HomeWork 2

Partha S Satpathy

May 3, 2017

library(fma)

## Warning: package 'fma' was built under R version 3.3.3

## Loading required package: forecast

## Warning: package 'forecast' was built under R version 3.3.3

library(fpp)

## Loading required package: expsmooth

## Loading required package: lmtest

## Warning: package 'lmtest' was built under R version 3.3.3

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.3.3

##   
## Attaching package: 'zoo'

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

## Loading required package: tseries

## Warning: package 'tseries' was built under R version 3.3.3

library(segmented)

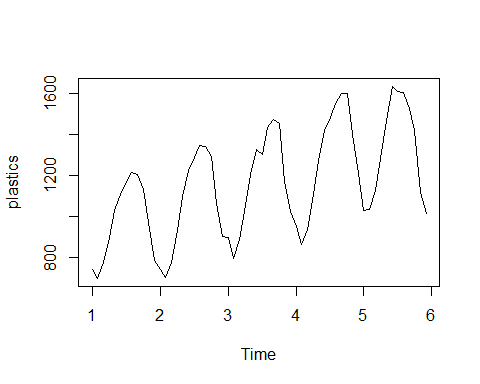
Ex 6 -Q1

ma3x5 = 1/3[1/5(yt-3 + yt-2 + yt-1 + yt + yt+1) + 1/5(yt-2 + yt-1 + yt + yt+1 + yt+2) + 1/5(yt-1 + yt + yt+1 + yt+2 + yt+3)] = 1/15yt-3 + 2/15 yt-2 + 3/15yt-1 + 3/15yt + 3/15yt+1 + 2/15 yt+2 + 1/15yt+3 = 0.067yt-3 + 0.13 yt-2 + 0.2 yt-1 + 0.2 yt + 0.2 yt+1 + 0.13 yt+2 + 0.067 yt+3 = ma7 with weights [0.067, 0.133, 0.200, 0.200, 0.200, 0.133,0.06]

#head(plastics)  
plastics

## Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec  
## 1 742 697 776 898 1030 1107 1165 1216 1208 1131 971 783  
## 2 741 700 774 932 1099 1223 1290 1349 1341 1296 1066 901  
## 3 896 793 885 1055 1204 1326 1303 1436 1473 1453 1170 1023  
## 4 951 861 938 1109 1274 1422 1486 1555 1604 1600 1403 1209  
## 5 1030 1032 1126 1285 1468 1637 1611 1608 1528 1420 1119 1013

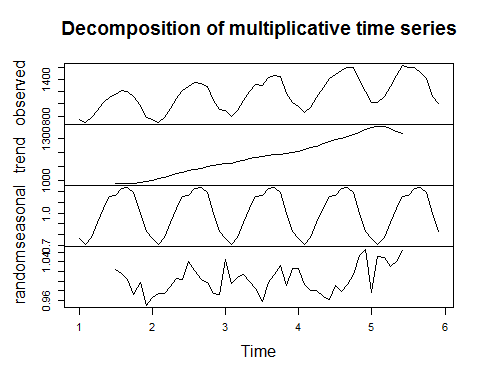
plot(plastics)



##Trend- Positive Upward  
##Seasonality - Increase with Summer and decreases when winter

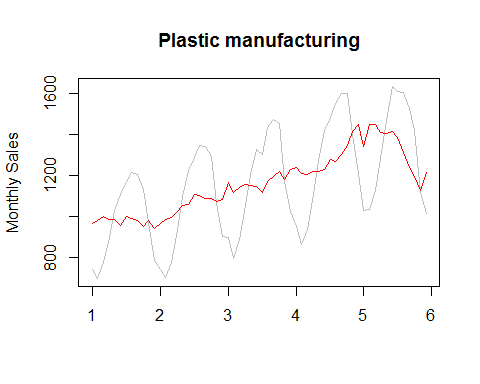
fit <- decompose(plastics, type="multiplicative")  
  
trend\_indices <- fit$trend  
seasonal\_indices <- fit$seasonal

plot(fit)

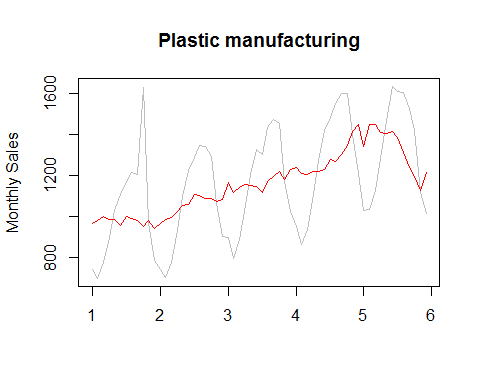


##Yes its showing the interpretation of a. The trend is positive and seasonality is present with summer and winter.

plot(plastics, col="grey",  
 main="Plastic manufacturing",  
 xlab="", ylab="Monthly Sales")  
lines(seasadj(fit),col="red",ylab="Seasonally adjusted")

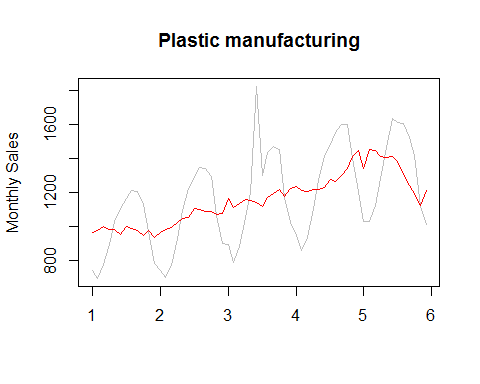


ts <- plastics  
ts[10] = ts[10] + 500  
  
fit2 <- decompose(ts, type="multiplicative")  
plot(ts, col="grey",  
 main="Plastic manufacturing",  
 xlab="", ylab="Monthly Sales")  
lines(seasadj(fit),col="red",ylab="Seasonally adjusted")

