Final_ARIMA

Xiaoqing Xia, Digvijay Yadav

2023-05-16

```r
library(readr)
tra <- read_csv("~/Desktop/QBS 126/Final Project/train_combine.csv")
```

```
## Rows: 420212 Columns: 20
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): Type
## dbl  (17): Store, Dept, Weekly_Sales, month, day, year, week, Temperature, F...
## lgl   (1): IsHoliday
## date  (1): Date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(tra)
```

```
## # A tibble: 6 x 20
##   Store  Dept Date       Weekl~1 IsHol~2 month   day  year  week Tempe~3 Fuel_~4
##   <dbl> <dbl> <date>       <dbl> <lgl>   <dbl> <dbl> <dbl> <dbl>   <dbl>   <dbl>
## 1     1     1 2010-02-05  24924. FALSE       2     5  2010     5    42.3    2.57
## 2    35     3 2010-02-05  14612. FALSE       2     5  2010     5    27.2    2.78
## 3    35     4 2010-02-05  26323. FALSE       2     5  2010     5    27.2    2.78
## 4    35     5 2010-02-05  36415. FALSE       2     5  2010     5    27.2    2.78
## 5    35     6 2010-02-05  11438. FALSE       2     5  2010     5    27.2    2.78
## 6    35     7 2010-02-05  23416. FALSE       2     5  2010     5    27.2    2.78
## # ... with 9 more variables: MarkDown1 <dbl>, MarkDown2 <dbl>, MarkDown3 <dbl>,
## #   MarkDown4 <dbl>, MarkDown5 <dbl>, CPI <dbl>, Unemployment <dbl>,
## #   Type <chr>, Size <dbl>, and abbreviated variable names 1: Weekly_Sales,
## #   2: IsHoliday, 3: Temperature, 4: Fuel_Price
```

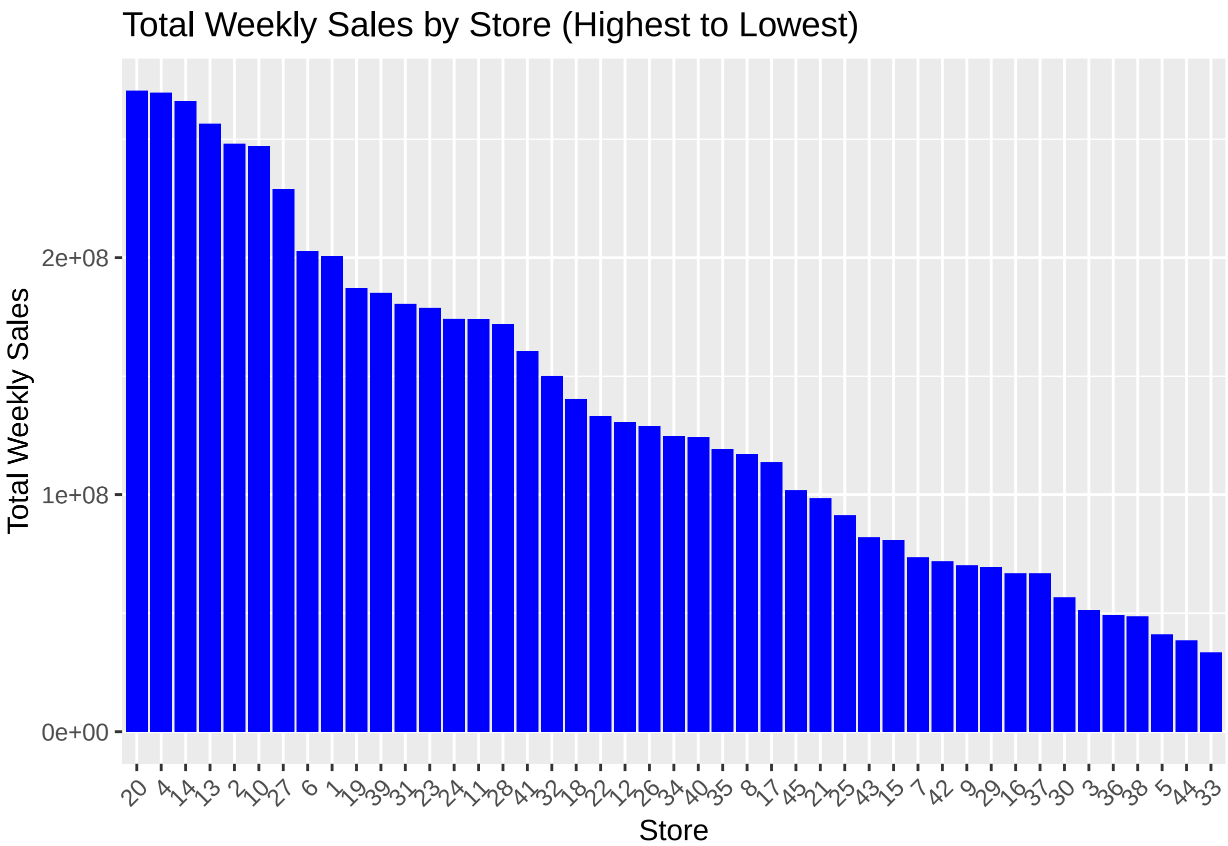
```r
# Set the seed for reproducibility
set.seed(123)
total_len <- nrow(tra)
train_len <- round(0.9 *total_len)
test_len <- total_len - train_len

# Split the dataset into training and test sets
train <- tra[1:train_len,]
test <- tra[(train_len + 1):total_len,]
```

Selecting Subset of Department and Store

```r
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# Calculate the total weekly sales by store
total_sales_by_store <- train %>%
  group_by(Store) %>%
  summarize(Total_Weekly_Sales = sum(Weekly_Sales)) %>%
  arrange(desc(Total_Weekly_Sales))

# Plot total weekly sales by store
ggplot(total_sales_by_store, aes(x = reorder(Store, Total_Weekly_Sales,decreasing = TRUE), y = Total_We
  geom_bar(stat = "identity", fill = "blue") +
  labs(x = "Store", y = "Total Weekly Sales") +
  ggtitle("Total Weekly Sales by Store (Highest to Lowest)") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

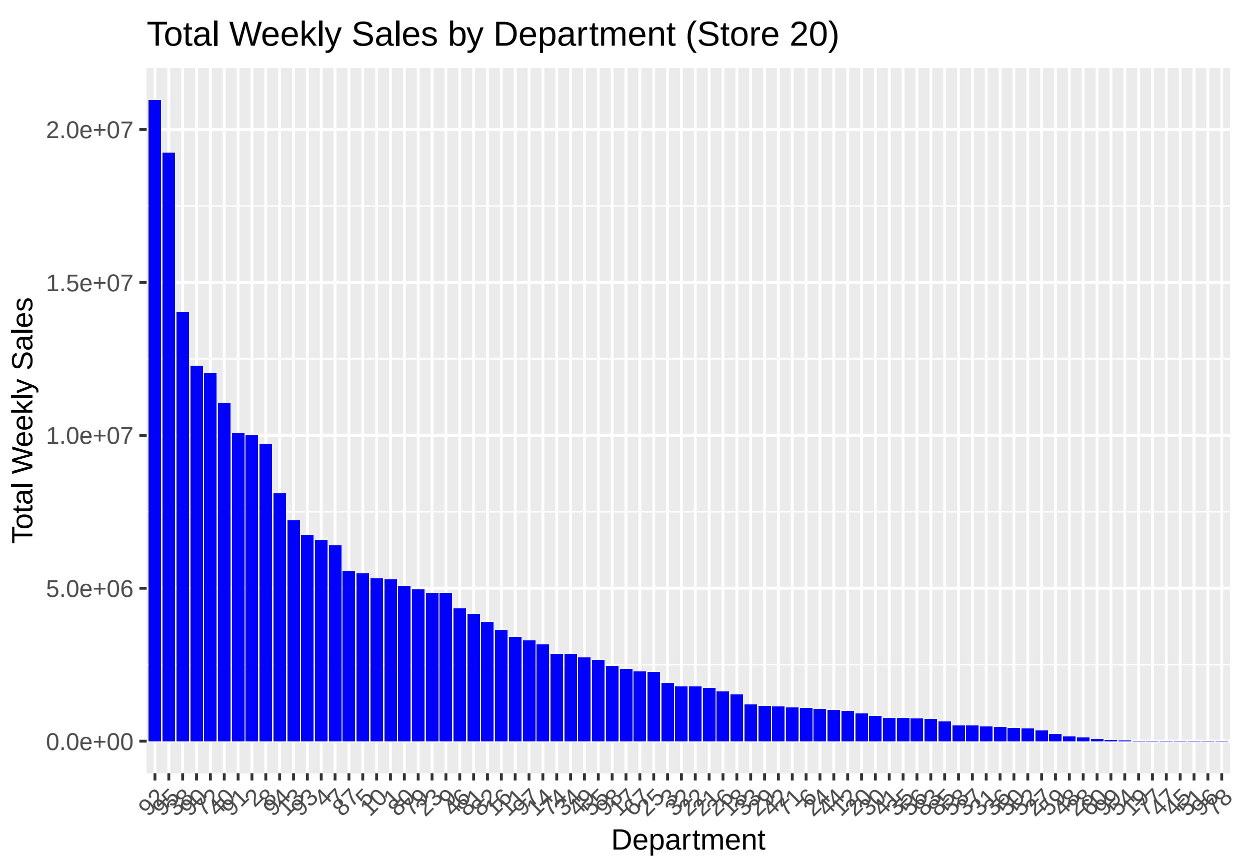



```r
head(total_sales_by_store)
```

```
## # A tibble: 6 x 2
##   Store Total_Weekly_Sales
##   <dbl>              <dbl>
## 1    20         270561862.
## 2     4         269659070.
## 3    14         266152272.
## 4    13         256595962.
## 5     2         248152053.
## 6    10         247075799
```

The top store with highest total weekly sales is 20.

```r
store_20_data <- train %>%
  filter(Store == 20) %>%
  group_by(Dept) %>%
  summarize(Total_Weekly_Sales = sum(Weekly_Sales)) %>%
  arrange(desc(Total_Weekly_Sales))

# Plot total weekly sales by department for Store 20
ggplot(store_20_data, aes(x = reorder(Dept, Total_Weekly_Sales,decreasing = TRUE), y = Total_Weekly_Sal
  geom_bar(stat = "identity", fill = "blue") +
  labs(x = "Department", y = "Total Weekly Sales") +
  ggtitle("Total Weekly Sales by Department (Store 20)") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The top department is 92. I will choose the 92nd department in Store 20.

