Math1318 Time Series Analysis / Math2204 Time series and Forecasting Final Project report

Declaration of Contributions:

No	**Name of Team Member**	**Contribution to project**
1	Stephanie Barwick	1/4
2	Yuting Shan	1/4
3	Fabian Caballero	1/4
4	Rasika Niwanthi Abeysekara	1/4
	Sum:	MUST EQUAL 1

RMIT University,
School of Science
2022

Research question: What are the most accurate forecasts for the rainfall at the Melbourne Botanical Gardens for the next 10 months?

General Introduction:

Monthly rainfall data is collected in many parts of the world for many reasons. Analysis of rainfall data facilitates policy decisions regarding water, such as cropping patterns and sowing dates for farming, dates for construction of public use infrastructure, such as roads or drainage, and details on providing drinking water to communities. Rainfall in Australia can be volatile, largely due to the atmospheric and oceanic fluctuations of the region.

The El Niño-Southern Oscillation, or ENSO, is the main contributor to Australian rainfall variability. ENSO is an irregular periodic variation in both sea surface temperatures and winds over the Pacific Ocean. The warming period of ENSO is referred to as El Niño and the cooling phase is referred to as La Niña. The effects of ENSO make it more difficult for long-term trends in rainfall to be identified. However, the successful identification of trends in rainfall data can allow for more proactive policies for water handling in periods of drought and flood.

Autoregressive Integrated Moving Average, or ARIMA, is a strategy used for modelling univariate time series data. Using both autoregressive and moving average, ARIMA uses the integrated element to differentiate, which allows it to model time series information that has trends. However, one limitation that ARIMA has is that it is unable to model data with seasonality. ARIMA requires data that is not seasonal, or alternatively has the seasonality removed. An alternative to ARIMA that does not share this limitation is Seasonal Autoregressive Integrated Moving Average, or SARIMA.

SARIMA is an extension of the ARIMA technique that allows the use of univariate time series data that include a seasonal component. To achieve this, SARIMA introduces three new hyperparameters that specify the Autoregression (AR), the differencing (I) and the moving average (MA) for the seasonal component of the time series. Additionally, a new parameter is included for the period of the seasonality.

SARIMA uses the same three initial trend elements, “p”, for autoregression, ”d”, for differencing, and “q”, for moving average. What is then added is “P”, for seasonal autoregression, “D”, for seasonal differencing, “Q”, for seasonal moving average, and “m”, the number of time steps for a single season.

An ARCH (Autoregressive conditionally heteroscedastic) model uses the variance of a time series, describing a changing, volatile variance. ARCH models are generally used in areas that have short periods of increased variance, such in econometric settings. An ARCH process is one where the variance at a point in time is conditional on the observations at the previous points. A GARCH

(Generalised Autoregressive conditionally heteroscedastic) model uses the values of past squared observations, as well as past variances to model variance at a given point in time.

Introduction of the dataset:

The original monthly rainfall data is extracted from the Bureau of Meteorology station number 86232. This series includes data from 1964 to 2022. There is missing data for certain years, including 1994 to 1998, as well as null values in between some years. Damaged instruments, absence and illness of an observer may result in gaps in records at stations.

A subset of the rainfall data after 1998 until the end of the 2021 has been used, where the completely missing years were removed, which resulted in 276 observations with 21 null values in different years. The observations are coming from consecutive months from the year 1999 onwards.

Imputation method for Null values:

The method used for imputation of the null values is by Median of monthly rainfall for the period of 1964- 2022. Then these values have been compared with the 'Mean rainfall (mm) for years 1970 to 2022' from the station 'Melbourne airport'.

Few stations of the Bureau of Meteorology have completely unbroken records of climate information. The Melbourne airport (station number 086282) is one of the stations which has collected unbroken information for the majority of years it has been running, for many different statistical elements.

The variance between the two elements were below 10% for over 65% of the data points and as such, it was decided that the Median per month would be used, after comparing with the 'Melbourne airport' rainfall data.

This report will discuss the methodology to find the best fitting model for monthly Rainfall Climate Data using trend models, ARMA, ARIMA, SARIMA, GARCH and ARCH and provide predictions for the next 12 observations using the best model.

Aim:
The aim of this report is to analyse the data for the monthly rainfall of the Melbourne Botanical Gardens, and then forecast the next 10 observations using the best applicable model.

Method:

- Step by step what we did/will do

1. Load and plot data- Using descriptive analysis, identify any unusual observations, potential issues, and gain an understanding of the trends and shape of the data.
2. Clean data, and ensure that it is suitable for further analysis.

3. Identify possible models to graph against this data.
4. Execute Linear Model to test if this model is suitable
5. Execute Quadratic Model to identify if this model is suitable
6. Test Seasonal Model to analyse if the model is viable to forecasting
7. Test Cosine Model to observe how viable is to forecast the values
8. Execute ARIMA and identify possible models that can fit the values
9. Apply ARCH/GARCH and identify possible GARCH model to join with the ARIMA models
10. Based on the previous results, organise data and the best possible model to forecast the following 10 months of rainfall in Melbourne

Results:
DESCRIPTIVE ANALYTICS :
Statistical Values :

The data is loaded with a frequency of monthly basis.

The data has a mean of 47.87 and a median of 43.30. It appears to be skewed to the right. Most of the data falls between 20-60, and the rainfall ranges from 1 to 161.8 with an interquartile range of 33.3 (see Appendix -Section 01 for Figure 01)

According to Appendix -Section 01, Figure 02 the histogram of the original data reflects a right skewness. The descriptive analysis provides the indications of existing trend, seasonality, behaviour, changing variance and change points of the time series data.

Time series Plot :

Figure 2: Time Series Plot based on our current dataset

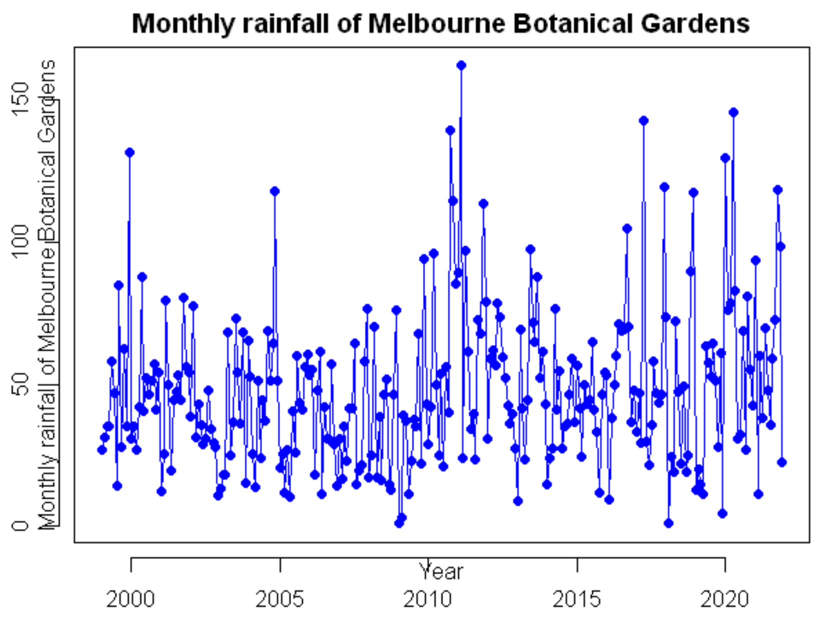


Figure 2 shows a time series graph of the monthly rainfall in the Melbourne Botanical Gardens. Based on the time series graph, there are no obvious trends presented in the dataset. The current plot appears to have a regular rise and fall, which can be an indication of seasonality. Since the data presents with a relatively stable mean, the plot does not appear to have a large change in variance. As the graph exhibits fluctuations in consecutive points, it indicates MA behaviour. There are no drastic changing points, and as such, no intervention points to consider.

For the periodic nature of the collected data, the first approach to understand the seasonality of the data is illustrating the time series plot with values labelled with the period of collection. According to Figure3 in Appendix, seasonality can be visible in the rainfall series where most of March & December months have been in higher ends while most January values fall in the lower end.

Scatter Plot :

A scatter plot of rainfall series (Figure 4 in Appendix) is used to analyse whether the consecutive rainfall values are correlated. As per Figure 02, a strong correlation between consecutive months cannot be observed

$$\rho = 0.1710105$$

Quantile-Quantile Plot :

As per Figure 5 in Appendix, the tails are off from the reference line in the QQ plot from both ends and does not seem to support well with the assumption of a normally distributed stochastic component. The Shapiro-Wilk normality test's p-value (2.897e-10) is lower than 0.05 significance level. Therefore, the null hypothesis is rejected which is 'the data is normally distributed'. The original monthly rainfall is not satisfactory with the normality assumption.

Shapiro-Wilk normality test

data: railfall_TS **W** = 0.92846, **p-value =** 2.897e-10

Auto-Correlation Plots :
Figure 3 shows the ACF plot and PACF plot of the time series. According to these plots, it seems there is only one significant lag in the ACF plot and one significant lag in the PACF plot and all other lags are within the dash lines. Also, there is no strong wave-pattern within the lags based on the ACF plot above. Therefore there is not sufficient evidence to support seasonality in the model.

Paired with the previous plots, Dickey-Fuller and Unit Root test produce a p-value lower than 0.05 (Alpha), which means that there is enough evidence to reject the null hypothesis that the Melbourne Rainfall data is not stationary .

Augmented Dickey-Fuller Test

ata: railfall_TS

Dickey-Fuller = -5.5098, Lag order = 6, p-value = 0.01 alternative hypothesis: stationary

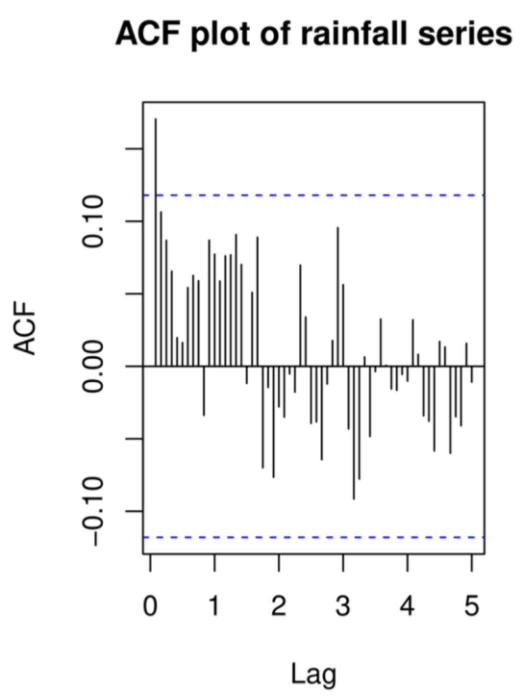
Phillips-Perron Unit Root Test

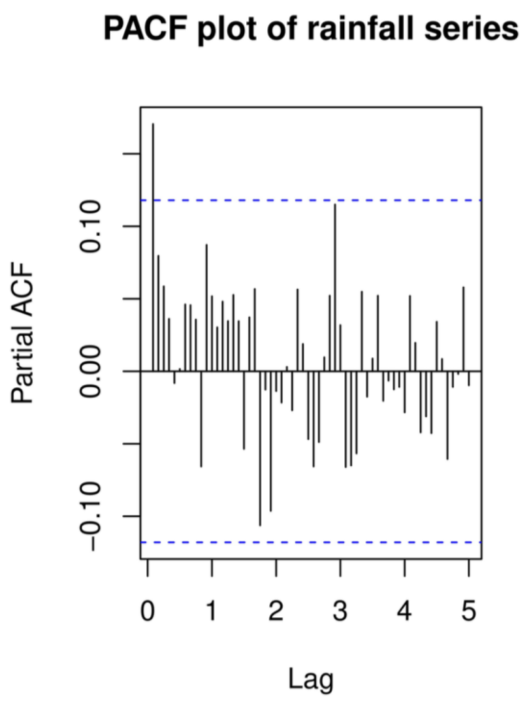
data: railfall_TS

Dickey-Fuller Z(alpha) = -251.46, Truncation lag parameter = 5, p-value = 0.01

alternative hypothesis: stationary

Figure 3: ACF and PACF plots of current time series





MODELS ANALYSIS:

The following findings are the result of the analysis of a variety of model techniques to identify the best time series model for the Melbourne Rainfall data.

Linear Model:

According to the results seen in Figure 07 in Appendix, the time series plot of standard residuals does not reflect a random pattern. Residuals have outliers over +4 in the histogram and right skewed asymmetric distribution is displayed.

The tails are off from the reference line in the QQ plot in this model, meaning it does not support the assumption of a normally distributed stochastic component. The Shapiro-Wilk normality test calculates the correlation between the residuals and the corresponding normal quantiles. The p-value (1.416e-09) is smaller than 0.05, therefore it is concluded that it is appropriate to reject the null hypothesis which is ‘the data is normally distributed’.

According to the ACF plot of this model, the first lag is above the horizontal dashed lines, or the confidence bound and a wave-like pattern can be seen in the ACF plot which confirms the seasonality which is not captured by the model.

Figure 08 (see Appendix) provides the trend line of the linear model. The fit is low and further confirmed by the Adjusted R squared value, which is 0.02332. According to the multiple R squared value of this model, about 2% of the variation in the rainfall series is explained by the linear trend model.

Quadratic Second Level and Third Level Model:

The time series plot of standard residuals does not reflect a good random pattern (see Figure 10 is Appendix). The residuals do not seem to capture larger outliers over +4. The QQ plot's tails are off and the Shapiro-Wilk normality test's p-value (1.785e-09) is lower than 0.05, therefore it is concluded it is appropriate to reject the null hypothesis which is 'the data is normally distributed'.

According to the ACF plot of this model, except for the first large lag which is above the confidence bound level all other lags are below the confidence bound level. The wave-like pattern is visible and has not captured the seasonality component. (See Appendix Figure 9 to 14 for more information)

Although the fitted line is improved slightly compared to the Linear trend model , both Quadratic second level and third level did not reflect expected improvements.

Seasonal Model & Cosine Model:

Seasonal model uses a cosine curve to generate the seasonality of the series data. As per Figure 15-20 in the Appendix section, all the coefficients are statistically significant at 5% level in this model and the adjusted R-squared value is 0.7556. This reflects a better model fitting compared to Linear and Quadratic trend models. However the model fitting is not quite perfect looking at the moving fitted line with the actual data.

With the cosine model as per Figure 21-23 in the Appendix section, all the coefficients aren't statistically significant at 5% level for this model and the adjusted R-squared value is 0.0115. Based on these statistical figures, it shows that the cosine model is not a good fit for the actual time series data compared to seasonal models.

Based on the analysis of seasonality and the previous results from the cosine and seasonal model, SARIMA cannot provide suitable models for the current data. However, it is possible that ARIMA models could offer suitable models for Melbourne Rainfall data because of the Moving Average behaviour of the data.

ARIMA Model:

In order to apply the ARIMA model the frequency is dropped from the time series object.

Transformation :

Transformation will help to improve normality within the data because the assessment of the Shapiro-wilk test rejects the null hypothesis that the raw data has a normal distribution.

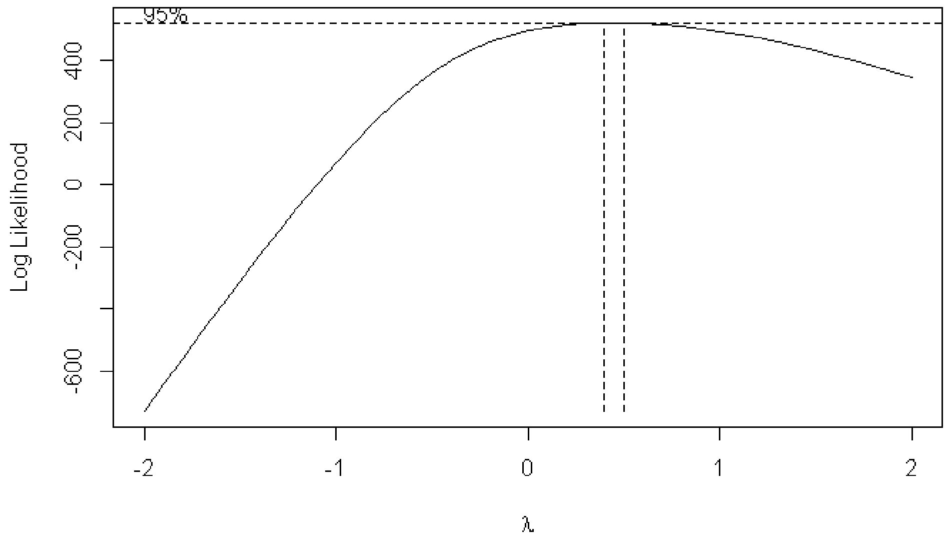
To evaluate the transformation the function box_cox_analysis() is applied.

Using the Shapiro-Wilk normality test,the tails of the QQplot are thinner and the P-value = 0.36 is greater than α=0.05.Thus, normality is improved by applying transformation with λ=0.4. Further tests such as Augmented dickey-fuller test and Phillips-Perron Unit Root test have indicated stationarity was not affected by the box-cox transformation.

ρ=0.15735

The correlation was not significantly affected by this transformation.

Figure 04. Lambda Likelihood plot of the Melbourne Rainfall data.



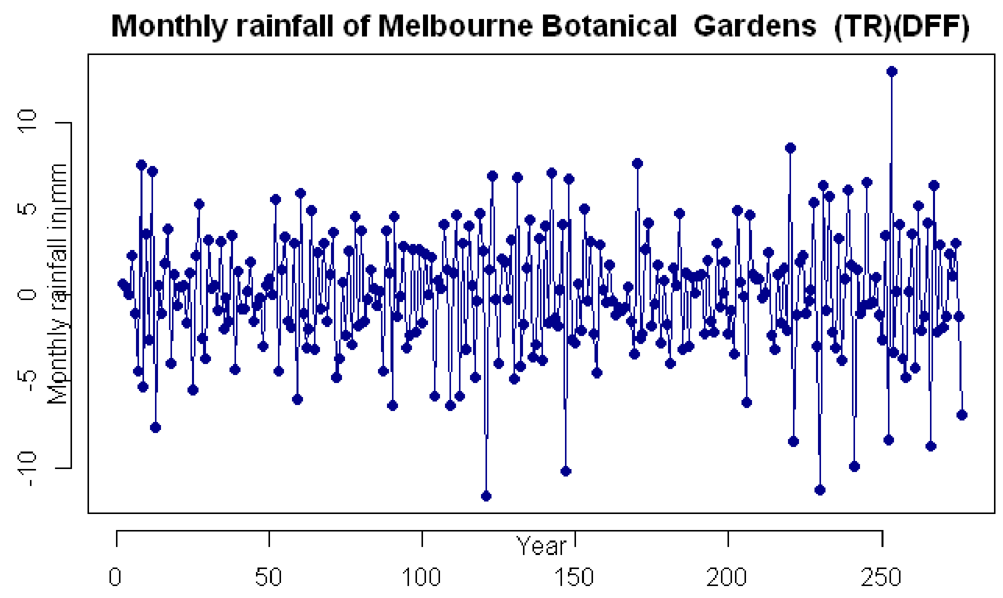
λ=0.4

P-value=0.36

<u>Difference :</u>

It is possible to apply differencing between the values to observe if the behaviour of the data could be improved compared to the transformation. However, the time series plot does not show any significant improvement that could suggest that a difference will get better with ARIMA models.

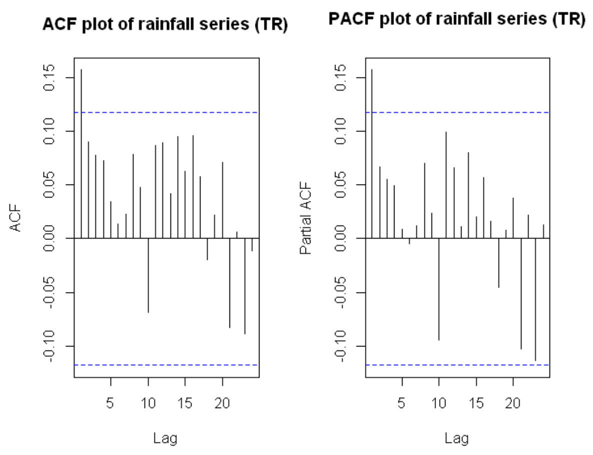
Figure 05: Time series plot of 1st Transformation of Monthly Rainfall



Specification :

- *ACF PACF SPECIFICATIONS* : The autocorrelation plots illustrate just a significant lag for ACF and PACF. Therefore, the possible models from this specification are ARIMA {1,0,1} {0,0,1}

Figure 06. ACF and PACF of Transformed Melbourne Rainfall data.



- *EACF SPECIFICATION :* Following the top left “o” the Extended Autocorrelation matrix (See Figure 25 in Appendix) suggest the following models : ARIMA {0,0,1} {0,0,2} {1,0,1} {1,0,2}. This specification shares a few models from the ACF and PACF graphs.
- *BIC* : (See Figure 26 in Appendix) Based on the max AR and MA from the eacf(), BIC will run with the same parameters to keep consistency in the analysis. The resulting graph

suggests ARIMA {1,0,0} as the strongest model in the first row and ARIMA {1,0,4} and {1,0,3} from the second and third line.

As a result, the following models are proposed to evaluate the quality of the model:

ARIMA {1,0,1} {0,0,1} {1,0,0} {0,0,2} {1,0,2} {1,0,3} {1,0,4}

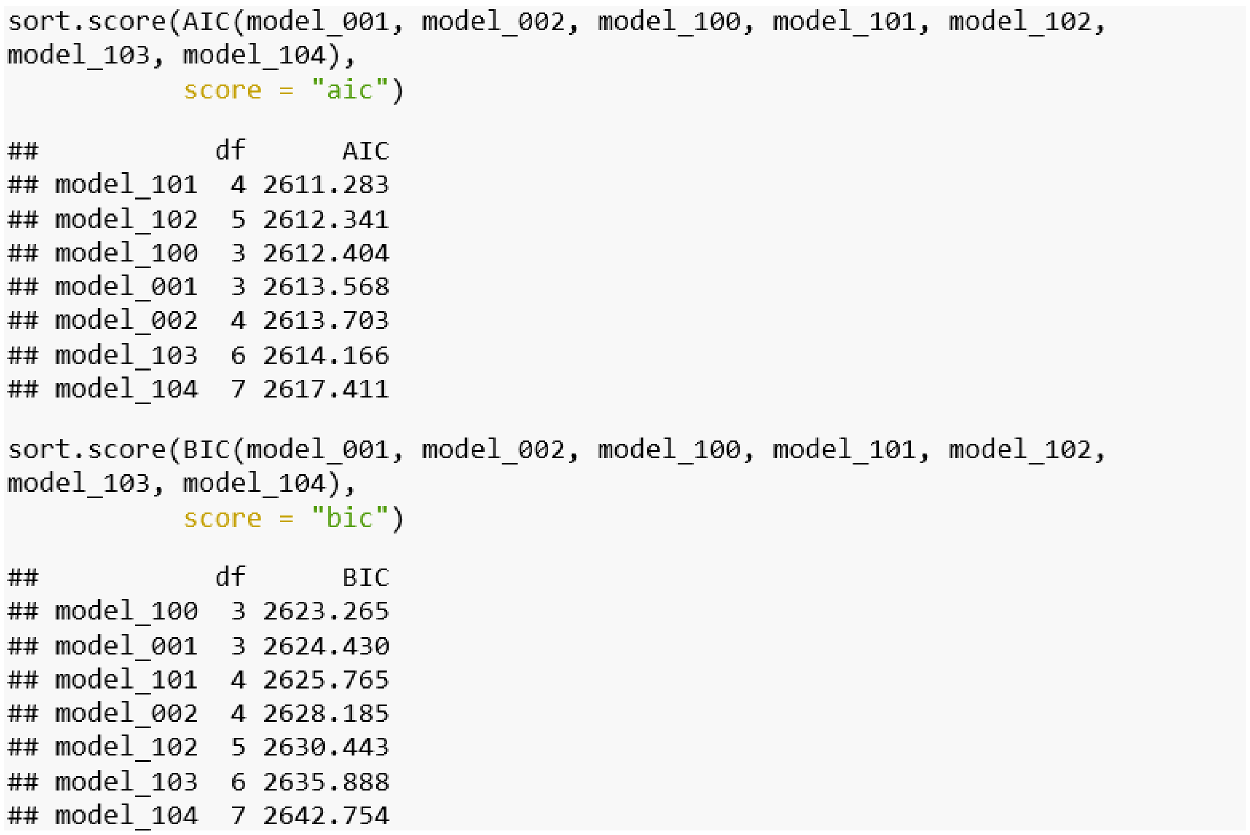
Model Diagnostics :

The function arima_modelling() will help to analyse the coefficients for each proposed model. Additionally, a for loop is applied in order to create and assign variables for future steps.

(Appendix Figure 27-33)

The coefficients give a result indicating that increment of moving average over one is not suggested to identify the fittest model. Therefore, the models with significant coefficients are : **ARIMA {1,0,0} {0,0,1}** However, it is important to identify which models are the best based on AIC and BIC results. For this reason, sort.score() is used to organise the models.

Figure 07. AIC and BIC ranking.



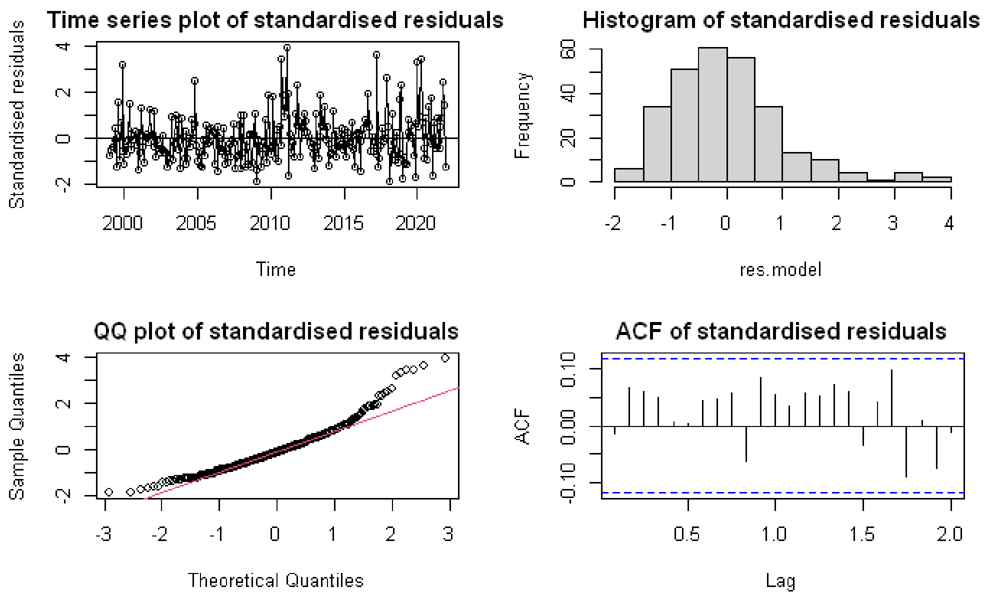
```
sort.score(AIC(model_001, model_002, model_100, model_101, model_102,
model_103, model_104),
           score = "aic")

##             df      AIC
## model_101  4 2611.283
## model_102  5 2612.341
## model_100  3 2612.404
## model_001  3 2613.568
## model_002  4 2613.703
## model_103  6 2614.166
## model_104  7 2617.411

sort.score(BIC(model_001, model_002, model_100, model_101, model_102,
model_103, model_104),
           score = "bic")

##             df      BIC
## model_100  3 2623.265
## model_001  3 2624.430
## model_101  4 2625.765
## model_002  4 2628.185
## model_102  5 2630.443
## model_103  6 2635.888
## model_104  7 2642.754
```

The results suggested that model_101 is better based on AIC and model_100 based on 100. However, it is important to highlight that model_101 did not show good results using the Maximum Likelihood method.

<u>Analysis of Residuals:</u>

The diagnostic tools will provide deep information about the final candidates models {1,0,1} and {1,0,0}.

