Forecasting: Holt-Winters

Dean D'souza

August 2, 2016

## Solutions:

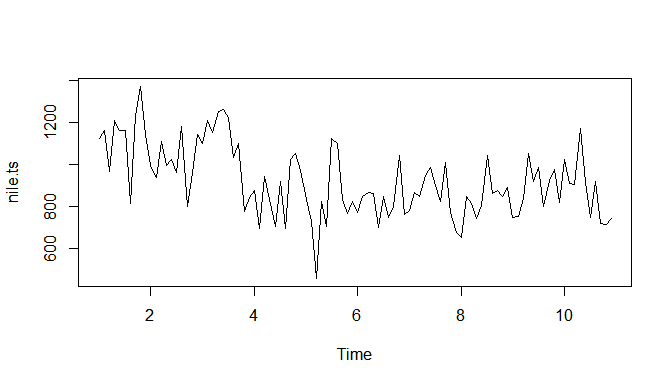
* Repeat the ts(), HoltWinters(), predict() and plot() functions on the Nile data as in these slides

We execute the R commands found in the lecture slides as follow:

# Setting up the nile data to be usable  
nile.ts<-ts(Nile,start=1,frequency=10)  
summary(nile.ts)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 456.0 798.5 893.5 919.4 1032.0 1370.0

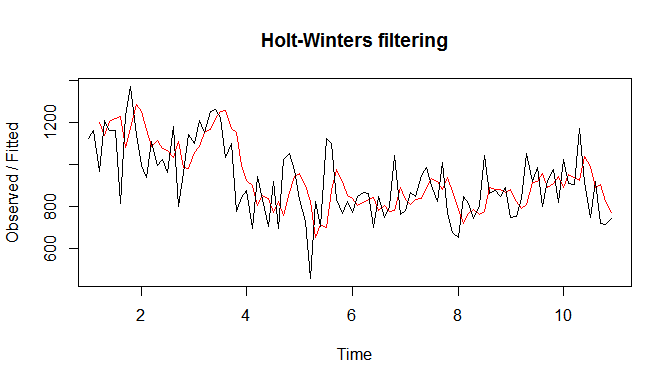
# Plotting the original data  
plot(nile.ts)



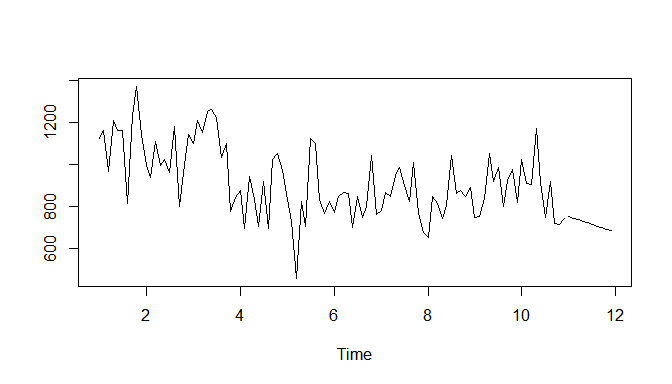
#Creating the Holt-Winters object without gamma  
nile.hw1<-HoltWinters(nile.ts,gamma = FALSE)  
nile.hw1

## Holt-Winters exponential smoothing with trend and without seasonal component.  
##   
## Call:  
## HoltWinters(x = nile.ts, gamma = FALSE)  
##   
## Smoothing parameters:  
## alpha: 0.4190643  
## beta : 0.05987705  
## gamma: FALSE  
##   
## Coefficients:  
## [,1]  
## a 756.913740  
## b -7.424597

