A Time Series Analysis of General Motors Company Monthly Sales in the US Market

Sara Mezuri

2023-04-24

Introduction

The American automotive industry is known to be one of the largest and most competitive markets globally, where General Motors Company is among the leading players who have been competing against each other for decades. The aim of this project is to perform a time series analysis of the General Motors Company monthly sales in the US market, for the past 2 decades which will provide insights into their market position, allowing us to uncover patterns, trends, and fluctuations within sequential data, enabling a deeper understanding of temporal relationships. It allows us to make predictions based on historical patterns, facilitating informed decision-making and strategic planning. This analytical approach is particularly valuable in forecasting future trends, identifying anomalies, and extracting meaningful insights from time-ordered datasets.

Objectives

The main objectives of this project are:

1. To analyze the sales performance of General Motors in the US market over the last two decades.
2. To identify trends, patterns and seasonality in the sales data of the company.
3. To recognize any of the factors that may have contributed to the increase or decrease in sales performance.
4. To fit a model on the dataset and make predictions for forthcoming data.

Data Collection and Processing

The data used for this project is the monthly sales data for the General Motors Company in the US market.

General Motors Sales Data (click here for the data)

```
# Read in the data

setwd("C:/Users/saram/Desktop/OU/Winter 2023/STA 5330 (Time series I)/Project")

mydata <- read.csv("SALES DATA.csv")

#Convert the data
mydata$Date<-as.Date(mydata$Date, format="%m/%d/%Y")
```

```r
# Print the first 6 rows of the data frame
head(mydata)
```

```
##         Date   Ford     GM
## 1 2005-01-01 174968 213931
## 2 2005-02-01 156003 190165
## 3 2005-03-01 145070 198088
## 4 2005-04-01 177840 215534
## 5 2005-05-01 144984 198960
## 6 2005-06-01 156917 210246
```

```r
# Print the last 6 rows of the data frame
tail(mydata)
```

```
##           Date   Ford     GM
## 214 2022-10-01 265698 533652
## 215 2022-11-01 263949 372586
## 216 2022-12-01 260741 368088
## 217 2023-01-01 281906 409418
## 218 2023-02-01 234208 296364
## 219 2023-03-01 183379 269125
```

```r
# Check the structure of the data
str(mydata)
```

```
## 'data.frame':    219 obs. of  3 variables:
##  $ Date: Date, format: "2005-01-01" "2005-02-01" ...
##  $ Ford: int  174968 156003 145070 177840 144984 156917 141633 157062 162819 151182 ...
##  $ GM  : int  213931 190165 198088 215534 198960 210246 182393 189370 182357 190371 ...
```

```r
# Check the summary of the data
summary(mydata)
```

```
##       Date                 Ford             GM
##  Min.   :2005-01-01   Min.   : 77815   Min.   : 88377
##  1st Qu.:2009-07-16   1st Qu.:164415   1st Qu.:200372
##  Median :2014-02-01   Median :188882   Median :236314
##  Mean   :2014-01-30   Mean   :190394   Mean   :240467
##  3rd Qu.:2018-08-16   3rd Qu.:216594   3rd Qu.:266241
##  Max.   :2023-03-01   Max.   :352164   Max.   :533652
```

To evaluate our model's predictions and compare them with actual values, we temporarily exclude the most recent 24 months. Subsequently, we will add them later on.

```r
# Length of the GM Sales Data column

n <- length(mydata$GM)
print(n)
```

```
## [1] 219
```

```r
# Create the data without the last 24 months

GM_Short <- mydata$GM[1:(n-24)]
head(GM_Short)
```

```
## [1] 213931 190165 198088 215534 198960 210246
```

```r
tail(GM_Short)
```

```
## [1] 307784 361590 299061 333899 300624 237351
```

```r
str(GM_Short)
```

```
##  int [1:195] 213931 190165 198088 215534 198960 210246 182393 189370 182357 190371 ...
```

```r
Date_Short <- mydata$Date[1:(n-24)]
head(Date_Short)
```

```
## [1] "2005-01-01" "2005-02-01" "2005-03-01" "2005-04-01" "2005-05-01"
## [6] "2005-06-01"
```

```r
tail(Date_Short)
```

```
## [1] "2020-10-01" "2020-11-01" "2020-12-01" "2021-01-01" "2021-02-01"
## [6] "2021-03-01"
```

```r
str(Date_Short)
```

```
##  Date[1:195], format: "2005-01-01" "2005-02-01" "2005-03-01" "2005-04-01" "2005-05-01" ...
```

```r
# Create the data containing only the last 24 months

GM_Predict <- mydata$GM[(n-23):n]
head(GM_Predict)
```

```
## [1] 324413 285560 290221 325729 352411 395262
```

```r
tail(GM_Predict)
```

```
## [1] 533652 372586 368088 409418 296364 269125
```

```r
str(GM_Predict)
```

```
##  int [1:24] 324413 285560 290221 325729 352411 395262 395217 329119 331474 348968 ...
```

```r
Date_Predict <- mydata$Date[(n-23):n]
head(Date_Predict)
```

```
## [1] "2021-04-01" "2021-05-01" "2021-06-01" "2021-07-01" "2021-08-01"
## [6] "2021-09-01"
```

```r
tail(Date_Predict)
```

```
## [1] "2022-10-01" "2022-11-01" "2022-12-01" "2023-01-01" "2023-02-01"
## [6] "2023-03-01"
```

```r
str(Date_Predict)
```

```
##  Date[1:24], format: "2021-04-01" "2021-05-01" "2021-06-01" "2021-07-01" "2021-08-01" ...
```

Exploratory Data Analysis

Trend, Seasonality and Stationarity Analysis

```r
# Create the sales plot

gm <- ts(mydata$GM) # create a time series object for GM Monthly Sales

plot(mydata$Date, gm,
     type = "o", xlab = "Date", ylab = "Monthly Sales", main = "Original General Motors Time Series")
```

```r
# Create the plot containing data from January 2005 to December 2020

gm_ts <- ts(GM_Short) # create a time series object for GM Shorten Monthly Sales

plot(Date_Short, gm_ts,
     type = "o", xlab = "Date", ylab = "Shorten Monthly Sales", main = "General Motors Shorten Time Seri
```

There is a change in trend and seasonality to be seen from the plot (fig. 2). As a result, the time series of sales do not appear to be stationary. A stationary time series is one whose statistical properties, such as the mean and variance, do not change over time, while a non-stationary time series has statistical properties that are time-dependent and can vary over time.

