Time Series Analysis (STAT 560)

Homework 5

Ajoy Kumar Saha, ID 101011922 And Md Mominul Islam, ID: 101009250

October 23, 2020

```
##********Defining Necessary Functions*******************
##Function for measures of accuracy
measacc.fs<- function(y,lambda){
      out<- firstsmooth(y,lambda)
      T<-length(y)
#Smoothed version of the original is the one step ahead prediction
#Hence the predictions (forecasts) are given as
      pred<-c(y[1],out[1:(T-1)])
      prederr<- (y - pred)
      SSE<-sum(prederr*prederr)
      MAPE<-100*sum(abs(prederr/y))/T
      MAD<-sum(abs(prederr))/T
      MSD<-sum(prederr*prederr)/T
      ret1<-c(SSE,MAPE,MAD,MSD)
      names(ret1)<-c("SSE","MAPE","MAD","MSD")
      return(ret1)
}


#Defining Smoothing Function
firstsmooth <- function(y, lambda, start=y[1]) {
           ## here the intial value is set to the first y value
          ytilde <- y
          ytilde[1] <- lambda*y[1]+(1-lambda)*start
          for (i in 2:length(y)) {
              ytilde[i] <- lambda*y[i] + (1-lambda)*ytilde[i-1]
          }
  ytilde
}
## Defingin Trig Function
tlsmooth<-function(y,gamma,y.tilde.start=y[1],lambda.start=1){
      T<-length(y)
      #Initialize the vectors
      Qt<-vector()
      Dt<-vector()
      y.tilde<-vector()
      lambda<-vector()
      err<-vector()
      #Set the starting values for the vectors
```

```
      lambda[1]=lambda.start
      y.tilde[1]=y.tilde.start
      Qt[1]<-0
      Dt[1]<-0
      err[1]<-0
      for (i in 2:T){
          err[i]<-y[i]-y.tilde[i-1]
          Qt[i]<-gamma*err[i]+(1-gamma)*Qt[i-1]
          Dt[i]<-gamma*abs(err[i])+(1-gamma)*Dt[i-1]
          lambda[i]<-abs(Qt[i]/Dt[i])
          y.tilde[i]=lambda[i]*y[i] + (1-lambda[i])*y.tilde[i-1]
      }
return(cbind(y.tilde,lambda,err,Qt,Dt))
}

# load data Table E4.4 for Exercise 4.8
library(readxl)
linear_data <- read_excel("Exercise4_8.xlsx")
linear_data1 <- as.matrix(linear_data)
```

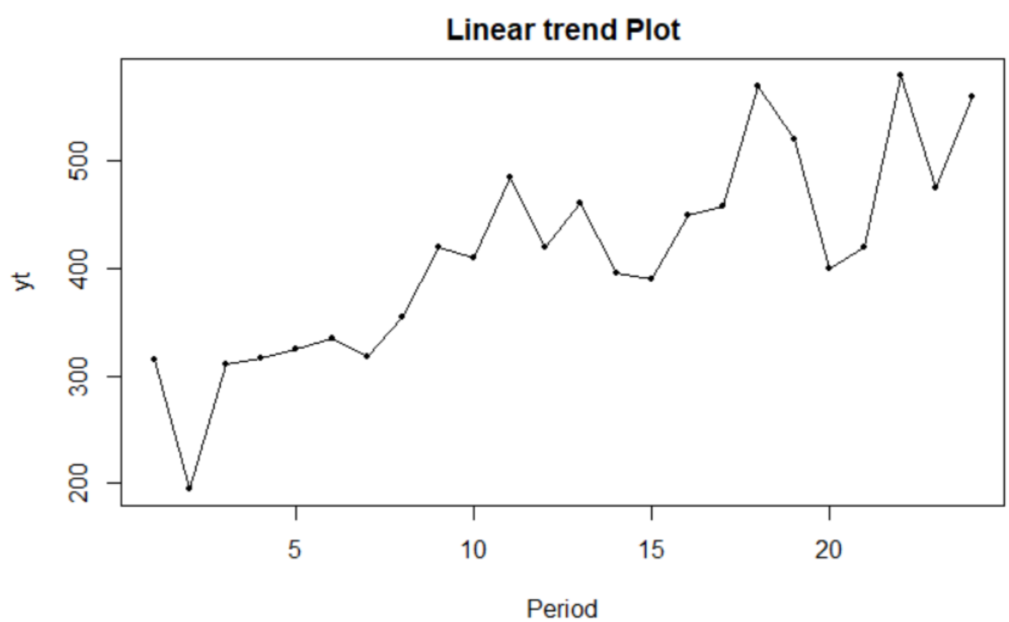
Exercise 4.8:

The data in Table E4.4 exhibit a linear trend.

