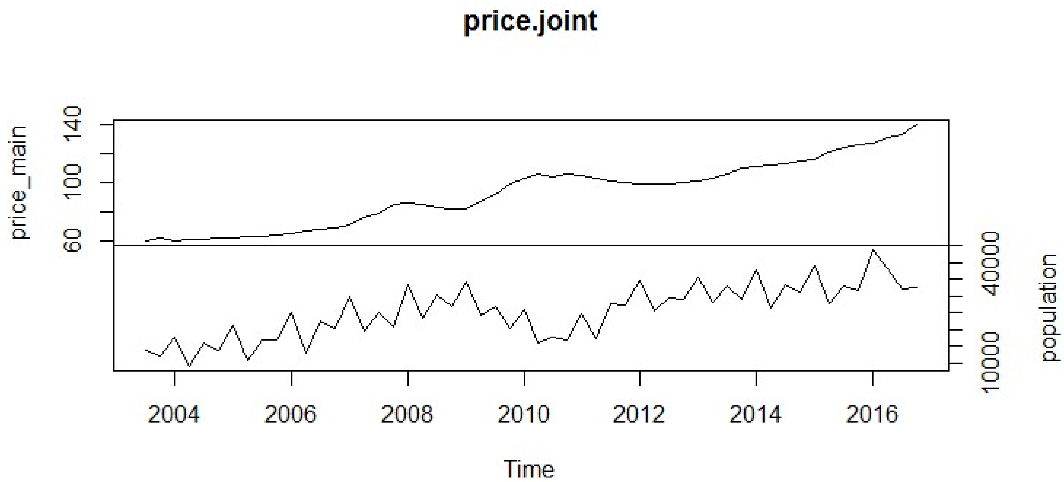
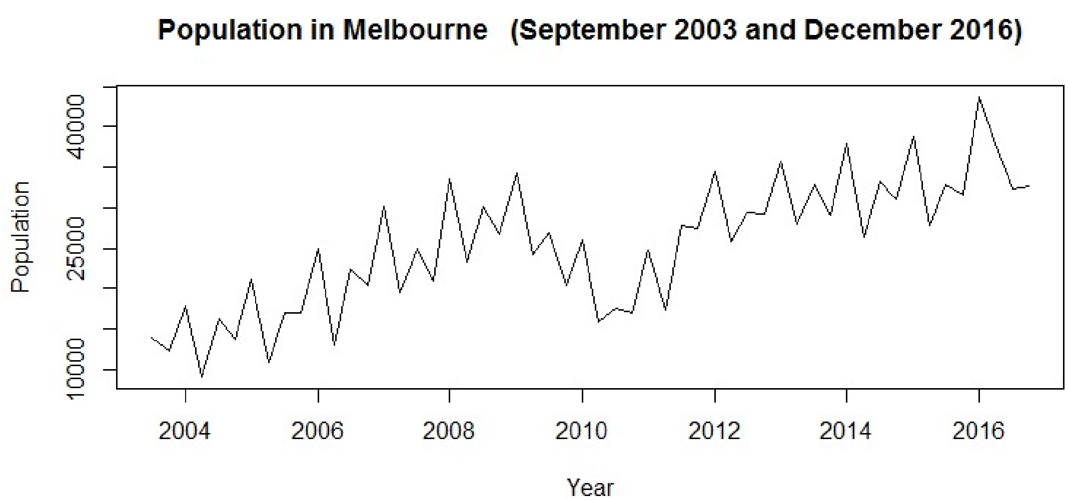
Forecasting Project

Priya Laxman

```
# Quietly load your packages
library("TSA")

library("forecast", "expsmooth")

library("x12", "Hmisc")

library("car", "AER")

library("tseries")
library("readxl")

library("stats")
library("ggplot2")

library("Hmisc")

library("forecast")
library("dLagM")

library("uroot")

library("xts")

library("dynlm")

#Read in the data

solardata<-read.csv("C:/Docs/Forecasting/Assignment 2/Data/data1.csv") #Data
for task 1
price<-read.csv("C:/Docs/Forecasting/Assignment 2/Data/data2.csv")
datax<-read.csv("C:/Docs/Forecasting/Assignment 2/Data/datax.csv") #Data for
task 1
dataxvals<-datax[,1] #vector for precipitation test/forecast values

#Task 2

price1 = ts(price, start=c(2003,3),frequency = 4)
price_main = price1[,2]
population = price1[,3]
price.joint=ts.intersect(price_main,population)
plot(price.joint,yax.flip=T)
```

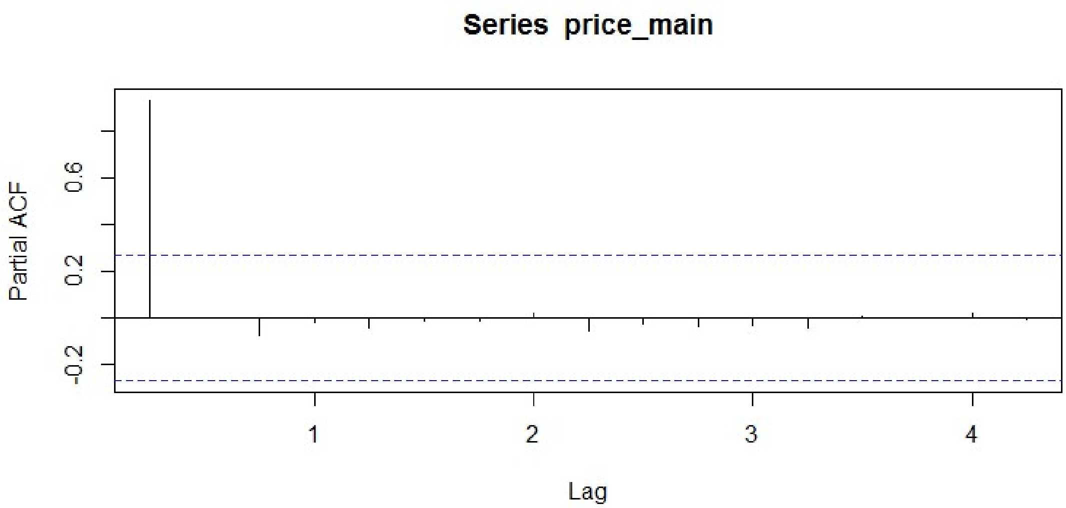



```r
#Features of above series as examined in the plot:
#a. Upward trend
#b. Rising levels increasing from first quarter to the fourth.
#c. Intervention:  The reduction in real mortgage rates since 2011 -
following reductions in the cash rate - has been closely associated with
both stronger housing price growth and strong dwelling construction more rece
ntly
#During 2011, there was a significant easing in monetary policy.

plot(population,ylab='Population',xlab='Year', main = "Population in Melbourn
e   (September 2003 and December 2016)")
```

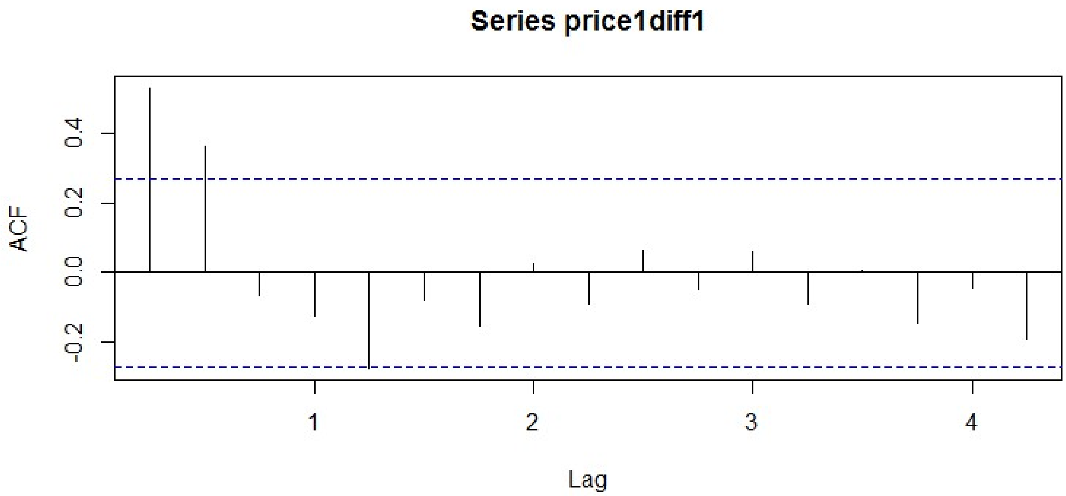



```r
acf(price_main)
```

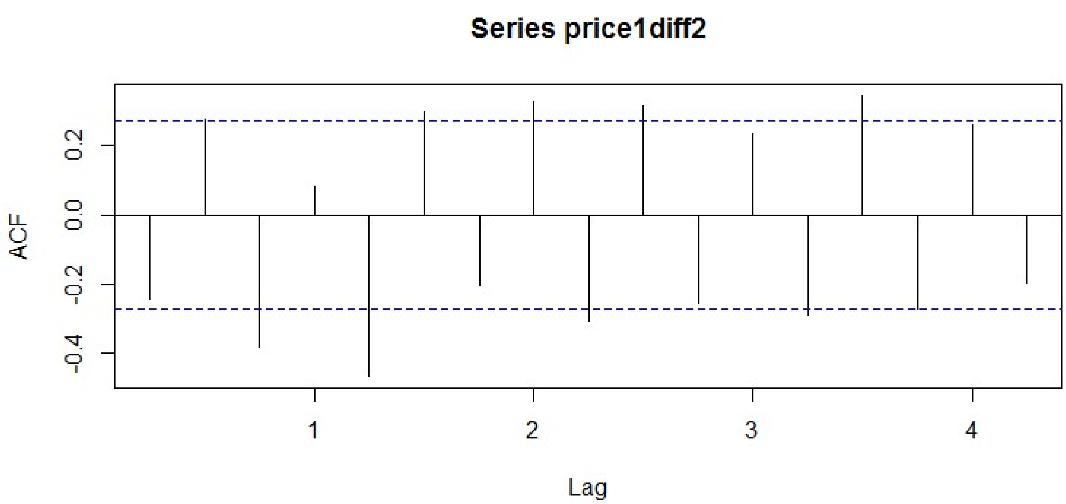

```r
pacf(price_main)
```



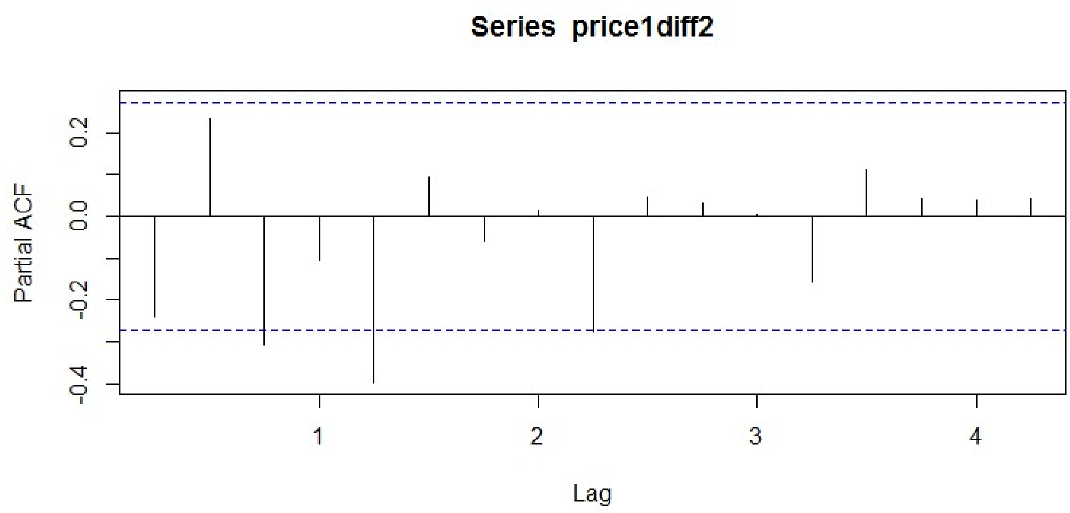

```r
#Trend exists. Decaying pattern in ACF plot and a very high first lag in
PACF plot which implies need to difference

price1diff1=diff(price_main)
price1diff2=diff(price_main,differences = 2)
acf(price1diff1)
```



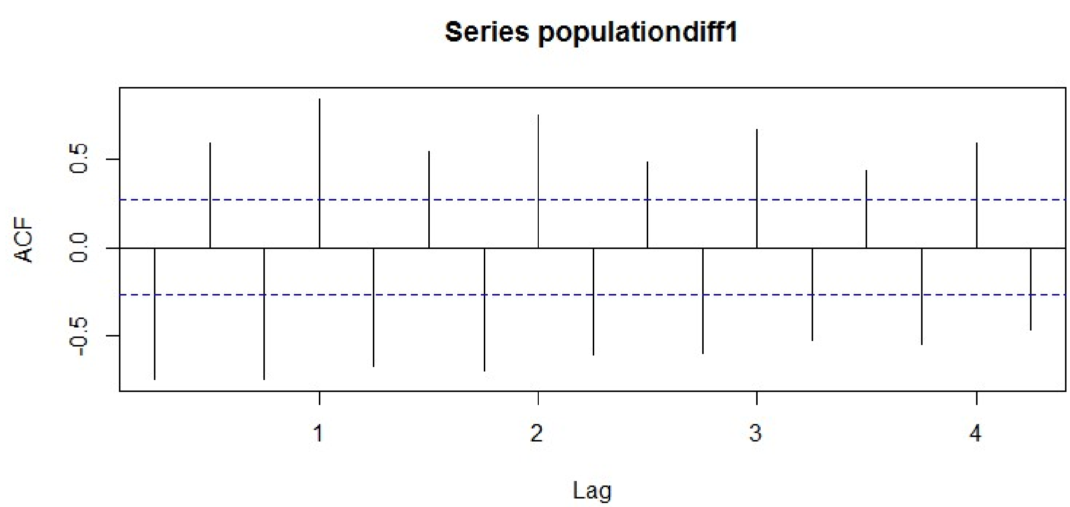

```
acf(price1diff2)
```



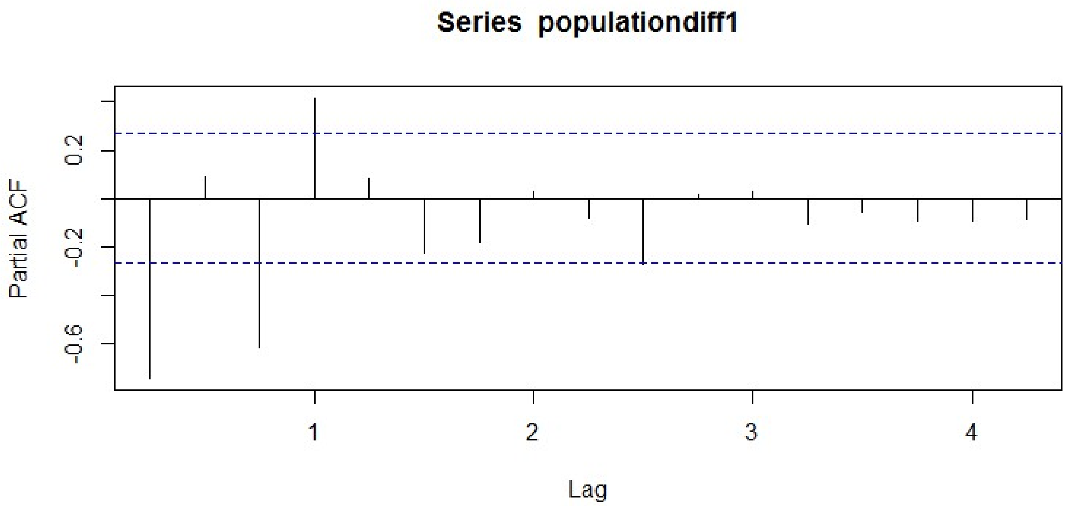

```
pacf(price1diff2)
```



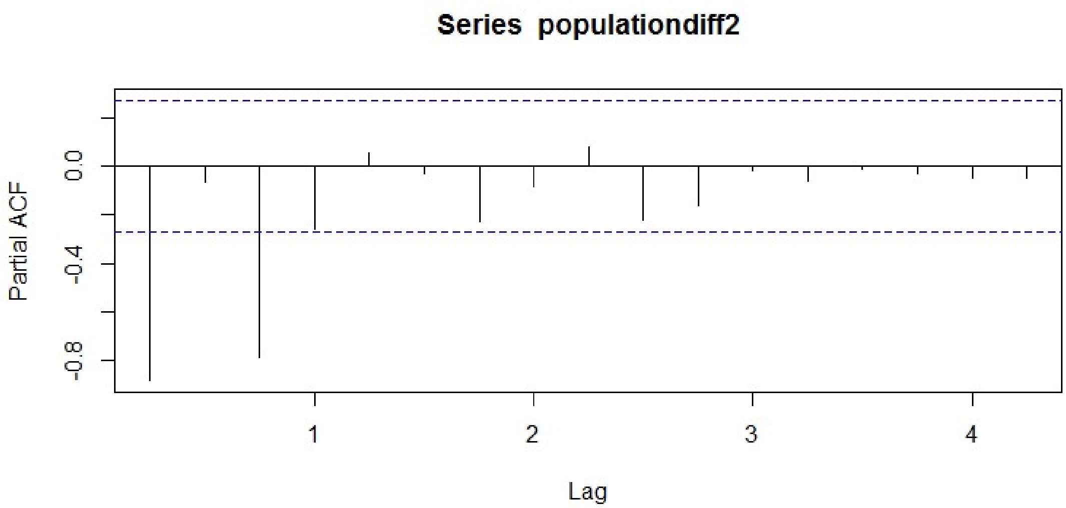

```
adf.test(price1diff2)

##
##  Augmented Dickey-Fuller Test
##
## data:  price1diff2
## Dickey-Fuller = -4.1299, Lag order = 3, p-value = 0.01077
## alternative hypothesis: stationary

#Stationary at difference of 2
populationdiff1=diff(population)
populationdiff2=diff(population,differences = 2)
acf(populationdiff1)
```




```
pacf(populationdiff1)
```

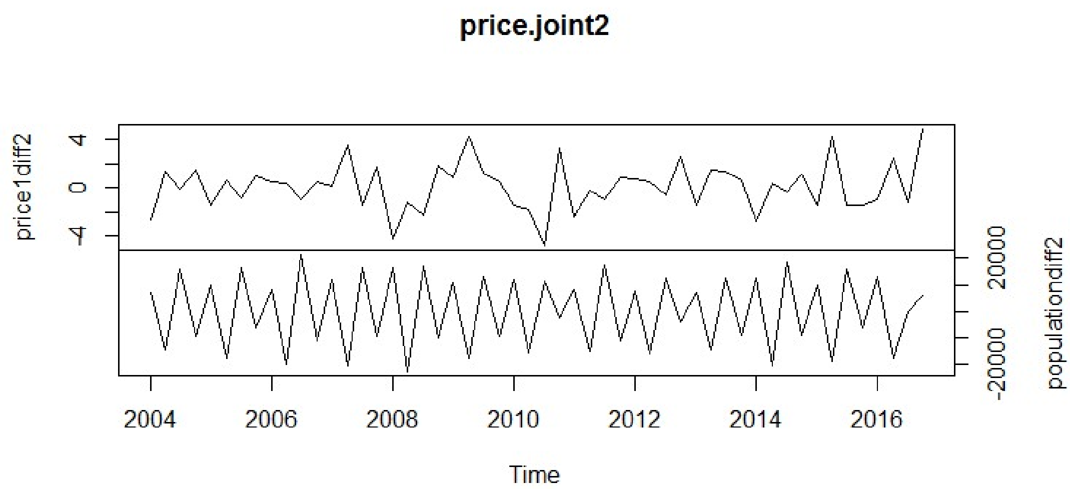



```
pacf(populationdiff2)
```

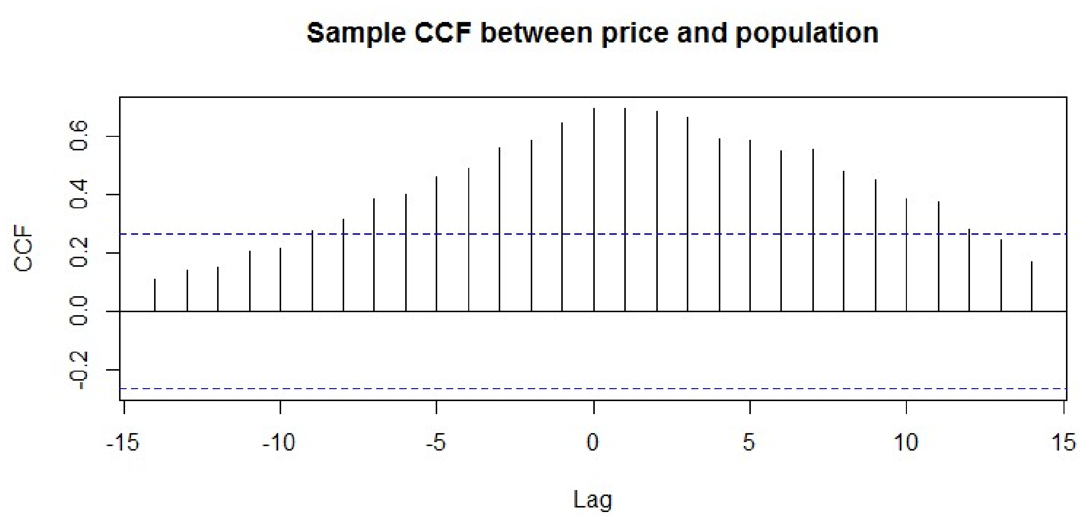



```
adf.test(populationdiff2)

## Warning in adf.test(populationdiff2): p-value smaller than printed p-value

##
##  Augmented Dickey-Fuller Test
##
## data:  populationdiff2
## Dickey-Fuller = -7.0677, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary

#Stationary at difference of 2

price.joint2=ts.intersect(price1diff2,populationdiff2)
```

```
plot(price.joint2,yax.flip=T)
```

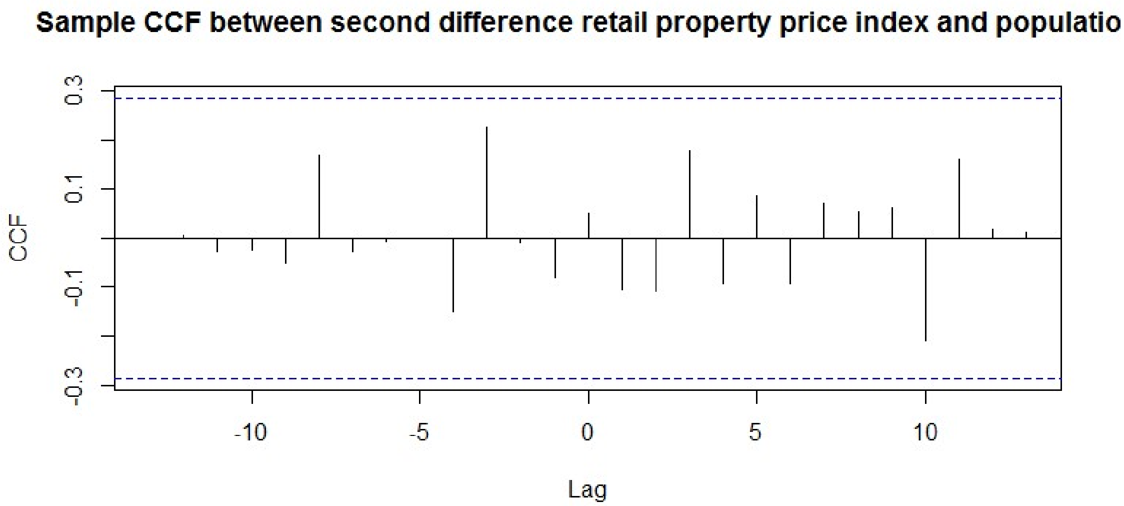



```
cor(price_main, population)

## [1] 0.6970439

cor(price1diff2, populationdiff2)

## [1] -0.5108004

#correlation of ~50% after making stationary the correlation has decreased.

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



```
#CCF of differenced data

ccf(as.vector(price.joint2[,1]), as.vector(price.joint2[,2]),ylab='CCF',
main = "Sample CCF between second difference retail property price index and
population")
```

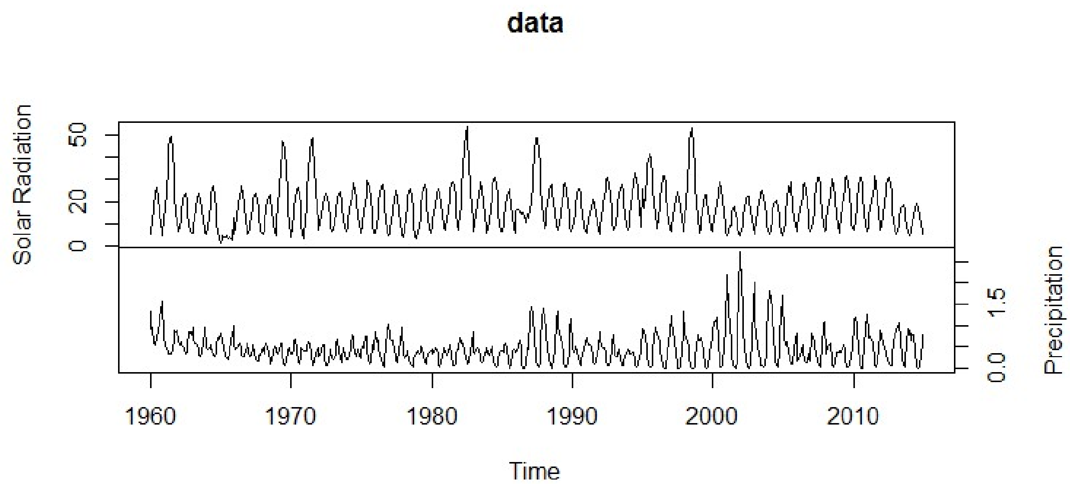

```
#When we display the stationary versions of series, the number of singificant
lags in the sample CCF plot is considerably decreased. However, this is not e
nough to conclude that there is no spurious correlation.

