ChulwalarCaseStudy

Rajeev, Armand, Najeeb, Marvin

July 21, 2016

Introduction

The purpose of this analysis is to test a variety of forecasting methods for predicting the exports of island of Chuwalhar, part of the island group Urbano. The data available are the past exports of flowers ranging from the years 2008 to 2013, planned exports from 2008 to 2013 for spices and teas, and a number of indicators likely related to exports (satisfaction indexes, export prices, temperature, etc). In addition, there are also a number of national holidays in March, April, and December which influence exports.

Set Up

```
library(fpp) # for time series forecasting and analysis
```

```
## Loading required package: forecast
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

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

```
## Loading required package: timeDate
```

```
## This is forecast 7.1
```

```
## Loading required package: fma
```

```
## Loading required package: tseries
```

```
## Loading required package: expsmooth
```

```
## Loading required package: lmtest
```

```
library(forecast) # for some other forecasting models

## The working directory will need changing depending on the computer and file system
!!
#setwd("/Users/rajeevkumar/Documents/R-Studio/ChulwalarCaseStudy")
```

Import the data

```
# As Is Data,Plan Data and Indicator Data files are imorted into R
ImportedAsIsData <- read.csv(file="./Data/ImportedAsIsDataChulwalar.csv", header = F,
sep=";", fill = T)
ImportedPlanData <- read.csv(file="./Data/ImportedPlanDataChulwalar.csv", header = F,
sep=";", fill = T)
ImportedIndicators <- read.csv(file="./Data/ImportedIndicatorsChulwalar.csv", header
= F, sep=";", fill = T)
str(ImportedAsIsData)
```

```
## 'data.frame':    98 obs. of  8 variables:
##  $ V1: Factor w/ 22 levels "","Apr","Aug",..: 19 9 8 13 2 14 11 10 3 18 ...
##  $ V2: int  2008 2313221 1950131 2346635 2039787 1756964 1458302 1679637 1639670 2
882886 ...
##  $ V3: int  2009 2610573 2371327 2743786 2125308 1850073 1836222 1797311 1851968 3
271171 ...
##  $ V4: int  2010 2760688 2918333 3227041 1613888 2550157 2317645 1474144 2148521 3
898571 ...
##  $ V5: int  2011 3112861 2926663 3294784 2577079 2774068 2378227 2222900 2991787 4
151531 ...
##  $ V6: int  2012 3093088 3679308 3433364 2714899 3011767 2726028 2483834 3055655 4
200796 ...
##  $ V7: int  2013 4119526 3535744 3560974 3760065 2959933 2787898 2828744 3084113 5
107775 ...
##  $ V8: int  2014 4308161 4155378 3924332 3659121 3898758 3313891 3595106 3502426 5
619059 ...
```

```
str(ImportedPlanData)
```

```
## 'data.frame':    97 obs. of  8 variables:
##  $ V1: Factor w/ 20 levels "","Apr","Aug",..: 19 7 6 11 2 12 9 8 3 15 ...
##  $ V2: int  2008 2243103 2162705 2720911 2011182 1877757 1819924 1682196 1893171 3
325711 ...
##  $ V3: int  2009 2547980 2247049 2731156 2020158 2098038 1927995 1783692 1907705 3
124040 ...
##  $ V4: int  2010 2965885 2751170 2906493 2383358 2246893 1992851 2023434 2244997 3
257717 ...
##  $ V5: int  2011 3113110 2883766 2957893 2601648 2370949 2339881 2105328 2341623 4
086297 ...
##  $ V6: int  2012 3895396 3588151 3787240 3036434 2907891 2707822 2619486 3784557 4
987460 ...
##  $ V7: int  2013 3580325 3863212 3606083 3213575 3139128 2998610 2785453 3083654 5
143757 ...
##  $ V8: int  2014 4474000 4185565 4278119 3985542 3605973 3515173 3269444 3656112 5
637391 ...
```

```
str(ImportedIndicators)
```

```
## 'data.frame':    195 obs. of  8 variables:
##  $ V1: Factor w/ 28 levels "","Apr","Aug",..: 7 16 12 20 2 19 18 17 3 26 ...
##  $ V2: num  2008 97.4 97.8 98.3 98.1 ...
##  $ V3: num  2009 98.3 98.9 98.7 98.8 ...
##  $ V4: num  2010 99 99.4 99.9 100 ...
##  $ V5: num  2011 101 101 102 102 ...
##  $ V6: num  2012 103 104 104 104 ...
##  $ V7: num  2013 104 105 106 105 ...
##  $ V8: num  2014 NA NA NA NA ...
```

Transform data into time series

In order to be able to work with the partial data sets later, these need to be split into individual vectors and converted into times series.

```
## I would not have this code in the final draft of the case study. These code need t
o be in an appendix, even if they are put into a makefile.
# Put each dataset into vectors for year betweeb 2008 to 2013 (2 to 7) leaving 2014
TotalAsIsVector <- c(ImportedAsIsData [2:13,2],ImportedAsIsData [2:13,3],ImportedAsIs
Data [2:13,4],ImportedAsIsData [2:13,5],ImportedAsIsData [2:13,6],ImportedAsIsData [2
:13,7])
EfakAsIsVector <- c(ImportedAsIsData [16:27,2],ImportedAsIsData [16:27,3],ImportedAsI
sData [16:27,4],ImportedAsIsData [16:27,5],ImportedAsIsData [16:27,6],ImportedAsIsDat
a [16:27,7])
WugeAsIsVector <- c(ImportedAsIsData [30:41,2],ImportedAsIsData [30:41,3],ImportedAsI
sData [30:41,4],ImportedAsIsData [30:41,5],ImportedAsIsData [30:41,6],ImportedAsIsDat
a [30:41,7])
TotalEtelAsIsVector <- c(ImportedAsIsData [44:55,2],ImportedAsIsData [44:55,3],Import
```

```
edAsIsData [44:55,4],ImportedAsIsData [44:55,5],ImportedAsIsData [44:55,6],ImportedAs
IsData [44:55,7])
BlueEtelAsIsVector <- c(ImportedAsIsData [58:69,2],ImportedAsIsData [58:69,3],Importe
dAsIsData [58:69,4],ImportedAsIsData [58:69,5],ImportedAsIsData [58:69,6],ImportedAsI
sData [58:69,7])
RedEtelAsIsVector <- c(ImportedAsIsData [72:83,2],ImportedAsIsData [72:83,3],Imported
AsIsData [72:83,4],ImportedAsIsData [72:83,5],ImportedAsIsData [72:83,6],ImportedAsIs
Data [72:83,7])
YearAsIsVector <- c(ImportedAsIsData [86,2],ImportedAsIsData [86,3],ImportedAsIsData
[86,4],ImportedAsIsData [86,5],ImportedAsIsData [86,6],ImportedAsIsData [86,7])
TotalAsIsVector_2014 <- c(ImportedAsIsData[2:13,8])

PlanVector <- c(ImportedPlanData[2:13,2],ImportedPlanData[2:13,3],ImportedPlanData[2:
13,4],ImportedPlanData[2:13,5],ImportedPlanData[2:13,6],ImportedPlanData[2:13,7])
EfakPlanVector <- c(ImportedPlanData[16:27,2],ImportedPlanData[16:27,3],ImportedPlanD
ata[16:27,4],ImportedPlanData[16:27,5],ImportedPlanData[16:27,6],ImportedPlanData[16:
27,7])
WugePlanVector <- c(ImportedPlanData[30:41,2],ImportedPlanData[30:41,3],ImportedPlanD
ata[30:41,4],ImportedPlanData[30:41,5],ImportedPlanData[30:41,6],ImportedPlanData[30:
41,7])
TotalEtelPlanVector <- c(ImportedPlanData[44:55,2],ImportedPlanData[44:55,3],Imported
PlanData[44:55,4],ImportedPlanData[44:55,5],ImportedPlanData[44:55,6],ImportedPlanDat
a[44:55,7])
BlueEtelPlanVector <- c(ImportedPlanData[58:69,2],ImportedPlanData[58:69,3],ImportedP
lanData[58:69,4],ImportedPlanData[58:69,5],ImportedPlanData[58:69,6],ImportedPlanData
[58:69,7])
RedEtelPlanVector <- c(ImportedPlanData[72:83,2],ImportedPlanData[72:83,3],ImportedPl
anData[72:83,4],ImportedPlanData[72:83,5],ImportedPlanData[72:83,6],ImportedPlanData[
72:83,7])
YearPlanVector <- c(ImportedPlanData[86,2],ImportedPlanData[86,3],ImportedPlanData[86
,4],ImportedPlanData[86,5],ImportedPlanData[86,6],ImportedPlanData[86,7])
PlanVector_2014 <- c(ImportedPlanData[2:13,8])

# The data is saved as a vector and needs to be converted into a time series
TotalAsIs<- ts(TotalAsIsVector , start=c(2008,1), end=c(2013,12), frequency=12)
EfakAsIs <- ts(EfakAsIsVector , start=c(2008,1), end=c(2013,12), frequency=12)
WugeAsIs <- ts(WugeAsIsVector, start=c(2008,1), end=c(2013,12), frequency=12)
TotalEtelAsIs<- ts(TotalEtelAsIsVector, start=c(2008,1), end=c(2013,12), frequency=12
)
BlueEtelAsIs <- ts(BlueEtelAsIsVector, start=c(2008,1), end=c(2013,12), frequency=12)
RedEtelAsIs <- ts(RedEtelAsIsVector, start=c(2008,1), end=c(2013,12), frequency=12)
YearAsIs <- ts(YearAsIsVector, start=c(2008,1), end=c(2013,12), frequency=12)
TotalAsIs_2014 <- ts(TotalAsIsVector_2014, start=c(2014,1), end=c(2014,12), frequency
=12)

TotalPlan <- ts(PlanVector , start=c(2008,1), end=c(2013,12), frequency=12)
EfakPlan <- ts(EfakPlanVector, start=c(2008,1), end=c(2013,12), frequency=12)
WugePlan <- ts(WugePlanVector, start=c(2008,1), end=c(2013,12), frequency=12)
TotalEtelPlan <- ts(TotalEtelPlanVector, start=c(2008,1), end=c(2013,12), frequency=1
2)
```

```
BlueEtelPlan <- ts(BlueEtelPlanVector, start=c(2008,1), end=c(2013,12), frequency=12)
RedEtelPlan <- ts(RedEtelPlanVector, start=c(2008,1), end=c(2013,12), frequency=12)
YearPlan <- ts(YearPlanVector, start=c(2008,1), end=c(2013,12), frequency=12)
TotalPlan_2014 <- ts(PlanVector_2014, start=c(2014,1), end=c(2014,12), frequency=12)

# Call up the time series to check everything has worked.

str(TotalAsIs)
```

```
##  Time-Series [1:72] from 2008 to 2014: 2313221 1950131 2346635 2039787 1756964 145
8302 1679637 1639670 2882886 2959716 ...
```

```
str(EfakAsIs)
```

```
##  Time-Series [1:72] from 2008 to 2014: 416589 472565 466539 370774 457741 384817 4
64502 389013 508370 495598 ...
```

```
str(WugeAsIs)
```

```
##  Time-Series [1:72] from 2008 to 2014: 414571 344579 429907 379606 305697 314582 3
46800 323618 578252 510031 ...
```

```
str(TotalEtelAsIs)
```

```
##  Time-Series [1:72] from 2008 to 2014: 1279668 1053325 1367520 1090725 873568 6444
79 772658 806741 1715265 1795751 ...
```

```
str(BlueEtelAsIs)
```

```
##  Time-Series [1:72] from 2008 to 2014: 425892 316631 353512 278711 212940 187849 2
06285 195810 448733 403327 ...
```

```
str(RedEtelAsIs)
```

```
##  Time-Series [1:72] from 2008 to 2014: 853776 736694 1014008 812014 660628 456630
566373 610931 1266532 1392424 ...
```

```
str(YearAsIs)
```

```
##  Time-Series [1:72] from 2008 to 2014: 26280011 29609916 32726772 37215503 4062967
6 45408410 26280011 29609916 32726772 37215503 ...
```

```
str(TotalAsIs_2014)
```

```
##  Time-Series [1:12] from 2014 to 2015: 4308161 4155378 3924332 3659121 3898758 331
3891 3595106 3502426 5619059 5274287 ...
```

```
str(TotalPlan)
```

```
##  Time-Series [1:72] from 2008 to 2014: 2243103 2162705 2720911 2011182 1877757 181
9924 1682196 1893171 3325711 2662148 ...
```

```
str(EfakPlan)
```

```
##  Time-Series [1:72] from 2008 to 2014: 492421 444995 665274 444369 487668 445242 4
43318 501222 546249 553286 ...
```

```
str(WugePlan)
```

```
##  Time-Series [1:72] from 2008 to 2014: 424190 388688 457796 363828 364246 358439 3
21255 370153 645618 470648 ...
```

```
str(TotalEtelPlan)
```

```
##  Time-Series [1:72] from 2008 to 2014: 1263613 1231125 1489621 1051346 933392 9320
47 855520 923070 2080877 1575579 ...
```

```
str(BlueEtelPlan)
```

```
##  Time-Series [1:72] from 2008 to 2014: 449227 373663 415732 331337 290942 287603 2
45390 284540 554127 467772 ...
```

```
str(RedEtelPlan)
```

```
##  Time-Series [1:72] from 2008 to 2014: 814386 857462 1073889 720009 642450 644444
610130 638530 1526750 1107807 ...
```

```
str(YearPlan)
```

```
##  Time-Series [1:72] from 2008 to 2014: 27883407 29387100 32780247 35224132 4394706
3 44152007 27883407 29387100 32780247 35224132 ...
```

```
str(TotalPlan_2014)
```

```
##  Time-Series [1:12] from 2014 to 2015: 4474000 4185565 4278119 3985542 3605973 351
5173 3269444 3656112 5637391 5157781 ...
```

Basic data analysis

Correlation between As Is and Plan Data

Test the correlation between As Is and Plan data in order to test how exact the planning is. Correlation is a measure of linear relationship between two variables.

```
cor(TotalAsIs, TotalPlan )
```

```
## [1] 0.9183402
```

```
cor(EfakAsIs , EfakPlan)
```

```
## [1] 0.9055081
```

```
cor(WugeAsIs, WugePlan)
```

```
## [1] 0.8788474
```

```
cor(TotalEtelAsIs, TotalEtelPlan)
```

```
## [1] 0.9159505
```

```
cor(BlueEtelAsIs , BlueEtelPlan)
```

```
## [1] 0.8044146
```

```
cor(RedEtelAsIs , RedEtelPlan)
```

```
## [1] 0.9106702
```

```
cor(YearAsIs, YearPlan)
```

```
## [1] 0.9627401
```

The results show a very high planning accuracy.

```
#Fit a Linear Model between Total As Is and Total Plan
TotalAsIs_lm <- lm(TotalAsIs ~ TotalPlan , data = TotalAsIs)
summary(TotalAsIs_lm)
```

```
##
## Call:
## lm(formula = TotalAsIs ~ TotalPlan, data = TotalAsIs)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -770214 -196776   26017  182579  672705
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.959e+04  1.521e+05   0.589    0.558
## TotalPlan   9.627e-01  4.959e-02  19.413   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 332600 on 70 degrees of freedom
## Multiple R-squared:  0.8433, Adjusted R-squared:  0.8411
## F-statistic: 376.9 on 1 and 70 DF,  p-value: < 2.2e-16
```

```
#As R-squared = 0.8433 and p value (p-value: < 2.2e-16) of slope is very low. A linea
r model is a good fit.

#Fit a Linear model considering trend and seasonality components of time series.
TotalAsIs_tslm <- tslm(TotalAsIs ~ TotalPlan )
summary(TotalAsIs_tslm)
```

```
##
## Call:
## tslm(formula = TotalAsIs ~ TotalPlan)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -770214 -196776   26017  182579  672705
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 8.959e+04  1.521e+05   0.589    0.558
## TotalPlan   9.627e-01  4.959e-02  19.413   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 332600 on 70 degrees of freedom
## Multiple R-squared:  0.8433, Adjusted R-squared:  0.8411
## F-statistic: 376.9 on 1 and 70 DF,  p-value: < 2.2e-16
```

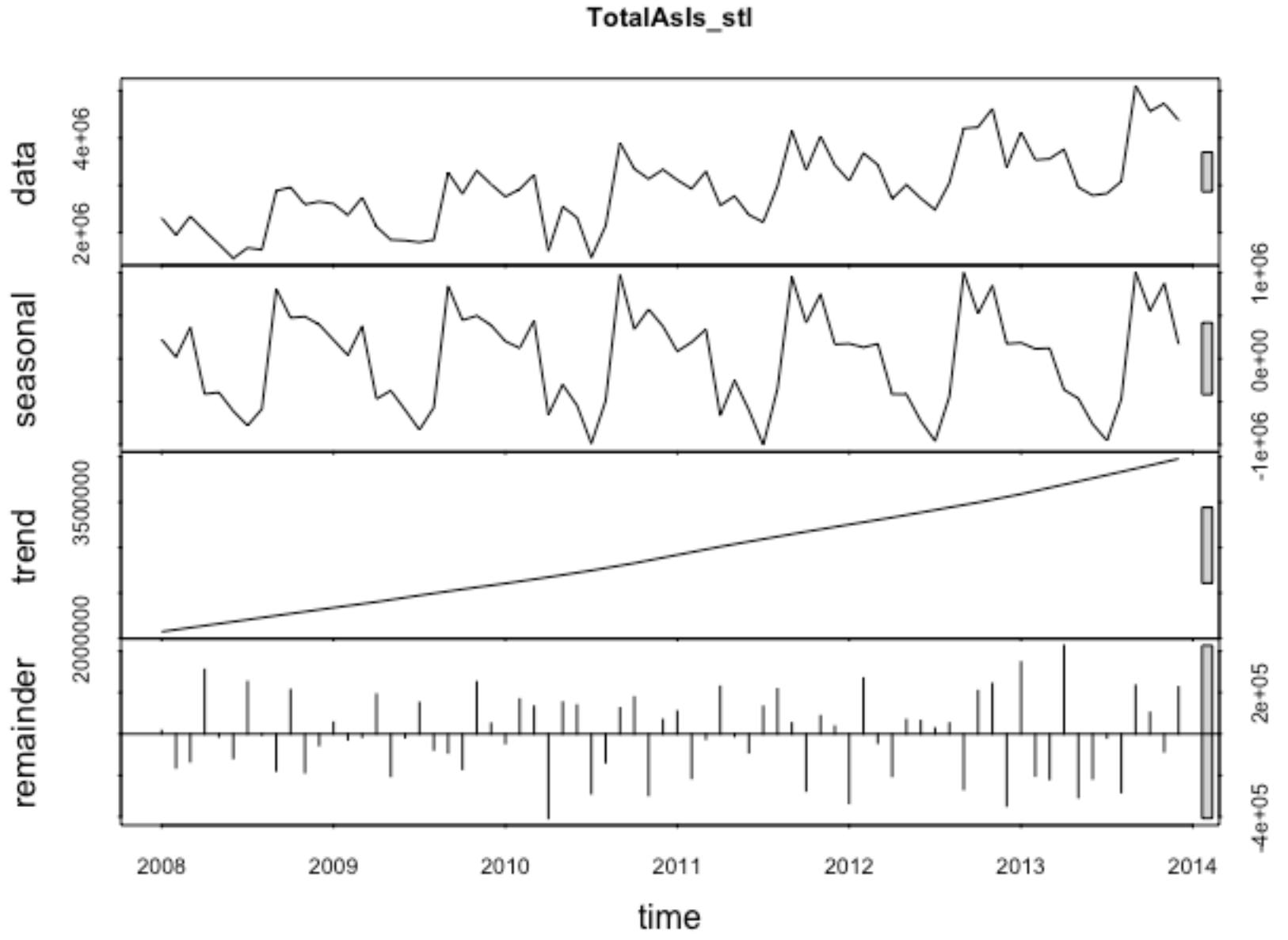
Use STL function for decomposition

The time series can be analysed using the stl function in order to seperate the trend, seasonality and remainder (remaining coincidential) components from one another.

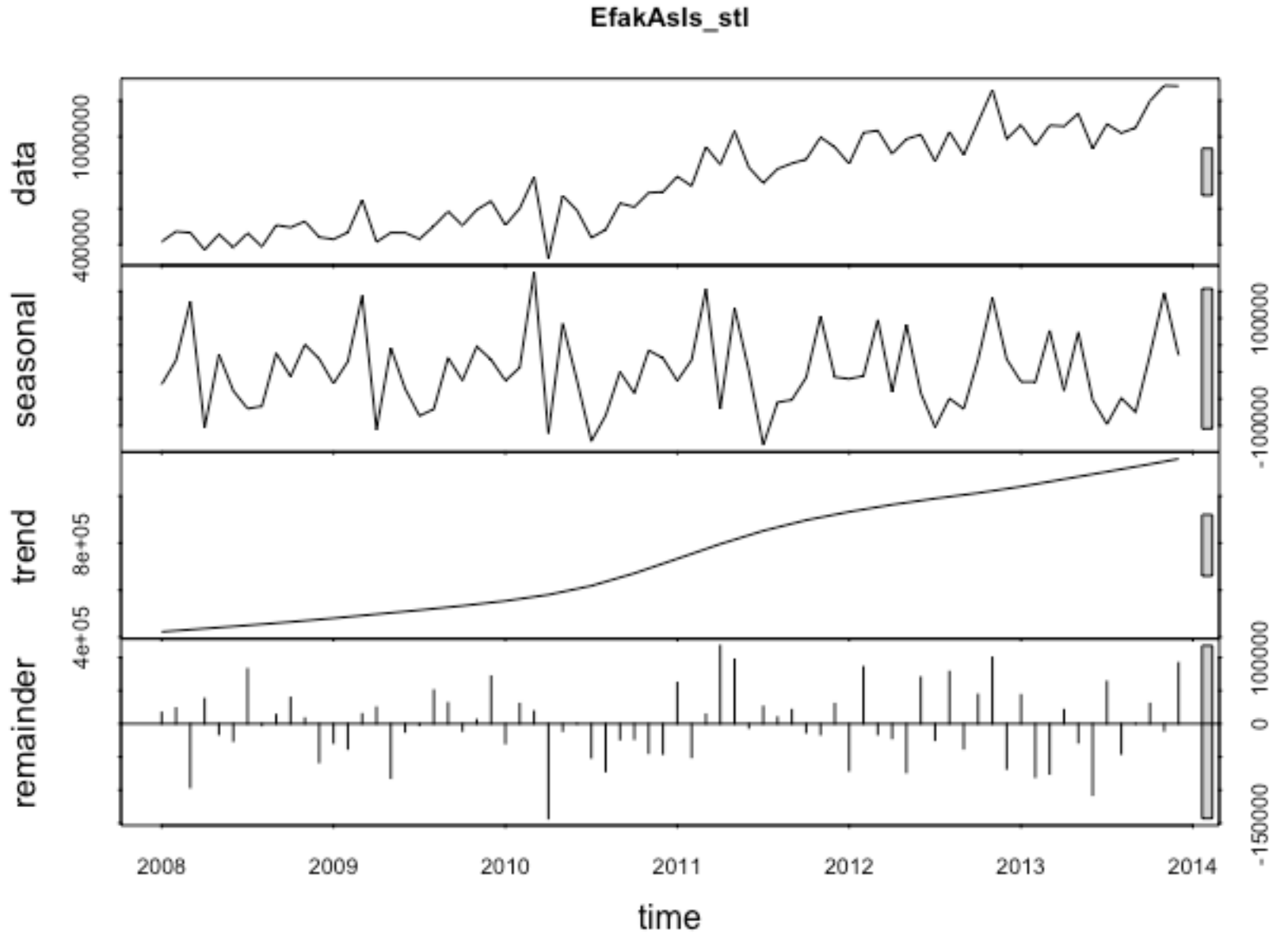
```
TotalAsIs_stl <- stl(TotalAsIs, s.window=5)
EfakAsIs_stl <- stl(EfakAsIs , s.window=5)
WugeAsIs_stl <- stl(WugeAsIs, s.window=5)
TotalEtelAsIs_stl <- stl(TotalEtelAsIs, s.window=5)
BlueEtelAsIs_stl <- stl(BlueEtelAsIs , s.window=5)
RedEtelAsIs_stl <- stl(RedEtelAsIs , s.window=5)
```

Thus the individual time series can be shown graphically and tabularly. The trend of the total exports is almost linear. A relatively uniform seaonality can be seen.

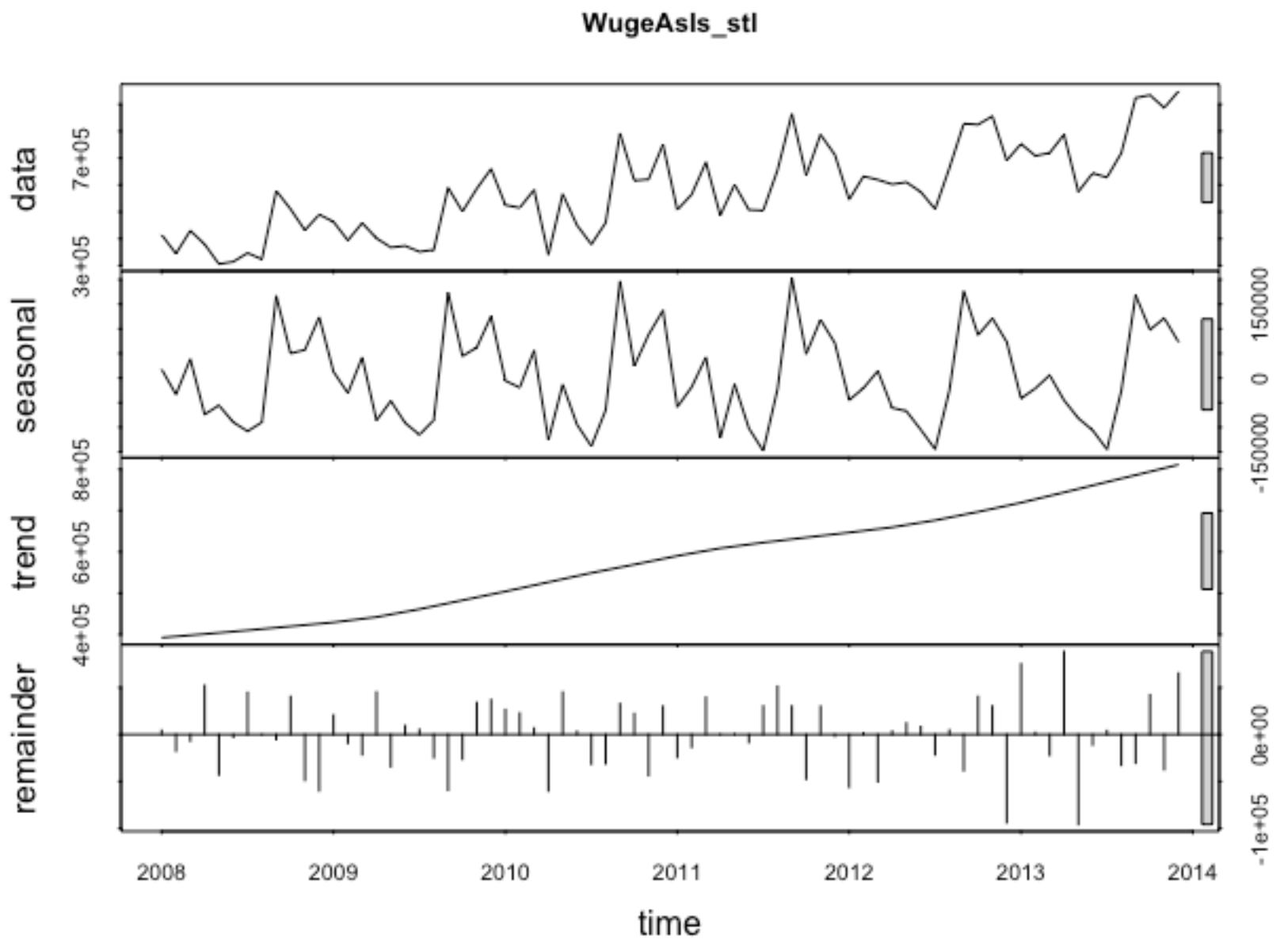
```
par(mfrow=c(3,2))
plot(TotalAsIs_stl, col="black", main="TotalAsIs_stl")
```



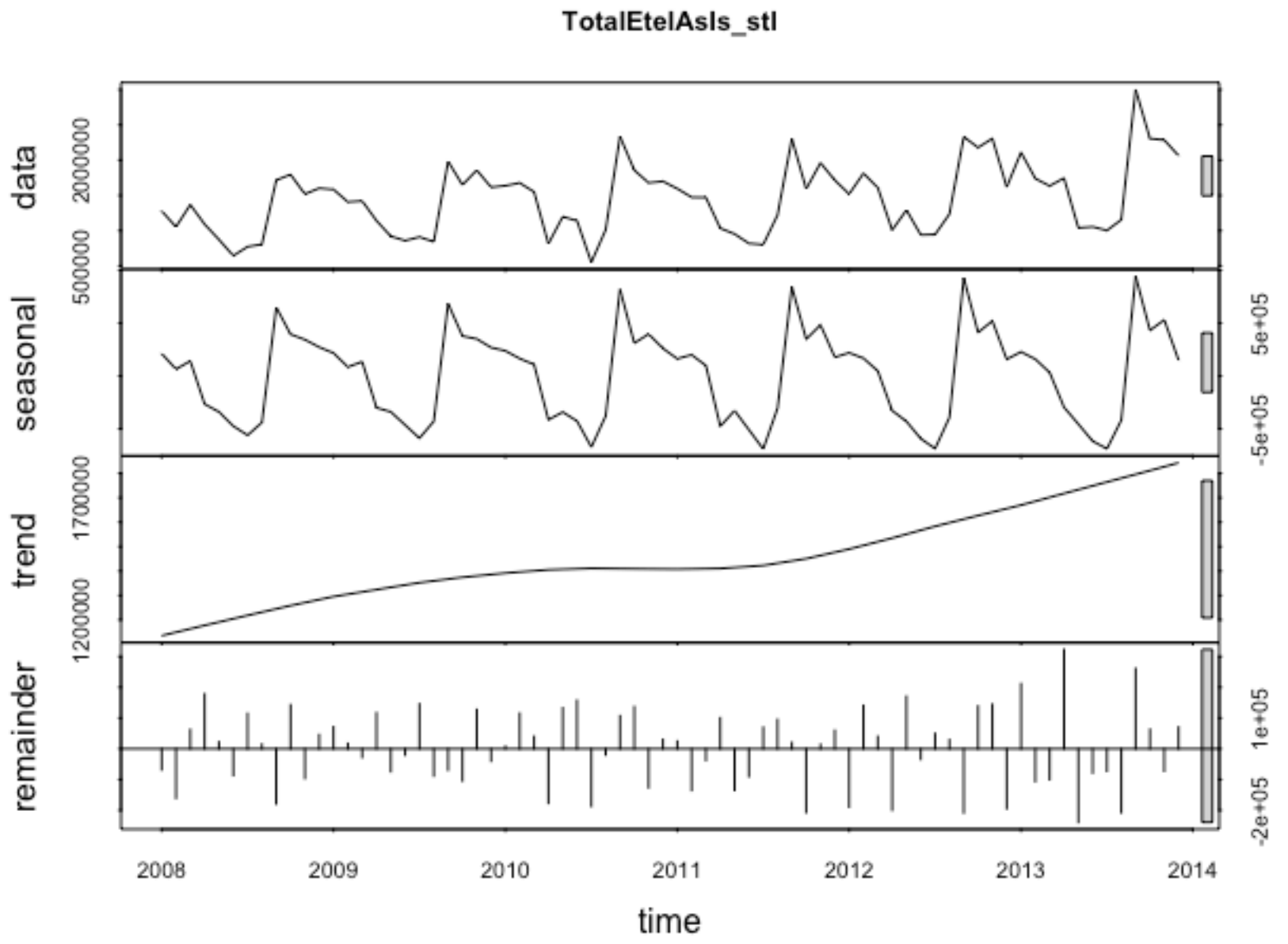

