Analysis of Australian Unemployment Rate

Osama Alfawzan

2023-06-11

Contents

Introduction

This report offers a thorough examination and forecast of Australia's unemployment rate. It covers various aspects of the predictive modeling process. The data used in this report is sourced from the official website of the **Australian Bureau of Statistics (ABS)**.

Visual analysis of the data involves the utilization of different types of graphs, such as time series, scatter plots, ACF, and PACF plots, to reveal patterns and autocorrelations present in the dataset. The ACF, PACF, EACF, and the Bayesian Information Criterion (BIC) table are used to determine the most suitable model. The process of parameter estimation is utilized to identify the most suitable parameters for the chosen model.

To evaluate the model's effectiveness, an analysis of residuals is conducted. Various residual analysis plots, such as time series, histograms, q-q plots, and ACF plots, are carefully reviewed to validate the model's assumptions.

Subsequently, a forecasting analysis is performed to predict the unemployment rates over the next **10 months**. This forecast holds significant value for policymakers, businesses, and stakeholders as it enhances their comprehension and enables them to plan accordingly for future labor market conditions.

Data Description

The analysis is conducted using a dataset comprising a time series of unemployment rates in Australia. This dataset includes **121** observations, covering the period from **March 2013** to **March 2023**. Each observation consists of the date and the corresponding unemployment rate for that specific timeframe. The main aim of this report is to analyze the historical data and provide a forecast of the unemployment rates for the upcoming **10 months**.

The dataset was sourced from the official **Australian Bureau of Statistics (ABS)** website and can be downloaded from **here**. **Table 1** below shows a random sample of the time series data.

Table 1: Random Sample of the Time Series Data

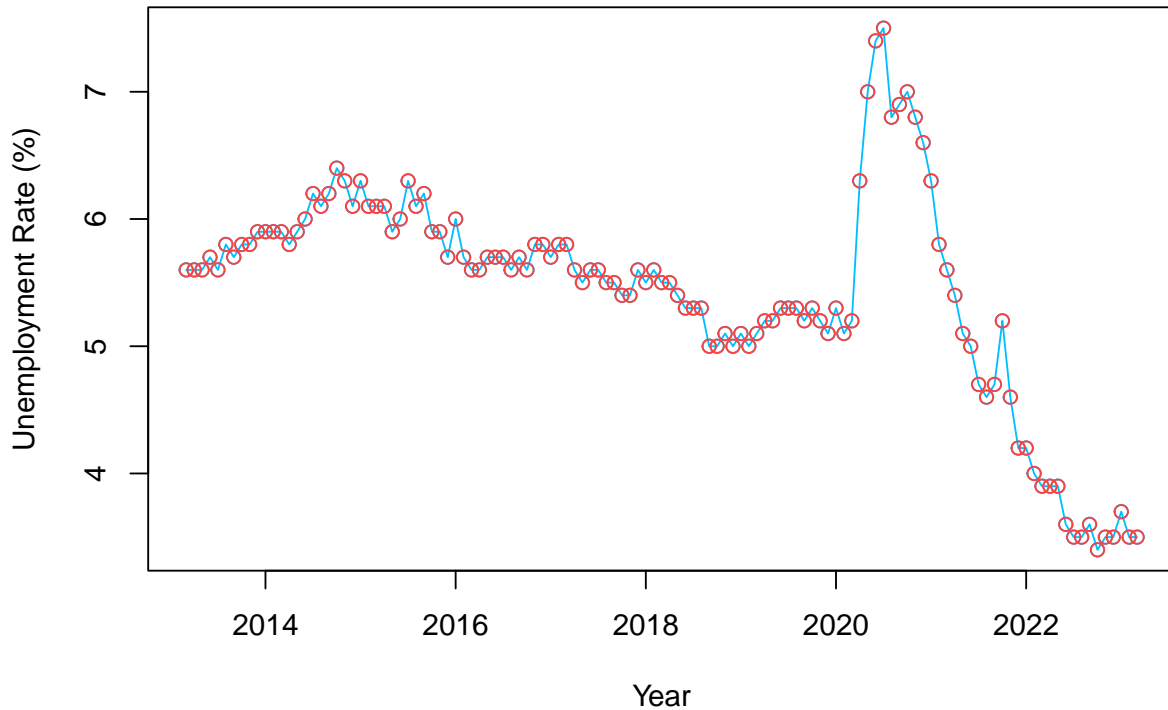|	Date	Unemployment Rate
72	2019-02-01	5.0
3	2013-05-01	5.6
87	2020-05-01	7.0
93	2020-11-01	6.8
73	2019-03-01	5.1

Descriptive Analysis

In this section, we will focus on the descriptive analysis of the unemployment rate time series data. We will begin by generating time series plots to visualize the data and identify any patterns or trends.

Next, we will look into the distribution of the unemployment rate by computing summary statistics, such as mean, median, and quartiles, to gain a better understanding of its variability. Lastly, we will calculate the correlation coefficient to examine the relationship between the unemployment rate in consecutive periods.

Time Series Plots

```r
plot(UnemploymentTS, type = "o", col = "deepskyblue", xlab = "Year",
     ylab = "Unemployment Rate (%)",
     main = "Figure 1. Australian Unemployment Rate from 2013 to 2023")
points(y = UnemploymentTS, x = time(UnemploymentTS),col = "brown1")
```

**Figure 1. Australian Unemployment Rate from 2013 to 2023**

The time series plot in **Figure 1** above illustrates the fluctuations in the unemployment rate over the given period. The unemployment rate ranges from **3.4%** to **7.5%**, with periods of relative stability followed by intermittent changes. There are notable peaks and dips in the data, indicating shifts in the unemployment rate over time. The COVID-19 pandemic significantly impacted unemployment rates in **2020-2021**.

However, vaccination efforts and the lifting of restrictions led to a decline in the unemployment rates, reaching its lowest point in **October 2022**. By visually examining the plot, we can observe the overall trend and patterns in the unemployment rate series, where the trend is mostly decreasing with a change in variance.

```r
plot(y = UnemploymentTS, x = zlag(UnemploymentTS),
     ylab = 'Unemployment Rate (%)',
     xlab = 'Previous Month Unemployment Rate',
     col = "brown1",
     main = "Figure 2. Scatter plot of neighboring unemployment rate")
```

Figure 2. Scatter plot of neighboring unemployment rate
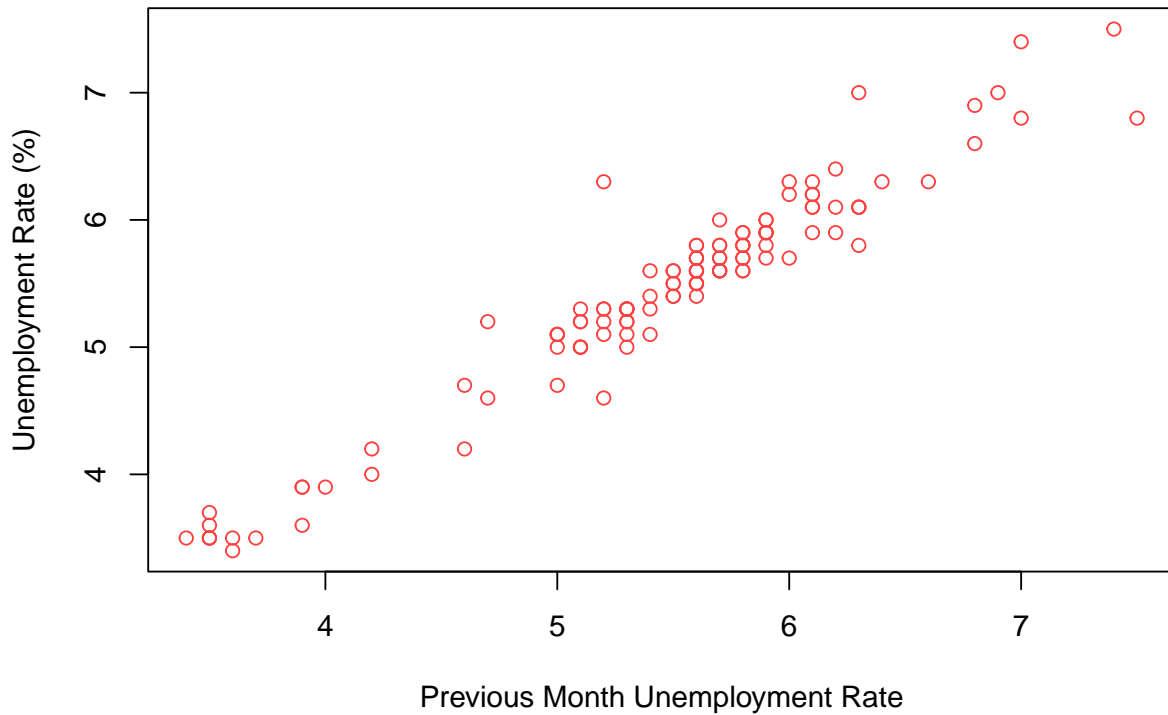
Figure 2 above is a scatter plot depicting the relationship between neighboring unemployment rates. The plot shows that as the unemployment rate increases, there is a corresponding increase in the neighboring unemployment rate, forming a diagonal pattern moving towards the top right of the plot. This indicates a positive correlation between neighboring unemployment rates.