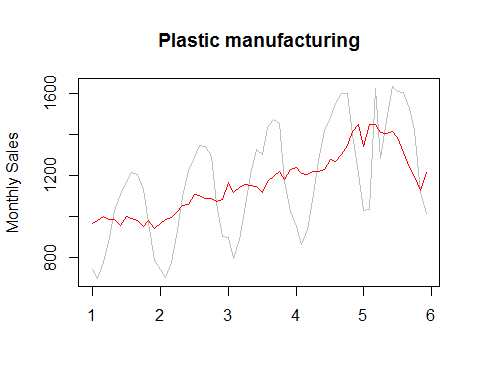


#The outlier put a spike on the graph

ts <- plastics  
ts[30] = ts[30] + 500  
  
fit2 <- decompose(ts, type="multiplicative")  
plot(ts, col="grey",  
 main="Plastic manufacturing",  
 xlab="", ylab="Monthly Sales")  
lines(seasadj(fit),col="red",ylab="Seasonally adjusted")

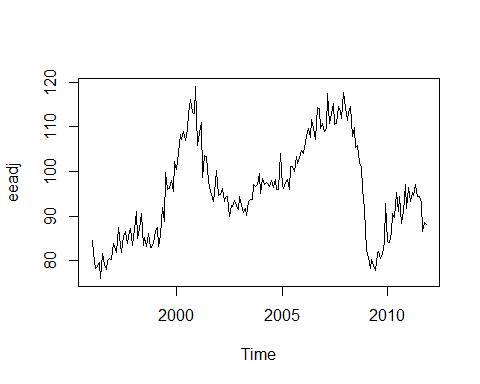


ts <- plastics  
ts[51] = ts[51] + 500  
  
fit2 <- decompose(ts, type="multiplicative")  
plot(ts, col="grey",  
 main="Plastic manufacturing",  
 xlab="", ylab="Monthly Sales")  
lines(seasadj(fit),col="red",ylab="Seasonally adjusted")

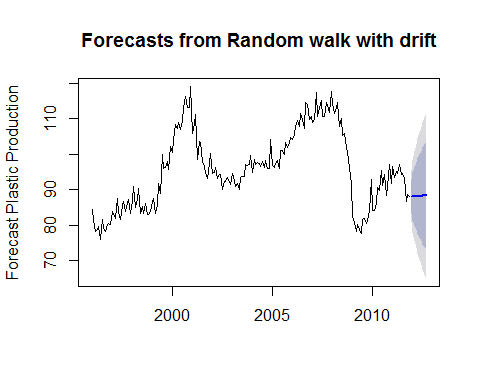


##There is no diference wherever the outlier is. It shows as a spike everywhere.

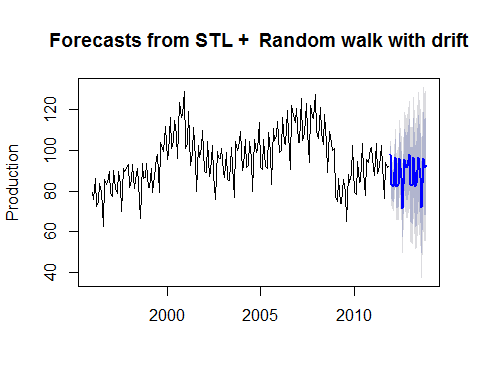
fit2 <- stl(elecequip, t.window=15, s.window="periodic", robust=TRUE)  
eeadj <- seasadj(fit2)  
plot(eeadj)



fit3 <- rwf(eeadj,drift=TRUE)  
plot(fit3, ylab="Forecast Plastic Production")



fcast <- forecast(fit2, method="rwdrift")  
plot(fcast, ylab="Production")



Ex 6, Q-3,a Seasonally adjusted series contain the remainder component as well as the trend-cycle. Therefore they are not “smooth” and “downturns” or “upturns” can be misleading. If the purpose is to look for turning points in the series, and interpret any changes in the series, then it is better to use the trend-cycle component rather than the seasonally adjusted data.

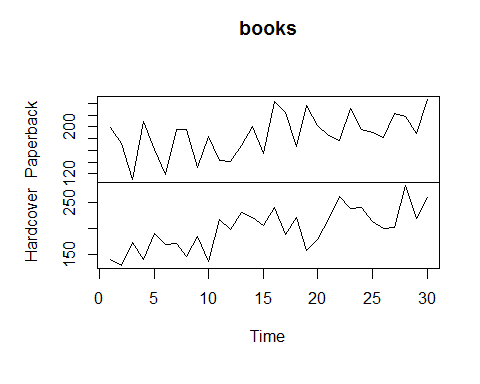
Ex6, Q-3.b The 1991-92 recession is particularly visible in the remainder aph. We can also see the same in the overall Data. However, there is no change in the trend and seasinality.

## Exercise 7

print(books)

## Time Series:  
## Start = 1   
## End = 30   
## Frequency = 1   
## Paperback Hardcover  
## 1 199 139  
## 2 172 128  
## 3 111 172  
## 4 209 139  
## 5 161 191  
## 6 119 168  
## 7 195 170  
## 8 195 145  
## 9 131 184  
## 10 183 135  
## 11 143 218  
## 12 141 198  
## 13 168 230  
## 14 201 222  
## 15 155 206  
## 16 243 240  
## 17 225 189  
## 18 167 222  
## 19 237 158  
## 20 202 178  
## 21 186 217  
## 22 176 261  
## 23 232 238  
## 24 195 240  
## 25 190 214  
## 26 182 200  
## 27 222 201  
## 28 217 283  
## 29 188 220  
## 30 247 259

plot(books)



paper <- books[,1]  
  
alpha <- seq(0,1,0.05)  
sse <- 0  
  
for (i in 1:21){  
 print(i)  
 fit <- ses(paper, initial='simple', alpha=alpha[i], h=3)  
 fitsse <- sum((paper - fitted(fit)))   
 sse <- c(sse,fitsse)  
}

## [1] 1  
## [1] 2  
## [1] 3  
## [1] 4  
## [1] 5  
## [1] 6  
## [1] 7  
## [1] 8  
## [1] 9  
## [1] 10  
## [1] 11  
## [1] 12  
## [1] 13  
## [1] 14  
## [1] 15  
## [1] 16  
## [1] 17  
## [1] 18  
## [1] 19  
## [1] 20  
## [1] 21

