Analysis of annual sales of muskrat furs

2023-08-20

```
library(TSA)
library(stats)
library(ecb)
library(dplyr)
library(ggplot2)
library(forecast)
library (tseries)
```

STUDENT: ESPOSITO FRANCESCO
ID: 5108223

Time series analysis of the annual sales of muskrat furs

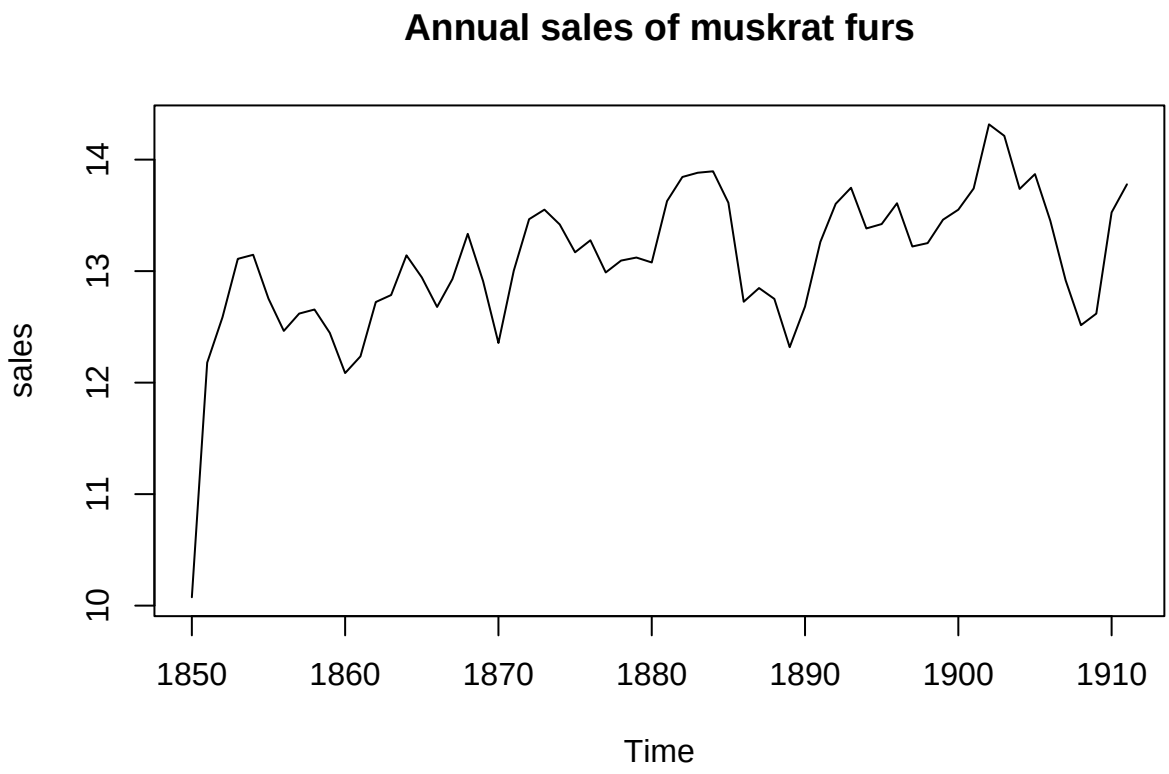
This series we're going to analyze considers the logarithms of the annual sales of muskrat furs by the Hudson's Bay Company for the years 1850-1911, with T=62 annual observations.

The sales of muskrat furs by the Hudson's Bay Company (HBC) can be considered a historical data point in the fur trade of Canada. The Hudson's Bay Company played a significant role in the North American fur trade for centuries.

From the mid-19th to the early 20th centuries, muskrat fur became an essential part of the fur trade, especially as other fur-bearing animals were becoming scarcer due to overhunting. Muskrat furs were used in a variety of goods, including fur coats and hats

Preliminary analysis

```
sales <- c(10.0752, 12.1791, 12.5863, 13.1102, 13.1466, 12.7531, 12.4638, 12.6191,
           12.6556, 12.4461, 12.0855, 12.2357, 12.7230, 12.7857, 13.1417, 12.9441,
           12.6786, 12.9292, 13.3344, 12.9096, 12.3556, 13.0036, 13.4657, 13.5514,
           13.4184, 13.1689, 13.2765, 12.9882, 13.0945, 13.1218, 13.0775, 13.6282,
           13.8444, 13.8824, 13.8953, 13.6134, 12.7252, 12.8483, 12.7509, 12.3177,
           12.6828, 13.2617, 13.6036, 13.7479, 13.3827, 13.4222, 13.6087, 13.2208,
           13.2515, 13.4611, 13.5512, 13.7413, 14.3164, 14.2131, 13.7369, 13.8702,
           13.4518, 12.9177, 12.5151, 12.6188, 13.5267, 13.7784)
sales <- ts(sales, start=c(1850), end = c(1911), frequency=1)
plot(sales, type ='l',main="Annual sales of muskrat furs")
```



As we can see from the first plot of the series, it's quite clear that there is a smooth increasing trend in the logarithm of the annual sales of the muskrat fur.

There could be several reasons that have influenced the (not constant) increase in annual muskrat sales over time. One of the main ones is that many of the animals that used to be considered ''fur-bearing animals" are now in short supply because of overhunting, so they have turned to alternative sources. The development of more sought-after techniques for trapping muskrat could also be an influential factor, increasing the availability of animals, greater production, and thus greater sales.

```
# Calculate Standard Deviation
std_dev <- sd(sales)
cat("Standard Deviation:", std_dev, "\n")
```

```
## Standard Deviation: 0.6465851
```

```
# Calculate Skewness
# install.packages("moments")  # Uncomment if 'moments' is not installed
library(moments)
```

```
##
## Caricamento pacchetto: 'moments'
```

```
## I seguenti oggetti sono mascherati da 'package:TSA':
##
##     kurtosis, skewness
```

```r
skewness_value <- skewness(sales)
cat("Skewness:", skewness_value, "\n")
```

```
## Skewness: -1.518078
```

```r
# Calculate Kurtosis
kurtosis_value <- kurtosis(sales)
cat("Kurtosis:", kurtosis_value, "\n")
```

```
## Kurtosis: 8.877432
```

```r
# Summary Statistics
cat("\nSummary Statistics:\n")
```

```
##
## Summary Statistics:
```

```r
print(summary(sales))
```

