MATH1318 - Time Series Analysis

Course Project

Time Series Analysis of Average Melbourne Temperature

(2000 - 2012)

Member 1 ID: s3803839
Member 1 Name: Sahil Gupta

Member 2 ID: s3778500
Member 2 Name: Aditya Kakde

Table of Contents

1. Introduction

In this report, we will be analyzing the time series for Melbourne Average Monthly temperatures. The purpose of this analysis is to come up with a suitable model which can capture as much of the behaviour of this time series as possible. Then using this model, we will be forecasting the average temperature of Melbourne for next 10 months.

We will start by performing a descriptive analysis on this time series. After this analysis, we will comment on which model would be suitable to use out of deterministic and stochastic. After deciding the model and effectively the direction of our analysis, we will try and come up with candidate models to capture the behaviour of this time series. This will be followed by performing diagnostic checks on each candidate model. Diagnostic checks will include estimating the parameters and analyzing the residuals for each model.

Once the model has been finalized, we will use this model to forecast average temperature for next 10 months. We would end this report by commenting on the forecast and analysing the goodness of fit for the chosen model.

2. Method

While performing the analysis for this time series, we will be using R language to implement all the required tests and techniques. Also, we will be using respective packages in R to plot all the required plots and the time series. Using RMarkdown, we will be capturing all the findings and results from our analysis.

The packages required for this analysis would be as follows:

- dplyr
- tidyselect
- knitr
- timeSeries
- fUnitRoots
- TSA
- lmtest
- forecast
- tseries
- FitAR
- FSAdata

We would also be using a user-defined function sort.score.R().

3. Data

We will extract the time series from a csv file. This file has been downloaded from the CensusAtSchool, New Zealand website. This file contains monthly minimum and maximum temperatures recorded for multiple cities. We have the temperatures from January 2000 to October 2012. As we have the monthly minimum and maximum temperature values, it would be a good idea to compute the mean temperature of each month for Melbourne. This mean value will be a good representation for variation of Melbourne temperatures across the year.

Date	MelbourneMin	Melbournemax	MelbourneAvg
2000M01	15.9	24.3	20.10
2000M02	18.9	28.7	23.80
2000M03	16.4	25.1	20.75
2000M04	12.7	20.8	16.75
2000M05	9.9	16.0	12.95
2000M06	7.7	14.6	11.15

Output 1: Sample of records with calculated average temperature
(refer Appendix 1 for R code for this output)

As we are using R, all the required data is present in a data frame.

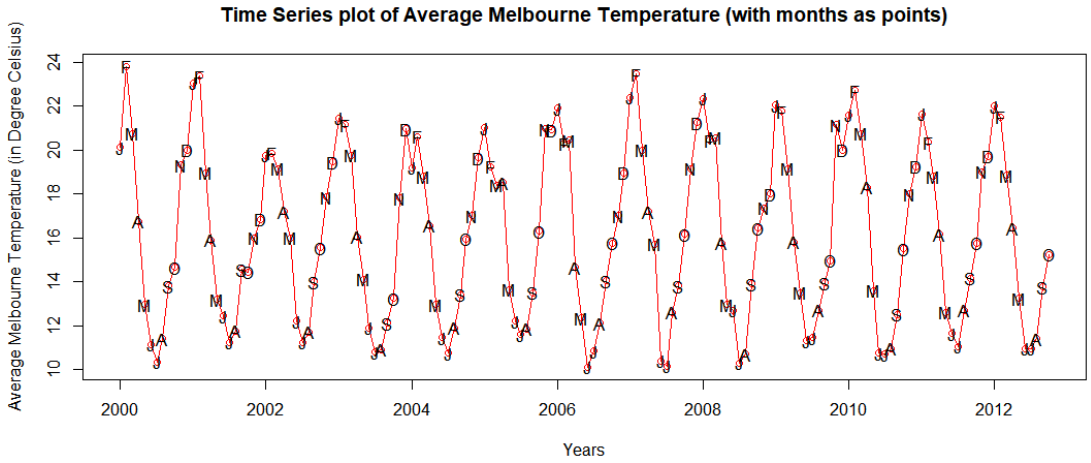
For us to carry out the analysis, we need to convert this data into a time series.

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2000	20.10	23.80	20.75	16.75	12.95	11.15	10.35	11.40	13.80	14.65	19.30	20.00
2001	23.05	23.35	18.95	15.90	13.20	12.45	11.25	11.75	14.55	14.45	16.00	16.85
2002	19.75	19.85	19.15	17.20	16.00	12.20	11.25	11.70	14.00	15.55	17.85	19.45
2003	21.45	21.15	19.75	16.05	14.10	11.90	10.80	10.90	12.10	13.25	17.80	20.95
2004	19.20	20.60	18.80	16.55	12.95	11.45	10.75	11.90	13.40	15.95	17.00	19.65
2005	21.00	19.25	18.40	18.50	13.65	12.20	11.60	11.85	13.50	16.30	20.95	20.90
2006	21.90	20.30	20.45	14.65	12.35	10.10	10.85	12.10	14.05	15.75	17.00	18.95
2007	22.40	23.45	20.00	17.20	15.70	10.40	10.15	12.60	13.80	16.15	19.15	21.25
2008	22.35	20.45	20.55	15.75	12.95	12.70	10.30	10.70	13.90	16.45	17.35	18.00
2009	22.05	21.75	19.15	15.80	13.50	11.30	11.45	12.70	13.95	14.95	21.15	20.00
2010	21.60	22.70	20.75	18.30	13.60	10.75	10.70	10.95	12.50	15.50	18.00	19.25
2011	21.65	20.35	18.80	16.15	12.60	11.65	11.05	12.70	14.15	15.75	19.00	19.70
2012	22.00	21.50	18.85	16.45	13.25	10.95	10.95	11.40	13.75	15.25		

Output 2: Time series data for Average Melbourne temperatures
(refer Appendix 2 for R code for this output)

Now that we have the time series ready at our disposal, we will start by plotting this time series and performing the descriptive analysis.

4. Descriptive Analysis

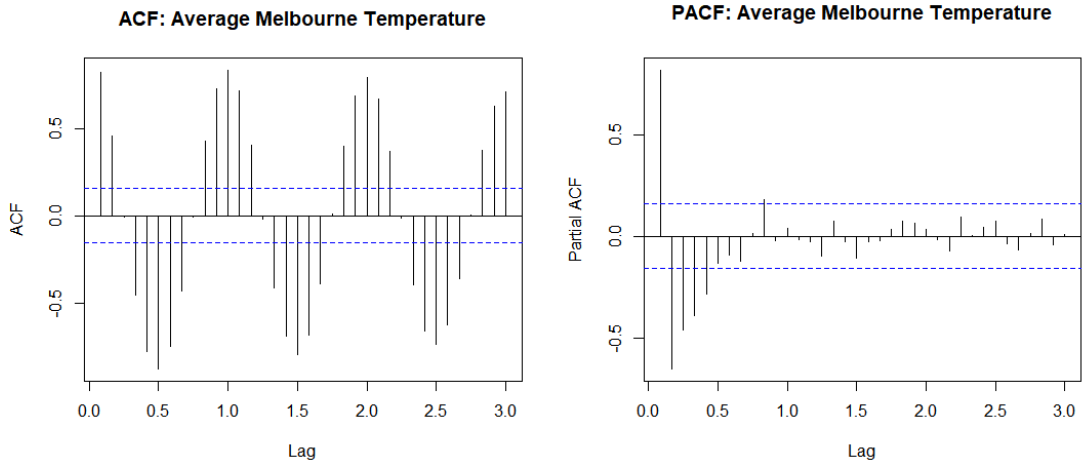


Output 3: Time series plot for Average Melbourne temperatures
(refer Appendix 3 for R code for this plot)

After plotting all the temperature values, we get the above time series. All the points represent temperature values for corresponding months. We can see that usually the temperatures are highest in January and February. Likewise, the temperatures are lowest in the months of June, July, and August. This pattern repeats itself across the entire time series. The basic properties for this time series would be:

1. **Trend:** At first glance, we cannot see any particular trend. We cannot be sure about this right now as the time series shows repeating patterns.
2. **Change in Variance:** Looking at the time series, the variance seems to change at around 2002 and again at around 2006. Again, we cannot be sure because of the repeating pattern in this time series.
3. **Seasonality:** Because of the repeating patterns, the time series strongly demonstrates the presence of seasonality.
4. **Behavior:** This time series seems to follow an auto-regressive behavior.
5. **Intervention Point:** In this time series, we cannot see any point beyond which the time series has changed drastically. So, we can say there is no intervention point in this time series.

Now that we have a basic understanding about the time series, we will plot the ACF(Auto Correlation Function) plot and PACF(Partial ACF) plot to analyze the presence of seasonality.

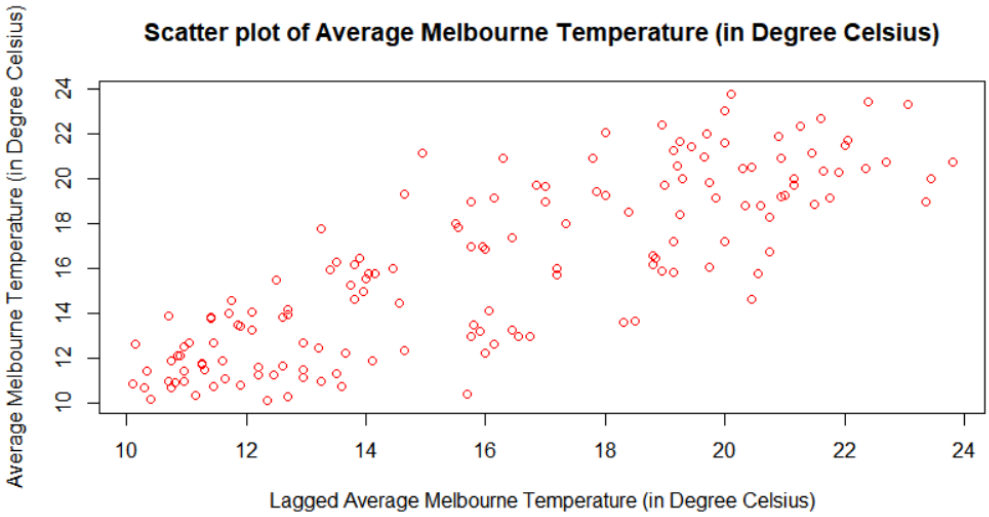


Output 4: ACF and PACF plot for Average Melbourne temperatures

(refer Appendix 4 for R code for this plot)

From the ACF plot, at lag 1, 2 and 3, we can see the repeating pattern which are significant. This demonstrates presence of seasonality in the time series. We cannot comment on the presence of trend from this plot due to seasonality. We can check for it once the seasonal component has been removed from the above plot.