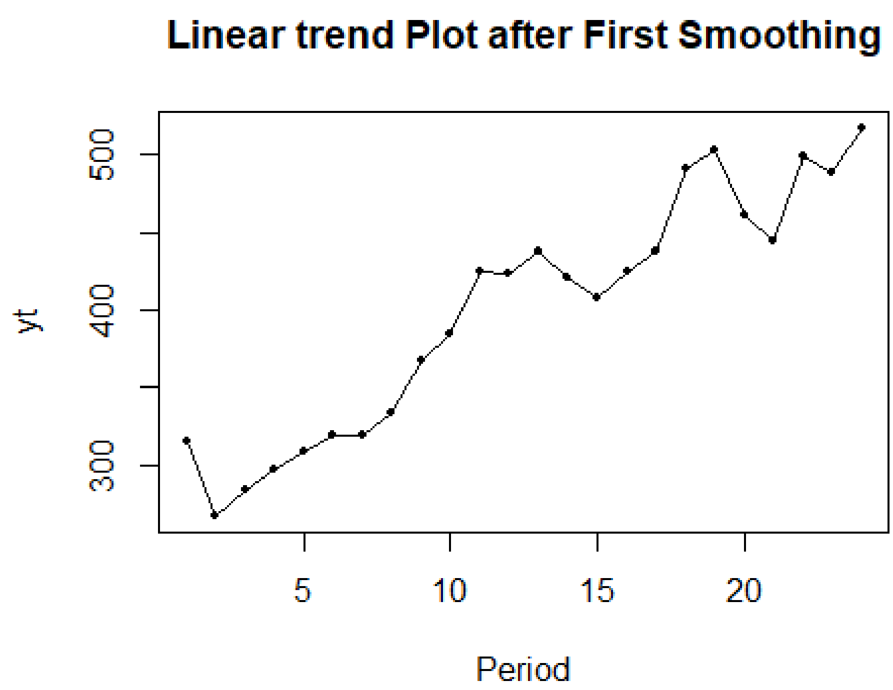
a. Verify that there is a trend by plotting the data

Answer:

```
plot(linear_data1, type ='p', pch=16, cex=.5, xlab='Period',ylab='yt', main =
"Linear trend Plot")
lines(linear_data1)
```



```
linear_data1_fs<-firstsmooth(y=linear_data1[,2],lambda=0.4)
plot(linear_data1_fs, type ='p', pch=16, cex=.5, xlab='Period',ylab='yt', mai
n = "Linear trend Plot after First Smoothing")
lines(linear_data1_fs)
```

