Time Series Forecasting for Mass Shootings in the U.S. (1966-2021)

PSTAT 174 Fall 2022 Final Project - Forecasting in R

Ezra Reyes Aguimatang

University of California, Santa Barbara

Contents

1 Abstract

According to the 2^{nd} Amendment of the United States Constitution, it states *"A well regulated Militia, being necessary to the security of a free State, the right of the people to keep and bear Arms, shall not be infringed"*. In light of the increase of mass shootings and more frequent gun violence in recent years, the discussion concerning the 2^{nd} Amendment has gravitated towards becoming a large looming controversy over the American people, and in turn these discussion are seemingly intensifying a political polarization across the country.

In this study, we will be utilizing a dataset from the Kaggle Database: Mass Shootings in the United States of America, which has recorded various attributes from a shocking 398 mass shootings that occurred in the United States between the years $1966 - 2021$. With this dataset, the aim of this study is to use *Time Series Analysis* and *Time Series Forecasting* in `R` as a means of predicting the future of the United States and its people, in terms of the presence of mass shootings and gun violence. Focusing on the `Date` and `Total_Victims` features of our dataset, this study will be forecasting the total number of victims (both fatalities and injuries) in the United States for future mass shootings if the state of the nation does not change.

After preliminary Exploratory Time Series Analysis, it was concluded that a Box-Cox Transformation of the series was best suited to creating a stationary series. After further examination of the transformed series, a first-order differencing at lag 1 was proven to make the series nearly entirely stationary. Next, after analyzing the ACF and PACF plots of the Box-Cox Transformed series differenced at lag 1, some preliminary assumed models were chosen, with a primary assumption of an $ARIMA(1,1,1)$ model.

Moving onto more in depth model comparison, 6 models were considered and compared using the Second-Order Akaike Information Criterion (AICc) and the Bayesian Information Criterion (BIC) tests. As a result, the suitable models were narrowed down to the $ARIMA(0,0,1)$, $ARIMA(1,0,1)$, and $ARIMA(1,0,0)$ models, no longer considering $ARIMA(1,1,1)$. Moving these 3 models onto the diagnostic testing stage of the study, after examining visualizations of these models' residuals using Histograms, Normal QQ Plots, Time Series Plots, ACF Plots, PACF Plots, ACF Plots of Squared Residuals; and by evaluating some statistical tests such as the Shapiro-Wilk Test, Box-Pierce Test, Box-Ljung Test, and Box-Ljung Test of Squared Residuals: all models proved to be suitable. Next, the invertibility of the models were evaluated using plots of their unit roots, which also did not prove to be a useful method of narrowing down a final model (as all models were invertible). Hence, our final model was chosen based off of previously computed AICc and BICs, leaving our final model chosen to be an $ARIMA(0,0,1)$ or in back-shift operator notation: $\Delta_1 BoxCox(U_t) = (1 + (-0.8818)_{(0.1859)}B)Z_t$. Although the best possible fitting model found, an $ARIMA(0,0,1)$ proved not to be the most accurate forecast for predicting the years $2016 - 2021$, an attempt to forecast 9 years into the future of the data $2022 - 2030$ was computed.

During this great time of sadness, fear, and chaos in the United States, its important to recognize the role that Data Science and statistical methods of research as a whole can play in helping the future of millions of lives. Although this study's forecasting model underestimated the number of mass shooting victims that would occur, perhaps this is more telling? In the hands of the wrong people, this type of research has the potential to spread possibly false statistics that discourage action. Throughout this time series forecasting analysis, it has become evident that these looming issues of gun violence and mass shootings have an exponential increase that could not be foreseen by statistical processes, meaning we have an extremely urgent matter that calls for a way to decrease its trend in order to save the lives of the millions it effects.

2 Introduction

The political divide among the people of the United States has always been present, but in more recent years, this gap between those of different political affiliations appears to widen more and more. With the growing use of social media to spread videos and ideas in a more efficient manner, new issues and conversations are brought to light, being more easily accessible than ever before. One of these topics of conversation is the issue of increased *Mass Shootings* and *Gun Violence* in the United States. According to various sources, the United States has consistently been the country awarded with the highest number of mass shootings. Although this idea is objective, factual, numerical, and has been widely known for years, the numbers of mass shootings per year seems to grow exponentially. A controversial topic that seems to be choosing between protecting the lives of primarily children in schools or protecting ones' selves with arms, is an issue in this country that has yet to be solved.

As stated prior, the Mass Shootings in the United States of America (1966-2021) dataset from Kaggle will be utilized in this study, which was originally compiled by Zeeshan-Ul-Hassan Usmani. This dataset was allegedly created using multiple sources including "Wikipedia, Mother Jones, Stanford, USA Today and other web sources" (as stated in his acknowledgements), and was given a Kaggle usability score of 7.94: a score out of 10 that Kaggle assigns data sets based on their completeness, credibility, and compatibility in order to inform users the validity of the data that is presented.

Rather than acting as a model of inference showing all the different factors that could explain the increase in the numbers of mass shootings, gun violence, and victims, this time series model will serve as a tool to forecast this dataset and show how these numbers will potentially look like in our near future if this current trend is to continue. Although this study isn't meant to take into account all or even many factors as to why this increase in mass shootings is occurring, creating a window into what this forecasted data may appear like in the future may hopefully incite some further discussion and possible urgency in the hands of all people who have the power to change the current and future state of this nation.

2.1 R Packages

For this study, data cleaning and time series analysis will be completed using the `R` programming language. We will be utilizing the following packages in `R` throughout the duration of this study…

```
library(knitr)
library(ggplot2)
library(ggfortify)
library(tinytex)
library(dplyr)
library(tidyverse)
library(MASS)
library(tseries)
library(devtools)
library(forecast)
library(UnitCircle)
library(MuMIn)
```

3 Dataset Overview and Cleaning

The Mass Shootings in the United States of America database has multiple files with different feature variables with some having more up to data observations than others. In this study the dataset version that will be used is the `Mass Shootings Dataset.csv` and `Mass shooting data.csv` files.

Before working with this data, we will first read the file in as an `R` dataframe to view its features and clean it as necessary.

```r
Mass_Shootings_Dataset_csv <- read_csv("archive/Mass Shootings Dataset.csv")
Mass_Shootings_Dataset_csv2 <- read_csv("archive/Mass shooting data.csv")
```

When taking a look at our dataset, we will see that we have 398 rows of observations and 13 column feature variables available. Since this dataset was not created specifically for the context and purposes of this project, a little cleaning is necessary before proceeding.

This data was not explicitly meant for time series analysis, so we will begin by changing the data types of the `Date` variable and also rename a few column feature variable names for better readability.

```r
Mass_Shootings_Dataset_csv$Date <- as.POSIXct(Mass_Shootings_Dataset_csv$Date,
                                              format="%m/%d/%Y")
colnames(Mass_Shootings_Dataset_csv)[1] <- "Event_ID"
colnames(Mass_Shootings_Dataset_csv)[8] <- "Total_Victims"
colnames(Mass_Shootings_Dataset_csv)[9] <- "Mental_Health_Issues"
```

	Feature Column Data Types
Event_ID	num
Title	chr
Location	chr
Date	POSIXct
Summary	chr
Fatalities	num
Injured	num
Total_Victims	num
Mental_Health_Issues	chr
Race	chr
Gender	chr
Latitude	num
Longitude	num

After looking at the structure of our data frame using the `str()` function, we see the data types of each of our column feature variables in the table above. For the intentions of this study, the variables that are most useful to us will be our `Date` and `Total_Victims` variables.

Now that we've done a little data cleaning and have gotten some insight into the feature columns of our data, we'll now take a look at the first 5 observations of our data for the 13 feature columns we have available to us.

```
head(Mass_Shootings_Dataset_csv[0:4], 5)
```

Event_ID	Title	Location	Date
1	Las Vegas Strip mass shooting	Las Vegas, NV	2017-10-01
2	San Francisco UPS shooting	San Francisco, CA	2017-06-14
3	Pennsylvania supermarket shooting	Tunkhannock, PA	2017-06-07
4	Florida awning manufacturer shooting	Orlando, Florida	2017-06-05
5	Rural Ohio nursing home shooting	Kirkersville, Ohio	2017-05-12

```
head(Mass_Shootings_Dataset_csv[5], 5)
```

Summary
NA
Jimmy Lam, 38, fatally shot three coworkers and wounded two others inside a UPS facility in San Francisco. Lam killed himself as law enforcement officers responded to the scene.
Randy Stair, a 24-year-old worker at Weis grocery fatally shot three of his fellow employees. He reportedly fired 59 rounds with a pair of shotguns before turning the gun on himself as another co-worker fled the scene for help and law enforcement responded.
John Robert Neumann, Jr., 45, a former employee of manufacturer Fiamma Inc. fatally shot five workers at the company, and then killed himself on the scene. He'd been fired from the company in April. The attack took place a week before the one-year anniversary of the Orlando nightclub massacre.
Thomas Hartless, 43, shot and killed a former girlfriend and another employee of a nursing home, and then fatally shot the Kirkersville police chief responding to the scene. Hartless' former girlfriend had recently obtained a court protection order against Hartless. Investigators later found more than 60 firearms in the home of Hartless, who was also found dead at the scene of the attack.

```
head(Mass_Shootings_Dataset_csv[6:13], 5)
```

Fatalities	Injured	Total_Victims	Mental_Health_Issues	Race	Gender	Latitude	Longitude
58	515	573	Unclear	NA	NA	NA	NA
3	2	5	Yes	Asian	M	NA	NA
3	0	3	Unclear	White	M	NA	NA
5	0	5	Unclear	NA	M	NA	NA
3	0	3	Yes	White	M	NA	NA

3.1 Making Data Time Series Appropriate

Since this data was not originally created for the purposes of Time Series Analysis, some necessary techniques of data cleaning and feature engineering are implemented before we can create our Time Series object.

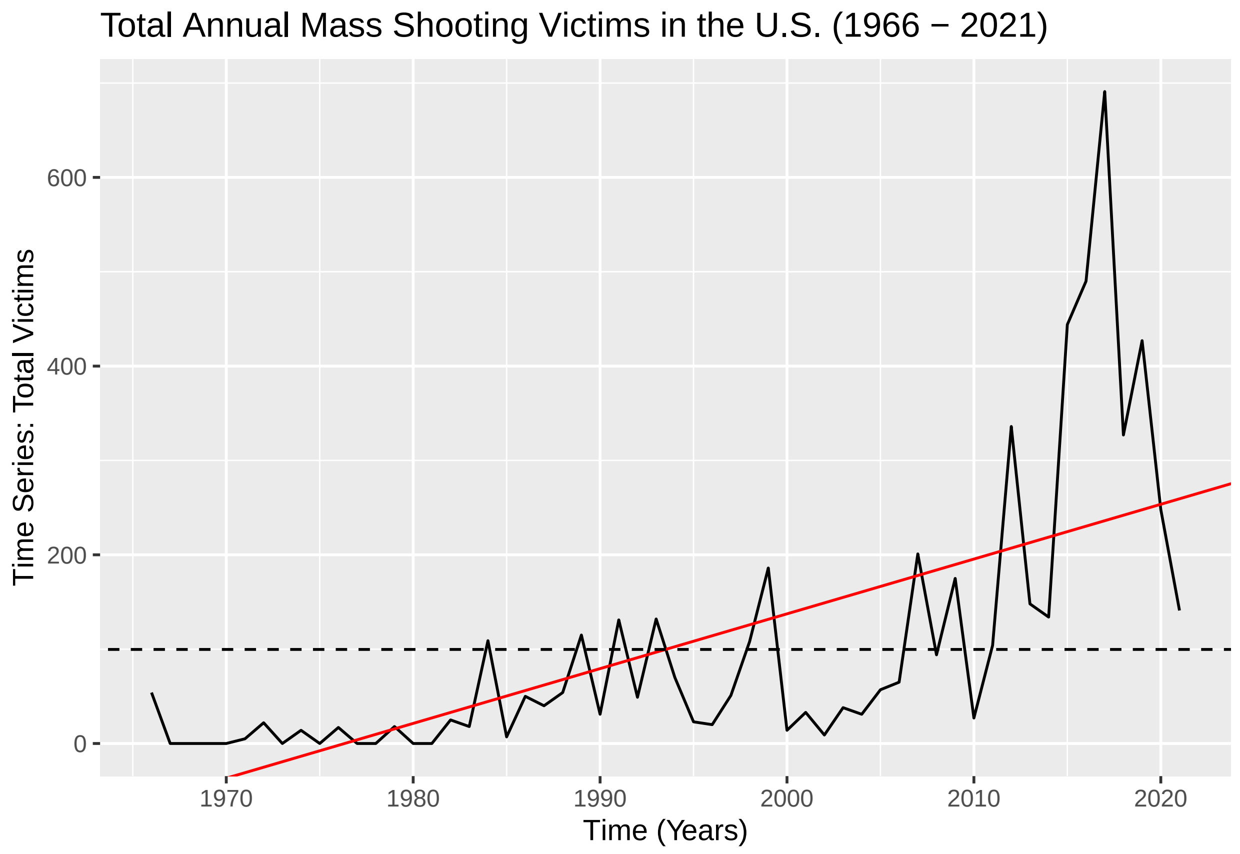
```r
# Subsetting only "Total_Victims" and "Date columns"
mass_shootings_df <- Mass_Shootings_Dataset_csv1[, c("Total_Victims", "Date")]
##head(mass_shootings_df)

# Adding a Year identifying Column based on the POSIXct "Date" Column
mass_shootings_df$Year <- as.numeric(format(mass_shootings_df$Date, format="%Y"))
##head(mass_shootings_df)
##unique(mass_shootings_df$Year)

# Creating a Data Frame of Values for Missing Years
Total_Victims <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Date <- c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA)
Year <- c(1981, 1980, 1978, 1977, 1975, 1973, 1970, 1969, 1968, 1967)
missing_obs_df <- data.frame(Total_Victims, Date, Year)

# Binding the Missing Years Data Frame to the Original Data Frame
mass_shootings_newdf <- rbind(mass_shootings_df, missing_obs_df)
mass_shootings_newdf <- mass_shootings_newdf[order(mass_shootings_newdf$Year,
                                                   decreasing=TRUE),]

# Creating a New Data Frame Summing and Grouping Row Observations by Year
Mass_Shootings_df <- mass_shootings_newdf
Mass_Shootings_df <- aggregate(Total_Victims ~ Year, Mass_Shootings_df, sum)
# Mass_Shootings_df
```

As seen above, we've subset the `Total_Victims` and `Date` columns being the two important aspects of our original data set that we want to analyze. Next, knowing that the `ts()` can only create a time series object with data that has no gaps in time, or `Date` in our case, a few steps had to be taken. Firstly, our `Date` column was originally in the format `%m/%d/%Y`, which denoted the date a mass shooting event had occurred. However, between the years 1966 – 2021 there was thankfully not an observation available for each day in this time period. Therefore, we had started with large gaps in our data. On top of this issue, the data not only had gaps between the day of events, but also had various missing months and years. As can be seen above we had a total of 10 missing year (1981, 1980, 1978, 1977, 1975, 1973, 1970, 1969, 1968, 1967). So in order to solve all these issues at once, I decided it was best to add a new column (`Year`) to our data to ID each observation by the year of their `Date`. By doing so, it was possible to aggregate the sums of `Total_Victims` for observations that have the same value for our new `Year` column, and lastly add observations for the missing `Years` in our data and setting all their values of `Total_Victims` to `0` (indicating that there were no victims of mass shootings that year).

4 Data Split and Time Series Plot

Now that we have a good idea of what exactly our data represents and finished the process of cleaning, we now want to view a plot of our time series object to now get a visual representation of our data.