Furthermore, we can proceed with the ADF test.

```r
library(tseries)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

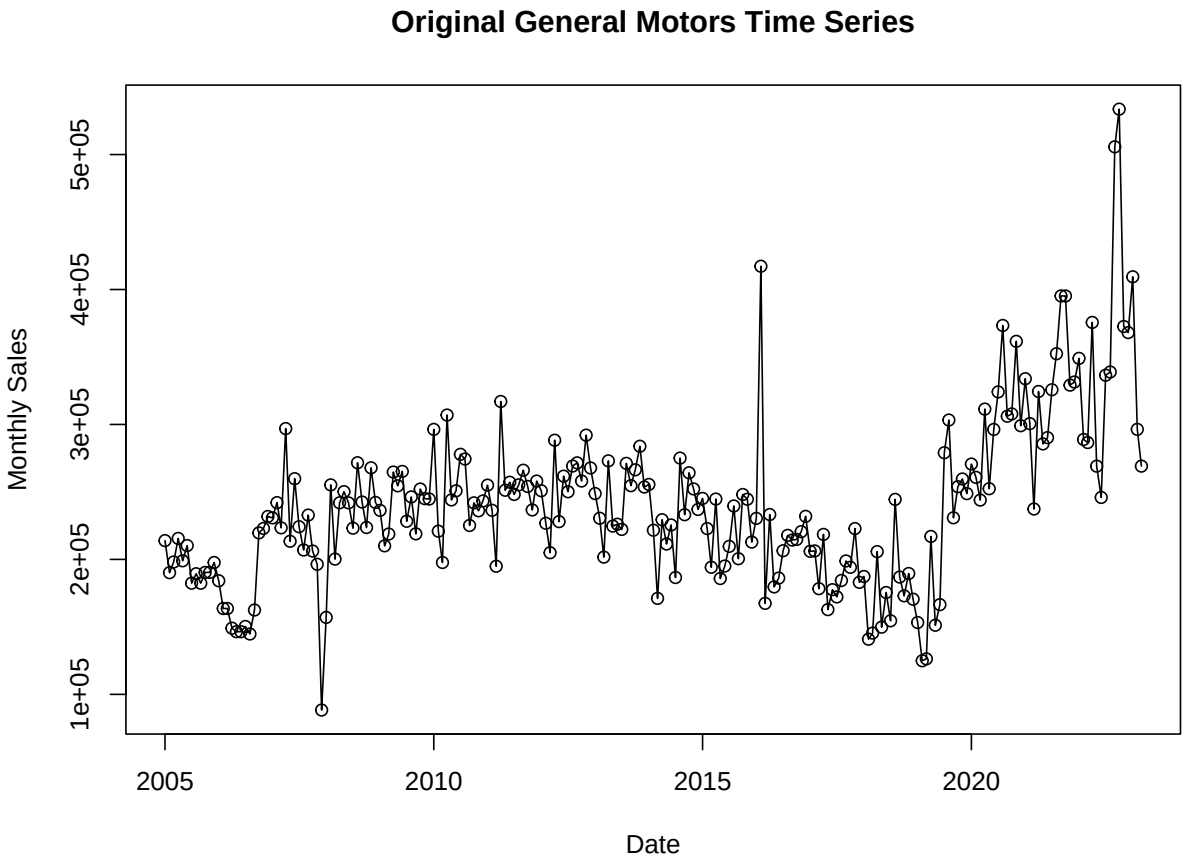


Figure 1: Original Monthly Sales Plot since 2005

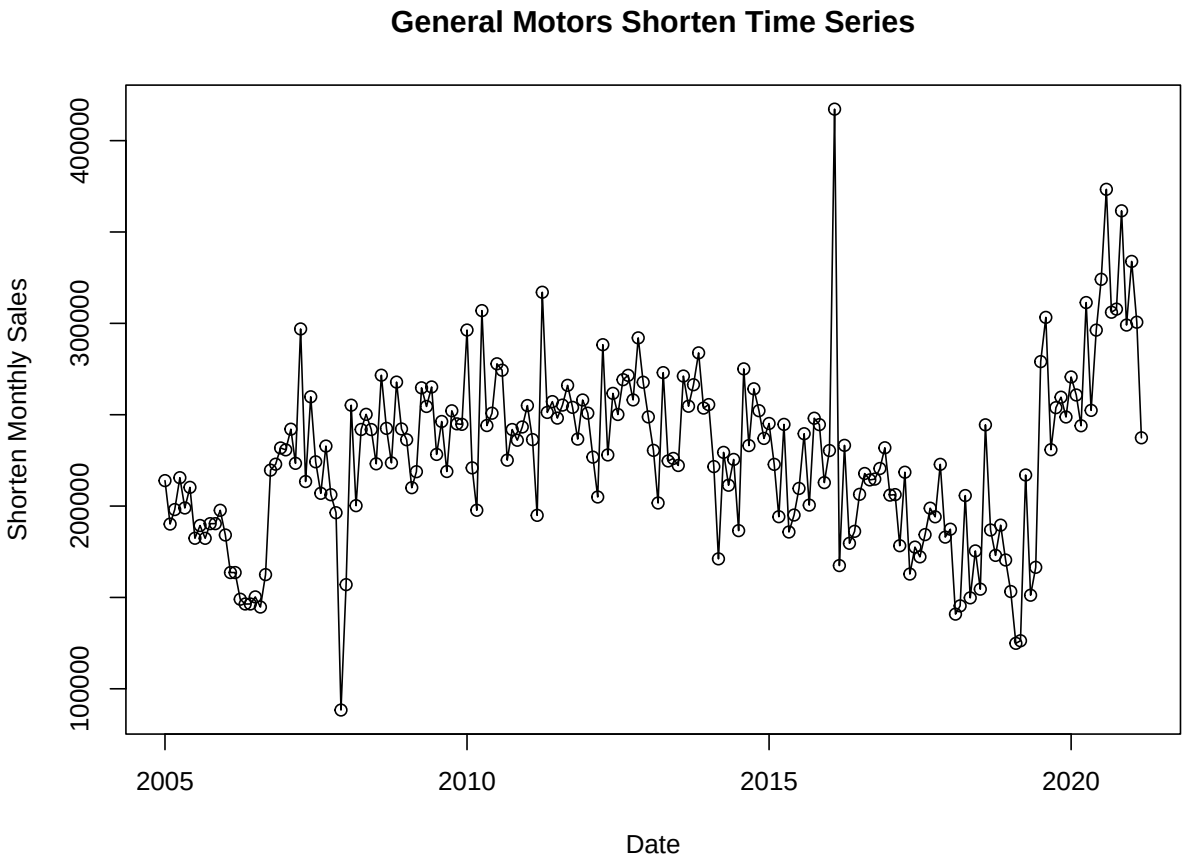


Figure 2: Shorten Monthly Sales Plot

```r
# Perform the Augmented Dickey-Fuller test

test <- adf.test(gm_ts)

# Print results

print(test)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  gm_ts
## Dickey-Fuller = -3.1198, Lag order = 5, p-value = 0.1074
## alternative hypothesis: stationary
```

```r
cat("p-value: ", test$p.value, "\n")
```

```
## p-value:  0.1073877
```

Clearly, from the ADF test result, the p-value for the time series is significantly greater than the significance level $\alpha = 0.05$, thus we fail to reject the null hypothesis.

Plotting the ACF and PACF graphs can also help understanding the time serie better. In time series analysis, ACF (Autocorrelation Function) and PACF (Partial Autocorrelation Function) plots are often used to determine the order of Autoregressive (AR) and Moving Average (MA) models.

The ACF plot displays the relationship between the values of a time series and its lagged values. The ACF plot can be used to identify the order of a MA model by observing the number of lags with significant autocorrelation.

The PACF plot displays the relationship between a time series and its lag values. The PACF plot can be used to identify the order of an AR model by observing the number of lags with significant partial autocorrelation.

```r
# Plot the acf and pacf of GM

par(mfrow = c(1,2))

acf(gm_ts, lag.max = 48)

pacf(gm_ts, lag.max = 48)
```

Transforming the Data

Next step is taking the logarithm of the time series. This is a common data transformation technique used in time series analysis. In our data set, we notice a variability in the data overtime, which makes it difficult to forecast the series accurately. Taking the logarithm of the series can help to stabilize the variance, by compressing the range of values for large observations and expanding the range for small observations.

```r
# Take the log of the time series

gm_ts_log <- log(gm_ts)
```

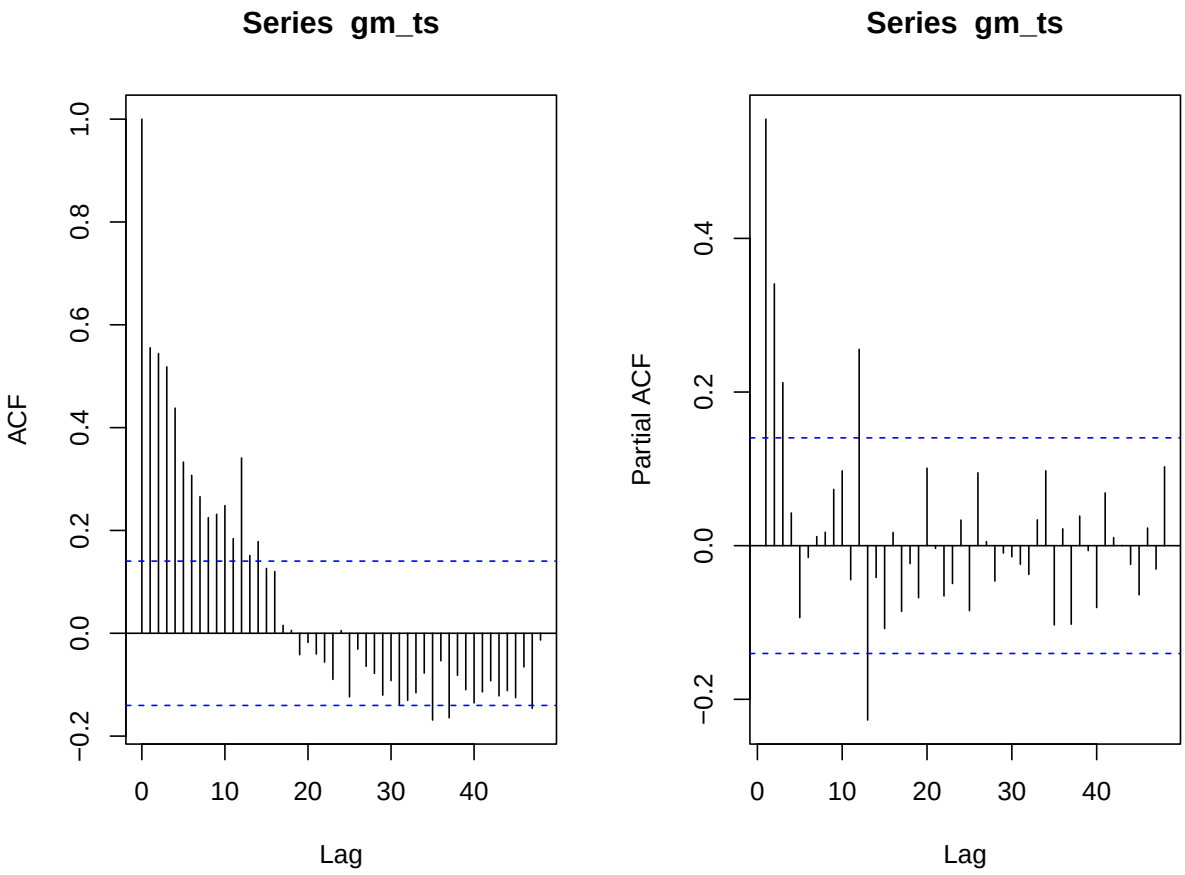


Figure 3: ACF and PACF graphs

