UE20CS312 - Data Analytics - Worksheet 3a - Basic Forecasting Techniques

PES University

Reshmi Pradeep, Dept. of CSE - PES2UG20CS270

2022-10-05

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.1
## v readr   2.1.2     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(fpp2)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
## -- Attaching packages ---------------------------------------------- fpp2 2.4 --
## v forecast  8.18     v expsmooth 2.3
## v fma       2.4
```

```
df <- read.csv('sales.csv')
head(df)
```

```
##   X       Date   Sales Holiday_Flag Temperature Fuel_Price      CPI
## 1 0 05-02-2010 2135144            0       43.76      2.598 126.4421
## 2 1 12-02-2010 2188307            1       28.84      2.573 126.4963
## 3 2 19-02-2010 2049860            0       36.45      2.540 126.5263
## 4 3 26-02-2010 1925729            0       41.36      2.590 126.5523
## 5 4 05-03-2010 1971057            0       43.49      2.654 126.5783
## 6 5 12-03-2010 1894324            0       49.63      2.704 126.6043
##   Unemployment Laptop_Demand
## 1        8.623             0
## 2        8.623             0
## 3        8.623             0
## 4        8.623             0
## 5        8.623             1
## 6        8.623             0
```

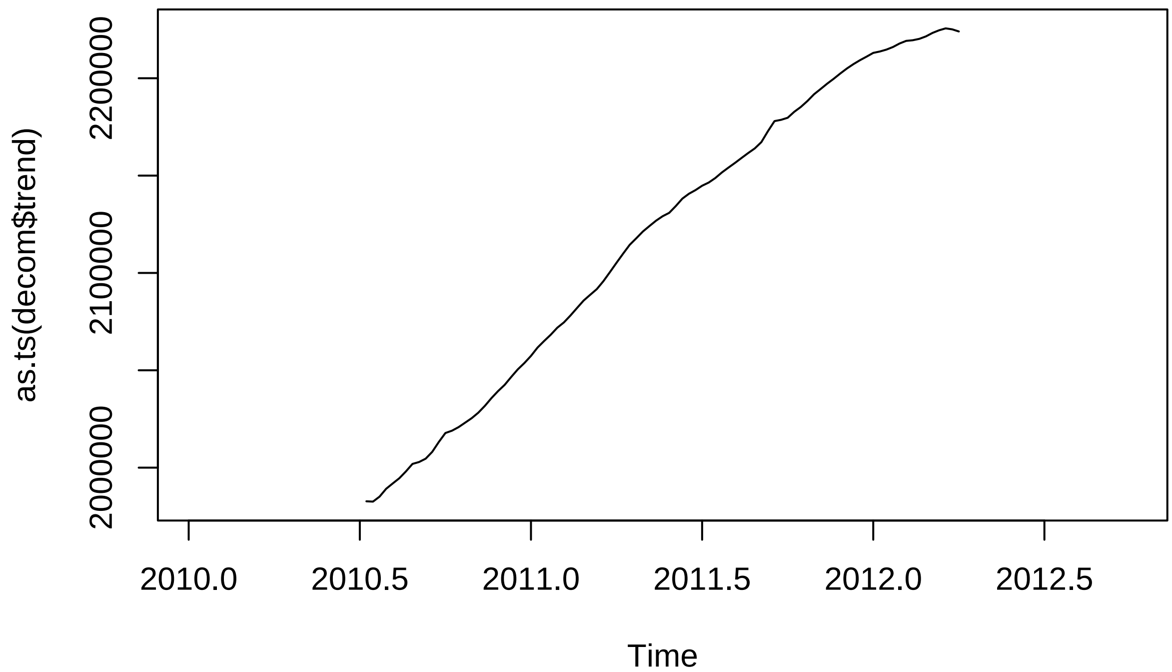
```r
sales <- df$Sales
head(sales)
```