```
plot(EfakAsIs_stl, col="black", main="EfakAsIs_stl")
```



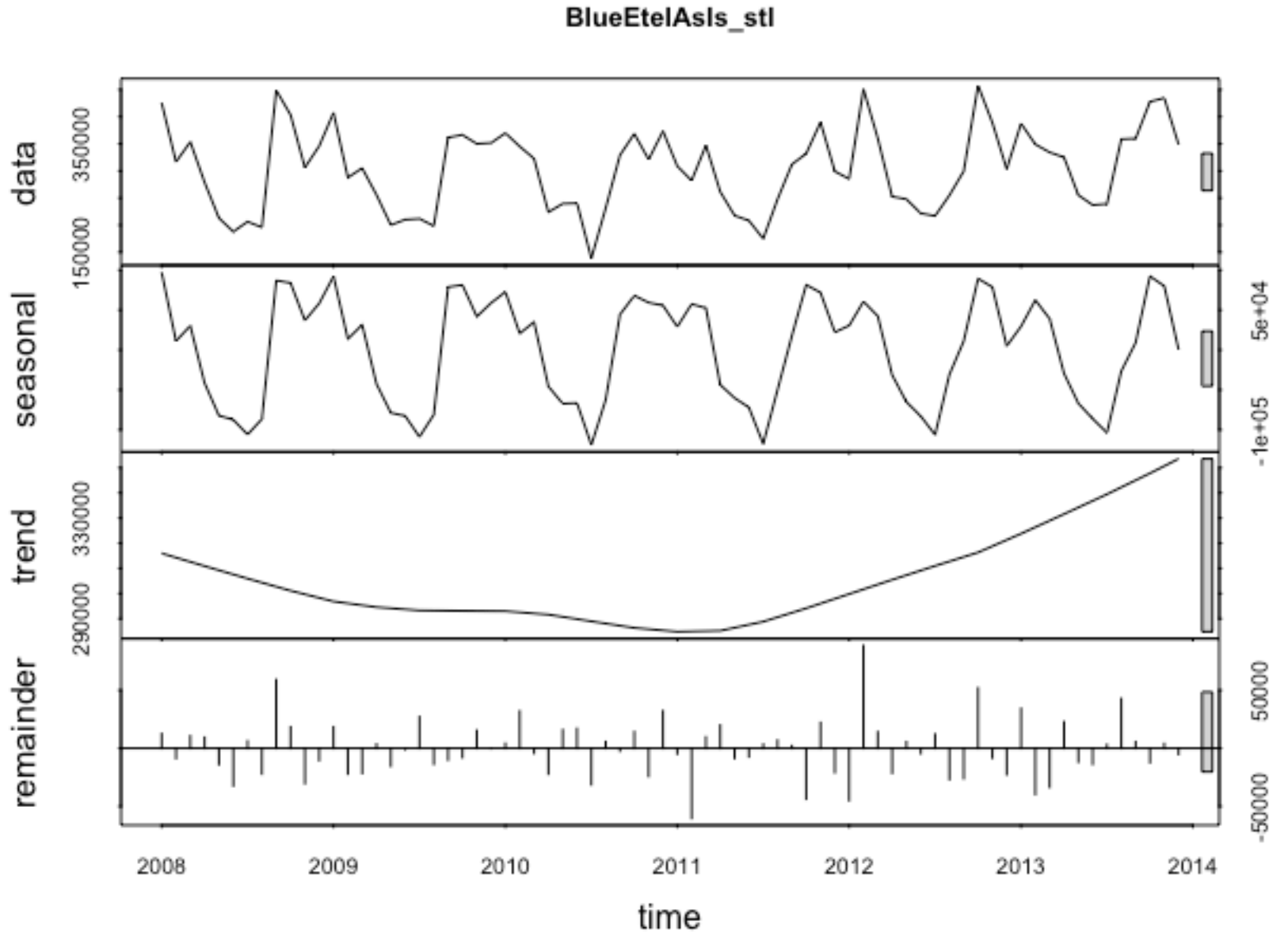

```
plot(WugeAsIs_stl, col="black", main="WugeAsIs_stl")
```



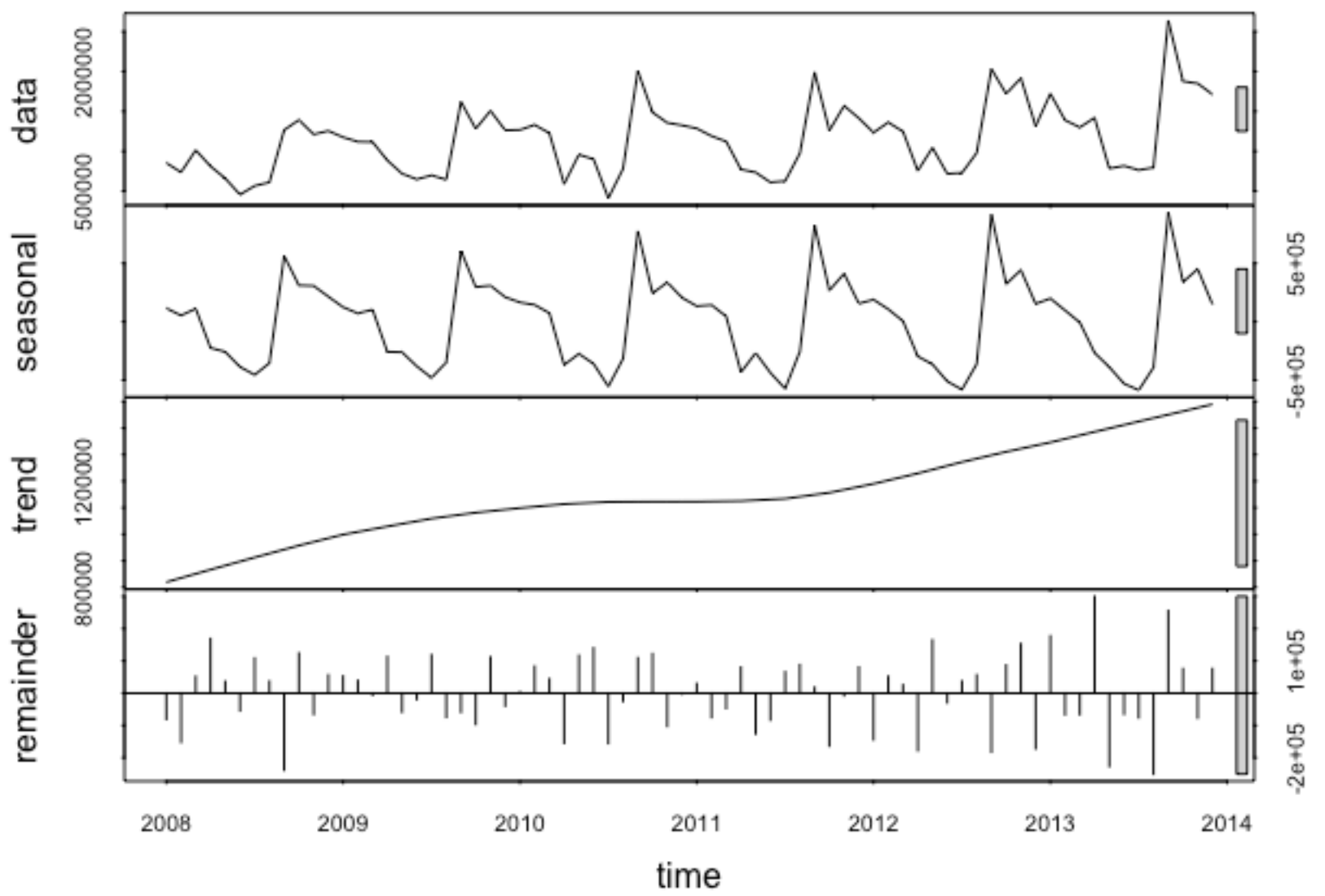

```
plot(TotalEtelAsIs_stl, col="black", main="TotalEtelAsIs_stl")
```




```
plot(BlueEtelAsIs_stl, col="black", main="BlueEtelAsIs_stl")
```

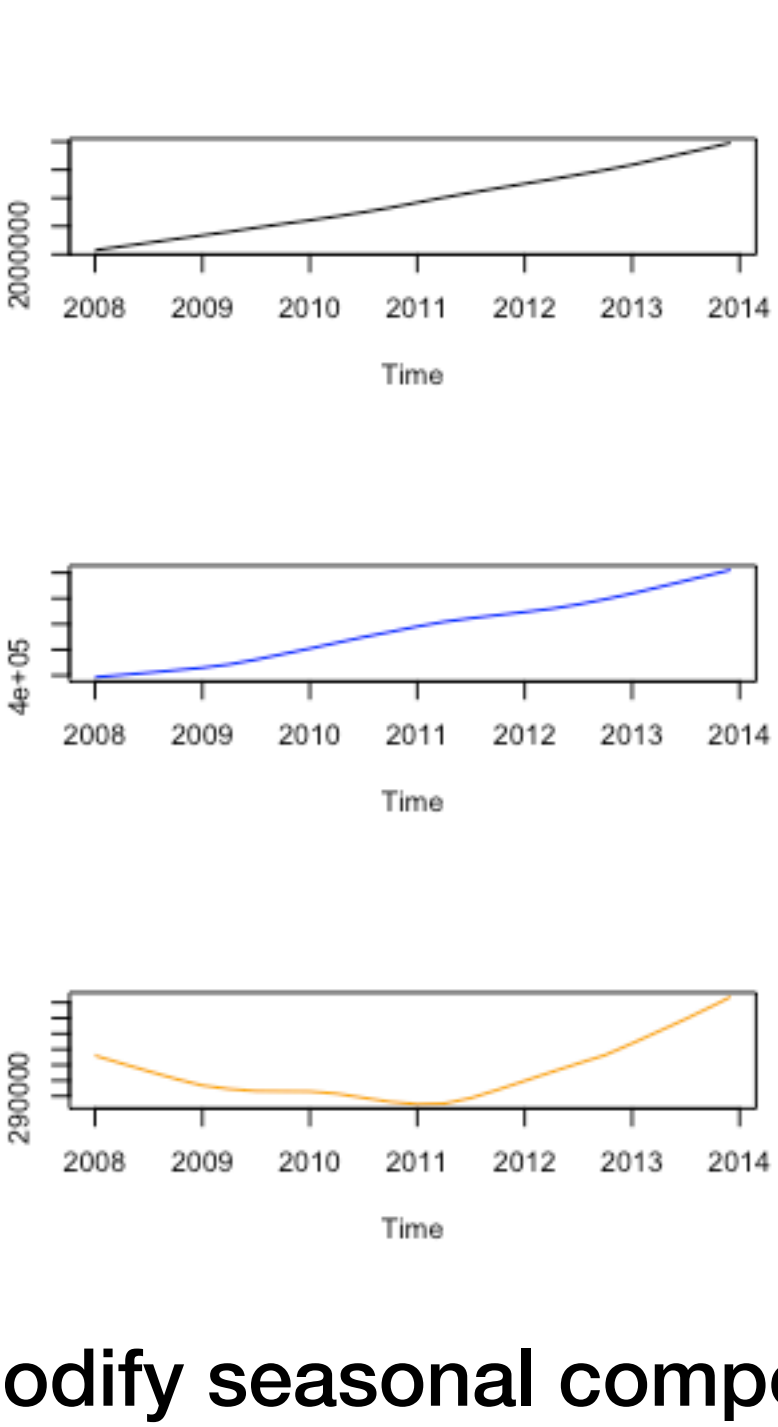


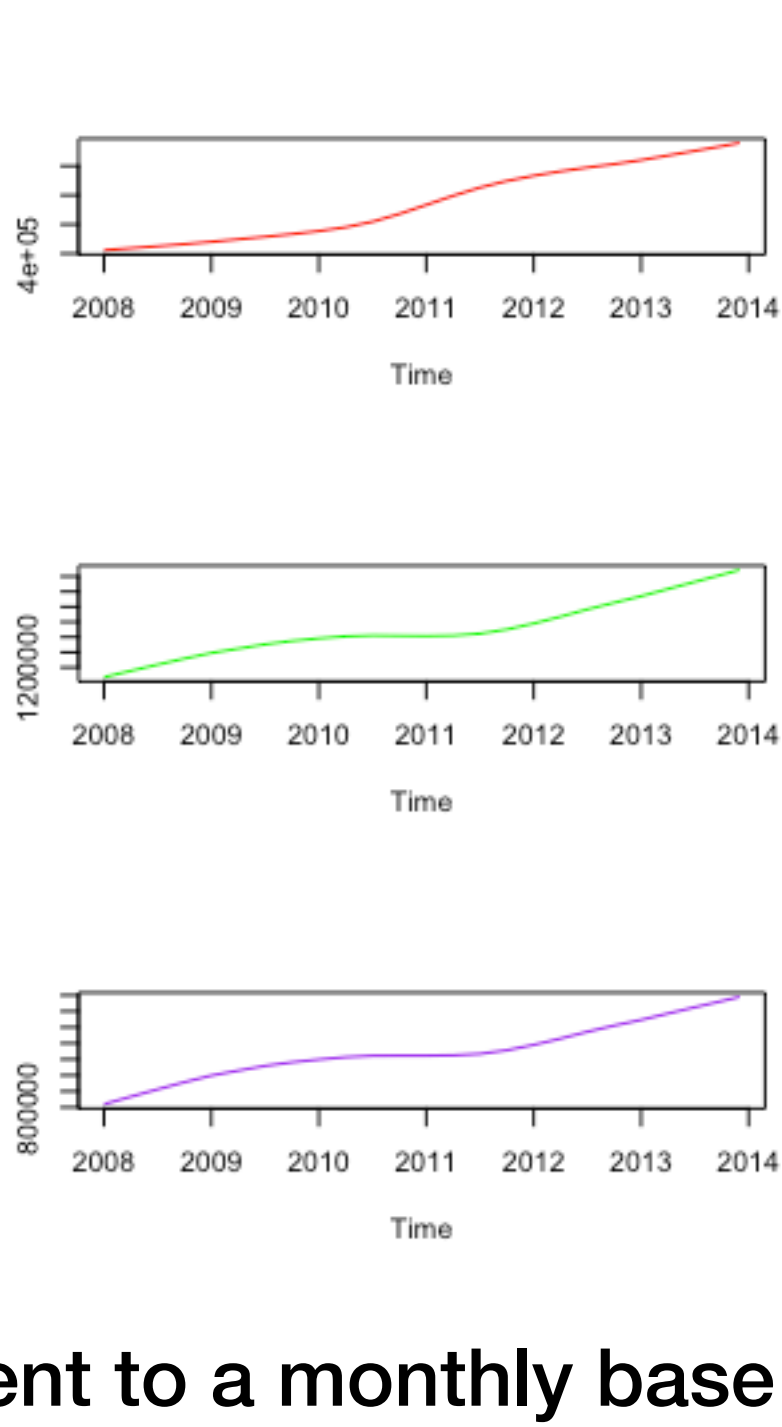

```
plot(RedEtelAsIs_stl, col="black", main="RedEtelAsIs_stl")
```



It is interesting to note that the almost linear trend is not seen in the individual segments. The individual trends run partially in opposite directions in the middle of the time scale, which causes the linear trend in the total As Is data.