```r
# Plot the logged time series

plot(Date_Short, gm_ts_log,
     type = "l", xlab = "Date", ylab = "Monthly Sales", main = "General Motors Logged Time Series")
```

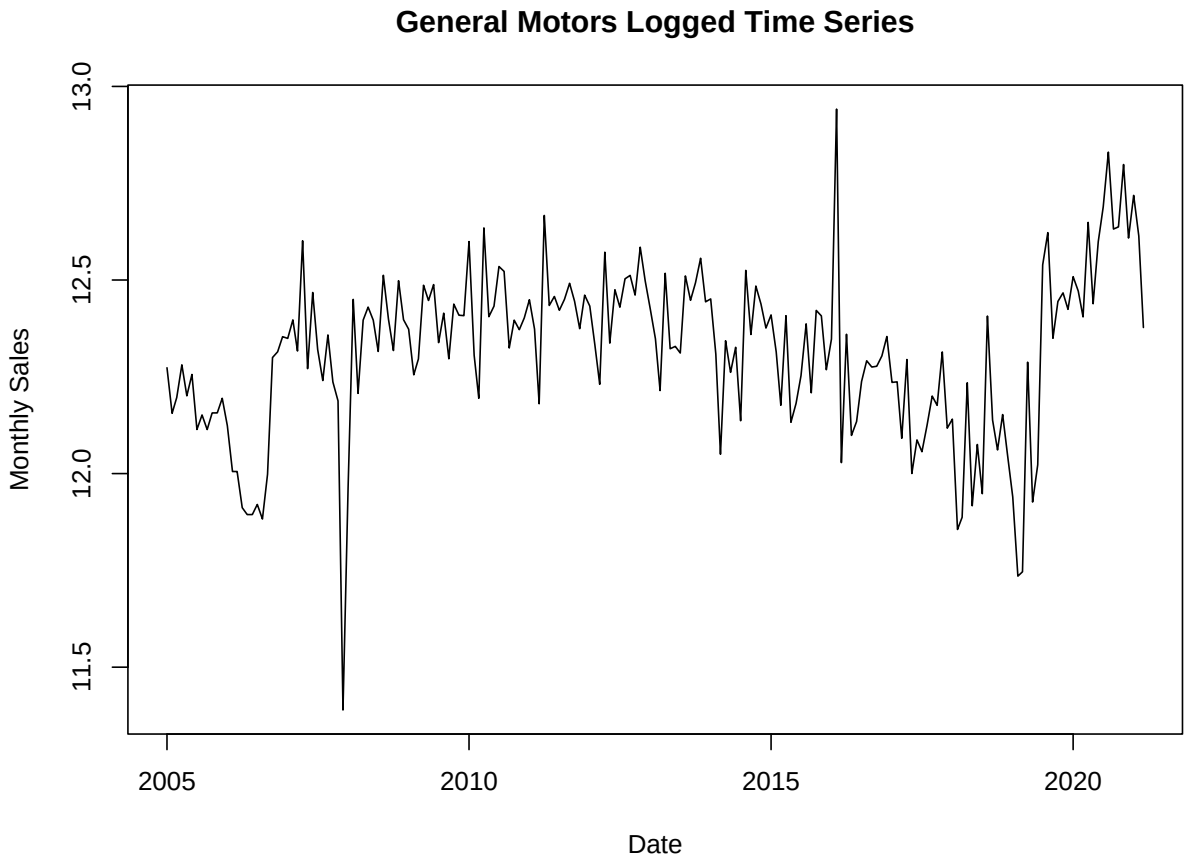


Figure 4: Monthly Sales Logged Time Series

Transformations such as logarithms can help to stabilize the variance of a time series. But when it comes to the mean of time series, differencing can help stabilize the mean of a time series by removing changes in the level of a time series, and therefore eliminating (or reducing) trend and seasonality.

The differenced series is the change between consecutive observations in the original series, and can be written as

$$X_t' = X_{t+1} - X_t$$

where X_t is a time series.

```r
# Take the first-order difference of the logged time series

gm_diff1 <- diff(gm_ts_log, lag = 1)
```

```r
# Plot the first-difference time series

plot(gm_diff1 , type = 'l',
     xlab = "time", ylab = "first-order difference", main = "GM Differenced Time Series")
```

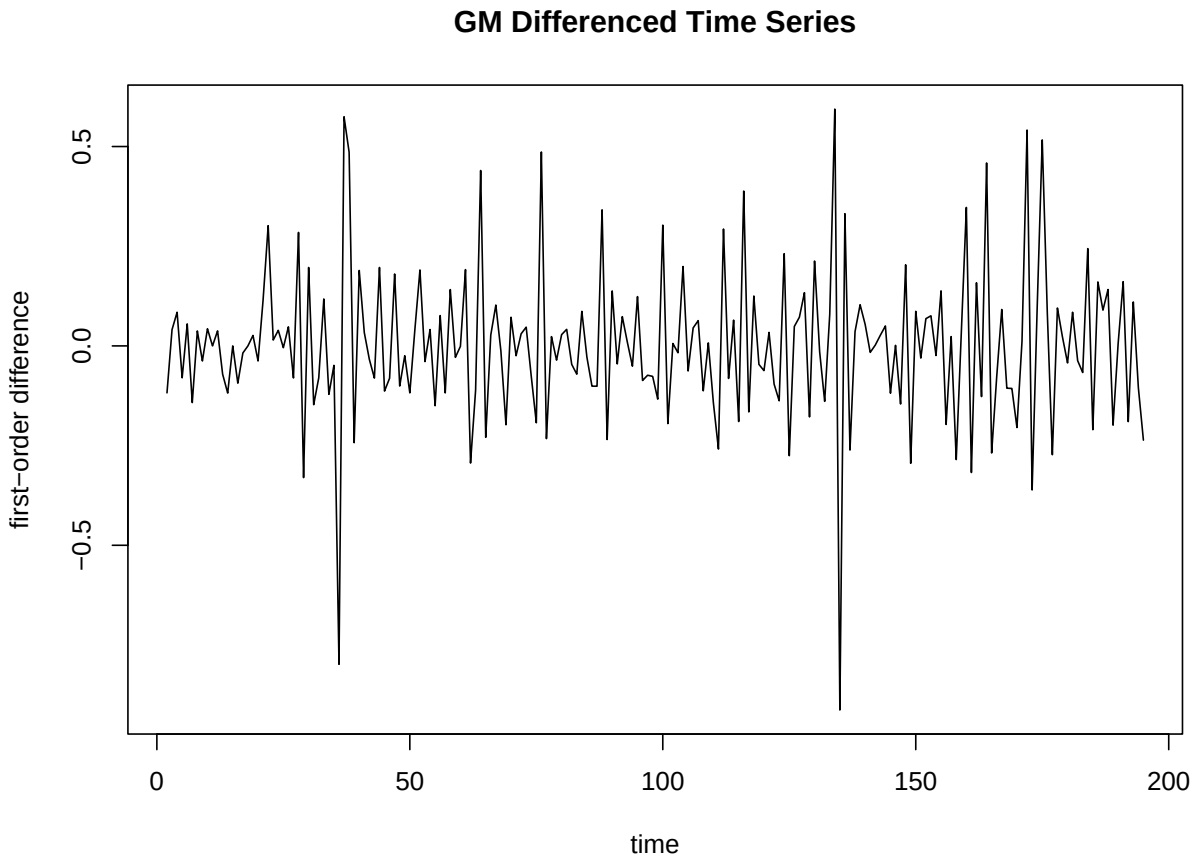


Figure 5: First-Order Differenced Time Series

```r
# Plot the ACF and PACF of the first-difference time series

par(mfrow = c(1, 2))

acf(gm_diff1, lag.max = 48)