```
## [1] 2135144 2188307 2049860 1925729 1971057 1894324
```

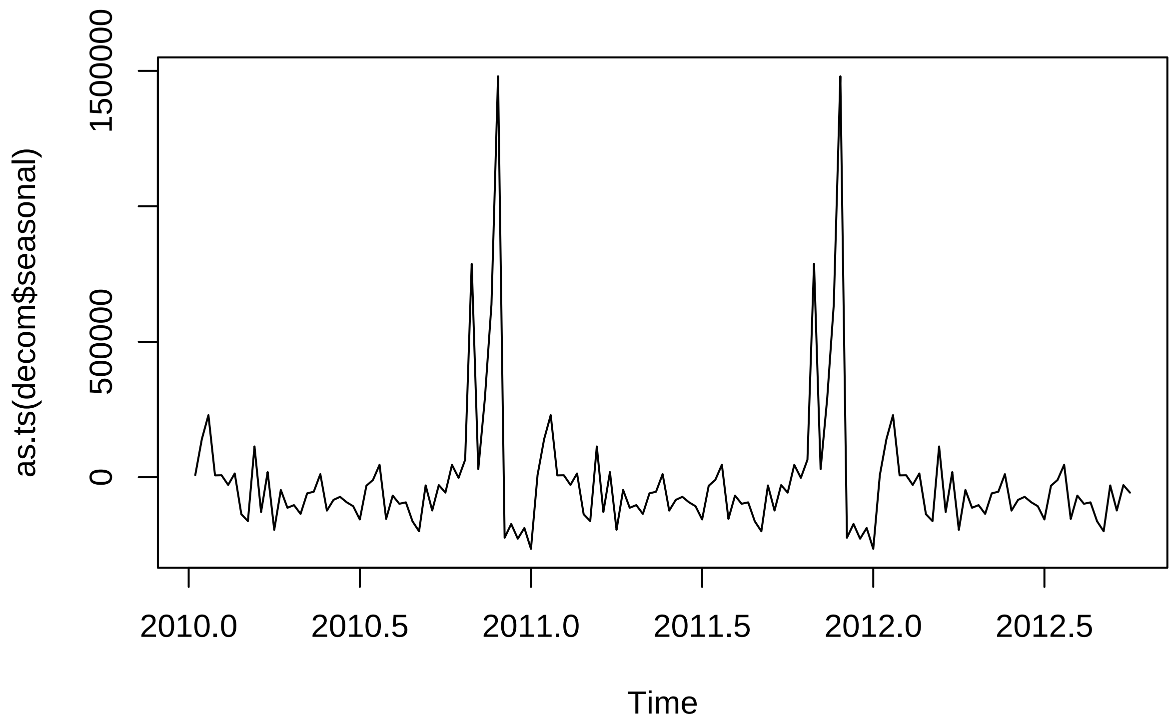
```r
sales_ts <- ts(sales, frequency = 52, start=c(2010, 2, 5))
sales_ts
```

```
## Time Series:
## Start = c(2010, 2)
## End = c(2012, 40)
## Frequency = 52
##   [1] 2135144 2188307 2049860 1925729 1971057 1894324 1897429 1762539 1979247
##  [10] 1818453 1851520 1802678 1817273 2000626 1875597 1903753 1857534 1903291
##  [19] 1870619 1929736 1846652 1881337 1812208 1898428 1848427 1796638 1907639
##  [28] 2007051 1997181 1848404 1935858 1865821 1899960 1810685 1842821 1951495
##  [37] 1867345 1927610 1933333 2013116 1999794 2097809 2789469 2102530 2302505
##  [46] 2740057 3526713 1794869 1862476 1865502 1886394 1814241 2119086 2187847
##  [55] 2316496 2078095 2103456 2039818 2116475 1944164 1900246 2074953 1960588
##  [64] 2220601 1878167 2063683 2002362 2015563 1986598 2065377 2073951 2141211
##  [73] 2008345 2051534 2066542 2049047 2036231 1989674 2160057 2105669 2232892
##  [82] 1988490 2078420 2093139 2075577 2031406 1929487 2166738 2074549 2207742
##  [91] 2151660 2281217 2203029 2243947 3004702 2180999 2508955 2771397 3676389
## [100] 2007106 2047766 1941677 2005098 1928721 2173374 2374661 2427640 2226662
## [109] 2206320 2202451 2214967 2091593 2089382 2470206 2105301 2144337 2064066
## [118] 2196968 2127661 2207215 2154138 2179361 2245257 2234191 2197300 2128363
## [127] 2224499 2100253 2175564 2048614 2174514 2193368 2283540 2125242 2081181
## [136] 2125105 2117855 2119439 2027620 2209835 2133026 2097267 2149594
```

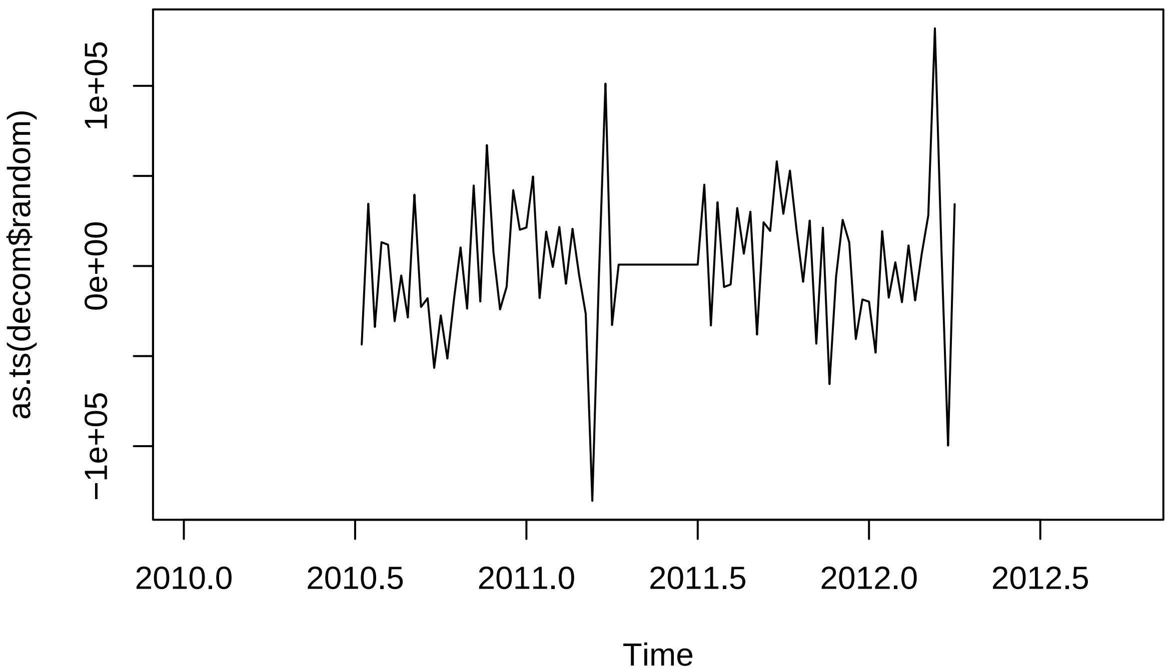
```r
###Problem 1

