Forecasting Project

Maitreya Milind Kadam

2025-09-26

```
knitr::opts_chunk$set(echo = TRUE)
library(TSA)
library(forecast)
library(x12)
library(tseries)
library(dLagM)
library(readr)
library(stats)
library(car)
library(dynlm)
library(lmtest)
library(tidyr)
library(dplyr)
library(knitr)
```

```
#UDF Function for acf,pacf
acf_pacf=function(x){
  var=deparse(substitute(x))
  par(mfrow=c(1,2))
  acf(x,main=paste('ACF plot for',var),lag.max=70)
  pacf(x,main=paste('PACF plot for',var),lag.max=70)
}
```

Introduction

This project consists of two time series analysis tasks. The 1st task focuses on analyzing and forecasting the monthly average horizontal solar radiation using appropriate time series techniques. The main task is to give the best 2 years forecasts using appropriate time series regression methods in terms of MASE. The 2nd task focuses on the relationship between the quaterly Residential Property Price Index (PPI) in Melbourne and the quaterly population change in Victoria over the period. The analysis focuses on determining whether the correlation between these two factors reflects a genuine association or is spurious. This analysis involves applying correlation analysis and stationarity tests to check the validity of the relation.

Task 1

Data Description

I have loaded the data1.csv which contains data about monthly average horizontal solar radiation and monthly precipitation. After that, I have converted the dataframe into time series object for visualisation and further analysis.

```
solar_ppt=read.csv('data1.csv')
solar=ts(solar_ppt$solar,start=c(1960,1),frequency = 12)
head(solar)
```

```
##            Jan       Feb       Mar       Apr       May       Jun
## 1960  5.051729  6.415832 10.847920 16.930264 24.030797 26.298202
```

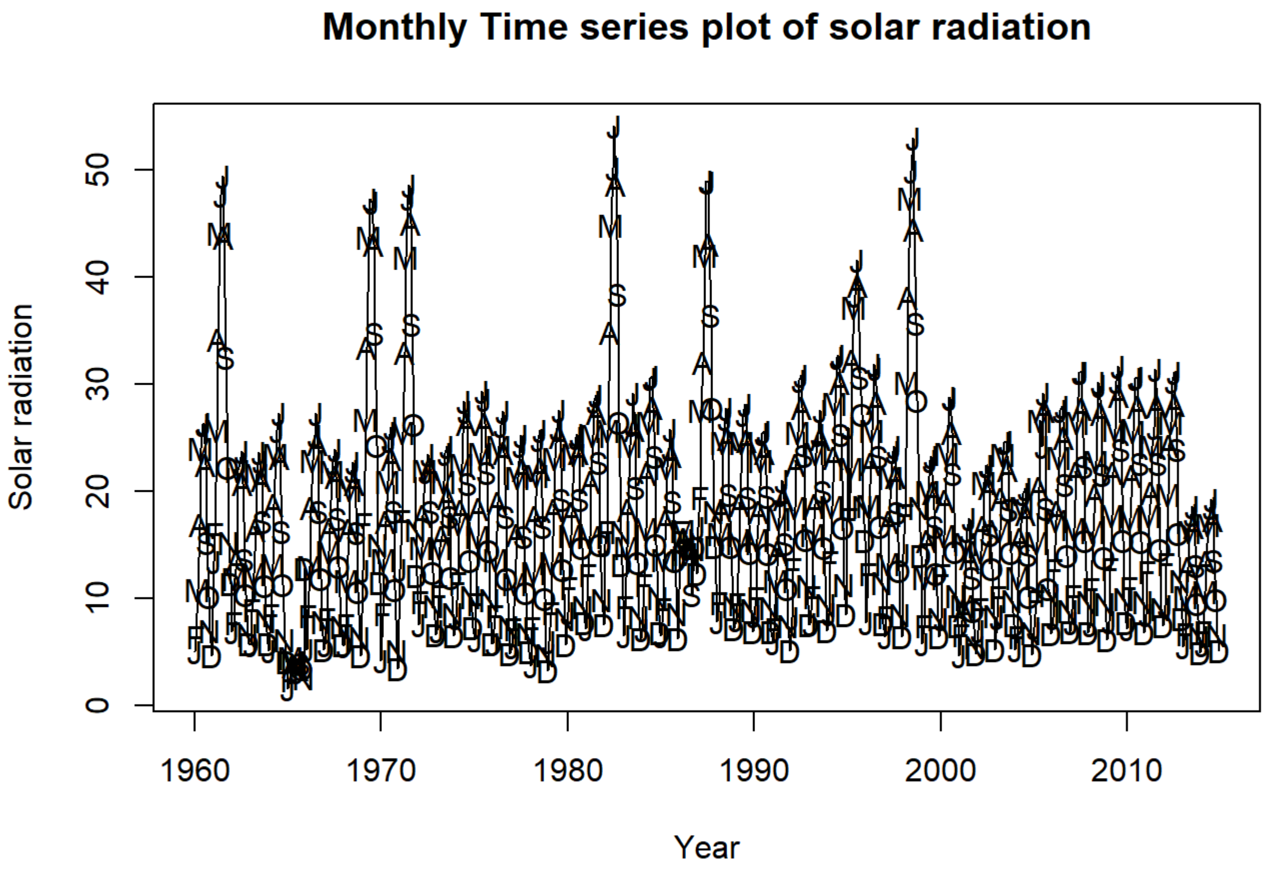
```
ppt=ts(solar_ppt$ppt,start=c(1960,1),frequency = 12)
head(ppt)
```

```
##        Jan   Feb   Mar   Apr   May   Jun
## 1960 1.333 0.921 0.947 0.615 0.544 0.703
```

```
#loading the dataset that are to be used for solar radiation forecasts
solar_fore=read.csv('data.x.csv')
```

Data Visualization

```
plot(solar, main = "Monthly Time series plot of solar radiation", ylab = "Solar radiation", x
lab = "Year")
points(y=solar,x=time(solar),pch=as.vector(season(solar)))
```

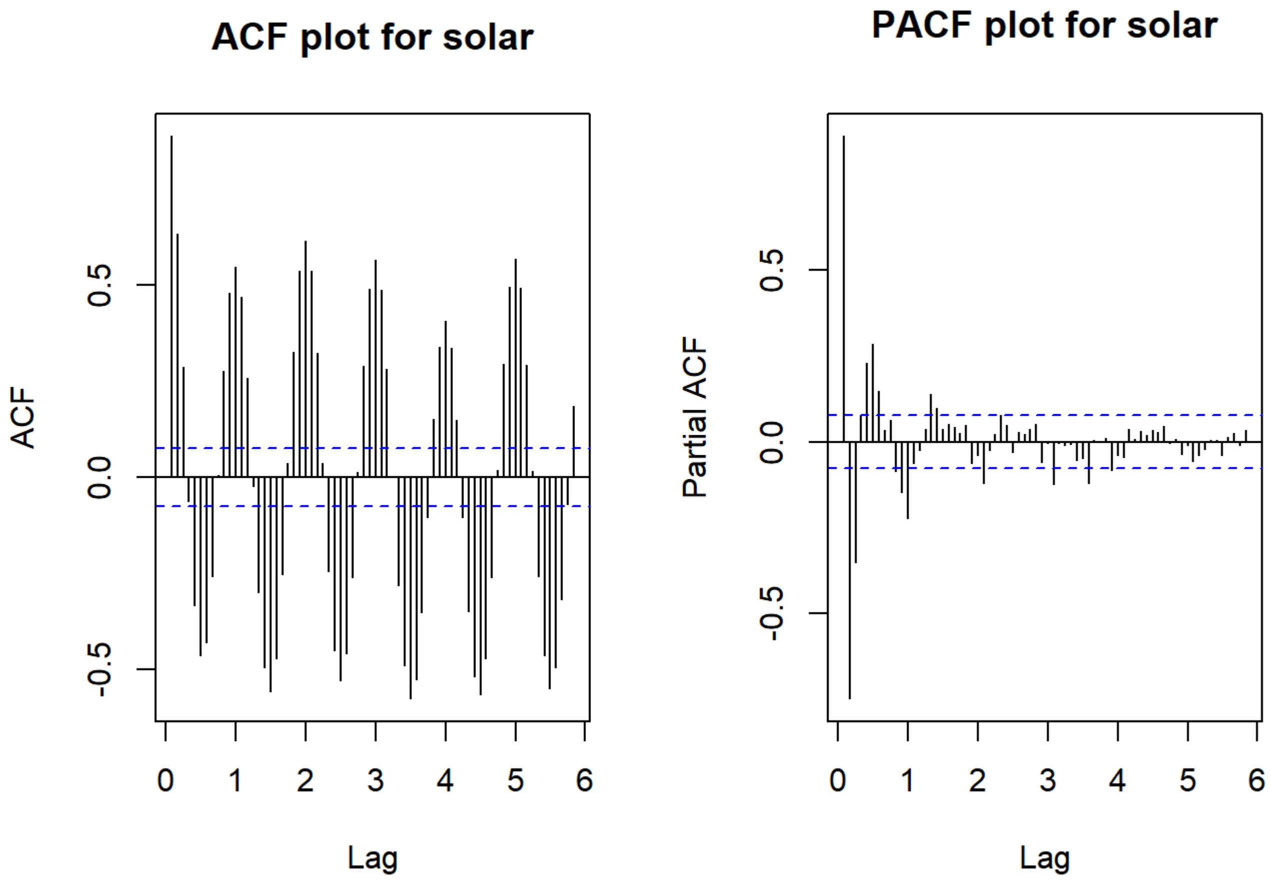


Based on the above figure, the following points can be inferred:

1. Trend: There appears to be no clear long-term upward or downward trend. The time series for the solar radiation appears to fluctuate around a stable mean.
2. Seasonality: The time series displays a strong seasonal pattern which includes peaks which are likely annual.

3. Variance: Due to the presence of seasonality changing variance is not easy to be noticed.
4. Change Points: The first change point appears to be around 1965 and the second change point appears to be around 1987.

```
#calls the udf function which is defined at the start.
acf_pacf(solar)
```



The ACF and PACF plot in the above figure clearly illustrates strong seasonal pattern which is indicated by a repeating annual pattern. However, the plots don’t suggest any trend.

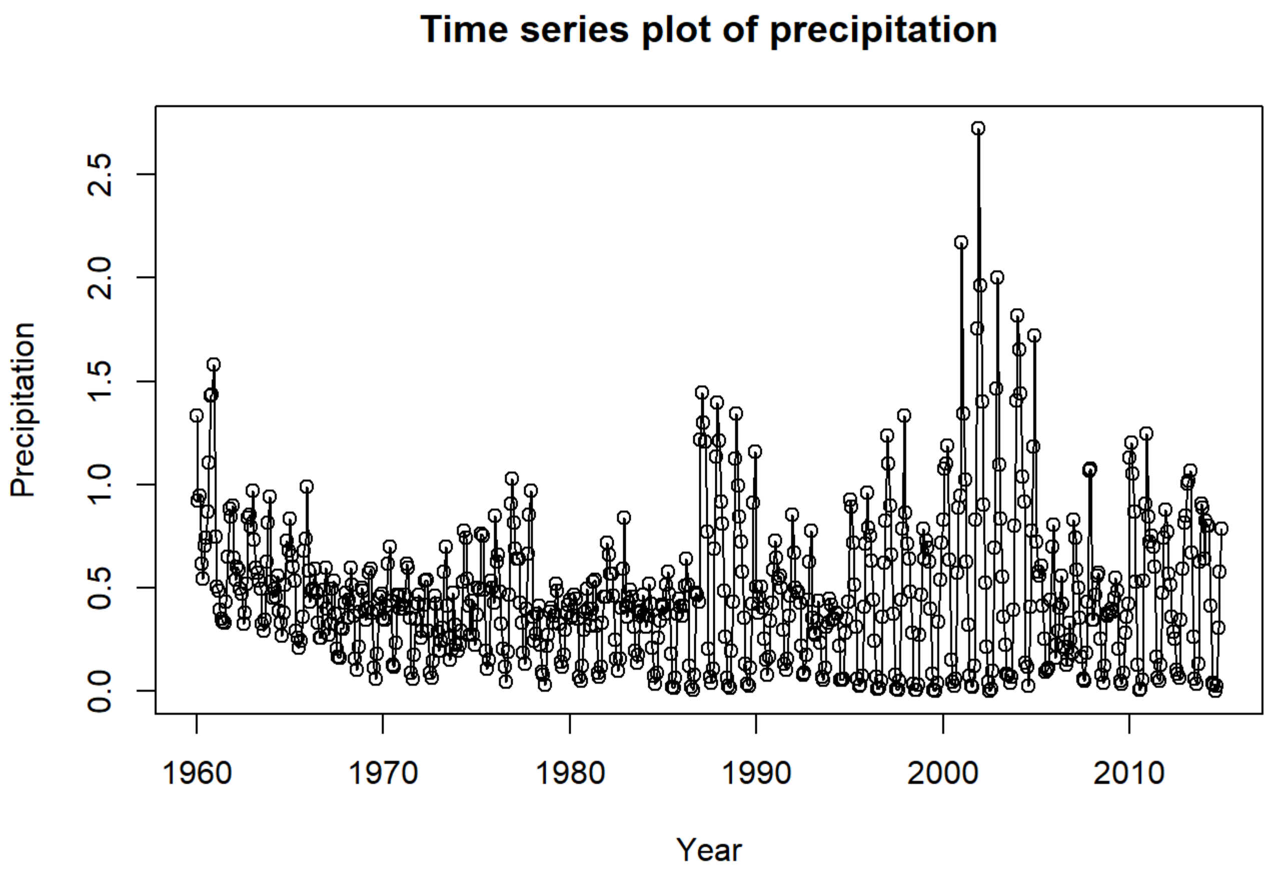
```
adf.test(solar)
```

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

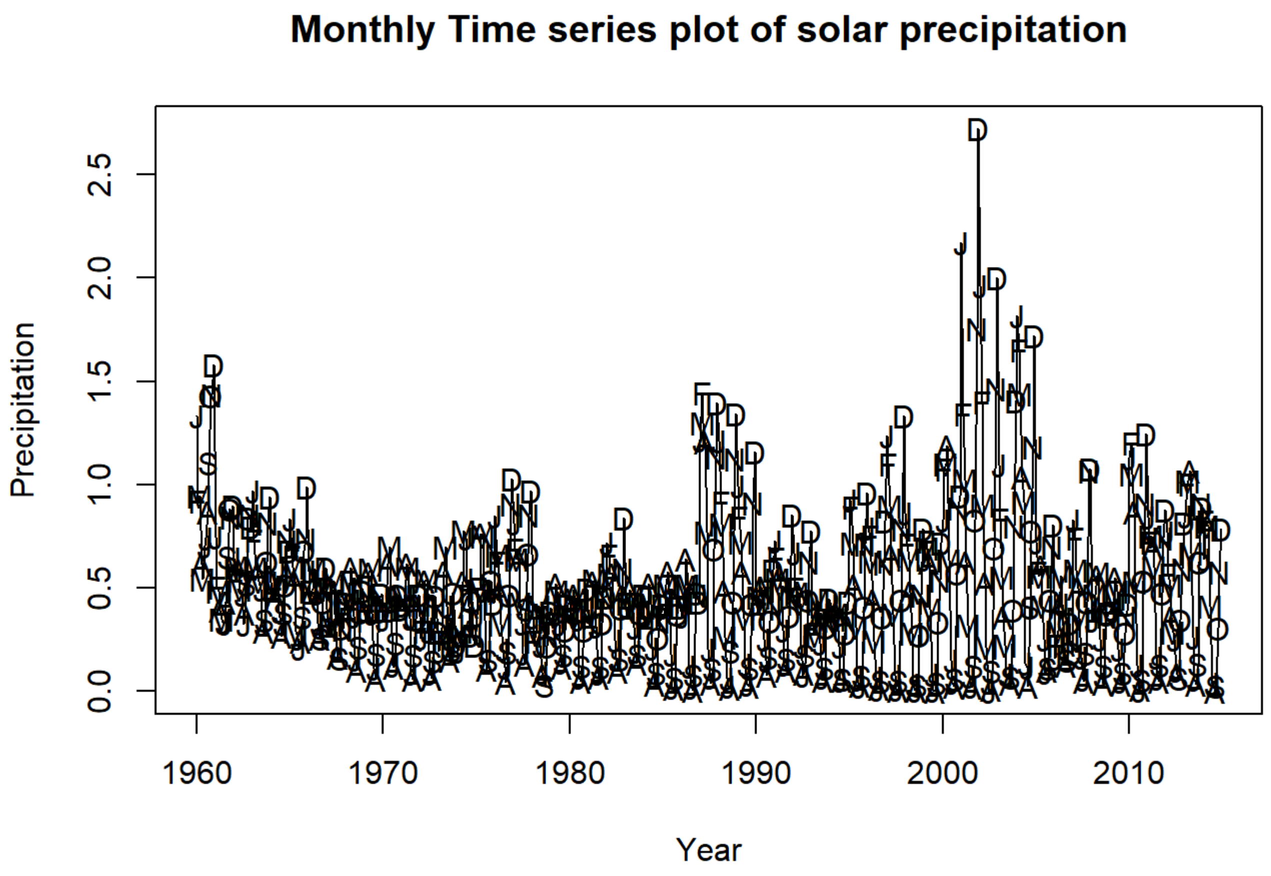
```
##
##  Augmented Dickey-Fuller Test
##
## data:  solar
## Dickey-Fuller = -4.7661, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
```