# Plotting the Holt-Winters object  
plot(nile.hw1)



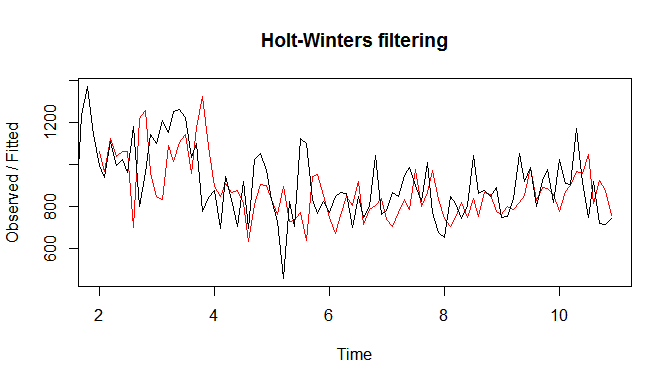
# Predicting future values with the Holt-winters Object  
nile.p1<-predict(nile.hw1, n.ahead=10)  
ts.plot(nile.ts,nile.p1)



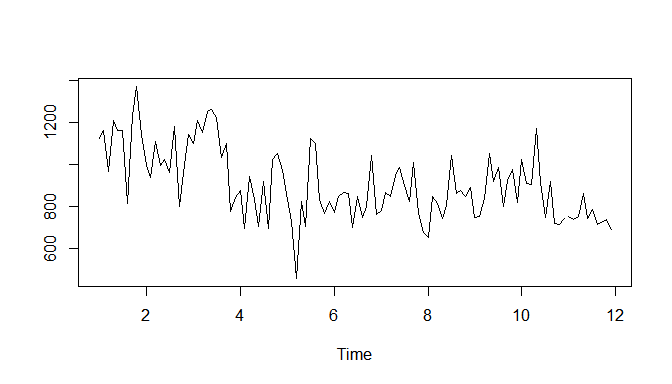
#Creating the new Holt-Winters object with gamma  
nile.hw2<-HoltWinters(nile.ts)  
nile.hw2

## Holt-Winters exponential smoothing with trend and additive seasonal component.  
##   
## Call:  
## HoltWinters(x = nile.ts)  
##   
## Smoothing parameters:  
## alpha: 0.2288315  
## beta : 0.01243079  
## gamma: 0.3072457  
##   
## Coefficients:  
## [,1]  
## a 739.945079  
## b -6.201272  
## s1 17.617748  
## s2 9.950334  
## s3 27.634432  
## s4 145.943614  
## s5 34.090131  
## s6 80.026344  
## s7 18.347819  
## s8 34.150186  
## s9 50.729266  
## s10 9.567410

# Plotting the new Holt-Winters object  
plot(nile.hw2)



# Predicting future values with the new Holt-winters Object  
nile.p2<-predict(nile.hw2, n.ahead=10)  
ts.plot(nile.ts,nile.p2)



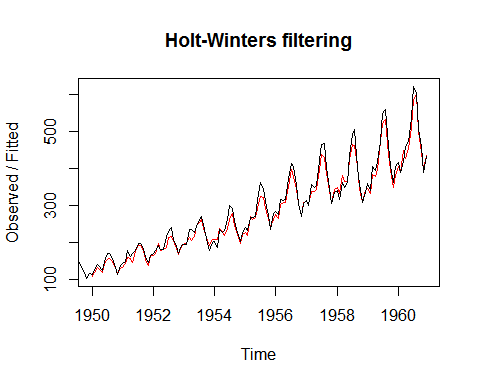
* Repeat the same functions with different values of alpha, beta, and gamma of your choosing on.
* AirPassengers

We perform the functions on the AirPassengers data set present in base R as follows:

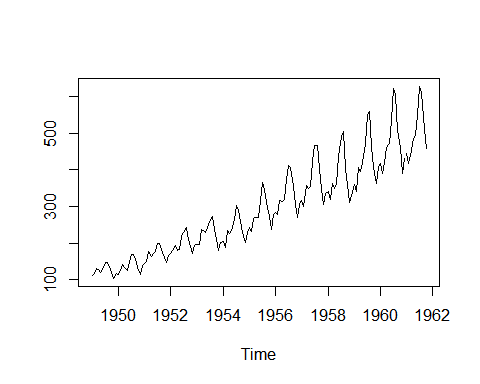
# creating the Holt-Winters Object  
ap.hw1<-HoltWinters(AirPassengers, alpha = 0.25,beta = FALSE, gamma = TRUE)  
ap.hw1