Based on this scatter plot, it can be inferred that neighboring unemployment rates can be a good predictor of future values.

Summary Statistics

Table 2: Summary Statistics for the Australian Unemployment Rate

Min.	3.400000
1st Qu.	5.100000
Median	5.600000
Mean	5.438016
3rd Qu.	5.900000
Max.	7.500000

The summary statistics for the Australian unemployment rate series in **Table 2** above provide key insights into the distribution of the data. The minimum unemployment rate recorded during the period was **3.4%**, indicating the lowest level reached. On the other hand, the maximum unemployment rate observed was **7.5%**, representing the highest level reached. The median unemployment rate, which represents the middle value of the series, was **5.6%**. The mean unemployment rate, calculated by averaging all the values, was **5.438%**. The first quartile, marking the 25th percentile of the data, was at **5.1%**, while the third quartile,

indicating the 75th percentile.

Correlation Coefficient

```
y = UnemploymentTS
x = zlag(UnemploymentTS)
index = 2:length(x)
print(paste("The correlation coefficient is:", cor(y[index], x[index])))
```

```
[1] "The correlation coefficient is: 0.967860343770281"
```

The correlation coefficient of **0.9678603** for the Australian unemployment rate series indicates a strong positive correlation between the values of the unemployment rate in subsequent months. This means that there is a consistent and significant relationship between the unemployment rates in consecutive months. The high correlation coefficient suggests that there is a strong linear association between the unemployment rate in the current month and the unemployment rate in the previous month. Therefore, the value of the unemployment rate in a particular month can be a good predictor of the unemployment rate in the following month.

Statistical Analysis and Preprocessing

In this section, we will analyze the unemployment rate time series data to better understand its patterns and prepare it for modeling. We will examine the autocorrelation and partial autocorrelation to identify any relationships between observations. Furthermore, we will assess the normality of the series using Q-Q plots and the Shapiro-Wilk test.

To address trends and non-stationarity, differencing will be applied, and unit-root tests will confirm the stationarity. These steps provide valuable insights into the unemployment rate time series characteristics, guiding the selection of suitable models for further analysis.

```
plot_unemployment <- function(data,title) {
  par(mfrow=c(2,2), cex.main = 0.7, mar = c(4, 4, 4, 1))
  qqnorm(data, col = "deepskyblue",
         main ="Normal Q-Q plot of the time series")
  qqline(data, col = "brown1")

  plot(data, type = 'o',
       col = "deepskyblue",
       ylab = 'Unemployment Rate (%)',
       main ="Time series plot of the series")
  points(y = data, x = time(data), col = "brown1")

  acf(data,
      main ="ACF plot of the series",
      col = "brown1",
      lag.max = 50)

  pacf(data,
       main ="PACF plot of the series",
       col = "brown1",
       lag.max = 50)
```

```
    mtext(title, side = 3, line = -1, outer = TRUE)}
```

The previous function, **plot_unemployment**, generates visual plots for the time series data. It includes a normal Q-Q plot, time series plot, ACF plot, and PACF plot. This function provides a concise way to visualize the characteristics of the unemployment rate time series data throughout the differencing phase.

ACF-PACF Analysis

```
plot_unemployment(UnemploymentTS,
                  "Figure 3. Australian Unemployment Rate Time Series")
```

Figure 3. Australian Unemployment Rate Time Series
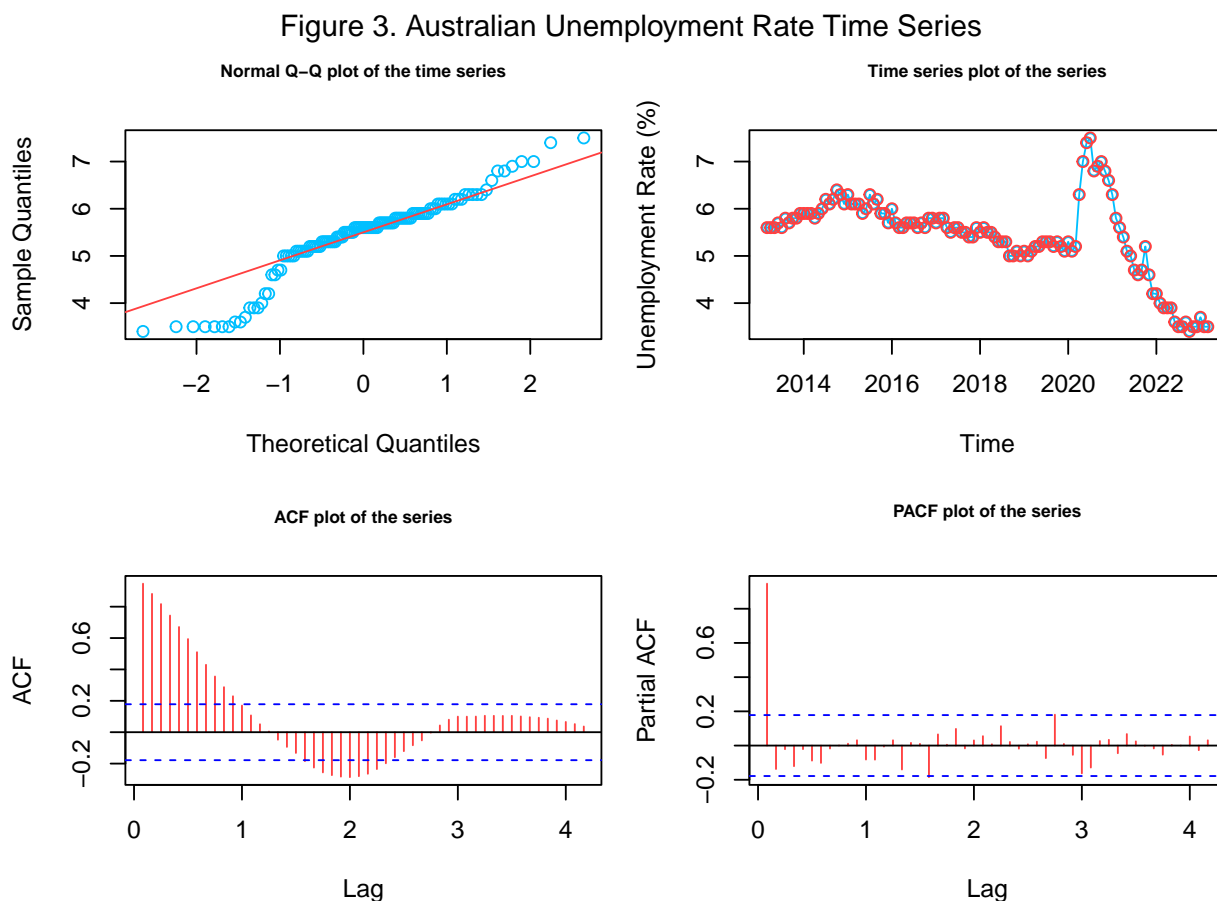
The Q-Q plot of the unemployment rate time series data in **Figure 3** above reveals a considarible departure from normality, indicating non-normality. This observation can be further validated by conducting the Shapiro-Wilk normality test. Additionally, the ACF plot displays a gradual decay pattern, while the PACF plot exhibits a significant first lag, indicating nonstationarity in the time series.