#Nearly all of the cross-correlations are significantly different from zero.
Obviously it is difficult to come up with a plausible reason for such a stron
g relationship between quarterly retail property price index and quarterly po
pulation change. The nonstationarity in the retail property price index serie
s and in the population series is more likely the cause of the spurious corre
lations found between the two series.

# Hence, we conduct pre whitening to remove spurious correlation and disentan
gle the linear association

prewhiten(as.vector(price.joint2[,1]), as.vector(price.joint2[,2]),ylab='CCF'
, main = "Sample CCF between second difference retail property price index an
d population")
```



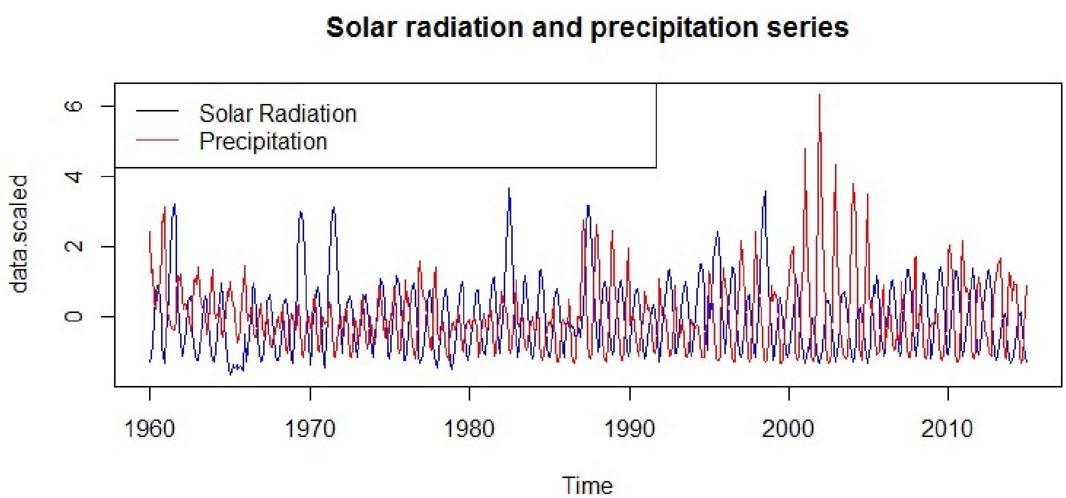

```
#There are no significant correlations in the differenced data and after prew
hitening. Thus, it seems that retail property price index and population chan
ge in Melbourne are in fact largely uncorrelated, and the strong cross-correl
ation pattern found between the raw data series is indeed spurious.

#Functions needed for Task 1

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

    options(warn=-1)

    if(!missing(...)) {# Several models
      models = list(model, ...)
      m = length(models)
      for (j in 1:m){
        if ((class(models[[j]])[1] == "polyDlm") | (class(models[[j]])[1] ==
"dlm") | (class(models[[j]])[1] == "koyckDlm") | (class(models[[j]])[1] == "a
rdlDlm")){
          Y.t = models[[j]]$model$model$y.t
          fitted = models[[j]]$model$fitted.values
        } else if (class(models[[j]])[1] == "lm"){
```

```
          Y.t = models[[j]]$model[,1]
          fitted = models[[j]]$fitted.values
        } else if (class(models[[j]])[1] == "dynlm"){
            Y.t = models[[j]]$model$Y.t
            fitted = models[[j]]$fitted.values
        } else {
          stop("MASE function works for lm, dlm, polyDlm, koyckDlm, and ardlD
lm objects. Please make sure that you are sending model object directly or se
nd a bunch of these objects to the function.")
        }
        # Y.t = models[[j]]$model$y.t
        # fitted = models[[j]]$fitted.values
        n = length(fitted)
        e.t = Y.t - fitted
        sum = 0
        for (i in 2:n){
          sum = sum + abs(Y.t[i] - Y.t[i-1] )
        }
        q.t = e.t / (sum/(n-1))
        if (j == 1){
          MASE = data.frame( n = n , MASE = mean(abs(q.t)))
          colnames(MASE) = c("n" , "MASE")
        } else {
          MASE = rbind(MASE, c(n , mean(abs(q.t))))
        }
      }
      Call <- match.call()
      row.names(MASE) = as.character(Call[-1L])
      MASE
    } else { # Only one model
      if ((class(model)[1] == "polyDlm") | (class(model)[1] == "dlm") | (clas
s(model)[1] == "koyckDlm") | (class(model)[1] == "ardlDlm")){
        Y.t = model$model$model$y.t
        fitted = model$model$fitted.values
      } else if (class(model)[1] == "lm"){
        Y.t = model$model[,1]
        fitted = model$fitted.values
      } else if (class(model)[1] == "dynlm"){
        Y.t = model$model$Y.t
        fitted = model$fitted.values
      } else {
        stop("MASE function works for lm, dlm, polyDlm, koyckDlm, and ardlDlm
objects. Please make sure that you are sending model object directly or send
one of these objects to the function.")
      }
      n = length(fitted)
      e.t = Y.t - fitted
      sum = 0
      for (i in 2:n){
        sum = sum + abs(Y.t[i] - Y.t[i-1] )
```

```
      }
      q.t = e.t / (sum/(n-1))
      MASE = data.frame( MASE = mean(abs(q.t)))
      colnames(MASE) = c("MASE")
      Call <- match.call()
      row.names(MASE) = as.character(Call[-1L])
      MASE
    }

}



View(solardata)
class(solardata)

## [1] "data.frame"

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

#Task 1

#convert to time series object from January 1960 to December 2014

solar = ts(solardata$solar,start = c(1960,1), frequency=12)
ppt = ts(solardata$ppt,start = c(1960,1), frequency=12)
solardata.ts = ts(solardata[,1:2],start = c(1960,1), frequency = 12)

# To create two separate time series plots in the same window
data = ts.intersect(solar , ppt)
colnames(data) =  c("Solar Radiation","Precipitation")
plot(data , yax.flip=T)
```

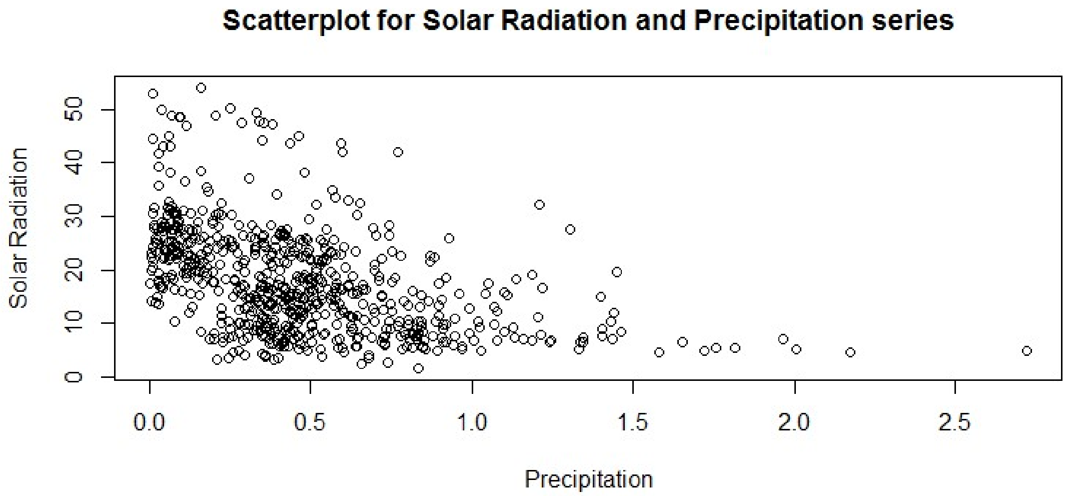



```
# We can scale and center both series to see in the same plot clearly
data.scaled = scale(solardata.ts)
plot(data.scaled, plot.type="s",col = c("blue", "red"), main = "Solar radiati
on and precipitation series")
legend("topleft",lty=1, text.width = 28, col=c("black","red"), c("Solar Radia
tion","Precipitation"))
```

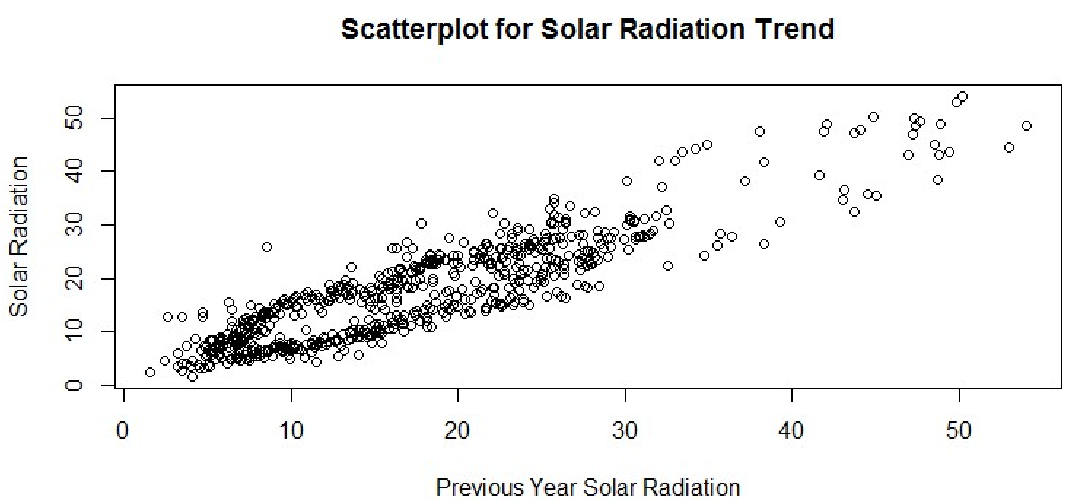



```
#Correlation check
cor(solardata.ts)

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

#Scatterplot for solar and ppt series
```

```
plot(y=solar,x=ppt,ylab='Solar Radiation',xlab='Precipitation', main="Scatter
plot for Solar Radiation and Precipitation series")
```

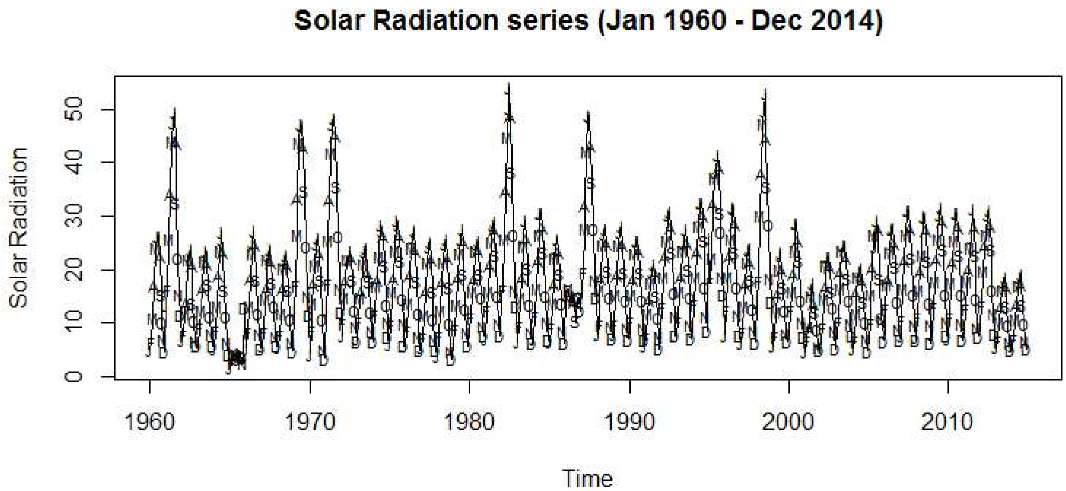



```
#There is slight negative correlation but it is lesser than 50% and cannot be
considered as significant amount of correlation

plot(y=solar,x=zlag(solar),ylab='Solar Radiation',xlab='Previous Year Solar R
adiation', main="Scatterplot for Solar Radiation Trend")
```

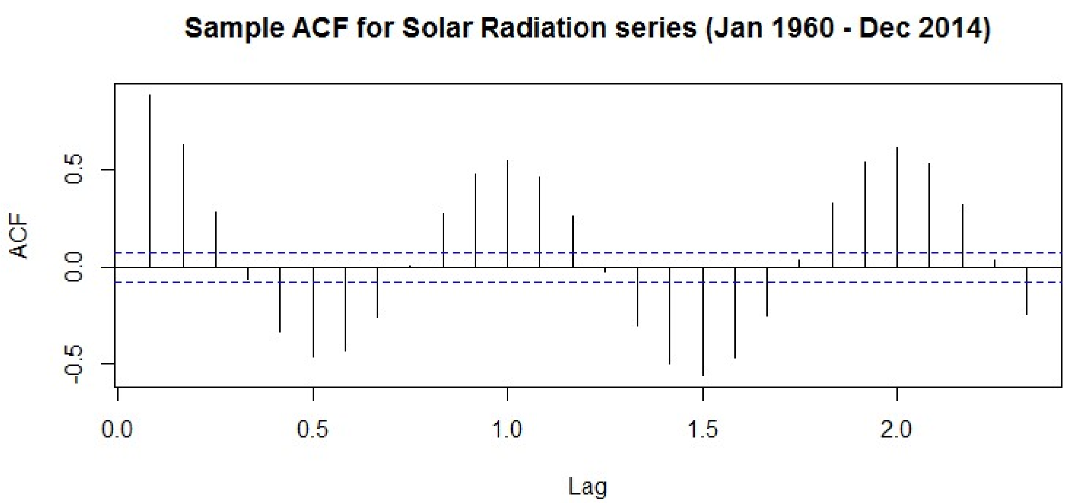



```
#As seen above there is a slight upward trend in the solar radiation series w
hen compared to previous year value

#Trend Stationarity Identification

plot(solar,ylab='Solar Radiation',xlab='Time', main = "Solar Radiation series
```

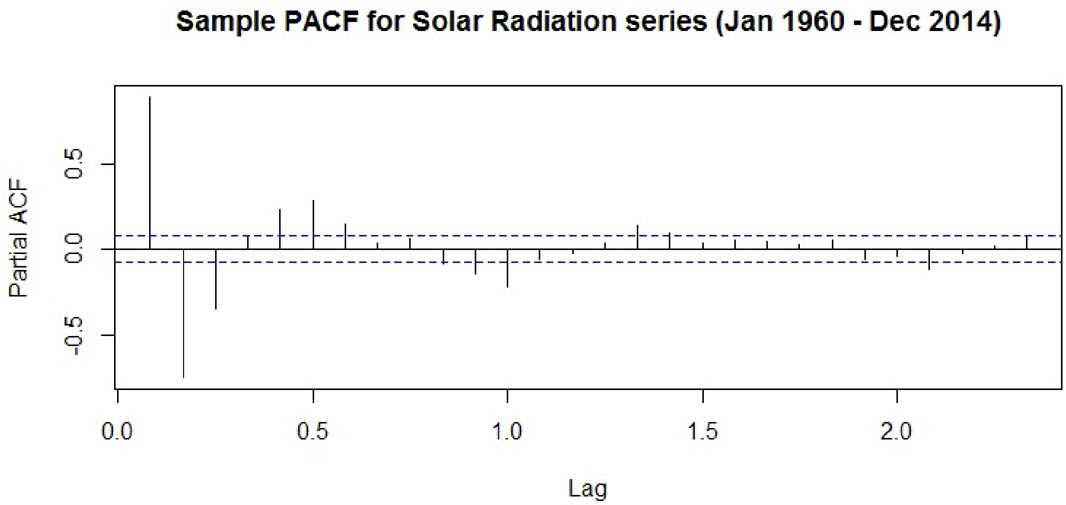
```
(Jan 1960 - Dec 2014)")
points(y=solar,x=time(solar), pch=as.vector(season(solar)), cex=0.7)
```



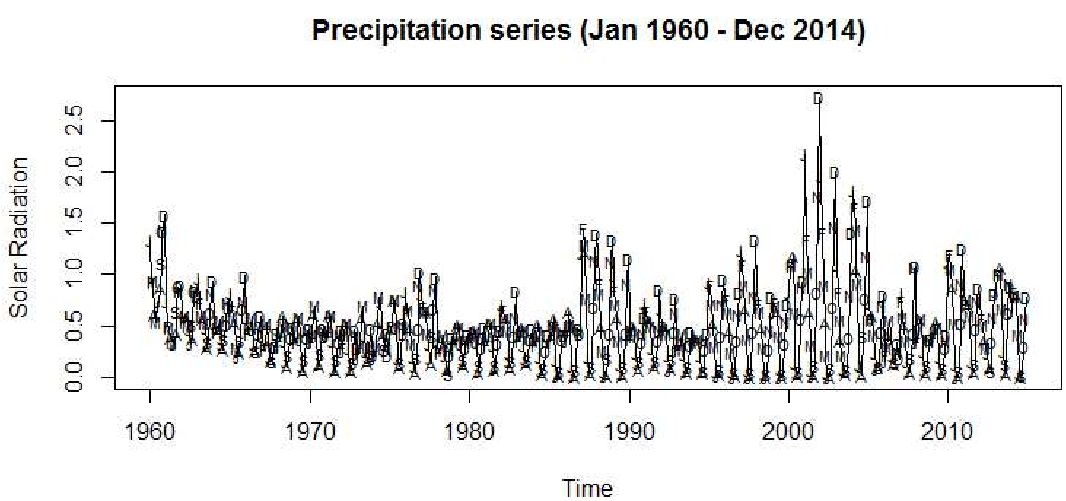

```
#As can be seen from the above plot, there is an apparent seasonal trend, wit
h low levels in January. The trend reaches its peak at mid year and then ther
e is a decreasing trend till the end of the year. The revolution around the S
un determines the fluctuations that are visible at monthly scales

#The variance is non constant and high levels are observed in the series duri
ng the years 1961, 1969, 1971, 1982, 1987, 1995, 1998. The series appears to
stabilize and variance decreases in the 2000s.

acf(solar, main="Sample ACF for Solar Radiation series (Jan 1960 - Dec 2014)"
)
```



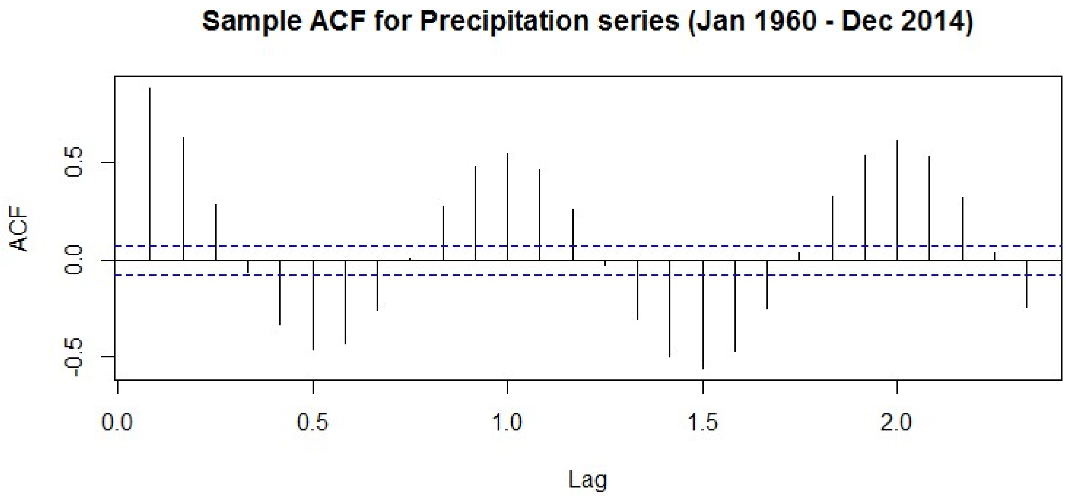
```r
pacf(solar, main="Sample PACF for Solar Radiation series (Jan 1960 - Dec 2014
)")
```



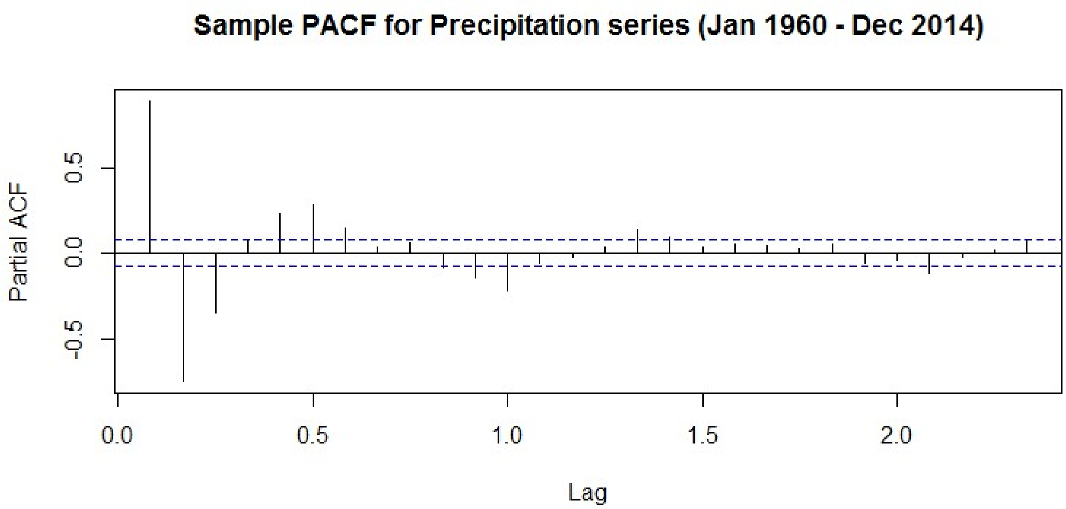

```r
#PACF and ACF reconfirm the seasonal trend as suggested by the plot and shows
seasonal trend

plot(ppt,ylab='Solar Radiation',xlab='Time', main = "Precipitation series (Ja
n 1960 - Dec 2014)")
points(y=ppt,x=time(ppt), pch=as.vector(season(ppt)), cex=0.7)
```



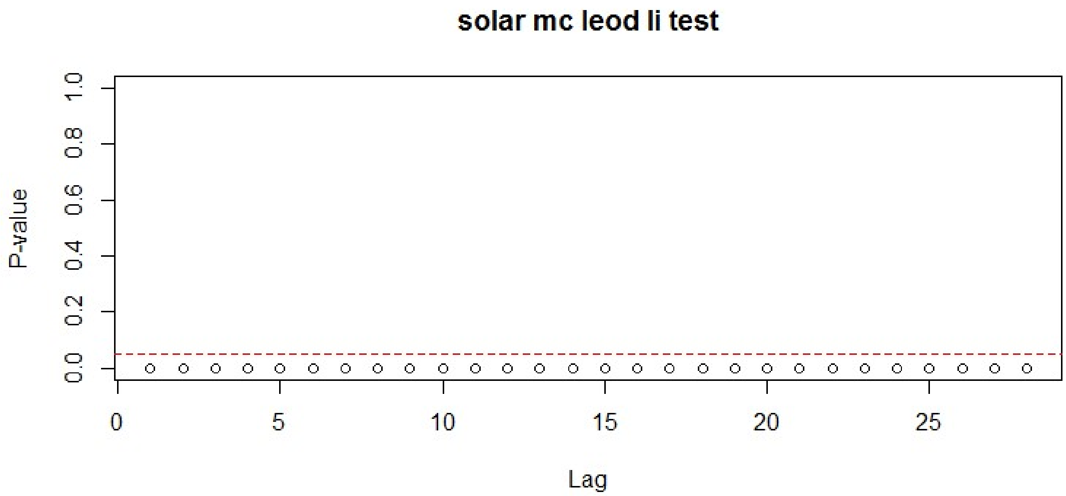

```r
acf(solar, main="Sample ACF for Precipitation series (Jan 1960 - Dec 2014)")
```



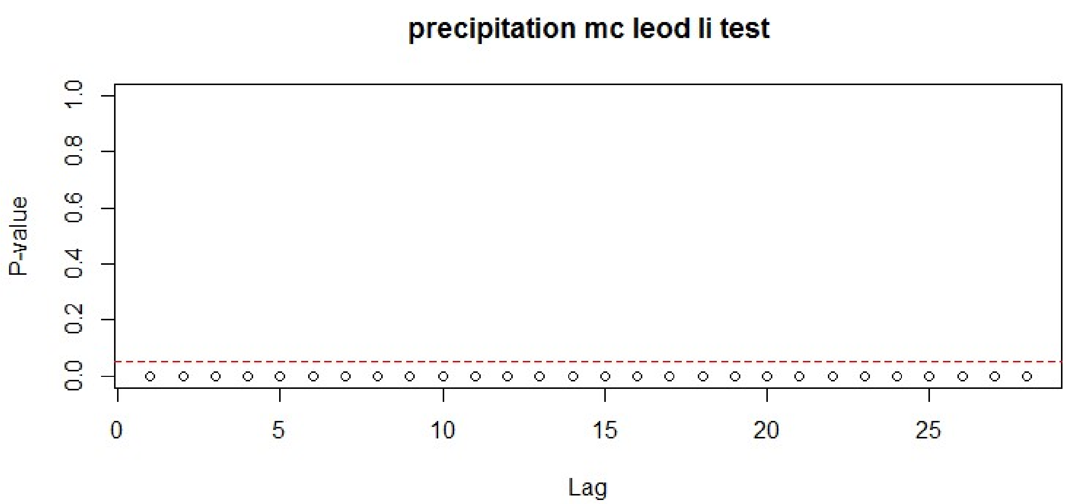

```r
pacf(solar, main="Sample PACF for Precipitation series (Jan 1960 - Dec 2014)"
)
```




```r
#Precipitation data also shos seasonality and changing variance.

