Time Series Analysis - Assignment - 2

Pratham Radhakrishna

3997064

Introduction

In this report, we undertake a comprehensive analysis to propose a set of possible ARIMA(p, d, q) models for a given dataset. The dataset has undergone descriptive analysis to understand its characteristics and identify potential patterns. We leverage various model specification tools covered in Modules 3, 4, and 5, including ACF-PACF plots, EACF (Extended Autocorrelation Function) analysis, and BIC (Bayesian Information Criterion) tables, to explore different combinations of autoregressive (AR) and moving average (MA) parameters, along with the differencing parameter (d) for ARIMA models.

Furthermore, after obtaining the set of possible ARIMA models, we fit each model to estimate the parameters using methods outlined in Module 6. Parameter estimates are crucial for understanding the relationships between past observations and predicting future values. We examine the output of parameter estimates and provide clear and correct comments on their significance and interpretation.

Finally, we employ goodness-of-fit metrics such as AIC (Akaike Information Criterion), BIC, MSE (Mean Squared Error), etc., to evaluate and compare the performance of the fitted models. These metrics aid in selecting the best model among the set of possible models, considering both the model's predictive accuracy and its complexity. Through this comprehensive analysis, we aim to identify the most suitable ARIMA model for the given dataset, facilitating reliable forecasting and decision-making.

Dataset

The dataset under analysis provides yearly Global Land Temperature Anomalies in Degrees Celsius relative to the base period of 1901-2000. Spanning from 1850 to 2023, it offers a comprehensive view of temperature variations over time. These anomalies represent deviations from the average temperature observed during the reference period, indicating whether temperatures in a given year were higher or lower than the long-term average. For instance, a negative anomaly indicates cooler temperatures compared to the 1901-2000 average, while a positive anomaly suggests warmer conditions. This dataset, sourced from NOAA National Centers for Environmental Information, facilitates the study of climate trends and patterns, enabling insights into long-term temperature changes on a global scale.

OBJECTIVE

The objective of this assignment is to analyze the dataset containing yearly Global Land Temperature Anomalies in Degrees Celsius against the base period 1901-2000, covering the years from 1850 to 2023. Leveraging techniques from time series analysis, we aim to:

1. Conduct descriptive analysis to understand the patterns and characteristics of the global land temperature anomalies dataset.
2. Propose a set of possible ARIMA(p, d, q) models using model specification tools such as ACF-PACF plots, EACF, and BIC tables.
3. Fit all proposed ARIMA models to the dataset to estimate their parameters.

4. Evaluate the goodness-of-fit of each model using metrics like AIC, BIC, and MSE.
5. Select the best-fitting ARIMA model for forecasting future global land temperature anomalies.

By achieving these objectives, we aim to provide insights into the long-term trends and variations in global land temperatures, aiding in our understanding of climate change dynamics and facilitating informed decision-making for environmental policies and interventions.

Descriptive Analysis and Model Specification

```
library(TSA)
```

```
## Warning: package 'TSA' was built under R version 4.1.3
```

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

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

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

```
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 4.1.3
```

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

```
#setting the file directory
setwd("C:/Users/Admin/Desktop/time series")
#reading the dataset
dataset <- read.csv("data.csv", header = FALSE)
#removing the first 5 rows
new_data<- dataset[-c(1:5), ]
colnames(new_data) <- c("Year", "Anomaly")
rownames(new_data) <- NULL
#converting it into numeric type
new_data$Anomaly <- as.numeric(new_data$Anomaly)
```

To Adjust the temperature anomalies to be non-negative, as described in the dataset description. Since the anomalies are calculated relative to a reference period and represent differences from the mean temperature during that period, you can shift the anomalies such that they are all positive.

```r
# Shift the anomalies to make them non-negative
min_anomaly <- min(new_data$Anomaly)
shifted_anomalies <- new_data$Anomaly - min_anomaly + 0.01  # Add a small value to avoid zero
```

```r
# Print the updated dataset
head(new_data)
```

```
##   Year Anomaly
## 1 1850   -0.52
## 2 1851   -0.33
## 3 1852   -0.28
## 4 1853   -0.40
## 5 1854   -0.21
## 6 1855   -0.31
```

```r
new_data$Anomaly_shifted <- shifted_anomalies
```

```r
#deleting the previous Anomaly column that has negative value
new_data <- new_data[, -which(colnames(new_data) == "Anomaly")]
```

```r
new_data
```

```
##     Year Anomaly_shifted
## 1   1850            0.33
## 2   1851            0.52
## 3   1852            0.57
## 4   1853            0.45
## 5   1854            0.64
## 6   1855            0.54
## 7   1856            0.55
## 8   1857            0.52
## 9   1858            0.44
## 10  1859            0.59
## 11  1860            0.19
## 12  1861            0.18
## 13  1862            0.01
## 14  1863            0.28
## 15  1864            0.20
## 16  1865            0.41
## 17  1866            0.34
## 18  1867            0.25
## 19  1868            0.36
## 20  1869            0.58
## 21  1870            0.29
## 22  1871            0.28
## 23  1872            0.25
## 24  1873            0.34
## 25  1874            0.37
## 26  1875            0.15
## 27  1876            0.21
## 28  1877            0.66
## 29  1878            0.78
## 30  1879            0.31
## 31  1880            0.32
## 32  1881            0.43
## 33  1882            0.26
## 34  1883            0.31
## 35  1884            0.15
## 36  1885            0.29
## 37  1886            0.31
## 38  1887            0.21
## 39  1888            0.54
## 40  1889            0.71
## 41  1890            0.33
## 42  1891            0.34
## 43  1892            0.33
## 44  1893            0.22
## 45  1894            0.28
## 46  1895            0.39
## 47  1896            0.44
## 48  1897            0.59
## 49  1898            0.43
## 50  1899            0.59
## 51  1900            0.73
## 52  1901            0.78
## 53  1902            0.57
## 54  1903            0.45
```

```
## 55  1904            0.44
## 56  1905            0.52
## 57  1906            0.76
## 58  1907            0.35
## 59  1908            0.40
## 60  1909            0.46
## 61  1910            0.58
## 62  1911            0.44
## 63  1912            0.35
## 64  1913            0.50
## 65  1914            0.73
## 66  1915            0.76
## 67  1916            0.46
## 68  1917            0.19
## 69  1918            0.26
## 70  1919            0.56
## 71  1920            0.58
## 72  1921            0.73
## 73  1922            0.62
## 74  1923            0.57
## 75  1924            0.65
## 76  1925            0.65
## 77  1926            0.85
## 78  1927            0.67
## 79  1928            0.80
## 80  1929            0.41
## 81  1930            0.79
## 82  1931            0.90
## 83  1932            0.85
## 84  1933            0.57
## 85  1934            0.88
## 86  1935            0.73
## 87  1936            0.79
## 88  1937            0.90
## 89  1938            1.10
## 90  1939            0.98
## 91  1940            0.99
## 92  1941            0.96
## 93  1942            0.92
## 94  1943            0.93
## 95  1944            1.10
## 96  1945            0.86
## 97  1946            0.83
## 98  1947            0.99
## 99  1948            0.87
## 100 1949            0.81
## 101 1950            0.63
## 102 1951            0.77
## 103 1952            0.85
## 104 1953            1.06
## 105 1954            0.81
## 106 1955            0.73
## 107 1956            0.53
## 108 1957            0.82
## 109 1958            0.90
## 110 1959            0.96
```