Nonstationarity implies that the mean, variance, or both change over time, making it challenging to utilize model specification tools effectively. Therefore, prior to employing tools such as ACF-PACF, EACF, and BIC table, it is necessary to transform the time series into a stationary form by eliminating any trend or nonstationary components.

Shapiro-Wilk Normality Test

```
    Shapiro-Wilk normality test

data:  UnemploymentTS
W = 0.92626, p-value = 5.226e-06
```
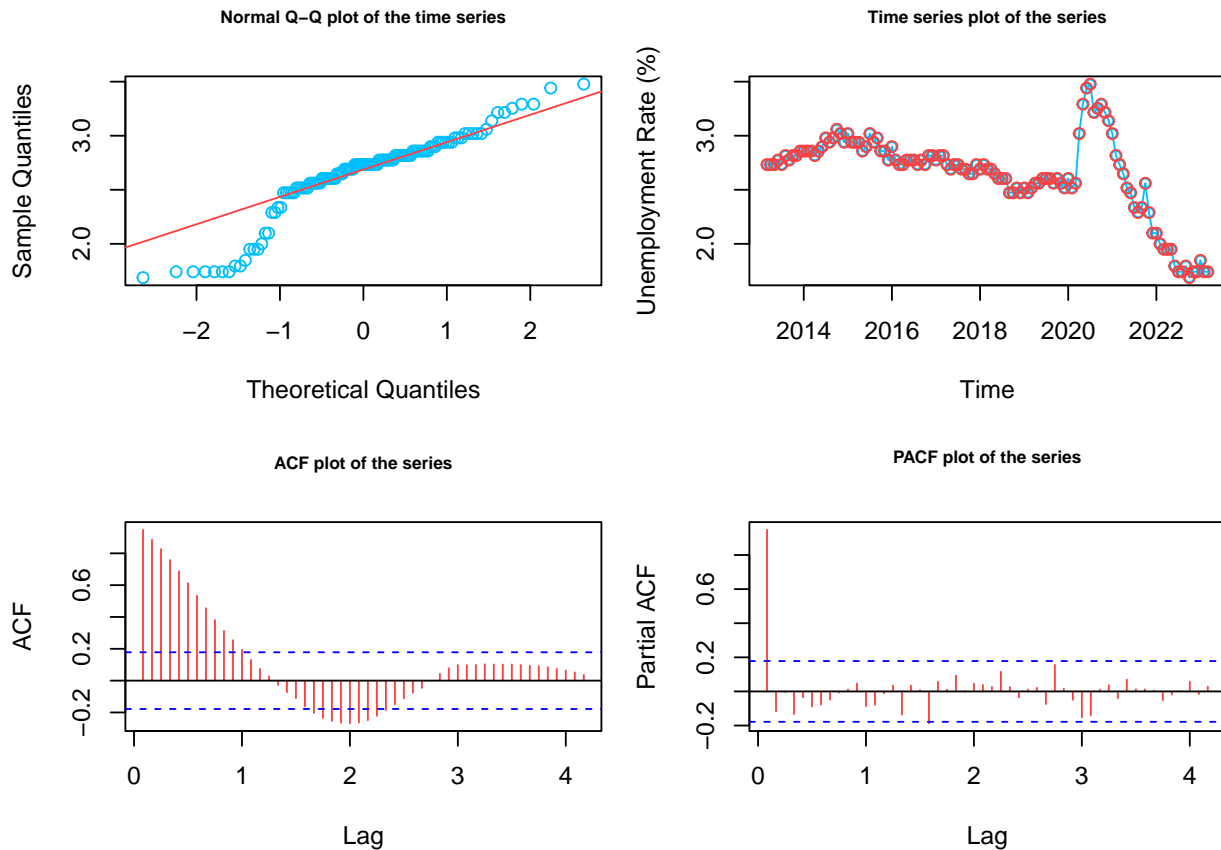
The Shapiro-Wilk normality test yielded a p-value of **5.226e-06**, which is below the significance level of **0.05**. This suggests that the unemployment rate time series data deviates significantly from a normal distribution. To address this non-normality, one possible approach is to consider transformations such as a Box-Cox transformation or log transformation, which can help stabilize the variation and improve the data's adherence to normality.

Transformation

In this section, we will apply two types of transformations, Box-Cox and log transformations, to address the non-normality of the unemployment rate time series data. We will use the **plot_unemployment** function to examine the impact, if any, of these transformations on the data's distribution. This analysis will help us determine whether the transformations successfully normalize the data or if alternative approaches need to be considered.

```
BC <- BoxCox.ar(UnemploymentTS, lambda = 0.5)
lambda <- BC$lambda[which(max(BC$loglike) == BC$loglike)]
BC.UnemploymentTS <- (UnemploymentTS^lambda-1)/lambda
plot_unemployment(BC.UnemploymentTS,
            "Figure 4. Box-Cox tranformation for the unemployment rate series")
```

Figure 4. Box–Cox tranformation for the unemployment rate series

Normal Q–Q plot of the time series

Time series plot of the series

ACF plot of the series
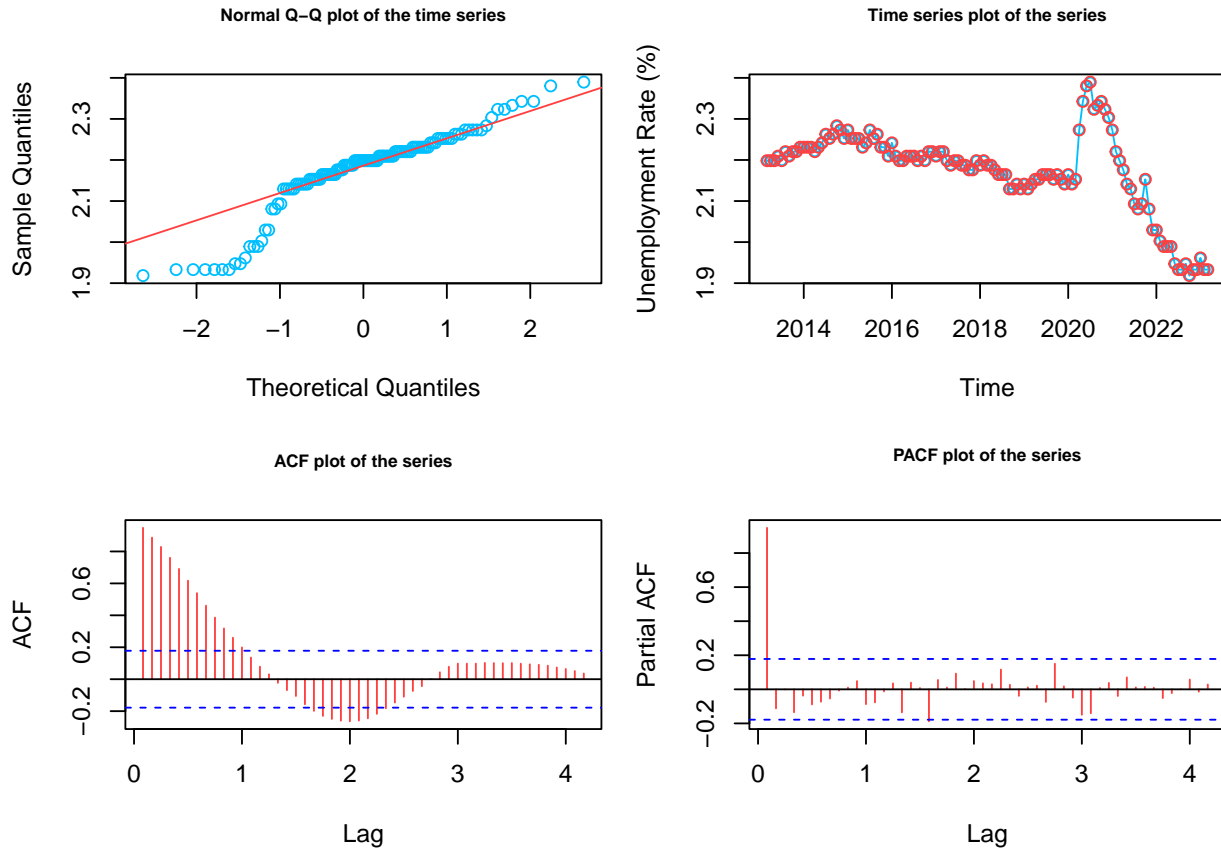
PACF plot of the series

After trying different lambda values for the Box-Cox transformation, it is observed from **Figure 4** above that the transformation did not significantly affect the normality of the unemployment rate time series data. The plots of the Box-Cox transformation, including the Q-Q plot, indicate that the distribution of the transformed data remains similar to the original data, suggesting that the Box-Cox transformation did not successfully address the non-normality.