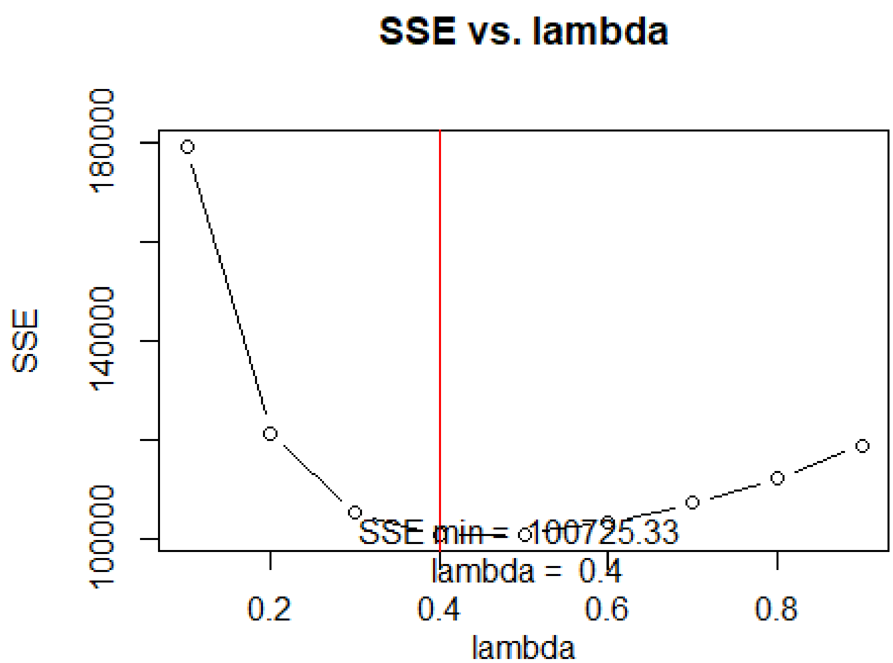


Explanation: From the above time series plot it seems that there is an increasing trend of yt over period. Therefore, yes, there is a trend. From the first order smooting plot, it also clearly shows that there is a liner incresing trend in this time series data.

b. Using the first 12 observations, develop an appropriate procedure for forecasting

Answer:

```
#Finding Out suitable Lamda Value for Low SSE for the first 12 Obeservations
lambda.vec<-seq(0.1, 0.9, 0.1)
sse.mydata<-function(sc){measacc.fs(linear_data1[,2],sc)[1]}
sse.vec<-sapply(lambda.vec, sse.mydata)
opt.lambda<-lambda.vec[sse.vec == min(sse.vec)]
plot(lambda.vec, sse.vec, type="b", main = "SSE vs. lambda\n",
xlab='lambda\n',ylab='SSE')
abline(v=opt.lambda, col = 'red')
mtext(text = paste("SSE min = ", round(min(sse.vec),2), "\n lambda = ", opt.l
ambda), side =1)
```



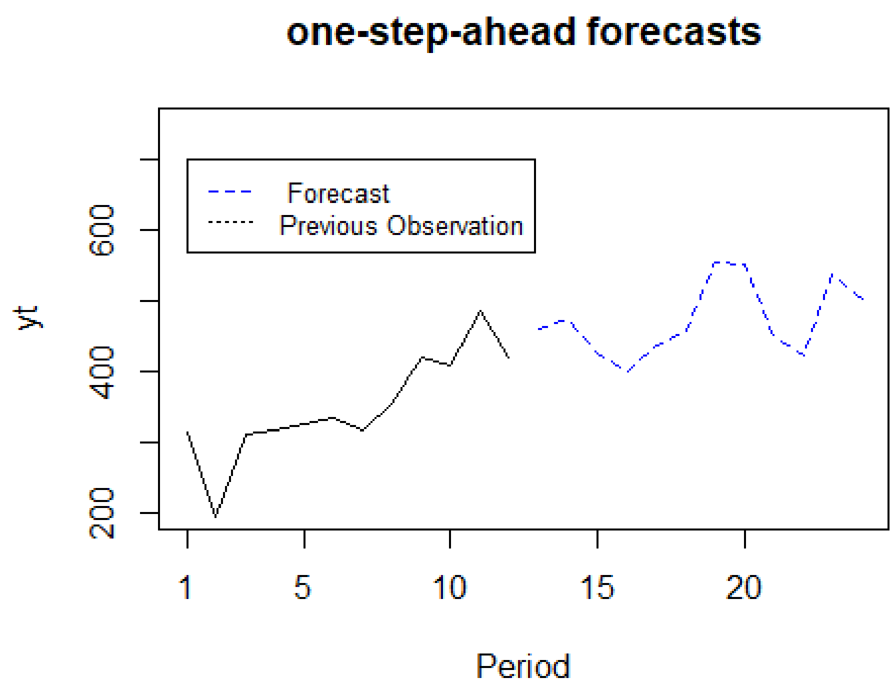
To find the optimum lamda, we find the minimum value of the sum of square error (SSE) for diffent lamda value.The above plot depicted the variation of SSE value for difent lamda value. The lamda value of 0.4 gives the minimum SSE of 100725.33. Therefore, the we selecte the optimum lamda = 0.4 for developing our forcasting model.

c. Forecast the last 12 observations and calculate the forecast errors. Does the forecasting procedure seem to be working satisfactorily?

