Introduction

This report concerns the implementations of two data analysis tasks with two separate datasets. The tasks to be carried out are detailed below:

- **Task 1:** Analysing and forecasting the amount of solar radiation received by a specific location by providing the optimal 2 years ahead forecast in terms of MASE. The models to be used are time series regression models, exponential smoothing, and state-space models. **Dynamic linear models were not explored in this report since there was no obvious intervention point in the time series for either variable.**
- **Task 2:** Analysing the correlation between quarterly Residential Property Price Index (PPI) in Melbourne and quarterly population change over the previous quarter in Victoria for the period between September 2003 and December 2016. The objective of the task is to determine whether the correlation between the two series is spurious.

Task 1

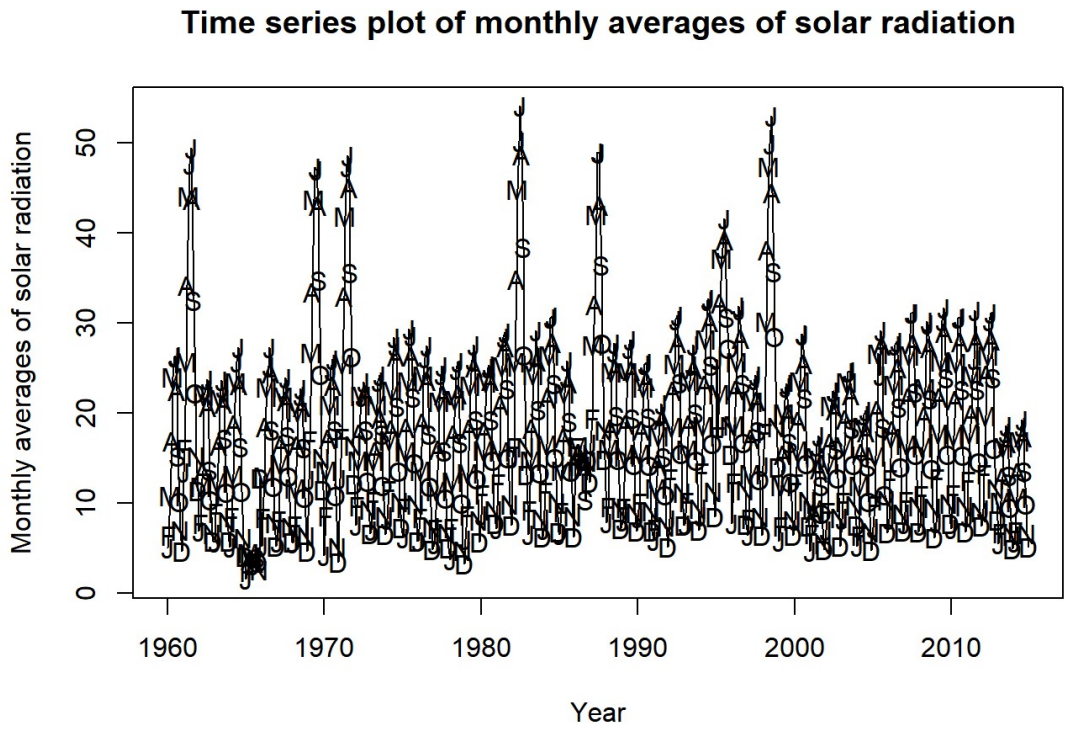
Reading Data and Pre-processing

The dataset is read in and each variable is converted to a time series object. Both solar radiation and precipitation series are plotted along with points denoting each month in order to identify any seasonality. The plots for both solar radiation and precipitation demonstrate seasonal peaks and seem to be roughly stationary since the variance and mean don't seem to change much over time.

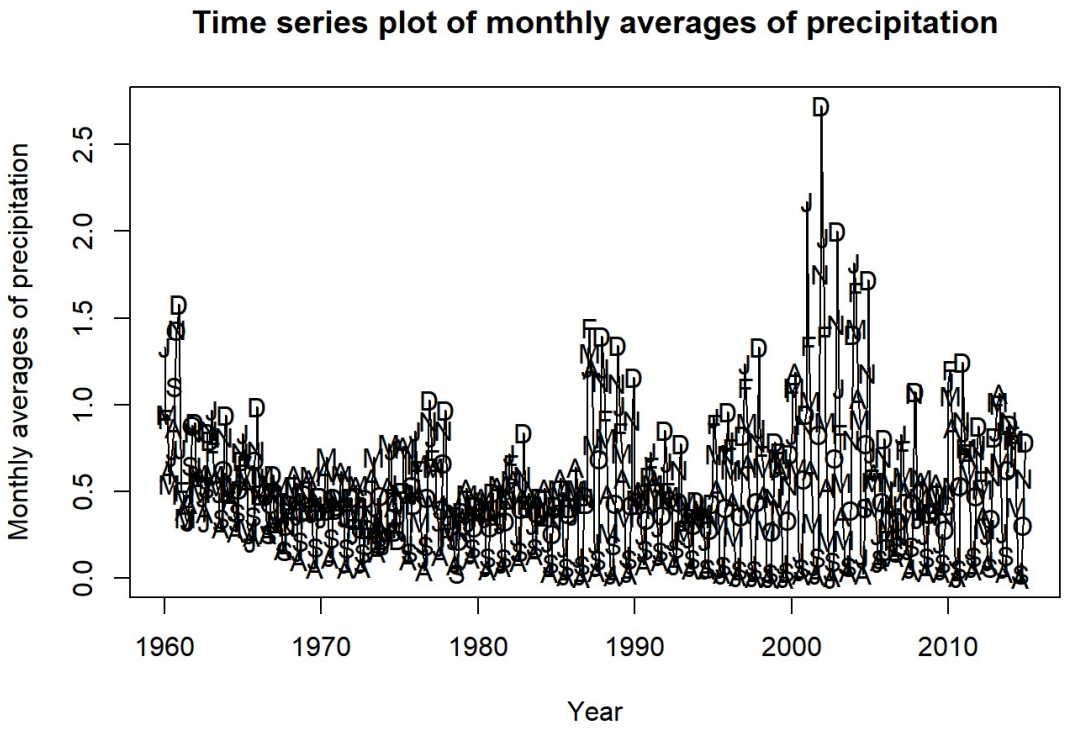
```
setwd("C:/Work/Master in Analytics/Semester 2/Forecasting/Forecasting Assignment 2")
solardata <- read.csv("data1.csv")
head(solardata)
```

```
##       solar   ppt
## 1  5.051729 1.333
## 2  6.415832 0.921
## 3 10.847920 0.947
## 4 16.930264 0.615
## 5 24.030797 0.544
## 6 26.298202 0.703
```

```
solar.ts <- ts(solardata$solar, start=c(1960,1), frequency=12)
ppt.ts <- ts(solardata$ppt, start=c(1960,1), frequency=12)


plot(solar.ts, ylab='Monthly averages of solar radiation',xlab='Year',
     main = "Time series plot of monthly averages of solar radiation")
points(y=solar.ts, x=time(solar.ts), pch=as.vector(season(solar.ts)))
```



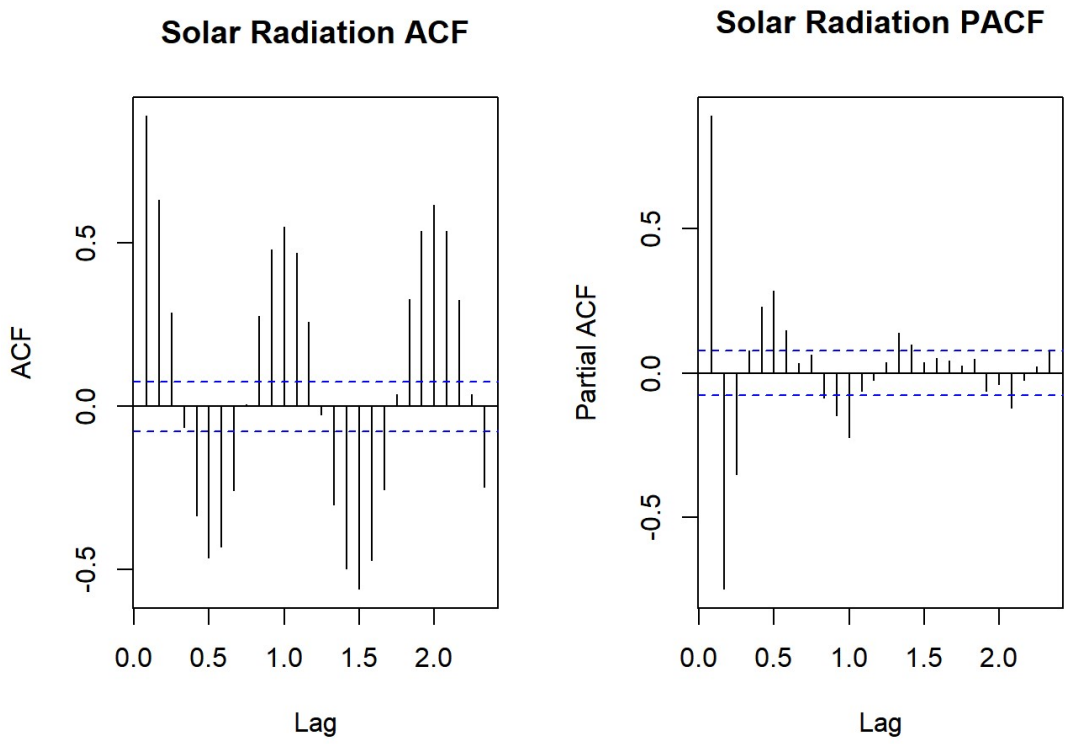

```
plot(ppt.ts, ylab='Monthly averages of precipitation',xlab='Year',
     main = "Time series plot of monthly averages of precipitation")
points(y=ppt.ts, x=time(ppt.ts), pch=as.vector(season(ppt.ts)))
```


Time series plot of monthly averages of precipitation
Monthly averages of precipitation
0.0
0.5
1.0
1.5
2.0
2.5
1960
1970
1980
1990
2000
2010
Year

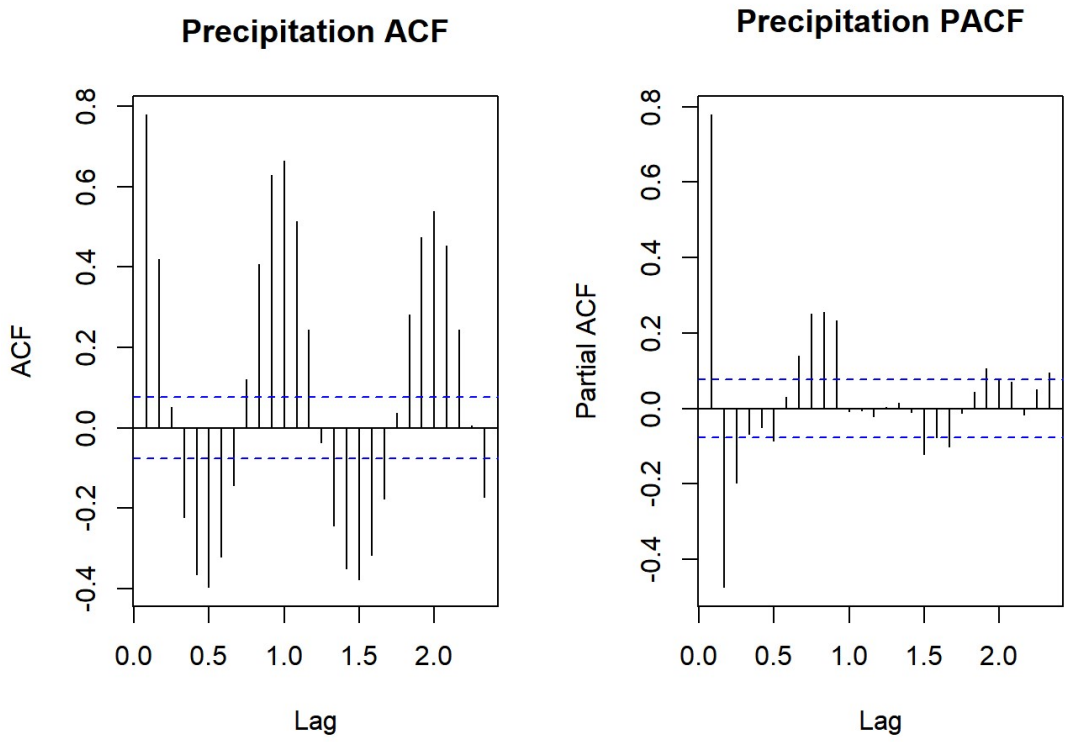
Exploring Stationarity of Variables

Plotting the ACF and PACF for solar radiation and precipitation demonstrates similar results for both variables. Both variables demonstrate an ACF with a periodic rise and fall, with most peaks lying above the 95% confidence interval, and the PACF plot shows multiple significant lags. This suggests that the series is non-stationary.

```
par(mfrow=c(1,2))
acf(solar.ts, max.lag = 24, main="Solar Radiation ACF")
pacf(solar.ts, max.lag = 24, main = "Solar Radiation PACF")
```




```
par(mfrow=c(1,2))
acf(ppt.ts, max.lag = 24, main="Precipitation ACF")
pacf(ppt.ts, max.lag = 24, main = "Precipitation PACF")
```



For further confirmation of non-stationarity in the series, an Augmented Dicky-Fuller test is performed on the solar data series. The absolute value of the test statistic is higher than the absolute of the critical values at 1%, 5% and 10% significance levels. The p-value is quite low (2.2e-16), all suggesting that the series is stationary due to the null hypothesis being rejected.

Further confirmation can be obtained using a Phillips-Perron unit root test in addtion to the ADF test. The Phillips-Perron test produces a Z-tau test statistic of -8.4301 which is more extreme than the critical values at 1%, 5%, and 10% levels. This implies a rejection of the null hypothesis once again, and suggests that the series is indeed stationary.

Testing precipitation for non-stationarity using an Augmented Dicky-Fuller test and a Phillips-Perron unit root test produces test statistics of -7.6 and 7.4 respectively, which are both more extreme than their given critical values at 1%, 5% and 10% significance levels. Therefore, it is possible to confirm that the series is stationary.

```
adf.solar = ur.df(solar.ts, type = "none", lags = 1, selectlags = "AIC")
summary(adf.solar)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.4027  -1.3527   0.5471   2.6802  19.7515
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## z.lag.1    -0.042790   0.006363  -6.725 3.82e-11 ***
## z.diff.lag  0.689323   0.028286  24.369  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.299 on 656 degrees of freedom
## Multiple R-squared:  0.4818, Adjusted R-squared:  0.4802
## F-statistic:   305 on 2 and 656 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -6.7251
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

```
pp.solar = ur.pp(solar.ts, type = "Z-tau", lags = "short")
summary(pp.solar)
```

```
##
## ####################################
## # Phillips-Perron Unit Root Test #
## ####################################
##
## Test regression with intercept
##
##
## Call:
## lm(formula = y ~ y.l1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.6478 -3.7383  0.3732  3.3766 16.4265
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.9384     0.3598   5.388 9.92e-08 ***
## y.l1          0.8911     0.0177  50.350  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.453 on 657 degrees of freedom
## Multiple R-squared:  0.7942, Adjusted R-squared:  0.7939
## F-statistic:  2535 on 1 and 657 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic, type: Z-tau  is: -8.4301
##
##          aux. Z statistics
## Z-tau-mu            7.3858
##
## Critical values for Z statistics:
##                      1pct      5pct     10pct
## critical values -3.442671 -2.866274 -2.569292
```

```
adf.ppt = ur.df(ppt.ts, type = "none", lags = 1, selectlags = "AIC")
summary(adf.ppt)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression none
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.05939 -0.04034  0.04809  0.14327  1.30427
##
## Coefficients:
##            Estimate Std. Error t value Pr(>|t|)
## z.lag.1    -0.10907    0.01434  -7.607 9.78e-14 ***
## z.diff.lag  0.38211    0.03593  10.635  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2113 on 656 degrees of freedom
## Multiple R-squared:  0.1813, Adjusted R-squared:  0.1788
## F-statistic: 72.65 on 2 and 656 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -7.6069
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau1 -2.58 -1.95 -1.62
```

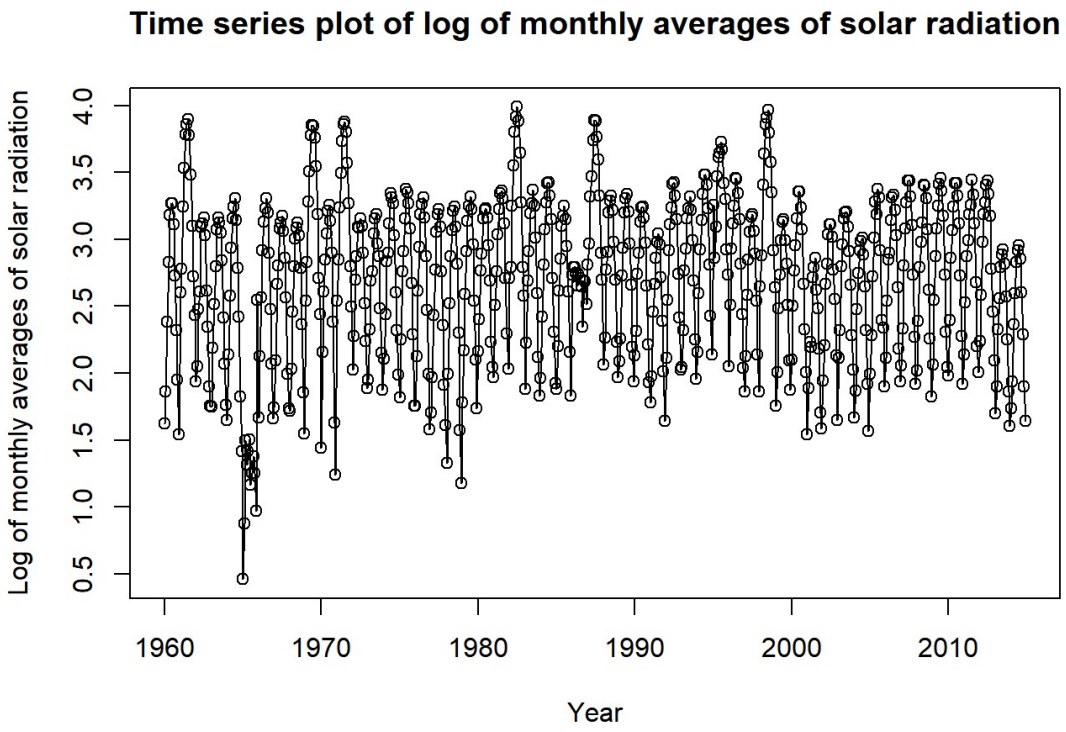
```
pp.ppt = ur.pp(ppt.ts, type = "Z-tau", lags = "short")
summary(pp.ppt)
```

```
##
## ###################################
## # Phillips-Perron Unit Root Test #
## ###################################
##
## Test regression with intercept
##
##
## Call:
## lm(formula = y ~ y.l1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.71989 -0.12860 -0.02614  0.09731  1.33078
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.10248    0.01422   7.209 1.55e-12 ***
## y.l1         0.77928    0.02420  32.195  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2204 on 657 degrees of freedom
## Multiple R-squared:  0.6121, Adjusted R-squared:  0.6115
## F-statistic:  1037 on 1 and 657 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic, type: Z-tau  is: -9.3544
##
##          aux. Z statistics
## Z-tau-mu             7.399
##
## Critical values for Z statistics:
##                      1pct      5pct     10pct
## critical values -3.442671 -2.866274 -2.569292
```

Applying Transformations to Solar Raidation

Log transformations are performed to the solar radiation series and the precipitation series to see if any improvements occur to the adf test. This produces a p-value of 0.01 for both solar radiation and precipitation.

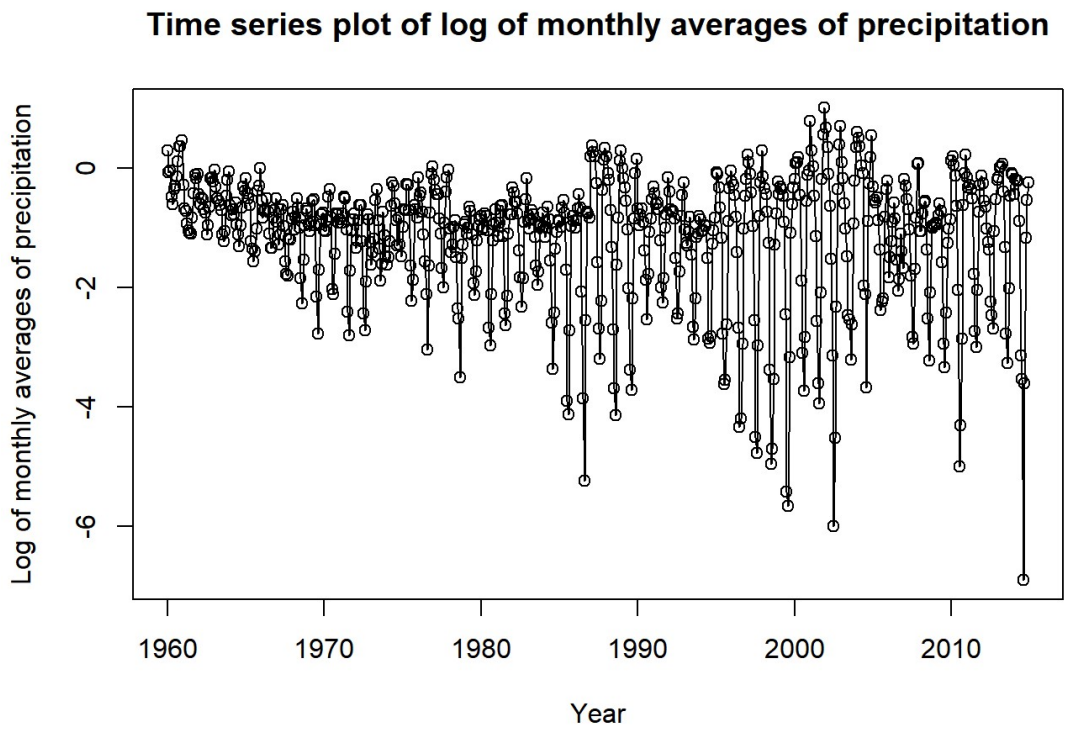
```
# Trying a log transformation to make the series stationary
log_solar = log(solar.ts)
plot(log_solar,ylab='Log of monthly averages of solar radiation',xlab='Year',type='o',
     main = "Time series plot of log of monthly averages of solar radiation")
```




```
adf.test(log_solar)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  log_solar
## Dickey-Fuller = -4.6658, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
```