In the next step, we will examine the impact of the log transformation on the normality of the data.

```
log_UnemploymentTS <- log(UnemploymentTS + abs(min(UnemploymentTS)) + 0.01)
plot_unemployment(log_UnemploymentTS,
                  "Figure 5. Log tranformation for the unemployment rate series")
```

Figure 5. Log tranformation for the unemployment rate series

Normal Q–Q plot of the time series

Time series plot of the series

ACF plot of the series

PACF plot of the series

The above plots of the log transformation in **Figure 5** reveal that, similar to the Box-Cox transformation, it had no significant impact on the normality of the unemployment rate time series data. The distribution of the log-transformed data remains similar to the original data, indicating that the log transformation did not successfully address the non-normality.

Consequently, we decided to proceed with the raw data for further analysis and apply differencing to address the nonstationarity in the series.

Differencing

Now, we will apply differencing to the unemployment rate series in order to make it stationary. We will begin with one differencing level to observe if the series becomes stationary or not. Stationarity is crucial for time series analysis as it ensures that the mean and variance of the series remain constant over time.

By examining the differenced series, we can determine if the initial differencing level is sufficient to achieve stationarity, or if further differencing is required.

```
diff.UnemploymentTS = diff(UnemploymentTS)
plot_unemployment(diff.UnemploymentTS,
            "Figure 6. The first difference for the unemployment rate series")
```

9

Figure 6. The first difference for the unemployment rate series
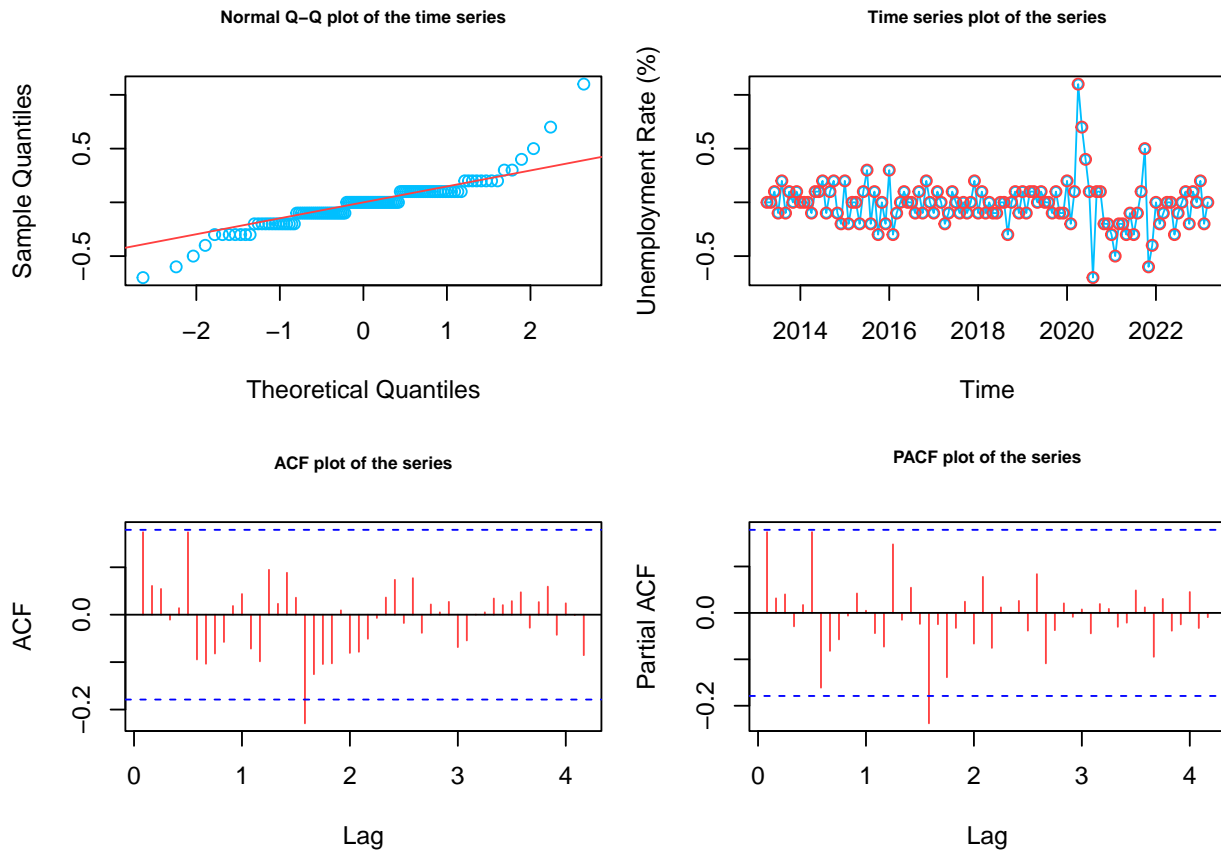
Figure 6 above presents a set of four plots depicting the first difference of the unemployment rate series. The Q-Q plot demonstrates an improvement in normality, indicating that the differencing process has reduced the non-normality in the data. The time series plot reveals a flat mean level, centered around zero, signifying the removal of the trend component. Moreover, there is no observable seasonality or change points in the time series plot.

In the ACF and PACF plots, there are two slightly significant autocorrelations within the confidence level, along with one autocorrelation that is both significant and above the confidence level.

In the next step, we will perform Unit-Root tests to confirm the stationarity of the series before proceeding with the model selection phase.

Unit-Root Tests

```
adf.test(diff.UnemploymentTS)
pp.test(diff.UnemploymentTS)
```

```
        Augmented Dickey-Fuller Test

data:  diff.UnemploymentTS
Dickey-Fuller = -4.4382, Lag order = 4, p-value = 0.01
alternative hypothesis: stationary
```

```
    Phillips-Perron Unit Root Test

data:  diff.UnemploymentTS
Dickey-Fuller Z(alpha) = -101.62, Truncation lag parameter = 4, p-value
= 0.01
alternative hypothesis: stationary
```

The results of the unit root tests further strengthen the evidence of stationarity in the differenced unemployment rate series. Both Dickey-Fuller test and the Phillips-Perron unit root test yield highly significant p-values of 0.01, indicating a strong rejection of the null hypothesis of non-stationarity. Therefore, we can conclude that differencing the time series once was sufficient to transform the nonstationary series into a stationary one. Based on these observations, it is reasonable to proceed with the EACF analysis to identify potential values for the ARIMA model parameters.

Model Selection

The suitability of GARCH, ARIMA, and ARMA models for the unemployment rate series depends on the characteristics of the data. GARCH (Generalized Autoregressive Conditional Heteroskedasticity) models are commonly used for modeling returns time series data. On the other hand, ARMA (Autoregressive Moving Average) models are suitable for stationary time series data.

Given that the unemployment rate series exhibited trends and non-stationarity, the most suitable model would be the ARIMA (Autoregressive Integrated Moving Average) model.

ARIMA Models

Going back to the acf and pacf plots in **Figure 6**, we observe two autocorrelations that are slightly significant within the confidence level, along with one autocorrelation that is both significant and above the confidence level. Based on the ACF and PACF plots, the proposed set of possible **ARIMA(p, d, q)** models includes **ARIMA(1, 1, 1), ARIMA(3, 1, 1), ARIMA(1, 1, 3), and ARIMA(3, 1, 3)**. The selected order of differencing is **1 (d)**, while the proposed **AR(p)** and **MA(q)** orders are determined by the significant autocorrelations observed in the ACF and PACF plots.