We will now check if there is any correlation between the adjacent points in the time series.




```
[1] 0.8215435
```

Output 5: Scatter plot for Average Melbourne temperatures
(refer Appendix 5 for R code for this plot and output)

From the above scatter plot and the output, there is a strong correlation between adjacent points of this time series. This suggests a strong autocorrelation, which means, the temperature value of previous months affect the temperature value for current month.

5. Deterministic Model: Seasonal

Now that we have got an understanding about the time series, we will start by applying a deterministic seasonal model. This is a regression approach where; we try to find a model which computes the dependent variable based on independent variables. While using the deterministic model, one crucial assumption to be made is that the data does not have any random component.

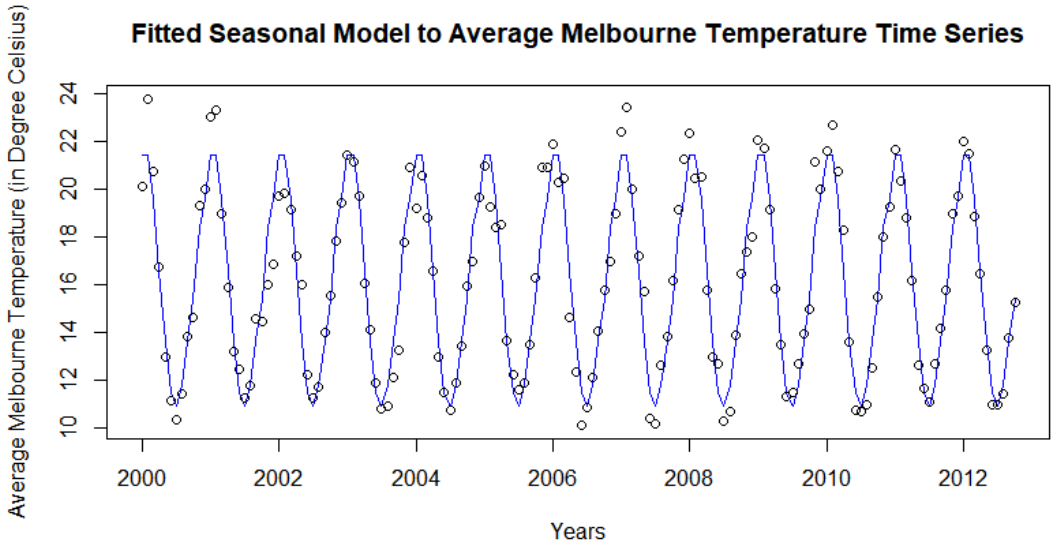
When we apply the seasonal model on our time series, we get below result:

```
Call:
lm(formula = tempAvg ~ month. - 1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.72917 -0.65000  0.01731  0.57500  2.77083

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
month.January    21.4231     0.2879   74.41   <2e-16 ***
month.February   21.4231     0.2879   74.41   <2e-16 ***
month.March      19.5654     0.2879   67.96   <2e-16 ***
month.April      16.5577     0.2879   57.51   <2e-16 ***
month.May        13.6000     0.2879   47.24   <2e-16 ***
month.June       11.4769     0.2879   39.86   <2e-16 ***
month.July       10.8808     0.2879   37.79   <2e-16 ***
month.August     11.7423     0.2879   40.79   <2e-16 ***
month.September  13.6500     0.2879   47.41   <2e-16 ***
month.October    15.3808     0.2879   53.42   <2e-16 ***
month.November   18.3792     0.2997   61.33   <2e-16 ***
month.December   19.5792     0.2997   65.34   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.038 on 142 degrees of freedom
Multiple R-squared:  0.9964,    Adjusted R-squared:  0.9961
F-statistic:  3254 on 12 and 142 DF,  p-value: < 2.2e-16
```

Output 6: Summary for fitted deterministic seasonal model
(refer Appendix 6 for R code for this output)

From the above results, we can see that co-efficients for all the independent variables are significant. We also got an unrealistic high R-squared value of 0.9961. This value basically represents the amount of variation explained by the model. In our case, the seasonal model explains 99.61% of the variation.

When we try and fit this model on our time series, we get below results:



Output 7: Fitted deterministic seasonal model over Average Melbourne Temperature Time Series (refer Appendix 7 for R code for this plot)

From the above plot, the model seems to be overfitted.

Now, looking at the assumption for Deterministic Models, the data should not have any random component. We cannot assure this condition on our data because we need to consider the natural variations in temperature which definitely have a strong random component. Also, when we look at the R-squared value and the fitted model, the seasonal model overfits our time series which is not really favourable.

Because of these reasons, we will not be using this model for forecasting.

We will now see how the stochastic seasonal model fits our time series.

6. Stochastic Model: SARIMA

Stochastic models are known to capture all the components in a time series. These components include the auto-regressive component, moving-average component, seasonal component, and the random component. The stochastic models are denoted as SARIMA(p, d, q)X(P, D, Q)_s. SARIMA stands for Seasonal Auto-Regressive Integrated Moving Average. p, d and q represent the parameters for the ordinary(non-seasonal) component and P, D and Q represent parameters for the seasonal model.

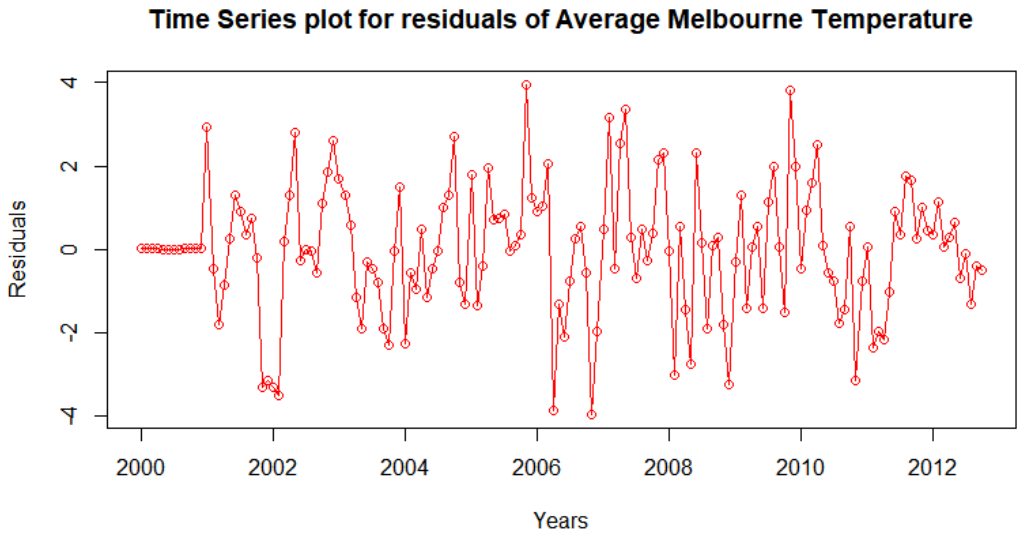
Our aim here would be to come up with appropriate values for parameters p, d, q, P, D and Q which would give us the most optimal model. We will start by finding the values for seasonal parameters P, D and Q.

6.1. Handling Seasonal Component

We are considering seasonal component in the beginning because, after handling and eliminating this component, we would be able to analyze the remaining components effectively.

6.1.1. *Eliminating Seasonality*

To eliminate the seasonal component, we will perform seasonal differencing. As this would be our first differencing, the value of D would be 1. We might need to perform second differencing after analysing the effects of 1st differencing.

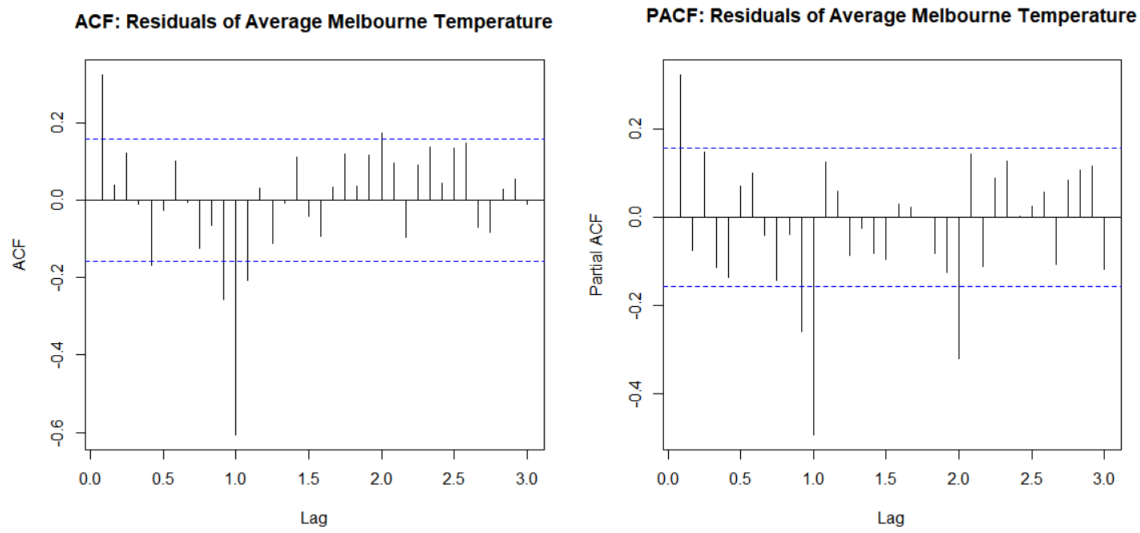


Output 8: Time Series plot of residuals of Average Melbourne Temperature
(refer Appendix 8 for R code for this plot)

Looking at the above output, this is the time series which is left out after eliminating the seasonality. Here we can see that there is no trend in the time series. We can see change in variance, but we will look at this in coming sections. Also, the series exhibits a moving average behaviour where the values seem to be fluctuating around the mean of 0.