```
log_ppt = log(ppt.ts)
plot(log_ppt,ylab='Log of monthly averages of precipitation',xlab='Year',type='o',
     main = "Time series plot of log of monthly averages of precipitation")
```




```
adf.test(log_ppt)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  log_ppt
## Dickey-Fuller = -10.948, Lag order = 8, p-value = 0.01
## alternative hypothesis: stationary
```

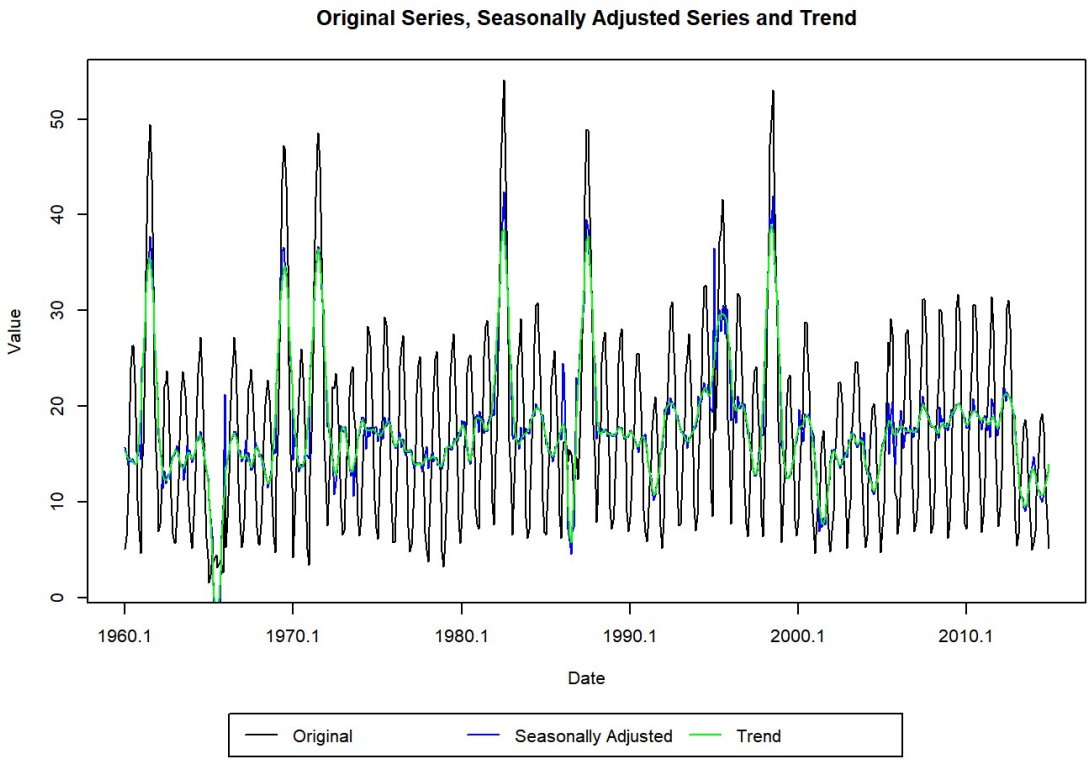
Verifying Seasonality

The seasonal components of the solar radiation series are explored using the X-12 ARIMA and stl functions. THe X-12 ARIMA function produces a plot with a seasonally adjusted series which has much less variation compared to the original series, suggesting that there are indeed seasonal components in the series. The results of the STL function show a sizable seasonal component with a rise and fall roughly occuring every year.

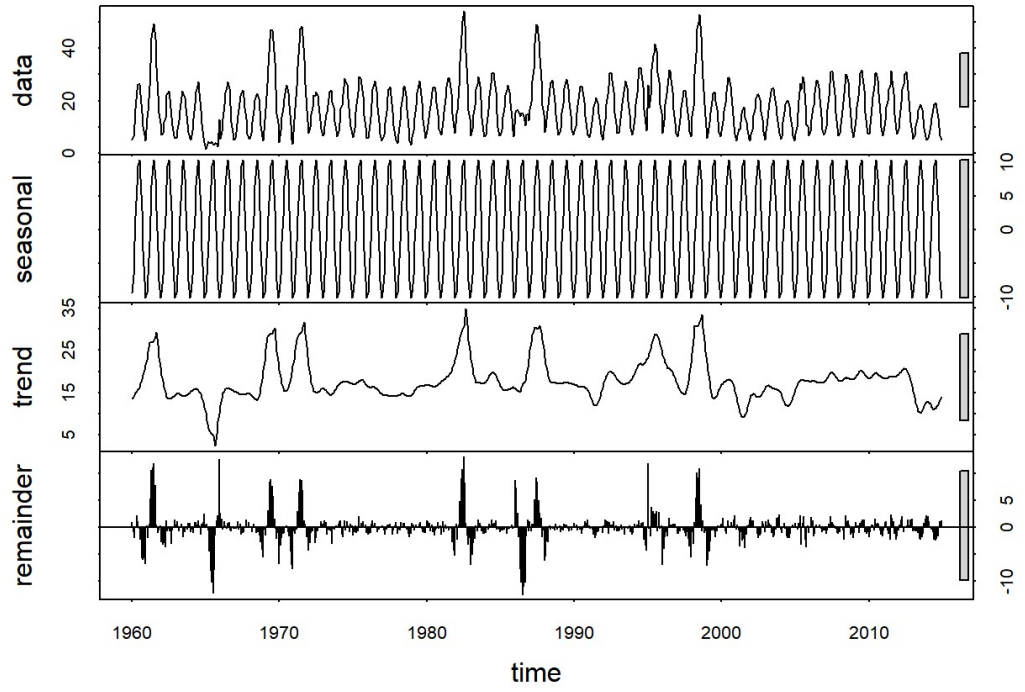
This is further confirmed by using the seasadj() function to obtain a seasonally adjusted series for solar radiation, which flattens out the periodic components of the series.

Using the X-12 ARIMA function on the precipitation series yields similar results, where the seasonally adjusted series looks much flatter compared to the original. The STL function decomposition reveals a seasonal component here as well, and the plot obtained using the seasadj() function displays a similar result to earlier with the periodic components of the series being flattened out.

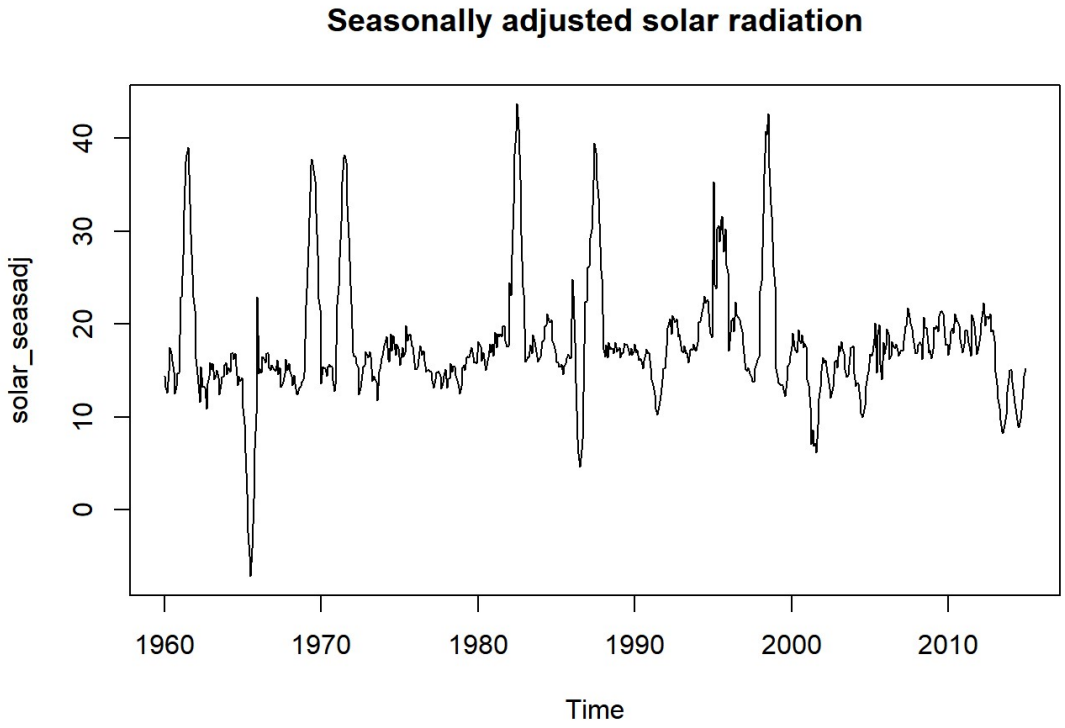
```
# solar: decomposing and adjusting seasonality

decomp_solar_x12 = x12(solar.ts)
plot(decomp_solar_x12, sa=TRUE, trend=TRUE)
```



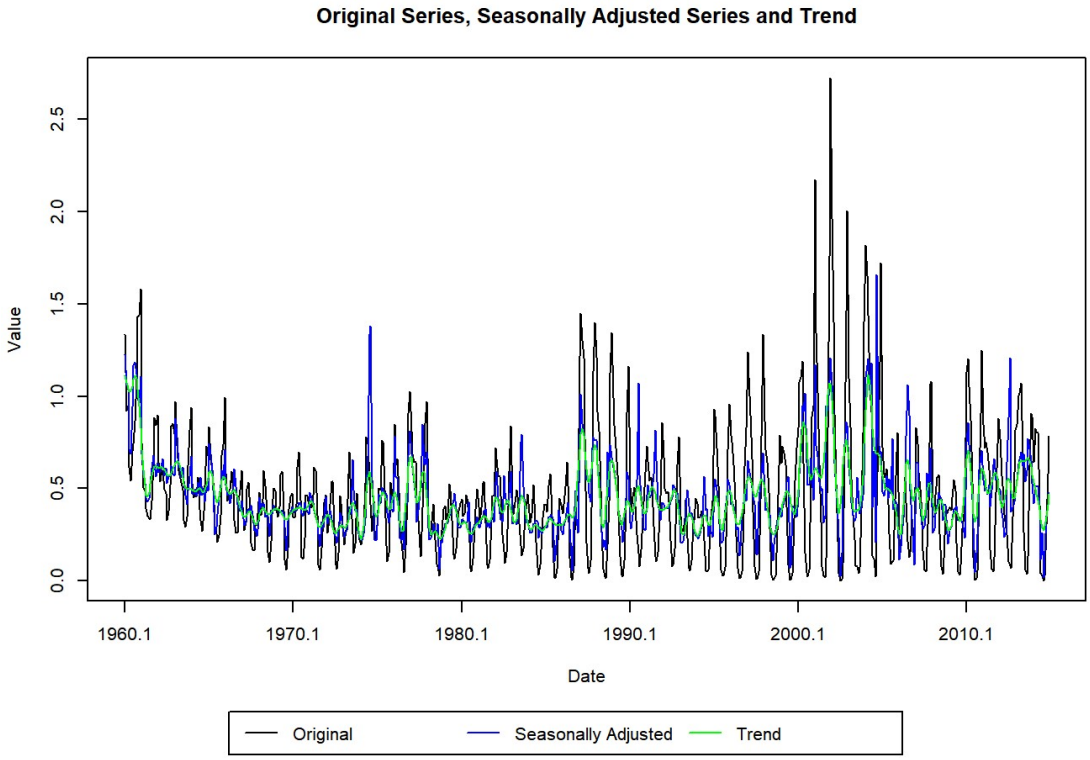

```
decomp_solar = stl(solar.ts, t.window = 12, s.window = "periodic", robust=TRUE)
plot(decomp_solar)
```



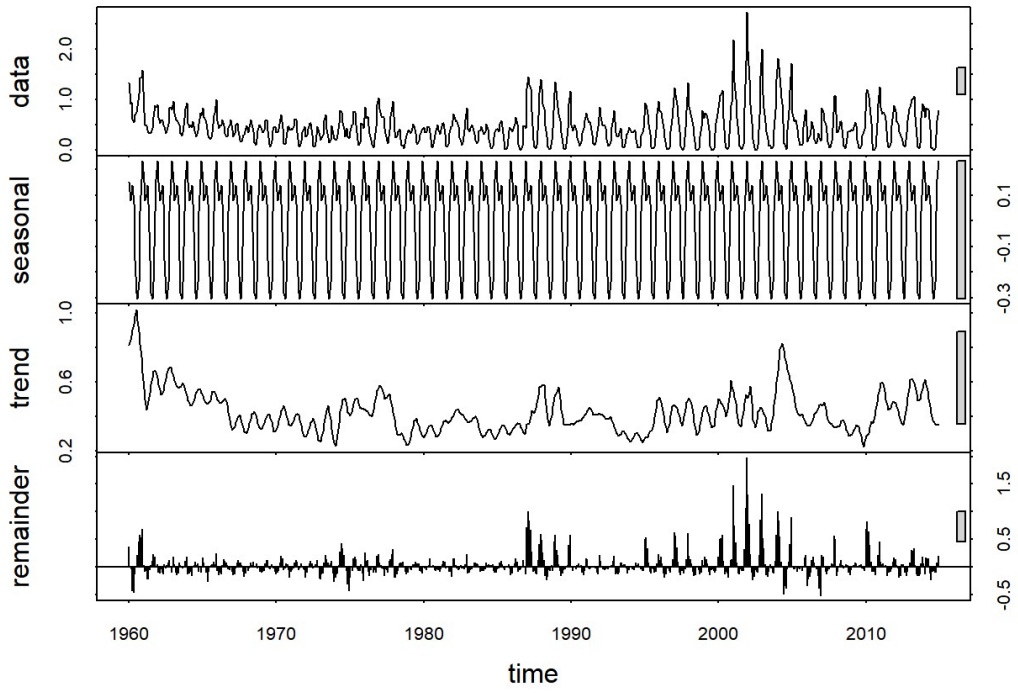

```
solar_seasadj = seasadj(decomp_solar)
plot(solar_seasadj, main = "Seasonally adjusted solar radiation")
```



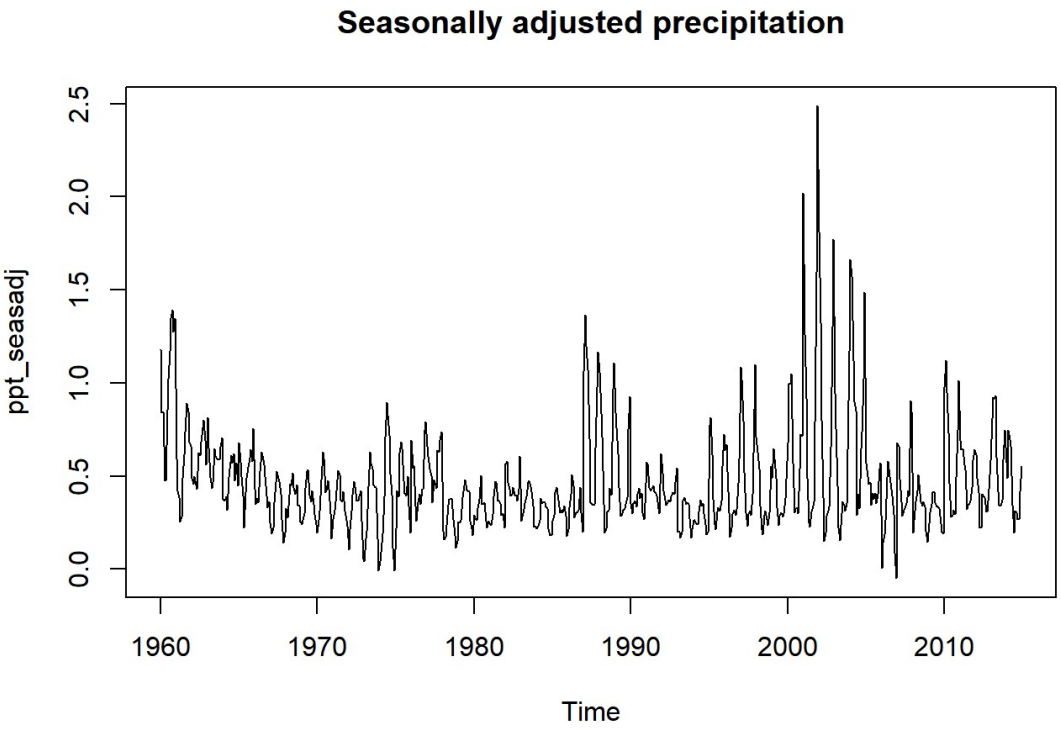

```
# precipitation: decomposing and adjusting seasonality

decomp_ppt_x12 = x12(ppt.ts)
plot(decomp_ppt_x12, sa=TRUE, trend=TRUE)
```




```
decomp_ppt = stl(ppt.ts, t.window = 12, s.window = "periodic", robust=TRUE)
plot(decomp_ppt)
```




```
ppt_seasadj = seasadj(decomp_ppt)
plot(ppt_seasadj, main = "Seasonally adjusted precipitation")
```

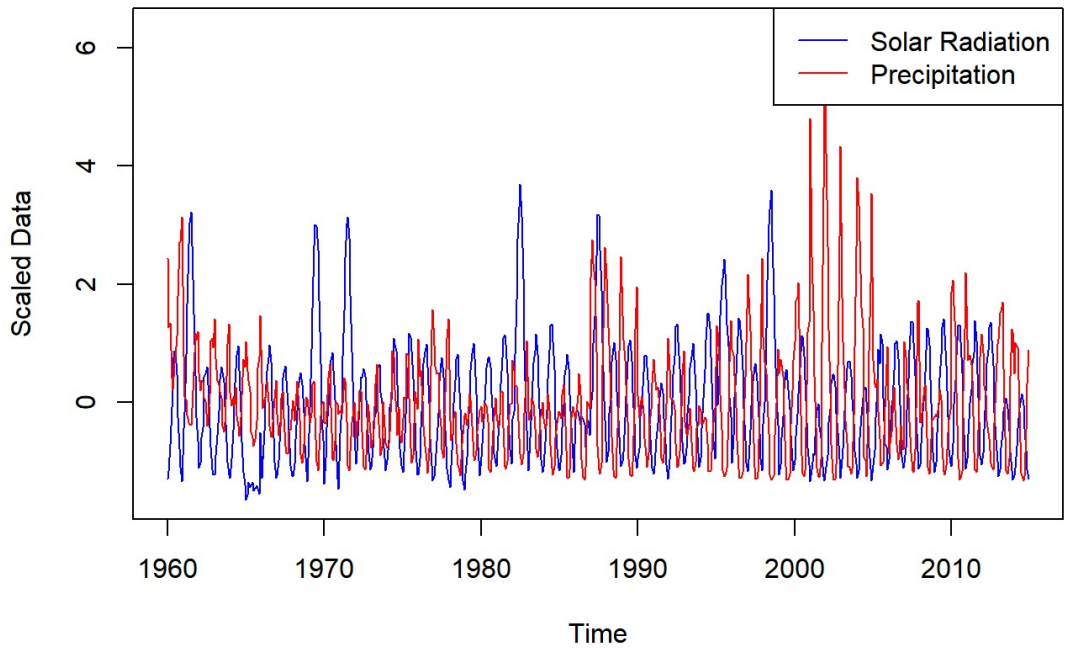

Seasonally adjusted precipitation
ppt_seasadj
0.0
0.5
1.0
1.5
2.0
2.5
1960
1970
1980
1990
2000
2010
Time

Correlating Predictor Variables with Dependent Variable

Since precipitation is to be used as a predictor for solar radiation, it would be useful to determine the correlation between the two series. This may be achieved visually by plotting the scaled series together on the same plot, so the solar radiation series is scaled and plotted along with the scaled precipitation series.

Close inspection of the plot reveals that peaks and troughs in the solar radiation series are immediately preceded by similar peaks and troughs in the precipitation series. This seems to imply that precipitation does indeed predict the future observations for solar radiation and is a suitable predictor variable.

```
solardata.ts <- ts(solardata, start=c(1960,1), frequency=12)
solardata_scaled = scale(solardata.ts)
plot(solardata_scaled, plot.type="s", col=c("blue", "red"), ylab="Scaled Data")
legend("topright", lty=1, col=c("blue", "red"), c("Solar Radiation", "Precipitation"))
```



Since a visual confirmation of the correlation between predictor and dependent variables is insufficient, the variables can be correlated using the cor() function in order to obtain a numerical representation of correlation. Obtaining the correlation values shows that precipitation and solar radiation have a correlation value of -0.45. The negative correlation implies that whenever there is precipitation, there is less solar radiation and vice-versa, which is a sensible conclusion.

```
#correlate variables
cor(solardata.ts)
```

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

Fitting Distributed Lag Models

Polynomial DLM

A polynomial DLM is first fitted using the polyDlm() function, using precipitation as a predictor variable to predict solar radiation. Prior to fitting, the ideal lag value is found using the finiteDLMauto() function, which is made to generate multiple models with q ranging from 1 to 10 and uses AIC error to determine the suitability of each model. The results of finiteDLMauto() suggest that the ideal value for q is 10, and a polynomial order of 2.