## Holt-Winters exponential smoothing without trend and with additive seasonal component.  
##   
## Call:  
## HoltWinters(x = AirPassengers, alpha = 0.25, beta = FALSE, gamma = TRUE)  
##   
## Smoothing parameters:  
## alpha: 0.25  
## beta : FALSE  
## gamma: TRUE  
##   
## Coefficients:  
## [,1]  
## a 415.845264  
## s1 27.478626  
## s2 1.042023  
## s3 36.298594  
## s4 69.700836  
## s5 76.194373  
## s6 133.182693  
## s7 211.340789  
## s8 193.810700  
## s9 93.653015  
## s10 42.162294  
## s11 -26.433647  
## s12 16.154736

# PLotting the Model  
plot(ap.hw1)



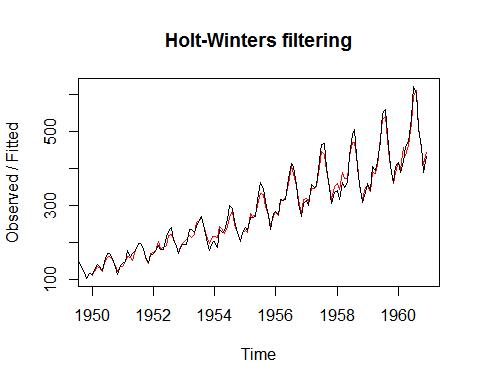
# Predicting the Future values  
ap.p1<-predict(ap.hw1, n.ahead=10)  
ts.plot(AirPassengers,ap.p1)



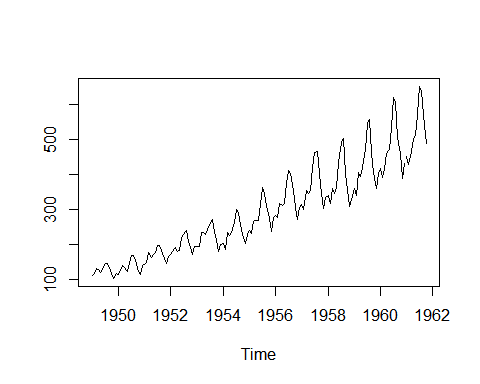
# creating the Holt-Winters Object such that the algorithm sets alpha, beta and gamma  
ap.hw2<-HoltWinters(AirPassengers)  
ap.hw2

## Holt-Winters exponential smoothing with trend and additive seasonal component.  
##   
## Call:  
## HoltWinters(x = AirPassengers)  
##   
## Smoothing parameters:  
## alpha: 0.2479595  
## beta : 0.03453373  
## gamma: 1  
##   
## Coefficients:  
## [,1]  
## a 477.827781  
## b 3.127627  
## s1 -27.457685  
## s2 -54.692464  
## s3 -20.174608  
## s4 12.919120  
## s5 18.873607  
## s6 75.294426  
## s7 152.888368  
## s8 134.613464  
## s9 33.778349  
## s10 -18.379060  
## s11 -87.772408  
## s12 -45.827781

# PLotting the Model  
plot(ap.hw2)



# Predicting the Future values  
ap.p2<-predict(ap.hw2, n.ahead=10)  
ts.plot(AirPassengers,ap.p2)



In the first model created above we set alpha to 0.25 to get a closer approximation for predicting the values. We also set Beta to False to perform exponential smoothing. This is to done so as to maintain the upward trend seen in the data. We also set gamma to true in order to take into account the cyclical effect seen in the data. The second model gives us the actual alpha, beta and gamma values that should be used for a model with good fit.

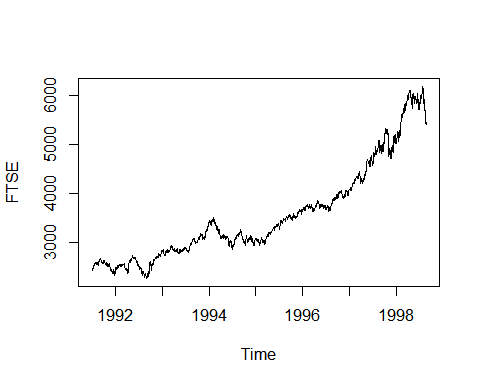
* EuStockMarkets
* When using the EuStockMarkets, choose one column