Based on the ADF test, the p-value of 0.01 is less than the threshold of 0.05 which rejects the null hypothesis of non-stationarity and therefore we can conclude that the solar radiation series is stationary.

```
plot(ppt, main = "Time series plot of precipitation", ylab = "Precipitation", xlab = "Year",t
ype='o')
```



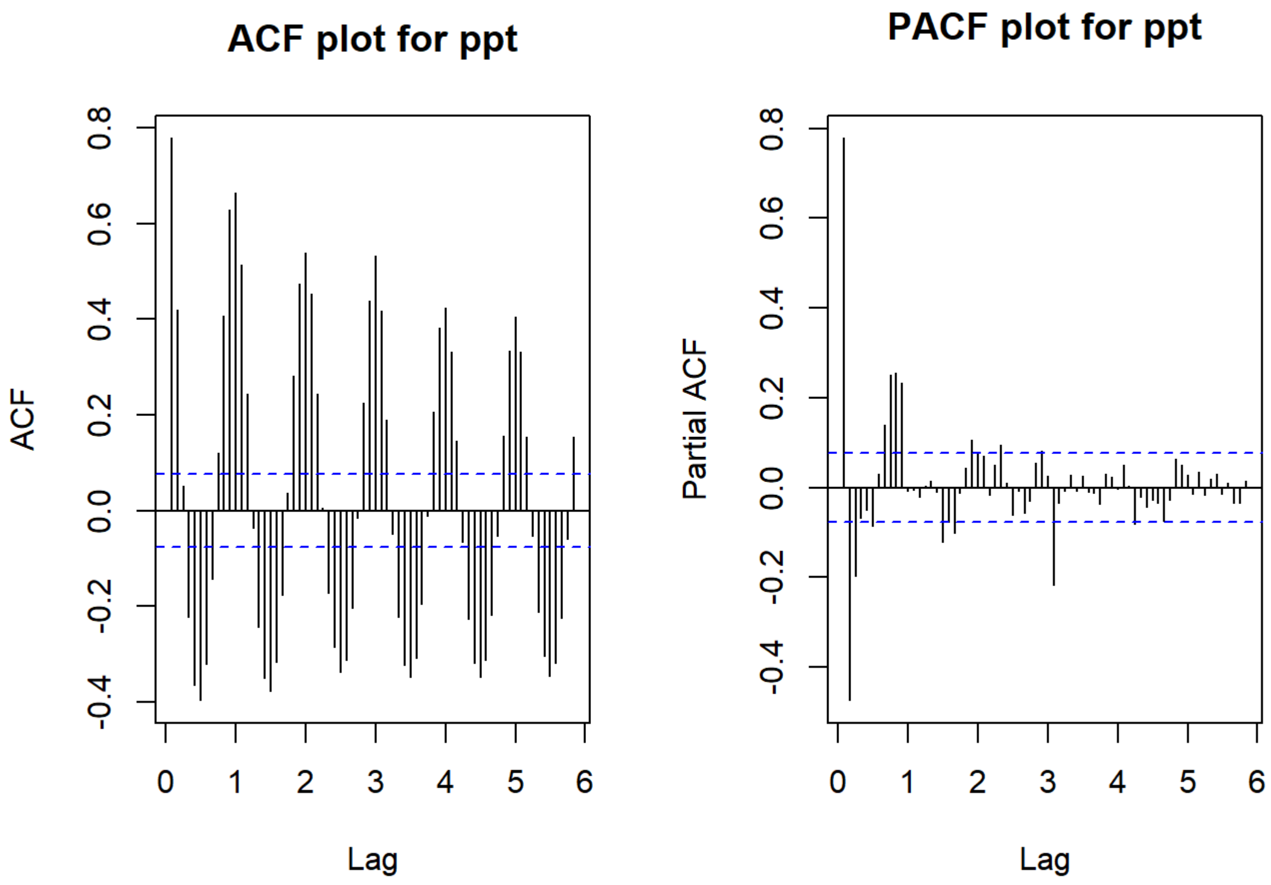
```
plot(ppt, main = "Monthly Time series plot of solar precipitation", ylab = "Precipitation", xlab = "Year")
points(y=ppt,x=time(ppt),pch=as.vector(season(ppt)))
```



Based on the above figure, the following points can be inferred:

1. Trend: The time series plot of precipitation indicates a gradual downward trend which is visible in the start of the series.
2. Seasonality: Lower precipitation values are observed in July, August while higher precipitation values are observed during December-January. This indicates that there is a clear seasonality as the pattern changes overtime.
3. Variance: Due to the presence of seasonality changing variance is not easy to be noticed.
4. Change Point: There are no change points in the above plot.

```
acf_pacf(ppt)
```



From the above ACF plot it can be observed that there is a presence of seasonal pattern which is indicated by the repetitive nature of the lags. In addition to that, there is also a possible existence of trend which is illustrated by the decaying pattern of seasonal lags.

```
adf.test(ppt)
```

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

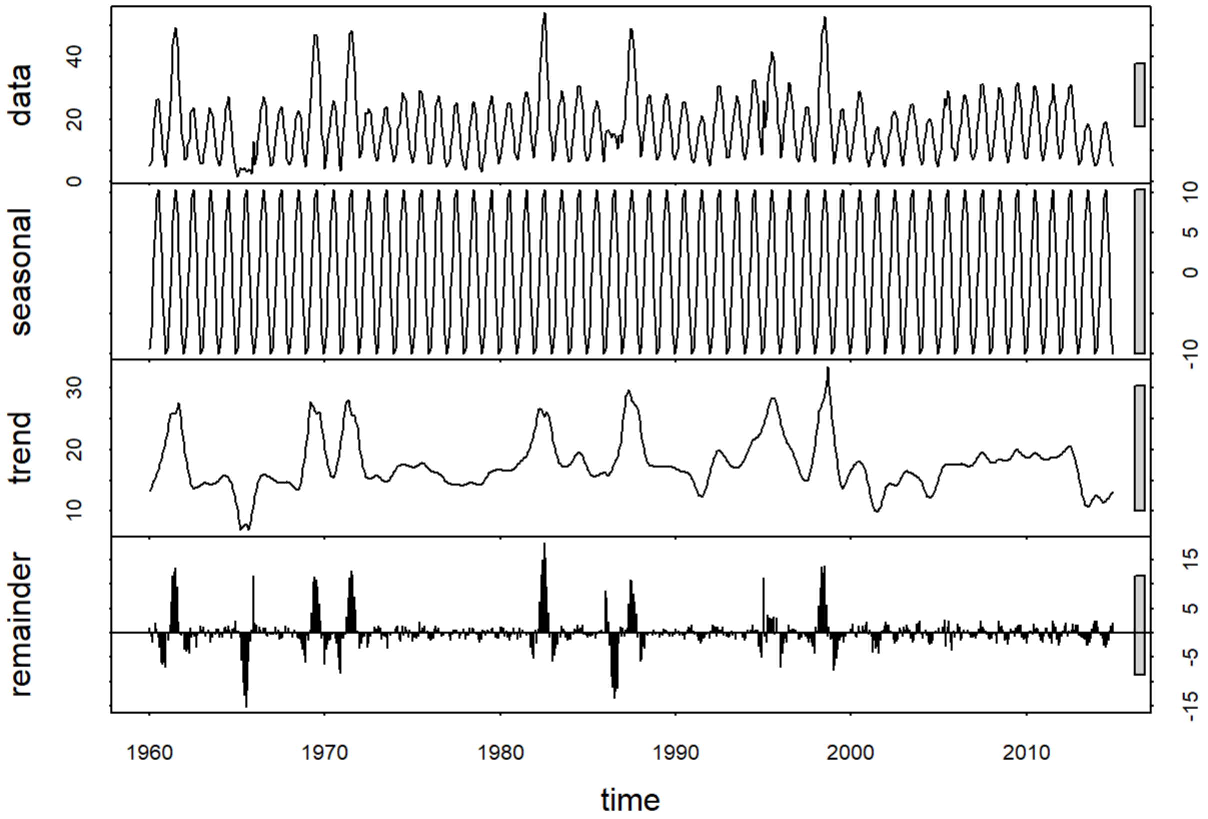
```
##
##  Augmented Dickey-Fuller Test
##
## data:  ppt
## Dickey-Fuller = -6.7438, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
```

The above results from the ADF tests helps in concluding that the precipitation series is stationary. The p-value of 0.01 which is less than the 0.05 threshold rejects the null hypothesis of non-stationarity.

Components of a Time Series Data

STL Decomposition for Solar radiation

```
stl_solar=stl(solar,t.window=15,s.window='periodic',robust=TRUE)
plot(stl_solar)
```

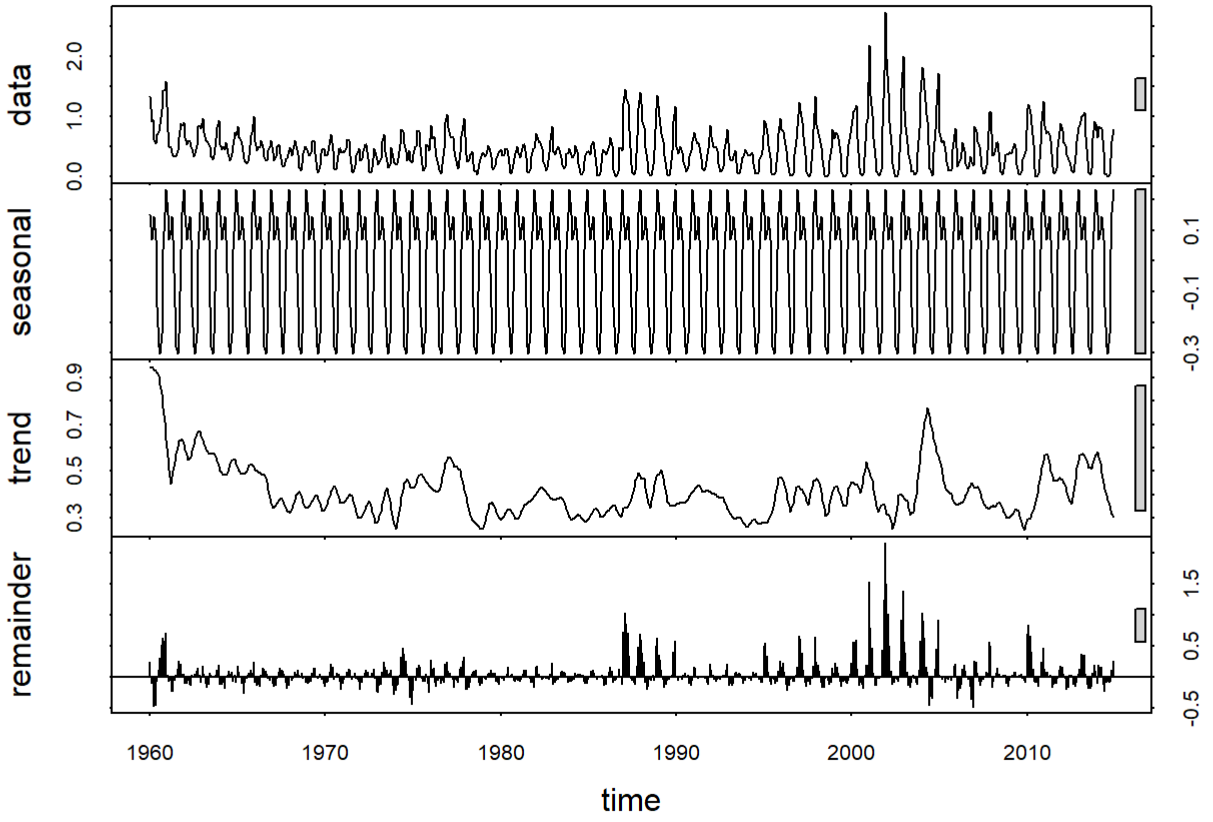


From the above plot, it can be inferrred that:

- The trend component for the solar radiation remains relatively stable over the time period with no presence of upward or downward movement.
- In terms of the change points, there are 2 change points that appear around the year 1965 and 1987.

STL Decomposition for Precipitation

```
stl_ppt=stl(ppt,t.window=15,s.window='periodic',robust=TRUE)
plot(stl_ppt)
```



From the above plot, it can be inferrred that:

- In terms of trend, the precipitation time series shows a gradual decline during the early part of the observed period.
- In terms of change points, there don’t seem to be any significant change points over the observed period for the time series.

Modelling with Time Series Regression Methods

```
cor(solar_ppt)
```

```
##            solar        ppt
## solar  1.0000000 -0.4540277
## ppt   -0.4540277  1.0000000
```

The negative correlation i.e. -0.45 between solar radiation and precipitation indicates that higher rainfall relates with lower solar radiation levels.

Finite Distributed Lag Model

```
for (j in 1:10){
  mod_1=dlm(x=solar_ppt$ppt,y=solar_ppt$solar,q=j)
  cat('q=',j,'AIC=',AIC(mod_1$model),'BIC=',BIC(mod_1$model),'MASE=',MASE(mod_1)$MASE,'\n')
}
```

```
## q= 1 AIC= 4728.713 BIC= 4746.676 MASE= 1.688457
## q= 2 AIC= 4712.649 BIC= 4735.095 MASE= 1.675967
## q= 3 AIC= 4688.551 BIC= 4715.478 MASE= 1.662703
## q= 4 AIC= 4663.6 BIC= 4695.003 MASE= 1.646357
## q= 5 AIC= 4644.622 BIC= 4680.499 MASE= 1.613848
## q= 6 AIC= 4637.489 BIC= 4677.837 MASE= 1.607532
## q= 7 AIC= 4632.716 BIC= 4677.532 MASE= 1.607042
## q= 8 AIC= 4625.986 BIC= 4675.267 MASE= 1.604806
## q= 9 AIC= 4615.084 BIC= 4668.827 MASE= 1.593121
## q= 10 AIC= 4602.658 BIC= 4660.858 MASE= 1.577996
```

We will proceed with the q value of 10 as it exhibits the smallest MASE value.

```
f_dlm=dlm(x=solar_ppt$ppt,y=solar_ppt$solar,q=10)
summary(f_dlm)
```

```
##
## Call:
## lm(formula = model.formula, data = design)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.9353  -5.4124  -0.7911   4.0184  30.8900
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.0105     1.0942  17.374  < 2e-16 ***
## x.t          -7.3843     1.8995  -3.887 0.000112 ***
## x.1          -0.4763     2.5395  -0.188 0.851288
## x.2          -0.1324     2.5734  -0.051 0.958980
## x.3           1.7902     2.5781   0.694 0.487691
## x.4           1.9686     2.5808   0.763 0.445877
## x.5           3.4928     2.5807   1.353 0.176402
## x.6           0.5243     2.5787   0.203 0.838943
## x.7           1.6762     2.5797   0.650 0.516088
## x.8           0.9282     2.5673   0.362 0.717817
## x.9           0.3754     2.5338   0.148 0.882272
## x.10         -5.3798     1.8760  -2.868 0.004272 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.256 on 638 degrees of freedom
## Multiple R-squared:  0.3081, Adjusted R-squared:  0.2962
## F-statistic: 25.82 on 11 and 638 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4602.658 4660.858
```

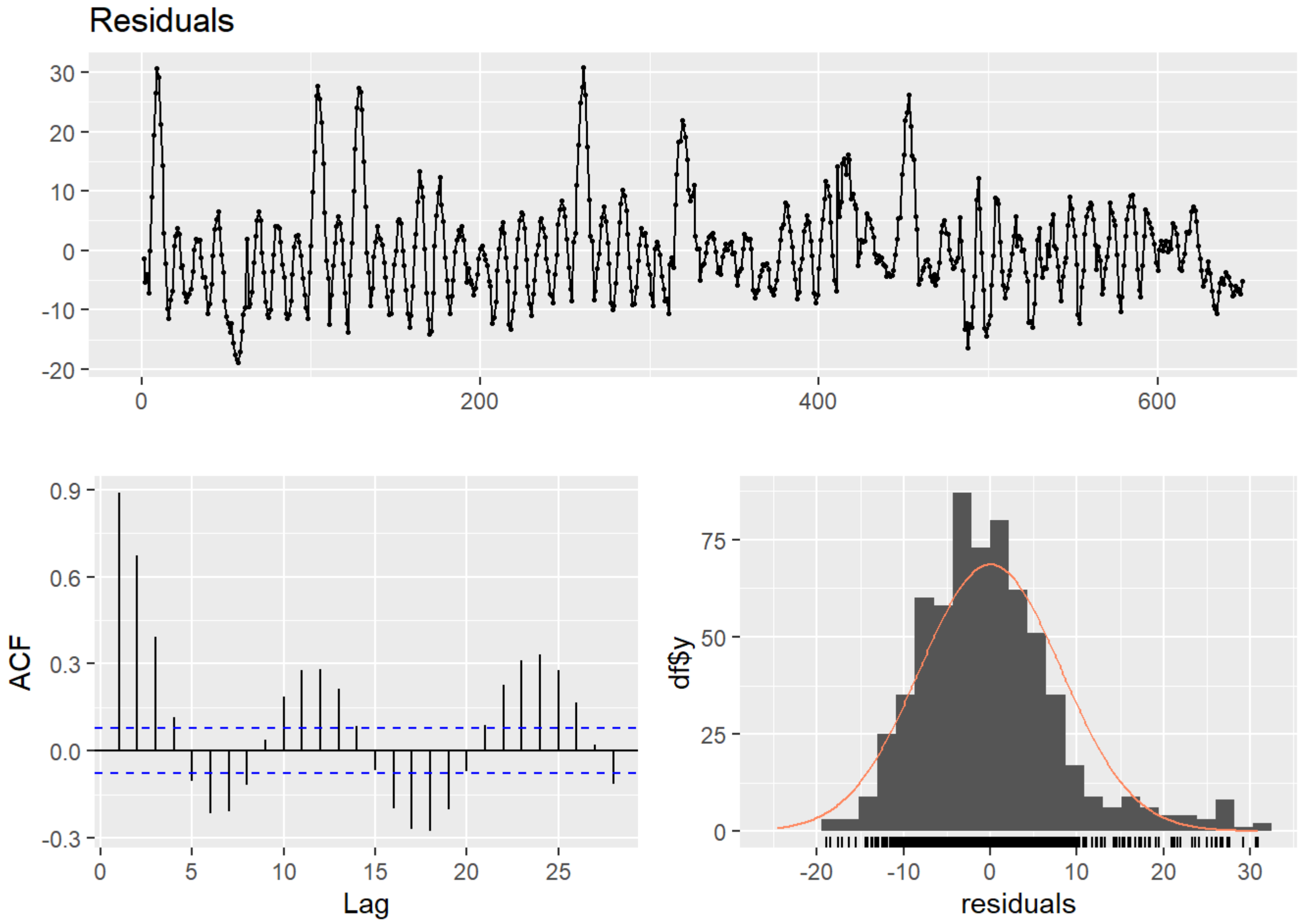
The Finite DLM model is statistically significant as the p-value is less than 0.05. The adjusted R-square value is 0.2962 which means that the model explains only 29.6% of the variation in the level of solar radiation. The model is not strong due to low explanatory power.

```
vif(f_dlm$model)
```

```
##      x.t      x.1      x.2      x.3      x.4      x.5      x.6      x.7 
## 4.244594 7.665259 7.910115 7.952633 7.957841 7.941836 7.911213 7.901898 
##      x.8      x.9     x.10 
## 7.847965 7.653512 4.228221
```