```
#PolyDLM
finiteDLMauto(x = as.vector(ppt.ts), y = as.vector(solar.ts), q.min = 1, q.max = 10,
              model.type = "poly", error.type = "AIC", trace = TRUE)
```

```
##     q - k    MASE      AIC      BIC   GMRAE     MBRAE R.Adj.Sq Ljung-Box
## 10 10 - 2 1.59255 4591.904 4614.289 1.40716  -0.02753  0.29924         0
## 9   9 - 2 1.60880 4607.861 4630.253 1.42573   0.36108  0.28915         0
## 8   8 - 2 1.61903 4620.470 4642.870 1.38665   0.27726  0.28205         0
## 7   7 - 2 1.61909 4627.974 4650.382 1.38654   0.56541  0.28073         0
## 6   6 - 2 1.61999 4634.526 4656.942 1.38760   0.34815  0.28104         0
## 5   5 - 2 1.63194 4645.250 4667.673 1.43173 -91.88214  0.27675         0
## 4   4 - 2 1.65329 4664.741 4687.171 1.47467   0.97378  0.26226         0
## 3   3 - 2 1.66635 4689.018 4711.457 1.45895   0.29153  0.24157         0
## 2   2 - 2 1.67597 4712.649 4735.095 1.42249   0.14593  0.22173         0
## 1   1 - 2 1.68846 4728.713 4746.676 1.43103  -1.07642  0.21036         0
```

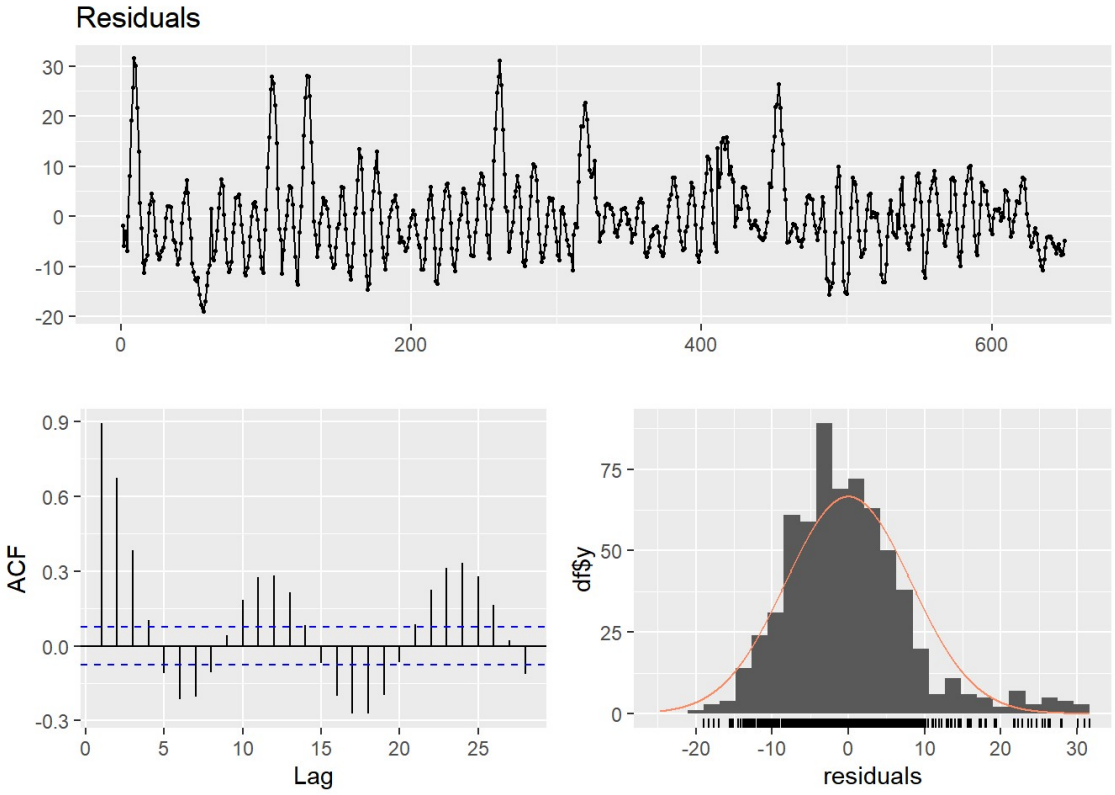
```
model.poly = polyDlm(x = as.vector(ppt.ts), y = as.vector(solar.ts), q = 10, k = 2, show.beta = TRUE)
```

```
## Estimates and t-tests for beta coefficients:
##         Estimate Std. Error t value  P(>|t|)
## beta.0    -5.040      0.435 -11.600 2.39e-28
## beta.1    -2.320      0.314  -7.400 4.33e-13
## beta.2    -0.188      0.251  -0.749 4.54e-01
## beta.3     1.370      0.237   5.770 1.23e-08
## beta.4     2.340      0.243   9.600 1.74e-20
## beta.5     2.720      0.248  11.000 7.32e-26
## beta.6     2.530      0.243  10.400 1.40e-23
## beta.7     1.750      0.235   7.430 3.39e-13
## beta.8     0.387      0.249   1.560 1.20e-01
## beta.9    -1.560      0.312  -4.990 7.68e-07
## beta.10   -4.090      0.433  -9.440 7.08e-20
```

```
summary(model.poly)
```

```
##
## Call:
## "Y ~ (Intercept) + X.t"
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.012  -5.343  -1.090   4.097  31.641
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 18.77806    1.08588   17.29   <2e-16 ***
## z.t0        -5.04398    0.43502  -11.60   <2e-16 ***
## z.t1         3.01112    0.18300   16.45   <2e-16 ***
## z.t2        -0.29153    0.01761  -16.56   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.237 on 646 degrees of freedom
## Multiple R-squared:  0.3025, Adjusted R-squared:  0.2992
## F-statistic: 93.38 on 3 and 646 DF,  p-value: < 2.2e-16
```

```
checkresiduals(model.poly$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  Residuals
## LM test = 580.37, df = 10, p-value < 2.2e-16
```

```
MASE(model.poly)
```

```
##                MASE
## model.poly 1.592555
```

The summary of the polynomial DLM model shows a low p-value of 2.2e-16, which suggests that

the model is significant. However, a low R-squared value of 0.2992 means that the model does not explain the data well. Additionally, the z values are significant for t, as well as the first and second lags for t.

The Breusch-Godfrey test for serial correlation produces a low p-value of 2.2e-16, which means that there is evidence of serial correlation in the model, making it less suitable to model solar radiation.

Analysing the residuals for the polynomial model shows that the residuals are relatively large and are not normally distributed due to the residuals being somewhat right skewed, suggesting that the model does not account for the data well and the errors in the model are not random. The ACF plot also shows peaks above the 95% confidence interval, suggesting that this is an inadequate model to describe solar radiation.

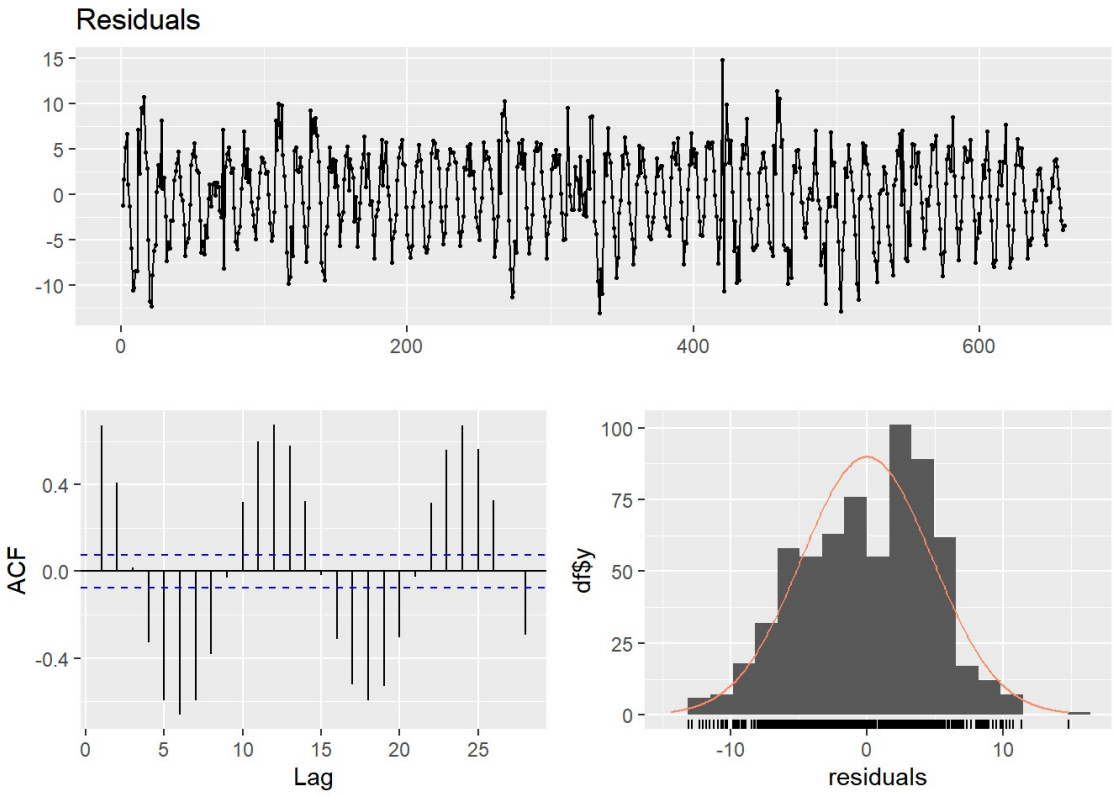
The MASE value for the polynomial model is 1.59, which is unsatisfactory since the fact that it is larger than 1 implies that it performs worse than a naive forecast in terms of mean absolute error.

Koyck (Geometric) DLM

```
# Kocyk
model.Koyck = koyckDlm(x = as.vector(ppt.ts), y = as.vector(solar.ts), intercept = TRUE)
summary(model.Koyck)
```

```
##
## 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 ***
## ---
## 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
##
## Diagnostic tests:
## NULL
##
##                              alpha     beta       phi
## Geometric coefficients:  -154.0203 5.346844 0.9854613
```

```
checkresiduals(model.Koyck$model)
```




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

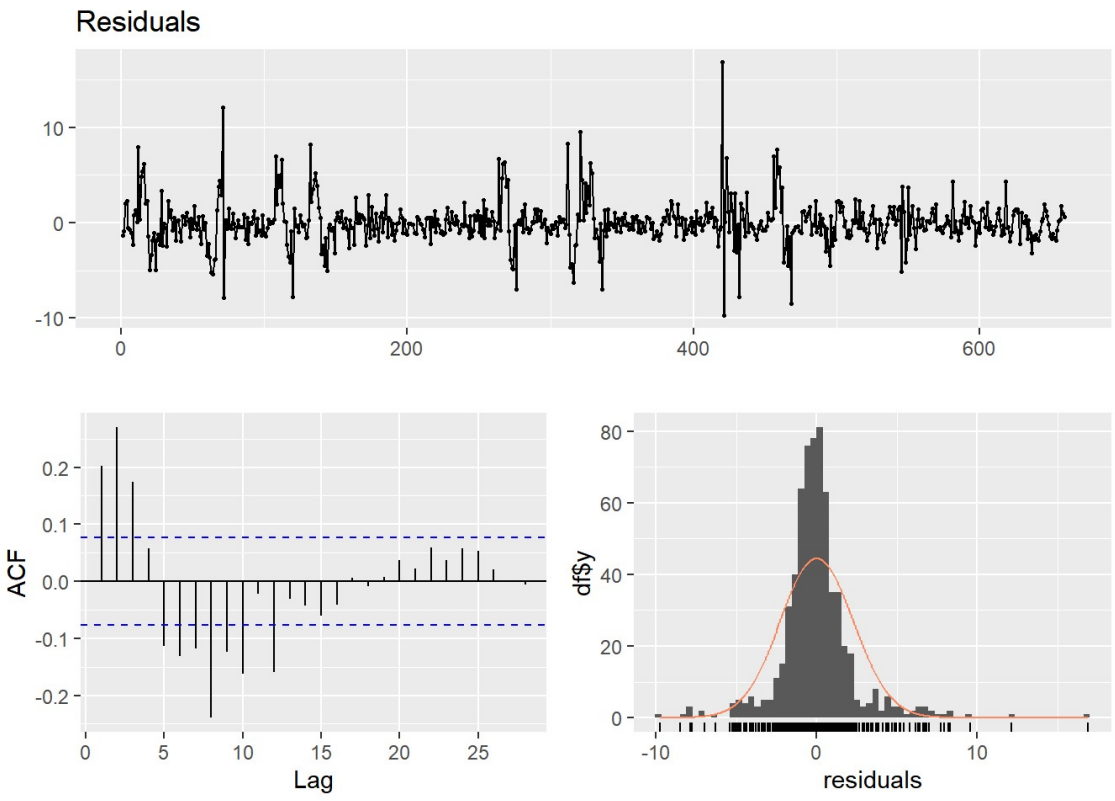
```
MASE(model.Koyck)
```

```
##                 MASE
## model.Koyck 1.032483
```

```
model.Koyck.adj = koyckDlm(x = as.vector(ppt_seasadj), y = as.vector(solar_seasadj), intercept = TRUE)
summary(model.Koyck.adj)
```

```
##
## Call:
## "Y ~ (Intercept) + Y.1 + X.t"
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.7507 -0.9835 -0.1133  0.7146 16.8996
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.37677    0.34297   4.014 6.65e-05 ***
## Y.1          0.93118    0.01418  65.675  < 2e-16 ***
## X.t         -0.32316    0.42273  -0.764    0.445
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.286 on 656 degrees of freedom
## Multiple R-Squared: 0.8688,  Adjusted R-squared: 0.8684
## Wald test:  2171 on 2 and 656 DF,  p-value: < 2.2e-16
##
## Diagnostic tests:
## NULL
##
##                             alpha       beta       phi
## Geometric coefficients:  20.00426 -0.3231641 0.9311762
```

```
checkresiduals(model.Koyck.adj$model)
```




```
##
##  Ljung-Box test
##
## data:  Residuals
## Q* = 193.13, df = 10, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 10
```

```
MASE(model.Koyck.adj)
```

```
##                      MASE
## model.Koyck.adj 0.9799924
```

A Koyck DLM is fitted using the koyckDlm() function, using precipitation as a predictor variable to predict solar radiation. The summary of the Koyck model shows a very low p-value of 2.2e-16, which suggests that the model is significant. A decent R-squared value of 0.7591 reveals that the model explains a good proportion of the data well.

The first lag of the Y value is significant and the X value at t are both significant, meaning that both are suitable predictor for solar radiation.

The Ljung-Box test produces a low p-value of 2.2e-16, which means that there is evidence of serial correlation in the model, making it less suitable to model solar radiation.

Analysing the residuals for the Koyck model shows that the residuals are relatively small compared to the polynomial model and are roughly normally distributed, suggesting that the model accounts for the data better than the polynomial model and the errors in the model are random. The ACF plot shows periodic peaks which extend beyond the 95% confidence interval, suggesting that the series is seasonal.

The MASE of the Koyck model was found to be 1.03, which makes it marginally worse than a naive forecast in terms of mean absolute error.

Another Koyck model is created using the seasonally adjusted versions of solar radiation and precipitation. This model produces a low p-value of 2.2e-16 and an even higher R-squared value of 0.8684, which implies that the Koyck model using seasonally adjusted predictor and output explains the series better than earlier. The residuals of the adjusted Koyck model are also normally distributed with lower variance compared to earlier. This supports the theory that this model explains the data better.

The MASE of the adjusted Koyck model was found to be 0.98, which makes it marginally better than a naive forecast in terms of mean absolute error.

The adjusted Koyck model also produced a p-value of 2.2e-16 for the Ljung-Box test, which implies that there is serial correlation in the data.

Finite DLM

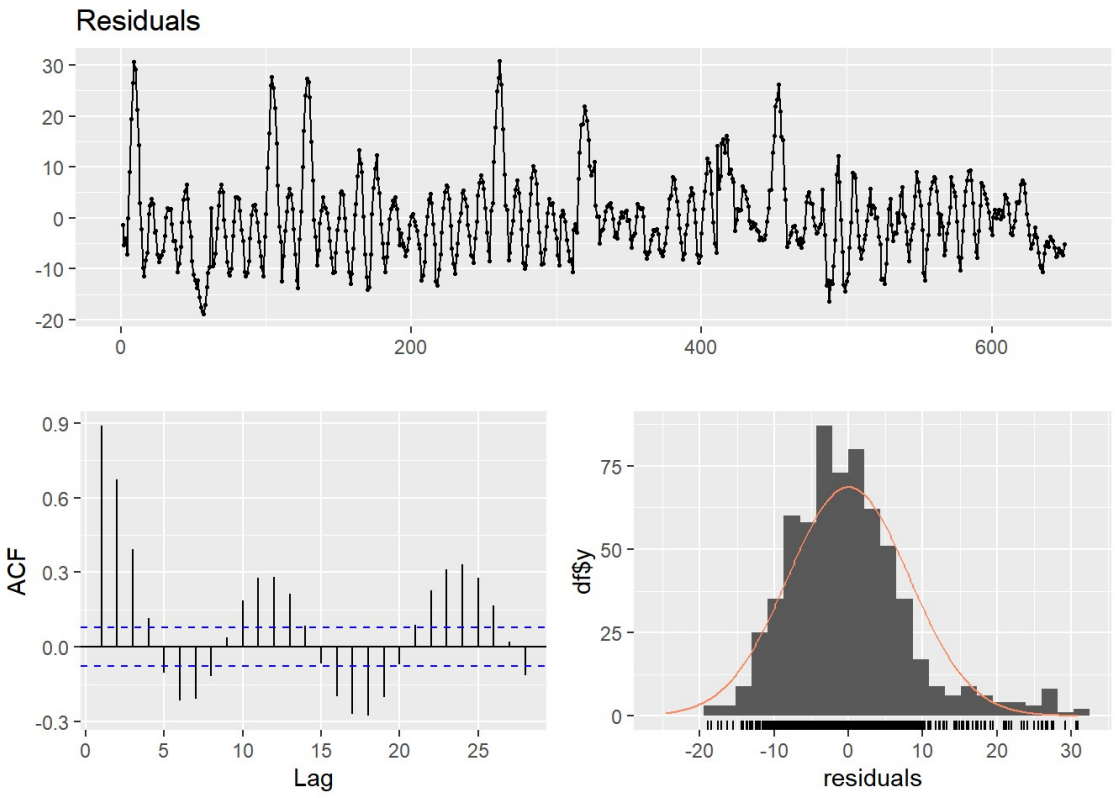
```
# Finite DLM
finiteDLMauto(x = as.vector(ppt.ts), y = as.vector(solar.ts), q.min = 1, q.max = 10,
              model.type = "dlm", error.type = "AIC", trace = TRUE)
```

```
##    q - k    MASE      AIC      BIC   GMRAE     MBRAE R.Adj.Sq Ljung-Box
## 10    10 1.57800 4602.658 4660.858 1.36315   0.09680  0.29615         0
## 9      9 1.59312 4615.084 4668.827 1.36372   2.68197  0.28882         0
## 8      8 1.60481 4625.986 4675.267 1.39224   1.79485  0.28251         0
## 7      7 1.60704 4632.716 4677.532 1.39082 -60.50677  0.28096         0
## 6      6 1.60753 4637.489 4677.837 1.39182   0.52400  0.28214         0
## 5      5 1.61385 4644.622 4680.499 1.39327   0.55267  0.28072         0
## 4      4 1.64636 4663.600 4695.003 1.46450  -1.04420  0.26577         0
## 3      3 1.66270 4688.551 4715.478 1.44133  -1.39824  0.24326         0
## 2      2 1.67597 4712.649 4735.095 1.42249   0.14593  0.22173         0
## 1      1 1.68846 4728.713 4746.676 1.43103  -1.07642  0.21036         0
```

```
model.dlm = dlm(x = as.vector(ppt.ts), y = as.vector(solar.ts), q = 10)
summary(model.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
```

```
checkresiduals(model.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
```

```
MASE(model.dlm)
```

```
##               MASE
## model.dlm 1.577996
```

A finite DLM is fitted using the dlm() function. Prior to fitting, the ideal lag value is found using the finiteDLMauto() function, which is made to generate multiple models with q ranging from 1 to 10 and uses AIC error to determine the suitability of each model. The results of finiteDLMauto() suggest that the ideal value for q is 10.

The summary of the finite DLM shows a high p-value of 0.2962, which suggests that the model is not significant. Additionally, a low R-squared value of 0.2962 means that the model does not explain the data well. Only x(t) and its 10th lag are significant, meaning that only those two lag components are satisfactory predictors of solar radiation.

The Breusch-Godfrey test for serial correlation produces a low p-value of 2.2e-16, which means that there is evidence of serial correlation in the model, making it less suitable to model solar radiation.

Analysing the residuals for the finite DLM shows that the residuals are relatively large and are not normally distributed since the reesiduals are somewhat right skewed, suggesting that the model does not account for the data well and the errors in the model are not random. The ACF plot also shows peaks above the 95% confidence interval, with the the first few peaks/lags being the largest. This suggests that this is an inadequate model to describe solar radiation.

The MASE of the DLM model is seen to be 1.58, which is almost as unsatisfactory as the polynomial model, meaning that a naive forecast performs better in terms of absolute mean error.

ARDL Model

```
#ARDL
columns = c("p","q","AIC","BIC")
df = data.frame(matrix(nrow = 0, ncol = length(columns)))
colnames(df) = columns

for(i in 1:10){
  for(j in 1:10){
    model.ARDL = ardlDlm(x = as.vector(ppt.ts), y = as.vector(solar.ts), p = i, q = j)
    new_row = data.frame(p = i, q=j, AIC=AIC(model.ARDL$model), BIC=BIC(model.ARDL$model))
    df = bind_rows(df, new_row)
  }
}

df[order(df$AIC),]
```

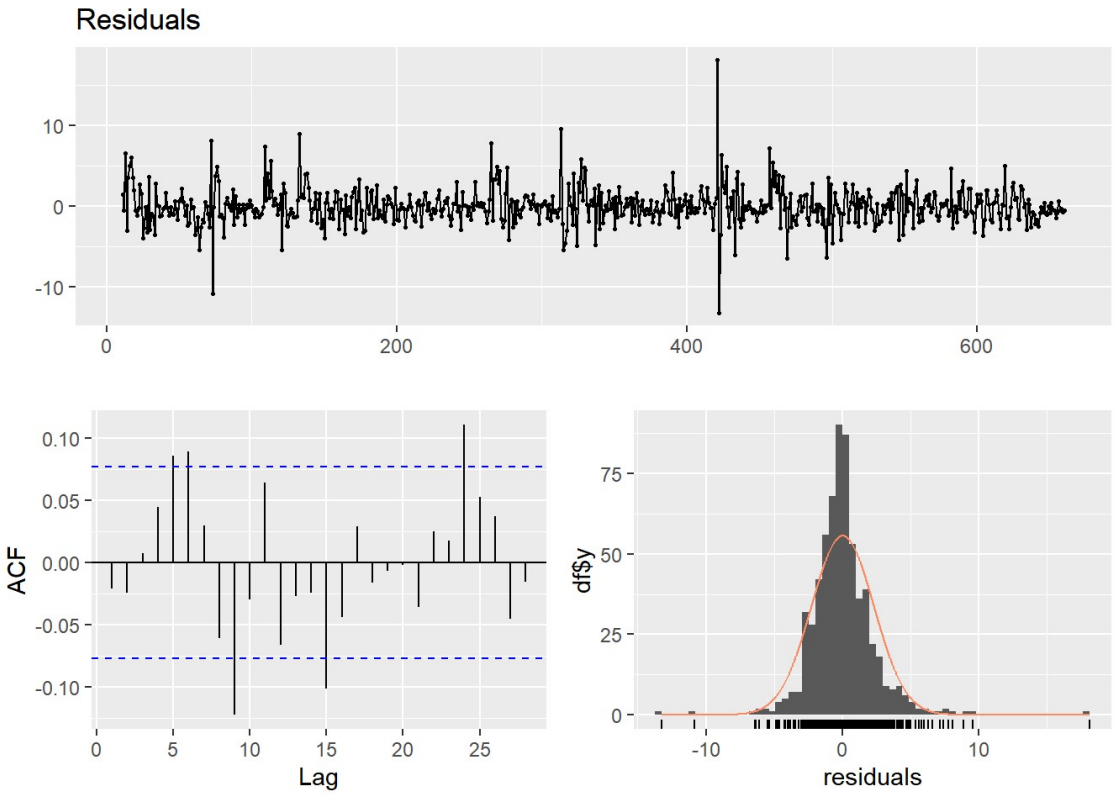
```
##      p  q      AIC      BIC
## 100 10 10 2962.601 3065.571
## 90   9 10 2963.123 3061.616
## 97  10  7 2968.654 3058.193
## 98  10  8 2970.239 3064.256
## 99  10  9 2972.143 3070.636
## 87   9  7 2972.197 3057.289
## 88   9  8 2973.907 3063.477
## 89   9  9 2975.583 3069.631
## 80   8 10 2977.316 3071.333
## 70   7 10 2980.152 3069.691
## 96  10  6 2980.553 3065.615
## 86   9  6 2983.773 3064.386
## 40   4 10 2986.780 3062.889
## 50   5 10 2988.572 3069.158
## 79   8  9 2990.093 3079.664
## 60   6 10 2990.569 3075.631
## 77   8  7 2991.824 3072.465
## 78   8  8 2993.801 3078.922
## 69   7  9 2994.189 3079.280
## 20   2 10 2995.768 3062.922
## 10   1 10 2996.457 3059.135
## 30   3 10 2996.590 3068.221
## 39   4  9 2998.610 3070.266
## 68   7  8 2998.638 3079.278
## 49   5  9 3000.385 3076.520
## 59   6  9 3002.341 3082.954
## 67   7  7 3002.413 3078.600
## 38   4  8 3003.603 3070.804
## 48   5  8 3005.420 3077.101
## 19   2  9 3005.437 3068.136
## 29   3  9 3005.950 3073.127
## 76   8  6 3006.280 3082.441
## 37   4  7 3006.430 3069.173
## 9    1  9 3006.442 3064.663
## 58   6  8 3007.386 3083.546
## 47   5  7 3008.224 3075.448
## 57   6  7 3010.203 3081.908
## 28   3  8 3010.775 3073.495
## 18   2  8 3010.800 3069.040
## 8    1  8 3011.894 3065.654
## 17   2  7 3013.616 3067.395
## 27   3  7 3013.636 3071.897
## 7    1  7 3014.566 3063.864
## 66   7  6 3022.352 3094.058
## 36   4  6 3027.070 3085.350
## 46   5  6 3029.060 3091.824
## 56   6  6 3030.976 3098.223
## 16   2  6 3031.898 3081.212
## 26   3  6 3032.418 3086.216
## 6    1  6 3033.200 3078.031
## 95  10  5 3046.099 3126.684
## 85   9  5 3048.439 3124.574
## 75   8  5 3068.742 3140.423
## 93  10  3 3072.759 3144.390
## 94  10  4 3073.910 3150.018
## 83   9  3 3074.896 3142.074
## 84   9  4 3075.884 3147.540
## 65   7  5 3091.439 3158.663
## 73   8  3 3093.999 3156.719
## 55   6  5 3094.149 3156.912
## 74   8  4 3095.328 3162.529
## 35   4  5 3096.024 3149.839
## 15   2  5 3097.288 3142.134
## 45   5  5 3097.877 3156.177
```