Answer:

```
lcpi<-0.4
T<-12
tau<-12
alpha.lev<-.05
yt.forecast<-rep(0,tau)
cl<-rep(0,tau)
cpi.smooth1<-rep(0,T+tau)
cpi.smooth2<-rep(0,T+tau)
for (i in 1:tau) {
cpi.smooth1[1:(T+i-1)]<-firstsmooth(y=linear_data1[1:(T+i-1),2],
lambda=lcpi)
cpi.smooth2[1:(T+i-1)]<-firstsmooth(y=cpi.smooth1[1:(T+i-1)],
lambda=lcpi)
yt.forecast[i]<-(2+(lcpi/(1-lcpi)))*cpi.smooth1[T+i-1]-
(1+(lcpi/(1-lcpi)))*cpi.smooth2[T+i-1]
cpi.hat<-2*cpi.smooth1[1:(T+i-1)]-cpi.smooth2[1:(T+i-1)]
sig.est<- sqrt(var(linear_data1[2:(T+i-1),2]- cpi.hat[1:(T+i-2)]))
```

```r
cl[i]<-qnorm(1-alpha.lev/2)*sig.est
}
plot(linear_data1[1:T+1],type="p", pch=16,cex=.5,xlab='Period',ylab='yt',
     xlim=c(1,24),ylim=c(200,750), main="one-step-ahead forecasts")
lines(linear_data1[1:T,2])
axis(1, seq(1,24,24), linear_data1[seq(1,24,24),1])
#points((T+1):(24),linear_data1[(T+1):(24),2],cex=.5)
lines((T+1):(T+tau),yt.forecast, lty =2, col= "blue")
#lines((T+1):(T+tau), yt.forecast+cl, lty =4, col = "red")
#lines((T+1):(T+tau),yt.forecast-cl, lty =6, col = "green")
legend(1, 700, legend=c(" Forecast", "Previous Observation"),
       col=c("blue", "Black"), lty=2:4, cex=0.8)
```



Explanation: Based on our optimum lamda value from 1st twelve observations, we developed a model. This model is used to forcast the last twelve observation. Above figure shows the plot of forcasted last 12 observations (blue das line) along with previous observations.

Table for forcasted errors

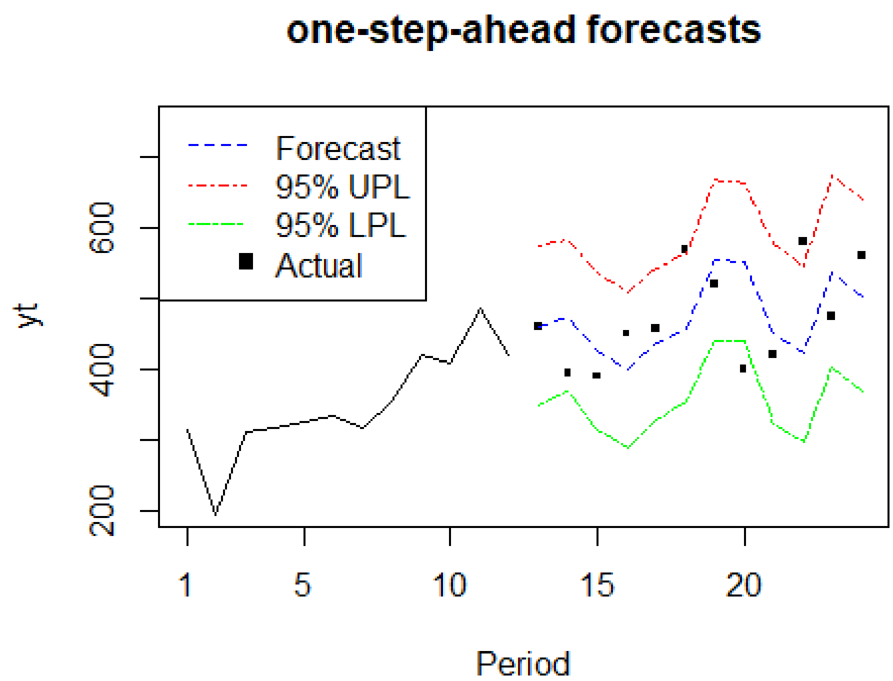
|————-| ———-|———| | Actual | Forecasted| Error | |————-| ———-|———| | 460 | 461.38 | -1.38 | | 395 | 475.67 | -80.67 | | 390 | 426.30 | -36.30 | | 450 | 399.53 | 50.47 | | 458 | 436.36 | 21.64 | | 570 | 458.20 | 111.80 | | 520 | 555.63 | -35.63 | | 400 | 553.01 | -153.01 | | 420 | 450.78 | -30.78 | | 580 | 421.85 | 158.15 | |————-| ———-|———|