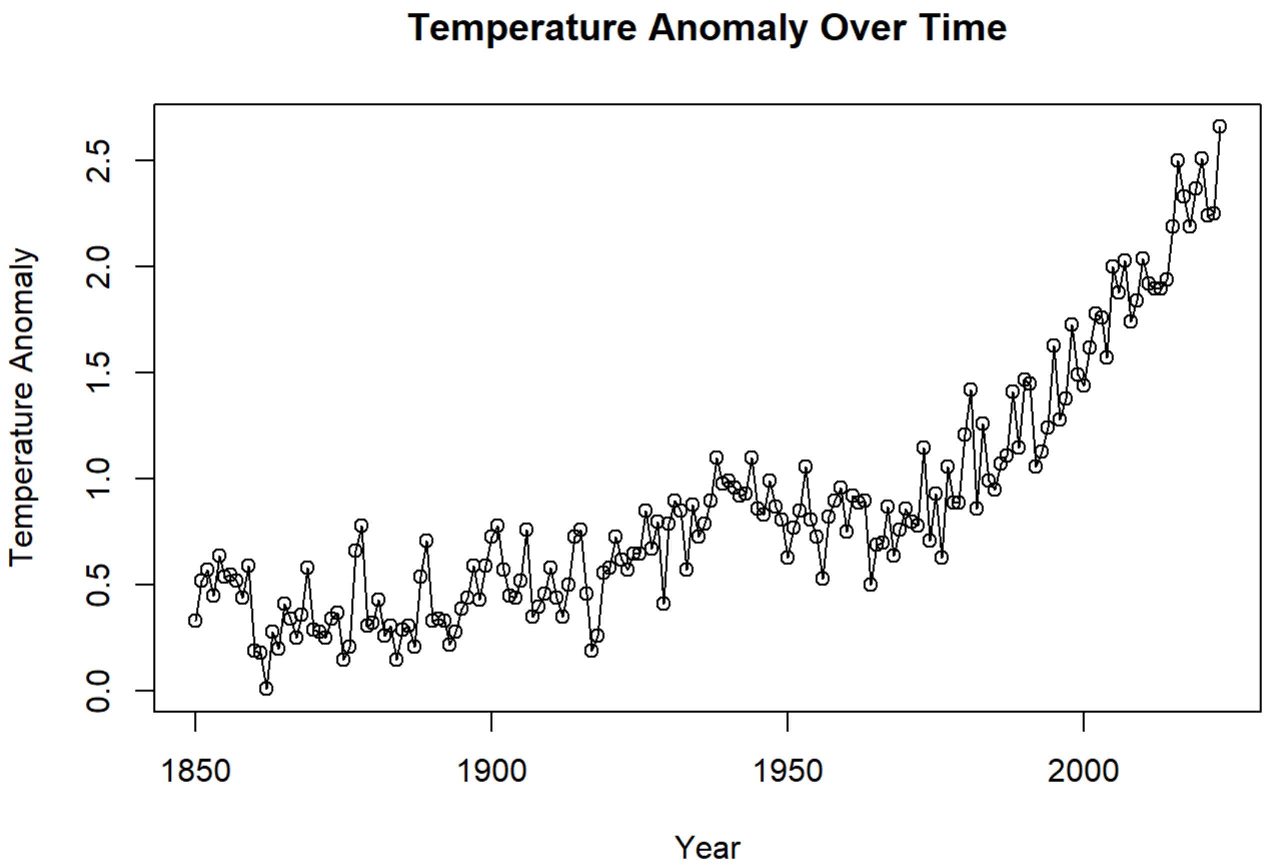
```
## 111 1960            0.75
## 112 1961            0.92
## 113 1962            0.89
## 114 1963            0.90
## 115 1964            0.50
## 116 1965            0.69
## 117 1966            0.70
## 118 1967            0.87
## 119 1968            0.64
## 120 1969            0.76
## 121 1970            0.86
## 122 1971            0.80
## 123 1972            0.78
## 124 1973            1.15
## 125 1974            0.71
## 126 1975            0.93
## 127 1976            0.63
## 128 1977            1.06
## 129 1978            0.89
## 130 1979            0.89
## 131 1980            1.21
## 132 1981            1.42
## 133 1982            0.86
## 134 1983            1.26
## 135 1984            0.99
## 136 1985            0.95
## 137 1986            1.07
## 138 1987            1.11
## 139 1988            1.41
## 140 1989            1.15
## 141 1990            1.47
## 142 1991            1.45
## 143 1992            1.06
## 144 1993            1.13
## 145 1994            1.24
## 146 1995            1.63
## 147 1996            1.28
## 148 1997            1.38
## 149 1998            1.73
## 150 1999            1.49
## 151 2000            1.44
## 152 2001            1.62
## 153 2002            1.78
## 154 2003            1.76
## 155 2004            1.57
## 156 2005            2.00
## 157 2006            1.88
## 158 2007            2.03
## 159 2008            1.74
## 160 2009            1.84
## 161 2010            2.04
## 162 2011            1.92
## 163 2012            1.90
## 164 2013            1.90
## 165 2014            1.94
## 166 2015            2.19
```

```
## 167 2016            2.50
## 168 2017            2.33
## 169 2018            2.19
## 170 2019            2.37
## 171 2020            2.51
## 172 2021            2.24
## 173 2022            2.25
## 174 2023            2.66
```

```
temperature_ts <- ts(new_data$Anomaly_shifted, start = c(1850), end = c(2023), frequency = 1)
temperature_ts
```

```
## Time Series:
## Start = 1850
## End = 2023
## Frequency = 1
##   [1] 0.33 0.52 0.57 0.45 0.64 0.54 0.55 0.52 0.44 0.59 0.19 0.18 0.01 0.28 0.20
##  [16] 0.41 0.34 0.25 0.36 0.58 0.29 0.28 0.25 0.34 0.37 0.15 0.21 0.66 0.78 0.31
##  [31] 0.32 0.43 0.26 0.31 0.15 0.29 0.31 0.21 0.54 0.71 0.33 0.34 0.33 0.22 0.28
##  [46] 0.39 0.44 0.59 0.43 0.59 0.73 0.78 0.57 0.45 0.44 0.52 0.76 0.35 0.40 0.46
##  [61] 0.58 0.44 0.35 0.50 0.73 0.76 0.46 0.19 0.26 0.56 0.58 0.73 0.62 0.57 0.65
##  [76] 0.65 0.85 0.67 0.80 0.41 0.79 0.90 0.85 0.57 0.88 0.73 0.79 0.90 1.10 0.98
##  [91] 0.99 0.96 0.92 0.93 1.10 0.86 0.83 0.99 0.87 0.81 0.63 0.77 0.85 1.06 0.81
## [106] 0.73 0.53 0.82 0.90 0.96 0.75 0.92 0.89 0.90 0.50 0.69 0.70 0.87 0.64 0.76
## [121] 0.86 0.80 0.78 1.15 0.71 0.93 0.63 1.06 0.89 0.89 1.21 1.42 0.86 1.26 0.99
## [136] 0.95 1.07 1.11 1.41 1.15 1.47 1.45 1.06 1.13 1.24 1.63 1.28 1.38 1.73 1.49
## [151] 1.44 1.62 1.78 1.76 1.57 2.00 1.88 2.03 1.74 1.84 2.04 1.92 1.90 1.90 1.94
## [166] 2.19 2.50 2.33 2.19 2.37 2.51 2.24 2.25 2.66
```

```
plot(temperature_ts, type = "o", xlab = "Year", ylab = "Temperature Anomaly", main = "Tempera
ture Anomaly Over Time")
```



After converting the data to time series format , we generated a time series plot and analyzed the following :

- Trend :We can see a clear upward trend.

- Seasonality: A cyclic pattern is seen, we can say that there is seasonality

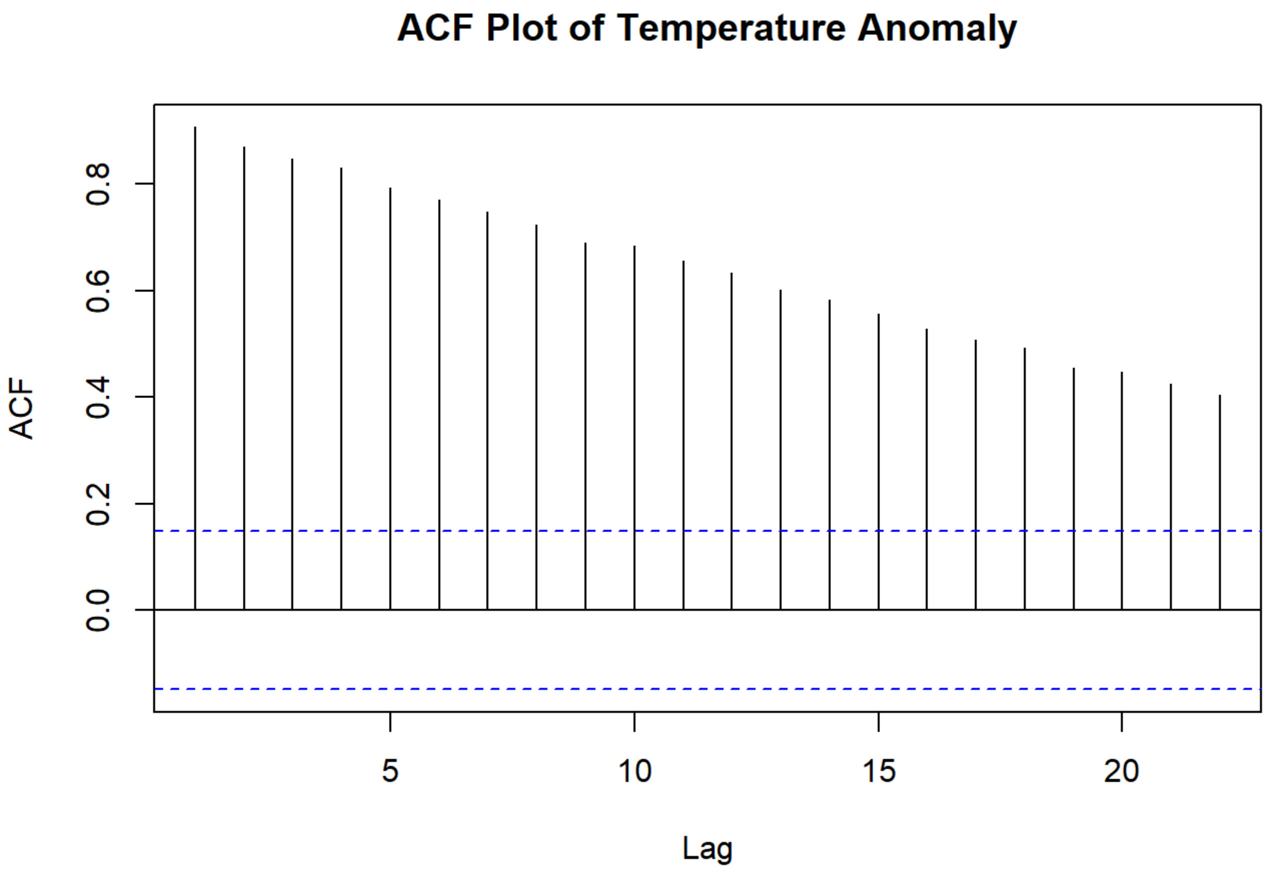
- Variance : no change in variance

- Behavior :Series appeared to be auto regressive due to multiple succeeding patterns.