The Finite DLM model does not exhibit multicollinearity as all the VIF values are less than 10

```
checkresiduals(f_dlm$model)
```



```
## 
##  Breusch-Godfrey test for serial correlation of order up to 15
## 
## data:  Residuals
## LM test = 588.43, df = 15, p-value < 2.2e-16
```

The residual plot clearly shows that the residuals are not randomly distributed. The ACF plot along with the Breusch-Godfrey test has p-value less than 0.05 which confirms serial correlation and seasonality. Therefore, the finite DLM model with q=10 fails to capture the autocorrelation and seasonal patterns which are present in the data.

Polynomial Distributed Lag Model

```
q_val=1:5
k_val=1:3
for (i in q_val){
  for (j in k_val){
    if (j<=i){
      mod_2=polyDlm(x=solar_ppt$ppt,y=solar_ppt$solar,q=i,k=j,show.beta=TRUE)
      cat('q=',i,'k=',j,'AIC=',AIC(mod_2$model),'BIC=',BIC(mod_2$model),'MASE=',MASE(mod_2)$MASE,'\n')
    }
  }
}
```

```
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0   -15.80       1.54  -10.20 7.49e-23
## beta.1     4.11       1.54    2.68 7.61e-03
## q= 1 k= 1 AIC= 4728.713 BIC= 4746.676 MASE= 1.688457
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0   -12.60      0.957  -13.20 3.14e-35
## beta.1    -3.20      0.361   -8.88 6.53e-18
## beta.2     6.19      0.954    6.49 1.73e-10
## q= 2 k= 1 AIC= 4710.708 BIC= 4728.664 MASE= 1.67675
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0   -12.90       1.76   -7.37 5.32e-13
## beta.1    -2.59       2.56   -1.01 3.12e-01
## beta.2     5.83       1.75    3.33 9.06e-04
## q= 2 k= 2 AIC= 4712.649 BIC= 4735.095 MASE= 1.675967
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0    -9.77      0.672  -14.50 9.70e-42
## beta.1    -4.35      0.355  -12.30 2.62e-31
## beta.2     1.07      0.353    3.03 2.56e-03
## beta.3     6.49      0.669    9.70 7.11e-21
## q= 3 k= 1 AIC= 4687.03 BIC= 4704.981 MASE= 1.666684
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0    -9.92       1.49   -6.66 5.80e-11
## beta.1    -4.24       1.06   -3.98 7.54e-05
## beta.2     1.18       1.07    1.11 2.69e-01
## beta.3     6.34       1.48    4.27 2.22e-05
## q= 3 k= 2 AIC= 4689.018 BIC= 4711.457 MASE= 1.666349
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0  -11.400       1.77  -6.450 2.13e-10
## beta.1   -0.566       2.58  -0.219 8.26e-01
## beta.2   -2.490       2.57  -0.967 3.34e-01
## beta.3    7.820       1.76   4.450 1.01e-05
## q= 3 k= 3 AIC= 4688.551 BIC= 4715.478 MASE= 1.662703
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0    -7.44      0.520  -14.30 1.68e-40
## beta.1    -3.99      0.340  -11.70 6.72e-29
## beta.2    -0.54      0.253   -2.13 3.32e-02
## beta.3     2.91      0.339    8.58 6.70e-17
## beta.4     6.36      0.518   12.30 2.91e-31
## q= 4 k= 1 AIC= 4663.361 BIC= 4681.305 MASE= 1.655786
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0  -8.2800       1.19 -6.9700 7.69e-12
## beta.1  -3.7600       0.45 -8.3500 4.23e-16
## beta.2   0.0481       0.79  0.0609 9.51e-01
## beta.3   3.1400       0.45  6.9900 6.92e-12
## beta.4   5.5200       1.18  4.6600 3.81e-06
## q= 4 k= 2 AIC= 4664.741 BIC= 4687.171 MASE= 1.653286
## Estimates and t-tests for beta coefficients:
```

```
##        Estimate Std. Error t value  P(>|t|)
## beta.0 -10.8000      1.620 -6.6600 5.77e-11
## beta.1   0.1090      1.770  0.0615 9.51e-01
## beta.2   0.0504      0.788  0.0640 9.49e-01
## beta.3  -0.7250      1.770 -0.4100 6.82e-01
## beta.4   7.9900      1.610  4.9700 8.71e-07
## q= 4 k= 3 AIC= 4661.618 BIC= 4688.535 MASE= 1.646472
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0   -5.480      0.436  -12.60 1.48e-32
## beta.1   -3.180      0.321   -9.89 1.44e-21
## beta.2   -0.874      0.244   -3.58 3.67e-04
## beta.3    1.430      0.244    5.86 7.49e-09
## beta.4    3.730      0.320   11.60 1.33e-28
## beta.5    6.030      0.435   13.90 1.64e-38
## q= 5 k= 1 AIC= 4648.873 BIC= 4666.811 MASE= 1.637665
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0   -7.480      0.952  -7.860 1.61e-14
## beta.1   -3.190      0.320  -9.970 6.95e-22
## beta.2    0.105      0.480   0.219 8.26e-01
## beta.3    2.410      0.480   5.020 6.73e-07
## beta.4    3.710      0.319  11.600 1.45e-28
## beta.5    4.020      0.951   4.230 2.67e-05
## q= 5 k= 2 AIC= 4645.25 BIC= 4667.673 MASE= 1.631939
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0  -10.200      1.420  -7.140 2.44e-12
## beta.1   -0.497      1.120  -0.445 6.56e-01
## beta.2    1.720      0.799   2.150 3.17e-02
## beta.3    0.792      0.799   0.991 3.22e-01
## beta.4    1.010      1.120   0.907 3.65e-01
## beta.5    6.680      1.420   4.720 2.95e-06
## q= 5 k= 3 AIC= 4640.873 BIC= 4667.781 MASE= 1.614195
```

The lowest MASE value is observed at q=5 and k=3, therefore I will proceed with those two values for Polynomial DLM modelling.

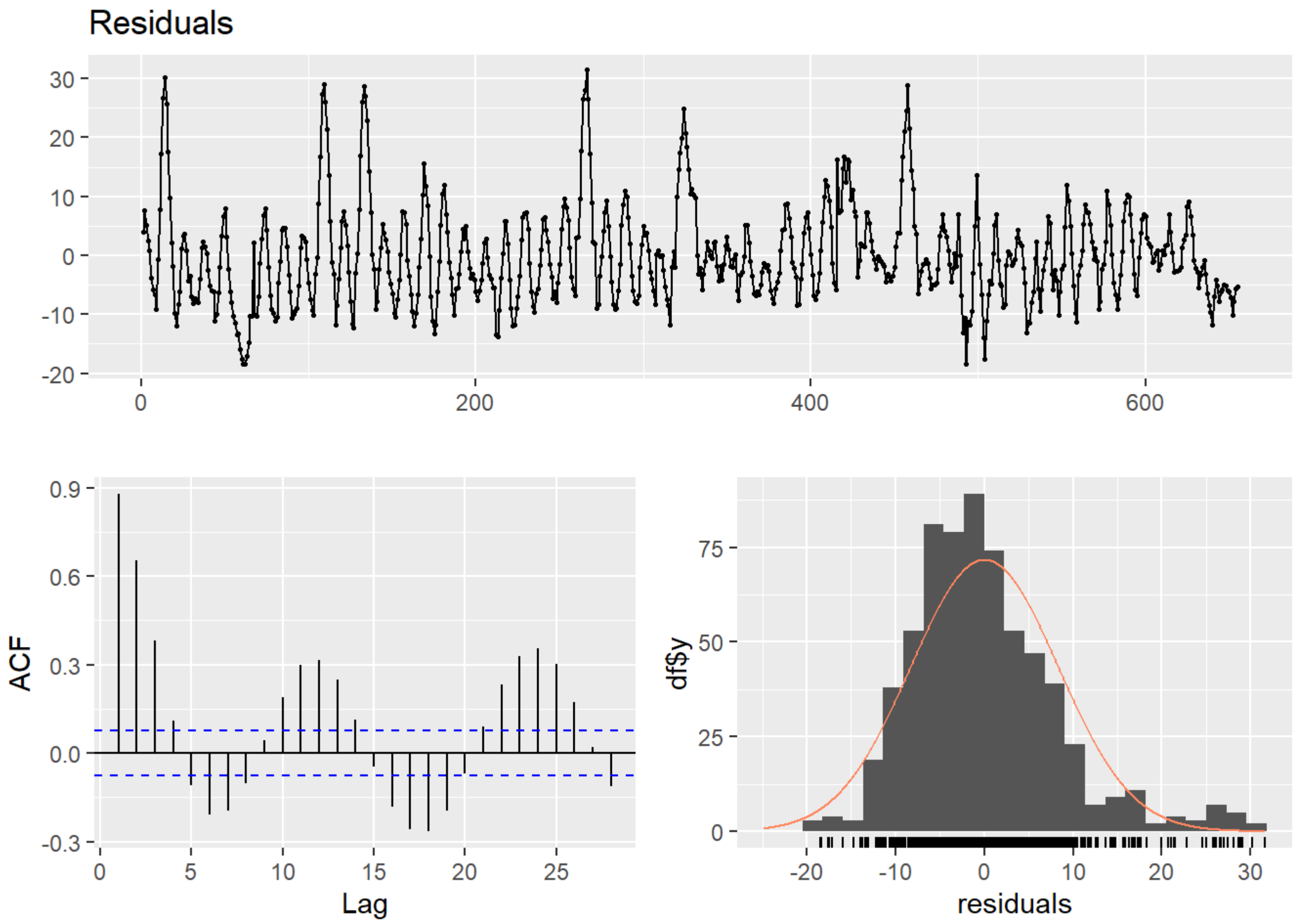
```
p_dlm=polyDlm(x=solar_ppt$ppt,y=solar_ppt$solar,q=5,k=3,show.beta=TRUE)
```

```
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0  -10.200      1.420  -7.140 2.44e-12
## beta.1   -0.497      1.120  -0.445 6.56e-01
## beta.2    1.720      0.799   2.150 3.17e-02
## beta.3    0.792      0.799   0.991 3.22e-01
## beta.4    1.010      1.120   0.907 3.65e-01
## beta.5    6.680      1.420   4.720 2.95e-06
```

```
summary(p_dlm)
```

```
##
## Call:
## "Y ~ (Intercept) + X.t"
##
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -18.471  -5.771  -1.350   4.368  31.562 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  18.0054     0.8318  21.646  < 2e-16 ***
## z.t0        -10.1532     1.4212  -7.144 2.44e-12 ***
## z.t1         14.8083     4.1109   3.602  0.00034 ***
## z.t2         -5.8674     2.1397  -2.742  0.00627 ** 
## z.t3          0.7158     0.2839   2.522  0.01192 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.318 on 650 degrees of freedom
## Multiple R-squared:  0.287,  Adjusted R-squared:  0.2827 
## F-statistic: 65.42 on 4 and 650 DF,  p-value: < 2.2e-16
```

```
checkresiduals(p_dlm$model)
```



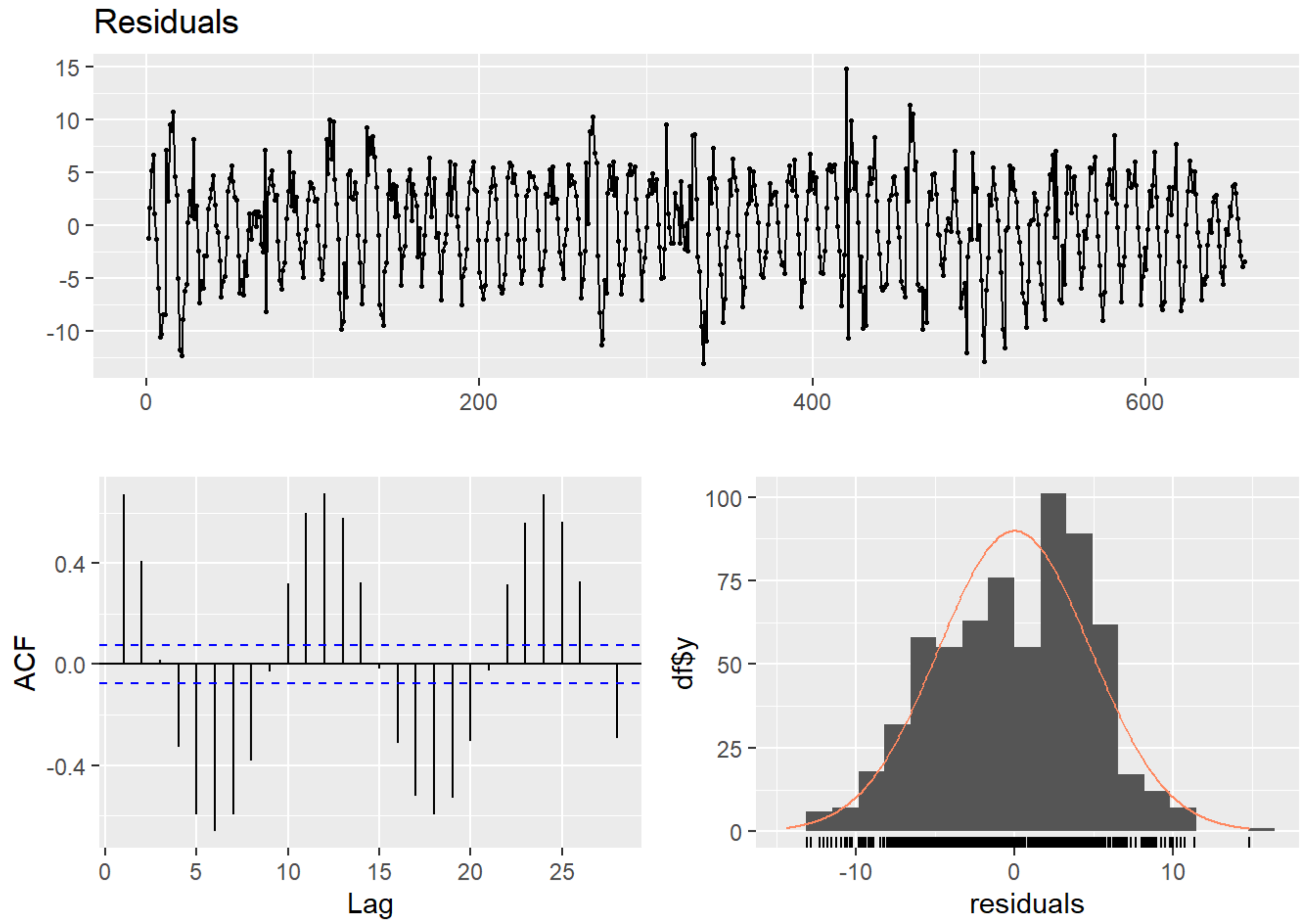
```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  Residuals
## LM test = 585.67, df = 10, p-value < 2.2e-16
```

Koyck Distributed Lag Model

```
koy_dlm=koyckDlm(x=solar_ppt$ppt,y=solar_ppt$solar)
summary(koy_dlm$model,diagnostics=TRUE)
```

```
##
## Call:
## "Y ~ (Intercept) + Y.1 + X.t"
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.0926  -3.5961   0.3176   3.6103  14.8399
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.23925    0.76549  -2.925  0.00356 **
## Y.1          0.98546    0.02424  40.650  < 2e-16 ***
## X.t          5.34684    0.84383   6.336 4.37e-10 ***
##
## Diagnostic tests:
##                  df1 df2 statistic p-value
## Weak instruments   1 656     710.7  <2e-16 ***
## Wu-Hausman         1 655     146.8  <2e-16 ***
## Sargan             0  NA        NA      NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.814 on 656 degrees of freedom
## Multiple R-Squared: 0.7598,  Adjusted R-squared: 0.7591
## Wald test:  1104 on 2 and 656 DF,  p-value: < 2.2e-16
```

```
checkresiduals(koy_dlm$model)
```



```
##
##  Ljung-Box test
##
## data:  Residuals
## Q* = 1413.2, df = 10, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 10
```

Based on the above residual analysis for the Koyck DLM model:

- The lags displayed in the ACF plot have a wave-like structure which is an indicator of serial correlation and seasonality.
- The p-value which is less than 0.05 suggests that the residuals are not normal.Therefore the Koyck DLM model is also not able to capture the autocorrelation and seasonality.