decom <- decompose(sales_ts, 'additive')
plot(as.ts(decom$trend))
```



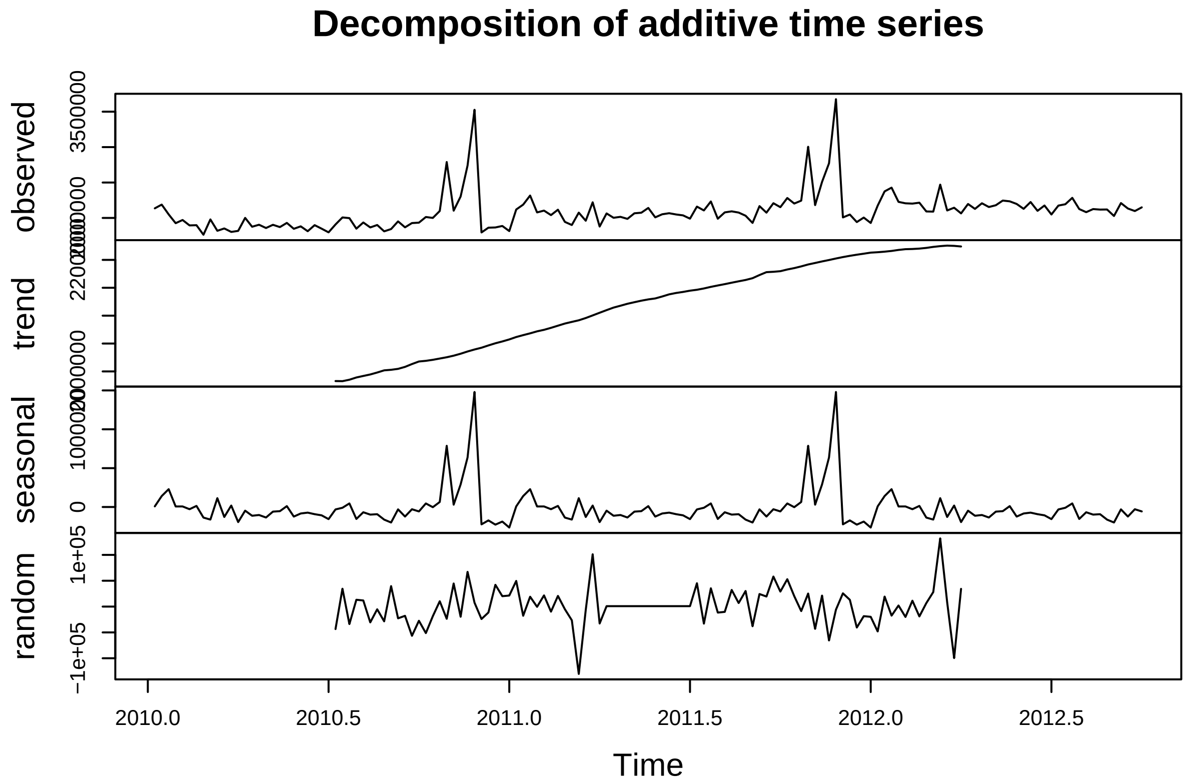