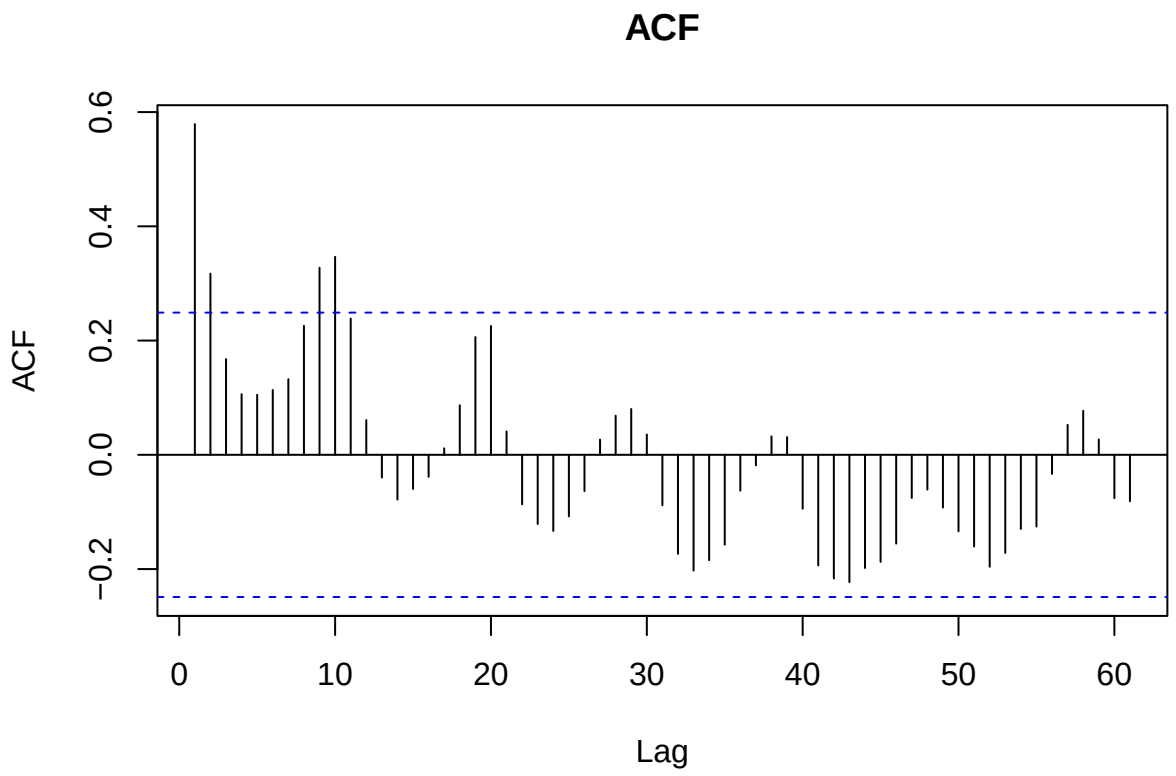
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.08   12.72   13.13   13.09   13.55   14.32
```

Skewness: -1.518078: indicates that the distribution is highly negatively skewed. In simpler terms, the left tail of the distribution is longer or fatter than the right tail. This means there are a significant number of lower values that are pulling the mean down. In the context of 'sales,' this could imply that there are periods (or items, or any other metric depending on what 'sales' signifies) that performed significantly worse compared to the rest.

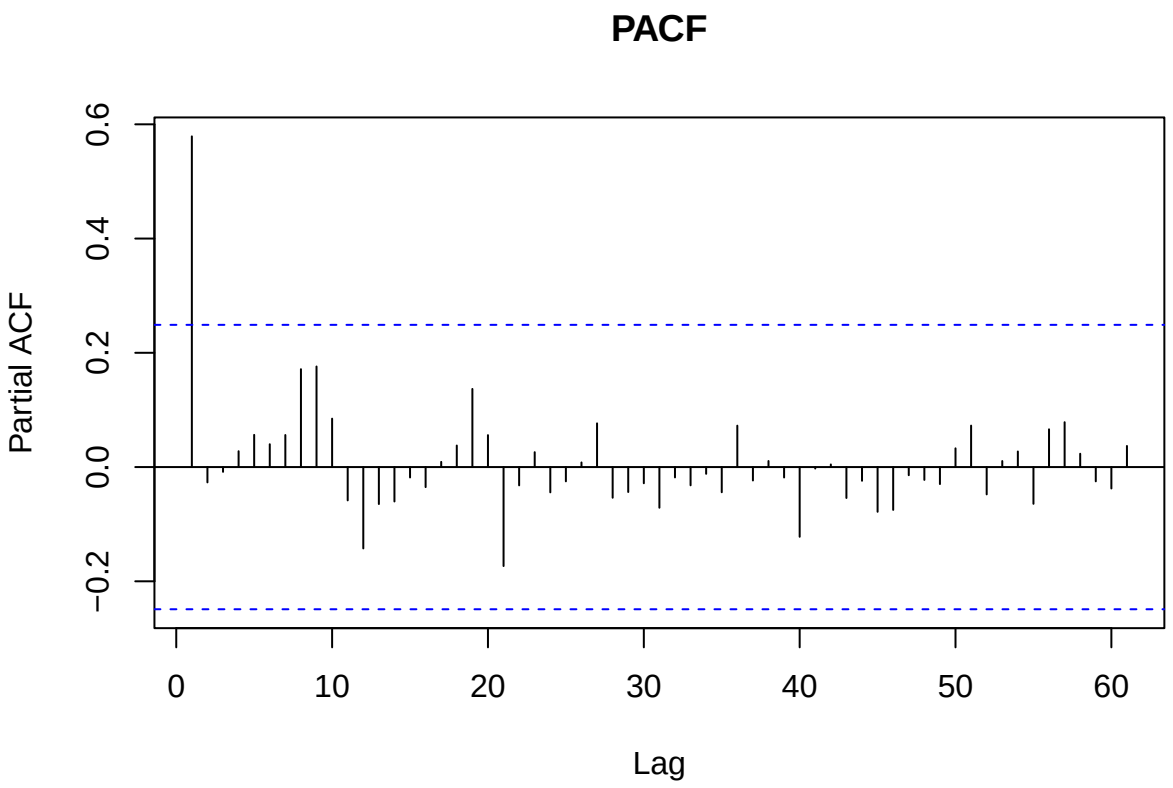
Kurtosis: 8.877432: is greater than 3 (the kurtosis of a normal distribution). This indicates that the distribution has heavier tails and a sharper peak than a normal distribution. A high kurtosis is usually a sign of outliers or extreme values that are significantly different from the majority of data. In a sales context, this might mean that there are times when sales were extraordinarily high or low.

Looking for confirmations:

```r
acf(sales, lag.max=62, main="ACF")
```




```r
pacf(sales,lag.max=62, main="PACF")
```



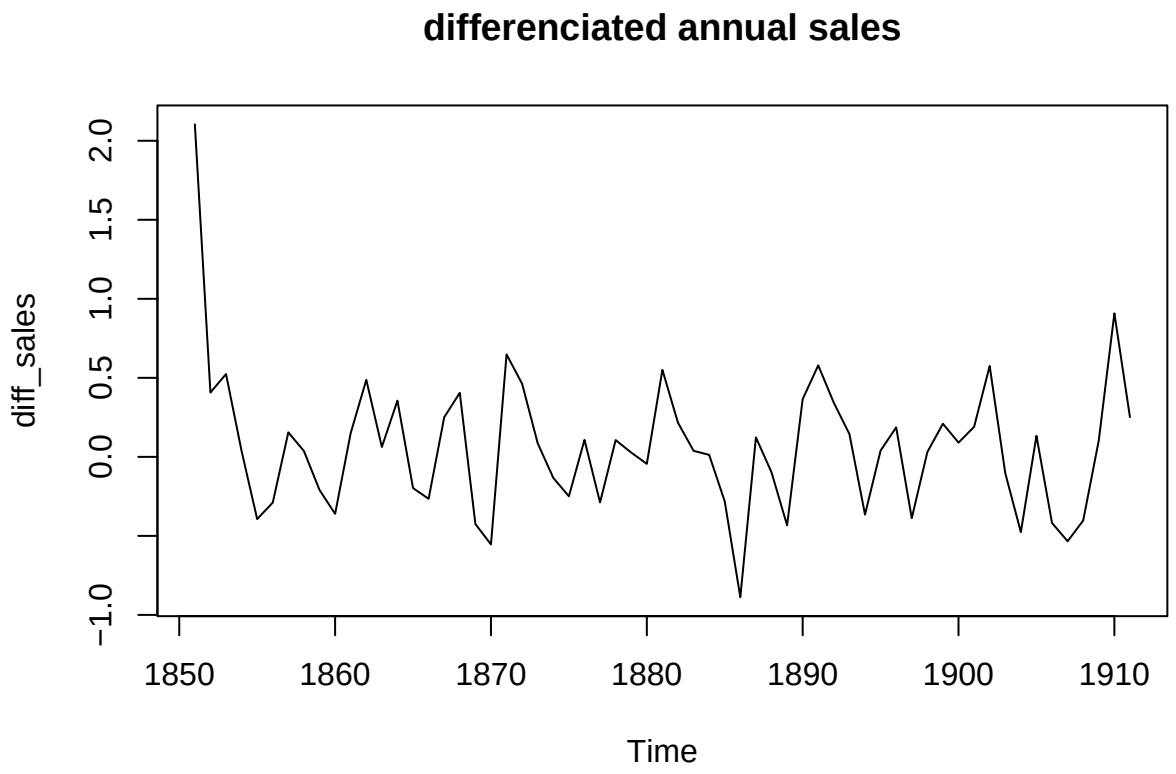
Interpretations of ACF/PACF plot:

1. The first two lags (at lags 1 and 2) indicate that there is a significant correlation between data points at time t and those at times $t-1$ and $t-2$. This is a common observation in time series and suggests some inertia or momentum in the data, where values that are closer together in time are more likely to be similar.
2. The lines at the ninth and tenth positions indicate that there is also a significant correlation between data points separated by 8 and 9 time periods. One might hypothesize that a cycle repeats approximately every 8/9 periods, but it's clear that this is not the case as there's no significance in the subsequent correlations.

Removing linear trend

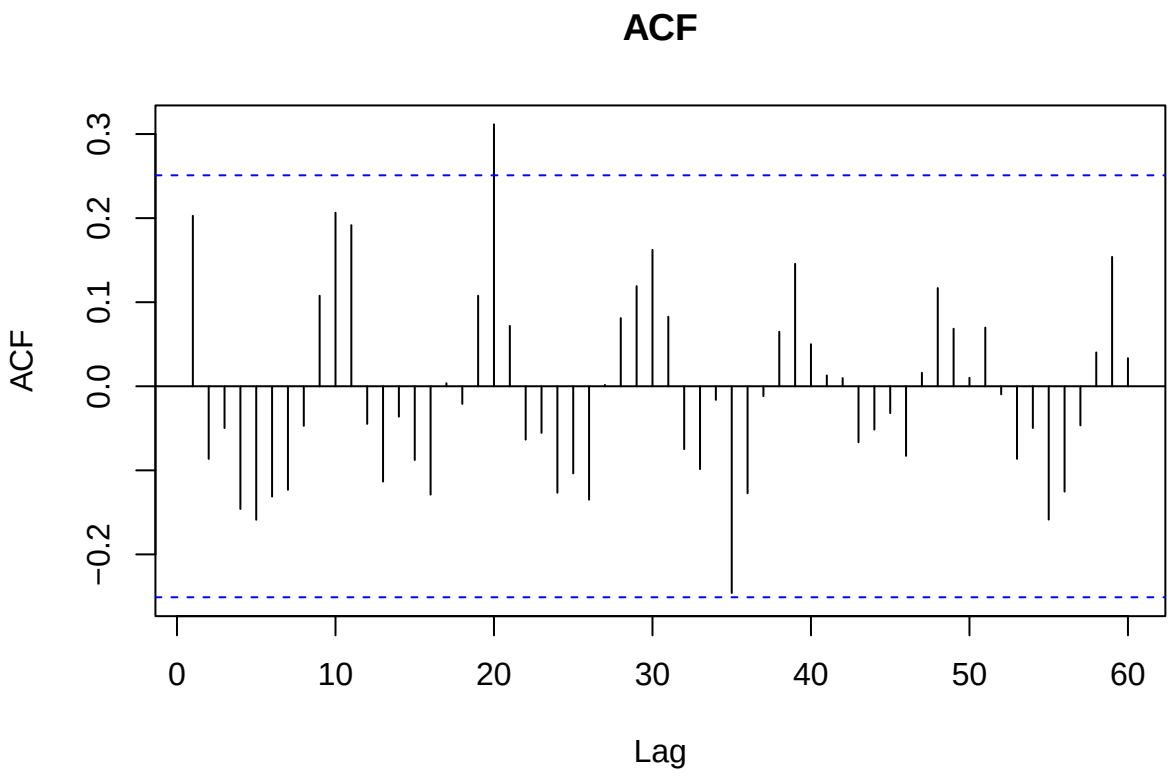
Since the data are already logarithmic transformations of the original data, we try to remove the trend by applying only first-order differentiation.

```
diff_sales <- diff(sales)
plot(diff_sales, type = 'l', main = 'differenciated annual sales')
```

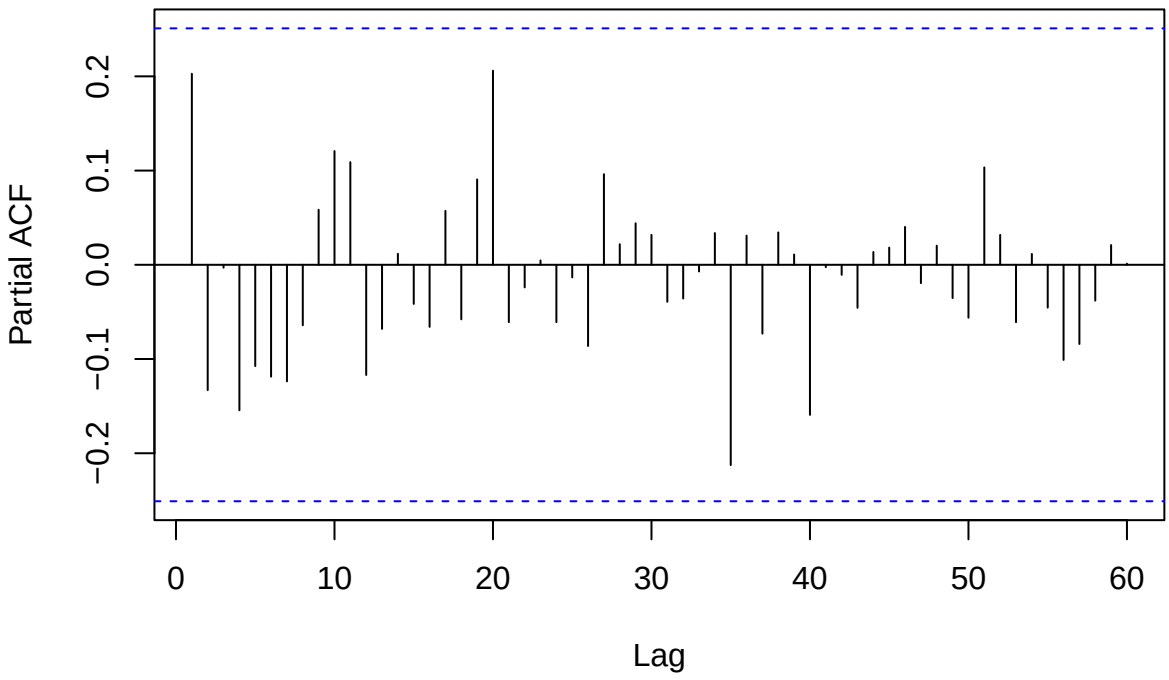


The application of the first-order differentiation on the time series was effective in eliminating the positive trend in the data. Nevertheless, the presence of seasonal patterns may still be present. Therefore, the examination of the autocorrelation function (ACF) and partial autocorrelation function (PACF) of the transformed data will be proceeded with.