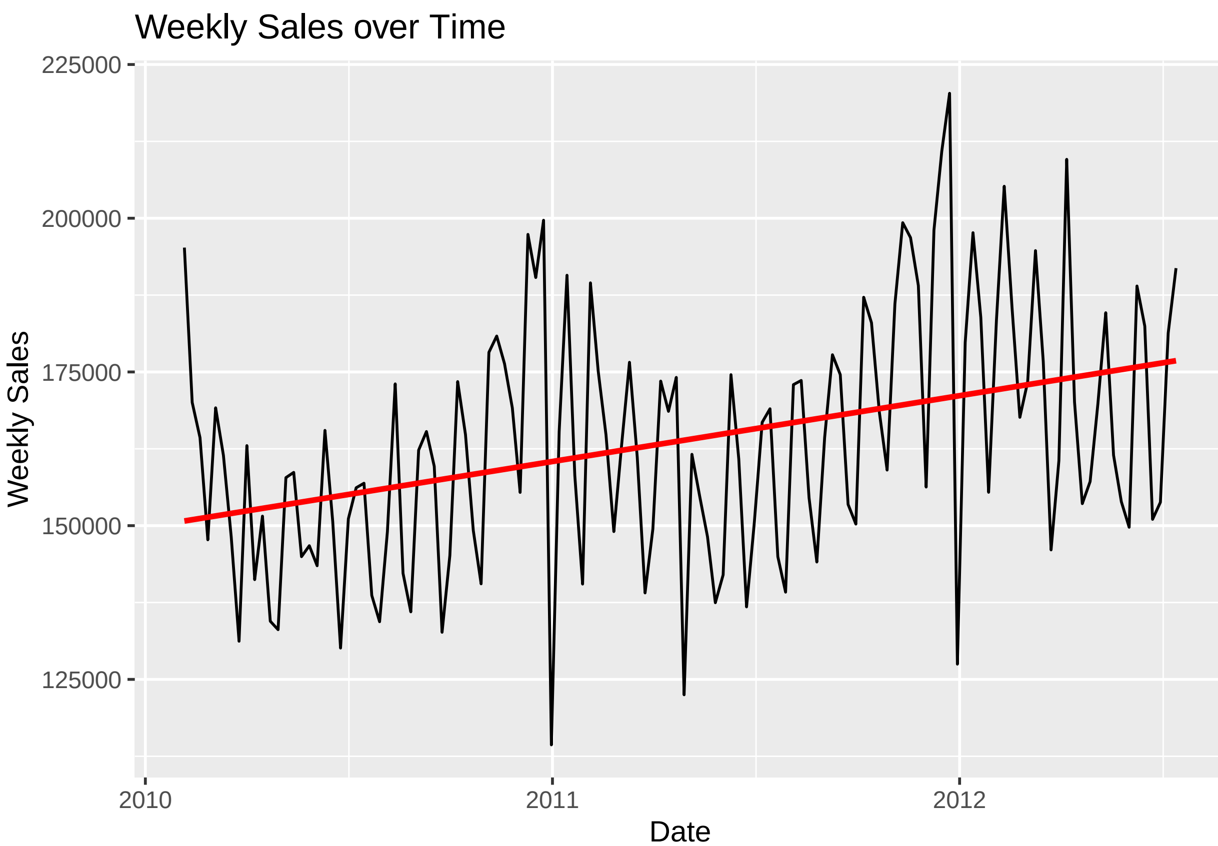
```r
train_data <- train %>%
  filter(Store == 20, Dept == 92)
train_data
```

```
## # A tibble: 128 x 20
##    Store  Dept Date       Weekly_Sales IsHoliday month   day  year  week Tempe~1
##    <dbl> <dbl> <date>            <dbl> <lgl>     <dbl> <dbl> <dbl> <dbl>   <dbl>
##  1    20    92 2010-02-05      195224. FALSE         2     5  2010     5    25.9
##  2    20    92 2010-02-12      170044. TRUE          2    12  2010     6    22.1
##  3    20    92 2010-02-19      164314. FALSE         2    19  2010     7    25.4
##  4    20    92 2010-02-26      147700. FALSE         2    26  2010     8    32.3
##  5    20    92 2010-03-05      169171. FALSE         3     5  2010     9    31.8
##  6    20    92 2010-03-12      161433. FALSE         3    12  2010    10    43.8
##  7    20    92 2010-03-19      148157. FALSE         3    19  2010    11    47.3
##  8    20    92 2010-03-26      131205. FALSE         3    26  2010    12    50.5
##  9    20    92 2010-04-02      163023. FALSE         4     2  2010    13    51
## 10    20    92 2010-04-09      141224. FALSE         4     9  2010    14    65.1
## # ... with 118 more rows, 10 more variables: Fuel_Price <dbl>, MarkDown1 <dbl>,
## #   MarkDown2 <dbl>, MarkDown3 <dbl>, MarkDown4 <dbl>, MarkDown5 <dbl>,
## #   CPI <dbl>, Unemployment <dbl>, Type <chr>, Size <dbl>, and abbreviated
## #   variable name 1: Temperature
```

```r
ggplot(train_data, aes(x = Date, y = Weekly_Sales)) +
  geom_line() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
```

```r
  labs(x = "Date", y = "Weekly Sales") +
  ggtitle("Weekly Sales over Time")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



The subset is not stationary.

```r
train_data
```

```
## # A tibble: 128 x 20
##    Store  Dept Date       Weekly_Sales IsHoliday month   day  year  week Tempe~1
##    <dbl> <dbl> <date>            <dbl> <lgl>     <dbl> <dbl> <dbl> <dbl>   <dbl>
##  1    20    92 2010-02-05      195224. FALSE         2     5  2010     5    25.9
##  2    20    92 2010-02-12      170044. TRUE          2    12  2010     6    22.1
##  3    20    92 2010-02-19      164314. FALSE         2    19  2010     7    25.4
##  4    20    92 2010-02-26      147700. FALSE         2    26  2010     8    32.3
##  5    20    92 2010-03-05      169171. FALSE         3     5  2010     9    31.8
##  6    20    92 2010-03-12      161433. FALSE         3    12  2010    10    43.8
##  7    20    92 2010-03-19      148157. FALSE         3    19  2010    11    47.3
##  8    20    92 2010-03-26      131205. FALSE         3    26  2010    12    50.5
##  9    20    92 2010-04-02      163023. FALSE         4     2  2010    13    51
## 10    20    92 2010-04-09      141224. FALSE         4     9  2010    14    65.1
## # ... with 118 more rows, 10 more variables: Fuel_Price <dbl>, MarkDown1 <dbl>,
## #   MarkDown2 <dbl>, MarkDown3 <dbl>, MarkDown4 <dbl>, MarkDown5 <dbl>,
## #   CPI <dbl>, Unemployment <dbl>, Type <chr>, Size <dbl>, and abbreviated
## #   variable name 1: Temperature
```

Seasonal Plot

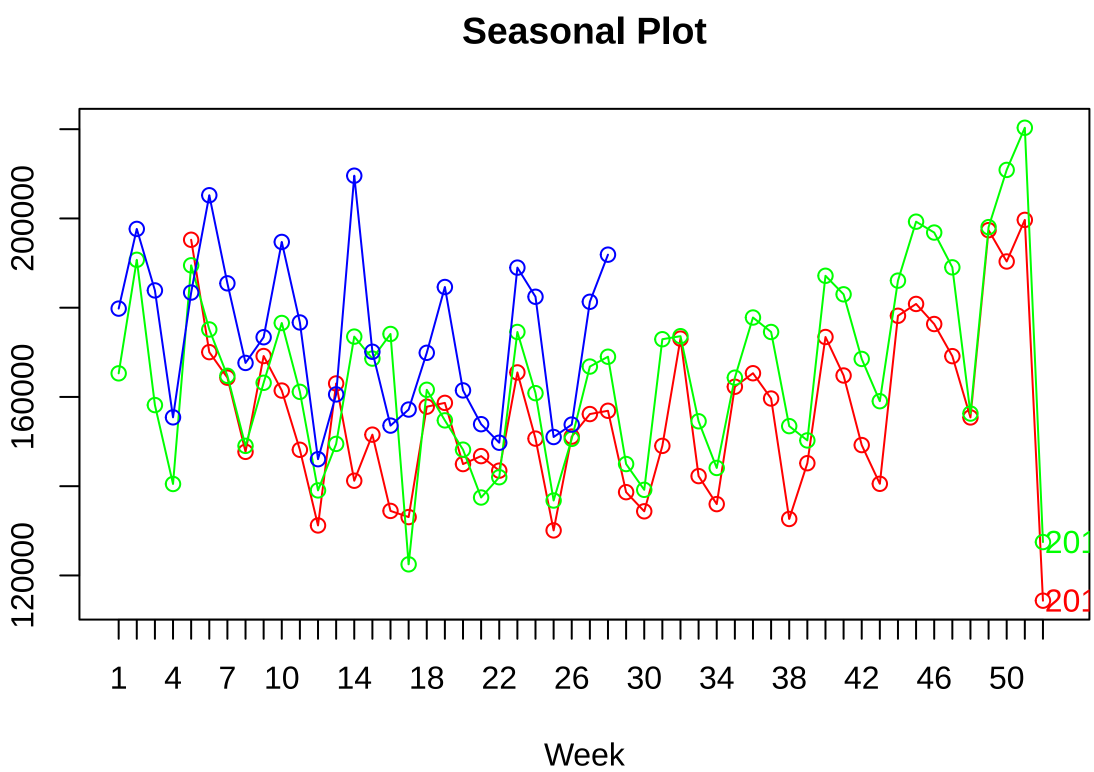
```
library(forecast)
```

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

```
train_df=ts(train_data$Weekly_Sales,start=c(2010,05) ,frequency= 52)
var(train_df)
```

```
## [1] 418398460
```

```
seasonplot(train_df, 52, col=rainbow(3), year.labels=TRUE, main="Seasonal Plot")
```




```
var(train_data$Weekly_Sales)
```

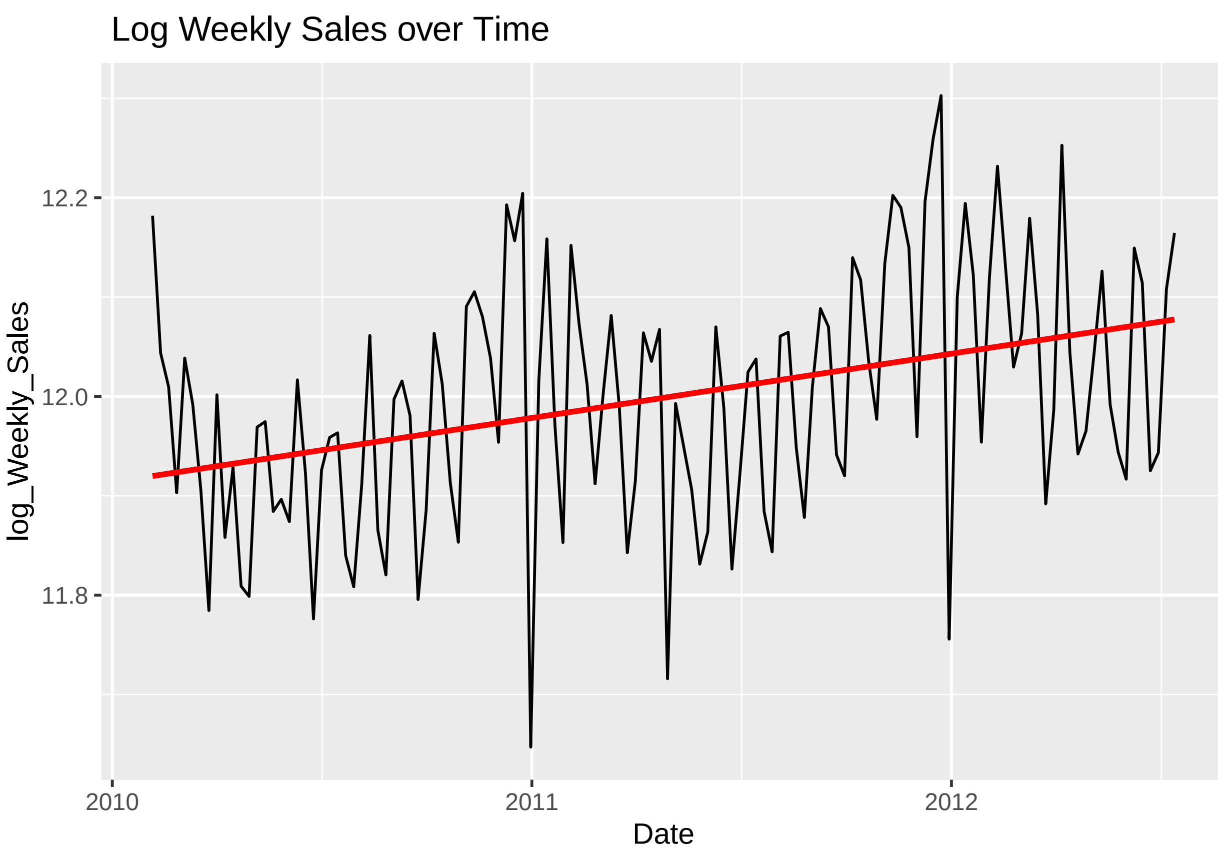
```
## [1] 418398460
```

Detrend Dataset

Since the value of weekly sales range from 0 to 700000, which means weekly sales contain large values. I prefer to use log function to standardize the variance by compressing large values and expanding small values.

