ALY6110_Final Project Time Series_Anish Shubhi Aditi Vignesh

install necessary packages

```
options(warn=-1)
library(knitr)
opts_chunk$set(echo = TRUE, results = 'hold')
library(data.table)
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(xtable)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday,
##     week, yday, year
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

```
library(TTR)
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

```
library(forecast)
```

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

```
## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.fracdiff    fracdiff
##   residuals.fracdiff fracdiff
```

```
options(warn=0)
setwd("C:/Users/anish/Downloads/Data Management and Big Data/Final Project")
```

Input file

```
hpcFile <- "C:/Users/anish/Downloads/Data Management and Big Data/Final Project/household_power_
consumption.txt"
HHPC <- read.table(hpcFile, header= TRUE,sep=";",as.is=TRUE)
head(HHPC)
```

```
##         Date     Time Global_active_power Global_reactive_power Voltage
## 1 16/12/2006 17:24:00               4.216                 0.418 234.840
## 2 16/12/2006 17:25:00               5.360                 0.436 233.630
## 3 16/12/2006 17:26:00               5.374                 0.498 233.290
## 4 16/12/2006 17:27:00               5.388                 0.502 233.740
## 5 16/12/2006 17:28:00               3.666                 0.528 235.680
## 6 16/12/2006 17:29:00               3.520                 0.522 235.020
##   Global_intensity Sub_metering_1 Sub_metering_2 Sub_metering_3
## 1           18.400          0.000          1.000             17
## 2           23.000          0.000          1.000             16
## 3           23.000          0.000          2.000             17
## 4           23.000          0.000          1.000             17
## 5           15.800          0.000          1.000             17
## 6           15.000          0.000          2.000             17
```

data cleaning

```
HHPC$Global_active_power <- as.numeric(HHPC$Global_active_power)
```

```
## Warning: NAs introduced by coercion
```

```
HHPC$Global_reactive_power <- as.numeric(HHPC$Global_reactive_power)
```

```
## Warning: NAs introduced by coercion
```

```
HHPC$Voltage <- as.numeric(HHPC$Voltage)
```

```
## Warning: NAs introduced by coercion
```

```
HHPC$Global_intensity <- as.numeric(HHPC$Global_intensity)
```

```
## Warning: NAs introduced by coercion
```

```
HHPC$Sub_metering_1 <- as.numeric(HHPC$Sub_metering_1)
```

```
## Warning: NAs introduced by coercion
```

```
HHPC$Sub_metering_2 <- as.numeric(HHPC$Sub_metering_2)
```

```
## Warning: NAs introduced by coercion
```

initial dataframe check

```
HHPC <-cbind(HHPC, as.Date(HHPC$Date, "%d/%m/%Y"), stringsAsFactors=FALSE)
colnames(HHPC)[10] <-"DateFormat"
HHPC <- HHPC[,c(ncol(HHPC), 1:(ncol(HHPC)-1))]
head(HHPC)
str(HHPC)
```

```
##   DateFormat       Date     Time Global_active_power Global_reactive_power
## 1 2006-12-16 16/12/2006 17:24:00               4.216                 0.418
## 2 2006-12-16 16/12/2006 17:25:00               5.360                 0.436
## 3 2006-12-16 16/12/2006 17:26:00               5.374                 0.498
## 4 2006-12-16 16/12/2006 17:27:00               5.388                 0.502
## 5 2006-12-16 16/12/2006 17:28:00               3.666                 0.528
## 6 2006-12-16 16/12/2006 17:29:00               3.520                 0.522
##   Voltage Global_intensity Sub_metering_1 Sub_metering_2 Sub_metering_3
## 1  234.84             18.4              0              1             17
## 2  233.63             23.0              0              1             16
## 3  233.29             23.0              0              2             17
## 4  233.74             23.0              0              1             17
## 5  235.68             15.8              0              1             17
## 6  235.02             15.0              0              2             17
## 'data.frame':    2075259 obs. of  10 variables:
##  $ DateFormat           : Date, format: "2006-12-16" "2006-12-16" ...
##  $ Date                 : chr  "16/12/2006" "16/12/2006" "16/12/2006" "16/12/2006" ...
##  $ Time                 : chr  "17:24:00" "17:25:00" "17:26:00" "17:27:00" ...
##  $ Global_active_power  : num  4.22 5.36 5.37 5.39 3.67 ...
##  $ Global_reactive_power: num  0.418 0.436 0.498 0.502 0.528 0.522 0.52 0.52 0.51 0.51 ...
##  $ Voltage              : num  235 234 233 234 236 ...
##  $ Global_intensity     : num  18.4 23 23 23 15.8 15 15.8 15.8 15.8 15.8 ...
##  $ Sub_metering_1       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Sub_metering_2       : num  1 1 2 1 1 2 1 1 1 2 ...
##  $ Sub_metering_3       : num  17 16 17 17 17 17 17 17 17 16 ...
```