#Mc Leod Li Test for conditional heteroscedasticity
McLeod.Li.test(y=solar, main="solar mc leod li test")
```

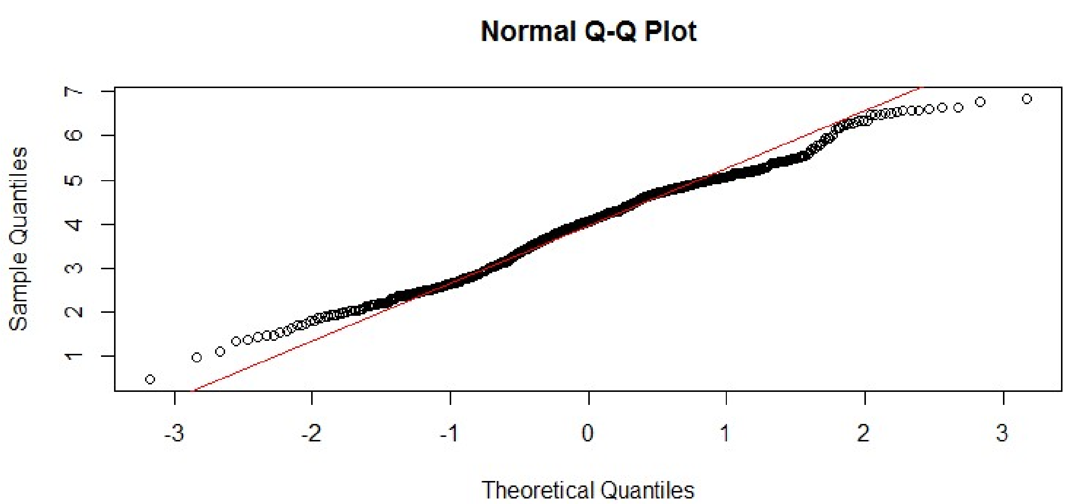



```r
McLeod.Li.test(y=ppt, main="precipitation mc leod li test")
```




```r
#From the Mc Leod Li Test, hence we reject the null hypothesis stating the ab
sence of conditional heteroscedasticity and there seems to be changing varian
ce in the solar time series. Hence a seasonal adjustment of both the series i
s needed.

#Shapiro test for normality
shapiro.test(solar)

##
##  Shapiro-Wilk normality test
##
## data:  solar
## W = 0.93637, p-value = 3.641e-16
```

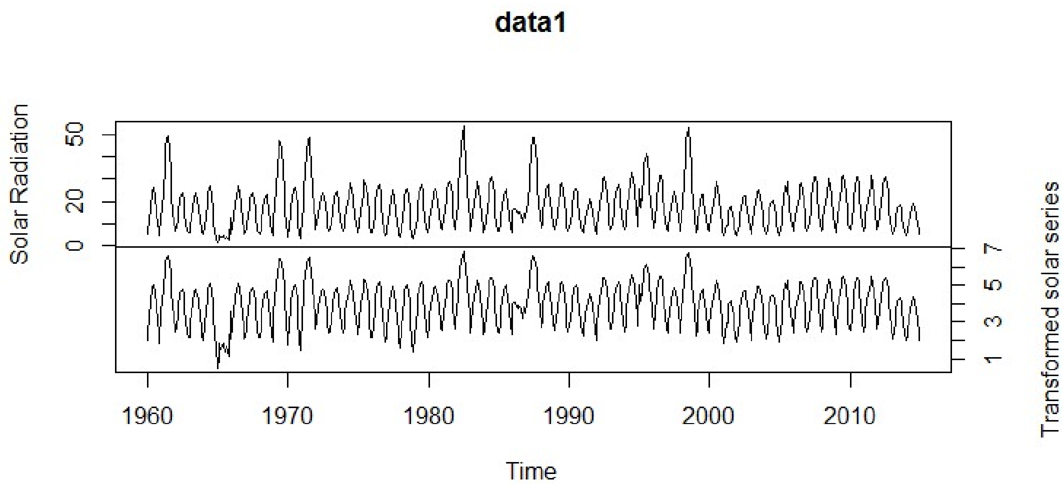
```
shapiro.test(ppt)

##
##  Shapiro-Wilk normality test
##
## data:  ppt
## W = 0.8947, p-value < 2.2e-16

#p-value implies data is not normal

#Time series with seasonality are not stationary - the trend and seasonality
will affect the value of the time series at different times

#The series shows:

#a. Seasonality
b. Changing Variance or Heteroscedasticity due to seasonal variation
c. Non stationarity caused by seasonality

#since trend is present we need to transform the data
#We proceed to finding approximate value of lambda using Box Cox power transf
ormation

lambda1=BoxCox.lambda(solar, method="loglik")  #lambda=0.25
BC.solar = ((solar^lambda1-1)/lambda1)
qqnorm(BC.solar)
qqline(BC.solar, col = 2)
```



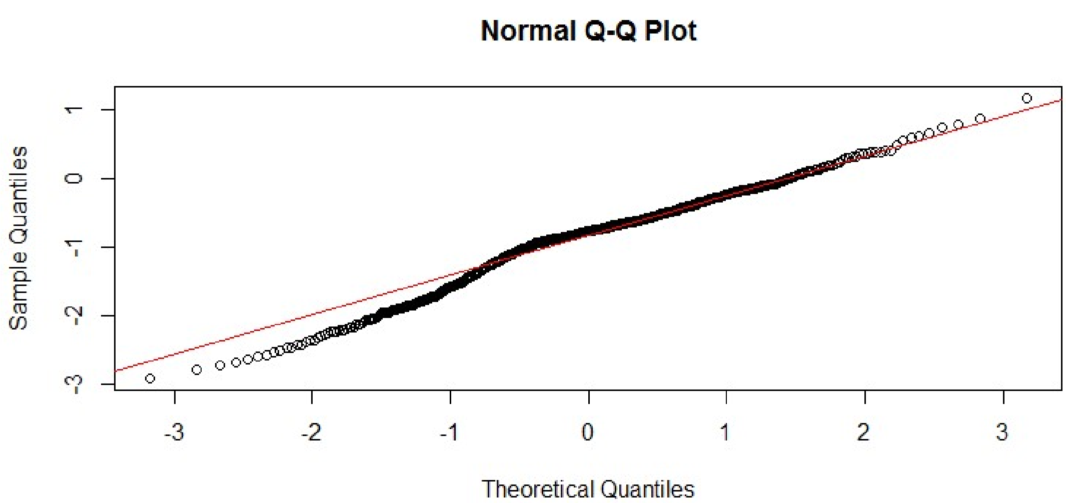

```
shapiro.test(BC.solar)

##
##  Shapiro-Wilk normality test
##
```

```
## data:  BC.solar
## W = 0.98834, p-value = 4.126e-05
```

```
data1= ts.intersect(solar , BC.solar)
colnames(data1) =  c("Solar Radiation","Transformed solar series")
plot(data1 , yax.flip=T)
```

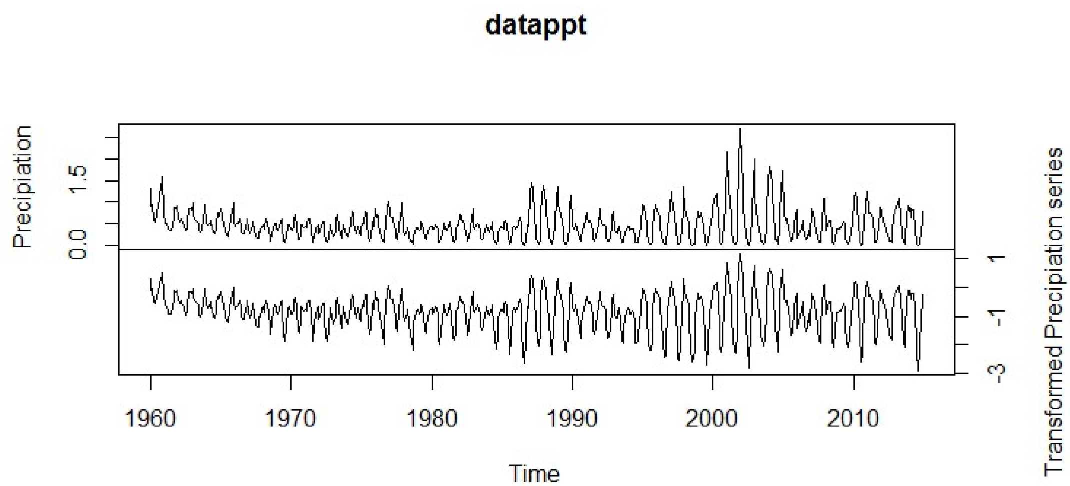



```
#As can be seen from the above plot, using lambda=0.25 transformation the dat
a bounces more closely around the mean level

lambda2=BoxCox.lambda(ppt, method="loglik")  #lambda=0.3
BC.ppt = ((ppt^lambda2-1)/lambda2)
qqnorm(BC.ppt)
qqline(BC.ppt, col = 2)
```

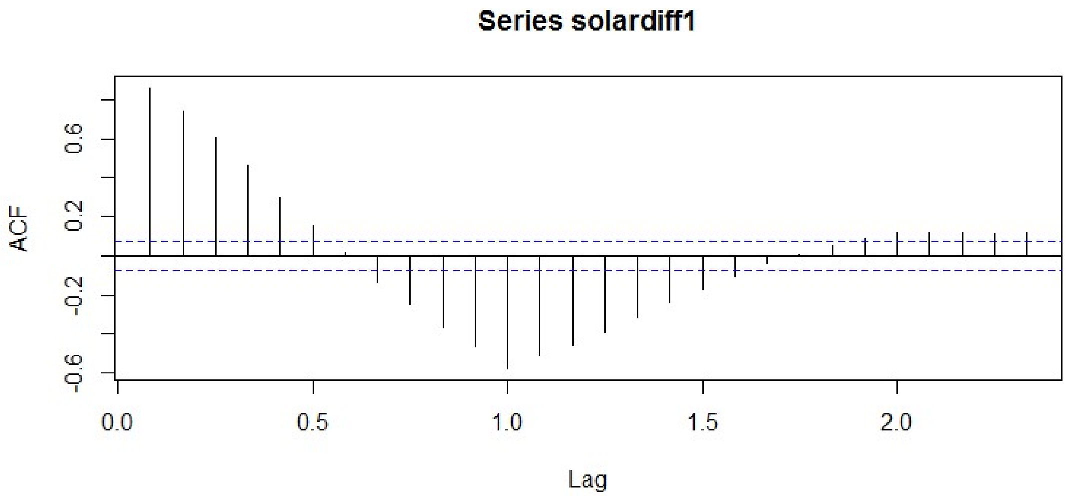



```
shapiro.test(BC.ppt)
```

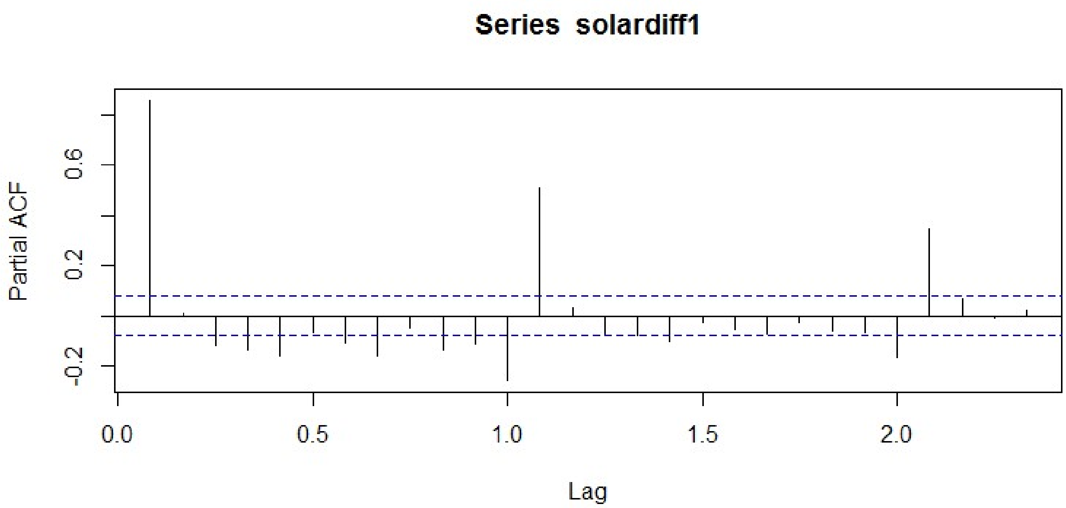
```
##
##  Shapiro-Wilk normality test
##
## data:  BC.ppt
## W = 0.9788, p-value = 3.498e-08

datappt= ts.intersect(ppt , BC.ppt)
colnames(datappt) =  c("Precipiation","Transformed Precipiation series")
plot(datappt, yax.flip=T)
```



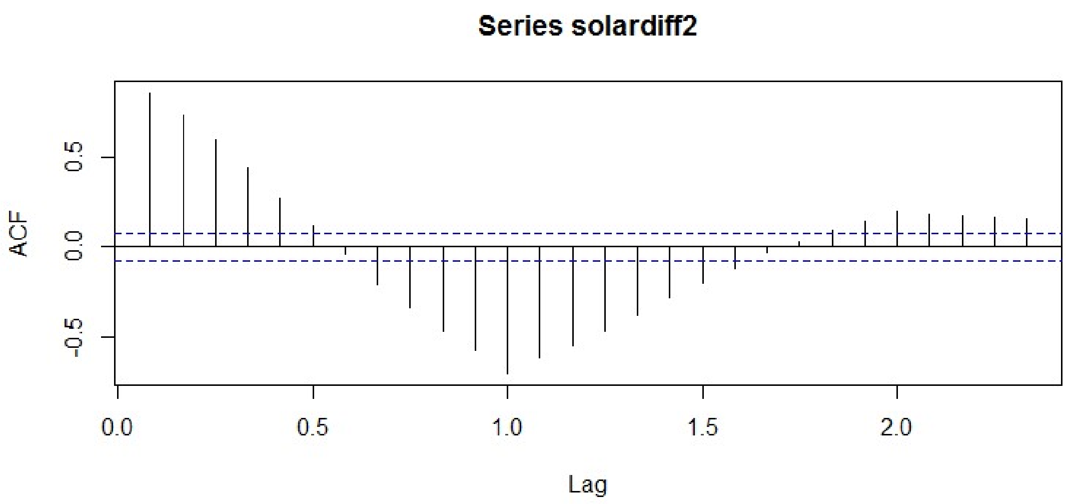

```
#As can be seen from the above plot, using lambda=0.25 transformation the dat
a bounces more closely around the mean level

#Now we proceed to seasonal differencing of solar data

#First difference:

solardiff1=diff(BC.solar,lag = frequency(BC.solar))
acf(solardiff1)
```



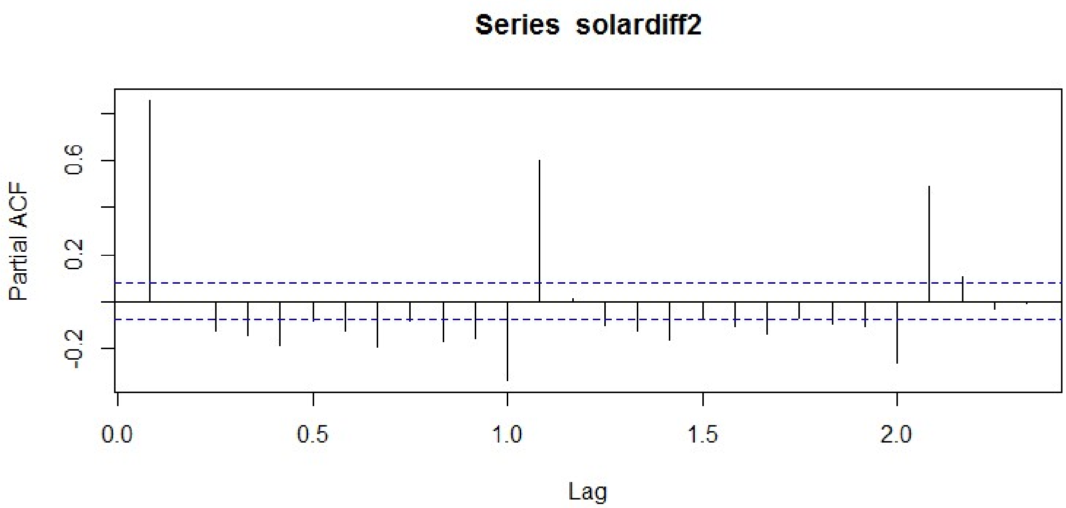

```
pacf(solardiff1)
```



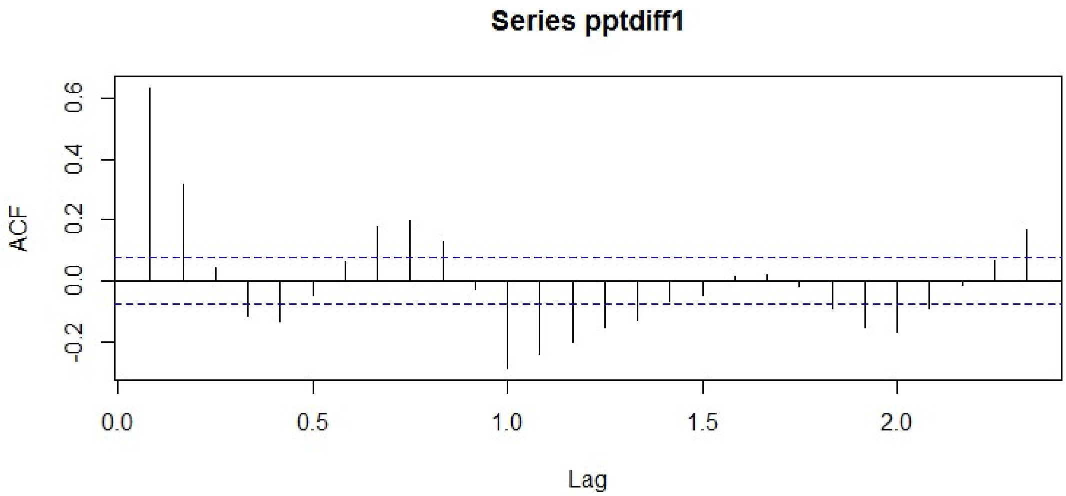

```
#Still pattern in lags exist. We proceed to second difference:

solardiff2=diff(BC.solar,differences=2,lag = frequency(BC.solar))
acf(solardiff2)
```



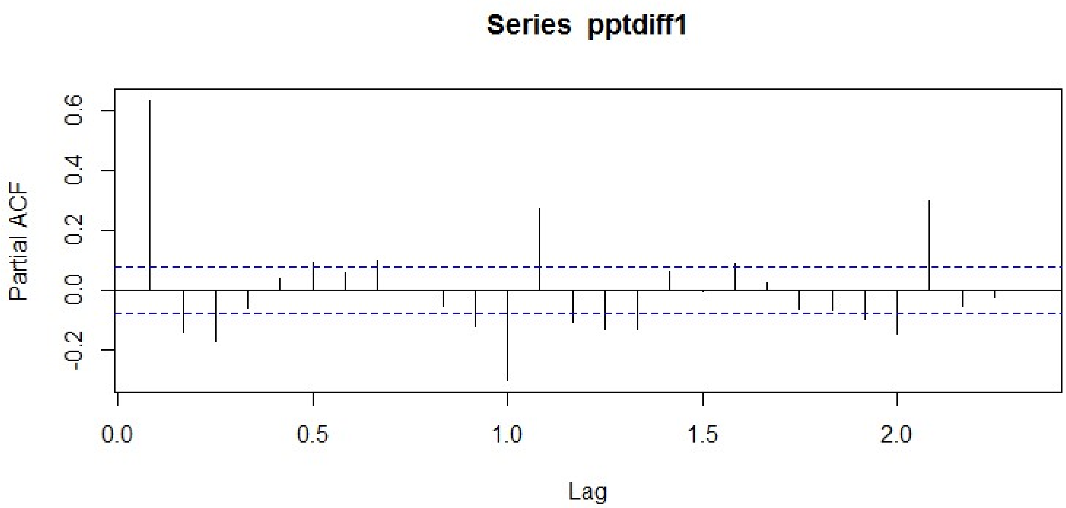