```
par(mfrow=c(3,2))
plot(TotalAsIs_stl$time.series[,"trend"], col="black")
plot(EfakAsIs_stl$time.series[,"trend"], col="red")
plot(WugeAsIs_stl$time.series[,"trend"], col="blue")
plot(TotalEtelAsIs_stl$time.series[,"trend"], col="green")
plot(BlueEtelAsIs_stl$time.series[,"trend"], col="orange")
plot(RedEtelAsIs_stl$time.series[,"trend"], col="purple")
```

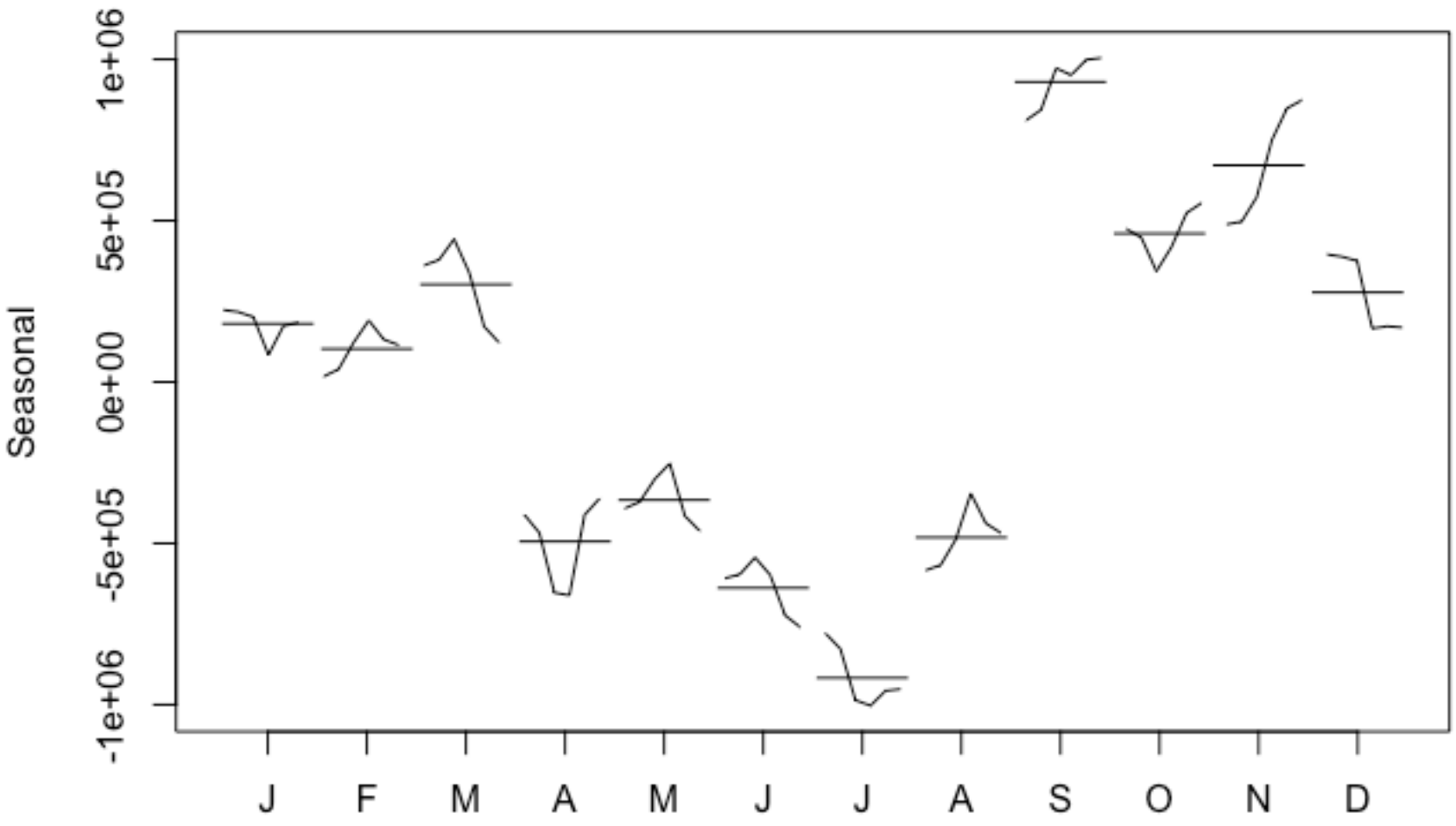




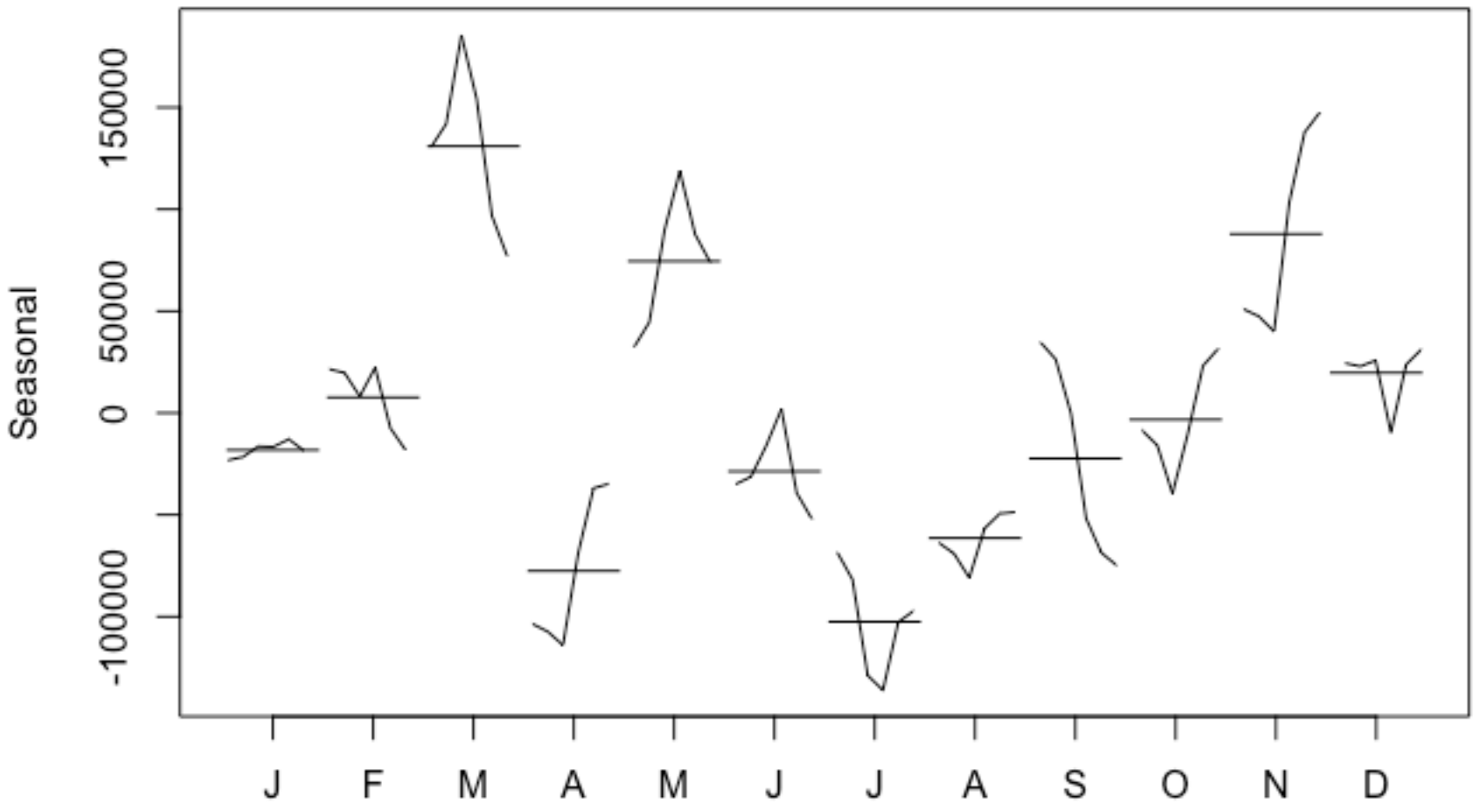
Modify seasonal component to a monthly base

The modification of the seasonlity component can also be changed into a monthly view. It only makes sense to do this if the seasonality componant as the trend looks almost identical and the remainder is then randomly spread.

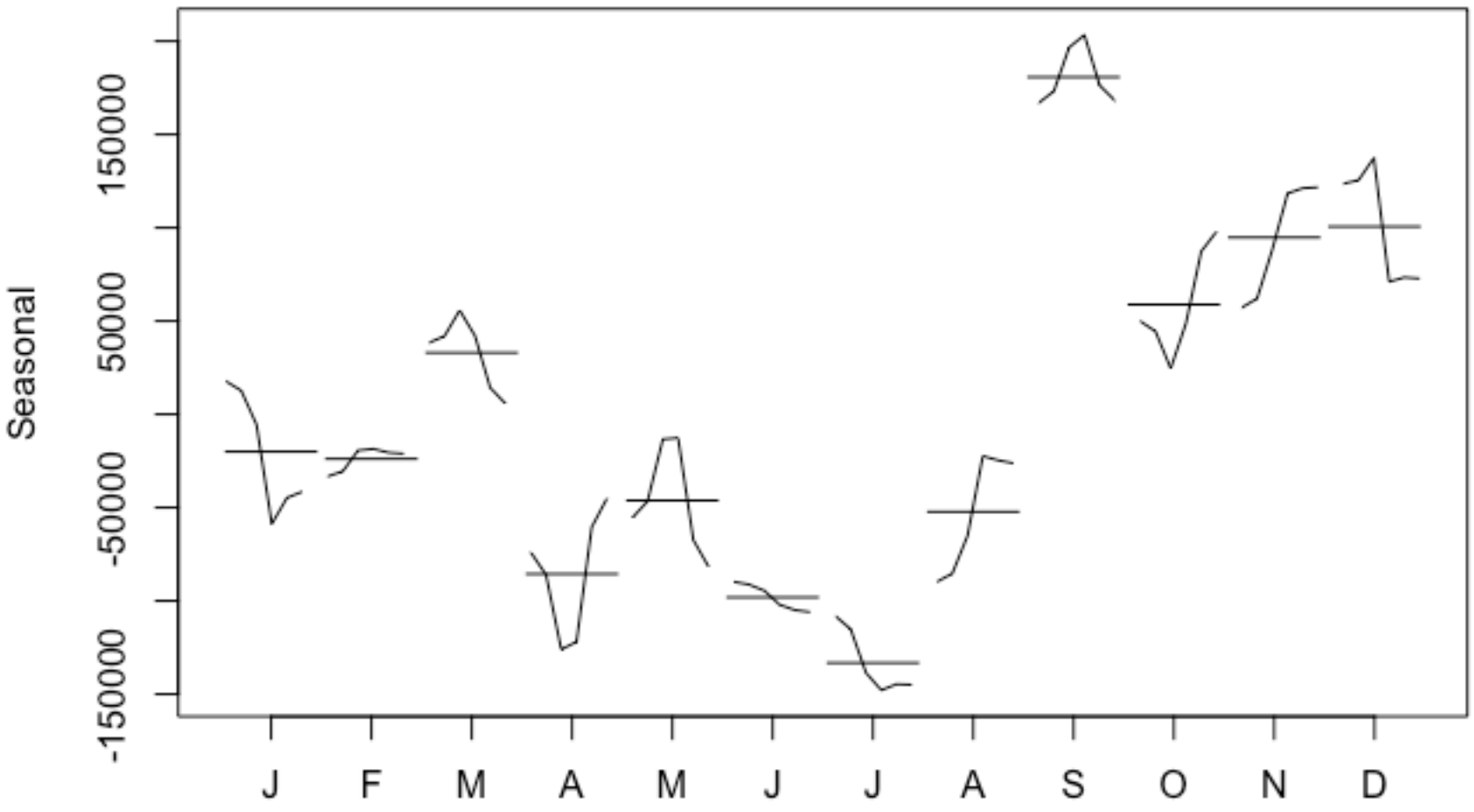
```
monthplot(TotalAsIs_stl$time.series[,"seasonal"], main="", ylab="Seasonal")
```



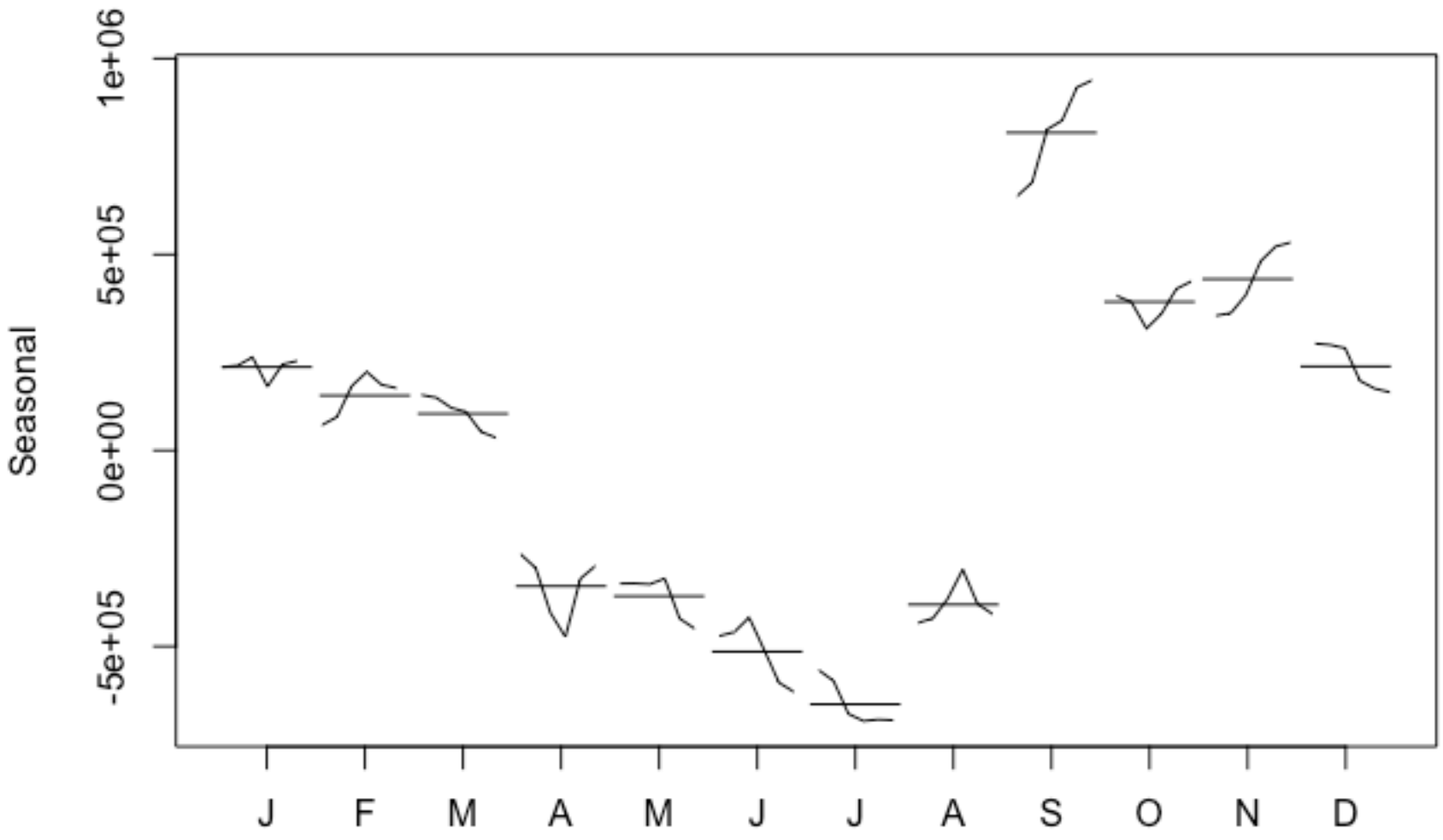

```
monthplot(EfakAsIs_stl$time.series[,"seasonal"], main="", ylab="Seasonal")
```



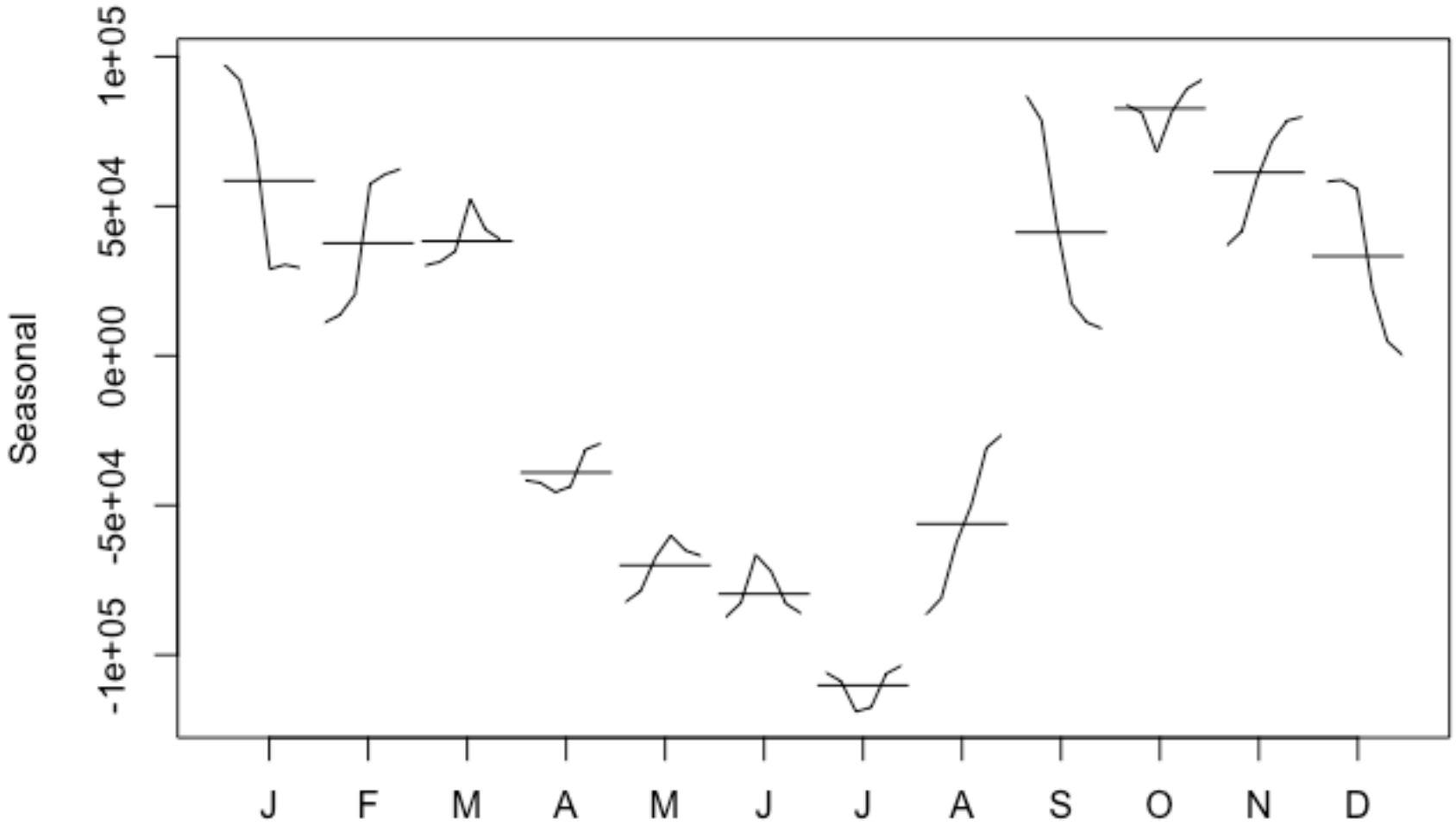

```
monthplot(WugeAsIs_stl$time.series[,"seasonal"], main="", ylab="Seasonal")
```



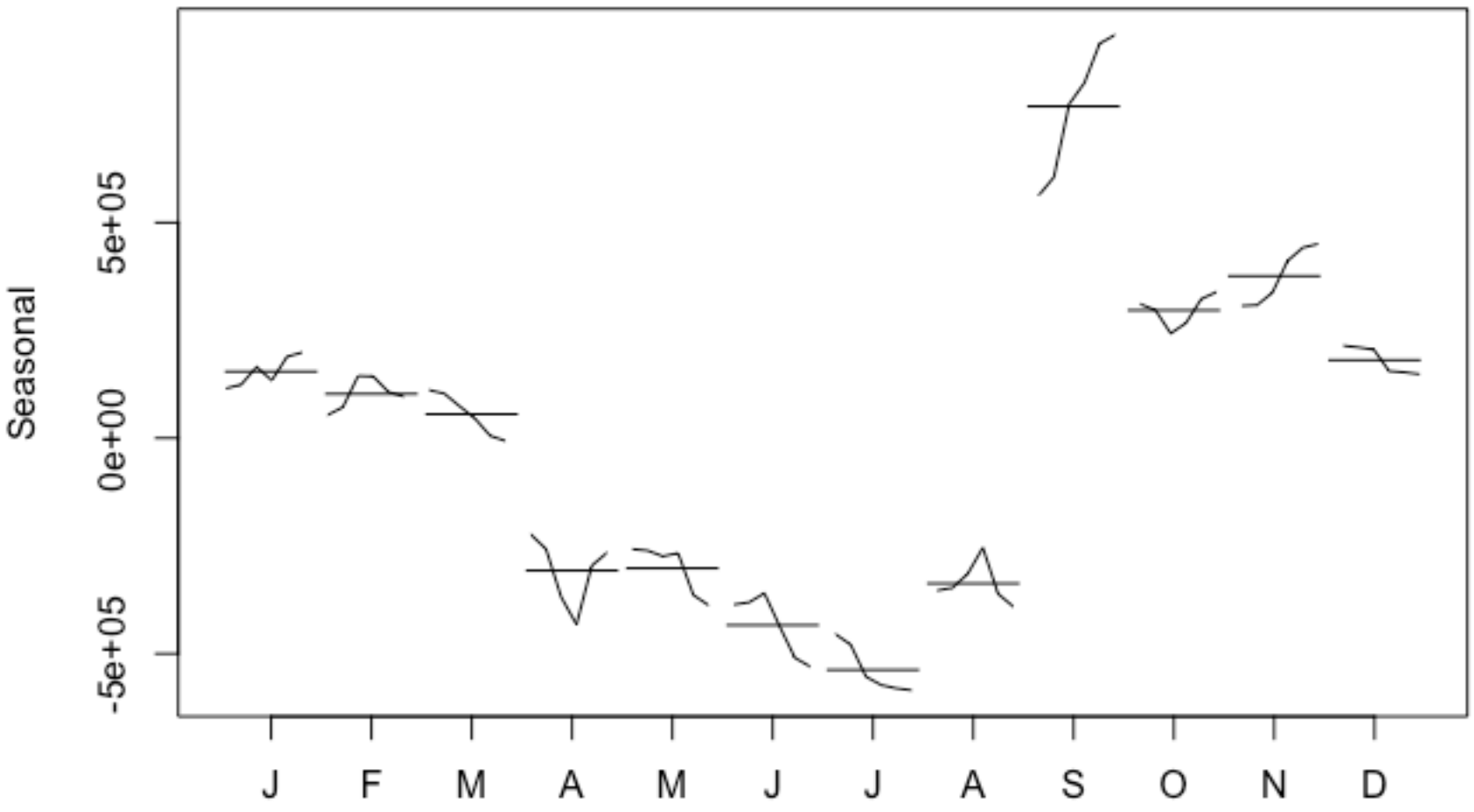

```
monthplot(TotalEtelAsIs_stl$time.series[,"seasonal"], main="", ylab="Seasonal")
```




```
monthplot(BlueEtelAsIs_stl$time.series[,"seasonal"], main="", ylab="Seasonal")
```