pacf(gm_diff1, lag.max = 48)
```

At times, the differenced data might not appear to be stationary, and it might be required to perform a second differencing to achieve a stationary series:

$$X_t'' = X_{t+1}' - X_t'.$$

In our case, that step is not neccesary.

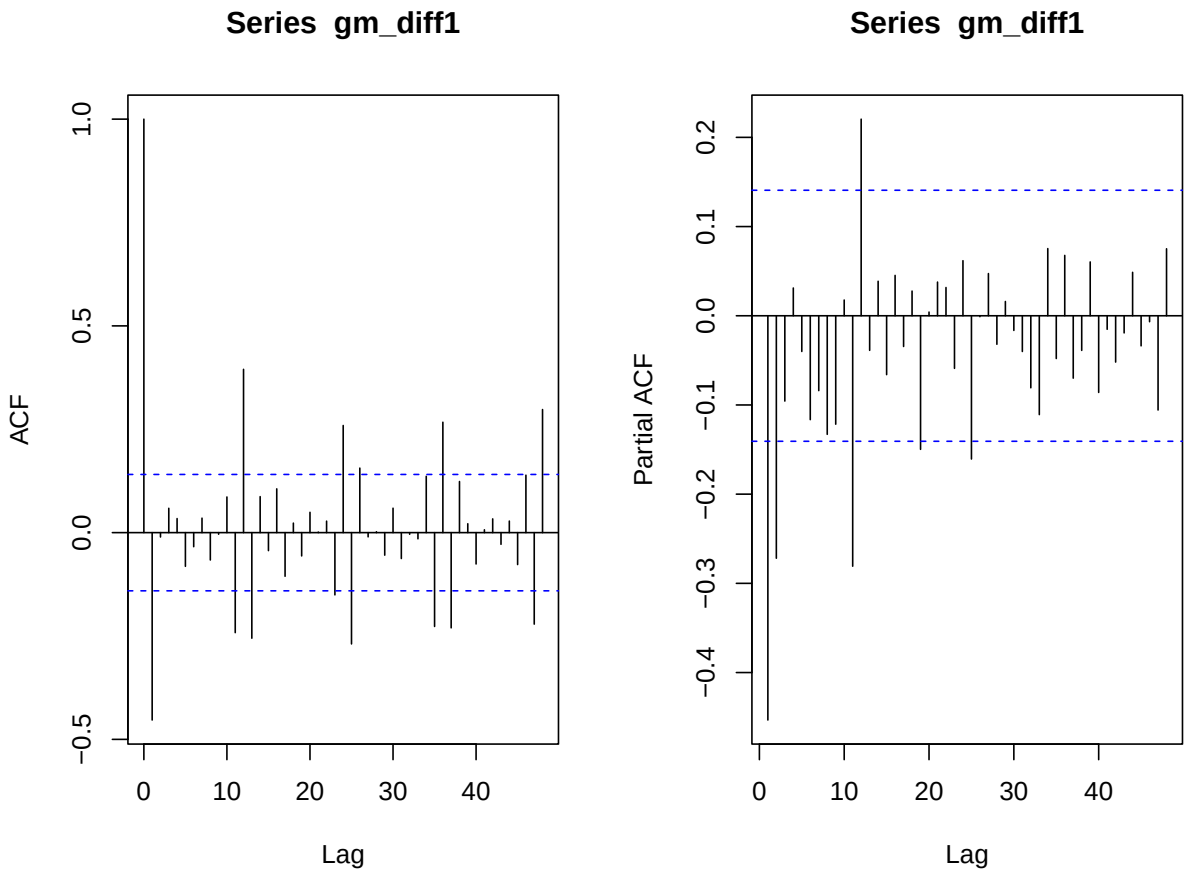


Figure 6: ACF and PACF of First-Order Differenced Time Series

Furthermore, in the case of a stationary time series at $lag = h$, it might be required to perform another differencing at $lag = h$. Seasonal differencing is defined as a difference between a value and a value with lag that is a multiple of S.

Seasonality in time series refers to the pattern of regular and predictable fluctuations that occur over fixed intervals of time, as days, weeks, months, or years.

In our case, the PACF shows a spike at lag 12, which may suggest seasonality and the ACF shows a strong correlation at lags 12,24,36 (fig.3 and fig.6).

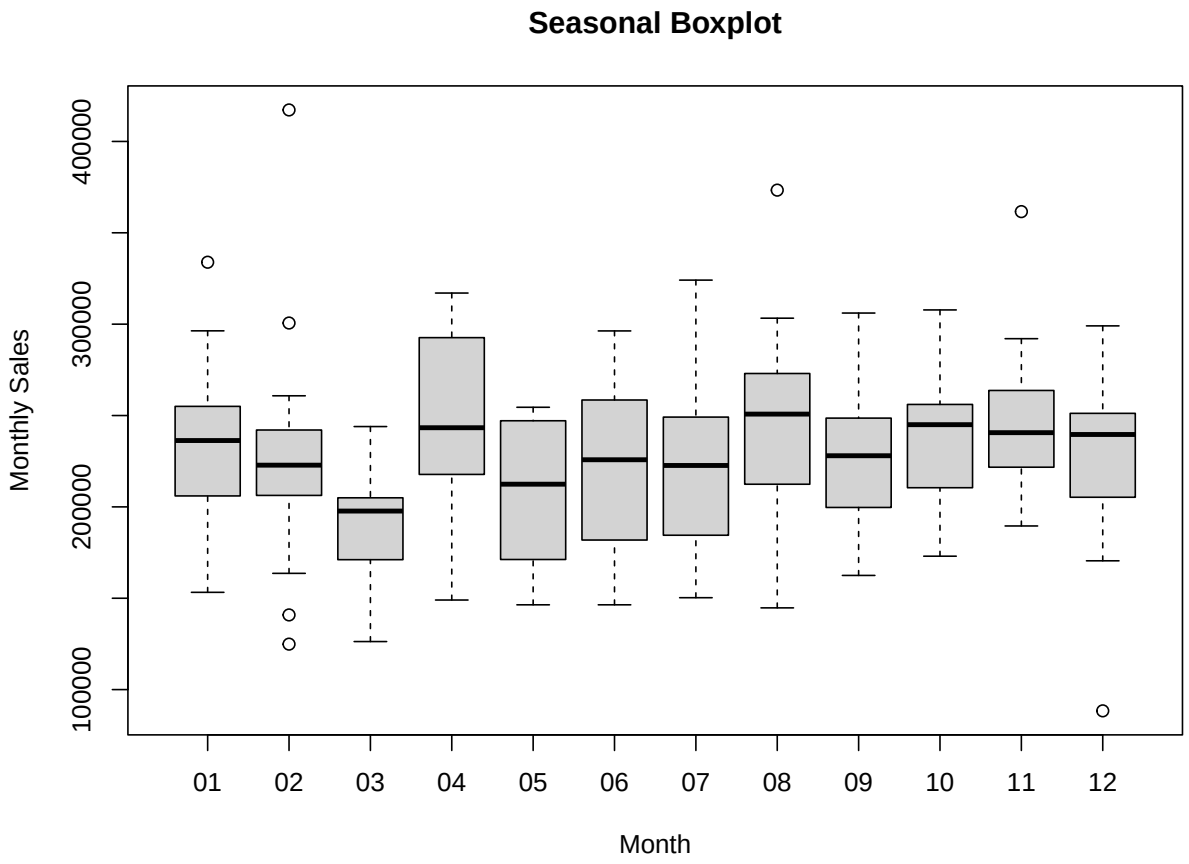
If a time series is seasonal at lag = 12, it means that there is a repeating pattern or regular fluctuation in the data that occurs every 12 time units. In the context of monthly data, a seasonal pattern at lag = 12 indicates a yearly seasonality, where the same or similar pattern tends to repeat every 12 months. Therefore, a seasonal pattern at lag = 12 suggests that the time series exhibits a recurring behavior on an annual basis. Understanding seasonality at lag = 12 allows to account for these recurring patterns when developing forecasting models.

Let's look at a seasonal sales boxplot to understand better the seasonality.

```r
# Create a data frame with date and sales
df <- data.frame(Date = Date_Short, Sales = gm_ts)

# Extract month and year from the date
df$Month <- format(df$Date, "%m")
df$Year <- format(df$Date, "%Y")

# Create a boxplot to visualize seasonality
boxplot(Sales ~ Month, data = df, xlab = "Month", ylab = "Monthly Sales", main = "Seasonal Boxplot")
```



This graph shows a visual of any recurring seasonal patterns or fluctuations. For example, the sales seem to be low during March, but they increase in April, then slightly decrease until August, when they increase back, and so on.

If $S = 12$, which occurs with monthly data, a seasonal difference is $(1 - B^{12})X_t = X_t - X_{t-12}$. Seasonal differencing removes seasonal trend and can also get rid of a seasonal random walk type of non-stationarity.

```
# Take the difference at lag 12 of the time series

gm_diff2 <- diff(gm_diff1, lag = 12)

# Plot the first-difference time series

plot(gm_diff2 , type = 'l', xlab = "time", ylab = "difference at lag 12",
     main = "GM Differenced Time Series at lag 12")
```