```
## 25   3  5 3098.808 3148.139
## 5    1  5 3100.283 3140.644
## 63   7  3 3117.291 3175.551
## 64   7  4 3118.610 3181.352
## 53   6  3 3119.154 3172.952
## 54   6  4 3120.480 3178.760
## 43   5  3 3127.103 3176.434
## 44   5  4 3127.868 3181.684
## 33   4  3 3131.289 3176.150
## 34   4  4 3131.424 3180.772
## 14   2  4 3132.962 3173.337
## 24   3  4 3134.777 3179.638
## 13   2  3 3137.634 3173.535
## 4    1  4 3138.399 3174.288
## 23   3  3 3139.409 3179.798
## 3    1  3 3143.522 3174.936
## 92  10  2 3165.653 3232.807
## 82   9  2 3168.825 3231.525
## 72   8  2 3180.780 3239.020
## 62   7  2 3212.372 3266.151
## 52   6  2 3214.331 3263.645
## 42   5  2 3221.853 3266.699
## 32   4  2 3224.285 3264.660
## 22   3  2 3226.623 3262.524
## 12   2  2 3229.051 3260.476
## 2    1  2 3239.416 3266.352
## 91  10  1 3486.250 3548.927
## 81   9  1 3503.723 3561.943
## 71   8  1 3538.751 3592.512
## 61   7  1 3575.799 3625.097
## 51   6  1 3586.116 3630.948
## 41   5  1 3599.402 3639.764
## 31   4  1 3602.664 3638.553
## 21   3  1 3608.793 3640.207
## 11   2  1 3639.223 3666.159
## 1    1  1 3712.311 3734.765
```

```
model.ARDL <- ardlDlm(x = as.vector(ppt.ts), y = as.vector(solar.ts), p = 10, q = 10)
summary(model.ARDL)
```

```
##
## Time series regression with "ts" data:
## Start = 11, End = 660
##
## Call:
## dynlm(formula = as.formula(model.text), data = data, start = 1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.2455  -1.1613  -0.1151   0.9928  18.0936
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.91075    0.49828   3.835 0.000138 ***
## X.t         -0.02897    0.55009  -0.053 0.958020
## X.1          0.74486    0.72435   1.028 0.304194
## X.2          0.63918    0.73466   0.870 0.384617
## X.3          1.18107    0.73413   1.609 0.108160
## X.4         -1.25531    0.73060  -1.718 0.086258 .
## X.5          0.09608    0.73000   0.132 0.895330
## X.6         -2.28228    0.72943  -3.129 0.001836 **
## X.7          2.35278    0.73479   3.202 0.001434 **
## X.8          0.75244    0.73775   1.020 0.308162
## X.9         -1.35804    0.73028  -1.860 0.063406 .
## X.10        -0.86665    0.55468  -1.562 0.118693
## Y.1          1.13991    0.03954  28.829  < 2e-16 ***
## Y.2          0.10168    0.05978   1.701 0.089454 .
## Y.3         -0.22380    0.05976  -3.745 0.000197 ***
## Y.4         -0.17681    0.05968  -2.963 0.003165 **
## Y.5         -0.21159    0.05935  -3.565 0.000392 ***
## Y.6          0.12216    0.05954   2.052 0.040608 *
## Y.7          0.14181    0.05964   2.378 0.017724 *
## Y.8         -0.03165    0.05921  -0.535 0.593134
## Y.9          0.16170    0.05923   2.730 0.006512 **
## Y.10        -0.12943    0.03859  -3.354 0.000844 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.321 on 628 degrees of freedom
## Multiple R-squared:  0.9462, Adjusted R-squared:  0.9444
## F-statistic: 525.8 on 21 and 628 DF,  p-value: < 2.2e-16
```

```
checkresiduals(model.ARDL$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 25
##
## data:  Residuals
## LM test = 85.129, df = 25, p-value = 1.755e-08
```

```
MASE(model.ARDL)
```

```
##                 MASE
## model.ARDL 0.3996089
```

```
# seasonally adjusted
df = data.frame(matrix(nrow = 0, ncol = length(columns)))
colnames(df) = columns

for(i in 1:10){
  for(j in 1:10){
    model.ARDL = ardlDlm(x = as.vector(ppt_seasadj), y = as.vector(solar_seasadj), p = i, q = j)
    new_row = data.frame(p = i, q=j, AIC=AIC(model.ARDL$model), BIC=BIC(model.ARDL$model))
    df = bind_rows(df, new_row)
  }
}

df[order(df$AIC),]
```

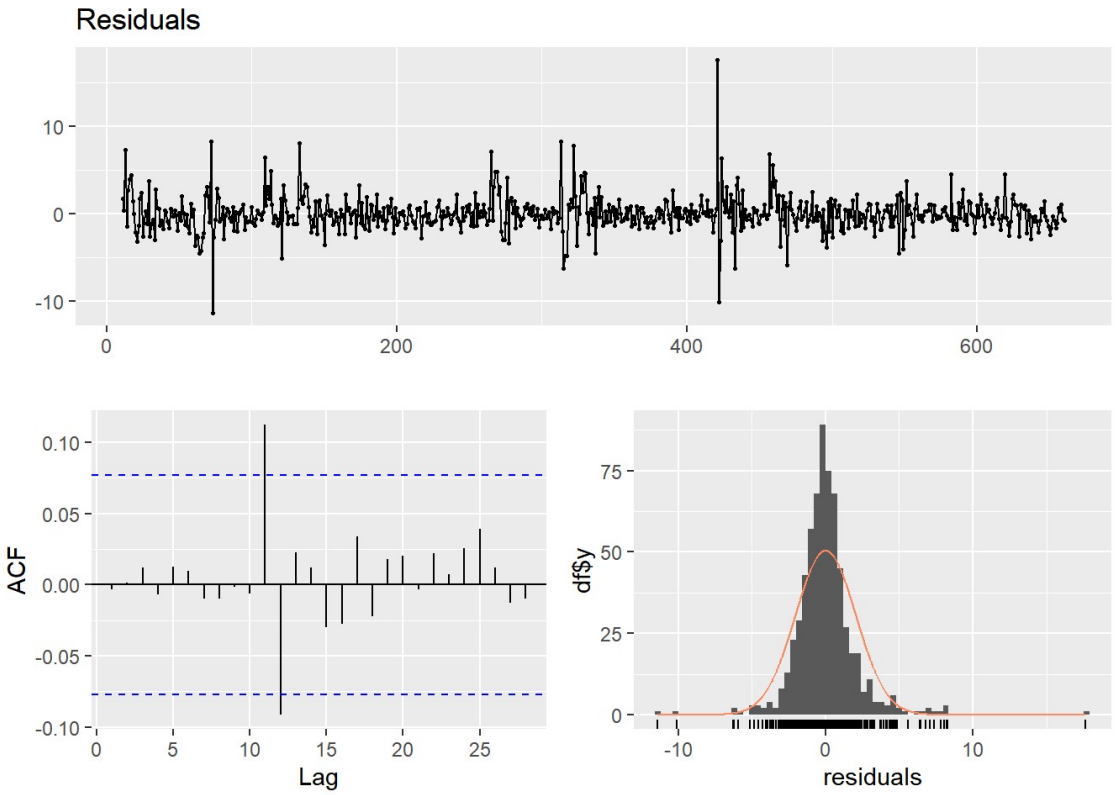
```
##      p  q      AIC      BIC
## 70   7 10 2816.259 2905.799
## 10   1 10 2816.770 2879.447
## 80   8 10 2817.613 2911.629
## 90   9 10 2817.693 2916.187
## 20   2 10 2817.913 2885.067
## 99  10  9 2819.078 2917.571
## 69   7  9 2819.196 2904.288
## 100 10 10 2819.693 2922.664
## 30   3 10 2819.876 2891.508
## 97  10  7 2819.935 2909.475
## 9    1  9 2819.996 2878.217
## 98  10  8 2820.223 2914.239
## 79   8  9 2820.565 2910.135
## 50   5 10 2820.651 2901.236
## 89   9  9 2820.727 2914.775
## 96  10  6 2820.803 2905.865
## 19   2  9 2821.015 2883.714
## 40   4 10 2821.242 2897.351
## 87   9  7 2821.632 2906.724
## 88   9  8 2821.984 2911.554
## 60   6 10 2822.241 2907.304
## 86   9  6 2822.418 2903.031
## 29   3  9 2822.983 2890.161
## 49   5  9 2823.978 2900.113
## 39   4  9 2824.484 2896.140
## 68   7  8 2824.491 2905.132
## 77   8  7 2825.399 2906.040
## 59   6  9 2825.633 2906.246
## 78   8  8 2825.915 2911.036
## 8    1  8 2826.002 2879.763
## 76   8  6 2826.280 2902.441
## 18   2  8 2826.938 2885.178
## 95  10  5 2827.542 2908.127
## 94  10  4 2828.100 2904.209
## 67   7  7 2828.105 2904.292
## 66   7  6 2828.678 2900.384
## 28   3  8 2828.936 2891.657
## 85   9  5 2829.044 2905.178
## 7    1  7 2829.437 2878.734
## 84   9  4 2829.591 2901.247
## 48   5  8 2829.601 2901.282
## 17   2  7 2830.230 2884.008
## 38   4  8 2830.358 2897.559
## 58   6  8 2831.106 2907.267
## 27   3  7 2832.214 2890.475
## 47   5  7 2832.690 2899.913
## 75   8  5 2833.034 2904.714
## 37   4  7 2833.623 2896.365
## 74   8  4 2833.779 2900.980
## 6    1  6 2833.846 2878.677
## 57   6  7 2834.155 2905.860
## 16   2  6 2834.669 2883.983
## 65   7  5 2835.197 2902.421
## 64   7  4 2835.829 2898.571
## 26   3  6 2836.665 2890.462
## 46   5  6 2837.036 2899.800
## 36   4  6 2837.891 2896.171
## 56   6  6 2838.416 2905.663
## 5    1  5 2842.569 2882.931
## 15   2  5 2843.515 2888.362
## 55   6  5 2843.607 2906.371
## 54   6  4 2844.627 2902.907
## 25   3  5 2845.513 2894.844
## 45   5  5 2845.816 2904.116
```

```
## 44   5  4 2846.710 2900.526
## 35   4  5 2846.799 2900.615
## 4    1  4 2847.900 2883.789
## 14   2  4 2848.592 2888.968
## 24   3  4 2850.587 2895.449
## 93  10  3 2850.589 2922.221
## 34   4  4 2852.020 2901.368
## 83   9  3 2852.403 2919.581
## 73   8  3 2857.245 2919.966
## 63   7  3 2858.856 2917.117
## 53   6  3 2867.636 2921.433
## 43   5  3 2869.686 2919.017
## 33   4  3 2873.760 2918.622
## 3    1  3 2874.001 2905.415
## 13   2  3 2874.862 2910.764
## 23   3  3 2876.862 2917.251
## 92  10  2 2911.248 2978.403
## 82   9  2 2913.186 2975.885
## 72   8  2 2916.350 2974.591
## 62   7  2 2918.103 2971.882
## 52   6  2 2925.006 2974.320
## 42   5  2 2926.760 2971.607
## 32   4  2 2930.024 2970.399
## 22   3  2 2932.507 2968.408
## 2    1  2 2933.063 2959.998
## 12   2  2 2934.122 2965.547
## 91  10  1 2940.785 3003.463
## 81   9  1 2942.468 3000.688
## 71   8  1 2945.722 2999.483
## 61   7  1 2947.718 2997.016
## 51   6  1 2954.230 2999.061
## 41   5  1 2956.059 2996.420
## 31   4  1 2959.636 2995.525
## 21   3  1 2961.966 2993.380
## 11   2  1 2963.793 2990.728
## 1    1  1 2966.553 2989.007
```

```
model.ARDL.adj <- ardlDlm(x = as.vector(ppt_seasadj), y = as.vector(solar_seasadj), p = 7, q = 10)
summary(model.ARDL.adj)
```

```
##
## Time series regression with "ts" data:
## Start = 11, End = 660
##
## Call:
## dynlm(formula = as.formula(model.text), data = data, start = 1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.4320  -0.9615  -0.1469   0.7731  17.6650
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.15076    0.45549   4.722 2.88e-06 ***
## X.t         -0.30359    0.50035  -0.607 0.544230
## X.1         -0.42039    0.68333  -0.615 0.538643
## X.2          0.20116    0.68990   0.292 0.770707
## X.3          0.42125    0.68882   0.612 0.541058
## X.4          0.34168    0.68755   0.497 0.619392
## X.5         -0.21380    0.68659  -0.311 0.755604
## X.6         -1.62374    0.68014  -2.387 0.017263 *
## X.7          1.38051    0.49442   2.792 0.005394 **
## Y.1          1.01086    0.03950  25.588  < 2e-16 ***
## Y.2          0.15241    0.05588   2.728 0.006558 **
## Y.3         -0.07229    0.05601  -1.291 0.197252
## Y.4         -0.12852    0.05594  -2.298 0.021912 *
## Y.5         -0.18955    0.05610  -3.379 0.000773 ***
## Y.6          0.05484    0.05605   0.978 0.328261
## Y.7          0.09736    0.05593   1.741 0.082239 .
## Y.8         -0.11061    0.05606  -1.973 0.048935 *
## Y.9          0.11680    0.05596   2.087 0.037277 *
## Y.10        -0.04614    0.03947  -1.169 0.242820
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.078 on 631 degrees of freedom
## Multiple R-squared:  0.8952, Adjusted R-squared:  0.8922
## F-statistic: 299.3 on 18 and 631 DF,  p-value: < 2.2e-16
```

```
checkresiduals(model.ARDL.adj$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 22
##
## data:  Residuals
## LM test = 37.997, df = 22, p-value = 0.01834
```

```
MASE(model.ARDL.adj)
```

```
##                     MASE
## model.ARDL.adj 0.9078539
```

An ARDL model is constructed using the ardlDlm() function. Before fitting the model, the optimal lag value for the independent series (p) and the optimal lag value for the dependent series (q) are determined through a process of multiple iterations. In each iteration, a new model is created with different p and q values. The AIC and BIC errors are computed for each model to assess their suitability. Based on the outcomes of this procedure, it is recommended that the optimal value for p is 10, while the optimal value for q is also 10.

The summary of the ARDL model shows a very low p-value of 2.2e-16, which suggests that the model is significant. A high R-squared value of 0.9444 makes it better than the seasonally adjusted Koyck model and reveals that the model explains most of the data well.

The 6th and 7th lags of the predictor as well as the 1st, 3rd, 4th, 5th, 6th, 7th, 9th, and 10th lags of solar radiation are significant.

The Breusch-Godfrey test produces a low p-value of 1.755e-08, which rejects the null hypothesis of independently distributed data. Therefore, it is possible to conclude that there is serial correlation in the model.

Analysing the residuals for the ARDL model shows that the residuals are relatively small compared to the polynomial model and are roughly normally distributed, suggesting that the

model accounts for the data well and the errors in the model are random. The ACF plot shows that most of the peaks are within the 95% confidence interval, with 5 peaks lying outside the interval. Finally, the MASE is calculated for the model and is seen to be 0.91, making it marginally better than a naive forecast.

Repeating the above procedure for the seasonally adjusted precipitation and solar radiation series produces a similar p-value of 2.2e-16 and a lower R-squared value of 0.8922. With this model, the 6th and 7th lags of the predictor variable are significant in addition to the 1st, 2nd, 4th, 5th, 8th, and 9th lags of solar radiation. Similar results are obtained for the residuals, with the residuals looking roughly normally distributed and a Breusch-Godfrey test p-value of 0.01834. This is higher than last time but still shows that serial correlation exists in the series. Overall the original ARDL model seems to explain the data better than the seasonally adjusted ARDL model due to its higher R-squared value. The MASE for the seasonally adjusted model is also 0.91.

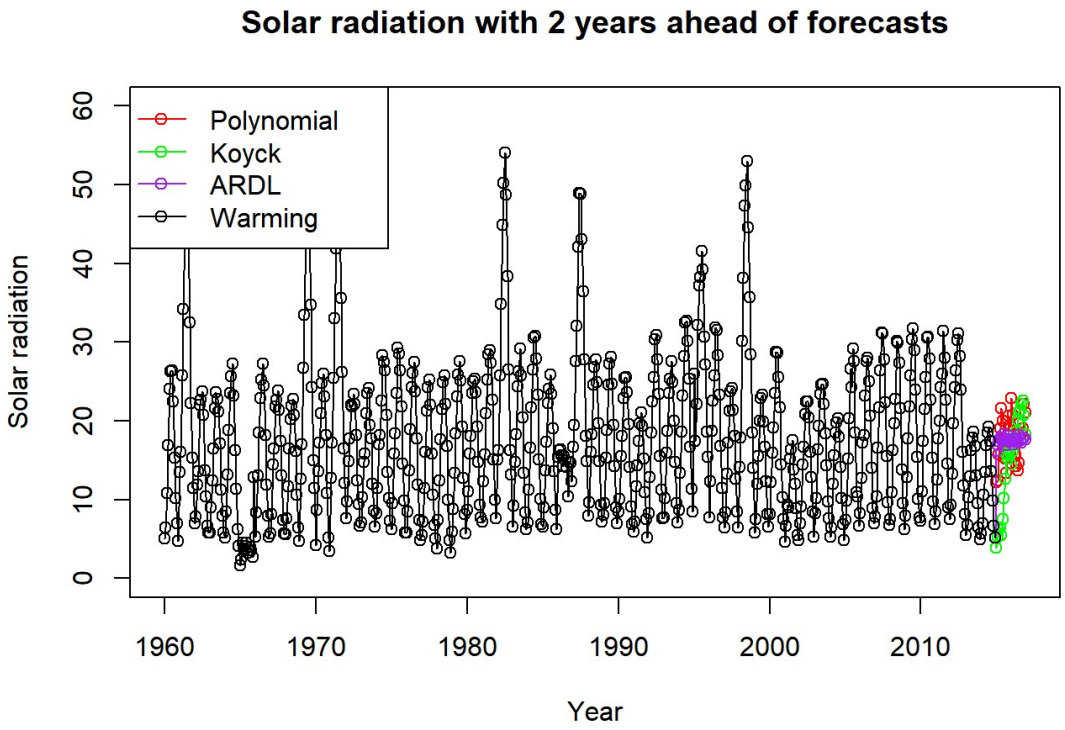
Forecasting with Time Series Regression Models

Two years ahead forecasts are created for the polynomial, Koyck, and ARDL models. The resulting plot shows that Koyck and polynomial models produces somewhat similar forecasts that have seasonal rises and dips similar to the historical data. However, the ARDL forecast sits around the mean level of the other two forecasts and remains mostly constant.