```
plot(as.ts(decom$seasonal))
```




```
plot(as.ts(decom$random))
```


as.ts(decom$random)
-1e+05
0e+00
1e+05
2010.0
2010.5
2011.0
2011.5
2012.0
2012.5
Time

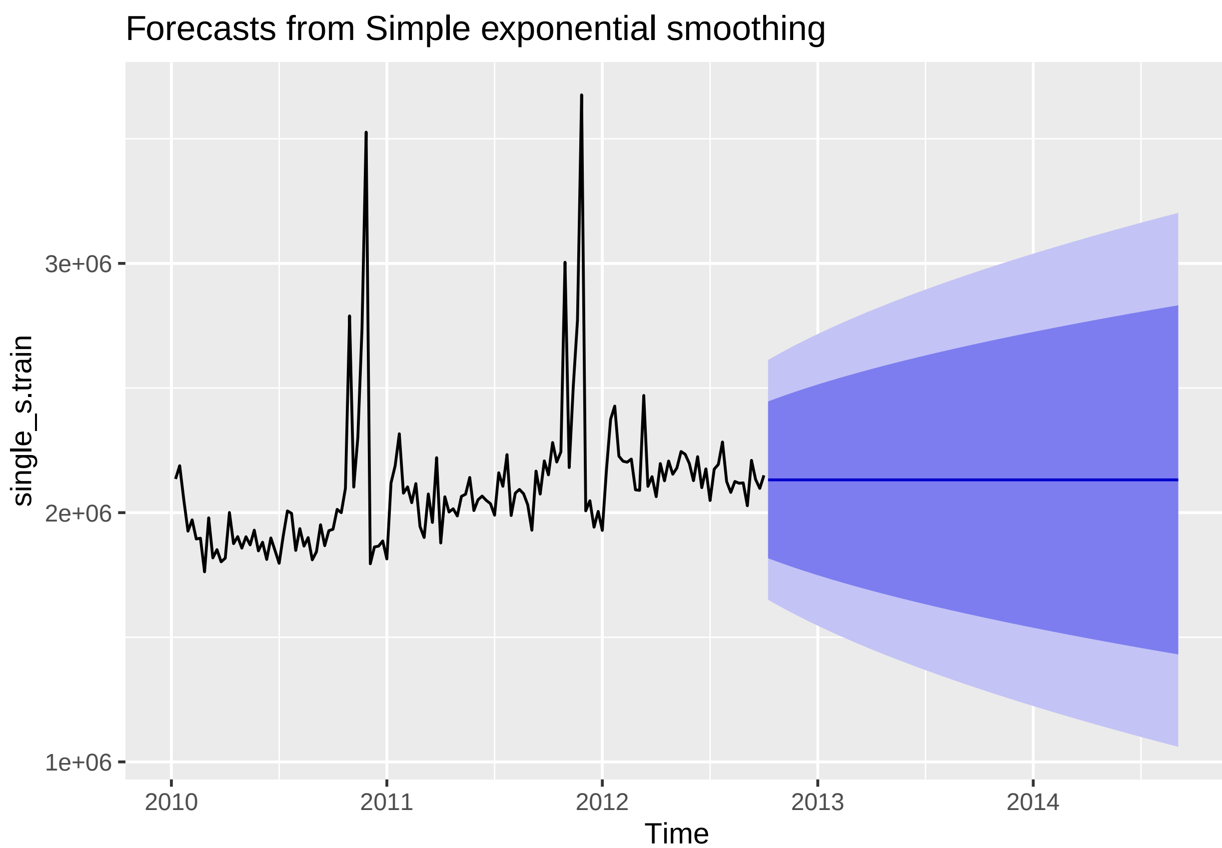
```
plot(decom)
```



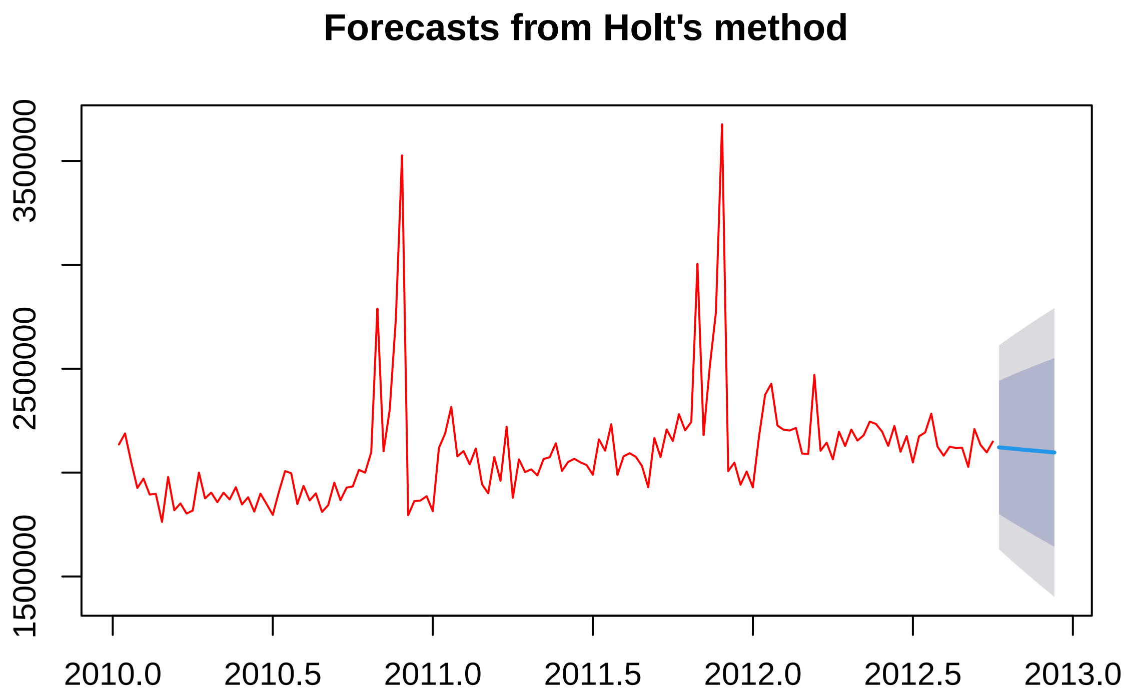