These initial model specifications provide a starting point for further analysis and refinement.

EACF Analysis

```
eacf(diff.UnemploymentTS)
```

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

According to the Extended Autocorrelation Function (EACF) model selection tool, the proposed set of possible models for the unemployment rate series includes **ARIMA(0, 1, 1), ARIMA(0, 1, 2), ARIMA(1, 1,**

0), ARIMA(1, 1, 1), ARIMA(1, 1, 2), and ARIMA(1, 1, 3). The orders of **AR(p)** and **MA(q)** were determined based on the significant autocorrelations observed in the EACF plot, and the selected order of differencing is **1 (d)**.

In addition to the EACF analysis, the BIC table will be utilized as another model selection tool to identify the set of possible models in the next step of the analysis.

BIC Table

```
res <- armasubsets(y=diff.UnemploymentTS,nar=4,nma=4,y.name='p',ar.method='ols')
plot(res)
title(main = "Figure 7. BIC table for the Australian Unemployment Rate series",
      line = 5.5, cex.main = 1)
```


Figure 7. BIC table for the Australian Unemployment Rate series
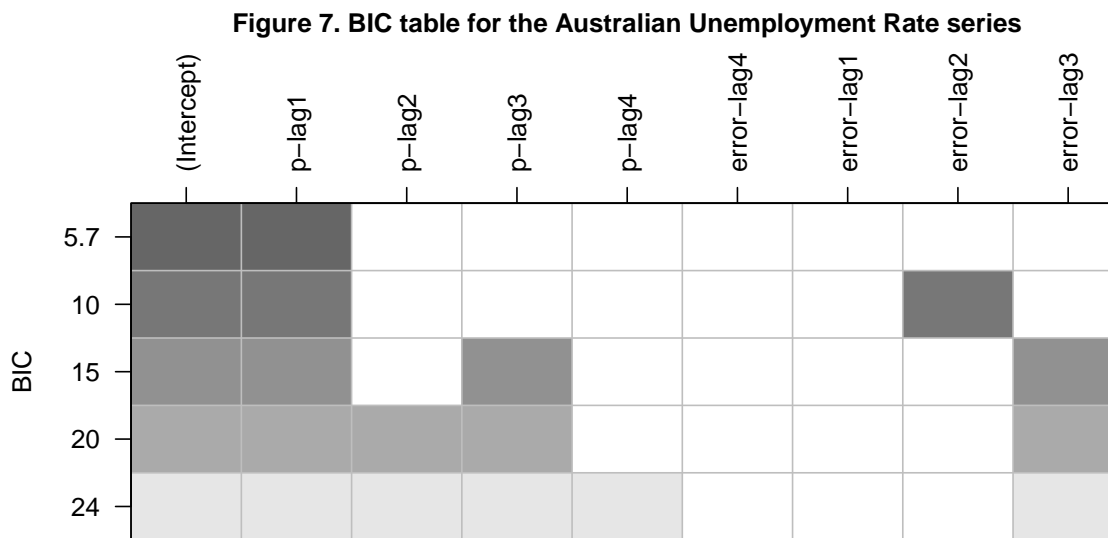
The BIC table depicted in Figure 7 was employed to determine the set of potential models by evaluating their corresponding BIC values. Based on the BIC analysis, the proposed models for consideration include **ARIMA(0, 1, 2), ARIMA(1, 1, 2), and ARIMA(1, 1, 3)**. These models were selected due to their comparatively lower BIC values in comparison to alternative models.

After performing the ACF-PACF analysis, EACF analysis, and considering the BIC values, we have identified a final set of models for the unemployment rate time series. These models include **ARIMA(0, 1, 1), ARIMA(0, 1, 2), ARIMA(1, 1, 0), ARIMA(1, 1, 1), ARIMA(1, 1, 2), ARIMA(3, 1, 1), ARIMA(1, 1, 3), and ARIMA(3, 1, 3)**. These models were chosen based on their ability to capture the observed autocorrelation patterns and their BIC values.

During model fitting, we will assess the performance of these models and determine the best model that provides accurate forecasts for the next 10 months of the unemployment rate series.

Model Evaluation

In this section, we will evaluate and select the best fitting model for the original unemployment rate series. To assess the goodness of fit and the significance of the model coefficients, we will perform coefficient tests.

Additionally, we will utilize AIC/BIC tables as model selection criteria, considering the models' information criteria values. Furthermore, we will employ error measures such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to identify the accuracy of the models' forecasts.

By conducting these evaluations and comparisons, we aim to identify the model that exhibits the best overall performance in terms of statistical significance, information criteria, and forecast accuracy.