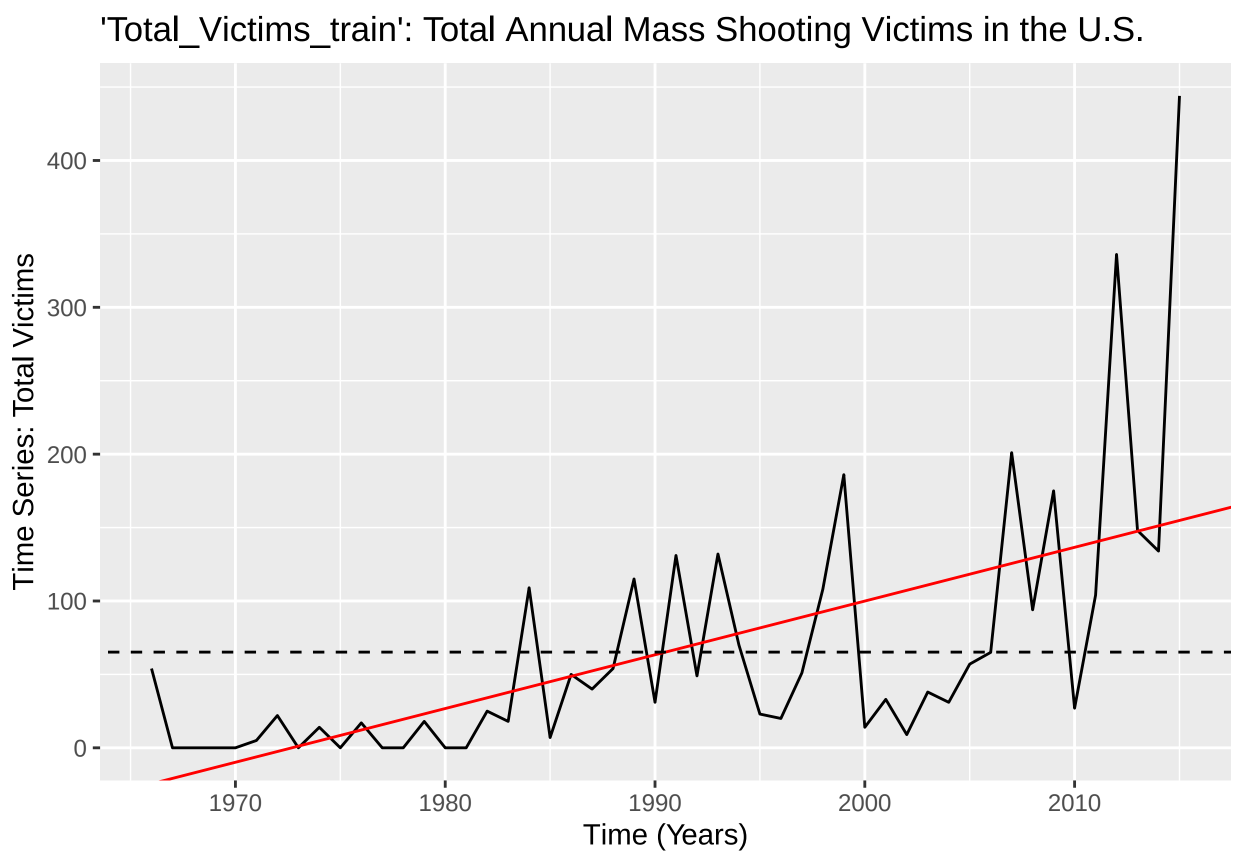
```r
Total_Victims_ts <- ts(data=Mass_Shootings_df[2],
                       start=1966, end=2021, frequency=1)
```



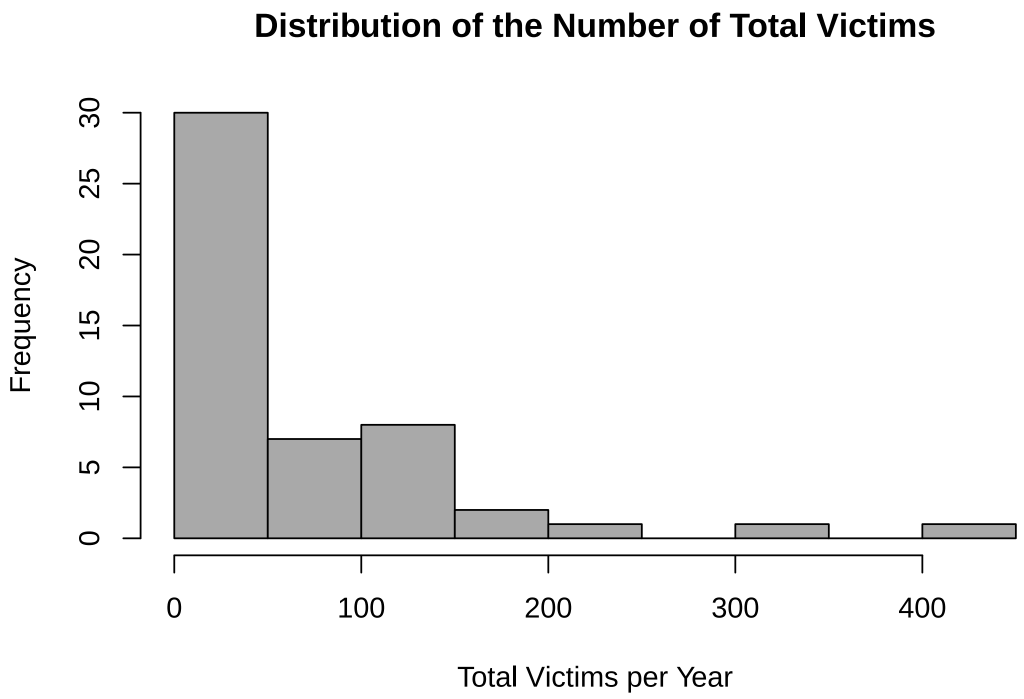
Since our time series `Total_Victims_ts` is an annual representation of total victims of mass shootings for each year, the `frequency` argument of the `ts()` function will be set to 1. From the plot above we do see a fairly strong increasing/upward trend, which allows us to conclude there has been a very clear gradual increase in the number of total victims of mass shootings (both fatalities and injuries) in the United States from 1966 – 2021. Knowing that this data pertains to the number of victims for mass shootings per year, and as we can see from the behavior of the plot, we start to conclude that this time series is most likely not seasonal but rather only has a fairly noticeable trend.

Now that we've created our Time Series object, before starting the process of analyzing our data for model selection, let us first create an 80/20 split our data into a training and testing set which we will call `Total_Victims_train` and `Total_Victims_test`.

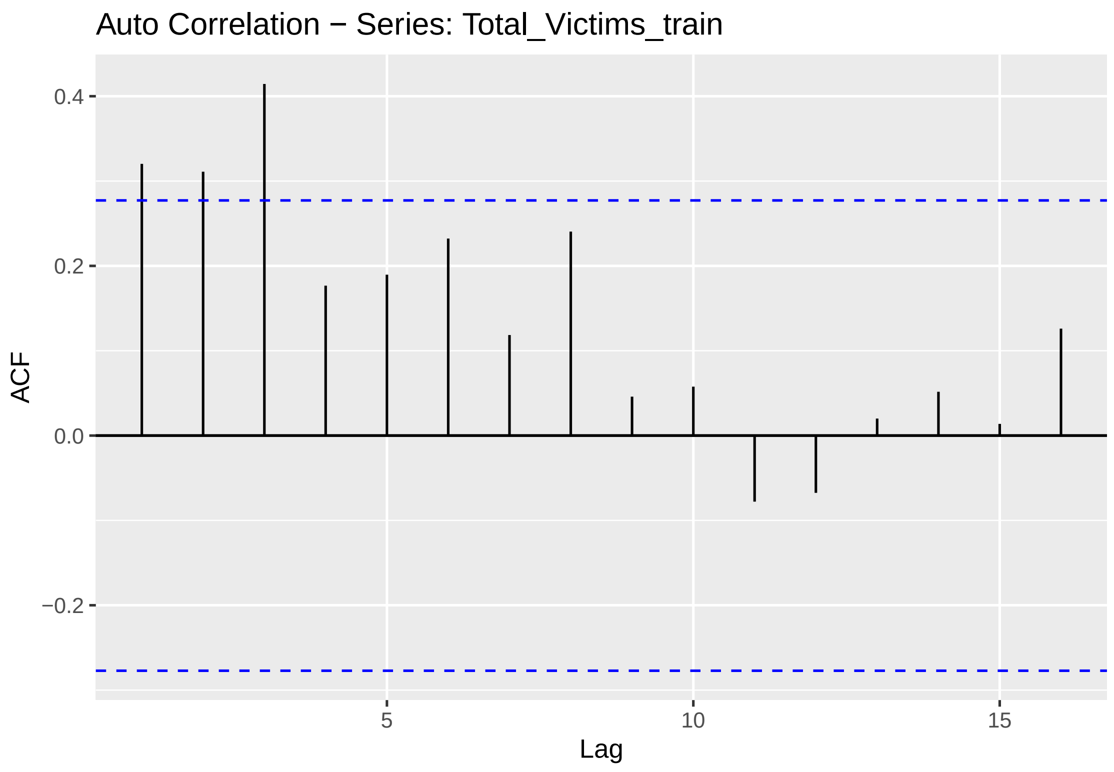
```
Total_Victims_train <- Mass_Shootings_df[1:50, ]
Total_Victims_test <- Mass_Shootings_df[50:56, ]
```



Now, before starting the process of analyzing our data for model selection, let us first take a closer look at our time series using a histogram and ACF plot.



Distribution of the Number of Total Victims
Frequency
0
5
10
15
20
25
30
0
100
200
300
400
Total Victims per Year

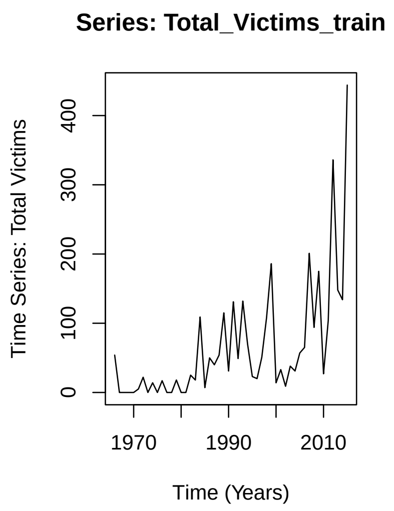


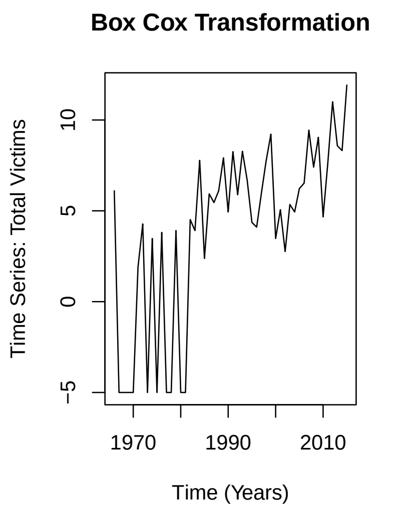
Auto Correlation − Series: Total_Victims_train
ACF
0.4
0.2
0.0
−0.2
5
10
15
Lag

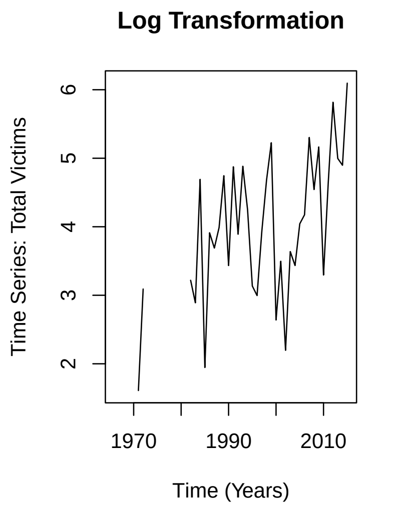
4.1 Transformations and Differencing

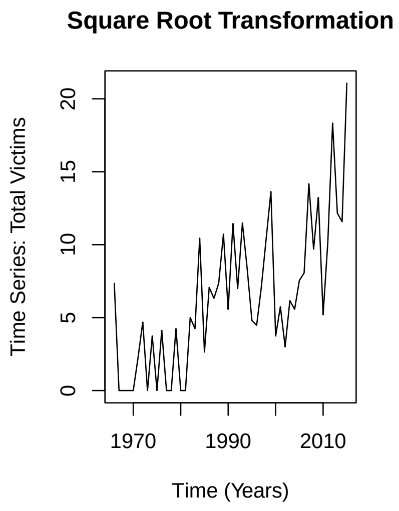
After looking at the initial time series plot of our data, the next step one considers in the model selection process are **Transformations** and **Differencings**. The purposes behind these techniques is to essentially manipulate our original time series model so that is it more **Stationary** and removes any trace of **Trend** or **Seasonality**. Looking at the 2 plots shown in the previous page ("Distribution of the Number of Total Victims" and "Auto Correlation - Series: Total_Victims_train"), we see the distribution of the amount of total victims per year has a very strong right-skew where many years have in between 0 and 50 victims of mass shootings. Now as we can see from the second plot, the ACF values confirm our suspicions that we do have a strong trend.

For these reasons, we want to transform our data in the hopes that we can stabilize variance and take a step closer to a stationary set of data. For this series we will attempt a Box-Cox Transformation, Log Transformation, and a Square-Root Transformation.

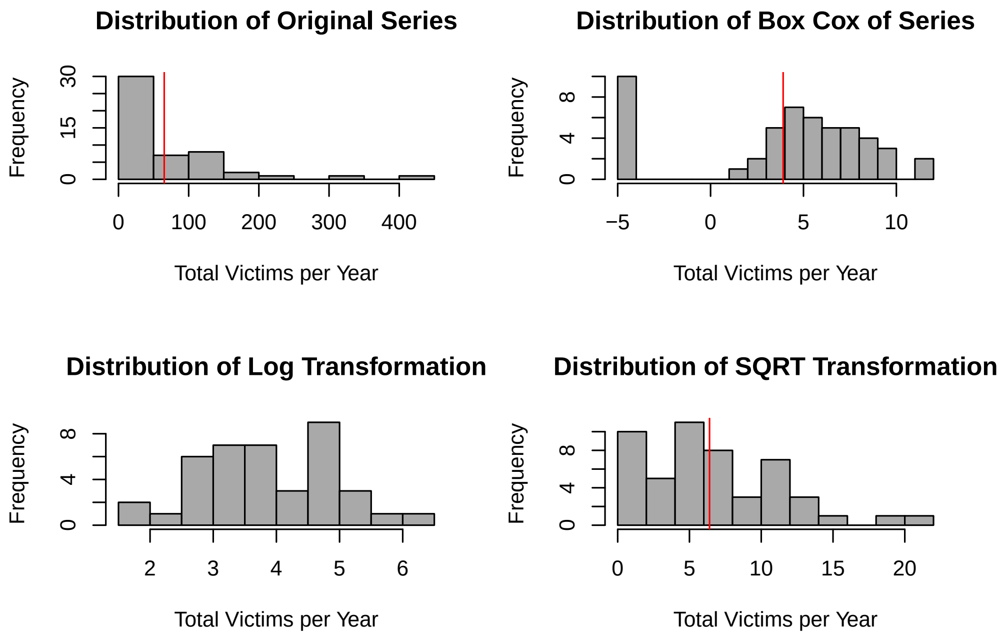


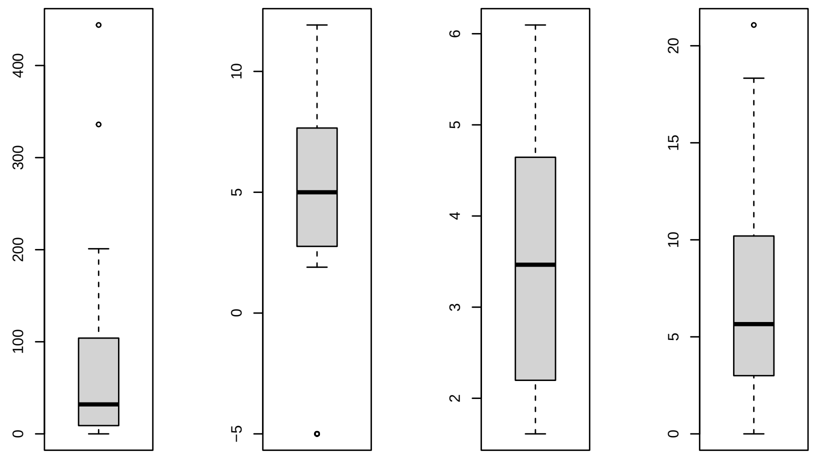






After examining the time series plots above of the possible transformations, we can somewhat confidently say that visually, these transformations seem to have greatly reduced variance especially for the years after 2010 when the amount of victims skyrocketed to over 600 by 2017. Although all of these plots seem to look like great transformations, we must further investigate which is the best transformation for our series.





Looking at visual the aspects of our plots above alone, we can conclude that our transformations that bring our original series, `Total_Victims_train`, looking the closest to a normal distribution, are the Box-Cox and Log Transformations. However, although our box plots for the Box-Cox

Transformation and Log Transformation (middle two box plots) seem to indicate that we have successfully normalized our data, we see from our histograms that the mean of our Log Transformation is not present in our plot and `R` informs us that we have a `-inf` mean value. In `R`, although a Log Transformation is perfect for discrete time series, it cannot handle values of zero hence our `-inf` mean value as well as an inability to calculate our variance or output an ACF or PACF plot. For this reason, we are not able utilize a Log Transformation for this study.