```
pacf(solardiff2)
```



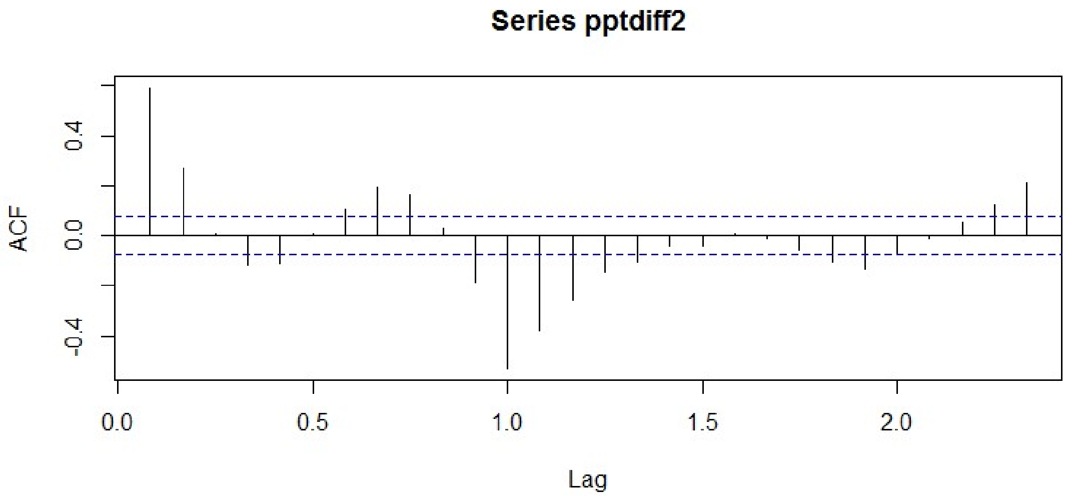

```
#pacf lags do not decay after second difference and there are random lags sig
nifying the differenced data is stationary. solardiff2 is the final transform
ed and seasonally differenced data

#Now we proceed to seasonal differencing of ppt data

#First difference:

pptdiff1=diff(BC.ppt,lag = frequency(BC.ppt))
acf(pptdiff1)
```



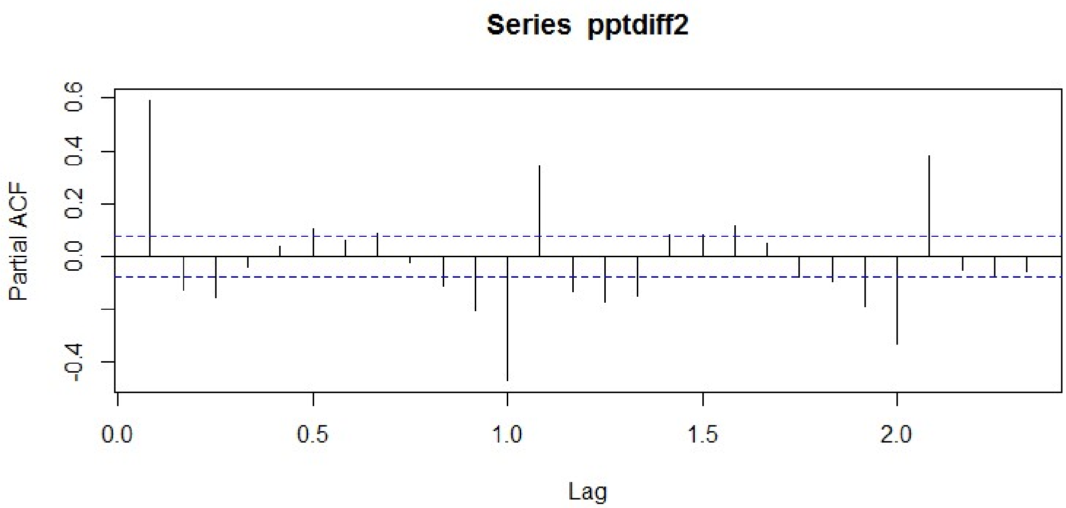

```
pacf(pptdiff1)
```




```
#Still pattern in lags exist. We proceed to second difference:

pptdiff2=diff(BC.ppt,differences=2,lag = frequency(BC.ppt))
acf(pptdiff2)
```




```r
pacf(pptdiff2)
```




```r
cor(BC.solar,BC.ppt)

## [1] -0.5071714

#Transformed series show greater correlation

#Checking dLagM models and USING THE dLagM PACKAGE

model.12 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 12 , show.summ
ary = TRUE)$model

##
## Call:
## lm(formula = y.t ~ ., data = design)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -18.563  -5.239  -0.796   4.137  32.430 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  19.5164     1.1151  17.501  < 2e-16 ***
## x.t          -5.8876     1.9508  -3.018  0.00265 ** 
## x.1           0.9993     2.5647   0.390  0.69694    
## x.2           0.4343     2.5571   0.170  0.86520    
## x.3           1.8763     2.5580   0.734  0.46352    
## x.4           1.7459     2.5587   0.682  0.49529    
## x.5           3.3279     2.5601   1.300  0.19410    
## x.6           0.7751     2.5617   0.303  0.76230    
## x.7           1.7937     2.5615   0.700  0.48402    
## x.8           0.2827     2.5593   0.110  0.91207    
## x.9          -1.1022     2.5615  -0.430  0.66712    
## x.10         -1.9333     2.5508  -0.758  0.44880    
## x.11         -0.5613     2.5532  -0.220  0.82605    
## x.12         -5.3492     1.9216  -2.784  0.00553 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 8.181 on 634 degrees of freedom
## Multiple R-squared:  0.3216, Adjusted R-squared:  0.3077 
## F-statistic: 23.12 on 13 and 634 DF,  p-value: < 2.2e-16
## 
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4578.787 4645.895
```

```
model.11 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 11 , show.summ
ary = TRUE)$model
```

```
## 
## Call:
## lm(formula = y.t ~ ., data = design)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -19.030  -5.271  -0.807   4.159  31.657 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  19.2922     1.1042  17.471  < 2e-16 ***
## x.t          -6.0353     1.9541  -3.089  0.00210 ** 
## x.1          -0.2507     2.5318  -0.099  0.92116    
## x.2           0.1376     2.5643   0.054  0.95724    
## x.3           1.5953     2.5680   0.621  0.53467    
## x.4           1.9434     2.5698   0.756  0.44978
```

```
## x.5           3.3519     2.5721   1.303  0.19299
## x.6           0.8739     2.5711   0.340  0.73403
## x.7           1.4599     2.5702   0.568  0.57023
## x.8           0.5150     2.5699   0.200  0.84122
## x.9          -0.7612     2.5605  -0.297  0.76636
## x.10         -0.7162     2.5244  -0.284  0.77674
## x.11         -5.2899     1.9296  -2.741  0.00629 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.22 on 636 degrees of freedom
## Multiple R-squared:  0.3149, Adjusted R-squared:  0.302
## F-statistic: 24.36 on 12 and 636 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4590.961 4653.617
```

```
model.10 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 10 , show.summ
ary = TRUE)$model
```

```
##
## Call:
## lm(formula = y.t ~ ., 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
```

```
model.9 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 9 , show.summar
y = TRUE)$model
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.989  -5.524  -0.926   4.113  31.653
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  18.5144     1.0785  17.167  < 2e-16 ***
## x.t          -8.8366     1.8321  -4.823 1.77e-06 ***
## x.1          -0.4598     2.5518  -0.180   0.8571
## x.2           0.2048     2.5833   0.079   0.9368
## x.3           1.8941     2.5904   0.731   0.4649
## x.4           2.0805     2.5913   0.803   0.4223
## x.5           3.1502     2.5905   1.216   0.2244
## x.6           0.7586     2.5893   0.293   0.7696
## x.7           2.2688     2.5769   0.880   0.3790
## x.8           2.0341     2.5430   0.800   0.4241
## x.9          -4.6741     1.8132  -2.578   0.0102 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.296 on 640 degrees of freedom
## Multiple R-squared:  0.2998, Adjusted R-squared:  0.2888
## F-statistic:  27.4 on 10 and 640 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4615.084 4668.827
```

```
model.8 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 8 , show.summar
y = TRUE)$model
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.594  -5.703  -1.197   4.183  31.840
##
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  17.8827     1.0509  17.016  < 2e-16 ***
## x.t         -10.0720     1.7675  -5.699 1.84e-08 ***
## x.1           0.1511     2.5494   0.059    0.953    
## x.2           0.4438     2.5912   0.171    0.864    
## x.3           2.0741     2.5977   0.798    0.425    
## x.4           1.6863     2.5964   0.649    0.516    
## x.5           3.3938     2.5973   1.307    0.192    
## x.6           1.3674     2.5825   0.530    0.597    
## x.7           3.3675     2.5396   1.326    0.185    
## x.8          -2.6471     1.7514  -1.511    0.131    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 8.327 on 642 degrees of freedom
## Multiple R-squared:  0.2924, Adjusted R-squared:  0.2825 
## F-statistic: 29.48 on 9 and 642 DF,  p-value: < 2.2e-16
## 
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4625.986 4675.267
```

```
model.7 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 7 , show.summar
y = TRUE)$model
```

```
## 
## Call:
## lm(formula = y.t ~ ., data = design)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -19.024  -5.780  -1.140   4.378  31.561 
## 
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  17.3522     0.9990  17.370  < 2e-16 ***
## x.t         -10.4305     1.7464  -5.972 3.87e-09 ***
## x.1           0.4790     2.5404   0.189    0.850    
## x.2           0.6702     2.5885   0.259    0.796    
## x.3           1.8691     2.5958   0.720    0.472    
## x.4           1.7578     2.5963   0.677    0.499    
## x.5           3.8639     2.5806   1.497    0.135    
## x.6           2.0049     2.5269   0.793    0.428    
## x.7           0.6935     1.7308   0.401    0.689    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 8.331 on 644 degrees of freedom
## Multiple R-squared:  0.2898, Adjusted R-squared:  0.281 
## F-statistic: 32.85 on 8 and 644 DF,  p-value: < 2.2e-16
```

```
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4632.716 4677.532
```

```
model.6 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 6 , show.summar
y = TRUE)$model
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.136  -5.796  -1.202   4.354  31.403
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.4381     0.9244  18.865  < 2e-16 ***
## x.t         -10.3695     1.7430  -5.949 4.42e-09 ***
## x.1           0.4397     2.5294   0.174   0.8621
## x.2           0.6780     2.5840   0.262   0.7931
## x.3           1.7948     2.5913   0.693   0.4888
## x.4           1.8136     2.5767   0.704   0.4818
## x.5           3.5003     2.5159   1.391   0.1646
## x.6           2.8893     1.7274   1.673   0.0949 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.322 on 646 degrees of freedom
## Multiple R-squared:  0.2898, Adjusted R-squared:  0.2821
## F-statistic: 37.66 on 7 and 646 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4637.489 4677.837
```

```
model.5 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 5 , show.summar
y = TRUE)$model
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.587  -5.811  -1.331   4.306  31.606
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  18.0351     0.8413  21.437  < 2e-16 ***
```

```
## x.t         -10.5535     1.7350  -6.083 2.02e-09 ***
## x.1           0.5363     2.5297   0.212 0.832166
## x.2           0.4982     2.5838   0.193 0.847167
## x.3           1.5183     2.5766   0.589 0.555878
## x.4           0.8421     2.5171   0.335 0.738073
## x.5           6.6487     1.7228   3.859 0.000125 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.329 on 648 degrees of freedom
## Multiple R-squared:  0.2873, Adjusted R-squared:  0.2807
## F-statistic: 43.54 on 6 and 648 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4644.622 4680.499
```

```
model.4 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 4 , show.summar
y = TRUE)$model
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.418  -5.743  -1.444   4.398  32.225
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.3727     0.7723  25.086  < 2e-16 ***
## x.t         -10.8482     1.7492  -6.202 9.93e-10 ***
## x.1           0.3550     2.5524   0.139    0.889
## x.2          -0.2807     2.5946  -0.108    0.914
## x.3          -0.4812     2.5406  -0.189    0.850
## x.4           7.9026     1.7380   4.547 6.49e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.411 on 650 degrees of freedom
## Multiple R-squared:  0.2714, Adjusted R-squared:  0.2658
## F-statistic: 48.42 on 5 and 650 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##      AIC      BIC
## 1 4663.6 4695.003
```

```
model.3 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 3 , show.summar
y = TRUE)$model
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.626  -5.831  -1.118   4.390  31.812
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  20.9176     0.7047  29.682  < 2e-16 ***
## x.t         -11.4184     1.7694  -6.453 2.13e-10 ***
## x.1          -0.5656     2.5773  -0.219    0.826
## x.2          -2.4870     2.5708  -0.967    0.334
## x.3           7.8193     1.7571   4.450 1.01e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.533 on 652 degrees of freedom
## Multiple R-squared:  0.2479, Adjusted R-squared:  0.2433
## F-statistic: 53.72 on 4 and 652 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4688.551 4715.478
```

```
model.2 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 2 , show.summar
y = TRUE)$model
```

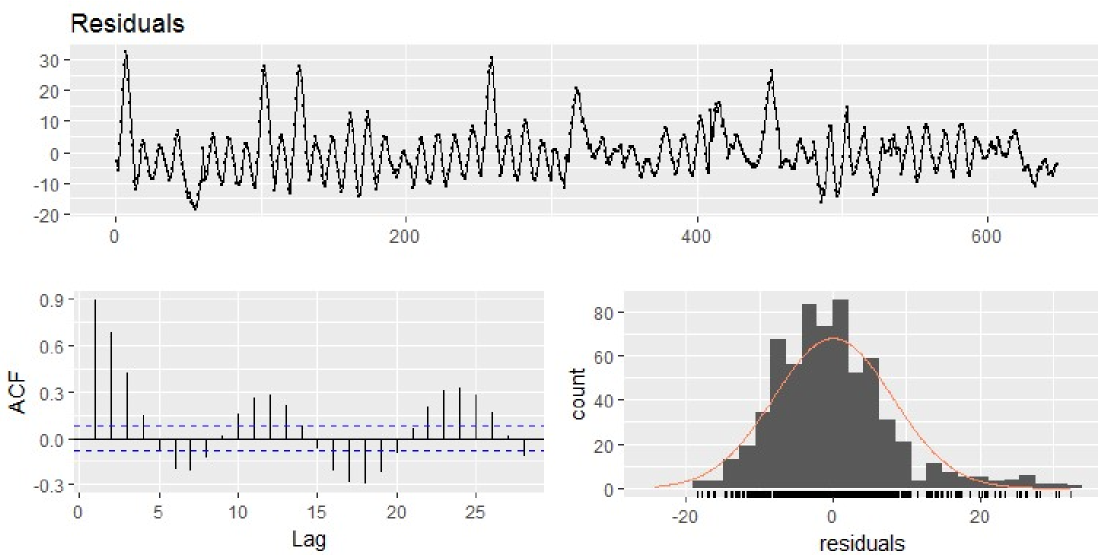
```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.481  -5.773  -0.921   4.576  31.726
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  22.3441     0.6374  35.054  < 2e-16 ***
## x.t         -12.9460     1.7577  -7.366 5.33e-13 ***
## x.1          -2.5903     2.5575  -1.013 0.311517
## x.2           5.8335     1.7499   3.334 0.000906 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.65 on 654 degrees of freedom
## Multiple R-squared:  0.2253, Adjusted R-squared:  0.2217
## F-statistic: 63.39 on 3 and 654 DF,  p-value: < 2.2e-16
##
```

```
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4712.649 4735.095
```

```
model.1 = dlm(x = as.vector(ppt) , y = as.vector(solar) , q = 1 , show.summar
y = TRUE)$model
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.816  -5.736  -0.742   4.717  32.283
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  23.2467     0.5839  39.814  < 2e-16 ***
## x.t         -15.7626     1.5425 -10.219  < 2e-16 ***
## x.1           4.1138     1.5365   2.677  0.00761 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.715 on 656 degrees of freedom
## Multiple R-squared:  0.2128, Adjusted R-squared:  0.2104
## F-statistic: 88.65 on 2 and 656 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4728.713 4746.676
```

```
checkresiduals(model.12)
```



```
##
##  Breusch-Godfrey test for serial correlation of order up to 17
##
## data:  object
## LM test = 594.22, df = 17, p-value < 2.2e-16
```

```
bgtest(model.12$model)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  model.12$model
## LM test = 527.93, df = 1, p-value < 2.2e-16
```

```
#According to this test and ACF plot, we can conclude that the serial correla
tion left in residuals is highly significant. This is true for all 12 simple
DLM models. Hence these models are not a good ft.  Also, from the time series
plot and histogram of residuals, there is an obvious non-random pattern and v
ery high residual values that violate general assumptions.

VIF.model.12 = vif(model.12)
VIF.model.12
```

```
##      x.t      x.1      x.2      x.3      x.4      x.5      x.6      x.7
## 4.432762 7.774629 7.820758 7.914873 7.941510 7.944820 7.943359 7.929999
##      x.8      x.9     x.10     x.11     x.12
## 7.916836 7.921508 7.867385 7.889225 4.508273
```

```
#VIF is less than 10 so there is not much effect of multicollinearity

modeltrans.12 = dlm(x = as.vector(ppt) , y = as.vector(BC.solar) , q = 12 , s
how.summary = TRUE)
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -3.09307 -0.58854  0.03028  0.59333  2.95031
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.256310   0.127145  33.476  < 2e-16 ***
## x.t         -0.666916   0.222424  -2.998 0.002820 **
## x.1          0.005911   0.292417   0.020 0.983878
## x.2         -0.015762   0.291557  -0.054 0.956903
## x.3          0.233509   0.291650   0.801 0.423635
## x.4          0.304722   0.291738   1.045 0.296650
## x.5          0.464108   0.291894   1.590 0.112335
## x.6          0.132652   0.292075   0.454 0.649861
```

```
## x.7          0.063866   0.292054   0.219 0.826972
## x.8         -0.094161   0.291806  -0.323 0.747041
## x.9         -0.136262   0.292053  -0.467 0.640970
## x.10        -0.118130   0.290840  -0.406 0.684756
## x.11         0.036064   0.291106   0.124 0.901446
## x.12        -0.801419   0.219091  -3.658 0.000275 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9328 on 634 degrees of freedom
## Multiple R-squared:  0.3628, Adjusted R-squared:  0.3498
## F-statistic: 27.77 on 13 and 634 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC     BIC
## 1 1764.642 1831.75
```

```
MASE.dynlm(modeltrans.12$model)
```

```
##                           MASE
## modeltrans.12$model 1.417123
```

```
bgtest(modeltrans.12$model)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  modeltrans.12$model
## LM test = 487.17, df = 1, p-value < 2.2e-16
```

```
#The DLM with transformation gives a better MASE value of 1.3355 as well as A
IC and BIC


mase_dlm=MASE.dynlm(model.1,model.2,model.3,model.4,model.5,model.6,model.7,m
odel.8,model.9,model.10,model.11,model.12,modeltrans.12$model)
mase_dlm
```

```
##                       n     MASE
## model.1             659 1.688457
## model.2             658 1.675967
## model.3             657 1.662703
## model.4             656 1.646357
## model.5             655 1.613848
## model.6             654 1.607532
## model.7             653 1.607042
## model.8             652 1.604806
## model.9             651 1.593121
## model.10            650 1.577996
## model.11            649 1.562127
```

```
## model.12            648 1.551600
## modeltrans.12$model 648 1.417123

#*modeltrans.12* with transformation is best in terms of MASE (1.41). However
, the difference is not huge so we will use AIC and BIC as measures of perfor
mances for DLM models. According to this model.12 is the best fit model with
lag length 12. We can still explore models with MASE<1

modeltrans.12.forecasts = dlmForecast(model = modeltrans.12 , x =dataxvals, h
= 2)$forecasts

#Residuals are high suggesting a better model could be found. Hence we move o
n to check for Polynomial Distributed Lag models.


#Polynomial Distributed Lag models


poly.11 = polyDlm(x = as.vector(ppt) , y = as.vector(solar) , q = 1 , k = 1,
show.beta = TRUE , show.summary = TRUE)

##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.816  -5.736  -0.742   4.717  32.283
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  23.2467     0.5839  39.814  < 2e-16 ***
## x.t         -15.7626     1.5425 -10.219  < 2e-16 ***
## x.1           4.1138     1.5365   2.677  0.00761 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.715 on 656 degrees of freedom
## Multiple R-squared:  0.2128, Adjusted R-squared:  0.2104
## F-statistic: 88.65 on 2 and 656 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4728.713 4746.676
##
## Call:
## lm(formula = y.t ~ ., data = z)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -17.816  -5.736  -0.742   4.717  32.283
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  23.2467     0.5839  39.814  < 2e-16 ***
## z.t0        -15.7626     1.5425 -10.219  < 2e-16 ***
## z.t1         19.8764     2.9067   6.838 1.84e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.715 on 656 degrees of freedom
## Multiple R-squared:  0.2128, Adjusted R-squared:  0.2104
## F-statistic: 88.65 on 2 and 656 DF,  p-value: < 2.2e-16
##
## 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
```

```
poly.21 = polyDlm(x = as.vector(ppt) , y = as.vector(solar) , q = 2 , k = 1,
show.beta = TRUE , show.summary = TRUE)
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.481  -5.773  -0.921   4.576  31.726
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  22.3441     0.6374  35.054  < 2e-16 ***
## x.t         -12.9460     1.7577  -7.366 5.33e-13 ***
## x.1          -2.5903     2.5575  -1.013 0.311517
## x.2           5.8335     1.7499   3.334 0.000906 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.65 on 654 degrees of freedom
## Multiple R-squared:  0.2253, Adjusted R-squared:  0.2217
## F-statistic: 63.39 on 3 and 654 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4712.649 4735.095
##
## Call:
## lm(formula = y.t ~ ., data = z)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -17.455  -5.735  -0.953   4.579  31.705 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  22.2978     0.6074   36.71   <2e-16 ***
## z.t0        -12.5903     0.9573  -13.15   <2e-16 ***
## z.t1          9.3889     0.8850   10.61   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 8.644 on 655 degrees of freedom
## Multiple R-squared:  0.2252, Adjusted R-squared:  0.2228 
## F-statistic:  95.2 on 2 and 655 DF,  p-value: < 2.2e-16
## 
## 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
```

```
poly.31 = polyDlm(x = as.vector(ppt) , y = as.vector(solar) , q = 3 , k = 1,
show.beta = TRUE , show.summary = TRUE)
```

```
## 
## Call:
## lm(formula = y.t ~ ., data = design)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -18.626  -5.831  -1.118   4.390  31.812 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  20.9176     0.7047  29.682  < 2e-16 ***
## x.t         -11.4184     1.7694  -6.453 2.13e-10 ***
## x.1          -0.5656     2.5773  -0.219    0.826    
## x.2          -2.4870     2.5708  -0.967    0.334    
## x.3           7.8193     1.7571   4.450 1.01e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 8.533 on 652 degrees of freedom
## Multiple R-squared:  0.2479, Adjusted R-squared:  0.2433 
## F-statistic: 53.72 on 4 and 652 DF,  p-value: < 2.2e-16
## 
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4688.551 4715.478
```

```
##
## Call:
## lm(formula = y.t ~ ., data = z)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.288  -5.841  -1.102   4.170  31.650
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  20.8794     0.6368   32.79   <2e-16 ***
## z.t0         -9.7724     0.6718  -14.55   <2e-16 ***
## z.t1          5.4204     0.4027   13.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.536 on 654 degrees of freedom
## Multiple R-squared:  0.245,  Adjusted R-squared:  0.2427
## F-statistic: 106.1 on 2 and 654 DF,  p-value: < 2.2e-16
##
## 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
```

```
poly.41 = polyDlm(x = as.vector(ppt) , y = as.vector(solar) , q = 4 , k = 1,
show.beta = TRUE , show.summary = TRUE)
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.418  -5.743  -1.444   4.398  32.225
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.3727     0.7723  25.086  < 2e-16 ***
## x.t         -10.8482     1.7492  -6.202 9.93e-10 ***
## x.1           0.3550     2.5524   0.139    0.889
## x.2          -0.2807     2.5946  -0.108    0.914
## x.3          -0.4812     2.5406  -0.189    0.850
## x.4           7.9026     1.7380   4.547 6.49e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.411 on 650 degrees of freedom
```

```
## Multiple R-squared:  0.2714, Adjusted R-squared:  0.2658
## F-statistic: 48.42 on 5 and 650 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##      AIC      BIC
## 1 4663.6 4695.003
##
## Call:
## lm(formula = y.t ~ ., data = z)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.507  -5.732  -1.440   4.357  32.154
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.0675     0.6753   28.23   <2e-16 ***
## z.t0         -7.4375     0.5204  -14.29   <2e-16 ***
## z.t1          3.4490     0.2269   15.20   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.429 on 653 degrees of freedom
## Multiple R-squared:  0.2649, Adjusted R-squared:  0.2627
## F-statistic: 117.7 on 2 and 653 DF,  p-value: < 2.2e-16
##
## 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
```

