1940233\_EndSem.R

anany

2022-05-10

#Register Number: 1940233  
# Class : 6CMS  
# End Semester exam  
  
#Importing the dataset For the total female births  
library(readxl)  
my\_data = read.csv("daily-total-female-births.csv")  
head(my\_data)

## Date Births  
## 1 1959-01-01 35  
## 2 1959-01-02 32  
## 3 1959-01-03 30  
## 4 1959-01-04 31  
## 5 1959-01-05 44  
## 6 1959-01-06 29

class(my\_data)

## [1] "data.frame"

#importing the required packages  
library(astsa)  
library(tseries)

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

library("xts")

## Loading required package: zoo

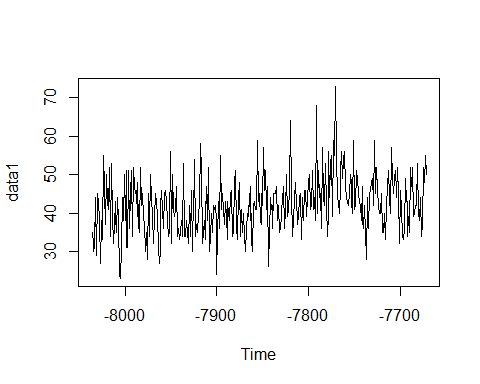
##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

my\_data$Date<-as.Date(my\_data$Date)  
data1<-ts(my\_data$Births,my\_data$Date)  
class(data1)

## [1] "ts"

#Thus , the data is converted into a time series  
ts.plot(data1)

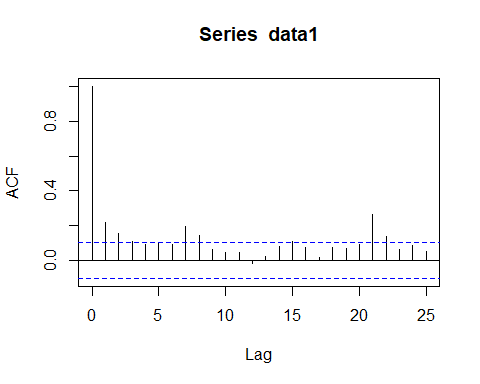


#The plot describes the time series model for the female births. There is   
#no trend. There seems to be a seasonal component in the data though.  
# The seasonal changes are additive in nature since the seasonal variation  
# is relatively constant over time  
  
#Checking for stationality  
acf(data1)  
adf.test(data1)

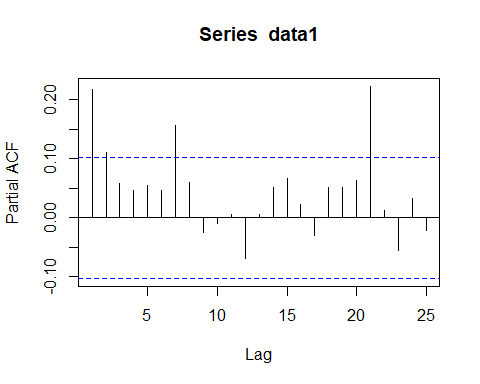
## Warning in adf.test(data1): p-value smaller than printed p-value

##   
## Augmented Dickey-Fuller Test  
##   
## data: data1  
## Dickey-Fuller = -5.1042, Lag order = 7, p-value = 0.01  
## alternative hypothesis: stationary

#Thus we can see that the data is stationary through the acf plot since   
#it gets cut off after lag 1  
  
#the ADF test shows that the data is stationary since the p value is less   
#that 0.05  
# Thus, the data is stationary and there is an absence of the trend component  
  
#Pacf and acf plots  
acf(data1)



pacf(data1)



# The pacf shows that the data does not get cut off after lag 1 with certain   
#outliers in between  
#The acf cuts off after 1 which shows the absence of Autoregression  
#From the acf and pacf, we can see that both tails off to 0 after   
#certain lags, which is an indication of an ARMA(p,q) model.  
#That is both AR and MA part will be included in the  
#model. But we cannot identify the order p,q of ARMA(p,q)   
#using acf and pacf plots  
library(forecast)

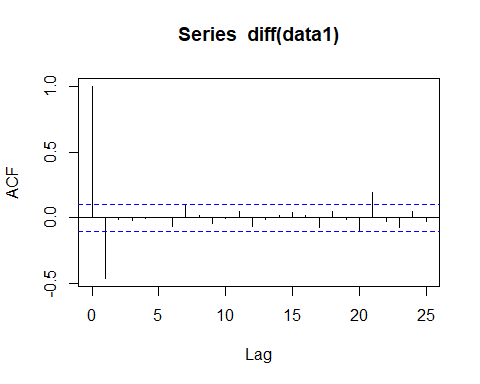
##   
## Attaching package: 'forecast'