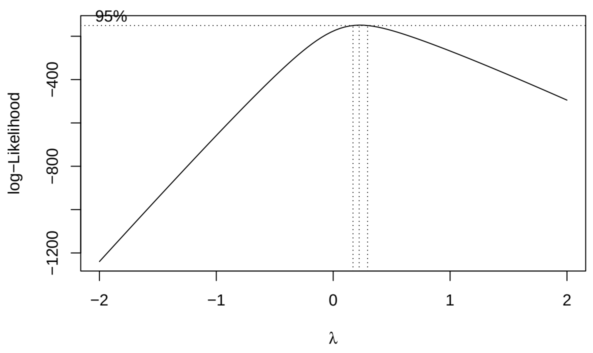
Now to choose the best transformation for our series, well take a look at the table below:

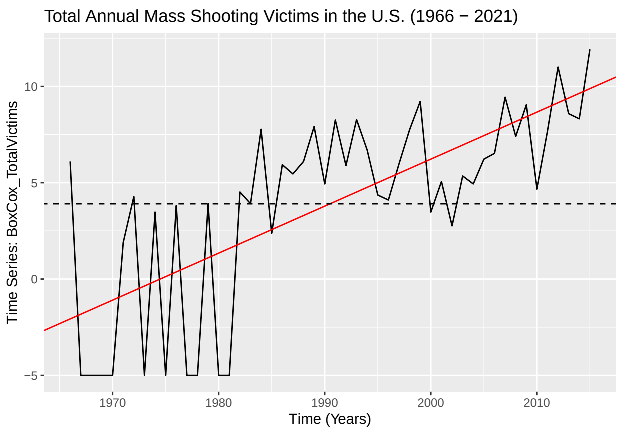
Series	Variance	Mean	Description
Total_Victims_train	7553.334	65.180	Original Time Series
BoxCox_TotalVictims	24.571	3.906	Box Cox Transformed Series
sqrt_TotalVictims	24.726	6.399	Square Root Transformed Series

Looking at the table above as well as keeping our previous plots in mind. Although both the Box-Cox Transformation and the Square Root Transformation stabilize our variance, we can conclude the Box-Cox Transformation is most suitable as it not only stabilizes our variance best, but it also brings the mean much closer to 0. To calculate our transformation we will use our lambda value of $\lambda \approx 0.2$ and substitute it into the following transformation:

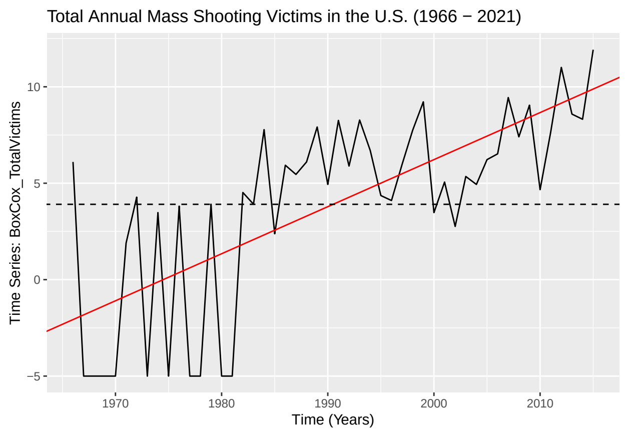
$$\frac{1}{\lambda}(X_t^\lambda - 1), \text{ where } X_t = \text{ Total\_Victims\_train}$$

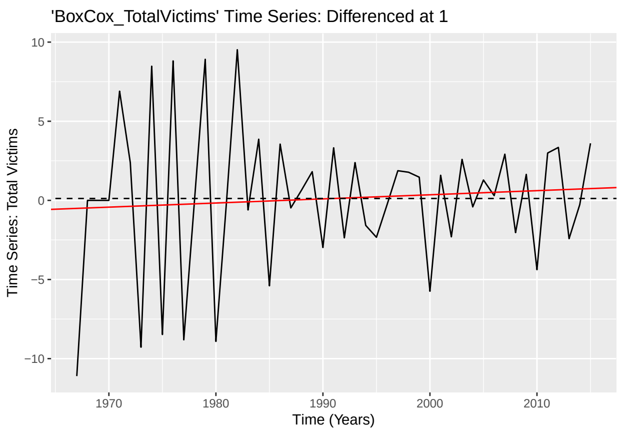
we can the 95% confidence interval for our value of λ in our Log-Likelihood plot below, and we can also see that the time series plot of our Box-Cox transformation sufficiently seems to reduce variance and trend in the visual sense as well.

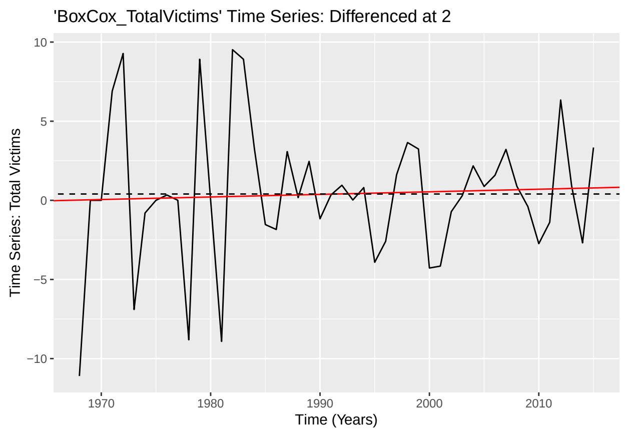


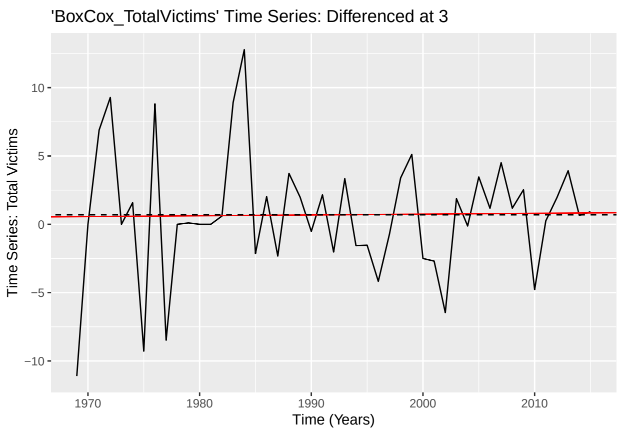


Now that we have narrowed down the suitable transformations down, finally choosing a Box-Cox Transformation for our series, we take a look at the levels of differencing that further stabilize the variance of our series while also bringing the mean to approximately zero or removing trend. Additionally, although we do not suspect seasonality, we will be checking for any sign of its presence in our series.









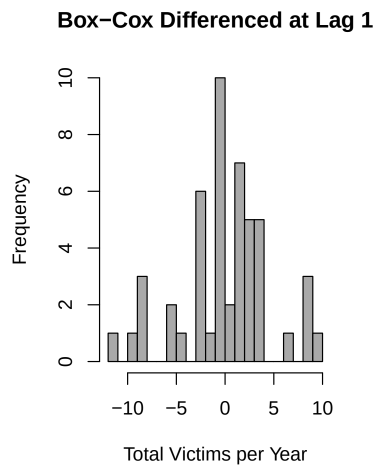
As we can see from our 4 Time Series plots above of our original Box-Cox transformed series, its differencing at lag 1, its differencing at lag 2, and at lag 3; we see that (visually) a differencing at lag 1, 2, and 3 all seem to stabilize our variance and reduce trend best in combination with a Box-Cox transformation: where differencing at lag 4 and above there was a fairly significant visual increase in trend and variance, a significant indication of over-differencing. Additionally, for the sake of discovering any seasonality, a first order differencing for higher values of lag were also calculated and plotted, but as we had guessed there did not seem to be evidence of seasonality after differencing and concluded that there was clear over-differencing for values of lag greater than 4. However, after calculating the variances at each level of differencing, we can take a closer look to verify at which lag value(s) does differencing most reduce trend and stabilize variance:

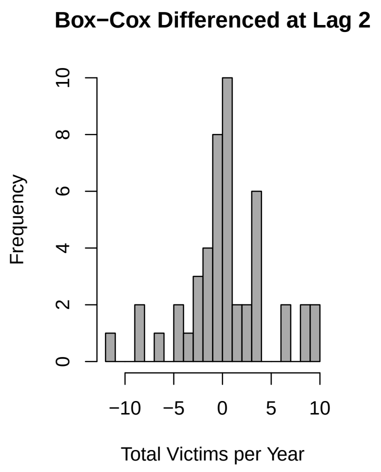
Series	Variance	Mean	Description
Total_Victims_train	7553.334	65.180	Original Time Series Object
BoxCox_TotalVictims	24.571	3.906	Box-Cox Transformation (prior to differencing)
BoxCox_TotalVictims_diff1	21.718	0.119	Box-Cox Transformation (difference at lag 1)
BoxCox_TotalVictims_diff2	19.405	0.399	Box-Cox Transformation (difference at lag 2)
BoxCox_TotalVictims_diff3	21.114	0.696	Box-Cox Transformation (difference at lag 3)

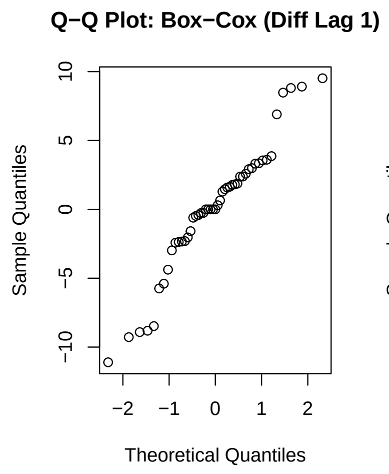
As evidenced by the table above, when differencing at lag 3, we see an increase in variance and can therefore suspect over-differencing: concluding that differencing at lag 1 and lag 2 are most suitable to making our series stationary. Now we can take a look at some plots to begin our model selection process.

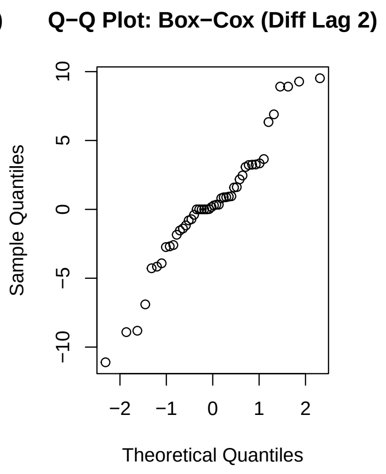
4.2 Distribution of Data (after transforming and differencing)

Now that we've taken a quick look at our ACF and PACF plots, lets do a check for normality by viewing our Box-Cox Transformed data after differencing. To do this we'll see our data's distribution using a histogram and we will also view a Normal Q-Q Plot.









To no surprise, our Box-Cox transformed series, differenced at lag 1 and lag 2, both have very similar distributions that both resemble approximately normal distributions. So, in order to assist us in our most suitable differenced series, we will utilize a Box-Ljung, KPSS, and ADF test to test for stationarity in trend and variance.

```r
Box.test(BoxCox_TotalVictims_diff1, lag=20, type="Ljung") # p-value = 0.004461
```

```
##
##  Box-Ljung test
##
## data:  BoxCox_TotalVictims_diff1
## X-squared = 40.387, df = 20, p-value = 0.004461
```

```r
kpss.test(BoxCox_TotalVictims_diff1, null="Trend") # p-value = 0.1
```

```
##
##  KPSS Test for Trend Stationarity
##
## data:  BoxCox_TotalVictims_diff1
## KPSS Trend = 0.062511, Truncation lag parameter = 3, p-value = 0.1
```

```r
adf.test(BoxCox_TotalVictims_diff1) # p-value = 0.01
```

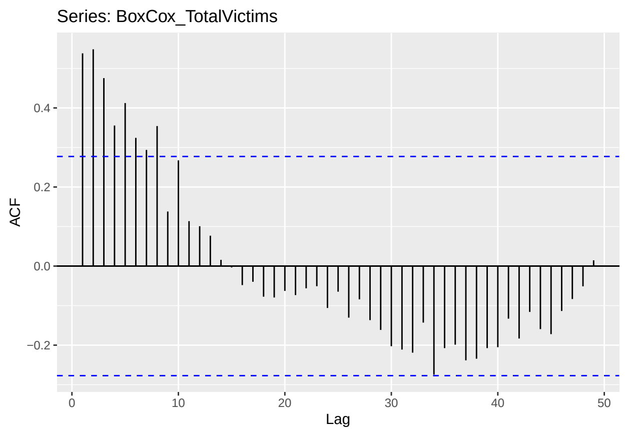
```
##
##  Augmented Dickey-Fuller Test
##
## data:  BoxCox_TotalVictims_diff1
## Dickey-Fuller = -5.8826, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

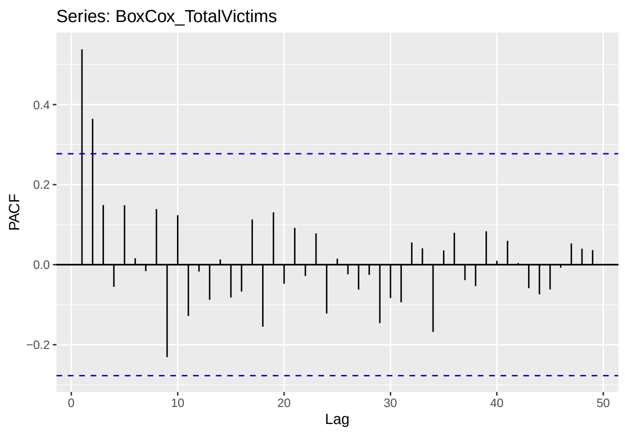
As our tests will show, using our original Box-Cox transformed series and Box-Cox series differenced at lag 3 as baselines for non-stationarity, our series differenced at lag 1 proves to be our most stationary time series. Therefore, we can move onto the model selection process using our stationary series, `BoxCox_TotalVictims_diff1`.

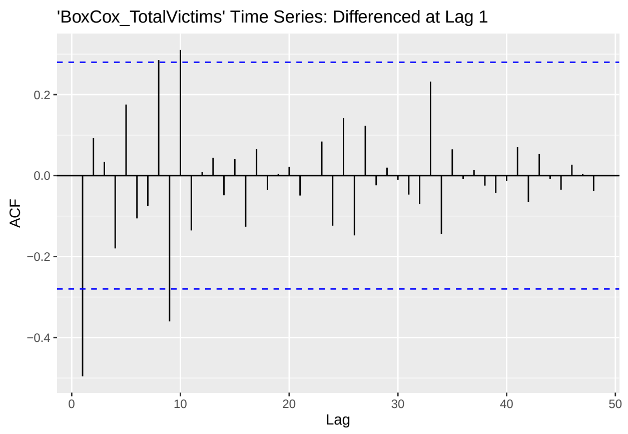
5 Model Selection

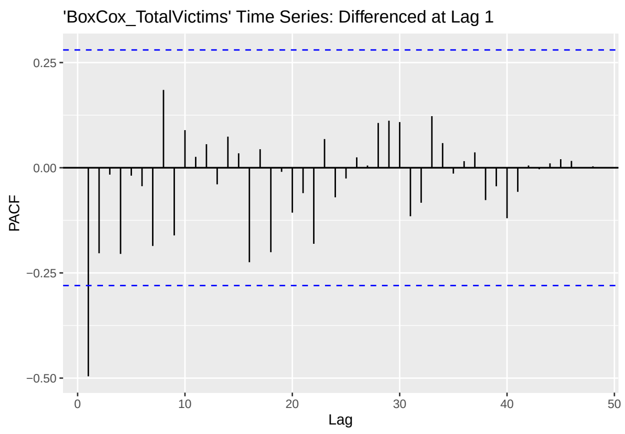
5.1 ACF and PACF (after transforming and differencing)

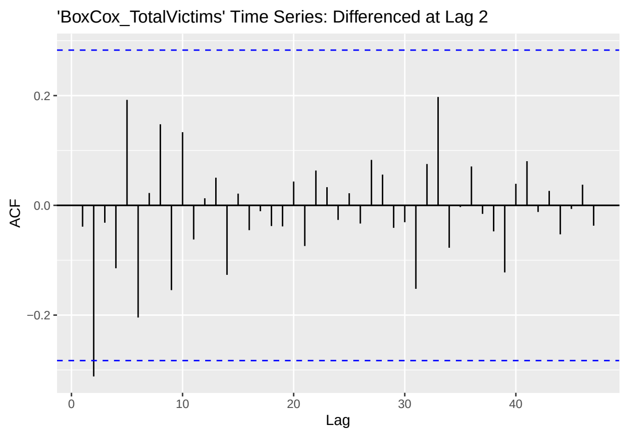
The next step in the model selection process is looking at the **ACF** and **PACF** plots of our time series model, also known as the **Auto Correlation Function** and **Partial Correlation Function**. As we discussed in the previous section, we saw in our table of variances and means that a differencing at lag 3 appeared to be over differencing, so we'll be taking a look at their ACF and PACF plots. Although we've already decided on differencing at lag 1 is best for making our Box-Cox transformed series more stationary, we'll be taking a look at our original transformed series as well as its difference at lag 2 for the purposes of comparison.

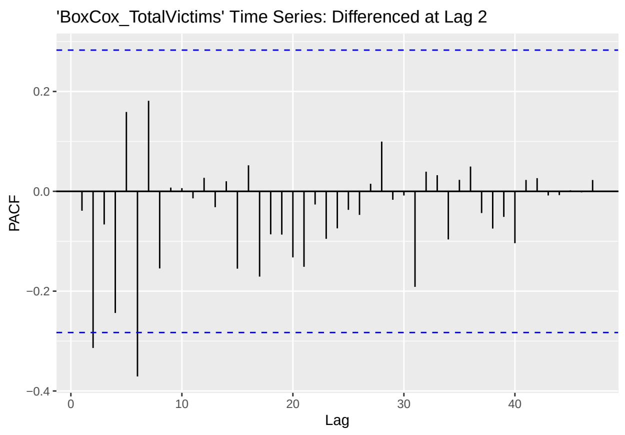












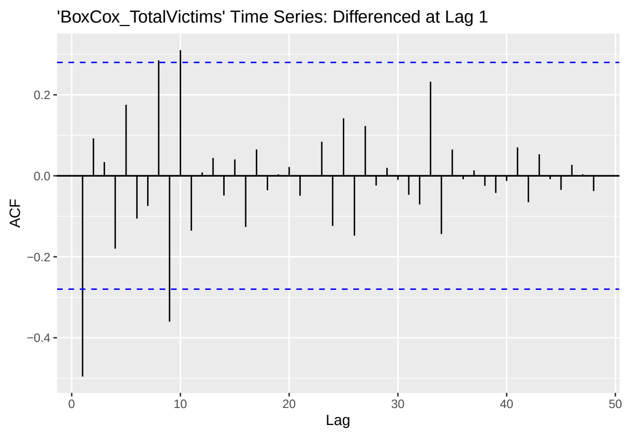
In comparison to the ACF and PACF plots of our original Box-Cox transformed series, differencing at lag 1 and lag 2 both seem to have made our plots of ACF and PACF much more stationary, as evidenced by rapid the decay of auto correlation and partial auto correlations. We also seem to have successfully first order differenced our series seeing no presence of seasonality. However, although not a defining factor of a non-stationary series, it should be noted that the ACF and PACF plot of the series differenced at lag 2 has a value of 0 (or below the confidence interval) at the first lag and a value above the confidence interval at the second lag. Due to this, it is possible that our series may not be completely stationary or the best combination of differencing, which could be an explanation for our earlier Box-Ljung, KPSS, and ADF test results that preferred differencing at lag 1.

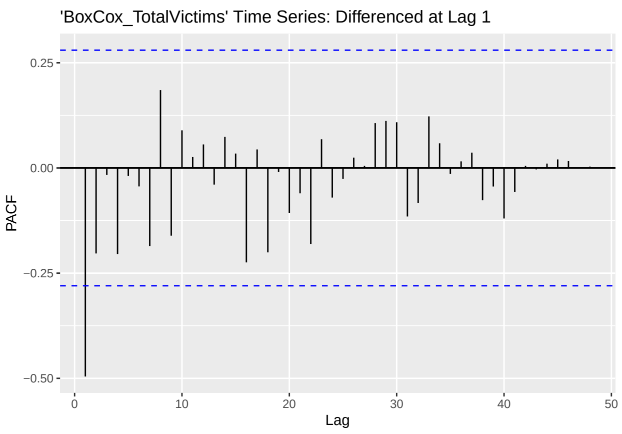
In `R`, the `boxcox()` function takes a time series object as input, and returns a list containing the transformed data, along with the lambda value. The lambda value is used to reverse the Box-Cox transformation after the model is fit, in order to make predictions on the original scale of the data. To calculate the optimal lambda value, the `boxcox()` function performs a search over a range of lambda values and selects the one that maximizes the log-likelihood function. This is done by using the `MASS` package in `R`. Once the optimal lambda value has been determined using the `boxcox()` function, it can be used in the `arima()` or other modeling functions in `R` to fit our stationary time series model. The lambda value can be passed to the lambda argument in the `arima()` function, which will apply the inverse Box-Cox transformation to the model predictions.

Knowing this information from our ACF and PACF plots, we'll begin coming up with some suitable models for our `BoxCox_TotalVictims_diff1` series, which at first glance I will guess resembles ARIMA judging from its appearance.

5.2 Model Identification

After carefully inspecting the nature of our ACF and PACF plots of `BoxCox_TotalVictims_diff1` series, which we can see again below:





Referring to our plots on the previous page, we can point out a few important characteristics concerning the ACF and PACF lag patterns. We will be looking for terms p, d, q, since we suspect an $ARIMA(p, d, q)$ model. Using our plots, we can think of some models that best represents our series:

- We differenced at lag 1 to remove our trend meaning we have $d = 1$.
- For our ACF plot we see a large value of ≈ -0.5 at lag 1 that seems to cut of at lag 1 and gradually decay and approach 0, so we'll say we have a parameter $q = 1$

- In our PACF plot we also see a large value of ≈ -0.5 at lag 1 as well, that gradually decays approaching 0, for this reason we'll suspect the presence of an AR term $p = 1$

This leaves us with a suspected model of $ARIMA(1, 1, 1)$ to test, however, using two different model selection criteria: the `AICc()` function, which calculates the **Second-order Akaike Information Criterion**, and the `BIC()` function, which calculates the **Bayesian Information Criterion**; we will be testing multiple ARIMA models in `R` including:

- $ARIMA(0, 0, 1)$, $ARIMA(1, 0, 1)$, $ARIMA(1, 0, 0)$, $ARIMA(1, 1, 1)$, $ARIMA(0, 1, 1)$, $ARIMA(1, 1, 0)$