```
poly.61 = polyDlm(x = as.vector(ppt) , y = as.vector(solar) , q = 6 , k = 1,
show.beta = TRUE , show.summary = TRUE)
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.136  -5.796  -1.202   4.354  31.403
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.4381     0.9244  18.865  < 2e-16 ***
## x.t         -10.3695     1.7430  -5.949 4.42e-09 ***
## x.1           0.4397     2.5294   0.174   0.8621
```

```
## x.2           0.6780     2.5840   0.262   0.7931
## x.3           1.7948     2.5913   0.693   0.4888
## x.4           1.8136     2.5767   0.704   0.4818
## x.5           3.5003     2.5159   1.391   0.1646
## x.6           2.8893     1.7274   1.673   0.0949 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.322 on 646 degrees of freedom
## Multiple R-squared:  0.2898, Adjusted R-squared:  0.2821
## F-statistic: 37.66 on 7 and 646 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4637.489 4677.837
##
## Call:
## lm(formula = y.t ~ ., data = z)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.284  -5.585  -1.406   4.373  32.143
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  15.0341     0.8062   18.65   <2e-16 ***
## z.t0         -3.7988     0.3912   -9.71   <2e-16 ***
## z.t1          1.5463     0.1065   14.52   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.481 on 651 degrees of freedom
## Multiple R-squared:  0.2568, Adjusted R-squared:  0.2545
## F-statistic: 112.5 on 2 and 651 DF,  p-value: < 2.2e-16
##
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
## beta.0   -3.800      0.391   -9.71 6.69e-21
## beta.1   -2.250      0.310   -7.26 1.10e-12
## beta.2   -0.706      0.249   -2.83 4.74e-03
## beta.3    0.840      0.225    3.73 2.05e-04
## beta.4    2.390      0.249    9.59 1.79e-20
## beta.5    3.930      0.309   12.70 3.44e-33
## beta.6    5.480      0.390   14.00 2.89e-39
```

```
poly.81 = polyDlm(x = as.vector(ppt) , y = as.vector(solar) , q = 8 , k = 1,
show.beta = TRUE , show.summary = TRUE)
```

```
##
## Call:
```

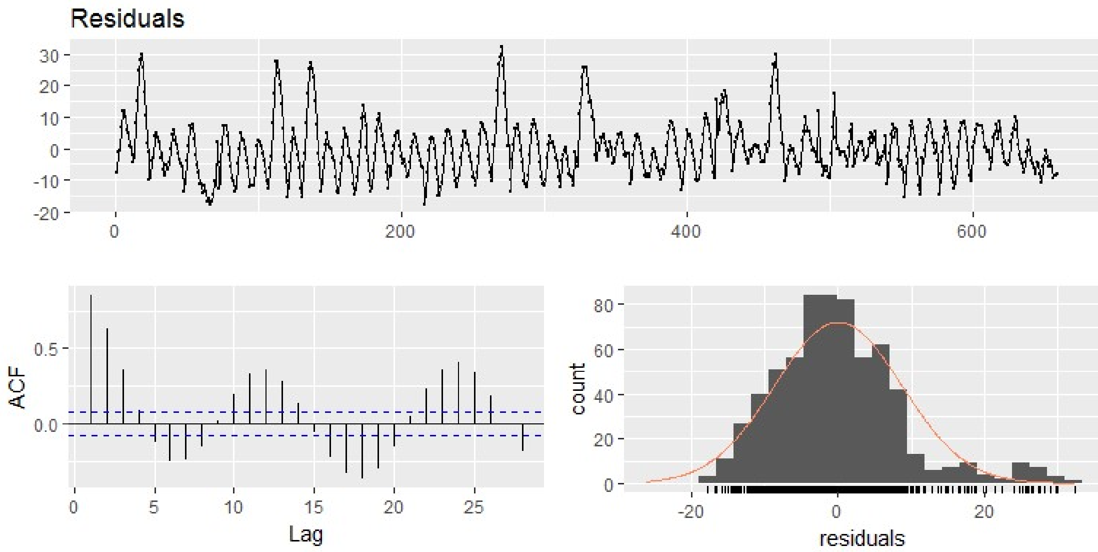
```
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.594  -5.703  -1.197   4.183  31.840
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  17.8827     1.0509  17.016  < 2e-16 ***
## x.t         -10.0720     1.7675  -5.699 1.84e-08 ***
## x.1           0.1511     2.5494   0.059    0.953
## x.2           0.4438     2.5912   0.171    0.864
## x.3           2.0741     2.5977   0.798    0.425
## x.4           1.6863     2.5964   0.649    0.516
## x.5           3.3938     2.5973   1.307    0.192
## x.6           1.3674     2.5825   0.530    0.597
## x.7           3.3675     2.5396   1.326    0.185
## x.8          -2.6471     1.7514  -1.511    0.131
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.327 on 642 degrees of freedom
## Multiple R-squared:  0.2924, Adjusted R-squared:  0.2825
## F-statistic: 29.48 on 9 and 642 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4625.986 4675.267
##
## Call:
## lm(formula = y.t ~ ., data = z)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.161  -6.774  -0.892   5.088  34.619
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12.9009     1.0456  12.338  < 2e-16 ***
## z.t0         -1.3981     0.3694  -3.784 0.000168 ***
## z.t1          0.6388     0.0714   8.948  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.136 on 649 degrees of freedom
## Multiple R-squared:  0.1389, Adjusted R-squared:  0.1362
## F-statistic: 52.32 on 2 and 649 DF,  p-value: < 2.2e-16
##
## Estimates and t-tests for beta coefficients:
##        Estimate Std. Error t value  P(>|t|)
```

```
## beta.0   -1.400      0.369  -3.780 1.68e-04
## beta.1   -0.759      0.317  -2.390 1.70e-02
## beta.2   -0.120      0.274  -0.439 6.61e-01
## beta.3    0.518      0.245   2.120 3.45e-02
## beta.4    1.160      0.234   4.950 9.68e-07
## beta.5    1.800      0.245   7.350 6.24e-13
## beta.6    2.430      0.274   8.890 6.16e-18
## beta.7    3.070      0.317   9.700 7.62e-21
## beta.8    3.710      0.369  10.100 3.20e-22
```

```
poly.121 = polyDlm(x = as.vector(ppt) , y = as.vector(solar) , q = 12 , k = 1
, show.beta = TRUE , show.summary = TRUE)
```

```
##
## Call:
## lm(formula = y.t ~ ., data = design)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.563  -5.239  -0.796   4.137  32.430
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.5164     1.1151  17.501  < 2e-16 ***
## x.t          -5.8876     1.9508  -3.018  0.00265 **
## x.1           0.9993     2.5647   0.390  0.69694
## x.2           0.4343     2.5571   0.170  0.86520
## x.3           1.8763     2.5580   0.734  0.46352
## x.4           1.7459     2.5587   0.682  0.49529
## x.5           3.3279     2.5601   1.300  0.19410
## x.6           0.7751     2.5617   0.303  0.76230
## x.7           1.7937     2.5615   0.700  0.48402
## x.8           0.2827     2.5593   0.110  0.91207
## x.9          -1.1022     2.5615  -0.430  0.66712
## x.10         -1.9333     2.5508  -0.758  0.44880
## x.11         -0.5613     2.5532  -0.220  0.82605
## x.12         -5.3492     1.9216  -2.784  0.00553 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.181 on 634 degrees of freedom
## Multiple R-squared:  0.3216, Adjusted R-squared:  0.3077
## F-statistic: 23.12 on 13 and 634 DF,  p-value: < 2.2e-16
##
## AIC and BIC values for the model:
##        AIC      BIC
## 1 4578.787 4645.895
##
## Call:
## lm(formula = y.t ~ ., data = z)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.089  -7.399  -1.043   5.762  38.362
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 21.42350    1.28430  16.681  < 2e-16 ***
## z.t0         1.26137    0.39535   3.191  0.00149 **
## z.t1        -0.30747    0.05554  -5.536  4.5e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.557 on 645 degrees of freedom
## Multiple R-squared:  0.05822,    Adjusted R-squared:  0.0553
## F-statistic: 19.94 on 2 and 645 DF,  p-value: 3.968e-09
##
## Estimates and t-tests for beta coefficients:
##         Estimate Std. Error t value  P(>|t|)
## beta.0    1.2600      0.395   3.190 1.49e-03
## beta.1    0.9540      0.349   2.730 6.43e-03
## beta.2    0.6460      0.305   2.120 3.47e-02
## beta.3    0.3390      0.267   1.270 2.04e-01
## beta.4    0.0315      0.234   0.134 8.93e-01
## beta.5   -0.2760      0.212  -1.300 1.94e-01
## beta.6   -0.5830      0.203  -2.870 4.21e-03
## beta.7   -0.8910      0.209  -4.260 2.33e-05
## beta.8   -1.2000      0.229  -5.240 2.17e-07
## beta.9   -1.5100      0.259  -5.820 9.53e-09
## beta.10  -1.8100      0.297  -6.110 1.69e-09
## beta.11  -2.1200      0.339  -6.250 7.40e-10
## beta.12  -2.4300      0.385  -6.310 5.38e-10

#q is lag order and k is order of the polynomial

#Now, all the distributed lag weights are significant at 5% level of signific
ance. Also, notice that the standard errors of estimators are much less than
their unconstrained counterparts. This implies that we have more precise esti
mates with polynomial DLM.
vif(poly.11$model)

##     z.t0     z.t1
## 9.224764 9.224764

checkresiduals(poly.11$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  object
## LM test = 568.25, df = 10, p-value < 2.2e-16
```

```
bgtest(poly.11$model)
```

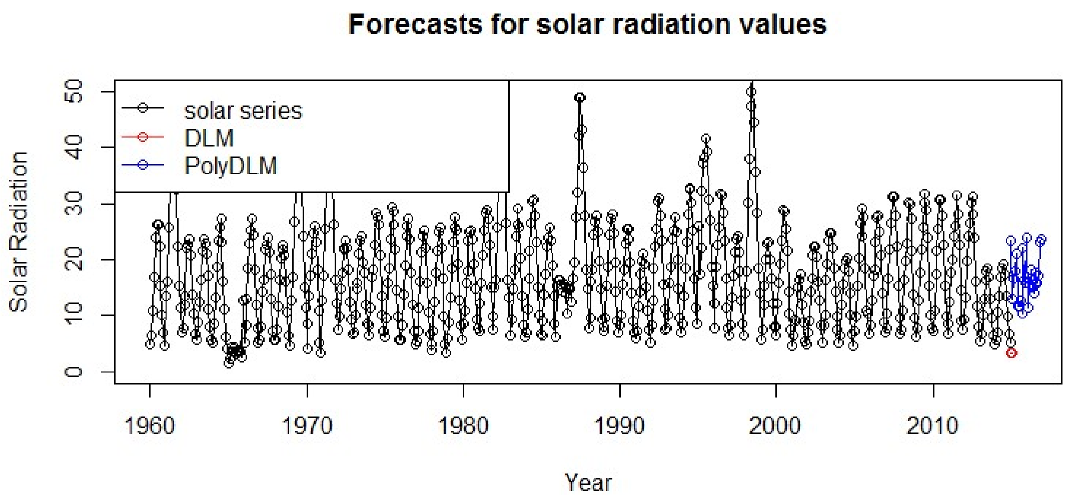
```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  poly.11$model
## LM test = 498.06, df = 1, p-value < 2.2e-16
```

```
mase_poly=MASE.dynlm(poly.11,poly.21,poly.31,poly.41,poly.61,poly.81,poly.121
)
mase_poly
```

```
##            n     MASE
## poly.11  659 1.688457
## poly.21  658 1.676750
## poly.31  657 1.666684
## poly.41  656 1.655786
## poly.61  654 1.646208
## poly.81  652 1.828150
## poly.121 648 1.914090
```

```
#*poly.11 has lowest MASE*

model_poly.forecasts = polyDlmForecast(model = poly.11 , x = dataxvals, h =24
)$forecasts

#*We can explore models with better MASE. Up until now, DLM models were a bet
ter fit that polynomial DLMs*
```

```r
plot(solar, type="o", xlim = c(1960, 2016), ylim = c(0,50),  ylab = "Solar Ra
diation", xlab = "Year", main="Forecasts for solar radiation values")
lines(ts(modeltrans.12.forecasts, start = c(2015,1),frequency=12),col="Red",t
ype="o")
lines(ts(model_poly.forecasts, start = c(2015,1),frequency=12),col="Blue",typ
e="o")
legend("topleft",lty=1, pch = 1, text.width = 20, col=c("black","red","blue")
, c("solar series", "DLM", "PolyDLM"))
```




```r
model_koyck = koyckDlm(x = as.vector(ppt) , y = as.vector(solar) , show.summa
ry = TRUE)

##
## Call:
## ivreg(formula = Y.t ~ Y.t_1 + X.t | Y.t_1 + X.t_1)
##
## 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.t_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
##
```

```
##                  alpha     beta       phi
## (Intercept) -154.0203 5.346844 0.9854613
```

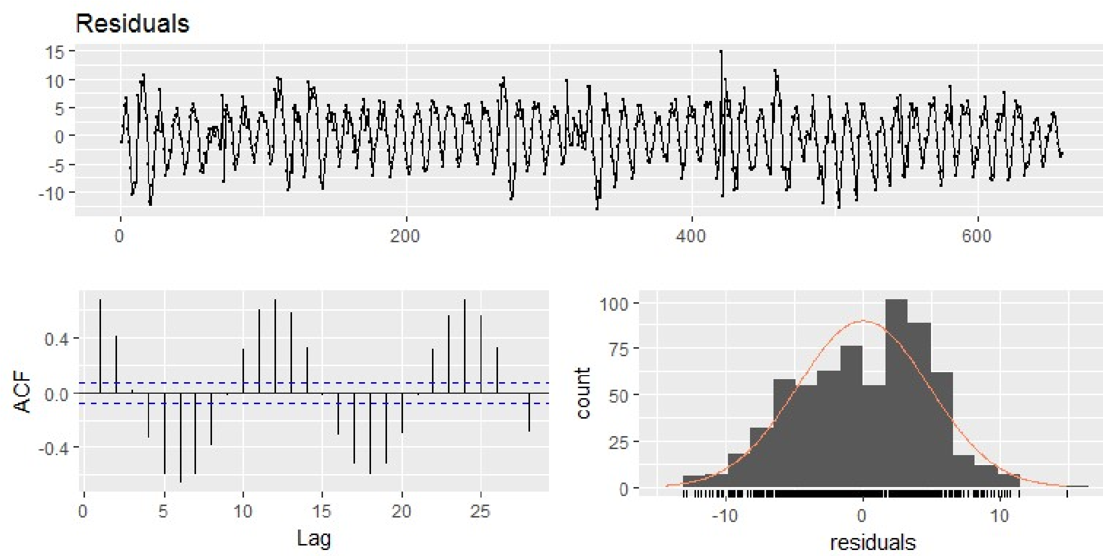
```r
summary(model_koyck$model, diagnostics = TRUE)
```

```
##
## Call:
## ivreg(formula = Y.t ~ Y.t_1 + X.t | Y.t_1 + X.t_1)
##
## 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.t_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
```

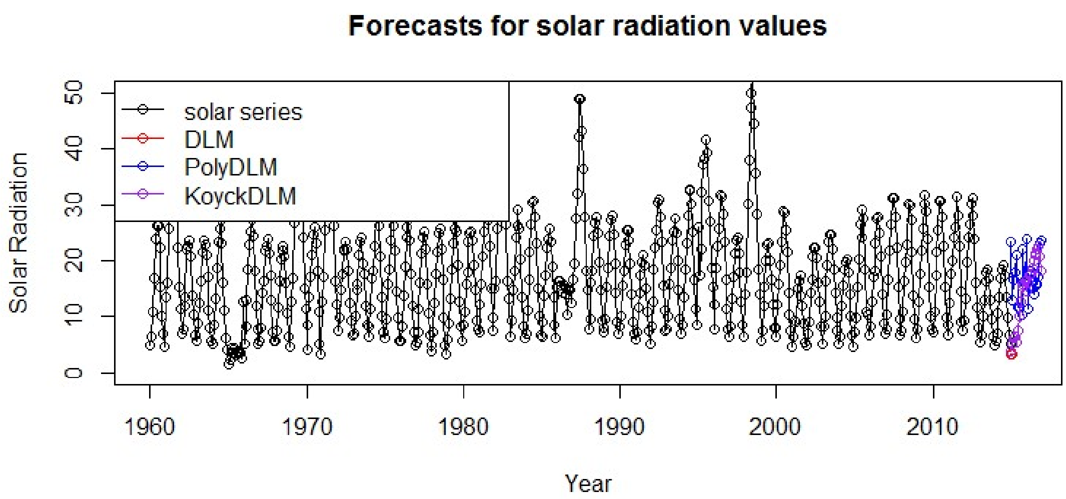
```r
#From the Wu-Hausman test result in the model output, we reject the null hypo
thesis that the correlation between between explanatory variable and the erro
r term is zero (There is no endogenetiy) at 5% level. So, there is a signific
ant correlation between explanatory variable and the error term at 5% level.
vif(model_koyck$model)
```

```
##    Y.t_1      X.t
## 1.605001 1.605001
```

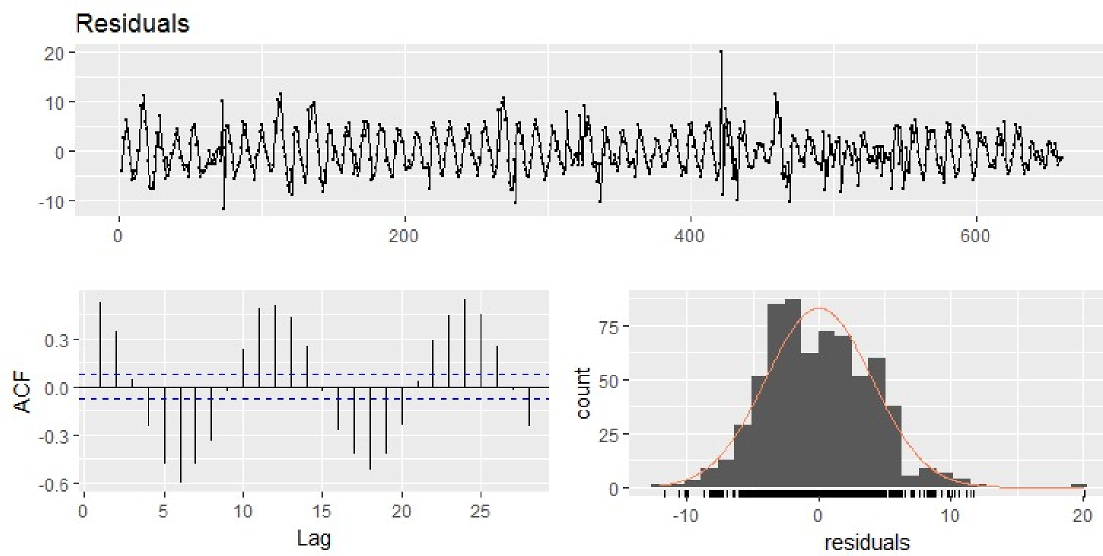
```r
checkresiduals(model_koyck$model)
```




```
bgtest(model_koyck$model)

##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  model_koyck$model
## LM test = 387.66, df = 1, p-value < 2.2e-16

model_koyck.forecasts = koyckDlmForecast(model = model_koyck , x = dataxvals,
h = 24)$forecasts
#high residuals and significant lags present in acf plot makes koyck model un
suitable

plot(solar, type="o", xlim = c(1960, 2016), ylim = c(0,50),  ylab = "Solar Ra
diation", xlab = "Year", main="Forecasts for solar radiation values")
lines(ts(modeltrans.12.forecasts, start = c(2015,1),frequency=12),col="Red",t
ype="o")
lines(ts(model_poly.forecasts, start = c(2015,1),frequency=12),col="Blue",typ
e="o")
lines(ts(model_koyck.forecasts, start = c(2015,1),frequency=12),col="Purple",
type="o")
legend("topleft",lty=1, pch = 1, text.width = 20, col=c("black","red","blue",
"purple"), c("solar series", "DLM", "PolyDLM", "KoyckDLM"))
```




```
#Autoregressive Distributed Lag Model


model_ardl.11 = ardlDlm(x = as.vector(ppt) , y = as.vector(solar) , p = 1 , q
= 1 , show.summary = TRUE)

##
## Time series regression with "ts" data:
## Start = 2, End = 660
##
## Call:
## dynlm(formula = formula(model.text))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.6739  -2.8807  -0.3641   2.8687  20.1193
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.81174    0.53016   1.531    0.126
## X.t         -6.99904    0.73480  -9.525   <2e-16 ***
## L(X.t, 1)    8.67630    0.71609  12.116   <2e-16 ***
## L(Y.t, 1)    0.91001    0.01851  49.161   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.027 on 655 degrees of freedom
## Multiple R-squared:  0.8321, Adjusted R-squared:  0.8314
## F-statistic:  1082 on 3 and 655 DF,  p-value: < 2.2e-16

checkresiduals(model_ardl.11$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  object
## LM test = 377.59, df = 10, p-value < 2.2e-16
```

```
bgtest(model_ardl.11$model)
```

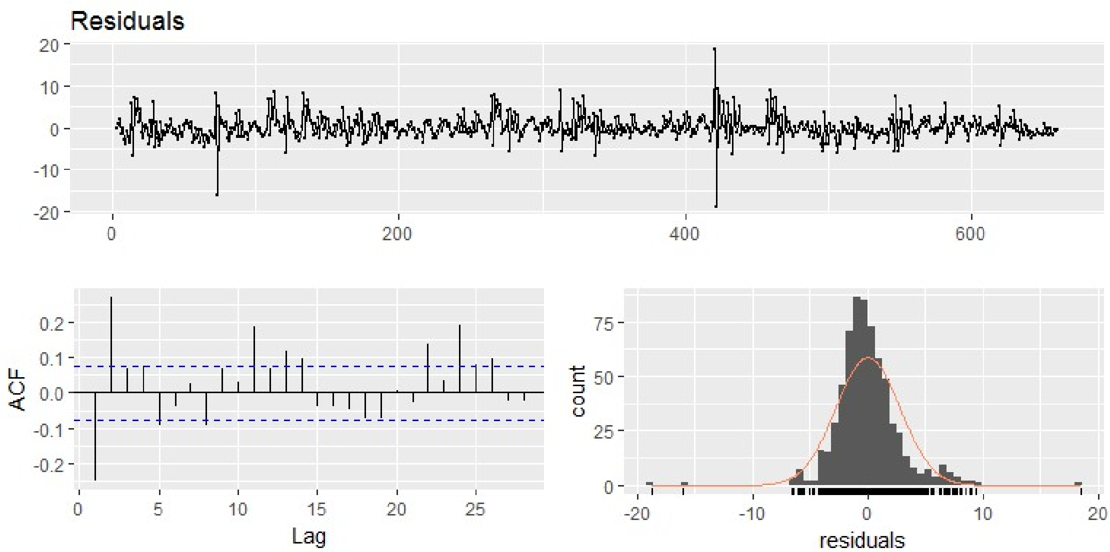
```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  model_ardl.11$model
## LM test = 233.05, df = 1, p-value < 2.2e-16
```

```
model_ardl.11.forecasts = ardlDlmForecast(model = model_ardl.11 ,  x = dataxv
als, h =24)$forecasts

model_ardl.22 = ardlDlm(x = as.vector(ppt) , y = as.vector(solar) , p = 2 , q
= 2 , show.summary = TRUE)
```

```
##
## Time series regression with "ts" data:
## Start = 3, End = 660
##
## Call:
## dynlm(formula = formula(model.text))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -18.7867  -1.5013  -0.2736   1.2345  18.5318
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.08758    0.39200   5.326 1.39e-07 ***
```

```
## X.t          -0.96803    0.59464  -1.628 0.104022
## L(X.t, 1)     0.70618    0.82880   0.852 0.394504
## L(X.t, 2)     2.09832    0.59665   3.517 0.000467 ***
## L(Y.t, 1)     1.51119    0.02823  53.539  < 2e-16 ***
## L(Y.t, 2)    -0.67673    0.02840 -23.829  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.797 on 652 degrees of freedom
## Multiple R-squared:  0.9192, Adjusted R-squared:  0.9186
## F-statistic:  1484 on 5 and 652 DF,  p-value: < 2.2e-16
```