Figure 08. Residuals plots for ARIMA {100}



Shapiro-Wilk normality test

data: res.model W = 0.93616, p-value = 1.517e-09

Figure 09. Residuals plots for ARIMA {100} with Ljung Box

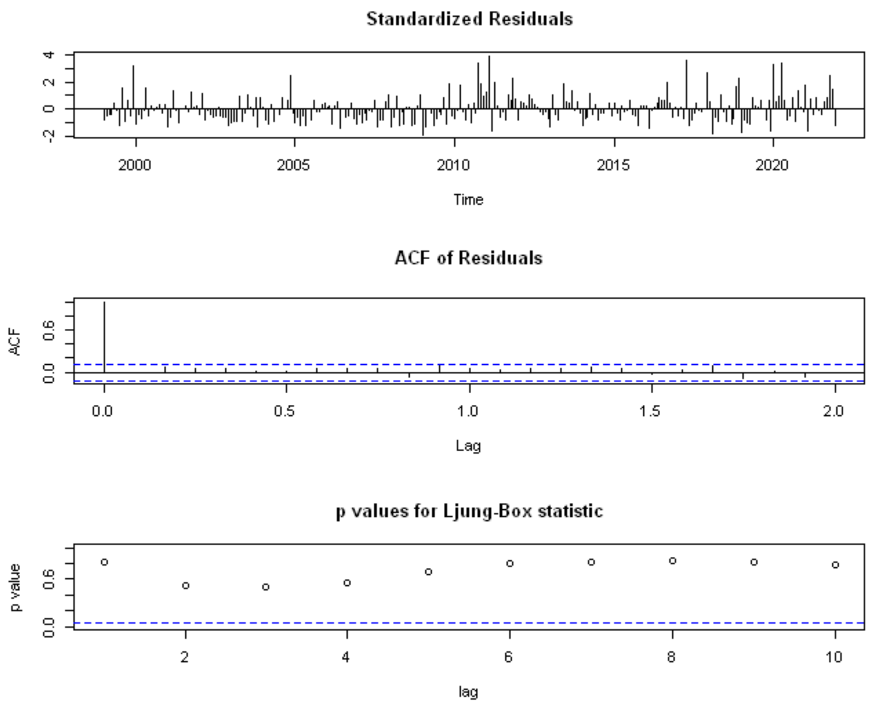
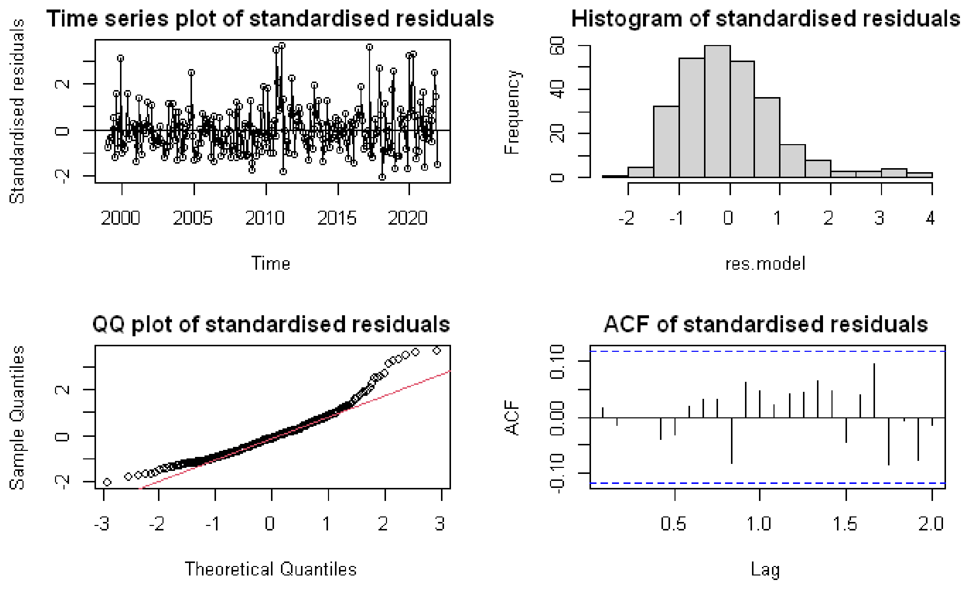


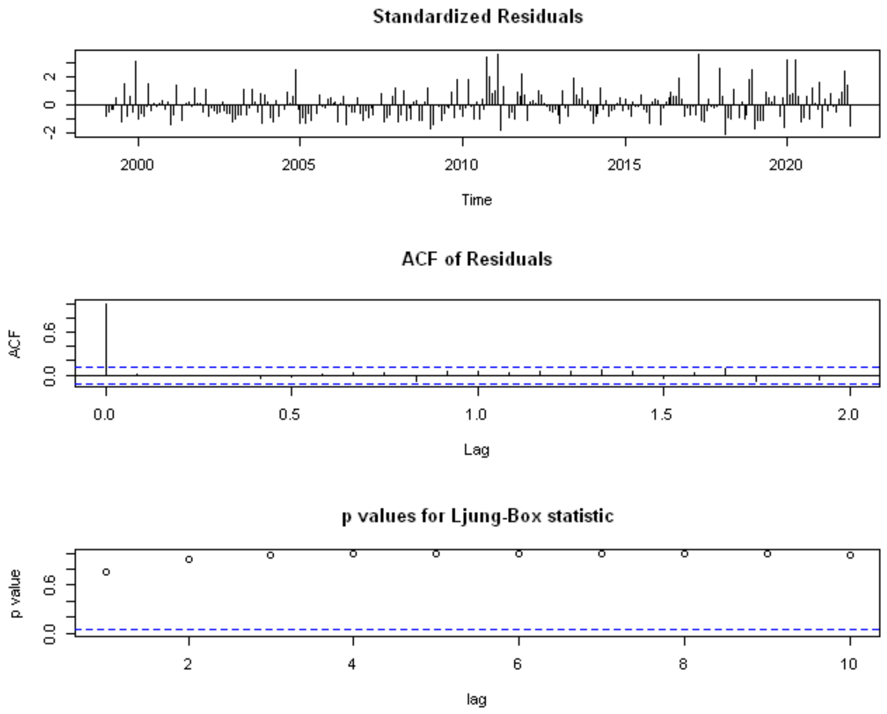
Figure 10. Residuals plots for ARIMA {101}



Shapiro-Wilk normality test

data: res.model W = 0.94089, p-value = 4.471e-09

Figure 11. Residuals plots for ARIMA {101} with Ljung Box



Both residuals have similar results. For instance, there is no normality distribution within the residuals of each model based on the Shapiro-Wilk test and the QQplot. Additionally, The time series plot for these models do not show a random pattern among the success points and the volatility of the data is still observable. Finally, there are large lags in the ACF plot in each model.

Although these models may not be the best option for the rainfall data of Melbourne, it is important to select one of these possible candidates because it could be useful for other models such as ARCH or GARCH. For this reason, overfitting was applied as a final diagnostic tool to analyse if any improvements could be made.

<u>Overfitting :</u>

These are the possible models applying overfitting (See Appendix Figure 34-37) :

ARIMA MODEL_100 : {2,0,0} {1,0,1}
ARIMA MODEL_101 : {2,0,1} {1,0,2}

From this model {1,0,1} is already one of the candidates to best model and {1,0,2} was already analysed with negative results in the coefficients. Then, it is important to analyse the two new models.

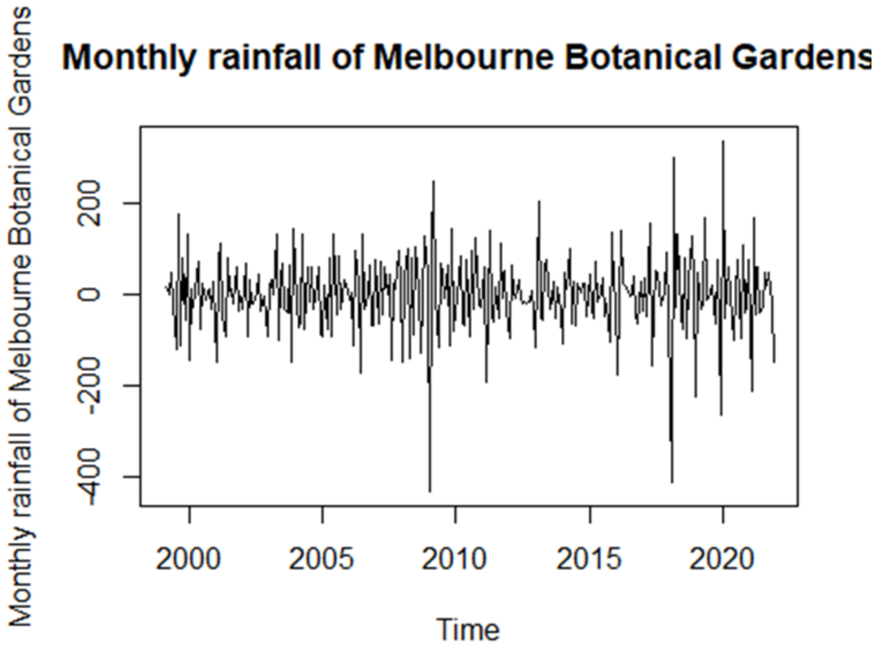
Selected Model : The over fitted models did not show any improvement for their coefficients, at least one coefficient was not significant. As a result, a considering previous analysis of the ARIMA model. The best candidate is the model ARIMA {1,0,0}. However in the analysis of residuals it can be seen that the volatility of the variance is still there. For this reason, GARCH and ARCH models will be tested, as it is likely that it would be able to reduce the variance in the model.

GARCH + ARCH Model:

We need to have this stationary series to test for the existence of ARCH effect or volatility clustering.

To achieve this, we define a return series for a time series. Similarly, the frequency is dropped to create the time series for this model.

Figure 12. Return time series plot



In the return series, the volatility is obvious and there is no sense of trend or seasonality.

<u>Test</u> :

Augmented Dickey-Fuller Test

data: r.rainfall **Dickey-Fuller =** -10.524**, Lag order** = 6, **p-value** = 0.01

alternative hypothesis: stationary

Shapiro-Wilk normality test

data: r.rainfall **W =** 0.9532, **p-value = 1**.02e-07

Figure 13. Q-Q plot returned time series

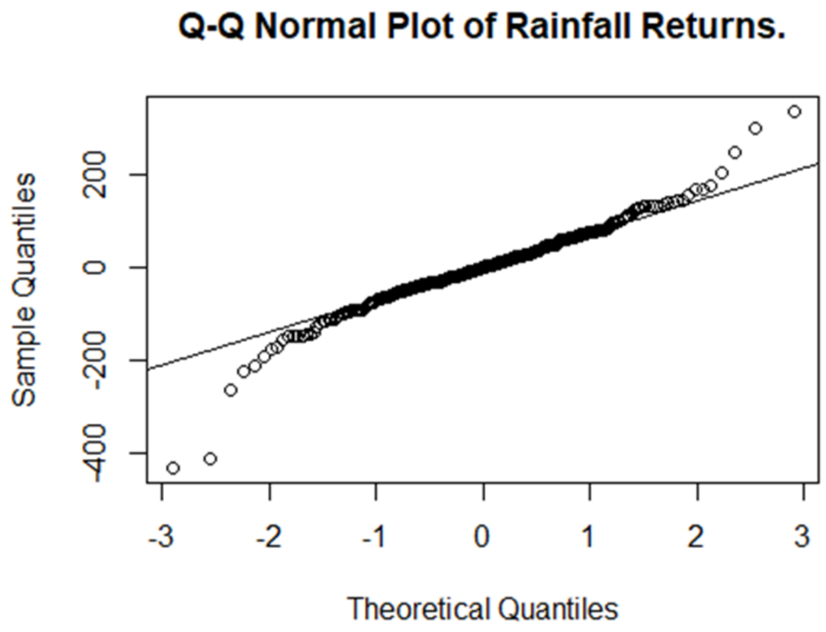
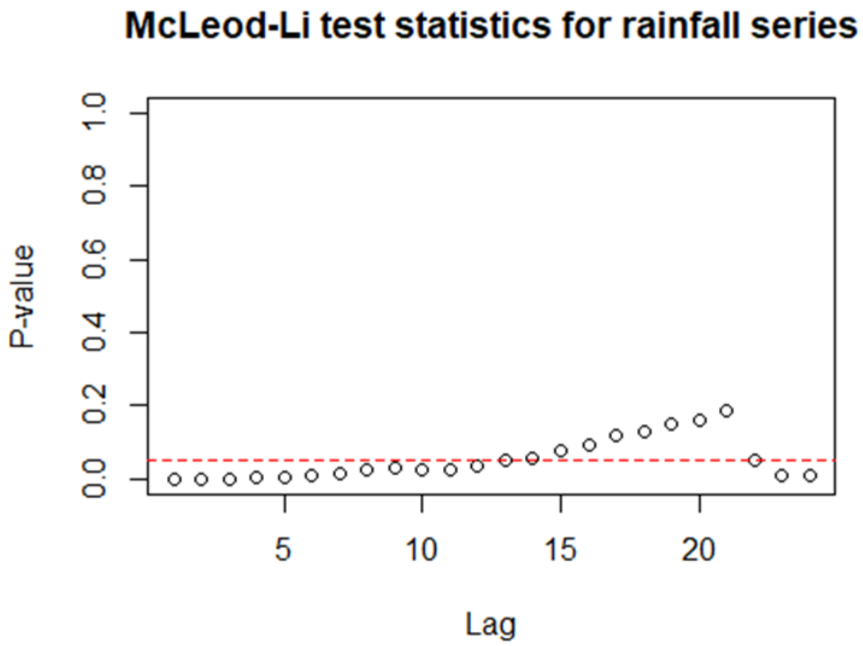


Figure 14. McLeod-li Test for Rainfall Time Series



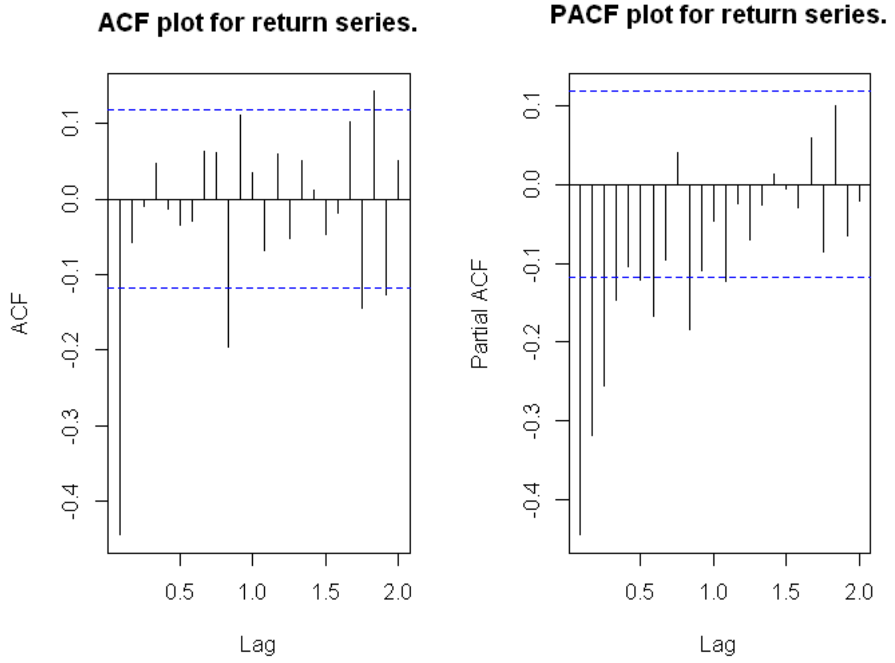
McLeod-Li test indicates volatility clustering in this series. # So, we need to capture both trend and changing variance by applying an ARMA + GARCH

<u>Specification :</u>

- ***ACF and PACF :*****We will first specify the orders of ARMA part. Then we will use the residuals of ARMA part to specify the orders of GARCH part. According to ACF and PACF,**

  P=4 Q=1
  ARMA {4,1}

  Figure 15. ACF and PACF time series.



- *EACF :* Following the top left “o” the Extended Autocorrelation matrix suggest the following models *{ARMA (0,1),ARMA (0,2),ARMA (1,1),ARMA (1,2)} (See Figure 38 in Appendix)*
- *BIC:* Based on the max AR and MA from the eacf(), BIC will run with the same parameters to keep consistency in the analysis. The resulting graph suggests ARMA (1,3),ARMA (2,3)} *(See Figure 39 in Appendix)*

As a result, the following models are proposed to evaluate the quality of the model:

ARMA (4,1), ARMA (0,1), ARMA (0,2), ARMA (1,1), ARMA (1,2),ARMA (1,3),ARMA (2,3)

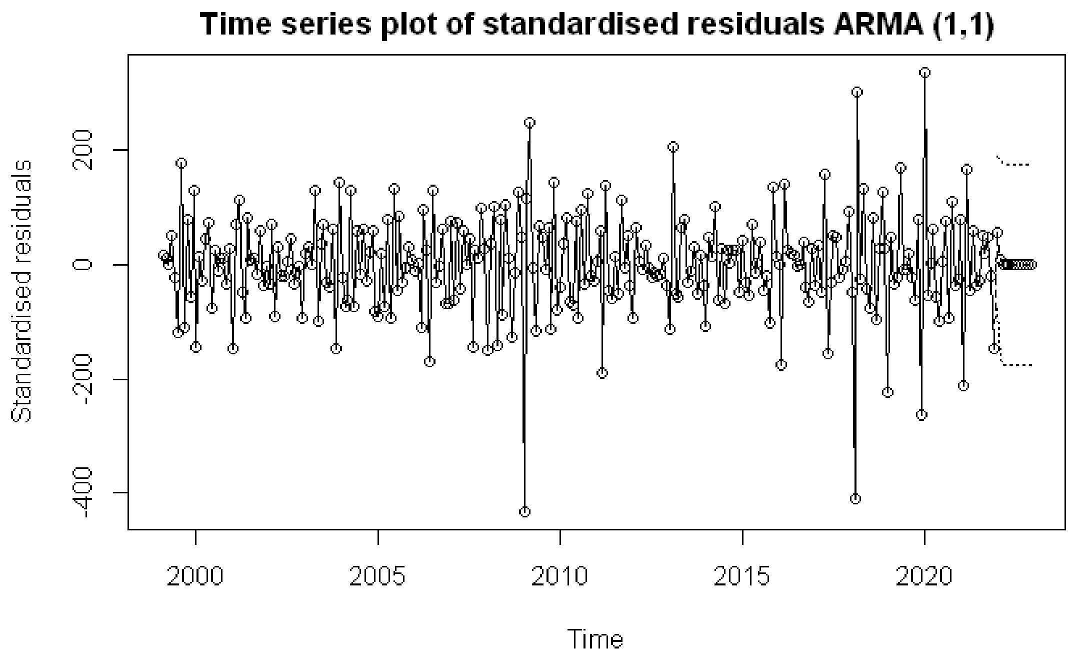
<u>Model Diagnostics :</u>

The function arima_modelling() will help to analyse the coefficients for each proposed model. Additionally, a for loop is applied in order to create and assign variables for future steps.

<u>Selecting GARCH model :</u>

Based on the previous results the current plot of this the current plot for the time series of Malborune Rainfall using ARMA (1,1) model :

Figure 24. Time series plot ARMA (1,1)



- *ABSOLUTE APPROACH:* The method of this approach is to transform the return series into absolute values. Then, it is possible to determine the possible GARCH models plotting specification tools.

  Figure 25. ACF and PACF Absolute transformation

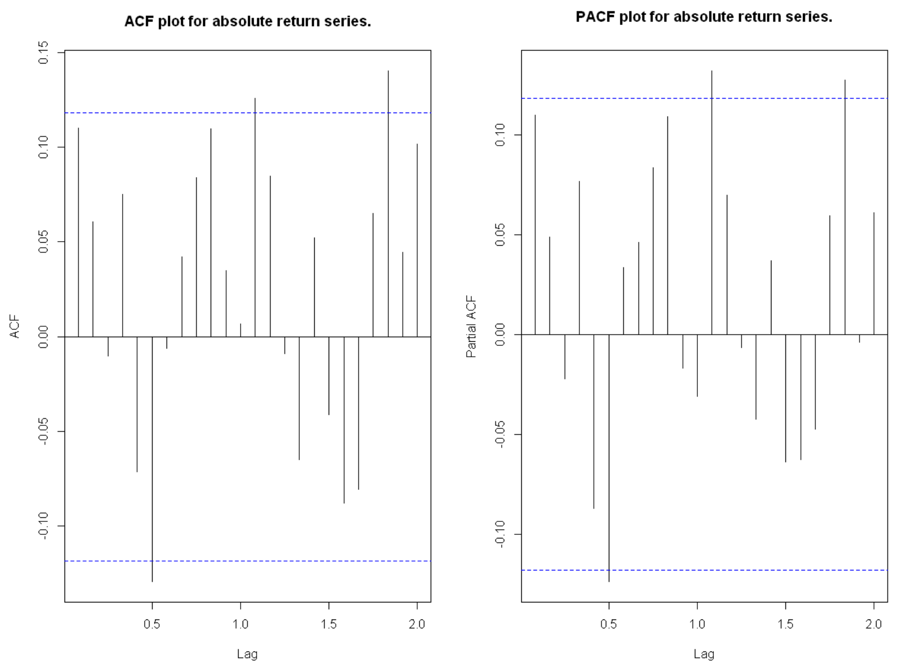
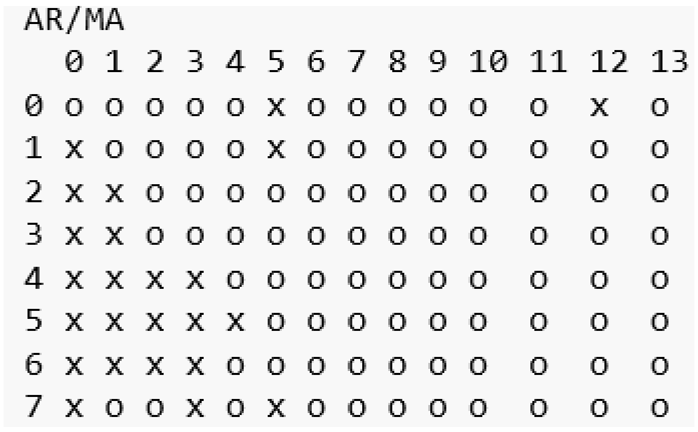


Figure 26. EACF Absolute transformation


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

As a result from the autocorrelation plots, GARCH {0,1} and {1,1} are selected based on the number of lags out of range. EACF did not show significant models because max(p,q) values were not able to provide a possible combination of models.

- *SQUARED APPROACH :* Similarly this method transforms the return series into squared values. Then, it is possible to determine the possible GARCH models plotting specification tools.

Figure 27. ACF and PACF Squared transformation

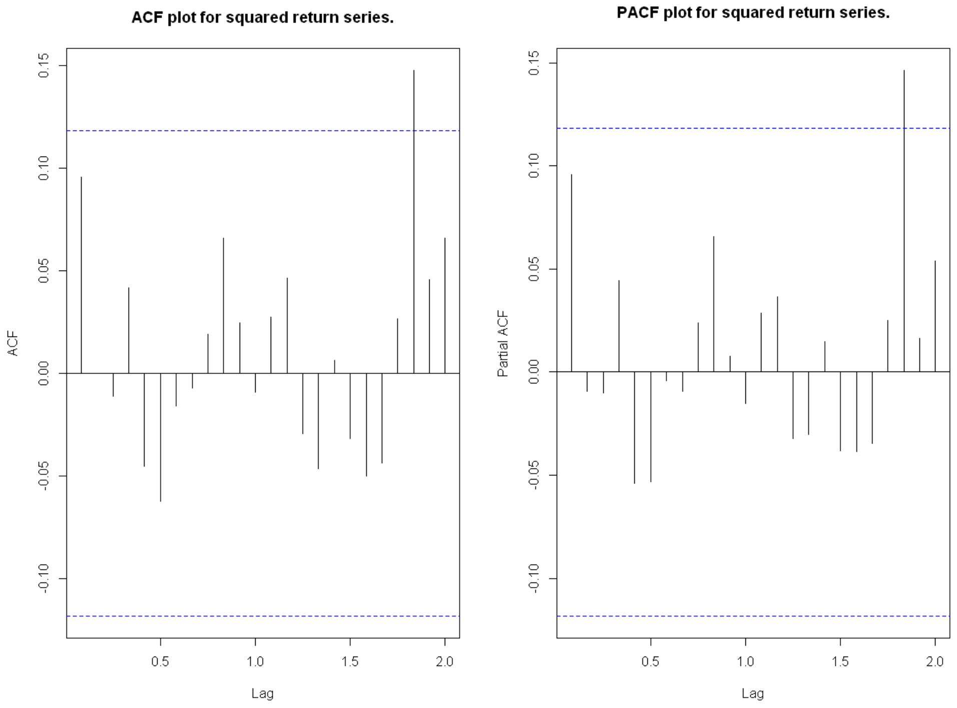
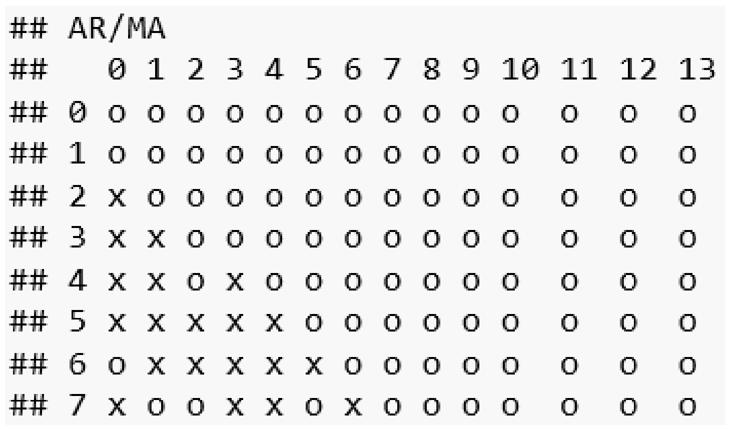


Figure 28. EACF Squared transformation



```
## 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 x o o o o o o o o o  o  o  o
## 4 x x o x o o o o o o o  o  o  o
## 5 x x x x x o o o o o o  o  o  o
## 6 o x x x x x o o o o o  o  o  o
## 7 x o o x x o x o o o o  o  o  o
```

Overall, none of the specification tools show a valid combination of values for P and Q, which could provide a GARCH model.

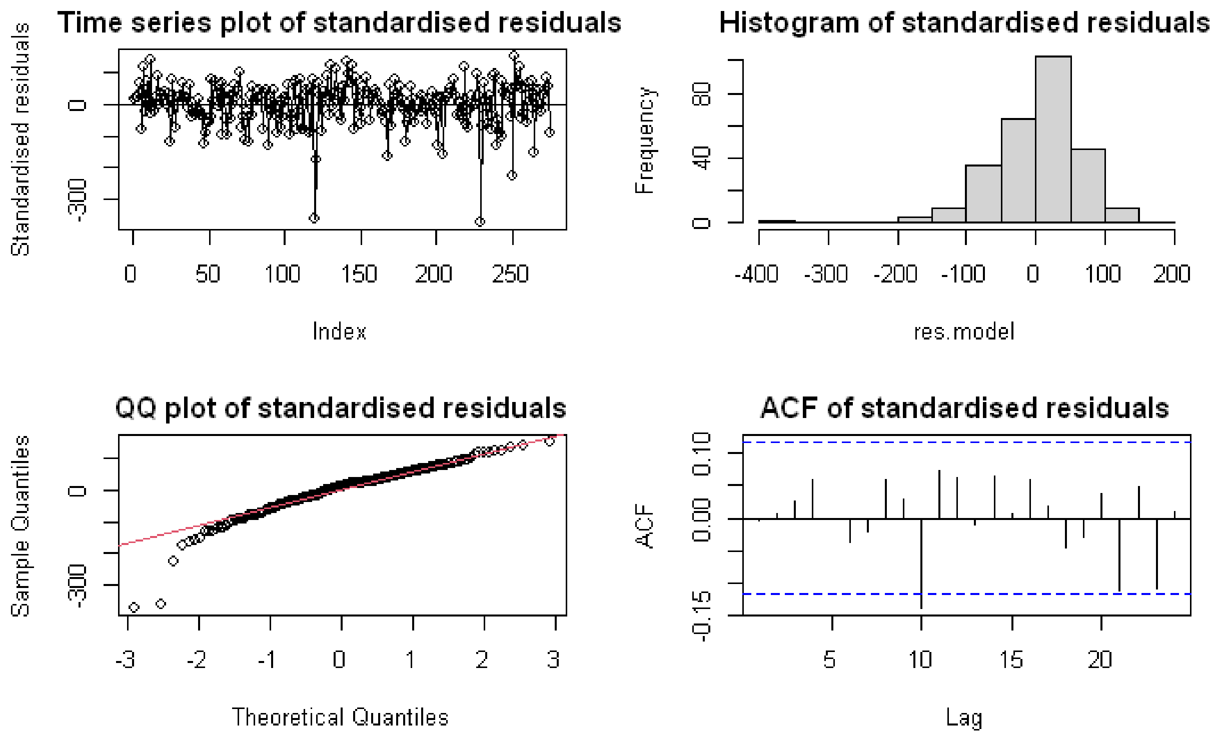
In conclusion, GARCH {0,1} and {1,1} are selected as possible models for this return time series. However, an evaluation of AIC and BIC illustrated that ARMA {1,1} + GARCH {0,1} is the best model to forecast the following 10 months of rainfall in Melbourne.