```r
log_data <- train_data %>%
  mutate(log_Weekly_Sales = log(Weekly_Sales))
ggplot(log_data, aes(x = Date, y = log_Weekly_Sales)) +
  geom_line() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(x = "Date", y = "log_Weekly_Sales") +
  ggtitle(" Log Weekly Sales over Time")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```r
var(log_data$log_Weekly_Sales)
```

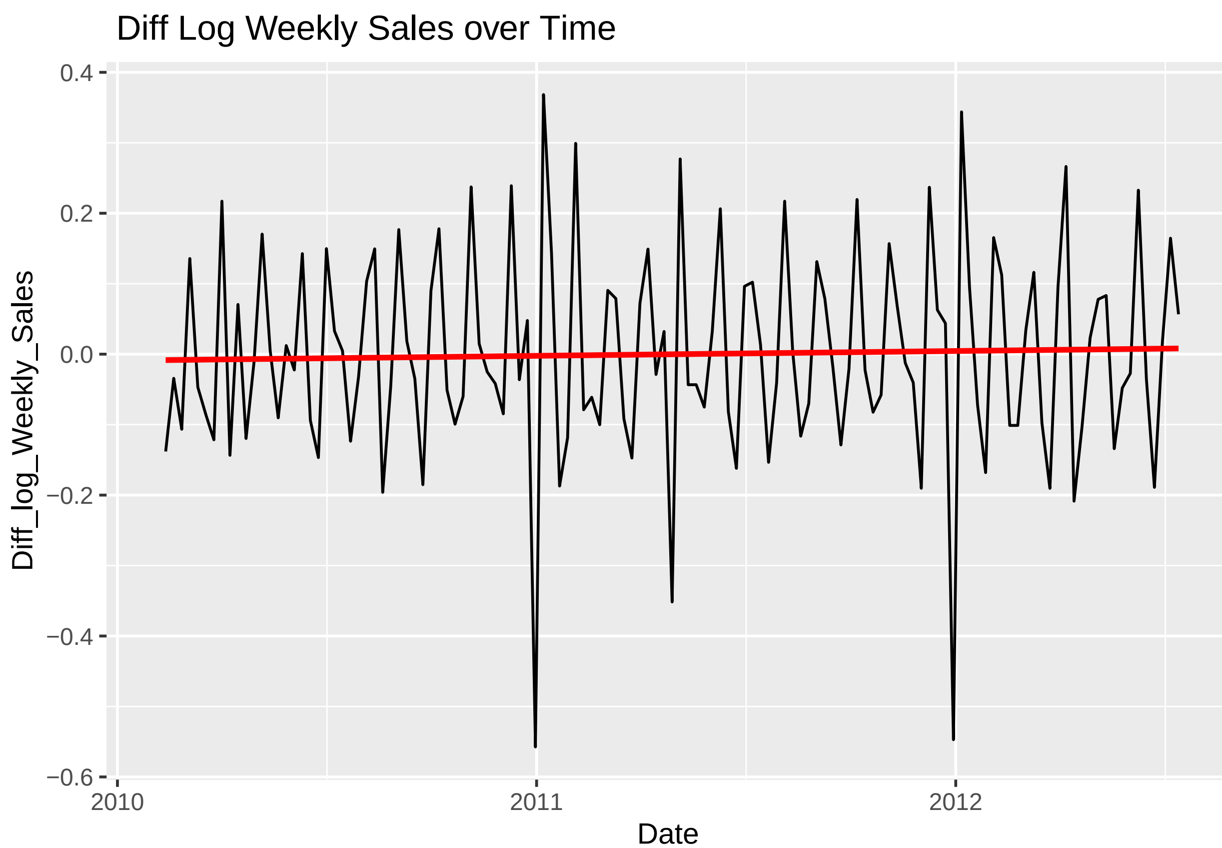
```
## [1] 0.01558717
```

difference it once

```r
log_diff_data <- log_data %>%
  mutate(diff_Weekly_Sales = log_Weekly_Sales- lag(log_Weekly_Sales))
ggplot(log_diff_data, aes(x = Date, y = diff_Weekly_Sales)) +
  geom_line() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(x = "Date", y = "Diff_log_Weekly_Sales") +
  ggtitle(" Diff Log Weekly Sales over Time")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 1 rows containing non-finite values (stat_smooth).

## Warning: Removed 1 row(s) containing missing values (geom_path).
```

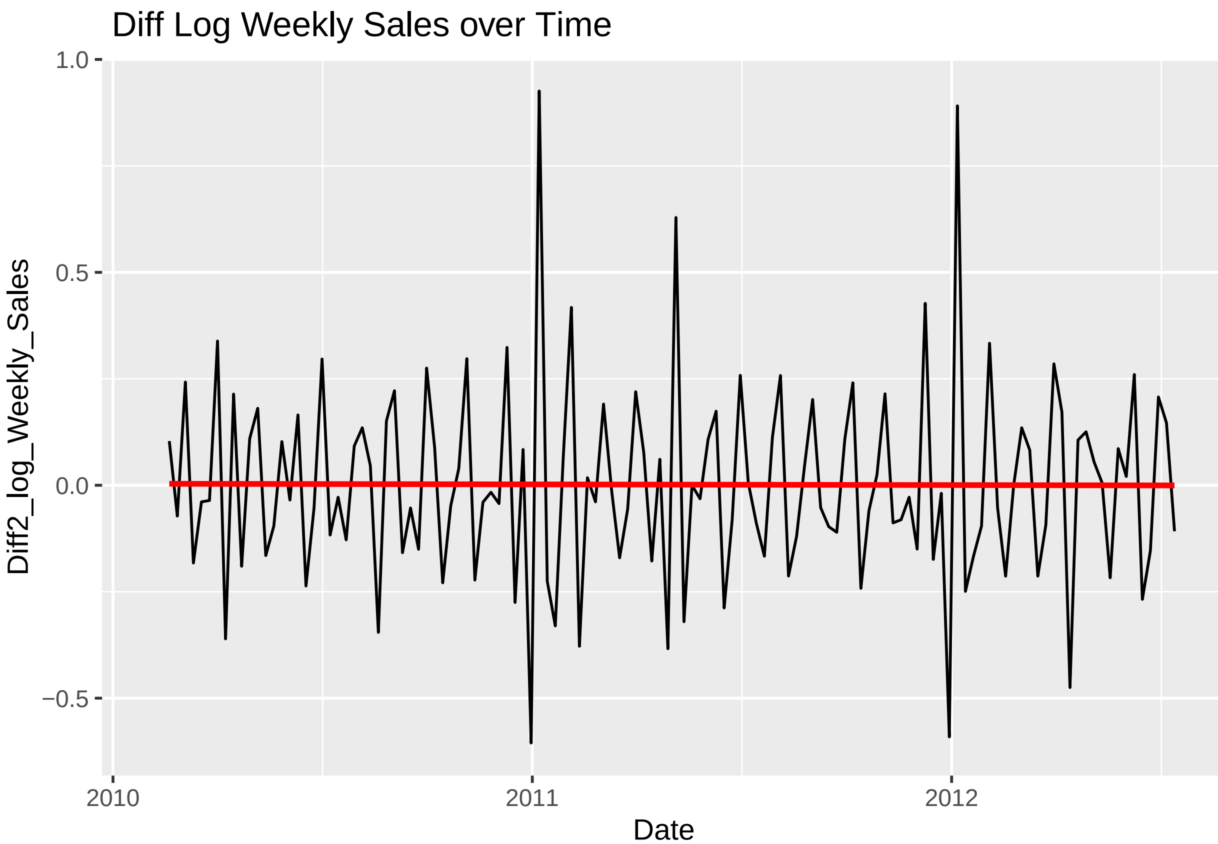



```r
clean_data <- na.omit(log_diff_data$diff_Weekly_Sales)
var(clean_data)
```

```
## [1] 0.02229928
```

```r
log_diff1_data <- log_diff_data %>%
  mutate(diff2_Weekly_Sales = diff_Weekly_Sales- lag(diff_Weekly_Sales))
ggplot(log_diff1_data, aes(x = Date, y = diff2_Weekly_Sales)) +
  geom_line() +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(x = "Date", y = "Diff2_log_Weekly_Sales") +
  ggtitle(" Diff Log Weekly Sales over Time")
```

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 2 rows containing non-finite values (stat_smooth).

## Warning: Removed 2 row(s) containing missing values (geom_path).
```




```
clean_data <- na.omit(log_diff1_data$diff2_Weekly_Sales)
var(clean_data)
```

```
## [1] 0.05597843
```

```
library(tseries)

# Perform ADF test
adf_result <- adf.test(na.omit(log_diff_data$diff_Weekly_Sales))
```

```
## Warning in adf.test(na.omit(log_diff_data$diff_Weekly_Sales)): p-value smaller
## than printed p-value
```

```
# Print the ADF test results
print(adf_result)
```

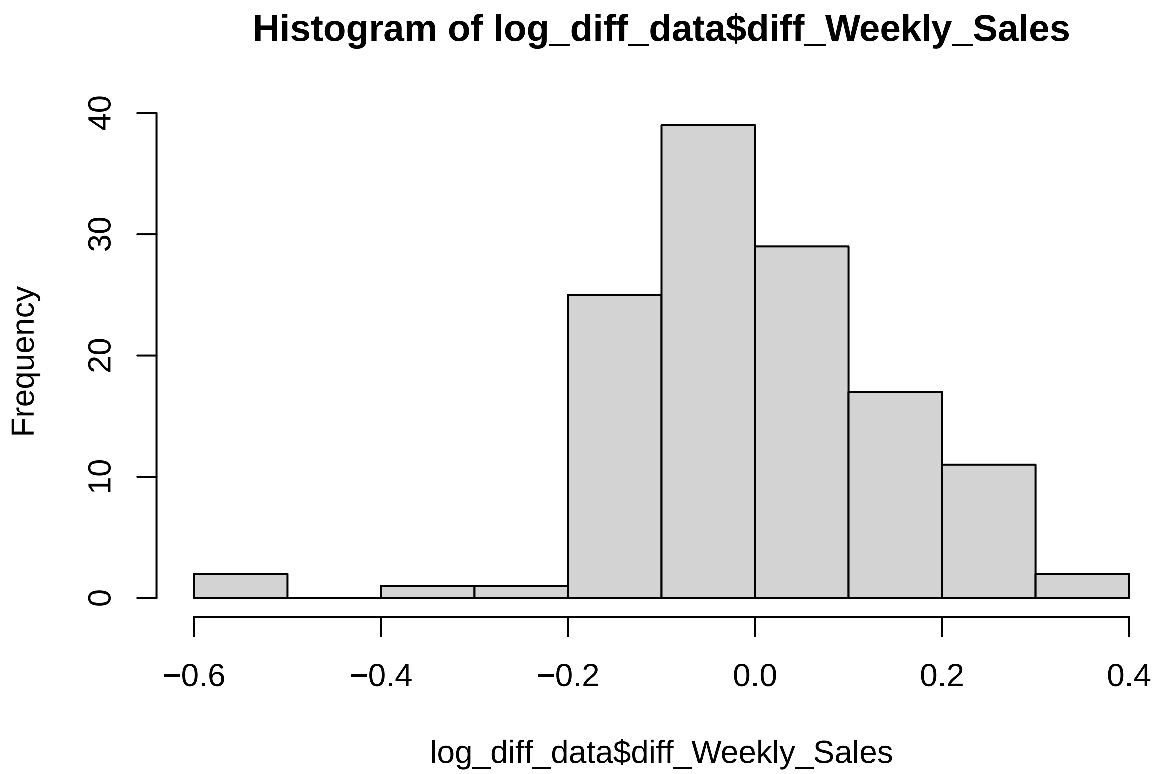
```
##
##  Augmented Dickey-Fuller Test
##
## data:  na.omit(log_diff_data$diff_Weekly_Sales)
## Dickey-Fuller = -5.7303, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

There is no trend in the subset. It's stationary now.

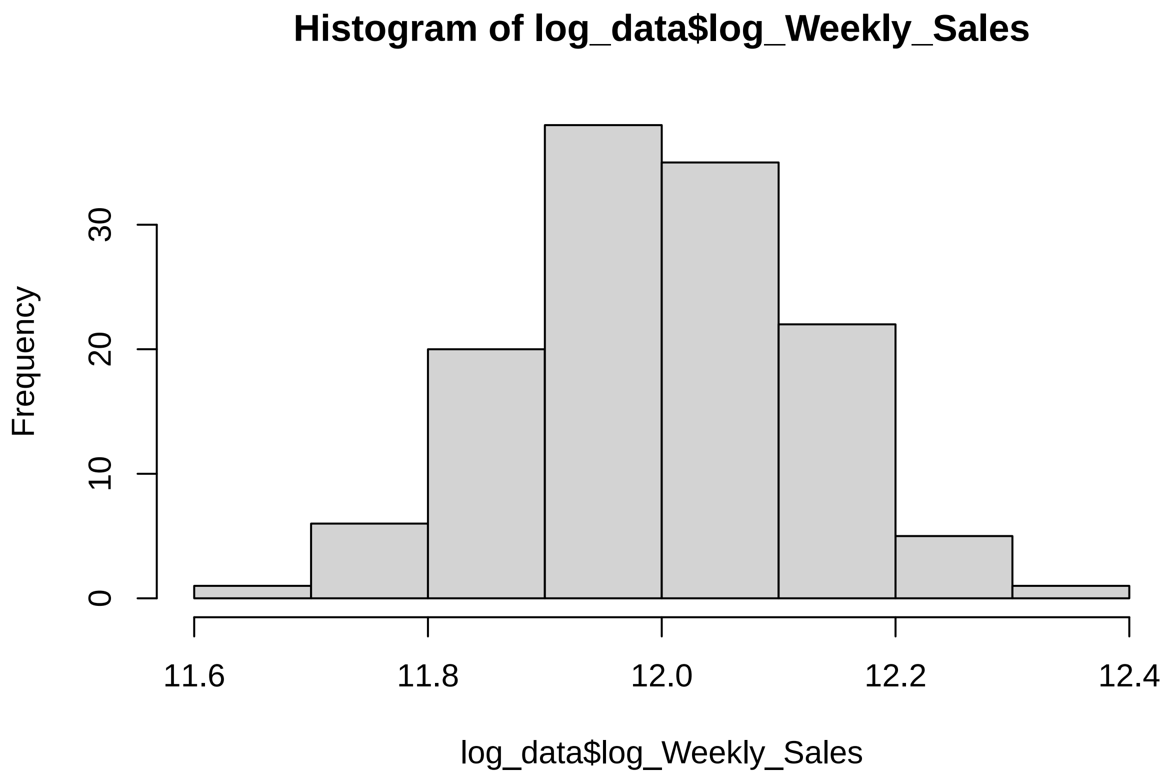
```r
train_df=ts(log_data$log_Weekly_Sales,start=c(2010,5) ,frequency =7)
var(train_df)
```