```
monthplot(RedEtelAsIs_stl$time.series[,"seasonal"], main="", ylab="Seasonal")
```



Correlation with external indicators

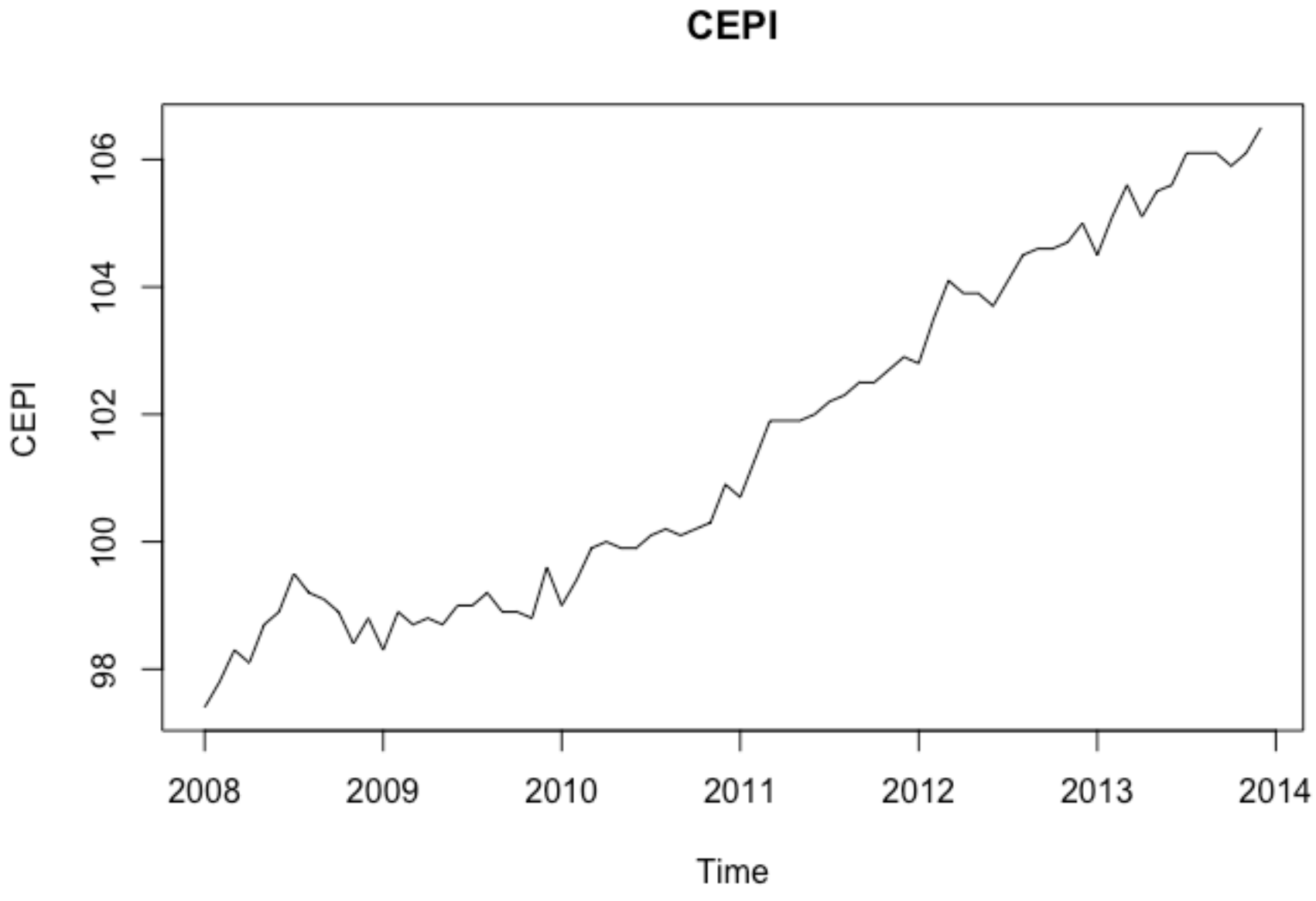
The indicators are as follows:

- Monthly Change in Export Price Index (CEPI)
- Monthly Satisfaction Index (SI) government based data
- Average monthly temperatures in Chulwalar
- Monthly births in Chulwalar
- Monthly Satisfaction Index (SI) external index
- Yearly Exports from Urbano
- Yearly number of Globalisation Party members in Chulwalar
- Monthly Average Export Price Index for Chulwalar
- Monthly Producer Price Index (PPI) for Etel in Chulwalar
- National Holidays
- Chulwalar Index (Total value of all companies in Chulwalar)
- Monthly Inflation rate in Chulwalar
- Proposed spending for National Holidays
- Influence of National Holiday

The indicators will be converted into individual vectors and subsequently converted into time series. The correlation of the indicators will then be tested against the As Is exports for Chulwalar.

Monthly Change in Export Price Index (CEPI)

```
CEPIVector <- c(ImportedIndicators[2:13,2],ImportedIndicators[2:13,3],ImportedIndicat
ors[2:13,4],ImportedIndicators[2:13,5],ImportedIndicators[2:13,6],ImportedIndicators[
2:13,7])
CEPI <- ts(CEPIVector , start=c(2008,1), end=c(2013,12), frequency=12)
plot(CEPI, main="CEPI")
```




```
cor(TotalAsIs, CEPI)
```

```
## [1] 0.663925
```

```
cor(EfakAsIs , CEPI)
```

```
## [1] 0.9303543
```

```
cor(WugeAsIs, CEPI)
```

```
## [1] 0.7618551
```

```
cor(TotalEtelAsIs, CEPI)
```

```
## [1] 0.339713
```

```
cor(BlueEtelAsIs , CEPI)
```

```
## [1] 0.1448837
```

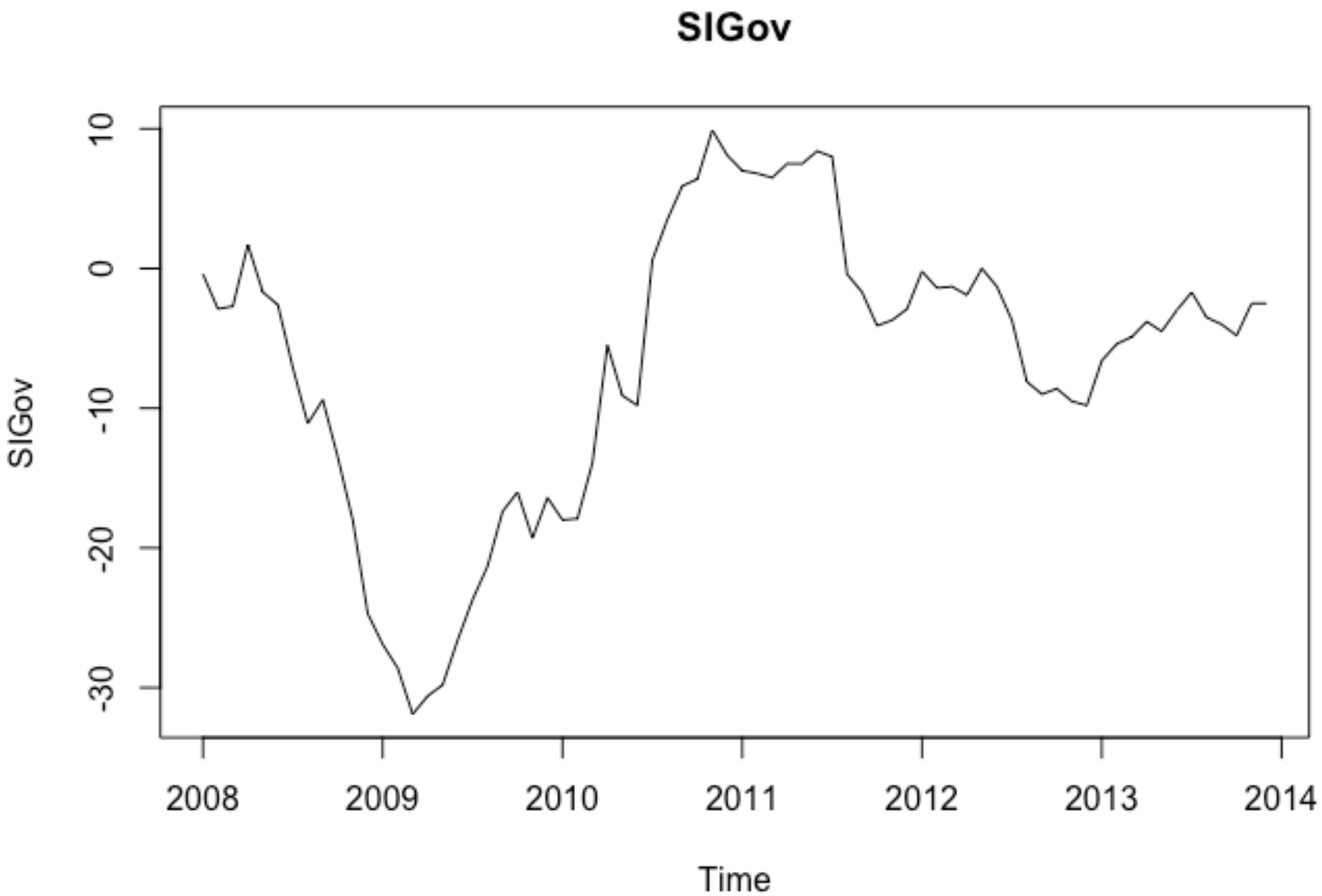
```
cor(RedEtelAsIs , CEPI)
```

```
## [1] 0.3587646
```

Total As Is and CEPI Correlation shows siginificant and positive correlation. (0.663925)

Monthly Satisfaction Index (SI) government based data

```
SIGovVector <- c(ImportedIndicators[16:27,2],ImportedIndicators[16:27,3],ImportedIndi
cators[16:27,4],ImportedIndicators[16:27,5],ImportedIndicators[16:27,6],ImportedIndic
ators[16:27,7])
SIGov <- ts(SIGovVector , start=c(2008,1), end=c(2013,12), frequency=12)
plot(SIGov, main="SIGov")
```




```
cor(TotalAsIs, SIGov)
```

```
## [1] 0.2007768
```

```
cor(EfakAsIs , SIGov)
```

```
## [1] 0.37934
```

```
cor(WugeAsIs, SIGov)
```

```
## [1] 0.3030266
```

```
cor(TotalEtelAsIs, SIGov)
```

```
## [1] 0.002556094
```

```
cor(BlueEtelAsIs , SIGov)
```

```
## [1] -0.04146932
```

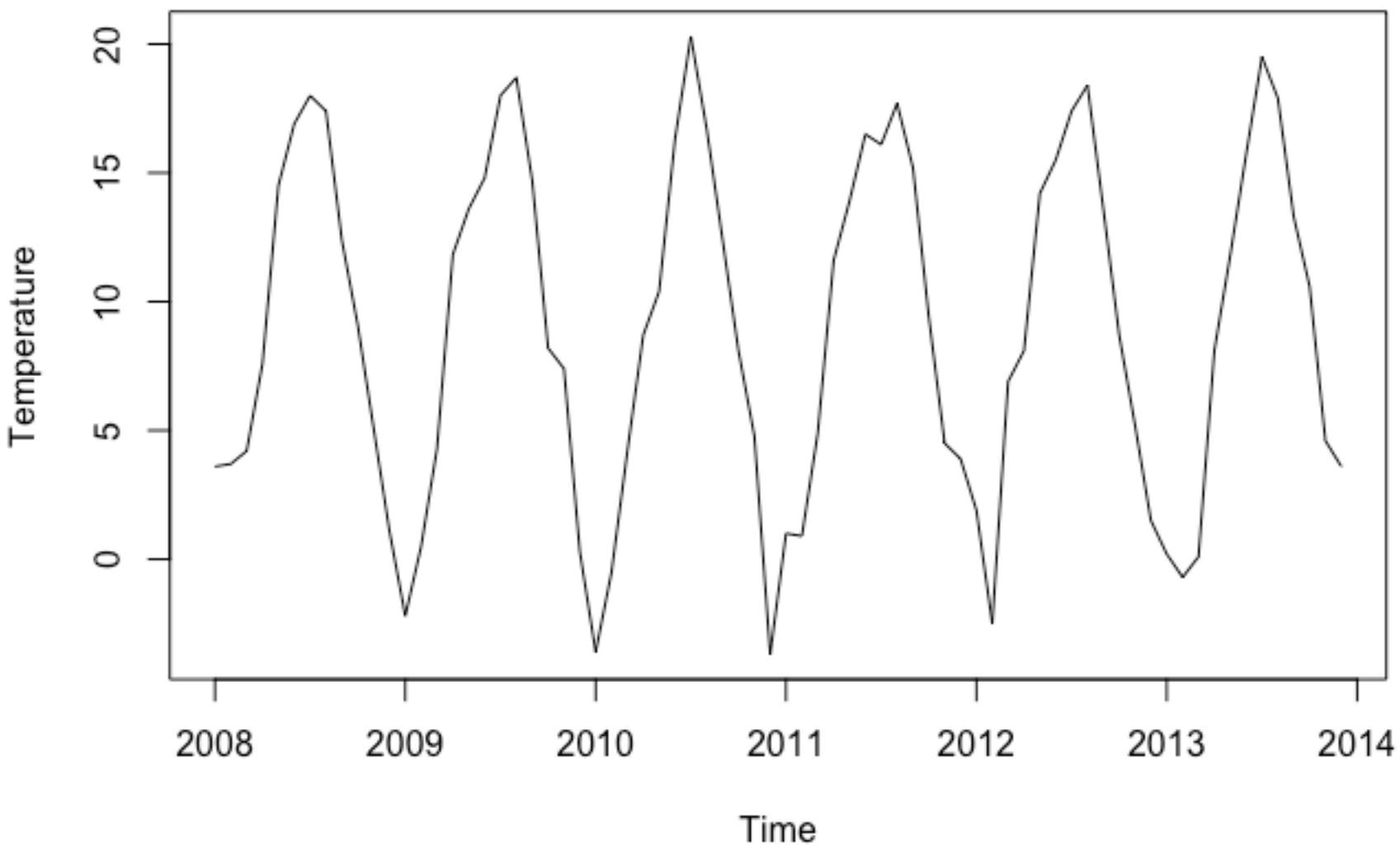
```
cor(RedEtelAsIs , SIGov)
```

```
## [1] 0.009978415
```

Total As Is and SI Correlation shows small and positive correlation.(0.2007768)

Average monthly temperatures in Chulwalar

```
TemperatureVector <- c(ImportedIndicators[30:41,2],ImportedIndicators[30:41,3],Import
edIndicators[30:41,4],ImportedIndicators[30:41,5],ImportedIndicators[30:41,6],Importe
dIndicators[30:41,7])
Temperature <- ts(TemperatureVector, start=c(2008,1), end=c(2013,12), frequency=12)
plot(Temperature, main="Temperature")
```




```
cor(TotalAsIs, Temperature)
```

```
## [1] -0.3429684
```

```
cor(EfakAsIs , Temperature)
```

```
## [1] -0.07951179
```

```
cor(WugeAsIs, Temperature)
```

```
## [1] -0.2045082
```

```
cor(TotalEtelAsIs, Temperature)
```

```
## [1] -0.453138
```

```
cor(BlueEtelAsIs , Temperature)
```

```
## [1] -0.6356067
```

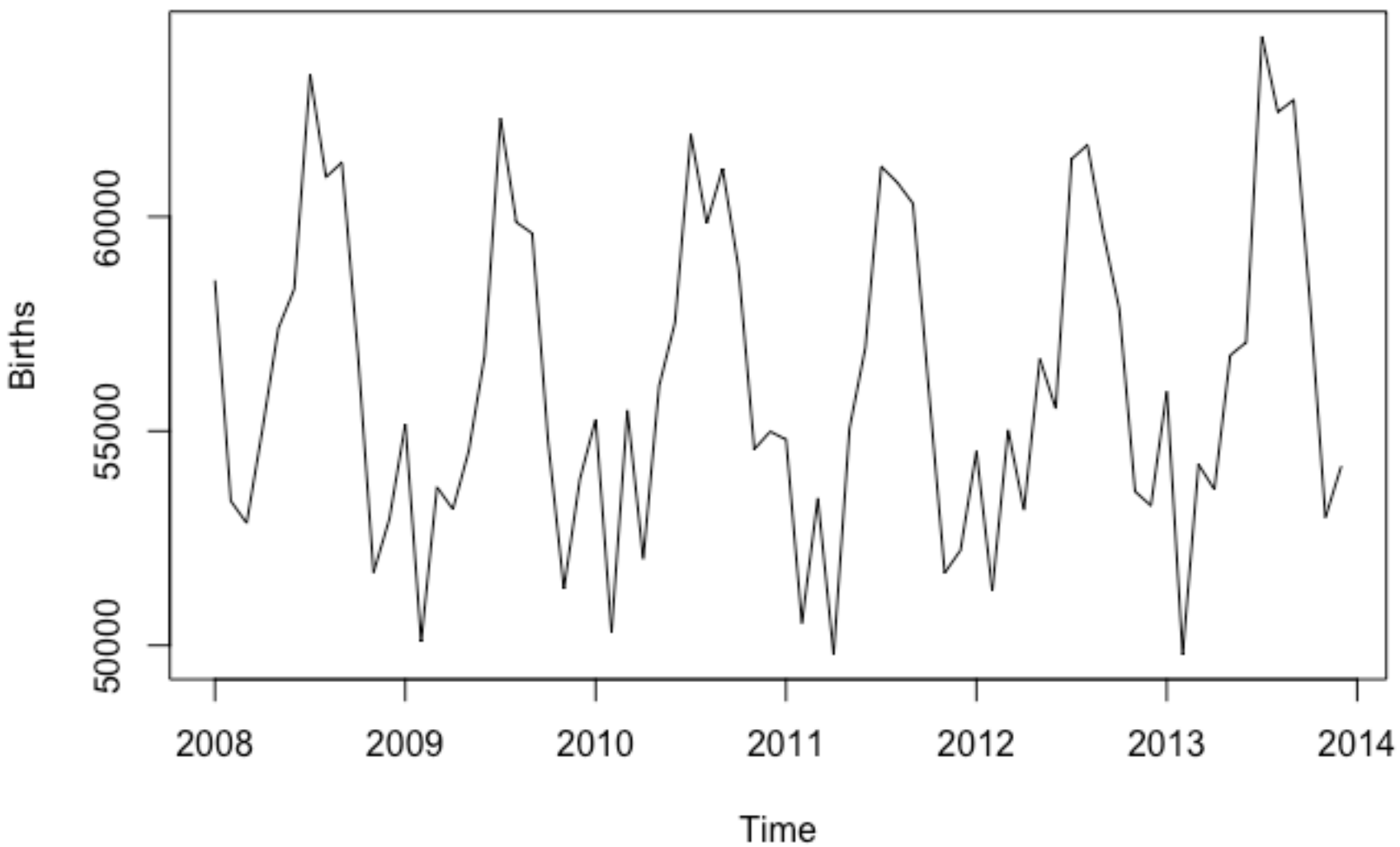
```
cor(RedEtelAsIs , Temperature)
```

```
## [1] -0.4028941
```

Total As Is and SI Correlation shows moderate and negative correlation. (-0.3429684)

Monthly births in Chulwalar

```
BirthsVector <- c(ImportedIndicators[44:55,2],ImportedIndicators[44:55,3],ImportedInd
icators[44:55,4],ImportedIndicators[44:55,5],ImportedIndicators[44:55,6],ImportedIndi
cators[44:55,7])
Births <- ts(BirthsVector, start=c(2008,1), end=c(2013,12), frequency=12)
plot(Births, main="Births")
```




```
cor(TotalAsIs, Births)
```

```
## [1] -0.1190228
```

```
cor(EfakAsIs , Births)
```

```
## [1] -0.05802961
```

```
cor(WugeAsIs, Births)
```

```
## [1] -0.007371339
```

```
cor(TotalEtelAsIs, Births)
```

```
## [1] -0.1504242
```

```
cor(BlueEtelAsIs , Births)
```

```
## [1] -0.2812913
```

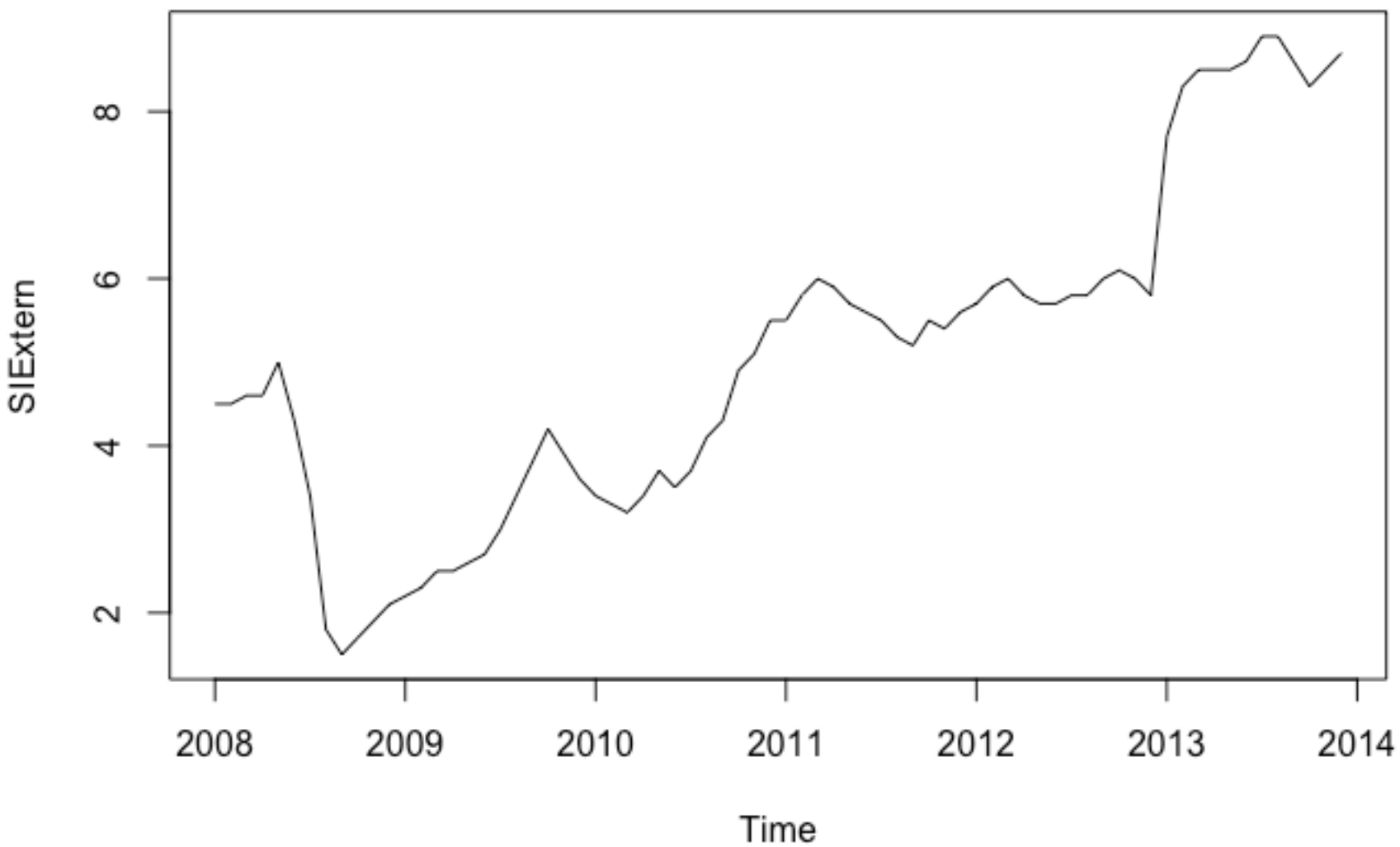
```
cor(RedEtelAsIs , Births)
```

```
## [1] -0.1217222
```

Total As Is and Birth Correlation shows small and negative correlation.(-0.1190228)

Monthly Satisfaction Index (SI) external index

```
SIExternVector <- c(ImportedIndicators[58:69,2],ImportedIndicators[58:69,3],ImportedI
ndicators[58:69,4],ImportedIndicators[58:69,5],ImportedIndicators[58:69,6],ImportedIn
dicators[58:69,7])
SIExtern <- ts(SIExternVector, start=c(2008,1), end=c(2013,12), frequency=12)
plot(SIExtern, main="SIExtern")
```