Now, to find possible values for P and Q, we will plot the Auto-Correlation Function(ACF) and Partial ACF(PACF) plots.

6.1.2. *Defining Seasonal Parameters: P, Q*

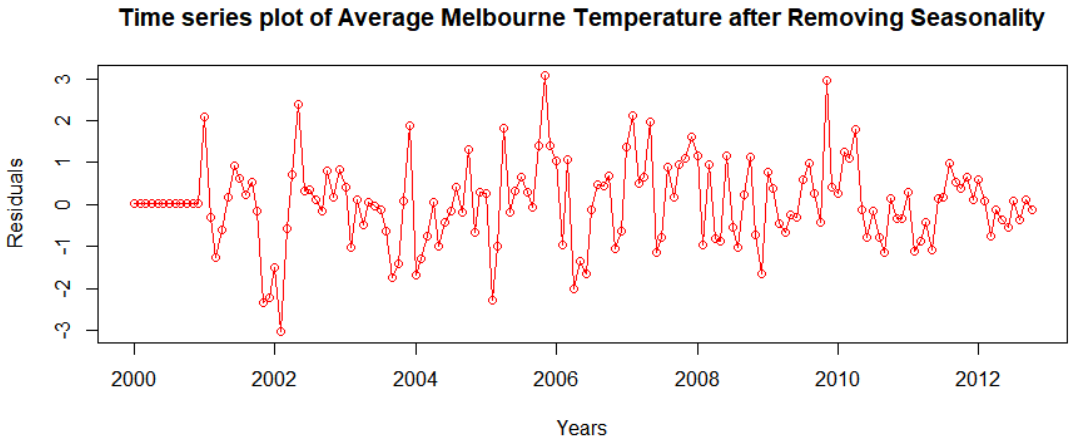


Output 9: ACF and PACF plot for Residuals of Average Melbourne temperatures
(refer Appendix 9 for R code for this plot)

Looking at the ACF plot, for lag 1, 2 and 3 where the seasonality is represented, we have significant value only at lag 1. So, from this ACF plot, we get the value for Q as 1.

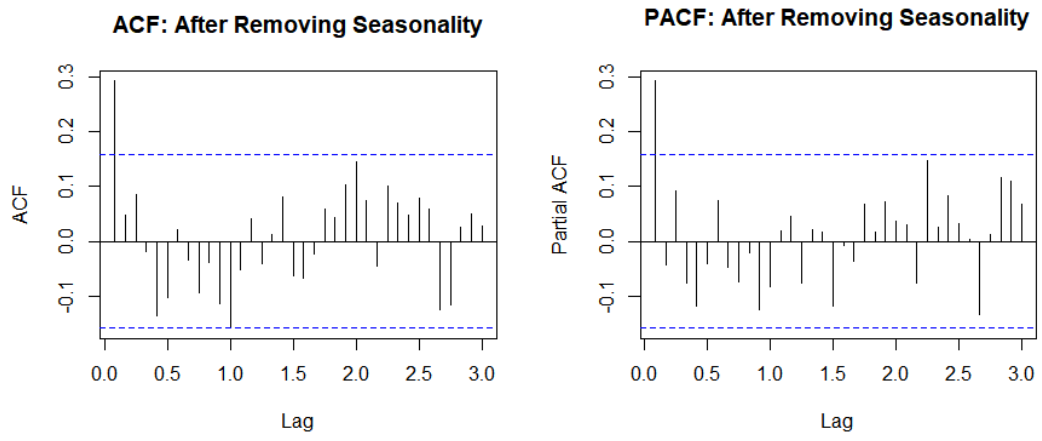
When we look at the PACF plot, again for lag 1, 2 and 3, where the seasonality is represented, we have significant values for all the lags under consideration. But when we look closely, there is a decreasing pattern for these values. In such situations where we see a trend, we can assume that there is no significant seasonal autoregressive component present. So, the value for P would be 0.

Now, we have the values for seasonal P, D and Q, which are, 0, 1 and 1, respectively. We will now use these values to capture and eliminate the seasonality. We will plot and check the remaining time series which does not have any seasonality and also check the ACF and PACF plots for the same.



Output 10: Time series plot of average Melbourne temperature after removing seasonality (refer Appendix 10 for R code for this plot)

We get the above times series after removing seasonality. We will be using this series to estimate the parameters for non-seasonal component.



Output 11: ACF and PACF plot for average Melbourne temperature after removing seasonality (refer Appendix 11 for R code for this plot)

Looking at both ACF and PACF plots, there are no significant values at lag 1, 2 or 3. This means that the seasonal component has been completely removed.

6.2. Model Estimation for non-Seasonal Component

Now that we have taken care of the seasonality, we can analyze the lags which are before lag 1. These lags represent the non-seasonal component. In the previous ACF and PACF plot, looking at the values for lags before lag 1, we do not see any specific pattern. Also, after first lag, there is drop in the values, which indicates there is no trend. So, we can say that the remaining time series is stationary, but, we need to first confirm this using a hypothesis test. This condition of stationarity would be our requirement to estimate the parameters.

So, to confirm if this series is stationary or not, we will perform the Augmented Dickey Fuller test.

6.2.1. ADF Test

The Dickey-Fuller unit-root test is used to test the null hypothesis that the process is difference nonstationary (the process is nonstationary but becomes stationary after first differencing). The alternative hypothesis is that the process is stationary.

The hypothesis test is given as follows:

H0: Process is Non−Stationary

HA: Process is Stationary

Decision Rules:

Reject,

- if p-value < 0.05 (significance level)
- if 95% CI of the parameter does not capture H0.

Otherwise, fail to reject H0H0.

Conclusion:

Test will be statistically significant if we reject H0. Otherwise, the test is not statistically significant.

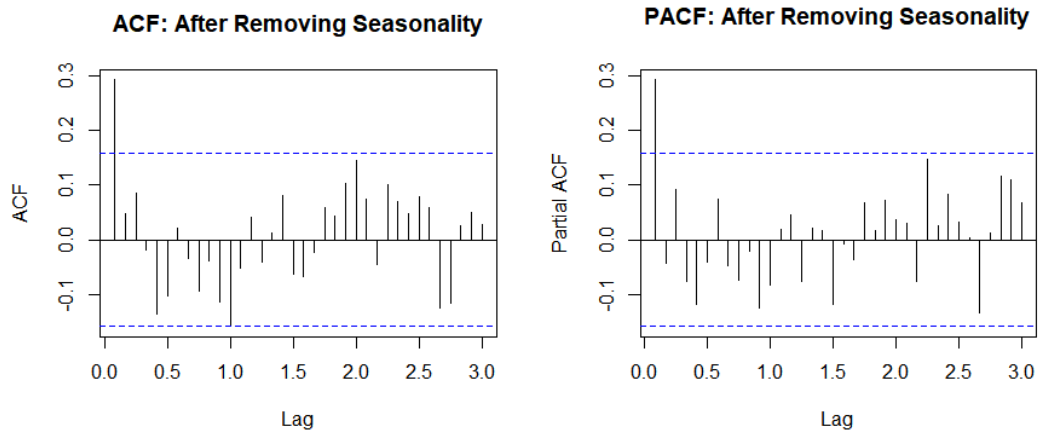
```
        Augmented Dickey-Fuller Test

data:  res_m2_temp
Dickey-Fuller = -5.3517, Lag order = 5, p-value = 0.01
alternative hypothesis: stationary
```

Output 12: Results of Augmented Dickey-Fuller Test
(refer Appendix 12 for R code for this output)

From the output, at significance level of 95%, we reject the Null Hypothesis as the p-value is 0.01 and which is less than 0.05. As the Null Hypothesis being that the series is not stationary, we can safely assume our time series under consideration to be stationary.

Now that we have confirmed the stationarity, we will come up with probable values for non-seasonal parameters p and q using ACF and PACF plots.

6.2.2. ACF and PACF Plots



Output 13: ACF and PACF plot for average Melbourne temperature after removing seasonality (refer Appendix 13 for R code for this plot)

From the above plot, we have just 1 significant lag in both ACF and PACF. So, from ACF we get the possible value for q as 1, and from PACF we get the possible value for p as 1 too.

So, from ACF and PACF plots, the possible candidate model which we get is,

SARIMA(1, 0, 1)X(0, 1, 1)_12

Now, to identify other possible values for the parameters p and q, we will compute the Extended Auto-Correlation Function(EACF) matrix.

6.2.3. EACF Method

The EACF method uses the fact that if the AR part of a mixed ARMA model is known, "filtering out" the autoregression from the observed time series results in a pure MA process that enjoys the cutoff property in its ACF.

```
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 o o o o o o o o o o o  o  o  o
2 x o x o o o o o o o o  o  o  o
3 x x x o o o o o o o o  o  o  o
4 x x o o o o o o o o o  o  o  o
5 x x o x o 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 x x o x o o o o o o  o  o  o
```

Output 14: EACF matrix for average Melbourne temperature after removing seasonality (refer Appendix 14 for R code for this matrix)