```r
###Problem 2
#single
single_s.train <- window(sales_ts, end=c(2020,40))
```

```
## Warning in window.default(x, ...): 'end' value not changed
```

```r
single_s.test <- window(sales_ts, start =c(2011,2))

single_ses <- ses(single_s.train,alpha =.2,h=100)
autoplot(forecast((single_ses),col="black"))
```



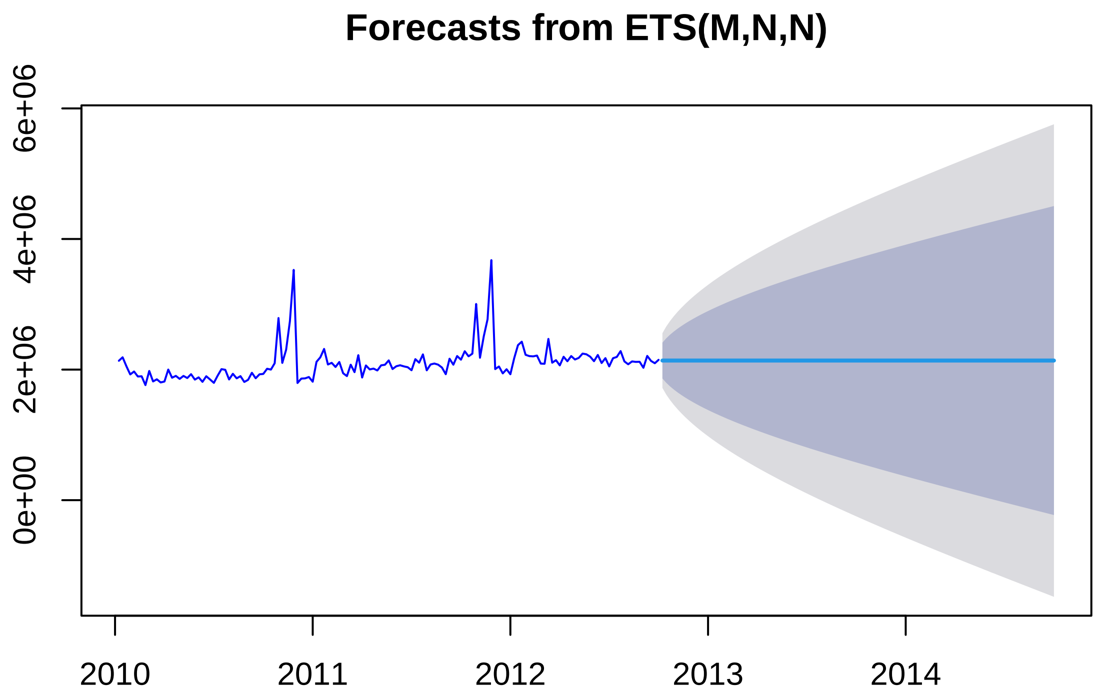

```
#Double
double_s <- holt(single_s.train)
plot(double_s,col="red")
```



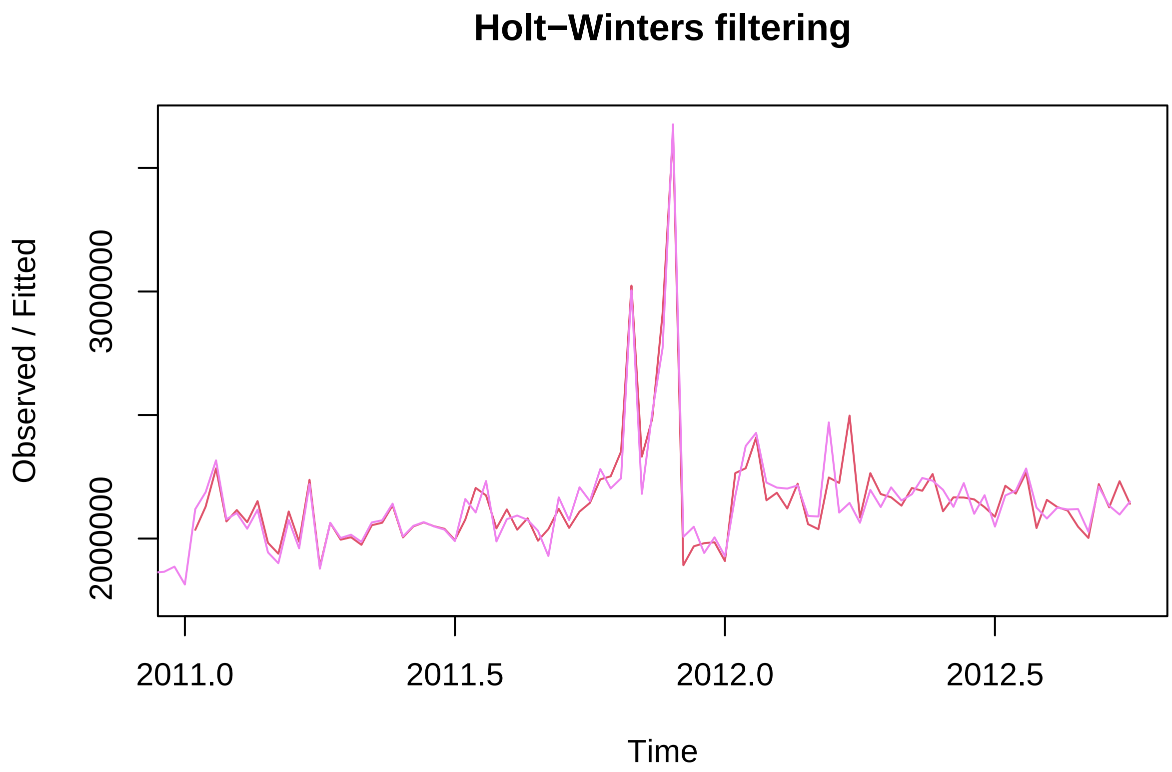