Stationarity and Normality Checks

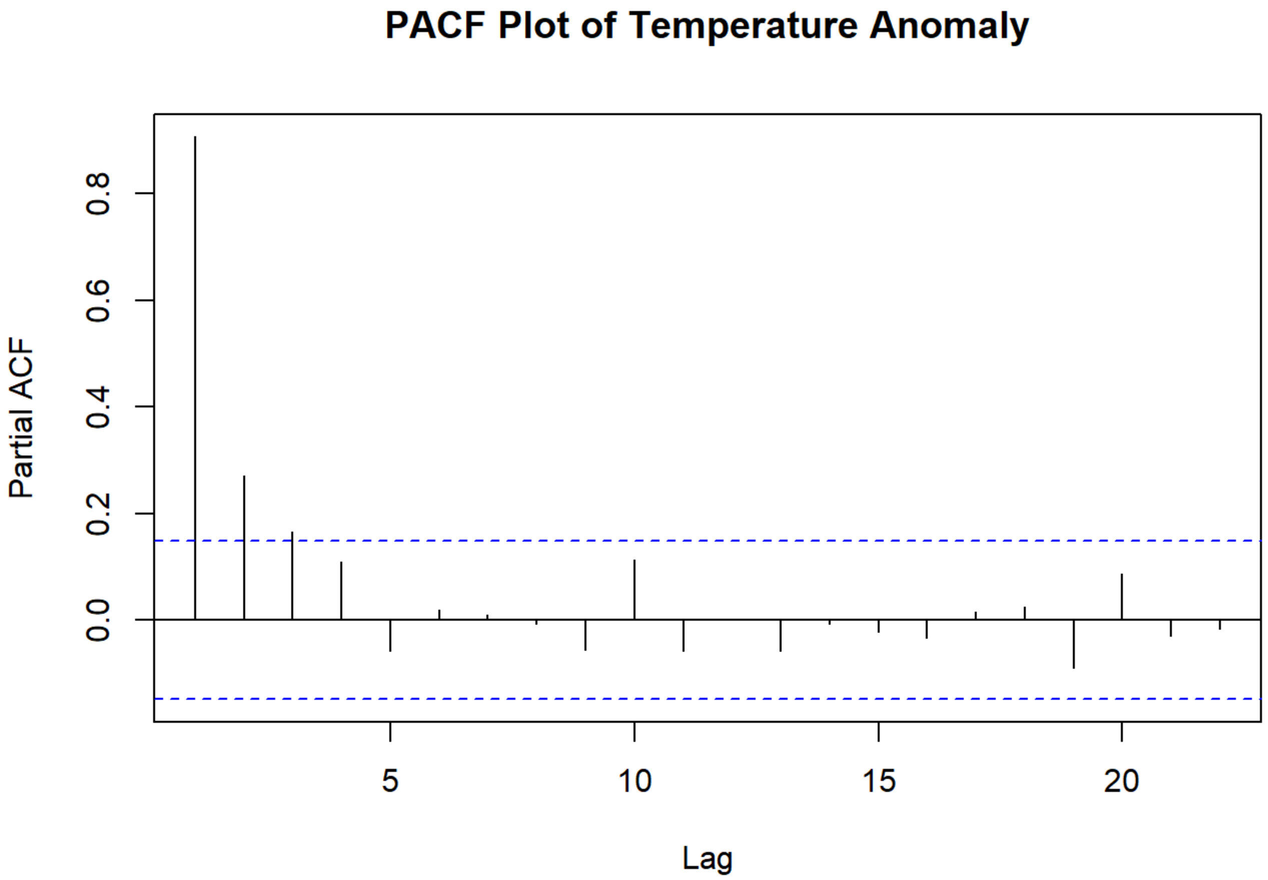
ACF and PACF Plots

```
acf(temperature_ts, main = "ACF Plot of Temperature Anomaly")
```



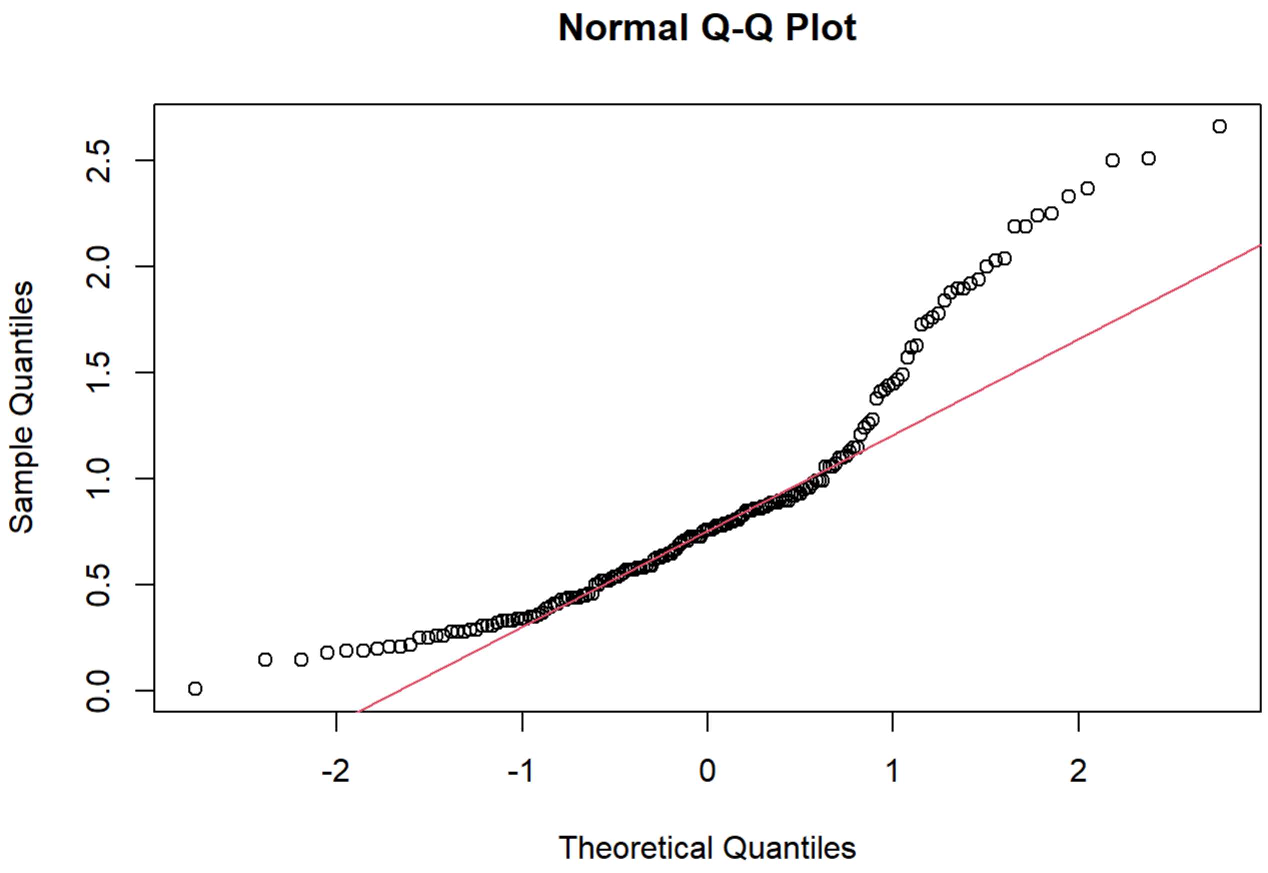
As we can see there is a decreasing trend or pattern observed in the above ACF plot which means it is a non stationary graph.

```
pacf(temperature_ts,main = "PACF Plot of Temperature Anomaly")
```



The first significant lag is very close to 1 which means that the above PACF plot is a non stationary graph.

```
qqnorm(temperature_ts)
qqline(temperature_ts, col = 2)
```



We can see from the plot that the points within -1 and 1 align with the straight line . It appears that the points outside this region seems to deviate from the straight line . We can say that the data is not normally distributed .

ADF Test

```
adf.test(temperature_ts)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  temperature_ts
## Dickey-Fuller = -0.84604, Lag order = 5, p-value = 0.9557
## alternative hypothesis: stationary
```

Shapiro Wilk Test

```
shapiro.test(temperature_ts)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  temperature_ts
## W = 0.8898, p-value = 4.675e-10
```

Since the p-value (0.9557) is much higher than the significance level (commonly 0.05), we fail to reject the null hypothesis. Therefore, we do not have enough evidence to reject the null hypothesis of non-stationarity. However, the test statistic is not very negative, indicating that the data might be non-stationary.

- fFrom the shapiro wilk since the p-value is extremely low (close to zero), we reject the null hypothesis of normality. Therefore, the data is not normally distributed.