We have made one-step-ahead forecast for next period using first 12 observations. To check the forcasting performance, we have estimated and plotted 95% upper prediction interval (UPL) and lower prediction level (LPL). We see from following figure that the forecast model almost capture all actual value of yt (for the period of last 12 observation) within the 95% UPL and 95% LPL. That means our one-step ahead forcast model is working satisfactorly as the real value plotted within the 95% prediction level.

```r
Actual <- linear_data1[13:24,2]
Forecast_Prediction <- yt.forecast
Error <- linear_data1[13:24,2] - yt.forecast

Table_Forecast <- data.frame (
  Actual,
  Forecast_Prediction,
  Error
)
Table_Forecast
```

```
##    Actual Forecast_Prediction       Error
## 1     460            461.3808   -1.380759
## 2     395            475.6683  -80.668271
## 3     390            426.3049  -36.304852
## 4     450            399.5252   50.474755
## 5     458            436.3605   21.639453
## 6     570            458.2036  111.796431
## 7     520            555.6345  -35.634485
## 8     400            553.0081 -153.008098
## 9     420            450.7813  -30.781302
## 10    580            421.8546  158.145352
## 11    475            539.1443  -64.144308
## 12    560            503.9055   56.094503
```

```r
plot(linear_data1[1:T+1],type="p", pch=18,cex=1,xlab='Period',ylab='yt',
     xlim=c(1,24),ylim=c(200,750), main="one-step-ahead forecasts")
lines(linear_data1[1:T,2])
axis(1, seq(1,24,24), linear_data1[seq(1,24,24),1])
points((T+1):(24),linear_data1[(T+1):(24),2],cex=.5, pch= 15, col = "black")
lines((T+1):(T+tau),yt.forecast, lty =2, col= "blue")
lines((T+1):(T+tau),yt.forecast+cl, lty =4, col = "red")
lines((T+1):(T+tau),yt.forecast-cl, lty =6, col = "green")

legend( x="topleft",
        legend=c("Forecast","95% UPL","95% LPL","Actual"),
        col=c("blue","red","green","black"), lwd=1, lty=c(2,4,6,NA),
        pch=c(NA,NA,NA,15), merge=FALSE )
```




```
# acf.forecast <- acf(yt.forecast, lag.max=12,type="correlation", plot =TRUE)
```

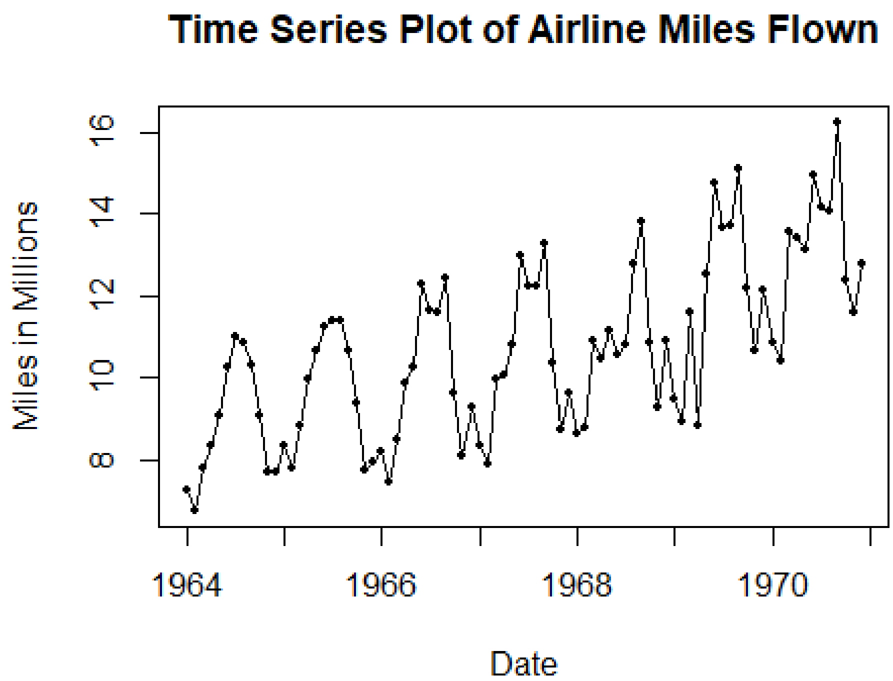
Exercise 4.27:

Table B.10 contains 7 years of monthly data on the number of airline miles flown in the United Kingdom. This is seasonal data.

a. Make a time series plot of the data and verify that it is seasonal.