Coefficient Tests

In this section, we will assess the significance of the estimated coefficients in our ARIMA models. The estimation of parameters will be carried out using two methods: maximum likelihood (ML) and conditional sum of squares (CSS). To determine the significance of the coefficients, we will examine the results of the z test.

```
# ARIMA(0, 1, 1)
print("ARIMA(0, 1, 1) CSS")
model_011_css = Arima(UnemploymentTS,order=c(0,1,1),method='CSS')
coeftest(model_011_css)
print("ARIMA(0, 1, 1) ML")
model_011_ml = Arima(UnemploymentTS,order=c(0,1,1),method='ML')
coeftest(model_011_ml)

# ARIMA(0, 1, 2)
print("ARIMA(0, 1, 2) CSS")
model_012_css = Arima(UnemploymentTS,order=c(0,1,2),method='CSS')
coeftest(model_012_css)
print("ARIMA(0, 1, 2) ML")
model_012_ml = Arima(UnemploymentTS,order=c(0,1,2),method='ML')
coeftest(model_012_ml)

# ARIMA(1, 1, 0)
print("ARIMA(1, 1, 0) CSS")
model_110_css = Arima(UnemploymentTS,order=c(1,1,0),method='CSS')
coeftest(model_110_css)
print("ARIMA(1, 1, 0) ML")
model_110_ml = Arima(UnemploymentTS,order=c(1,1,0),method='ML')
coeftest(model_110_ml)

# ARIMA(1, 1, 1)
print("ARIMA(1, 1, 1) CSS")
model_111_css = Arima(UnemploymentTS,order=c(1,1,1),method='CSS')
coeftest(model_111_css)
print("ARIMA(1, 1, 1) ML")
model_111_ml = Arima(UnemploymentTS,order=c(1,1,1),method='ML')
coeftest(model_111_ml)

# ARIMA(1, 1, 2)
print("ARIMA(1, 1, 2) CSS")
model_112_css = Arima(UnemploymentTS,order=c(1,1,2),method='CSS')
coeftest(model_112_css)
print("ARIMA(1, 1, 2) ML")
model_112_ml = Arima(UnemploymentTS,order=c(1,1,2),method='ML')
coeftest(model_112_ml)
```

```
# ARIMA(3, 1, 1)
print("ARIMA(3, 1, 1) CSS")
model_311_css = Arima(UnemploymentTS,order=c(3,1,1),method='CSS')
coeftest(model_311_css)
print("ARIMA(3, 1, 1) ML")
model_311_ml = Arima(UnemploymentTS,order=c(3,1,1),method='ML')
coeftest(model_311_ml)

# ARIMA(1, 1, 3)
print("ARIMA(1, 1, 3) CSS")
model_113_css = Arima(UnemploymentTS,order=c(1,1,3),method='CSS')
coeftest(model_113_css)
print("ARIMA(1, 1, 3) ML")
model_113_ml = Arima(UnemploymentTS,order=c(1,1,3),method='ML')
coeftest(model_113_ml)

# ARIMA(3, 1, 3)
print("ARIMA(3, 1, 3) CSS")
model_313_css = Arima(UnemploymentTS,order=c(3,1,3),method='CSS')
coeftest(model_313_css)
print("ARIMA(3, 1, 3) ML")
model_313_ml = Arima(UnemploymentTS,order=c(3,1,3),method='ML')
coeftest(model_313_ml)
```

```
[1] "ARIMA(0, 1, 1) CSS"

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ma1 0.167505   0.086879   1.928  0.05385 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

[1] "ARIMA(0, 1, 1) ML"

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ma1 0.166215   0.086566  1.9201  0.05485 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

[1] "ARIMA(0, 1, 2) CSS"

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ma1 0.169050   0.091090  1.8558  0.06348 .
ma2 0.047297   0.092890  0.5092  0.61064
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

[1] "ARIMA(0, 1, 2) ML"
```

```
z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ma1 0.167744   0.090697  1.8495  0.06439 .
ma2 0.046702   0.092101  0.5071  0.61210
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

[1] "ARIMA(1, 1, 0) CSS"

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ar1 0.179533   0.089804  1.9992  0.04559 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

[1] "ARIMA(1, 1, 0) ML"

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ar1 0.178051   0.089393  1.9918  0.04639 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

[1] "ARIMA(1, 1, 1) CSS"

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ar1  0.45595    0.51185  0.8908   0.3730
ma1 -0.28851    0.55380 -0.5210   0.6024

[1] "ARIMA(1, 1, 1) ML"

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ar1  0.45275    0.51305  0.8825   0.3775
ma1 -0.28637    0.55532 -0.5157   0.6061

[1] "ARIMA(1, 1, 2) CSS"

z test of coefficients:

     Estimate Std. Error z value Pr(>|z|)
ar1  0.645698   0.517023  1.2489   0.2117
ma1 -0.473717   0.524868 -0.9025   0.3668
ma2 -0.048386   0.142446 -0.3397   0.7341

[1] "ARIMA(1, 1, 2) ML"

z test of coefficients:
```

```
     Estimate Std. Error z value Pr(>|z|)
ar1  0.641870   0.524010  1.2249    0.2206
ma1 -0.471309   0.532180 -0.8856    0.3758
ma2 -0.047896   0.142449 -0.3362    0.7367


[1] "ARIMA(3, 1, 1) CSS"


z test of coefficients:


       Estimate Std. Error z value  Pr(>|z|)
ar1 -0.6216644  0.1789821 -3.4733  0.000514 ***
ar2  0.1768080  0.1100047  1.6073  0.107994
ar3  0.0095153  0.0919487  0.1035  0.917578
ma1  0.8025558  0.1578917  5.0830 3.716e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


[1] "ARIMA(3, 1, 1) ML"


z test of coefficients:


     Estimate Std. Error z value Pr(>|z|)
ar1 -0.028127   0.797329 -0.0353   0.9719
ar2  0.060498   0.164643  0.3674   0.7133
ar3  0.053530   0.093111  0.5749   0.5654
ma1  0.198685   0.794259  0.2502   0.8025


[1] "ARIMA(1, 1, 3) CSS"


z test of coefficients:


     Estimate Std. Error z value Pr(>|z|)
ar1 -0.102155   0.804121 -0.1270   0.8989
ma1  0.275295   0.800781  0.3438   0.7310
ma2  0.083344   0.184098  0.4527   0.6508
ma3  0.057026   0.093194  0.6119   0.5406


[1] "ARIMA(1, 1, 3) ML"


z test of coefficients:


     Estimate Std. Error z value Pr(>|z|)
ar1 -0.101420   0.814730 -0.1245   0.9009
ma1  0.273122   0.811352  0.3366   0.7364
ma2  0.081722   0.183865  0.4445   0.6567
ma3  0.056030   0.092432  0.6062   0.5444


[1] "ARIMA(3, 1, 3) CSS"


z test of coefficients:


       Estimate Std. Error z value  Pr(>|z|)
ar1 -0.8996568  0.0024596 -365.77 < 2.2e-16 ***
```

```
ar2  0.6081039  0.0032344  188.01 < 2.2e-16 ***
ar3  0.7138483        NaN     NaN       NaN
ma1  1.1797949        NaN     NaN       NaN
ma2 -0.3842887        NaN     NaN       NaN
ma3 -0.8494774        NaN     NaN       NaN
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

[1] "ARIMA(3, 1, 3) ML"

z test of coefficients:

    Estimate Std. Error z value Pr(>|z|)
ar1 -0.63090    0.37814 -1.6684  0.09523 .
ar2 -0.32780    0.44470 -0.7371  0.46105
ar3  0.52797    0.35940  1.4690  0.14182
ma1  0.85781    0.41379  2.0731  0.03817 *
ma2  0.52182    0.52274  0.9982  0.31816
ma3 -0.38481    0.41753 -0.9216  0.35672
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The coefficient test results above indicate the significance of the coefficients in each ARIMA model. Based on the results, the best model would be **ARIMA(3, 1, 1) CSS**, as it has a highly significant coefficient for the **ar1 and ma1** terms with estimates of **-0.6216644 and 0.8025558**, respectively. The p-values for these coefficients are below the conventional significance level of **0.05**, indicating their statistical significance in the model. The **ar2** coefficient has a p-value of **0.107994**, which is slightly above **0.05** but still relatively close to the significance level. The **ar3** coefficient has a p-value of **0.917578**, indicating it is not statistically significant.

AIC/BIC

In this section, we will use Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) to compare the proposed set of models and identify the model with the best fit. We will present the AIC and BIC tables, sorted from low to high based on the relative score using `sort.score` function below. This will allow us to compare the models relative performance and assess their adequacy for the Arctic series.

```
# This function is a custom function developed by one of Haydar's previous students.
# It is a handy function for sorting the models scores.

sort.score <- function(x, score = c("bic", "aic")){
  if (score == "aic"){
    x[with(x, order(AIC)),]
  } else if (score == "bic") {
    x[with(x, order(BIC)),]
  } else {
    warning('score = "x" only accepts valid arguments ("aic","bic")')
  }
}
```

I would like to acknowledge the work of the student who developed the sort.score function used in this report. Their contribution has been invaluable, and I am grateful for their effort.

```
sort.score(AIC(model_011_ml, model_012_ml, model_110_ml, model_111_ml,
               model_112_ml, model_311_ml, model_113_ml, model_313_ml), score = "aic")
```

Table 3: AIC table for the proposed set of models

	df	AIC
model_110_ml	2	-27.76544
model_011_ml	2	-27.46599
model_111_ml	3	-26.00279
model_012_ml	3	-25.72830
model_313_ml	7	-24.60048
model_112_ml	4	-24.09656
model_311_ml	5	-22.18631
model_113_ml	5	-22.15357

The AIC table above shows the AIC scores and degrees of freedom for the proposed set of ARIMA models. Lower AIC scores indicate better fitting models. Among the models listed, **ARIMA(1,1,0)** (model_110_ml) has the lowest AIC score of **-27.76544**. This suggests that it provides a better balance between model fit and complexity compared to the other models. Therefore, **ARIMA(1,1,0)** model is considered the best model based on its low AIC score.

```
sort.score(BIC(model_011_ml, model_012_ml, model_110_ml, model_111_ml,
               model_112_ml, model_311_ml, model_113_ml, model_313_ml), score = "bic" )
```

Table 4: BIC table for the proposed set of models

	df	BIC
model_110_ml	2	-22.190452
model_011_ml	2	-21.891012
model_111_ml	3	-17.640314
model_012_ml	3	-17.365821
model_112_ml	4	-12.946591
model_311_ml	5	-8.248851
model_113_ml	5	-8.216109
model_313_ml	7	-5.088033

Table 4 above displays the BIC scores and degrees of freedom for various ARIMA models. The BIC is a criterion that balances model fit and complexity, similar to the AIC, but with a stronger penalty for model complexity. Lower BIC scores indicate better fitting models. Based on the BIC table, the model with the lowest BIC score is **ARIMA(1,1,0)** (model_110_ml) with a BIC score of **-22.190452**. This model strikes a good balance between fit and complexity. Therefore, **ARIMA(1,1,0)** is considered the best model based on its low BIC score.

Error Measures

Next, we will use error measures such as ME, RMSE, MAE, MPE, MAPE, MASE, and ACF1 to assess the accuracy and performance of the proposed set of models. These measures provide valuable insights into bias, overall prediction error, percentage error, seasonal and trend adjustments. By considering these error measures, we can identify the best model that has good accuracy and predictive capabilities.