Figure 29. AIC and BIC values

```
                              AIC      BIC
ARMA(1,1)+GARCH(0,1) 11.30813 11.37389
ARMA(1,1)+GARCH(1,1) 11.30657 11.38548
```

Additionally, the analysis of residuals suggests that volatility is significantly reduced for the selected model.

Figure 30. Residual analysis



FORECASTING

Figure 31. Standard deviation superimposed

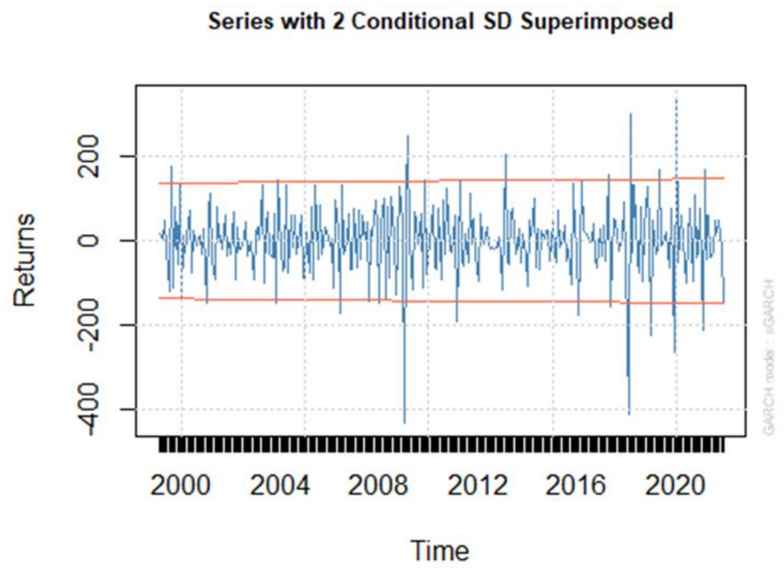


Figure 32. Conditional Standard Deviation Vs Returns

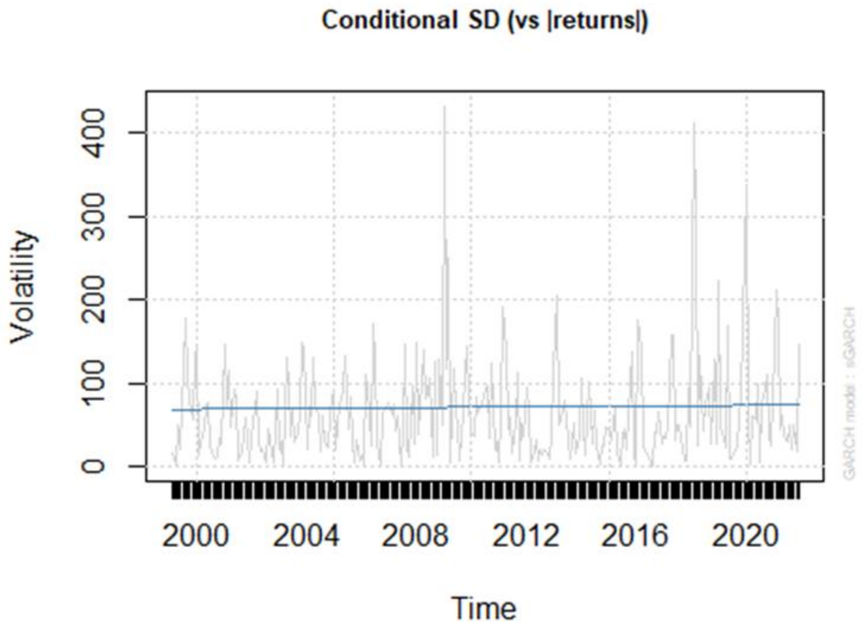
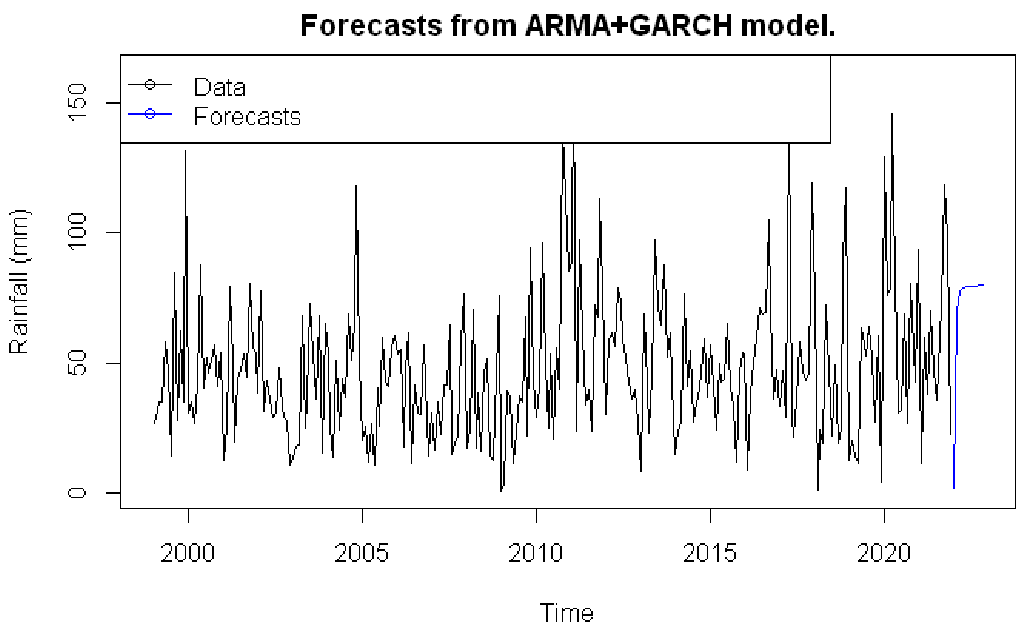


Figure 33. Forecasting of the following 10 months



Discussion:

Trend models:

Model	P value of Overall model	P value of coefficients	Adjusted R Squared
Linear Trend model	not statistically significant at 5% significance level	not statistically significant at 5% significance level	0.02332
Quadratic Trend model - Order 2	not statistically significant at 5% significance level	not statistically significant at 5% significance level	0.02352
Quadratic Trend model - Order 3	not statistically significant at 5% significance level	not statistically significant at 5% significance level	0.02351
Seasonal Trend model	Statistically significant at 5% significance level	Statistically significant at 5% significance level	0.7556

Cosine model	not statistically significant at 5% significance level	not statistically significant at 5% significance level	0.0115

ARIMA models

	p	d	q	ARIMA models
ACF /PACF	1	0	1	ARIMA (1,0,1)
EACF	0,1	0	1,2	ARIMA (0,0,1), ARIMA (0,0,2), ARIMA (1,0,1),ARIMA (1,0,2)
BIC table	1	0	0	ARIMA (1,0,0)

GARCH+ARCH Model

	p	q	GARCH+ARCH Model
ACF /PACF	4	1	ARMA (4,1)
EACF	0,1	1,2	ARMA(0,1), ARMA (1,1) ARMA(0,2), ARMA(1,2)
BIC table	1,2	3	ARMA(1,3), ARMA(2,3)

There were many possible models that were considered for this project, and each was considered carefully to decide on the most accurate.

The linear model was decided against, as the p-value (1.416e-09) was smaller than 0.05. As well as this, according to the ACF plot of this model, the first lag is above the confidence lines and a wave-like pattern can be seen in the ACF plot which implies there is seasonality in the data which was not captured by the model.

Because of this, the next model tested was the quadratic model. This model was similarly rejected, as the QQ plot's tails were straying from the expected bounds, and the Shapiro-Wilk normality test's p-value (1.785e-09) was lower than 0.05. According to the ACF plot of this model, except for the first large lag which is above the confidence bound level all other lags are below the confidence bound level. The wave-like pattern is visible and has not captured the seasonality component.

Based on the results of the seasonal and cosine models, it was reasonable to assume that those models do not have enough evidence to prove that the data has seasonality, or to suitably describe the data. Therefore, the SARIMA model was bypassed, as it was concluded it would not provide any important results, as an established frequency is required to proceed with these types of models.

Although the selected ARIMA model achieved the parsimony principle and a normal distribution, the model is not accurate enough to map the volatility of variance of the Melbourne Rainfall. For this reason, it is necessary to try methods such as ARCH/GARCH models to address the volatility of the data, and to get the most accurate forecasting possible.

Finally, ARCH/GARCH was tested, as ARCH/GARCH models are able to control the volatility of the variance for data. The results of this are visible on the analysis of residuals, which show time series plot a high degree of randomness compared to previous models.

The forecast was created by an ARIMA + GARCH model, as this method provides a prediction of the mean and the variance for the Melbourne Rainfall time series. This was the optimal choice, as the selected model was able to deal with the moving average behaviour, stationarity and changing variance of the time series in accordance with the type of data and the current models applied.

The forecasted data looks reasonable (see Figure 33 above) compared to the historical values. The fit of the forecasted 10 months aligns with the historical series. These forecasts are also reasonably in line with the graphs created by the Bureau of Meteorology.

The selected best model is ARMA {1,1} + GARCH {0,1} for forecasting. In Figure 33, we observe larger fluctuations cluster together. This means that there is volatility clustering in the rainfall series.

The GARCH model assumes that the conditional mean of the time series is zero which does not hold always in rainfall series data. In this case the conditional mean structure is modelled by ARMA {1,1) model.

Compared to all the tested models for rainfall series , ARMA {1,1} + GARCH {0,1} has given the best fit in terms of parameter significance and diagnostics. This chosen hybrid model predicts the changing variance of rainfall series with the existence of volatility clustering.

One limitation encountered when beginning to analyse this data, was the fact there were so many missing values. As entire years of observations were missing, it was important to consider what methods should be implemented to clean the data and make meaningful forecasts with it. Cutting the missing years was a logical decision, but did cut out a large portion of possible data that could have given further insight into the rainfall in the Melbourne Botanical Gardens. It may have been advantageous to consider more rainfall stations' data, especially those that were more consistent with data gathering.

When attempting to create the Ljung Box plot of various models, there was a repeated issue with the visualisation of the Ljung Box. Further investigation into why this was an issue would be beneficial, to create a more cohesive summary of the models used.

Rainfall data is difficult to predict, as rainfall is affected by many different variables, such as wind speed, temperature and ENSO patterns. Though the forecasts are as accurate as possible using the chosen model, the amount of variables that affect rainfall contribute to making the issue something that requires more information to be more accurately predicted. In the future, using multivariate data to create forecasts that take these variables into account would be beneficial. If further research were to be conducted, the World Climate Research Program's Coupled Model Intercomparison Project would be a resource that could be of assistance choosing which data to analyse to create more accurate forecasts.

The analysis of rainfall data has real world impacts. Rainfall data facilitates decisions about cropping patterns and sewing dates on farms, the updating of irrigation, the construction of roads, and drinking water distribution in both urban and rural areas. Governments and farmers both need accurate rainfall data, and in the case of the Melbourne Botanical Gardens, the governing board of the gardens need this information to keep the plants healthy and not over or under watered.

Further analysis of this data could be completed in the future. Comparing this data and the forecasts against the ENSO variations could give more insight into how both El Nino and La Niña weather patterns affect the rainfall in Melbourne. As well as this, comparing the rainfall data against the rainfall in other parts of Melbourne, as well as other major cities in Australia would bring a more holistic image of how rainfall behaves and functions in Australia. This combined with a multivariate approach would allow for forecasts that account for more of the variables affecting rainfall.

Conclusion:
The purpose of this analysis was to create forecasts for the next ten months of rainfall in the Melbourne Botanical gardens. It was identified that the best fitting model to create these forecasts would be an ARMA {1,1} + GARCH {0,1} model, as it adequately addressed the moving average behaviour, stationarity and changing variance of the data. This model could be improved upon in the future by referencing the techniques used in the World Climate Research Program’s Coupled Model Intercomparison Project. This analysis shows a practical analysis of real-world data, which can be used to improve the response to rainfall changes in farming and legislative areas, as well as many others.

Appendix:
Section 01: Figures and plots for reference
1.1 Original data

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00   27.68   43.30   47.87   60.98  161.80

## [1] 33.3
```

Figure 01. Summary statistics of rainfall series

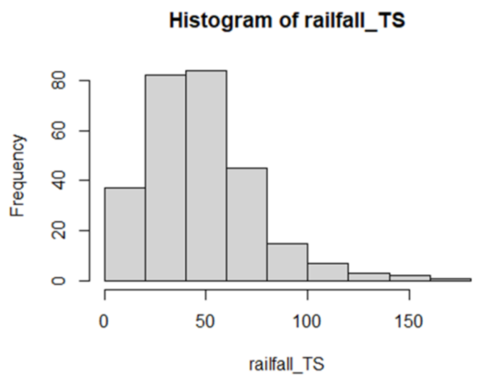


Figure 02. Histogram of rainfall series

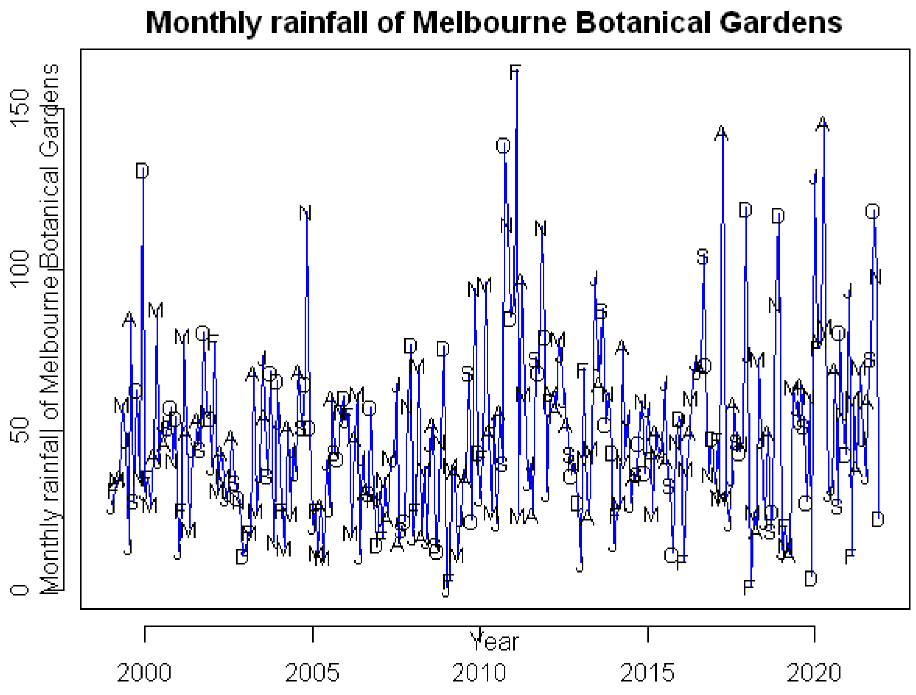


Figure 03. Seasonality plot of time series

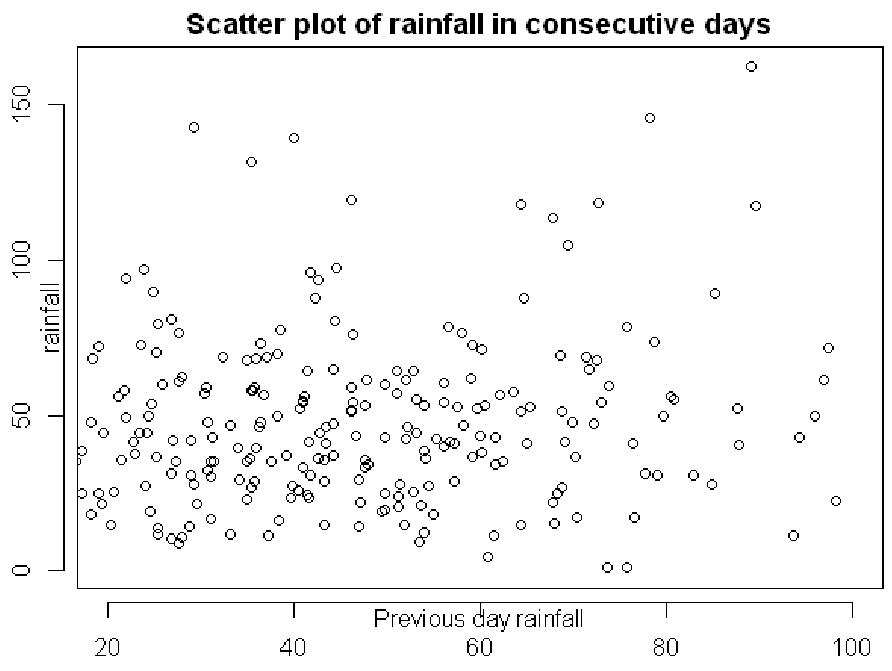


Figure 04. Scatter plot of time series

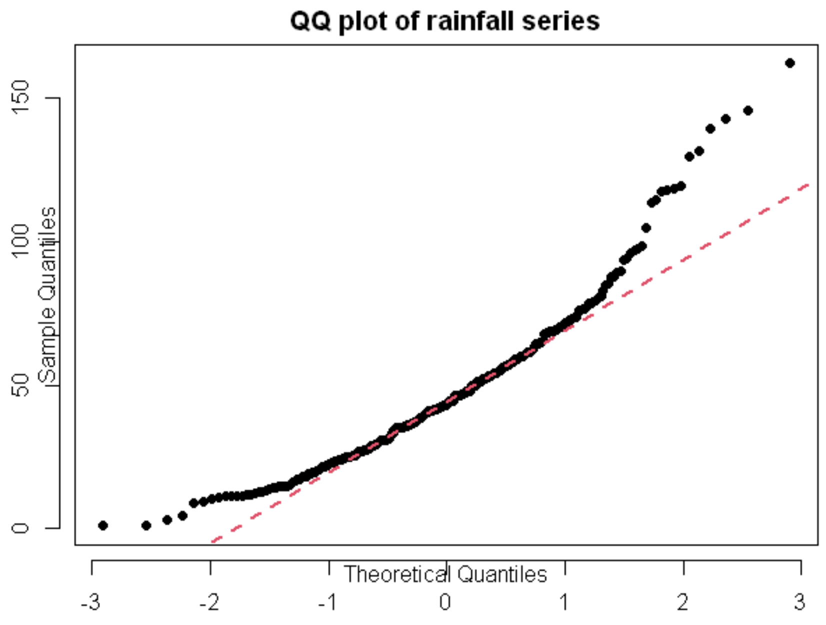


Figure 05. QQ plot of time series

1.2 Linear trend model

```
railfall_TS_lm = lm(railfall_TS ~ time(railfall_TS)) # label the linear trend
model as model1
summary(railfall_TS_lm)

##
## Call:
## lm(formula = railfall_TS ~ time(railfall_TS))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -51.863 -18.720  -4.483  13.275 113.505
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -1321.7040   497.9513  -2.654  0.00841 **
## time(railfall_TS)     0.6812     0.2477   2.750  0.00635 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.32 on 274 degrees of freedom
## Multiple R-squared:  0.02687,    Adjusted R-squared:  0.02332
## F-statistic: 7.565 on 1 and 274 DF,  p-value: 0.006348
```

Figure 06. Linear trend model for current time series

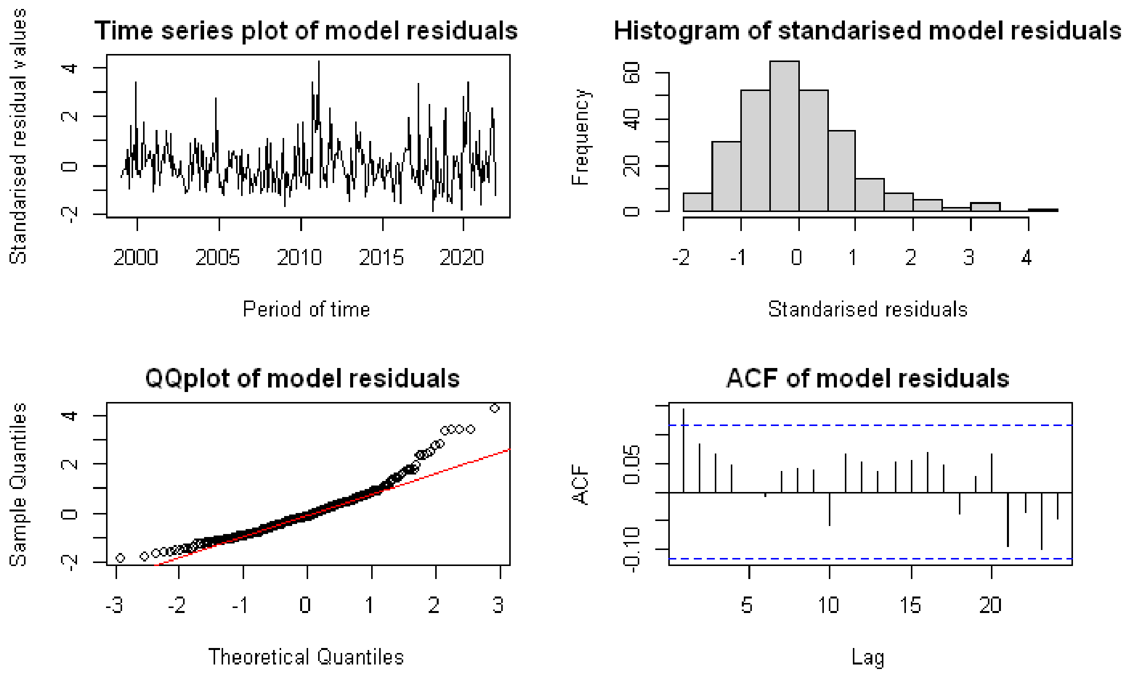


Figure 07: Summary of residuals for linear model

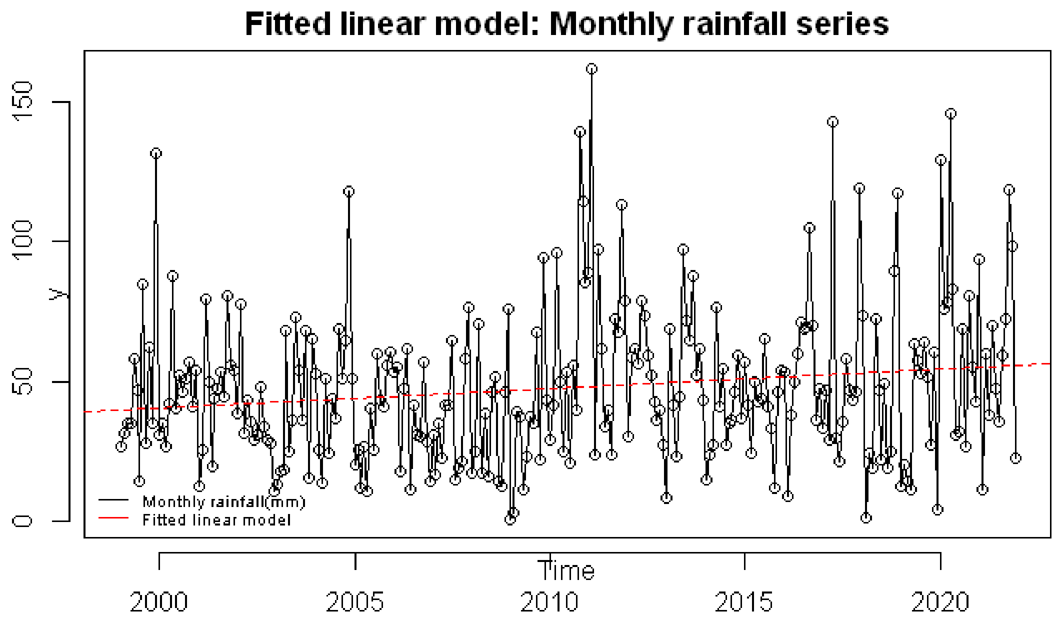


Figure 08:Fitted linear model to monthly rainfall series

1.3 Quadratic Trend model

```
t = time(railfall_TS)
t2 = t^2
railfall_TS_q2 = lm(railfall_TS~t+t2)
summary(railfall_TS_q2)

##
## Call:
## lm(formula = railfall_TS ~ t + t2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.466 -18.384  -4.198  13.247 115.379
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.720e+05  1.686e+05   1.020    0.309
## t           -1.717e+02  1.677e+02  -1.024    0.307
## t2           4.288e-02  4.170e-02   1.028    0.305
##
## Residual standard error: 27.32 on 273 degrees of freedom
## Multiple R-squared:  0.03062,    Adjusted R-squared:  0.02352
## F-statistic: 4.312 on 2 and 273 DF,  p-value: 0.01434
```

Figure 09: Quadratic model of order 2 for current time series

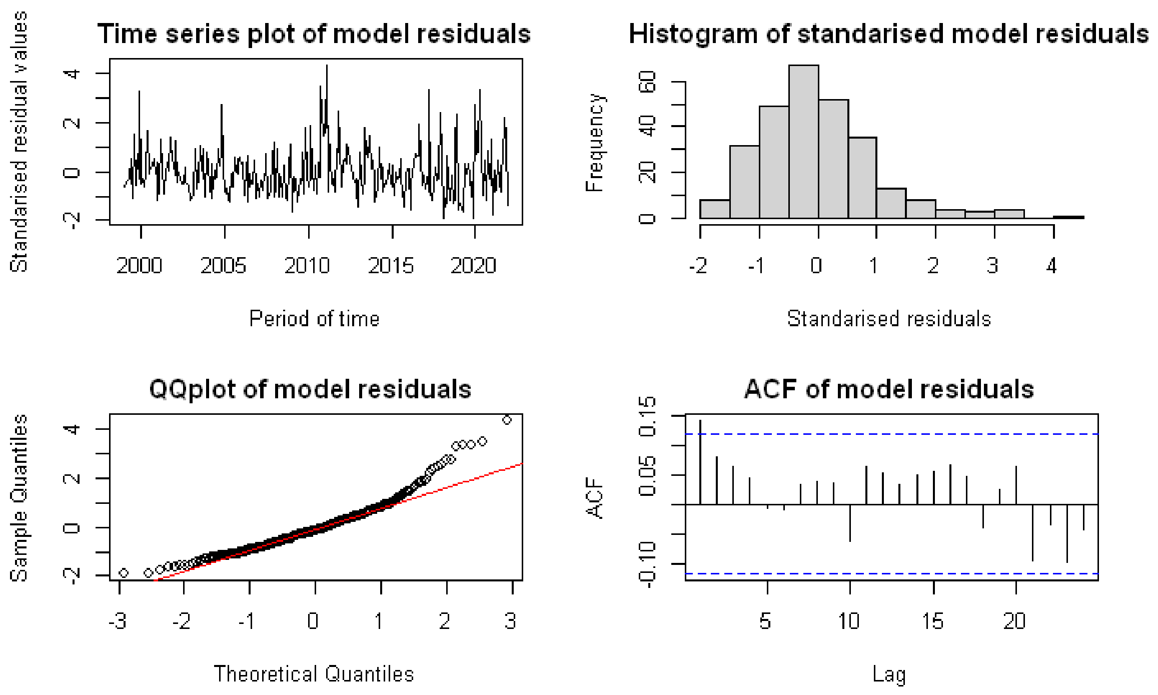


Figure 10: Summary of residuals for quadratic model of order 2

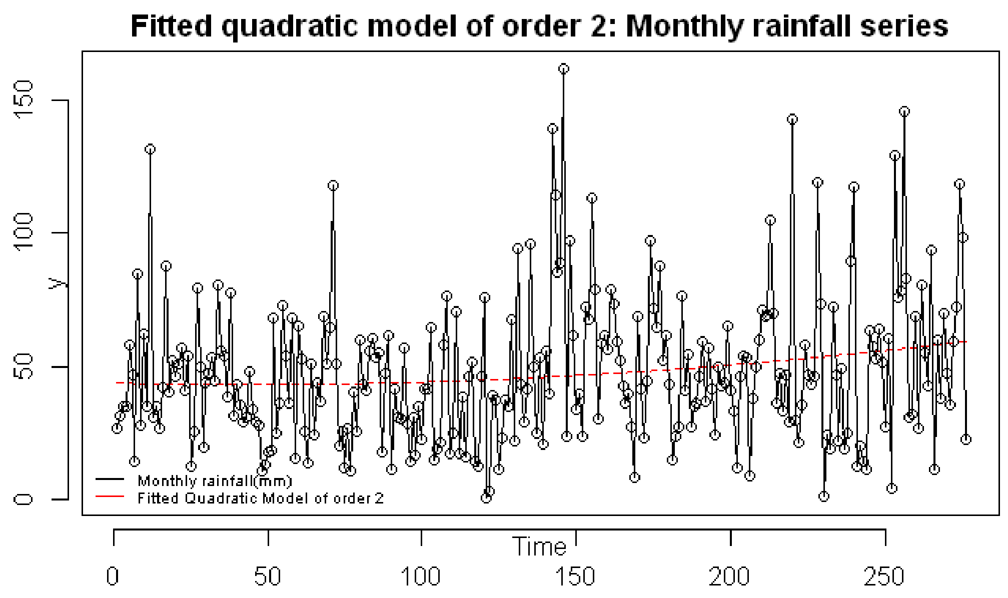


Figure 11: Fitted quadratic model of order 2: monthly rainfall series