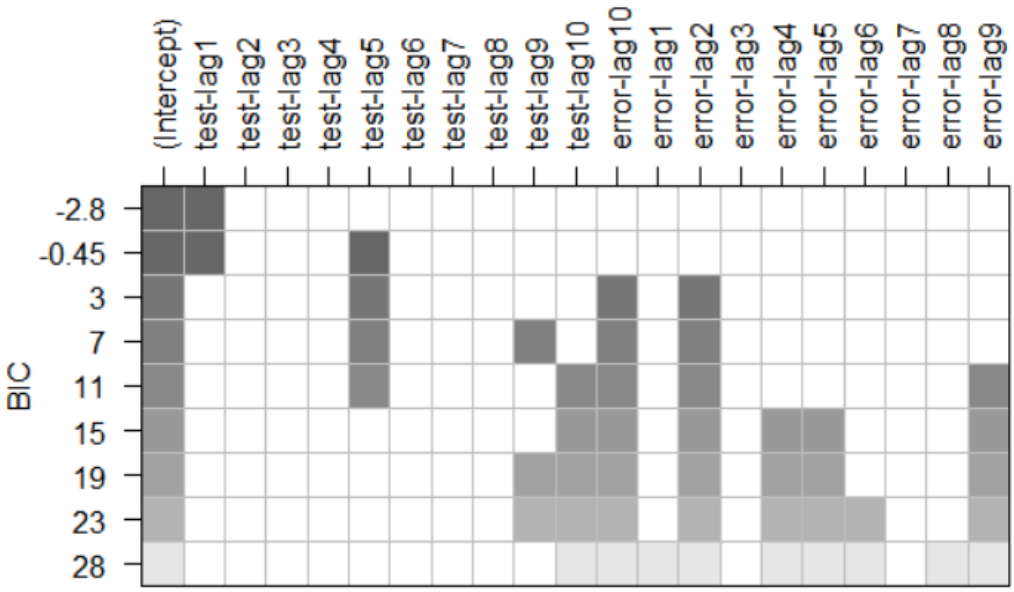
From the above matrix, we get the vertex at point (0, 1). So, the possible values for p and q which we could consider are (0, 1), (0, 2) and (1, 2). Based on the values for p and q, the possible candidate models which we get from the EACF matrix are

- **SARIMA(0, 0, 1)X(0, 1, 1)_12**
- **SARIMA(0, 0, 2)X(0, 1, 1)_12**
- **SARIMA(1, 0, 2)X(0, 1, 1)_12**

6.2.4. BIC Table

Bayesian Information Criterion (BIC) or Schwartz's Bayesian Criterion given as:

$$BIC = -2log\ (maximum\ likelihood) + klog\ (n)$$



Output 15: Bayesian Information Criterion table for average Melbourne temperature after removing seasonality (refer Appendix 15 for R code for this plot)

From the above BIC table, the values which get for p are 1, 5, 9 and 10. Similarly, the values which we get for q are 2, 4, 5, 9 and 10. Here would not be considering the p and q values of 9 and 10 as they will form very big models and violate the Principle of Parsimony.

So, the possible candidate models from the BIC table would be

- **SARIMA(1, 0, 2)X(0, 1, 1)_12**
- **SARIMA(1, 0, 4)X(0, 1, 1)_12**
- **SARIMA(1, 0, 5)X(0, 1, 1)_12**
- **SARIMA(5, 0, 2)X(0, 1, 1)_12**
- **SARIMA(5, 0, 4)X(0, 1, 1)_12**
- **SARIMA(5, 0, 5)X(0, 1, 1)_12**

6.3. Proposed Candidate Models

Now, considering the candidate models from ACF/PACF plots, EACF matrix and the BIC table, we get below models:

- SARIMA(1,0,1)x(0,1,1)_12
- SARIMA(0,0,1)x(0,1,1)_12
- SARIMA(0,0,2)x(0,1,1)_12
- SARIMA(1,0,2)x(0,1,1)_12
- SARIMA(1,0,4)x(0,1,1)_12
- SARIMA(1,0,5)x(0,1,1)_12
- SARIMA(5,0,2)x(0,1,1)_12
- SARIMA(5,0,4)x(0,1,1)_12
- SARIMA(5,0,5)x(0,1,1)_12

We will now perform multiple diagnostic checks to find the most optimal model. We will start by analysing the significance of non-seasonal parameters for the above models. As these parameters represent the Auto-Regressive and Moving-Average components, we will be able to identify and eliminate models which have insignificant components present. After this we will sort the models based on their AIC and BIC scores and select the ones having least AIC and BIC scores.

After shortlisting the models, we will analyze the residuals for all the shortlisted models. Based on the results on this analysis, we will finalize the most optimal model and use it for forecasting the average Melbourne temperature.

6.4. Diagnostic Check - Parameter Estimation for Non-Seasonal Component

Here, we will start by checking the significance of estimated parameters. For this we will be using Maximum Likelihood and the Conditional Sum of Squares method.

6.4.1. Parameter Estimation on proposed models

1. For model SARIMA(1,0,1)x(0,1,1)_12

```
z test of coefficients:

     Estimate Std. Error  z value  Pr(>|z|)
ar1  -0.15957    0.13860  -1.1513    0.2496
ma1   0.49624    0.11705   4.2395  2.24e-05 ***
sma1 -0.80705    0.05183 -15.5712 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


z test of coefficients:

      Estimate Std. Error z value Pr(>|z|)
ar1   0.046246   0.333555  0.1386   0.8897
ma1   0.276413   0.330726  0.8358   0.4033
sma1 -0.999999   0.121504 -8.2301   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output 16: Coefficients test for model SARIMA(1,0,1)x(0,1,1)_12
(refer Appendix 16 for R code for this output

From the above output, as per CSS method, we have moving-average(MA) component as significant and the auto-regressive(AR) component as not significant at significance level of 95%. As per the ML method, both AR and MA components are insignificant.

The above results suggest that this model is not optimal because as shown by ML and CSS method, we have at least 1 component which is not significant.

2. For model SARIMA(0,0,1)x(0,1,1)_12

```
z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ma1   0.307171   0.083835   3.664 0.0002483 ***
sma1 -0.752184   0.052039 -14.454 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ma1   0.320063   0.081891  3.9084  9.29e-05 ***
sma1 -0.999992   0.121678 -8.2183 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output 17: Coefficients test for model SARIMA(0,0,1)x(0,1,1)_12
(refer Appendix 17 for R code for this output)

From the above output, as per CSS method, we have both the MA and AR components as significant. As per the ML method too, both the components are significant.

The above results suggest that this model can be optimal as we have both the AR and MA components as significant at significance level of 95%.

3. For model SARIMA(0,0,2)x(0,1,1)_12

```
z test of coefficients:

      Estimate Std. Error  z value  Pr(>|z|)
ma1   0.303957   0.083417   3.6438 0.0002686 ***
ma2  -0.016887   0.079168  -0.2133 0.8310850
sma1 -0.752932   0.052144 -14.4394 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


z test of coefficients:

       Estimate Std. Error z value  Pr(>|z|)
ma1   0.3217612  0.0842207  3.8205 0.0001332 ***
ma2   0.0087943  0.0803726  0.1094 0.9128698
sma1 -0.9999970  0.1216098 -8.2230 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output 18: Coefficients test for model SARIMA(0,0,2)x(0,1,1)_12
(refer Appendix 18 for R code for this output)

This is an overfitted version of previous model. When we look at the output of test, we can see that both the ML and CSS test suggest that the second MA component is not significant at significance level of 95%.

4. For model SARIMA(1,0,2)x(0,1,1)_12

```
z test of coefficients:

      Estimate Std. Error  z value  Pr(>|z|)
ar1  -0.291963   0.132254  -2.2076   0.02727 *
ma1   0.661936   0.142659   4.6400 3.484e-06 ***
ma2   0.158428   0.088536   1.7894   0.07355 .
sma1 -0.825923   0.049086 -16.8260 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


z test of coefficients:

     Estimate Std. Error z value Pr(>|z|)
ar1   0.47067    0.64592  0.7287   0.4662
ma1  -0.14498    0.64997 -0.2231   0.8235
ma2  -0.11601    0.22835 -0.5080   0.6114
sma1 -1.00000    0.12048 -8.3000   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Output 19: Coefficients test for model SARIMA(1,0,2)x(0,1,1)_12
(refer Appendix 19 for R code for this output)

As seen in the above output, as per the CSS method, we have AR(1) and MA(1) components to be significant. The MA(2) component is not significant. This also seems to be an overfitted version of the previous model, but it gives the AR component as significant at the significance level of 95%.

5. For model SARIMA(1,0,4)x(0,1,1)_12

```
z test of coefficients:

      Estimate Std. Error  z value  Pr(>|z|)
ar1  -0.301901   0.130871  -2.3069   0.02106 *
ma1   0.678963   0.141374   4.8026 1.566e-06 ***
ma2   0.271531   0.111952   2.4254   0.01529 *
ma3   0.228412   0.119443   1.9123   0.05584 .
ma4   0.128992   0.090495   1.4254   0.15404
sma1 -0.829694   0.048998 -16.9333 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


z test of coefficients:

      Estimate Std. Error z value Pr(>|z|)
ar1  -0.258284   0.624977 -0.4133   0.6794
ma1   0.577224   0.617482  0.9348   0.3499
ma2   0.144000   0.213226  0.6753   0.4995
ma3   0.143242   0.125436  1.1420   0.2535
ma4   0.088973   0.104713  0.8497   0.3955
sma1 -0.999989   0.115090 -8.6887   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Output 20: Coefficients test for model SARIMA(1,0,4)x(0,1,1)_12
(refer Appendix 20 for R code for this output)

As seen in the first output which is as per the CSS method, we have MA(3) and MA(4) components which are not significant. We have the other components as significant.

As per the ML method, we do not have any significant components.

6. For model SARIMA(1,0,5)x(0,1,1)_12

```
z test of coefficients:

      Estimate Std. Error  z value  Pr(>|z|)
ar1  -0.297059   0.136196  -2.1811   0.02918 *
ma1   0.666326   0.154091   4.3242 1.531e-05 ***
ma2   0.256310   0.127149   2.0158   0.04382 *
ma3   0.218711   0.123748   1.7674   0.07716 .
ma4   0.107456   0.126925   0.8466   0.39721
ma5  -0.024417   0.101068  -0.2416   0.80910
sma1 -0.828907   0.049500 -16.7456 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1   0.671523   0.235464  2.8519  0.004346 **
ma1  -0.377648   0.239532 -1.5766  0.114886
ma2  -0.156698   0.114525 -1.3682  0.171239
ma3   0.079779   0.100694  0.7923  0.428190
ma4  -0.071091   0.094577 -0.7517  0.452246
ma5  -0.141352   0.091233 -1.5494  0.121296
sma1 -0.999998   0.119041 -8.4004 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Output 21: Coefficients test for model SARIMA(1,0,5)x(0,1,1)_12
(refer Appendix 21 for R code for this output)

This model is an over-fitted version of the previous model. As per the result from CSS method, MA(4) and MA(5) components are insignificant at significance level of 95%, which means adding more components over the previous model is not of much use.

Looking at the second output which is as per the ML method, except of AR(1) component, we do not have any other significant components.