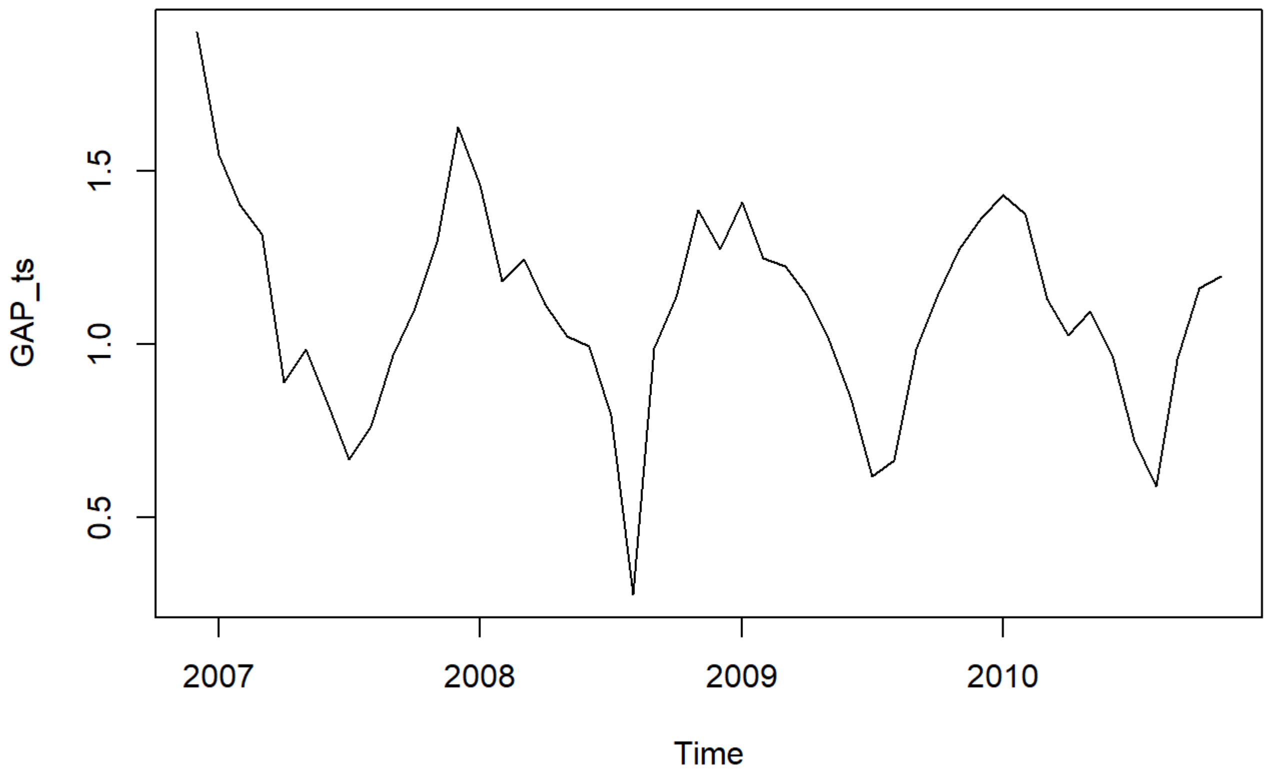
cleaning 2

```
HHPC <-cbind(HHPC, month(HHPC$DateFormat, label = TRUE, abbr = TRUE), stringsAsFactors=FALSE) #m
onth extraction
colnames(HHPC)[11] <-"Month"
HHPC <- HHPC[,c(ncol(HHPC), 1:(ncol(HHPC)-1))]
HHPC <-cbind(HHPC, year(HHPC$DateFormat), stringsAsFactors=FALSE) #year extraction
colnames(HHPC)[12] <-"Year"
HHPC <- HHPC[,c(ncol(HHPC), 1:(ncol(HHPC)-1))]

HHPCsml <- HHPC
HHPCsml$DateFormat <- NULL #eliminate unnecessary cols
HHPCsml$Date <- NULL
HHPCsml$Time <- NULL

HHPCTrain <- HHPCsml #create training set
head(HHPCTrain,5)
```