```
# Plot the ACF and PACF of the difference time series at lag = 12

par(mfrow = c(1,2))

acf(gm_diff2, lag.max = 48)

pacf(gm_diff2, lag.max = 48)
```

We run the ADF test again to make sure that now we have a stationary series.

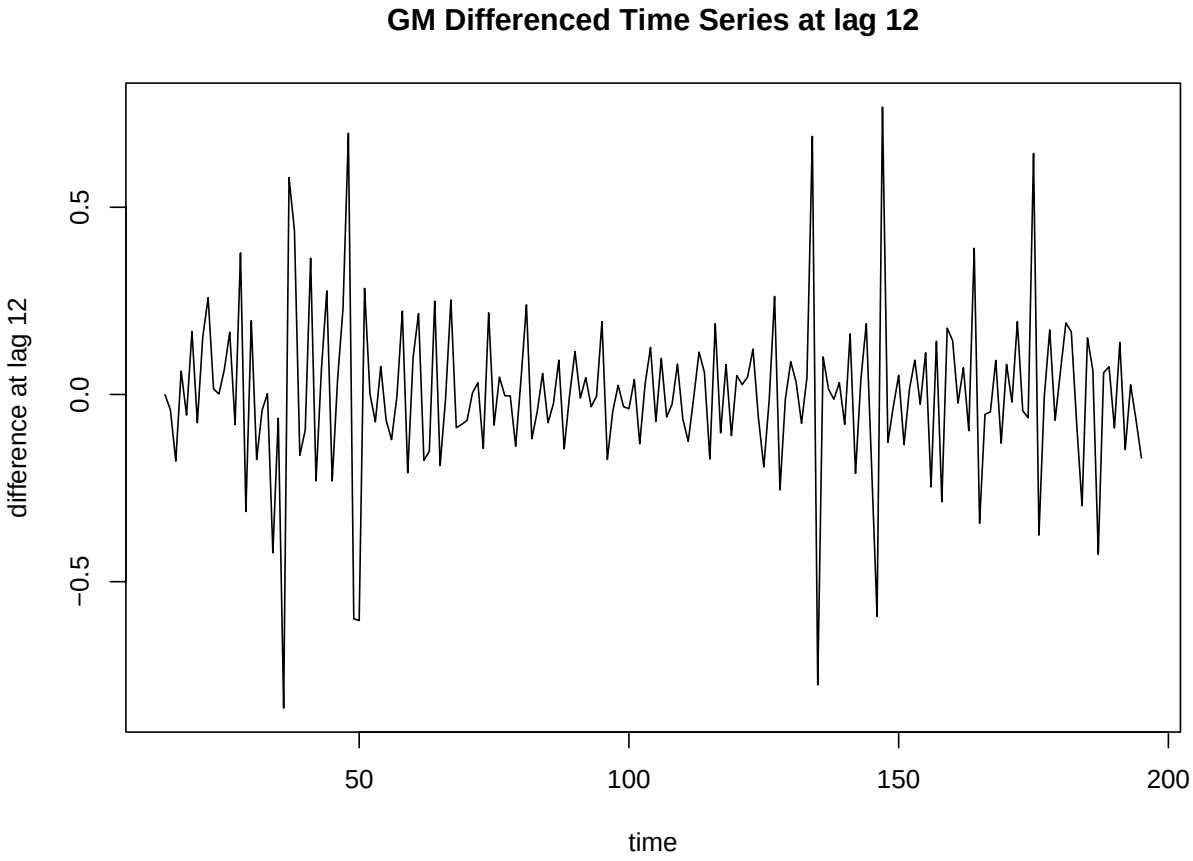


Figure 7: Differenced Time Series at lag 12

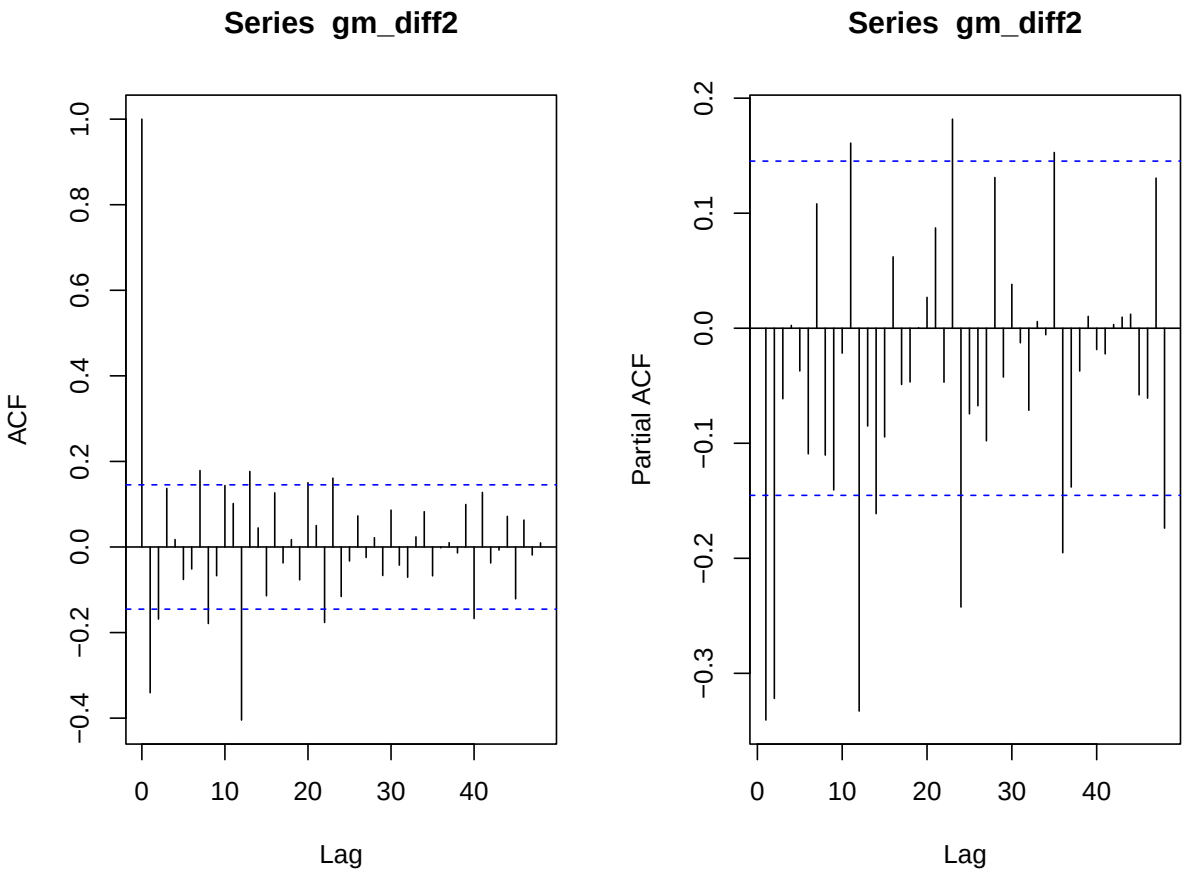


Figure 8: ACF and PACF of Differenced Time Series at lag = 12

```r
library(tseries)

# Perform the Augmented Dickey-Fuller test

test_1 <- adf.test(gm_diff2)

# Print results

print(test_1)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  gm_diff2
## Dickey-Fuller = -6.9659, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```r
cat("p-value: ", test_1$p.value, "\n")
```

```
## p-value:  0.01
```

As we can tell $p = 0.01 < 0.05$, thus we reject the null hypothesis, i.e the time series is now stationary.

Model Specification

Rationale for choosing the model

Clearly, the time series represent a SARIMA $(p, d, q)x(P, D, Q)$ model. A SARIMA model is an ARIMA model with a seasonal component, where

- p and seasonal P indicate the autoregressive order.
- d and seasonal D indicate differencing that must be done to stationarize series.
- q and Q indicate the moving average order.

By definition, if d and D are non-negative integers, then $\{X_t\}$ is a seasonal ARIMA $(p, d, q)x(P, D, Q)s$ process with period s if the differenced series $Y_t = (1 - B)^d(1 - B^s)^D X_t$ is a causal ARMA process defined by

$$\phi(B)\Phi(B^s)Y_t = \theta(B)\Theta(B^s)Z_t, Zt \sim WN(0, \sigma^2)$$

where the non-seasonal components are

- AR: $\phi(z) = 1 - \phi_1 z - \ldots - \phi_p z^p$
- MA: $\theta(z) = 1 + \theta_1 z + \ldots + \theta_q z^q$

and the seasonal components are

- seasonal AR: $\Phi(z) = 1 - \Phi_1 z - \ldots - \Phi_p z^p$

- seasonal MA: $\Theta(z) = 1 + \Theta_1 z + \ldots + \Theta_q z^q$.

For our both time series, $D = 1$ and $d = 1$. To determine p, q, P and Q, we can refer to the ACF and PACF plots.

From the ACF and PACF plots, for the sales time series we notice a significant spike on the ACF plot before the first seasonal $lag = 12$ and three noticeable lags on the PACF plot at lags $12, 24, 36$. This may suggest that $P = 3$ and $Q = 1$.

However, in this case, basing my analysis only on the ACF and PACF plots,may result are in uncertain and unsufficient conclusions.

To determine the best p, P, q and Q, we use the following code

```
# Load the necessary packages
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.2.3
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
# Choosing the best model for GM Time Series

# Set values for d, D, and s
d <- 1
D <- 1
s <- 12

# Initialize variables to store minimum AIC and corresponding parameters
min_aic_gm <- Inf
best_params_gm <- c()

# Loop over values of p, q, P, and Q
for (p in 1:3) {
  for (q in 1:3) {
    for (P in 1:3) {
      for (Q in 1:3) {
        # Check if the sum of parameters is less than or equal to 10
        if (p + d + q + P + D + Q <= 10) {
          # Fit ARIMA model
          model_gm <- try(arima(log(gm_ts), order=c(p-1, d, q-1),
                                seasonal=list(order=c(P-1, D, Q-1),period=s)), silent = TRUE)
          # Check for errors in model fitting
          if (class(model_gm) == "try-error") {
            next
          }
          # Perform Ljung-Box test on residuals of fitted model
          test <- Box.test(model_gm$residuals, lag=log(length(model_gm$residuals)))
          # Calculate SSE of fitted model
          sse <- sum(model_gm$residuals^2)
          # Update minimum AIC and corresponding parameters if a better model is found
          if (model_gm$aic < min_aic_gm) {
```