```r
Smodel_011_css <- accuracy(model_011_css)[1:7]
Smodel_012_css <- accuracy(model_012_css)[1:7]
Smodel_110_css <- accuracy(model_110_css)[1:7]
Smodel_111_css <- accuracy(model_111_css)[1:7]
Smodel_112_css <- accuracy(model_112_css)[1:7]
Smodel_311_css <- accuracy(model_311_css)[1:7]
Smodel_113_css <- accuracy(model_113_css)[1:7]
Smodel_313_css <- accuracy(model_313_css)[1:7]

df.Smodels <- data.frame(rbind(Smodel_011_css, Smodel_012_css,
                               Smodel_110_css, Smodel_111_css,
                               Smodel_112_css, Smodel_311_css,
                               Smodel_113_css, Smodel_313_css))
colnames(df.Smodels) <- c("ME", "RMSE", "MAE", "MPE", "MAPE",
                          "MASE", "ACF1")
rownames(df.Smodels) <- c('ARIMA(0,1,1)', 'ARIMA(0,1,2)', 'ARIMA(1,1,0)',
                          'ARIMA(1,1,1)', 'ARIMA(1,1,2)', 'ARIMA(3,1,1)',
                          'ARIMA(1,1,3)', 'ARIMA(3,1,3)')
round(df.Smodels,  digits = 3)
```

Table 5: Error Measurements table for the proposed set of models

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
ARIMA(0,1,1)	-0.015	0.211	0.146	-0.393	2.687	0.216	0.003
ARIMA(0,1,2)	-0.014	0.211	0.145	-0.378	2.682	0.216	-0.001
ARIMA(1,1,0)	-0.014	0.211	0.145	-0.376	2.684	0.216	-0.011
ARIMA(1,1,1)	-0.013	0.211	0.144	-0.351	2.663	0.214	0.000
ARIMA(1,1,2)	-0.013	0.211	0.144	-0.338	2.659	0.214	-0.005
ARIMA(3,1,1)	-0.014	0.210	0.143	-0.374	2.638	0.212	-0.002
ARIMA(1,1,3)	-0.014	0.211	0.145	-0.358	2.672	0.215	-0.005
ARIMA(3,1,3)	-0.012	0.187	0.129	-0.280	2.363	0.191	0.124

Based on the error measurement table above, the best model appears to be **ARIMA(3,1,3)** based on several metrics. It has a very low ME of **-0.012**, indicating minimal bias in predictions. Additionally, it has the lowest RMSE of **0.187**, indicating lower overall prediction error. The MAE of **0.129** is also the lowest among the models, indicating better accuracy. Furthermore, the MASE of **0.191** suggests that it performs well in adjusting for seasonal and trend patterns.

Finally, although the ACF1 value of **0.124** is not particularly strong, it is the highest among the models, indicating a relatively better fit in capturing patterns. Therefore, **ARIMA(3,1,3)** is the best model based on these error measures.

On the other hand, when considering all the model evaluation techniques used as an average, including coefficient tests, AIC/BIC tables, and error measures, the best model for prediction would be **ARIMA(1,1,0)**. This model consistently performs well across various evaluation metrics, including low AIC and BIC scores, as well as relatively low error measures such as RMSE, MAE, and MASE. Therefore, **ARIMA(1,1,0)** is the best choice based on the overall evaluation.

Residual Analysis

```r
residual.analysis <- function(model, title, std = TRUE,start = 2,
                              class = c("ARIMA","GARCH","ARMA-GARCH", "fGARCH")[1]){
  library(TSA)
  library(FitAR)
  if (class == "ARIMA"){
    if (std == TRUE){
      res.model = rstandard(model)
    }else{
      res.model = residuals(model)
    }
  }else if (class == "GARCH"){
    res.model = model$residuals[start:model$n.used]
  }else if (class == "ARMA-GARCH"){
    res.model = model@fit$residuals
  }else if (class == "fGARCH"){
    res.model = model@residuals
  }else {
    stop("The argument 'class' must be either 'ARIMA' or 'GARCH' ")
  }
  par(mfrow=c(3,2))
  plot(res.model,type='o',ylab='Standardised residuals',
       main="Time series plot of standardised residuals")
  abline(h=0)
  hist(res.model,main="Histogram of standardised residuals")
  qqnorm(res.model,main="QQ plot of standardised residuals")
  qqline(res.model, col = 2)
  acf(res.model,main="ACF of standardised residuals")
  print(shapiro.test(res.model))
  k=0
  LBQPlot(res.model, lag.max = 30, StartLag = k + 1, k = 0, SquaredQ = FALSE)
  mtext(title, side = 3, line = -1.2, outer = TRUE)
  par(mfrow=c(1,1))
}
```
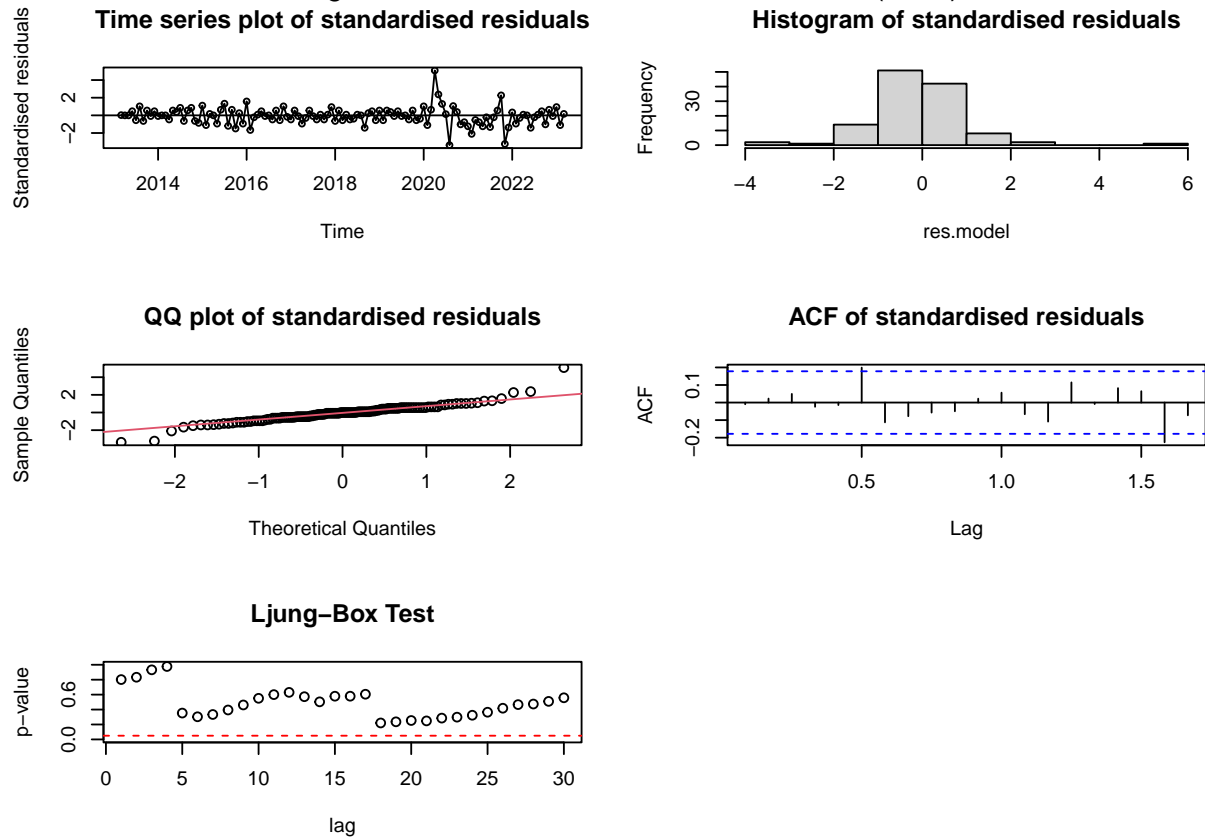
The previous `residual.analysis` function performs residual analysis for different models (ARIMA, GARCH, ARMA-GARCH, and fGARCH). It generates diagnostic plots and tests, such as time series plot, histogram, QQ plot, ACF, Shapiro-Wilk test, and Ljung-Box test. This function was taken from the unit's website on Canvas.