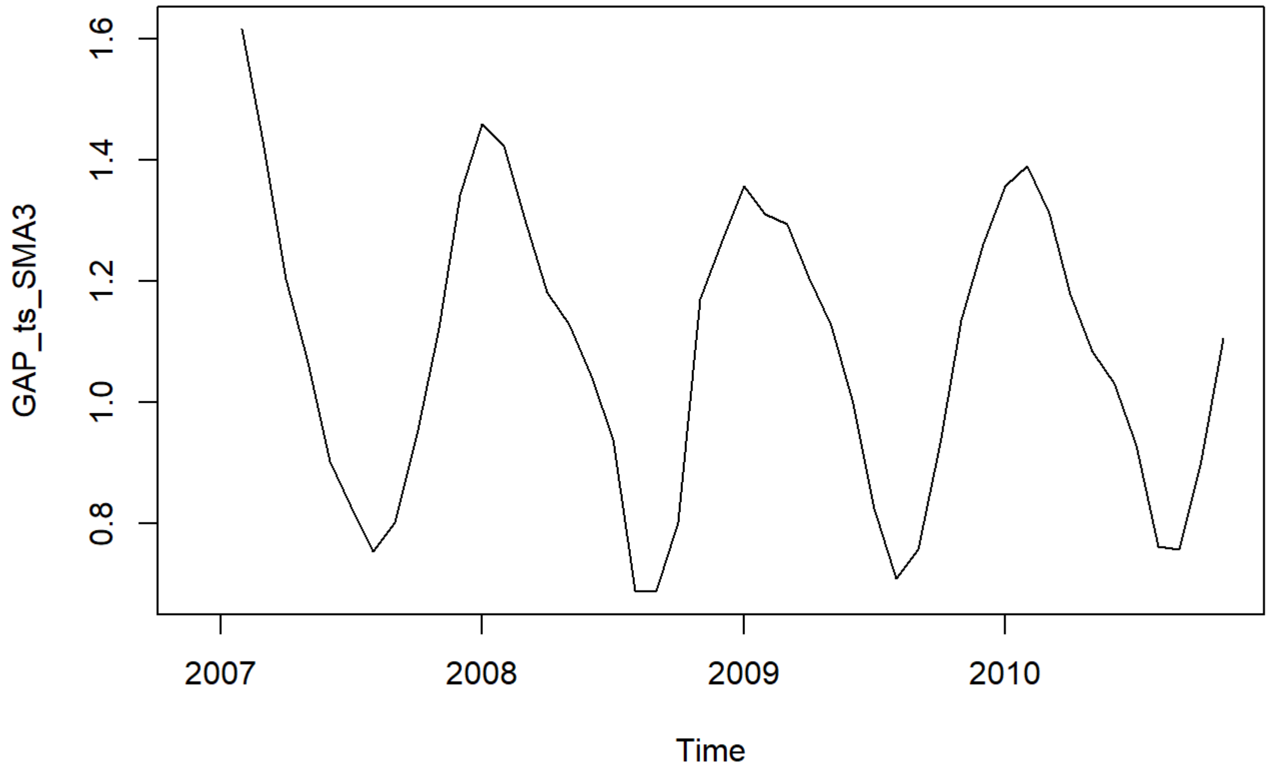
```
##   Year Month Global_active_power Global_reactive_power Voltage
## 1 2006   Dec               4.216                 0.418  234.84
## 2 2006   Dec               5.360                 0.436  233.63
## 3 2006   Dec               5.374                 0.498  233.29
## 4 2006   Dec               5.388                 0.502  233.74
## 5 2006   Dec               3.666                 0.528  235.68
##   Global_intensity Sub_metering_1 Sub_metering_2 Sub_metering_3
## 1             18.4              0              1             17
## 2             23.0              0              1             16
## 3             23.0              0              2             17
## 4             23.0              0              1             17
## 5             15.8              0              1             17
```