AutoRegressive Distributed Lag Model

```
for(i in 1:5){
  for(j in 1:5){
    mod3 = ardlDlm(x=solar_ppt$ppt,y=solar_ppt$solar, p = i, q = j)
    cat("p = ", i, "q = ", j, "AIC = ", AIC(mod3$model), "BIC = ", BIC(mod3$model),'MASE=',MA
SE(mod3)$MASE, "\n")
  }
}
```

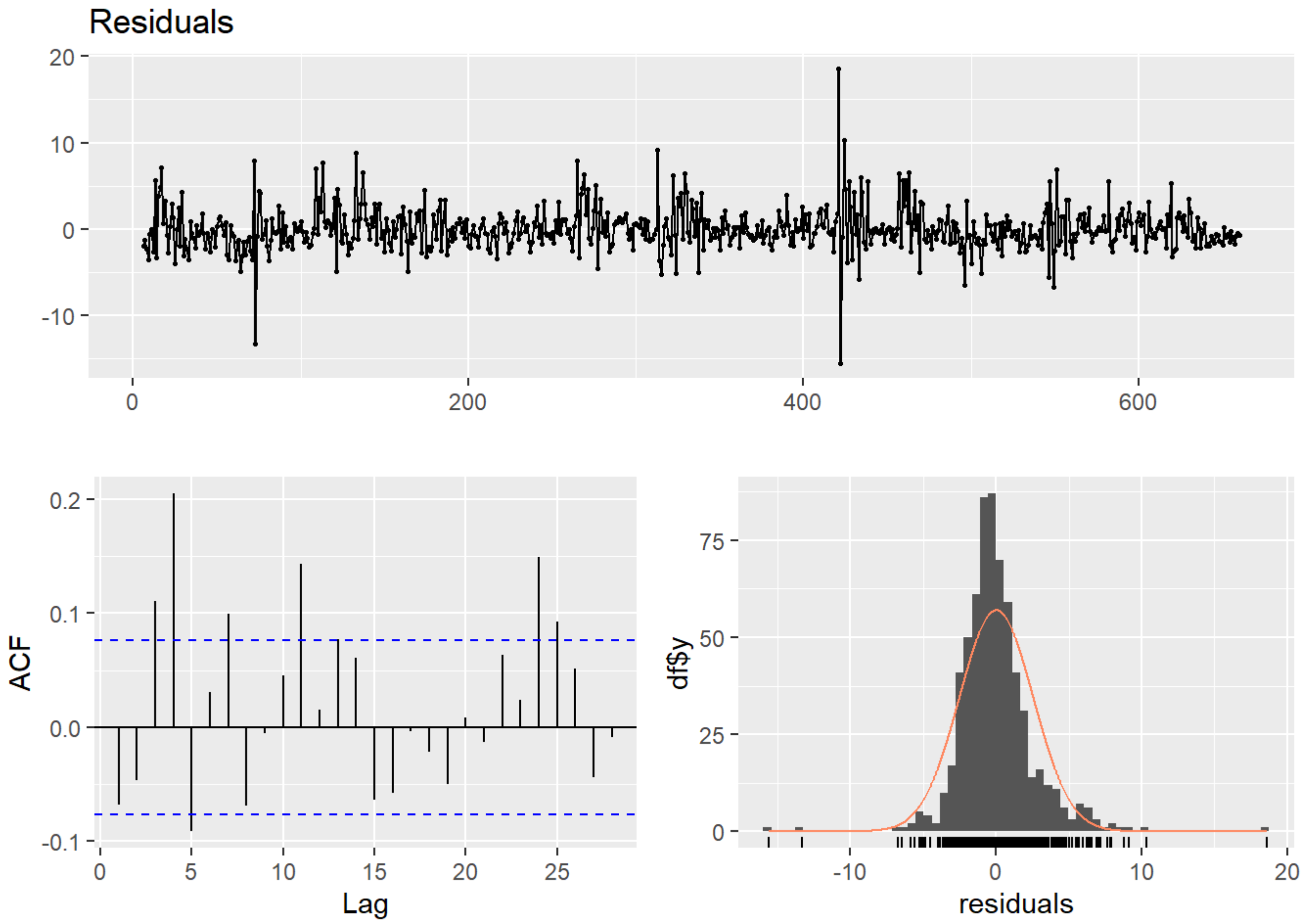
```
## p =  1 q =  1 AIC =  3712.311 BIC =  3734.765 MASE= 0.8392434
## p =  1 q =  2 AIC =  3239.416 BIC =  3266.352 MASE= 0.4971918
## p =  1 q =  3 AIC =  3143.522 BIC =  3174.936 MASE= 0.4740063
## p =  1 q =  4 AIC =  3138.399 BIC =  3174.288 MASE= 0.4697571
## p =  1 q =  5 AIC =  3100.283 BIC =  3140.644 MASE= 0.450425
## p =  2 q =  1 AIC =  3639.223 BIC =  3666.159 MASE= 0.7834855
## p =  2 q =  2 AIC =  3229.051 BIC =  3260.476 MASE= 0.4951319
## p =  2 q =  3 AIC =  3137.634 BIC =  3173.535 MASE= 0.4738939
## p =  2 q =  4 AIC =  3132.962 BIC =  3173.337 MASE= 0.4702773
## p =  2 q =  5 AIC =  3097.288 BIC =  3142.134 MASE= 0.4503599
## p =  3 q =  1 AIC =  3608.793 BIC =  3640.207 MASE= 0.7572489
## p =  3 q =  2 AIC =  3226.623 BIC =  3262.524 MASE= 0.4955334
## p =  3 q =  3 AIC =  3139.409 BIC =  3179.798 MASE= 0.4737144
## p =  3 q =  4 AIC =  3134.777 BIC =  3179.638 MASE= 0.4701162
## p =  3 q =  5 AIC =  3098.808 BIC =  3148.139 MASE= 0.4502885
## p =  4 q =  1 AIC =  3602.664 BIC =  3638.553 MASE= 0.7580664
## p =  4 q =  2 AIC =  3224.285 BIC =  3264.66 MASE= 0.4959949
## p =  4 q =  3 AIC =  3131.289 BIC =  3176.15 MASE= 0.4695096
## p =  4 q =  4 AIC =  3131.424 BIC =  3180.772 MASE= 0.4665123
## p =  4 q =  5 AIC =  3096.024 BIC =  3149.839 MASE= 0.4479481
## p =  5 q =  1 AIC =  3599.402 BIC =  3639.764 MASE= 0.7572617
## p =  5 q =  2 AIC =  3221.853 BIC =  3266.699 MASE= 0.4954501
## p =  5 q =  3 AIC =  3127.103 BIC =  3176.434 MASE= 0.4675479
## p =  5 q =  4 AIC =  3127.868 BIC =  3181.684 MASE= 0.4651969
## p =  5 q =  5 AIC =  3097.877 BIC =  3156.177 MASE= 0.4479311
```

The lowest MASE value is observed at p=5 and q=5, therefore I will proceed with those two values for Autoregressive DLM modelling.

```
autoreg_dlm = ardlDlm(x=solar_ppt$ppt,y=solar_ppt$solar,p=5,q=5)
summary(autoreg_dlm)
```

```
##
## Time series regression with "ts" data:
## Start = 6, End = 660
##
## Call:
## dynlm(formula = as.formula(model.text), data = data, start = 1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.5959  -1.3825  -0.2646   1.0410  18.5812
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.50740    0.45434   5.519 4.96e-08 ***
## X.t         -0.61416    0.54804  -1.121 0.262863
## X.1          0.78299    0.77670   1.008 0.313788
## X.2          1.26543    0.79241   1.597 0.110772
## X.3          0.75184    0.79227   0.949 0.342998
## X.4         -1.00181    0.77678  -1.290 0.197617
## X.5         -0.21024    0.55439  -0.379 0.704639
## Y.1          1.27063    0.03867  32.861  < 2e-16 ***
## Y.2         -0.01727    0.06264  -0.276 0.782907
## Y.3         -0.40297    0.06043  -6.669 5.56e-11 ***
## Y.4         -0.23273    0.06229  -3.737 0.000203 ***
## Y.5          0.21571    0.03802   5.673 2.12e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.548 on 643 degrees of freedom
## Multiple R-squared:  0.9338, Adjusted R-squared:  0.9327
## F-statistic: 824.9 on 11 and 643 DF,  p-value: < 2.2e-16
```

```
checkresiduals(autoreg_dlm$model)
```



```
##
##  Breusch-Godfrey test for serial correlation of order up to 15
##
## data:  Residuals
## LM test = 107.98, df = 15, p-value = 3.937e-16
```

Based on the above residual analysis for AutoRegressive DLM model:

- There is a presence of changing variance in the residuals and they show signs of non-randomness.
- There are some high spikes in the ACF plot which indicate autocorrelation and seasonality.
- The histogram plot also contains long tails which suggests that the normality of the residuals is violated.

Upon fitting the Finite, Polynomial, Koyck, and AutoRegressive DLM model it turns out that these models were not able to capture the autocorrelation and seasonality present in the solar radiation series. I have created a dataframe named ‘accurate’ to store and compare the AIC, BIC, and MASE values for all the models fitted and which are going to be fitted further.

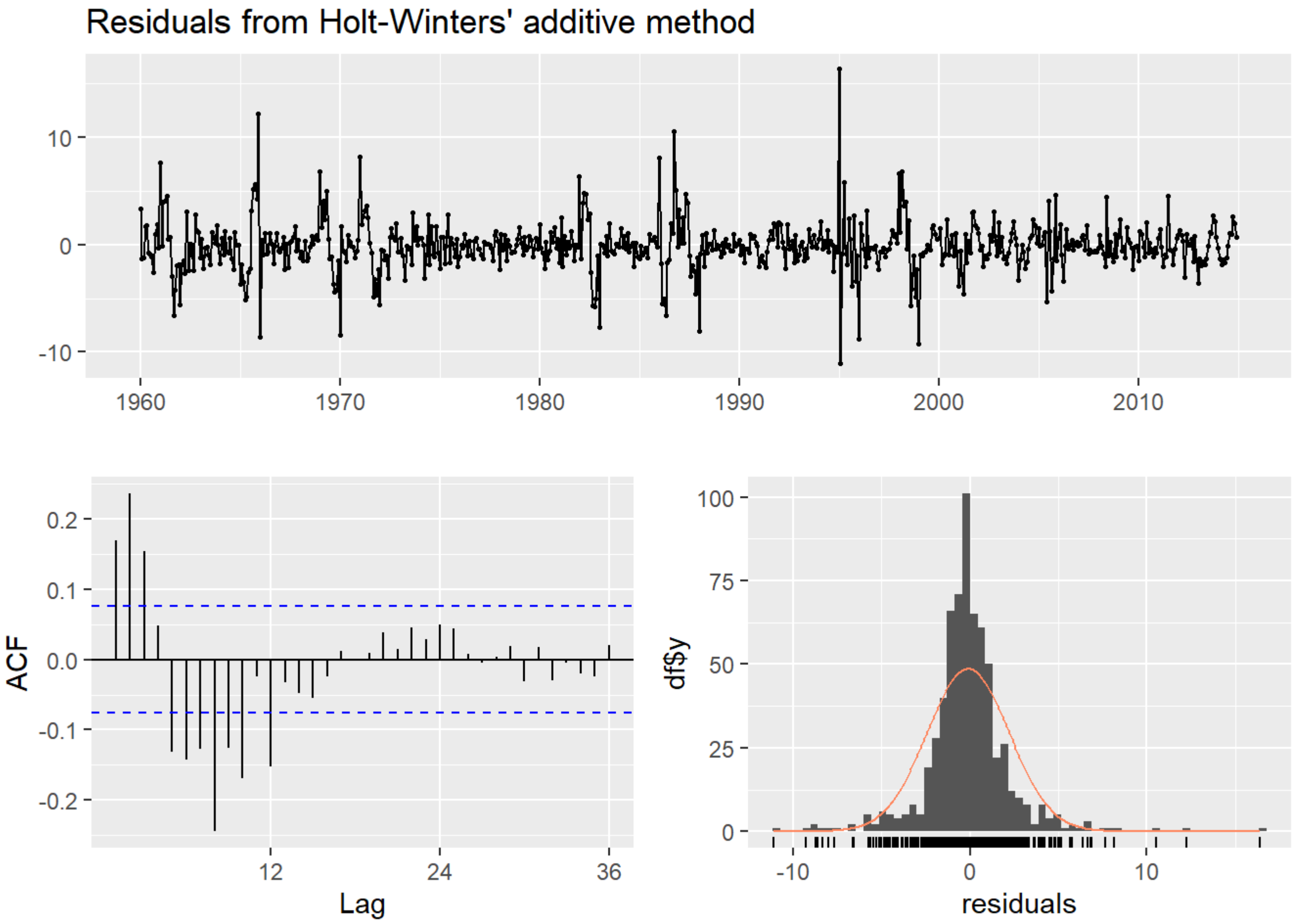
```
attr(koy_dlm$model,'class')='lm'
mods=c('Finite DLM','Poly DLM','Koyck', 'AutoReg DLM')
mase=MASE(f_dlm$model,p_dlm$model,koy_dlm$model,autoreg_dlm)$MASE
accurate=data.frame(mods,mase)
colnames(accurate)=c('Models','MASE')
head(accurate)
```

```
##        Models      MASE
## 1  Finite DLM 1.5779955
## 2    Poly DLM 1.6141953
## 3       Koyck 1.0324829
## 4 AutoReg DLM 0.4479311
```

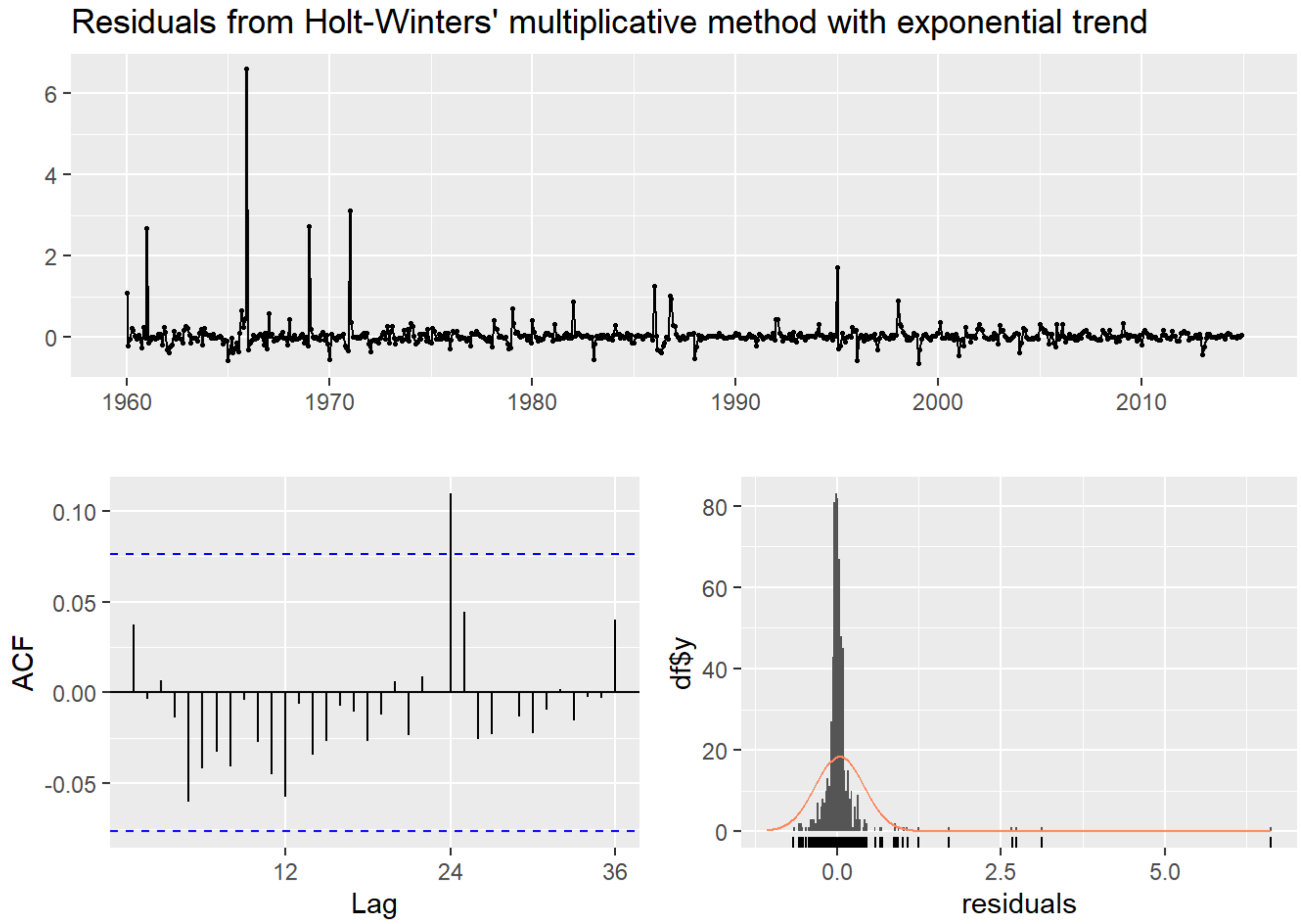
Exponential Smoothing Methods

As there is a presence of seasonality in the given time series we can try another method which is the exponential smoothing but for that we will focus on the models that include additive or multiplicative seasonality.