```
predictor <- read.csv("data.x.csv")
length(model.dlm)
```

```
## [1] 5
```

```
model.poly.Frc = dLagM::forecast(model.poly, x = t(predictor), h=24)
model.koyck.Frc = dLagM::forecast(model.Koyck, x = t(predictor), h=24)
model.ARDL.Frc = dLagM::forecast(model.ARDL, x = t(predictor), h=24)

plot(ts(c(as.vector(solar.ts), model.poly.Frc$forecasts), start = 1960, frequency = 12), type="o", col="red",
ylim=c(0, 60),
     ylab="Solar radiation", xlab="Year", main="Solar radiation with 2 years ahead of forecasts")
lines(ts(c(as.vector(solar.ts), model.koyck.Frc$forecasts), start = 1960, frequency = 12), col="green",type="
o")
lines(ts(c(as.vector(solar.ts), model.ARDL.Frc$forecasts), start = 1960, frequency = 12), col="purple",type="
o")
lines(ts(as.vector(solar.ts), start = 1960, frequency = 12),col="black",type="o")
legend("topleft",lty=1, pch = 1, text.width = 11, col=c( "red", "green","purple", "black"),
       c("Polynomial", "Koyck", "ARDL", "Warming"))
```


Solar radiation with 2 years ahead of forecasts
Polynomial
Koyck
ARDL
Warming
Solar radiation
0
10
20
30
40
50
60
1960
1970
1980
1990
2000
2010
Year

Exponential Smoothing Models

Three types of exponential smoothing models may be created for the solar radiation dataset. These are listed as follows:

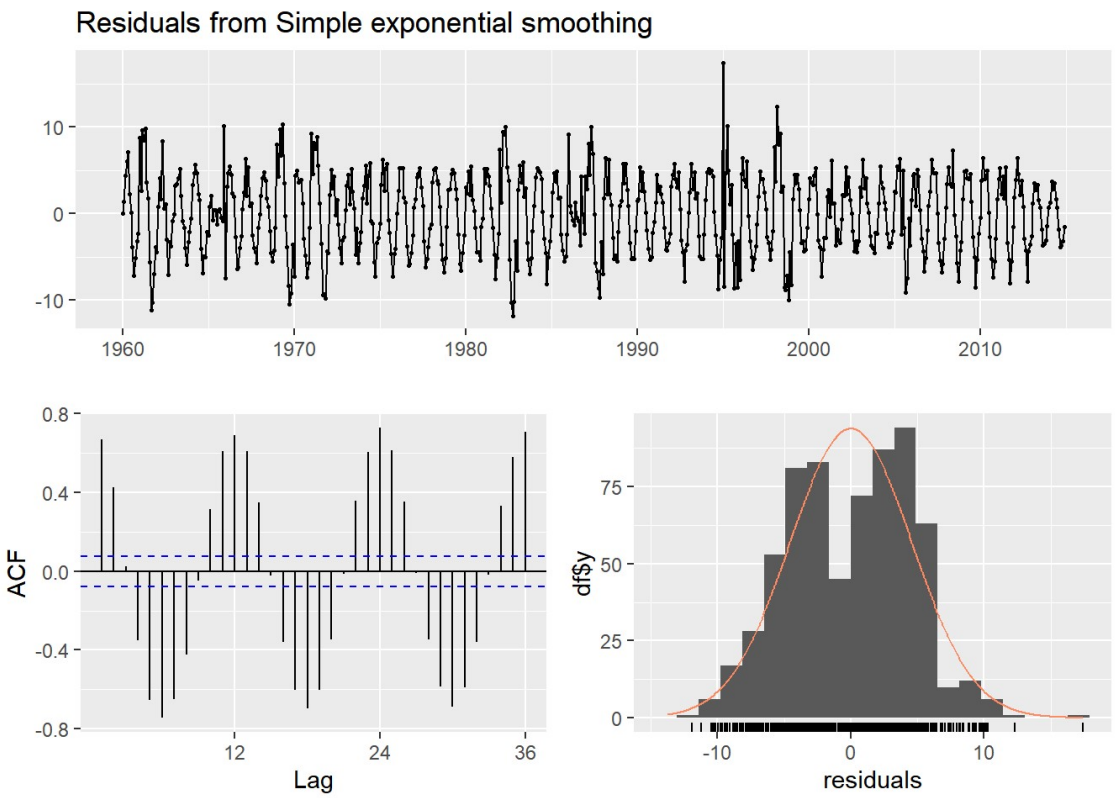
- **Simple exponential smoothing model:** Best for series with no trend or seasonality
- **Holt's linear trend model:** Best for series with seasonal components
- **Holt-Winter's trend and seasonality model:** Best for series with both trend and seasonal components

The three above models are created for the solar radiation series using the ses() and holt() functions.

```
fit1.ses <- ses(solar.ts, initial="simple", h=24)
summary(fit1.ses)
```

```
##
## Forecast method: Simple exponential smoothing
##
## Model Information:
## Simple exponential smoothing
##
## Call:
##  ses(y = solar.ts, h = 24, initial = "simple")
##
##   Smoothing parameters:
##     alpha = 1
##
##   Initial states:
##     l = 5.0517
##
##   sigma:  4.5688
## Error measures:
##                        ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 0.0001462894 4.568777 3.876091 -5.211851 27.29823 0.636771
##                   ACF1
## Training set 0.6677846
##
## Forecasts:
##          Point Forecast       Lo 80    Hi 80      Lo 95    Hi 95
## Jan 2015        5.14828  -0.7068426 11.00340  -3.806357 14.10292
## Feb 2015        5.14828  -3.1321139 13.42867  -7.515490 17.81205
## Mar 2015        5.14828  -4.9930900 15.28965 -10.361607 20.65817
## Apr 2015        5.14828  -6.5619655 16.85853 -12.760995 23.05756
## May 2015        5.14828  -7.9441725 18.24073 -14.874898 25.17146
## Jun 2015        5.14828  -9.1937832 19.49034 -16.786013 27.08257
## Jul 2015        5.14828 -10.3429188 20.63948 -18.543464 28.84002
## Aug 2015        5.14828 -11.4125081 21.70907 -20.179260 30.47582
## Sep 2015        5.14828 -12.4170884 22.71365 -21.715633 32.01219
## Oct 2015        5.14828 -13.3672440 23.66380 -23.168771 33.46533
## Nov 2015        5.14828 -14.2709655 24.56753 -24.550893 34.84745
## Dec 2015        5.14828 -15.1344604 25.43102 -25.871495 36.16806
## Jan 2016        5.14828 -15.9626655 26.25923 -27.138125 37.43469
## Feb 2016        5.14828 -16.7595836 27.05614 -28.356906 38.65347
## Mar 2016        5.14828 -17.5285132 27.82507 -29.532883 39.82944
## Apr 2016        5.14828 -18.2722113 28.56877 -30.670271 40.96683
## May 2016        5.14828 -18.9930099 29.28957 -31.772637 42.06920
## Jun 2016        5.14828 -19.6929024 29.98946 -32.843030 43.13959
## Jul 2016        5.14828 -20.3736087 30.67017 -33.884081 44.18064
## Aug 2016        5.14828 -21.0366254 31.33319 -34.898077 45.19464
## Sep 2016        5.14828 -21.6832636 31.97982 -35.887025 46.18359
## Oct 2016        5.14828 -22.3146804 32.61124 -36.852694 47.14925
## Nov 2016        5.14828 -22.9319027 33.22846 -37.796654 48.09321
## Dec 2016        5.14828 -23.5358467 33.83241 -38.720306 49.01687
```

```
checkresiduals(fit1.ses)
```

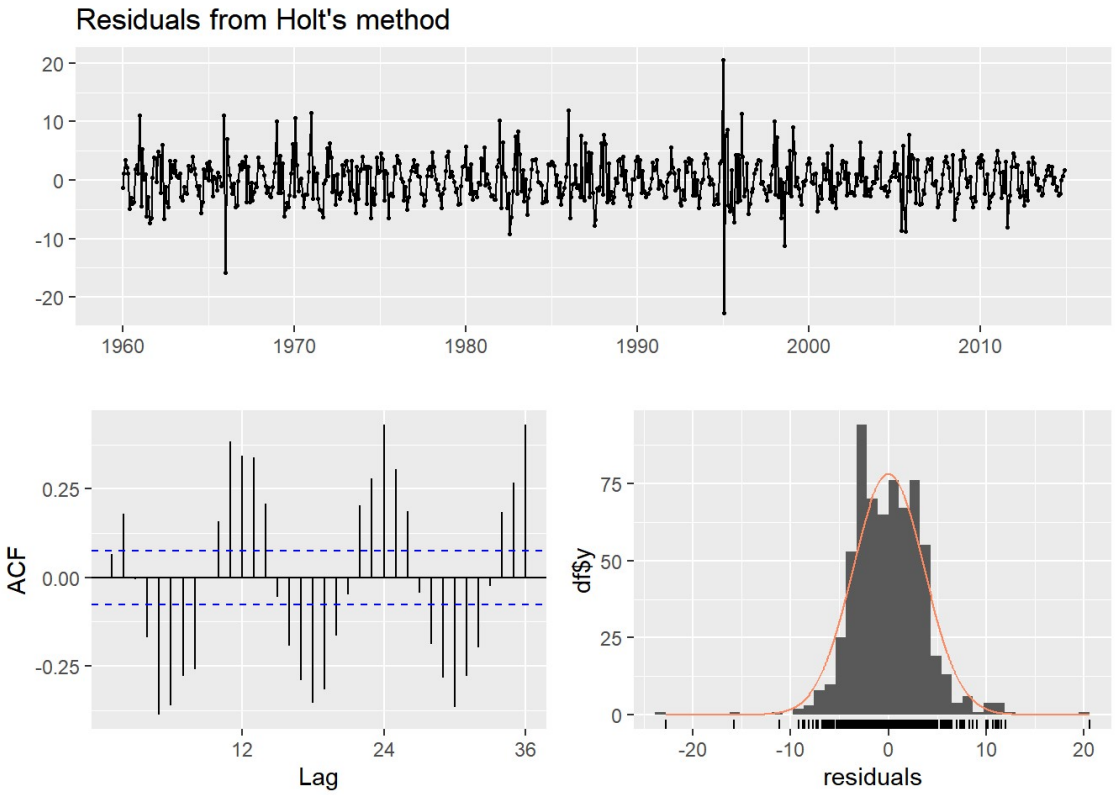



```
##
##  Ljung-Box test
##
## data:  Residuals from Simple exponential smoothing
## Q* = 4227.3, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit2.holt <- holt(solar.ts, initial="simple", h=24)
summary(fit2.holt)
```

```
##
## Forecast method: Holt's method
##
## Model Information:
## Holt's method
##
## Call:
##  holt(y = solar.ts, h = 24, initial = "simple")
##
##   Smoothing parameters:
##     alpha = 0.9165
##     beta  = 1
##
##   Initial states:
##     l = 5.0517
##     b = 1.3641
##
##   sigma:  3.6493
## Error measures:
##                         ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.004941411 3.649286 2.806374 6.556498 21.13578 0.4610361
##                    ACF1
## Training set 0.06518525
##
## Forecasts:
##          Point Forecast       Lo 80      Hi 80      Lo 95     Hi 95
## Jan 2015      3.3755382   -1.301210   8.052286   -3.77693  10.52801
## Feb 2015      1.7507183   -8.358924  11.860360  -13.71065  17.21208
## Mar 2015      0.1258983  -16.851847  17.103643  -25.83932  26.09112
## Apr 2015     -1.4989216  -26.473564  23.475721  -39.69434  36.69650
## May 2015     -3.1237415  -37.070990  30.823507  -55.04158  48.79409
## Jun 2015     -4.7485614  -48.543955  39.046832  -71.72784  62.23071
## Jul 2015     -6.3733813  -60.819126  48.072363  -89.64096  76.89420
## Aug 2015     -7.9982012  -73.839431  57.843028 -108.69367  92.69727
## Sep 2015     -9.6230212  -87.558671  68.312629 -128.81531 109.56927
## Oct 2015    -11.2478411 -101.938399  79.442717 -149.94708 127.45140
## Nov 2015    -12.8726610 -116.945936  91.200614 -172.03900 146.29368
## Dec 2015    -14.4974809 -132.553051 103.558089 -195.04789 166.05293
## Jan 2016    -16.1223008 -148.735022 116.490421 -218.93596 186.69135
## Feb 2016    -17.7471207 -165.469968 129.975727 -243.66972 208.17548
## Mar 2016    -19.3719407 -182.738335 143.994454 -269.21928 230.47540
## Apr 2016    -20.9967606 -200.522511 158.528990 -295.55770 253.56418
## May 2016    -22.6215805 -218.806526 173.563365 -322.66056 277.41740
## Jun 2016    -24.2464004 -237.575807 189.083006 -350.50557 302.01277
## Jul 2016    -25.8712203 -256.816988 205.074547 -379.07229 327.32985
## Aug 2016    -27.4960402 -276.517752 221.525671 -408.34188 353.34980
## Sep 2016    -29.1208602 -296.666697 238.424977 -438.29691 380.05519
## Oct 2016    -30.7456801 -317.253233 255.761873 -468.92117 407.42981
## Nov 2016    -32.3705000 -338.267485 273.526485 -500.19957 435.45857
## Dec 2016    -33.9953199 -359.700221 291.709581 -532.11798 464.12734
```

```
checkresiduals(fit2.holt)
```

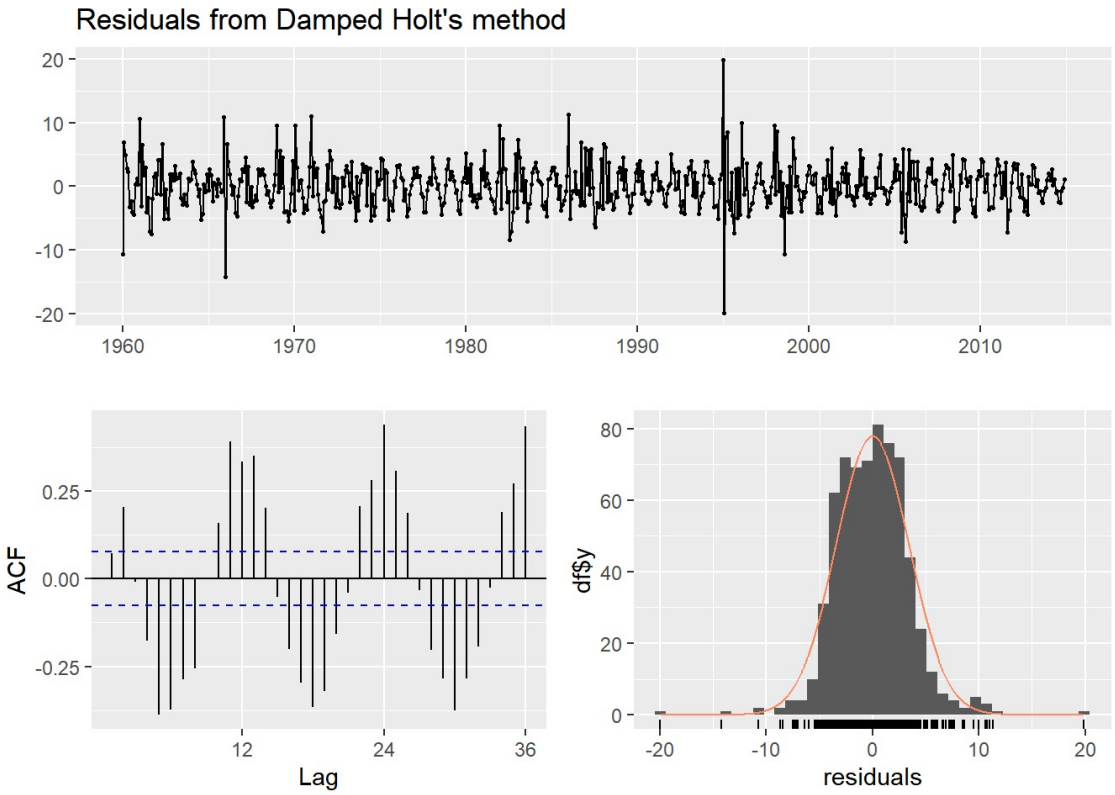



```
##
##  Ljung-Box test
##
## data:  Residuals from Holt's method
## Q* = 1096.3, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit3.holt <- holt(solar.ts, damped=TRUE, initial="simple", h=24) # Fit with additive damped trend
summary(fit3.holt)
```

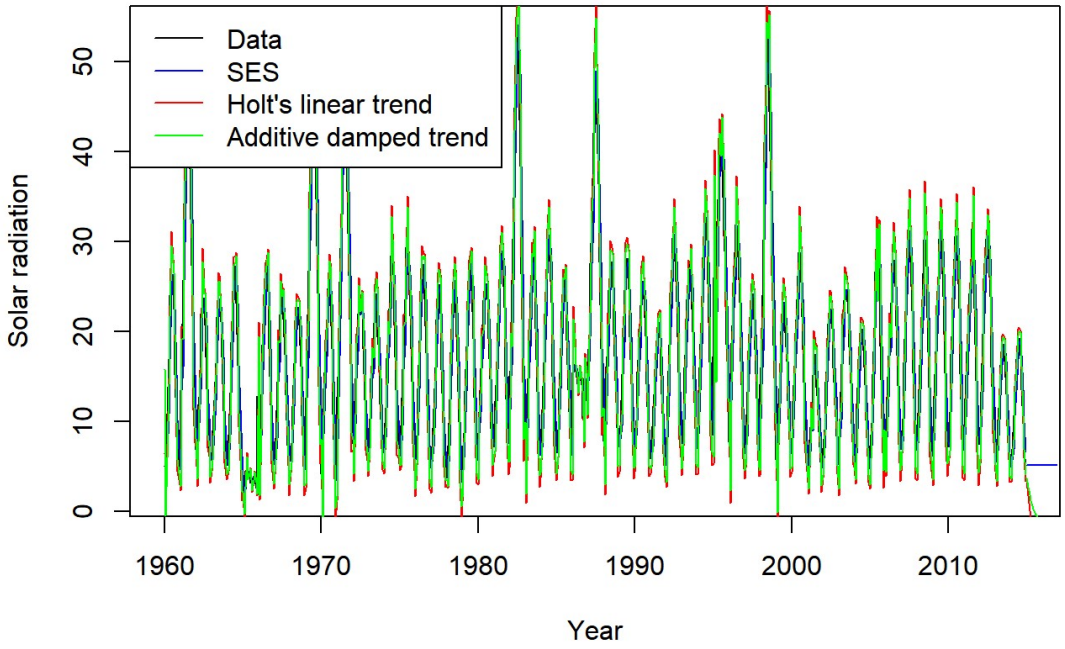
```
##
## Forecast method: Damped Holt's method
##
## Model Information:
## Damped Holt's method
##
## Call:
##  holt(y = solar.ts, h = 24, damped = TRUE, initial = "simple")
##
##   Smoothing parameters:
##     alpha = 0.9278
##     beta  = 0.9278
##     phi   = 0.8
##
##   Initial states:
##     l = 13.7194
##     b = 2.5785
##
##   sigma:  3.4657
##
##      AIC     AICc      BIC
## 5932.524 5932.652 5959.477
##
## Error measures:
##                       ME     RMSE      MAE      MPE     MAPE     MASE
## Training set -0.00831309 3.452592 2.638613 3.791232 19.25658 0.433476
##                    ACF1
## Training set 0.07287524
##
## Forecasts:
##          Point Forecast       Lo 80    Hi 80       Lo 95     Hi 95
## Jan 2015     3.77468888  -0.6668422  8.21622   -3.018047  10.56742
## Feb 2015     2.73586859  -5.9098357 11.38157  -10.486595  15.95833
## Mar 2015     1.90481212 -11.3560708 15.16570  -18.375958  22.18558
## Apr 2015     1.23996674 -16.7554651 19.23540  -26.281671  28.76160
## May 2015     0.70809029 -22.0017301 23.41791  -34.023583  35.43976
## Jun 2015     0.28258901 -27.0463637 27.61154  -41.513437  42.07862
## Jul 2015    -0.05781212 -31.8700954 31.75447  -48.710501  48.59488
## Aug 2015    -0.33013310 -36.4696231 35.80936  -55.600714  54.94045
## Sep 2015    -0.54798994 -40.8505077 39.75453  -62.185372  61.08939
## Oct 2015    -0.72227547 -45.0230940 43.57854  -68.474531  67.02998
## Nov 2015    -0.86170394 -49.0000892 47.27668  -74.483011  72.75960
## Dec 2015    -0.97324674 -52.7951082 50.84861  -80.227945  78.28145
## Jan 2016    -1.06248101 -56.4218036 54.29684  -85.727259  83.60230
## Feb 2016    -1.13386844 -59.8933599 57.62562  -90.998756  88.73102
## Mar 2016    -1.19097841 -63.2222137 60.84026  -96.059566  93.67761
## Apr 2016    -1.23666639 -66.4199175 63.94658 -100.925846  98.45251
## May 2016    -1.27321679 -69.4970922 66.95066 -105.612630 103.06620
## Jun 2016    -1.30245712 -72.4634340 69.85852 -110.133780 107.52887
## Jul 2016    -1.32584939 -75.3277524 72.67605 -114.501994 111.85029
## Aug 2016    -1.34456321 -78.0980276 75.40890 -118.728857 116.03973
## Sep 2016    -1.35953427 -80.7814739 78.06241 -122.824909 120.10584
## Oct 2016    -1.37151112 -83.3846076 80.64159 -126.799718 124.05670
## Nov 2016    -1.38109261 -85.9133126 83.15113 -130.661967 127.89978
## Dec 2016    -1.38875780 -88.3729032 85.59539 -134.419528 131.64201
```

