700, 0.760), main = "Forecast of
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(vix_train_bc)+1):(length(vix_train_bc)+12), pred.tr$pred, col="red")
```

```
forecast(fit.A) # prints forecasts with prediction bounds in a table
#par(cex = 0.9)
# Forecast original data
pred.orig <- exp(pred.tr$pred)
fit.A <- arima(vix_train, order=c(1,1,0), seasonal = list(order = c(1,1,1), period = 8), method="ML")
forecast(fit.A) # prints forecasts with prediction bounds in a table
pred.tr <- predict(fit.A, n.ahead = 12)
U.tr= pred.tr$pred + 2*pred.tr$se # upper bound of prediction interval
L.tr= pred.tr$pred - 2*pred.tr$se # lower bound
ts.plot(vix_train, xlim=c(1, length(vix_train) + 12), ylim = c(-15, 40), main = "Forecast of Original Da
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(vix_train)+1):(length(vix_train)+12), pred.tr$pred, col="red")
# Zoom in on the graph
fit.A <- arima(vix_train, order=c(1,1,0), seasonal = list(order = c(1,1,1), period = 8), method="ML")
pred.tr <- predict(fit.A, n.ahead = 12)
U.tr= pred.tr$pred + 2*pred.tr$se # upper bound of prediction interval
L.tr= pred.tr$pred - 2*pred.tr$se # lower bound
ts.plot(vix_train, xlim=c(50, length(vix_train) + 12), ylim = c(-15, 40), main = "Zoomed in Forecast of
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(vix_train)+1):(length(vix_train)+12), pred.tr$pred, col="red")
# Zoomed in forecasts on true values (vix)
ts.plot(vix, xlim=c(50, length(vix_train)+ 12), ylim = c(-15, 40), col = "red", main = "Zoomed in Foreca
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(vix_train)+1):(length(vix_train)+12), pred.tr$pred, col=" purple")
points((length(vix_train)+1):(length(vix_train)+12), pred.tr$orig, col = "black")
```