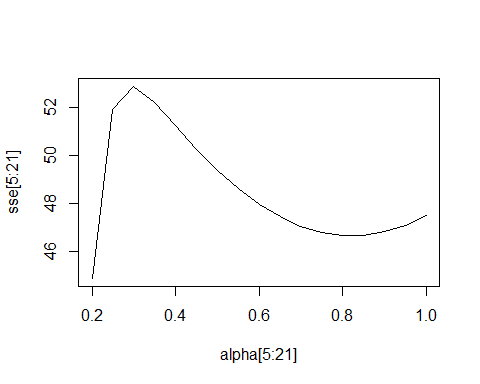
alpha

## [1] 0.00 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40 0.45 0.50 0.55 0.60 0.65  
## [15] 0.70 0.75 0.80 0.85 0.90 0.95 1.00

sse

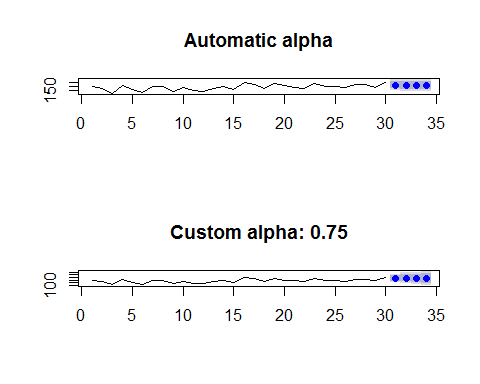
## [1] 0.00000 -378.00000 -77.48985 17.05107 44.87901 51.94123  
## [7] 52.90100 52.24224 51.25594 50.28218 49.40049 48.62893  
## [13] 47.97497 47.44545 47.04681 46.78404 46.65983 46.67402  
## [19] 46.82334 47.10121 47.49777 48.00000

plot(alpha[5:21],sse[5:21],type = 'l')



##The minimum error is coming for alpha = 0.75

fit1 <- ses(paper, initial='simple', h=4)  
fit2 <- ses(paper, initial='simple', alpha=0.75, h=4)  
par(mfrow=c(2,1))  
plot(fit1, main="Automatic alpha")  
plot(fit2, main="Custom alpha: 0.75")



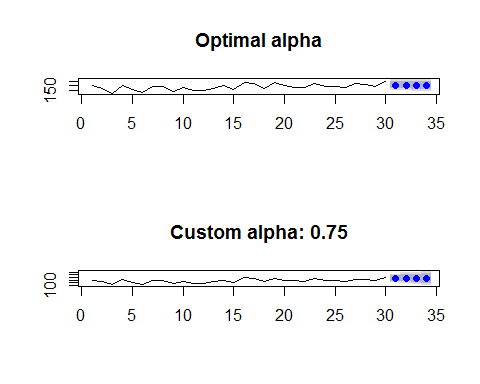
fit1 <- ses(paper, initial='optimal', h=4)  
sum((paper - fitted(fit1)))

## [1] 215.2864

fit2 <- ses(paper, initial='simple', alpha=0.75, h=4)  
sum((paper - fitted(fit2)))

## [1] 46.65983

par(mfrow=c(2,1))  
plot(fit1, main="Optimal alpha")  
plot(fit2, main="Custom alpha: 0.75")



##Optimal is not an improvement. It's sse is too high.

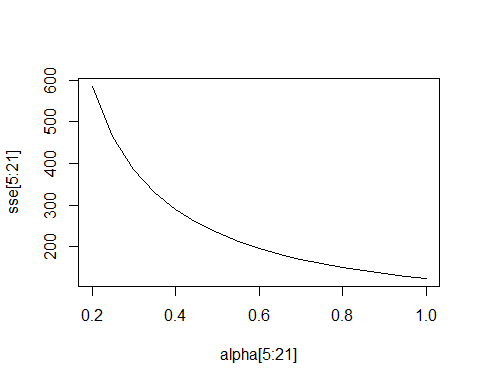
hardcover <- books[,2]  
  
alpha <- seq(0,1,0.05)  
sse <- 0  
  
for (i in 1:21){  
 #print(i)  
 fit <- ses(hardcover, initial='simple', alpha=alpha[i], h=3)  
 fitsse <- sum((hardcover - fitted(fit)))   
 sse <- c(sse,fitsse)  
}  
alpha

## [1] 0.00 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40 0.45 0.50 0.55 0.60 0.65  
## [15] 0.70 0.75 0.80 0.85 0.90 0.95 1.00

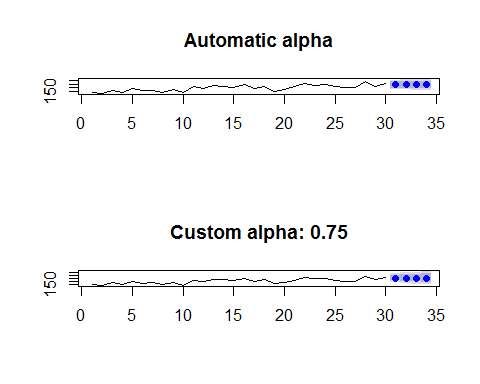
sse

## [1] 0.0000 1795.0000 1127.9566 780.6708 584.8486 465.0875 386.1845  
## [8] 330.8279 289.9775 258.6009 233.7191 213.4710 196.6436 182.4209  
## [15] 170.2422 159.7155 150.5632 142.5857 135.6372 129.6090 124.4182  
## [22] 120.0000

plot(alpha[5:21],sse[5:21],type = 'l')



optAlpha <- 1 ##As found from the graph  
##----------------------------  
  
fit1 <- ses(hardcover, initial='simple', h=4)  
fit2 <- ses(hardcover, initial='simple', alpha=0.75, h=4)  
par(mfrow=c(2,1))  
plot(fit1, main="Automatic alpha")  
plot(fit2, main="Custom alpha: 0.75")