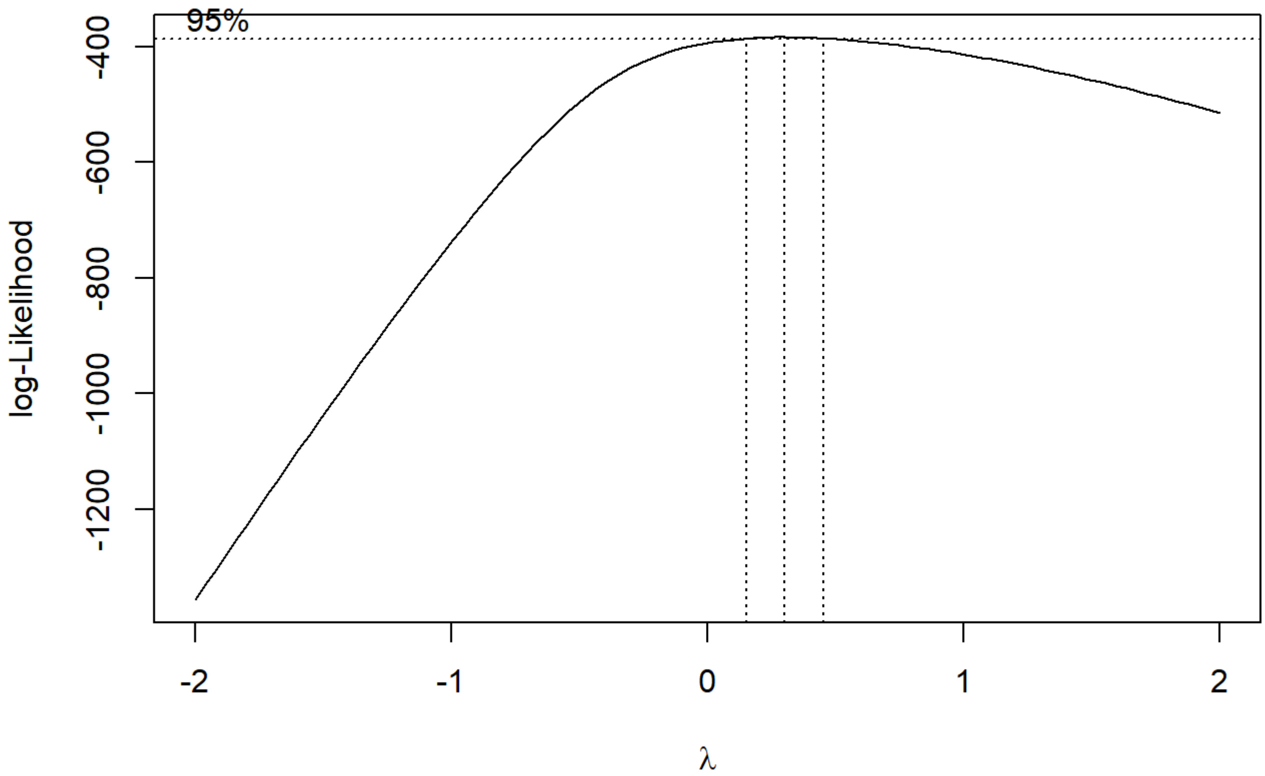
These results suggest that further analysis or transformation might be needed before applying a time series model. Non-stationary data might require differencing to achieve stationarity, and non-normality might indicate the need for transformation or different model selection.

Transformation

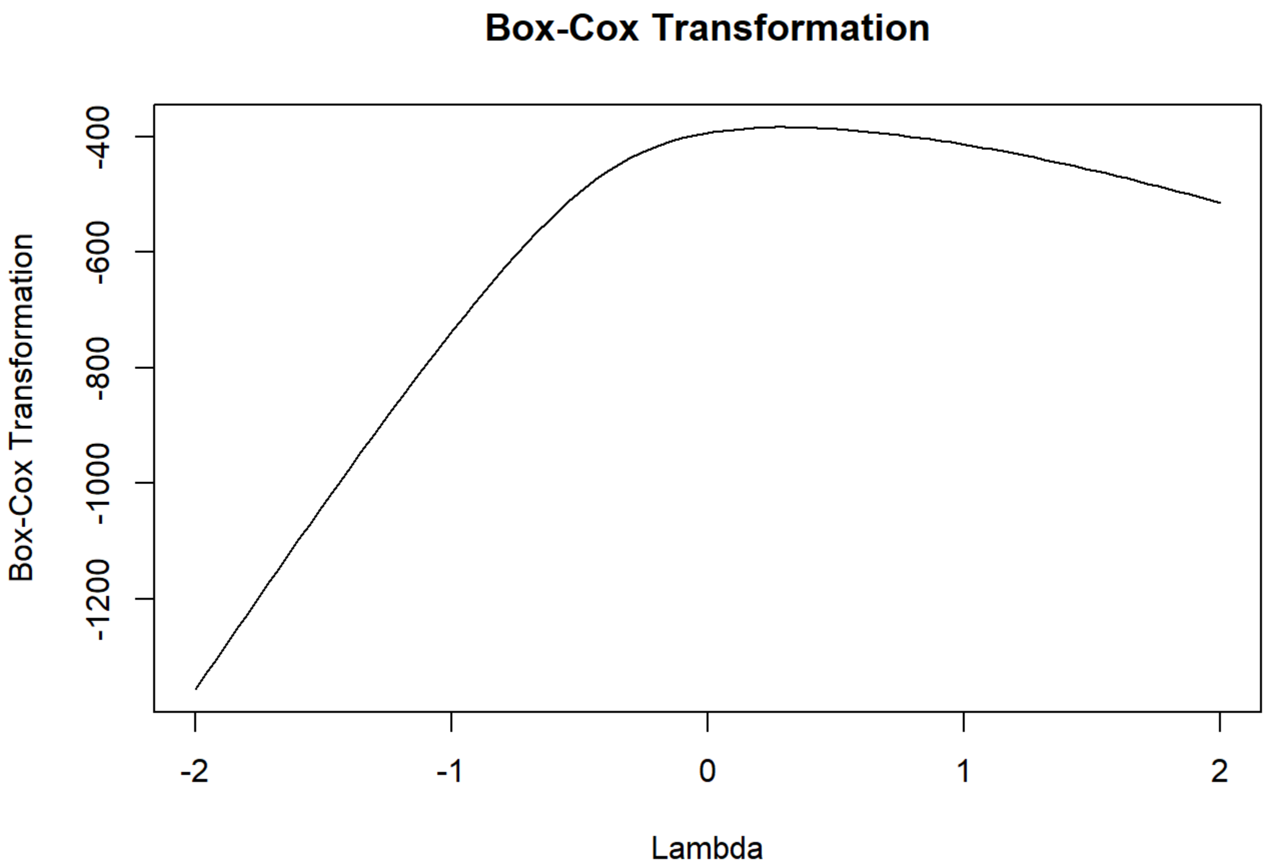
The Box-Cox transformation is a family of power transformations that includes the logarithmic transformation as a special case. It can handle a wider range of transformations, including both positive and negative values, by introducing a parameter lambda (λ) that determines the transformation applied.

```
# Load the MASS package
library(MASS)

# Perform Box-Cox transformation
transformed_data <- boxcox(temperature_ts ~ 1)
```




```
# Plot the Box-Cox transformation
plot(transformed_data$x, transformed_data$y, type = "l",
     xlab = "Lambda", ylab = "Box-Cox Transformation", main = "Box-Cox Transformation")
```



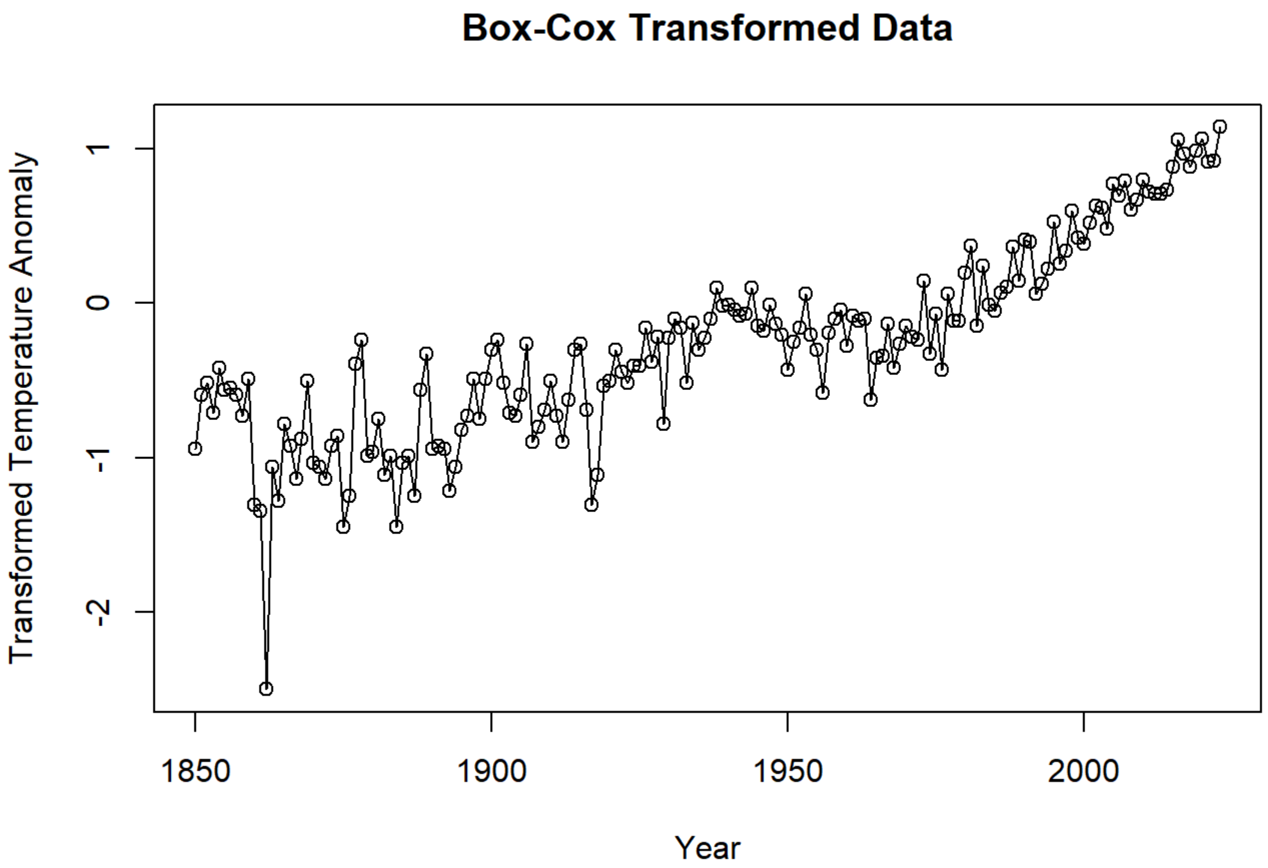

```
# Print the optimal lambda value
print(paste("Optimal lambda value:", transformed_data$x[which.max(transformed_data$y)]))
```

```
## [1] "Optimal lambda value: 0.303030303030303"
```