```
            min_aic_gm <- model_gm$aic
            best_params_gm <- c(p-1, d, q-1, P-1, D, Q-1)
          }
          # Print AIC, SSE, and p-value of fitted model
          cat(p-1, d, q-1, P-1, D, Q-1, "AIC:", model_gm$aic, "SSE:", sse,
              "p-value:", test$p.value, "\n")
        }
      }
    }
  }
}
```

```
## 0 1 0 0 1 0 AIC: -36.5376 SSE: 8.624564 p-value: 5.03415e-06
## 0 1 0 0 1 1 AIC: -116.0022 SSE: 5.020715 p-value: 5.729521e-05
## 0 1 0 0 1 2 AIC: -115.4274 SSE: 4.865225 p-value: 3.807871e-05
## 0 1 0 1 1 0 AIC: -67.21705 SSE: 7.123069 p-value: 2.566065e-05
## 0 1 0 1 1 1 AIC: -115.034 SSE: 4.910817 p-value: 4.361298e-05
## 0 1 0 1 1 2 AIC: -114.3673 SSE: 4.885259 p-value: 3.192667e-05
## 0 1 0 2 1 0 AIC: -90.01579 SSE: 6.07934 p-value: 1.720706e-05
## 0 1 0 2 1 1 AIC: -116.4787 SSE: 4.905773 p-value: 2.216956e-05
## 0 1 0 2 1 2 AIC: -116.0276 SSE: 4.835726 p-value: 3.975161e-05
## 0 1 1 0 1 0 AIC: -73.46286 SSE: 6.951662 p-value: 0.3407345
## 0 1 1 0 1 1 AIC: -147.4136 SSE: 4.222169 p-value: 0.2948956
## 0 1 1 0 1 2 AIC: -147.3999 SSE: 4.108932 p-value: 0.2796996
## 0 1 1 1 1 0 AIC: -104.6396 SSE: 5.721588 p-value: 0.5724313
## 0 1 1 1 1 1 AIC: -146.9892 SSE: 4.127919 p-value: 0.2947198
## 0 1 1 1 1 2 AIC: -145.9324 SSE: 4.12024 p-value: 0.2425352
## 0 1 1 2 1 0 AIC: -125.081 SSE: 4.951473 p-value: 0.2635543
## 0 1 1 2 1 1 AIC: -147.1641 SSE: 4.127417 p-value: 0.1837136
## 0 1 2 0 1 0 AIC: -73.0854 SSE: 6.888575 p-value: 0.506225
## 0 1 2 0 1 1 AIC: -148.6715 SSE: 4.130425 p-value: 0.6984616
## 0 1 2 0 1 2 AIC: -148.3829 SSE: 4.02042 p-value: 0.6430829
## 0 1 2 1 1 0 AIC: -105.1794 SSE: 5.636543 p-value: 0.8294886
## 0 1 2 1 1 1 AIC: -148.0138 SSE: 4.041096 p-value: 0.6654043
## 0 1 2 2 1 0 AIC: -125.8063 SSE: 4.873773 p-value: 0.6188385
## 1 1 0 0 1 0 AIC: -56.93349 SSE: 7.621065 p-value: 0.0002527716
## 1 1 0 0 1 1 AIC: -133.2552 SSE: 4.5519 p-value: 0.0009339387
## 1 1 0 0 1 2 AIC: -132.9115 SSE: 4.437536 p-value: 0.0006441894
## 1 1 0 1 1 0 AIC: -88.25902 SSE: 6.269592 p-value: 0.002197579
## 1 1 0 1 1 1 AIC: -132.5464 SSE: 4.460074 p-value: 0.0007499509
## 1 1 0 1 1 2 AIC: -131.4502 SSE: 4.448845 p-value: 0.0004719753
## 1 1 0 2 1 0 AIC: -108.5681 SSE: 5.433454 p-value: 0.0002376293
## 1 1 0 2 1 1 AIC: -132.8712 SSE: 4.454545 p-value: 0.0002768135
## 1 1 1 0 1 0 AIC: -72.57129 SSE: 6.90815 p-value: 0.3976194
## 1 1 1 0 1 1 AIC: -148.5091 SSE: 4.127658 p-value: 0.5027411
## 1 1 1 0 1 2 AIC: -148.08 SSE: 4.023038 p-value: 0.4440494
## 1 1 1 1 1 0 AIC: -105.2838 SSE: 5.627878 p-value: 0.6479687
## 1 1 1 1 1 1 AIC: -147.7564 SSE: 4.041247 p-value: 0.4659481
## 1 1 2 0 1 0 AIC: -79.30037 SSE: 6.560016 p-value: 0.6160368
## 1 1 2 0 1 1 AIC: -146.9158 SSE: 4.135462 p-value: 0.804575
## 1 1 2 1 1 0 AIC: -106.9721 SSE: 5.513624 p-value: 0.8057709
## 2 1 0 0 1 0 AIC: -74.86694 SSE: 6.822464 p-value: 0.9743984
## 2 1 0 0 1 1 AIC: -147.1076 SSE: 4.184608 p-value: 0.8232415
```

```
## 2 1 0 0 1 2 AIC: -147.6816 SSE: 4.04414 p-value: 0.8632313
## 2 1 0 1 1 0 AIC: -102.9894 SSE: 5.719341 p-value: 0.8360974
## 2 1 0 1 1 1 AIC: -147.0431 SSE: 4.07391 p-value: 0.8569774
## 2 1 0 2 1 0 AIC: -125.5191 SSE: 4.88221 p-value: 0.8824884
## 2 1 1 0 1 0 AIC: -73.62991 SSE: 6.793407 p-value: 0.9941215
## 2 1 1 0 1 1 AIC: -146.5947 SSE: 4.144076 p-value: 0.8729816
## 2 1 1 1 1 0 AIC: -103.9641 SSE: 5.602845 p-value: 0.4399763
## 2 1 2 0 1 0 AIC: -71.66004 SSE: 6.792209 p-value: 0.9953421
```

```
# Print the best parameters and minimum AIC
cat("Best Parameters: p, d, q, P, D, Q =", best_params_gm, "\n")
```

```
## Best Parameters: p, d, q, P, D, Q = 0 1 2 0 1 1
```

```
cat("Minimum AIC =", min_aic_gm, "\n")
```

```
## Minimum AIC = -148.6715
```

The output of the code includes the AIC, SSE and the p-value of each fitted model. The AIC (Akaike Information Criterion) is calculated as

$$AIC = 2k - 2ln(L),$$

where k is the number of parameters in the model, and L is the likelihood function of the model. The AIC is a measure of the quality of a statistical model, with lower values indicating better fit. The SSE is the sum of squared errors of the fitted model, which measures how well the model fits the data.

The Ljung-Box test is a statistical test for autocorrelation in a time series. The test is used to assess whether the residuals (i.e., the differences between the observed values and the values predicted by a model) of a time series model are independently distributed. Therefore, the null hypothesis of the Ljung-Box test is that the residuals of the fitted model are independently distributed.

Regarding the p-value of the Ljung-Box test, if the p-value of the Ljung-Box test is small (less than 0.05), this provides evidence against the null hypothesis. Hence, a small p-value suggests that the model may not be a good fit for the data.

Conversely, if the p-value of the Ljung-Box test is large (greater than 0.05), this suggests that there is no evidence against the null hypothesis, and that the residuals are independently distributed. In this case, the model is likely a good fit for the data.

From the results of the code above, we choose as the best fitting model for the time series to be $(0,1,2)x(0,1,1)$ with the smallest $AIC = -148.6715$ and $p - value = 0.6984616$ from Ljung-Box test.

Model Fitting

Parameter estimation

For a SARIMA$(0,1,2)x(0,1,1)_{12}$, the equation would be

$$(1-B)^1(1+B^{12})^1 X_t = (1+\theta_1 B+\theta_2 B^2)(1+\Theta_1 B^{12})Z_t$$

where $Z_t \sim (0,\sigma^2)$.

```r
# We fit a SARIMA(0,1,2)(0,1,1) model to GM time series

gm.fit <- arima(gm_diff2, order = c(0, 1, 2),seasonal = list(order = c(0, 1, 1),period = 12)
                ,method = "ML")

print(gm.fit)
```

```
##
## Call:
## arima(x = gm_diff2, order = c(0, 1, 2), seasonal = list(order = c(0, 1, 1),
##     period = 12), method = "ML")
##
## Coefficients:
##           ma1     ma2     sma1
##       -1.5205  0.5205  -1.0000
## s.e.   0.0764  0.0726   0.0543
##
## sigma^2 estimated as 0.04101:  log likelihood = 8.83,  aic = -9.66
```

Therefore, for the GM model, the equation would be

$$(1-B)^1(1+B^{12})^1 X_t = (1-1.5205B+0.5205B^2)(1-1.0000B^{12})Z_t$$

where

$$Z_t \sim (0, 0.04101)$$

Model Diagnostics

Residual Analysis

As mentioned above, residuals in time series are what is left over after fitting a model. For most of the time series models, residuals are equal to difference between the observation and corresponding fitted values. To make sure that the models we have choosen are a good fit, we need to make sure that the residuals are white noise.