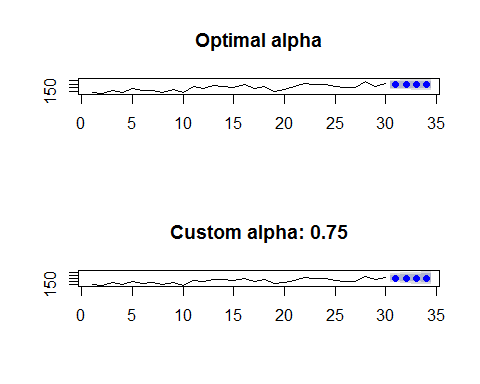
##----------------------------  
fit1 <- ses(hardcover, initial='optimal', h=4)  
sum((hardcover - fitted(fit1)))

## [1] 275.0075

fit2 <- ses(hardcover, initial='simple', alpha=0.75, h=4)  
sum((hardcover - fitted(fit2)))

## [1] 150.5632

par(mfrow=c(2,1))  
plot(fit1, main="Optimal alpha")  
plot(fit2, main="Custom alpha: 0.75")



##Optimal Alpha works best

fit1 <- holt(paper, alpha=0.8, beta=0.2, initial="simple", h=5)   
sum((paper - fitted(fit1)))

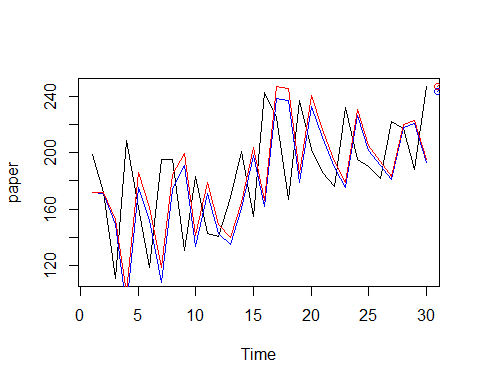
## [1] 212.902

fit2 <- holt(paper, alpha=0.8, beta=0.2, initial="simple", exponential=TRUE, h=5)  
sum((paper - fitted(fit2)))

## [1] 42.15357

##Exponential smoothing works better than the normal one

plot(paper)  
lines(fitted(fit1), col="blue")   
lines(fitted(fit2), col="red")  
lines(fit1$mean, col="blue", type="o")   
lines(fit2$mean, col="red", type="o")



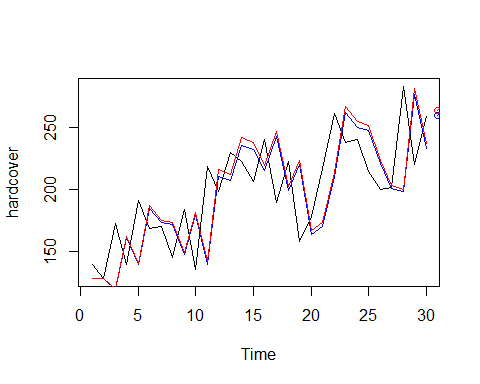
fit1 <- holt(hardcover, alpha=0.8, beta=0.2, initial="simple", h=5)   
sum((hardcover - fitted(fit1)))

## [1] 103.8246

fit2 <- holt(hardcover, alpha=0.8, beta=0.2, initial="simple", exponential=TRUE, h=5)  
sum((hardcover - fitted(fit2)))

## [1] 12.29919

plot(hardcover)  
lines(fitted(fit1), col="blue")   
lines(fitted(fit2), col="red")  
lines(fit1$mean, col="blue", type="o")   
lines(fit2$mean, col="red", type="o")



fit1$upper[,2] - fit1$lower[,2]

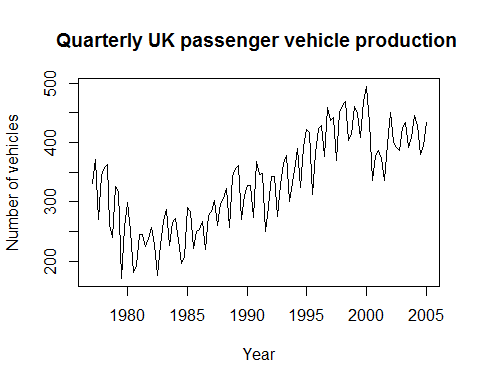
## Time Series:  
## Start = 31   
## End = 35   
## Frequency = 1   
## [1] 150.7754 235.5186 316.2691 397.7727 481.5379

fit2$upper[,2] - fit2$lower[,2]

## Time Series:  
## Start = 31   
## End = 35   
## Frequency = 1   
## [1] 221.4873 336.0263 453.2694 616.7795 807.0461

#Exponential method works better as the 95% interval is smaller

cars\_data <- ukcars  
plot(cars\_data, main="Quarterly UK passenger vehicle production ", xlab="Year", ylab="Number of vehicles")



decomposed <- stl(cars\_data, s.window="periodic", robust=TRUE)  
seasonal <- decomposed$time.series[,1]  
  
cars\_seasonal <- cars\_data - seasonal

fit\_1 <- holt(cars\_seasonal, h=8, damped = TRUE)  
  
lastyear <- rep(decomposed$time.series[110:113,"seasonal"],2)  
reseasonalized\_fc1 <- fit\_1$mean + lastyear  
  
summary(fit\_1)

##   
## Forecast method: Damped Holt's method  
##   
## Model Information:  
## Damped Holt's method   
##   
## Call:  
## holt(y = cars\_seasonal, h = 8, damped = TRUE)   
##   
## Smoothing parameters:  
## alpha = 0.5666   
## beta = 3e-04   
## phi = 0.9117   
##   
## Initial states:  
## l = 346.0865   
## b = -9.7583   
##   
## sigma: 25.2032  
##   
## AIC AICc BIC   
## 1275.490 1276.283 1291.854   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set 2.518454 25.20318 20.5804 0.3038991 6.585405 0.6707052  
## ACF1  
## Training set 0.0353549  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 2005 Q2 407.4117 375.1125 439.7109 358.0144 456.8090  
## 2005 Q3 407.4110 370.2796 444.5425 350.6234 464.1987  
## 2005 Q4 407.4104 366.0038 448.8170 344.0845 470.7364  
## 2006 Q1 407.4099 362.1274 452.6924 338.1563 476.6635  
## 2006 Q2 407.4094 358.5554 456.2634 332.6936 482.1251  
## 2006 Q3 407.4089 355.2254 459.5924 327.6011 487.2167  
## 2006 Q4 407.4085 352.0939 462.7230 322.8122 492.0048  
## 2007 Q1 407.4081 349.1291 465.6871 318.2780 496.5382

fit\_2 <- holt(cars\_seasonal, h=8)  
  
reseasonalized\_fc2 <- fit\_2$mean + lastyear  
  
summary(fit\_2)