```
## [1] AICc Comparison of All Models
```

	Model	AICc
ARIMA(0, 0, 1)	arima001_fit	271.0137
ARIMA(1, 0, 1)	arima101_fit	273.3521
ARIMA(1, 0, 0)	arima100_fit	279.4466
ARIMA(1, 1, 1)	arima111_fit	279.644
ARIMA(0, 1, 1)	arima011_fit	292.1284
ARIMA(1, 1, 0)	arima110_fit	307.1699

```
## [1] BIC Comparison of All Models
```

	Model	BIC
ARIMA(0, 0, 1)	arima001_fit	276.1559
ARIMA(1, 0, 1)	arima101_fit	280.0103
ARIMA(1, 0, 0)	arima100_fit	284.5887
ARIMA(1, 1, 1)	arima111_fit	284.7121
ARIMA(0, 1, 1)	arima011_fit	295.6042
ARIMA(1, 1, 0)	arima110_fit	310.6457

To my surprise, it seems that out of all of our models tested, an $ARIMA(1,1,1)$ does not seem to have the lowest AICc or BIC values but rather ranks as our 4th best model for our `BoxCox_TotalVictims_diff1` series: with an AICc of 279.644 and a BIC of 284.7121. However interestingly enough, an $ARIMA(0,0,1)$ model does seem to have the lowest values for the Second-order Akaike Information Criterion (AICc), with an AICc of 271.0137, and for the Bayesian Information Criterion (BIC), with a BIC of 276.1559.

Even though we have differenced at lag 1, our best fitting model does not have a parameter of $d = 1$. Although it is first instinct to suspect that this differencing at lag 1 should result in a differencing term of $d = 1$, this is not always the case. A possible reason for this could be that the residual series of the $ARIMA(0,0,1)$ model appears to be white noise in these calculations, and after taking into account all of the information in our series, it concludes that no further differencing is needed. As can be seen from our calculations above, it seems that an $ARIMA(0,0,1)$ model is a better fit for our stationary time series, despite not including a differencing term.

Now since we still need to go through one more step before choosing a final model for forecasting, we will be looking at our models with the 3 lowest values of AICc and BIC:

- Our "best" and lowest AICc and BIC scoring model:

$$ARIMA(0,0,1)$$

- The 2^{nd} lowest AICc and BIC scoring model:

$$ARIMA(1,0,1)$$

- And our 3^{rd} lowest AICc and BIC scoring model:

$$ARIMA(1,0,0)$$

Therefore, after this step of model criterion comparison, we can be confident in proceeding onto our next step of **Model Diagnostic Checking** with our primarily assumed 3 best fitting models.

- Our "best" and lowest AICc and BIC scoring model:

$$ARIMA(0,0,1)$$

- The 2^{nd} lowest AICc and BIC scoring model:

$$ARIMA(1,0,1)$$

- And our 3^{rd} lowest AICc and BIC scoring model:

$$ARIMA(1,0,0)$$

6 Model Diagnostic Checking

After computing multiple criterion for all of our assumed models in our previous step of preliminary **Model Selection**, we will now begin **Model Diagnostic Checking** for our top 3 best fitting models that were chosen based on having the lowest AICc and BIC scores:

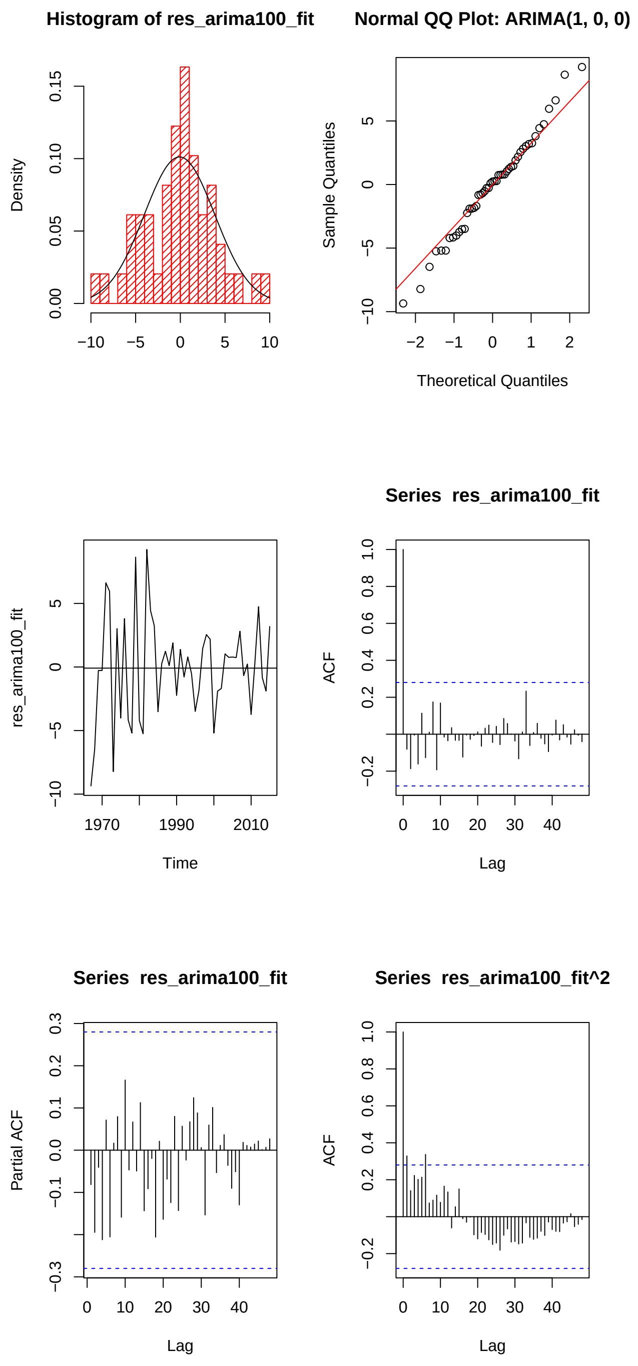
- $ARIMA(0,0,1)$
- $ARIMA(1,0,1)$
- $ARIMA(1,0,0)$

In this step we will first be using a multitude of methods to compare our models that will focus on the nature of their residuals. By visualizing the residuals of the fitted models with Histograms, Normal QQ Plots, Time Series Plots, ACF Plots, PACF Plots, ACF Plots of Squared Residuals; and by evaluating some statistical tests such as the Shapiro-Wilk Test, Box-Pierce Test, Box-Ljung Test, and Box-Ljung Test of Squared Residuals; we will be checking for the normality and stationarity of the residuals.

Next, we will be calculating unit root test for of our models to check invertibility of our fitted series, and eliminating any models that are not, in order to select our final model that is suitable for our final step of forecasting. When referring to the plots of our models' unit roots, we will be plotting our roots and inverse roots. In order for our models to pass this test of invertibility, both the roots and inverse roots must lie inside our unit circle.

6.0.1 Diagnostic Checking of Model: ARIMA(1, 0, 0)

We will start by evaluating the residuals of our 3rd best fitting model, $ARIMA(1, 0, 0)$.



```
##
##  Shapiro-Wilk normality test
##
## data:  res_arima100_fit
## W = 0.98867, p-value = 0.9154
##
##  Box-Pierce test
##
## data:  res_arima100_fit
## X-squared = 9.4521, df = 9, p-value = 0.3966
##
##  Box-Ljung test
##
## data:  res_arima100_fit
## X-squared = 11.333, df = 9, p-value = 0.2536
##
##  Box-Ljung test
##
## data:  res_arima100_fit^2
## X-squared = 22.93, df = 10, p-value = 0.01101
##
## Call:
## ar(x = res_arima100_fit, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  15.55
```

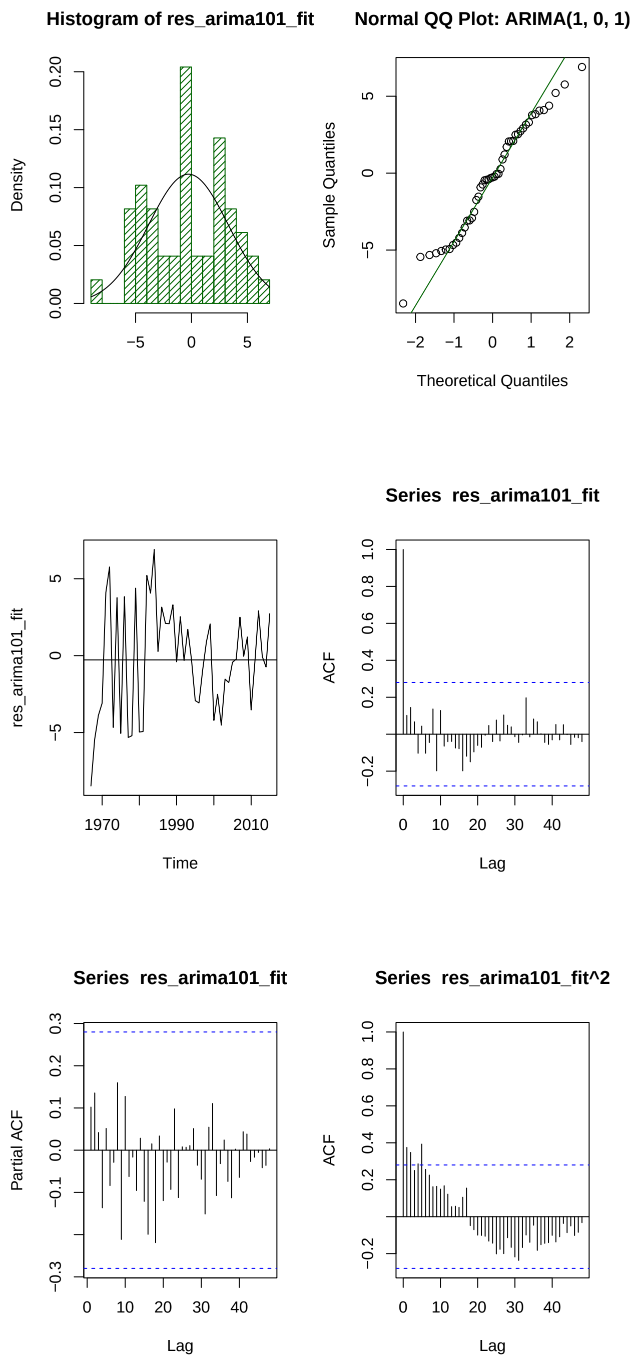
After examining our plots and tests above of our 3rd best fitting model, ARIMA(1, 0, 0), we see:

- The Histogram of the Residuals: seems to very closely follow a normal distribution with a mean of approximately 0, and no clear signs of skewness
- The Normal Q-Q Plot of the Residuals: follows a straight line indicating probable normality
- The Time Series Plot of Residuals: has a seemingly stationary appearance with no signs of trend or large spikes that would indicate unstable variance
- The ACF Plot of Residuals: has an ACF of entirely zero at all lags
- The PACF Plot of Residuals: has no PACF values that are non-zero for all lags
- The ACF Plot of Squared Residuals: seems to have non-zero lags at lag 1 and lag 6
- The Shapiro-Wilk Normality Test of Residuals: with a high $p-value = 0.9154$
- The Box-Pierce Test of Residuals: with a large $p-value = 0.3966$
- The Ljung-Box Test of Residuals: with a $p-value = 0.2536$ which is greater than our significance level of $\alpha = 0.05$
- The Ljung-Box Test of Squared Residuals: with a small $p-value = 0.01101$

Therefore, given the appearance of our plots above as well as the results of our tests, we can conclude the residuals of our fitted $ARIMA(1, 0, 0)$ model are stationary and pass all tests that indicate goodness of fit. Hence, we have no reason to reject this model as a non-suitable final model for forecasting, and can therefore move this model onto the next step of checking for invertibility.

6.0.2 Diagnostic Checking of Model: ARIMA(1, 0, 1)

Now we will move onto evaluating the residuals of our 2nd best fitting model, $ARIMA(1, 0, 1)$.