Optimal lambda value: 0.303030303030303”

```
library(MASS)
# Perform Box-Cox transformation
lambda=0.30
transformed_data1=(temperature_ts^lambda-1)/lambda

# Plot the transformed data
plot(transformed_data1, type = "o", xlab = "Year", ylab = "Transformed Temperature Anomaly",
main = "Box-Cox Transformed Data")
```



ADF Test

```
adf.test(transformed_data1)
```

```
## Warning in adf.test(transformed_data1): p-value smaller than printed p-value
```

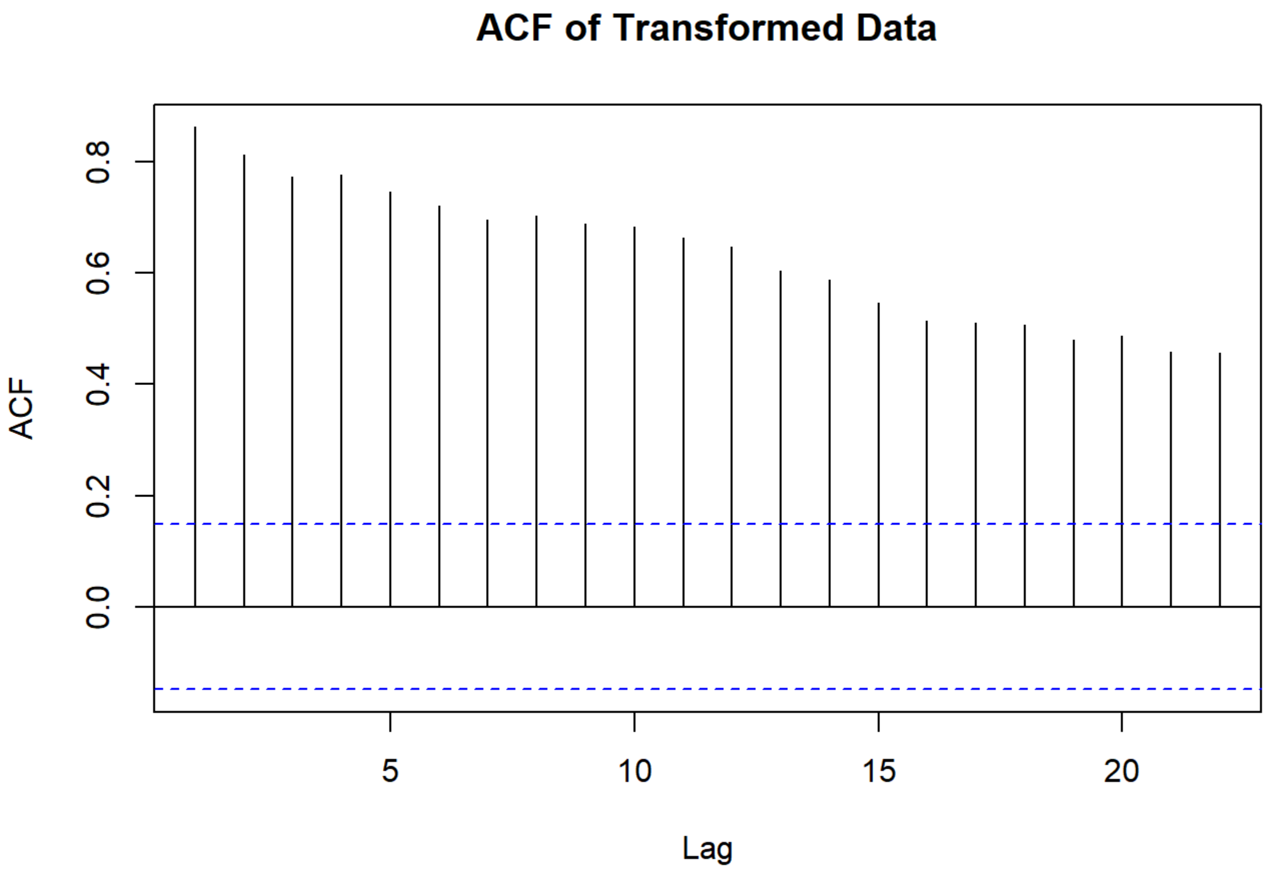
```
##
##  Augmented Dickey-Fuller Test
##
## data:  transformed_data1
## Dickey-Fuller = -4.0737, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

Shapiro Wilk Test

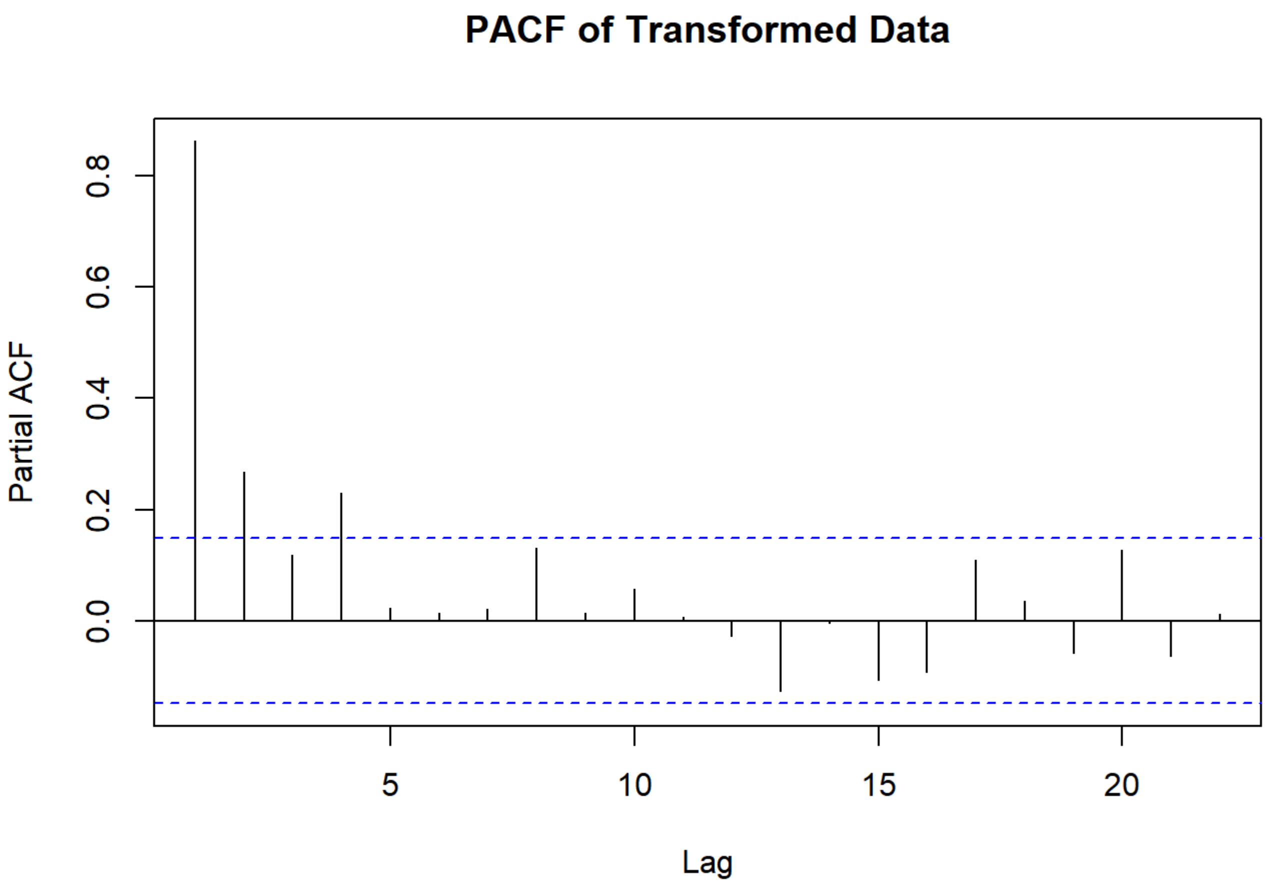
```
shapiro.test(transformed_data1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  transformed_data1
## W = 0.98226, p-value = 0.02572
```