```
## [1] 0.01558717
```

```r
hist(log_diff_data$diff_Weekly_Sales)
```

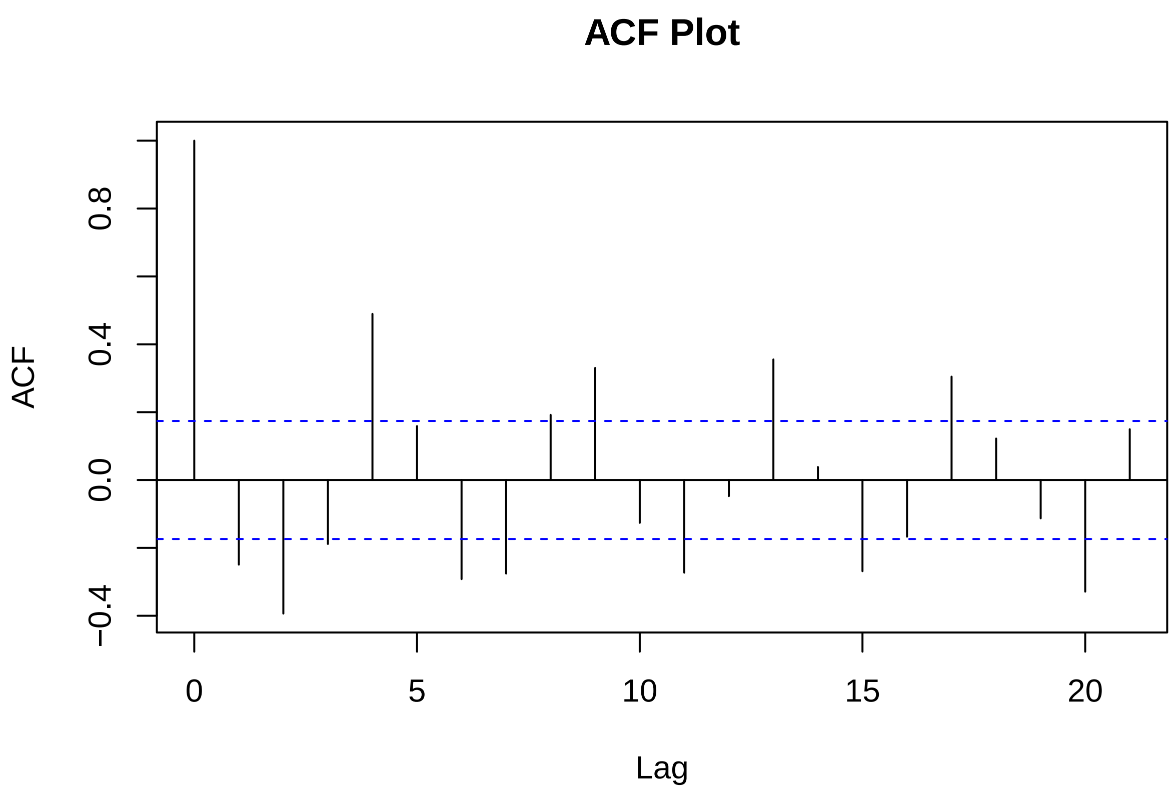



```r
hist(log_data$log_Weekly_Sales)
```

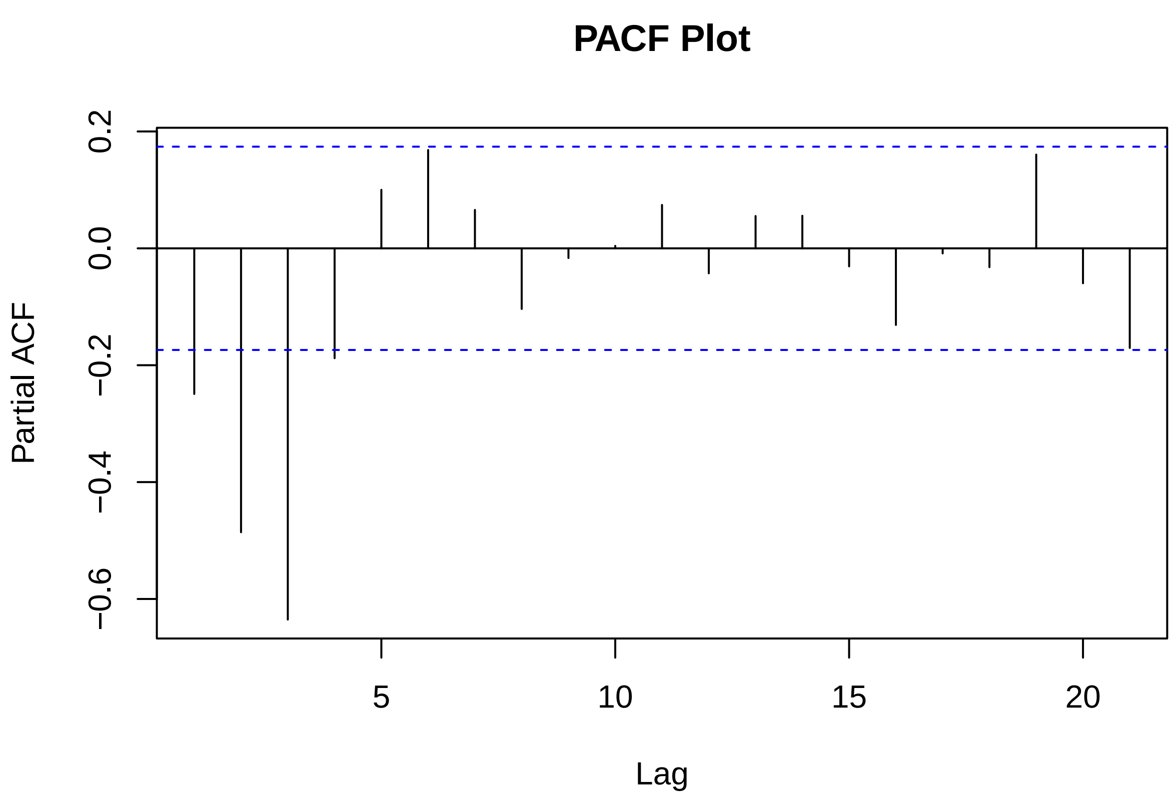


Modeling ARIMA Model

```
library(forecast)
acf(na.omit(log_diff_data$diff_Weekly_Sales), main = "ACF Plot")
```




```r
pacf(na.omit(log_diff_data$diff_Weekly_Sales), main = "PACF Plot")
```

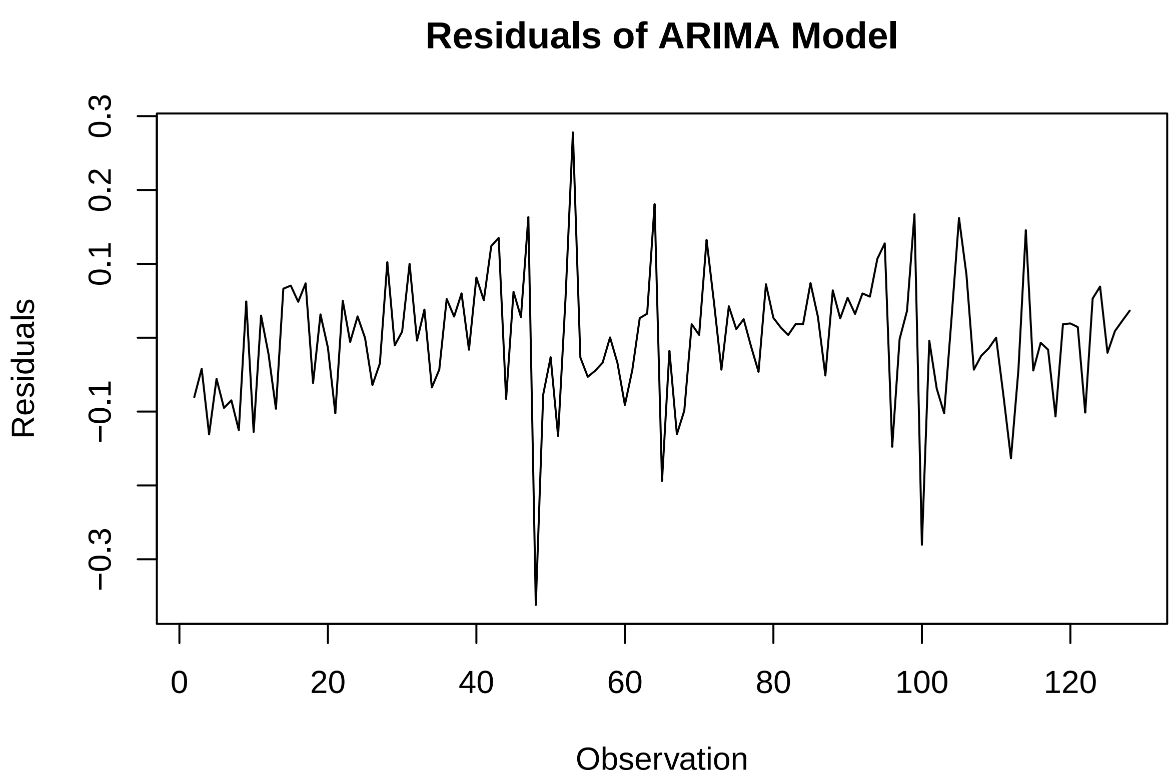