```
acf(diff_sales, lag.max=62, main="ACF")
```




```
pacf(diff_sales,lag.max=62, main="PACF")
```



A wavelike trend in the ACF graph would indicatethe presence of a seasonal component of the time series and that it is not adequately modeled.
Regarding the only significant peak in the graph, if the peak at lag 20 is the only significant one and there are no significant peaks at multiples of 20 (such as 40, 60, etc.), then it is likely that the peak represents a unique feature rather than a periodic or seasonal regularity.

Model specification

In order to assess the goodness of fit of our models, statistical measures such as the *Akaike Information Criterion* (AIC) and *Bayesian Information Criterion* (BIC) are used. To facilitate the process, a function that computes the AIC and BIC for various model specifications is implemented.

Below are a table and plots summarizing the results, allowing to identify the model with the lowest values of these information criteria as the preferred choice.

We'll start defining *sales.ma* in order to get the mean adjusted time series (without applying any transformation on the data, because it's already applied).

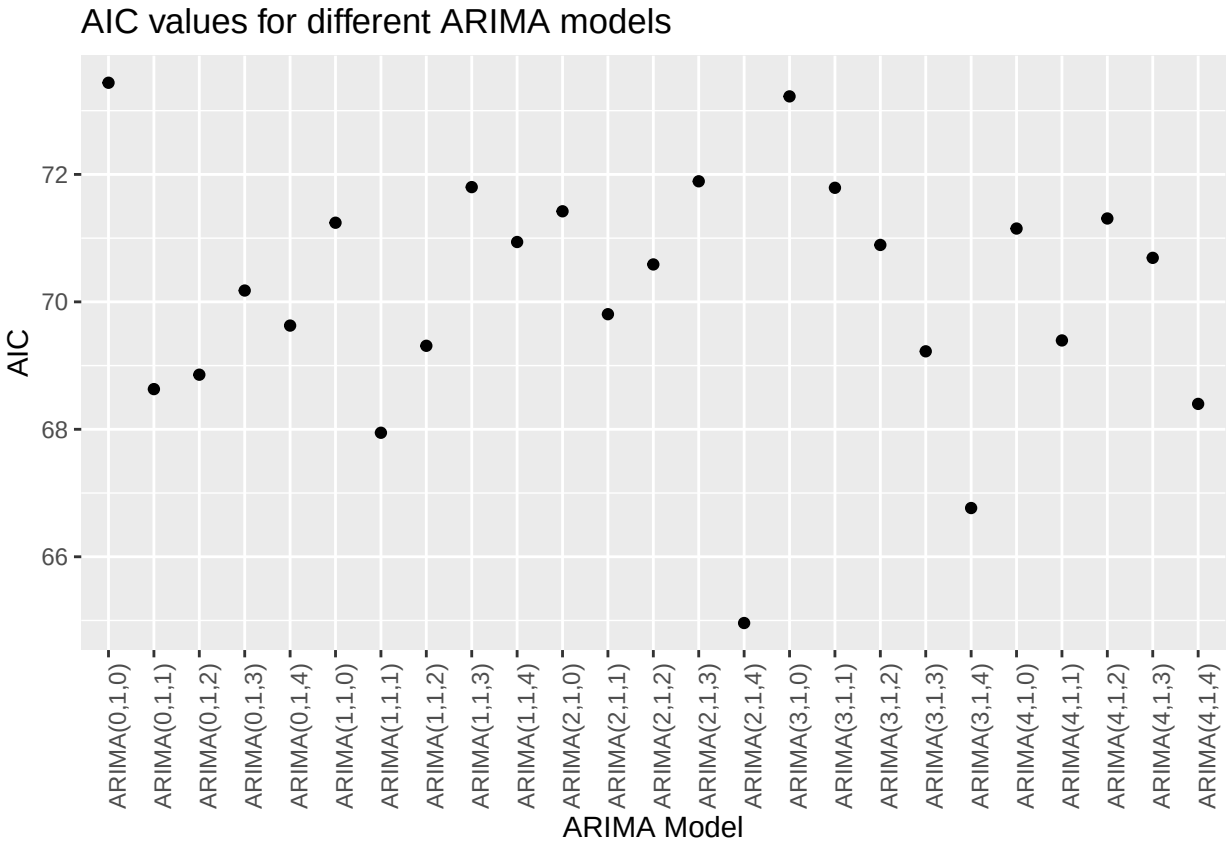
We've seen that a *differentiation (I)* on the data is needed in order to remove the trend, thus the built loop iterates remaining constant the order *d=1*.

```
sales.ma = sales - mean(sales)

P <- 4 #AR order
Q <- 4 #MA order
results <- as.data.frame(matrix(NA, nrow = (P+1) * (Q+1), ncol = 2))
colnames(results) = c("AIC", "BIC")
index = 1
```

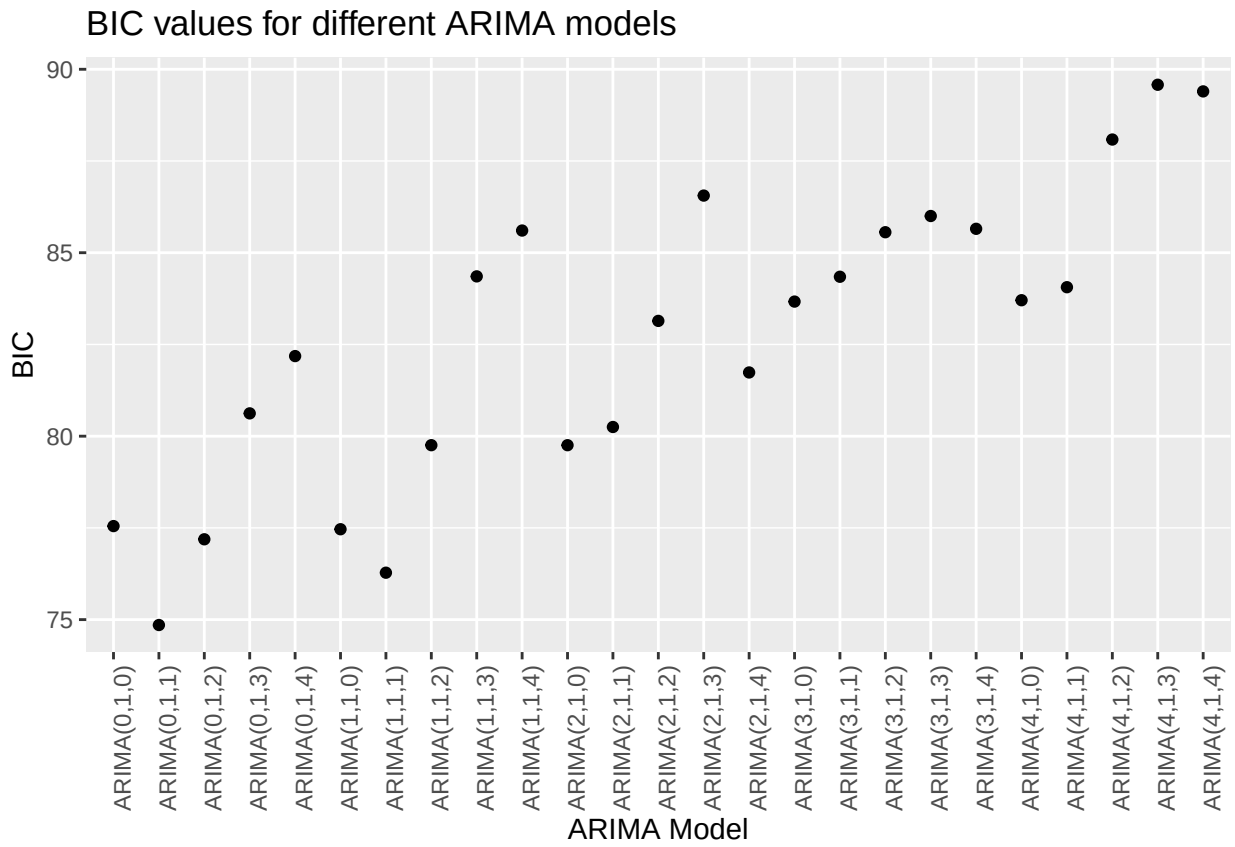
```
for (p  in 0:P) {
  for(q in 0:Q){
    results[index, 1] = arima(sales.ma, order = c(p,1,q), method = "ML")$aic
    results[index,2] = BIC(arima(sales.ma, order = c(p,1,q), method = "ML"))

    rownames(results)[index] = paste("ARIMA(", p, ",1,", q, ")", sep = "")
    index = index + 1
  }
}