```
# Plot ACF
acf(transformed_data1, main = "ACF of Transformed Data")
```




```
# Plot PACF
pacf(transformed_data1, main = "PACF of Transformed Data")
```

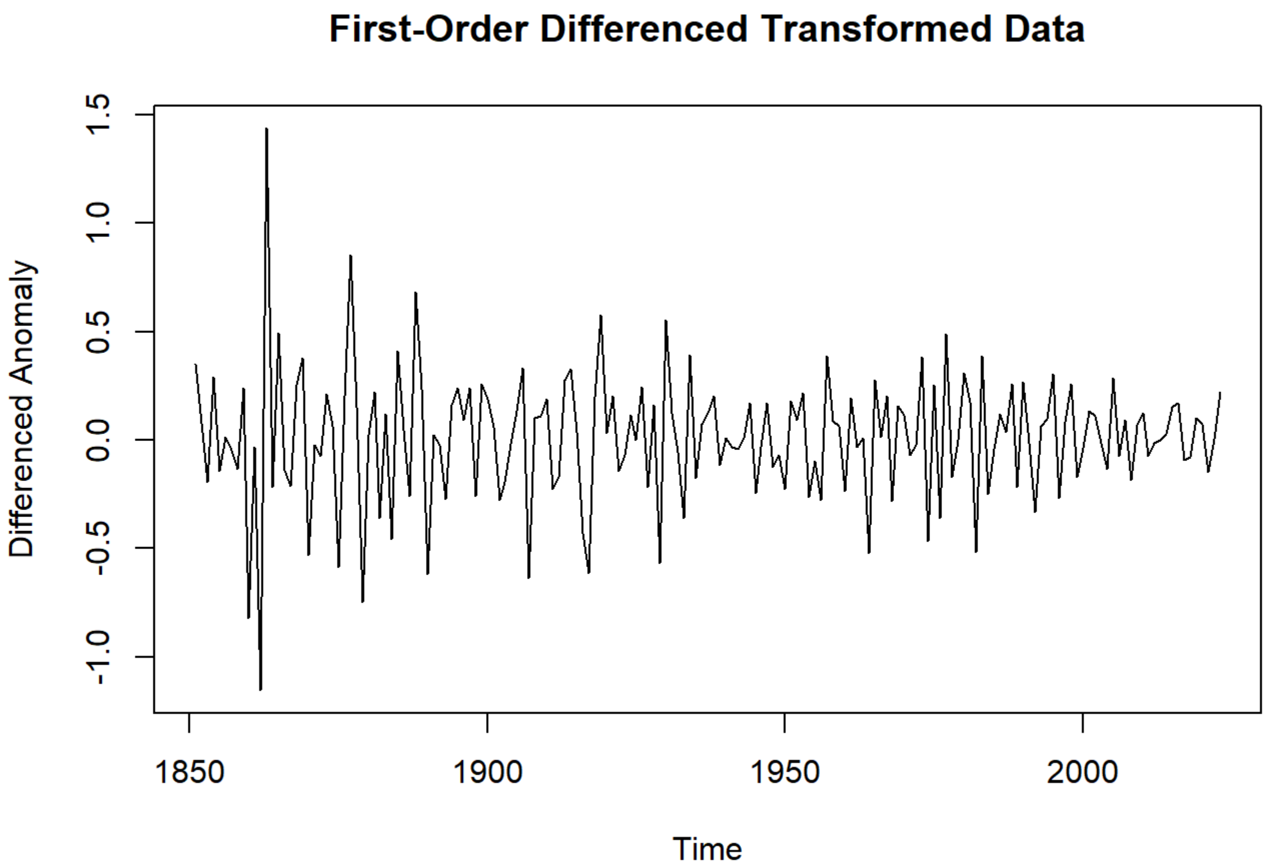


ACF and PACF plots of the transformed data still show evidence of non-stationarity despite the change in the p-value from the Augmented Dickey-Fuller (ADF) test, it suggests that while the data may exhibit stationarity in terms of its mean and variance, there might still be autocorrelation present in the series.

Differencing

Differencing involves computing the difference between consecutive observations in the time series. It is commonly used to remove trends or seasonality from the data, making it stationary. First-order differencing (taking the difference between consecutive observations) is often used to remove linear trends, while higher-order differencing may be necessary for removing higher-order trends.

```
# First-order differencing
transformed_data_diff1 <- diff(transformed_data1, differences = 1)
# Plot first-order differenced series
plot(transformed_data_diff1, type = "l", main = "First-Order Differenced Transformed Data", y
lab = "Differenced Anomaly")
```



After first level of differencing we can see that the time series plot is more stablelized .

ADF and PP Test

```
# Augmented Dickey-Fuller (ADF) Test
adf_result <- adf.test(transformed_data_diff1)
```

```
## Warning in adf.test(transformed_data_diff1): p-value smaller than printed
## p-value
```

```
print(adf_result)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  transformed_data_diff1
## Dickey-Fuller = -7.5525, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```
# Phillips-Perron Unit Root Test
pp_result <- pp.test(transformed_data_diff1)
```

```
## Warning in pp.test(transformed_data_diff1): p-value smaller than printed p-value
```

```
print(pp_result)
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  transformed_data_diff1
## Dickey-Fuller Z(alpha) = -188.96, Truncation lag parameter = 4, p-value
## = 0.01
## alternative hypothesis: stationary
```

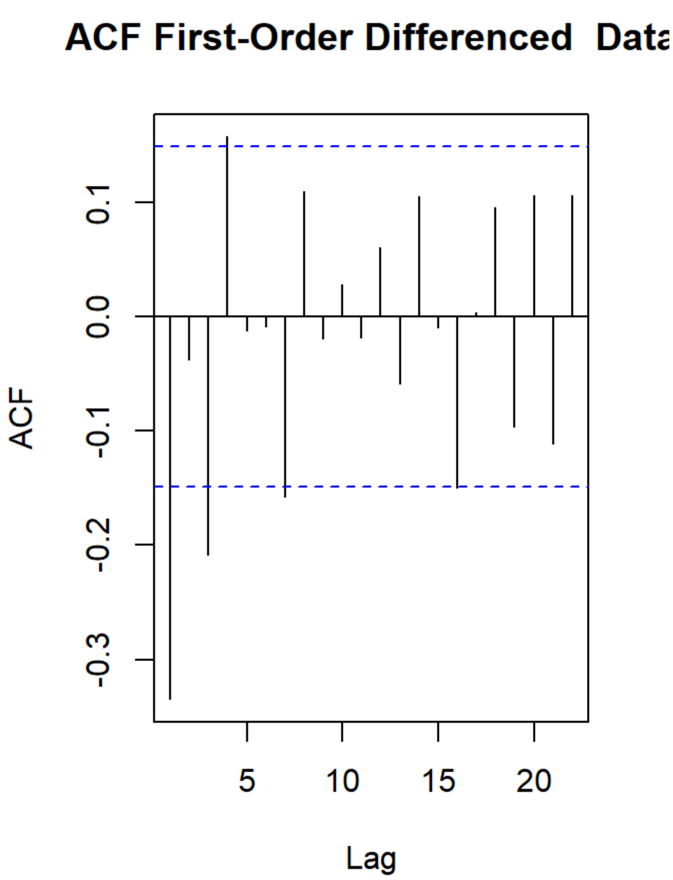
- With a p-value of 0.01, which is smaller than the typical significance level of 0.05, we reject the null hypothesis of non-stationarity.With a p-value of 0.01, which again is smaller than the significance level, we reject the null hypothesis of non-stationarity.
- Therefore, we conclude that the transformed_data_diff1 series is stationary.

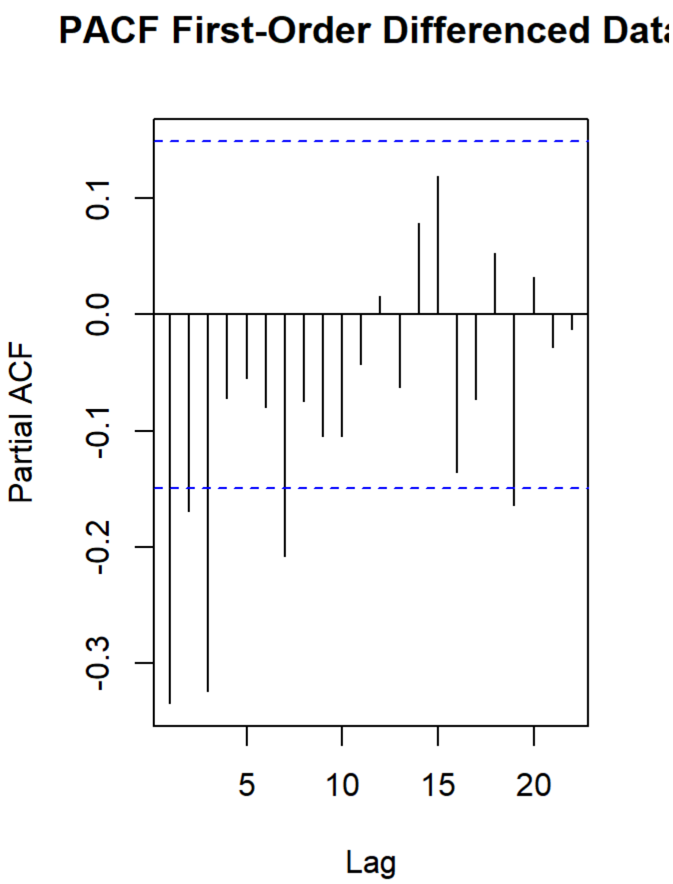
Overall, both tests provide strong evidence supporting the stationarity of the transformed dataset, suggesting that it does not exhibit trends or systematic patterns over time.

Model Specification

ACF and PACF plots