```
cor(TotalAsIs, SIExtern)
```

```
## [1] 0.5883122
```

```
cor(EfakAsIs , SIExtern)
```

```
## [1] 0.8358147
```

```
cor(WugeAsIs, SIExtern)
```

```
## [1] 0.6786552
```

```
cor(TotalEtelAsIs, SIExtern)
```

```
## [1] 0.2865672
```

```
cor(BlueEtelAsIs , SIExtern)
```

```
## [1] 0.1604768
```

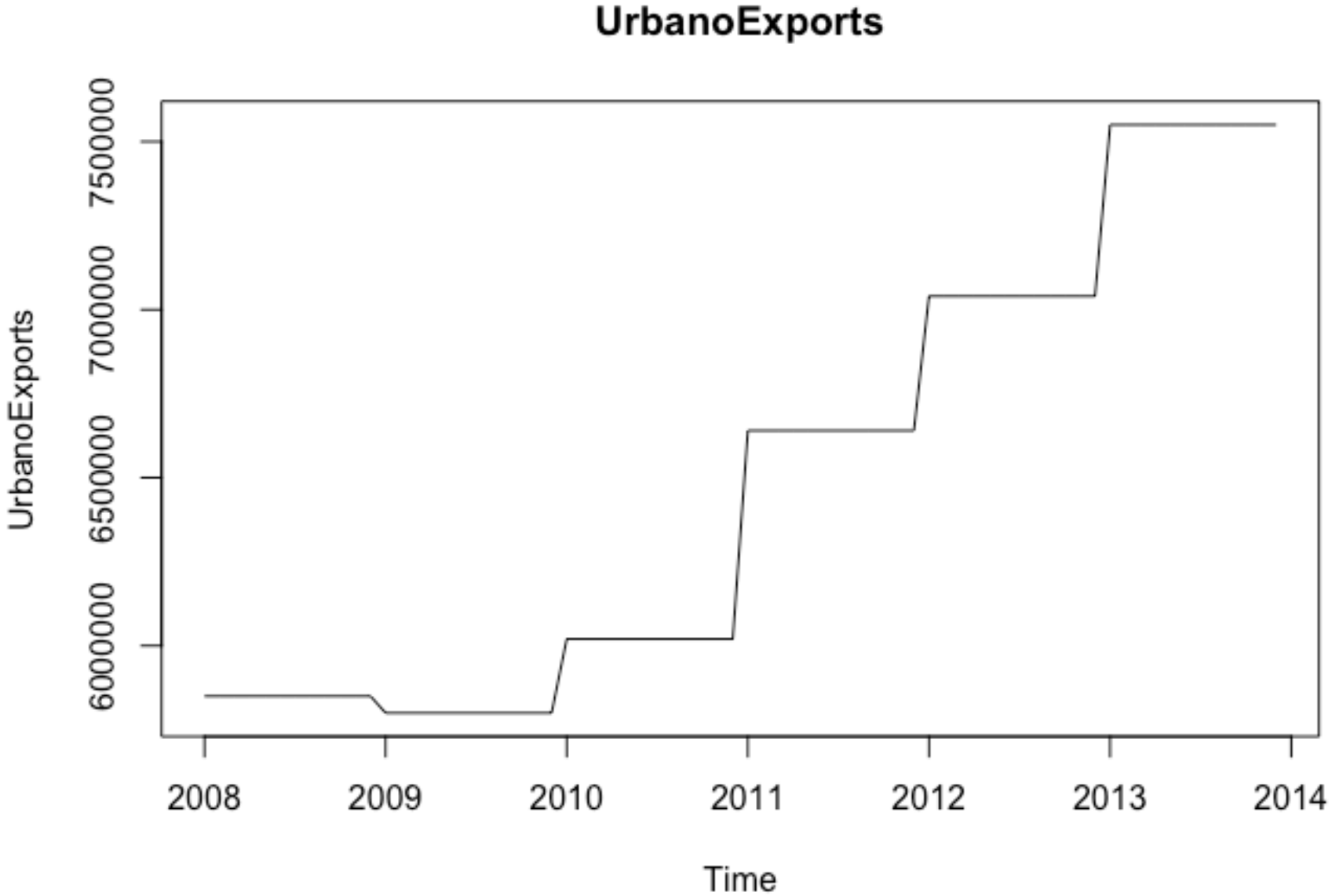
```
cor(RedEtelAsIs , SIExtern)
```

```
## [1] 0.2960946
```

Total As Is and SI Correlation shows moderate and positive correlation. (0.5883122)

Yearly exports from Urbano

```
UrbanoExportsVector <- c(ImportedIndicators[72:83,2],ImportedIndicators[72:83,3],Impo
rtedIndicators[72:83,4],ImportedIndicators[72:83,5],ImportedIndicators[72:83,6],Impor
tedIndicators[72:83,7])
UrbanoExports <- ts(UrbanoExportsVector, start=c(2008,1), end=c(2013,12), frequency=1
2)
plot(UrbanoExports, main="UrbanoExports")
```




```
cor(TotalAsIs, UrbanoExports)
```

```
## [1] 0.638178
```

```
cor(EfakAsIs , UrbanoExports)
```

```
## [1] 0.9163565
```

```
cor(WugeAsIs, UrbanoExports)
```

```
## [1] 0.7118468
```

```
cor(TotalEtelAsIs, UrbanoExports)
```

```
## [1] 0.3182532
```

```
cor(BlueEtelAsIs , UrbanoExports)
```

```
## [1] 0.1655794
```

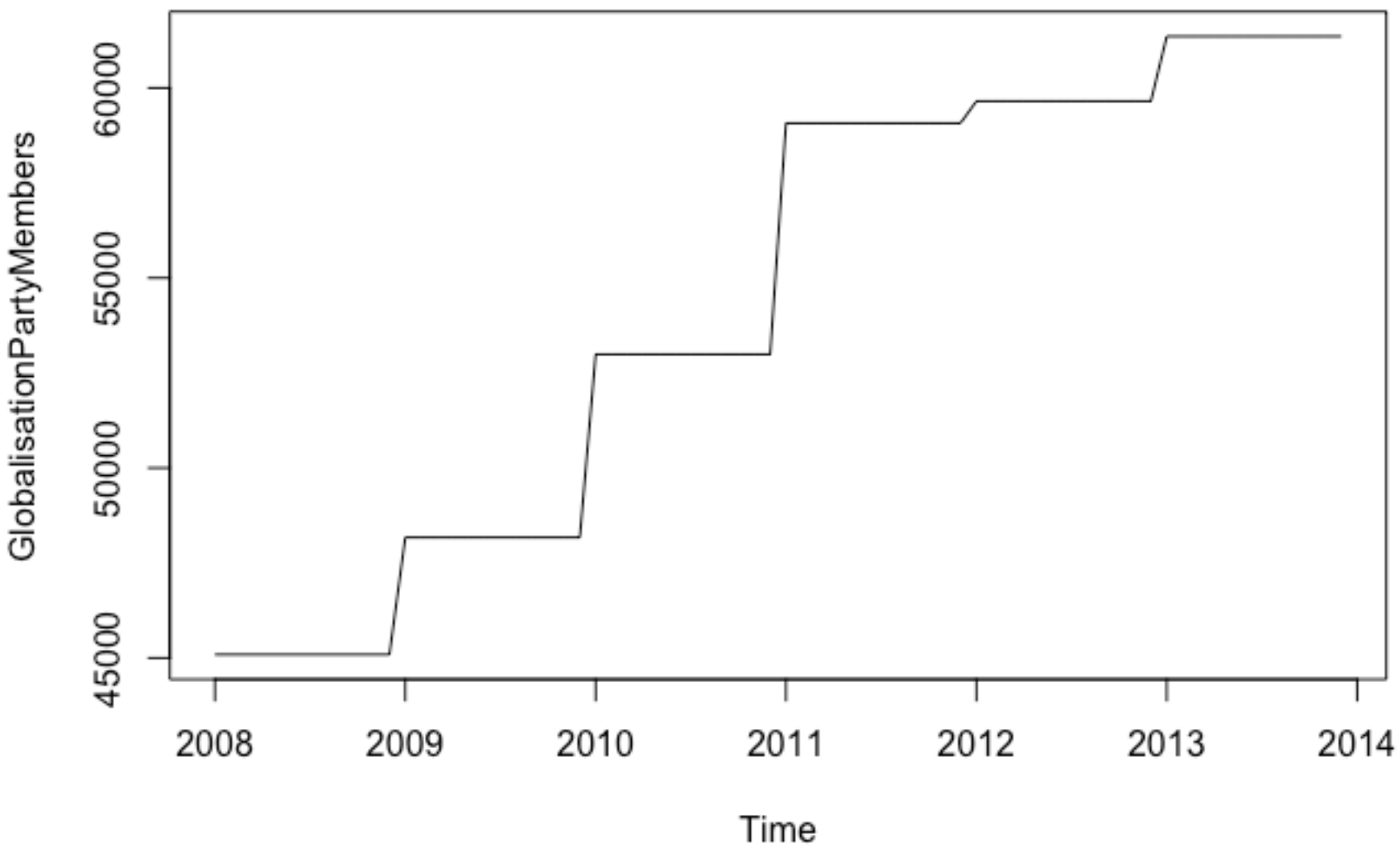
```
cor(RedEtelAsIs , UrbanoExports)
```

```
## [1] 0.3309962
```

Total As Is and Urbano Correlation shows moderate and positive correlation. (0.638178)

Yearly number of Globalisation Party members in Chulwalar

```
GlobalisationPartyMembersVector <- c(ImportedIndicators[86:97,2],ImportedIndicators[8
6:97,3],ImportedIndicators[86:97,4],ImportedIndicators[86:97,5],ImportedIndicators[86
:97,6],ImportedIndicators[86:97,7])
GlobalisationPartyMembers <- ts(GlobalisationPartyMembersVector, start=c(2008,1), end
=c(2013,12), frequency=12)
plot(GlobalisationPartyMembers, main="GlobalisationPartyMembers")
```




```
cor(TotalAsIs, GlobalisationPartyMembers)
```

```
## [1] 0.630084
```

```
cor(EfakAsIs , GlobalisationPartyMembers)
```

```
## [1] 0.8963942
```

```
cor(WugeAsIs, GlobalisationPartyMembers)
```

```
## [1] 0.7193864
```

```
cor(TotalEtelAsIs, GlobalisationPartyMembers)
```

```
## [1] 0.2994635
```

```
cor(BlueEtelAsIs , GlobalisationPartyMembers)
```

```
## [1] 0.08547266
```

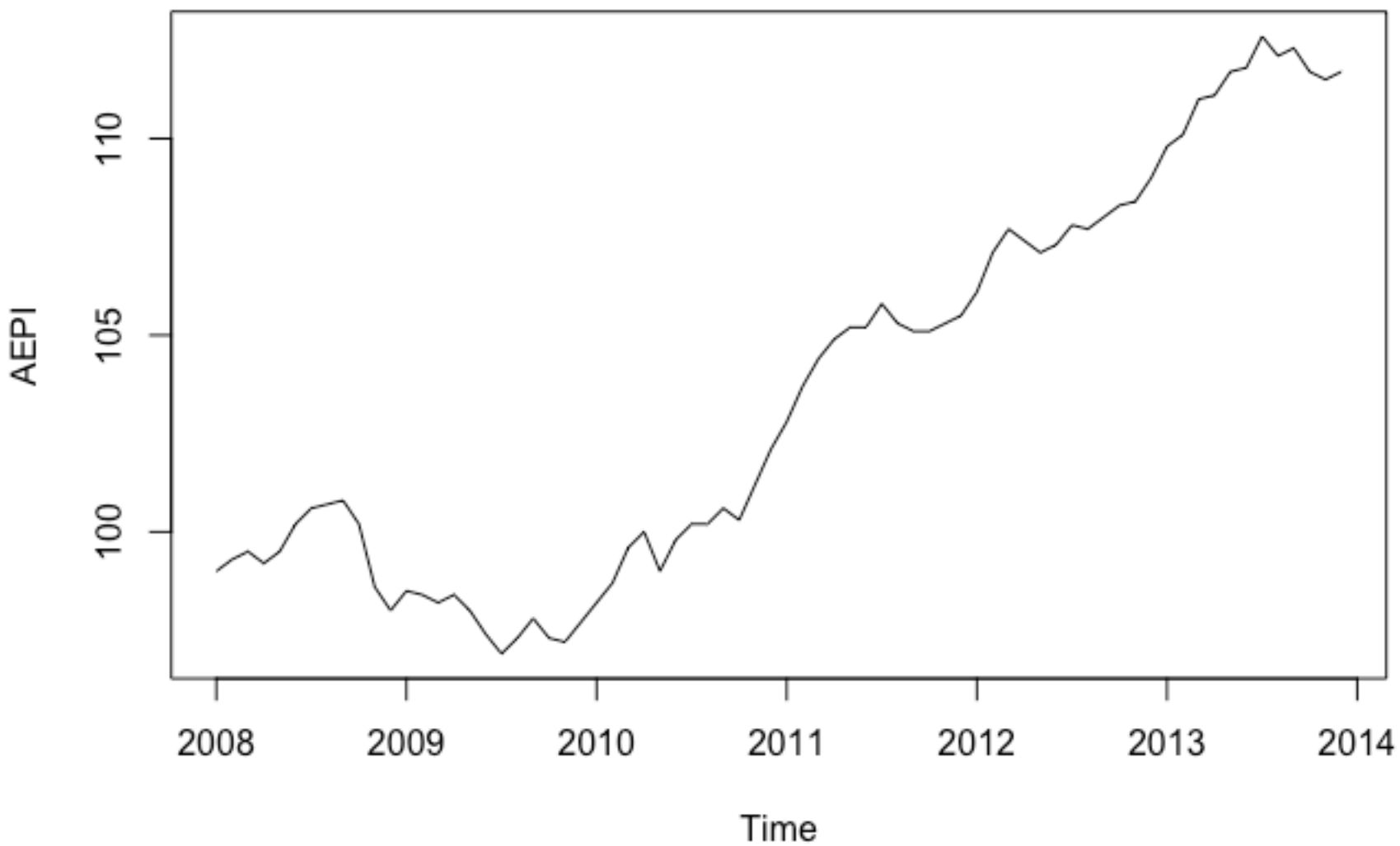
```
cor(RedEtelAsIs , GlobalisationPartyMembers)
```

```
## [1] 0.3234832
```

Total As Is and Yearly number of Globalisation Party members Correlation shows moderate and positive correlation. (0.630084)

Monthly Average Export Price Index for Chulwalar

```
AEPIVector <- c(ImportedIndicators[100:111,2],ImportedIndicators[100:111,3],ImportedI
ndicators[100:111,4],ImportedIndicators[100:111,5],ImportedIndicators[100:111,6],Impo
rtedIndicators[100:111,7])
AEPI <- ts(AEPIVector, start=c(2008,1), end=c(2013,12), frequency=12)
plot(AEPI, main="AEPI")
```




```
cor(TotalAsIs, AEPI)
```

```
## [1] 0.625232
```

```
cor(EfakAsIs , AEPI)
```

```
## [1] 0.9056624
```

```
cor(WugeAsIs, AEPI)
```

```
## [1] 0.7159733
```

```
cor(TotalEtelAsIs, AEPI)
```

```
## [1] 0.3035506
```

```
cor(BlueEtelAsIs , AEPI)
```

```
## [1] 0.1577964
```

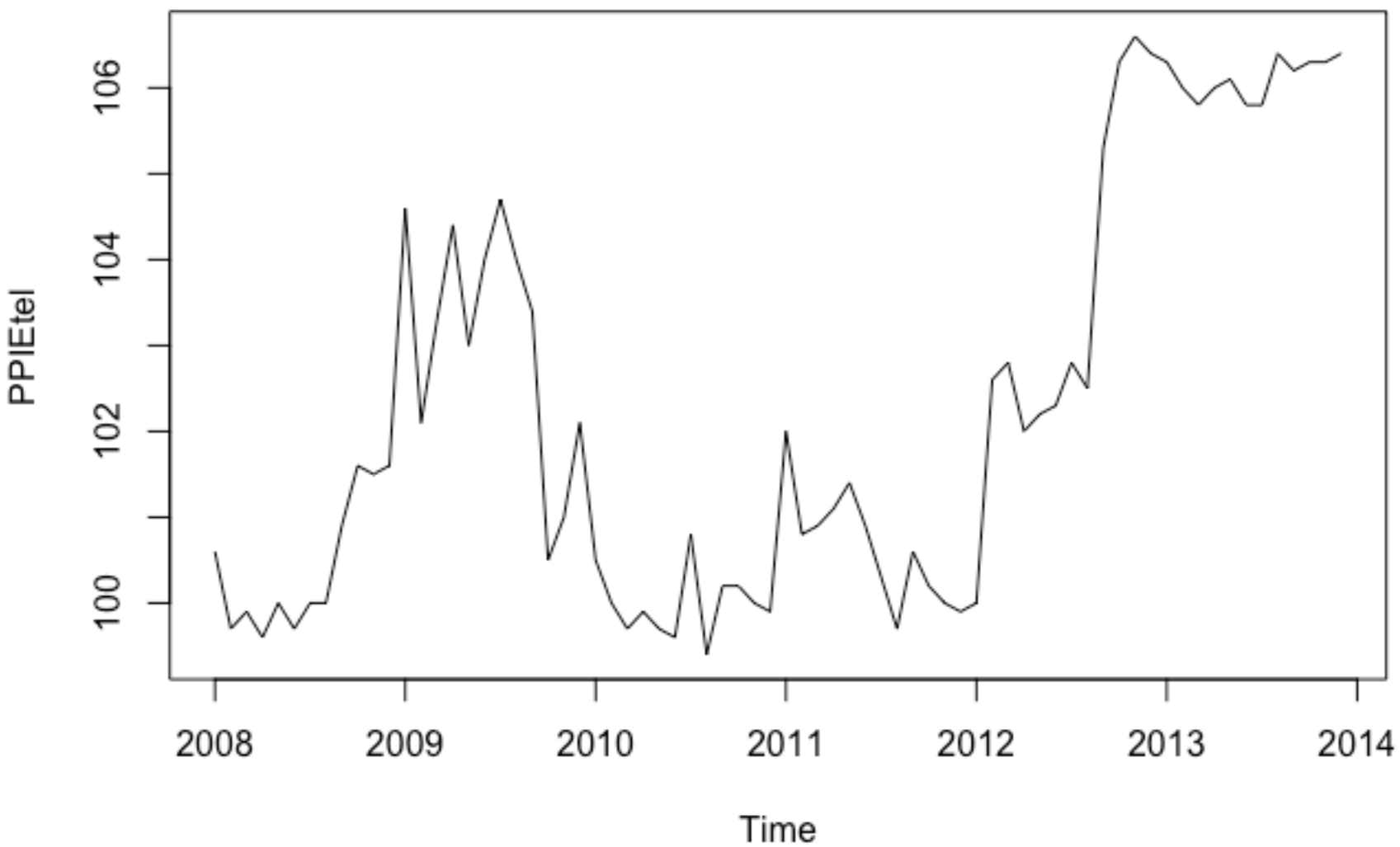
```
cor(RedEtelAsIs , AEPI)
```

```
## [1] 0.3157277
```

Total As Is and AEPI Correlation shows moderate and positive correlation. (0.625232)

Monthly Producer Price Index (PPI) for Etel in Chulwalar

```
PPIEtelVector <- c(ImportedIndicators[114:125,2],ImportedIndicators[114:125,3],Import
edIndicators[114:125,4],ImportedIndicators[114:125,5],ImportedIndicators[114:125,6],I
mportedIndicators[114:125,7])
PPIEtel <- ts(PPIEtelVector, start=c(2008,1), end=c(2013,12), frequency=12)
plot(PPIEtel, main="PPIEtel")
```




```
cor(TotalAsIs, PPIEtel)
```

```
## [1] 0.4836129
```

```
cor(EfakAsIs , PPIEtel)
```

```
## [1] 0.5865375
```

```
cor(WugeAsIs, PPIEtel)
```

```
## [1] 0.4920865
```

```
cor(TotalEtelAsIs, PPIEtel)
```

```
## [1] 0.3374707
```

```
cor(BlueEtelAsIs , PPIEtel)
```

```
## [1] 0.2445472
```

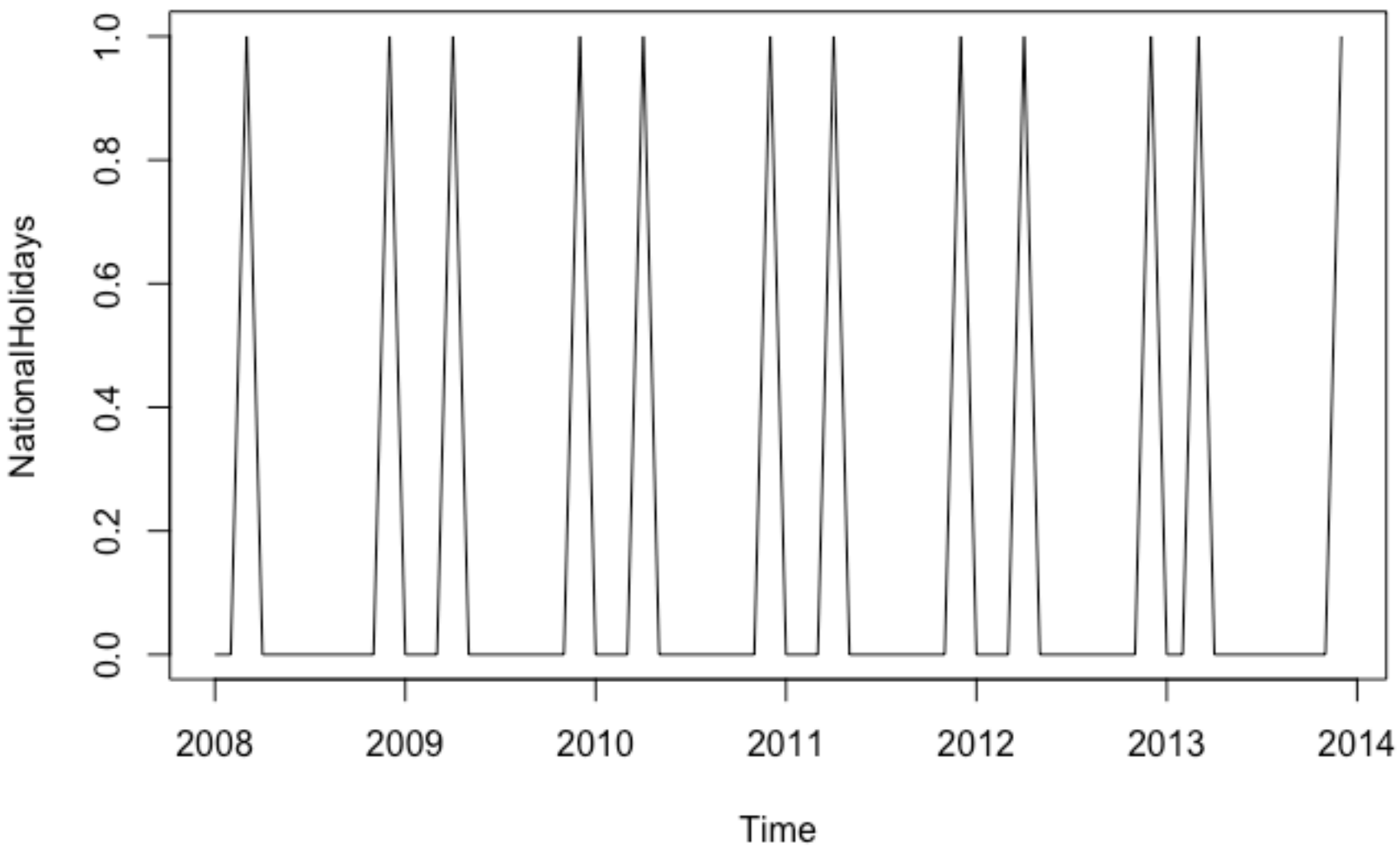
```
cor(RedEtelAsIs , PPIEtel)
```

```
## [1] 0.3391872
```

Total As Is and PPI Correlation shows moderate and positive correlation. (0.4836129)

National Holidays