```r
weekly_sales_ts <- ts(log_diff_data$diff_Weekly_Sales, frequency =52)
arima_model <- auto.arima(log_diff_data$diff_Weekly_Sales)

print(arima_model)
```

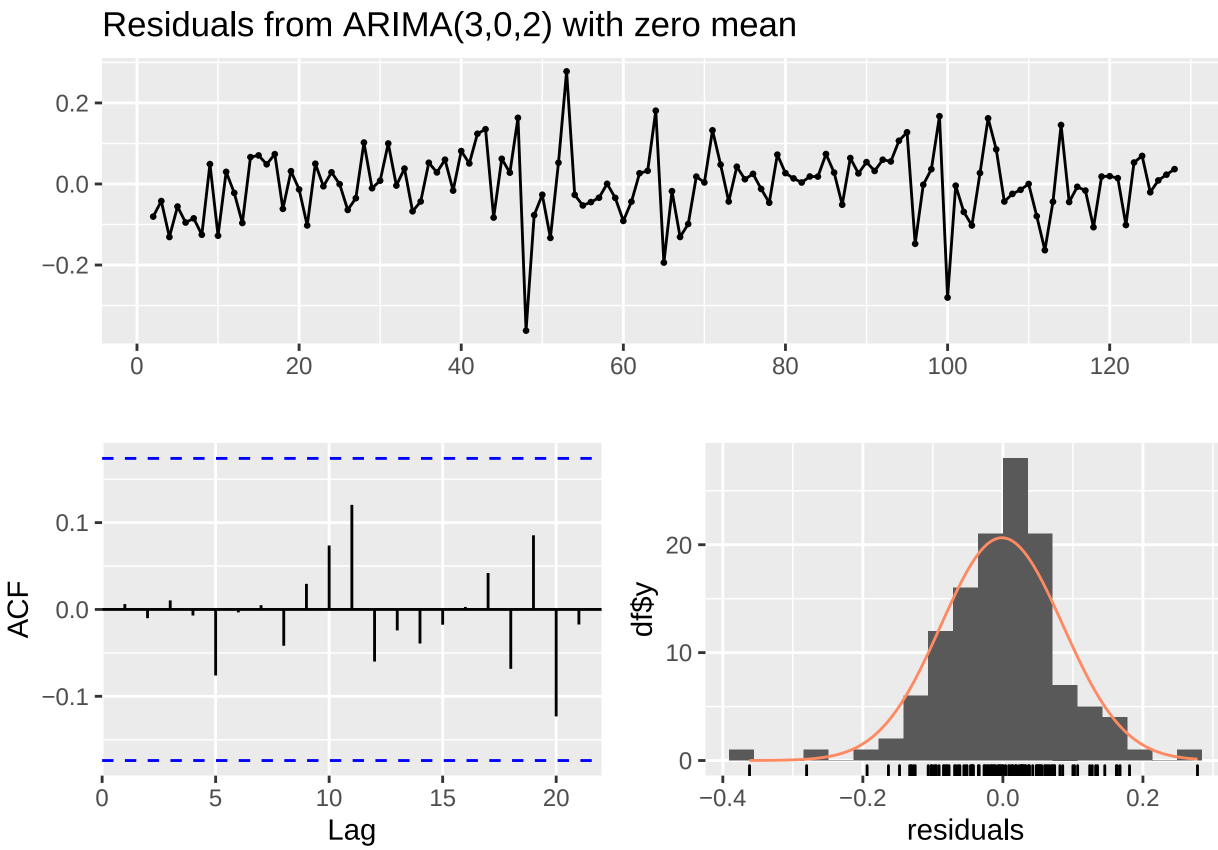
```
## Series: log_diff_data$diff_Weekly_Sales
## ARIMA(3,0,2) with zero mean
##
## Coefficients:
##           ar1      ar2      ar3      ma1     ma2
##       -0.3321  -0.7572  -0.5184  -0.5802  0.4143
## s.e.   0.1247   0.0485   0.0995   0.1285  0.1210
##
## sigma^2 = 0.007863:  log likelihood = 128.52
## AIC=-245.05   AICc=-244.35   BIC=-227.98
```

Checking Residuals

```r
# Get the residuals of the ARIMA model
residuals_arima <- residuals(arima_model)

# Plot the residuals
plot(residuals_arima, type = "l", main = "Residuals of ARIMA Model", xlab = "Observation", ylab = "Resi
```

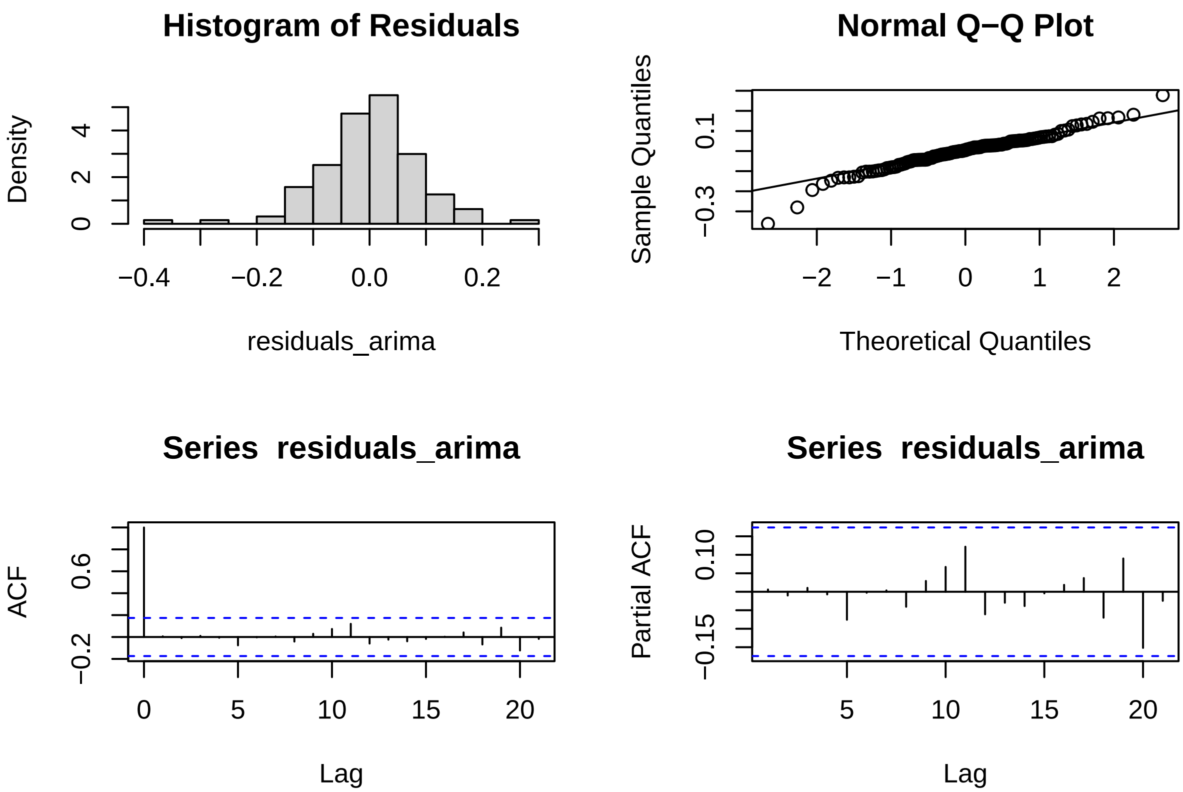



```r
checkresiduals(arima_model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,0,2) with zero mean
## Q* = 1.9413, df = 5, p-value = 0.8572
##
## Model df: 5.   Total lags used: 10
```

```r
par(mfrow = c(2, 2))
# Generate histogram and QQ-plot of residuals
hist(residuals_arima, breaks = "FD", freq = FALSE, main = "Histogram of Residuals")
qqnorm(residuals_arima)
qqline(residuals_arima)
acf(residuals_arima)
pacf(residuals_arima)
```



I can discover that all of the ACF and PACF are approximately within the 95% confidence interval, which means the model contain the constant variance of error.

Independence Test

```r
shapiro_test <- shapiro.test(residuals_arima)

print(shapiro_test)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals_arima
## W = 0.96371, p-value = 0.001762
```

Correlation Test

```r
box_ljung_test <- Box.test(residuals_arima, lag = 20, type = "Ljung-Box")

print(box_ljung_test)
```

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

```
##
## data:  residuals_arima
## X-squared = 9.2469, df = 20, p-value = 0.9799
```

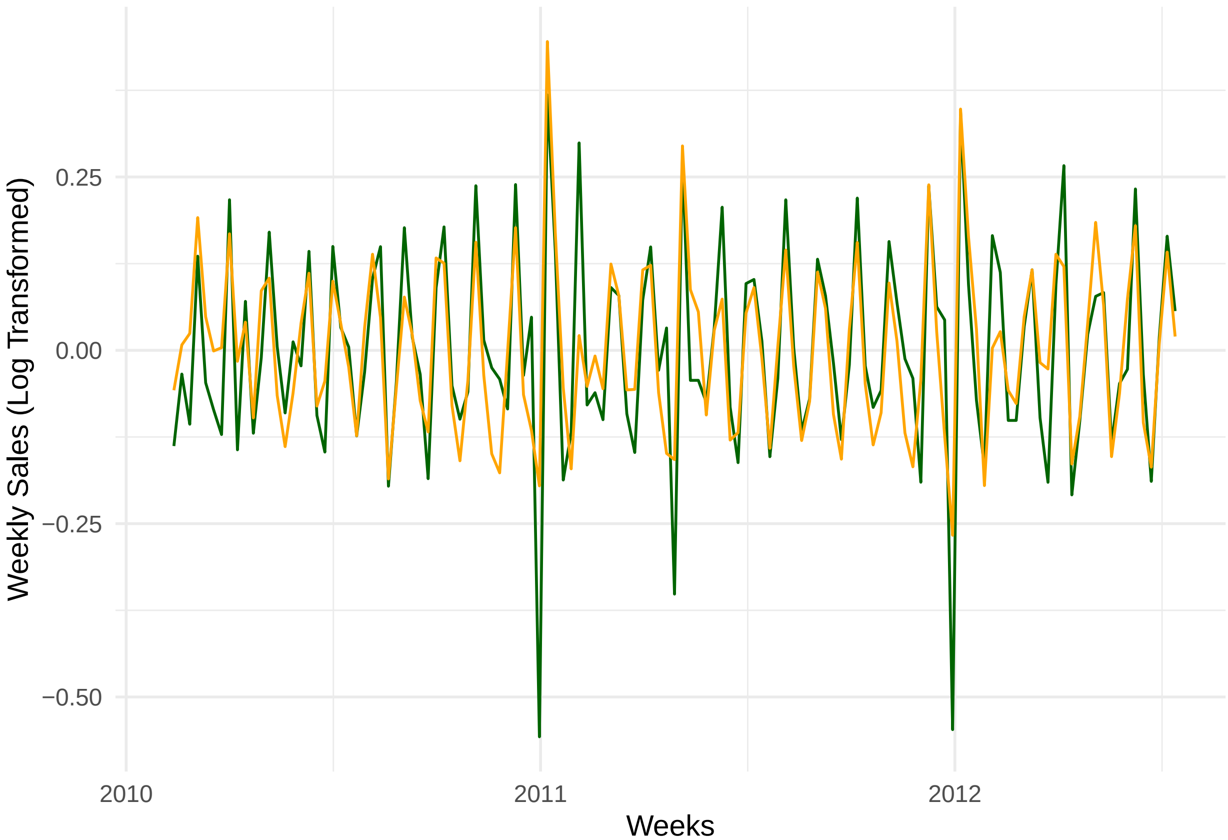
```
box_pierce_test <- Box.test(residuals_arima, lag = 20, type = "Box-Pierce")

print(box_pierce_test)
```

```
##
##  Box-Pierce test
##
## data:  residuals_arima
## X-squared = 8.0743, df = 20, p-value = 0.9914
```

ARIMA Prediction

```
# Obtain predicted values
pred = c(NA,na.omit(log_diff_data$diff_Weekly_Sales) - residuals_arima)

ggplot()+
  geom_line(aes(log_diff_data$Date,log_diff_data$diff_Weekly_Sales),color = 'darkgreen')+
  geom_line(aes(log_diff_data$Date,pred),color='orange')+
  #geom_point(aes(log_diff_datas$Date, test_data$diff_Weekly_Sales), color = 'blue') +
  xlab("Weeks")+
  ylab("Weekly Sales (Log Transformed)") +
  theme_minimal()
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
## Removed 1 row(s) containing missing values (geom_path).
```