cleaning 3 for time series forecasting

```
HHPCsml <- group_by(HHPCsml, Year, Month) #grouping month & year
HHPCsml <- summarise(HHPCsml, MeanGAP = mean(Global_active_power, na.rm = TRUE),
                     MeanGRP = mean(Global_reactive_power, na.rm = TRUE),
                     MeanVolt = mean(Voltage, na.rm = TRUE),
                     MeanGI = mean(Global_intensity, na.rm = TRUE),
                     MeanSubm1 = mean(Sub_metering_1, na.rm = TRUE),
                     MeanSubm2 = mean(Sub_metering_2, na.rm = TRUE),
                     MeanSubm3 = mean(Sub_metering_3, na.rm = TRUE))
HHPCsml <- arrange(HHPCsml, Year, Month) #reducing data using mean
GAP_vector <- vector( mode ="numeric") #mean global active power calculation
GAP_vector <- HHPCsml$MeanGAP
GAP_ts <- ts(GAP_vector, frequency = 12, start = c(2006,12), end = c(2010,11))
plot.ts(GAP_ts)
```

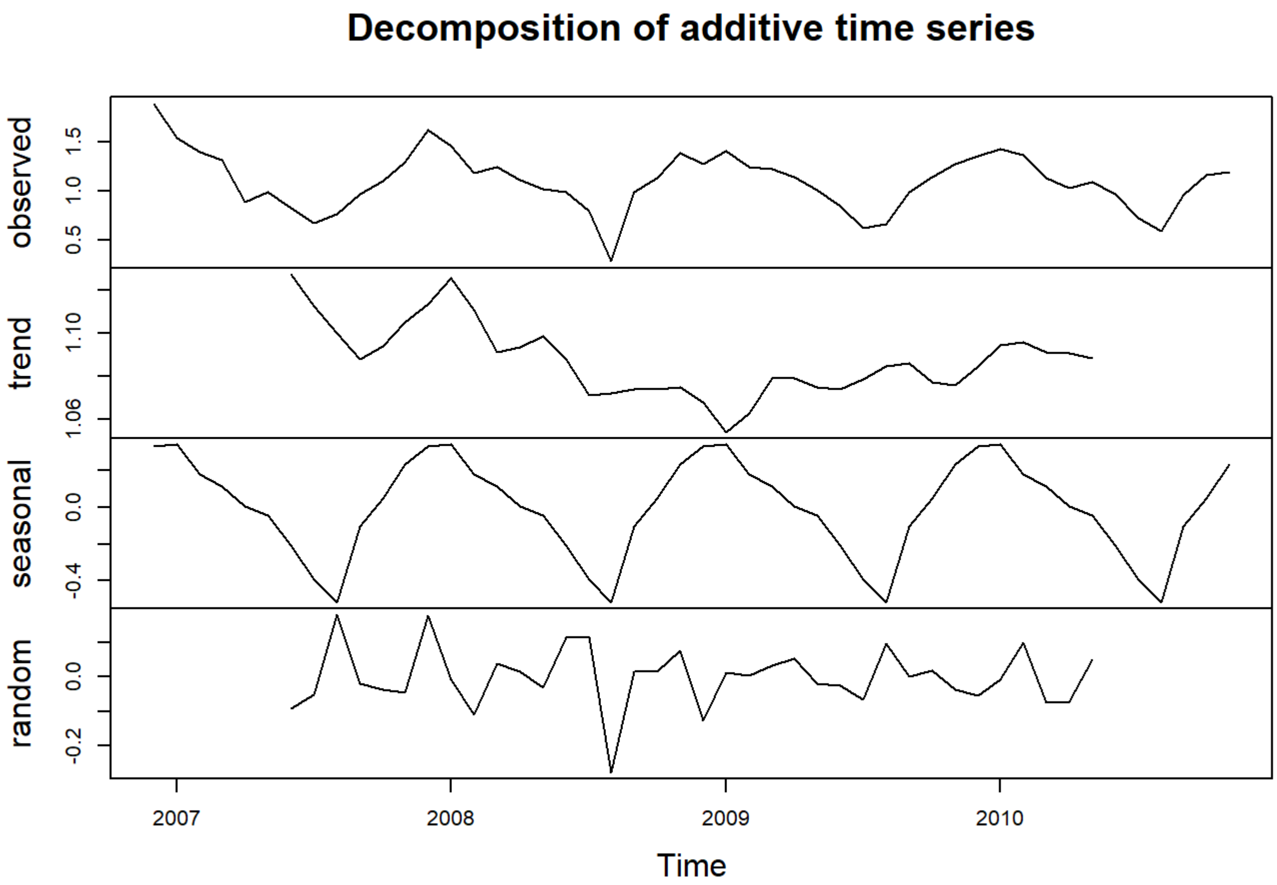


```
GAP_ts_SMA3 <- SMA(GAP_ts, n=3) #smoothing curve using moving average parameter
plot.ts(GAP_ts_SMA3)
```