```
checkresiduals(fit3.holt)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Damped Holt's method
## Q* = 1134.4, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
plot(solar.ts, type="l", ylab="Solar radiation", xlab="Year", fcol="white", plot.conf=FALSE)
lines(fitted(fit1.ses), col="blue")
lines(fitted(fit2.holt), col="red")
lines(fitted(fit3.holt), col="green")
lines(fit1.ses$mean, col="blue", type="l")
lines(fit2.holt$mean, col="red", type="l")
lines(fit3.holt$mean, col="green", type="l")
legend("topleft", lty=1, col=c("black","blue","red","green"),
       c("Data","SES", "Holt's linear trend", "Additive damped trend"))
```



Model Summary

Model	AIC	BIC	MASE	Ljung-Box p-value	Residual Normality
Simple SES	NA	NA	0.636771	2.2e-16	Not normal
Simple Holt	NA	NA	0.4610361	2.2e-16	Not normal
Additive Damped Trend Holt SES	5932.524	5959.477	0.433476	2.2e-16	Normal

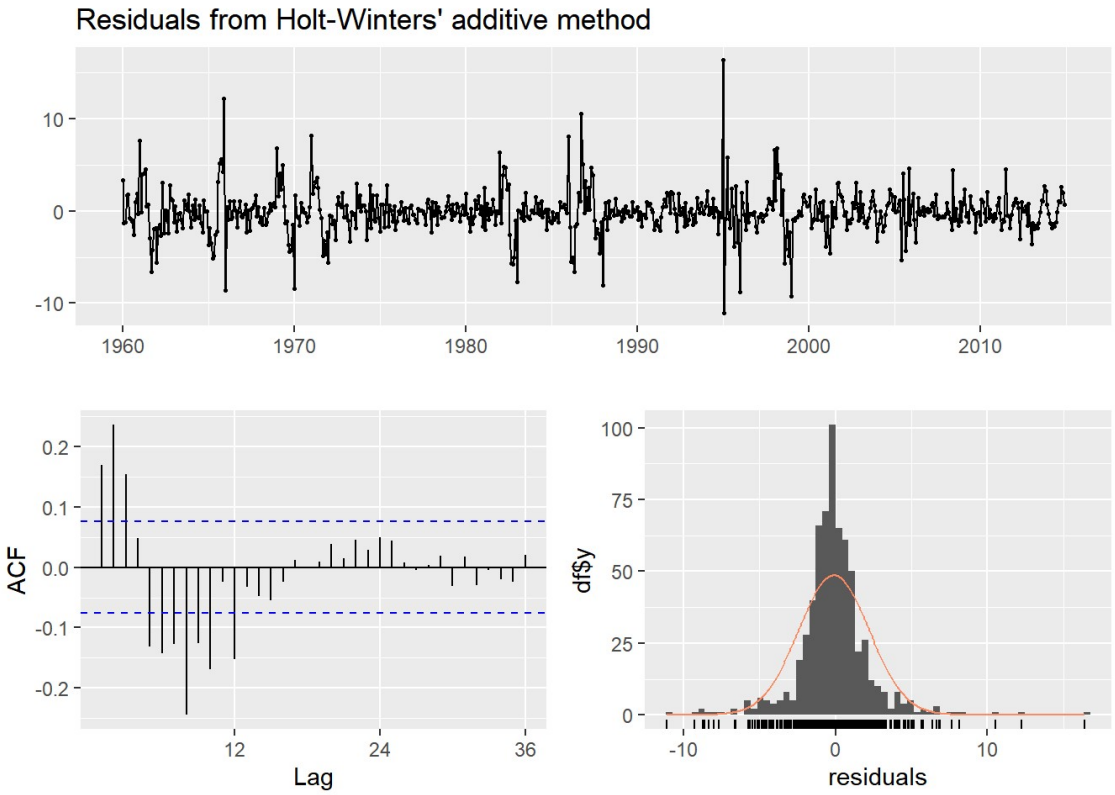
Checking the summaries and residuals for each model reveals that all the models are significant with p-values of 2.2e-16. The additive damped trend model has the lowest MASE from the three models and the most normally distributed residuals based on the residual plots. This suggests that the additive damped trend model is the best model for solar radiation thus far.

State Space Models

```
fit5.hw <- hw(solar.ts,seasonal="additive", h=24)
summary(fit5.hw)
```

```
##
## Forecast method: Holt-Winters' additive method
##
## Model Information:
## Holt-Winters' additive method
##
## Call:
##  hw(y = solar.ts, h = 24, seasonal = "additive")
##
##   Smoothing parameters:
##     alpha = 0.9993
##     beta  = 0.0019
##     gamma = 1e-04
##
##   Initial states:
##     l = 11.3306
##     b = 0.209
##     s = -10.7542 -8.4968 -3.227 3.0768 7.9264 10.9122
##            10.1422 7.3322 2.2353 -1.9528 -7.3626 -9.8316
##
##   sigma:  2.3575
##
##      AIC     AICc      BIC
## 5434.708 5435.662 5511.076
##
## Error measures:
##                       ME     RMSE      MAE       MPE     MAPE    MASE      ACF1
## Training set -0.1179476 2.328736 1.504489 -2.230359 12.68101 0.24716 0.1703355
##
## Forecasts:
##          Point Forecast      Lo 80     Hi 80      Lo 95    Hi 95
## Jan 2015       6.127399  3.1061581  9.148641   1.506810 10.74799
## Feb 2015       8.655602  4.3802222 12.930983   2.116973 15.19423
## Mar 2015      14.122221  8.8814747 19.362968   6.107191 22.13725
## Apr 2015      18.366625 12.3095941 24.423656   9.103196 27.63005
## May 2015      23.522264 16.7439512 30.300578  13.155729 33.88880
## Jun 2015      26.390991 18.9586817 33.823300  15.024255 37.75773
## Jul 2015      27.219228 19.1837595 35.254697  14.930039 39.50842
## Aug 2015      24.290519 15.6920179 32.889020  11.140246 37.44079
## Sep 2015      19.495874 10.3670365 28.624711   5.534522 33.45723
## Oct 2015      13.253765  3.6218809 22.885648  -1.476930 27.98446
## Nov 2015       8.042144 -2.0695730 18.153861  -7.422393 23.50668
## Dec 2015       5.840110 -4.7313921 16.411612 -10.327607 22.00783
## Jan 2016       6.819616 -4.1942247 17.833457 -10.024600 23.66383
## Feb 2016       9.347819 -2.0927719 20.788410  -8.149055 26.84469
## Mar 2016      14.814438  2.9609172 26.667959  -3.313958 32.94283
## Apr 2016      19.058842  6.8048110 31.312872   0.317919 37.79976
## May 2016      24.214481 11.5711780 36.857784   4.878218 43.55074
## Jun 2016      27.083208 14.0608586 40.105557   7.167243 46.99917
## Jul 2016      27.911445 14.5194061 41.303485   7.430089 48.39280
## Aug 2016      24.982736 11.2296053 38.735867   3.949138 46.01633
## Sep 2016      20.188091  6.0818048 34.294377  -1.385612 41.76179
## Oct 2016      13.945981 -0.5061083 28.398071  -8.156582 36.04855
## Nov 2016       8.734361 -6.0566988 23.525420 -13.886613 31.35533
## Dec 2016       6.532327 -8.5913309 21.655984 -16.597311 29.66196
```

```
checkresiduals(fit5.hw)
```

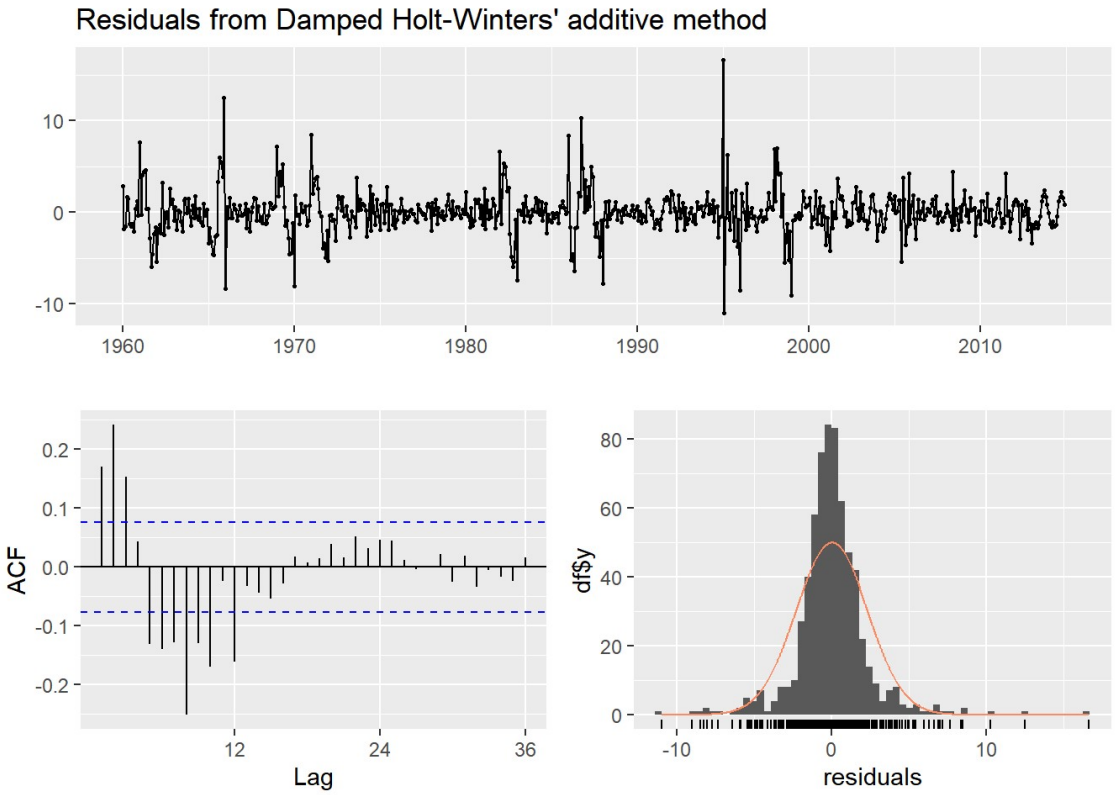



```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' additive method
## Q* = 205.55, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit6.hw <- hw(solar.ts,seasonal="additive",damped = TRUE, h=24)
summary(fit6.hw)
```

```
##
## Forecast method: Damped Holt-Winters' additive method
##
## Model Information:
## Damped Holt-Winters' additive method
##
## Call:
##  hw(y = solar.ts, h = 24, seasonal = "additive", damped = TRUE)
##
##   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
##
## 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
##
## Forecasts:
##          Point Forecast     Lo 80     Hi 80       Lo 95    Hi 95
## Jan 2015       5.945198  2.940530  8.949866   1.3499548 10.54044
## Feb 2015       8.479778  4.230548 12.729009   1.9811413 14.97842
## Mar 2015      13.784309  8.579902 18.988716   5.8248548 21.74376
## Apr 2015      17.598568 11.588776 23.608360   8.4073847 26.78975
## May 2015      22.632258 15.912803 29.351713  12.3557383 32.90878
## Jun 2015      25.599150 18.238024 32.960277  14.3412786 36.85702
## Jul 2015      26.761775 18.810497 34.713053  14.6013444 38.92221
## Aug 2015      23.722251 15.221610 32.222892  10.7216429 36.72286
## Sep 2015      18.222624  9.205955 27.239293   4.4328191 32.01243
## Oct 2015      12.293259  2.788471 21.798047  -2.2430606 26.82958
## Nov 2015       7.504219 -2.464877 17.473315  -7.7421977 22.75064
## Dec 2015       5.150488 -5.262286 15.563263 -10.7744758 21.07545
## Jan 2016       5.947275 -4.891175 16.785726 -10.6287044 22.52326
## Feb 2016       8.481728 -2.766251 19.729707  -8.7205712 25.68403
## Mar 2016      13.786139  2.142988 25.429291  -4.0205242 31.59280
## Apr 2016      17.600287  5.574906 29.625668  -0.7909464 35.99152
## May 2016      22.633871 10.238009 35.029734   3.6760353 41.59171
## Jun 2016      25.600665 12.845046 38.356283   6.0926298 45.10870
## Jul 2016      26.763197 13.657667 39.868726   6.7200186 46.80637
## Aug 2016      23.723586 10.277223 37.169950   3.1591478 44.28802
## Sep 2016      18.223877  4.445085 32.002669  -2.8489663 39.29672
## Oct 2016      12.294435 -1.808972 26.397843  -9.2748652 33.86374
## Nov 2016       7.505324 -6.915414 21.926061 -14.5492911 29.55994
## Dec 2016       5.151525 -9.579726 19.882776 -17.3779790 27.68103
```

```
checkresiduals(fit6.hw)
```

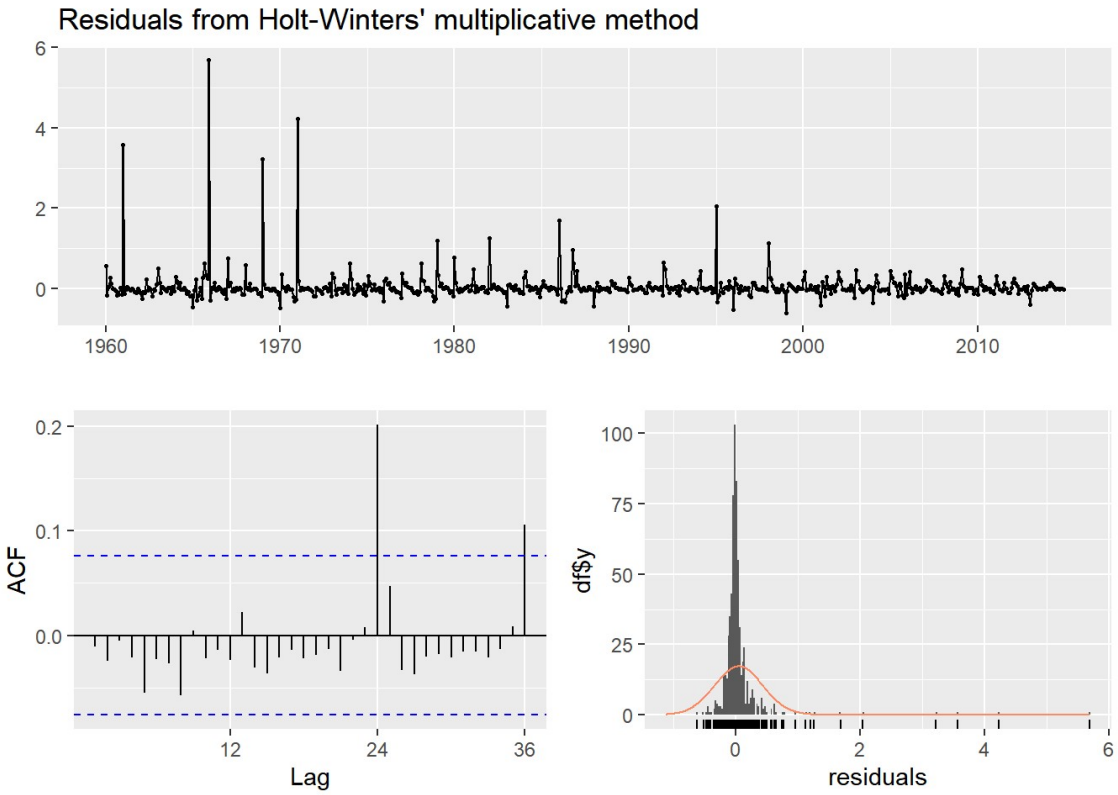



```
##
##  Ljung-Box test
##
## data:  Residuals from Damped Holt-Winters' additive method
## Q* = 210.76, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit7.hw <- hw(solar.ts,seasonal="multiplicative", h=24)
summary(fit7.hw)
```

```
##
## Forecast method: Holt-Winters' multiplicative method
##
## Model Information:
## Holt-Winters' multiplicative method
##
## Call:
##  hw(y = solar.ts, h = 24, seasonal = "multiplicative")
##
##   Smoothing parameters:
##     alpha = 0.8251
##     beta  = 1e-04
##     gamma = 0.0259
##
##   Initial states:
##     l = 10.5659
##     b = -0.2371
##     s = 0.5024 0.6292 0.9319 1.2229 1.4923 1.6309
##            1.5606 1.3672 1.0278 0.8128 0.5106 0.3114
##
##   sigma:  0.3971
##
##      AIC     AICc      BIC
## 6648.746 6649.699 6725.114
##
## Error measures:
##                       ME    RMSE      MAE        MPE     MAPE      MASE
## Training set 0.04281136 2.12539 1.359297 -0.4896895 10.87401 0.2233077
##                    ACF1
## Training set 0.06551473
##
## Forecasts:
##          Point Forecast        Lo 80     Hi 80       Lo 95     Hi 95
## Jan 2015       5.610335   2.75534413  8.465326   1.2440033  9.976666
## Feb 2015       6.512806   2.03809262 10.987520  -0.3306776 13.356290
## Mar 2015       8.786120   1.33667747 16.235562  -2.6068190 20.179058
## Apr 2015      10.192785  -0.02672876 20.412298  -5.4366123 25.822181
## May 2015      12.512597  -1.96157245 26.986766  -9.6237347 34.648928
## Jun 2015      14.061609  -4.41048190 32.533701 -14.1890164 42.312235
## Jul 2015      14.502088  -6.89909775 35.903274 -18.2282011 47.232377
## Aug 2015      12.945620  -8.35024040 34.241481 -19.6235881 45.514829
## Sep 2015      10.352210  -8.52365174 29.228072 -18.5159294 39.220350
## Oct 2015       7.418279  -7.51119146 22.347750 -15.4143760 30.250935
## Nov 2015       4.989730  -6.05900058 16.038460 -11.9078451 21.887305
## Dec 2015       3.871776  -5.53876625 13.282319 -10.5204066 18.263960
## Jan 2016       4.199400  -7.01753622 15.416336 -12.9554236 21.354224
## Feb 2016       4.838892  -9.30305408 18.980837 -16.7893479 26.467131
## Mar 2016       6.477174 -14.21845674 27.172805 -25.1740619 38.128410
## Apr 2016       7.452633 -18.56883703 33.474103 -32.3437711 47.249037
## May 2016       9.069749 -25.52978297 43.669280 -43.8456686 61.985166
## Jun 2016      10.099485 -31.99838423 52.197355 -54.2836502 74.482621
## Jul 2016      10.315199 -36.68017471 57.310572 -61.5580226 82.188420
## Aug 2016       9.113765 -36.29212001 54.519651 -60.3285438 78.556075
## Sep 2016       7.208700 -32.09306283 46.510463 -52.8981594 67.315559
## Oct 2016       5.105869 -25.38360449 35.595343 -41.5237568 51.735495
## Nov 2016       3.391945 -18.81647619 25.600367 -30.5729044 37.356795
## Dec 2016       2.597256 -16.07148862 21.266001 -25.9541251 31.148637
```

```
checkresiduals(fit7.hw)
```

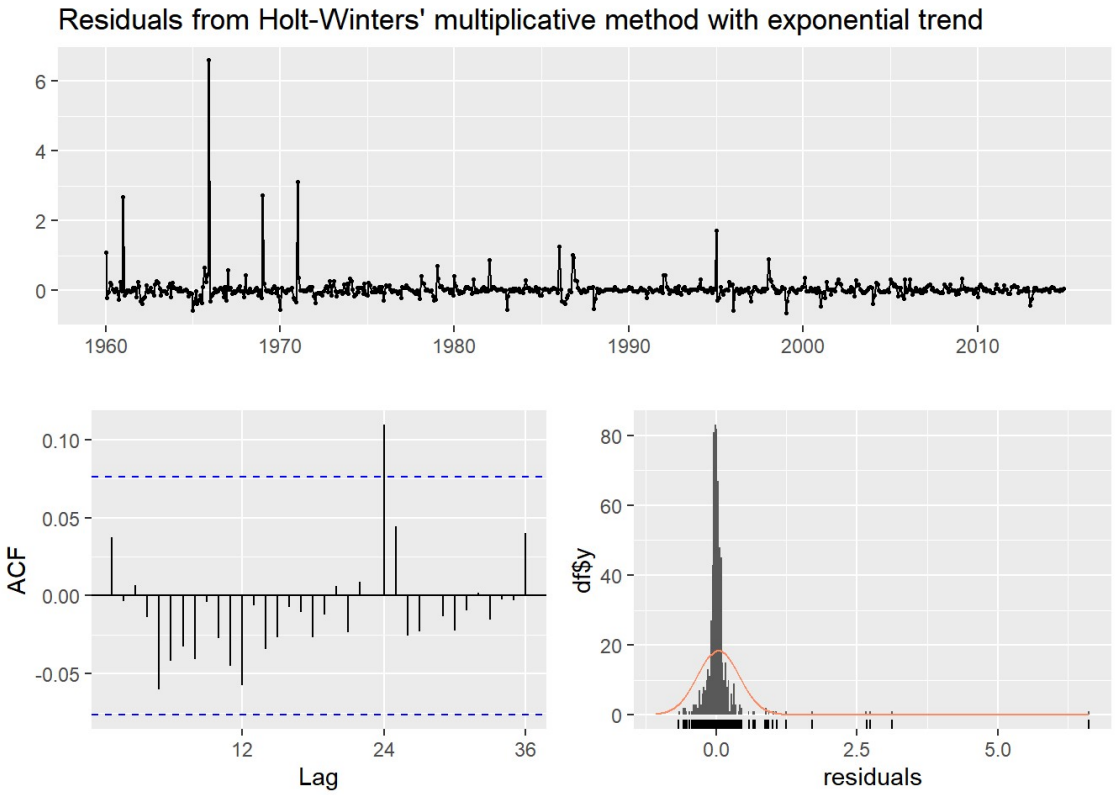



```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method
## Q* = 38.585, df = 24, p-value = 0.03017
##
## Model df: 0.   Total lags used: 24
```

```
fit8.hw <- hw(solar.ts,seasonal="multiplicative",exponential = TRUE, h=24)
summary(fit8.hw)
```

```
##
## Forecast method: Holt-Winters' multiplicative method with exponential trend
##
## Model Information:
## Holt-Winters' multiplicative method with exponential trend
##
## Call:
##  hw(y = solar.ts, h = 24, seasonal = "multiplicative", exponential = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.6499
##     beta  = 1e-04
##     gamma = 0.0597
##
##   Initial states:
##     l = 9.3759
##     b = 0.9889
##     s = 0.3552 0.4789 0.952 1.0553 1.4608 1.7926
##            1.6746 1.4529 1.1145 0.8672 0.5333 0.2627
##
##   sigma:  0.3755
##
##      AIC     AICc      BIC
## 6584.208 6585.161 6660.576
##
## Error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE     ACF1
## Training set 0.06388152 2.208727 1.412454 -1.303792 11.28449 0.2320404 0.153871
##
## Forecasts:
##          Point Forecast     Lo 80     Hi 80      Lo 95     Hi 95
## Jan 2015       5.797870 2.9828897  8.531286 1.42193901  9.899812
## Feb 2015       7.513359 3.3557135 12.164457 1.65600046 15.431449
## Mar 2015      10.758212 4.2782752 18.515268 2.24732415 24.171365
## Apr 2015      12.694087 4.6130448 22.843208 2.15476472 30.886029
## May 2015      15.340074 5.0225551 28.268214 2.29927869 40.406193
## Jun 2015      17.239431 4.9292116 32.473814 2.41387440 46.897956
## Jul 2015      18.004700 4.7199266 35.875938 2.18958859 53.874878
## Aug 2015      16.206596 3.9195735 33.504144 1.80972430 52.305241
## Sep 2015      13.011256 2.8682077 26.807056 1.28023937 44.497234
## Oct 2015       9.215869 1.8706729 19.690045 0.83261037 32.318339
## Nov 2015       6.214762 1.1681461 13.416052 0.51491733 23.032044
## Dec 2015       4.565667 0.7765353  9.941973 0.32677670 17.525757
## Jan 2016       5.220661 0.8073444 11.694033 0.33122964 21.597453
## Feb 2016       6.765364 0.9628256 15.453767 0.40303480 29.315403
## Mar 2016       9.687175 1.3429910 22.241604 0.45429286 44.015025
## Apr 2016      11.430324 1.4748069 26.909352 0.57481368 55.223559
## May 2016      13.812889 1.6858510 31.965611 0.58320055 67.536162
## Jun 2016      15.523155 1.8056897 36.937003 0.65787219 72.360570
## Jul 2016      16.212237 1.7482777 39.852927 0.59045820 80.944766
## Aug 2016      14.593143 1.4099054 35.201430 0.48693355 72.783897
## Sep 2016      11.715916 1.0499570 27.752366 0.37507717 60.224446
## Oct 2016       8.298381 0.6988630 20.001012 0.25088692 46.766649
## Nov 2016       5.596049 0.4365067 13.902330 0.12931021 30.874739
## Dec 2016       4.111130 0.3107000 10.179641 0.09830758 23.128783
```