7. For model SARIMA(5,0,2)x(0,1,1)_12

```
z test of coefficients:

      Estimate Std. Error  z value Pr(>|z|)
ar1   0.713127   0.436531   1.6336   0.1023
ar2  -0.468013   0.477642  -0.9798   0.3272
ar3   0.226980   0.176399   1.2867   0.1982
ar4  -0.134025   0.105566  -1.2696   0.2042
ar5  -0.069414   0.101266  -0.6855   0.4930
ma1  -0.375792   0.438253  -0.8575   0.3912
ma2   0.305241   0.389583   0.7835   0.4333
sma1 -0.796495   0.052606 -15.1408   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


z test of coefficients:

      Estimate Std. Error z value Pr(>|z|)
ar1   0.480432   0.709398  0.6772   0.4983
ar2  -0.406155   0.452672 -0.8972   0.3696
ar3   0.203323   0.164632  1.2350   0.2168
ar4  -0.057476   0.150584 -0.3817   0.7027
ar5  -0.103066   0.116949 -0.8813   0.3782
ma1  -0.170435   0.715860 -0.2381   0.8118
ma2   0.305726   0.344803  0.8867   0.3753
sma1 -0.999994   0.114832 -8.7084   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output 22: Coefficients test for model SARIMA(5,0,2)x(0,1,1)_12
(refer Appendix 22 for R code for this output)

This is a big model which was suggested by BIC table. As seen in the output, we do not have any AR and MA components as significant. So, the above result suggests that adding new components does not add any significance to the model.

8. For model SARIMA(5,0,4)x(0,1,1)_12

```
z test of coefficients:

      Estimate Std. Error  z value  Pr(>|z|)
ar1   0.939648   0.178174   5.2738 1.337e-07 ***
ar2  -0.136648   0.124009  -1.1019 0.2704959
ar3  -0.599200   0.083088  -7.2117 5.527e-13 ***
ar4   0.782618   0.053448  14.6425 < 2.2e-16 ***
ar5  -0.277999   0.011551 -24.0666 < 2.2e-16 ***
ma1  -0.549820   0.161465  -3.4052 0.0006612 ***
ma2  -0.199760   0.083023  -2.4061 0.0161251 *
ma3   0.802349   0.081337   9.8645 < 2.2e-16 ***
ma4  -0.797346   0.124100  -6.4250 1.319e-10 ***
sma1 -0.801749   0.054653 -14.6699 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1   0.017665         NA      NA        NA
ar2  -0.142374         NA      NA        NA
ar3   0.483473         NA      NA        NA
ar4   0.414609         NA      NA        NA
ar5  -0.252283         NA      NA        NA
ma1   0.311936         NA      NA        NA
ma2   0.180711         NA      NA        NA
ma3  -0.411415         NA      NA        NA
ma4  -0.586118         NA      NA        NA
sma1 -0.999999   0.147827 -6.7647 1.336e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output 23: Coefficients test for model SARIMA(5,0,4)x(0,1,1)_12
(refer Appendix 23 for R code for this output)

From the above output, in the results which we get using CSS method, except for AR(2) component, all other components are significant. This suggest a good model. We will decide if we should consider this model after sorting all the models based on AIC and BIC scores.

9. For model SARIMA(5,0,5)x(0,1,1)_12

```
z test of coefficients:

       Estimate Std. Error  z value  Pr(>|z|)
ar1   0.9329278  0.0650350  14.3450 < 2.2e-16 ***
ar2  -0.1306872  0.0902932  -1.4474  0.147795
ar3  -0.5578322  0.0676510  -8.2457 < 2.2e-16 ***
ar4   0.7406635  0.0386144  19.1810 < 2.2e-16 ***
ar5  -0.2419794  0.0109124 -22.1748 < 2.2e-16 ***
ma1  -0.5575022         NA        NA        NA
ma2  -0.2071144  0.0750613  -2.7593  0.005793 **
ma3   0.7811954  0.0506004  15.4385 < 2.2e-16 ***
ma4  -0.8190736  0.0110288 -74.2668 < 2.2e-16 ***
ma5  -0.0024479         NA        NA        NA
sma1 -0.8016168  0.0554265 -14.4627 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


z test of coefficients:

      Estimate Std. Error z value  Pr(>|z|)
ar1  -0.015663   0.232520 -0.0674 0.9462921
ar2   0.095880   0.167580  0.5721 0.5672242
ar3   0.339168   0.133201  2.5463 0.0108877 *
ar4   0.514185   0.152085  3.3809 0.0007225 ***
ar5  -0.494750   0.185431 -2.6681 0.0076281 **
ma1   0.354784   0.304318  1.1658 0.2436811
ma2  -0.046932   0.246667 -0.1903 0.8491009
ma3  -0.287297   0.192000 -1.4963 0.1345645
ma4  -0.665075   0.224579 -2.9614 0.0030621 **
ma5   0.220289   0.230169  0.9571 0.3385288
sma1 -0.999991   0.158300 -6.3170 2.666e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Output 24: Coefficients test for model SARIMA(5,0,5)x(0,1,1)_12
(refer Appendix 24 for R code for this output)

Looking at the results from CSS method, for MA(1) and MA(5) components, the test was not able to find any values. Except for AR(2) and the components which were failed to capture, we have all other components as significant.

Now that we know the significance of components for all the candidate models, we will sort these models based on their AIC and BIC score.

6.4.2. AIC and BIC Sort

We will first sort the models based on their AIC scores.

	df	AIC
model_001_ml	3	436.4285
model_101_ml	4	438.4088
model_002_ml	4	438.4166
model_102_ml	5	440.2582
model_105_ml	8	442.3082
model_104_ml	7	442.8981
model_502_ml	9	444.1256
model_504_ml	11	445.7989
model_505_ml	12	446.4032

Output 25: Models sorted as per AIC scores
(refer Appendix 25 for R code for this output)

Looking at the above output, models SARIMA(0,0,1)x(0,1,1)_12, SARIMA(1,0,1)x(0,1,1)_12 and SARIMA(0,0,2)x(0,1,1)_12 are the ones having lowest AIC score. Out of these models, we know from the parameter estimation that, we have all the components as significant for model SARIMA(0,0,1)x(0,1,1)_12. We will still be considering other models for residual analysis to check if we get satisfactory behavior of the residuals.

We will now sort the models as per their BIC score and check if we get the same models.

	df	BIC
model_001_ml	3	445.2960
model_101_ml	4	450.2321
model_002_ml	4	450.2399
model_102_ml	5	455.0373
model_104_ml	7	463.5889
model_105_ml	8	465.9548
model_502_ml	9	470.7280
model_504_ml	11	478.3130
model_505_ml	12	481.8732

Output 26: Models sorted as per BIC scores
(refer Appendix 26 for R code for this output)

From the above output, we get same 3 models after sorting them by their BIC scores.

6.4.3. Overfitting

After performing the AIC and BIC sort, we usually try and overfit the selected models to check for more potential models. But here, we have already performed the parameter estimation for the overfitted version of selected models.

So, now we will proceed ahead with the residual analysis for the models SARIMA(0,0,1)x(0,1,1)_12, SARIMA(0,0,2)x(0,1,1)_12 and SARIMA(1,0,1)x(0,1,1)_12.

6.5. Diagnostic Check - Residual Analysis for Non-Seasonal Component

The purpose of residual analysis is to check if the residuals behave like white noise, i.e., they are normally distributed, they do not have any trend and there are no significant autocorrelations. For all the 3 models chosen in previous step, we will check for normality of the residuals using the Shapiro-Wilk test for normality. Next, we will check if there is any trend and if there exist any significant autocorrelation.

The Shapiro-Wilk test is used to test the normality of residuals. The hypothesis test is given as follows:

H0: Residuals are normally distributed

HA: Residuals are not normally distributed

Decision Rules:

Reject,

- if p-value < 0.1 (significance level)
- if 90% CI of the parameter does not capture H0.

Otherwise, fail to reject H0.

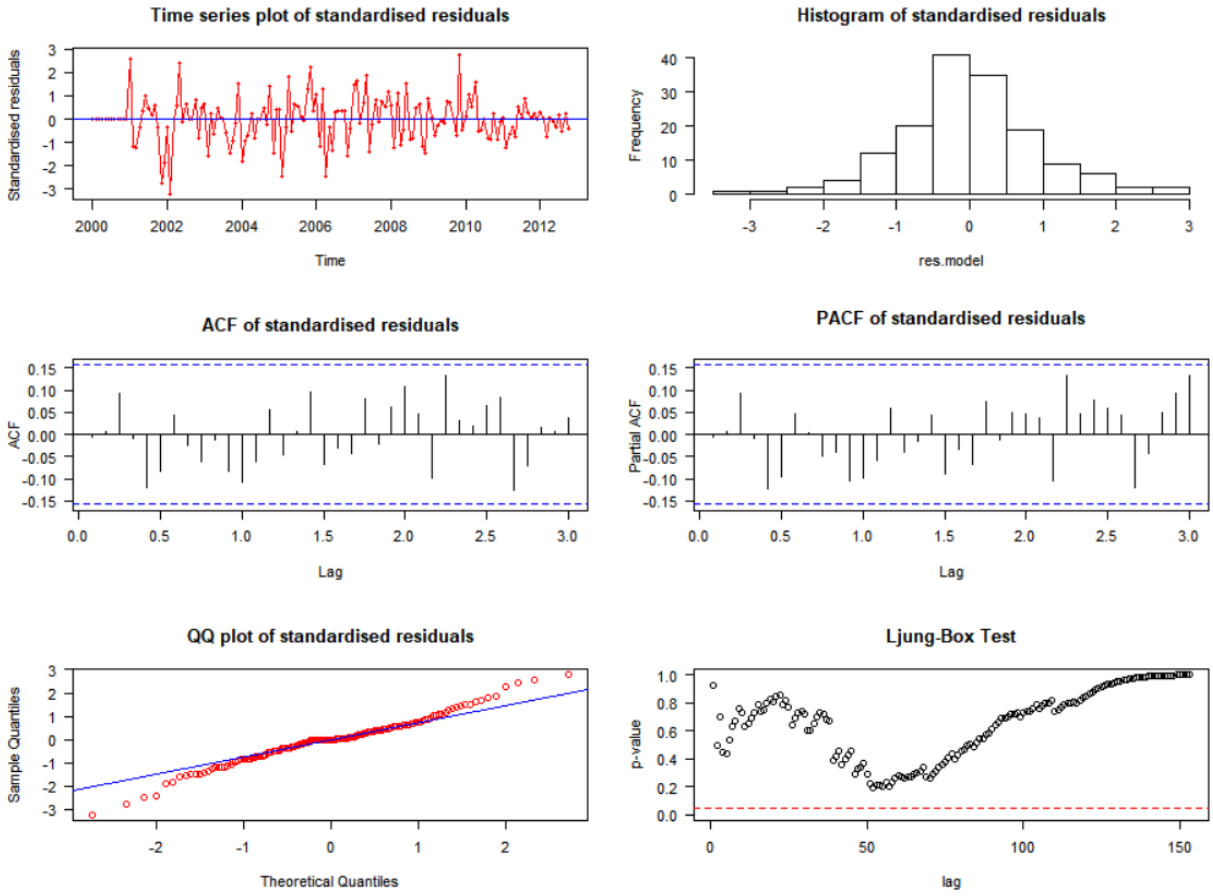
Conclusion:

Test will be statistically significant if we reject H0. Otherwise, the test is not statistically significant.

Here, we will be using the residuals of the model fitted using CSS method. This is because, for the 3 chosen models, we referred to the significance of co-efficients obtained by the CSS method.

6.5.1. For model SARIMA(0,0,1)x(0,1,1)_12