knitr::kable(results)
```

	AIC	BIC
ARIMA(0,1,0)	73.43864	77.54951
ARIMA(0,1,1)	68.63086	74.85261
ARIMA(0,1,2)	68.85706	77.18969
ARIMA(0,1,3)	70.17796	80.62146
ARIMA(0,1,4)	69.62789	82.18226
ARIMA(1,1,0)	71.24199	77.46373
ARIMA(1,1,1)	67.94596	76.27859
ARIMA(1,1,2)	69.31053	79.75403
ARIMA(1,1,3)	71.80089	84.35526
ARIMA(1,1,4)	70.93927	85.60452
ARIMA(2,1,0)	71.42137	79.75399
ARIMA(2,1,1)	69.80672	80.25021
ARIMA(2,1,2)	70.58751	83.14188
ARIMA(2,1,3)	71.89218	86.55743
ARIMA(2,1,4)	64.96001	81.73613
ARIMA(3,1,0)	73.22457	83.66806
ARIMA(3,1,1)	71.79028	84.34465
ARIMA(3,1,2)	70.89339	85.55863
ARIMA(3,1,3)	69.22405	86.00017
ARIMA(3,1,4)	66.76463	85.65162
ARIMA(4,1,0)	71.15068	83.70505
ARIMA(4,1,1)	69.39472	84.05996
ARIMA(4,1,2)	71.30862	88.08474
ARIMA(4,1,3)	70.69088	89.57787
ARIMA(4,1,4)	68.39888	89.39675

```
ggplot(results, aes(x = rownames(results), y = AIC)) +
  geom_point() +
  xlab("ARIMA Model") +
  ylab("AIC") +
  ggtitle("AIC values for different ARIMA models") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```




```
ggplot(results, aes(x = rownames(results), y = BIC)) +
  geom_point() +
  geom_line() +
  xlab("ARIMA Model") +
  ylab("BIC") +
  ggtitle("BIC values for different ARIMA models") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



As we can see from the plot, AIC and BIC suggest two different models. In this case, we must decide which of the two criteria fits our case based on their characteristics. We can distinguish the main characteristics thus:

- *AIC*: Aims to select a model that has good prediction ability. It tends to favor more complex models than BIC if they improve the goodness of fit.
- *BIC*: Has a heavier penalty for model complexity and aims to identify the true model (assuming the true model is one of the candidates). BIC can be seen as a criterion that tries to balance accuracy and simplicity, tilting the scale a bit more toward simplicity.

Thus, since our dataset is not large and our goal is to try to make forecasts, AIC fits our case more.

As we can see, the lowest AIC is referred to the model: *ARIMA(2, 1, 4)*.

But there is an observation to be made!

Our loop is performing an exhaustive search on a grid of ARIMA models specified by the maximum values of p, d and q that we provided. This approach is more likely to find a model that strictly minimizes the AIC, at the cost of increasing the complexity of the model (which could lead to overfitting).

To find the right trade-off between complexity and efficiency, again based on AIC, we can use the '*auto.arima*' function.

The *'auto.arima'* function is a sophisticated optimization algorithm that uses a combination of statistical tests, AIC, and other heuristics to select an appropriate model. It is programmed to find a good trade-off between model fit and complexity, which could lead to a less complex model with slightly higher AIC but better generalization properties.

Let's apply it.

```r
# Fit the ARIMA model
model1 <- auto.arima(sales.ma, method="ML")
model1
```

```
## Series: sales.ma
## ARIMA(1,1,1)
##
## Coefficients:
##           ar1     ma1
##       -0.4258  0.9202
## s.e.   0.1749  0.0745
##
## sigma^2 = 0.1697:  log likelihood = -31.97
## AIC=69.95   AICc=70.37   BIC=76.28
```

As we can see, the suggest model by the function is a much simpler model than that suggested by the previously constructed loop.

It maintains the degree *d=1* for *differentiation*, and also provides degrees for AR and MA both open to *1*. All this, generating an AIC value of about 69, which is not significantly larger than the previous *ARIMA(2, 1, 4)* model, which provided AIC value of about 64.95.

An *ARIMA(1, 1, 1)* model can be written as:

$$(1 - \phi_1 L)(1 - L)Y_t = (1 + \theta_1 L)\epsilon_t$$

Where:
- Y_t represents the value of the time series at time t.
- L is the lag operator, so that $LY_t = Y_{t-1}$.
- ϕ_1 is the parameter for the autoregressive component of order 1.
- θ_1 is the parameter for the moving average component of order 1.
- ϵ_t is white noise error at time t with zero mean and constant variance.

Explanation of the Parameters:
- The part $(1 - \phi_1 L)$ represents the AR(1) component. If ϕ_1 is positive, it means the series has a positive correlation with its previous value. If it's negative, it has a negative correlation.
- $(1 - L)$ represents the differencing operation. This makes the series stationary by differencing each point with its previous value: $Y_t - Y_{t-1}$.
- The part $(1 + \theta_1 L)$ represents the MA(1) component. This captures the correlation between the current error and past errors.

Model estimation

In this step we're gonna use the ML method in order to estimate the parameters. Maximum Likelihood (ML) is a commonly used method for estimating the parameters of a statistical model. The idea is to find the set of parameters that maximizes the likelihood of the observed data given the model.

Given a time series dataset $\{Y_1, Y_2, ..., Y_T\}$, and a model (e.g., ARIMA) that is characterized by a set of parameters θ, the likelihood $L(\theta; Y)$ measures how well the model explains the data. Maximum Likelihood Estimation (MLE) aims to find the θ that maximizes this likelihood.

This method provides a unified approach to estimation, which is computationally efficient and asymptotically unbiased. Moreover it gives us a way to compare different models via likelihood ratio tests or information criteria like AIC and BIC.

```
model1
```

```
## Series: sales.ma
## ARIMA(1,1,1)
##
## Coefficients:
##           ar1     ma1
##       -0.4258  0.9202
## s.e.   0.1749  0.0745
##
## sigma^2 = 0.1697:  log likelihood = -31.97
## AIC=69.95   AICc=70.37   BIC=76.28
```

In the given model estimation, the ARIMA model is estimated on the mean adjusted log-transformed time series data using the *maximum likelihood estimation* method.