```
##
##  Shapiro-Wilk normality test
##
## data:  res_arima101_fit
## W = 0.97336, p-value = 0.3275
##
##  Box-Pierce test
##
## data:  res_arima101_fit
## X-squared = 6.6287, df = 8, p-value = 0.5772
##
##  Box-Ljung test
##
## data:  res_arima101_fit
## X-squared = 7.9729, df = 8, p-value = 0.4361
##
##  Box-Ljung test
##
## data:  res_arima101_fit^2
## X-squared = 42.009, df = 10, p-value = 7.472e-06
##
## Call:
## ar(x = res_arima101_fit, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  12.79
```

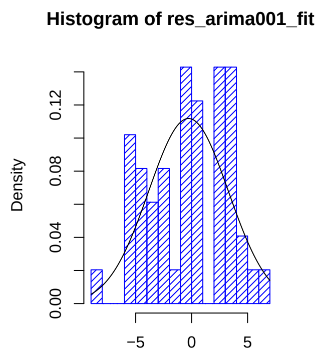
After examining our plots and tests above of our 2nd best fitting model, ARIMA(1, 0, 1), we see:

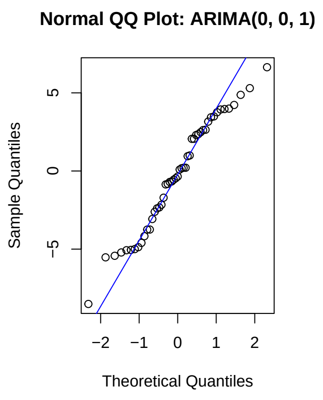
- The Histogram of the Residuals: seems to very closely follow a normal distribution with a mean of approximately 0, although has sparsity in the region adjacent to the mean
- The Normal Q-Q Plot of the Residuals: appears to follow a straight line indicating probable normality, but has sample quantiles that deviate from normality at the tails
- The Time Series Plot of Residuals: has a seemingly stationary appearance with no signs of trend, but has a somewhat large spike around 1985 that may indicate unstable variance
- The ACF Plot of Residuals: has an ACF of entirely zero at all lags
- The PACF Plot of Residuals: has no PACF values that are non-zero for all lags
- The ACF Plot of Squared Residuals: seems to have non-zero lags at lag 1, 2, 4 and lag 5
- The Shapiro-Wilk Normality Test of Residuals: with a high $p-value = 0.3275$
- The Box-Pierce Test of Residuals: with a large $p-value = 0.5772$
- The Ljung-Box Test of Residuals: with a $p-value = 0.4361$ which is greater than our significance level of $\alpha = 0.05$
- The Ljung-Box Test of Squared Residuals: with a small $p-value = 7.472e-06$

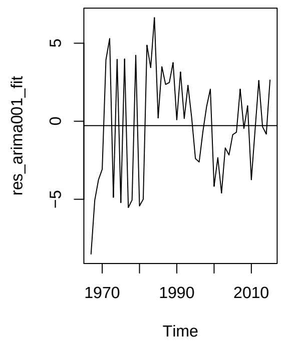
Therefore, given the appearance of our plots above as well as the results of our tests, we can conclude the residuals of our fitted $ARIMA(1,0,1)$ model seem to indicate stationary and pass all tests that verify goodness of fit. Hence, we have no reason to reject this model as a non-suitable final model for forecasting, and can therefore move this model onto the test of invertibility.

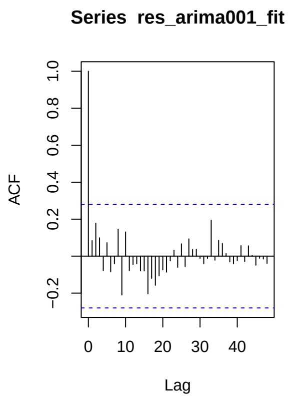
6.0.3 Diagnostic Checking of Model: ARIMA(0, 0, 1)

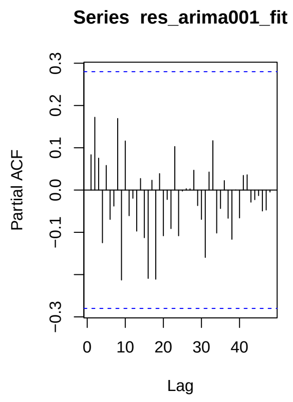
Finally we will evaluate the residuals of our model with the lowest AICc and BIC scores, our best fitting model: $ARIMA(0, 0, 1)$.

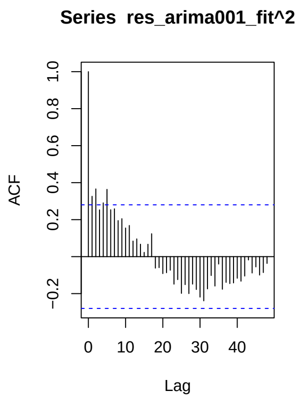












