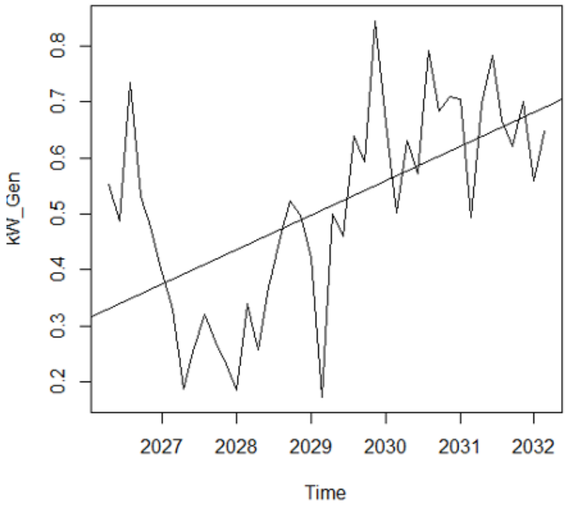
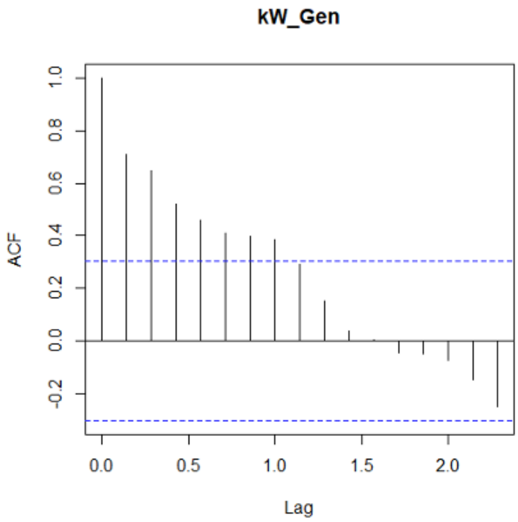
```
#Following packages are essential for Time Series forecasting
install.packages("fUnitRoots")
install.packages("lmtest")
install.packages("FitAR")
install.packages("forecast")
install.packages("haven")
install.packages("lmtest")
install.packages("tseries")

#Read the packages
library(haven)
library(fUnitRoots)
library(lmtest)
library(FitAR)
library(forecast)
library(lmtest)
library(tseries)

#Set the working directory
dir = 'C://OSU 2019-2021//Semester - 3//BAN 5753//Week14'
setwd(dir)
getwd()

#Read SAS data
df = read_sas('solarpv.sas7bdat')

#Understanding data
head(df,15)             #Top 15 records
summary(df)             #Summary of the data
nrow(df)                #Number of rows
ncol(df)                #Number of columns
names(df)               #Names of the columns
class(df$EDT)           #Data Type of EDT
class(df$kW_Gen)        #Data Type of Power Generation
```