```
        Shapiro-Wilk normality test

data:  res.model
W = 0.97666, p-value = 0.01019
```



Output 27: Residual Analysis for model SARIMA(0,0,1)x(0,1,1)_12
(refer Appendix 27 for R code for this output and plot)

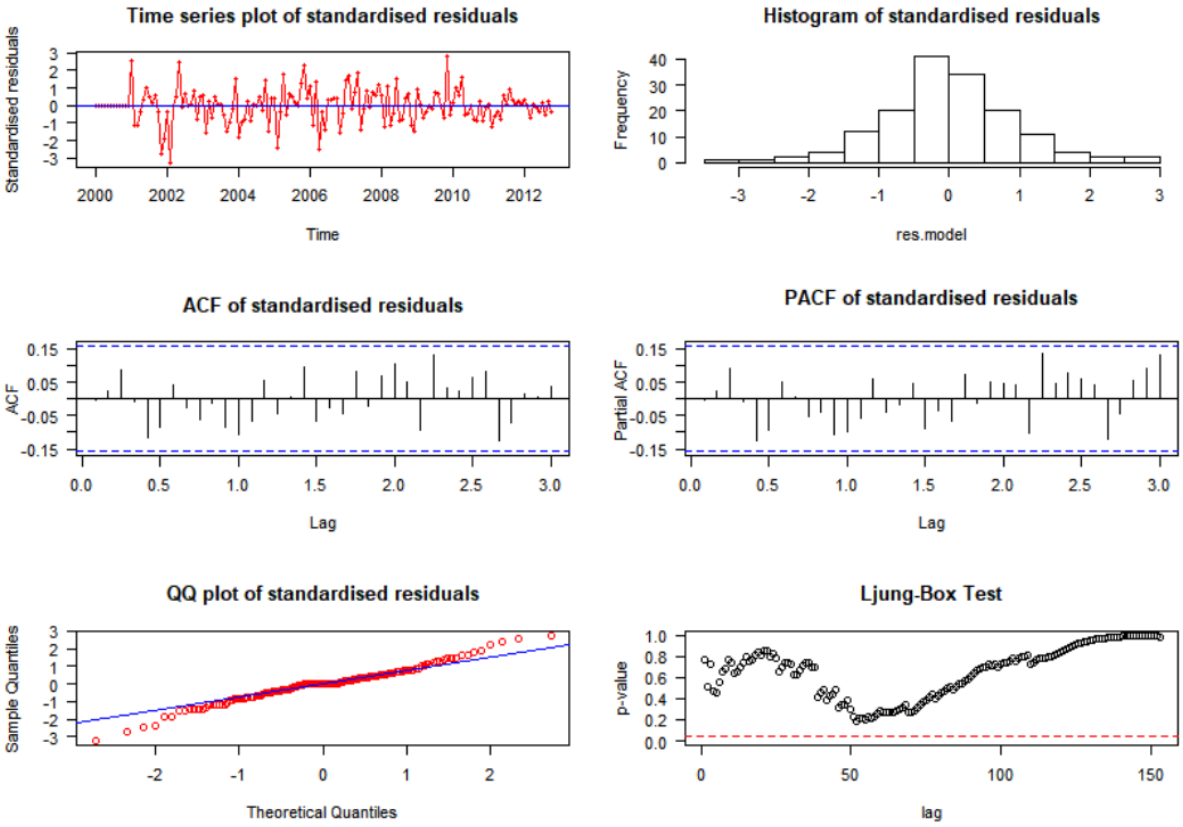
Looking at output from the Shapiro-Wilk normality test, with p-value = 0.01019 and significance level of 90%, we fail to reject the hypothesis, which means that the residuals are normally distributed.

Looking at the time series plot, the values are fluctuating along the constant mean of zero and are not showing any trend. Looking at the histogram and the qqplot, we see that the residuals are normally distributed. Both ACF and PACF plots demonstrate the behaviour of white noise. And, when we finally look at the Ljung Box test output, we can see that we do not have any point below the confidence interval, which means there is no significant auto-correlation present in the residuals.

All these points support this model as an optimal model.

6.5.2. For model SARIMA(0,0,2)x(0,1,1)_12

```
        Shapiro-Wilk normality test

data:  res.model
W = 0.97668, p-value = 0.01025
```



Output 28: Residual Analysis for model SARIMA(0,0,2)x(0,1,1)_12
(refer Appendix 28 for R code for this output and plot)

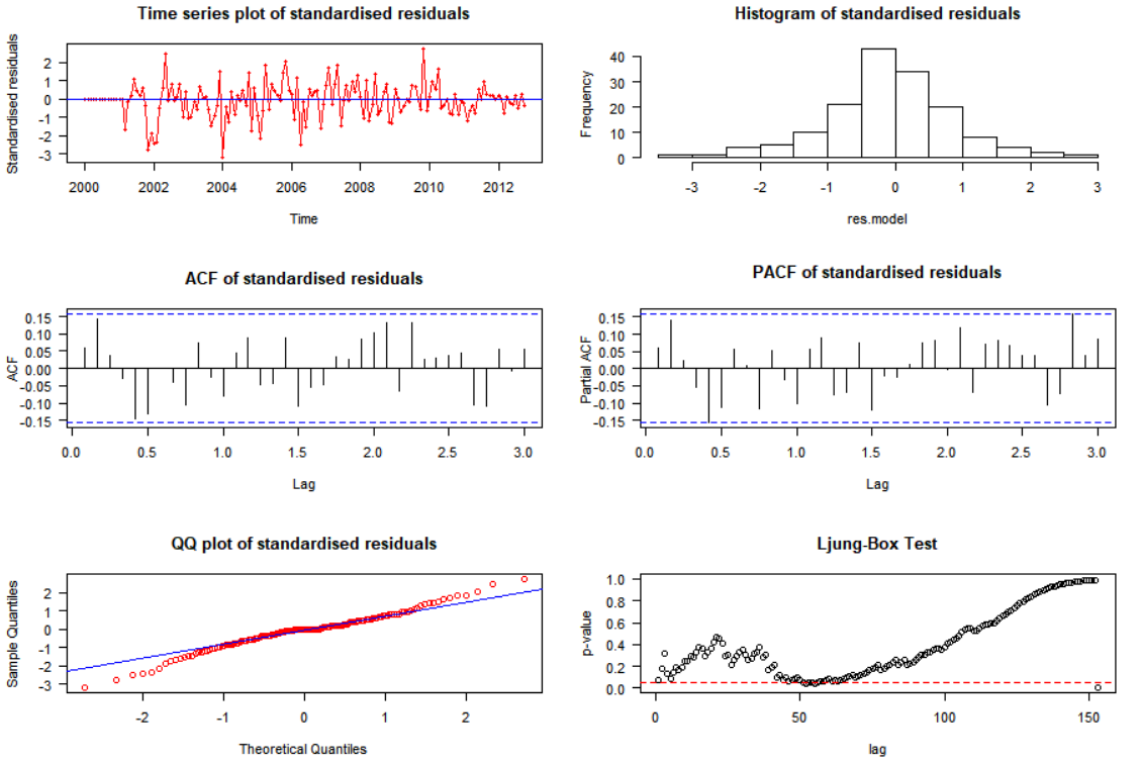
Looking at output from the Shapiro-Wilk normality test, with p-value = 0.01025 and significance level of 90%, we fail to reject the hypothesis, which means that the residuals are normally distributed.

Looking at the time series plot, the values are fluctuating along the constant mean of zero and are not showing any trend. Looking at the histogram and the qqplot, we see that the residuals are normally distributed. Both ACF and PACF plots demonstrate the behaviour of white noise. And, when we finally look at the Ljung Box test output, we can see that we do not have any point below the confidence interval, which means there is no significant auto-correlation present in the residuals.

All these points support this model too as an optimal model.

6.5.3. For model SARIMA(1,0,1)x(0,1,1)_12