```r
library(TSA)
```

```
## Warning: package 'TSA' was built under R version 4.2.3
```

```
## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast
```

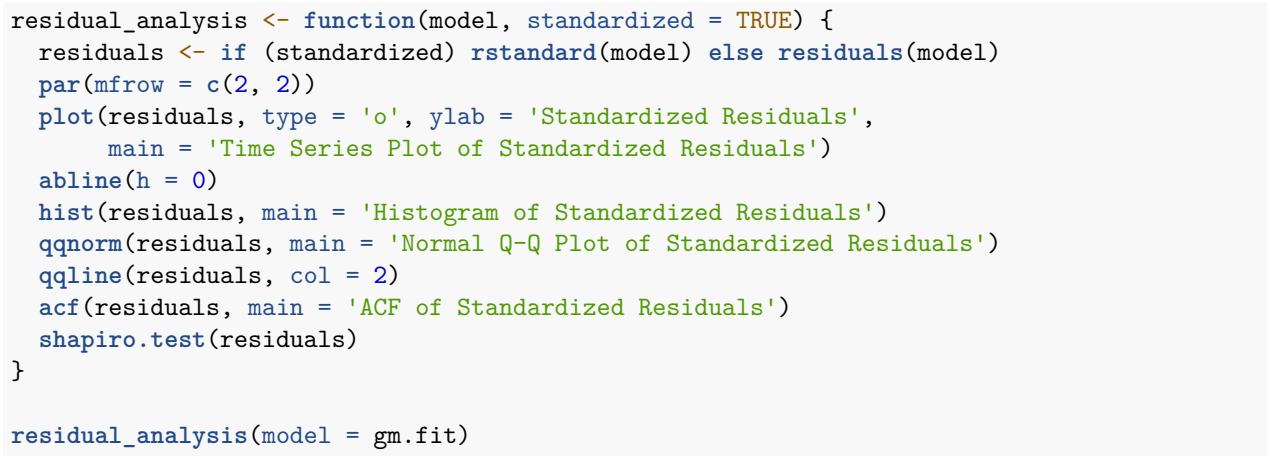
```
##
## Attaching package: 'TSA'
```

```
## The following objects are masked from 'package:stats':
##
##     acf, arima
```

```
## The following object is masked from 'package:utils':
##
##     tar
```

```
residual_analysis <- function(model, standardized = TRUE) {
  residuals <- if (standardized) rstandard(model) else residuals(model)
  par(mfrow = c(2, 2))
  plot(residuals, type = 'o', ylab = 'Standardized Residuals',
       main = 'Time Series Plot of Standardized Residuals')
  abline(h = 0)
  hist(residuals, main = 'Histogram of Standardized Residuals')
  qqnorm(residuals, main = 'Normal Q-Q Plot of Standardized Residuals')
  qqline(residuals, col = 2)
  acf(residuals, main = 'ACF of Standardized Residuals')
  shapiro.test(residuals)
}

residual_analysis(model = gm.fit)
```



```
##
##  Shapiro-Wilk normality test
##
## data:  residuals
## W = 0.87008, p-value = 2.052e-11
```

From the residual analysis, we conclude the following, for the chosen model:

- From the Time Series plot, the residuals shown no trend, i.e stationary.
- From the histogram, we notice a normal distribution of the residuals.
- The QQplot also displays normality.
- And, the ACF plot represents white noise residuals.

Thus, we can say that the model we chose, fits the best.

Forecasting

The last step would be forecasting the monthly sales for the General Motors Company for the next 24 months.

```
# Predict log-transformed GM time series 24 time points ahead

n.ahead <- 24
x.forecast_2 <- predict(gm.fit, n.ahead = n.ahead)

# Plot the forecasts with prediction intervals
UU <- x.forecast_2$pred + 1.96 * x.forecast_2$se
LL <- x.forecast_2$pred - 1.96 * x.forecast_2$se
```

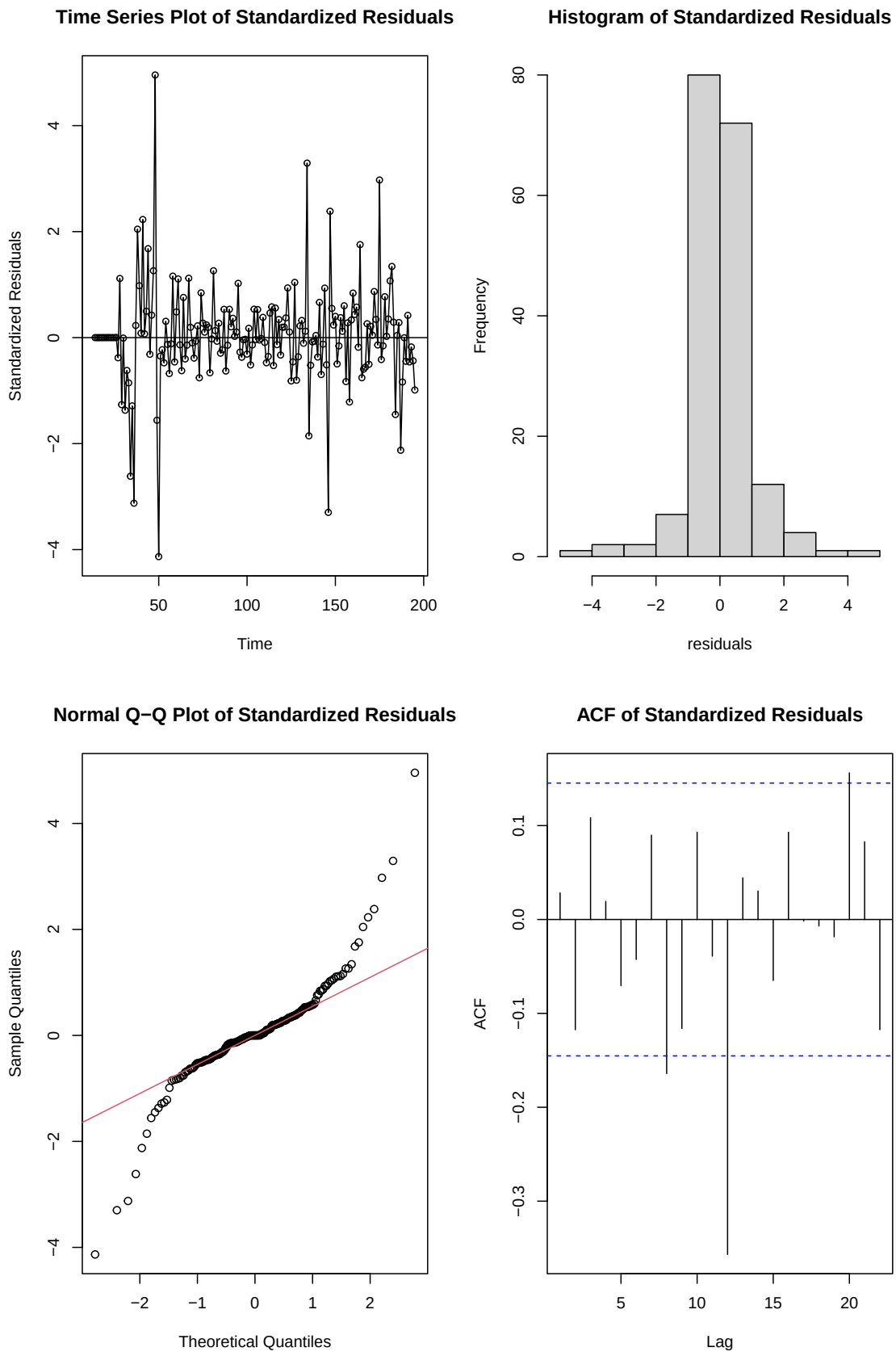


Figure 9: Residual Analysis for gm.fit model