```
##
##  Shapiro-Wilk normality test
##
## data:  res_arima001_fit
## W = 0.9684, p-value = 0.209
##
##  Box-Pierce test
##
## data:  res_arima001_fit
## X-squared = 7.4106, df = 9, p-value = 0.5945
##
##  Box-Ljung test
##
## data:  res_arima001_fit
## X-squared = 8.8873, df = 9, p-value = 0.4477
##
##  Box-Ljung test
##
## data:  res_arima001_fit^2
## X-squared = 42.52, df = 10, p-value = 6.059e-06
##
## Call:
## ar(x = res_arima001_fit, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  12.72
```

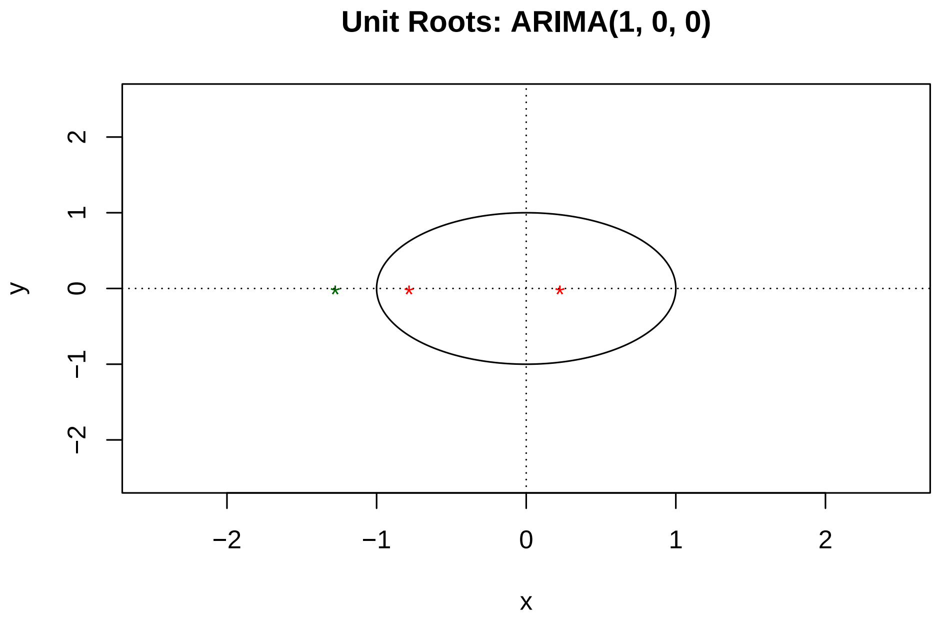
After examining our plots and tests above of our best fitting model, ARIMA(0, 0, 1), we can see:

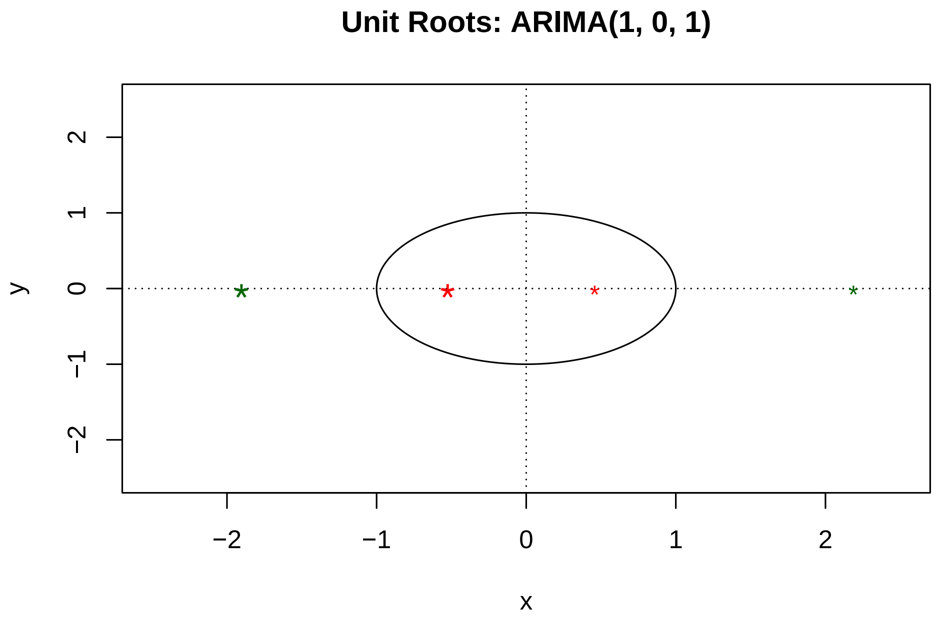
- The Histogram of the Residuals: seems to very closely follow a normal distribution with a mean of approximately 0, and no clear signs of skewness
- The Normal Q-Q Plot of the Residuals: appears to follow a straight line indicating probable normality, but has sample quantiles that deviate from the normality line at both tail ends
- The Time Series Plot of Residuals: has a seemingly stationary appearance with no signs of trend, but has a fairly large spike around the year 1985 that may indicate unstable variance
- The ACF Plot of Residuals: has an ACF of entirely zero at all lags
- The PACF Plot of Residuals: has no PACF values that are non-zero for all lags
- The ACF Plot of Squared Residuals: seems to have non-zero lags at lag 1, 2 and 5
- The Shapiro-Wilk Normality Test of Residuals: with a somewhat high $p - value = 0.209$ which is still larger than our significance level of $\alpha = 0.05$
- The Box-Pierce Test of Residuals: with a large $p - value = 0.5945$
- The Ljung-Box Test of Residuals: with a fairly large $p - value = 0.4477$
- The Ljung-Box Test of Squared Residuals: with a small $p - value = 6.059e - 06$

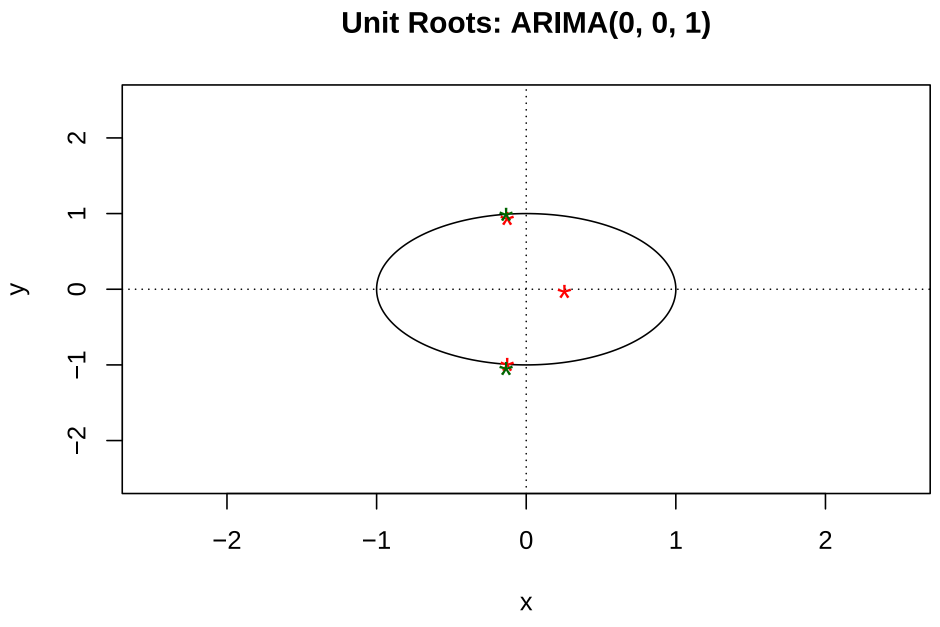
Therefore, given the appearance of our plots above as well as the results of our tests, we can conclude the residuals of our fitted $ARIMA(0, 0, 1)$ model are stationary and pass all tests that indicate goodness of fit. Hence, we have no reason to reject this model as a non-suitable final model for forecasting, and can therefore move this model onto the next step of checking for invertibility.

6.0.4 Unit Root Testing

Now that we've verified that all 3 of our assumed models' residuals display normality, stationarity, and also show no signs of dependence in the residuals, we can now move onto our testing of invertibility. To accomplish this, we'll be looking at the Unit Root Plots of our 3 assumed models to check for invertibility, which again can be indicated by having the roots (the green points) outside the unit circle and inverted roots (the red points) within the unit circle.







Looking at our plots above, it seems that all of our fitted models seem to pass the test for invertibility having all roots (green) outside the unit circle and all inverted roots (red) within the unit circle. Due to this, we will be defaulting to choosing our originally chosen best fitting model as our final model for the final step of this study. Due to having the lowest values of AICc and BIC in comparison to our other 2 chosen models, our final Time Series model that is most suitable for the last stage of **Forecasting** is our model in the plot above:

$$ARIMA(0, 0, 1)$$

7 Forecasting

Considering the primary assumed best fitting model was an $ARIMA(1,1,1)$, it is somewhat surprising to see our final selected model. After having an initial 6 ARIMA models to choose from, after comparing model criterion with **AICc** and **BIC**, computing **Diagnostic Tests of Residuals**, and **Unit Root Testing** for **Invertibility**, we have successfully chosen our final model:

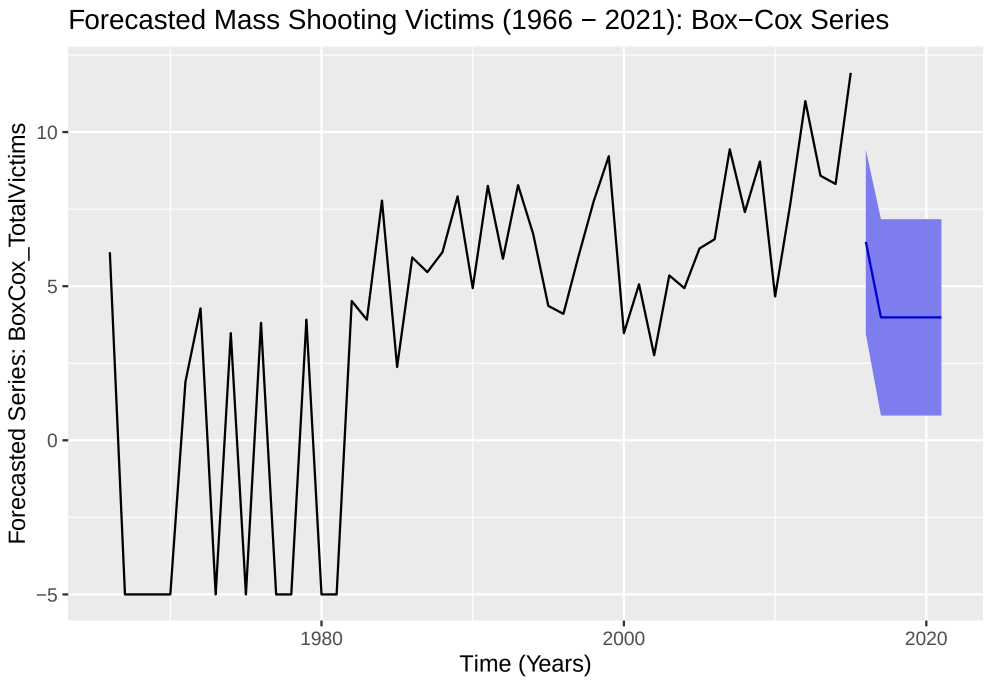
$$ARIMA(0,0,1)$$

which we can re-write using **Back-shift Operator Notation** as:

$$\begin{aligned}\Delta_1 BoxCox(U_t) &= \theta(B)Z_t \\ &= (1+(-0.8818)_{(0.1859)}B)Z_t \\ \text{where } U_t &= \text{ `Total\_Victims\_train` and} \\ \text{where } \hat{\sigma}_Z^2 &= 12.72\end{aligned}$$

Knowing this, we can now begin our forecasting. For this study, we will be forecasting the next 6 observations, which in this context is the next 6 years of data $(2016-2021)$. To test the accuracy of our chosen model, we will first begin by forecasting our non-differenced Box-Cox transformed series.

7.0.1 Forecasting: Box-Cox Transformed Series (prior to differencing)

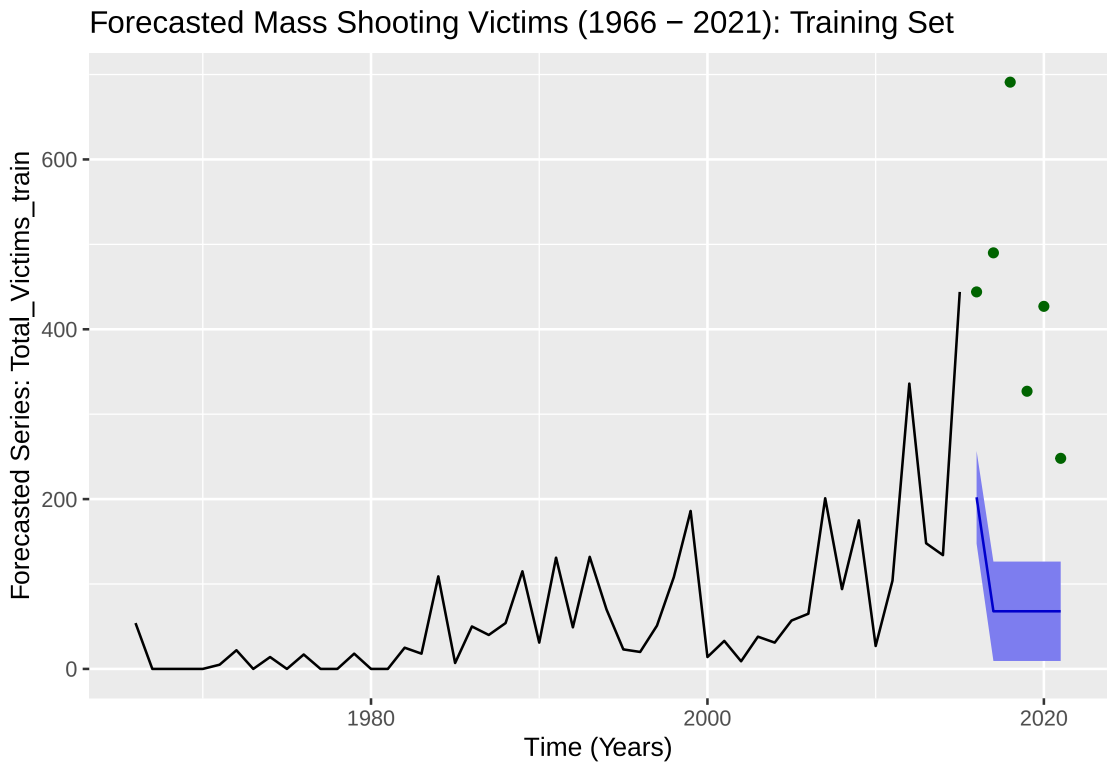


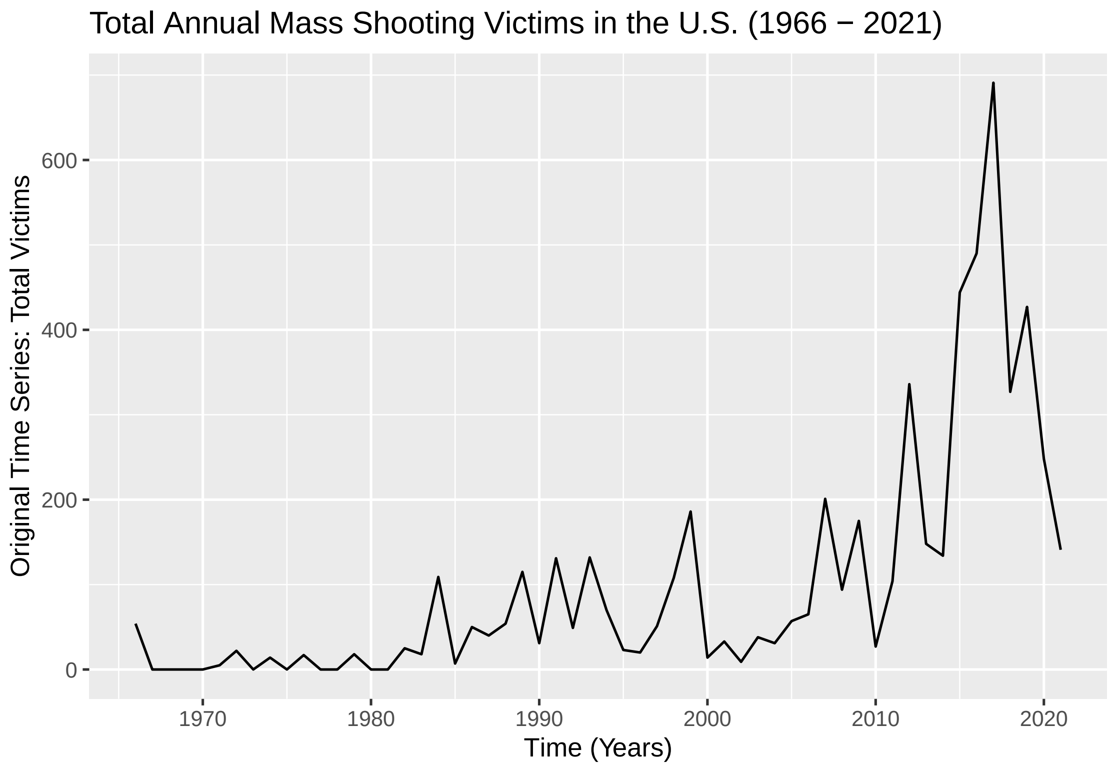
As we can see above, we have successfully plotted the forecasted data of our Box-Cox Transformed series prior to differencing (`BoxCox_TotalVictims`). Looking 6 years into the future, we now have data for the years 2016 – 2021 for a Box-Cox Transformation of our series. Looking at the table of our predicted values below, it seems that for the years 2017 – 2021, we have a plateau of values.

	Point Forecast	Lo 50	Hi 50
2016	6.439673	3.4481644	9.431181
2017	3.988219	0.8023181	7.174120
2018	3.988219	0.8023181	7.174120
2019	3.988219	0.8023181	7.174120
2020	3.988219	0.8023181	7.174120
2021	3.988219	0.8023181	7.174120

Now, we can test the accuracy of our chosen ARIMA model by comparing our forecasted training series (`Total_Victims_train`) against the true observed values of our testing series (`Total_Victims_test`).

7.0.2 Forecasting: Training Split Subset of Original Series



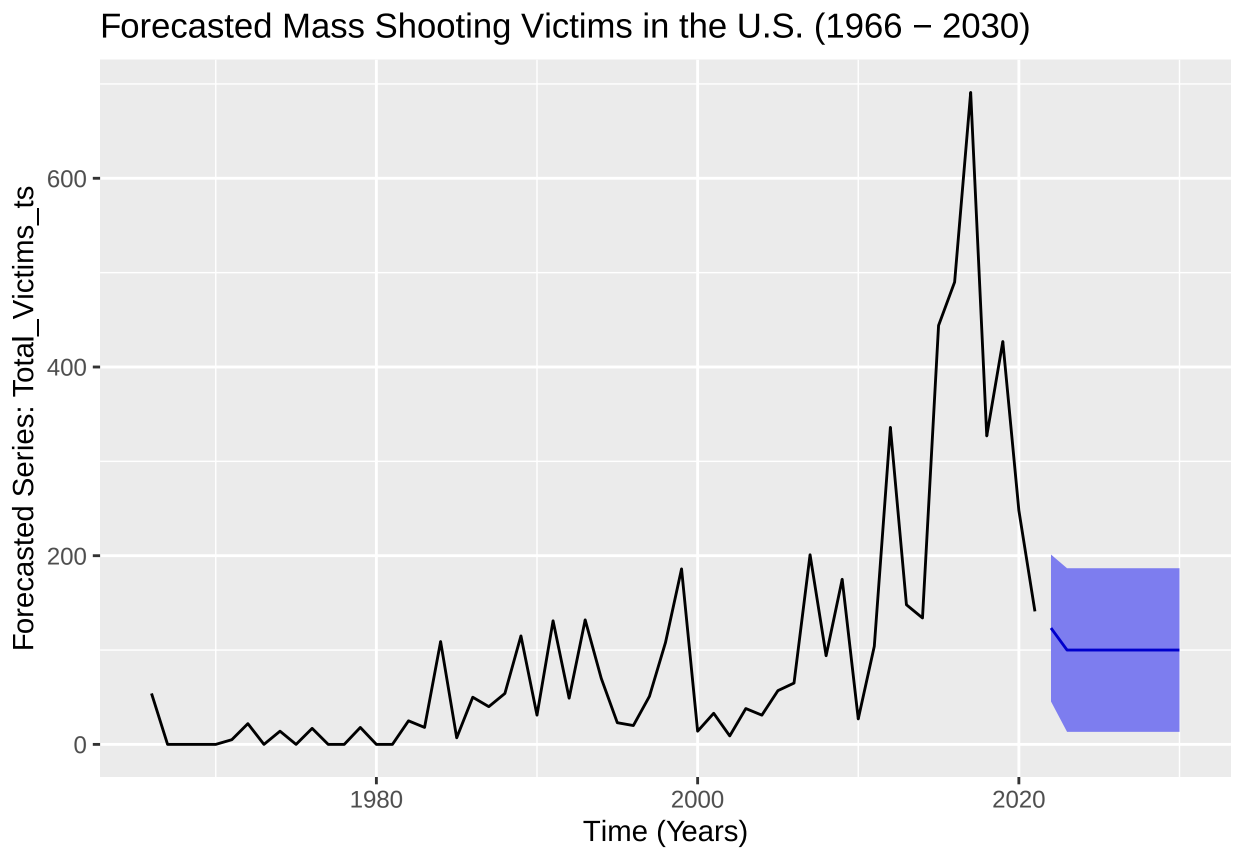


Looking at our plots above, we can see that in the first plot, our forecasting was unfortunately not the most accurate in comparison to the original data (represented by the green points). We can further see this in comparison to the second plot, which is the time series plot of our original series (`Total_Victims_ts`). Looking at the table of values below, our predicted number of total mass shooting victims in the U.S. for the year 2016 is about 242 less than that of the true value of 444 victims. We also see that like our forecasting of the Box-Cox Transformed series, we have a plateau for the years 2017 – 2021.

	Point Forecast	Lo 50	Hi 50	Original Data
2016	202.15616	147.536950	256.7754	444
2017	67.90388	9.377381	126.4304	490
2018	67.90388	9.377381	126.4304	691
2019	67.90388	9.377381	126.4304	327
2020	67.90388	9.377381	126.4304	427
2021	67.90388	9.377381	126.4304	248

7.0.3 Forecasting: Forecasted Total Mass Shooting Victims in the U.S. (1966 - 2030)

Although our model was not the best at predicting our original values, lets see what our model predicts for the total number of mass shooting victims in the U.S., 9 years after the scope of our data for the years $2022 - 2030$.



Just as expected, we do see a plateau in values. At year 2023 we see our forecasted series plateaus at a value of approximately 100.0001.

	Point Forecast	Lo 50	Hi 50
2022	123.3563	45.48666	201.2260
2023	100.0001	13.28799	186.7122
2024	100.0001	13.28799	186.7122
2025	100.0001	13.28799	186.7122
2026	100.0001	13.28799	186.7122
2027	100.0001	13.28799	186.7122
2028	100.0001	13.28799	186.7122
2029	100.0001	13.28799	186.7122
2030	100.0001	13.28799	186.7122

8 Conclusion

Starting the model selection process, 6 models were considered and compared using the Second-Order Akaike Information Criterion (AICc) and the Bayesian Information Criterion (BIC) tests. As a result, the suitable models were narrowed down to the $ARIMA(0,0,1)$, $ARIMA(1,0,1)$, and $ARIMA(1,0,0)$ models. After moving these 3 models assumptions onto the diagnostic testing stage of this study, after examining visualizations of these models' residuals using Histograms, Normal QQ Plots, Time Series Plots, ACF Plots, PACF Plots, ACF Plots of Squared Residuals; and by evaluating some statistical tests such as the Shapiro-Wilk Test, Box-Pierce Test, Box-Ljung Test, and Box-Ljung Test of Squared Residuals: all models proved to be suitable for our original series. Additionally, a test of invertibility of the models were evaluated using plots of their unit roots, which also did not prove to be a useful method of narrowing down the options for a final suitable model (as all models were invertible). Finally, the final chosen model was selected based off of previously computed AICc and BIC evaluations, leaving a final model of an $ARIMA(0,0,1)$ or in back-shift operator notation: $\Delta_1 BoxCox(U_t) = (1 + (-0.8818)_{(0.1859)}B)Z_t$.

As evidenced by the previous section, although the seemingly best possible fitting model was found, an $ARIMA(0,0,1)$, it had unfortunately proven itself to not to be the most accurate forecasting model for predicting the years $2016-2021$ for our original series. When tested against the testing split of the data set of our original series, we saw that our forecasted number of "Total Mass Shooting Victims in the U.S. for the year 2016" was about 242 victims less than that of the true value of 444 victims. We also see that like our forecasting of the Box-Cox Transformed series, we computed a plateau for the years $2017-2021$. Even so, we attempted to compute the forecast of 9 years of data into the future (for the years $2022-2030$).

To conclude, although the process of forecasting was not entirely successful, its important to note the limitations that surround this study. As a data set that was not only recent in terms of observed years, this data represents an issue that perhaps expands far past the univariate time series forecasting of this study. Predicting the future of such a daunting and unpredictable series of events was expected to be no easy task from the start. Being a topic of political controversy, one that has not seemed to be touched by the world of Data Science, is bound to not have any concrete or easy pattern of data to predict.

During this great time of sadness, fear, and chaos in the United States, its important to recognize the role that Data Science and statistical methods of research as a whole can play in helping the future of millions of lives. Although this study's forecasting model underestimated the number of mass shooting victims that would occur, perhaps this is more telling? In the hands of the wrong people, this type of research has the potential to spread possibly false statistics that discourage action. Throughout this time series forecasting analysis, it has become evident that these looming issues of gun violence and mass shootings have an exponential increase that could not be foreseen by statistical processes, meaning we have an extremely urgent matter that calls for a way to decrease its trend in order to save the lives of all those it effects.

9 References

Brockwell, P. J., & Davis, R. A. (2016). Introduction to Time Series and Forecasting (Third). Cham Springer International Publishing.

ggfortify : Extension to ggplot2 to handle some popular packages - R software and data visualization - Easy Guides - Wiki - STHDA. (n.d.). Www.sthda.com. Retrieved March 26, 2023, from http://www.sthda.com/english/wiki/ggfortify-extension-to-ggplot2-to-handle-some-popular-packages-r-software-and-data-visualization#plotting-ts-objects

Hyndman, R., & Athanasopoulos, G. (2018). Forecasting: Principles and Practice. Otexts.com; OTexts. https://otexts.com/fpp2/

Hyndman, R., Athanasopoulos, G., Bergmeir, C., Caceres, G., Chhay, L., Kuroptev, K., O'Hara-Wild, M., Petropoulos, F., Razbash, S., Wang, E., & Yasmeen, F. (2023). Forecasting Functions for Time Series and Linear Models. https://cran.r-project.org/web/packages/forecast/forecast.pdf

Package "forecast". Methods and tools for displaying and analysing univariate time series forecasts including exponential smoothing via state space models and automatic ARIMA modelling.

Plotting Time Series with ggplot2 and ggfortify. (n.d.). Rstudio-Pubs-Static.s3.Amazonaws.com. Retrieved March 26, 2023, from https://rstudio-pubs-static.s3.amazonaws.com/36420_264efd5eaf084e37a7771f3ae54016e4.html

Shumway, R. H., & Stoffer, D. S. (2017). Time series analysis and its applications : with R examples (Fourth). Springer International.

USMANI, Z.-U.-H. (2022, January). US Mass Shootings. Www.kaggle.com. https://www.kaggle.com/datasets/zusmani/us-mass-shootings-last-50-years?resource=download&select=Mass+Shootings+Dataset.csv.

10 Code Appendix

```r
# Packages Used
library(knitr)
library(ggplot2)
library(ggfortify)
library(tinytex)
library(dplyr)
library(tidyverse)
library(MASS)
library(tseries)
library(devtools)
library(forecast)
library(UnitCircle)
library(MuMIn)

# Reading in CSV files
Mass_Shootings_Dataset_csv <- read_csv("archive/Mass Shootings Dataset.csv")
Mass_Shootings_Dataset_csv2 <- read_csv("archive/Mass shooting data.csv")
Mass_Shootings_Dataset_csv$Date <- as.POSIXct(Mass_Shootings_Dataset_csv$Date,
                                              format="%m/%d/%Y")
colnames(Mass_Shootings_Dataset_csv)[1] <- "Event_ID"
colnames(Mass_Shootings_Dataset_csv)[8] <- "Total_Victims"
colnames(Mass_Shootings_Dataset_csv)[9] <- "Mental_Health_Issues"
##str(Mass_Shootings_Dataset_csv)
datacolumntypes_df <- data.frame(
  Data.Types=c("num", "chr", "chr", "POSIXct", "chr", "num", "num", "num", "chr",
               "chr", "chr", "num", "num"),
  row.names=c("Event_ID", "Title", "Location", "Date", "Summary", "Fatalities", "Injured",
              "Total_Victims", "Mental_Health_Issues", "Race", "Gender", "Latitude",
              "Longitude")
  )
colnames(datacolumntypes_df)[1] <- "Feature Column Data Types"
datacolumntypes_df