```
checkresiduals(model_ardl.22$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 10
##
## data:  object
## LM test = 161.59, df = 10, p-value < 2.2e-16
```

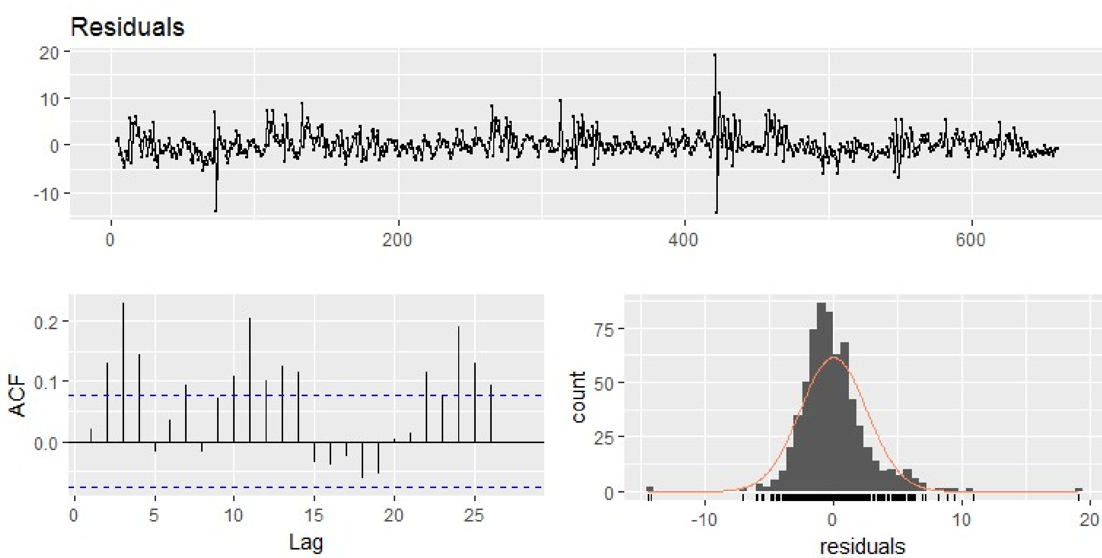
```
bgtest(model_ardl.22$model)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  model_ardl.22$model
## LM test = 79.25, df = 1, p-value < 2.2e-16
```

```
model_ardl.22.forecasts = ardlDlmForecast(model = model_ardl.22 ,  x = dataxv
als, h =24)$forecasts

model_ardl.33 = ardlDlm(x = as.vector(ppt) , y = as.vector(solar) , p = 3 , q
= 3 , show.summary = TRUE)
```

```
##
## Time series regression with "ts" data:
## Start = 4, End = 660
##
## Call:
## dynlm(formula = formula(model.text))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.4265  -1.5232  -0.2725   1.1582  19.0683
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.09000    0.39589   7.805 2.39e-14 ***
## X.t         -0.55917    0.56261  -0.994   0.3206
## L(X.t, 1)    1.00698    0.79376   1.269   0.2050
## L(X.t, 2)    1.84292    0.79195   2.327   0.0203 *
## L(X.t, 3)   -0.26711    0.56697  -0.471   0.6377
## L(Y.t, 1)    1.26560    0.03696  34.244  < 2e-16 ***
## L(Y.t, 2)   -0.13823    0.06139  -2.252   0.0247 *
## L(Y.t, 3)   -0.35408    0.03644  -9.715  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.619 on 649 degrees of freedom
## Multiple R-squared:  0.9295, Adjusted R-squared:  0.9287
## F-statistic:  1222 on 7 and 649 DF,  p-value: < 2.2e-16
```

```
checkresiduals(model_ardl.33$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 11
##
```

```
## data:  object
## LM test = 131.77, df = 11, p-value < 2.2e-16
```

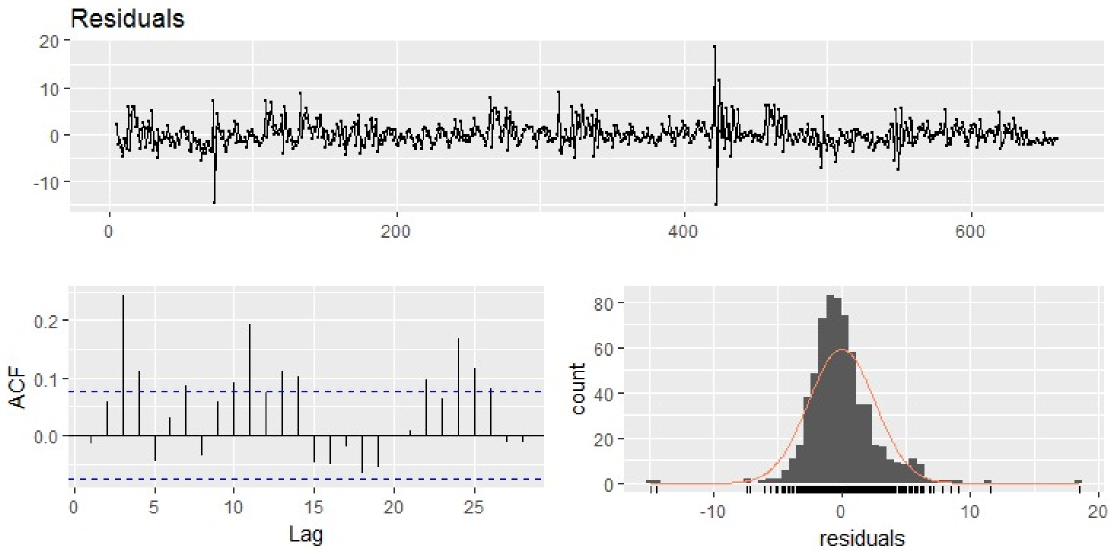
```
bgtest(model_ardl.33$model)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  model_ardl.33$model
## LM test = 2.3861, df = 1, p-value = 0.1224
```

```
model_ardl.33.forecasts = ardlDlmForecast(model = model_ardl.33 ,  x = dataxv
als, h = 24)$forecasts

model_ardl.4 = ardlDlm(x = as.vector(ppt) , y = as.vector(solar) , p = 4 , q
= 4 , show.summary = TRUE)
```

```
##
## Time series regression with "ts" data:
## Start = 5, End = 660
##
## Call:
## dynlm(formula = formula(model.text))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.8794  -1.4604  -0.2545   1.0939  18.5875
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.13899    0.43084   7.286 9.36e-13 ***
## X.t         -0.58279    0.56072  -1.039   0.2990
## L(X.t, 1)    0.83001    0.79438   1.045   0.2965
## L(X.t, 2)    1.43386    0.80749   1.776   0.0763 .
## L(X.t, 3)    1.03081    0.79394   1.298   0.1946
## L(X.t, 4)   -1.30042    0.56524  -2.301   0.0217 *
## L(Y.t, 1)    1.28691    0.03934  32.714  < 2e-16 ***
## L(Y.t, 2)   -0.11558    0.06168  -1.874   0.0614 .
## L(Y.t, 3)   -0.43741    0.06139  -7.125 2.79e-12 ***
## L(Y.t, 4)    0.05271    0.03887   1.356   0.1755
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.608 on 646 degrees of freedom
## Multiple R-squared:  0.9304, Adjusted R-squared:  0.9294
## F-statistic: 958.9 on 9 and 646 DF,  p-value: < 2.2e-16
```

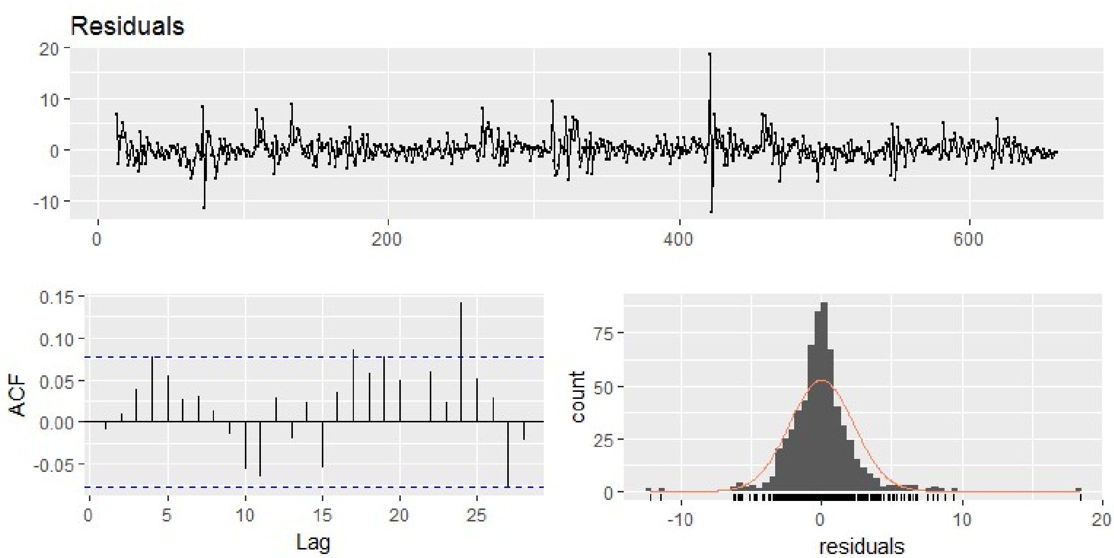
```
checkresiduals(model_ardl.4$model)
```




```
##
##  Breusch-Godfrey test for serial correlation of order up to 13
##
## data:  object
## LM test = 131.3, df = 13, p-value < 2.2e-16

bgtest(model_ardl.4$model)

##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  model_ardl.4$model
## LM test = 13.553, df = 1, p-value = 0.000232

model_ardl.4.forecasts = ardlDlmForecast(model = model_ardl.4 ,  x = dataxval
s, h = 24)$forecasts

#residuals increase for ardl(4,4)

#ardl(3,3) is most suitable

model_ardl.212 = ardlDlm(x = as.vector(ppt) , y = as.vector(solar) , p = 2 ,
q = 12 , show.summary = TRUE)$model

##
## Time series regression with "ts" data:
## Start = 13, End = 660
##
## Call:
## dynlm(formula = formula(model.text))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -12.1868  -1.1351  -0.0724   0.9020  18.4912
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.19845    0.44250   4.968 8.71e-07 ***
## X.t         -0.71587    0.51322  -1.395 0.163542
## L(X.t, 1)    0.90035    0.70426   1.278 0.201565
## L(X.t, 2)    0.79846    0.51301   1.556 0.120106
## L(Y.t, 1)    1.10198    0.03881  28.394  < 2e-16 ***
## L(Y.t, 2)    0.09343    0.05838   1.600 0.110029
## L(Y.t, 3)   -0.20031    0.05841  -3.429 0.000645 ***
## L(Y.t, 4)   -0.13771    0.05861  -2.350 0.019102 *
## L(Y.t, 5)   -0.13849    0.05870  -2.359 0.018611 *
## L(Y.t, 6)    0.09723    0.05902   1.647 0.099969 .
## L(Y.t, 7)    0.05951    0.05938   1.002 0.316646
## L(Y.t, 8)   -0.11452    0.05900  -1.941 0.052695 .
## L(Y.t, 9)    0.12576    0.05866   2.144 0.032412 *
## L(Y.t, 10)   0.08448    0.05836   1.448 0.148239
## L(Y.t, 11)   0.10356    0.05828   1.777 0.076047 .
## L(Y.t, 12)  -0.22359    0.03832  -5.836 8.57e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.311 on 632 degrees of freedom
## Multiple R-squared:  0.9461, Adjusted R-squared:  0.9448
## F-statistic:   739 on 15 and 632 DF,  p-value: < 2.2e-16
```

```
checkresiduals(model_ardl.212)
```



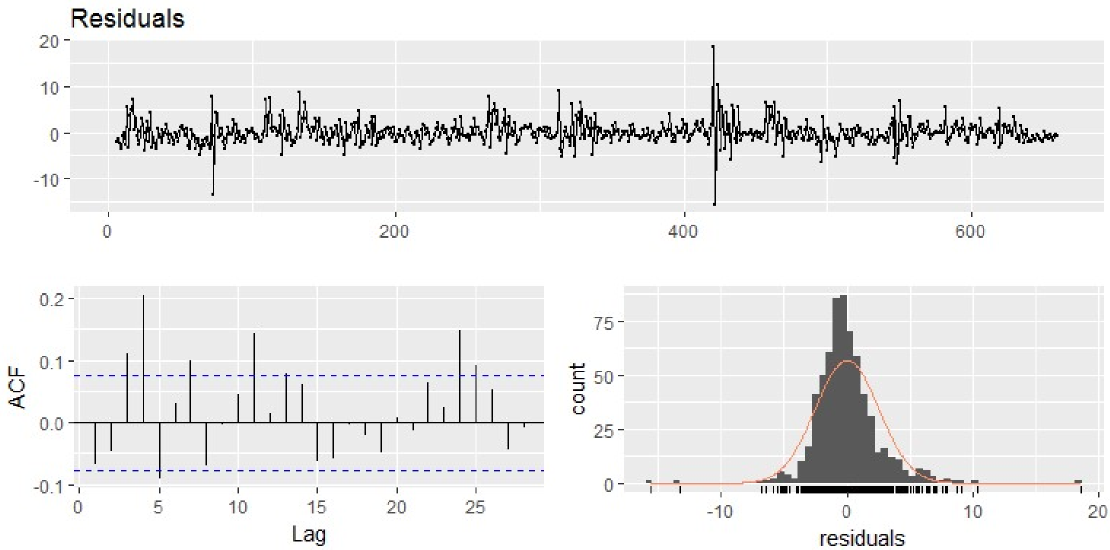

```
##
##  Breusch-Godfrey test for serial correlation of order up to 19
##
```

```
## data:  object
## LM test = 77.01, df = 19, p-value = 6.058e-09
```

```
model_ardl.5 = ardlDlm(x = as.vector(ppt) , y = as.vector(solar) , p = 5 , q
= 5 , show.summary = TRUE)
```

```
##
## Time series regression with "ts" data:
## Start = 6, End = 660
##
## Call:
## dynlm(formula = formula(model.text))
##
## 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
## L(X.t, 1)    0.78299    0.77670   1.008 0.313788
## L(X.t, 2)    1.26543    0.79241   1.597 0.110772
## L(X.t, 3)    0.75184    0.79227   0.949 0.342998
## L(X.t, 4)   -1.00181    0.77678  -1.290 0.197617
## L(X.t, 5)   -0.21024    0.55439  -0.379 0.704639
## L(Y.t, 1)    1.27063    0.03867  32.861  < 2e-16 ***
## L(Y.t, 2)   -0.01727    0.06264  -0.276 0.782907
## L(Y.t, 3)   -0.40297    0.06043  -6.669 5.56e-11 ***
## L(Y.t, 4)   -0.23273    0.06229  -3.737 0.000203 ***
## L(Y.t, 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(model_ardl.5$model)
```




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

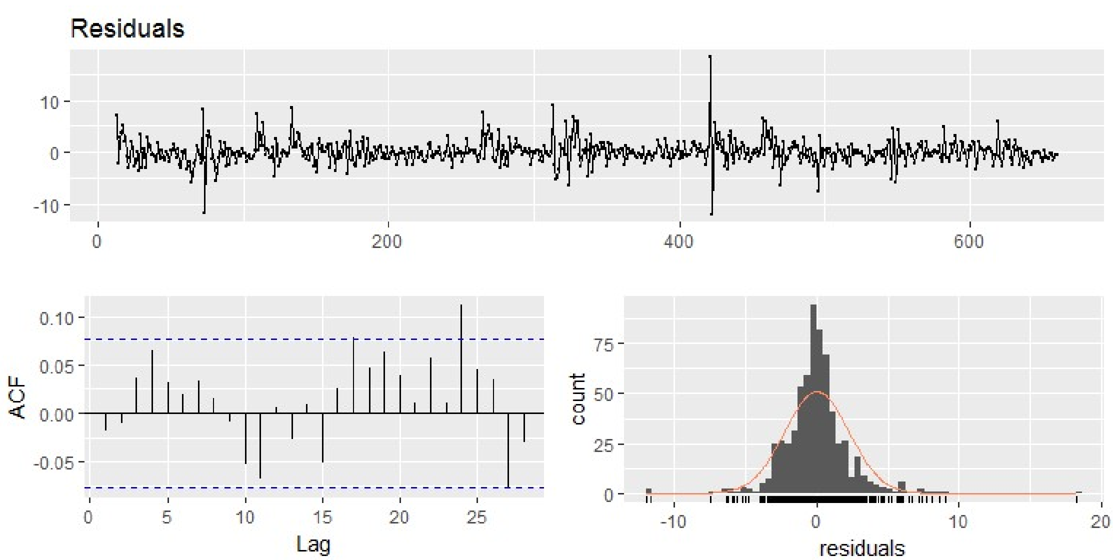
```
bgtest(model_ardl.5$model)
```

```
##
##  Breusch-Godfrey test for serial correlation of order up to 1
##
## data:  model_ardl.5$model
## LM test = 60.323, df = 1, p-value = 8.049e-15
```

```
model_ardl.5.forecasts = ardlDlmForecast(model = model_ardl.5 ,  x = dataxval
s, h = 24)$forecasts

model_ardl.412 = ardlDlm(x = as.vector(ppt) , y = as.vector(solar) , p = 4 ,
q = 12 , show.summary = TRUE)
```

```
##
## Time series regression with "ts" data:
## Start = 13, End = 660
##
## Call:
## dynlm(formula = formula(model.text))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.9724  -1.0789  -0.0976   0.8486  18.2957
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.67282    0.46162   5.790 1.11e-08 ***
```

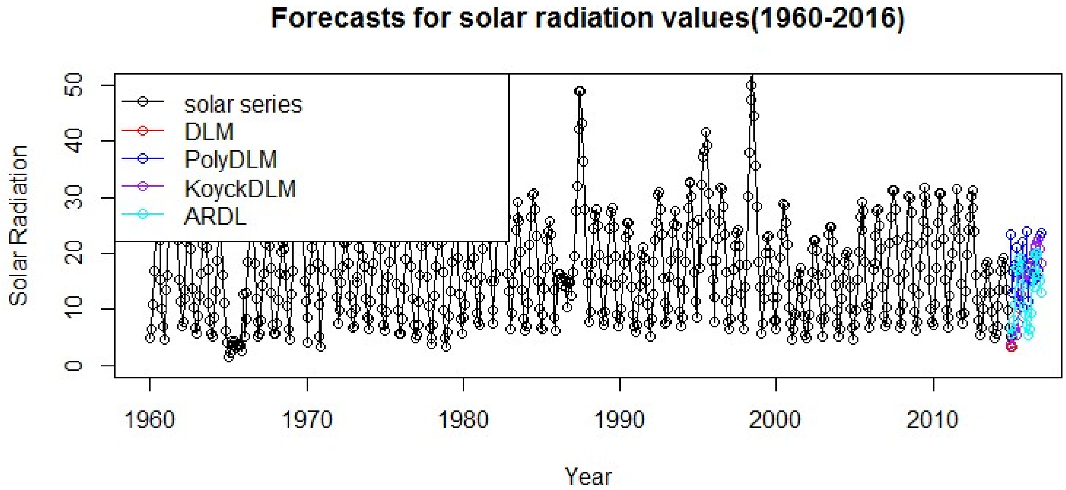
```
## X.t          -0.82870    0.51221  -1.618 0.106186    
## L(X.t, 1)     0.62344    0.71250   0.875 0.381907    
## L(X.t, 2)     0.75238    0.72656   1.036 0.300813    
## L(X.t, 3)     1.31225    0.71371   1.839 0.066440 .  
## L(X.t, 4)    -1.82167    0.51026  -3.570 0.000384 ***
## L(Y.t, 1)     1.09712    0.03858  28.437  < 2e-16 ***
## L(Y.t, 2)     0.11012    0.05803   1.897 0.058227 .  
## L(Y.t, 3)    -0.19360    0.05792  -3.342 0.000880 ***
## L(Y.t, 4)    -0.14384    0.05820  -2.471 0.013719 *  
## L(Y.t, 5)    -0.15506    0.05835  -2.658 0.008070 ** 
## L(Y.t, 6)     0.08815    0.05854   1.506 0.132625    
## L(Y.t, 7)     0.06003    0.05891   1.019 0.308567    
## L(Y.t, 8)    -0.10063    0.05876  -1.712 0.087299 .  
## L(Y.t, 9)     0.13781    0.05865   2.350 0.019096 *  
## L(Y.t, 10)    0.06422    0.05826   1.102 0.270788    
## L(Y.t, 11)    0.11164    0.05795   1.927 0.054488 .  
## L(Y.t, 12)   -0.22660    0.03805  -5.955 4.32e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 2.29 on 630 degrees of freedom
## Multiple R-squared:  0.9472, Adjusted R-squared:  0.9458 
## F-statistic: 664.8 on 17 and 630 DF,  p-value: < 2.2e-16

checkresiduals(model_ardl.412$model)
```




```
## 
##  Breusch-Godfrey test for serial correlation of order up to 21
## 
## data:  object
## LM test = 70.386, df = 21, p-value = 3.047e-07
```

```
model_ardl.412.forecasts = ardlDlmForecast(model = model_ardl.412 , x = datax
vals, h =24)$forecasts

aic.models_ardl = AIC(model_ardl.11$model,model_ardl.22$model,model_ardl.33$m
odel,model_ardl.4$model,model_ardl.5$model, model_ardl.412$model)
sort.score(aic.models_ardl, score="aic")

##                      df      AIC
## model_ardl.412$model 19 2932.294
## model_ardl.5$model   13 3097.877
## model_ardl.4$model   11 3131.424
## model_ardl.33$model   9 3139.409
## model_ardl.22$model   7 3229.051
## model_ardl.11$model   5 3712.311

bic.models_ardl = BIC(model_ardl.11$model,model_ardl.22$model,model_ardl.33$m
odel,model_ardl.4$model,model_ardl.5$model, model_ardl.412$model)
sort.score(bic.models_ardl, score="bic")

##                      df      BIC
## model_ardl.412$model 19 3017.298
## model_ardl.5$model   13 3156.177
## model_ardl.33$model   9 3179.798
## model_ardl.4$model   11 3180.772
## model_ardl.22$model   7 3260.476
## model_ardl.11$model   5 3734.765

mase_ardl=MASE.dynlm(model_ardl.11$model,model_ardl.22$model,model_ardl.33$mo
del,model_ardl.4$model,model_ardl.5$model, model_ardl.412$model)
mase_ardl

##                        n      MASE
## model_ardl.11$model  659 0.8392434
## model_ardl.22$model  658 0.4951319
## model_ardl.33$model  657 0.4737144
## model_ardl.4$model   656 0.4665123
## model_ardl.5$model   655 0.4479311
## model_ardl.412$model 648 0.3857942