```
t = time(railfall_TS)
t3 = t^3
railfall_TS_q3 = lm(railfall_TS~t+t3)
summary(railfall_TS_q3)

##
## Call:
## lm(formula = railfall_TS ~ t + t3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.464 -18.386  -4.198  13.244 115.378
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.141e+05  1.124e+05   1.016    0.311
## t           -8.546e+01  8.384e+01  -1.019    0.309
## t3           7.104e-06  6.914e-06   1.027    0.305
##
## Residual standard error: 27.32 on 273 degrees of freedom
## Multiple R-squared:  0.03062,    Adjusted R-squared:  0.02351
## F-statistic: 4.311 on 2 and 273 DF,  p-value: 0.01435
```
Figure 12: Quadratic model of order 3 for current time series

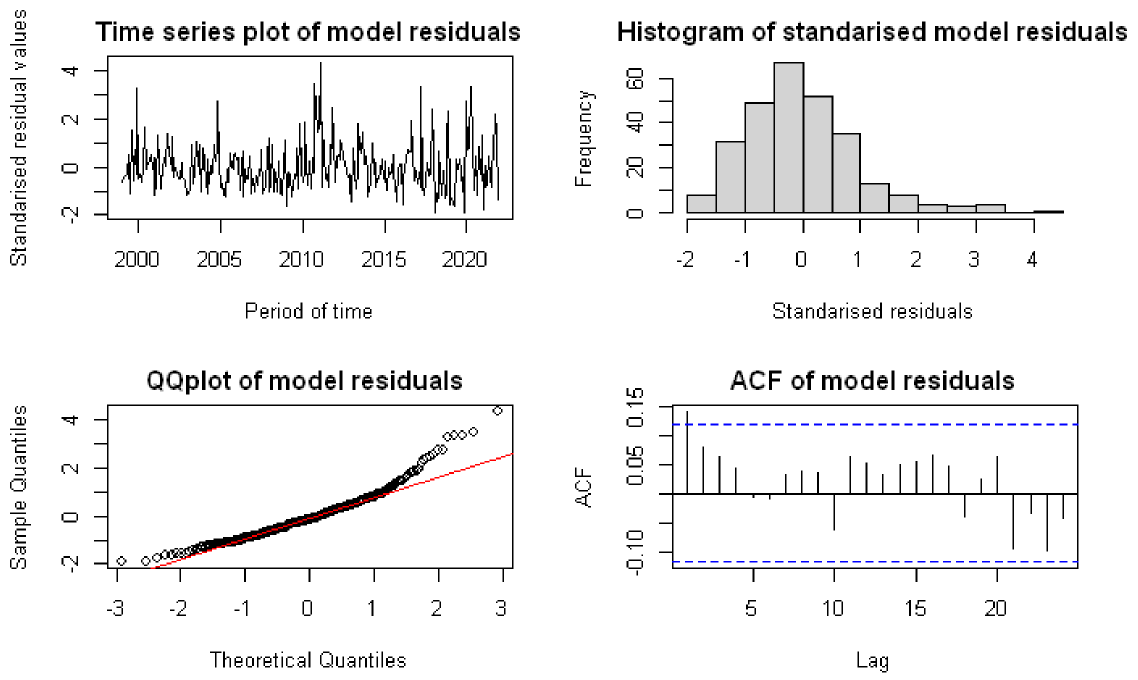


Figure 13:Summary of residuals for quadratic model of order 3

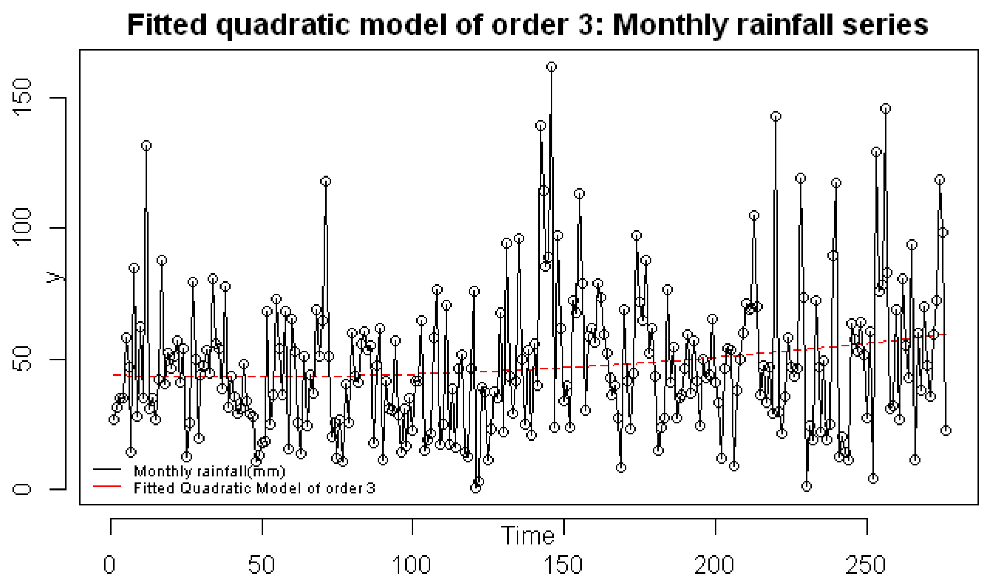
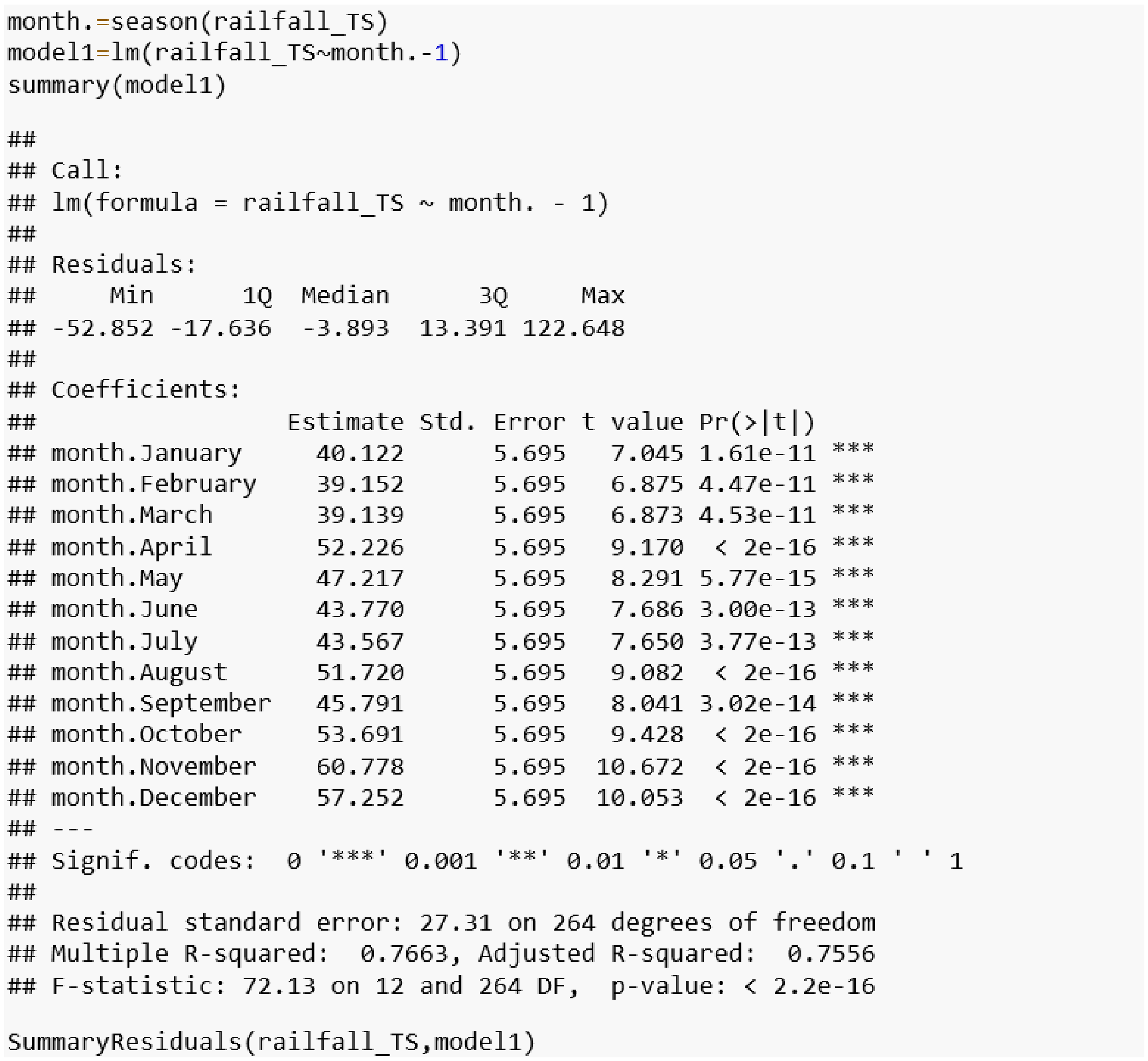


Figure 14: Fitted quadratic model of order 3: Monthly rainfall series

1.4 Seasonal Trend model



```
month.=season(railfall_TS)
model1=lm(railfall_TS~month.-1)
summary(model1)
```

```
##
## Call:
## lm(formula = railfall_TS ~ month. - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.852 -17.636  -3.893  13.391 122.648
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## month.January     40.122      5.695   7.045 1.61e-11 ***
## month.February    39.152      5.695   6.875 4.47e-11 ***
## month.March       39.139      5.695   6.873 4.53e-11 ***
## month.April       52.226      5.695   9.170  < 2e-16 ***
## month.May         47.217      5.695   8.291 5.77e-15 ***
## month.June        43.770      5.695   7.686 3.00e-13 ***
## month.July        43.567      5.695   7.650 3.77e-13 ***
## month.August      51.720      5.695   9.082  < 2e-16 ***
## month.September   45.791      5.695   8.041 3.02e-14 ***
## month.October     53.691      5.695   9.428  < 2e-16 ***
## month.November    60.778      5.695  10.672  < 2e-16 ***
## month.December    57.252      5.695  10.053  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.31 on 264 degrees of freedom
## Multiple R-squared:  0.7663, Adjusted R-squared:  0.7556
## F-statistic: 72.13 on 12 and 264 DF,  p-value: < 2.2e-16
```

```
SummaryResiduals(railfall_TS,model1)
```

Figure 15: Seasonal model for current time series

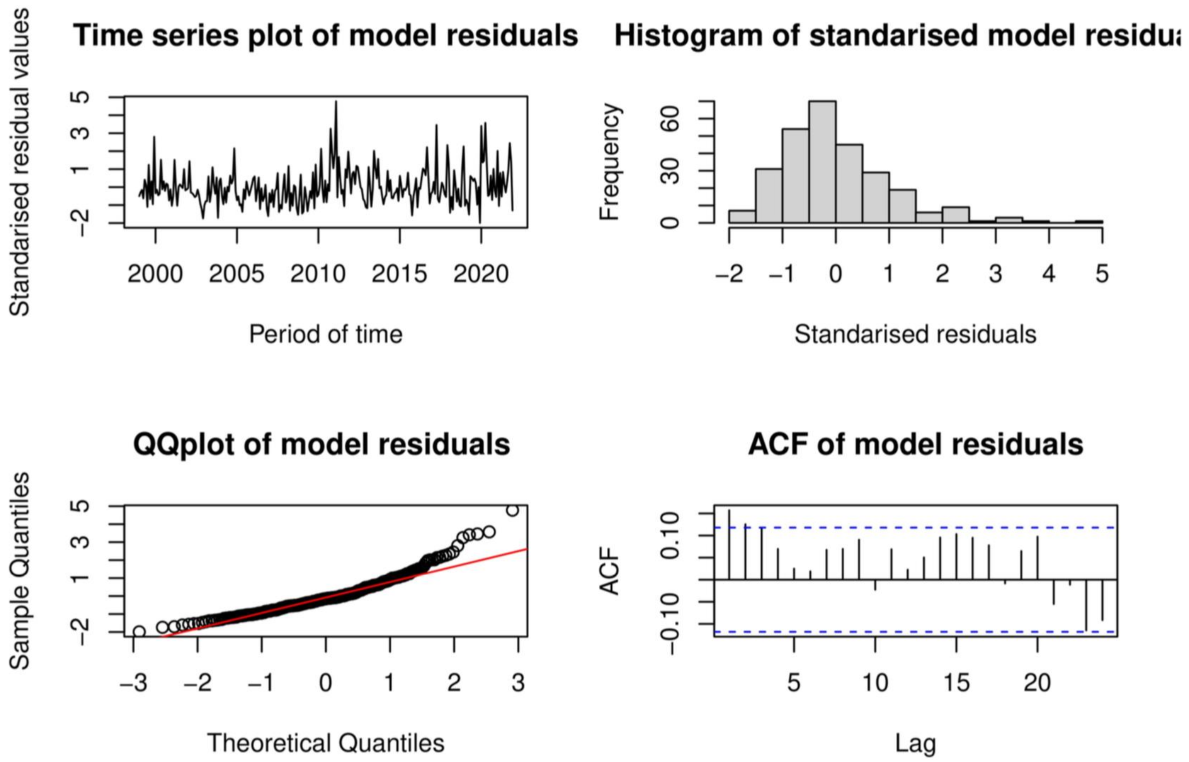


Figure 16: Residuals for Seasonal model

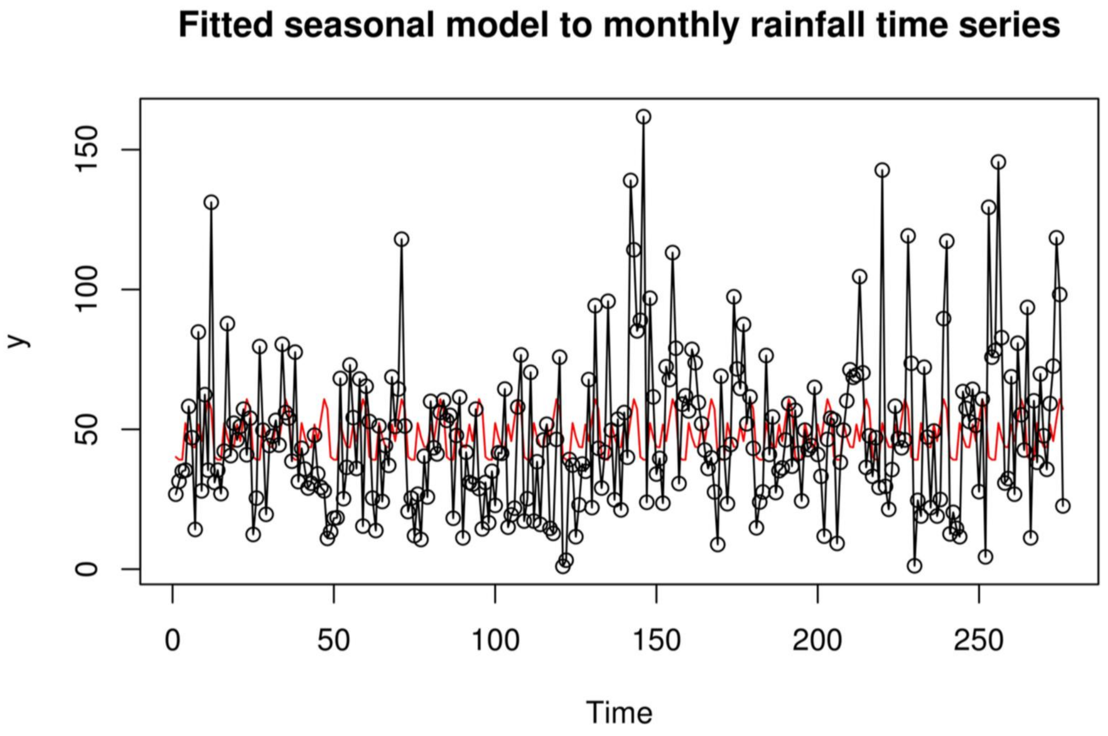
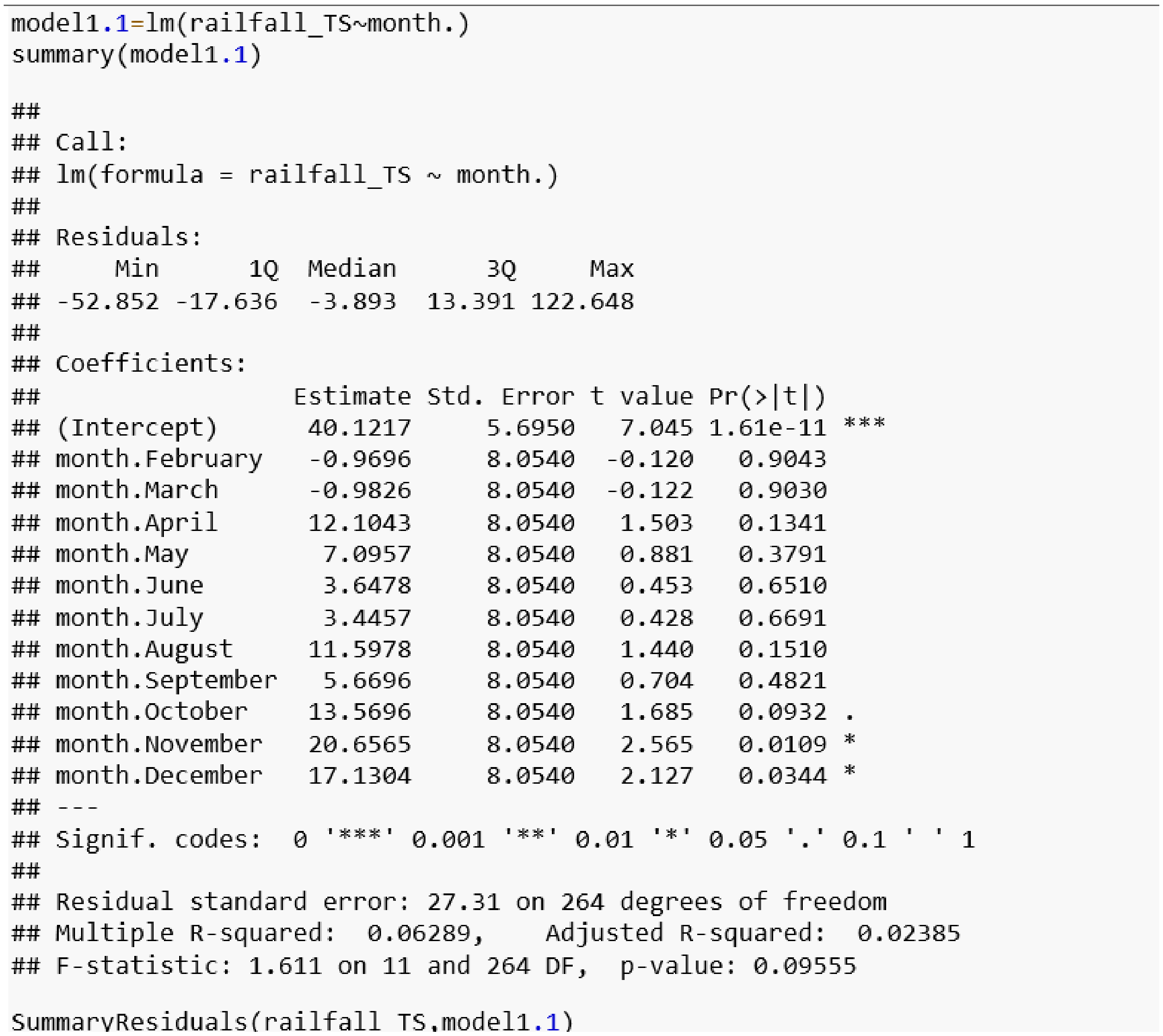


Figure 17: Fitted seasonal model to monthly rainfall time series



```
model1.1=lm(railfall_TS~month.)
summary(model1.1)

##
## Call:
## lm(formula = railfall_TS ~ month.)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.852 -17.636  -3.893  13.391 122.648
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      40.1217     5.6950   7.045 1.61e-11 ***
## month.February   -0.9696     8.0540  -0.120   0.9043
## month.March      -0.9826     8.0540  -0.122   0.9030
## month.April      12.1043     8.0540   1.503   0.1341
## month.May         7.0957     8.0540   0.881   0.3791
## month.June        3.6478     8.0540   0.453   0.6510
## month.July        3.4457     8.0540   0.428   0.6691
## month.August     11.5978     8.0540   1.440   0.1510
## month.September   5.6696     8.0540   0.704   0.4821
## month.October    13.5696     8.0540   1.685   0.0932 .
## month.November   20.6565     8.0540   2.565   0.0109 *
## month.December   17.1304     8.0540   2.127   0.0344 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.31 on 264 degrees of freedom
## Multiple R-squared:  0.06289,    Adjusted R-squared:  0.02385
## F-statistic: 1.611 on 11 and 264 DF,  p-value: 0.09555

SummaryResiduals(railfall_TS,model1.1)
```

Figure 18: Summary of Seasonal model with intercept

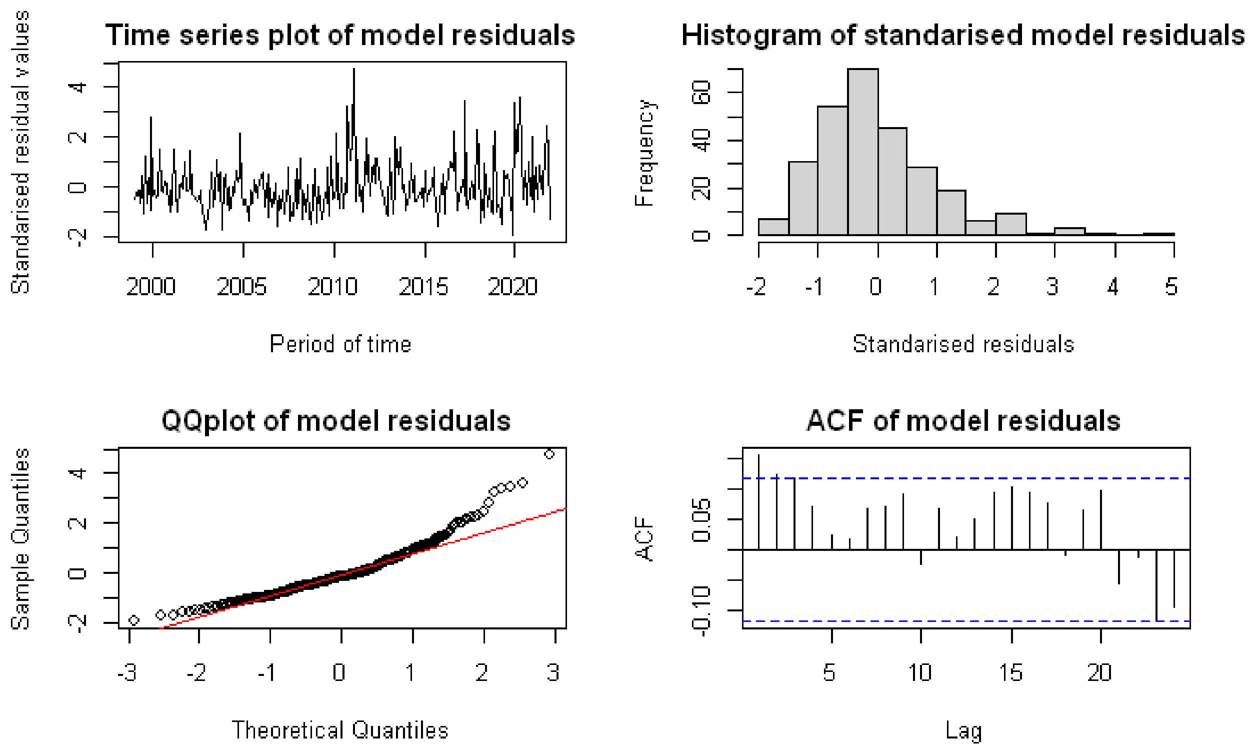


Figure 19: Residuals for Seasonal model with Intercept

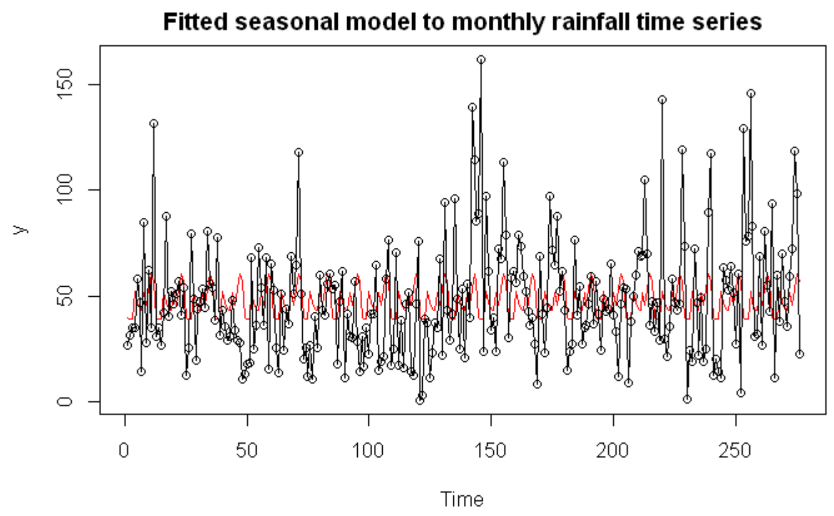


Figure 20: Fitted Seasonal model with intercept

1.5 Cosine model

```
har. <- harmonic(railfall_TS,1)
data<- data.frame(railfall_TS,har.)
model2 <- lm(railfall_TS~cos.2.pi.t.+sin.2.pi.t.,data=data)
summary(model2)

##
## Call:
## lm(formula = railfall_TS ~ cos.2.pi.t. + sin.2.pi.t., data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.002 -19.036  -4.018  13.926 116.482
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  47.8688     1.6544  28.935   <2e-16 ***
## cos.2.pi.t.   0.1335     2.3397   0.057   0.9545
## sin.2.pi.t.  -5.3325     2.3397  -2.279   0.0234 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.48 on 273 degrees of freedom
## Multiple R-squared:  0.01868,    Adjusted R-squared:  0.0115
## F-statistic: 2.599 on 2 and 273 DF,  p-value: 0.07619