# Printing First 5 Observations of CSV Dataframe
head(Mass_Shootings_Dataset_csv[0:4], 5)
head(Mass_Shootings_Dataset_csv[5], 5)
head(Mass_Shootings_Dataset_csv[6:13], 5)

# Subsetting only "Total_Victims" and "Date columns"
mass_shootings_df <- Mass_Shootings_Dataset_csv1[, c("Total_Victims", "Date")]
##head(mass_shootings_df)

# Adding a Year identifying Column based on the POSIXct "Date" Column
mass_shootings_df$Year <- as.numeric(format(mass_shootings_df$Date, format="%Y"))
##head(mass_shootings_df)
```

```r
##unique(mass_shootings_df$Year)

# Creating a Data Frame of Values for Missing Years
Total_Victims <- c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
Date <- c(NA, NA, NA, NA, NA, NA, NA, NA, NA, NA)
Year <- c(1981, 1980, 1978, 1977, 1975, 1973, 1970, 1969, 1968, 1967)
missing_obs_df <- data.frame(Total_Victims, Date, Year)

# Binding the Missing Years Data Frame to the Original Data Frame
mass_shootings_newdf <- rbind(mass_shootings_df, missing_obs_df)
mass_shootings_newdf <- mass_shootings_newdf[order(mass_shootings_newdf$Year,
                                                   decreasing=TRUE),]

# Creating a New Data Frame Summing and Grouping Row Observations by Year
Mass_Shootings_df <- mass_shootings_newdf
Mass_Shootings_df <- aggregate(Total_Victims ~ Year, Mass_Shootings_df, sum)
## Mass_Shootings_df
Total_Victims_ts <- ts(data=Mass_Shootings_df[2],
                       start=1966, end=2021, frequency=1)
##attributes(Total_Victims_ts)

# Time Series Plot of Original Series
ts_coefficients <- coef(
  lm(Total_Victims~time(Total_Victims_ts), data=Total_Victims_ts)
  )
ts_intercept <- ts_coefficients[1]
ts_slope <- ts_coefficients[2]
ts_mean <- mean(Total_Victims_ts)

autoplot(Total_Victims_ts,
         ts.geom="line",
         xlab="Time (Years)", ylab="Time Series: Total Victims",
         main="Total Annual Mass Shooting Victims in the U.S. (1966 - 2021)") +
  geom_abline(intercept=ts_intercept,
              slope=ts_slope,
              color="red") +
  geom_abline(intercept=ts_mean,
              slope=0,
              linetype="dashed",
              color="black")

# Training and Test Split of Data
Total_Victims_train <- Mass_Shootings_df[1:50, ]
Total_Victims_test <- Mass_Shootings_df[50:56, ]
##attributes(Total_Victims_train)

# Time Series Plot of Training Set
```

```r
orig_coefficients <- coef(
  lm(Total_Victims_train~time(Total_Victims_train), data=Total_Victims_train)
  )
orig_intercept <- orig_coefficients[1]
orig_slope <- orig_coefficients[2]
orig_mean <- mean(Total_Victims_train)

autoplot(Total_Victims_train,
         ts.geom="line",
         xlab="Time (Years)", ylab="Time Series: Total Victims",
         main="'Total_Victims_train': Total Annual Mass Shooting Victims in the U.S.") +
  geom_abline(intercept=orig_intercept,
              slope=orig_slope,
              color="red") +
  geom_abline(intercept=orig_mean,
              slope=0,
              linetype="dashed",
              color="black")

hist(Total_Victims_train,
     breaks=15,
     main="Distribution of the Number of Total Victims",
     xlab="Total Victims per Year",
     col="dark gray")
#acf(Total_Victims_train, plot=FALSE)
autoplot(acf(Total_Victims_train, plot=FALSE),
         xlab="Lag", ylab="ACF",
         main="Auto Correlation - Series: Total_Victims_train") +
  geom_abline(intercept=0,
              slope=0,
              color="black")

# Transformation Calculations
## Box-Cox Transformation:
t = 1:length(Total_Victims_train)
BoxCox_Transform <- boxcox(Total_Victims_train+.00001 ~ as.numeric(t), plotit=FALSE)
lambda <- BoxCox_Transform$x[which(BoxCox_Transform$y == max(BoxCox_Transform$y))]
#lambda
BoxCox_TotalVictims <- (1/lambda)*(Total_Victims_train^lambda-1)
## Logarithmic Transformation:
log_TotalVictims <- log(Total_Victims_train)
## Square-Root Transformation:
sqrt_TotalVictims <- sqrt(Total_Victims_train)
par(mfrow=c(1, 2))

# ORIGINAL PLOT
plot.ts(Total_Victims_train,
         xlab="Time (Years)", ylab="Time Series: Total Victims",
         main="Series: Total_Victims_train")
# BOX COX PLOT
plot.ts(BoxCox_TotalVictims,
         xlab="Time (Years)", ylab="Time Series: Total Victims",
         main="Box Cox Transformation")

par(mfrow=c(1, 2))
# LOG PLOT
plot.ts(log_TotalVictims,
         xlab="Time (Years)", ylab="Time Series: Total Victims",
         main="Log Transformation")
# SQRT PLOT
plot.ts(sqrt_TotalVictims,
         xlab="Time (Years)", ylab="Time Series: Total Victims",
         main="Square Root Transformation")

# Histograms of Transformations
par(mfrow=c(2, 2))

hist(Total_Victims_train,
     breaks=15,
     main="Distribution of Original Series",
     xlab="Total Victims per Year",
     col="dark gray")
abline(v=mean(Total_Victims_train), col="red")
hist(BoxCox_TotalVictims,
     breaks=15,
     main="Distribution of Box Cox of Series",
     xlab="Total Victims per Year",
     col="dark gray")
abline(v=mean(BoxCox_TotalVictims), col="red")
hist(log_TotalVictims,
     breaks=15,
     main="Distribution of Log Transformation",
     xlab="Total Victims per Year",
     col="dark gray")
abline(v=mean(log_TotalVictims), col="red")
hist(sqrt_TotalVictims,
     breaks=15,
     main="Distribution of SQRT Transformation",
     xlab="Total Victims per Year",
     col="dark gray")
abline(v=mean(sqrt_TotalVictims), col="red")
par(mfrow=c(1, 4))

# Box Plots of Transformations
```

```r
boxplot(Total_Victims_train, horizontal=FALSE)
boxplot(BoxCox_TotalVictims, horizontal=FALSE)
boxplot(log_TotalVictims, horizontal=FALSE)
boxplot(sqrt_TotalVictims, horizontal=FALSE)

# Table of Variances and Means of Transformations
Series <- c("Total_Victims_train", "BoxCox_TotalVictims", "sqrt_TotalVictims")
Variance <- round(c(var(Total_Victims_train), var(BoxCox_TotalVictims),
                    var(sqrt_TotalVictims)), digits=3)
Mean <- round(c(mean(Total_Victims_train), mean(BoxCox_TotalVictims),
                mean(sqrt_TotalVictims)), digits=3)
Description <- c("Original Time Series", "Box Cox Transformed Series",
                 "Square Root Transformed Series")
kable(data.frame(Series, Variance, Mean, Description))

# Box-Cox Transformation Calculations
boxcox(Total_Victims_train+.00001 ~ as.numeric(t))

orig_variance <- var(BoxCox_TotalVictims)
#orig_variance

orig_coefficients <- coef(
  lm(BoxCox_TotalVictims~time(BoxCox_TotalVictims), data=BoxCox_TotalVictims)
  )
orig_intercept <- orig_coefficients[1]
orig_slope <- orig_coefficients[2]
orig_mean <- mean(BoxCox_TotalVictims)

# ORIGINAL BOX COX
autoplot(BoxCox_TotalVictims,
         ts.geom="line",
         xlab="Time (Years)", ylab="Time Series: BoxCox_TotalVictims",
         main="Total Annual Mass Shooting Victims in the U.S. (1966 - 2021)") +
  geom_abline(intercept=orig_intercept,
              slope=orig_slope,
              color="red") +
  geom_abline(intercept=orig_mean,
              slope=0,
              linetype="dashed",
              color="black")

BoxCox_TotalVictims_diff1 <- diff(BoxCox_TotalVictims, lag=1)
BoxCox_TotalVictims_diff2 <- diff(BoxCox_TotalVictims, lag=2)
BoxCox_TotalVictims_diff3 <- diff(BoxCox_TotalVictims, lag=3)
BoxCox_TotalVictims_diff112 <- diff(diff(BoxCox_TotalVictims, lag=2), lag=1)
BoxCox_TotalVictims_diff212 <- diff(diff(BoxCox_TotalVictims, lag=2), lag=2)
BoxCox_TotalVictims_diff312 <- diff(diff(BoxCox_TotalVictims, lag=2), lag=3)
```

```r
# Calculations of the Series Differenced at Various Lags that were not included
# (including checking for seasonal differencing with second-order differencing)
##27.300, 20.597, 18.728, 21.047, 23.445, 18.559, 24.227, 25.460, 18.727, 29.882,
###19.732, 30.051, 27.852
##60.220, 35.847, 40.222
##35.847, 48.25, 38.26

##4.648, 0.152, 0.511, 0.878
##0.235, 0.248, 0.338
##0.248, .346, 0.434

ts_variance <- var(Total_Victims_train)
ts_mean <- mean(Total_Victims_train)
#ts_variance
diff1_variance <- var(BoxCox_TotalVictims_diff1)
#diff1_variance
diff2_variance <- var(BoxCox_TotalVictims_diff2)
#diff2_variance
diff3_variance <- var(BoxCox_TotalVictims_diff3)
#diff3_variance
diff112_variance <- var(BoxCox_TotalVictims_diff112)
#diff112_variance
diff212_variance <- var(BoxCox_TotalVictims_diff212)
#diff212_variance
diff312_variance <- var(BoxCox_TotalVictims_diff312)
#diff312_variance

diff1_coefficients <- coef(
  lm(BoxCox_TotalVictims_diff1~time(BoxCox_TotalVictims_diff1),
     data=BoxCox_TotalVictims_diff1)
  )
diff1_intercept <- diff1_coefficients[1]
diff1_slope <- diff1_coefficients[2]
diff1_mean <- mean(BoxCox_TotalVictims_diff1)

diff2_coefficients <- coef(
  lm(BoxCox_TotalVictims_diff2~time(BoxCox_TotalVictims_diff2),
     data=BoxCox_TotalVictims_diff2)
  )
diff2_intercept <- diff2_coefficients[1]
diff2_slope <- diff2_coefficients[2]
diff2_mean <- mean(BoxCox_TotalVictims_diff2)

diff3_coefficients <- coef(
  lm(BoxCox_TotalVictims_diff3~time(BoxCox_TotalVictims_diff3),
     data=BoxCox_TotalVictims_diff3)
  )
```

```r
diff3_intercept <- diff3_coefficients[1]
diff3_slope <- diff3_coefficients[2]
diff3_mean <- mean(BoxCox_TotalVictims_diff3)

diff112_coefficients <- coef(
  lm(BoxCox_TotalVictims_diff112~time(BoxCox_TotalVictims_diff112),
     data=BoxCox_TotalVictims_diff112)
  )
diff112_intercept <- diff112_coefficients[1]
diff112_slope <- diff112_coefficients[2]
diff112_mean <- mean(BoxCox_TotalVictims_diff112)

diff212_coefficients <- coef(
  lm(BoxCox_TotalVictims_diff212~time(BoxCox_TotalVictims_diff212),
     data=BoxCox_TotalVictims_diff212)
  )
diff212_intercept <- diff212_coefficients[1]
diff212_slope <- diff212_coefficients[2]
diff212_mean <- mean(BoxCox_TotalVictims_diff212)

diff312_coefficients <- coef(
  lm(BoxCox_TotalVictims_diff312~time(BoxCox_TotalVictims_diff312),
     data=BoxCox_TotalVictims_diff312)
  )
diff312_intercept <- diff312_coefficients[1]
diff312_slope <- diff312_coefficients[2]
diff312_mean <- mean(BoxCox_TotalVictims_diff312)

# ORIGINAL BOX COX
autoplot(BoxCox_TotalVictims,
         ts.geom="line",
         xlab="Time (Years)", ylab="Time Series: BoxCox_TotalVictims",
         main="Total Annual Mass Shooting Victims in the U.S. (1966 - 2021)") +
  geom_abline(intercept=orig_intercept,
              slope=orig_slope,
              color="red") +
  geom_abline(intercept=orig_mean,
              slope=0,
              linetype="dashed",
              color="black")
# DIFF 1 BOX COX
autoplot(BoxCox_TotalVictims_diff1,
         ts.geom="line",
         xlab="Time (Years)", ylab="Time Series: Total Victims",
         main="'BoxCox_TotalVictims' Time Series: Differenced at 1") +
  geom_abline(intercept=diff1_intercept,
              slope=diff1_slope,
               color="red") +
  geom_abline(intercept=diff1_mean,
              slope=0,
              linetype="dashed",
              color="black")
# DIFF 2 BOX COX
autoplot(BoxCox_TotalVictims_diff2,
         ts.geom="line",
         xlab="Time (Years)", ylab="Time Series: Total Victims",
         main="'BoxCox_TotalVictims' Time Series: Differenced at 2") +
  geom_abline(intercept=diff2_intercept,
              slope=diff2_slope,
              color="red") +
  geom_abline(intercept=diff2_mean,
              slope=0,
              linetype="dashed",
              color="black")
# DIFF 3 BOX COX
autoplot(BoxCox_TotalVictims_diff3,
         ts.geom="line",
         xlab="Time (Years)", ylab="Time Series: Total Victims",
         main="'BoxCox_TotalVictims' Time Series: Differenced at 3") +
  geom_abline(intercept=diff3_intercept,
              slope=diff3_slope,
              color="red") +
  geom_abline(intercept=diff3_mean,
              slope=0,
              linetype="dashed",
              color="black")
# # DIFF 1 and 12 BOX COX
# autoplot(BoxCox_TotalVictims_diff112,
#          ts.geom="line",
#          xlab="Time (Years)", ylab="Time Series: Total Victims",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 12 and 1") +
#   geom_abline(intercept=diff112_intercept,
#               slope=diff112_slope,
#               color="red") +
#   geom_abline(intercept=diff112_mean,
#               slope=0,
#               linetype="dashed",
#               color="black")
# # DIFF 2 and 12 BOX COX
# autoplot(BoxCox_TotalVictims_diff212,
#          ts.geom="line",
#          xlab="Time (Years)", ylab="Time Series: Total Victims",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 12 and 2") +
#   geom_abline(intercept=diff212_intercept,
#               slope=diff212_slope,
#               color="red") +
#   geom_abline(intercept=diff212_mean,
#               slope=0,
#               linetype="dashed",
#               color="black")
# # DIFF 3 and 12 BOX COX
# autoplot(BoxCox_TotalVictims_diff312,
#          ts.geom="line",
#          xlab="Time (Years)", ylab="Time Series: Total Victims",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 12 and 3") +
#   geom_abline(intercept=diff312_intercept,
#               slope=diff312_slope,
#               color="red") +
#   geom_abline(intercept=diff312_mean,
#               slope=0,
#               linetype="dashed",
#               color="black")

# Table of Variances and Means
Series <- c("Total_Victims_train", "BoxCox_TotalVictims", "BoxCox_TotalVictims_diff1",
            "BoxCox_TotalVictims_diff2", "BoxCox_TotalVictims_diff3")
Variance <- round(c(ts_variance, orig_variance, diff1_variance,
                    diff2_variance, diff3_variance), digits=3)
Mean <- round(c(ts_mean, orig_mean, diff1_mean, diff2_mean, diff3_mean), digits=3)
Description <- c("Original Time Series Object",
                 "Box-Cox Transformation (prior to differencing)",
                 "Box-Cox Transformation (difference at lag 1)",
                 "Box-Cox Transformation (difference at lag 2)",
                 "Box-Cox Transformation (difference at lag 3)")
kable(data.frame(Series, Variance, Mean, Description))

# ACF and PACF Plots of Box-Cox
par(mfrow=c(1, 2))
acf(BoxCox_TotalVictims, plot=TRUE, lag.max=50, main="Box-Cox Transformation")
pacf(BoxCox_TotalVictims, plot=TRUE, lag.max=50, main="Box-Cox Transformation")

# ACF and PACF Plots of Box-Cox Differenced lag 1
par(mfrow=c(1, 2))
acf(BoxCox_TotalVictims_diff1, plot=TRUE, lag.max=50, main="Box-Cox Differenced at Lag 1")
pacf(BoxCox_TotalVictims_diff1, plot=TRUE, lag.max=50, main="Box-Cox Differenced at Lag 1")

# ACF and PACF Plots of Box-Cox Differenced lag 2
par(mfrow=c(1, 2))
acf(BoxCox_TotalVictims_diff2, plot=TRUE, lag.max=50, main="Box-Cox Differenced at Lag 2")
pacf(BoxCox_TotalVictims_diff2, plot=TRUE, lag.max=50, main="Box-Cox Differenced at Lag 2")
```