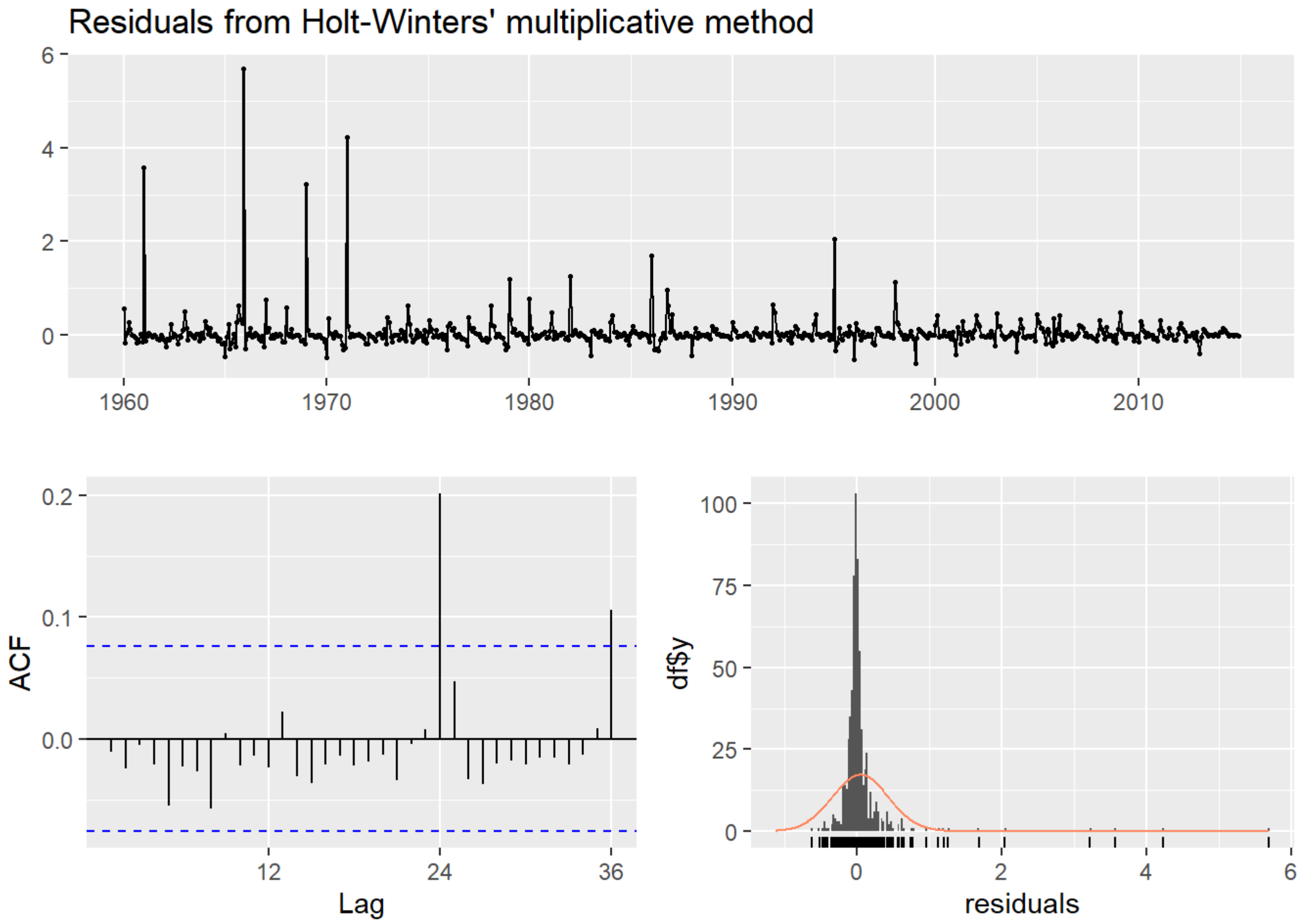
```
expo=c(T,F)
season=c('additive',"multiplicative")
damp=c(T,F)
exp=expand.grid(expo,season,damp)
exp=exp[-c(1,5),]
expo_smth_aic=array(NA,6)
expo_smth_bic=array(NA,6)
expo_smth_mase=array(NA,6)
lvl=array(NA,dim=c(6,3))
for (i in 1:6){
  hw=hw(solar,expo = exp[i,1],seasonal=toString(exp[i,2],damp=exp[i,3]))
  expo_smth_aic[i]=hw$model$aic
  expo_smth_bic[i]=hw$model$bic
  expo_smth_mase[i]=accuracy(hw)[6]
  lvl[i,1]=exp[i,1]
  lvl[i,2]=toString(exp[i,2])
  lvl[i,3]=exp[i,3]
  checkresiduals(hw)
}
```



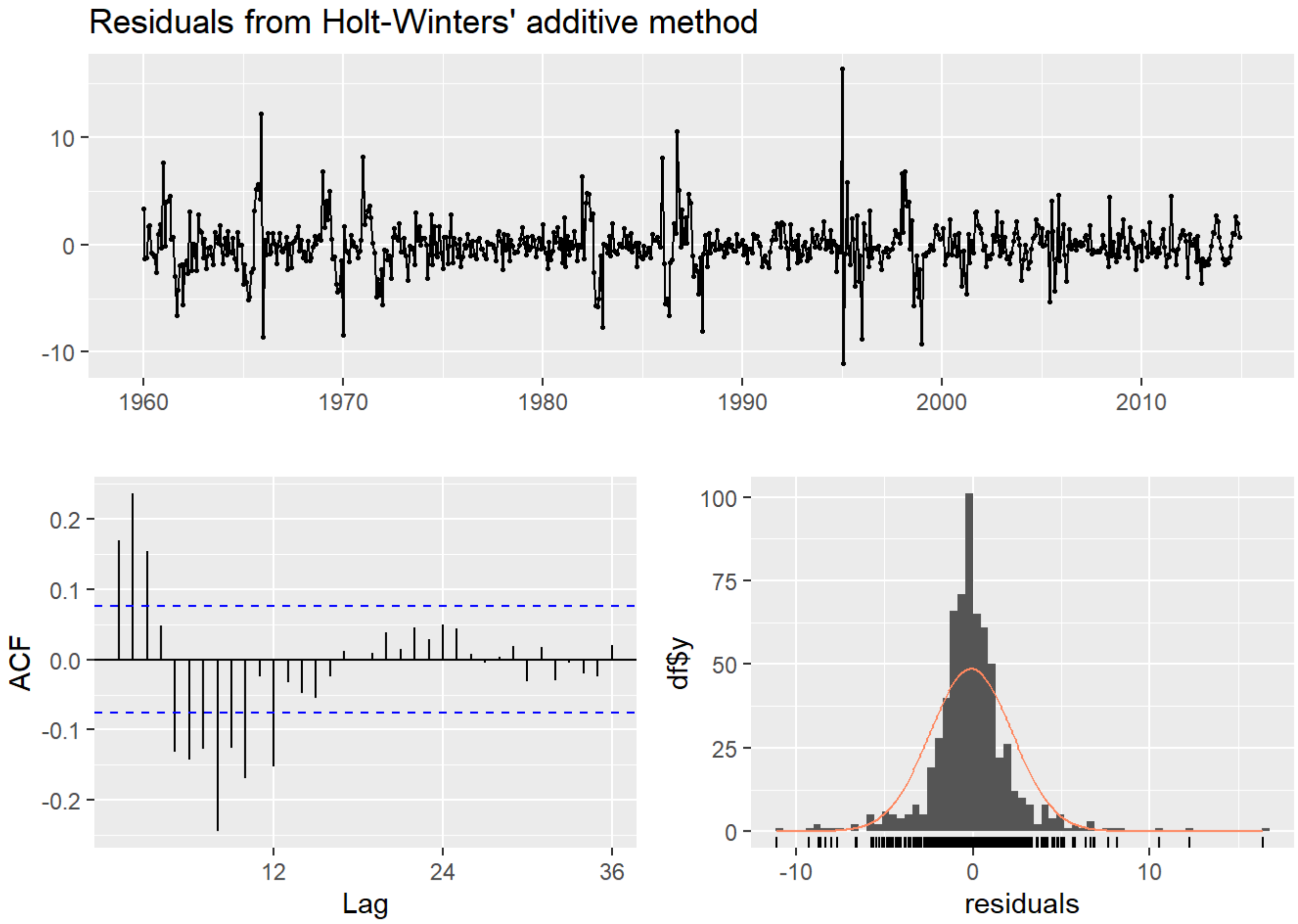
```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 205.55, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```



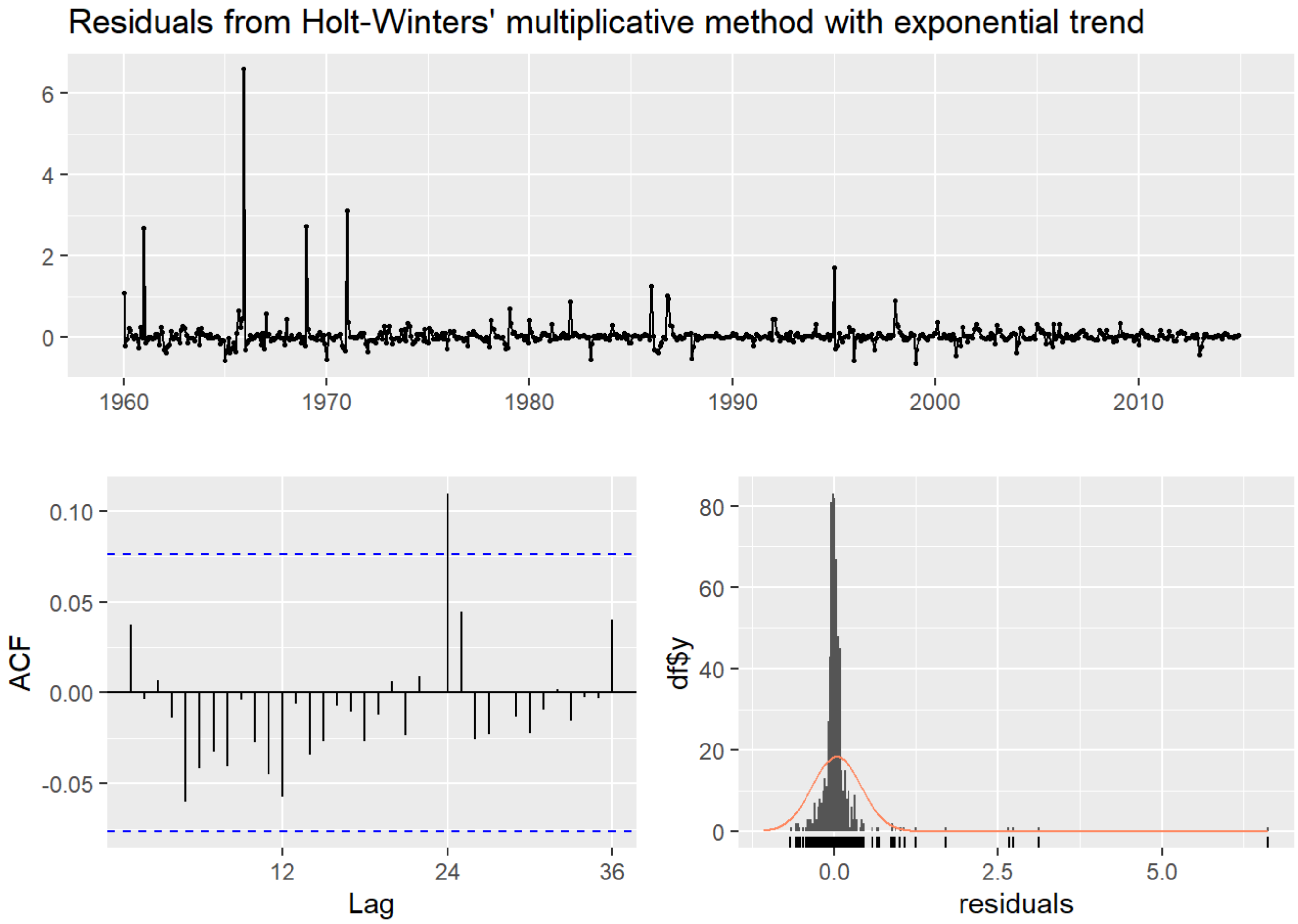
```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method with exponential trend
## Q* = 21.246, df = 24, p-value = 0.6242
##
## Model df: 0.   Total lags used: 24
```



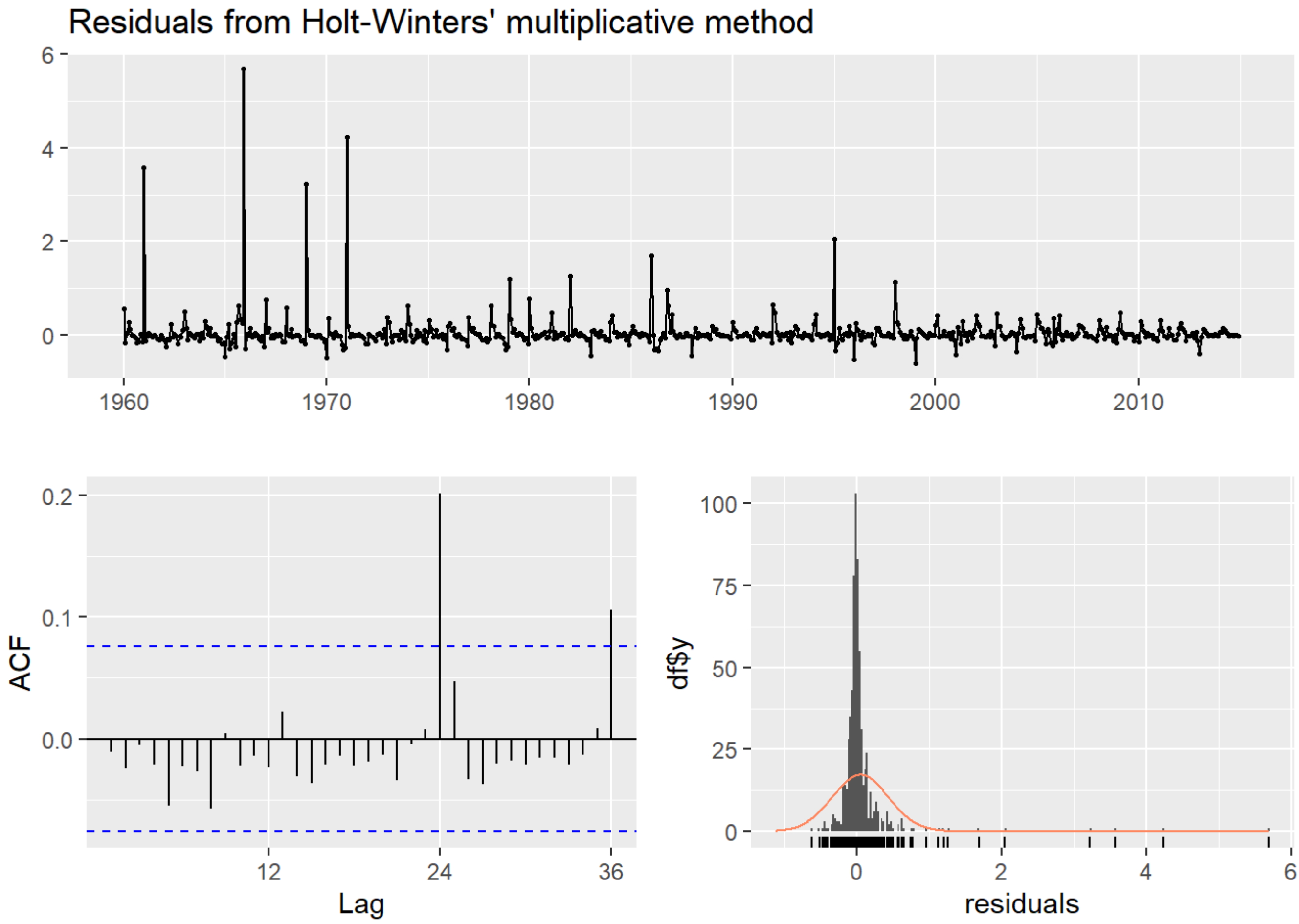
```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method
## Q* = 38.585, df = 24, p-value = 0.03017
##
## Model df: 0.   Total lags used: 24
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 205.55, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```


Residuals from Holt-Winters' multiplicative method with exponential trend

```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method with exponential trend
## Q* = 21.246, df = 24, p-value = 0.6242
##
## Model df: 0.   Total lags used: 24
```



```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method
## Q* = 38.585, df = 24, p-value = 0.03017
##
## Model df: 0.   Total lags used: 24
```