We perform the functions on the EUStockMarkets data set present in base R as follows (The column chosen for the purposes of this assignment is the UK FTSE column):

# Getting the data ready for Holt-Winters  
FTSE<-EuStockMarkets[,4]  
head(FTSE)

## [1] 2443.6 2460.2 2448.2 2470.4 2484.7 2466.8

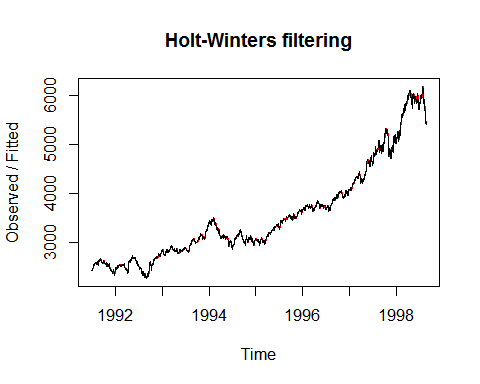
# Verifying the time series  
plot(FTSE)



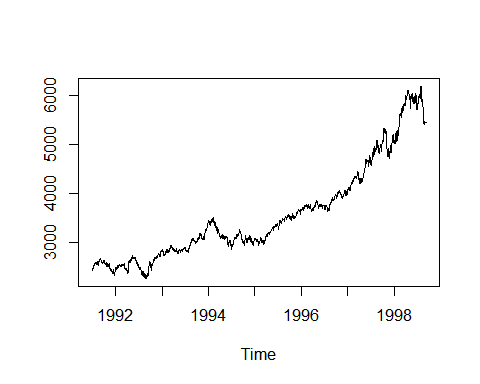
# creating the Holt-Winters Object  
FTSE.hw1<-HoltWinters(FTSE, alpha = 0.9 ,beta = FALSE , gamma = FALSE)  
FTSE.hw1

## Holt-Winters exponential smoothing without trend and without seasonal component.  
##   
## Call:  
## HoltWinters(x = FTSE, alpha = 0.9, beta = FALSE, gamma = FALSE)  
##   
## Smoothing parameters:  
## alpha: 0.9  
## beta : FALSE  
## gamma: FALSE  
##   
## Coefficients:  
## [,1]  
## a 5450.064

# PLotting the Model  
plot(FTSE.hw1)



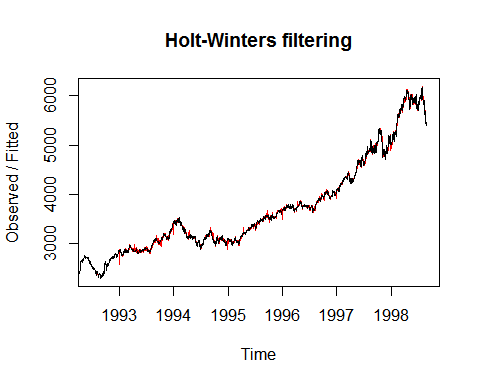
# Predicting the Future values  
FTSE.p1<-predict(FTSE.hw1, n.ahead=10)  
ts.plot(FTSE,FTSE.p1)



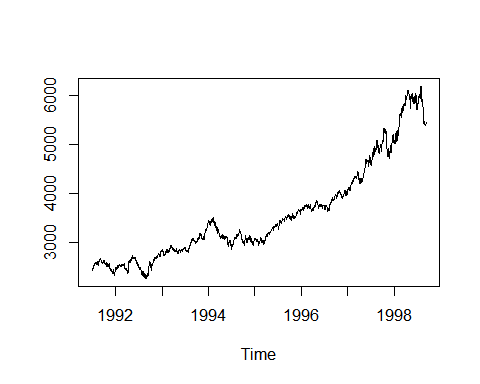
# creating the Holt-Winters Object such that the algorithm sets alpha, beta and gamma  
FTSE.hw2<-HoltWinters(FTSE)  
FTSE.hw2