```
test_data <- test %>%
  filter(Store == 20, Dept == 92)
```

```
log_datas <- test_data %>%
  mutate(log_Weekly_Sales = log(Weekly_Sales))

log_diff_datas <- log_datas %>%
  mutate(diff_Weekly_Sales = log_Weekly_Sales- lag(log_Weekly_Sales))
```

```
# Predict sales for the test dataset
predicted_test <- c(NA,predict(arima_model, n.ahead = nrow(test_data)))
length(log_diff_datas$Date)
```

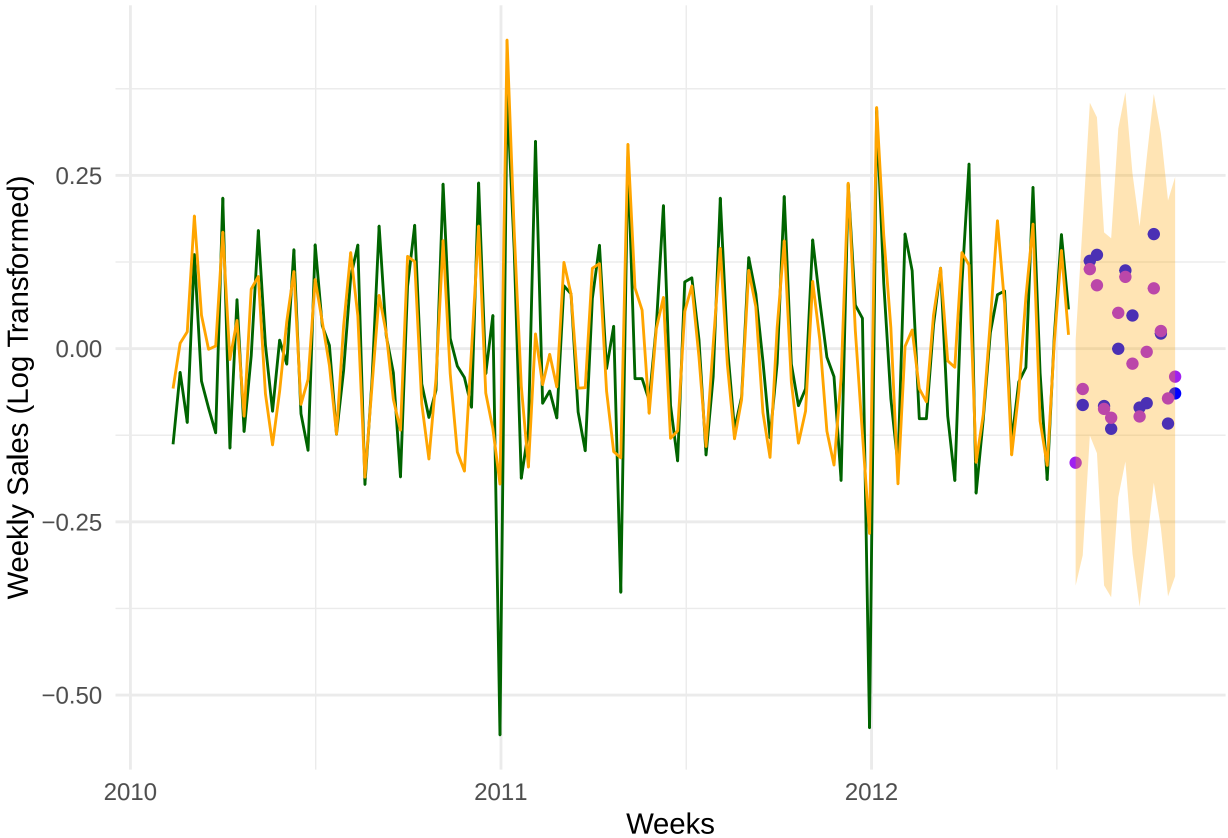
```
## [1] 15
```

```
ggplot()+
  geom_line(aes(log_diff_data$Date,log_diff_data$diff_Weekly_Sales),color = 'darkgreen')+
  geom_line(aes(log_diff_data$Date,pred),color='orange')+
  geom_point(aes(log_diff_datas$Date, log_diff_datas$diff_Weekly_Sales), color = 'blue') +
  geom_point(aes(log_diff_datas$Date, predicted_test$pred), color = 'purple') +
  geom_ribbon(aes(x = log_diff_datas$Date, ymin = predicted_test$pred - 2 * predicted_test$se, ymax = p
  xlab("Weeks")+
  ylab("Weekly Sales (Log Transformed)") +
  theme_minimal()
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```

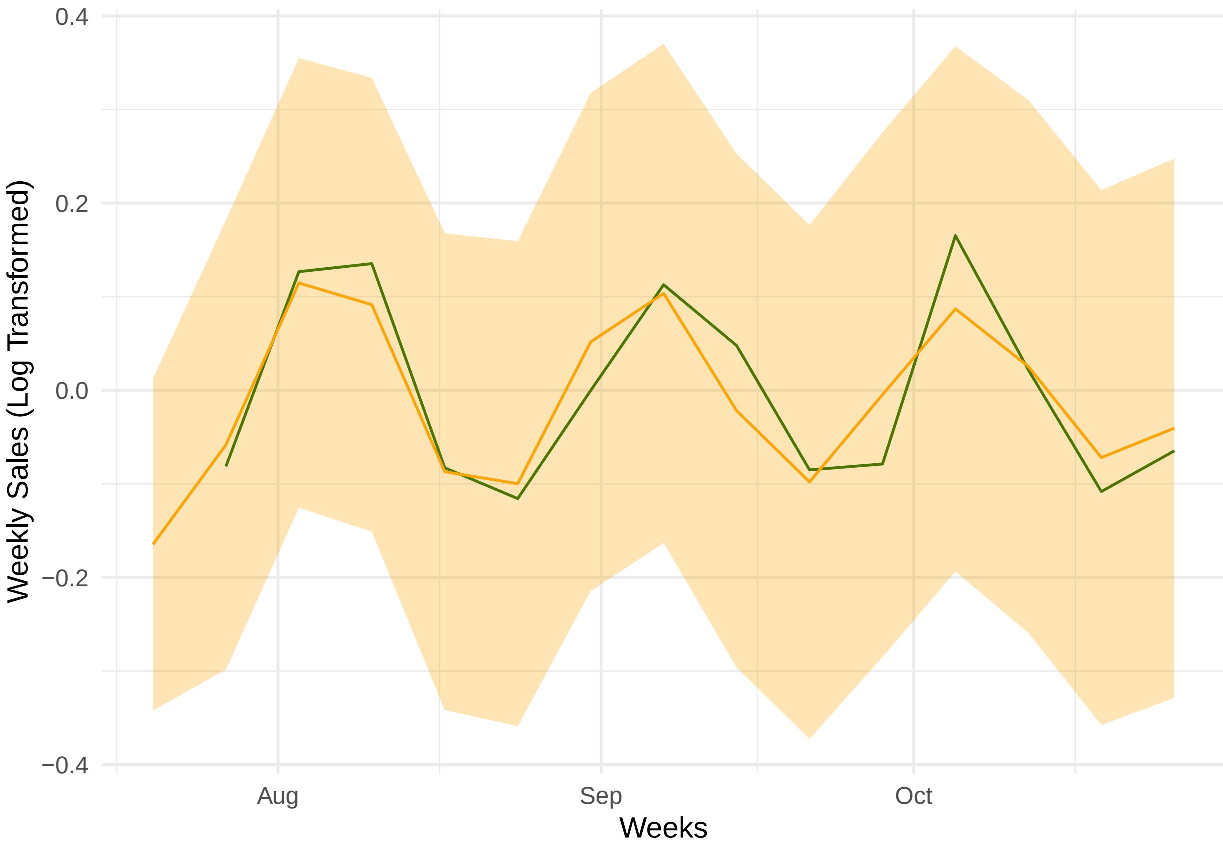
```
## Removed 1 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```




```
ggplot()+
  geom_line(aes(log_diff_datas$Date,log_diff_datas$diff_Weekly_Sales),color = 'darkgreen')+
  geom_line(aes(log_diff_datas$Date,predicted_test$pred),color='orange')+
  geom_ribbon(aes(x = log_diff_datas$Date, ymin = predicted_test$pred - 2 * predicted_test$se, ymax = pr
  xlab("Weeks") +
  xlab("Weeks")+
  ylab("Weekly Sales (Log Transformed)") +
  theme_minimal()
```

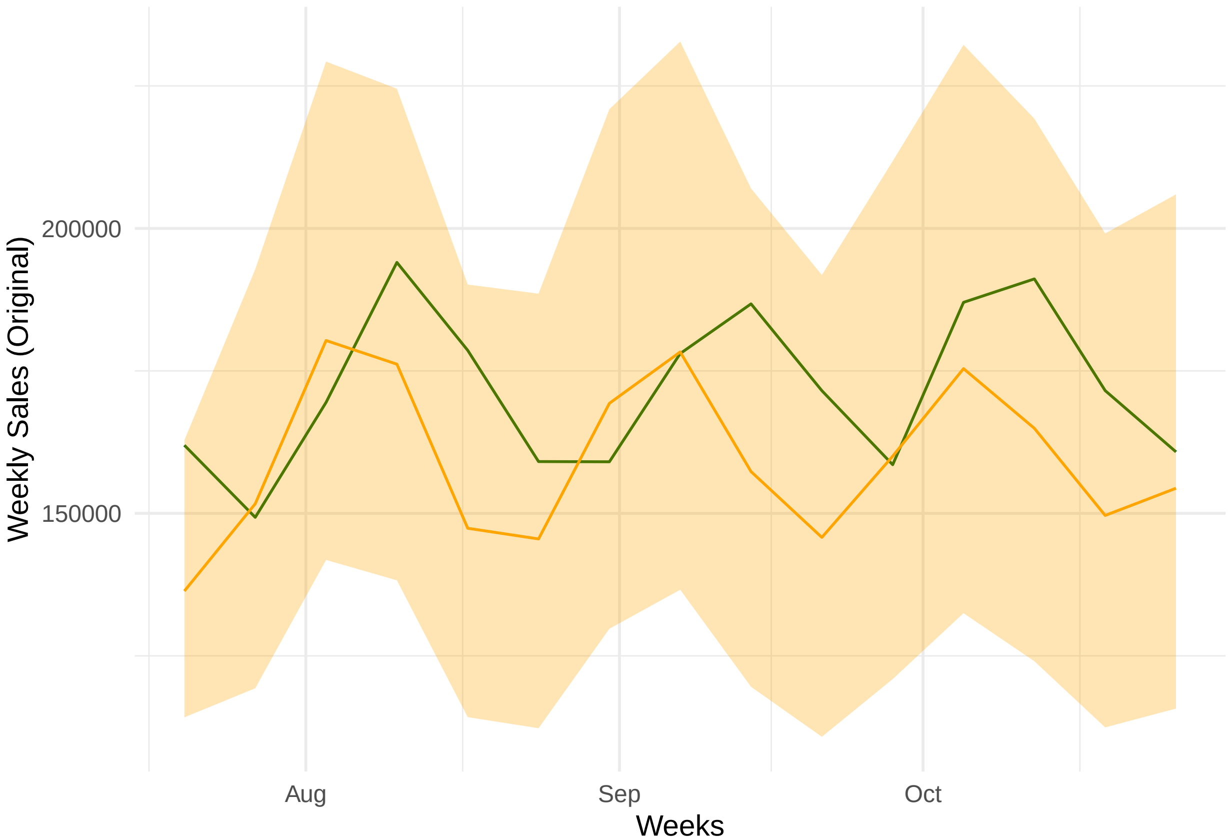
```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```