```
# Plot ACF and PACF for first-order differenced series
par(mfrow = c(1, 2))
acf(transformed_data_diff1, main = "ACF First-Order Differenced  Data")
pacf(transformed_data_diff1, main = "PACF First-Order Differenced Data")
```





The ACF and PACF plots shows good significant lags without any trends or patterns after first level of differencing .

For the P in PACF values , we see there are 5 significant lags in the plot.

For the Q values in ACF , we can see there are 4 significant lags and one lag in the 15 mark just over the line . So , I can consider 4 or 5 .

Set of arima models obtained

(5,1,5)

(5,1,4)

EACF

```
install.packages("TSA")
```

```
## Warning: package 'TSA' is in use and will not be installed
```

```
library(TSA)
```

```
eacf(transformed_data_diff1)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x o x x o o x o o o o  o  o  o
## 1 x o x o o o x o o o o  o  o  o
## 2 x x x o o o x o o o o  o  o  o
## 3 x o x o o o o o o o o  o  o  o
## 4 x o o x x o x o o o o  o  o  o
## 5 x o o x o o x o o o o  o  o  o
## 6 x o x x o x x o o o o  o  o  o
## 7 x x x x x x x o o o o  o  o  o
```

possible sets of arima mode from EACF is

(1,1,3)

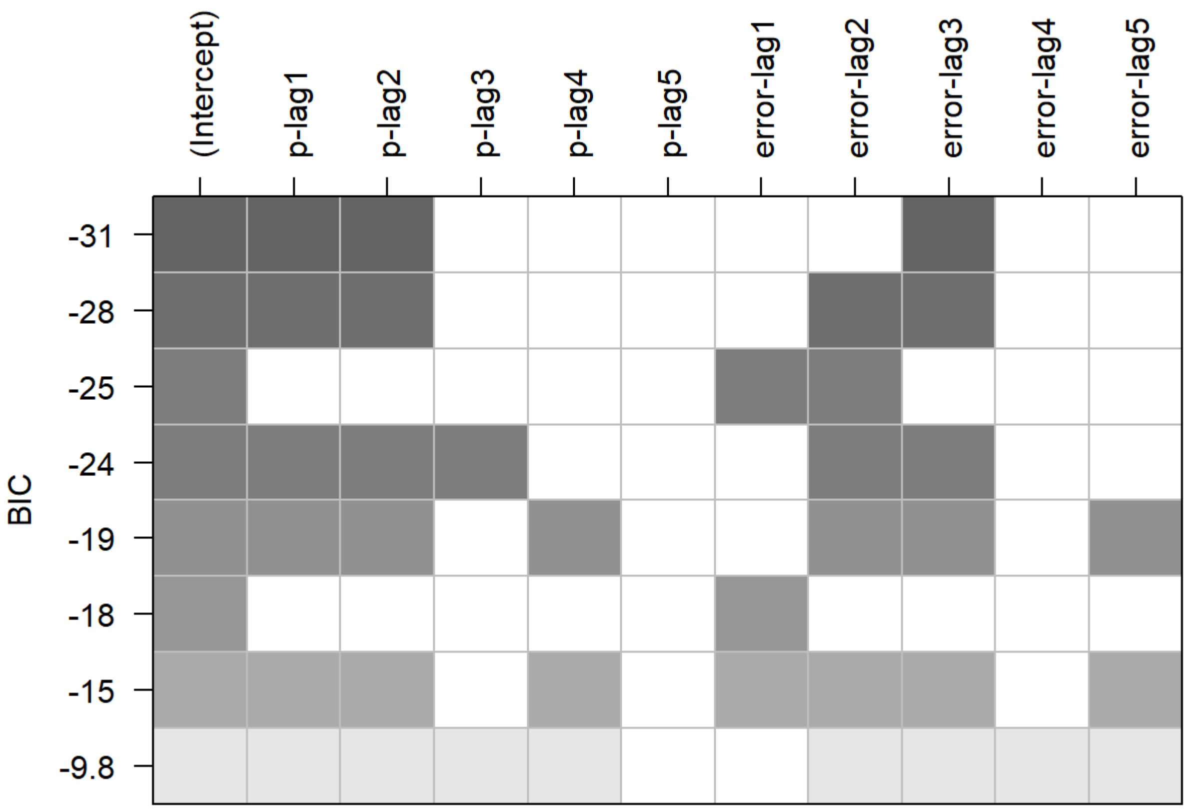
(1,1,4)

(2,1,3)

(2,1,4)

BIC

```
res=armasubsets(y= transformed_data_diff1,nar=5,nma=5,y.name='p',ar.method = 'ols')
plot(res)
```



The possible models from BIC table is

(1,1,3)

(2,1,3)

(1,1,2,)

The Final candidate set of models are

ARIMA(5,1,5)

ARIMA(5,1,4)

ARIMA(1,1,3)

ARIMA(1,1,4)

ARIMA(2,1,3)

ARIMA(2,1,4)

ARIMA(1,1,2)

Parameter Estimation

```
library(lmtest)
```

```
## Warning: package 'lmtest' was built under R version 4.1.3
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 4.1.3
```

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

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

ARIMA(5,1,5)

```
model_515_ml = arima(temperature_ts,order=c(5,1,5),method='ML')
coeftest(model_515_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1 -0.097420   0.124988 -0.7794 0.4357260
## ar2  0.055931   0.063663  0.8785 0.3796519
## ar3  0.400948   0.058449  6.8598 6.898e-12 ***
## ar4  0.792829   0.089639  8.8447 < 2.2e-16 ***
## ar5 -0.242292   0.103968 -2.3305 0.0197822 *
## ma1 -0.449046   0.099763 -4.5011 6.759e-06 ***
## ma2 -0.220678   0.106902 -2.0643 0.0389896 *
## ma3 -0.541031   0.082489 -6.5588 5.424e-11 ***
## ma4 -0.430230   0.117033 -3.6762 0.0002368 ***
## ma5  0.751456   0.078925  9.5212 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_515_css = arima(temperature_ts,order=c(5,1,5),method='CSS')
coeftest(model_515_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1 -0.078176   0.146017 -0.5354 0.5923784
## ar2  0.027551   0.091662  0.3006 0.7637435
## ar3  0.425354   0.077341  5.4997 3.804e-08 ***
## ar4  0.767733   0.141635  5.4205 5.944e-08 ***
## ar5 -0.199259   0.122098 -1.6320 0.1026879
## ma1 -0.492087   0.129110 -3.8114 0.0001382 ***
## ma2 -0.188449   0.143448 -1.3137 0.1889443
## ma3 -0.559000   0.097790 -5.7163 1.088e-08 ***
## ma4 -0.334641   0.166768 -2.0066 0.0447895 *
## ma5  0.669899   0.095699  7.0001 2.558e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_514_ml = arima(temperature_ts,order=c(5,1,4),method='ML')
```

```
## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1
```

```
coeftest(model_514_ml)
```

```
## Warning in sqrt(diag(se)): NaNs produced
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value Pr(>|z|)
## ar1 -0.817412        NaN      NaN      NaN
## ar2 -1.217428   0.073135 -16.6464  < 2e-16 ***
## ar3 -0.589399        NaN      NaN      NaN
## ar4 -0.129425   0.138765  -0.9327  0.35098
## ar5 -0.146259   0.069113  -2.1162  0.03433 *
## ma1  0.322915        NaN      NaN      NaN
## ma2  0.728719        NaN      NaN      NaN
## ma3 -0.239436        NaN      NaN      NaN
## ma4 -0.203853        NaN      NaN      NaN
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ARIMA(5,1,4)

```
model_514_css = arima(temperature_ts,order=c(5,1,4),method='CSS')
coeftest(model_514_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1  0.673280   0.160169  4.2036 2.627e-05 ***
## ar2 -0.080549   0.329095 -0.2448    0.8066
## ar3  0.292281   0.402771  0.7257    0.4680
## ar4  0.055241   0.134604  0.4104    0.6815
## ar5 -0.179322   0.117038 -1.5322    0.1255
## ma1 -1.265135   0.169553 -7.4616 8.549e-14 ***
## ma2  0.313876   0.340634  0.9214    0.3568
## ma3 -0.332175   0.618652 -0.5369    0.5913
## ma4  0.409075   0.303069  1.3498    0.1771
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ARIMA(1,1,3)

```
model_113_ml = arima(temperature_ts,order=c(1,1,3),method='ML')
coeftest(model_113_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
## ar1 -0.945462   0.055334 -17.0864 < 2.2e-16 ***
## ma1  0.461733   0.094271   4.8979 9.685e-07 ***
## ma2 -0.570200   0.067049  -8.5042 < 2.2e-16 ***
## ma3 -0.171510   0.073709  -2.3268   0.01997 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_113_css = arima(temperature_ts,order=c(1,1,3),method='CSS')
coeftest(model_113_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1  0.481992   0.271729  1.7738 0.076096 .
## ma1 -1.050049   0.291332 -3.6043 0.000313 ***
## ma2  0.116334   0.161129  0.7220 0.470300
## ma3  0.145225   0.096376  1.5069 0.131847
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ARIMA(1,1,4)

```
model_114_ml = arima(temperature_ts,order=c(1,1,4),method='ML')
coeftest(model_114_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1 -0.109766   0.402299 -0.2728   0.7850
## ma1 -0.412507   0.396838 -1.0395   0.2986
## ma2 -0.211924   0.199095 -1.0644   0.2871
## ma3 -0.108472   0.092116 -1.1776   0.2390
## ma4  0.177834   0.081037  2.1945   0.0282 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_114_css = arima(temperature_ts,order=c(1,1,4),method='CSS')
coeftest(model_114_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1  0.078748   1.470740  0.0535   0.9573
## ma1 -0.602180   1.464733 -0.4111   0.6810
## ma2 -0.130138   0.638252 -0.2039   0.8384
## ma3 -0.101098   0.204569 -0.4942   0.6212
## ma4  0.206978   0.134261  1.5416   0.1232
```