```
coefs = model1$coef
ar1 = coefs[1]
ma1 = coefs[2]
coefs
```

```
##        ar1        ma1
## -0.4257589  0.9202429
```

ar1 = -0.4258: The coefficient for the autoregressive term of lag 1. A negative value suggests that an increase at lag 1 is followed by a decrease in the next period (or vice versa), providing a kind of "mean-reversion" behavior.
ma1 = 0.9202: The coefficient for the moving average term of lag 1. A high positive value like this indicates that the model takes into account a significant portion of the lag 1 forecast error when modeling the current value.

The obtained estimates of the parameters can be indicated as:

- $\phi_1 = -0.4257589$
- $\theta_1 = 0.9202429$

Model diagnostic

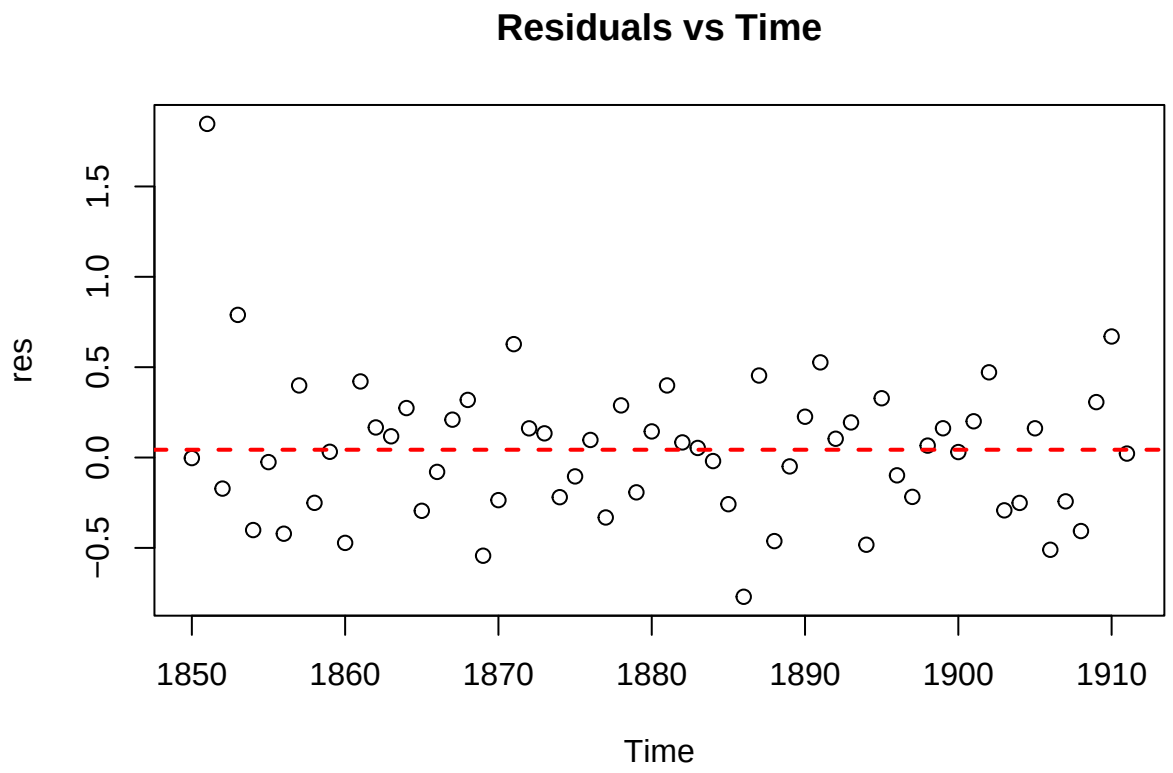
This step has the goal of check the goodness-of-fit of a given model. It's driven by the residual analysis:

$$\text{Residual}_t = Y_t - \hat{Y}_t$$

Adopting the ML method, if the adopted model is suitable for the data and the parameter estimates are close to the true (unknown) value, then the residuals should be: - independent; - identically distributed; - Normal random variables with zero means and σ_a^2 standard deviations.

When the residuals are identically distributed over time, the residuals are not influenced by any particular time points or trends in the time series. in case they aren't identically distributed over time, statistical inferences, such as hypothesis tests or confidence intervals, may be biased or incorrect, and predictions made using the model may not be reliable.

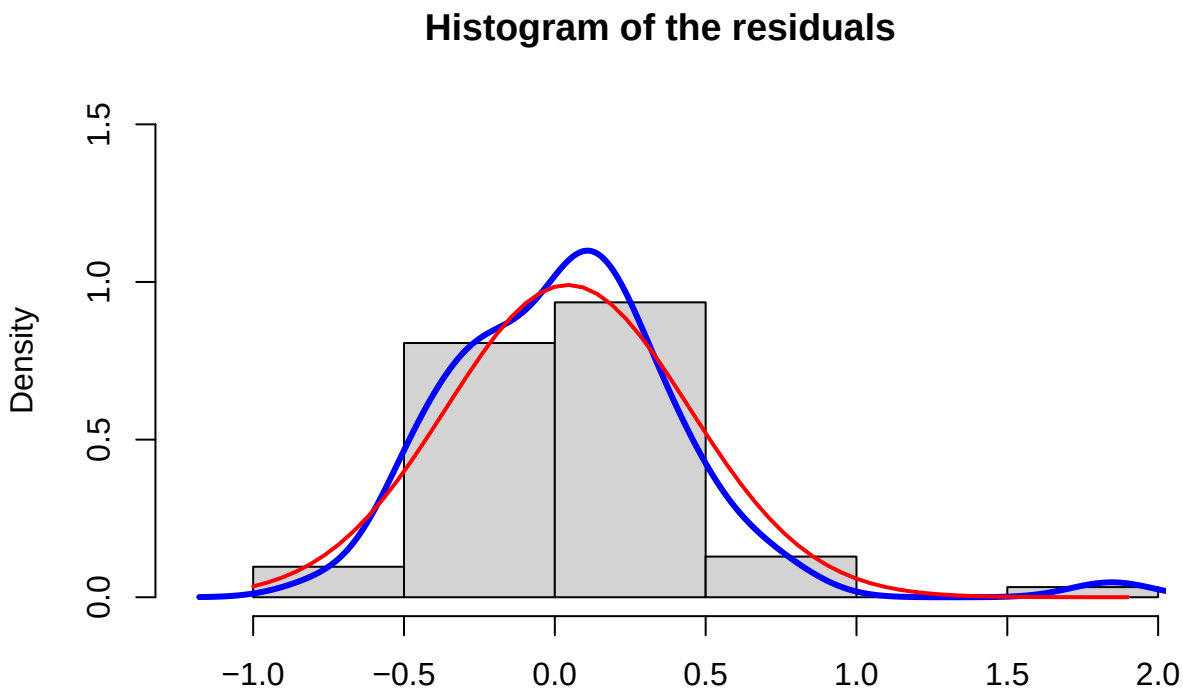
```
res = model1$residuals
plot(res, type = "p", main = "Residuals vs Time")
abline(h = mean(res), col = "red", lwd = 2, lty = 2)
```



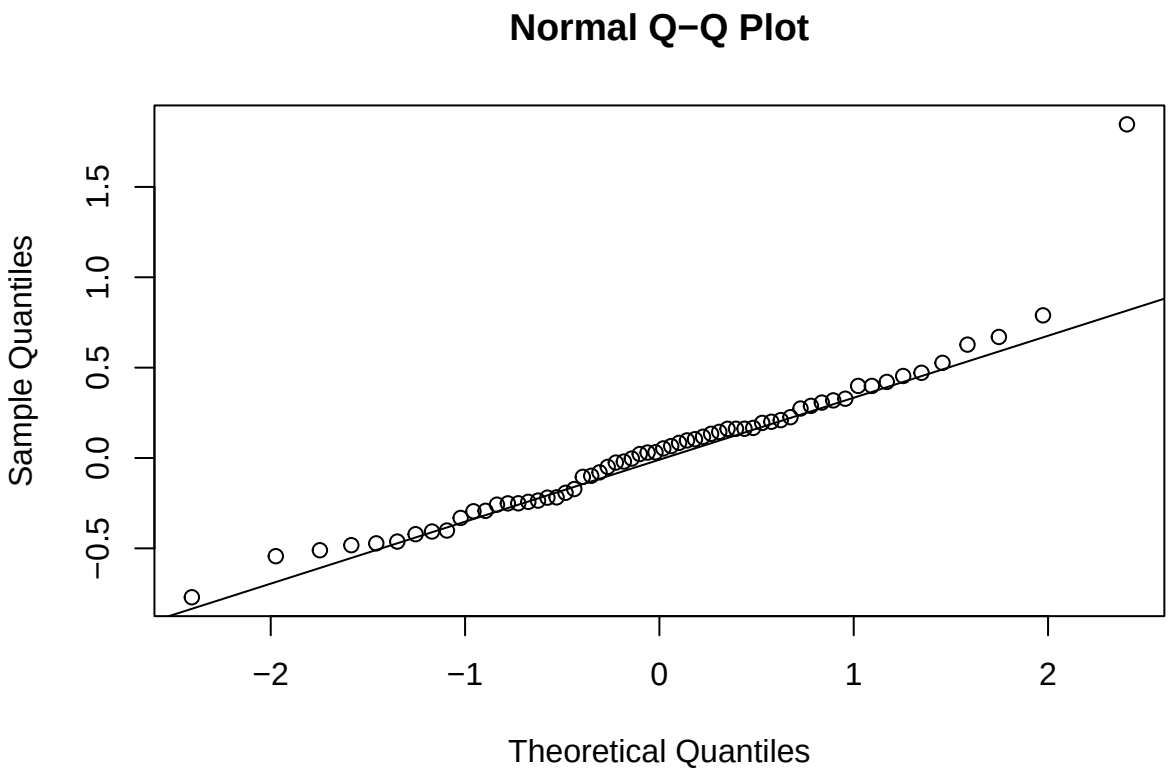
Apparently there appears to be heteroschedastic behavior, but plot visualization alone is not sufficient to firmly express whether the behavior of the residuals reflects a normal distribution or not.

Therefore, let us investigate the analysis further.