## Holt-Winters exponential smoothing with trend and additive seasonal component.  
##   
## Call:  
## HoltWinters(x = FTSE)  
##   
## Smoothing parameters:  
## alpha: 0.8759868  
## beta : 0.0001496175  
## gamma: 1  
##   
## Coefficients:  
## [,1]  
## a 5668.6646378  
## b 0.9828073  
## s1 -245.8423679  
## s2 -263.5349111  
## s3 -280.3752755  
## s4 -271.0715691  
## s5 -274.3183739  
## s6 -269.5911955  
## s7 -284.1485059  
## s8 -275.9178652  
## s9 -258.2009758  
## s10 -236.5120897  
## s11 -213.0011399  
## s12 -232.6971760  
## s13 -249.3084052  
## s14 -256.3139132  
## s15 -243.3912316  
## s16 -208.7800406  
## s17 -220.9764916  
## s18 -232.3524458  
## s19 -173.4137700  
## s20 -95.3782455  
## s21 -56.0730747  
## s22 -32.3727451  
## s23 -28.8292439  
## s24 -4.3458449  
## s25 -14.8967221  
## s26 -34.2961951  
## s27 -50.6786249  
## s28 -49.6287981  
## s29 -58.0167237  
## s30 -37.3400894  
## s31 -118.6577959  
## s32 -127.0755749  
## s33 -110.6700850  
## s34 -105.1027480  
## s35 -86.0134626  
## s36 -82.7301828  
## s37 -62.8960893  
## s38 -63.4800558  
## s39 -71.3733087  
## s40 -62.2023246  
## s41 -57.8513554  
## s42 -43.5712526  
## s43 -18.3143462  
## s44 13.6422025  
## s45 18.9083109  
## s46 14.1879237  
## s47 21.4051518  
## s48 2.4344119  
## s49 -16.2388121  
## s50 11.3181543  
## s51 17.4894541  
## s52 36.2158289  
## s53 58.4203147  
## s54 53.5017804  
## s55 54.0413496  
## s56 55.0731316  
## s57 60.0516318  
## s58 65.1354354  
## s59 63.0484744  
## s60 45.4841830  
## s61 50.3465704  
## s62 42.5575832  
## s63 43.1285044  
## s64 56.2302301  
## s65 77.2508839  
## s66 89.0470229  
## s67 69.6148281  
## s68 59.3105236  
## s69 84.1932558  
## s70 116.0749370  
## s71 116.8150236  
## s72 132.9573487  
## s73 132.9210073  
## s74 119.4454504  
## s75 103.3547786  
## s76 113.8613433  
## s77 117.3371563  
## s78 102.3221506  
## s79 96.7130623  
## s80 62.4558384  
## s81 68.9992865  
## s82 61.0358389  
## s83 59.4960157  
## s84 64.8589956  
## s85 103.1608816  
## s86 139.1136191  
## s87 164.4472994  
## s88 169.4670843  
## s89 170.2775512  
## s90 150.0853376  
## s91 37.6133133  
## s92 -15.4391975  
## s93 -34.2018993  
## s94 -40.9137619  
## s95 -35.6303484  
## s96 -20.5115264  
## s97 -39.7031984  
## s98 -55.8281224  
## s99 -56.4756254  
## s100 -37.0740533  
## s101 -34.1002231  
## s102 -36.3533782  
## s103 -28.6595029  
## s104 -6.3809719  
## s105 -8.4925395  
## s106 9.4690580  
## s107 26.0966385  
## s108 11.9508364  
## s109 16.2526643  
## s110 6.9538905  
## s111 -2.1619416  
## s112 12.8892174  
## s113 24.2725840  
## s114 20.7400910  
## s115 26.9918638  
## s116 47.4893825  
## s117 51.8267423  
## s118 45.2695210  
## s119 34.1998167  
## s120 24.1515422  
## s121 23.2871531  
## s122 16.4993965  
## s123 7.4820002  
## s124 6.2415123  
## s125 10.6408479  
## s126 8.6014868  
## s127 29.1708322  
## s128 30.0428765  
## s129 29.8099203  
## s130 33.5212452  
## s131 36.0880257  
## s132 42.9779117  
## s133 51.7435164  
## s134 52.3912941  
## s135 55.4917580  
## s136 52.6275968  
## s137 50.0562436  
## s138 71.9709925  
## s139 54.4235582  
## s140 