```
checkresiduals(fit8.hw)
```

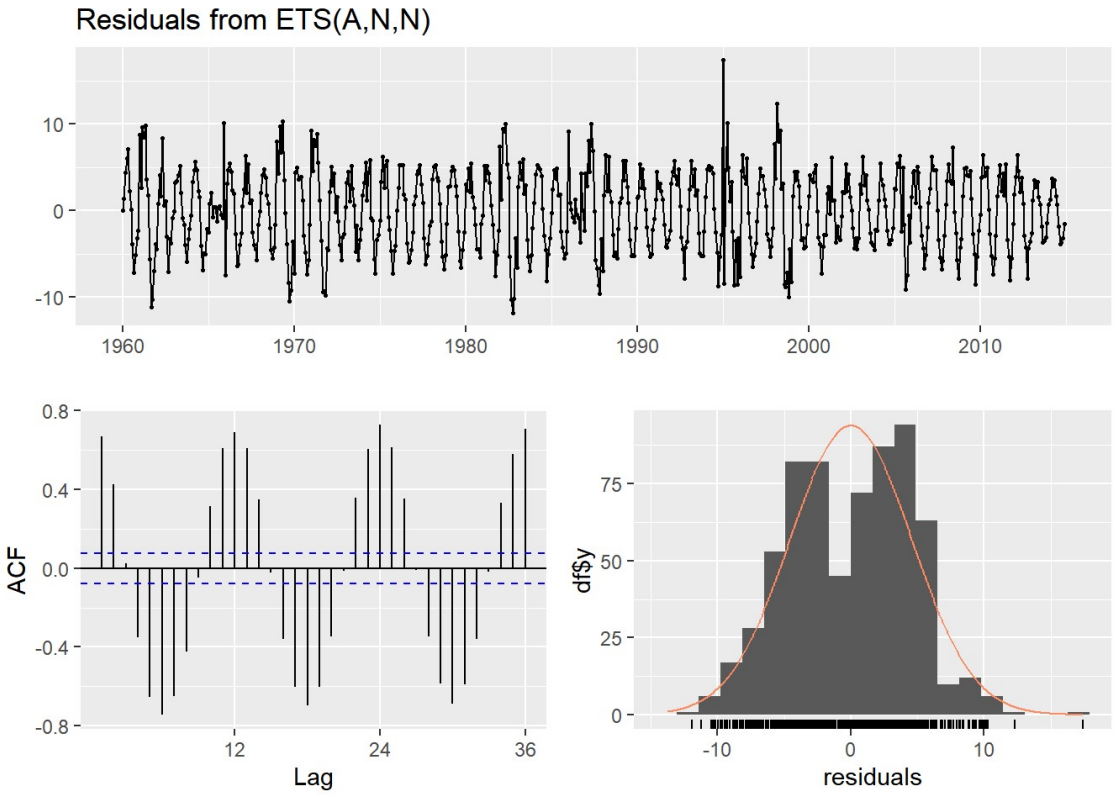



```
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method with exponential trend
## Q* = 21.246, df = 24, p-value = 0.6242
##
## Model df: 0.   Total lags used: 24
```

```
fit1.etsA = ets(solar.ts, model="ANN")
summary(fit1.etsA)
```

```
## ETS(A,N,N)
##
## Call:
##  ets(y = solar.ts, model = "ANN")
##
##   Smoothing parameters:
##     alpha = 0.9999
##
##   Initial states:
##     l = 5.0575
##
##   sigma:  4.576
##
##      AIC     AICc      BIC
## 6296.371 6296.407 6309.847
##
## Training set error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.0001378357 4.569082 3.876391 -5.213129 27.30052 0.6368203
##                   ACF1
## Training set 0.6678374
```

```
checkresiduals(fit1.etsA)
```

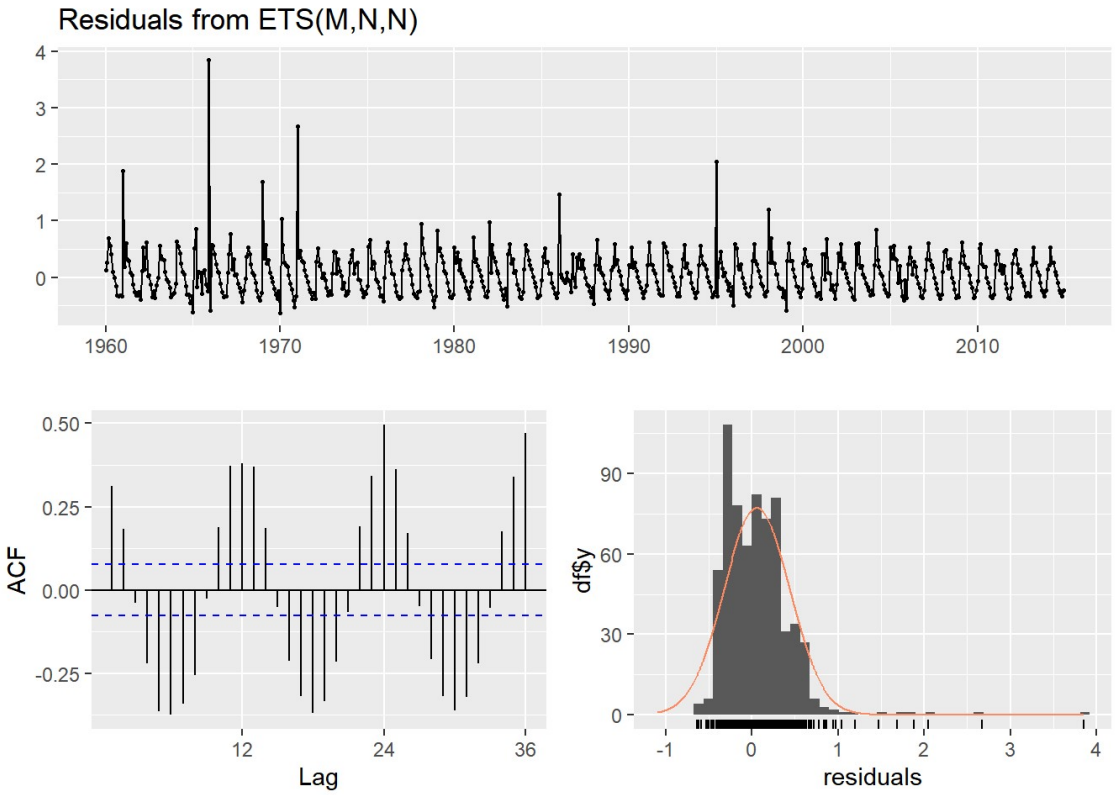



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,N,N)
## Q* = 4227.6, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit1.etsM = ets(solar.ts, model="MNN")
summary(fit1.etsM)
```

```
## ETS(M,N,N)
##
## Call:
##  ets(y = solar.ts, model = "MNN")
##
##   Smoothing parameters:
##     alpha = 0.9999
##
##   Initial states:
##     l = 4.4856
##
##   sigma:  0.3868
##
##      AIC     AICc      BIC
## 6619.776 6619.812 6633.253
##
## Training set error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.001004352 4.569136 3.877241 -5.195978 27.31733 0.6369599
##                    ACF1
## Training set 0.6678785
```

```
checkresiduals(fit1.etsM)
```

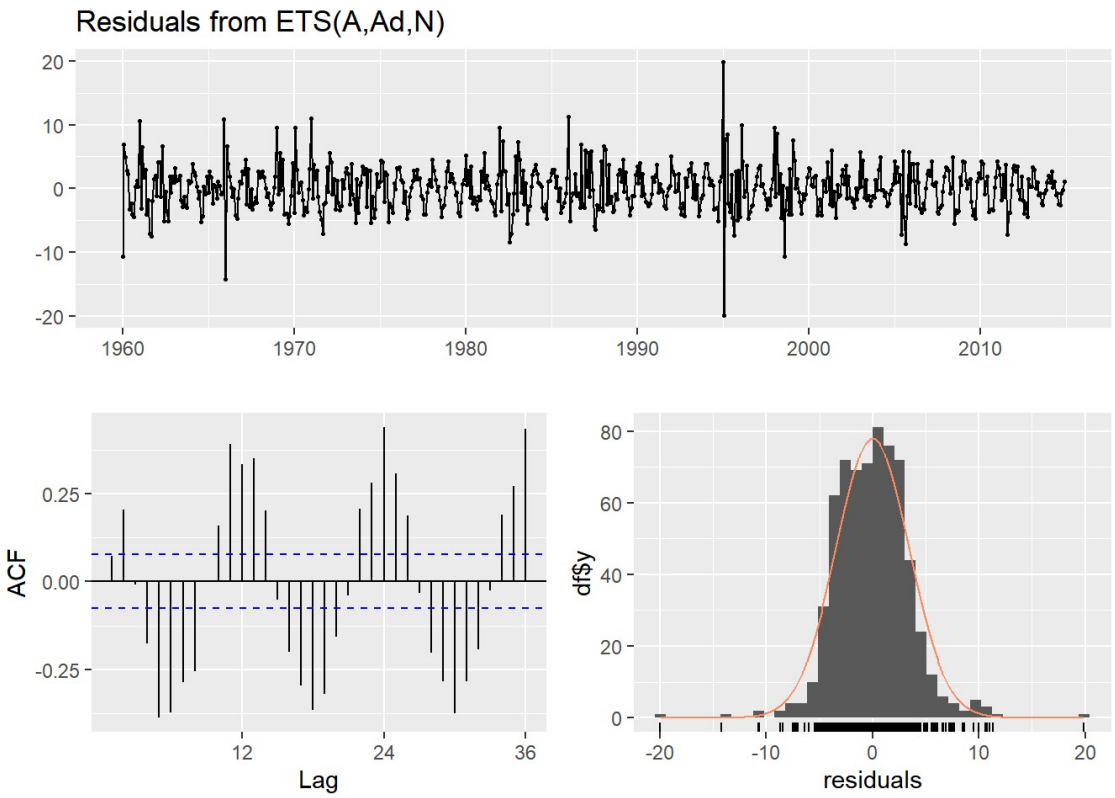



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,N,N)
## Q* = 1334.8, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit2.etsA = ets(solar.ts, model="AAN")
summary(fit2.etsA)
```

```
## ETS(A,Ad,N)
##
## Call:
##  ets(y = solar.ts, model = "AAN")
##
##   Smoothing parameters:
##     alpha = 0.9279
##     beta  = 0.9279
##     phi   = 0.8
##
##   Initial states:
##     l = 13.7196
##     b = 2.5786
##
##   sigma:  3.4657
##
##      AIC     AICc      BIC
## 5932.524 5932.653 5959.478
##
## Training set error measures:
##                       ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.008312543 3.452593 2.638571 3.790894 19.25631 0.4334691
##                    ACF1
## Training set 0.07272876
```

```
checkresiduals(fit2.etsA)
```

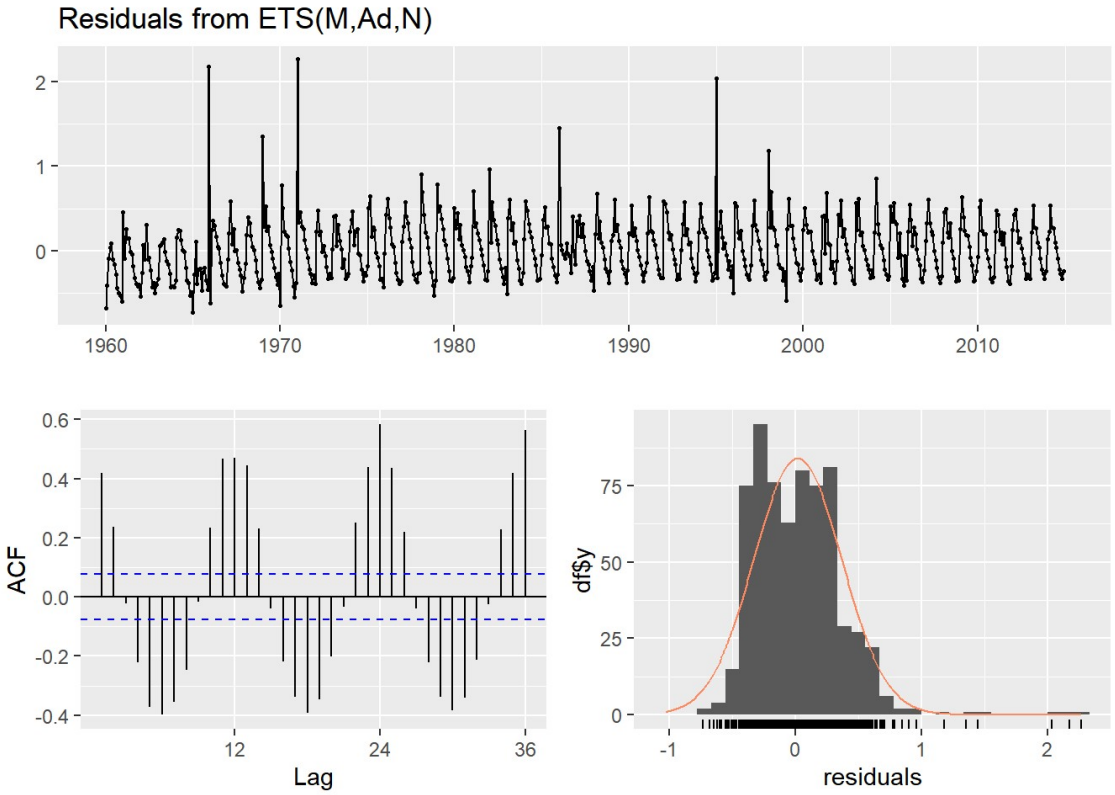



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,Ad,N)
## Q* = 1134, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit2.etsM = ets(solar.ts, model="MAN", damped = TRUE)
summary(fit2.etsM)
```

```
## ETS(M,Ad,N)
##
## Call:
##  ets(y = solar.ts, model = "MAN", damped = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.9815
##     beta  = 1e-04
##     phi   = 0.98
##
##   Initial states:
##     l = 9.9304
##     b = 5.8356
##
##   sigma:  0.35
##
##      AIC     AICc      BIC
## 6540.866 6540.995 6567.819
##
## Training set error measures:
##                      ME     RMSE      MAE       MPE     MAPE      MASE      ACF1
## Training set -0.4464292 4.75387 4.007741 -9.78188 29.60799 0.6583987 0.6870915
```

```
checkresiduals(fit2.etsM)
```

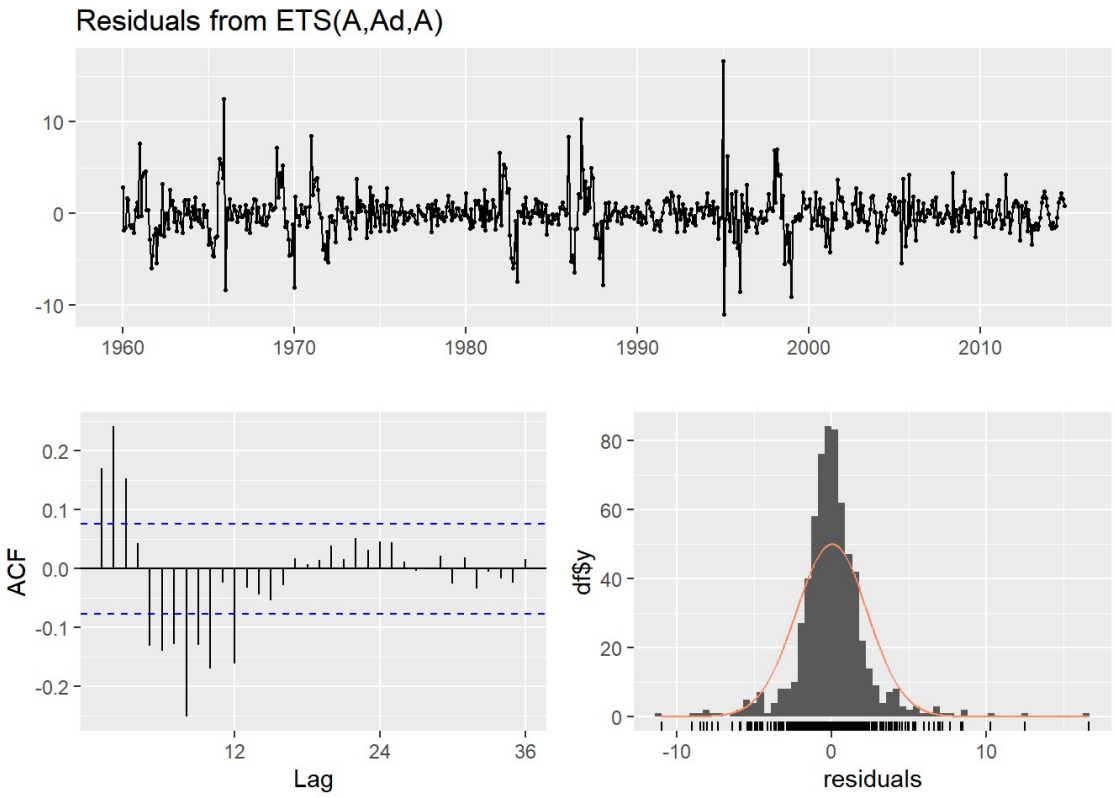



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,Ad,N)
## Q* = 1745.9, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit3.etsA = ets(solar.ts, model="AAA")
summary(fit3.etsA)
```

```
## ETS(A,Ad,A)
##
## Call:
##  ets(y = solar.ts, model = "AAA")
##
##   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
```

```
checkresiduals(fit3.etsA)
```

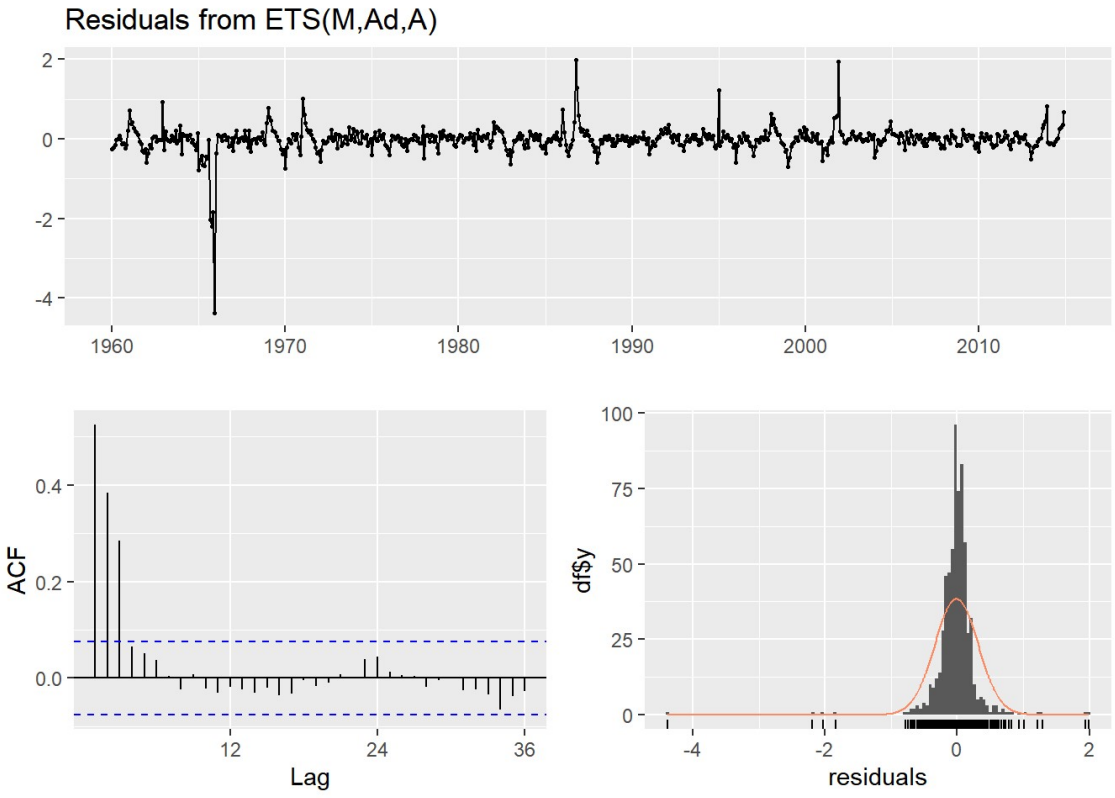



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(A,Ad,A)
## Q* = 210.76, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit3.etsM = ets(solar.ts, model="MAA")
summary(fit3.etsM)
```

```
## ETS(M,Ad,A)
##
## Call:
##  ets(y = solar.ts, model = "MAA")
##
##   Smoothing parameters:
##     alpha = 0.5273
##     beta  = 0.0035
##     gamma = 0.0104
##     phi   = 0.8
##
##   Initial states:
##     l = 11.5982
##     b = 2.6304
##     s = -10.4131 -6.5688 -2.8947 0.153 7.6029 8.4081
##            8.1855 8.4954 2.4863 -2.2582 -6.2468 -6.9497
##
##   sigma:  0.3329
##
##      AIC     AICc      BIC
## 6469.079 6470.146 6549.940
##
## Training set error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.02086702 3.282879 2.311939 -5.186754 18.98277 0.3798095
##                   ACF1
## Training set 0.551665
```

```
checkresiduals(fit3.etsM)
```