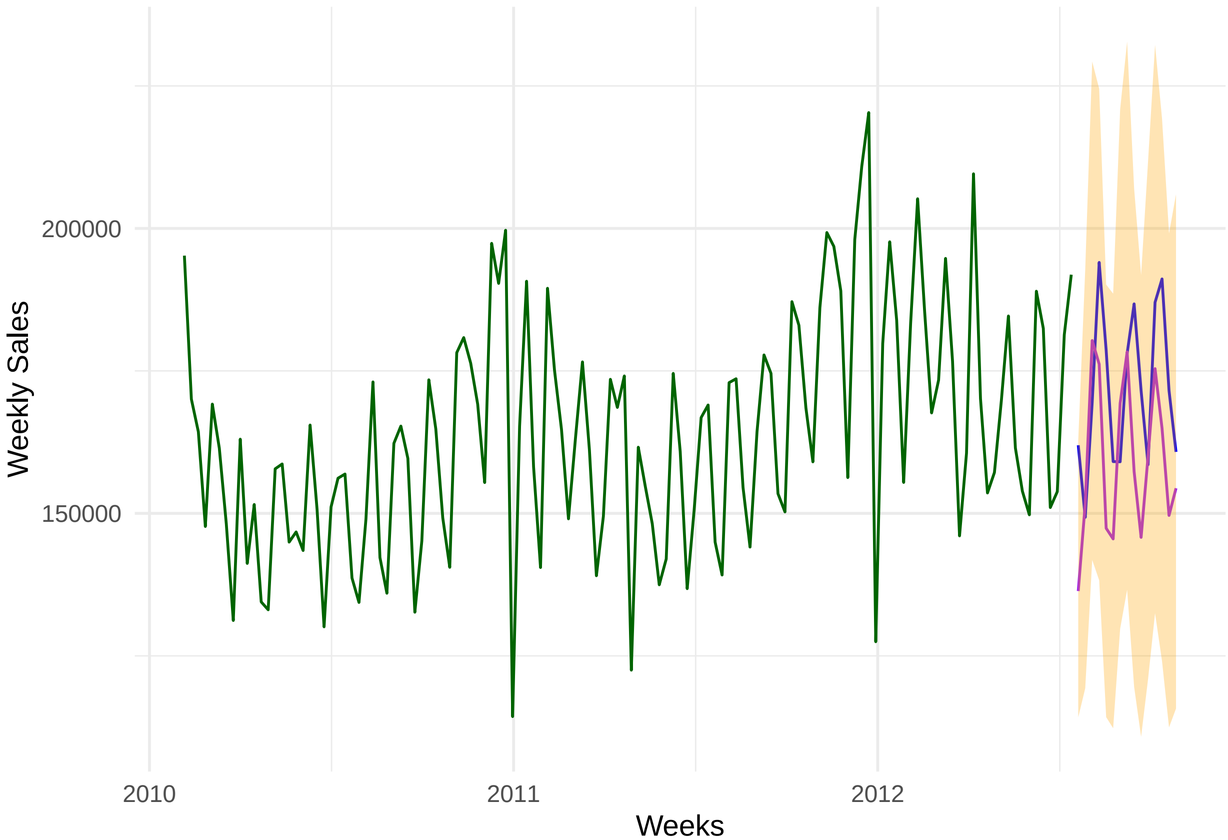

```
# Reverse differencing
previous_observation <- tail(log_datas$log_Weekly_Sales, 1) # Last observed value in the original datas
predicted_original <- predicted_test$pred + previous_observation

original =  exp(predicted_original)
```

```
ci_lower <- exp(predicted_test$pred - 2 * predicted_test$se + previous_observation)
ci_upper <- exp(predicted_test$pred + 2 * predicted_test$se + previous_observation)
```

```
ggplot()+
  geom_line(aes(log_diff_datas$Date,test_data$Weekly_Sales),color = 'darkgreen')+
  geom_line(aes(log_diff_datas$Date,original),color='orange')+
  geom_ribbon(aes(x = log_diff_datas$Date, ymin = ci_lower, ymax = ci_upper), alpha = 0.3, fill = 'orang
  xlab("Weeks") +
  xlab("Weeks")+
  ylab("Weekly Sales (Original)") +
  theme_minimal()
```




```
ggplot()+
  geom_line(aes(log_diff_data$Date,train_data$Weekly_Sales),color = 'darkgreen')+
  geom_line(aes(log_diff_datas$Date, test_data$Weekly_Sales), color = 'blue') +
  geom_line(aes(log_diff_datas$Date,original), color = 'purple') +
  geom_ribbon(aes(x = log_diff_datas$Date, ymin = ci_lower, ymax = ci_upper), alpha = 0.3, fill = 'orang
  xlab("Weeks")+
  ylab("Weekly Sales") +
  theme_minimal()
```



Calculate Mean Absolute Error

```
mae <- mean(abs(test_data$Weekly_Sales)-original)
mae
```

```
## [1] 12283.3
```

Calculate R2 Score

```
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(forecast)
data <- data.frame(actual = test_data$Weekly_Sales, predicted = original)
r_squared <- 1 - sum((test_data$Weekly_Sales - original)^2) / sum((test_data$Weekly_Sales - mean(test_da
r_squared
```

```
## [1] -0.9835209
```

```r
accuracy <- accuracy(original, test_data$Weekly_Sales)
#r2_score <- accuracy$R2
tss <- sum((test_data$Weekly_Sales - mean(test_data$Weekly_Sales))^2)
rss <- sum((test_data$Weekly_Sales - original)^2)
r2_score <- 1 - (rss / tss)
r2_score
```

```
## [1] -0.9835209
```

```r
store_data <- train %>%
  filter(Store == 20)
```

Finding best model for different Stores and different Department

```r
a <- list(20,4,14,13,2)
arima_lists <- list()
for (i in a){
# Select data for one store (e.g., Store 1)
store_data <- train[train$Store == i, ]

# Group the data by department and calculate the total weekly sales
department_sales <- aggregate(Weekly_Sales ~ Dept, data = store_data, FUN = sum)

# Sort the departments by weekly sales in descending order
top_departments <- department_sales[order(department_sales$Weekly_Sales, decreasing = TRUE), ]

# Select the top 5 departments
top_departments <- top_departments[1, ]

# Subset the original data for the selected store and top 5 departments
subset_data <- store_data[store_data$Dept %in% top_departments$Dept, ]

log_data <- subset_data %>%
  mutate(log_Weekly_Sales = log(Weekly_Sales))

log_diff_data <- log_data %>%
  mutate(diff_Weekly_Sales = log_Weekly_Sales- lag(log_Weekly_Sales))

weekly_sales_ts <- ts(log_diff_data$diff_Weekly_Sales, frequency =52)
arima_model <- auto.arima(log_diff_data$diff_Weekly_Sales,
                        stationary=TRUE, seasonal=TRUE,approximation=TRUE,trace= TRUE)
arima_lists[[i]] = arima_model

  # Plot the forecast
#plot(forecast_result, main = colnames(log_diff_data)[i])
}
```

```
##
##  Fitting models using approximations to speed things up...
##
##  ARIMA(2,0,2) with non-zero mean : -227.3883
```

```
##  ARIMA(0,0,0) with non-zero mean : -119.5033
##  ARIMA(1,0,0) with non-zero mean : -125.4804
##  ARIMA(0,0,1) with non-zero mean : -171.2494
##  ARIMA(0,0,0) with zero mean     : -121.568
##  ARIMA(1,0,2) with non-zero mean : -177.9419
##  ARIMA(2,0,1) with non-zero mean : -205.6253
##  ARIMA(3,0,2) with non-zero mean : -244.8544
##  ARIMA(3,0,1) with non-zero mean : -241.226
##  ARIMA(4,0,2) with non-zero mean : -241.5836
##  ARIMA(3,0,3) with non-zero mean : -242.7467
##  ARIMA(2,0,3) with non-zero mean : -235.8184
##  ARIMA(4,0,1) with non-zero mean : -240.5373
##  ARIMA(4,0,3) with non-zero mean : Inf
##  ARIMA(3,0,2) with zero mean     : -247.0929
##  ARIMA(2,0,2) with zero mean     : -229.5846
##  ARIMA(3,0,1) with zero mean     : -243.4117
##  ARIMA(4,0,2) with zero mean     : -243.8581
##  ARIMA(3,0,3) with zero mean     : -245.0189
##  ARIMA(2,0,1) with zero mean     : -207.782
##  ARIMA(2,0,3) with zero mean     : -238.0544
##  ARIMA(4,0,1) with zero mean     : -242.7621
##  ARIMA(4,0,3) with zero mean     : Inf
##
##  Now re-fitting the best model(s) without approximations...
##
##  ARIMA(3,0,2) with zero mean     : -244.349
##
##  Best model: ARIMA(3,0,2) with zero mean
##
##
##  Fitting models using approximations to speed things up...
##
##  ARIMA(2,0,2) with non-zero mean : -254.0498
##  ARIMA(0,0,0) with non-zero mean : -173.4003
##  ARIMA(1,0,0) with non-zero mean : -205.4895
##  ARIMA(0,0,1) with non-zero mean : -239.8445
##  ARIMA(0,0,0) with zero mean     : -175.4638
##  ARIMA(1,0,2) with non-zero mean : -243.0788
##  ARIMA(2,0,1) with non-zero mean : -240.836
##  ARIMA(3,0,2) with non-zero mean : -276.8719
##  ARIMA(3,0,1) with non-zero mean : -275.0446
##  ARIMA(4,0,2) with non-zero mean : -275.1951
##  ARIMA(3,0,3) with non-zero mean : -276.0514
##  ARIMA(2,0,3) with non-zero mean : -261.8005
##  ARIMA(4,0,1) with non-zero mean : Inf
##  ARIMA(4,0,3) with non-zero mean : Inf
##  ARIMA(3,0,2) with zero mean     : -279.0512
##  ARIMA(2,0,2) with zero mean     : -256.222
##  ARIMA(3,0,1) with zero mean     : -277.1428
##  ARIMA(4,0,2) with zero mean     : -277.4036
##  ARIMA(3,0,3) with zero mean     : -278.2859
##  ARIMA(2,0,1) with zero mean     : -242.8465
##  ARIMA(2,0,3) with zero mean     : -263.9729
##  ARIMA(4,0,1) with zero mean     : -275.9146
```

```
## ARIMA(4,0,3) with zero mean     : -275.2214
##
## Now re-fitting the best model(s) without approximations...
##
## ARIMA(3,0,2) with zero mean     : -278.703
##
## Best model: ARIMA(3,0,2) with zero mean
##
##
## Fitting models using approximations to speed things up...
##
## ARIMA(2,0,2) with non-zero mean : -191.4769
## ARIMA(0,0,0) with non-zero mean : -89.28341
## ARIMA(1,0,0) with non-zero mean : -156.8511
## ARIMA(0,0,1) with non-zero mean : -152.6343
## ARIMA(0,0,0) with zero mean     : -91.19869
## ARIMA(1,0,2) with non-zero mean : -191.5384
## ARIMA(0,0,2) with non-zero mean : -153.8846
## ARIMA(1,0,1) with non-zero mean : -191.4922
## ARIMA(1,0,3) with non-zero mean : -190.1515
## ARIMA(0,0,3) with non-zero mean : -153.0439
## ARIMA(2,0,1) with non-zero mean : -191.9101
## ARIMA(2,0,0) with non-zero mean : -166.1016
## ARIMA(3,0,1) with non-zero mean : -206.0149
## ARIMA(3,0,0) with non-zero mean : -204.0873
## ARIMA(4,0,1) with non-zero mean : -202.205
## ARIMA(3,0,2) with non-zero mean : -207.5465
## ARIMA(4,0,2) with non-zero mean : -204.0265
## ARIMA(3,0,3) with non-zero mean : -205.4713
## ARIMA(2,0,3) with non-zero mean : -189.5097
## ARIMA(4,0,3) with non-zero mean : -203.1536
## ARIMA(3,0,2) with zero mean     : -209.019
## ARIMA(2,0,2) with zero mean     : -192.5281
## ARIMA(3,0,1) with zero mean     : -207.0828
## ARIMA(4,0,2) with zero mean     : -205.5381
## ARIMA(3,0,3) with zero mean     : -206.9386
## ARIMA(2,0,1) with zero mean     : -193.0298
## ARIMA(2,0,3) with zero mean     : -190.5483
## ARIMA(4,0,1) with zero mean     : -203.5534
## ARIMA(4,0,3) with zero mean     : -204.7599
##
## Now re-fitting the best model(s) without approximations...
##
## ARIMA(3,0,2) with zero mean     : -192.9187
##
## Best model: ARIMA(3,0,2) with zero mean
##
##
## Fitting models using approximations to speed things up...
##
## ARIMA(2,0,2) with non-zero mean : -264.6944
## ARIMA(0,0,0) with non-zero mean : -199.7421
## ARIMA(1,0,0) with non-zero mean : -234.3772
## ARIMA(0,0,1) with non-zero mean : -264.5985
```