ARIMA(2,1,3)

```
model_213_ml = arima(temperature_ts,order=c(2,1,3),method='ML')
coeftest(model_213_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value Pr(>|z|)
## ar1 -0.388813   0.020400 -19.0590  < 2e-16 ***
## ar2 -0.970151   0.020541 -47.2303  < 2e-16 ***
## ma1 -0.132068   0.061466  -2.1486  0.03166 *
## ma2  0.708912   0.039284  18.0458  < 2e-16 ***
## ma3 -0.609584   0.062929  -9.6868  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_213_css = arima(temperature_ts,order=c(2,1,3),method='CSS')
coeftest(model_213_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1 -0.358426   0.056471 -6.3471 2.194e-10 ***
## ar2 -0.888335   0.093282 -9.5231 < 2.2e-16 ***
## ma1 -0.149460   0.078887 -1.8946   0.05814 .
## ma2  0.559230   0.115741  4.8317 1.354e-06 ***
## ma3 -0.541872   0.072696 -7.4539 9.061e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ARIMA(2,1,4)

```
model_214_ml = arima(temperature_ts,order=c(2,1,4),method='ML')
coeftest(model_214_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value Pr(>|z|)
## ar1 -0.390570   0.020771 -18.8038   <2e-16 ***
## ar2 -0.970789   0.020734 -46.8208   <2e-16 ***
## ma1 -0.112625   0.078359  -1.4373   0.1506
## ma2  0.692137   0.058554  11.8204   <2e-16 ***
## ma3 -0.601953   0.064034  -9.4005   <2e-16 ***
## ma4 -0.027362   0.071832  -0.3809   0.7033
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_214_css = arima(temperature_ts,order=c(2,1,4),method='CSS')
coeftest(model_214_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1 -0.354430   0.064101 -5.5292 3.217e-08 ***
## ar2 -0.870605   0.155874 -5.5853 2.333e-08 ***
## ma1 -0.162940   0.097880 -1.6647  0.095973 .
## ma2  0.550520   0.170057  3.2373  0.001207 **
## ma3 -0.539571   0.084103 -6.4156 1.403e-10 ***
## ma4  0.022012   0.089763  0.2452  0.806281
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ARIMA(1,1,2)

```
model_112_ml = arima(temperature_ts,order=c(1,1,2),method='ML')
coeftest(model_112_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1 -0.093303   0.337026 -0.2768   0.7819
## ma1 -0.436637   0.326070 -1.3391   0.1805
## ma2 -0.157691   0.198694 -0.7936   0.4274
```

```
model_112_css = arima(temperature_ts,order=c(1,1,2),method='CSS')
coeftest(model_112_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1 -0.059494   0.294478 -0.2020   0.8399
## ma1 -0.472704   0.290131 -1.6293   0.1033
## ma2 -0.136202   0.181942 -0.7486   0.4541
```

```
AIC(model_515_ml,model_514_ml,model_113_ml,model_114_ml,model_213_ml,model_214_ml,model_112_m
l)
```

```
##              df        AIC
## model_515_ml 11 -113.37709
## model_514_ml 10 -104.03061
## model_113_ml  5 -100.29695
## model_114_ml  6 -100.28465
## model_213_ml  6 -109.63879
## model_214_ml  7 -107.78274
## model_112_ml  4  -98.69114
```

```
aic_scores <- c(
  AIC(model_515_ml),
  AIC(model_514_ml),
  AIC(model_113_ml),
  AIC(model_114_ml),
  AIC(model_213_ml),
  AIC(model_214_ml),
  AIC(model_112_ml)
)
# Sort the AIC scores
sorted_aic_scores <- sort(aic_scores)

# Display the sorted AIC scores
sorted_aic_scores
```

```
## [1] -113.37709 -109.63879 -107.78274 -104.03061 -100.29695 -100.28465  -98.69114
```

```
BIC(model_515_ml,model_514_ml,model_113_ml,model_114_ml,model_213_ml,model_214_ml,model_112_m
l)
```

```
##              df       BIC
## model_515_ml 11 -78.69089
## model_514_ml 10 -72.49769
## model_113_ml  5 -84.53049
## model_114_ml  6 -81.36490
## model_213_ml  6 -90.71904
## model_214_ml  7 -85.70970
## model_112_ml  4 -86.07797
```

```
# Create a vector of BIC values
bic_values <- c(
  BIC(model_515_ml),
  BIC(model_514_ml),
  BIC(model_113_ml),
  BIC(model_114_ml),
  BIC(model_213_ml),
  BIC(model_214_ml),
  BIC(model_112_ml)
)

# Sort the BIC values
sorted_bic_values <- sort(bic_values)

# Display the sorted BIC values
sorted_bic_values
```

```
## [1] -90.71904 -86.07797 -85.70970 -84.53049 -81.36490 -78.69089 -72.49769
```

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.1.3
```

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

```
model.515A = Arima(temperature_ts,order=c(5,1,5),method='ML')
model.514A = Arima(temperature_ts,order=c(5,1,4),method='ML')
model.113A = Arima(temperature_ts,order=c(1,1,3),method='ML')
model.114A = Arima(temperature_ts,order=c(1,1,4),method='ML')
model.213A = Arima(temperature_ts,order=c(2,1,3),method='ML')
model.214A = Arima(temperature_ts,order=c(2,1,4),method='ML')
model.112A = Arima(temperature_ts,order=c(1,1,2),method='ML')
```

```
Smodel.515A <- accuracy(model.515A)[1:7]
Smodel.514A <- accuracy(model.514A)[1:7]
Smodel.113A <- accuracy(model.113A)[1:7]
Smodel.114A <- accuracy(model.114A)[1:7]
Smodel.213A <- accuracy(model.213A)[1:7]
Smodel.214A <- accuracy(model.214A)[1:7]
Smodel.112A <- accuracy(model.112A)[1:7]
```

```
df.Smodels <- data.frame(
rbind(Smodel.515A, Smodel.514A,Smodel.113A,Smodel.114A,Smodel.213A,Smodel.214A,Smodel.112A))
colnames(df.Smodels) <- c("ME", "RMSE", "MAE", "MPE", "MAPE",
"MASE", "ACF1")
rownames(df.Smodels) <- c("ARIMA(5,1,5)", "ARIMA(5,1,4)", "ARIMA(1,1,3)",
"ARIMA(1,1,4)","ARIMA(2,1,3)","ARIMA(2,1,4)","ARIMA(1,1,2)")
```

```
df.Smodels
```

```
##                      ME      RMSE       MAE       MPE     MAPE      MASE
## ARIMA(5,1,5) 0.01771218 0.1601244 0.1280727 -22.29674 39.20832 0.7961400
## ARIMA(5,1,4) 0.02866676 0.1660190 0.1322603 -21.06205 38.98809 0.8221714
## ARIMA(1,1,3) 0.03168658 0.1751369 0.1378862 -20.45087 39.09062 0.8571439
## ARIMA(1,1,4) 0.02990477 0.1741026 0.1353121 -21.30195 39.41291 0.8411422
## ARIMA(2,1,3) 0.02872176 0.1671030 0.1340269 -21.89238 40.00940 0.8331531
## ARIMA(2,1,4) 0.02920538 0.1670730 0.1340301 -21.88559 40.00042 0.8331732
## ARIMA(1,1,2) 0.03162485 0.1770231 0.1396776 -21.66049 40.51916 0.8682799
##                      ACF1
## ARIMA(5,1,5) -0.006131200
## ARIMA(5,1,4) -0.031166271
## ARIMA(1,1,3) -0.050308018
## ARIMA(1,1,4) -0.028579254
## ARIMA(2,1,3) -0.009880209
## ARIMA(2,1,4) -0.029482923
## ARIMA(1,1,2) -0.033496546
```

1. **Selection Process**:
    - We evaluate each model based on their RMSE, MAE, MPE, and MAPE.

- Lower values of RMSE, MAE, MPE, and MAPE indicate better predictive accuracy.
- The model with the lowest values across these metrics is considered the best choice.

2. **ARIMA(5,1,5)**:

   This model has the lowest RMSE and MAE among all the fitted models, indicating that, on average, its predictions are closest to the actual values.

   - Additionally, while not the lowest, its MPE and MAPE values are also reasonable compared to other models, suggesting that its predictions are relatively accurate in terms of percentage differences.

   - Overall, considering all the error metrics, ARIMA(5,1,5) provides the best balance of accuracy and precision in predicting the time series data.

CONCLUSION

Based on these metrics, the ARIMA(5,1,5) model has the lowest RMSE and MAE, indicating better performance in predicting the time series data. Therefore, the ARIMA(5,1,5) model is the best among the fitted set of possible models.