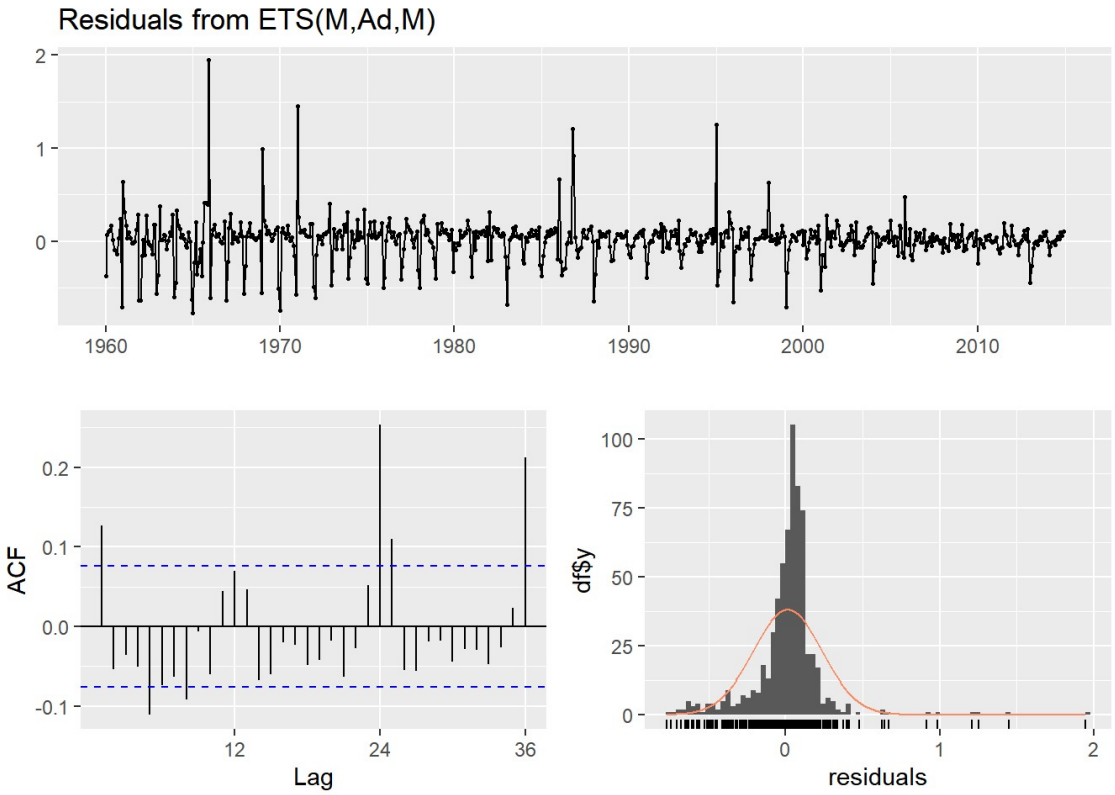


```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,Ad,A)
## Q* = 346.85, df = 24, p-value < 2.2e-16
##
## Model df: 0.   Total lags used: 24
```

```
fit4.etsM = ets(solar.ts, model="MAM")
summary(fit4.etsM)
```

```
## ETS(M,Ad,M)
##
## Call:
##  ets(y = solar.ts, model = "MAM")
##
##   Smoothing parameters:
##     alpha = 0.7922
##     beta  = 6e-04
##     gamma = 0.0707
##     phi   = 0.98
##
##   Initial states:
##     l = 9.3693
##     b = 1.2404
##     s = 0.6953 0.3066 0.5802 0.9879 1.3594 1.5469
##            1.4723 1.4055 1.227 0.9573 0.6959 0.7656
##
##   sigma:  0.2274
##
##      AIC     AICc      BIC
## 5953.502 5954.569 6034.363
##
## Training set error measures:
##                     ME    RMSE      MAE       MPE     MAPE      MASE      ACF1
## Training set 0.1750605 3.02791 1.961614 -5.497209 17.21119 0.3222574 0.2565219
```

```
checkresiduals(fit4.etsM)
```

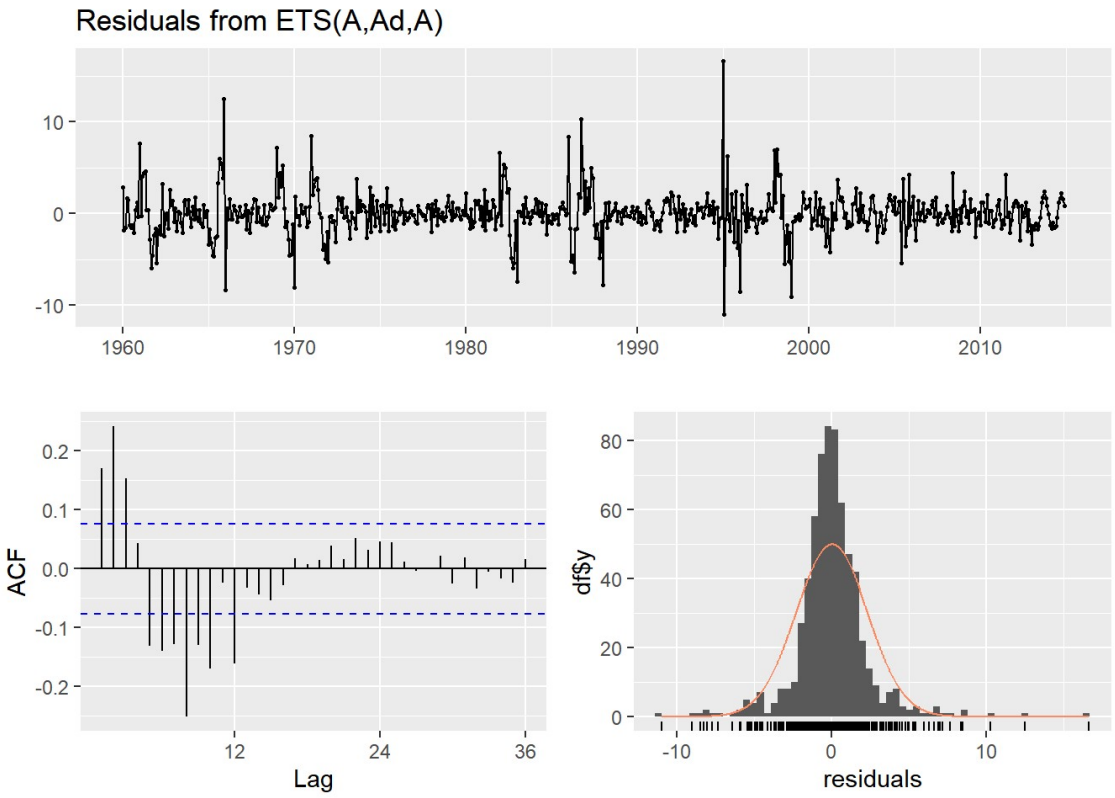



```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,Ad,M)
## Q* = 102.28, df = 24, p-value = 1.222e-11
##
## Model df: 0.   Total lags used: 24
```

```
fit5 = ets(solar.ts)
summary(fit5)
```

```
## ETS(A,Ad,A)
##
## Call:
##  ets(y = solar.ts)
##
##   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
```

```
checkresiduals(fit5)
```




```
##
##  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
```

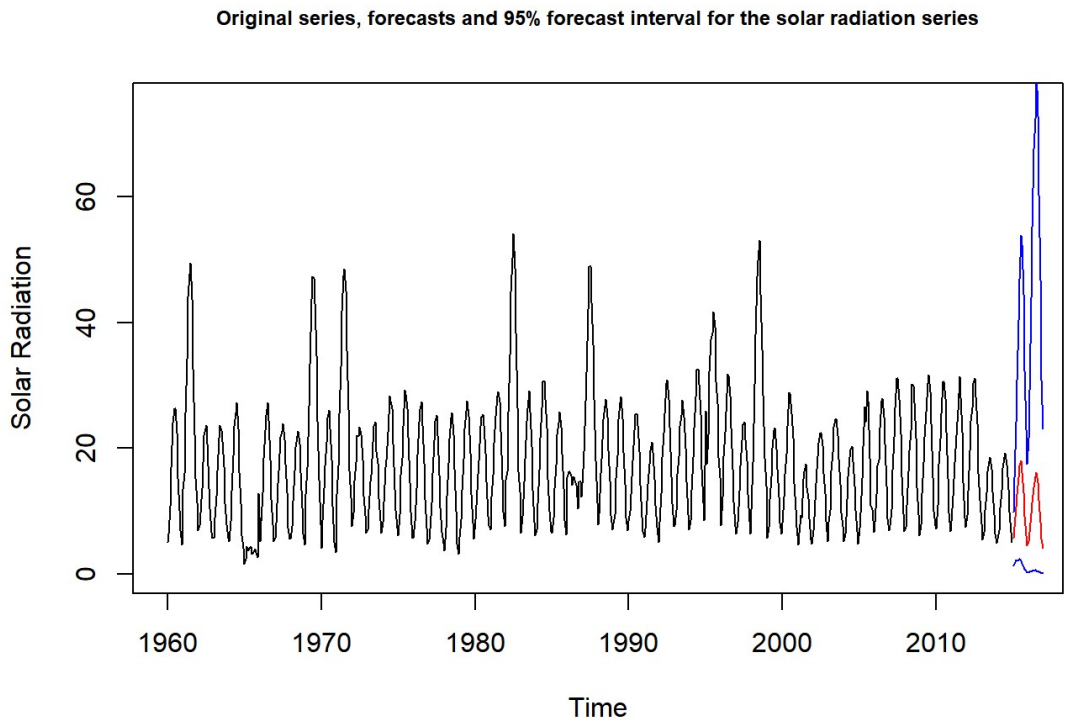
Model Summary

Model	AIC	BIC	MASE	Ljung-Box p-value	Residual Normality
Holt-Winter's Additive Seasonal	5434.708	5511.076	0.24716	2.2e-16	Normal
Holt-Winter's Additive Seasonal Damped	5428.422	5509.282	0.2461797	2.2e-16	Normal
Holt-Winter's Multiplicative Seasonal	6648.746	6725.114	0.2233077	0.03017	Normal
Holt-Winter's Multiplicative Seasonal Exponential	6584.208	6660.576	0.2320404	0.6242	Normal
ANN State Space	6296.371	6309.847	0.6368203	2.2e-16	Not Normal
MNN State Space	6619.776	6633.253	0.6369599	2.2e-16	Not Normal
AAN State Space	5932.524	5959.478	0.4334691	2.2e-16	Normal
MAN State Space	6540.866	6567.819	0.6583987	2.2e-16	Not Normal
AAA State Space	5428.422	5509.282	0.2461797	2.2e-16	Normal
MAA State Space	6469.079	6549.940	0.3798095	2.2e-16	Normal
MAM State Space	5953.502	6034.363	0.3222574	1.222e-11	Not Normal

Several more models are created which are listed in the above table. Their AIC, BIC, MASE, Ljung-Box p-value, and residual normality are listed as well. It is possible to shortlist the Holt-Winter's Additive Seasonal, Holt-Winter's Multiplicative Seasonal, Holt-Winters Multiplicative Seasonal Exponential, and AAA State Space based on their low MASE values. Of these, the Holt-Winter's Multiplicative Exponential Seasonal model is the only one that fails to reject the null hypothesis for the Ljung-Box test due to its p-value of 0.6242. This means, there is no serial correlation in the model. Holt-Winter's Multiplicative Seasonal also has a relatively high p-value of 0.03 for the Ljung-Box test but it is not high enough to reject the null hypothesis. The AAA State Space has the lowest AIC value as well as the lowest BIC value. All residual distributions for the shortlisted models are normally distributed. The ANN, MNN, MAN, and MAM state space models have non-normal distributions in their residuals as well as relatively high AIC and BIC values compared to the shortlisted models.

Based on these, the final model chosen was the Holt-Winter's Multiplicative Exponential Seasonal model due to its overall combination of no serial correlation and low MASE compared to the other models.

```
upper.95.int = fit8.hw$upper[,2]
lower.95.int = fit8.hw$lower[,2]
centre = fit8.hw$mean

plot(solar.ts, xlim = c(1960,2016), ylim = c(0,75),
     ylab="Solar Radiation",
     main = "Original series, forecasts and 95% forecast interval for the solar radiation series",
     cex.main=0.75)
lines(centre, col = "red")
lines(upper.95.int, col = "blue")
lines(lower.95.int, col = "blue")
```



Plotting the forecast along with the confidence intervals shows a forecast (red) that appears seasonal in nature similar to historical data. THe confidence intervals seem quite wide, especially the upper bound. ## Task 2

The objective of this task is to identify any spurious correlation that may exist between the Residential Property Price Index (PPI) in Melbourne and the quarterly population change over the previous quarter in Victoria. One initial way to determine this visually is by plotting the two series together to see if there are any similarities in the trends for both series. The plot obtained shows that the PPI and population change do seem correlated since they follow a similar trend, which may cause spurious correlation.

This is further confirmed by plotting the cross-correlation function (CCF) for the two series. The result of the CCF function demonstrates a clear case of spurious correlation since almost all the peaks lie beyond the 95% confidence intervals. Thus, it is possible to conclude that there xists spurious correlation between the two series.

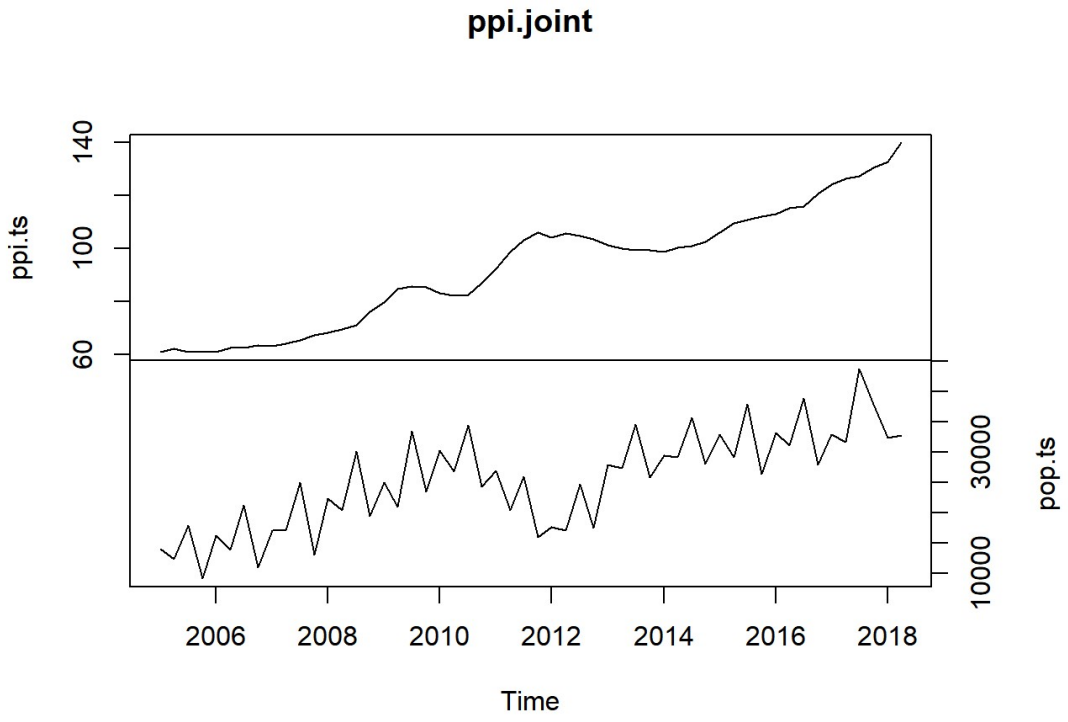
In order to remove the effects of cross correlation between the series, pre-whitening can be applied to both using the prewhiten() function. Pre-whitening attempts to remove dependence between the two input series to create a stationary series. Prior to applying the function, the second-order differences are obtained for both PPI and population change. These are then fed to the prewhiten() function and the new CCF is plotted.

Studying the new CCF plot shows that traces of cross-correlation are lost after pre-whitening. All the CCF peaks lie within the 95% confidence intervals and no obvious pattern is visible in the CCF plot.

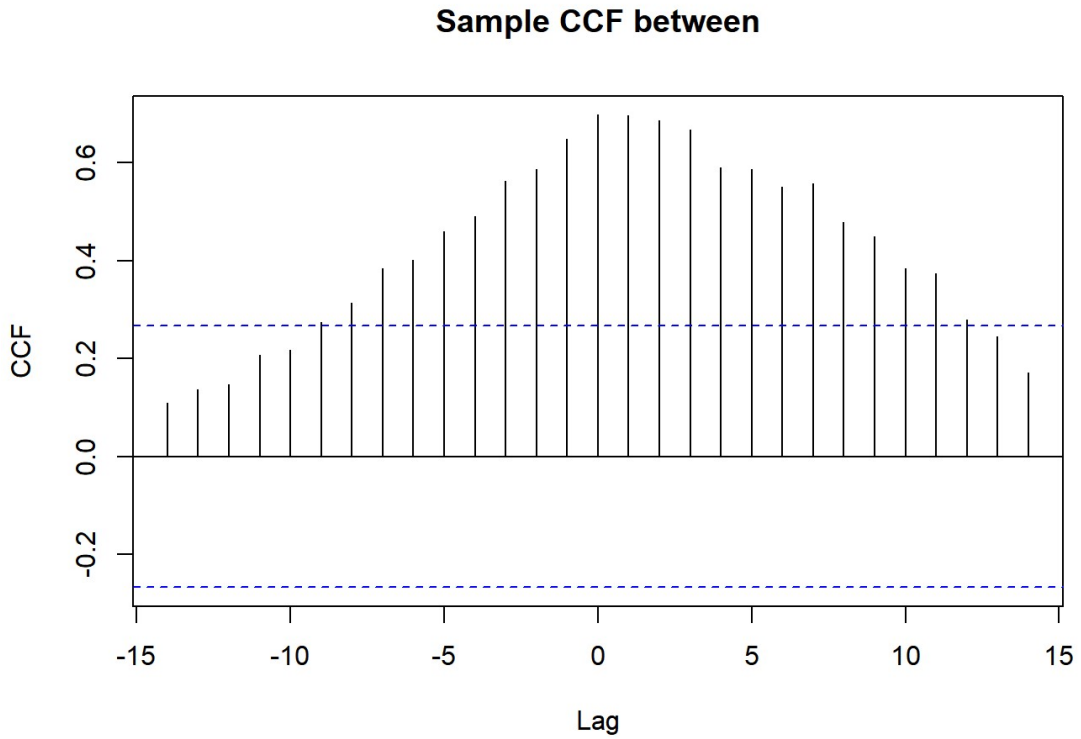
```
ppidata <- read.csv("data2.csv")
head(ppidata)
```

```
##    Quarter price change
## 1 Sep-2003  60.7  14017
## 2 Dec-2003  62.1  12350
## 3 Mar-2004  60.8  17894
## 4 Jun-2004  60.9   9079
## 5 Sep-2004  60.9  16210
## 6 Dec-2004  62.4  13788
```

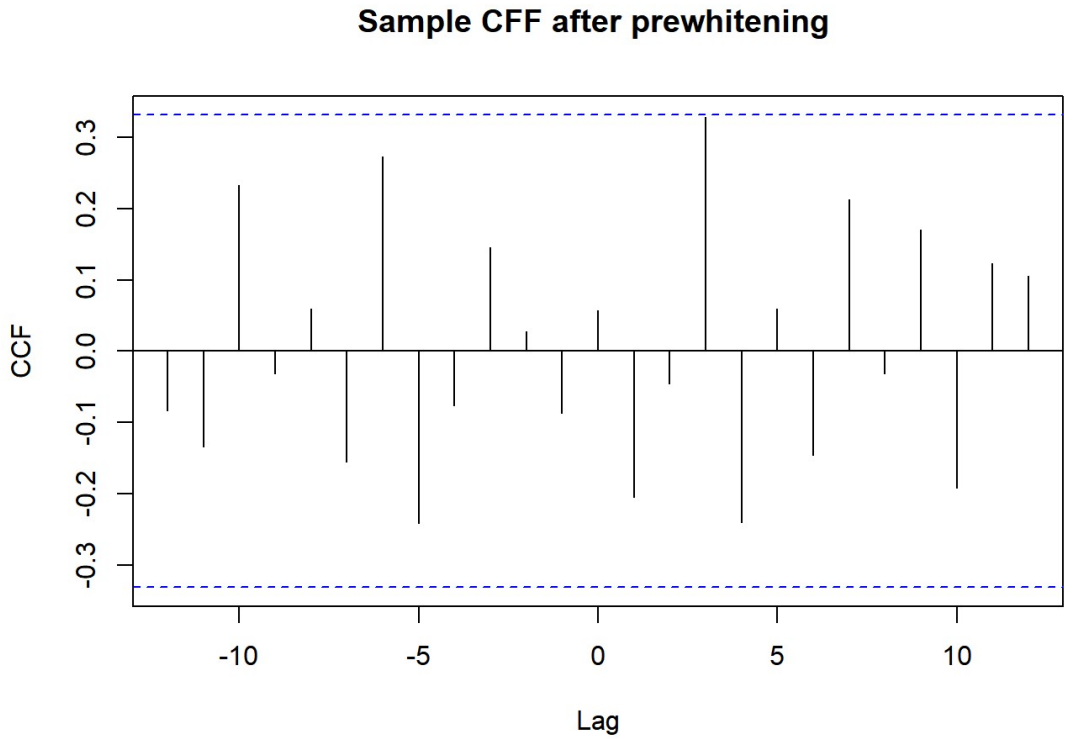
```
ppidata.ts <- ts(ppidata, start=c(2003,9), frequency=4)
ppi.ts <- ppidata.ts[,2]
pop.ts <- ppidata.ts[,3]
ppi.joint=ts.intersect(ppi.ts,pop.ts)
plot(ppi.joint, yax.flip=T)
```




```
ccf(as.vector(ppi.joint[,1]), as.vector(ppi.joint[,2]),
    ylab='CCF', main = "Sample CCF between")
```




```
me.dif=ts.intersect(diff(diff(ppi.ts,4)),diff(diff(pop.ts,4)))
prewhiten(as.vector(me.dif[,1]),as.vector(me.dif[,2]),ylab='CCF',
          main="Sample CFF after prewhitening ")
```



Conclusion

The procedures explored in this report explored various models including time series regression, exponential smoothing, and state-space models. It was discovered that both solar radiation and precipitation were both stationary since the ADF and Phillips-Perron tests had their null

hypotheses rejected. Further exploration of the data showed that there were seasonal components in the solar radiation and precipitation series. Next, various Distributed Lag Models, including the polynomial DLM, Koyck DLM, finite DLM, and ARDL model, were fitted these were later analysed by interpreting their summaries and residuals. Analysing the diagnostics from each model revealed that, for the solar radiation series, the ARDL model was the best time series regression model due to its goodness of fit and relatively low MASE compared to other models. The Koyck model was a close second with marginally less significant p-value and slightly higher residuals. Both the polynomial and finite DLM models were not significant and had low R-squared values, failing to describe the data properly.

Next, Holt-Winter's models and state space models were implemented for the series, producing various models based on whether the errors, trend, and seasonality were included, additive, or multiplicative. From analysing the MASE of these models as well as their AIC and BIC values and residuals, the AAN AAA State Space model and Holt-Winter's Multiplicative Exponential Seasonal model were chosen due to low MASE, AIC, BIC and normal residual distributions.

The purpose of the next task was to identify spurious correlation between the Residential Property Price Index (PPI) in Melbourne and the quarterly population change over the previous quarter in Victoria. A visual look at the plots of both series showed that they do seem correlated, and the CCF plot confirmed it due to multiple peaks beyond the 95% confidence intervals. Pre-whitening the 2nd differences of the series and feeding to the prewhiten() function removed correlations between the series and produced a CCF plot which confirmed the lack of cross-correlation with the differenced series.