Answer:

```
# Load data for 4.27 exercise
library(readxl)
AppendixB_datafile <- read_excel("AppendixB_datafile.xlsx", sheet = "B.10-FLO
WN", skip = 3)
airline_data<- ts(AppendixB_datafile[,2], start = c(1964,1), freq = 12)
plot(airline_data,type="p", pch=16,cex=.5,xlab='Date',ylab='Miles in Millions
',
main="Time Series Plot of Airline Miles Flown")
lines(airline_data[,1])
```

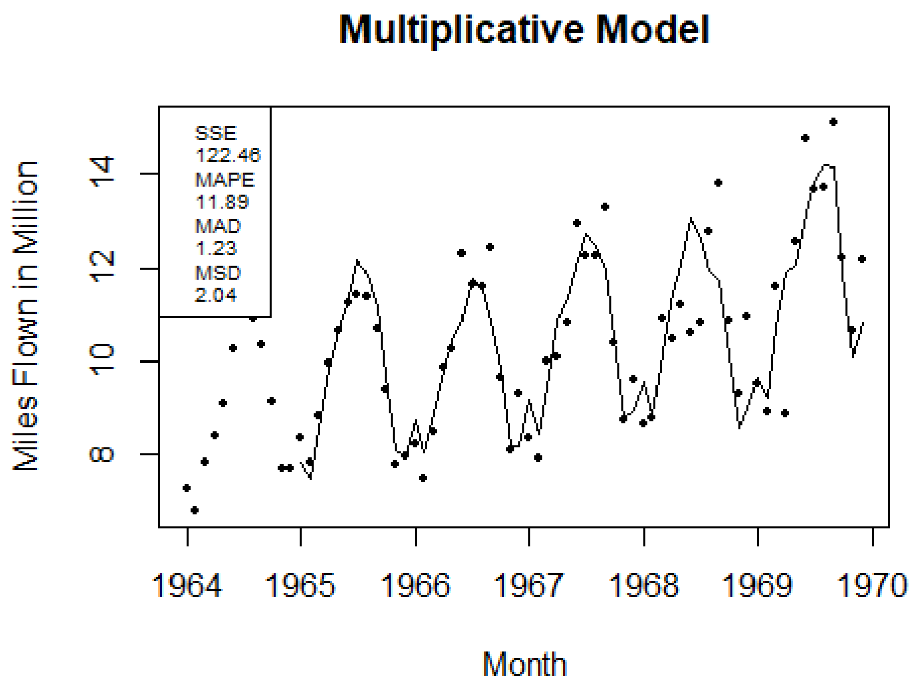


Explanation: From the time series plot, we can say that data is seasonal becasue the miles flown had been varies like as a year cycle. The value of flown increses from january to July then decreasing again to December of each year. Therefore, we can that the time series is varying seasonally.

b. Use Winters'multiplicative method for the first 6 years to develop a forecasting method for this data. How well does this smoothing procedure work?

Answer:

```
airline1<-AppendixB_datafile[1:72,]
airline1.ts<-ts(airline1[,2], start = c(1964, 1), freq = 12)
fly.hw.multi<-HoltWinters(airline1.ts,alpha=0.2,beta=0.2,gamma=0.2, seasonal=
"multiplicative")
plot(airline1.ts,type="p", pch=16,cex=.5, xlab='Month', ylab='Miles Flown in
Million', main="Multiplicative Model")

lines(fly.hw.multi$fitted[,1])
acc.multi <- measacc.fs(fly.hw.multi$fitted[,1],.4)
legend( x="topleft", legend = c("SSE",round(min(acc.multi[1]), 2), "MAPE",rou
nd(min(acc.multi[2]), 2),"MAD",round(min(acc.multi[3]), 2),"MSD", round(min(a
cc.multi[4]), 2)), cex=0.6)
```