##   
## Forecast method: Holt's method  
##   
## Model Information:  
## Holt's method   
##   
## Call:  
## holt(y = cars\_seasonal, h = 8)   
##   
## Smoothing parameters:  
## alpha = 0.6012   
## beta = 1e-04   
##   
## Initial states:  
## l = 343.3854   
## b = 0.6617   
##   
## sigma: 25.3907  
##   
## AIC AICc BIC   
## 1275.166 1275.726 1288.803   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set -0.1407116 25.39072 20.14514 -0.5931913 6.500319 0.6565204  
## ACF1  
## Training set 0.02953472  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 2005 Q2 408.3701 375.8306 440.9096 358.6052 458.1350  
## 2005 Q3 409.0302 371.0594 447.0010 350.9588 467.1015  
## 2005 Q4 409.6903 366.9717 452.4088 344.3579 475.0226  
## 2006 Q1 410.3503 363.3600 457.3407 338.4848 482.2158  
## 2006 Q2 411.0104 360.1043 461.9166 333.1562 488.8646  
## 2006 Q3 411.6705 357.1278 466.2132 328.2546 495.0864  
## 2006 Q4 412.3306 354.3779 470.2832 323.6997 500.9615  
## 2007 Q1 412.9906 351.8168 474.1644 319.4334 506.5479

fit\_3 <- ets(cars\_seasonal)  
fit\_3

## ETS(A,N,N)   
##   
## Call:  
## ets(y = cars\_seasonal)   
##   
## Smoothing parameters:  
## alpha = 0.6115   
##   
## Initial states:  
## l = 319.5835   
##   
## sigma: 25.2942  
##   
## AIC AICc BIC   
## 1270.304 1270.525 1278.487

summary(fit\_1)

##   
## Forecast method: Damped Holt's method  
##   
## Model Information:  
## Damped Holt's method   
##   
## Call:  
## holt(y = cars\_seasonal, h = 8, damped = TRUE)   
##   
## Smoothing parameters:  
## alpha = 0.5666   
## beta = 3e-04   
## phi = 0.9117   
##   
## Initial states:  
## l = 346.0865   
## b = -9.7583   
##   
## sigma: 25.2032  
##   
## AIC AICc BIC   
## 1275.490 1276.283 1291.854   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set 2.518454 25.20318 20.5804 0.3038991 6.585405 0.6707052  
## ACF1  
## Training set 0.0353549  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 2005 Q2 407.4117 375.1125 439.7109 358.0144 456.8090  
## 2005 Q3 407.4110 370.2796 444.5425 350.6234 464.1987  
## 2005 Q4 407.4104 366.0038 448.8170 344.0845 470.7364  
## 2006 Q1 407.4099 362.1274 452.6924 338.1563 476.6635  
## 2006 Q2 407.4094 358.5554 456.2634 332.6936 482.1251  
## 2006 Q3 407.4089 355.2254 459.5924 327.6011 487.2167  
## 2006 Q4 407.4085 352.0939 462.7230 322.8122 492.0048  
## 2007 Q1 407.4081 349.1291 465.6871 318.2780 496.5382

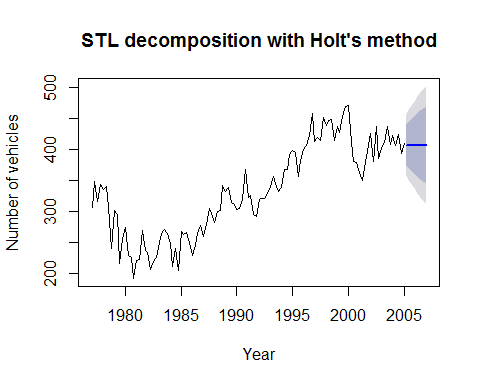
summary(fit\_2)

##   
## Forecast method: Holt's method  
##   
## Model Information:  
## Holt's method   
##   
## Call:  
## holt(y = cars\_seasonal, h = 8)   
##   
## Smoothing parameters:  
## alpha = 0.6012   
## beta = 1e-04   
##   
## Initial states:  
## l = 343.3854   
## b = 0.6617   
##   
## sigma: 25.3907  
##   
## AIC AICc BIC   
## 1275.166 1275.726 1288.803   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set -0.1407116 25.39072 20.14514 -0.5931913 6.500319 0.6565204  
## ACF1  
## Training set 0.02953472  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 2005 Q2 408.3701 375.8306 440.9096 358.6052 458.1350  
## 2005 Q3 409.0302 371.0594 447.0010 350.9588 467.1015  
## 2005 Q4 409.6903 366.9717 452.4088 344.3579 475.0226  
## 2006 Q1 410.3503 363.3600 457.3407 338.4848 482.2158  
## 2006 Q2 411.0104 360.1043 461.9166 333.1562 488.8646  
## 2006 Q3 411.6705 357.1278 466.2132 328.2546 495.0864  
## 2006 Q4 412.3306 354.3779 470.2832 323.6997 500.9615  
## 2007 Q1 412.9906 351.8168 474.1644 319.4334 506.5479

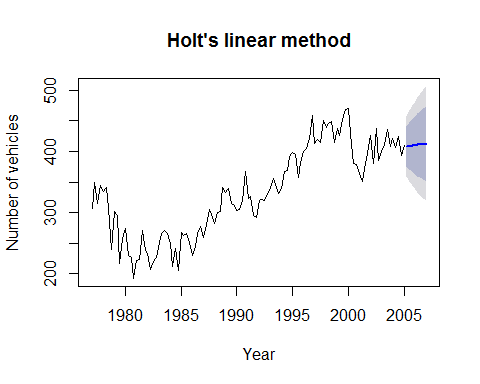
summary(fit\_3)