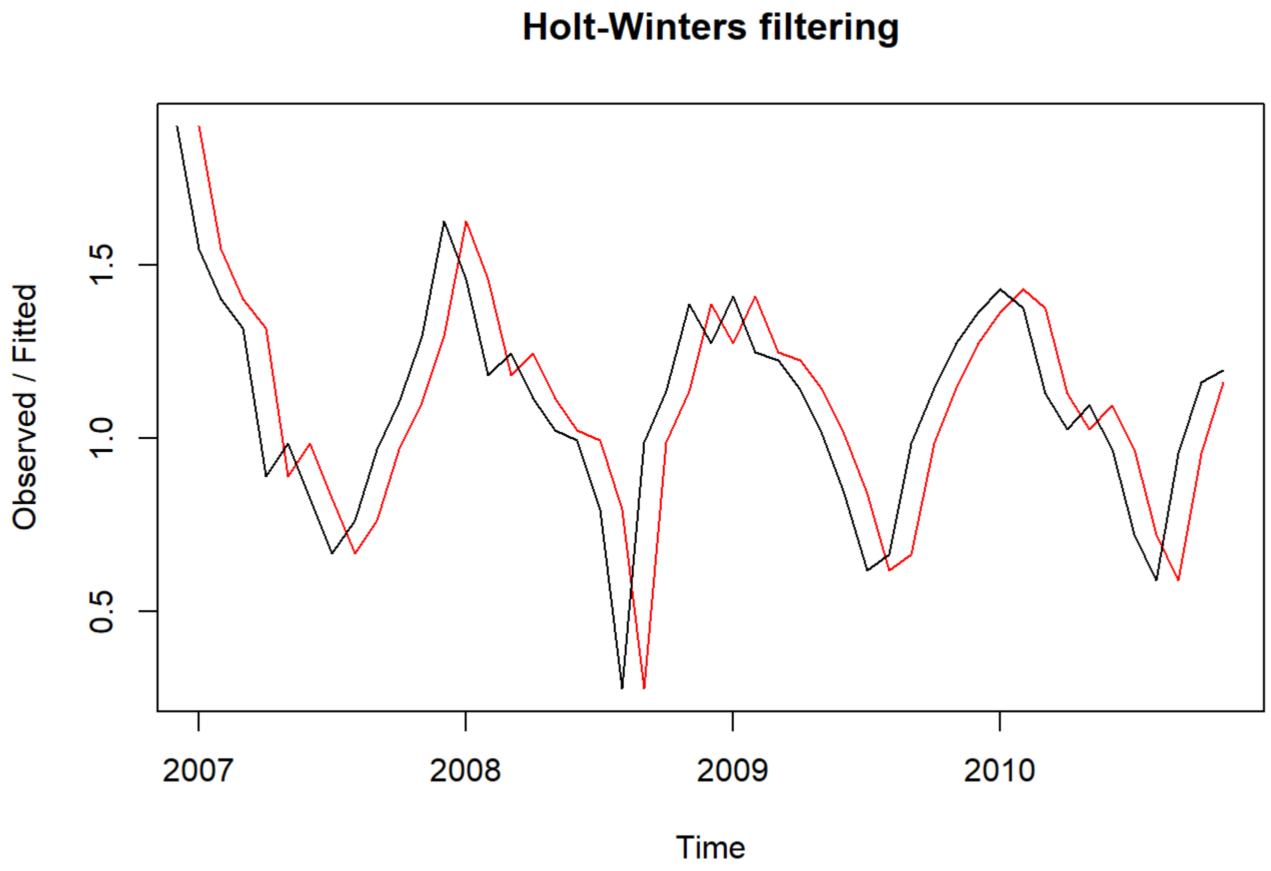


time series analysis and forecasting

```
GAP_ts_components <- decompose(GAP_ts)
plot(GAP_ts_components)
```



```
GAPforecasts <- HoltWinters(GAP_ts, beta=FALSE, gamma=FALSE)
GAPforecasts
plot(GAPforecasts)
```