```r
residual.analysis(model = model_110_ml,
                  title = "Figure 8. Standardised residuals for ARIMA(1,1,0)")
```

```
    Shapiro-Wilk normality test

data:  res.model
W = 0.90998, p-value = 6.022e-07
```

Figure 8. Standardised residuals for ARIMA(1,1,0)

The output of the `residual.analysis` function in **Figure 8** provides insights into the behavior of the model's residuals. The time series plot shows no notable seasonality or variance change, although a few potential outliers are present. The histogram suggests a relatively symmetric shape, and the Q-Q plot indicates a good normality, despite a few outliers. The ACF plot demonstrates minimal residual autocorrelation, and the Ljung-Box test confirms the absence of significant autocorrelation at all lags. Additionally, the Shapiro-Wilk test yields a very low p-value (**p-value = 6.022e-07**), rejecting the assumption of normality for the residuals.

Overall, these observations suggest that the **ARIMA(1,1,0)** model effectively captures the underlying patterns and randomness in the data.

Forecasting

In the code below, we will fit the **ARIMA(1,1,0)** model to the original unemployment rate time series. Using the `forecast` function from the `forecast` package, we will generate a forecast for the next **10 months**. Then, we will plot the forecast to provide a visual representation of the predicted values. This forecast will allow us to anticipate the future behavior of the unemployment rate in Australia and help businesses and stakeholders make informed decisions based on the projected values.

```
fit = Arima(UnemploymentTS,c(1,1,0), method = "ML")
frc = forecast::forecast(fit,h=10)
as.data.frame(frc)
```

21

Table 6: Forecasted Australian unemployment rates for Apr 2023 to Jan 2024

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
Apr 2023	3.5	3.227242	3.772758	3.082853	3.917147
May 2023	3.5	3.078520	3.921480	2.855402	4.144598
Jun 2023	3.5	2.964720	4.035280	2.681359	4.318641
Jul 2023	3.5	2.870378	4.129622	2.537076	4.462923
Aug 2023	3.5	2.788309	4.211691	2.411563	4.588437
Sep 2023	3.5	2.714751	4.285249	2.299065	4.700935
Oct 2023	3.5	2.647512	4.352487	2.196233	4.803767
Nov 2023	3.5	2.585202	4.414797	2.100938	4.899062
Dec 2023	3.5	2.526874	4.473126	2.011732	4.988268
Jan 2024	3.5	2.471849	4.528151	1.927579	5.072421

The above **Table 6** represents the forecasted values for Australian unemployment rates for the period from **April 2023** to **January 2024**.

- `Point Forecast` indicates the estimated unemployment rate for each month.
- `Lo 80` and `Hi 80` represent the lower and upper bounds of the **80%** confidence interval for the forecasted values.
- `Lo 95` and `Hi 95` represent the lower and upper bounds of the **95%** confidence interval for the forecasted values.

```
plot(frc, type = "o", col = "brown1", xlab = "Year",
     ylab = "Unemployment Rate (%)",
     main = "Figure 9. Australian Unemployment Rate forecast for Apr 2023 to Jan 2024",
     cex.main = 0.9)
```

Figure 9. Australian Unemployment Rate forecast for Apr 2023 to Jan 2024
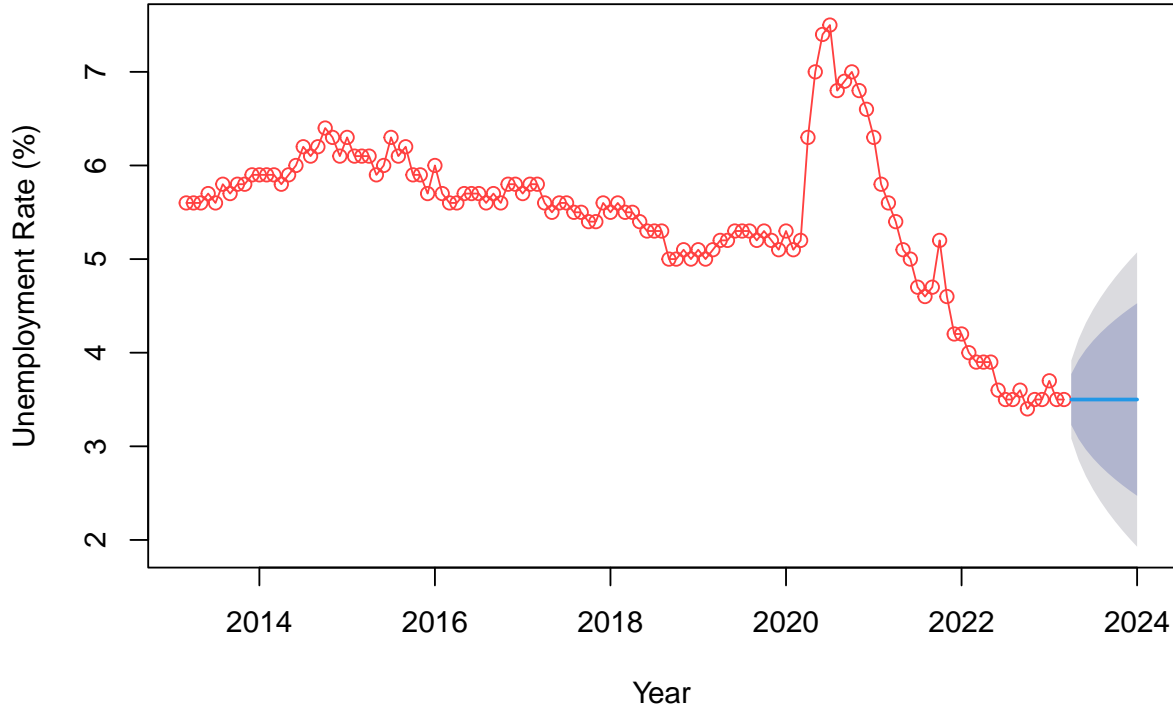
The plot in **Figure 9** provides a visual representation of the forecasted unemployment rates and its lower and upper bounds for the **80%** and **95%** confidence intervals. The forecasted unemployment rates are all at a value of **3.5**, indicating that the model predicts a stable unemployment rate for the forecast period. The shaded areas determined by the confidence intervals, illustrate the uncertainty associated with the forecasts.

Conclusion

In conclusion, our report aimed to analyze and forecast the Australian unemployment rates based on a time series analysis. We began by conducting descriptive analysis, which included time series plots, summary statistics, and correlation coefficient calculations. This allowed us to gain insights into the patterns and relationships present in the unemployment rate series.

To prepare the series for modeling, we performed statistical analysis and preprocessing steps. This involved conducting ACF-PACF analysis, Shapiro-Wilk normality test, and unit-root tests to ensure stationarity. After confirming the need for differencing, we proceeded with ARIMA modeling due to its suitability for our series. The selection of the ARIMA models was based on ACF-PACF analysis, EACF analysis, and the BIC table. The proposed set of models included **ARIMA(0, 1, 1), ARIMA(0, 1, 2), ARIMA(1, 1, 0), ARIMA(1, 1, 1), ARIMA(1, 1, 2), ARIMA(3, 1, 1), ARIMA(1, 1, 3), and ARIMA(3, 1, 3)**. These models were evaluated using coefficient tests, AIC/BIC tables, and error measures, along with residual analysis.

After careful evaluation of the models, considering all the techniques used on average, the **ARIMA(1, 1, 0)** model was the best choice for predicting the future unemployment rates. Therefore, we fitted the **ARIMA(1, 1, 0)** model to the original unemployment rate time series.

The forecasted unemployment rates for the next **10 months** consistently predict a value of **3.5**. This implies that the **ARIMA(1, 1, 0)** model projects a stable unemployment rate for the forecasted period. The forecasted rates indicate a consistent level of **3.5%** unemployment rate, suggesting relative stability in the job market over the period from **April 2023** to **January 2024**.