```
        Shapiro-Wilk normality test

data:  res.model
W = 0.97559, p-value = 0.007745
```



Output 29: Residual Analysis for model SARIMA(1,0,1)x(0,1,1)_12
(refer Appendix 29 for R code for this output and plot)

Looking at output from the Shapiro-Wilk normality test, with p-value = 0.007745 and significance level of 90%, we reject the hypothesis, which means that the residuals are not normally distributed.

Looking at the time series plot, the values are fluctuating along the constant mean of zero and are not showing any trend. Looking at the histogram and the qqplot, we see that the residuals are normally distributed. But as per the Shapiro-Wilk test, this normality is not significant. Both ACF and PACF plots demonstrate the behaviour of white noise. And, when we finally look at the Ljung Box test output, we can see that we do have few points below the confidence interval, which means there is significant auto-correlation present in the residuals.

All these points do not support this model to be an optimal one.

So, from the above analysis, we have 2 models which are optimal, viz., SARIMA(0,0,1)x(0,1,1)_12 and SARIMA(0,0,2)x(0,1,1)_12. Now, we had seen that in model SARIMA(0,0,2)x(0,1,1)_12, MA(2) component was not significant. Also, as per the Principle of Parsimony, we go for the smaller model.

So, we will be choosing SARIMA(0,0,1)x(0,1,1)_12 model as the most optimal model and using this to forecast the average Melbourne temperature.

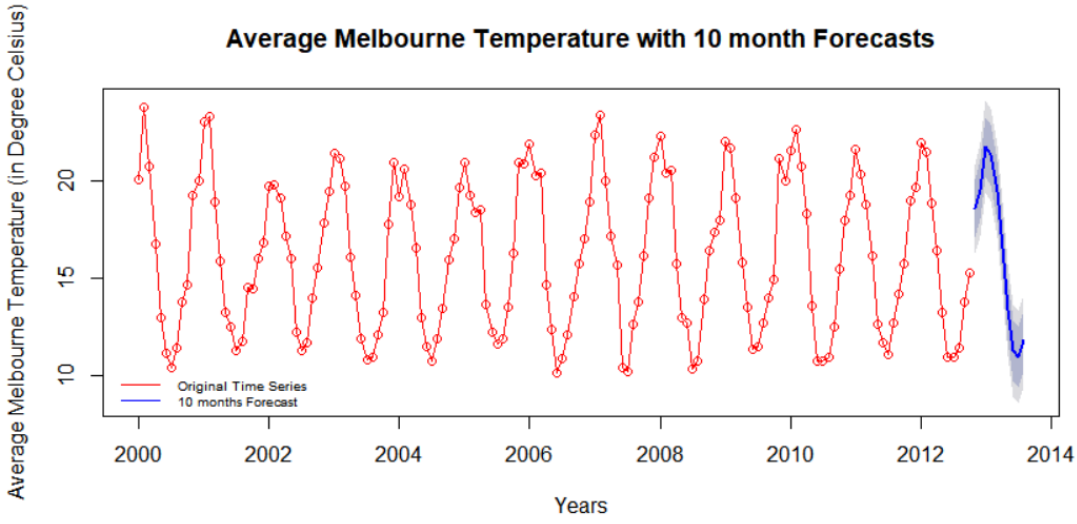
7. Forecasting

Having finalized SARIMA(0,0,1)x(0,1,1)_12 model, we will use this model to forecast the average Melbourne temperature for next 10 months.

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
Nov 2012	18.59376	17.106304	20.08121	16.318894	20.86862
Dec 2012	19.57953	18.023496	21.13556	17.199782	21.95927
Jan 2013	21.75643	20.200541	23.31232	19.376904	24.13595
Feb 2013	21.37962	19.823731	22.93551	19.000094	23.75914
Mar 2013	19.45523	17.899343	21.01112	17.075706	21.83475
Apr 2013	16.56733	15.011445	18.12322	14.187808	18.94686
May 2013	13.35558	11.799690	14.91146	10.976053	15.73510
Jun 2013	11.28396	9.728075	12.83985	8.904438	13.66349
Jul 2013	10.89709	9.341202	12.45298	8.517565	13.27661
Aug 2013	11.79562	10.239731	13.35151	9.416094	14.17514

Output 30: 10 months forecast for average Melbourne temperatures
(refer Appendix 30 for R code for this table)

Above table displays the forecasted values for next 10 months. We can also see the 80% and 95% confidence interval for this forecast. This result would be more intuitive once we plot it along with the original time series.



Output 31: Time series plot of 10 months forecast for average Melbourne temperatures
(refer Appendix 31 for R code for this plot)

From the above output, our model captured the seasonality very well. It is also doing a fair job in forecasting future values which follow the pattern of the original time series.

8. Conclusion

1. We were able to determine which modelling technique was to be used. As there is no specific statistic test which tells us about the model to choose out of stochastic or deterministic models, we chose stochastic model as it is better known to capture the seasonality and randomness for any time series.
2. Next, we were able to successfully handle the seasonality and define the seasonal parameters P, D and Q. After this, we were also able to model the non-seasonal part.
3. After modelling the seasonal and non-seasonal part, we ended up with few possible models. Using the parameter estimation and residual analysis, we were able to find out the most optimal model.
4. Using this model, we were successfully able to forecast the average Melbourne temperatures for next 10 months.
5. The forecast followed the pattern of the original time series well and also displayed 80% and 95% confidence interval for the predicted temperature values.

9. Appendix

A. R code of residual.analysis()

```
residual.analysis <- function(model, std = TRUE, start = 2, class =
c("ARIMA","GARCH","ARMA-GARCH")[1]){
  # If you have an output from arima() function use class = "ARIMA"
  # If you have an output from garch() function use class = "GARCH".
  # 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)
  library(FitAR)
  if (class == "ARIMA"){
    if (std == TRUE){
      res.model = rstandard(model)
    }else{
      res.model = rstandard(model)
    }
  }else if (class == "GARCH"){
    res.model = model$residuals[start:model$n.used]
  }else if (class == "ARMA-GARCH"){
      res.model = model@fit$residuals
  }else {
    stop("The argument 'class' must be either 'ARIMA' or 'GARCH' ")
  }
  par(mfrow=c(3,2), las = 1)
  plot(res.model,type='o', col = "red",
       ylab='Standardised residuals',
       main="Time series plot of standardised residuals")
  abline(h=0, col = "blue")
  hist(res.model, main="Histogram of standardised residuals")
  acf(res.model, main="ACF of standardised residuals", lag.max = 36)
  pacf(res.model, main="PACF of standardised residuals", lag.max = 36)
  qqnorm(res.model, main="QQ plot of standardised residuals", col = "red")
  qqline(res.model, col = "blue")
  print(shapiro.test(res.model))
  k=0
  LBQPlot(res.model, length(model$residuals)-1, StartLag = k + 1, k = 0, SquaredQ
= FALSE)
}
```

B. R code for importing packages and user-defined function sort.score()

```
library(dplyr)
library(tidyselect)
library(knitr)
library(timeSeries)
library(fUnitRoots)
library(TSA)
library(lmtest)
library(forecast)
library(tseries)
library(FitAR)
library(FSAdata)

source('sort.score.R')
```

1. **R Code for Output 1: Sample of records with calculated average temperature**

```
#import data
melbTemp <- read.csv("TempWorld2.csv")

#compute mean
melbTemp$MelbourneAvg <- (melbTemp$Melbournemax + melbTemp$MelbourneMin)/2

#display computed mean
head(melbTemp[, c('Date', 'MelbourneMin', 'Melbournemax', 'MelbourneAvg')])
```

2. **R Code for Output 2: Time series data for Average Melbourne temperatures**

```
#convert to time series
tempAvg <- ts(as.vector(melbTemp$MelbourneAvg), start=2000, frequency = 12)

#display time series data
tempAvg
```

3. **R Code for Output 3: Time series plot for Average Melbourne temperatures**

```
#set the frame
par(mfrow = c(1,1))

#plot time series
plot(tempAvg, type="o", col = "red", lwd = 1.5,
     xlab = "Years",
     ylab = "Average Melbourne Temperature (in Degree Celsius)",
     main = "Time Series plot of Average Melbourne Temperature (with months as
points)")
points(y = tempAvg, x = time(tempAvg), pch = as.vector(season(tempAvg)))
```

4. **R Code for Output 4: ACF and PACF plot for Average Melbourne temperatures**

```
#set the frame
par(mfrow = c(1,2))

#plot ACF
acf(tempAvg, main = "ACF: Average Melbourne Temperature", lag.max = 36)

#plot PACF
pacf(tempAvg, main = "PACF: Average Melbourne Temperature", lag.max = 36)
```

5. **R Code for Output 5: Scatter plot for Average Melbourne temperatures**

```
#set the frame
par(mfrow = c(1,1))

#plot scatter plot
plot(y = tempAvg, x = zlag(tempAvg), col = "red",
     xlab = 'Lagged Average Melbourne Temperature (in Degree Celsius)',
     ylab = 'Average Melbourne Temperature (in Degree Celsius)',
     main = 'Scatter plot of Average Melbourne Temperature (in Degree Celsius)')

#Reading the average temperature data into y
y = tempAvg
```

```
#Generating first lag of the average temperature series
x = zlag(tempAvg)

#Creating an index to get rid of the first NA value in x
index = 2:length(x)

#Calculating the correlation between numerical values in x and y
cor(y[index],x[index])
```

6. **R Code for Output 6: Summary for fitted deterministic seasonal model**

```
#identify season
month. = season(tempAvg)

#fit the model
seasonal_model = lm( tempAvg ~ month.-1)

#print model summary
summary(seasonal_model)
```

7. **R Code for Output 7: Fitted deterministic seasonal model over Average Melbourne Temperature Time Series**

```
#set the frame
par(mfrow = c(1,1))

#fit the seasonal model
plot(ts(fitted(seasonal_model), freq=12, start=2000),
        xlab = 'Years',
        ylab = 'Average Melbourne Temperature (in Degree Celsius)',
        type = 'l',
        ylim = range(c(fitted(seasonal_model), tempAvg)),
        main = "Fitted Seasonal Model to Average Melbourne Temperature Time
Series",
     col = "blue",  lwd = 1.5)
points(tempAvg)
```

8. **R Code for Output 8: Time Series plot of residuals of Average Melbourne Temperature**

```
#perform 1st differencing
m1_temp = arima(tempAvg, order=c(0,0,0), seasonal=list(order=c(0,1,0), period =
12))

#extract residuals
res_m1_temp = residuals(m1_temp);

#set the frame
par(mfrow = c(1,1))

#plot the time series for residuals
plot(res_m1_temp, type="o", col = "red", lwd = 1.5,
     xlab = 'Years',
     ylab = 'Residuals',
     main = "Time Series plot for residuals of Average Melbourne Temperature")
```

9. **R Code for Output 9: ACF and PACF plot for Residuals of Average Melbourne temperatures**

```
#set the frame
par(mfrow = c(1,2))