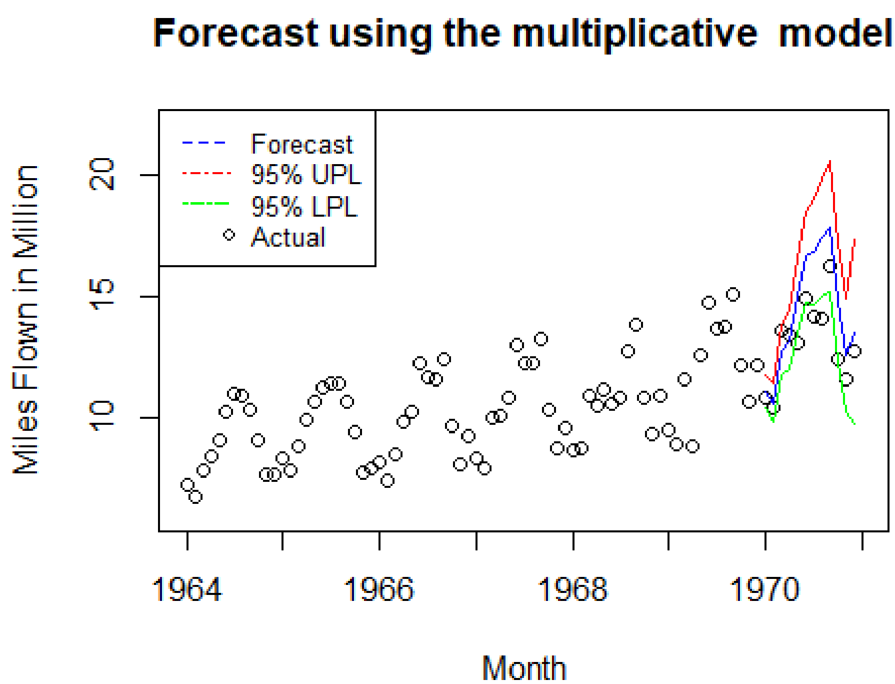
From Winters'multiplicative method for the first 6 years, we get the above smooting line that forecasting well. It seems that the multiplicative mdoel work well and however it does not able to get the all the peak value for each season after smoothing.

c. Make one-step-ahead forecasts of the last 12 months. Determine the forecast errors. How well did your procedure work in forecasting the new data?

Answer:

```
airline2 <- AppendixB_datafile[73:84,]

airline2.ts<-ts(airline2[,2], start = c(1970,1), freq = 12)

airline2.forecast<-predict(fly.hw.multi, n.ahead=12, prediction.interval =
TRUE)
plot(airline1.ts,type="p", pch=21,cex=1,xlab='Month',ylab='Miles Flown in Mil
lion', xlim=c(1964,1971), ylim=c(6,22), main='Forecast using the multiplicati
ve  model' )

points(airline2.ts)
lines(airline2.forecast[,1],col ='blue' )
lines(airline2.forecast[,2],col ='red')
lines(airline2.forecast[,3],col ='green')

legend( x="topleft",
        legend=c("Forecast","95% UPL","95% LPL","Actual"),
```

```
col=c("blue","red","green","black"), cex= .8,lwd=.5, lty=c(2,4,6,NA),
pch=c(NA,NA,NA,21), merge=FALSE )
```



From above figure, it is observing that the multiplicative model performing well for the one-step-ahead forecasts of the last 12 months. Forecasted points are closely follow the trend of seasonal pattern. It also seems that actual values are lies between the 95% upper prediction level (UPL) and 95% lower prediction levels.