## ETS(A,N,N)   
##   
## Call:  
## ets(y = cars\_seasonal)   
##   
## Smoothing parameters:  
## alpha = 0.6115   
##   
## Initial states:  
## l = 319.5835   
##   
## sigma: 25.2942  
##   
## AIC AICc BIC   
## 1270.304 1270.525 1278.487   
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set 1.268464 25.29416 20.22177 -0.1453187 6.505687 0.6590176  
## ACF1  
## Training set 0.0263301

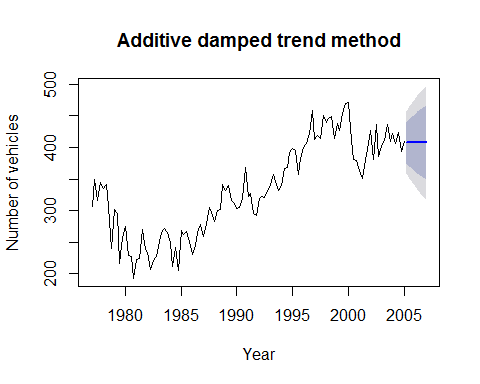
pred\_3 <- predict(fit\_3)  
plot(pred\_3, main="STL decomposition with Holt's method", xlab="Year", ylab="Number of vehicles")



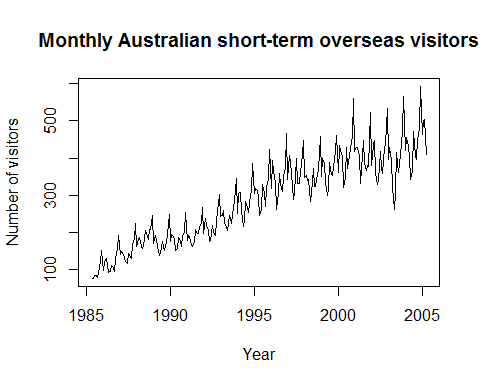
pred\_2 <- predict(fit\_2)  
plot(pred\_2, main="Holt's linear method", xlab="Year", ylab="Number of vehicles")



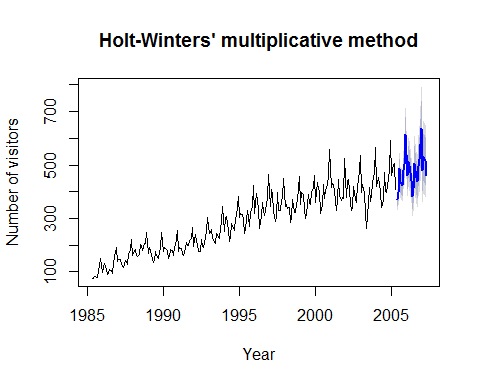
pred\_1 <- predict(fit\_1)  
plot(pred\_1, main="Additive damped trend method", xlab="Year", ylab="Number of vehicles")



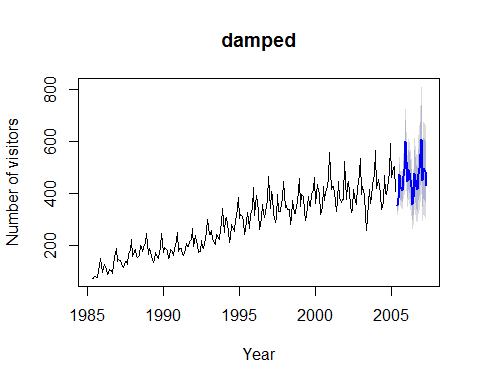
visitor\_data <- visitors  
plot(visitor\_data, main="Monthly Australian short-term overseas visitors", xlab="Year", ylab="Number of visitors")



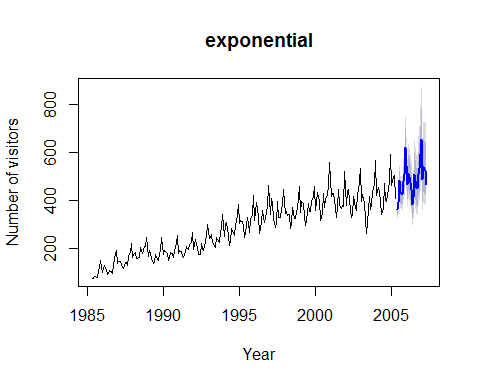
Forecast\_twoyears <- hw(visitor\_data, h=24, seasonal='multiplicative')  
plot(Forecast\_twoyears, main="Holt-Winters' multiplicative method", xlab="Year", ylab="Number of visitors")



Forecast\_twoyears1 <- hw(visitors, h=24, seasonal='multiplicative', damped=TRUE)  
plot(Forecast\_twoyears1, main="damped", xlab="Year", ylab="Number of visitors")



Forecast\_twoyears2 <- hw(visitors, h=24, seasonal='multiplicative', exponential=TRUE)  
plot(Forecast\_twoyears2, main="exponential", xlab="Year", ylab="Number of visitors")



plot(Forecast\_twoyears,ylab="Monthly Australian visitors", plot.conf=FALSE,  
 fcol="white", xlab="Year")

## Warning in plot.window(xlim, ylim, log, ...): "plot.conf" is not a  
## graphical parameter

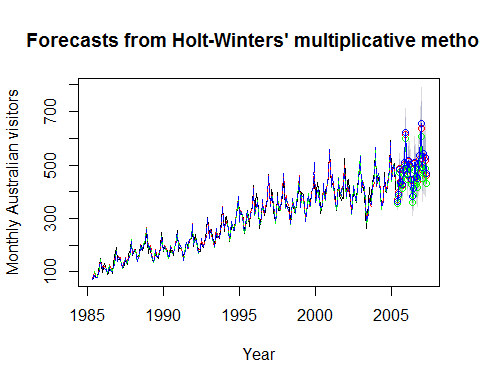
## Warning in title(main = main, xlab = xlab, ylab = ylab, ...): "plot.conf"  
## is not a graphical parameter