```
single_s.hw <- ets(single_s.train)
```

```
## Warning in ets(single_s.train): I can't handle data with frequency greater than
## 24. Seasonality will be ignored. Try stlf() if you need seasonal forecasts.
```

```
plot(forecast(single_s.hw),col="blue")
```




```r
#Triple
triple_s <- HoltWinters(sales_ts, alpha=0.2, beta=0.5, gamma=0.8, seasonal = "additive")
plot(triple_s,col = "violet")
```

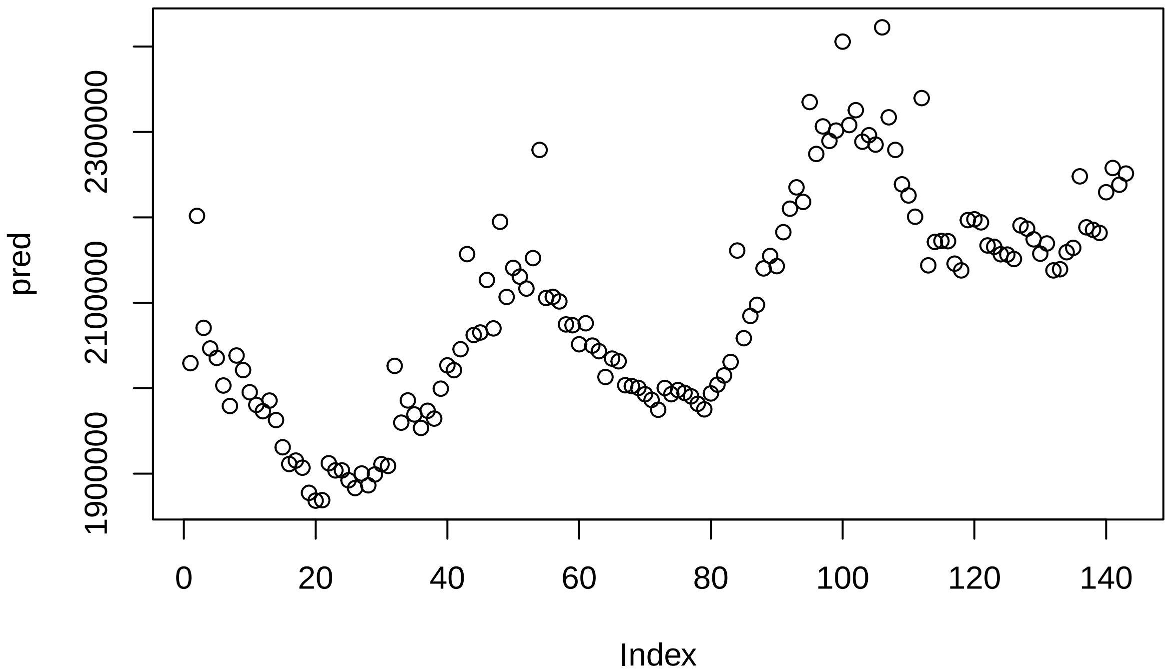