SummaryResiduals(railfall_TS,model2)
```

Figure 21: Cosine model for current time series

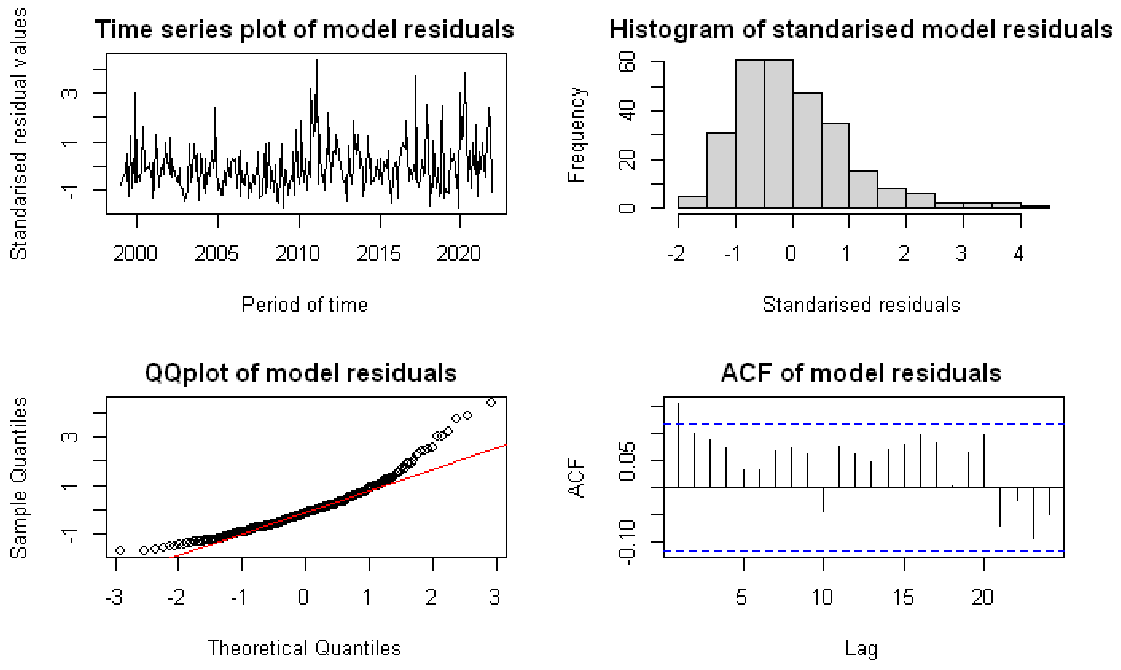


Figure 22: Residuals for the cosine model

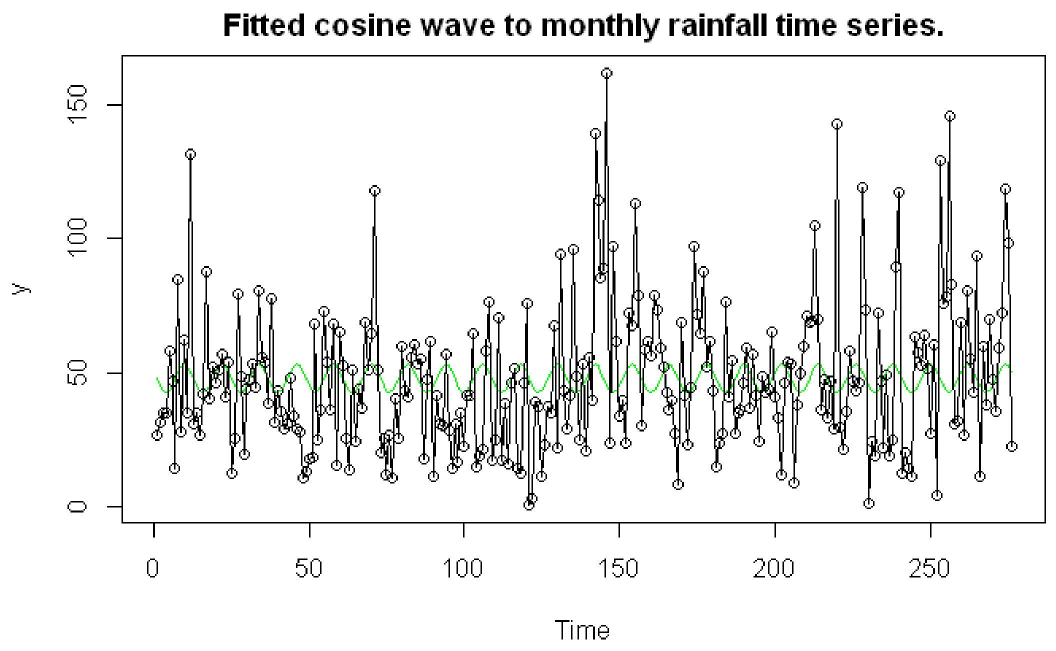


Figure 23: Fitted cosine wave to monthly rainfall time series

1.6 ARIMA model

<u>Transformation</u> :

Shapiro-Wilk normality test

data: box_cox W = 0.99411, p-value = 0.3594

Augmented Dickey-Fuller Test

data: X Dickey-Fuller = -5.5098, Lag order = 6, p-value = 0.01

alternative hypothesis: stationary

Phillips-Perron Unit Root Test

data: X Dickey-Fuller Z(alpha) = -251.46, Truncation lag parameter = 5,

P-value = 0.01 alternative hypothesis: stationary

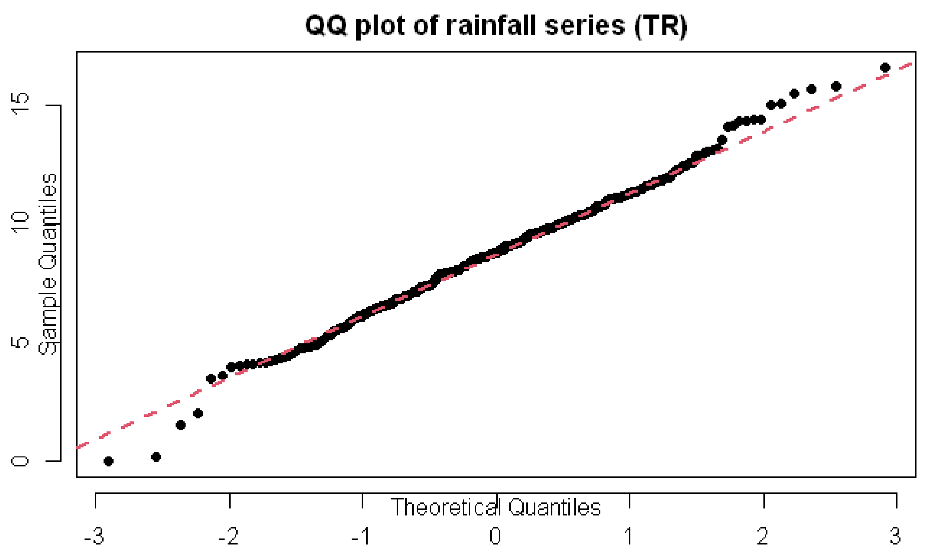
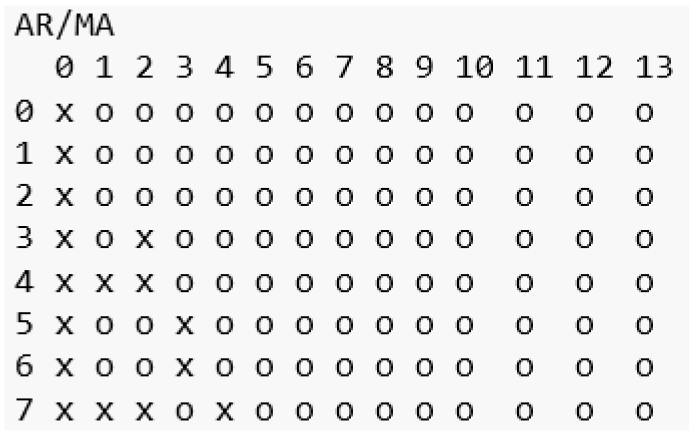


Figure 24: Quantile-Quantile plot of Transformed data



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

Figure 25: Extended Auto-Correlation Functio**n**

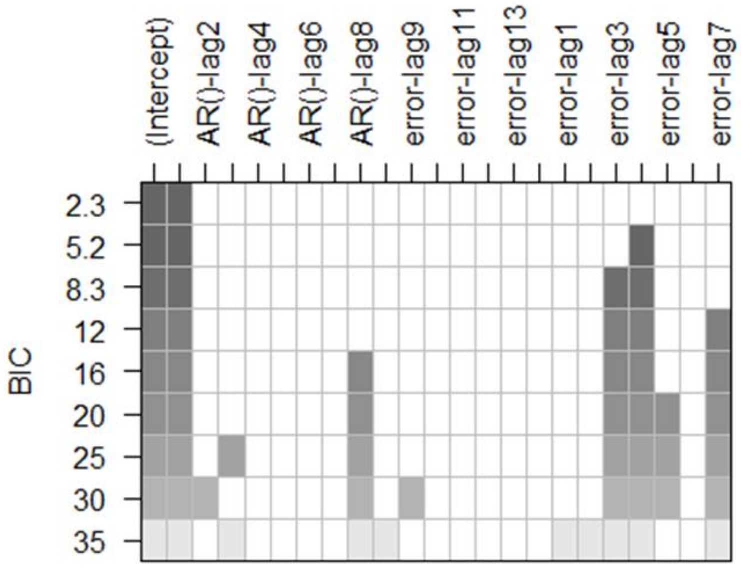
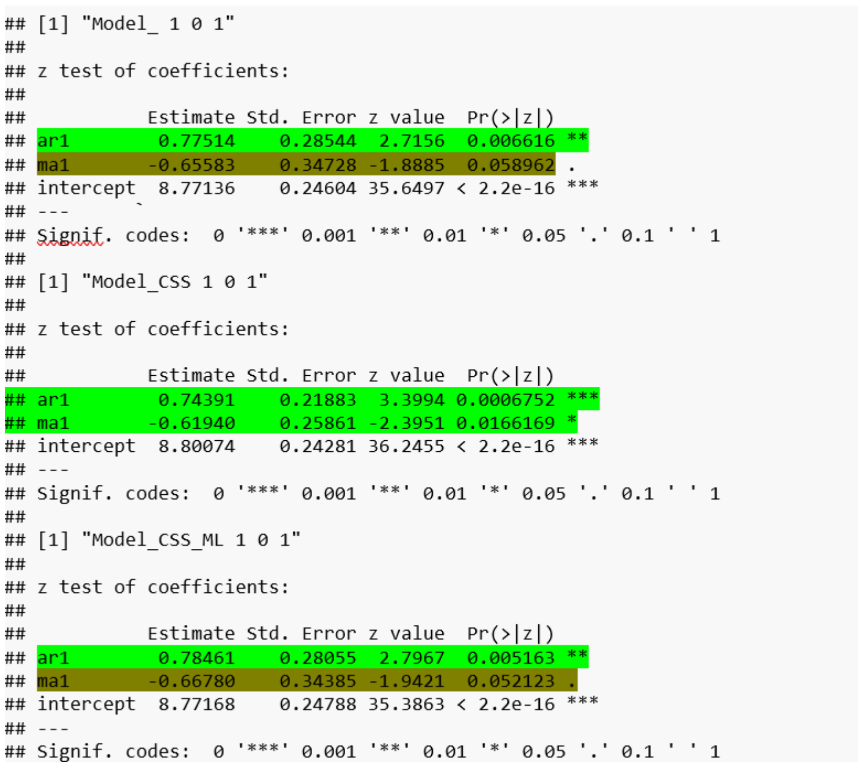


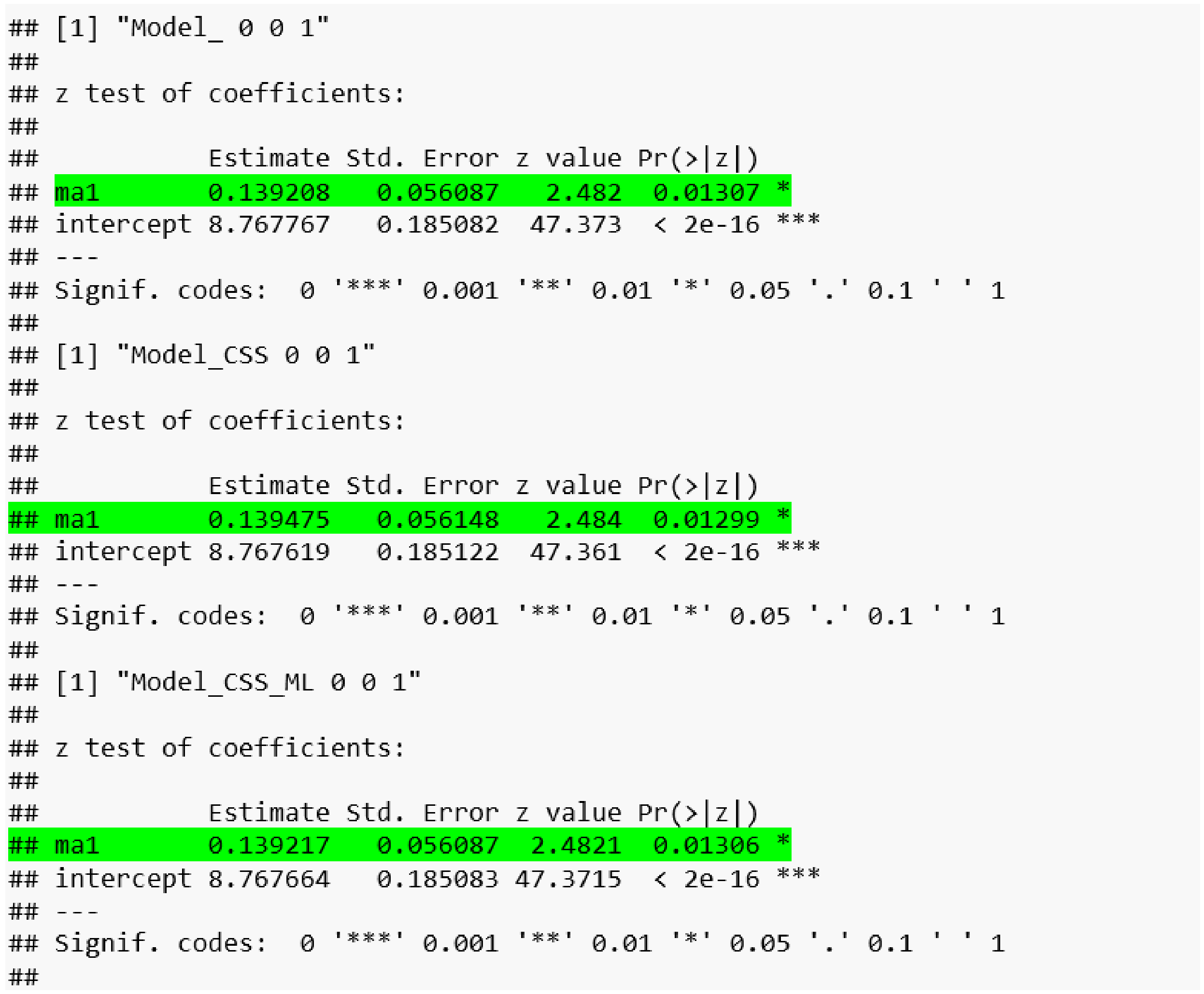
Figure 26: BIC table

Model Diagnostics checking



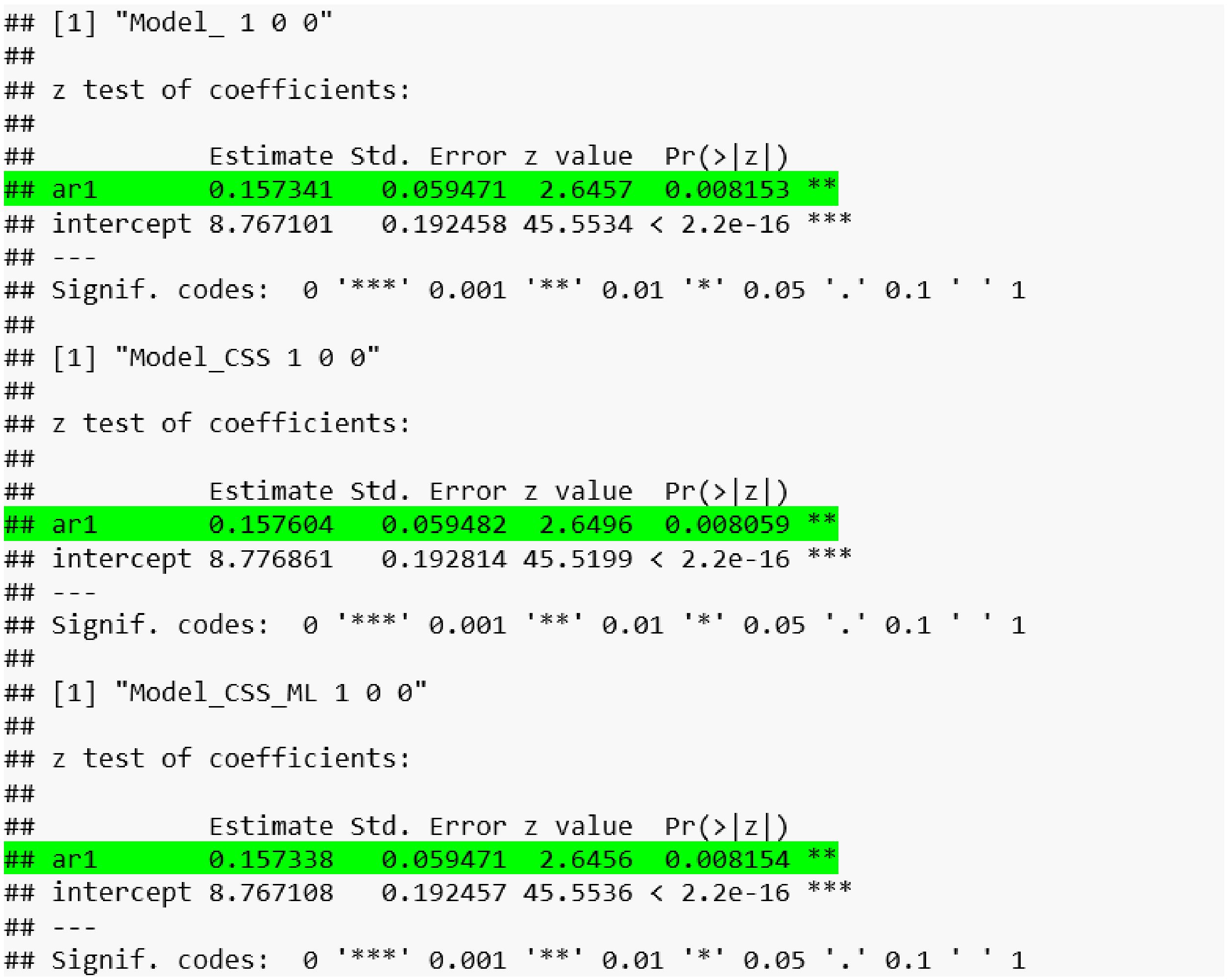
```
## [1] "Model_ 1 0 1"
##
## z test of coefficients:
##
##           Estimate Std. Error z value  Pr(>|z|)
## ar1        0.77514    0.28544  2.7156  0.006616 **
## ma1       -0.65583    0.34728 -1.8885  0.058962 .
## intercept  8.77136    0.24604 35.6497 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 1 0 1"
##
## z test of coefficients:
##
##           Estimate Std. Error z value  Pr(>|z|)
## ar1        0.74391    0.21883  3.3994 0.0006752 ***
## ma1       -0.61940    0.25861 -2.3951 0.0166169 *
## intercept  8.80074    0.24281 36.2455 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 1 0 1"
##
## z test of coefficients:
##
##           Estimate Std. Error z value  Pr(>|z|)
## ar1        0.78461    0.28055  2.7967  0.005163 **
## ma1       -0.66780    0.34385 -1.9421  0.052123 .
## intercept  8.77168    0.24788 35.3863 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 27: Model ARIMA {101}



```
## [1] "Model_ 0 0 1"
##
## z test of coefficients:
##
##           Estimate Std. Error z value Pr(>|z|)
## ma1       0.139208   0.056087   2.482  0.01307 *
## intercept 8.767767   0.185082  47.373  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 0 0 1"
##
## z test of coefficients:
##
##           Estimate Std. Error z value Pr(>|z|)
## ma1       0.139475   0.056148   2.484  0.01299 *
## intercept 8.767619   0.185122  47.361  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 0 0 1"
##
## z test of coefficients:
##
##           Estimate Std. Error z value Pr(>|z|)
## ma1       0.139217   0.056087  2.4821  0.01306 *
## intercept 8.767664   0.185083 47.3715  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

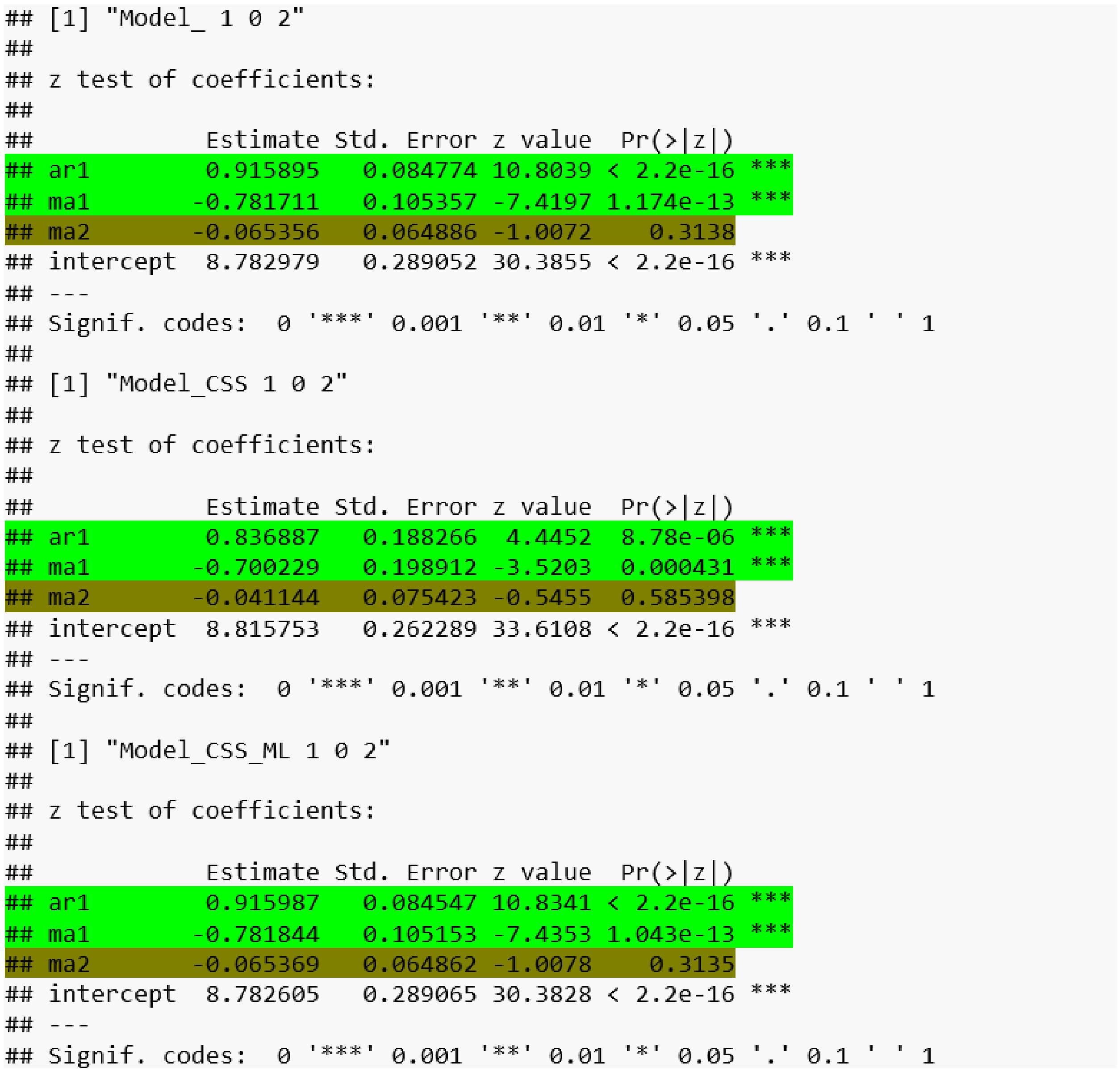
Figure 28:Model ARIMA {001}



```
## [1] "Model_ 1 0 0"
##
## z test of coefficients:
##
##           Estimate Std. Error z value  Pr(>|z|)
## ar1       0.157341   0.059471  2.6457  0.008153 **
## intercept 8.767101   0.192458 45.5534 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 1 0 0"
##
## z test of coefficients:
##
##           Estimate Std. Error z value  Pr(>|z|)
## ar1       0.157604   0.059482  2.6496  0.008059 **
## intercept 8.776861   0.192814 45.5199 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 1 0 0"
##
## z test of coefficients:
##
##           Estimate Std. Error z value  Pr(>|z|)
## ar1       0.157338   0.059471  2.6456  0.008154 **
## intercept 8.767108   0.192457 45.5536 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 29: Model ARIMA {100}

```
## [1] "Model_ 0 0 2"
##
## z test of coefficients:
##
##           Estimate Std. Error z value Pr(>|z|)
## ma1       0.142054   0.060333  2.3545  0.01855 *
## ma2       0.066235   0.056863  1.1648  0.24409
## intercept 8.766841   0.195751 44.7857  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 0 0 2"
##
## z test of coefficients:
##
##           Estimate Std. Error z value Pr(>|z|)
## ma1       0.142293   0.060413  2.3553  0.01851 *
## ma2       0.066429   0.057021  1.1650  0.24402
## intercept 8.766524   0.195812 44.7700  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 0 0 2"
##
## z test of coefficients:
##
##           Estimate Std. Error z value Pr(>|z|)
## ma1       0.142040   0.060334  2.3542  0.01856 *
## ma2       0.066249   0.056863  1.1651  0.24399
## intercept 8.766734   0.195751 44.7851  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```
Figure 30: Model ARIMA {002}

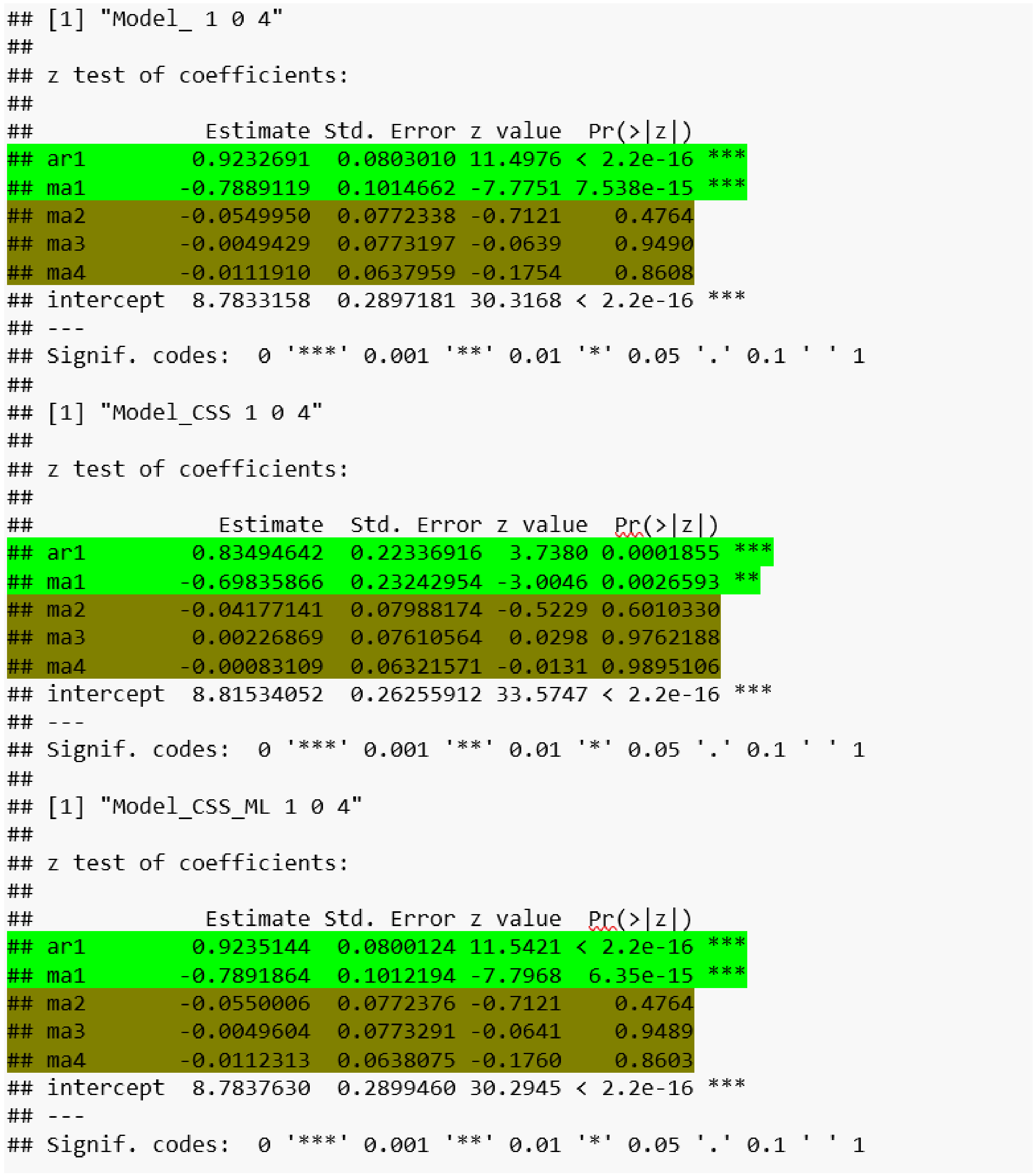