## Warning in axis(1, ...): "plot.conf" is not a graphical parameter

## Warning in axis(2, ...): "plot.conf" is not a graphical parameter

## Warning in box(...): "plot.conf" is not a graphical parameter

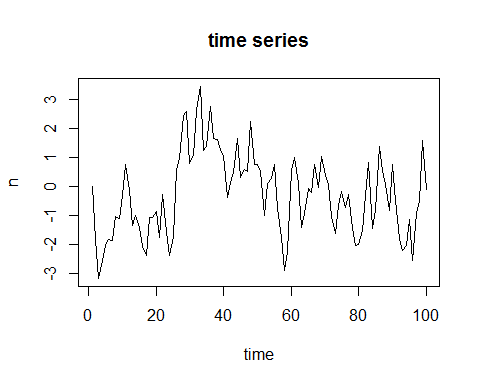
lines(fitted(Forecast\_twoyears), col="red", lty=2)  
lines(fitted(Forecast\_twoyears1), col="green", lty=2)  
lines(fitted(Forecast\_twoyears2), col='blue', lty=2)  
lines(Forecast\_twoyears$mean, type="o", col="red")  
lines(Forecast\_twoyears1$mean, type="o", col="green")  
lines(Forecast\_twoyears2$mean, type="o", col="blue")



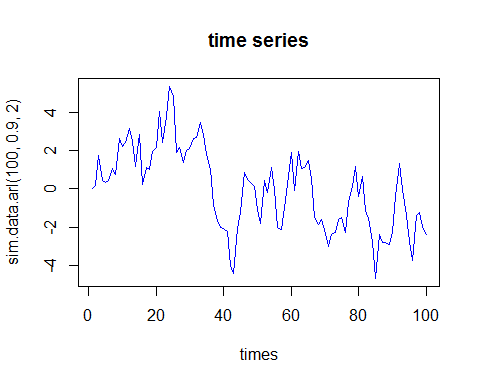
vis\_fit1 <- holt(visitor\_data, seasonal='multiplicative')  
vis\_fit2 <- ets(visitor\_data)

## Exercise 8.11

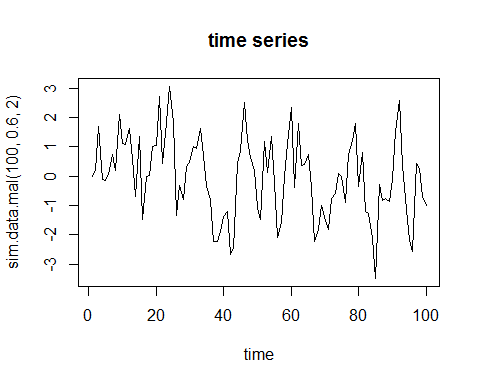
#a  
n <- ts(numeric(100))  
e<- rnorm(100)  
for(i in 2:100)  
 n[i] <- 0.6\*n[i-1] + e[i]  
  
#b  
plot(n, main="time series", xlab="time")



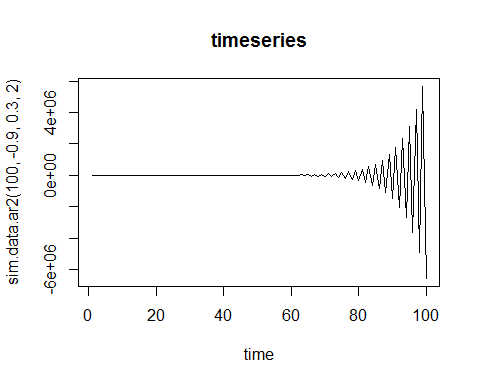
sim.data.arl <- function(n.obs, phi, seed.nr){  
 set.seed(seed.nr)  
 a <- ts(numeric(n.obs))  
 e <- rnorm(n.obs)  
 for (i in 2:n.obs)  
 a[i] <- phi\*a[i-1] + e[i]  
 return(a)  
}  
plot(sim.data.arl(100, 0.9, 2), main= "time series", xlab="times")  
lines(sim.data.arl(100, 0.9, 2), col="blue")



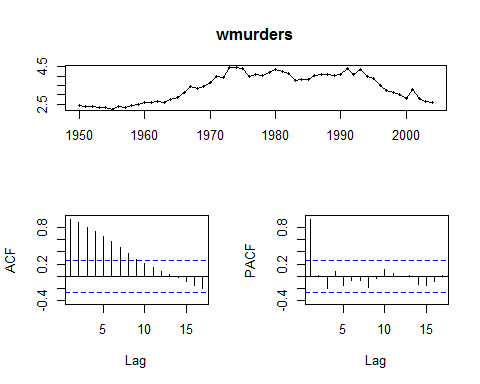
#c  
sim.data.mal <- function(n.obs, theta, seed.nr){  
 set.seed(seed.nr)  
 a <- ts(numeric(n.obs))  
 e <- rnorm(n.obs)  
 for (i in 2:n.obs)  
 a[i] <- theta\*a[i-1] + e[i]  
 return(a)  
}  
  
#d  
plot(sim.data.mal(100, 0.6, 2), main="time series", xlab="time")



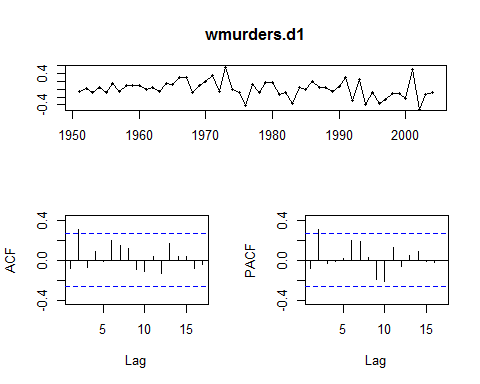
#e  
sim.data.armal1 <- function(n.obs, theta, seed.nr){  
 set.seed(seed.nr)  
 a <- ts(numeric(n.obs))  
 e <- rnorm(n.obs)  
 for (i in 2:n.obs)  
 a[i] <- phi\*a[i-1] + theta\*e[i-1] + e[i]  
 return(a)  
}  
  
#f  
sim.data.ar2 <- function(n.obs, phi1, phi2, seed.nr){  
 set.seed(seed.nr)  
 a <- ts(numeric(n.obs))  
 e <- rnorm(n.obs)  
 for(i in 3: n.obs)  
 a[i] <- phi1\*a[i-1] + phi2\*a[i-2] + e[i]  
 return(a)  
}  
plot(sim.data.ar2(100, -0.9, 0.3, 2), main="timeseries", xlab="time")