```r
# Plot the log-transformed time series
plot((length(gm_diff2) + 1):(length(gm_diff2) + n.ahead),
     x.forecast_2$pred,
     xlim = c(1, length(gm_diff2) + n.ahead),
     ylim = c(min(gm_diff2, UU, LL), max(gm_diff2, UU, LL)),
     col = "blue",
     type = "l",
     xlab = "Time in Months",
     ylab = "Log-Transformed GM",
     main = "Log-Transformed Time Series and Forecast")

lines((length(gm_diff2) + 1):(length(gm_diff2) + n.ahead), UU,
      col = "green", lty = "dashed", type = "l")
lines((length(gm_diff2) + 1):(length(gm_diff2) + n.ahead), LL,
      col = "green", lty = "dashed", type = "l")
lines(1:length(gm_diff2), gm_diff2, col = "red", type = "l")
```

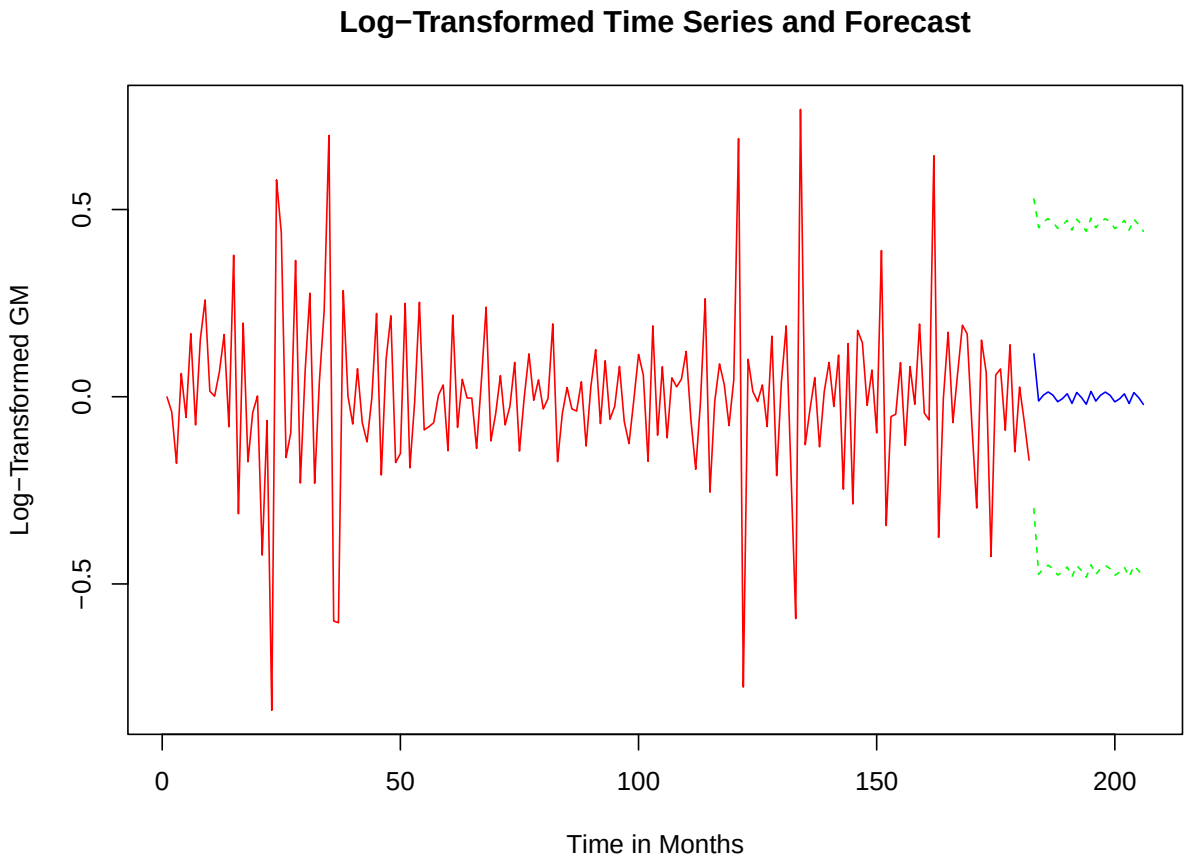


Figure 10: The forecasted values of General Motors Time Series for 24 months

```r
# Plot the original time series
plot((length(gm) + 1):(length(gm) + n.ahead),
     exp(x.forecast_2$pred),
```

```
     xlim = c(1, length(gm) + n.ahead),
     ylim = c(min(gm, exp(UU), exp(LL)), max(gm, exp(UU), exp(LL))),
     col = "blue",
     type = "l",
     xlab = "Time in Months",
     ylab = "Original GM",
     main = "Original Time Series and Forecast")

lines((length(log(gm)) + 1):(length(log(gm)) + n.ahead), exp(UU), col = "blue", lty = "dashed")
lines((length(gm) + 1):(length(gm) + n.ahead), exp(LL), col = "blue", lty = "dashed")
lines(1:length(gm), gm, col = "red")
```

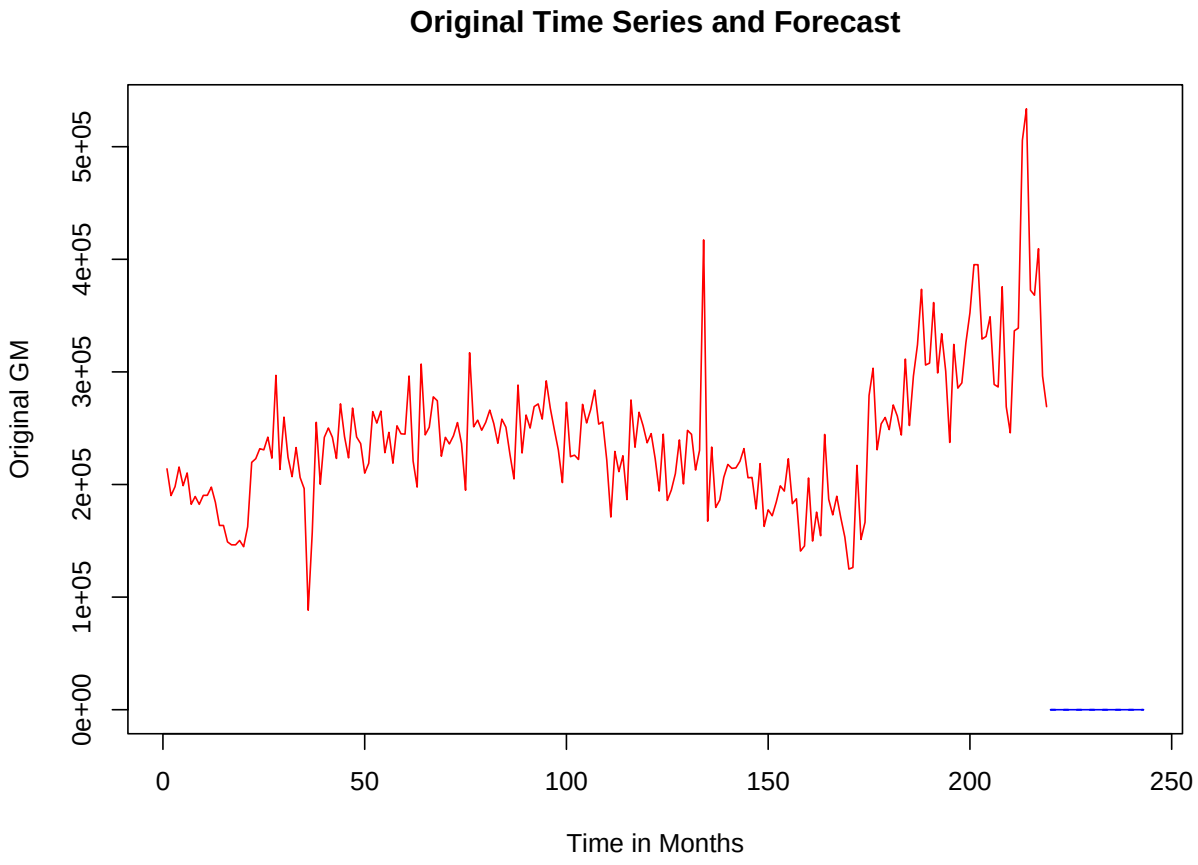


Figure 11: The forecasted values of General Motors Time Series for 24 months

Conclusions

Market Analysis: The time series analysis provides valuable insights into the monthly sales performance of General Motors Company in the US automotive market over the past two decades.

At first, a consistently steady trend is evident in the sales data. Notably, there is a spike around the year 2016, attributed to several factors including the favorable conditions of the overall automotive market in the United States and the ongoing economic recovery post the global financial crisis of 2008. Subsequently, a notable decline is observed during the 2019-2020 timeframe, attributed to the impact of the COVID-19 pandemic. Following this downturn, there is a subsequent increase in sales, characterized by a more sustained pattern.

On the model selection, the SARIMA(0,1,2)(0,1,1) model was identified as the best-fit model for the time series. The model selection process involved considering various combinations of parameters, and the chosen model demonstrated the lowest AIC, indicating a good balance between model complexity and goodness of fit.

The original time series exhibited non-stationarity, but after applying first-order differencing and seasonal differencing at lag 12, a stationary series was achieved. This transformation is essential for building accurate forecasting models.

The diagnostic analysis of the selected model's residuals indicates that the residuals exhibit characteristics of white noise, suggesting that the model adequately captures the underlying patterns in the data.

The forecasting section provides a glimpse into the future by predicting General Motors' monthly sales for the next 24 months. The forecasted values, along with prediction intervals, offer a range of potential outcomes and highlight the uncertainty associated with future predictions.

In contrast to the initial data available for this timeframe, variations are noticeable, but it appears that the prediction intervals encompass the original data. This implies that the prediction error is likely to fall within the specified interval.

References

1. Brockwell, Peter J., Davis, Richard A. (2016). Introduction to Time Series and Forecasting, Third Edition

2. Cryer, J.D., Chan, K-S., (2008). Time Series Analysis with applications in R, Second Edition

3. Prabhakaran, S. (2021). ARIMA Model - Complete Guide to Time Series Forecasting in Python (https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/)

4. Graves, A. (2020). Time Series Forecasting with a SARIMA Model (https://towardsdatascience.com/time-series-forecasting-with-a-sarima-model-db051b7ae459)