```
## [1] "Model_ 1 0 2"
##
## z test of coefficients:
##
##            Estimate Std. Error z value  Pr(>|z|)
## ar1        0.915895   0.084774 10.8039 < 2.2e-16 ***
## ma1       -0.781711   0.105357 -7.4197 1.174e-13 ***
## ma2       -0.065356   0.064886 -1.0072    0.3138
## intercept  8.782979   0.289052 30.3855 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 1 0 2"
##
## z test of coefficients:
##
##            Estimate Std. Error z value  Pr(>|z|)
## ar1        0.836887   0.188266  4.4452  8.78e-06 ***
## ma1       -0.700229   0.198912 -3.5203  0.000431 ***
## ma2       -0.041144   0.075423 -0.5455  0.585398
## intercept  8.815753   0.262289 33.6108 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 1 0 2"
##
## z test of coefficients:
##
##            Estimate Std. Error z value  Pr(>|z|)
## ar1        0.915987   0.084547 10.8341 < 2.2e-16 ***
## ma1       -0.781844   0.105153 -7.4353 1.043e-13 ***
## ma2       -0.065369   0.064862 -1.0078    0.3135
## intercept  8.782605   0.289065 30.3828 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 31: Model ARIMA {102}

```
## [1] "Model_ 1 0 3"
##
## z test of coefficients:
##
##              Estimate Std. Error z value  Pr(>|z|)
## ar1          0.922158   0.079535 11.5944 < 2.2e-16 ***
## ma1         -0.787260   0.100477 -7.8352 4.681e-15 ***
## ma2         -0.056240   0.077823 -0.7227    0.4699
## ma3         -0.013695   0.060473 -0.2265    0.8208
## intercept  8.784536   0.291372 30.1489 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 1 0 3"
##
## z test of coefficients:
##
##               Estimate Std. Error z value  Pr(>|z|)
## ar1          0.8323279  0.2245383  3.7068 0.0002099 ***
## ma1         -0.6955596  0.2328779 -2.9868 0.0028191 **
## ma2         -0.0415305  0.0804204 -0.5164 0.6055626
## ma3          0.0020274  0.0650363  0.0312 0.9751308
## intercept  8.8143727  0.2619308 33.6515 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 1 0 3"
##
## z test of coefficients:
##
##              Estimate Std. Error z value  Pr(>|z|)
## ar1          0.921088   0.080534 11.4373 < 2.2e-16 ***
## ma1         -0.786314   0.101336 -7.7595 8.529e-15 ***
## ma2         -0.055970   0.077809 -0.7193    0.4719
## ma3         -0.013527   0.060450 -0.2238    0.8229
## intercept  8.783876   0.290260 30.2621 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

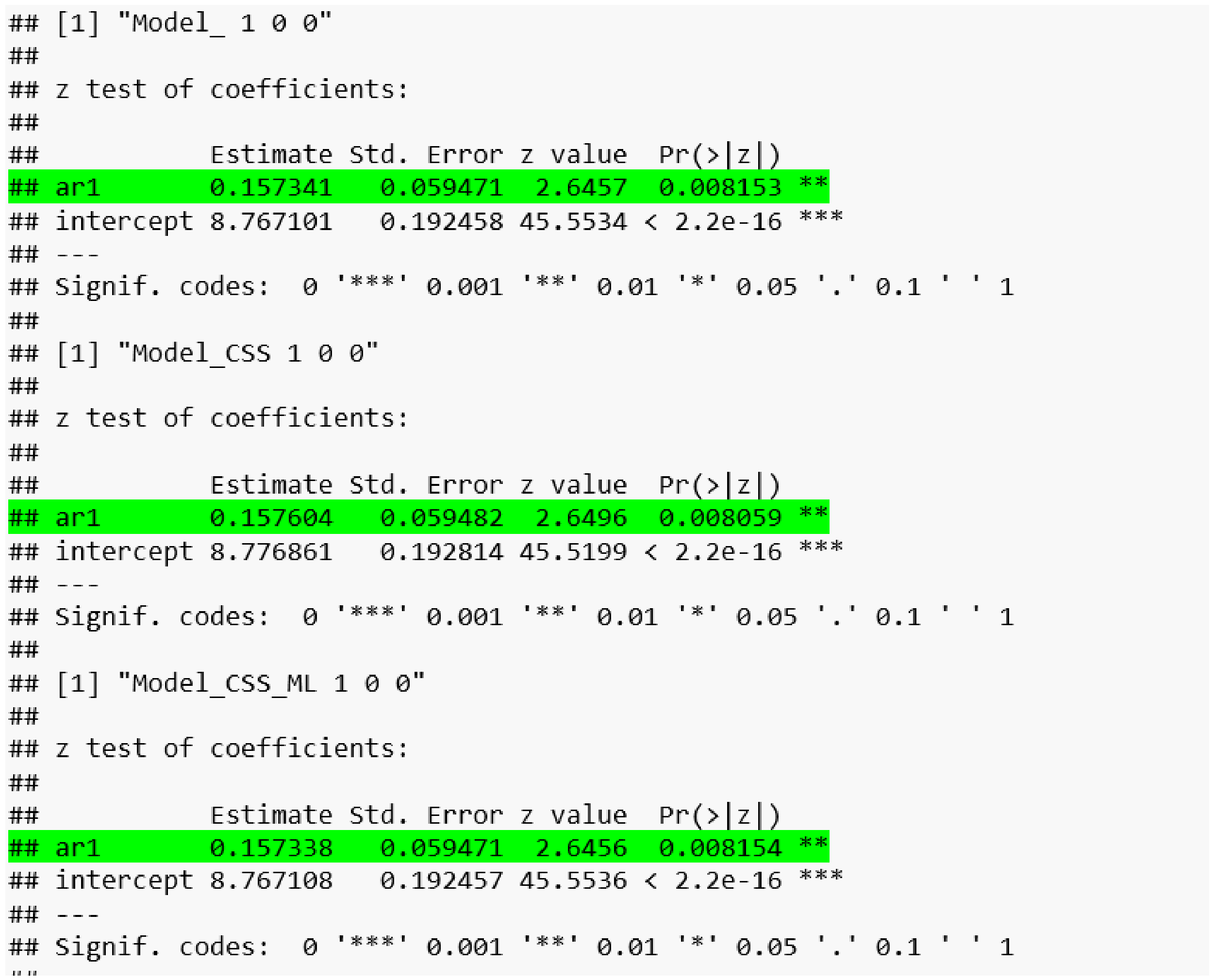
Figure 32: Model ARIMA {103}



```
## [1] "Model_ 1 0 4"
##
## z test of coefficients:
##
##             Estimate Std. Error z value  Pr(>|z|)
## ar1        0.9232691  0.0803010 11.4976 < 2.2e-16 ***
## ma1       -0.7889119  0.1014662 -7.7751 7.538e-15 ***
## ma2       -0.0549950  0.0772338 -0.7121    0.4764
## ma3       -0.0049429  0.0773197 -0.0639    0.9490
## ma4       -0.0111910  0.0637959 -0.1754    0.8608
## intercept  8.7833158  0.2897181 30.3168 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 1 0 4"
##
## z test of coefficients:
##
##              Estimate  Std. Error z value  Pr(>|z|)
## ar1        0.83494642  0.22336916  3.7380 0.0001855 ***
## ma1       -0.69835866  0.23242954 -3.0046 0.0026593 **
## ma2       -0.04177141  0.07988174 -0.5229 0.6010330
## ma3        0.00226869  0.07610564  0.0298 0.9762188
## ma4       -0.00083109  0.06321571 -0.0131 0.9895106
## intercept  8.81534052  0.26255912 33.5747 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 1 0 4"
##
## z test of coefficients:
##
##             Estimate Std. Error z value  Pr(>|z|)
## ar1        0.9235144  0.0800124 11.5421 < 2.2e-16 ***
## ma1       -0.7891864  0.1012194 -7.7968  6.35e-15 ***
## ma2       -0.0550006  0.0772376 -0.7121    0.4764
## ma3       -0.0049604  0.0773291 -0.0641    0.9489
## ma4       -0.0112313  0.0638075 -0.1760    0.8603
## intercept  8.7837630  0.2899460 30.2945 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

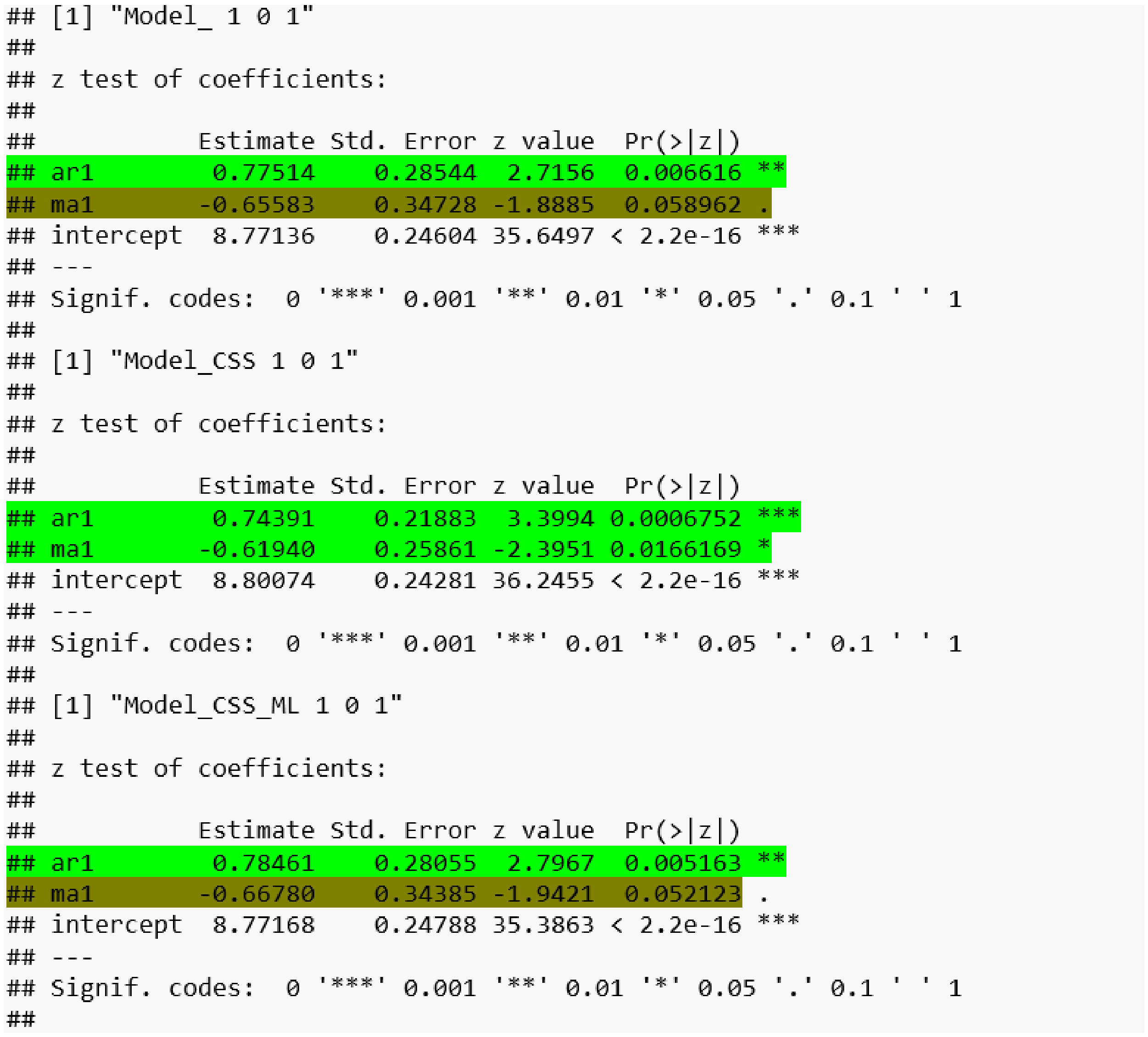
Figure 33: Model ARIMA {104}

Overfitting models



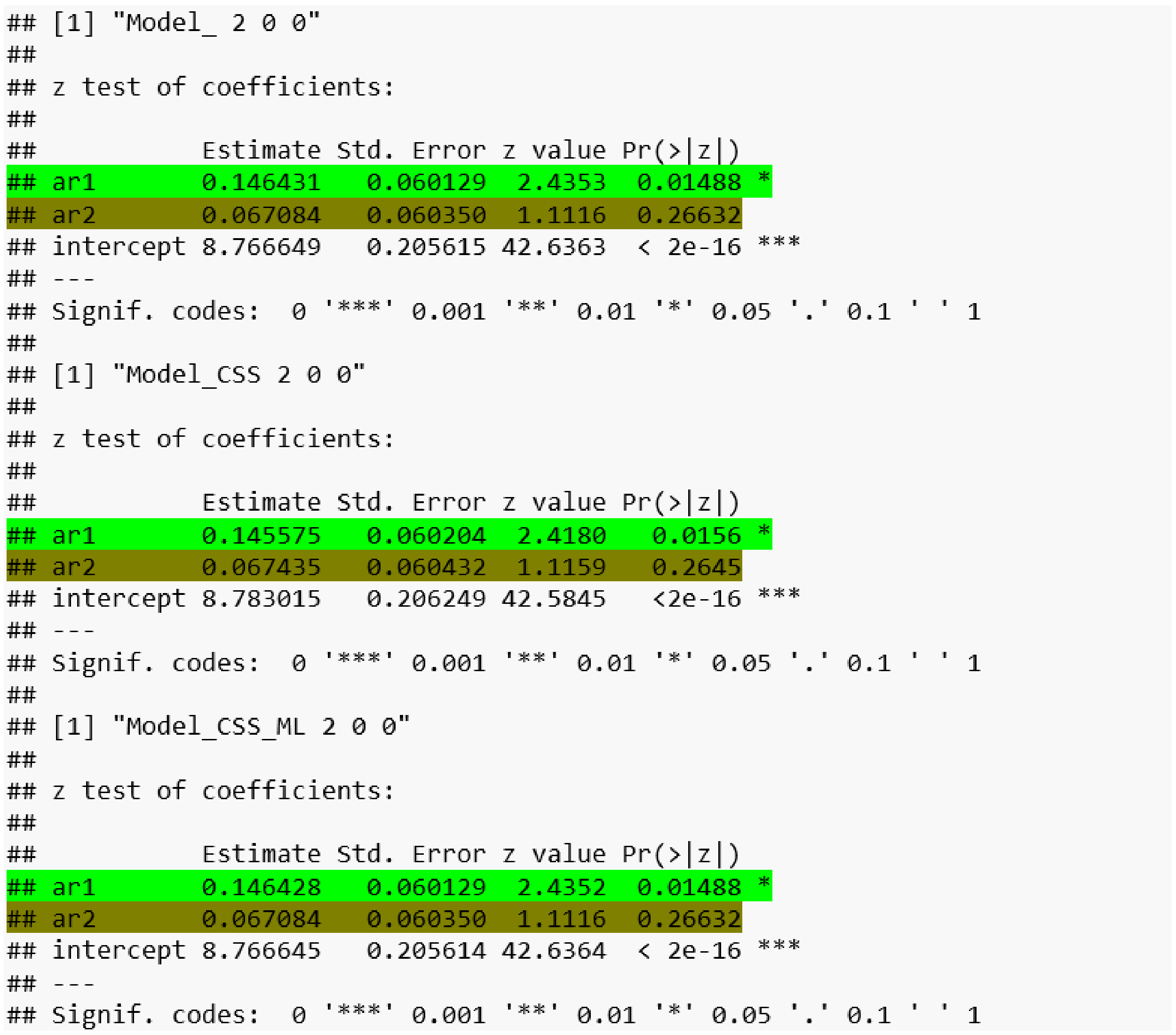
```
## [1] "Model_ 1 0 0"
##
## z test of coefficients:
##
##           Estimate Std. Error z value  Pr(>|z|)
## ar1       0.157341   0.059471  2.6457  0.008153 **
## intercept 8.767101   0.192458 45.5534 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 1 0 0"
##
## z test of coefficients:
##
##           Estimate Std. Error z value  Pr(>|z|)
## ar1       0.157604   0.059482  2.6496  0.008059 **
## intercept 8.776861   0.192814 45.5199 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 1 0 0"
##
## z test of coefficients:
##
##           Estimate Std. Error z value  Pr(>|z|)
## ar1       0.157338   0.059471  2.6456  0.008154 **
## intercept 8.767108   0.192457 45.5536 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 34: Model ARIMA {100}



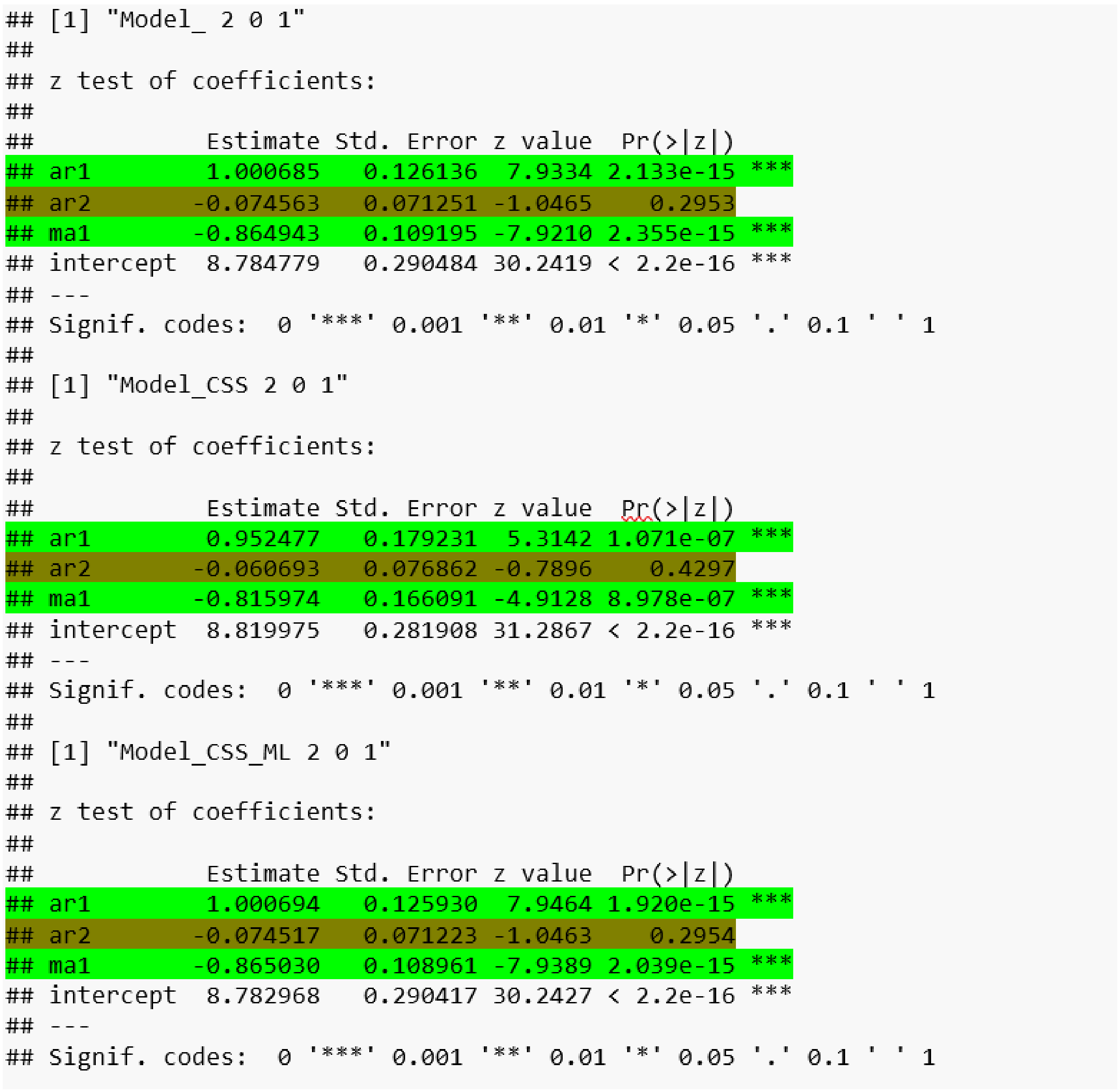
```
## [1] "Model_ 1 0 1"
##
## z test of coefficients:
##
##            Estimate Std. Error z value  Pr(>|z|)
## ar1         0.77514    0.28544  2.7156  0.006616 **
## ma1        -0.65583    0.34728 -1.8885  0.058962 .
## intercept   8.77136    0.24604 35.6497 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 1 0 1"
##
## z test of coefficients:
##
##            Estimate Std. Error z value  Pr(>|z|)
## ar1         0.74391    0.21883  3.3994 0.0006752 ***
## ma1        -0.61940    0.25861 -2.3951 0.0166169 *
## intercept   8.80074    0.24281 36.2455 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 1 0 1"
##
## z test of coefficients:
##
##            Estimate Std. Error z value  Pr(>|z|)
## ar1         0.78461    0.28055  2.7967  0.005163 **
## ma1        -0.66780    0.34385 -1.9421  0.052123 .
## intercept   8.77168    0.24788 35.3863 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

Figure 35: Model ARIMA {101}



```
## [1] "Model_ 2 0 0"
##
## z test of coefficients:
##
##           Estimate Std. Error z value Pr(>|z|)
## ar1       0.146431   0.060129  2.4353  0.01488 *
## ar2       0.067084   0.060350  1.1116  0.26632
## intercept 8.766649   0.205615 42.6363  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 2 0 0"
##
## z test of coefficients:
##
##           Estimate Std. Error z value Pr(>|z|)
## ar1       0.145575   0.060204  2.4180   0.0156 *
## ar2       0.067435   0.060432  1.1159   0.2645
## intercept 8.783015   0.206249 42.5845   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 2 0 0"
##
## z test of coefficients:
##
##           Estimate Std. Error z value Pr(>|z|)
## ar1       0.146428   0.060129  2.4352  0.01488 *
## ar2       0.067084   0.060350  1.1116  0.26632
## intercept 8.766645   0.205614 42.6364  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

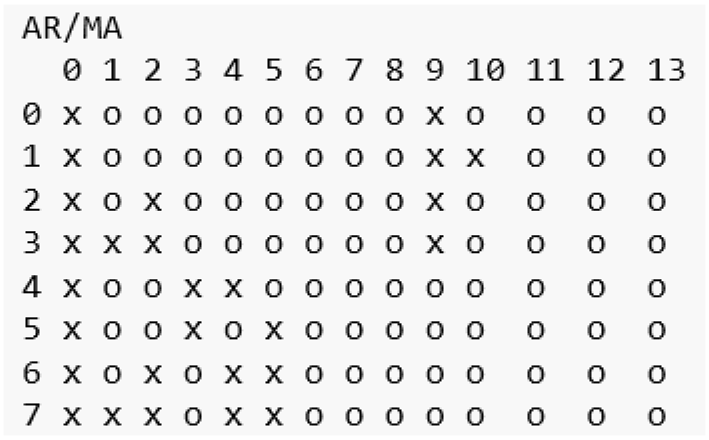
Figure 36: Model ARIMA {200}



```
## [1] "Model_ 2 0 1"
##
## z test of coefficients:
##
##             Estimate Std. Error z value  Pr(>|z|)
## ar1         1.000685   0.126136  7.9334 2.133e-15 ***
## ar2        -0.074563   0.071251 -1.0465    0.2953
## ma1        -0.864943   0.109195 -7.9210 2.355e-15 ***
## intercept   8.784779   0.290484 30.2419 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS 2 0 1"
##
## z test of coefficients:
##
##             Estimate Std. Error z value  Pr(>|z|)
## ar1         0.952477   0.179231  5.3142 1.071e-07 ***
## ar2        -0.060693   0.076862 -0.7896    0.4297
## ma1        -0.815974   0.166091 -4.9128 8.978e-07 ***
## intercept   8.819975   0.281908 31.2867 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## [1] "Model_CSS_ML 2 0 1"
##
## z test of coefficients:
##
##             Estimate Std. Error z value  Pr(>|z|)
## ar1         1.000694   0.125930  7.9464 1.920e-15 ***
## ar2        -0.074517   0.071223 -1.0463    0.2954
## ma1        -0.865030   0.108961 -7.9389 2.039e-15 ***
## intercept   8.782968   0.290417 30.2427 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 37: Model ARIMA {201}

1.6 ARCH + GARCH model



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

Figure 38. EACF matrix of the returned time series data of Melbourne Rainfall

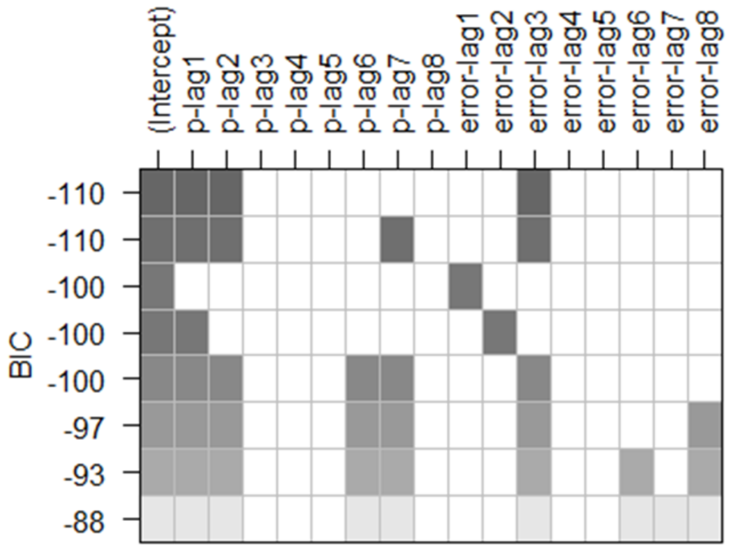
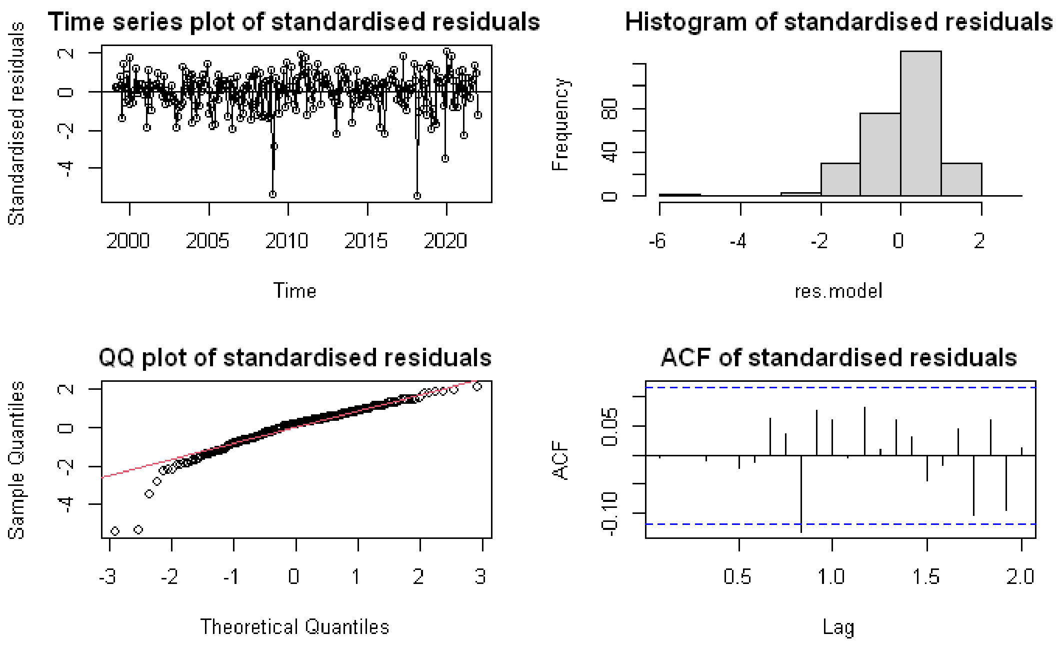
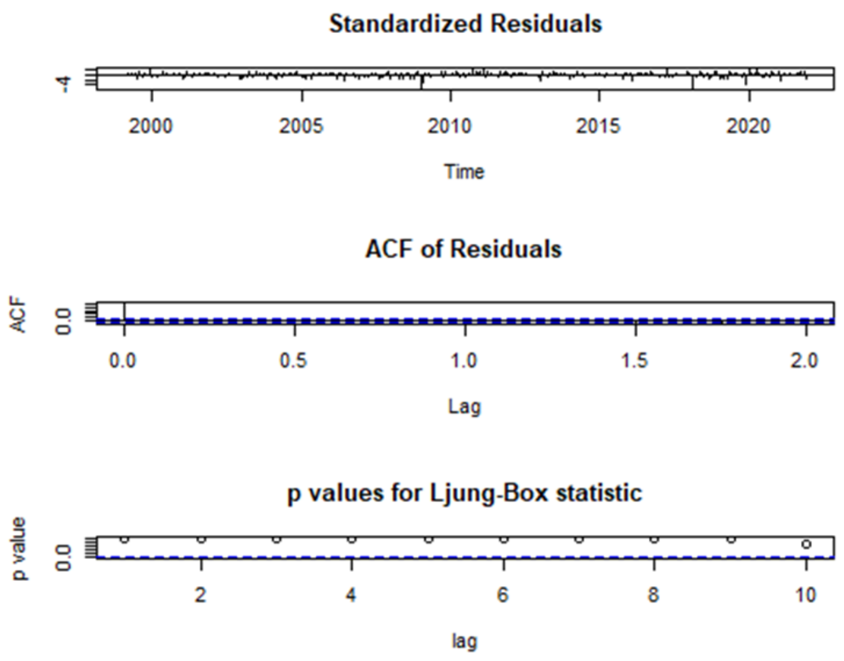