#a  
data(wmurders)  
tsdisplay(wmurders)



wmurders.d1 <- diff(wmurders)  
tsdisplay(wmurders.d1)



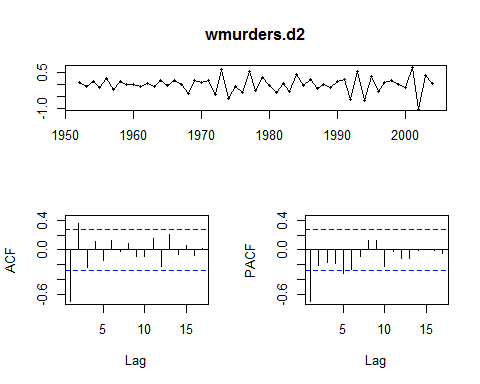
adf.test(wmurders.d1)

##   
## Augmented Dickey-Fuller Test  
##   
## data: wmurders.d1  
## Dickey-Fuller = -3.7688, Lag order = 3, p-value = 0.02726  
## alternative hypothesis: stationary

kpss.test(wmurders.d1)

##   
## KPSS Test for Level Stationarity  
##   
## data: wmurders.d1  
## KPSS Level = 0.58729, Truncation lag parameter = 1, p-value =  
## 0.02379

wmurders.d2 <- diff(diff(wmurders))  
tsdisplay(wmurders.d2)



adf.test(wmurders.d2)

## Warning in adf.test(wmurders.d2): p-value smaller than printed p-value

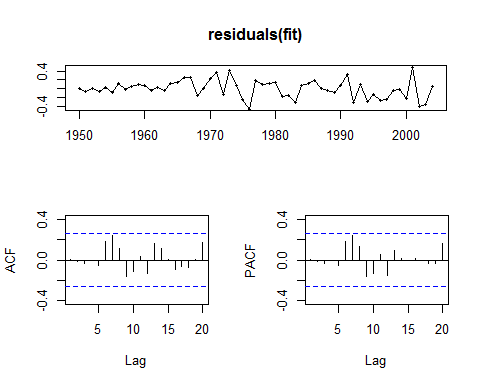
##   
## Augmented Dickey-Fuller Test  
##   
## data: wmurders.d2  
## Dickey-Fuller = -5.1646, Lag order = 3, p-value = 0.01  
## alternative hypothesis: stationary

kpss.test(wmurders.d2)

## Warning in kpss.test(wmurders.d2): p-value greater than printed p-value

##   
## KPSS Test for Level Stationarity  
##   
## data: wmurders.d2  
## KPSS Level = 0.030483, Truncation lag parameter = 1, p-value = 0.1

fit <- Arima(wmurders, order=c(0, 1, 2))  
tsdisplay(residuals(fit), lag.max=20)



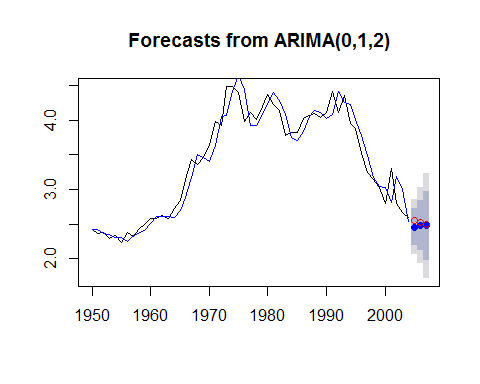
Box.test(residuals(fit), lag=24, fitdf=4, type="Ljung")

##   
## Box-Ljung test  
##   
## data: residuals(fit)  
## X-squared = 22.127, df = 20, p-value = 0.3337

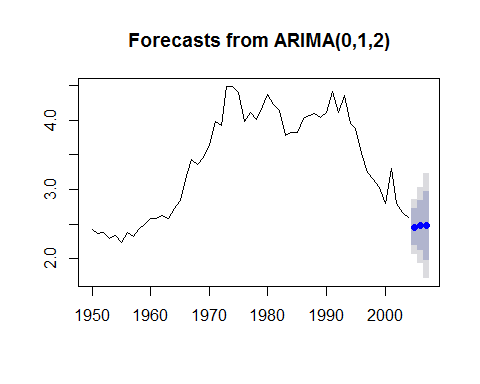
##As the test shows that the residual is only White Noise, we can conclude that ARIMA is the best model.  
  
#b  
#A constant will bring drift which we do not have. So, we will not use constant.  
  
##c  
#(1−B)(1−Bm)yt=(1−B−Bm+Bm+1)yt=yt−yt−1−yt−m+yt−m−1  
  
##d  
fit <- Arima(wmurders, order=c(0, 1, 2))  
tsdisplay(residuals(fit), lag.max=20)  
  
##e  
fcast <- forecast(fit, h=3)  
fcast$mean

## Time Series:  
## Start = 2005   
## End = 2007   
## Frequency = 1   
## [1] 2.458450 2.477101 2.477101

## Time Series:  
## Start = 2005   
## End = 2007   
## Frequency = 1   
## [1] 2.458450 2.477101 2.477101  
toforecast <- 3  
yt <- fit$x  
et <- fit$residuals  
theta1 <- as.numeric(fit$coef['ma2'])  
theta2 <- as.numeric(fit$coef['ma1'])  
  
for (h in 1:toforecast){  
 n <- length(yt)  
 y\_tp1 <- 2 \* yt[n] - yt[n - 1] + theta1 \* et[n] + theta2 \* et[n - 1]  
 yt <- c(yt, y\_tp1)  
 et <- c(et, 0)}  
  
f <- yt[(length(yt) - toforecast + 1):length(yt)]  
plot(fcast)  
lines(fit$x - fit$residuals, col='blue')  
points(c(2005, 2006, 2007), f, col='red')



##f  
plot(fcast)



##g  
auto.arima(wmurders)

## Series: wmurders   
## ARIMA(1,2,1)   
##   
## Coefficients:  
## ar1 ma1  
## -0.2434 -0.8261  
## s.e. 0.1553 0.1143  
##   
## sigma^2 estimated as 0.04632: log likelihood=6.44  
## AIC=-6.88 AICc=-6.39 BIC=-0.97