```
###Problem 3
model<- lm(sales ~ (Holiday_Flag + Unemployment + Laptop_Demand + Temperature + Fuel_Price + CPI),data =
pred = predict(model)
summary(model)
```

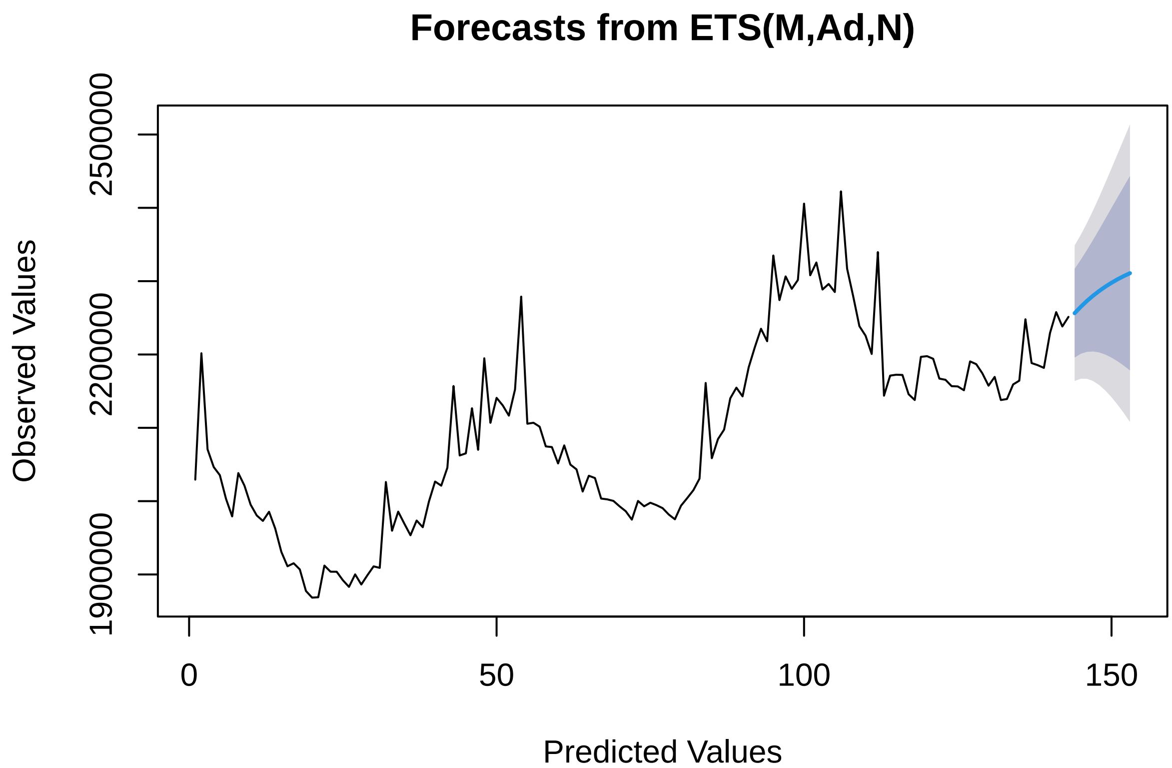
```
##
## Call:
## lm(formula = sales ~ (Holiday_Flag + Unemployment + Laptop_Demand +
##     Temperature + Fuel_Price + CPI), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -399969  -88853  -26444   41799 1456693
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -4640749    6896266  -0.673  0.50213
## Holiday_Flag    104846      80358   1.305  0.19419
## Unemployment    -25260      57459  -0.440  0.66091
## Laptop_Demand     3051       4475   0.682  0.49653
## Temperature      -4137       1412  -2.930  0.00398 **
## Fuel_Price     -106728     103421  -1.032  0.30391
## CPI              58100      52430   1.108  0.26976
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 238800 on 136 degrees of freedom
## Multiple R-squared:  0.2291, Adjusted R-squared:  0.1951
## F-statistic: 6.736 on 6 and 136 DF,  p-value: 2.892e-06
```