```
NationalHolidaysVector <- c(ImportedIndicators[170:181,2],ImportedIndicators[170:181,
3],ImportedIndicators[170:181,4],ImportedIndicators[170:181,5],ImportedIndicators[170
:181,6],ImportedIndicators[170:181,7])
NationalHolidays <- ts(NationalHolidaysVector, start=c(2008,1), end=c(2013,12), frequ
ency=12)
plot(NationalHolidays, main="NationalHolidays")
```




```
cor(TotalAsIs, NationalHolidays)
```

```
## [1] -0.007883708
```

```
cor(EfakAsIs , NationalHolidays)
```

```
## [1] 0.001235706
```

```
cor(WugeAsIs, NationalHolidays)
```

```
## [1] 0.06505569
```

```
cor(TotalEtelAsIs, NationalHolidays)
```

```
## [1] -0.01081446
```

```
cor(BlueEtelAsIs , NationalHolidays)
```

```
## [1] 0.02903763
```

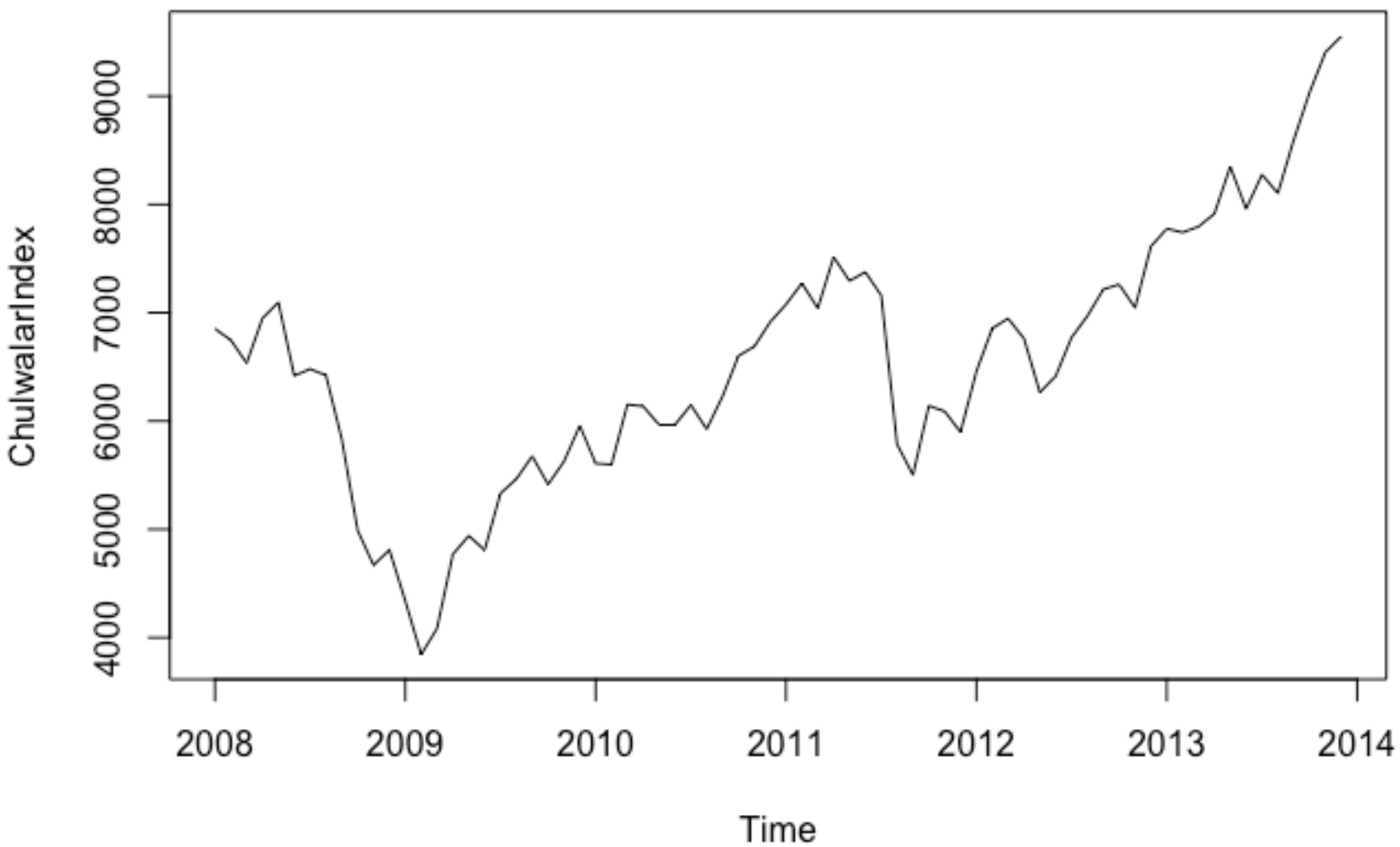
```
cor(RedEtelAsIs , NationalHolidays)
```

```
## [1] -0.01717636
```

Total As Is and National Holidays Correlation shows almost no correlation. (-0.007883708)

Chulwalar Index (Total value of all companies in Chulwalar)

```
ChulwalarIndexVector <- c(ImportedIndicators[128:139,2],ImportedIndicators[128:139,3]
,ImportedIndicators[128:139,4],ImportedIndicators[128:139,5],ImportedIndicators[128:1
39,6],ImportedIndicators[128:139,7])
ChulwalarIndex <- ts(ChulwalarIndexVector, start=c(2008,1), end=c(2013,12), frequency
=12)
plot(ChulwalarIndex, main="ChulwalarIndex")
```




```
cor(TotalAsIs, ChulwalarIndex)
```

```
## [1] 0.4837017
```

```
cor(EfakAsIs , ChulwalarIndex)
```

```
## [1] 0.7129557
```

```
cor(WugeAsIs, ChulwalarIndex)
```

```
## [1] 0.5721568
```

```
cor(TotalEtelAsIs, ChulwalarIndex)
```

```
## [1] 0.2209171
```

```
cor(BlueEtelAsIs , ChulwalarIndex)
```

```
## [1] 0.1469233
```

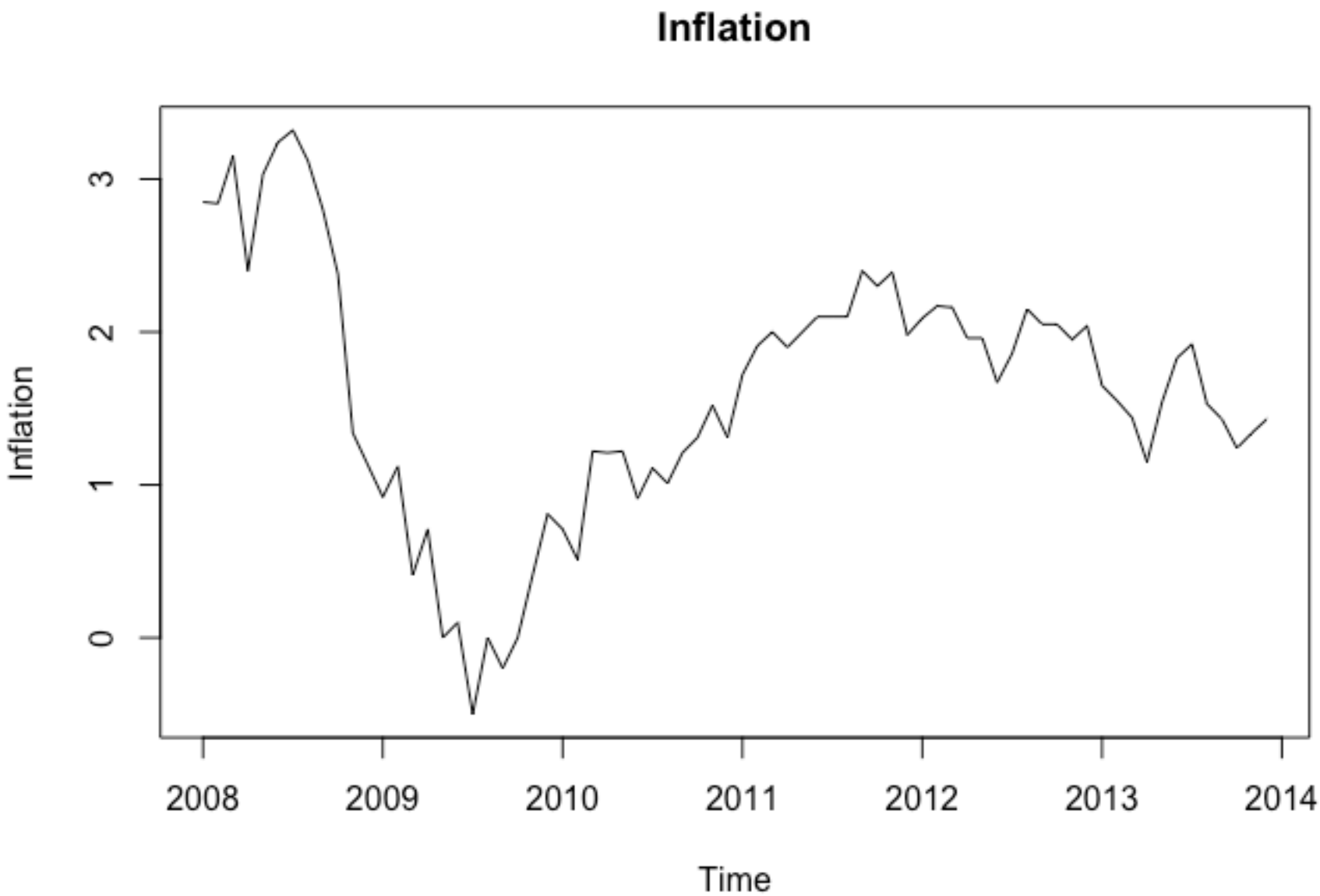
```
cor(RedEtelAsIs , ChulwalarIndex)
```

```
## [1] 0.2242922
```

Total As Is and ChulwalarIndex Correlation shows moderate and positive correlation. (0.4837017)

Monthly Inflation rate in Chulwalar

```
InflationVector <- c(ImportedIndicators[142:153,2],ImportedIndicators[142:153,3],Impo
rtedIndicators[142:153,4],ImportedIndicators[142:153,5],ImportedIndicators[142:153,6]
,ImportedIndicators[142:153,7])
Inflation <- ts(InflationVector, start=c(2008,1), end=c(2013,12), frequency=12)
plot(Inflation, main="Inflation")
```




```
cor(TotalAsIs, Inflation)
```

```
## [1] 0.002438708
```

```
cor(EfakAsIs , Inflation)
```

```
## [1] 0.1454134
```

```
cor(WugeAsIs, Inflation)
```

```
## [1] 0.03191332
```

```
cor(TotalEtelAsIs, Inflation)
```

```
## [1] -0.08378282
```

```
cor(BlueEtelAsIs , Inflation)
```

```
## [1] 0.02117817
```

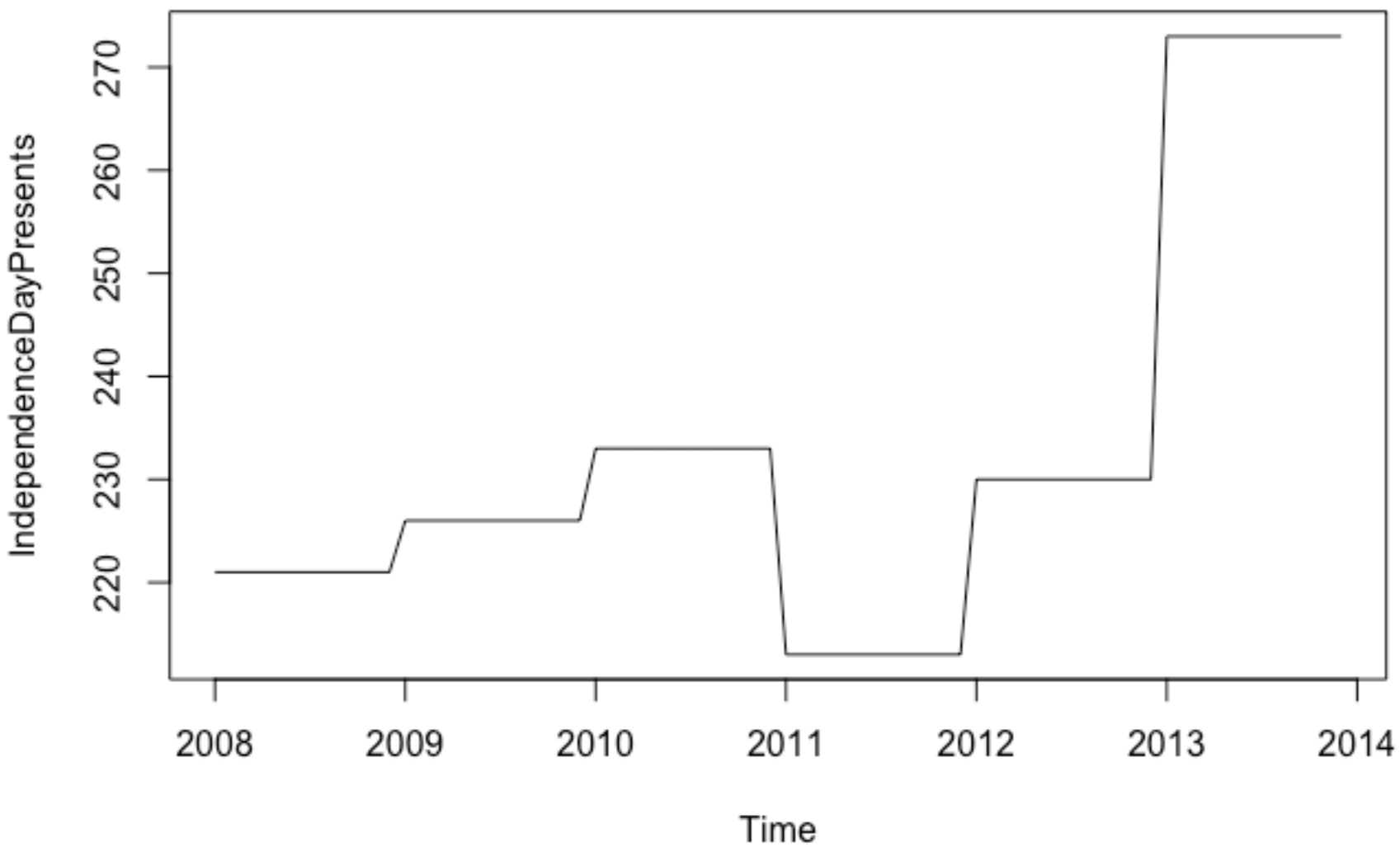
```
cor(RedEtelAsIs , Inflation)
```

```
## [1] -0.0982151
```

Total As Is and Inflation shows no correlation.(0.002438708)

Proposed spending for Independence day presents

```
IndependenceDayPresentsVector <- c(ImportedIndicators[156:167,2],ImportedIndicators[1
56:167,3],ImportedIndicators[156:167,4],ImportedIndicators[156:167,5],ImportedIndicat
ors[156:167,6],ImportedIndicators[156:167,7])
IndependenceDayPresents <- ts(IndependenceDayPresentsVector, start=c(2008,1), end=c(2
013,12), frequency=12)
plot(IndependenceDayPresents, main="IndependenceDayPresents")
```




```
cor(TotalAsIs, IndependenceDayPresents)
```

```
## [1] 0.4359522
```

```
cor(EfakAsIs , IndependenceDayPresents)
```

```
## [1] 0.5243145
```

```
cor(WugeAsIs, IndependenceDayPresents)
```

```
## [1] 0.4892437
```

```
cor(TotalEtelAsIs, IndependenceDayPresents)
```

```
## [1] 0.2872013
```

```
cor(BlueEtelAsIs , IndependenceDayPresents)
```

```
## [1] 0.2110373
```

```
cor(RedEtelAsIs , IndependenceDayPresents)
```

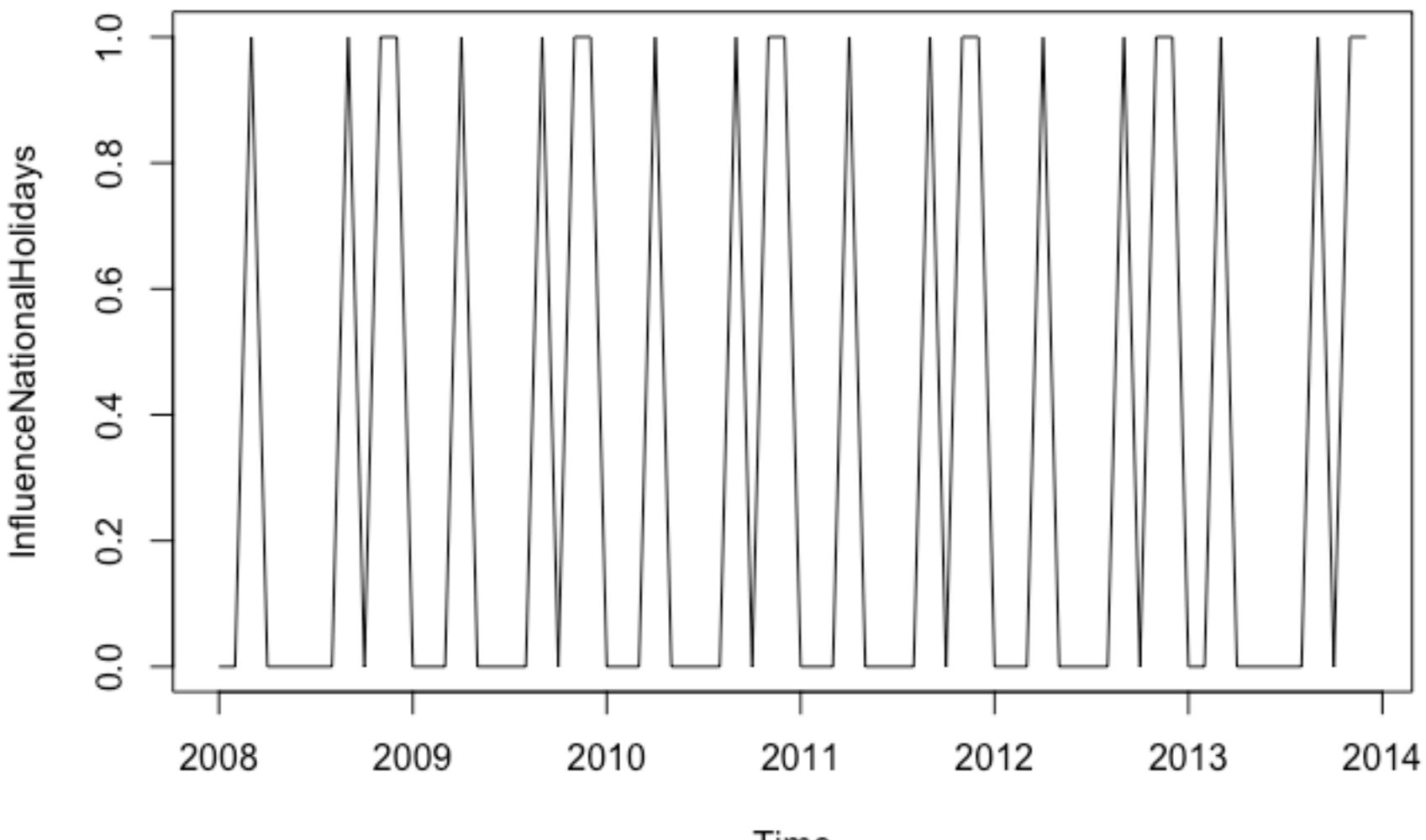
```
## [1] 0.2881631
```

Total As Is and Independence Day Presents spending Correlation shows moderate and positive correlation.(0.4359522)

Influence of National Holidays :

This indicator is an experiment where the influence of National Holidays is extended into the months leading up to the holiday. However later tests show that this indicator is no better for forecasting than the orignial National Holidays indicator.

```
InfluenceNationalHolidaysVector <- c(ImportedIndicators[184:195,2],ImportedIndicators
[184:195,3],ImportedIndicators[184:195,4],ImportedIndicators[184:195,5],ImportedIndic
ators[184:195,6],ImportedIndicators[184:195,7])
InfluenceNationalHolidays <- ts(InfluenceNationalHolidaysVector, start=c(2008,1), end
=c(2013,12), frequency=12)
plot(InfluenceNationalHolidays, main="InfluenceNationalHolidays")
```




```
cor(TotalAsIs, InfluenceNationalHolidays)
```

```
## [1] 0.3717463
```

```
cor(EfakAsIs , InfluenceNationalHolidays)
```

```
## [1] 0.09926836
```

```
cor(WugeAsIs, InfluenceNationalHolidays)
```

```
## [1] 0.3712288
```

```
cor(TotalEtelAsIs, InfluenceNationalHolidays)
```

```
## [1] 0.4535836
```

```
cor(BlueEtelAsIs , InfluenceNationalHolidays)
```

```
## [1] 0.2792198
```

```
cor(RedEtelAsIs , InfluenceNationalHolidays)
```

```
## [1] 0.4643512
```

Total As Is and Influence of National Holidays Correlation shows moderate correlation.(0.3717463)

Check that the data import has worked

```
str(CEPIVector)
```

```
##  num [1:72] 97.4 97.8 98.3 98.1 98.7 98.9 99.5 99.2 99.1 98.9 ...
```

```
str(SIGovVector)
```

```
##  num [1:72] -0.4 -2.9 -2.7 1.7 -1.7 -2.6 -7.1 -11.1 -9.4 -13.5 ...
```

```
str(TemperatureVector)
```

```
##  num [1:72] 3.6 3.7 4.2 7.6 14.5 16.9 18 17.4 12.4 9.1 ...
```

```
str(BirthsVector)
```

```
##  num [1:72] 58519 53370 52852 55048 57398 ...
```

```
str(SIExternVector)
```

```
##  num [1:72] 4.5 4.5 4.6 4.6 5 4.3 3.4 1.8 1.5 1.7 ...
```

```
str(UrbanoExportsVector)
```

```
##  num [1:72] 5850000 5850000 5850000 5850000 5850000 5850000 5850000 5850000 585000
0 5850000 ...
```