Based on the above results, there is a slight improvement in the residuals as compared to the DLM models in capturing the correlation and seasonality present in the series. Therefore, the results from the exponential smoothing methods are added to the earlier created dataframe so as to compare based on the MASE values.

```
vals=data.frame(lvl,expo_smth_mase)
colnames(vals)=c('Trend','Seasonality','Damped','MASE')
vals$Trend=factor(vals$Trend,levels=c(T,F),labels=c('multiplicative','additive'))
vals$Damped=factor(vals$Damped,levels=c(T,F),labels=c('damped','N'))
vals=unite(vals,col='Models',c('Trend','Seasonality','Damped'))
accurate=rbind(accurate,vals)
accurate
```

```
##                                    Models      MASE
## 1                              Finite DLM 1.5779955
## 2                                Poly DLM 1.6141953
## 3                                   Koyck 1.0324829
## 4                             AutoReg DLM 0.4479311
## 5                additive_additive_damped 0.2471600
## 6  multiplicative_multiplicative_damped 0.2320404
## 7          additive_multiplicative_damped 0.2233077
## 8                     additive_additive_N 0.2471600
## 9         multiplicative_multiplicative_N 0.2320404
## 10              additive_multiplicative_N 0.2233077
```

State-Space Model Variations

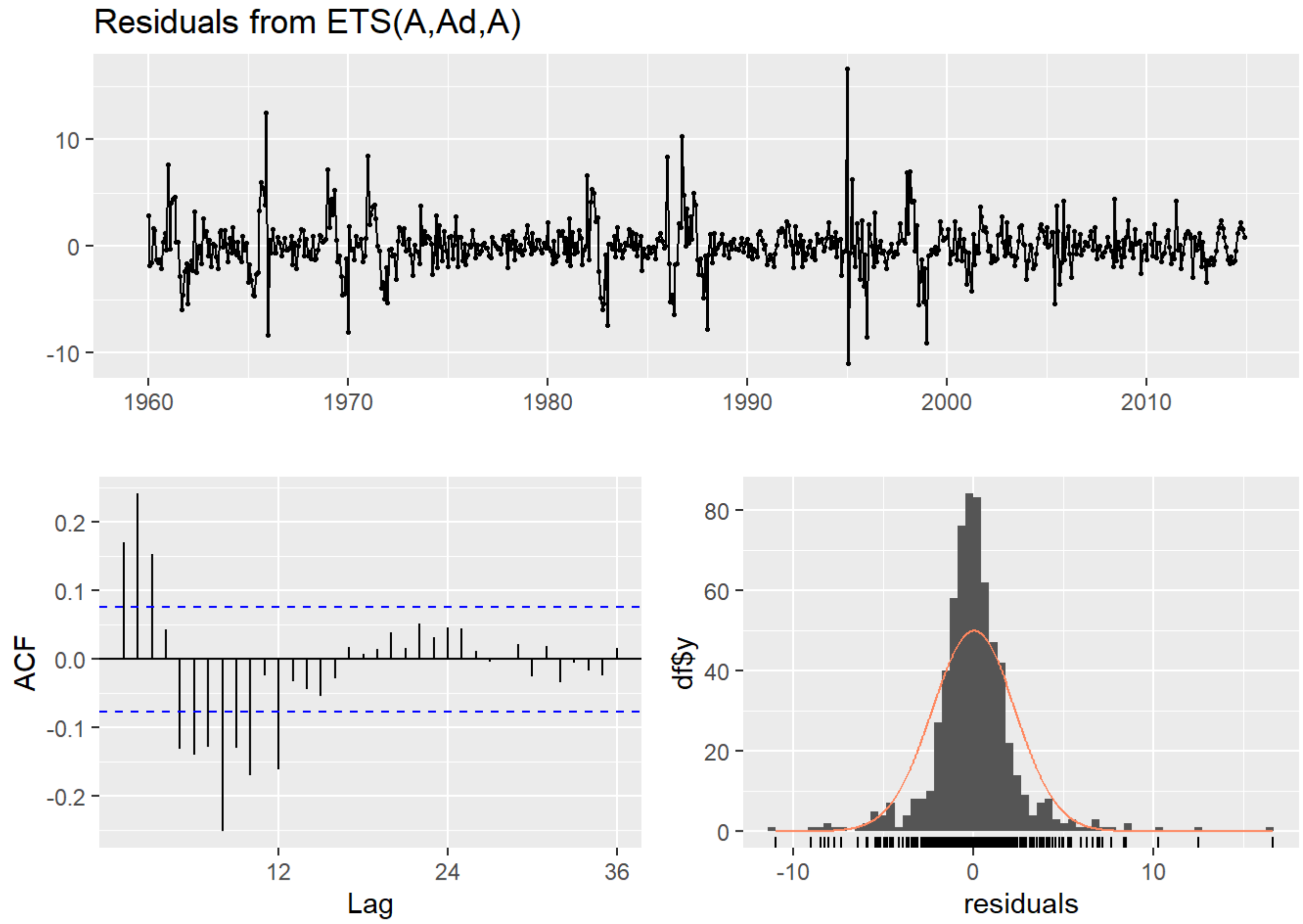
```
var=c('AAA','MAA','MAM','MMM')
damps=c(T,F)
statespace_models=expand.grid(var,damps)
statespace_aic=array(NA,8)
statespace_bic=array(NA,8)
statespace_mase=array(NA,8)
mod=array(NA,dim=c(8,2))
for (i in 1:8){
  s_space=ets(solar,model=toString(statespace_models[i,1]),damped=statespace_models[i,2])
  statespace_aic[i]=s_space$aic
  statespace_bic[i]=s_space$bic
  statespace_mase[i]=accuracy(s_space)[6]
  mod[i,1]=toString(statespace_models[i,1])
  mod[i,2]=statespace_models[i,2]
}
```

```
statespace <- ets(solar)
summary(statespace)
```

```
## ETS(A,Ad,A) 
## 
## Call:
## ets(y = solar) 
## 
##   Smoothing parameters:
##     alpha = 0.9999 
##     beta  = 1e-04 
##     gamma = 1e-04 
##     phi   = 0.9388 
## 
##   Initial states:
##     l = 11.154 
##     b = 0.7632 
##     s = -10.4919 -8.137 -3.348 2.5794 8.08 11.1219
##            9.9586 6.9916 1.9573 -1.8565 -7.1607 -9.6946
## 
##   sigma:  2.3446
## 
##      AIC     AICc      BIC 
## 5428.422 5429.489 5509.282 
## 
## Training set error measures:
##                        ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.01091357 2.314163 1.498521 -1.468083 12.44796 0.2461797
##                   ACF1
## Training set 0.1700724
```

Based on the above results, the model suggested by the software is ETS(A,Ad,A) which includes additive errors, additive damped trend and additive seasonality in it.

```
checkresiduals(statespace)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,Ad,A)
## Q* = 210.76, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

Based on the above results, the measures for the state space models are also added to the dataframe created earlier.

```
acc_measure=data.frame(mod,statespace_mase)
acc_measure$X2=factor(acc_measure$X2,levels=c(T,F),labels=c('Damped','N'))
acc_measure=unite(acc_measure,'Model',c('X1','X2'))
colnames(acc_measure)=c('Models','MASE')
accurate=rbind(accurate,acc_measure)
accurate=arrange(accurate, MASE)
kable(accurate, caption = "Models sorted by MASE")
```

Models sorted by MASE

Models	MASE
additive_multiplicative_damped	0.2233077
additive_multiplicative_N	0.2233077
multiplicative_multiplicative_damped	0.2320404
multiplicative_multiplicative_N	0.2320404
AAA_Damped	0.2461797
additive_additive_damped	0.2471600

Models	MASE
additive_additive_N	0.2471600
AAA_N	0.2471600
MMM_Damped	0.3201193
MAM_Damped	0.3222574
MAM_N	0.3721664
MAA_Damped	0.3798095
AutoReg DLM	0.4479311
MAA_N	0.4748561
MMM_N	0.5292151
Koyck	1.0324829
Finite DLM	1.5779955
Poly DLM	1.6141953

Based on the above result, the model that gives the lowest MASE is the additive_multiplicative_damped model with a MASE value of 0.2233077.

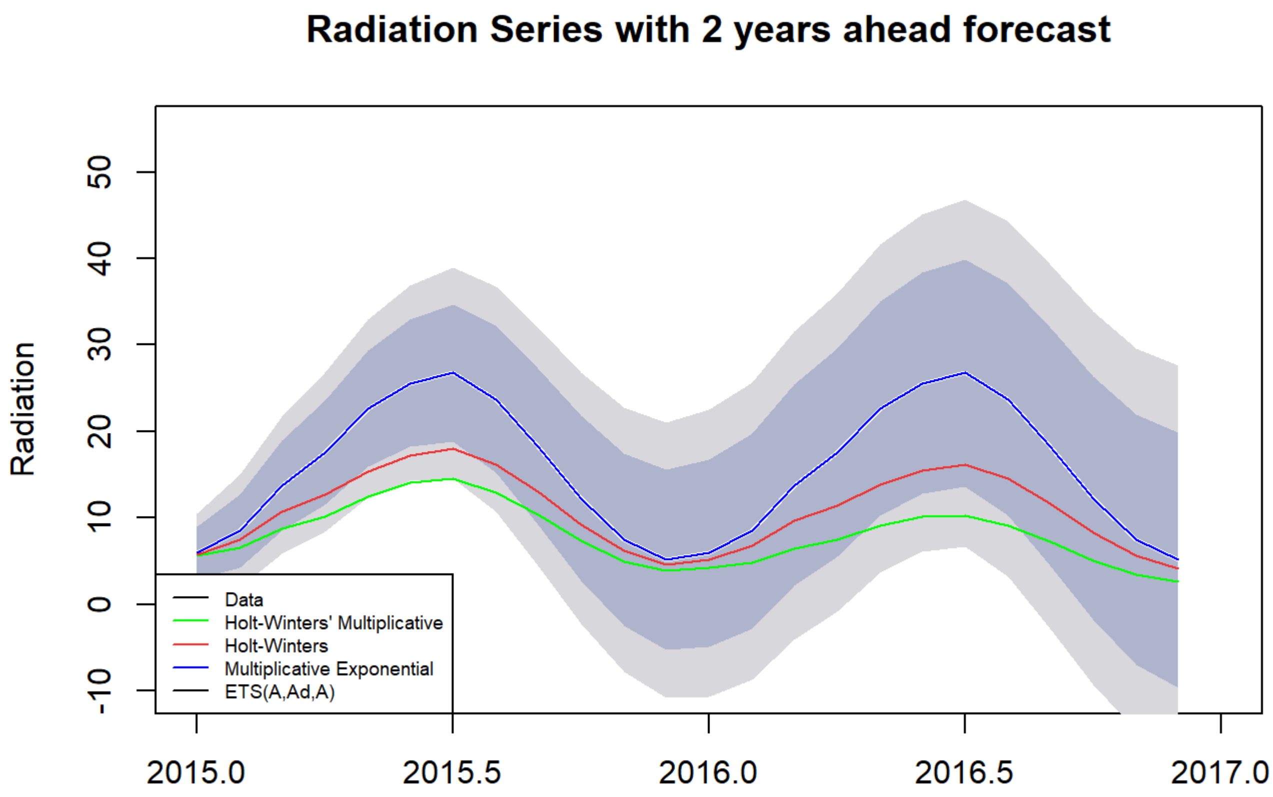
Now as we checked all the models based on their MASE values, to select the best model for forecasting I compared three options. The Holt-Winters multiplicative method gave the lowest MASE and best captured autocorrelation and seasonality. The version with a multiplicative trend had the 2nd best MASE. The ETS(A,Ad,A) model which was suggested by the software had the lowest MASE between the state space models but it was unable to capture autocorrelation which was present in the data.

Forecasting

I have loaded the data.x which is to be used for forecasting the solar radiation for the next 2 years. However, the models which are selected based on the MASE values i.e. Holt-Winters(both standard and exponential) and ETS(A,Ad,A) are univariate in their approach. That means, these models rely only on the historical values of the target variable which is solar radiation in this case to capture the level, trend, and seasonality. As they do not include external predictors(data.x), adding data.x would not influence the forecast and it would be unnecessary.

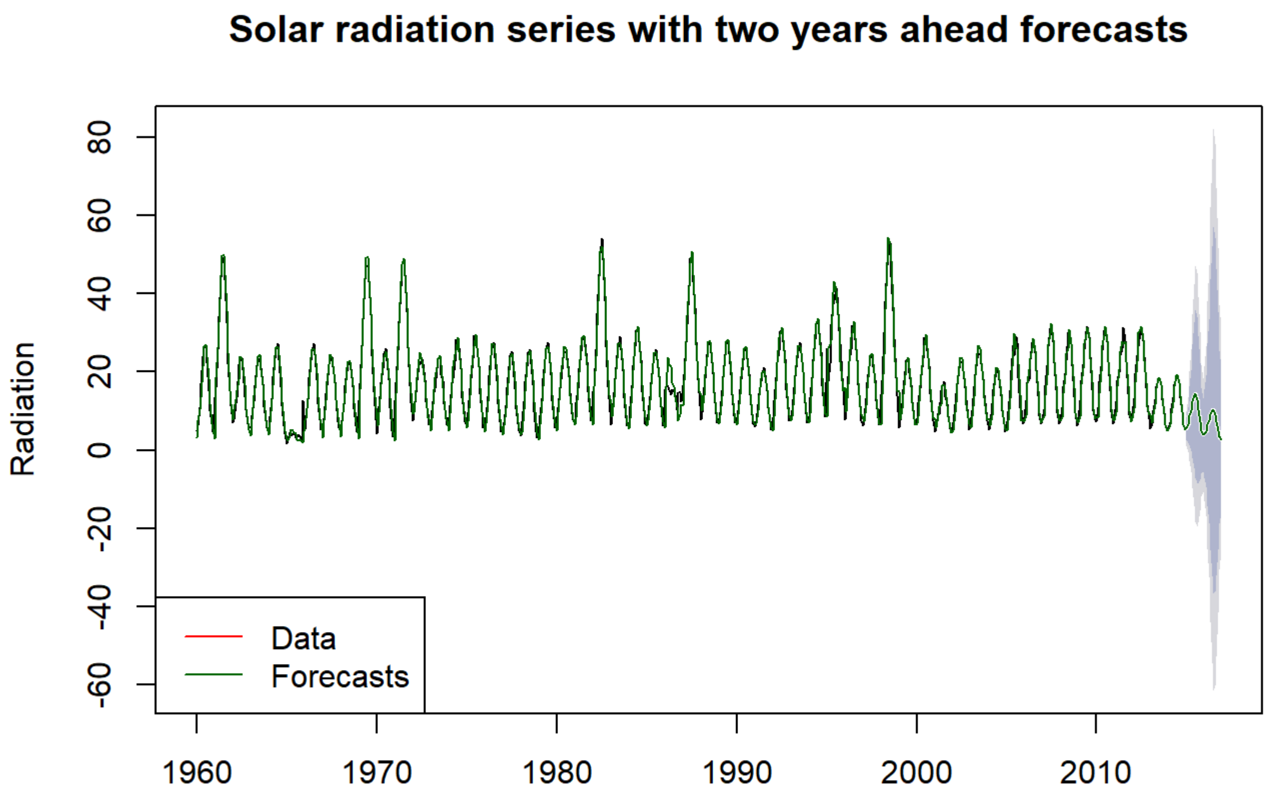
```
forecast1 = hw(solar, seasonal="multiplicative", h=2*frequency(solar))
forecast2 =hw(solar, seasonal="multiplicative", exponential=T, h=2*frequency(solar))
forecast3 = ets(solar, model="AAA", damped = T)
fore_radiation=forecast::forecast(forecast3)
```

```
plot(fore_radiation,xlim=c(2015,2017), fcol = "white", main = "Radiation Series with 2 years
ahead forecast", ylab = "Radiation", ylim = c(-10,55))
lines(fitted(forecast1), col = "green")
lines(forecast1$mean, col = "green", lwd = 1)
lines(fitted(forecast2), col = "brown2")
lines(forecast2$mean, col = "brown2", lwd = 1)
lines(fitted(forecast3), col = "blue")
lines(fore_radiation$mean, col = "blue", lwd = 1)
legend("bottomleft", lty = 1,cex=0.6, col = c("black", "green", "brown2", "blue"), c("Data",
"Holt-Winters' Multiplicative", "Holt-Winters","Multiplicative Exponential", "ETS(A,Ad,A)"))
```



Based on the above plot, we can see that all the 3 models produce fitted values that are generally close to the original data. However the state-space model showed the largest difference. The forecasts from the models vary noticeably. The state-space model predicts the highest peaks. The Holt-Winters multiplicative model suggests a decline in the solar radiation over the period of 2 years, it's version of a multiplicative trend showed stable levels for the 1st year which is followed by a slight drop in the 2nd year. All in all, based on residual plots and the comparison of the MASE values, I chose the Holt-Winters multiplicative model for forecasting the solar radiation 2 years ahead.