Figure 39. BIC of the returned time series data of Melbourne Rainfall

```
## [1] "Model_ 4 0 1"
##
## z test of coefficients:
##
##            Estimate Std. Error  z value Pr(>|z|)
## ar1        0.134480   0.060337   2.2288  0.02583 *
## ar2        0.027223   0.060779   0.4479  0.65422
## ar3        0.046046   0.060869   0.7565  0.44936
## ar4        0.074916   0.060382   1.2407  0.21472
## ma1       -0.999999   0.011835 -84.4930  < 2e-16 ***
## intercept  0.084221   0.070541   1.1939  0.23251
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

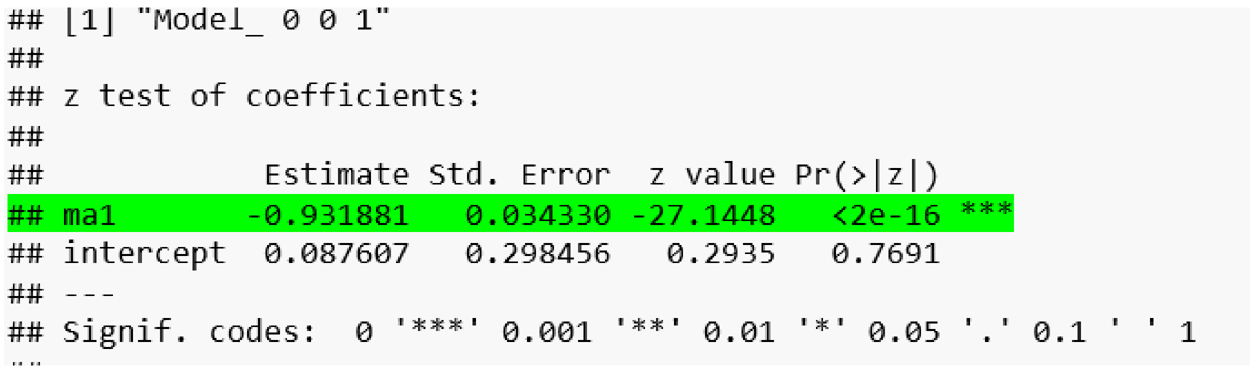




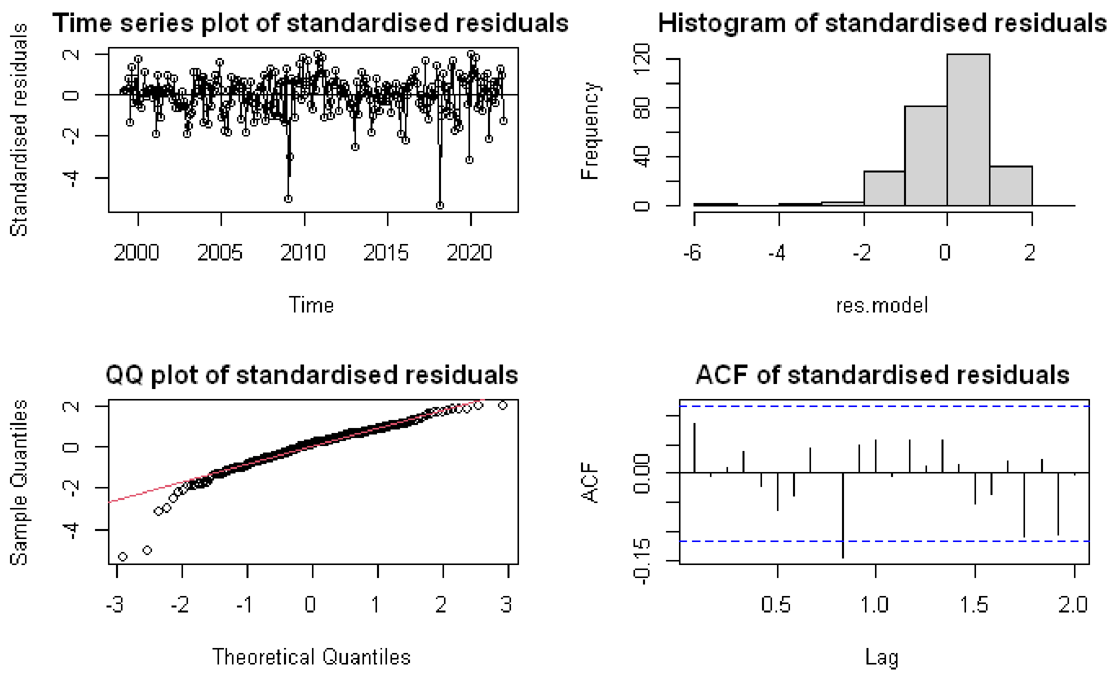
Shapiro-Wilk normality test

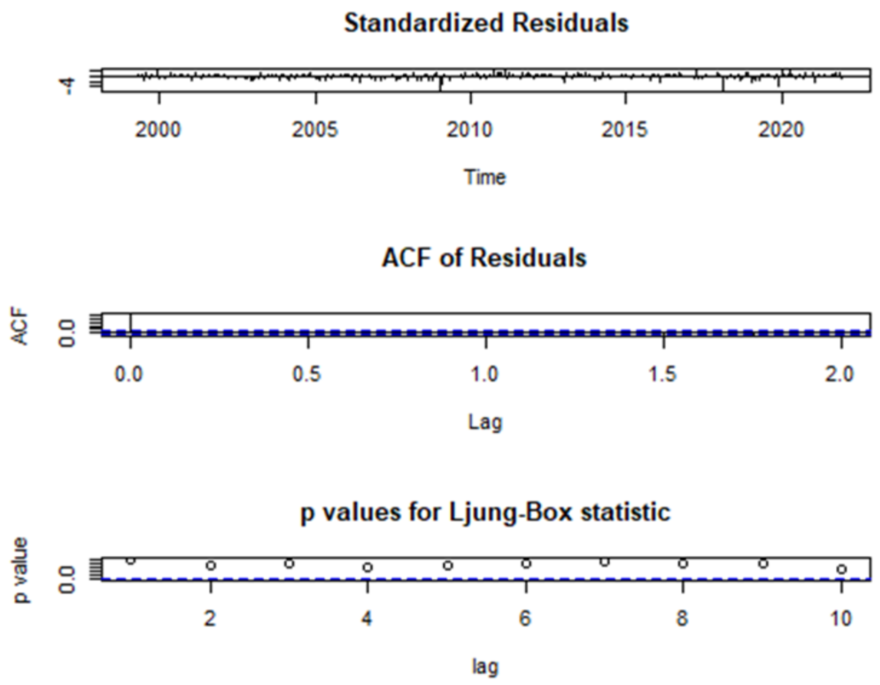
data: res.mode**l W** = 0.91699, **p-value =** 3.178e-11

Figure 40. Model ARMA {4,1}



```
## [1] "Model_ 0 0 1"
##
## z test of coefficients:
##
##             Estimate Std. Error  z value Pr(>|z|)
## ma1       -0.931881   0.034330 -27.1448   <2e-16 ***
## intercept  0.087607   0.298456   0.2935   0.7691
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



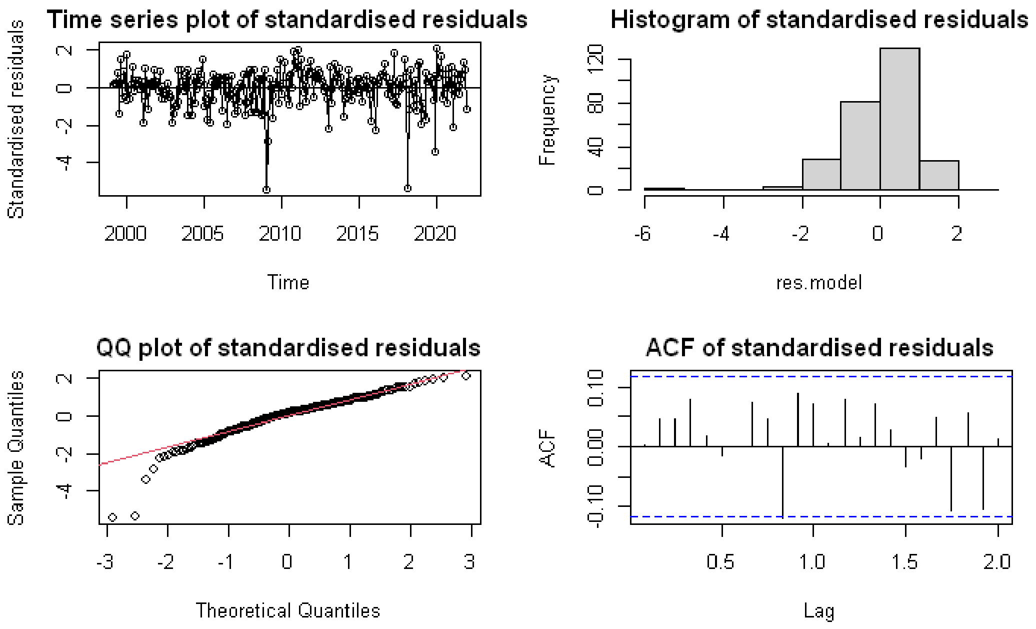


Shapiro-Wilk normality test

data: res.model W = 0.92722, p-value = 2.37e-10

Figure 41. Model ARMA {0,1}

```
## [1] "Model_ 0 0 2"
##
## z test of coefficients:
##
##              Estimate Std. Error  z value Pr(>|z|)
## ma1         -0.864555   0.059539 -14.5209  < 2e-16 ***
## ma2         -0.135445   0.057688  -2.3479  0.01888 *
## intercept    0.081683   0.058234   1.4027  0.16071
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

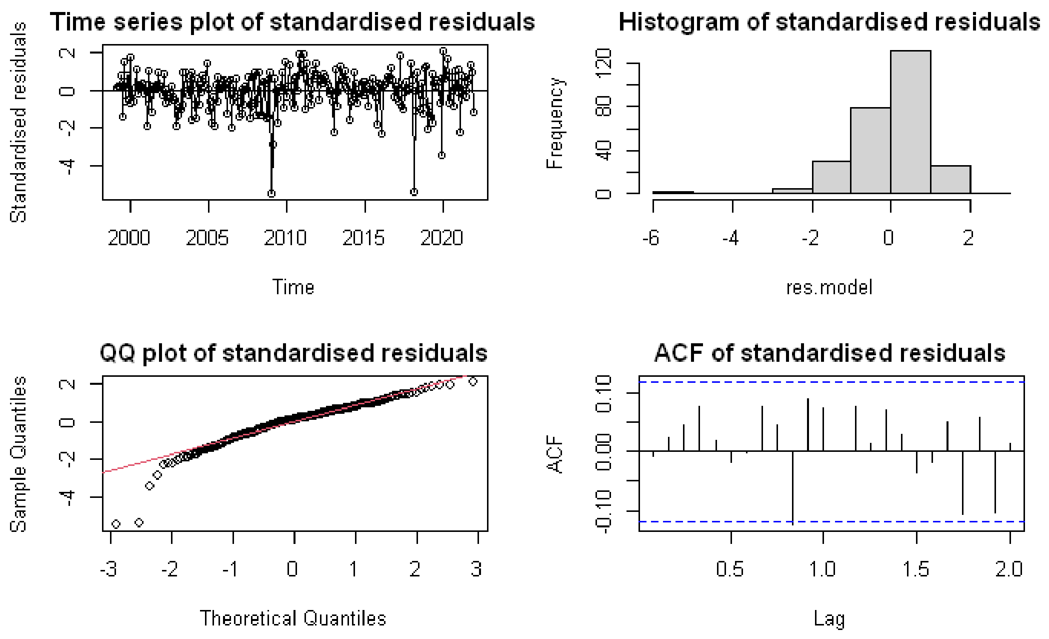


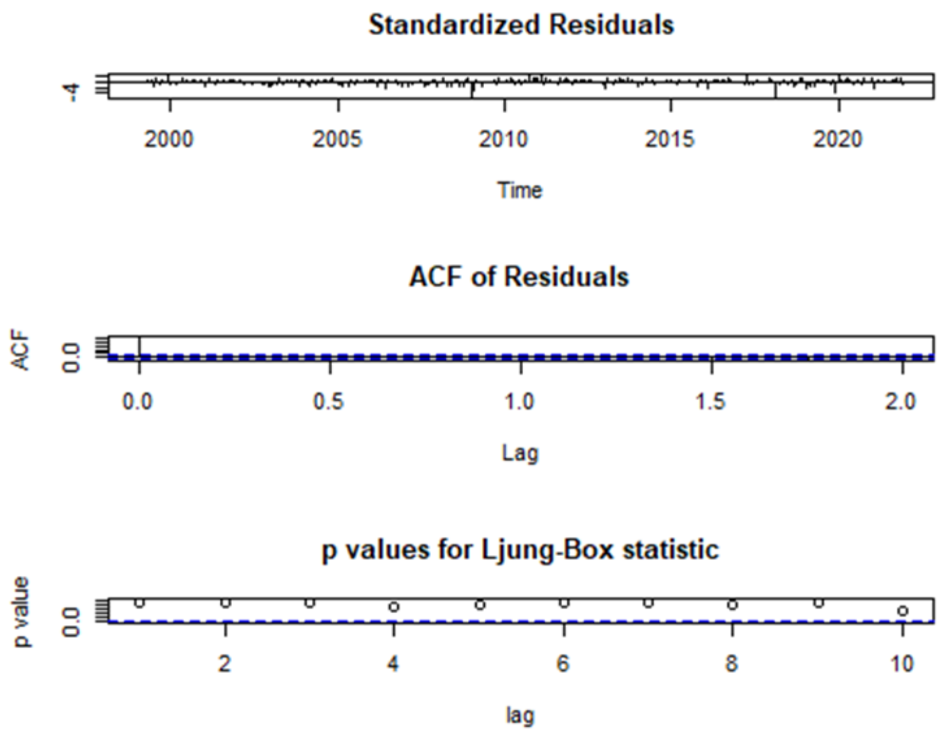
Shapiro-Wilk normality test

data: res.model **W** = 0.91721, **p-value** = 3.318e-11

Figure 42. Model ARMA {0,2}

```
## [1] "Model_ 1 0 1"
##
## z test of coefficients:
##
##             Estimate Std. Error  z value Pr(>|z|)
## ar1         0.145395   0.059845   2.4295  0.01512 *
## ma1        -1.000000   0.014186 -70.4922  < 2e-16 ***
## intercept   0.081703   0.059933   1.3632  0.17281
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



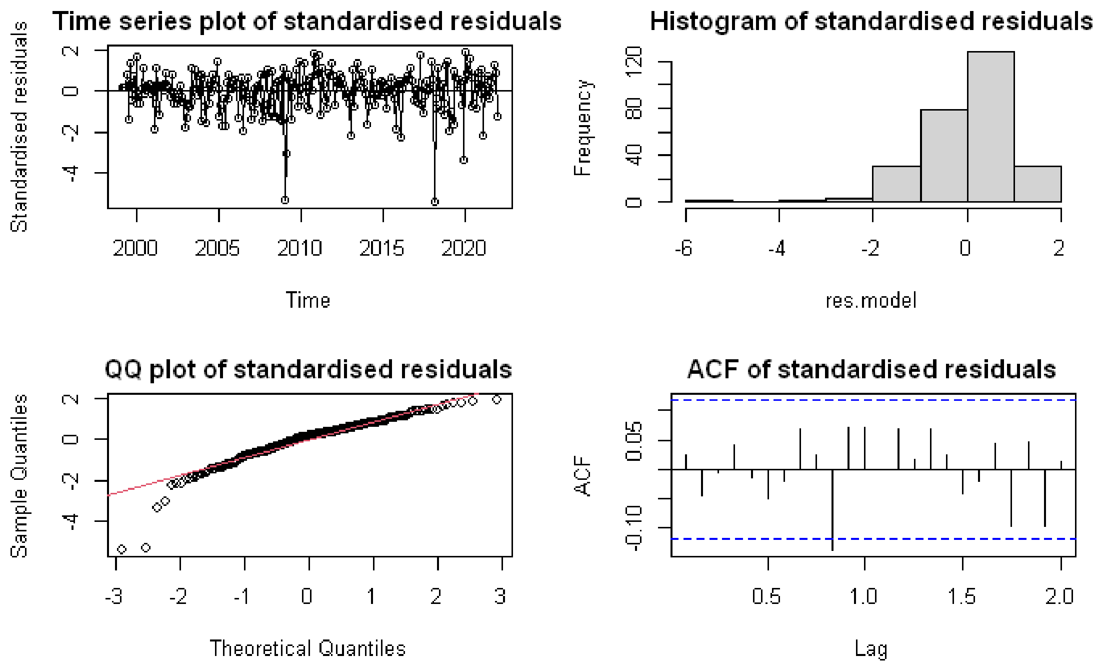


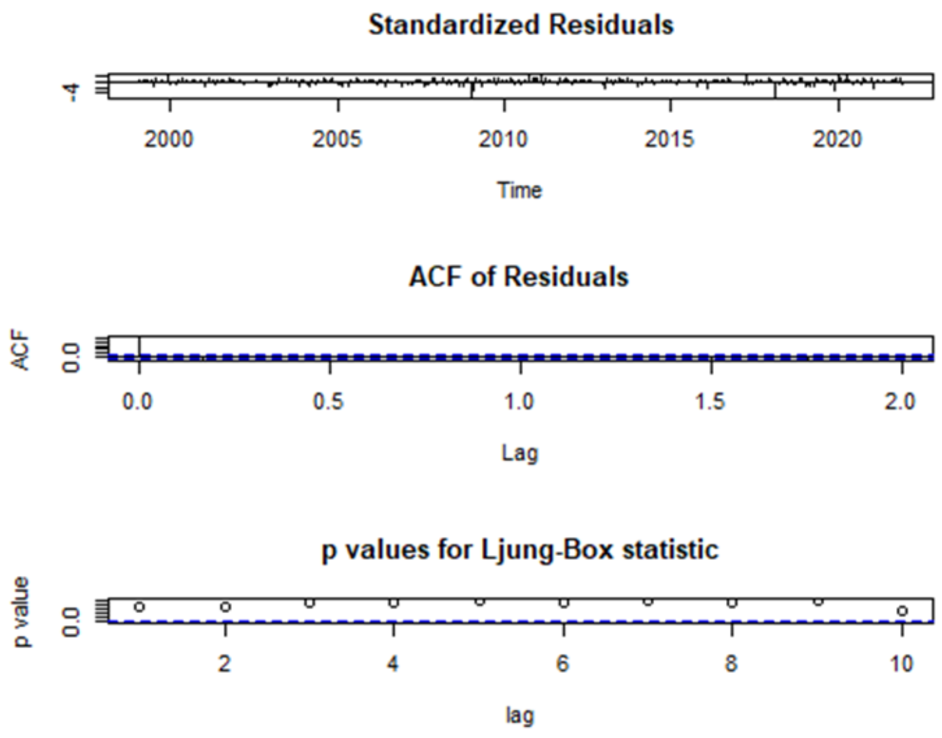
Shapiro-Wilk normality test

data: res.model **W =** 0.9159**, p-value =** 2.594e-11

Figure 43. Model ARMA {1,1}

```
## [1] "Model_ 1 0 2"
##
## z test of coefficients:
##
##            Estimate Std. Error z value Pr(>|z|)
## ar1        0.761589   0.240137  3.1715 0.001517 **
## ma1       -1.655392   0.282544 -5.8589 4.66e-09 ***
## ma2        0.655443   0.282283  2.3219 0.020236 *
## intercept  0.084837   0.072986  1.1624 0.245089
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



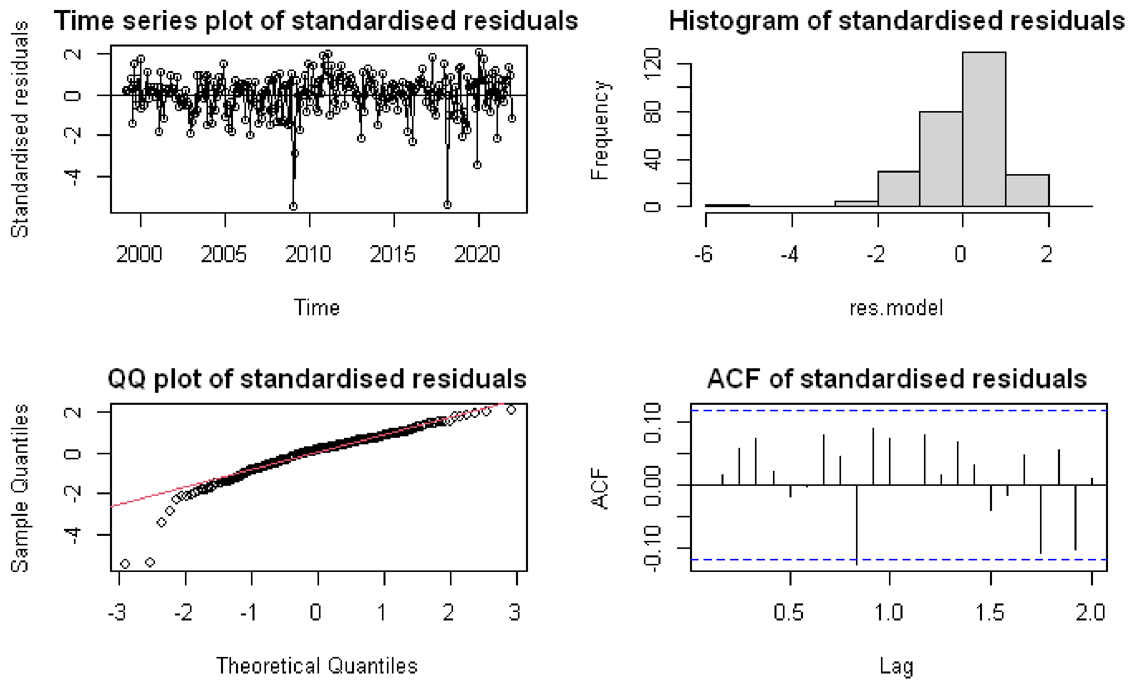


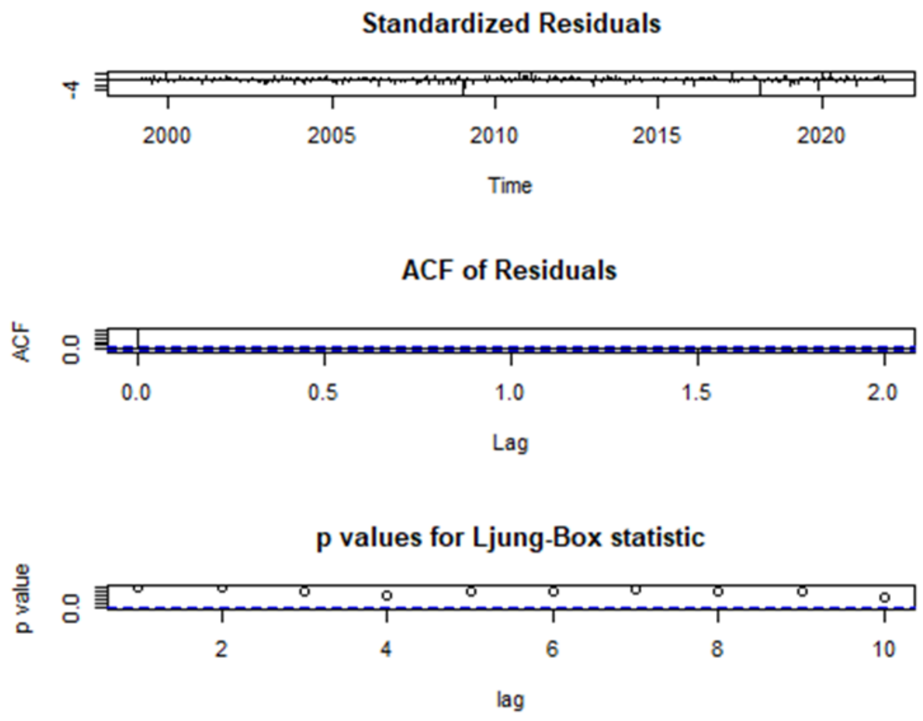
Shapiro-Wilk normality test

data: res.model W = 0.91376, p-value = 1.747e-11

Figure 44. Model ARMA {1,2}

```
## [1] "Model_ 1 0 3"

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

##
## z test of coefficients:
##
##             Estimate Std. Error z value Pr(>|z|)
## ar1        -0.332149        NaN     NaN      NaN
## ma1        -0.529051        NaN     NaN      NaN
## ma2        -0.396224        NaN     NaN      NaN
## ma3        -0.074693        NaN     NaN      NaN
## intercept   0.081634   0.059453  1.3731   0.1697
```

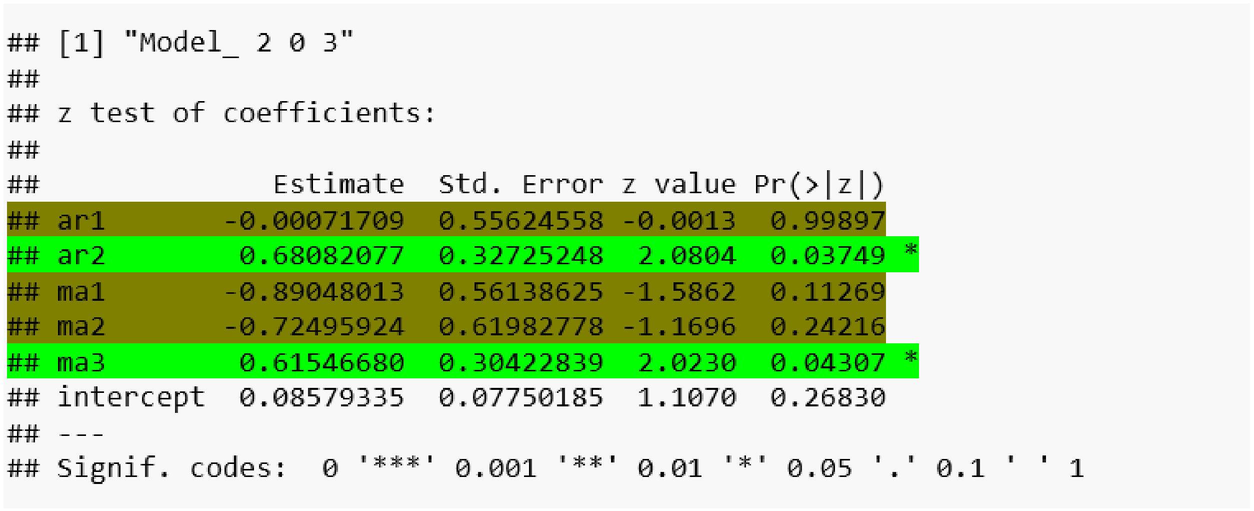




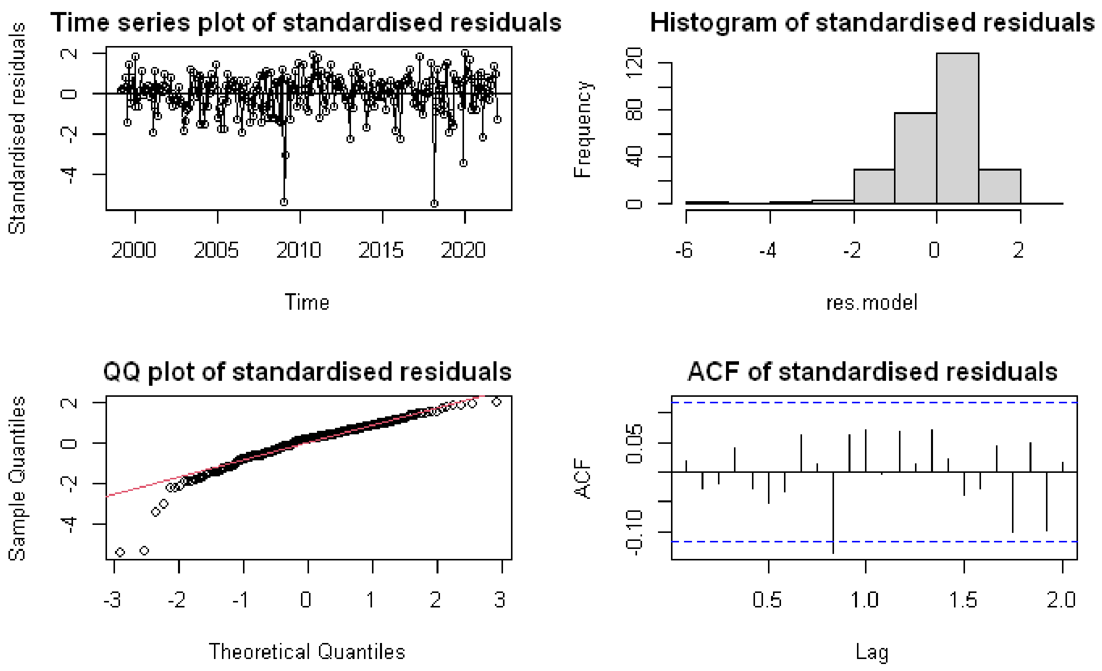
Shapiro-Wilk normality test

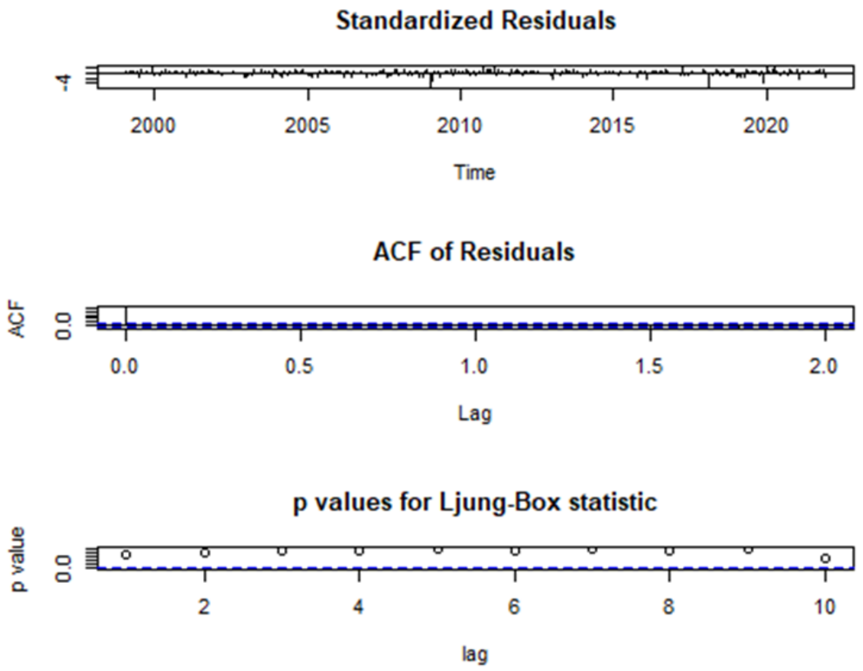
data: res.model W = 0.91506, p-value = 2.219e-11

Figure 45. Model ARMA {1,3}