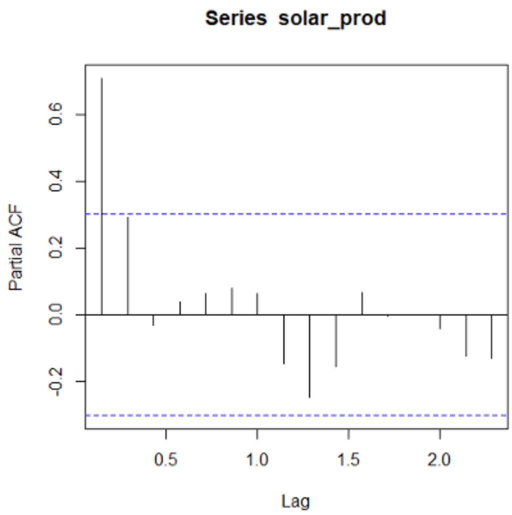
```
> head(df,15)                   #Top 15 records
# A tibble: 15 x 4
     EDT kW_Gen Cloud_Cover cosval
   <dbl>  <dbl>       <dbl>  <dbl>
 1 20001  0.553        4.75 -0.301
 2 20008  0.487        5.34 -0.413
 3 20015  0.734        2.29 -0.519
 4 20022  0.531        4.92 -0.618
 5 20029  0.471        5.52 -0.708
 6 20036  0.394        5.72 -0.788
 7 20043  0.330        5.02 -0.856
 8 20050  0.188        6.57 -0.912
 9 20057  0.262        6.03 -0.954
10 20064  0.320        4.52 -0.983
11 20071  0.273        5.20 -0.998
12 20078  0.232        6.27 -0.998
13 20085  0.185        6.40 -0.983
14 20092  0.339        4.49 -0.954
15 20099  0.258        6.08 -0.912
> summary(df)                   #Summary of the data
      EDT            kW_Gen         Cloud_Cover         cosval
 Min.   :20001   Min.   :0.1730   Min.   :2.286   Min.   :-0.99759
 1st Qu.:20073   1st Qu.:0.3753   1st Qu.:4.591   1st Qu.:-0.78784
 Median :20145   Median :0.5124   Median :5.288   Median :-0.30064
 Mean   :20145   Mean   :0.5111   Mean   :5.190   Mean   :-0.08111
 3rd Qu.:20216   3rd Qu.:0.6592   3rd Qu.:5.821   3rd Qu.: 0.76241
 Max.   :20288   Max.   :0.8446   Max.   :6.571   Max.   : 0.99808
> nrow(df)                      #Number of rows
[1] 42
> ncol(df)                      #Number of columns
[1] 4
> names(df)                     #Names of the columns
[1] "EDT"         "kW_Gen"       "Cloud_Cover" "cosval"
> class(df$EDT)                 #Data Type of EDT
[1] "numeric"
> class(df$kW_Gen)
[1] "numeric"
```

```
#Subsetting the data to include
solar_prod = subset(df,select = c("kW_Gen"))

#Converting EDT to time format
df$EDT = as.Date(df$EDT, origin = "1970-01-01")
print(head(df$EDT,5))      #First recorded date for power generation
print(tail(df$EDT,5))      #Last recorded date for power generation

#Convert solar production to timeseries data with origin as 2025-10-05 with 7 day interval
solar_prod = ts(solar_prod,frequency = 7, start = c(2025,10,05))

#Plot the trend with a mean line
plot(solar_prod)
abline(reg=lm(solar_prod~time(solar_prod)))
```




```
#Calculate ACF for the timeseries
acf(solar_prod)

#Calculate PACF for the timeseries
pacf(solar_prod)
```





#LjungBox Chi Square test to check for White noice in the timeseries
Box.test(solar_prod, type="Ljung-Box")

```
        Box-Ljung test

data:  solar_prod
X-squared = 22.669, df = 1, p-value = 1.925e-06
```

#Create a ARIMA with (p=1, d=0, q=0), to create a different model you can change the value of p,q,d in the list “order = c(1,0,0) in the order of p,q,d
fitARIMA <- arima(solar_prod, order=c(1,0,0),method="ML")
fitARIMA   #Print all detials of the ARIMA Model

```
Call:
arima(x = solar_prod, order = c(1, 0, 0), method = "ML")

Coefficients:
         ar1  intercept
      0.7039     0.5202
s.e.  0.1051     0.0621

sigma^2 estimated as 0.01574:  log likelihood = 27.24,  aic = -48.49
```