```
plot(pred)
```



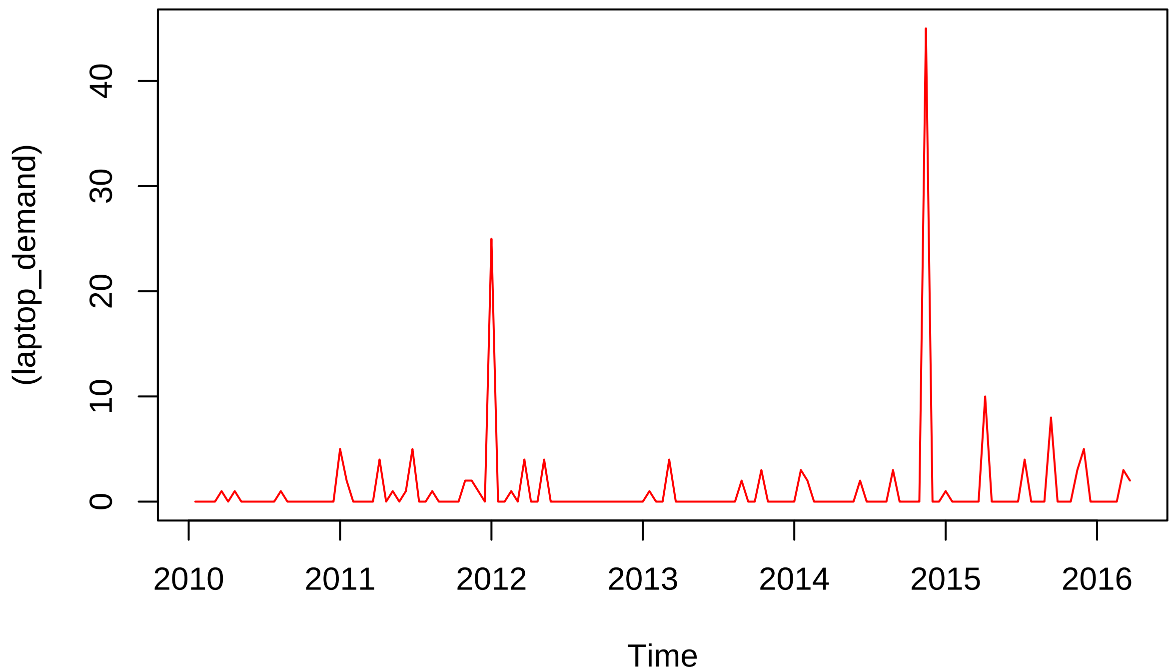

```
plot(predict(pred),df$Sales,xlab="Predicted Values",ylab="Observed Values")
abline(a = 0, b = 1, col = "blue", lwd = 2)
```



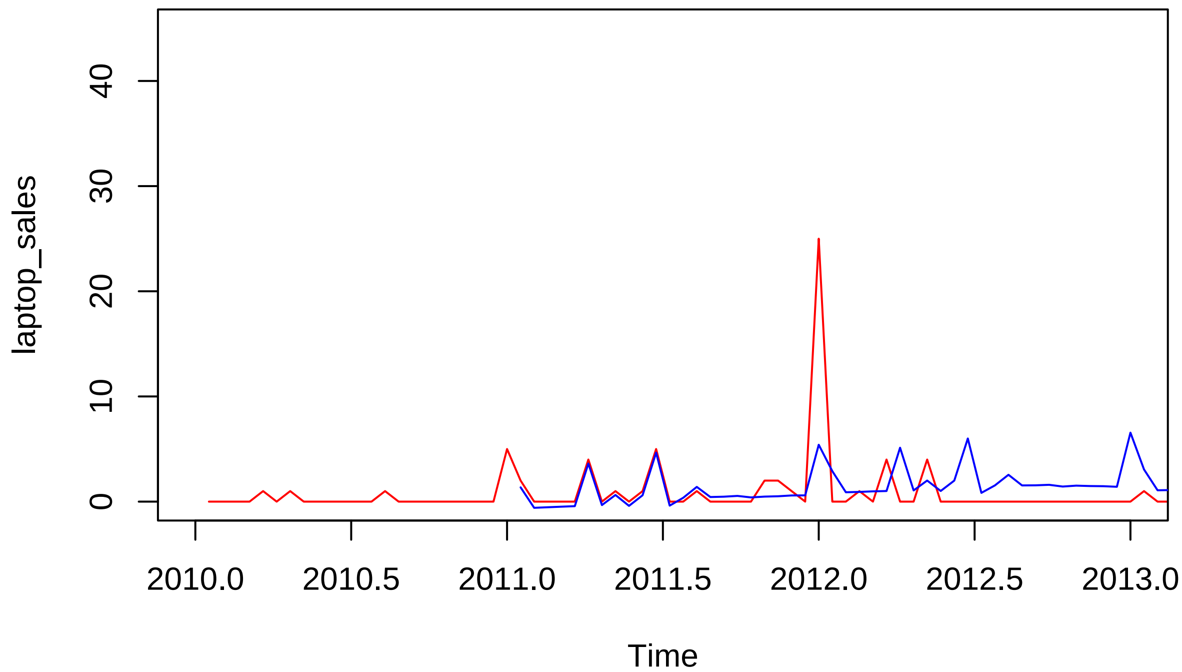

```
###Problem 4
laptop_demand <- ts(df$Laptop_Demand, frequency = 23, start=c(2010, 2, 5))
head(laptop_demand)
```

```
## Time Series:
## Start = c(2010, 2)
## End = c(2010, 7)
## Frequency = 23
## [1] 0 0 0 0 1 0
```

```
plot.ts((laptop_demand),col="red")
```




```
Holt_laptop_demand <- HoltWinters(laptop_demand)
plot(laptop_demand, ylab="laptop_sales", xlim=c(2010,2013),col="red")
lines(Holt_laptop_demand$fitted[,1], col="blue")
```




```
###Problem 5
accuracy(single_ses)
```

```
##                     ME     RMSE      MAE       MPE     MAPE      MASE      ACF1
## Training set 4200.174 243659.2 134818.6 -0.696094 6.033297 0.9359191 0.1700732
```

```
accuracy(double_s)
```

```
##                    ME   RMSE      MAE       MPE     MAPE      MASE       ACF1
## Training set 22671.9 246842 137824.2 0.2836668 6.190693 0.9567842 0.08812095
```

```
accuracy(single_s.hw)
```

```
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 1479.281 261214.4 137002.4 -0.6663062 6.222284 0.9510793
##                     ACF1
## Training set -0.2159863
```

```
accuracy(forecast(triple_s))
```

```
##                      ME     RMSE      MAE         MPE     MAPE      MASE
## Training set -1069.781 70337.69 47855.84 -0.07872944 2.203233 0.3322184
##                     ACF1
## Training set 0.02295977
```

```
accuracy(model)
```

```
##                        ME     RMSE      MAE        MPE     MAPE      MASE
## Training set 1.63082e-12 232909.2 126479.4 -0.8960906 5.601944 0.7825363
```

```
#Triple Exponential Smoothing is the best Exponential Smoothing method and regression is not better tha
```