```
## [1] "Model_ 2 0 3"
##
## z test of coefficients:
##
##                 Estimate  Std. Error z value Pr(>|z|)
## ar1          -0.00071709  0.55624558 -0.0013  0.99897
## ar2           0.68082077  0.32725248  2.0804  0.03749 *
## ma1          -0.89048013  0.56138625 -1.5862  0.11269
## ma2          -0.72495924  0.61982778 -1.1696  0.24216
## ma3           0.61546680  0.30422839  2.0230  0.04307 *
## intercept     0.08579335  0.07750185  1.1070  0.26830
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



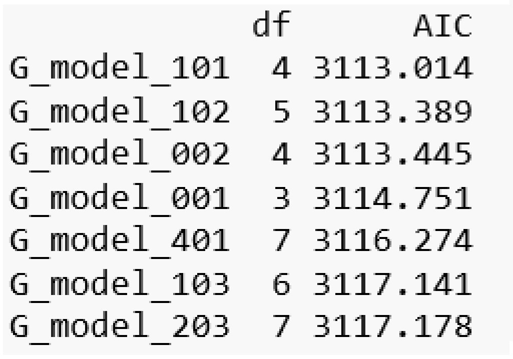


Shapiro-Wilk normality test

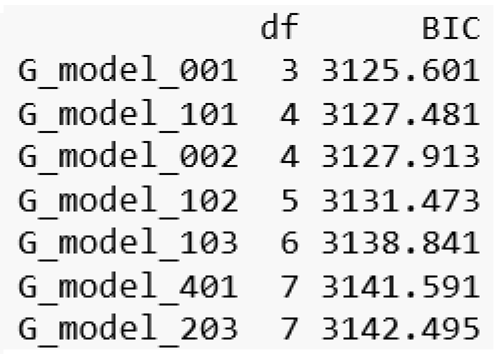
data: res.model W = 0.91593, p-value = 2.607e-11

ARMA(1,1) fitted to the rainfall return series is the best model in terms of parameter significance and diagnostics.

Figure 46. Model ARMA {2,3}



```
            df      AIC
G_model_101  4 3113.014
G_model_102  5 3113.389
G_model_002  4 3113.445
G_model_001  3 3114.751
G_model_401  7 3116.274
G_model_103  6 3117.141
G_model_203  7 3117.178
```



```
            df      BIC
G_model_001  3 3125.601
G_model_101  4 3127.481
G_model_002  4 3127.913
G_model_102  5 3131.473
G_model_103  6 3138.841
G_model_401  7 3141.591
G_model_203  7 3142.495
```

lowest BIC - model_001 second lowest - model_101

lowest AIC - model_101, second lowest - model_102

Figure 47. AIC and BIC values for GARCH models

Section 02: R code of the project

# - Set UP -

```
library(tseries)
library(dplyr)
library(TSA)
library(lmtest)
library(tidyr)
library(forecast)
library(data.table)
library(rugarch)
```

# - Functions -

```
residual.analysis <- function(model, std = TRUE,start = 2, class =
c("ARIMA","GARCH","ARMA-GARCH", "garch", "fGARCH")[1]){
  # If you have an output from arima() function use class = "ARIMA"
  # If you have an output from garch() function use class = "GARCH".
  # If you have an output from garchFit() function use class = "fGARCH" - added by HD -
5/5/21
  # If you have an output from garch() function from tseries package use class = "garch" -
added by HD - 20/5/21
  # Please note that you should use tseries package to be able to run this function for GARCH
models.
  # If you have an output from ugarchfit() function use class = "ARMA-GARCH"
  library(TSA)

  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 == "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(2,2))
  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
  tsdiag(model)
  par(mfrow=c(1,1))
}

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")')
 }
}
arima_modelling <- function(X, arima_list, sarima_list = c(0,0,0), freq = 1, method =
"THREE"){ # Data = Time series, list(arima_list) ARIMA models list
  library(lmtest)
  library(forecast)
  for (i in 1:length(arima_list)){

    if (method == "THREE"){
    print(paste("Model_",arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3])) # Print model to
analyse

    print(coeftest(arima(X,
                        order = c(arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3]),
                        seasonal =list(order = sarima_list,
                                      period = freq),
                        method = "ML"))) #Print coefficient test for each model


    print(paste("Model_CSS",arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3])) # Print
model to analyse

    print(coeftest(arima(X,
                        order = c(arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3]),
                        seasonal =list(order = sarima_list,
                                      period = freq),
                        method = "CSS"))) #Print coefficient test for each model


    print(paste("Model_CSS_ML",arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3])) # Print
model to analyse

    print(coeftest(arima(X,
                        order = c(arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3]),
                        seasonal =list(order = sarima_list,
                                      period = freq),
                        method = "CSS-ML"))) #Print coefficient test for each model
    }
    if (method == "ML"){
      print(paste("Model_",arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3])) # Print model
to analyse
```

```
     print(coeftest(arima(X,
                      order = c(arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3]),
                      seasonal =list(order = sarima_list,
                                  period = freq),
                      method = "ML"))) #Print coefficient test for each model
     }
   if (method == "CSS"){
     print(paste("Model_CSS",arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3])) # Print
model to analyse

     print(coeftest(arima(X,
                      order = c(arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3]),
                      seasonal =list(order = sarima_list,
                                  period = freq),
                      method = "CSS"))) #Print coefficient test for each model
   }
   if (method == "CSS-ML"){
     print(paste("Model_CSS_ML",arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3])) #
Print model to analyse

     print(coeftest(arima(X,
                      order = c(arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3]),
                      seasonal =list(order = sarima_list,
                                  period = freq),
                      method = "CSS-ML")))
   }
   if (method == "ARIMA"){
     print(paste("Model_CSS_ML",arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3])) #
Print model to analyse

     print(coeftest(Arima(X,
                      order = c(arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3]))))

   }
  }
}

SummaryResiduals <- function(data,model){
  Res_model=rstudent(model)
  Shapiro<-shapiro.test(Res_model)
  par(mfrow=c(2,2))
  plot(as.vector(time(data)),Res_model, type="l",
       ylab="Standarised residual values",
       xlab="Period of time",
       main="Time series plot of model residuals")
  hist(Res_model,
      xlab = "Standarised residuals",
      main = "Histogram of standarised model residuals")
  qqnorm(Res_model,
        main = "QQplot of model residuals ")
  qqline(Res_model,col="red")
  acf(Res_model,
     main = "ACF of model residuals")
  par(mfrow=c(1,1))
  Shapiro
} # Analysis of residuals trends

box_cox_analysis <- function(X,
                             title = "Time series plot (TRANSFORMED)",
                             ylab="Values",
                             xlab="Period",
                             title2 = "QQplot of Time series (TRANSFORMED)"){
  library(tseries)
  library(TSA)
  box_cox = BoxCox.ar(X)
  lambda <- box_cox$lambda[which(max(box_cox$loglike) == box_cox$loglike)]
  box_cox = (X^lambda-1)/lambda
  plot(box_cox,
      main = title,
      ylab = ylab,
      xlab = xlab,
      line = 0.6,
      type = 'o',
      lwd = 1)

  qqnorm(y=box_cox,
        main = title2, line=0.6, lwd = 1, pch=19)
  qqline(y=box_cox, col = 2, lwd = 2, lty = 2) # tails are quite off from both ends
  print(shapiro.test(box_cox))

  y = box_cox
  x = zlag(box_cox)       # Generate first lag
  index = 2:length(x)    # Create an index to get rid of the first NA value in x
  print(adf.test(X))
  print(pp.test(X))


  print(paste("CORRELATION = ",cor(y[index],x[index])))
  print(paste("LAMBDA =",lambda))
}

# **Data**
```

```
Monthly_rainfall = read.csv(file.choose(), header=TRUE)
Monthly_rainfall# check the data


# Reshaping the data-frame to long format
Monthly_rainfall.long <- pivot_longer(Monthly_rainfall, cols=2:13, names_to = "Month",
values_to = "Rainfall")
head(Monthly_rainfall.long,n=15)

summary(Monthly_rainfall.long)

# change character data type into numeric for rainfall column
Monthly_rainfall.long$Rainfall <- as.numeric(Monthly_rainfall.long$Rainfall)
head(Monthly_rainfall.long,n=15)
tail(Monthly_rainfall.long,n=15)


#-------------------------------
# Imputing Median per Month for missing values

# transform a data frame into a data.table to imputation
setDT(Monthly_rainfall.long)

Monthly_rainfall.long <- Monthly_rainfall.long[, Rainfall := ifelse(is.na(Rainfall),
                          median(Rainfall, na.rm = TRUE),
                          Rainfall),
                      by = Month]


#check values
head(Monthly_rainfall.long,n=15)

Monthly_rainfall.long

## Year 1994-1998 data is missing , therefore we will continue with data after 1999 and before
2022
RF_after_1999 <- Monthly_rainfall.long %>%
         dplyr::filter(Year > 1998 & Year  < 2022)
railfall_TS= ts(RF_after_1999$Rainfall,start=1999, frequency = 12)
```

#Descriptive Analytics****

```
summary(railfall_TS)
IQR(railfall_TS)
hist(railfall_TS, main = "Rainfall of Melbourne Botanical Gardens")
```

```
#*Time series plot*
p1=plot(railfall_TS
        , ylab="Monthly rainfall of Melbourne Botanical Gardens"
        , xlab="Year"
        , main="Monthly rainfall of Melbourne Botanical Gardens"
        , col='blue'
        , line=0.6
        , lwd = 1, pch=19
        , type='o')

plot(y=railfall_TS
     ,x=zlag(railfall_TS)
     ,ylab='rainfall'
     , xlab='Previous day rainfall'
     , main = "Scatter plot of rainfall in consecutive days"
     ,line=0.5
     , xlim = c(20, 100))


y = railfall_TS
x = zlag(railfall_TS)       # Generate first lag
index = 2:length(x)    # Create an index to get rid of the first NA value in x
cor(y[index],x[index])


#**Testing the data**

#*Normality*
qqnorm(y=railfall_TS  , main = "QQ plot of rainfall series", line=0.6, lwd = 1, pch=19)
qqline(y=railfall_TS  , col = 2, lwd = 2, lty = 2) # tails are quite off from both ends
shapiro.test(railfall_TS )

#*Stationary*
#NO STATIONARY p value greater than alpha

acf(railfall_TS, main = "ACF plot of rainfall series")
pacf(railfall_TS, main = "PACF plot of rainfall series")

# ACF - just one lag is above confidence boundary
# PACF - just one lag is above confidence boundary

adf.test(railfall_TS) # p-value = 0.01 (< 0.05 stationary)
pp.test(railfall_TS) # p-value = 0.01 (< 0.05 stationary)

#*Seasonality*
```

```
p1=plot(railfall_TS
        , ylab="Monthly rainfall of Melbourne Botanical Gardens"
        , xlab="Year"
        , main="Monthly rainfall of Melbourne Botanical Gardens"
        , col='blue'
        , line=0.6
        , lwd = 1) #pch=19
#, type='o')
points(y=railfall_TS,x=time(railfall_TS), pch=as.vector(season(railfall_TS)), col =
"black",lwd = 1.5, cex = 0.9)

par(mfrow =c(1,2))
acf(railfall_TS,lag.max = 60, main = "ACF plot of rainfall series")
pacf(railfall_TS,lag.max = 60,main = "PACF plot of rainfall series")
par(mfrow =c(1,1))
```

TREND MODELS

#- Linear Model -

```
railfall_TS_lm = lm(railfall_TS ~ time(railfall_TS)) # label the linear trend model as model1
summary(railfall_TS_lm)

plot(railfall_TS,type='o',ylab='y', main = "Fitted linear model: Monthly rainfall
series",line=0.5)
abline(railfall_TS_lm, col = 'red', lty=2) # add the fitted least squares line from model1

legend("bottomleft",lty=1, bty = "n" ,text.width = 8, col=c("black","red"),
      c("Monthly rainfall(mm)", "Fitted linear model"),cex=0.6)

SummaryResiduals(railfall_TS,railfall_TS_lm)
```

- Quadratic Model (2nd Order) -

```
t = time(railfall_TS)
t2 = t^2
railfall_TS_q2 = lm(railfall_TS~t+t2)
summary(railfall_TS_q2)

plot(ts(fitted(railfall_TS_q2)), ylim = c(min(c(fitted(railfall_TS_q2), as.vector(railfall_TS))),
max(c(fitted(railfall_TS_q2),as.vector(railfall_TS)))),
     ylab='y' , main = "Fitted quadratic model of order 2: Monthly rainfall series",
type="l",lty=2,col="red",line=0.5)
lines(as.vector(railfall_TS),type="o")
legend("bottomleft",lty=1, bty = "n" ,text.width = 8, col=c("black","red"),
      c("Monthly rainfall(mm)", "Fitted Quadratic Model of order 2"),cex=0.6)
```

```
SummaryResiduals(railfall_TS,railfall_TS_q2)

#- Quadratic Model (3rd Order) -

t = time(railfall_TS)
t3 = t^3
railfall_TS_q3 = lm(railfall_TS~t+t3)
summary(railfall_TS_q3)

plot(ts(fitted(railfall_TS_q3)), ylim = c(min(c(fitted(railfall_TS_q3), as.vector(railfall_TS))),
max(c(fitted(railfall_TS_q3),as.vector(railfall_TS)))),
    ylab='y' , main = "Fitted quadratic model of order 3: Monthly rainfall series",
type="l",lty=2,col="red",line=0.5)
lines(as.vector(railfall_TS),type="o")
legend("bottomleft",lty=1, bty = "n" ,text.width = 8, col=c("black","red"),
      c("Monthly rainfall(mm)", "Fitted Quadratic Model of order 3"),cex=0.6)

SummaryResiduals(railfall_TS,railfall_TS_q3)

#- Seasonal Model -

month.=season(railfall_TS)
model1=lm(railfall_TS~month.-1)
summary(model1)
SummaryResiduals(railfall_TS,model1)
plot(ts(fitted(model1)), ylab='y',main="Fitted seasonal model to monthly rainfall time series",
    ylim=c(min(c(fitted(model1),as.vector(railfall_TS))),
          max(c(fitted(model1), as.vector(railfall_TS)))),col="red")
lines(as.vector(railfall_TS),type="o")

#- Seasonal model with intercept -
model1.1=lm(railfall_TS~month.)
summary(model1.1)
SummaryResiduals(railfall_TS,model1.1)
plot(ts(fitted(model1.1)), ylab='y',main="Fitted seasonal model to monthly rainfall time
series",
    ylim=c(min(c(fitted(model1.1),as.vector(railfall_TS))),
          max(c(fitted(model1.1), as.vector(railfall_TS)))),col="red")
lines(as.vector(railfall_TS),type="o")

#- Cosine model -
har. <- harmonic(railfall_TS,1)
data<- data.frame(railfall_TS,har.)
model2 <- lm(railfall_TS~cos.2.pi.t.+sin.2.pi.t.,data=data)
summary(model2)
```

```
SummaryResiduals(railfall_TS,model2)
plot(ts(fitted(model2)),ylab='y',main="Fitted cosine wave to monthly rainfall time series.",
     ylim=c(min(c(fitted(model2),as.vector(railfall_TS))),
            max(c(fitted(model2), as.vector(railfall_TS)))
     ), col="green" )
lines(as.vector(railfall_TS),type="o")

#- ARIMA Models -
railfall_TS_2= ts(RF_after_1999$Rainfall)
box_cox_analysis(railfall_TS_2,
                 title = "Monthly rainfall of Melbourne Botanical Gardens (TR)",
                 ylab = "Monthly rainfall in (mm)",
                 xlab="Year",
                 title2 = "QQ plot of rainfall series (TR)")

railfall_TS_2_BC = (railfall_TS_2^(0.4)-1)/(0.4)

# _ Difference_

railfall_TS_2_BC_DFF1 <- diff(railfall_TS_2_BC)
plot(railfall_TS_2_BC_DFF1
        , ylab="Monthly rainfall in mm"
        , xlab="Year"
        , main="Monthly rainfall of Melbourne Botanical  Gardens  (TR)(DFF)"
        , col='dark blue'
        , line=0.6
        , lwd = 1, pch=19
        , type='o')

#_ ACF PACF _
par(mfrow=c(1,2))
acf(railfall_TS_2_BC, main = "ACF plot of rainfall series (TR)")
pacf(railfall_TS_2_BC, main = "PACF plot of rainfall series (TR)")
par(mfrow=c(1,1))

# _ EACF _
EACF <- eacf(railfall_TS_2_BC)

#_BIC_
plot(armasubsets(y = railfall_TS_2_BC, nar = 8, nma = 14, y.name = "AR()", ar.method =
"ols"))


#*MODELS PROPOSED*
arima_list <- list(c(1,0,1),c(0,0,1),c(1,0,0),c(0,0,2),
                   c(1,0,2),c(1,0,3),c(1,0,4))
```

```
arima_modelling(X = (railfall_TS_2^(0.4)-1)/(0.4),
            arima_list = arima_list)

for (i in 1:length(arima_list)){
  assign(paste0('model_',arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3]),
      arima(railfall_TS,
          order = c(arima_list[[i]][1],arima_list[[i]][2],arima_list[[i]][3]),
          method = "ML"))
}
sort.score(AIC(model_001, model_002, model_100, model_101, model_102, model_103,
model_104),
        score = "aic")

sort.score(BIC(model_001, model_002, model_100, model_101, model_102, model_103,
model_104),
        score = "bic")

Model__100<-Arima(y = (railfall_TS_2^(0.4)-1)/(0.4),order = c(1,0,0))
Model__101<-Arima(y = (railfall_TS_2^(0.4)-1)/(0.4),order = c(1,0,1))
residual.analysis(model = Model__100)

residual.analysis(model = Model__101)
```

#*Overfitting

```
arima_list2 <- list(c(1,0,0),c(1,0,1),c(2,0,0),c(2,0,1))

arima_modelling(X = (railfall_TS_2^(0.4)-1)/(0.4),
            arima_list = arima_list2,method = "ARIMA")

for (i in 1:length(arima_list2)){
  assign(paste0('Model__',arima_list2[[i]][1],arima_list2[[i]][2],arima_list2[[i]][3]),
      arima(railfall_TS,
          order = c(arima_list2[[i]][1],arima_list2[[i]][2],arima_list2[[i]][3])))
}
```

#*GARCH / ARCH*

```
r.rainfall=diff(log(railfall_TS))*100
par(mfrow=c(1,1))
plot(r.rainfall,ylab='Monthly rainfall of Melbourne Botanical Gardens',
    main = "Monthly rainfall of Melbourne Botanical Gardens")
adf.test(r.rainfall)# 0.01 The stationarity of return series is confirmed by the ADF test.

qqnorm(r.rainfall,main="Q-Q Normal Plot of Rainfall Returns.")
qqline(r.rainfall)
```

```
shapiro.test(r.rainfall)

par(mfrow=c(1,2))
acf(r.rainfall, main="ACF plot for return series.")
pacf(r.rainfall, main="PACF plot for return series.")

par(mfrow=c(1,1))
McLeod.Li.test(y=r.rainfall,main="McLeod-Li test statistics for rainfall series")

# - EACF -
eacf(r.rainfall)

# - BIC -
plot(armasubsets(y=r.rainfall,nar=8,nma=8,y.name='p',ar.method='ols'))

#_MODELS_
arima_list_GARCH <-list(c(4,0,1),c(0,0,1),c(0,0,2),c(1,0,1),c(1,0,2),c(1,0,3),c(2,0,3))

arima_modelling(X = r.rainfall,
            arima_list = arima_list_GARCH,method = "ML")

for (i in 1:length(arima_list_GARCH)){

assign(paste0('G_model_',arima_list_GARCH[[i]][1],arima_list_GARCH[[i]][2],arima_list_G
ARCH[[i]][3]),
     arima(r.rainfall,
          order =
c(arima_list_GARCH[[i]][1],arima_list_GARCH[[i]][2],arima_list_GARCH[[i]][3]),
          method = "ML"))
}

residual.analysis(model = G_model_401, std = TRUE,start = 2, class = "ARIMA")
residual.analysis(model = G_model_001, std = TRUE,start = 2, class = "ARIMA")
residual.analysis(model = G_model_002, std = TRUE,start = 2, class = "ARIMA")
residual.analysis(model = G_model_101, std = TRUE,start = 2, class = "ARIMA")
residual.analysis(model = G_model_102, std = TRUE,start = 2, class = "ARIMA")
residual.analysis(model = G_model_103, std = TRUE,start = 2, class = "ARIMA")
residual.analysis(model = G_model_203, std = TRUE,start = 2, class = "ARIMA")

sort.score(AIC(G_model_401,G_model_001,G_model_002,
            G_model_101,G_model_102,G_model_103,G_model_203), score = "aic")



sort.score(BIC(G_model_401,G_model_001,G_model_002,
            G_model_101,G_model_102,G_model_103,G_model_203), score = "bic")
```

```
plot(G_model_101,type='o',ylab='Standardised residuals', main="Time series plot of
standardised residuals ARMA (1,1)")
#ABS AND SQT
abs.r.res.Rainfall = abs(rstandard(G_model_101))
par(mfrow=c(1,2))
acf(abs.r.res.Rainfall, main="ACF plot for absolute return series.")
pacf(abs.r.res.Rainfall, main="PACF plot for absolute return series.")
eacf(abs.r.res.Rainfall)
par(mfrow=c(1,1))

sq.r.res.Rainfall= rstandard(G_model_101)^2
par(mfrow=c(1,2))
acf(sq.r.res.Rainfall, main="ACF plot for squared return series.")
pacf(sq.r.res.Rainfall, main="PACF plot for squared return series.")
par(mfrow=c(1,1))
eacf(sq.r.res.Rainfall)
```

#*ARMA + GARCH

```
model_101_01 <- fGarch::garchFit(~ arma(1,1)+garch(1,0),
                              data = r.rainfall, trace=F)

model_101_11 <- fGarch::garchFit(~ arma(1,1)+garch(1,1),
                              data = r.rainfall, trace=F)
residual.analysis(model = model_101_01, std = TRUE,start = 2, class = "fGARCH")
residual.analysis(model = model_101_11, std = TRUE,start = 2, class = "fGARCH")

df <- data.frame(AIC = c(model_101_01@fit$ics[1],model_101_11@fit$ics[1]),
              BIC = c(model_101_01@fit$ics[2],model_101_11@fit$ics[2]))
rownames(df) <- c("ARMA(1,1)+GARCH(0,1)","ARMA(1,1)+GARCH(1,1)")
df

spec <- ugarchspec(variance.model = list(model = "sGARCH",
                                    garchOrder = c(0, 1)
),
mean.model = list(armaOrder = c(1, 1)))
model_101_11_2 <- ugarchfit(spec = spec, data = r.rainfall,
                          solver = "hybrid",
                          solver.control = list(trace=0))

residual.analysis(model = model_101_11_2, class = "ARMA-GARCH")
plot(model_101_11_2,which = 1)
plot(model_101_11_2,which = 3)
```

# *FORECASTING ARMA + GARCH

```
forc = ugarchforecast(model_101_11_2,n.ahead=10,data=r.rainfall)
```

```
forc

Rainfall.positive = railfall_TS + min(abs(railfall_TS))+0.1

firstObs <- matrix(c(log(Rainfall.positive)[1]),1)
log.rainfall.diff1.back = diffinv(r.rainfall, xi = firstObs)
log.rainfall.diff1.back = exp(log.rainfall.diff1.back)
log.rainfall.diff1.back.original = log.rainfall.diff1.back - (min(abs(railfall_TS))+0.1)
log.rainfall.diff1.back.original - railfall_TS # Make sure you are doing it correctly!

frc <- forc@forecast$seriesFor
lastObs <- matrix(c(log(Rainfall.positive)[276]),1)
log.rainfall.diff1.back = diffinv(frc, xi = lastObs)
log.rainfall.diff1.back.frc = log.rainfall.diff1.back - (min(abs(railfall_TS))+0.1)

plot(railfall_TS, xlim= c(1999, 2022.83), ylim = c(min(railfall_TS),
                                       max(railfall_TS)),
    ylab = "Rainfall (mm)",
    main = "Forecasts from ARMA+GARCH model.")
lines(ts(as.vector(log.rainfall.diff1.back.frc), start = c(2022),frequency = 12), col="blue",
type="l")
legend("topleft", lty=1, pch=1, col=c("black","blue"), text.width = 18,
      c("Data", "Forecasts"))
```

References:

Brownlee, J., 2018. *A Gentle Introduction to SARIMA for Time Series Forecasting in Python*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/sarima-for-time-series-forecasting-in-python/ > [Accessed 4 May 2022].

Brownlee, J., 2018. *How to Model Volatility with ARCH and GARCH for Time Series Forecasting in Python*. [online] Machine Learning Mastery. Available at: <https://machinelearningmastery.com/develop-arch-and-garch-models-for-time-series-forecasting-in-python/ > [Accessed 17 May 2022].

Bureau of Meteorology. 2022. *Climate Driver Update*. [online] Available at: <http://www.bom.gov.au/climate/enso/ > [Accessed 4 May 2022].

Bureau of Meteorology. 2022. *Monthly rainfall- Melbourne Botanical Gardens*. [online] Available at: <http://www.bom.gov.au/jsp/ncc/cdio/weatherData/av?p_nccObsCode=139&p_display_type=dataFile&p_startYear=&p_c=&p_stn_num=086232 > [Accessed 4 May 2022].

Foo, K., 2018. *Seasonal lags: SARIMA modelling and forecasting*. [online] Medium. Available at: <https://medium.com/@kfoofw/seasonal-lags-sarima-model-fa671a858729 > [Accessed 4 May 2022].

Kumar, R., 2020. *Time Series Model(s) — ARCH and GARCH*. [online] Medium. Available at: <https://medium.com/@ranjithkumar.rocking/time-series-model-s-arch-and-garch-2781a982b448 > [Accessed 17 May 2022].

PennState: Statistics Online Courses. 2022. *11.1 ARCH/GARCH Models | STAT 510*. [online] Available at: <https://online.stat.psu.edu/stat510/lesson/11/11.1 > [Accessed 17 May 2022].