#plot ACF
acf(res_m1_temp, lag.max = 36, main = "ACF: Residuals of Average Melbourne
Temperature")

#plot PACF
pacf(res_m1_temp, lag.max = 36, main = "PACF: Residuals of Average Melbourne
Temperature")
```

10. **R Code for Output 10: Time series plot of average Melbourne temperature after removing seasonality**

```
#apply seasonal (0, 1, 1) model to capture seasonality
m2_temp = arima(tempAvg, order=c(0,0,0), seasonal=list(order=c(0,1,1), period =
12))

#capture remaining components other than seasonality
res_m2_temp = residuals(m2_temp);

#set the frame
par(mfrow=c(1,1))

#plot the time series for remaining components
plot(res_m2_temp, type="o", col = "red", lwd = 1.5,
     xlab = 'Years',
     ylab = 'Residuals',
     main = "Time series plot of Average Melbourne Temperature after Removing
Seasonality")
```

11. **R Code for Output 11: ACF and PACF plot for average Melbourne temperature after removing seasonality**

```
#set the frame
par(mfrow=c(1,2))

#plot ACF
acf(res_m2_temp, lag.max = 36, main = "ACF: After Removing Seasonality")

#plot PACF
pacf(res_m2_temp, lag.max = 36, main = "PACF: After Removing Seasonality")
```

12. **R Code for Output 12: Results of Augmented Dickey-Fuller Test**

```
#ADF Test
adf.test(res_m2_temp)
```

13. **R Code for Output 13: ACF and PACF plot for average Melbourne temperature after removing seasonality**

```
#set the frame
par(mfrow=c(1,2))

#plot ACF
acf(res_m2_temp, lag.max = 36, main = "ACF: After Removing Seasonality")

#plot PACF
pacf(res_m2_temp, lag.max = 36, main = "PACF: After Removing Seasonality")
```

14. **R Code for Output 14: EACF matrix for average Melbourne temperature after removing seasonality**

```
#compute EACF
eacf(res_m2_temp)
```

15. **R Code for Output 15: Bayesian Information Criterion table for average Melbourne temperature after removing seasonality**

```
#set the frame
par(mfrow=c(1,1))

#compute the BIC table
res = armasubsets(y=res_m2_temp, nar = 10, nma = 10, y.name = 'test',
ar.method='ols')

#plot the BIC table
plot(res)
```

16. **R Code for Output 16: Coefficients test for model SARIMA(1,0,1)x(0,1,1)_12**

```
#fit the model using CSS
model_101_css = arima(tempAvg, order=c(1,0,1),seasonal=list(order=c(0,1,1),
period=12),method = "CSS")

#perform co-efficients test
coeftest(model_101_css)

#fit the model using ML
model_101_ml = arima(tempAvg, order=c(1,0,1),seasonal=list(order=c(0,1,1),
period=12),method = "ML")

#perform co-efficients test
coeftest(model_101_ml)
```

17. **R Code for Output 17: Coefficients test for model SARIMA(0,0,1)x(0,1,1)_12**

```
#fit the model using CSS
model_001_css = arima(tempAvg,order=c(0,0,1),seasonal=list(order=c(0,1,1),
period=12),method = "CSS")

#perform co-efficients test
coeftest(model_001_css)

#fit the model using ML
```

```
model_001_ml = arima(tempAvg,order=c(0,0,1),seasonal=list(order=c(0,1,1),
period=12),method = "ML")

#perform co-efficients test
coeftest(model_001_ml)
```

18. R Code for Output 18: Coefficients test for model SARIMA(0,0,2)x(0,1,1)_12

```
#fit the model using CSS
model_002_css = arima(tempAvg,order=c(0,0,2),seasonal=list(order=c(0,1,1),
period=12),method = "CSS")

#perform co-efficients test
coeftest(model_002_css)

#fit the model using ML
model_002_ml = arima(tempAvg,order=c(0,0,2),seasonal=list(order=c(0,1,1),
period=12),method = "ML")

#perform co-efficients test
coeftest(model_002_ml)
```

19. R Code for Output 19: Coefficients test for model SARIMA(1,0,2)x(0,1,1)_12

```
#fit the model using CSS
model_102_css = arima(tempAvg,order=c(1,0,2),seasonal=list(order=c(0,1,1),
period=12),method = "CSS")

#perform co-efficients test
coeftest(model_102_css)

#fit the model using ML
model_102_ml = arima(tempAvg,order=c(1,0,2),seasonal=list(order=c(0,1,1),
period=12),method = "ML")

#perform co-efficients test
coeftest(model_102_ml)
```

20. R Code for Output 20: Coefficients test for model SARIMA(1,0,4)x(0,1,1)_12

```
#fit the model using CSS
model_104_css = arima(tempAvg,order=c(1,0,4),seasonal=list(order=c(0,1,1),
period=12),method = "CSS")

#perform co-efficients test
coeftest(model_104_css)

#fit the model using ML
model_104_ml = arima(tempAvg,order=c(1,0,4),seasonal=list(order=c(0,1,1),
period=12),method = "ML")

#perform co-efficients test
coeftest(model_104_ml)
```

21. R Code for Output 21: Coefficients test for model SARIMA(1,0,5)x(0,1,1)_12

```
#fit the model using CSS
model_105_css = arima(tempAvg,order=c(1,0,5),seasonal=list(order=c(0,1,1),
period=12),method = "CSS")

#perform co-efficients test
coeftest(model_105_css)

#fit the model using ML
model_105_ml = arima(tempAvg,order=c(1,0,5),seasonal=list(order=c(0,1,1),
period=12),method = "ML")

#perform co-efficients test
coeftest(model_105_ml)
```

22. R Code for Output 22: Coefficients test for model SARIMA(5,0,2)x(0,1,1)_12

```
#fit the model using CSS
model_502_css = arima(tempAvg,order=c(5,0,2),seasonal=list(order=c(0,1,1),
period=12),method = "CSS")

#perform co-efficients test
coeftest(model_502_css)

#fit the model using ML
model_502_ml = arima(tempAvg,order=c(5,0,2),seasonal=list(order=c(0,1,1),
period=12),method = "ML")

#perform co-efficients test
coeftest(model_502_ml)
```

23. R Code for Output 23: Coefficients test for model SARIMA(5,0,4)x(0,1,1)_12

```
#fit the model using CSS
model_504_css = arima(tempAvg,order=c(5,0,4),seasonal=list(order=c(0,1,1),
period=12),method = "CSS")

#perform co-efficients test
coeftest(model_504_css)

#fit the model using ML
model_504_ml = arima(tempAvg,order=c(5,0,4),seasonal=list(order=c(0,1,1),
period=12),method = "ML")

#perform co-efficients test
coeftest(model_504_ml)
```

24. R Code for Output 24: Coefficients test for model SARIMA(5,0,5)x(0,1,1)_12

```
#fit the model using CSS
model_505_css = arima(tempAvg,order=c(5,0,5),seasonal=list(order=c(0,1,1),
period=12),method = "CSS")

#perform co-efficients test
coeftest(model_505_css)

#fit the model using ML
```

```
model_505_ml = arima(tempAvg,order=c(5,0,5),seasonal=list(order=c(0,1,1),
period=12),method = "ML")

#perform co-efficients test
coeftest(model_505_ml)
```

25. R Code for Output 25: Models sorted as per AIC scores

```
#sort based on AIC values
aic = sort.score(AIC(model_101_ml, model_001_ml, model_002_ml, model_102_ml,
model_104_ml, model_105_ml,model_502_ml,model_504_ml,model_505_ml), score =
"aic")

#display the sorted models
aic
```

26. R Code for Output 26: Models sorted as per BIC scores

```
#sort based on BIC values
bic = sort.score(BIC(model_101_ml, model_001_ml, model_002_ml, model_102_ml,
model_104_ml, model_105_ml,model_502_ml,model_504_ml,model_505_ml), score =
"bic")

#display the sorted models
bic
```

27. R Code for Output 27: Residual Analysis for model SARIMA(0,0,1)x(0,1,1)_12

```
#residual analysis for model SARIMA(0,0,1)x(0,1,1)_12
#Refer Appendix A for function definition of residual.analysis()
residual.analysis(model = model_001_css, class = "ARIMA")
```

28. R Code for Output 28: Residual Analysis for model SARIMA(0,0,2)x(0,1,1)_12

```
#residual analysis for model SARIMA(0,0,2)x(0,1,1)_12
#Refer Appendix A for function definition of residual.analysis()
residual.analysis(model = model_002_css, class = "ARIMA")
```

29. R Code for Output 29: Residual Analysis for model SARIMA(1,0,1)x(0,1,1)_12

```
#residual analysis for model SARIMA(1,0,1)x(0,1,1)_12
#Refer Appendix A for function definition of residual.analysis()
residual.analysis(model = model_101_css, class = "ARIMA")
```

30. R Code for Output 30: 10 months forecast for average Melbourne temperatures

```
#fitting the model
fit = Arima(tempAvg, order=c(0,0,1), seasonal = list(order=c(0,1,1), period =
12), method = "CSS")

#predicting the next 10 values
temp_forecast = forecast(fit, h = 10)

#presenting the forecast in a table form
temp_forecast
```

31. R Code for Output 31: Time series plot of 10 months forecast for average Melbourne temperatures

```
#set the frame
par(mfrow=c(1,1))

#plotting the forecast
plot(temp_forecast, type="o", lwd = 1.5, col = "red",
     xlab = "Years",
     ylab = "Average Melbourne Temperature (in Degree Celsius)",
     main = "Average Melbourne Temperature with 10-month Forecasts")

#displaying the legend
legend("bottomleft", lty = 1, bty = "n" , col = c("red", "blue"), cex = 0.6,
      legend = c("Original Time Series", "10 months Forecast"))
```

10. References

1. MATH1318 Time Series Analysis notes prepared by Dr. Haydar Demirha and Zeynep Kalaylioglu

2. Canvas, n.d., Modules, viewed on 16th May 2020, https://rmit.instructure.com/courses/67182/modules

3. Time Series Datasets (2013) by M. Forster and Rachel Passmore at CensusAtSchool, NZ, viewed on 15th May 2020, https://new.censusatschool.org.nz/wp-content/uploads/2013/02/TimeSeriesDatasets_130207.zip