#Autoregressive DLM model_ardl.412 has lowest AIC and BIC and MASE value of 0
.38. Residuals are randomly distributed. *So far, model_ardl.412 is most suit
able in tersm of residuals and MASE*

plot(solar, type="o", xlim = c(1960, 2016), ylim = c(0,50),  ylab = "Solar Ra
diation", xlab = "Year", main="Forecasts for solar radiation values(1960-2016
)")
lines(ts(modeltrans.12.forecasts, start = c(2015,1),frequency=12),col="Red",t
ype="o")
lines(ts(model_poly.forecasts, start = c(2015,1),frequency=12),col="Blue",typ
e="o")
lines(ts(model_koyck.forecasts, start = c(2015,1),frequency=12),col="Purple",
```

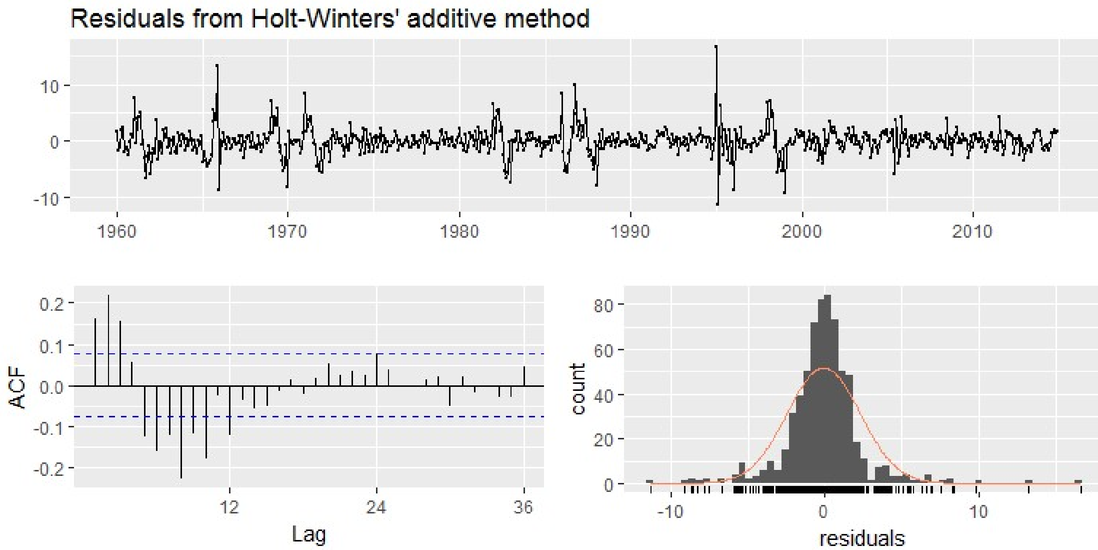
```
type="o")
lines(ts(model_ardl.412.forecasts, start = c(2015,1),frequency=12),col="Cyan"
,type="o")
legend("topleft",lty=1, pch = 1, text.width = 20, col=c("black","red","blue",
"purple","cyan"), c("solar series", "DLM", "PolyDLM", "KoyckDLM", "ARDL"))
```




```
#Exponential smoothing models


#Since seasonality exists in the series, we will consider Holt-Winters' Trend
and Seasonality Method and start off with that.

fit5.hw = hw(solar,seasonal="additive", h=2*frequency(solar))
summary(fit5.hw)

##
## Forecast method: Holt-Winters' additive method
##
## Model Information:
## Holt-Winters' additive method
##
## Call:
##  hw(y = solar, h = 2 * frequency(solar), seasonal = "additive")
##
##   Smoothing parameters:
##     alpha = 0.9968
##     beta  = 0.0079
##     gamma = 0.0027
##
##   Initial states:
##     l = 12.813
##     b = 0.4276
##     s=-10.6349 -7.3748 -2.6593 2.7233 7.775 11.0058
```

```
##            9.8199 6.1144 1.8544 -1.8065 -7.0856 -9.7316
##
##   sigma:  2.3699
##
##      AIC     AICc      BIC
## 5457.817 5458.770 5534.185
##
## Error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.08375221 2.369864 1.547273 -1.615444 12.99165 0.2541887
##                   ACF1
## Training set 0.163735
##
## Forecasts:
##          Point Forecast      Lo 80     Hi 80       Lo 95    Hi 95
## Jan 2015       5.899303   2.862201  8.936406   1.2544557 10.54415
## Feb 2015       8.536199   4.213959 12.858438   1.9259038 15.14649
## Mar 2015      13.828280   8.509665 19.146895   5.6941602 21.96240
## Apr 2015      17.502130  11.334239 23.670021   8.0691544 26.93511
## May 2015      21.822830  14.898340 28.747319  11.2327369 32.41292
## Jun 2015      25.314433  17.698277 32.930589  13.6665276 36.96234
## Jul 2015      26.552786  18.293496 34.812075  13.9212921 39.18428
## Aug 2015      23.394989  14.530464 32.259514   9.8378675 36.95211
## Sep 2015      18.270816   8.831599 27.710033   3.8347798 32.70685
## Oct 2015      12.811417   2.822722 22.800112  -2.4649740 28.08781
## Nov 2015       8.147760  -2.369208 18.664727  -7.9365542 24.23207
## Dec 2015       5.037795  -5.991806 16.067396 -11.8305235 21.90611
## Jan 2016       5.789632  -5.734380 17.313644 -11.8348239 23.41409
## Feb 2016       8.426527  -3.578240 20.431294  -9.9331799 26.78623
## Mar 2016      13.718608   1.245117 26.192099  -5.3579498 32.79517
## Apr 2016      17.392458   4.460922 30.323995  -2.3846202 37.16954
## May 2016      21.713158   8.333114 35.093201   1.2501467 42.17617
## Jun 2016      25.204761  11.384778 39.024744   4.0689210 46.34060
## Jul 2016      26.443114  12.190926 40.695302   4.6462733 48.23995
## Aug 2016      23.285317   8.607936 37.962698   0.8381999 45.73243
## Sep 2016      18.161144   3.064951 33.257337  -4.9264910 41.24878
## Oct 2016      12.701745  -2.807433 28.210923 -11.0174965 36.42099
## Nov 2016       8.038088  -7.878738 23.954914 -16.3045970 32.38077
## Dec 2016       4.928123 -11.393259 21.249506 -20.0332775 29.88952
```

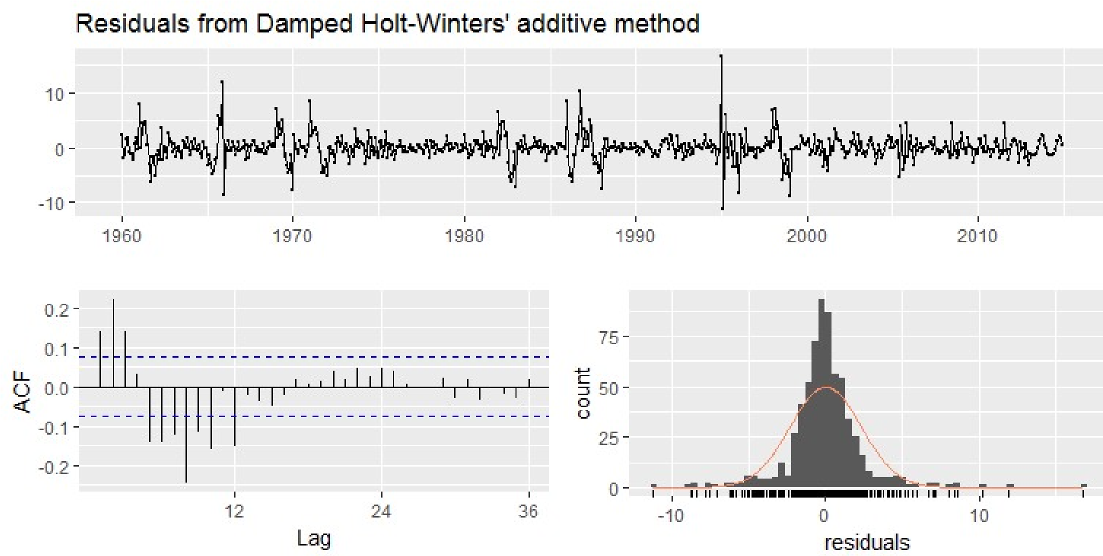
```r
checkresiduals(fit5.hw)
```




```
##
##  Ljung-Box test
##
## data:  residuals
## Q* = 193.75, df = 8, p-value < 2.2e-16
##
## Model df: 16.   Total lags used: 24

#                      ME      RMSE       MAE        MPE      MAPE       MASE
ACF1
#Training set -0.08375221 2.369864 1.547273 -1.615444 12.99165 0.2541887 0.16
3735
#This model of Holt Winters additive trend has the low MASE value and residua
ls


fit6.hw = hw(solar,seasonal="additive",damped = TRUE, h=2*frequency(solar))
summary(fit6.hw)

##
## Forecast method: Damped Holt-Winters' additive method
##
## Model Information:
## Damped Holt-Winters' additive method
##
## Call:
##  hw(y = solar, h = 2 * frequency(solar), seasonal = "additive",
##
##  Call:
##      damped = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.9998
```

```
##     beta  = 0.0305
##     gamma = 1e-04
##     phi   = 0.8002
##
##   Initial states:
##     l = 11.3091
##     b = 1.1812
##     s=-10.2162 -8.1852 -3.0863 2.7434 7.8222 10.7833
##            9.7852 6.9704 2.0583 -2.0649 -7.1175 -9.4927
##
##   sigma:  2.3047
##
##      AIC     AICc      BIC
## 5423.009 5424.076 5503.869
##
## Error measures:
##                         ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.0003458727 2.304693 1.479661 -1.293116 12.22459 0.2430814
##                  ACF1
## Training set 0.139996
##
## Forecasts:
##          Point Forecast      Lo 80     Hi 80      Lo 95    Hi 95
## Jan 2015       5.957157   3.003574  8.910739   1.440042 10.47427
## Feb 2015       8.400404   4.131166 12.669642   1.871168 14.92964
## Mar 2015      13.507280   8.214295 18.800264   5.412359 21.60220
## Apr 2015      17.669457  11.502016 23.836898   8.237170 27.10174
## May 2015      22.618021  15.672023 29.564018  11.995034 33.24101
## Jun 2015      25.460126  17.804536 33.115715  13.751912 37.16834
## Jul 2015      26.479246  18.167125 34.791368  13.766954 39.19154
## Aug 2015      23.536535  14.610566 32.462504   9.885444 37.18763
## Sep 2015      18.470192   8.965802 27.974582   3.934482 33.00590
## Oct 2015      12.651719   2.598992 22.704447  -2.722601 28.02604
## Nov 2015       7.564516  -3.010553 18.139586  -8.608657 23.73769
## Dec 2015       5.537388  -5.537363 16.612140 -11.399982 22.47476
## Jan 2016       6.268556  -5.285594 17.822705 -11.401991 23.93910
## Feb 2016       8.649592  -3.365932 20.665115  -9.726566 27.02575
## Mar 2016      13.706684   1.246017 26.167352  -5.350262 32.76363
## Apr 2016      17.829025   4.937925 30.720125  -1.886212 37.54426
## May 2016      22.745710   9.437586 36.053834   2.392691 43.09873
## Jun 2016      25.562305  11.849441 39.275169   4.590290 46.53432
## Jul 2016      26.561013  12.454711 40.667314   4.987286 48.13474
## Aug 2016      23.601966   9.112668 38.091263   1.442497 45.76143
## Sep 2016      18.522551   3.659937 33.385165  -4.207855 41.25296
## Oct 2016      12.693618  -2.533307 27.920543 -10.593954 35.98119
## Nov 2016       7.598044  -7.984791 23.180879 -16.233845 31.42993
## Dec 2016       5.564218 -10.366745 21.495181 -18.800087 29.92852
```

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



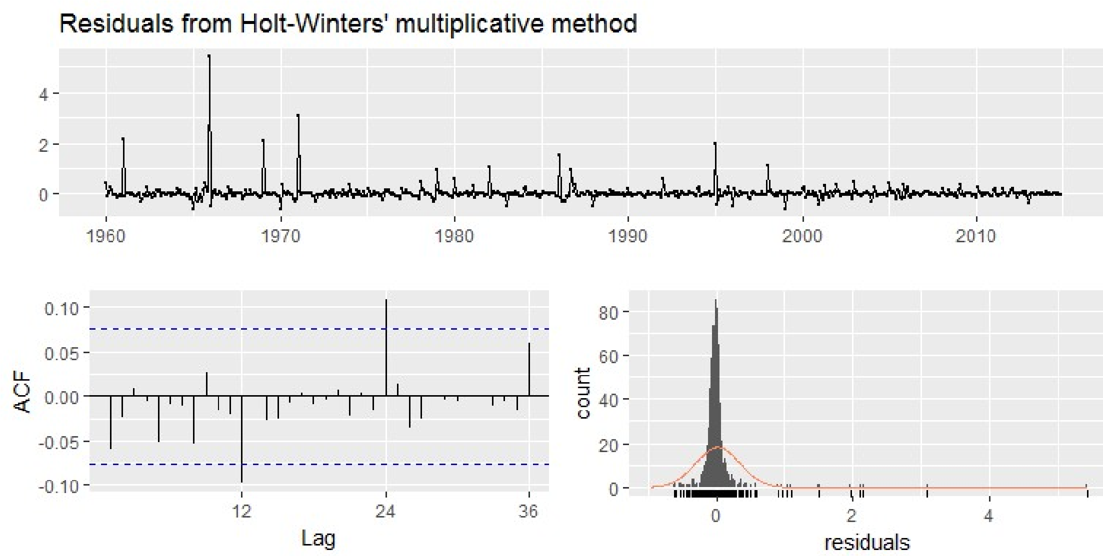

```
##
##  Ljung-Box test
##
## data:  residuals
## Q* = 187.39, df = 7, p-value < 2.2e-16
##
## Model df: 17.   Total lags used: 24
```

```
#                         ME      RMSE       MAE        MPE      MAPE       MASE
ACF1
#Training set -0.0003458727 2.304693 1.479661 -1.293116 12.22459 0.2430814 0.
139996
#Damped hotl winters model has an even lower MASE

fit7.hw = hw(solar,seasonal="multiplicative", h=2*frequency(solar))
summary(fit7.hw)
```

```
##
## Forecast method: Holt-Winters' multiplicative method
##
## Model Information:
## Holt-Winters' multiplicative method
##
## Call:
##  hw(y = solar, h = 2 * frequency(solar), seasonal = "multiplicative")
##
##   Smoothing parameters:
##     alpha = 0.9181
##     beta  = 1e-04
##     gamma = 0.0155
##
##   Initial states:
##     l = 9.0986
```

```
##     b = 0.0427
##     s=0.4397 0.5864 0.8389 1.1545 1.4509 1.62
##            1.5856 1.4029 1.0993 0.8686 0.5587 0.3944
##
##   sigma:  0.3238
##
##      AIC     AICc      BIC
## 6420.503 6421.456 6496.871
##
## Error measures:
##                      ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.1060967 2.062279 1.255284 -2.17078 10.01439 0.2062203
##                     ACF1
## Training set -0.07132262
##
## Forecasts:
##          Point Forecast       Lo 80     Hi 80       Lo 95     Hi 95
## Jan 2015       5.608518   3.2813132  7.935723   2.0493654  9.167671
## Feb 2015       6.942511   2.9455771 10.939445   0.8297282 13.055293
## Mar 2015      10.231515   2.9702619 17.492769  -0.8736134 21.336644
## Apr 2015      12.735791   2.1575755 23.314007  -3.4421936 28.913776
## May 2015      16.245849   0.9089146 31.582784  -7.2099681 39.701667
## Jun 2015      18.528840  -0.9834884 38.041169 -11.3126913 48.370372
## Jul 2015      19.215693  -3.0649048 41.496291 -14.8595407 53.290927
## Aug 2015      17.268548  -4.5683751 39.105472 -16.1281441 50.665241
## Sep 2015      13.834786  -5.1066311 32.776204 -15.1336119 42.803185
## Oct 2015       9.939120  -4.7099408 24.588180 -12.4646849 32.342924
## Nov 2015       6.823089  -3.9531865 17.599364  -9.6578022 23.303980
## Dec 2015       5.368708  -3.6833678 14.420784  -8.4752472 19.212663
## Jan 2016       5.847459  -4.6648771 16.359795 -10.2297716 21.924689
## Feb 2016       7.237310  -6.5718178 21.046437 -13.8819283 28.356547
## Mar 2016      10.664548 -10.8839742 32.213071 -22.2910729 43.620169
## Apr 2016      13.273050 -15.0733145 41.619415 -30.0789736 56.625074
## May 2016      16.928946 -21.2202792 55.078171 -41.4152591 75.273151
## Jun 2016      19.305400 -26.5334907 65.144291 -50.7991337 89.409934
## Jul 2016      20.018432 -30.0003163 70.037181 -56.4786424 96.515507
## Aug 2016      17.987618 -29.2553832 65.230620 -54.2643175 90.239554
## Sep 2016      14.409021 -25.3310708 54.149113 -46.3682047 75.186247
## Oct 2016      10.350337 -19.5996717 40.300345 -35.4542484 56.154922
## Nov 2016       7.104483 -14.4472403 28.656207 -25.8560336 40.065000
## Dec 2016       5.589417 -12.1736547 23.352489 -21.5768568 32.755691
```

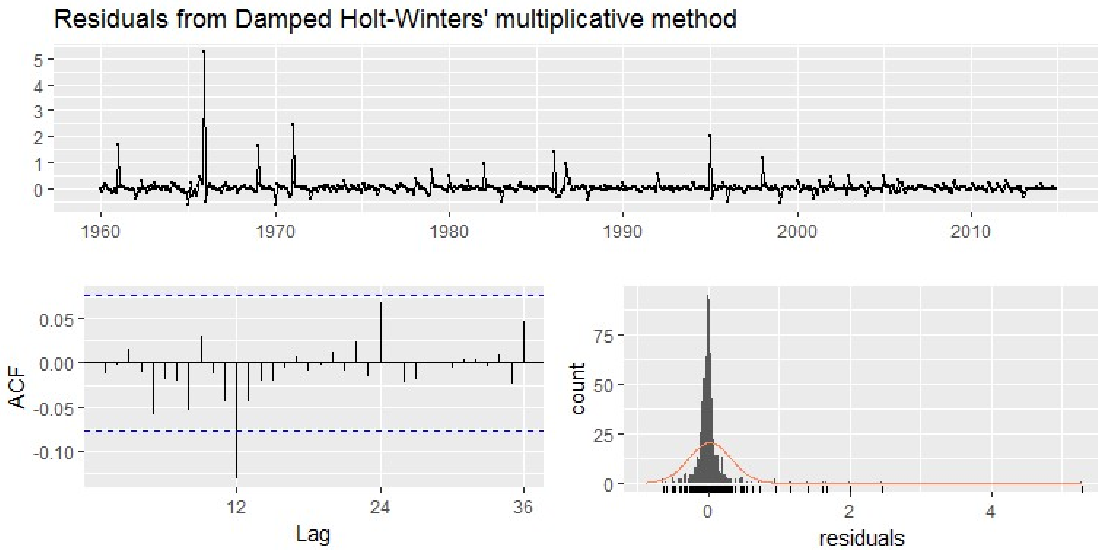
```
checkresiduals(fit7.hw)
```




```
##
##  Ljung-Box test
##
## data:  residuals
## Q* = 23.735, df = 8, p-value = 0.002539
##
## Model df: 16.   Total lags used: 24

#                      ME     RMSE      MAE      MPE     MAPE      MASE
ACF1
#Training set -0.1060967 2.062279 1.255284 -2.17078 10.01439 0.2062203 -0.071
32262


#Next we check the exponential trend model with multiplicative seasonality.

fit8.hw = hw(solar,seasonal="multiplicative",damped = TRUE, h=2*frequency(sol
ar))
summary(fit8.hw)

##
## Forecast method: Damped Holt-Winters' multiplicative method
##
## Model Information:
## Damped Holt-Winters' multiplicative method
##
## Call:
##  hw(y = solar, h = 2 * frequency(solar), seasonal = "multiplicative",
##
##  Call:
##      damped = TRUE)
##
##   Smoothing parameters:
```

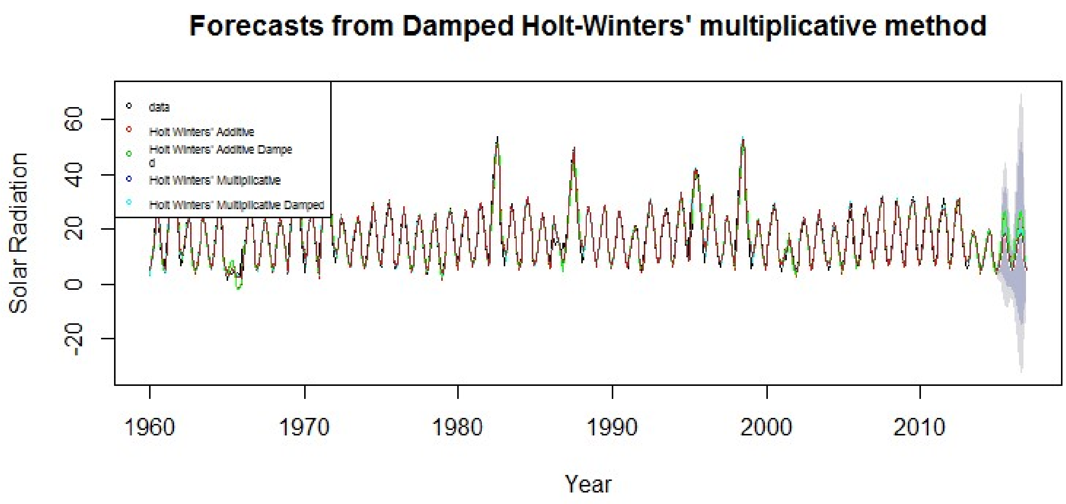
```
##     alpha = 0.7951
##     beta  = 4e-04
##     gamma = 1e-04
##     phi   = 0.8805
##
##   Initial states:
##     l = 9.9766
##     b = 1.3498
##     s=0.4466 0.5613 0.8259 1.1521 1.4418 1.6021
##            1.549 1.393 1.1118 0.8795 0.5884 0.4484
##
##   sigma:  0.3011
##
##      AIC     AICc      BIC
## 6327.138 6328.205 6407.999
##
## Error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.03547783 2.039583 1.240267 -2.200423 10.02395 0.2037532
##                    ACF1
## Training set 0.05899635
##
## Forecasts:
##          Point Forecast        Lo 80     Hi 80       Lo 95     Hi 95
## Jan 2015       5.210042   3.19992817  7.220156   2.1358381  8.284246
## Feb 2015       6.829713   3.40403682 10.255390   1.5905933 12.068833
## Mar 2015      10.207852   4.08059108 16.335112   0.8370152 19.578688
## Apr 2015      12.899256   4.00597381 21.792538  -0.7018454 26.500357
## May 2015      16.163790   3.67436373 28.653216  -2.9371391 35.264719
## Jun 2015      17.974152   2.66137944 33.286925  -5.4447128 41.393017
## Jul 2015      18.588747   1.33021801 35.847275  -7.8058952 44.983388
## Aug 2015      16.729045  -0.05085655 33.508946  -8.9335997 42.391689
## Sep 2015      13.367488  -1.02047080 27.755447  -8.6369962 35.371972
## Oct 2015       9.582696  -1.42585953 20.591251  -7.2534366 26.418828
## Nov 2015       6.511184  -1.43743113 14.459799  -5.6451738 18.667542
## Dec 2015       5.182438  -1.51602944 11.880905  -5.0619837 15.426859
## Jan 2016       5.209234  -1.89803200 12.316499  -5.6603911 16.078858
## Feb 2016       6.828787  -2.98013876 16.637712  -8.1726702 21.830244
## Mar 2016      10.206643  -5.19389071 25.607176 -13.3464407 33.759726
## Apr 2016      12.897924  -7.50542324 33.301271 -18.3063028 44.102150
## May 2016      16.162336 -10.59690033 42.921573 -24.7623846 57.087057
## Jun 2016      17.972747 -13.12429372 49.069788 -29.5860728 65.531567
## Jul 2016      18.587486 -14.97709822 52.152070 -32.7451158 69.920088
## Aug 2016      16.728063 -14.76001179 48.216137 -31.4287916 64.884917
## Sep 2016      13.366811 -12.83345715 39.567079 -26.7030412 53.436663
## Oct 2016       9.582278  -9.95698803 29.121543 -20.3004505 39.465006
## Nov 2016       6.510941  -7.28874491 20.310626 -14.5938572 27.615738
## Dec 2016       5.182272  -6.22524971 16.589794 -12.2640271 22.628572
```

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