```
plot(forecast1, fcol = "white", main = "Solar radiation series with two years ahead forecast
s", ylab = "Radiation")
lines(fitted(forecast1), col = "darkgreen")
lines(forecast1$mean, col = "darkgreen", lwd = 1)
legend("bottomleft", lty = 1, col = c("red", "darkgreen"), c("Data", "Forecasts"))
```



Task 2

Data Description

I have loaded the data2.csv which contains data about the quaterly Residential PPI in Melbourne and population change in Victoria over the previous quarter from September 2003 to December 2016. After that, I have converted the dataframe into time series object for visualisation and further analysis.

```
ppi_population =read_csv('data2.csv')
```

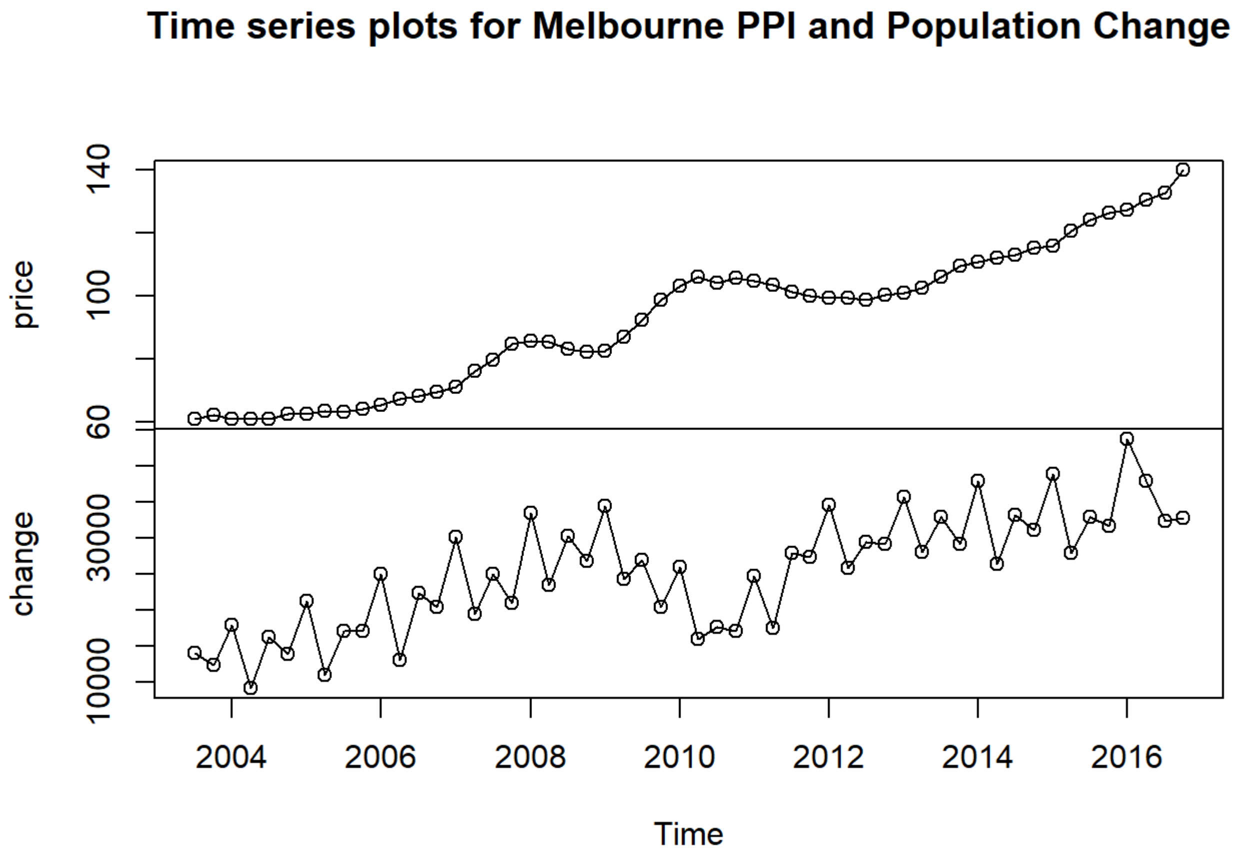
```
## Rows: 54 Columns: 3
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (1): Quarter
## dbl (2): price, change
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
cppi=ts(ppi_population[,2:3],start=c(2003,3),frequency=4)
quaterly_ppi=ts(ppi_population$price,start=c(2003,3),frequency=4)
pop_change=ts(ppi_population$change,start=c(2003,3),frequency=4)
head(cppi)
```

```
##         price change
## 2003 Q3  60.7  14017
## 2003 Q4  62.1  12350
## 2004 Q1  60.8  17894
## 2004 Q2  60.9   9079
## 2004 Q3  60.9  16210
## 2004 Q4  62.4  13788
```

Data Visualisation

```
plot(cppi,main='Time series plots for Melbourne PPI and Population Change',type='o')
```



Upon visualizing both the time series, it can be clearly seen that they have upward trends, and the population change appears to be seasonal. The plot also shows a possible correlation between the two and we will explore the correlation between them in the further steps.

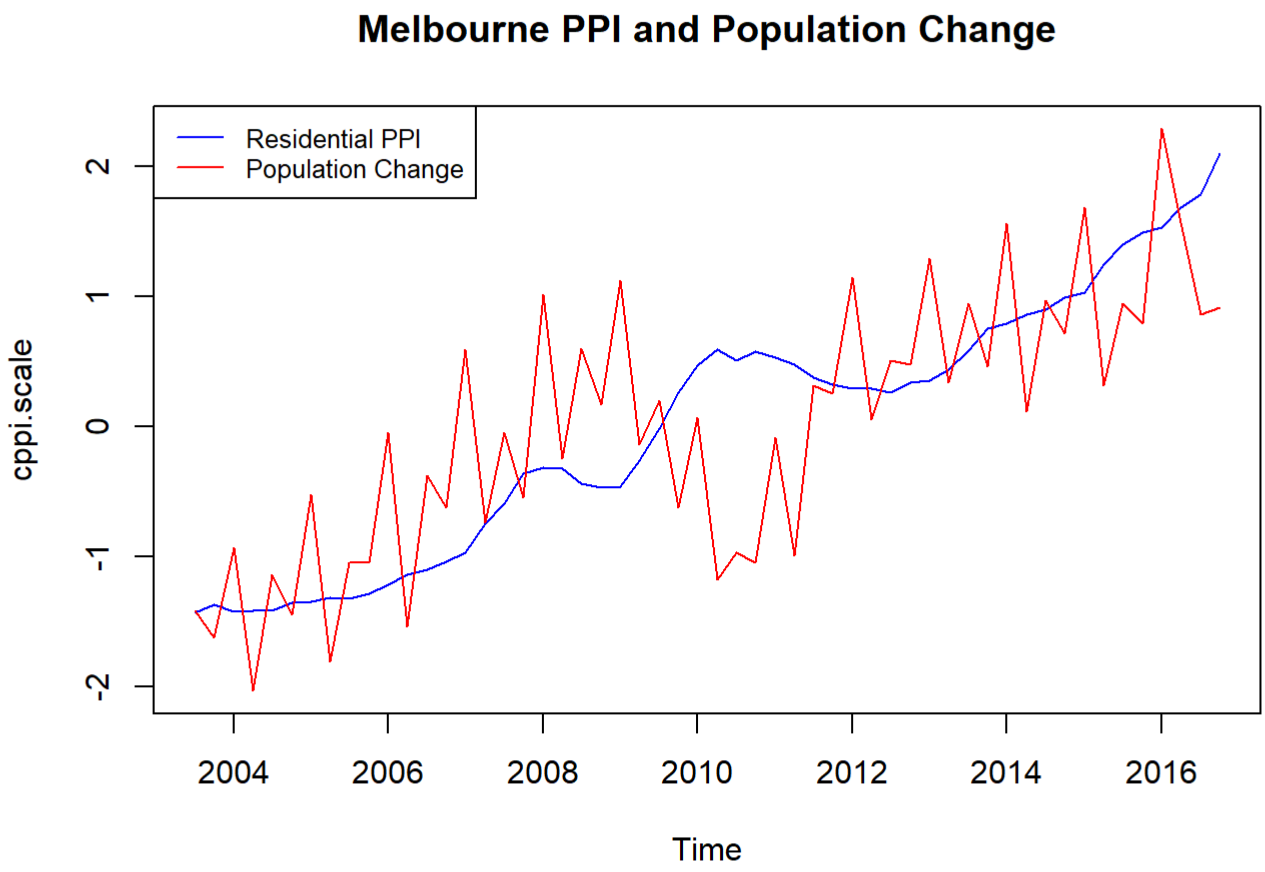
Correlation

```
cor(cppi)
```

```
##            price    change
## price  1.0000000 0.6970439
## change 0.6970439 1.0000000
```

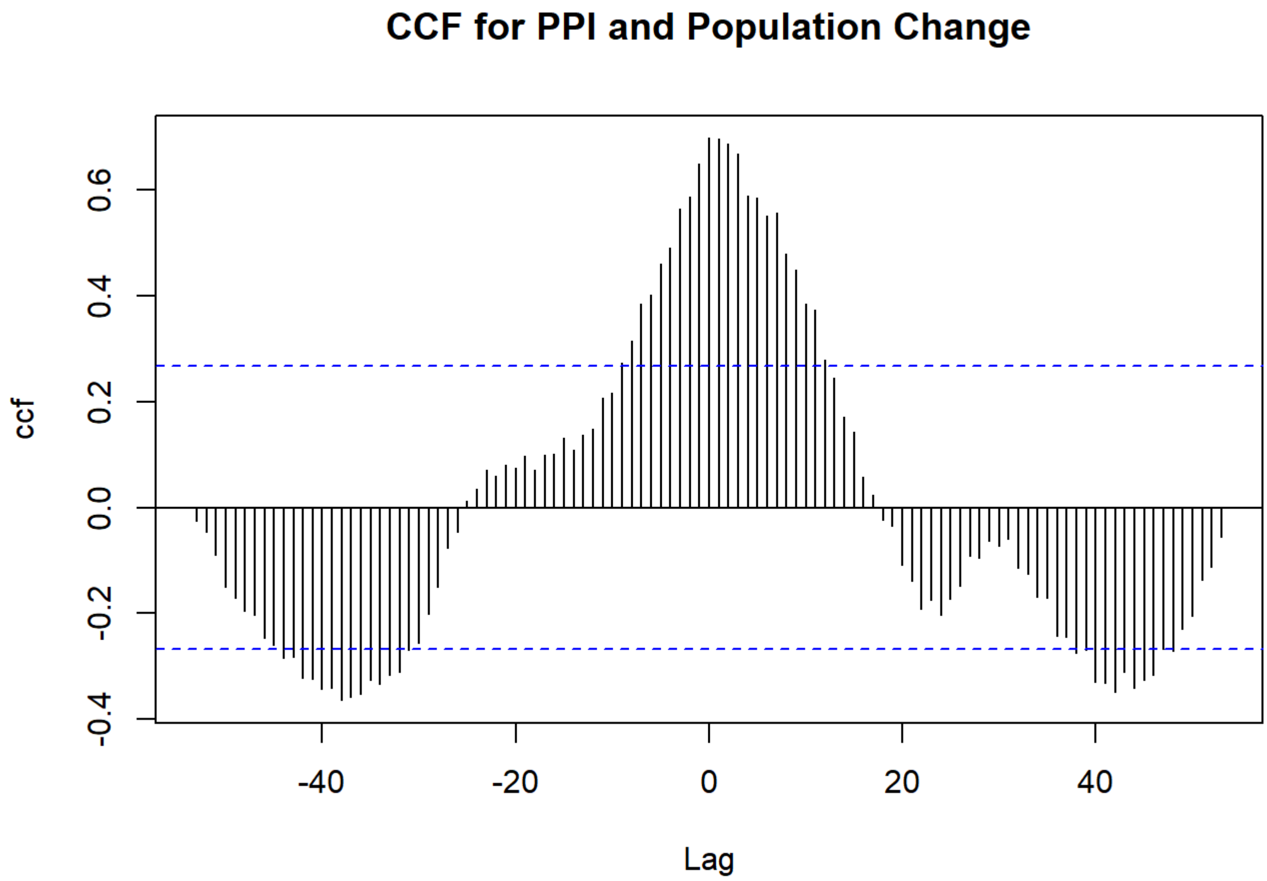
Based on the above correlation matrix between the two variables, there is a strong correlation i.e. 0.697 between Melbourne PPI and Population Change which indicates that they tend to move together.

```
cppi.scale = scale(cppi)
plot(cppi.scale, plot.type="s", col=c("blue", "red"), main = "Melbourne PPI and Population Ch
ange")
legend("topleft",
       legend=c("Residential PPI", "Population Change"),
       col=c("blue", "red"),
       lty=1,
       cex=0.8)
```



Based on the above plot, it is clear that both the time series are correlated with each other. Both the time series plot shows upward trend. The strong correlation between the two suggests a linked relation.

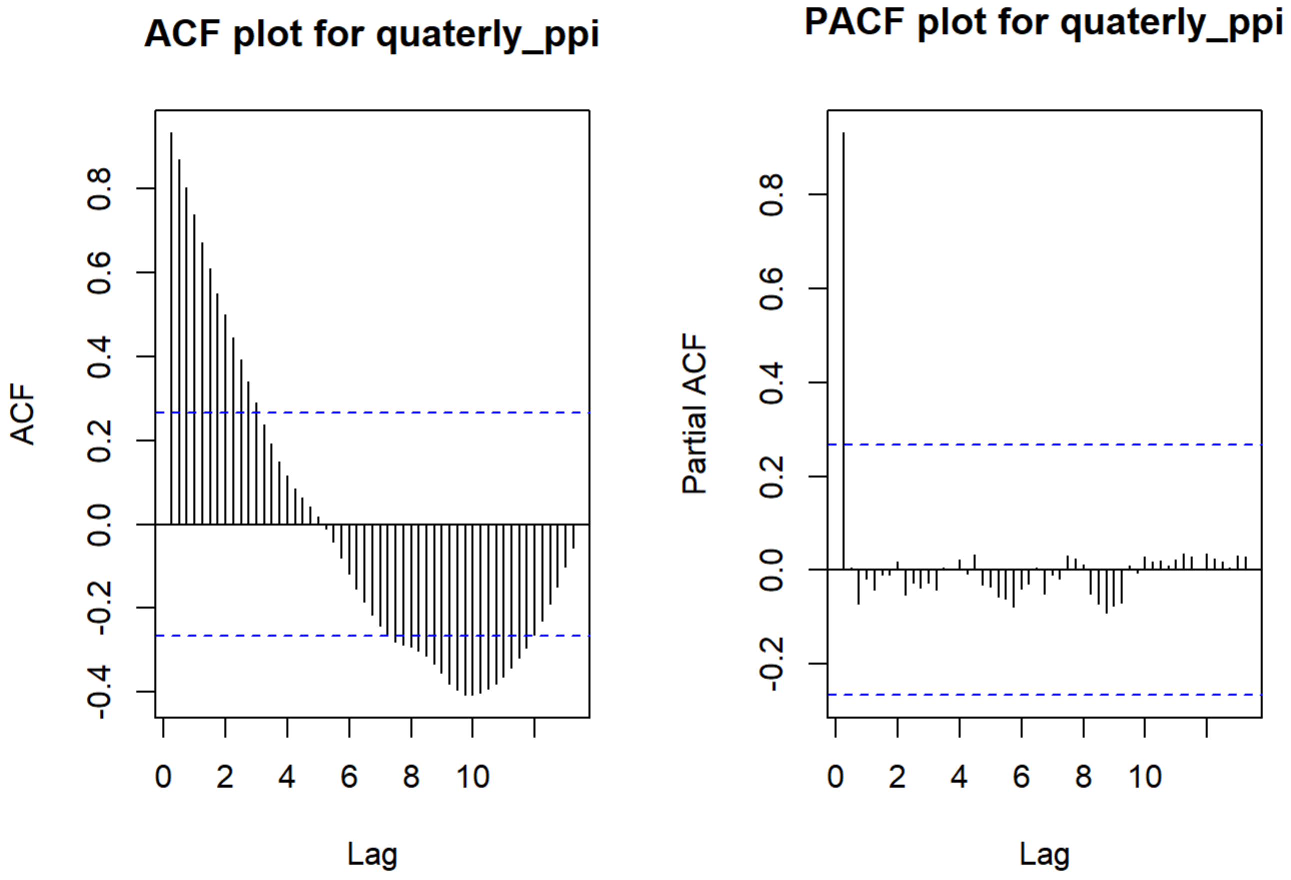
```
ccf(as.vector(quaterly_ppi),as.vector(pop_change),ylab='ccf',main='CCF for PPI and Population
Change',lag.max = 70)
```



From the above CCF plot for PPI and Population Change, there are several lags in the CCF plot that are significantly different from zero. This indicates cross-correlation between Melbourne’s PPI and Population Change. This might also be possible to to nonstationarity present in the dataset which could lead to misleading correlations.

Presence of Nonstationarity

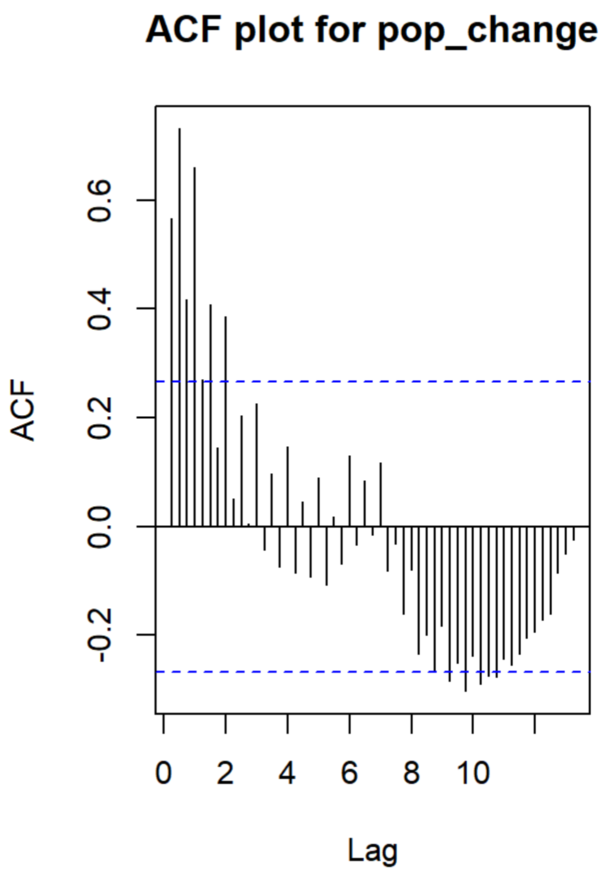
```
acf_pacf(quaterly_ppi)
```

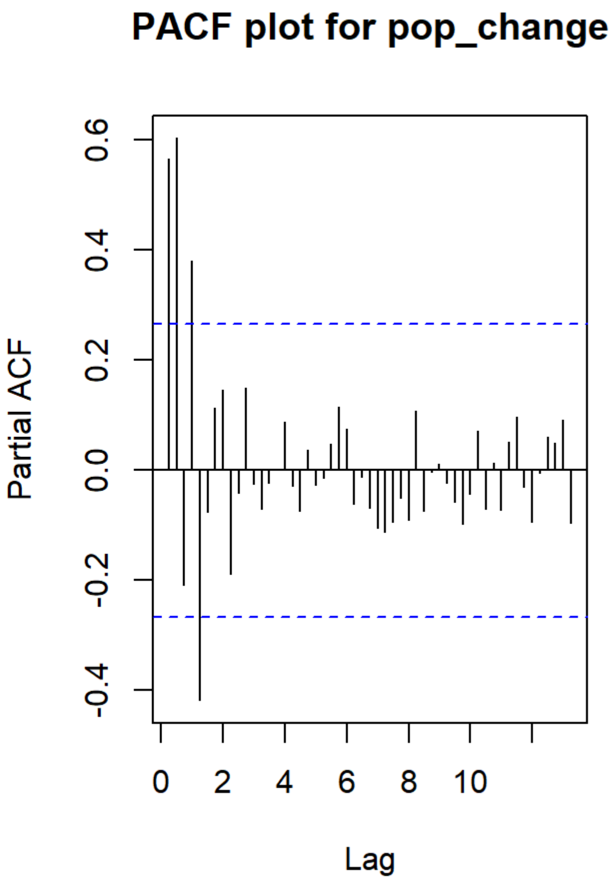


Based on the above ACF and PACF plots for Melbourne PPI:

- The lags in the ACF plot for Melbourne PPI shows a decaying pattern which is an indicator that the series is nonstationary, and it is also evident that there is no seasonality present in the time series.
- The PACF plot for Melbourne PPI indicates partial autocorrelation at the initial lags which suggest that the past values have an impact on current values.