```
str(GlobalisationPartyMembersVector)
```

```
##  num [1:72] 45089 45089 45089 45089 45089 ...
```

```
str(AEPIVector)
```

```
##  num [1:72] 99 99.3 99.5 99.2 99.5 ...
```

```
str(PPIEtelVector)
```

```
##  num [1:72] 100.6 99.7 99.9 99.6 100 ...
```

```
str(NationalHolidaysVector)
```

```
##  num [1:72] 0 0 1 0 0 0 0 0 0 0 ...
```

```
str(ChulwalarIndexVector)
```

```
##  num [1:72] 6852 6748 6535 6949 7097 ...
```

```
str(InflationVector)
```

```
##  num [1:72] 2.85 2.84 3.15 2.4 3.03 3.24 3.32 3.12 2.8 2.38 ...
```

```
str(IndependenceDayPresentsVector)
```

```
##  num [1:72] 221 221 221 221 221 221 221 221 221 221 ...
```

Forecasting models with smoothing and related approaches

Exponential Smoothing uses past values to calculate a forecast. The strength with which each value influences the forecast is weakened with help of a smoothing parameter. Thus we are dealing with a weighted average, whose values fade out the longer ago they were in the past.

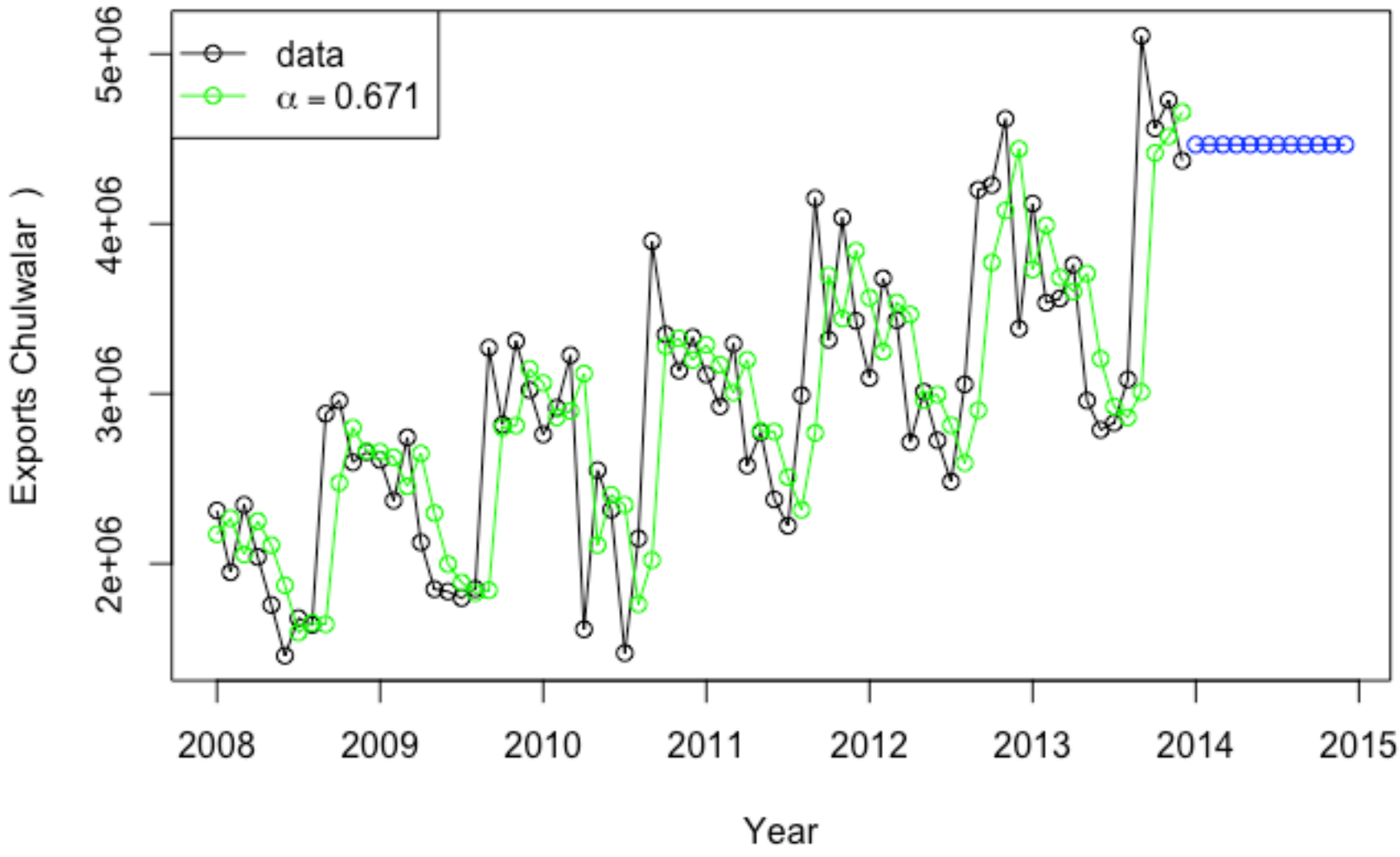
The Akaike's Information Criterion(AIC/AICc) or the Bayesian Information Criterion (BIC) should be at minimum.

Simple expontential smoothing

```
Model_ses <- ses(TotalAsIs, h=12)
summary(Model_ses)
```

```
##
## Forecast method: Simple exponential smoothing
##
## Model Information:
## Simple exponential smoothing
##
## Call:
##  ses(x = TotalAsIs, h = 12)
##
##   Smoothing parameters:
##     alpha = 0.671
##
##   Initial states:
##     l = 2173226.7433
##
##   sigma:  609507
##
##      AIC     AICc      BIC
## 2230.058 2230.232 2234.612
##
## Error measures:
##                     ME   RMSE      MAE       MPE     MAPE     MASE
## Training set 47469.84 609507 429997.1 -1.511008 15.02336 1.172074
##                     ACF1
## Training set 0.02384493
##
## Forecasts:
##          Point Forecast   Lo 80   Hi 80   Lo 95   Hi 95
## Jan 2014        4466448 3685333 5247562 3271836 5661059
## Feb 2014        4466448 3525801 5407094 3027853 5905042
## Mar 2014        4466448 3389650 5543245 2819628 6113267
## Apr 2014        4466448 3268880 5664015 2634926 6297969
## May 2014        4466448 3159220 5773675 2467215 6465680
## Jun 2014        4466448 3058072 5874823 2312524 6620371
## Jul 2014        4466448 2963718 5969177 2168221 6764674
## Aug 2014        4466448 2874947 6057948 2032458 6900437
## Sep 2014        4466448 2790873 6142022 1903878 7029017
## Oct 2014        4466448 2710821 6222074 1781448 7151447
## Nov 2014        4466448 2634263 6298632 1664363 7268532
## Dec 2014        4466448 2560778 6372117 1551977 7380918
```

```
plot(Model_ses, plot.conf=FALSE, ylab="Exports Chulwalar  )", xlab="Year", main="", f
col="white", type="o")
lines(fitted(Model_ses), col="green", type="o")
lines(Model_ses$mean, col="blue", type="o")
legend("topleft",lty=1, col=c(1,"green"), c("data", expression(alpha == 0.671)),pch=1
)
```



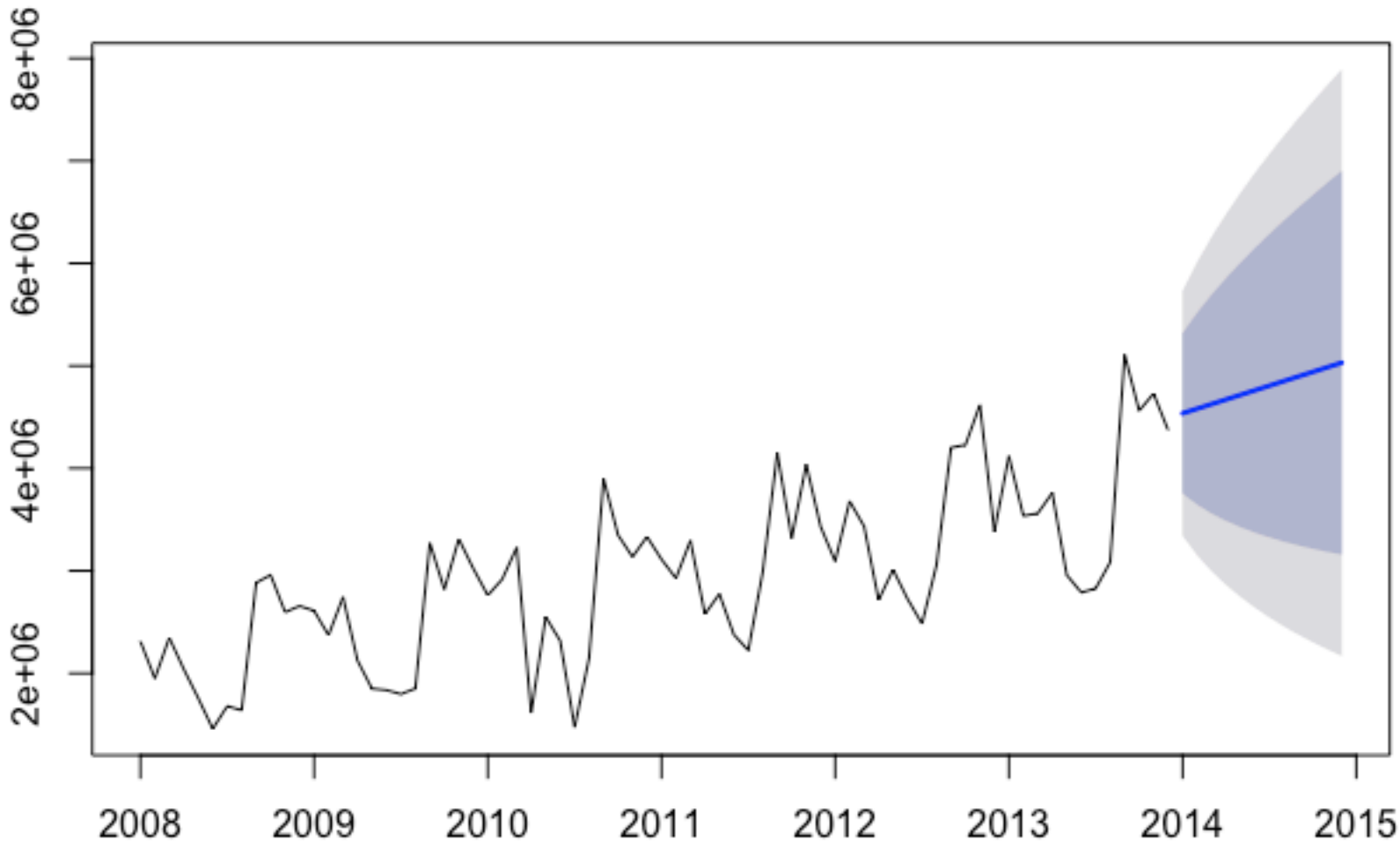
Holt's linear trend method

Holt added to the model in order to forecast using trends as well. For this it is necessary to add a beta, which determines the trend. If neither alpha nor beta is stated, both parameters will be optimised using ets(). The trend is exponential if the intercepts(level) and the gradient (slope) are multiplied with eachother. The values are worse. As the Beta was very low in the optimisation, the forecast is very similar to the ses() model.

```
Model_holt_1 <- holt(TotalAsIs,h=12)
summary(Model_holt_1)
```

```
##
## Forecast method: Holt's method
##
## Model Information:
## Holt's method
##
## Call:
##  holt(x = TotalAsIs, h = 12)
##
##   Smoothing parameters:
##     alpha = 0.6571
##     beta  = 1e-04
##
##   Initial states:
##     l = 2040390.7764
##     b = 45050.7514
##
##   sigma:  608119.1
##
##      AIC     AICc      BIC
## 2233.730 2234.327 2242.837
##
## Error measures:
##                     ME     RMSE      MAE       MPE     MAPE     MASE
## Training set -16586.9 608119.1 441110.7 -3.88925 15.75307 1.202367
##                     ACF1
## Training set 0.03462672
##
## Forecasts:
##          Point Forecast   Lo 80   Hi 80   Lo 95   Hi 95
## Jan 2014        4536367 3757031 5315703 3344475 5728259
## Feb 2014        4581298 3648703 5513894 3155016 6007580
## Mar 2014        4626230 3562188 5690271 2998918 6253541
## Apr 2014        4671161 3490181 5852141 2865008 6477314
## May 2014        4716092 3428721 6003463 2747228 6684956
## Jun 2014        4761024 3375378 6146669 2641862 6880185
## Jul 2014        4805955 3328531 6283379 2546429 7065480
## Aug 2014        4850886 3287035 6414738 2459182 7242591
## Sep 2014        4895818 3250047 6541588 2378829 7412807
## Oct 2014        4940749 3216925 6664573 2304387 7577111
## Nov 2014        4985680 3187164 6784196 2235088 7736273
## Dec 2014        5030612 3160363 6900860 2170314 7890909
```

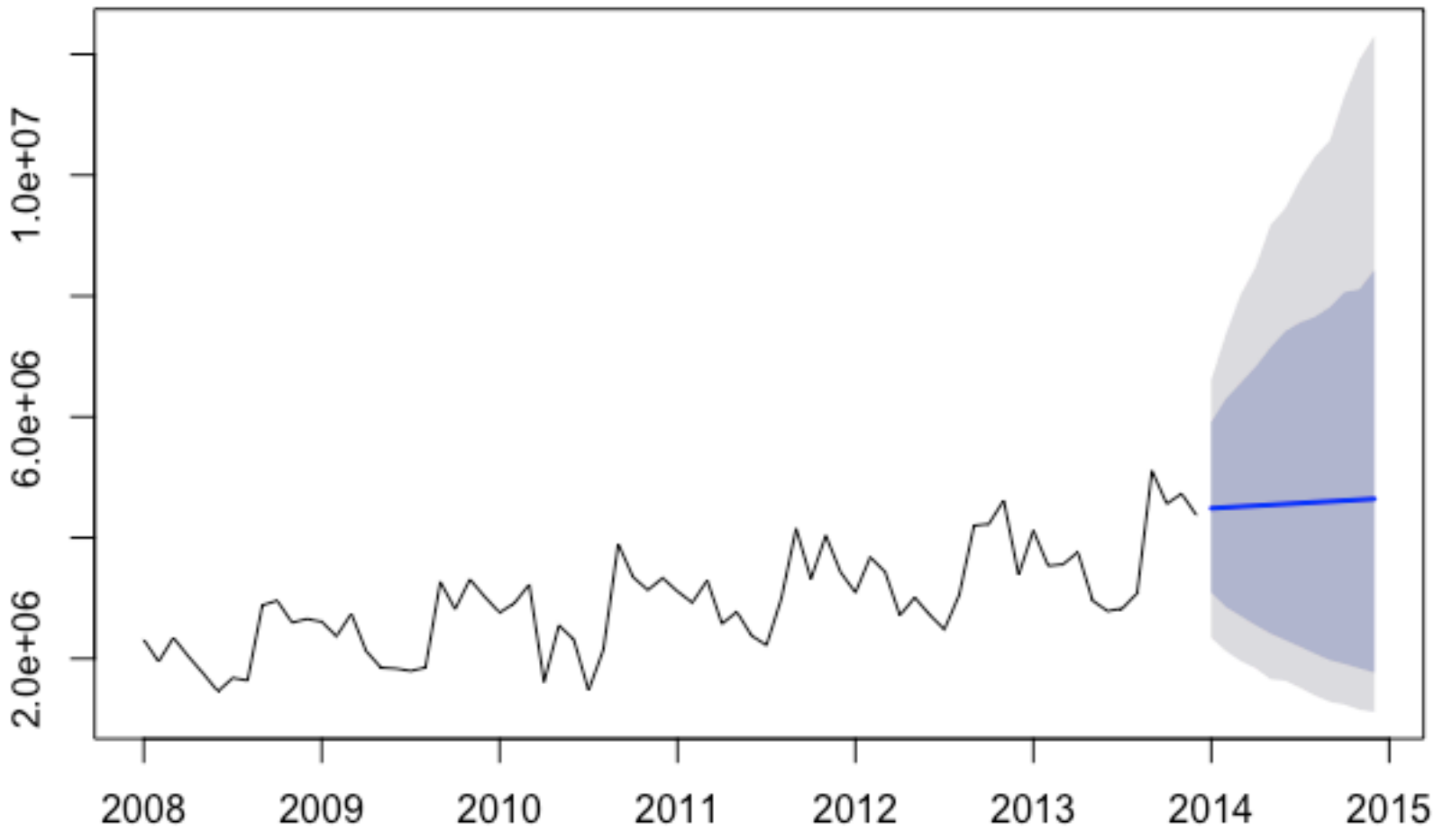
```
plot(Model_holt_1)
```




```
# expoential trend
Model_holt_2<- holt(TotalAsIs, exponential=TRUE,h=12)
summary(Model_holt_2)
```

```
##
## Forecast method: Holt's method with exponential trend
##
## Model Information:
## Holt's method with exponential trend
##
## Call:
##  holt(x = TotalAsIs, h = 12, exponential = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.6637
##     beta  = 1e-04
##
##   Initial states:
##     l = 2041538.9468
##     b = 1.0029
##
##   sigma:  0.2438
##
##      AIC     AICc      BIC
## 2251.010 2251.607 2260.116
##
## Error measures:
##                    ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 37825.61 609787.5 433018.9 -1.838214 15.18487 1.180311
##                    ACF1
## Training set 0.02918287
##
## Forecasts:
##          Point Forecast   Lo 80   Hi 80   Lo 95    Hi 95
## Jan 2014        4488281 3094898 5910687 2349620  6603924
## Feb 2014        4502175 2856202 6299434 2127480  7367867
## Mar 2014        4516113 2706599 6560865 1962547  8027730
## Apr 2014        4530094 2554679 6828071 1842829  8474630
## May 2014        4544118 2419440 7145669 1661460  9172500
## Jun 2014        4558186 2308140 7414707 1637562  9449500
## Jul 2014        4572297 2199659 7559619 1520535  9922793
## Aug 2014        4586452 2083781 7651099 1392637 10301999
## Sep 2014        4600650 1978251 7814131 1288887 10570402
## Oct 2014        4614893 1914094 8062570 1242499 11309487
## Nov 2014        4629180 1843426 8106689 1157210 11911356
## Dec 2014        4643510 1772119 8427091 1121709 12300372
```

```
plot(Model_holt_2)
```



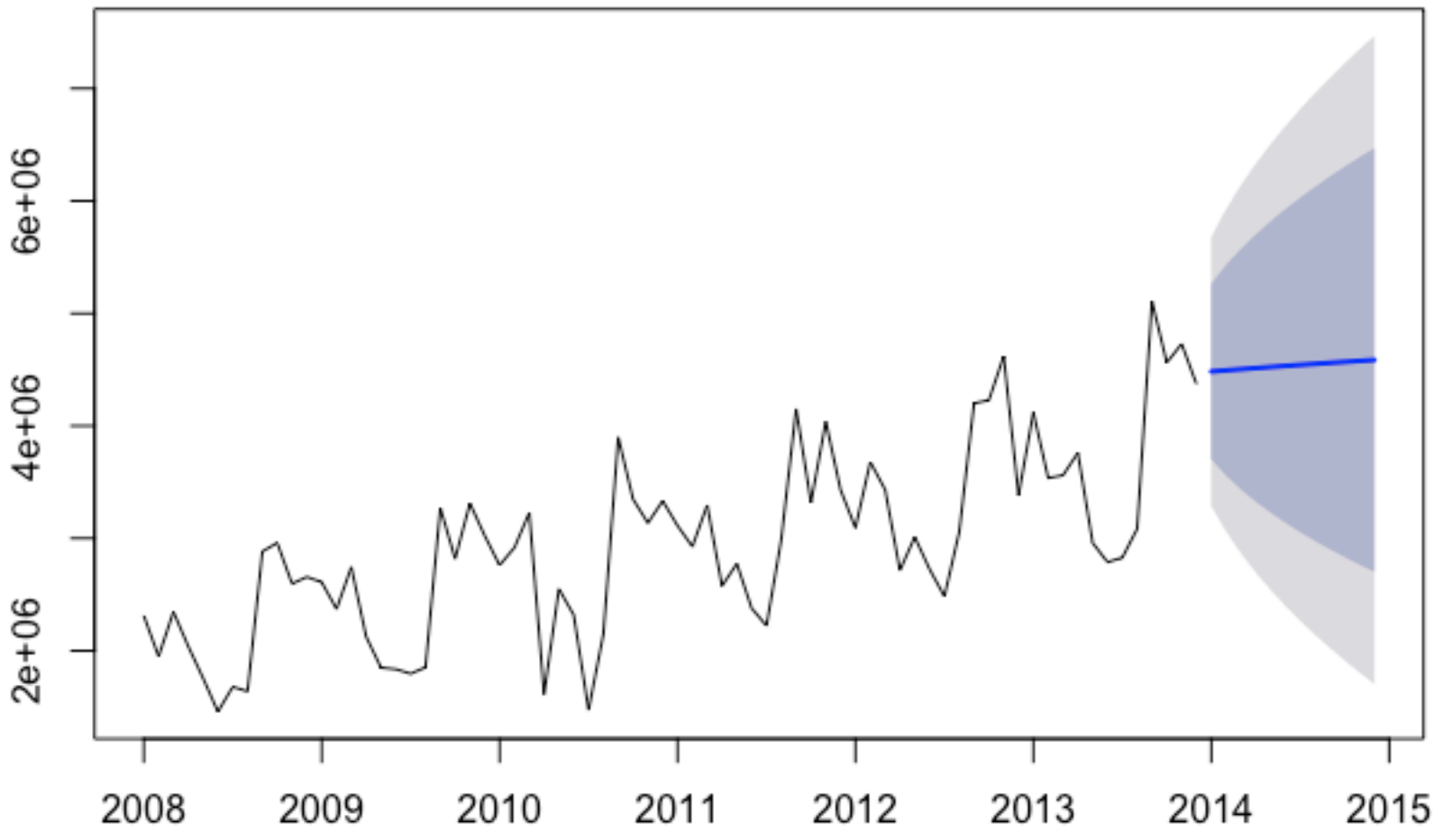
Dampened trends

As such simple trends tend to forecast the future to positively, we have added a dampener. This also works for exponential trends. We also plot the level and slope individually for each model.