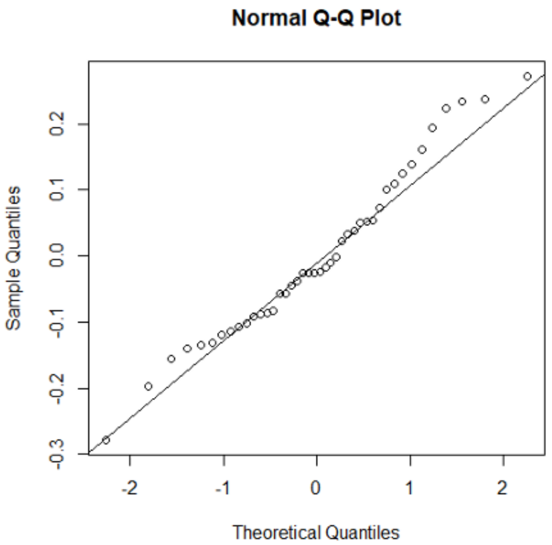
coeftest(fitARIMA)   #Check coefficient of the ARIMA Model

```
z test of coefficients:

          Estimate Std. Error z value  Pr(>|z|)    
ar1       0.703913   0.105141  6.6950 2.157e-11 ***
intercept 0.520190   0.062109  8.3754 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

#Perform aCF, PACF and Ljung Box (White Noise) test - We will use the residuals for this
acf(fitARIMA$residuals)
pacf(fitARIMA$residuals)
Box.test(fitARIMA$residuals, type="Ljung-Box")

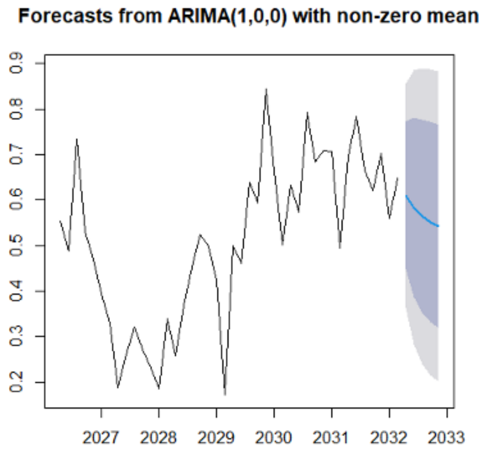
```
        Box-Ljung test

data:  fitARIMA$residuals
X-squared = 1.5826, df = 1, p-value = 0.2084
```

qqnorm(fitARIMA$residuals)
qqline(fitARIMA$residuals)

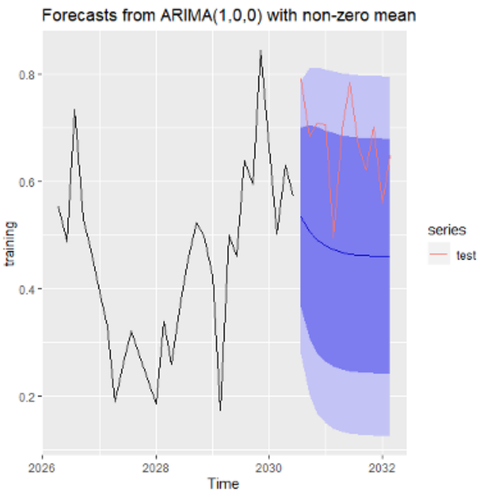


#Diagnostic Checking

```
arima_bic = AIC(fitARIMA , k = log(length(solar_prod)))
print(arima_bic)

#Forecast for the next 5 weeks based on 42 weeks
arima_fore = forecast(fitARIMA, h = 5)
accuracy(arima_fore)
plot(fitARIMA)
plot(arima_fore)
```




```
#Training and Testing for time series using multiple time series
training <- subset(solar_prod, end=length(solar_prod)-12)  #Creating a training data set 1-30
test <- subset(solar_prod, start=length(solar_prod)-11)    #Creating a test data set 31-42

solar_prod_ts <- Arima(training, order=c(1,0,0),method="ML")   #Arima Model with p=1

#Plot train + Test on the graph
solar_prod_ts %>%
  forecast(h=12) %>%
  autoplot() + autolayer(test)
```




```
#Check the accuracy of the model on the test data
solar_prod_ts_test <- Arima(test, model=solar_prod_ts)
accuracy(solar_prod_ts_test)

                     ME      RMSE       MAE      MPE     MAPE     MASE       ACF1
Training set 0.07797087 0.1322323 0.1088544 9.933352 15.92886 1.099678 -0.3671251
#The above output is for the test dataset, R Script labels it as a training set
```