```
acf_pacf(pop_change)
```





Based on the above ACF and PACF plots for Population Change:

- The lags in the ACF plot for Population Change also show a decaying pattern and they indicate that the series is non-stationary.
- The PACF plot for Population Change has multiple spikes across different lags which suggests that there is a complex autocorrelation present

```
adf.test(quaterly_ppi)
```

```
## 
##  Augmented Dickey-Fuller Test
## 
## data:  quaterly_ppi
## Dickey-Fuller = -1.3264, Lag order = 3, p-value = 0.8458
## alternative hypothesis: stationary
```

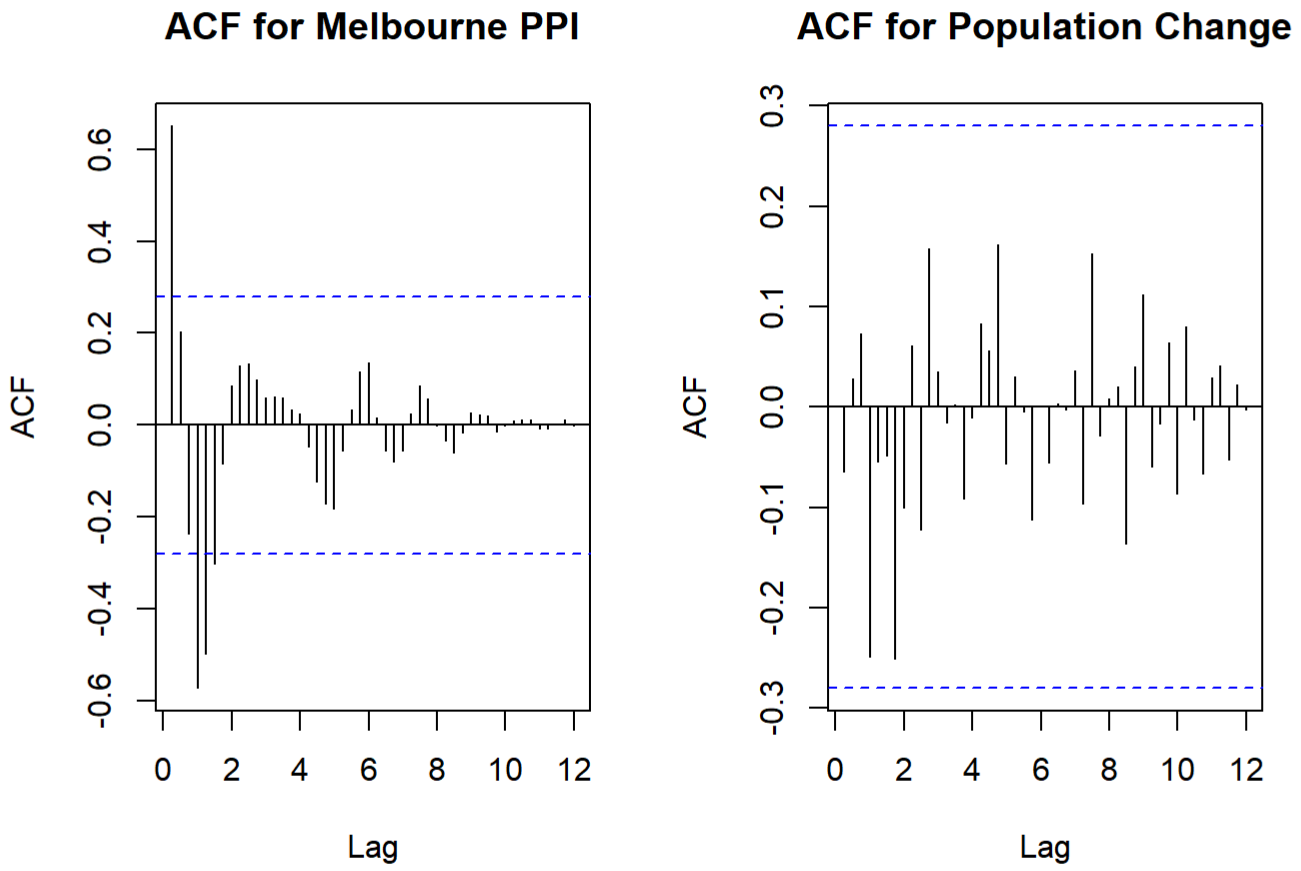
```
adf.test(pop_change)
```

```
## 
##  Augmented Dickey-Fuller Test
## 
## data:  pop_change
## Dickey-Fuller = -1.603, Lag order = 3, p-value = 0.7344
## alternative hypothesis: stationary
```

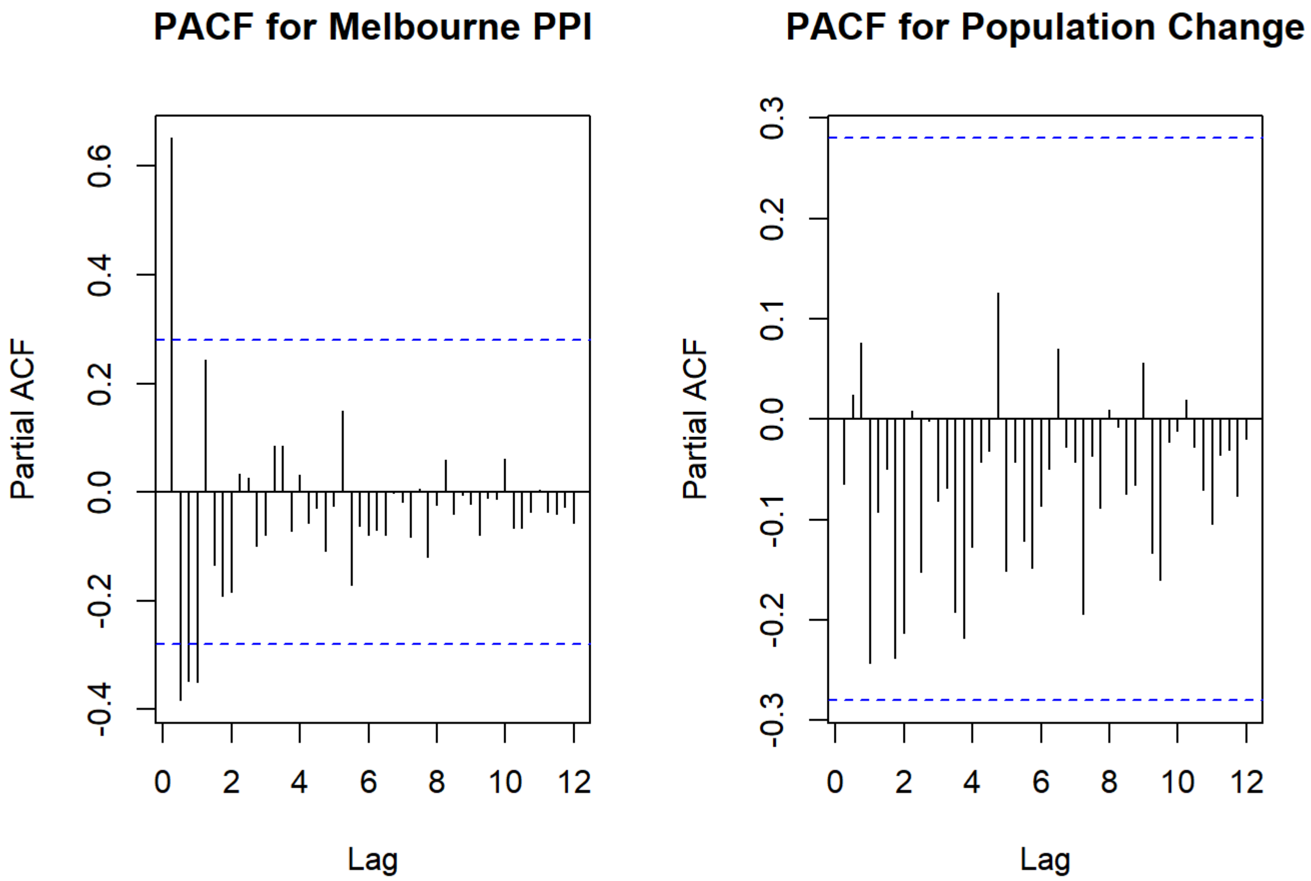
Based on the above ADF tests for both the time series, the p-value is 0.8458 and 0.7344 which are both greater than the significant threshold value i.e. >0.05 and therefore we can conclude that both the series are non-stationary and hence we need to make them stationary before applying the prewhitening approach to them.

```
difference <- ts.intersect(diff(diff(quaterly_ppi,4)), diff(diff(pop_change,4)))
```

```
par(mfrow=c(1,2))
acf(difference[,1],lag.max=70,main='ACF for Melbourne PPI')
acf(difference[,2],lag.max=70,main='ACF for Population Change')
```



```
par(mfrow=c(1,2))
pacf(difference[,1],lag.max=70,main='PACF for Melbourne PPI')
pacf(difference[,2],lag.max=70,main='PACF for Population Change')
```



```
adf.test(difference[,1])
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  difference[, 1]
## Dickey-Fuller = -5.1122, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

```
adf.test(difference[,2])
```

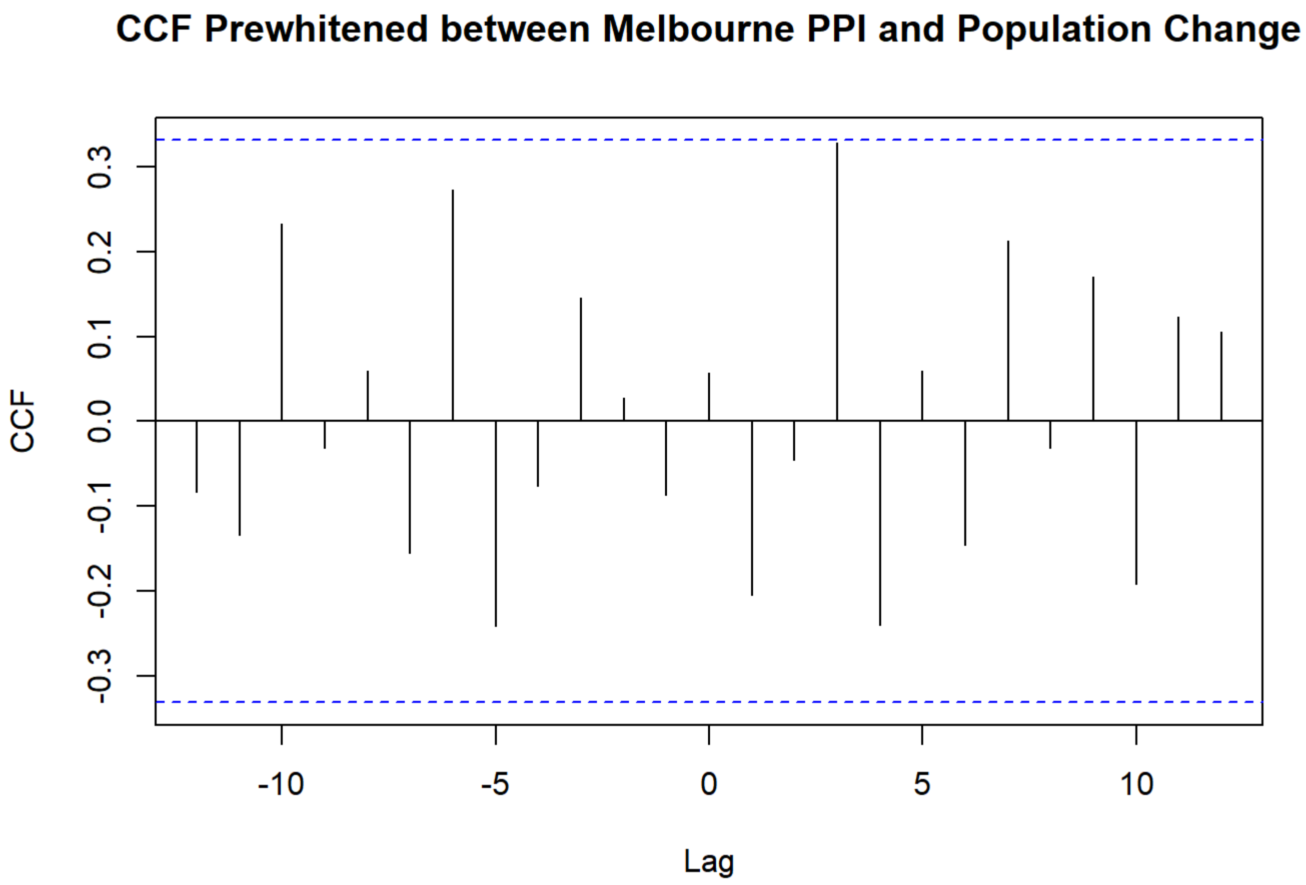
```
##
##  Augmented Dickey-Fuller Test
##
## data:  difference[, 2]
## Dickey-Fuller = -3.8985, Lag order = 3, p-value = 0.02136
## alternative hypothesis: stationary
```

In order to remove the non-stationarity presence in both the time series, I applied the seasonal differencing for both to ensure that there is consistency among both of them. Therefore, based on the above ADF results it can be concluded that both the series are stationary as the p-values are 0.01 and 0.02136 which is less than the significant threshold i.e. <0.05.

Now, as both the time series are stationary we can move ahead with the prewhitening process to differentiate the relationship between the series from their autocorrelation.

Prewhitening

```
prewhiten(as.vector(difference[,1]), as.vector(difference[,2]), ylab='CCF', main = "CCF Prewhitened between Melbourne PPI and Population Change")
```



After performing prewhitening, the above CCF plot indicates that there is no significant correlation between the Melbourne PPI and Population change. This suggests that the link in the data was probably misleading due to nonstationarity present.

Conclusion

By performing Task 1, most of the models that were used to forecast the solar radiation for the next 2 years were unable to capture seasonality and autocorrelation which was present in the data. However 3 models performed well out of which the Holt-Winters multiplicative method worked best as it gave the lowest MASE. This suggests that the solar radiation will decrease but the confidence interval suggests that the forecast are not that much reliable.

For Task 2, upon visualizing the initial plots, both the Melbourne PPI and Population Change indicated upward trends and their correlation indicated that both of them were related to each other. However, upon making the time series stationary and then using prewhitening, that correlation turned out to be spurious.