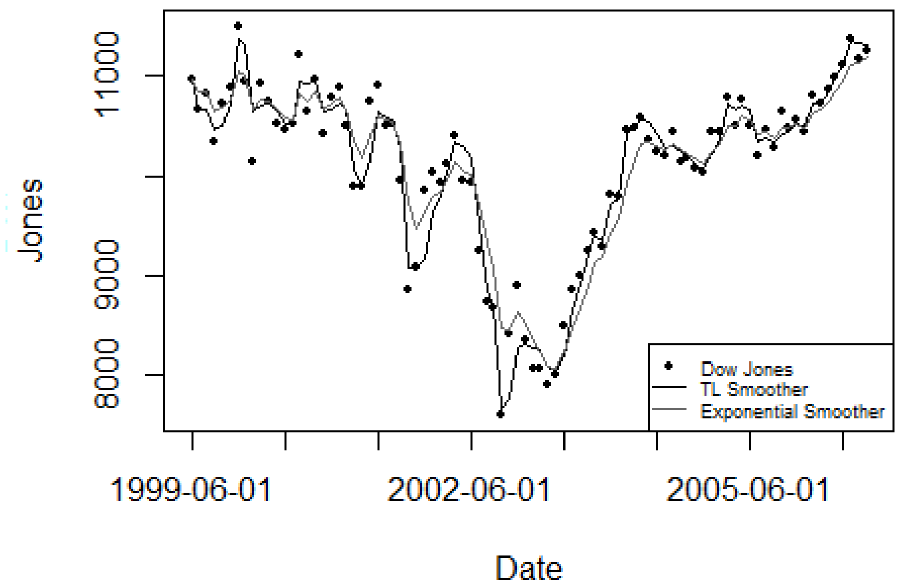
Actual	Predict	Error
10.8	11.05085	-0.25
10.4	10.61594	-0.22
13.6	12.7601	0.84
13.4	13.18773	0.21
13.1	15.02865	-1.93
14.9	16.62897	-1.73
14.1	16.83737	-2.74
14.1	17.39488	-3.29
16.2	17.85084	-1.65
12.4	14.78945	-2.39
11.6	12.58429	-0.98
12.8	13.52265	-0.72

Example 4.6:

show the plot similar to Figure 4.25 by letting the simple exponential smoother with = 0.4 and TL smoother with = 0.4. Print the first 10 calculations in a table that is similar to Table 4.9 on page 276.

Answer:

```
library(readxl)
Example4_6 <- read_excel("Example4_6.xlsx")

dji.data<-as.data.frame(Example4_6)
out.tl.dji<-tlsmooth(dji.data[,2],0.4)

#Obtain the exponential smoother for Dow Jones Index
dji.smooth1<-firstsmooth(y=dji.data[,2],lambda=0.4)
#Plot the data together with TL and exponential smoother for comparison
plot(dji.data[,2],type="p", pch=16,cex=.5,xlab='Date',ylab='Dow
Jones',xaxt='n')
axis(1, seq(1,85,12), dji.data[seq(1,85,12),1])
lines(out.tl.dji[,1])
lines(dji.smooth1,col="grey40")
legend(x="bottomright",
        legend=c("Dow Jones","TL Smoother","Exponential Smoother"),
pch=c(16, NA, NA),lwd=c(NA,.5,.5),cex=.55,col=c("black",
"black","grey40"))
```




```
Date <- dji.data[1:10,1]
Dow_Jones <- dji.data[1:10,2]
Smoothed <- out.tl.dji[1:10,1]
Lambda <- out.tl.dji[1:10,2]
Error <- out.tl.dji[1:10,3]
Qt<- out.tl.dji[1:10,4]
Dt<- out.tl.dji[1:10,5]


Table_Trigg_Leach_Smoother <- data.frame (
  Date,
  Dow_Jones,
  Smoothed,
  Lambda,
  Error,
  Qt,
  Dt
)
Table_Trigg_Leach_Smoother

##          Date Dow_Jones Smoothed     Lambda     Error        Qt       Dt
## 1  1999-06-01   10970.8 10970.80 1.00000000    0.0000   0.00000   0.0000
## 2  1999-07-01   10655.2 10655.20 1.00000000 -315.6000 -126.24000 126.2400
## 3  1999-08-01   10829.3 10662.51 0.04198536  174.1000   -6.10400 145.3840
## 4  1999-09-01   10337.0 10462.11 0.61566314 -325.5097 -133.86626 217.4343
## 5  1999-10-01   10729.9 10492.31 0.11279687  267.7946  26.79810 237.5784
## 6  1999-11-01   10877.8 10713.51 0.57381146  385.4882 170.27416 296.7423
```

```
## 7  1999-12-01   11497.1 11376.12 0.84560785   783.5907  415.60076 491.4817
## 8  2000-01-01   10940.5 11306.37 0.16010797  -435.6198   75.11256 469.1369
## 9  2000-02-01   10128.3 10639.38 0.56616891 -1178.0736 -426.16189 752.7116
## 10 2000-03-01   10921.9 10710.78 0.25271480   282.5151 -142.69111 564.6330
```