## The following object is masked from 'package:astsa':  
##   
## gas

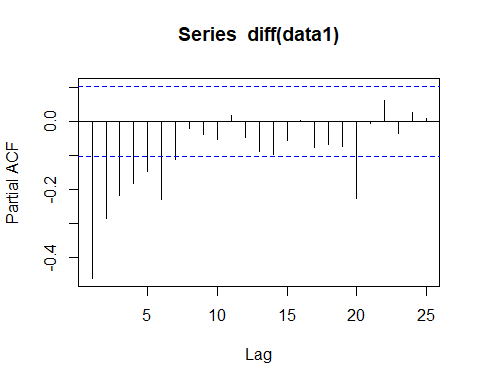
fit=auto.arima(data1,seasonal="TRUE")  
fit

## Series: data1   
## ARIMA(0,1,2)   
##   
## Coefficients:  
## ma1 ma2  
## -0.8478 -0.1079  
## s.e. 0.0497 0.0496  
##   
## sigma^2 = 49.49: log likelihood = -1226.79  
## AIC=2459.57 AICc=2459.64 BIC=2471.26

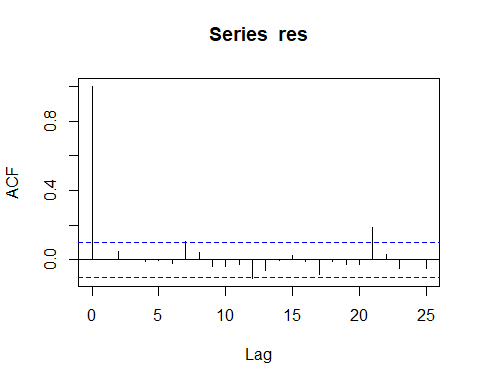
#Thus, the best fitted model would be an MA(2) process or an ARIMA(0,1,2)   
#process  
  
#The model can be written as   
# -0.8487-0.1079  
#The best fitted arima model is ARIMA(0,1,2), which means that if we differenced the data  
#set once, the model will be the simple MA(2) process. You can verify this with respect to the  
#differenced data set.  
#4.  
acf(diff(data1))



pacf(diff(data1))



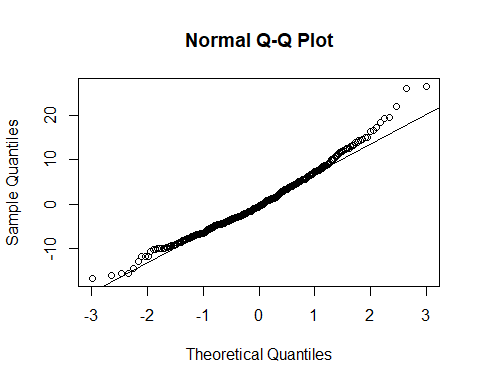
res=resid(fit)  
acf(res)## to check whether the residuals are uncorrelated or not



Box.test(res,lag=10,fitdf=2)

##   
## Box-Pierce test  
##   
## data: res  
## X-squared = 7.0964, df = 8, p-value = 0.5263

# Test for autocorrelation, lag is the number  
# of lags at which to estimate the auto-correlation, fitdf can be taken as the  
# sum of p and q  
  
qqnorm(res)  
qqline(res)



#thus the data may not be normal  
shapiro.test(res)##Normality test

##   
## Shapiro-Wilk normality test  
##   
## data: res  
## W = 0.98029, p-value = 6.849e-05

#since the p value is very small, we reject the null hypothesis. Thus,  
#the data is not normal  
  
 #thus the assumptions are not satisfied since the data is not normally  
#distributed but the data is uncorrelated  
  
#Forecast  
forecast=forecast(data1,h=12)  
forecast

## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## -7671 43.78505 34.72404 52.84607 29.92742 57.64269  
## -7670 43.78505 34.71377 52.85634 29.91172 57.65839  
## -7669 43.78505 34.70351 52.86660 29.89603 57.67408  
## -7668 43.78505 34.69326 52.87685 29.88036 57.68975  
## -7667 43.78505 34.68303 52.88708 29.86471 57.70540  
## -7666 43.78505 34.67281 52.89730 29.84907 57.72104  
## -7665 43.78505 34.66259 52.90752 29.83345 57.73665  
## -7664 43.78505 34.65239 52.91772 29.81785 57.75225  
## -7663 43.78505 34.64220 52.92790 29.80227 57.76784  
## -7662 43.78505 34.63203 52.93808 29.78671 57.78340  
## -7661 43.78505 34.62186 52.94825 29.77116 57.79895  
## -7660 43.78505 34.61171 52.95840 29.75563 57.81448

plot(forecast)