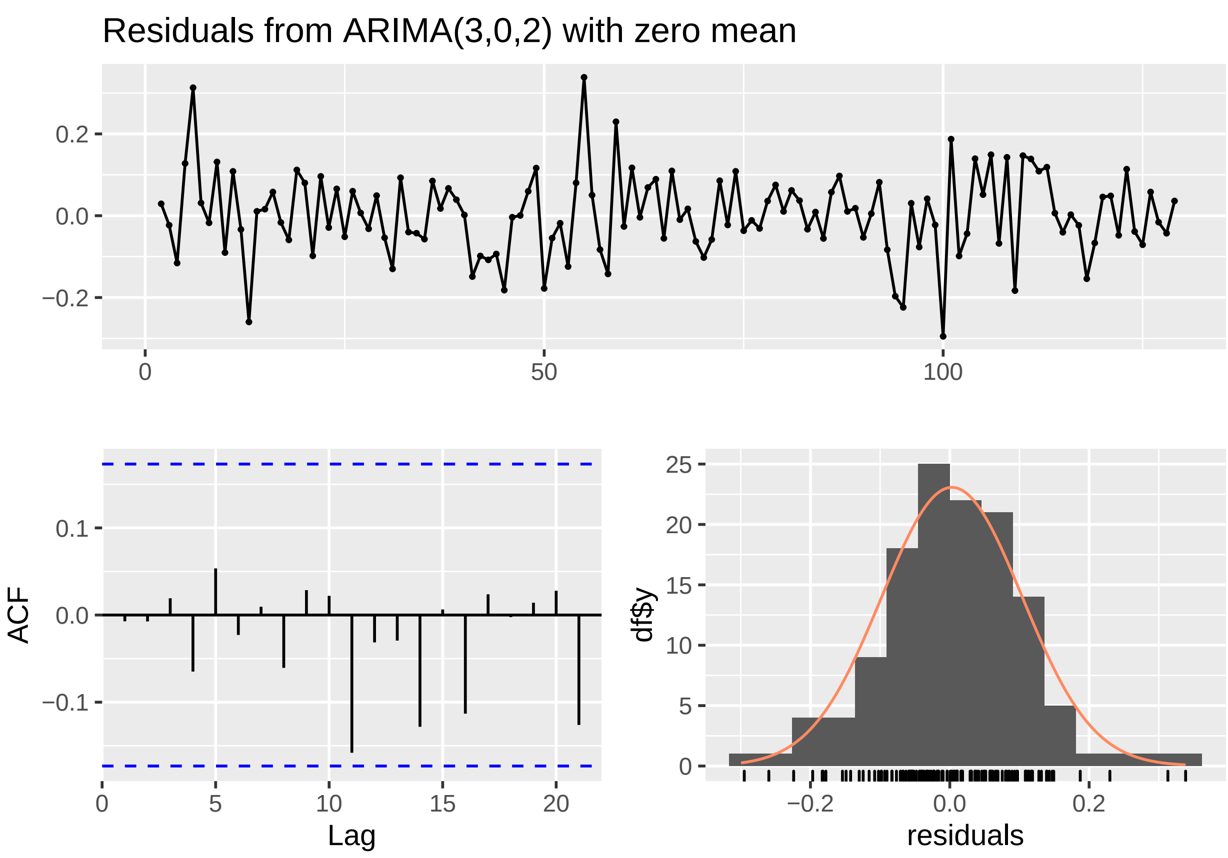
```
##  ARIMA(0,0,0) with zero mean     : -201.8067
##  ARIMA(1,0,2) with non-zero mean : -265.2979
##  ARIMA(0,0,2) with non-zero mean : -264.1474
##  ARIMA(1,0,1) with non-zero mean : -265.6464
##  ARIMA(2,0,1) with non-zero mean : -263.3145
##  ARIMA(2,0,0) with non-zero mean : -241.3964
##  ARIMA(1,0,1) with zero mean     : -267.5235
##  ARIMA(0,0,1) with zero mean     : -266.6217
##  ARIMA(1,0,0) with zero mean     : -236.4613
##  ARIMA(2,0,1) with zero mean     : -265.3133
##  ARIMA(1,0,2) with zero mean     : -267.1907
##  ARIMA(0,0,2) with zero mean     : -266.2271
##  ARIMA(2,0,0) with zero mean     : -243.5062
##  ARIMA(2,0,2) with zero mean     : -266.8647
##
##  Now re-fitting the best model(s) without approximations...
##
##  ARIMA(1,0,1) with zero mean     : -266.4405
##
##  Best model: ARIMA(1,0,1) with zero mean
##
##
##  Fitting models using approximations to speed things up...
##
##  ARIMA(2,0,2) with non-zero mean : -230.6469
##  ARIMA(0,0,0) with non-zero mean : -163.084
##  ARIMA(1,0,0) with non-zero mean : -180.2667
##  ARIMA(0,0,1) with non-zero mean : -214.4135
##  ARIMA(0,0,0) with zero mean     : -165.1469
##  ARIMA(1,0,2) with non-zero mean : -209.0389
##  ARIMA(2,0,1) with non-zero mean : -217.1593
##  ARIMA(3,0,2) with non-zero mean : -239.4233
##  ARIMA(3,0,1) with non-zero mean : -240.7299
##  ARIMA(3,0,0) with non-zero mean : -242.9328
##  ARIMA(2,0,0) with non-zero mean : -195.2025
##  ARIMA(4,0,0) with non-zero mean : -240.1568
##  ARIMA(4,0,1) with non-zero mean : Inf
##  ARIMA(3,0,0) with zero mean     : -245.0944
##  ARIMA(2,0,0) with zero mean     : -197.3316
##  ARIMA(4,0,0) with zero mean     : -242.3606
##  ARIMA(3,0,1) with zero mean     : -242.9276
##  ARIMA(2,0,1) with zero mean     : -219.3249
##  ARIMA(4,0,1) with zero mean     : -241.6836
##
##  Now re-fitting the best model(s) without approximations...
##
##  ARIMA(3,0,0) with zero mean     : -245.4432
##
##  Best model: ARIMA(3,0,0) with zero mean
```

Apply ARIMA(3,0,2) with no Means for different subset of the dataset

```r
arima_model <- arima_model

new_data <- subset(train, Store == 23 & Dept == 42)

log_data <- new_data %>%
  mutate(log_Weekly_Sales = log(Weekly_Sales))

log_diff_data <- log_data %>%
  mutate(diff_Weekly_Sales = log_Weekly_Sales- lag(log_Weekly_Sales))

model <- arima(log_diff_data$diff_Weekly_Sales, order = c(3, 0, 2), include.mean = FALSE)
model
```

```
##
## Call:
## arima(x = log_diff_data$diff_Weekly_Sales, order = c(3, 0, 2), include.mean = FALSE)
##
## Coefficients:
##          ar1      ar2      ar3      ma1     ma2
##       0.5276  -0.4766  -0.1411  -1.0004  0.6753
## s.e.  0.2611   0.1403   0.1288   0.2414  0.2172
##
## sigma^2 estimated as 0.01012:  log likelihood = 112.12,  aic = -212.23
```

```r
checkresiduals(model)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(3,0,2) with zero mean
## Q* = 1.7909, df = 5, p-value = 0.8773
##
## Model df: 5.   Total lags used: 10
```

```r
#plot(actual_values, type = "l", col = "blue", xlim = c(0,200))
#lines(forecasted_values, col = "red")
#legend("topleft", legend = c("Actual", "Forecasted"), col = c("blue", "red"), lty = 1)
```

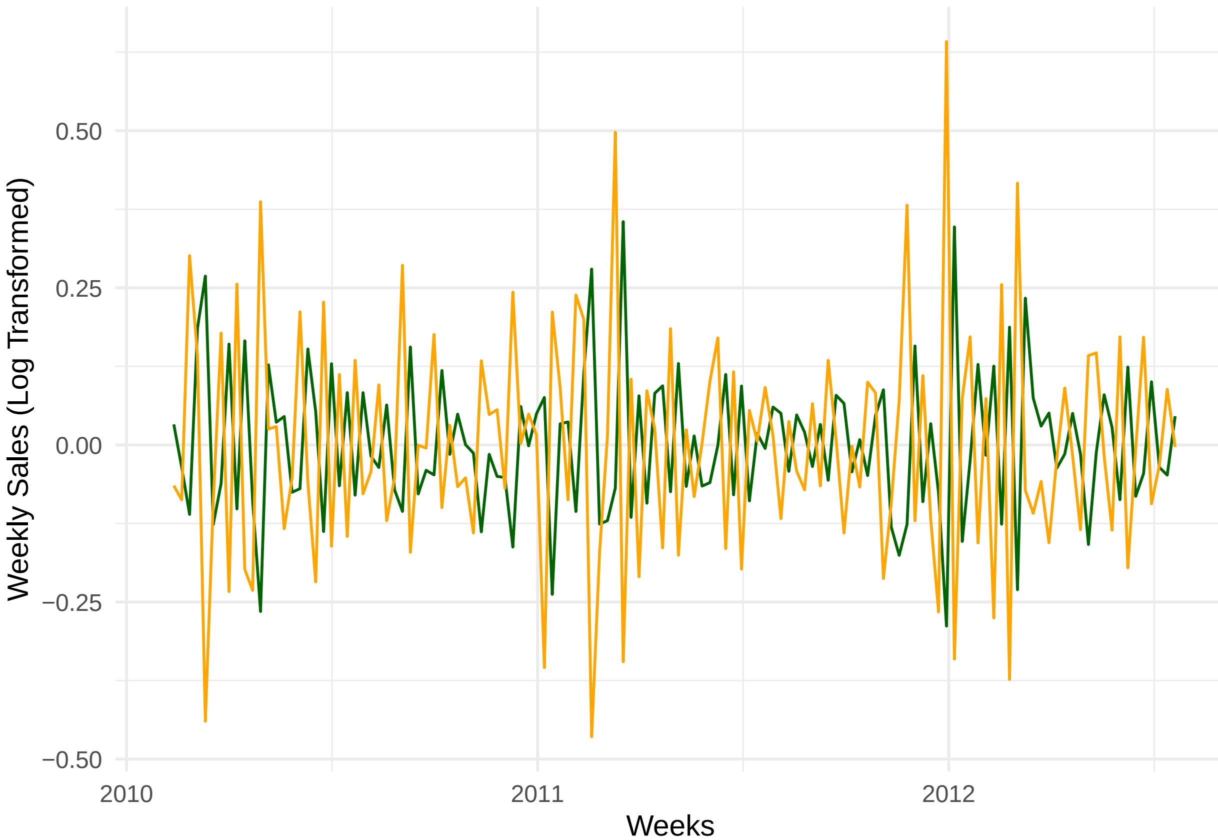
```r
residuals_arima <- residuals(model)
pred = na.omit(log_diff_data$diff_Weekly_Sales) - residuals_arima
```

```
## Warning in `-.default`(na.omit(log_diff_data$diff_Weekly_Sales),
## residuals_arima): longer object length is not a multiple of shorter object
## length
```

```r
ggplot()+
  geom_line(aes(log_diff_data$Date,log_diff_data$diff_Weekly_Sales),color = 'darkgreen')+
  geom_line(aes(log_diff_data$Date,pred),color='orange')+
  #geom_point(aes(log_diff_datas$Date, test_data$diff_Weekly_Sales), color = 'blue') +
  xlab("Weeks")+
  ylab("Weekly Sales (Log Transformed)") +
  theme_minimal()
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```



Weekly Sales (Log Transformed)
0.50
0.25
0.00
−0.25
−0.50
2010
2011
2012
Weeks