```
# Histograms of Box-Cox Differenced Series
par(mfrow=c(1, 2))
hist(BoxCox_TotalVictims_diff1,
     breaks=15,
     main="Box-Cox Differenced at Lag 1",
     xlab="Total Victims per Year",
     col="dark gray")
hist(BoxCox_TotalVictims_diff2,
     breaks=15,
     main="Box-Cox Differenced at Lag 2",
     xlab="Total Victims per Year",
     col="dark gray")

# Q-Q Plots of Box-Cox Differenced Series
par(mfrow=c(1, 2))
qqnorm(BoxCox_TotalVictims_diff1, main="Q-Q Plot: Box-Cox (Diff Lag 1)")
qqnorm(BoxCox_TotalVictims_diff2, main="Q-Q Plot: Box-Cox (Diff Lag 2)")

# DIFF AT LAG 1 AND 2 PASS STATIONARITY TESTS
Box.test(BoxCox_TotalVictims, lag=20, type="Ljung") # p-value = 7.917e-11
kpss.test(BoxCox_TotalVictims, null="Trend") # p-value = p-value = 0.1
adf.test(BoxCox_TotalVictims) # p-value = 0.2346

Box.test(BoxCox_TotalVictims_diff1, lag=20, type="Ljung") # p-value = 0.004461
kpss.test(BoxCox_TotalVictims_diff1, null="Trend") # p-value = 0.1
adf.test(BoxCox_TotalVictims_diff1) # p-value = 0.01

Box.test(BoxCox_TotalVictims_diff2, lag=20, type="Ljung") # p-value = 0.6876
kpss.test(BoxCox_TotalVictims_diff2, null="Trend") # p-value = 0.1
adf.test(BoxCox_TotalVictims_diff2) # p-value = 0.01

Box.test(BoxCox_TotalVictims_diff3, lag=20, type="Ljung") # p-value = 0.7198
kpss.test(BoxCox_TotalVictims_diff3, null="Trend") # p-value = 0.1
adf.test(BoxCox_TotalVictims_diff3) # p-value = 0.01915

# ACF Plots

# autoplot(acf(Total_Victims_train, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="ACF",
#          main="Series: Total_Victims_train") +
#   geom_abline(intercept=0,
#               slope=0,
#               color="black")
# autoplot(pacf(Total_Victims_train, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="PACF",
#          main="Series: Total_Victims_train") +
#   geom_abline(intercept=0,
```

```r
#                slope=0,
#                color="black")

autoplot(acf(BoxCox_TotalVictims, plot=FALSE, lag.max=50),
         xlab="Lag", ylab="ACF",
         main="Series: BoxCox_TotalVictims") +
  geom_abline(intercept=0,
              slope=0,
              color="black")
autoplot(pacf(BoxCox_TotalVictims, plot=FALSE, lag.max=50),
         xlab="Lag", ylab="PACF",
         main="Series: BoxCox_TotalVictims") +
  geom_abline(intercept=0,
              slope=0,
              color="black")

autoplot(acf(BoxCox_TotalVictims_diff1, plot=FALSE, lag.max=50),
         xlab="Lag", ylab="ACF",
         main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 1") +
  geom_abline(intercept=0,
              slope=0,
              color="black")
autoplot(pacf(BoxCox_TotalVictims_diff1, plot=FALSE, lag.max=50),
         xlab="Lag", ylab="PACF",
         main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 1") +
  geom_abline(intercept=0,
              slope=0,
              color="black")

autoplot(acf(BoxCox_TotalVictims_diff2, plot=FALSE, lag.max=50),
         xlab="Lag", ylab="ACF",
         main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 2") +
  geom_abline(intercept=0,
              slope=0,
              color="black")
autoplot(pacf(BoxCox_TotalVictims_diff2, plot=FALSE, lag.max=50),
         xlab="Lag", ylab="PACF",
         main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 2") +
  geom_abline(intercept=0,
              slope=0,
              color="black")

# autoplot(acf(BoxCox_TotalVictims_diff3, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="ACF",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 3") +
#   geom_abline(intercept=0,
#                slope=0,
#                color="black")
# autoplot(pacf(BoxCox_TotalVictims_diff3, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="PACF",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 3") +
#   geom_abline(intercept=0,
#                slope=0,
#                color="black")

# autoplot(acf(BoxCox_TotalVictims_diff112, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="ACF",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 12 and 1") +
#   geom_abline(intercept=0,
#                slope=0,
#                color="black")
# autoplot(pacf(BoxCox_TotalVictims_diff112, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="PACF",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 12 and 1") +
#   geom_abline(intercept=0,
#                slope=0,
#                color="black")

# autoplot(acf(BoxCox_TotalVictims_diff212, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="ACF",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 12 and 2") +
#   geom_abline(intercept=0,
#                slope=0,
#                color="black")
# autoplot(pacf(BoxCox_TotalVictims_diff212, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="PACF",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 12 and 2") +
#   geom_abline(intercept=0,
#                slope=0,
#                color="black")

# autoplot(acf(BoxCox_TotalVictims_diff312, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="ACF",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 12 and 3") +
#   geom_abline(intercept=0,
#                slope=0,
#                color="black")
# autoplot(pacf(BoxCox_TotalVictims_diff312, plot=FALSE, lag.max=50),
#          xlab="Lag", ylab="PACF",
#          main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 12 and 3") +
#   geom_abline(intercept=0,
#                slope=0,
#                color="black")
```

```r
autoplot(acf(BoxCox_TotalVictims_diff1, plot=FALSE, lag.max=50),
         xlab="Lag", ylab="ACF",
         main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 1") +
  geom_abline(intercept=0,
              slope=0,
              color="black")
autoplot(pacf(BoxCox_TotalVictims_diff1, plot=FALSE, lag.max=50),
         xlab="Lag", ylab="PACF",
         main="'BoxCox_TotalVictims' Time Series: Differenced at Lag 1") +
  geom_abline(intercept=0,
              slope=0,
              color="black")

# ARIMA MODELS

# Worst/Largest AICc: 307.1699
arima110_fit <- arima(BoxCox_TotalVictims_diff1,
      order = c(1, 1, 0),
      method = "ML")

# 5th Lowest AICc: 292.1284
arima011_fit <- arima(BoxCox_TotalVictims_diff1,
      order = c(0, 1, 1),
      method = "ML")

# 4th Lowest AICc: 279.644
arima111_fit <- arima(BoxCox_TotalVictims_diff1,
      order = c(1, 1, 1),
      method = "ML")

# 3rd Lowest AICc: 279.4466
arima100_fit <- arima(BoxCox_TotalVictims_diff1,
      order = c(1, 0, 0),
      method = "ML")

# 2nd Lowest AICc: 273.3521
arima101_fit <- arima(BoxCox_TotalVictims_diff1,
      order = c(1, 0, 1),
      method = "ML")

# Best/Lowest AICc: 271.0137
arima001_fit <- arima(BoxCox_TotalVictims_diff1,
      order = c(0, 0, 1),
      method = "ML")

# Table of AICc Values
print("AICc Comparison of All Models", quote = FALSE)
```

```r
AICc_arima110 <- AICc(arima110_fit)
AICc_arima011 <- AICc(arima011_fit)
AICc_arima111 <- AICc(arima111_fit)
AICc_arima100 <- AICc(arima100_fit)
AICc_arima101 <- AICc(arima101_fit)
AICc_arima001 <- AICc(arima001_fit)

Model <- c("arima001_fit", "arima101_fit", "arima100_fit",
           "arima111_fit", "arima011_fit", "arima110_fit")
AICc <- c(271.0137, 273.3521, 279.4466, 279.644, 292.1284, 307.1699)
AICc_df <- data.frame(cbind(Model, AICc),
                      row.names=c("ARIMA(0, 0, 1)", "ARIMA(1, 0, 1)",
                                  "ARIMA(1, 0, 0)", "ARIMA(1, 1, 1)",
                                  "ARIMA(0, 1, 1)", "ARIMA(1, 1, 0)"))
AICc_df

# Table of BIC Values
print("BIC Comparison of All Models", quote = FALSE)
BIC_output <- BIC(arima001_fit, arima101_fit, arima100_fit,
                  arima111_fit, arima011_fit, arima110_fit)

Model <- c("arima001_fit", "arima101_fit", "arima100_fit",
           "arima111_fit", "arima011_fit", "arima110_fit")
BIC <- c(276.1559, 280.0103, 284.5887, 284.7121, 295.6042, 310.6457)
BIC_df <- data.frame(cbind(Model, BIC),
                     row.names=c("ARIMA(0, 0, 1)", "ARIMA(1, 0, 1)",
                                 "ARIMA(1, 0, 0)", "ARIMA(1, 1, 1)",
                                 "ARIMA(0, 1, 1)", "ARIMA(1, 1, 0)"))
BIC_df

# Evaluating the residuals of ARIMA(1, 0, 0), our 3rd best fitting model,
##(according to AICc and BIC)
res_arima100_fit <- residuals(arima100_fit)
mean_res_arima100 <- mean(res_arima100_fit)
stdev_res_arima100 <- sqrt(var(res_arima100_fit))

par(mfrow=c(1, 2))

hist(res_arima100_fit, density=20, breaks=20, col="red",
     xlab="", prob=TRUE)
curve(dnorm(x, mean_res_arima100, stdev_res_arima100), add=TRUE)

qqnorm(res_arima100_fit, main= "Normal QQ Plot: ARIMA(1, 0, 0)")
qqline(res_arima100_fit, col="red")

par(mfrow=c(1, 2))
```

```r
plot.ts(res_arima100_fit)
arima100_lm <- lm(res_arima100_fit ~ as.numeric(1:length(res_arima100_fit)));
abline(arima100_lm, col="red")
abline(h=mean_res_arima100, col="black")

acf(res_arima100_fit, lag.max=50)

par(mfrow=c(1, 2))

pacf(res_arima100_fit, lag.max=50)

acf(res_arima100_fit^2, lag.max=50)

shapiro.test(res_arima100_fit)
Box.test(res_arima100_fit, lag = 10, type = c("Box-Pierce"), fitdf = 1)
Box.test(res_arima100_fit, lag = 10, type = c("Ljung-Box"), fitdf = 1)
Box.test(res_arima100_fit^2, lag = 10, type = c("Ljung-Box"), fitdf = 0)
ar(res_arima100_fit, aic = TRUE, order.max = NULL, method = c("yule-walker"))

# Evaluating the residuals of ARIMA(1, 0, 1), our 2nd best fitting model,
##(according to AICc and BIC)
res_arima101_fit <- residuals(arima101_fit)
mean_res_arima101 <- mean(res_arima101_fit)
stdev_res_arima101 <- sqrt(var(res_arima101_fit))

par(mfrow=c(1, 2))

hist(res_arima101_fit, density=20, breaks=20, col="darkgreen",
     xlab="", prob=TRUE)
curve(dnorm(x, mean_res_arima101, stdev_res_arima101), add=TRUE)

qqnorm(res_arima101_fit, main= "Normal QQ Plot: ARIMA(1, 0, 1)")
qqline(res_arima101_fit, col="darkgreen")

par(mfrow=c(1, 2))

plot.ts(res_arima101_fit)
arima101_lm <- lm(res_arima101_fit ~ as.numeric(1:length(res_arima101_fit)));
abline(arima101_lm, col="darkgreen")
abline(h=mean_res_arima101, col="black")

acf(res_arima101_fit, lag.max=50)

par(mfrow=c(1, 2))

pacf(res_arima101_fit, lag.max=50)
```

```r
acf(res_arima101_fit^2, lag.max=50)

shapiro.test(res_arima101_fit)
Box.test(res_arima101_fit, lag = 10, type = c("Box-Pierce"), fitdf = 2)
Box.test(res_arima101_fit, lag = 10, type = c("Ljung-Box"), fitdf = 2)
Box.test(res_arima101_fit^2, lag = 10, type = c("Ljung-Box"), fitdf = 0)
ar(res_arima101_fit, aic = TRUE, order.max = NULL, method = c("yule-walker"))

# Evaluating the residuals of ARIMA(0, 0, 1), our best fitting model,
##(according to AICc and BIC)
res_arima001_fit <- residuals(arima001_fit)
mean_res_arima001 <- mean(res_arima001_fit)
stdev_res_arima001 <- sqrt(var(res_arima001_fit))

par(mfrow=c(1, 2))

hist(res_arima001_fit, density=20, breaks=20, col="blue",
     xlab="", prob=TRUE)
curve(dnorm(x, mean_res_arima001, stdev_res_arima001), add=TRUE)

qqnorm(res_arima001_fit, main= "Normal QQ Plot: ARIMA(0, 0, 1)")
qqline(res_arima001_fit, col="blue")

par(mfrow=c(1, 2))

plot.ts(res_arima001_fit)
arima001_lm <- lm(res_arima001_fit ~ as.numeric(1:length(res_arima001_fit)));
abline(arima001_lm, col="blue")
abline(h=mean_res_arima001, col="black")

acf(res_arima001_fit, lag.max=50)

par(mfrow=c(1, 2))

pacf(res_arima001_fit, lag.max=50)

acf(res_arima001_fit^2, lag.max=50)

shapiro.test(res_arima001_fit)
Box.test(res_arima001_fit, lag = 10, type = c("Box-Pierce"), fitdf = 1)
Box.test(res_arima001_fit, lag = 10, type = c("Ljung-Box"), fitdf = 1)
Box.test(res_arima001_fit^2, lag = 10, type = c("Ljung-Box"), fitdf = 0)
ar(res_arima001_fit, aic = TRUE, order.max = NULL, method = c("yule-walker"))

# Unit Roots Plotting Helper Function
plot.roots <- function(ar.roots=NULL, ma.roots=NULL, size=2.5, angles=FALSE,
                       special=NULL, sqecial=NULL, main="", first.col="red",
                       second.col="darkgreen", my.pch="*") {
  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)
  }
}

# Unit Roots of ARIMA(1, 0, 0): ar1 = -0.5585
arima100_roots <- polyroot(c(1, -arima100_fit$coef))
plot.roots(ar.roots=arima100_roots, main="Unit Roots: ARIMA(1, 0, 0)")
```

```r
# Unit Roots of ARIMA(1, 0, 1): ar1 = -0.0680, ma1 = -0.8208
arima101_roots <- polyroot(c(1, -arima101_fit$coef[-2]))
plot.roots(ar.roots=arima101_roots, ma.roots=arima101_roots[2],
           main="Unit Roots: ARIMA(1, 0, 1)")

# Unit Roots of ARIMA(0, 0, 1): ma1 = -0.8818
arima001_roots <- polyroot(c(1, 0, -arima001_fit$coef))
plot.roots(ma.roots=arima001_roots, main="Unit Roots: ARIMA(0, 0, 1)")

# Forecast Box-Cox Transformed Series
arima001_BC <- arima(BoxCox_TotalVictims,
                     order = c(0, 0, 1),
                     method = "ML")
arima001_BC.forecast <- forecast::forecast(arima001_BC,
                                 level=c(50), h=6)
autoplot(arima001_BC.forecast, ts.colour='black', ts.connect=TRUE,
         predict.colour='blue', predict.geom='line', predict.linetype='dashed',
         conf.int.colour='#000000', conf.int.fill='#000000',
         xlab='Time (Years)', ylab='Forecasted Series: BoxCox_TotalVictims',
         main='Forecasted Mass Shooting Victims (1966 - 2021): Box-Cox Series')
arima001_BC.forecast_df <- as.data.frame(arima001_BC.forecast)
arima001_BC.forecast_df

# Forecast Training Set Series
arima001_train <- arima(Total_Victims_train,
                     order = c(0, 0, 1),
                     method = "ML")
arima001_train.forecast <- forecast(arima001_train,
                                 level=c(50), h=6)
autoplot(arima001_train.forecast,
         xlab="Time (Years)", ylab="Forecasted Series: Total_Victims_train",
         main="Forecasted Mass Shooting Victims (1966 - 2021): Training Set") +
  geom_point(mapping=aes(x=time(Total_Victims_test), y=Total_Victims_test))

autoplot(Total_Victims_ts,
         ts.geom="line",
         xlab="Time (Years)", ylab="Original Time Series: Total Victims",
         main="Total Annual Mass Shooting Victims in the U.S. (1966 - 2021)")
arima001_train.forecast_df <- as.data.frame(arima001_train.forecast)
arima001_train.forecast_df["Original Data"] <- c(444, 490, 691, 327, 427, 248)
arima001_train.forecast_df

# Forecast Original Series up to 2030
arima001_orig_ts <- arima(Total_Victims_ts,
                     order = c(0, 0, 1),
                     method = "ML")
arima001_orig_ts.forecast <- forecast::forecast(arima001_orig_ts,
                                  level=c(50), h=9)
autoplot(arima001_orig_ts.forecast,
         xlab="Time (Years)", ylab="Forecasted Series: Total_Victims_ts",
         main="Forecasted Mass Shooting Victims in the U.S. (1966 - 2030)")
arima001_orig_ts.forecast_df <- as.data.frame(arima001_orig_ts.forecast)
arima001_orig_ts.forecast_df
```