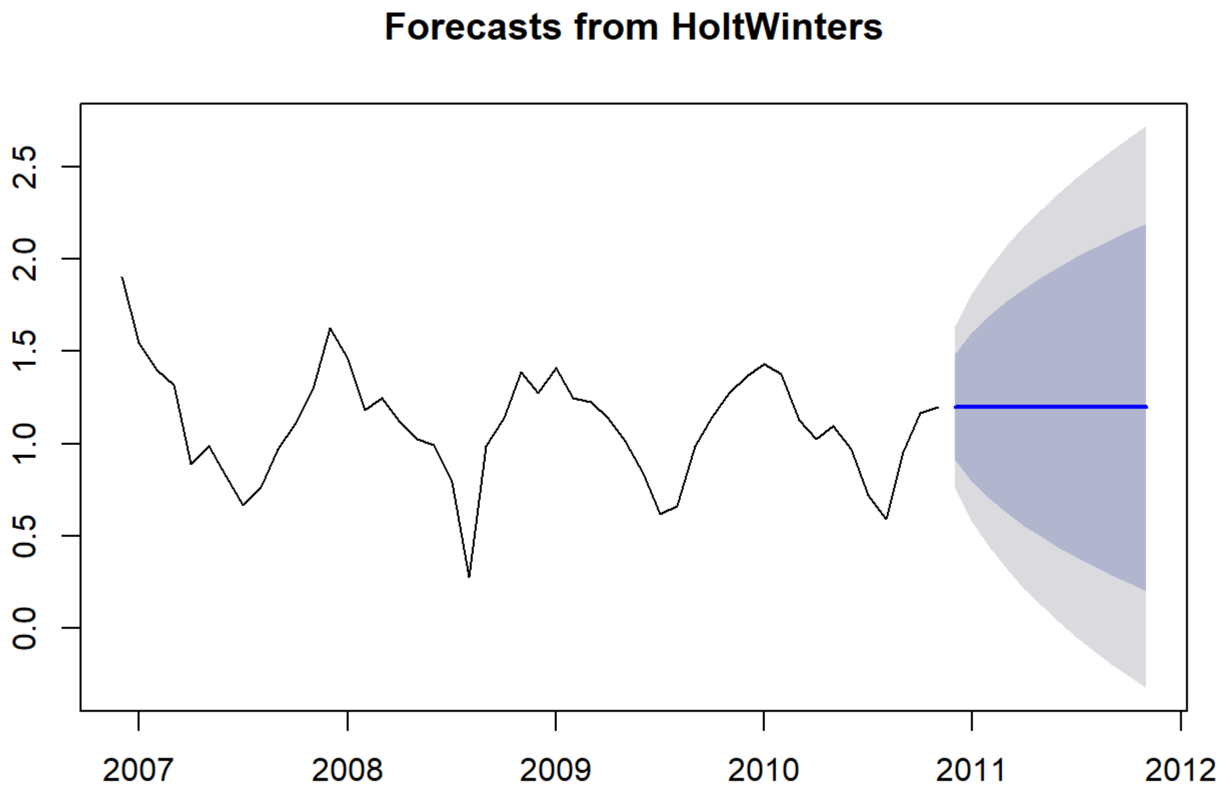


```
GAPforecasts$SSE #sum of squared errors
```

```
## Holt-Winters exponential smoothing without trend and without seasonal component.
## 
## Call:
## HoltWinters(x = GAP_ts, beta = FALSE, gamma = FALSE)
## 
## Smoothing parameters:
##  alpha: 0.9999472
##  beta : FALSE
##  gamma: FALSE
## 
## Coefficients:
##       [,1]
## a 1.196853
## [1] 2.318598
```

confidence interval forecast

```
GAPforecasts2 <- forecast:::forecast.HoltWinters(GAPforecasts, h=12)
GAPforecasts2
plot(GAPforecasts2)
```



```
##          Point Forecast     Lo 80    Hi 80       Lo 95    Hi 95
## Dec 2010       1.196853 0.9097887 1.483917  0.75782624 1.635879
## Jan 2011       1.196853 0.7908937 1.602812  0.57599192 1.817713
## Feb 2011       1.196853 0.6996608 1.694045  0.43646334 1.957242
## Mar 2011       1.196853 0.6227475 1.770958  0.31883456 2.074871
## Apr 2011       1.196853 0.5549853 1.838720  0.21520116 2.178504
## May 2011       1.196853 0.4937234 1.899982  0.12150921 2.272196
## Jun 2011       1.196853 0.4373872 1.956318  0.03535044 2.358355
## Jul 2011       1.196853 0.3849507 2.008755 -0.04484427 2.438550
## Aug 2011       1.196853 0.3357012 2.058004 -0.12016486 2.513870
## Sep 2011       1.196853 0.2891199 2.104586 -0.19140488 2.585110
## Oct 2011       1.196853 0.2448149 2.148890 -0.25916344 2.652869
## Nov 2011       1.196853 0.2024821 2.191223 -0.32390595 2.717611
```

CONCLUSION : The forecast.HoltWinters() function provided the forecast for a year, a 80% prediction interval for the forecast, and a 95% prediction interval for the forecast. For example, the forecasted mean global active power for November 2011 is about 1.1968, with a 95% prediction interval of (-0.323, 2.7176). with a sum of squared error = 2.32