```
##
##  Ljung-Box test
##
## data:  residuals
## Q* = 24.591, df = 7, p-value = 0.0008964
##
## Model df: 17.   Total lags used: 24
```

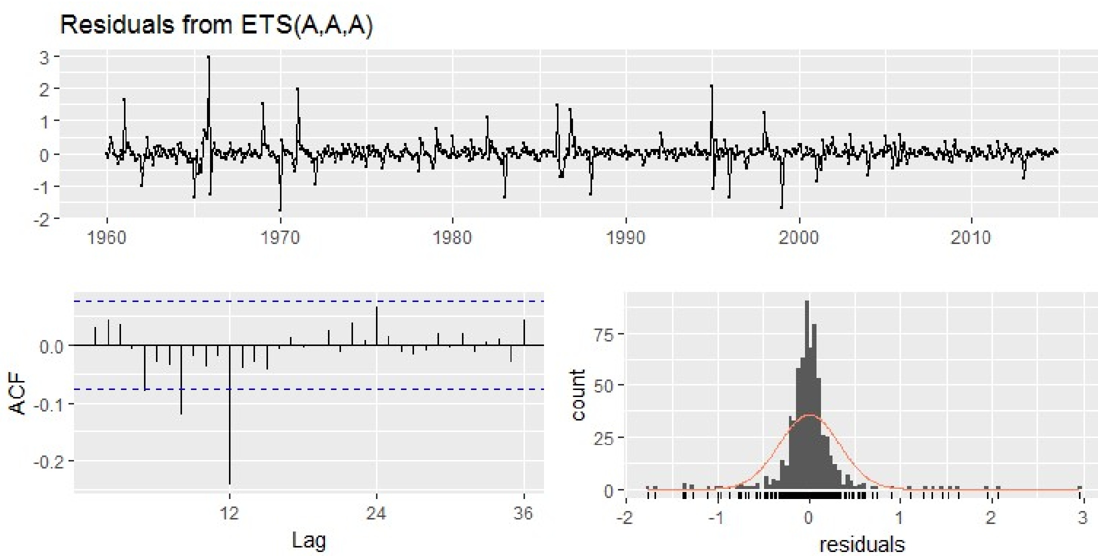
```
plot(fit8.hw,ylab="Solar Radiation",type="l", fcol="white", xlab="Year")
lines(fitted(fit5.hw), col="red", lty=1)
lines(fitted(fit6.hw), col="green", lty=1)
lines(fitted(fit7.hw), col="cyan", lty=1)
lines(fitted(fit8.hw), col="brown", lty=1)
lines(fit5.hw$mean, type="l", col="red")
lines(fit6.hw$mean, type="l", col="green")
lines(fit7.hw$mean, type="l", col="cyan")
lines(fit8.hw$mean, type="l", col="brown")
legend("topleft",lty=0.5, pch=1, col=1:5,cex=0.5,
c("data","Holt Winters' Additive", "Holt Winters' Additive Dampe
d", "Holt Winters' Multiplicative", "Holt Winters' Multiplicative Damped"))
```




```
#                      ME       RMSE       MAE        MPE       MAPE       MASE
ACF1
#Training set -0.03547783 2.039583 1.240267 -2.200423 10.02395 0.2037532 0.05
899635

#From MASE and residual measures, we can notice that the Holts Winters model
with damped trend and multiplicative seasonal model has lowest MASE (0.2)

#Next, we explore the suitability of general State space models.
#Since the solar radiation series exhibits seasonality, we will only evaluate
models that incorporate the seasonal factor

#With additive error, trend and seasonality:

fit3.etsA = ets(solar, model="AAA",lambda = 0.25)
summary(fit3.etsA)

## ETS(A,A,A)
##
## Call:
##  ets(y = solar, model = "AAA", lambda = 0.25)
##
##   Box-Cox transformation: lambda= 0.25
##
##   Smoothing parameters:
##     alpha = 0.8785
##     beta  = 2e-04
##     gamma = 1e-04
##
##   Initial states:
##     l = 3.3172
##     b = -5e-04
##     s=-1.4335 -0.9913 -0.2535 0.4465 0.929 1.185
```

```
##              1.1069 0.8365 0.3412 -0.0641 -0.7965 -1.3064
##
##   sigma:  0.3276
##
##      AIC     AICc      BIC
## 2845.986 2846.939 2922.354
##
## Training set error measures:
##                      ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.03183877 2.036973 1.241284 -1.814031 9.856229 0.2039203
##                    ACF1
## Training set 0.01551632
```

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

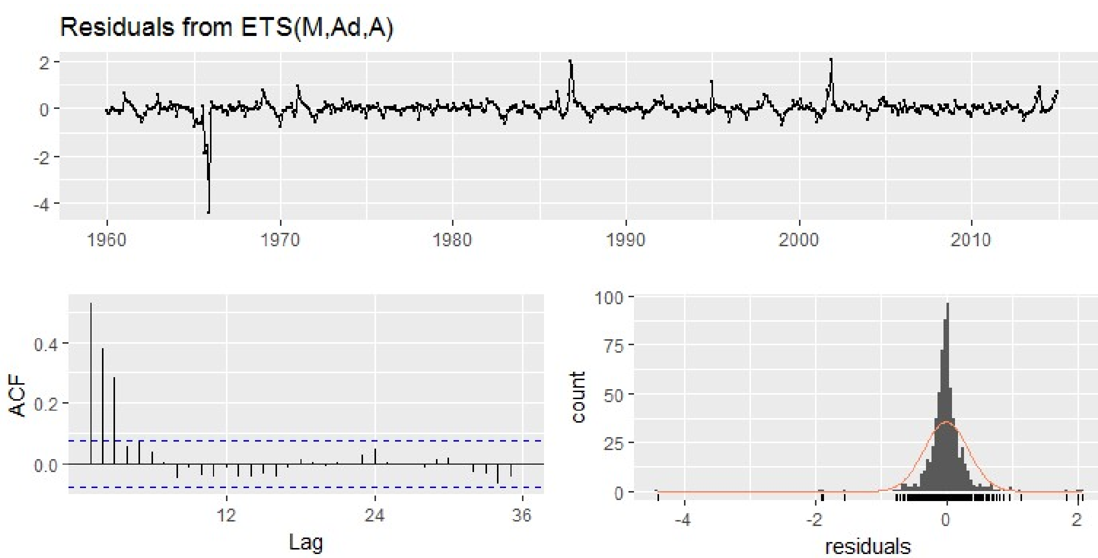



```
##
##  Ljung-Box test
##
## data:  residuals
## Q* = 67.317, df = 8, p-value = 1.677e-11
##
## Model df: 16.   Total lags used: 24
```

```r
#MASE=0.2

#There are many lags in the ACF and residuals are higher than holt winters mu
ltiplicative model. Moreover, MASE is comparable to holt winters multiplicati
ve model.

#With multiplicative error, additive trend and seasonality:

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

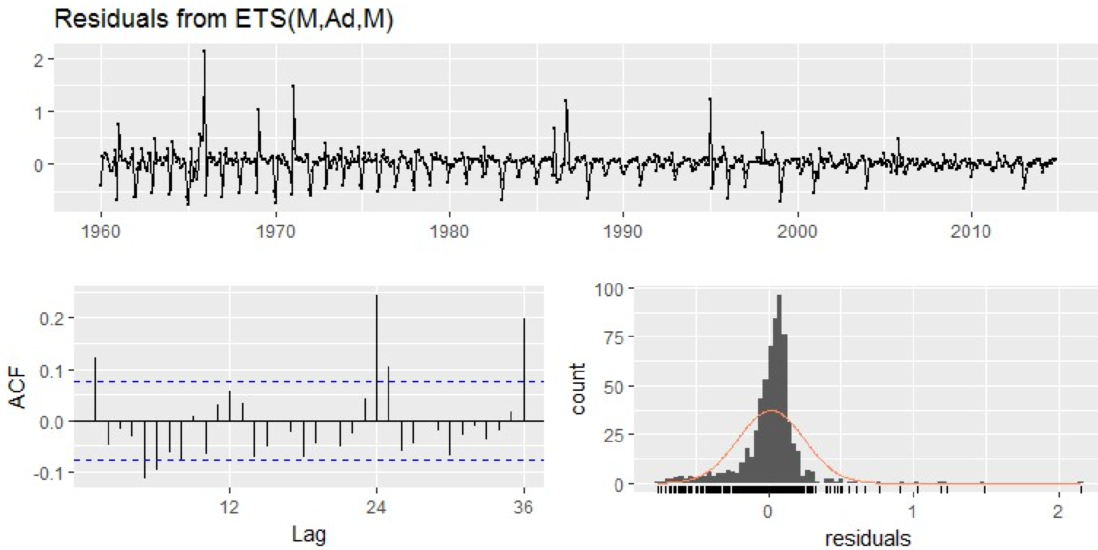
```
## ETS(M,Ad,A)
##
## Call:
##  ets(y = solar, model = "MAA")
##
##   Smoothing parameters:
##     alpha = 0.478
##     beta  = 8e-04
##     gamma = 1e-04
##     phi   = 0.8495
##
##   Initial states:
##     l = 10.7367
##     b = 2.9076
##     s=-10.3436 -7.8261 -3.4126 0.1089 7.7705 10.7246
##            9.8295 7.1223 2.5865 -2.0162 -6.9922 -7.5514
##
##   sigma:  0.335
##
##      AIC     AICc      BIC
## 6492.852 6493.919 6573.712
##
## Training set error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.04002889 3.335056 2.289546 -5.087836 19.54459 0.3761306
##                   ACF1
## Training set 0.6061329
```

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




```
##
##  Ljung-Box test
##
```

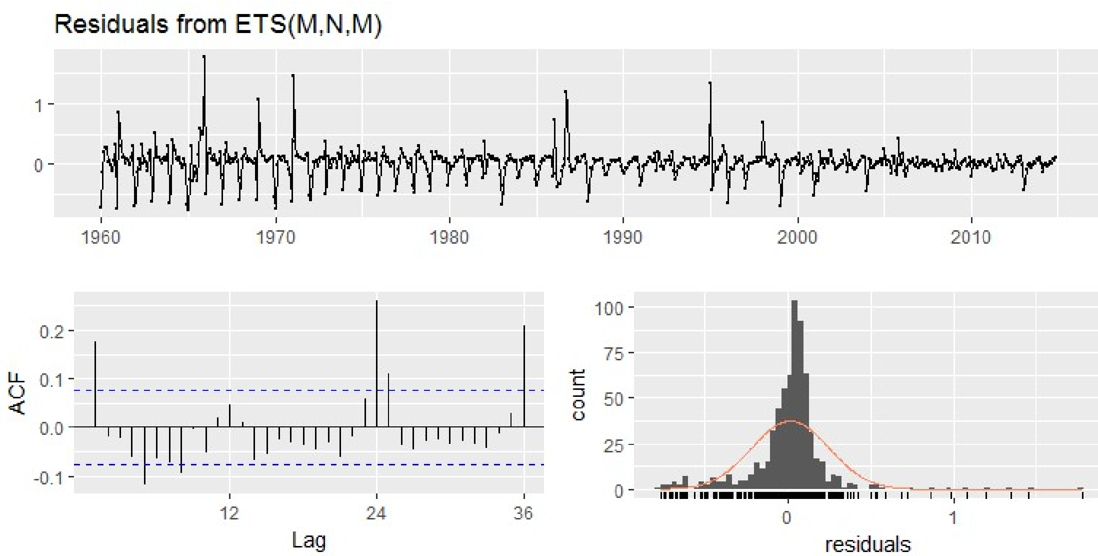
```
## data:  residuals
## Q* = 349.61, df = 7, p-value < 2.2e-16
##
## Model df: 17.   Total lags used: 24

#The above measures show that multiplicative errors perform poorly with this series.

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

## ETS(M,Ad,M)
##
## Call:
##  ets(y = solar, model = "MAM")
##
##   Smoothing parameters:
##     alpha = 0.7842
##     beta  = 1e-04
##     gamma = 0.0661
##     phi   = 0.9613
##
##   Initial states:
##     l = 10.4979
##     b = 0.7605
##     s=0.6918 0.3215 0.6002 1.001 1.3928 1.4728
##            1.4421 1.4614 1.2139 0.9745 0.6685 0.7595
##
##   sigma:  0.2294
##
##      AIC     AICc      BIC
## 5974.796 5975.863 6055.656
##
## Training set error measures:
##                     ME   RMSE      MAE       MPE     MAPE      MASE
## Training set 0.2739231 3.004 1.989601 -4.834858 17.12599 0.3268551
##                   ACF1
## Training set 0.2643485

checkresiduals(fit4.etsM)
```




```
##
##  Ljung-Box test
##
## data:  residuals
## Q* = 95.319, df = 7, p-value < 2.2e-16
##
## Model df: 17.   Total lags used: 24

#                     ME   RMSE      MAE       MPE     MAPE      MASE      ACF1
#Training set 0.2739231 3.004 1.989601 -4.834858 17.12599 0.3268551 0.2643485

#The above measures show that multiplicative errors and multiplicative season
ality perform poorly with this series.


#multiplicative error and seasonality with no trend

fit6.etsM = ets(solar, model="MNM")
summary(fit6.etsM)

## ETS(M,N,M)
##
## Call:
##  ets(y = solar, model = "MNM")
##
##   Smoothing parameters:
##     alpha = 0.7065
##     gamma = 0.0804
##
##   Initial states:
##     l = 21.5335
##     s=0.8906 0.3179 0.6025 0.9817 1.2849 1.4813
##            1.5419 1.375 1.1558 0.9043 0.6746 0.7896
```

```
##
##   sigma:  0.2323
##
##      AIC     AICc      BIC
## 5988.832 5989.577 6056.215
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.2126627 3.195065 2.043712 -5.568916 17.95583 0.3357446
##                   ACF1
## Training set 0.2896291
```

```
checkresiduals(fit6.etsM)
```

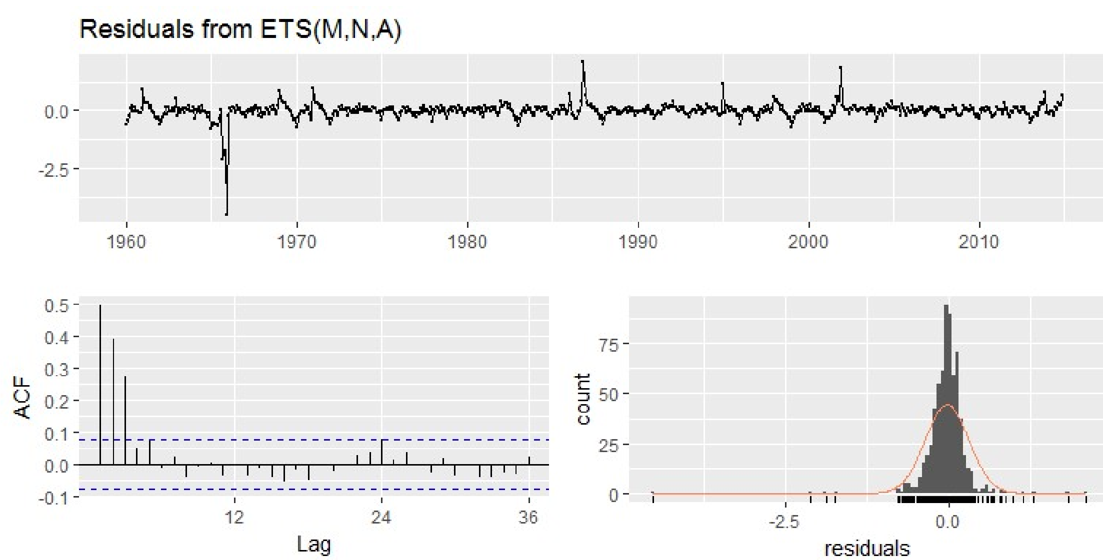



```
##
##  Ljung-Box test
##
## data:  residuals
## Q* = 110.43, df = 10, p-value < 2.2e-16
##
## Model df: 14.   Total lags used: 24
```

```
#This model still does not outperform the Holt Winter model with multiplicati
ve seasonality.

fit7.etsM = ets(solar, model="MNA")
summary(fit7.etsM)
```

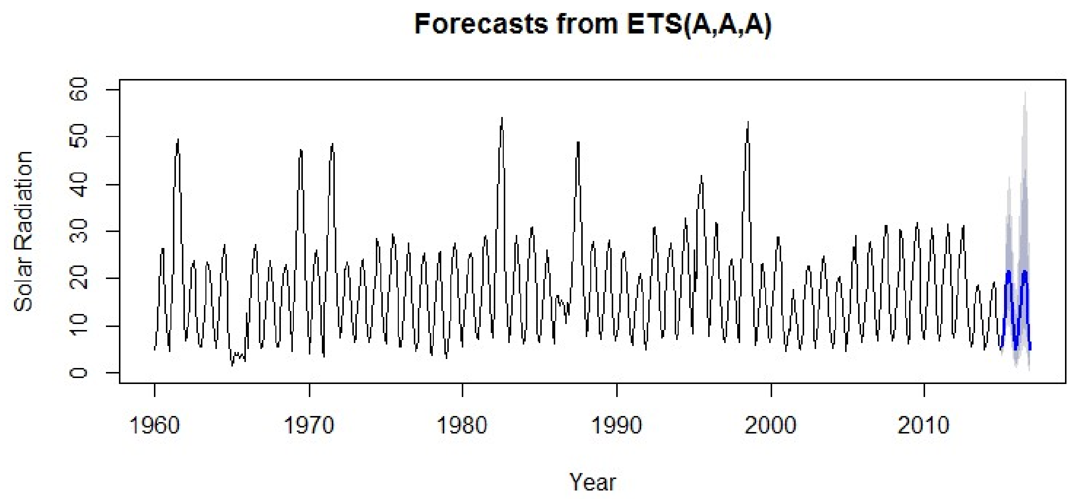
```
## ETS(M,N,A)
##
## Call:
##  ets(y = solar, model = "MNA")
##
##   Smoothing parameters:
```

```
##     alpha = 0.4777
##     gamma = 1e-04
##
##   Initial states:
##     l = 21.5697
##     s=-10.1753 -7.1745 -4.0165 0.0827 7.1147 7.8517
##            12.2277 6.0807 2.1198 -0.5072 -6.0681 -7.5357
##
##   sigma:  0.334
##
##      AIC     AICc      BIC
## 6496.630 6497.376 6564.014
##
## Training set error measures:
##                       ME    RMSE       MAE       MPE      MAPE      MASE
## Training set -0.02316152 3.6531 2.621824 -6.455377 20.94911 0.4307179
##                   ACF1
## Training set 0.4615006

checkresiduals(fit7.etsM)
```




```
##
##  Ljung-Box test
##
## data:  residuals
## Q* = 335.01, df = 10, p-value < 2.2e-16
##
## Model df: 14.   Total lags used: 24

#This model has high MASE and isnot suitable.

#fit3.ETSA model AAA With additive error, trend and seasonality has lowest MA
SE of 0.2
```

```r
plot(forecast(fit3.etsA), ylab="Solar Radiation",type="l", xlab="Year")
```




```r
#State Space Models

#Since the data is seasonal and has heteroscedasticity we use non linear inno
vations state space models and seasonal or multiplicative trend approaches fo
r model fitting



#A Multiplicative Seasonal and Error Model: ETS(M,A,M)

#multiplicative seasonal Holt-Winters' model

fit_mam <- hw(solar,seasonal="multiplicative", h=2*frequency(solar))
summary(fit_mam)

##
## Forecast method: Holt-Winters' multiplicative method
##
## Model Information:
## Holt-Winters' multiplicative method
##
## Call:
##  hw(y = solar, h = 2 * frequency(solar), seasonal = "multiplicative")
##
##   Smoothing parameters:
##     alpha = 0.9181
##     beta  = 1e-04
##     gamma = 0.0155
##
##   Initial states:
```

```
##     l = 9.0986
##     b = 0.0427
##     s=0.4397 0.5864 0.8389 1.1545 1.4509 1.62
##            1.5856 1.4029 1.0993 0.8686 0.5587 0.3944
##
##   sigma:  0.3238
##
##      AIC     AICc      BIC
## 6420.503 6421.456 6496.871
##
## Error measures:
##                      ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.1060967 2.062279 1.255284 -2.17078 10.01439 0.2062203
##                     ACF1
## Training set -0.07132262
##
## Forecasts:
##          Point Forecast       Lo 80     Hi 80       Lo 95     Hi 95
## Jan 2015       5.608518   3.2813132  7.935723   2.0493654  9.167671
## Feb 2015       6.942511   2.9455771 10.939445   0.8297282 13.055293
## Mar 2015      10.231515   2.9702619 17.492769  -0.8736134 21.336644
## Apr 2015      12.735791   2.1575755 23.314007  -3.4421936 28.913776
## May 2015      16.245849   0.9089146 31.582784  -7.2099681 39.701667
## Jun 2015      18.528840  -0.9834884 38.041169 -11.3126913 48.370372
## Jul 2015      19.215693  -3.0649048 41.496291 -14.8595407 53.290927
## Aug 2015      17.268548  -4.5683751 39.105472 -16.1281441 50.665241
## Sep 2015      13.834786  -5.1066311 32.776204 -15.1336119 42.803185
## Oct 2015       9.939120  -4.7099408 24.588180 -12.4646849 32.342924
## Nov 2015       6.823089  -3.9531865 17.599364  -9.6578022 23.303980
## Dec 2015       5.368708  -3.6833678 14.420784  -8.4752472 19.212663
## Jan 2016       5.847459  -4.6648771 16.359795 -10.2297716 21.924689
## Feb 2016       7.237310  -6.5718178 21.046437 -13.8819283 28.356547
## Mar 2016      10.664548 -10.8839742 32.213071 -22.2910729 43.620169
## Apr 2016      13.273050 -15.0733145 41.619415 -30.0789736 56.625074
## May 2016      16.928946 -21.2202792 55.078171 -41.4152591 75.273151
## Jun 2016      19.305400 -26.5334907 65.144291 -50.7991337 89.409934
## Jul 2016      20.018432 -30.0003163 70.037181 -56.4786424 96.515507
## Aug 2016      17.987618 -29.2553832 65.230620 -54.2643175 90.239554
## Sep 2016      14.409021 -25.3310708 54.149113 -46.3682047 75.186247
## Oct 2016      10.350337 -19.5996717 40.300345 -35.4542484 56.154922
## Nov 2016       7.104483 -14.4472403 28.656207 -25.8560336 40.065000
## Dec 2016       5.589417 -12.1736547 23.352489 -21.5768568 32.755691

#Error measures:
#ME      RMSE      MAE      MPE     MAPE      MASE        ACF1
#-0.1060967 2.062279 1.255284 -2.17078 10.01439 0.2062203 -0.07132262

#MAM model:
```

```r
fit.solar.mam = ets(solar, model="MAM")
summary(fit.solar.mam)
```

```
## ETS(M,Ad,M)
##
## Call:
##  ets(y = solar, model = "MAM")
##
##   Smoothing parameters:
##     alpha = 0.7842
##     beta  = 1e-04
##     gamma = 0.0661
##     phi   = 0.9613
##
##   Initial states:
##     l = 10.4979
##     b = 0.7605
##     s=0.6918 0.3215 0.6002 1.001 1.3928 1.4728
##            1.4421 1.4614 1.2139 0.9745 0.6685 0.7595
##
##   sigma:  0.2294
##
##      AIC     AICc      BIC
## 5974.796 5975.863 6055.656
##
## Training set error measures:
##                     ME   RMSE      MAE       MPE     MAPE      MASE
## Training set 0.2739231 3.004 1.989601 -4.834858 17.12599 0.3268551
##                   ACF1
## Training set 0.2643485
```

```r
#Training set error measures:
 #                    ME   RMSE      MAE       MPE     MAPE      MASE      ACF1
#Training set 0.2739231 3.004 1.989601 -4.834858 17.12599 0.3268551 0.2643485

#According to MASE value multiplicative seasonal Holt-Winters' model performs
better for this series.

#Also it is possible to implement a model with multiplicative trend, multipli
cative seasonal component and multiplicative errors. We fit this model to sol
ar radiation series as well.

fit.solar.MMM = ets(solar, model="MMM")
summary(fit.solar.MMM)
```

```
## ETS(M,M,M)
##
## Call:
##  ets(y = solar, model = "MMM")
##
```

```
##   Smoothing parameters:
##     alpha = 0.7228
##     beta  = 8e-04
##     gamma = 0.0867
##
##   Initial states:
##     l = 10.93
##     b = 1.0255
##     s=0.9057 0.3029 0.5813 1.2518 1.3012 1.4324
##            1.583 1.4286 1.0283 0.8702 0.5773 0.7373
##
##   sigma:  0.2269
##
##      AIC     AICc      BIC
## 6001.785 6002.738 6078.153
##
## Training set error measures:
##                      ME     RMSE      MAE       MPE    MAPE      MASE
## Training set -0.4255136 3.176437 1.867805 -8.598519 17.3914 0.3068462
##                   ACF1
## Training set 0.1676206

#multiplicative seasonal Holt-Winters' model and  fit.etsA With additive erro
r, trend and seasonality are the most suitable models with lowest MASE of 0.2
and lowest number of residuals, which are normally distributed. However, fit.
etsA AAA type of model has even lower number of residuals and could be consid
ered a better fit.
```