43.1415867  
## s141 57.9798843  
## s142 65.3395508  
## s143 38.9152843  
## s144 20.1346809  
## s145 6.4366168  
## s146 -9.0042988  
## s147 -4.0157607  
## s148 -18.6912019  
## s149 -20.3490768  
## s150 -28.5647401  
## s151 -41.9707821  
## s152 -30.3105597  
## s153 -26.9102820  
## s154 -22.8429748  
## s155 -35.4703643  
## s156 -43.1938543  
## s157 -36.3842553  
## s158 -57.0093126  
## s159 -76.5753996  
## s160 -86.0104952  
## s161 -94.3360428  
## s162 -86.2323416  
## s163 -78.0115598  
## s164 -64.4926349  
## s165 8.8643266  
## s166 74.9354853  
## s167 98.1908163  
## s168 127.0400498  
## s169 149.4997614  
## s170 149.2649906  
## s171 150.8354818  
## s172 153.0265711  
## s173 128.5140019  
## s174 119.4339125  
## s175 135.9719569  
## s176 160.2705530  
## s177 164.0940973  
## s178 170.6407383  
## s179 162.5648422  
## s180 163.4947759  
## s181 152.4033530  
## s182 185.3838049  
## s183 190.9636981  
## s184 212.4300101  
## s185 227.8435462  
## s186 235.6573211  
## s187 240.4527128  
## s188 238.6049519  
## s189 221.5711035  
## s190 213.6049637  
## s191 214.4070804  
## s192 200.0265944  
## s193 209.5142957  
## s194 200.2205782  
## s195 207.8027736  
## s196 196.7870915  
## s197 210.2461965  
## s198 206.1878628  
## s199 199.0910266  
## s200 196.2020400  
## s201 192.9681512  
## s202 200.9850302  
## s203 174.5852865  
## s204 176.4338119  
## s205 163.9154214  
## s206 142.3577359  
## s207 135.4339178  
## s208 136.0595409  
## s209 112.5418023  
## s210 109.9492167  
## s211 102.9372427  
## s212 97.9942586  
## s213 83.4305780  
## s214 51.2403071  
## s215 48.5589445  
## s216 30.4813865  
## s217 25.2299139  
## s218 27.4829612  
## s219 15.7296685  
## s220 -11.2719167  
## s221 -19.2522161  
## s222 -6.8681243  
## s223 -14.6230383  
## s224 -36.5773941  
## s225 -42.3355664  
## s226 -56.8120342  
## s227 -45.8599343  
## s228 -43.1774099  
## s229 -34.1205190  
## s230 -43.4767510  
## s231 -57.7177872  
## s232 -66.2712199  
## s233 -60.4860844  
## s234 -66.3620942  
## s235 -91.6385451  
## s236 -110.8186628  
## s237 -117.2581393  
## s238 -136.1601702  
## s239 -154.6724999  
## s240 -160.7966427  
## s241 -193.2209027  
## s242 -185.9712193  
## s243 -179.9723741  
## s244 -152.0752177  
## s245 -164.6066211  
## s246 -153.6928944  
## s247 -145.8368639  
## s248 -143.4001273  
## s249 -161.9960568  
## s250 -196.9254493  
## s251 -218.3569641  
## s252 -244.1553477  
## s253 -258.0991254  
## s254 -254.2911032  
## s255 -232.5106897  
## s256 -230.4202205  
## s257 -227.7819877  
## s258 -209.0036566  
## s259 -219.8214687  
## s260 -213.6646378

# PLotting the Model  
plot(FTSE.hw2)



# Predicting the Future values  
FTSE.p2<-predict(FTSE.hw2, n.ahead=10)  
ts.plot(FTSE,FTSE.p2)



In the first model created above we set alpha to 0.9 to get a closer approximation for predicting the values. We also set Beta to FALSE to perform exponential smoothing. We also set gamma to False as we do not want to fit a cyclical model (as the data does not show significant cyclical effects). The second model gives us the actual alpha, beta and gamma values that should be used for a model with good fit.