```
hist(res,main="Histogram of the residuals",xlab="",freq=F, xlim=c(-1.2,1.90), ylim = c(0, 1.5))
lines(density(res),col="blue",lwd=3)

zz=seq(-1,1.9,length=62)
f.zz=dnorm(zz, mean(res),sd(res))
lines(zz,f.zz,col="red",lwd=2)
```




```
qqnorm(res)
qqline(res)
```



Graphically, the residuals do not appear to behave in a way that is obviously different from the normal distribution. A departure from the Normality behavior is shown at the tails due to the presence of the outliers that are affecting the normality.

In order not to limit ourselves to a graphical interpretation that may be affected by bias, we can make the evaluation objective with a test.

The test is the Shapiro-Wilk test, which is based on comparing the order of the values in our sample with what would be expected from a sample of a normal distribution. This test is generally considered more powerful (i.e., it is more likely to reject the null hypothesis when it is false).

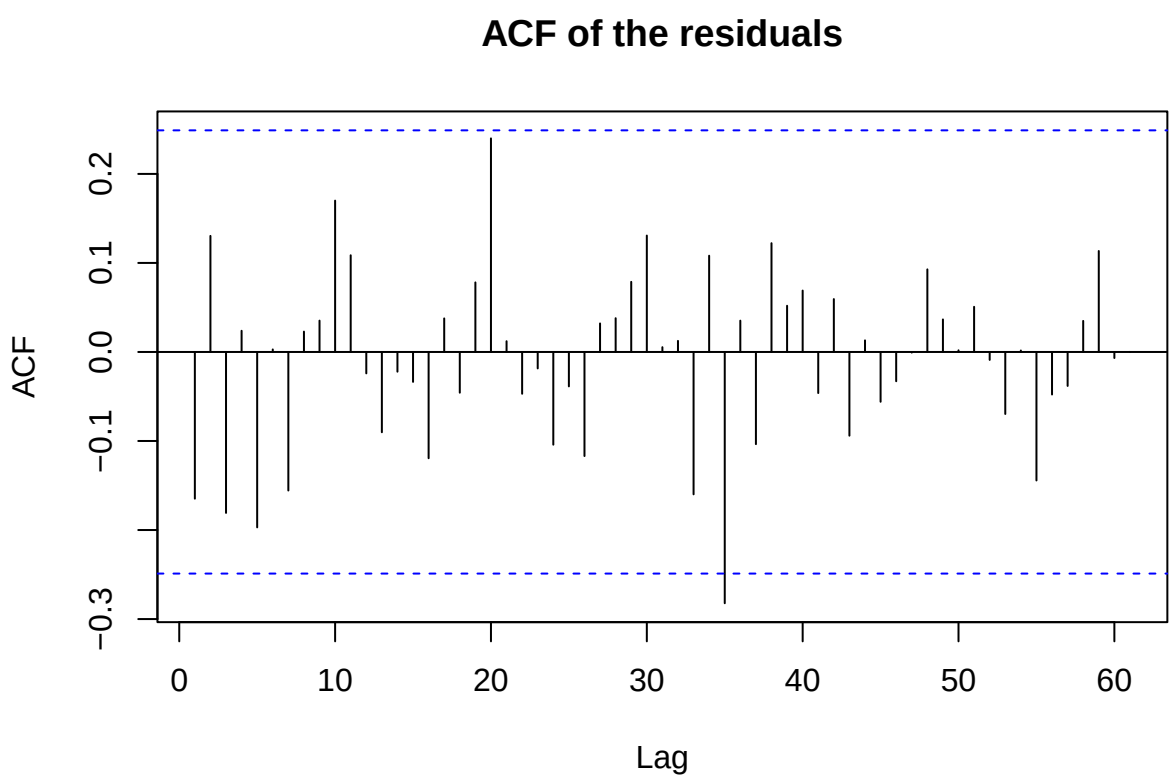
```
shapiro.test(res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.91466, p-value = 0.0003751
```

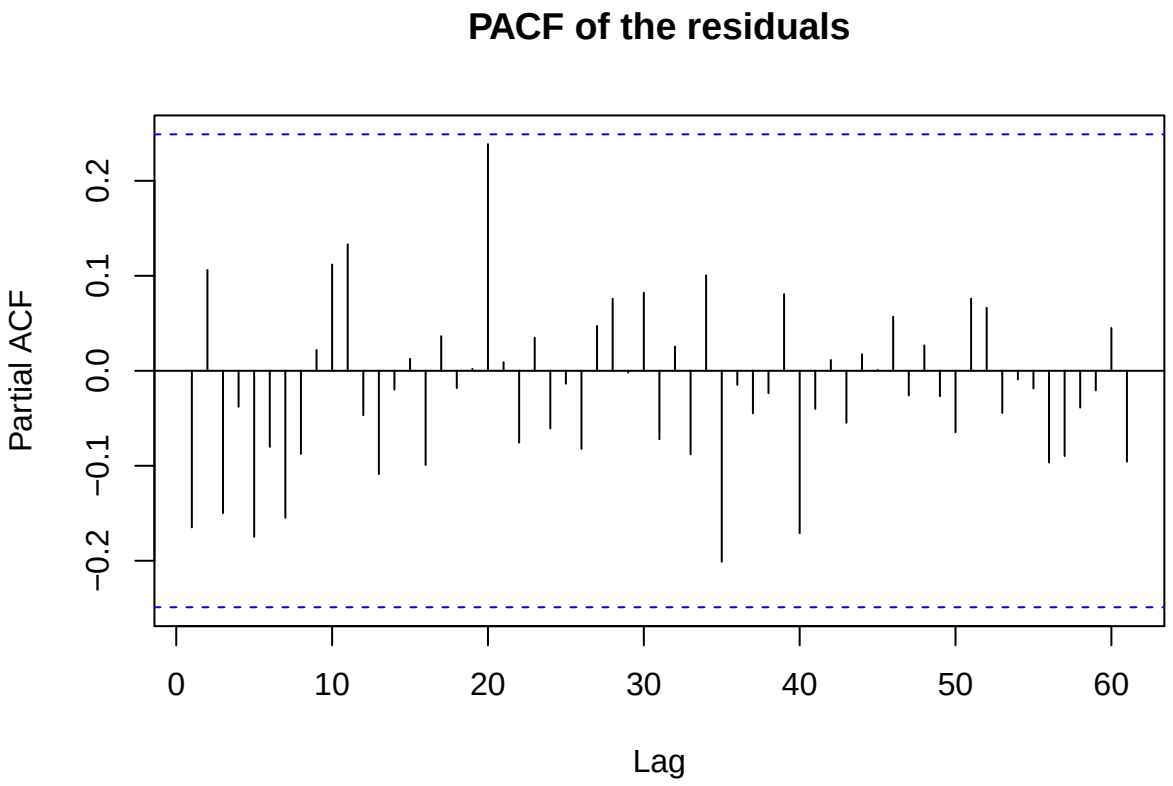
The residuals are Normal if the p-value is above 0.05 and is not the case, cause p-value is above the threshold.

Let's check now the independence of residuals with ACF and PACF:

```
acf(as.numeric(res), lag.max = 63, main = "ACF of the residuals")
```




```
pacf(as.numeric(res), lag.max = 63, main = "PACF of the residuals")
```

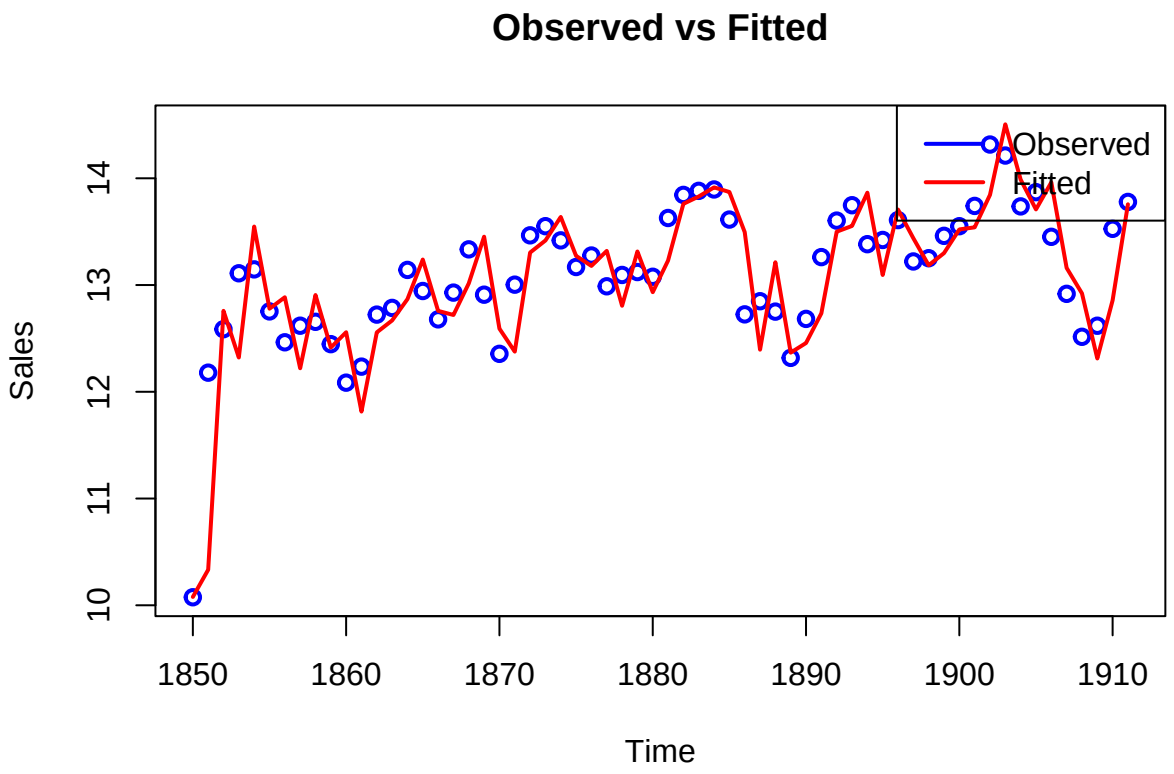


As shown from the plots of ACF and PACF, the residuals are uncorrelated. There is no significant correlations beyond the 35th lag.

Model fitting

Now let's compare the behavior of the fitted values with respect to the observed values:

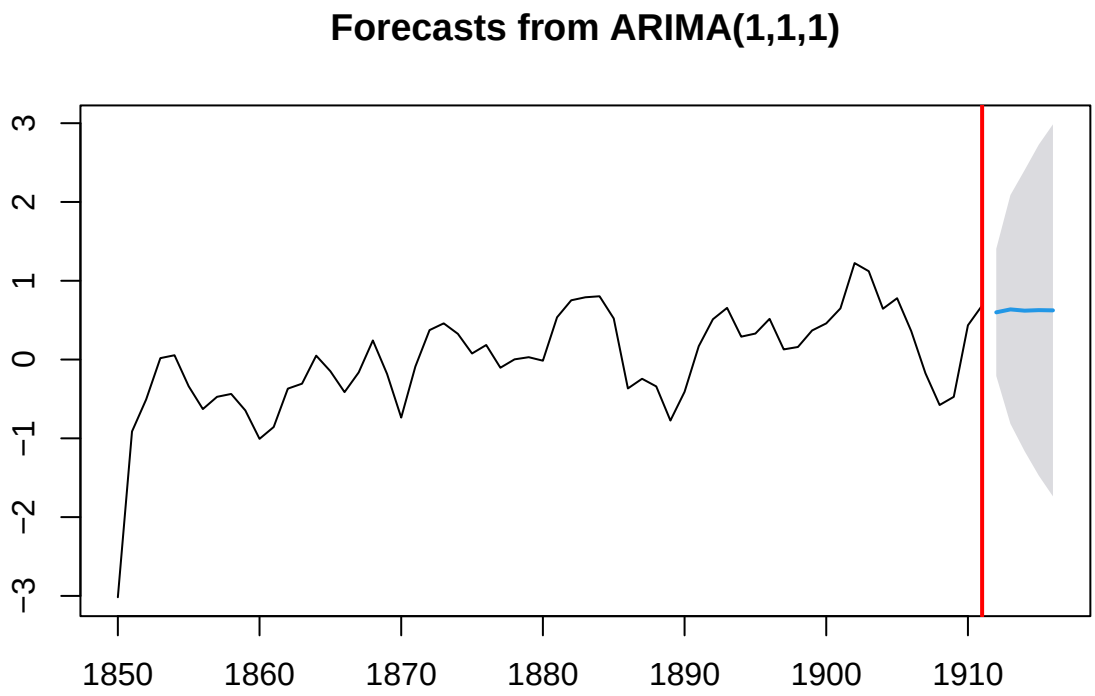
```
# Calculate the fitted values
sales.fit <- sales - res

# Plot the observed vs fitted values
plot(sales, type="p", col="blue", lwd=2, ylim=c(min(sales, sales.fit), max(sales, sales.fit)), ylab="Sa
lines(sales.fit, col="red", lwd=2)
legend("topright", legend=c("Observed", "Fitted"), col=c("blue", "red"), lty=1, lwd=2)
```



The line of fitted values closely follows the line of observed values through the entire time series, suggesting that the ARIMA(1,1,1) model provided a quite good estimate of the actual data. Overall, the goodness of fit between the observed and fitted values suggests that the model is reliable for making predictions.

Forecasting

```
# Generate forecasts, including confidence intervals
sales.for <- forecast(model1, h=5, level=c(95))  # forecasts 95% confidence interval

# Plot forecasts with confidence intervals
plot(sales.for)
abline(v=1911, col="red", lwd=2)
```



It is evident that the predicted values are almost all the same. This can be explained for various reasons like:

- *ARIMA Model*: the ARIMA model can forecast a time series as a "random walk," particularly when the data show a more or less stable trend. An ARIMA(1,1,1) model, as specified in the code, can behave like a random walk if the differencing order (the second parameter, which is 1 in this case) is applied. In a random walk, the best prediction for the next value is the last observed value, which can explain a nearly horizontal forecast line.
- *Nature of the Data*: If the data have low noise and show a constant trend or seasonality, the model might simply forecast a continuation of that trend or seasonality, resulting in a horizontal line.

In essence, the near-constant predicted values might be due to the nature of both the data and the model being used.

Conclusion of the Historical Series Analysis of Muskrat Furs Sales.

The historical series analysis of muskrat furs sales, represented through their logarithms, was conducted using an ARIMA(1, 1, 1) model. Since the dataset was not very large, the choice of a relatively simple model was deemed appropriate to minimize the risk of overfitting.

Implications and Final Considerations. The analysis provides a basis for understanding the temporal dynamics of annual muskrat furs sales, but is limited by the sample size and assumptions of the ARIMA model. The non-normality of the residuals could indicate the presence of factors not captured by the model, which could be an area for further investigation. In addition, the near-constant prediction suggests that it might be useful to explore additional variables or modeling approaches to improve prediction accuracy.