```
Model_holt_3 <- holt(TotalAsIs, damped=TRUE,h=12)
summary(Model_holt_3)
```

```
##
## Forecast method: Damped Holt's method
##
## Model Information:
## Damped Holt's method
##
## Call:
##  holt(x = TotalAsIs, h = 12, damped = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.6613
##     beta  = 2e-04
##     phi   = 0.98
##
##   Initial states:
##     l = 2040392.5761
##     b = 45053.25
##
##   sigma:  608787.2
##
##      AIC     AICc      BIC
## 2235.888 2236.797 2247.272
##
## Error measures:
##                    ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 15578.94 608787.2 436909.7 -2.797612 15.46526 1.190916
##                    ACF1
## Training set 0.03351419
##
## Forecasts:
##          Point Forecast   Lo 80   Hi 80   Lo 95   Hi 95
## Jan 2014        4483618 3703426 5263811 3290417 5676819
## Feb 2014        4493914 3558436 5429391 3063224 5924603
## Mar 2014        4504003 3435520 5572486 2869899 6138107
## Apr 2014        4513891 3327168 5700614 2698955 6328827
## May 2014        4523581 3229332 5817829 2544198 6502963
## Jun 2014        4533077 3139534 5926619 2401837 6664316
## Jul 2014        4542383 3056128 6028638 2269352 6815413
## Aug 2014        4551503 2977955 6125051 2144969 6958036
## Sep 2014        4560440 2904162 6216719 2027381 7093499
## Oct 2014        4569199 2834101 6304298 1915595 7222803
## Nov 2014        4577783 2767264 6388301 1808834 7346732
## Dec 2014        4586195 2703249 6469141 1706477 7465913
```

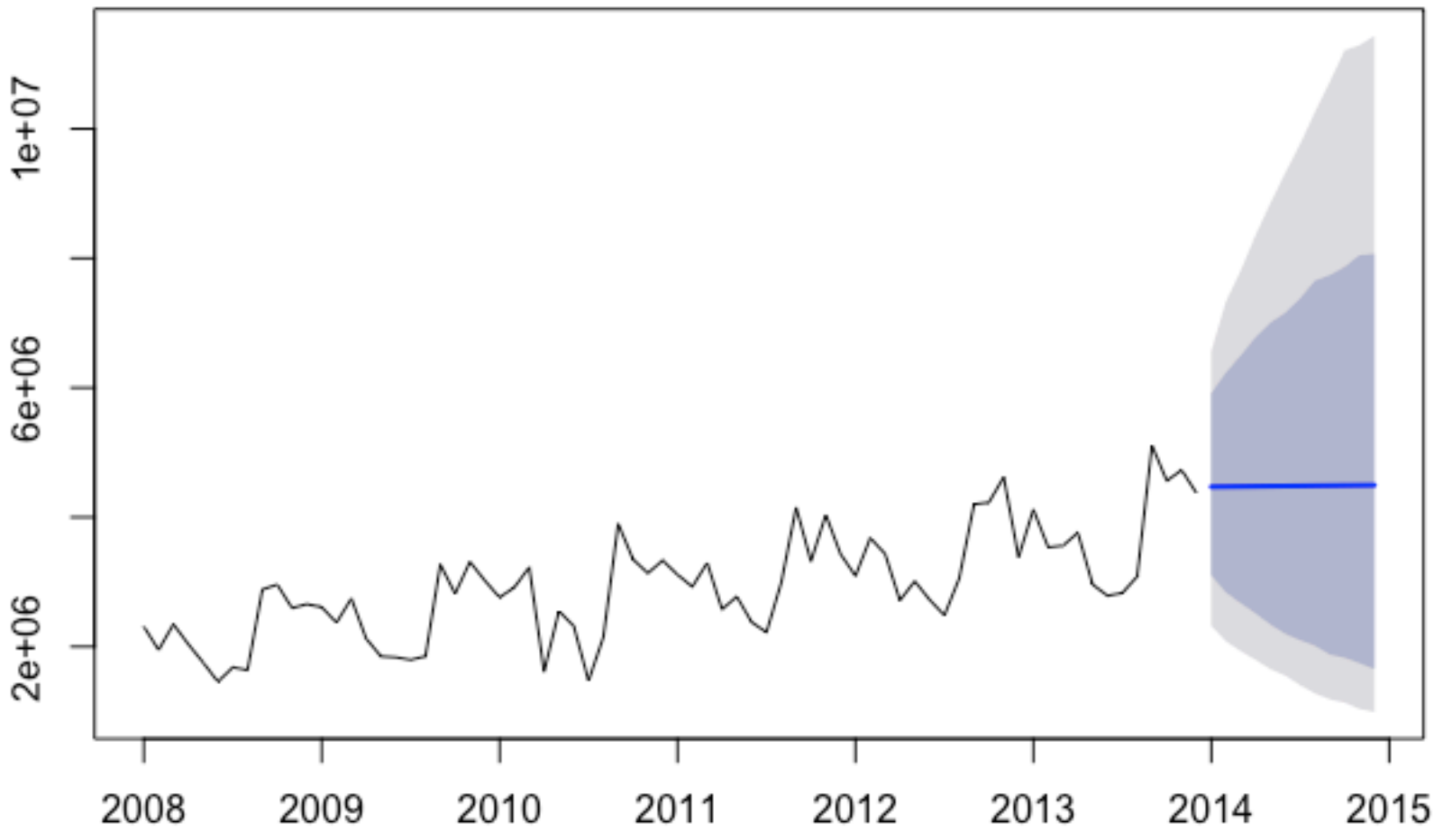
```
plot(Model_holt_3)
```



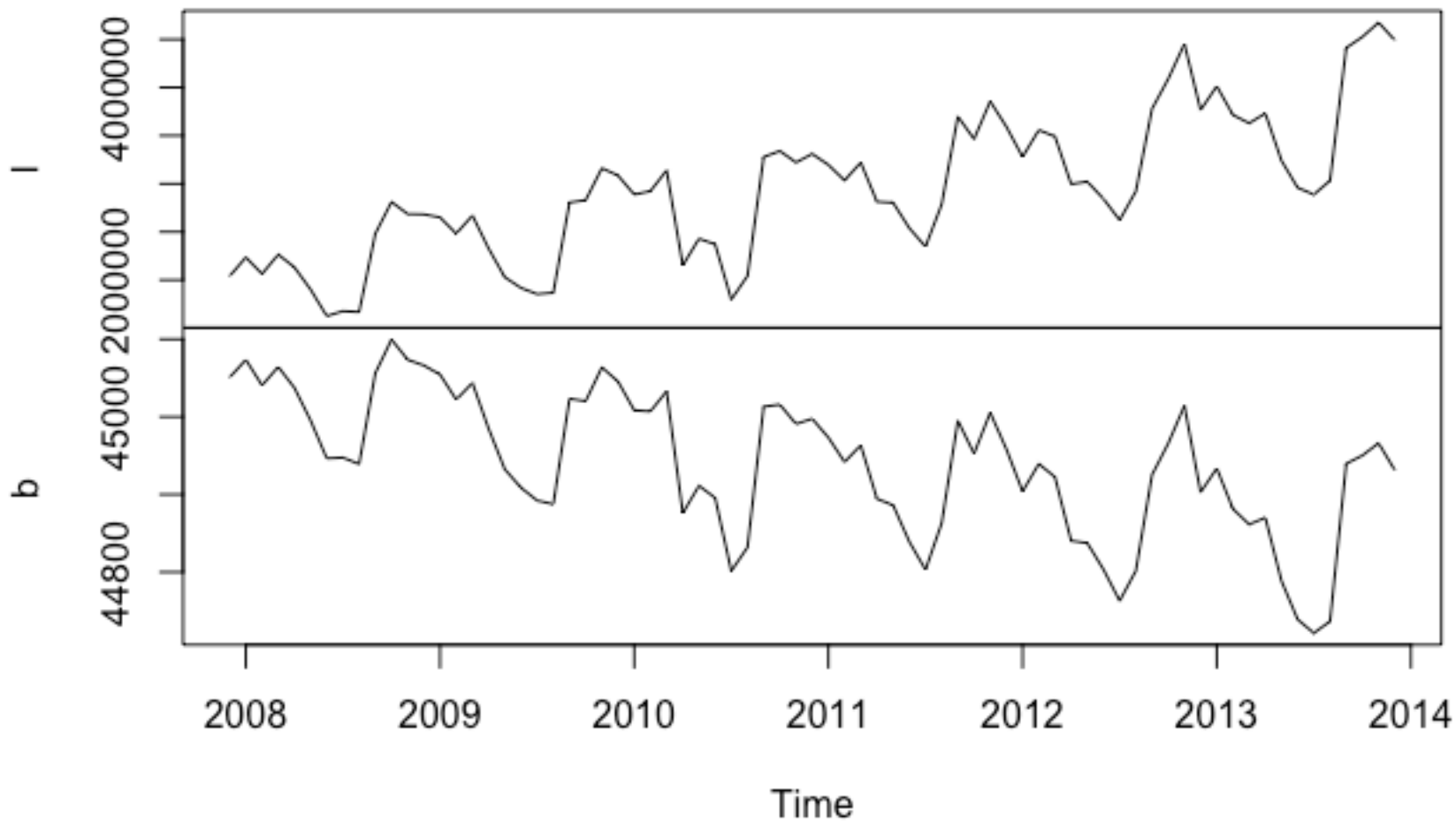

```
Model_holt_4 <- holt(TotalAsIs, exponential=TRUE, damped=TRUE,h=12)
summary(Model_holt_4)
```

```
##
## Forecast method: Damped Holt's method with exponential trend
##
## Model Information:
## Damped Holt's method with exponential trend
##
## Call:
##  holt(x = TotalAsIs, h = 12, damped = TRUE, exponential = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.6679
##     beta  = 1e-04
##     phi   = 0.9799
##
##   Initial states:
##     l = 2041541.9705
##     b = 1.0019
##
##   sigma:  0.2449
##
##      AIC     AICc      BIC
## 2253.216 2254.125 2264.600
##
## Error measures:
##                    ME     RMSE      MAE       MPE     MAPE     MASE
## Training set 46119.56 609906.7 432069.1 -1.549114 15.11987 1.177722
##                   ACF1
## Training set 0.0254941
##
## Forecasts:
##          Point Forecast   Lo 80   Hi 80     Lo 95    Hi 95
## Jan 2014        4470648 3105903 5911918 2325890.3  6555924
## Feb 2014        4473164 2840943 6232758 2084779.2  7329443
## Mar 2014        4475630 2673928 6497669 1933483.2  7809752
## Apr 2014        4478047 2514949 6779636 1798334.8  8358043
## May 2014        4480418 2345182 7004960 1655057.4  8850377
## Jun 2014        4482742 2201289 7155064 1556028.7  9318394
## Jul 2014        4485020 2102177 7379142 1408443.7  9762813
## Aug 2014        4487253 2018119 7652447 1280581.3 10255600
## Sep 2014        4489443 1881373 7737737 1189358.2 10723781
## Oct 2014        4491589 1831755 7863969 1139258.4 11209287
## Nov 2014        4493694 1749502 8045763 1040915.5 11288776
## Dec 2014        4495757 1656484 8066075  995324.8 11432763
```

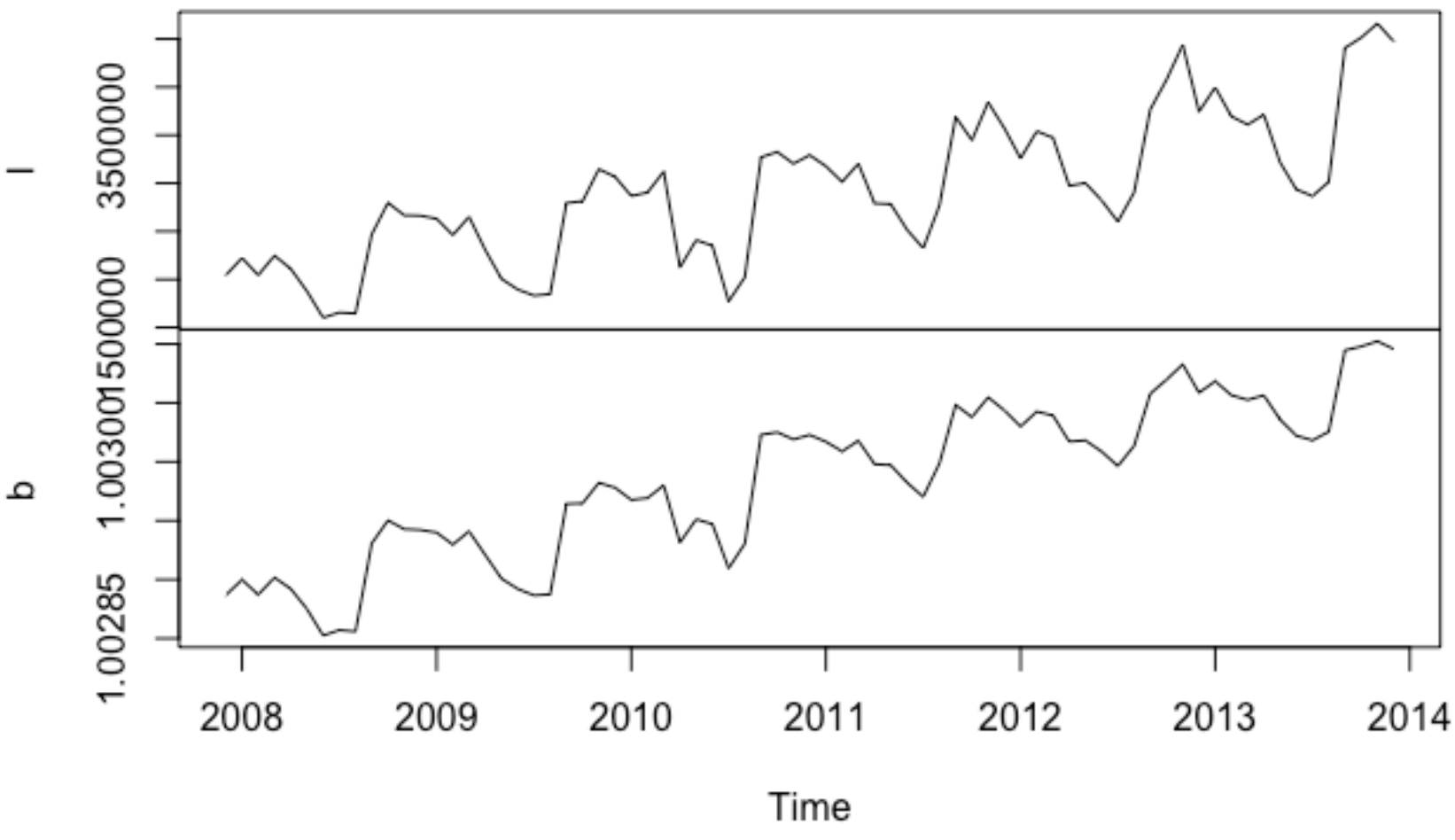
```
plot(Model_holt_4)
```



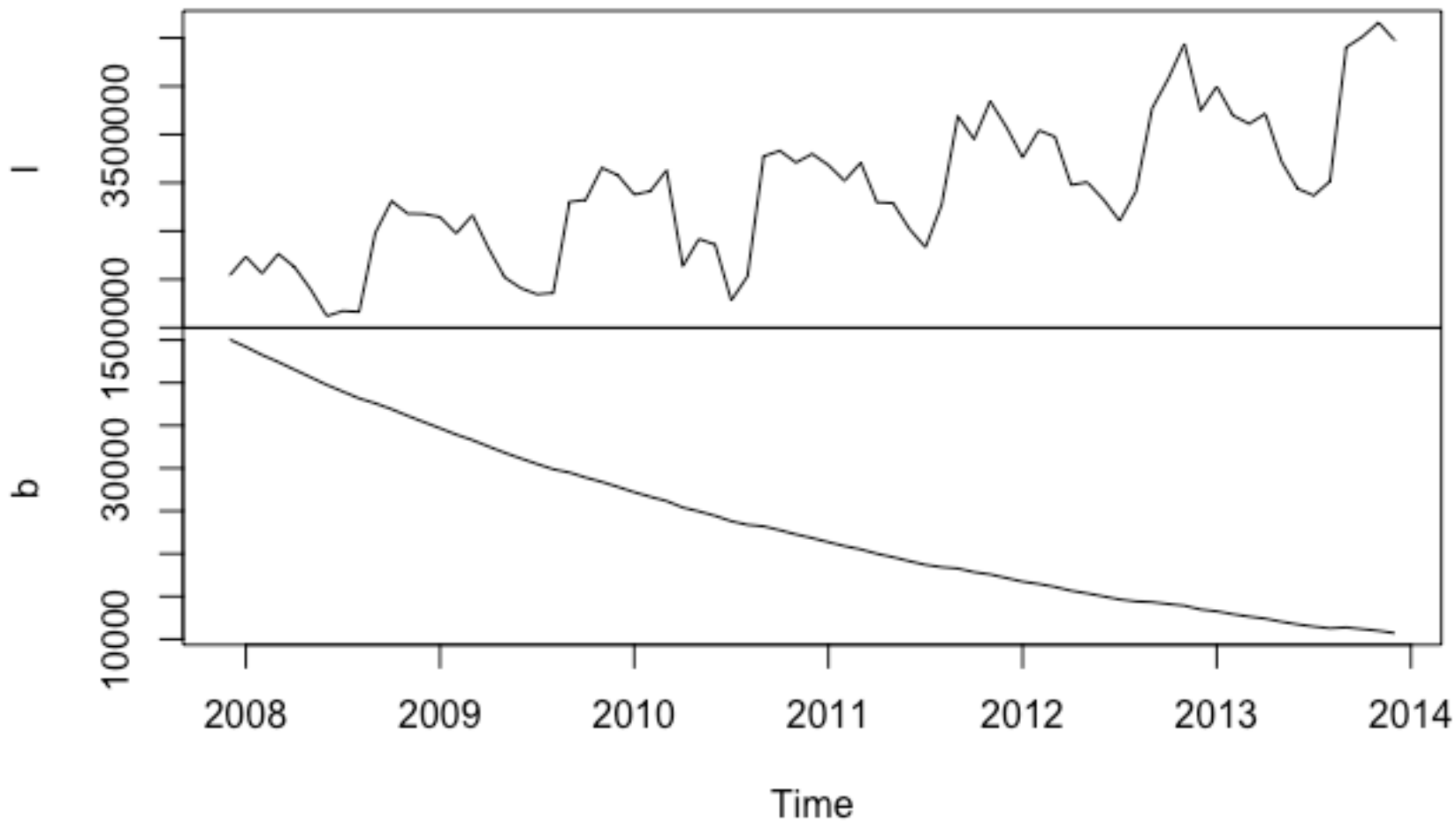

```
# level and slope can be plotted individually for each model.
plot(Model_holt_1$model$state)
```



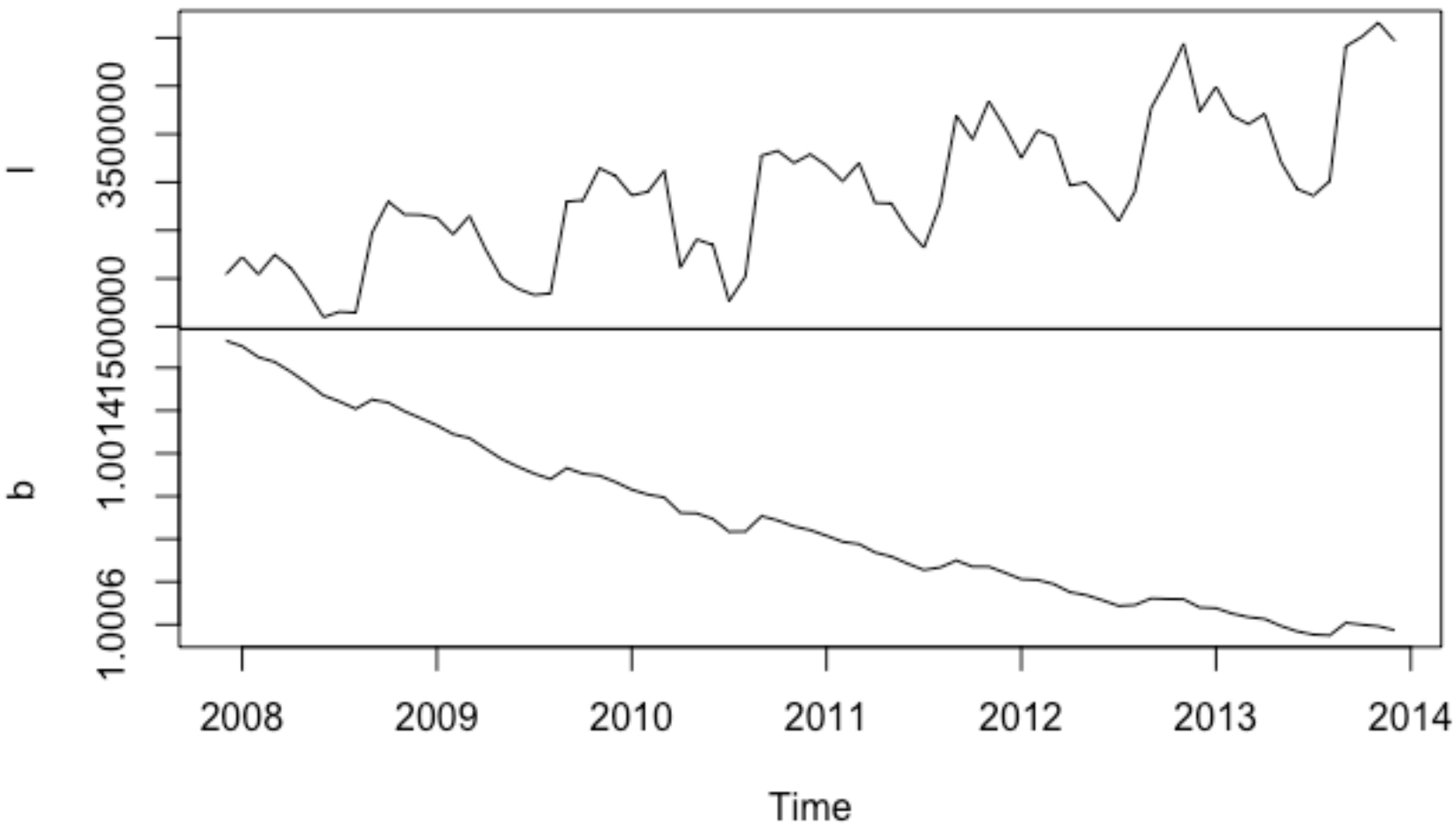

```
plot(Model_holt_2$model$state)
```




```
plot(Model_holt_3$model$state)
```

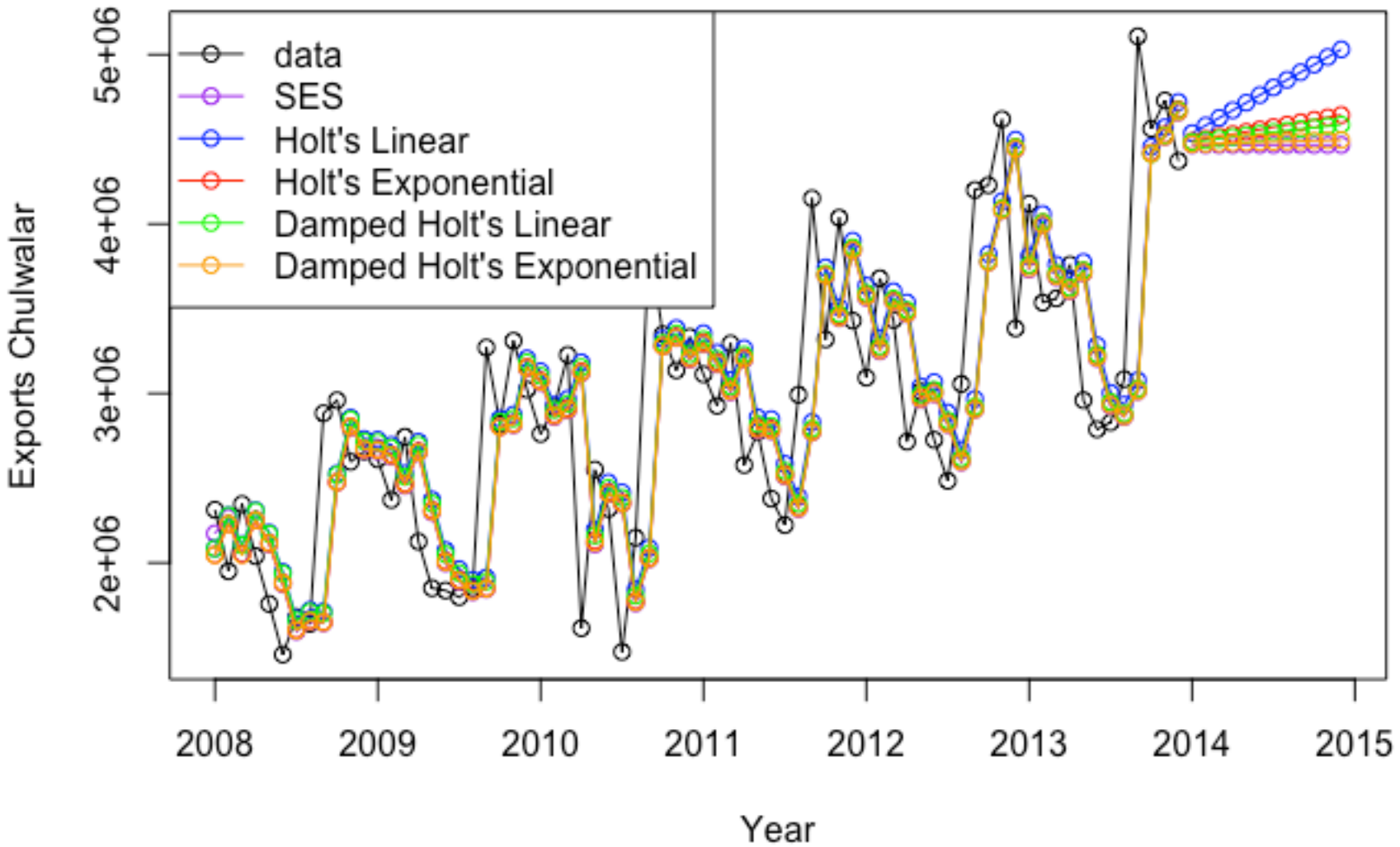



```
plot(Model_holt_4$model$state)
```




```
plot(Model_holt_1, plot.conf=FALSE, ylab="Exports Chulwalar", xlab="Year", main="Fore
casts for Non-Seasonal Methods", fcol="white", type="o")

lines(fitted(Model_ses), col="purple", type="o")
lines(fitted(Model_holt_1), col="blue", type="o")
lines(fitted(Model_holt_2), col="red", type="o")
lines(fitted(Model_holt_3), col="green", type="o")
lines(fitted(Model_holt_4), col="orange", type="o")
lines(Model_ses$mean, col="purple", type="o")
lines(Model_holt_1$mean, col="blue", type="o")
lines(Model_holt_2$mean, col="red", type="o")
lines(Model_holt_3$mean, col="green", type="o")
lines(Model_holt_4$mean, col="orange", type="o")
legend("topleft",lty=1, col=c(1,"purple","blue","red","green","orange"), c("data", "S
ES","Holt's Linear", "Holt's Exponential", "Damped Holt's Linear", "Damped Holt's Exp
onential"),pch=1)
```



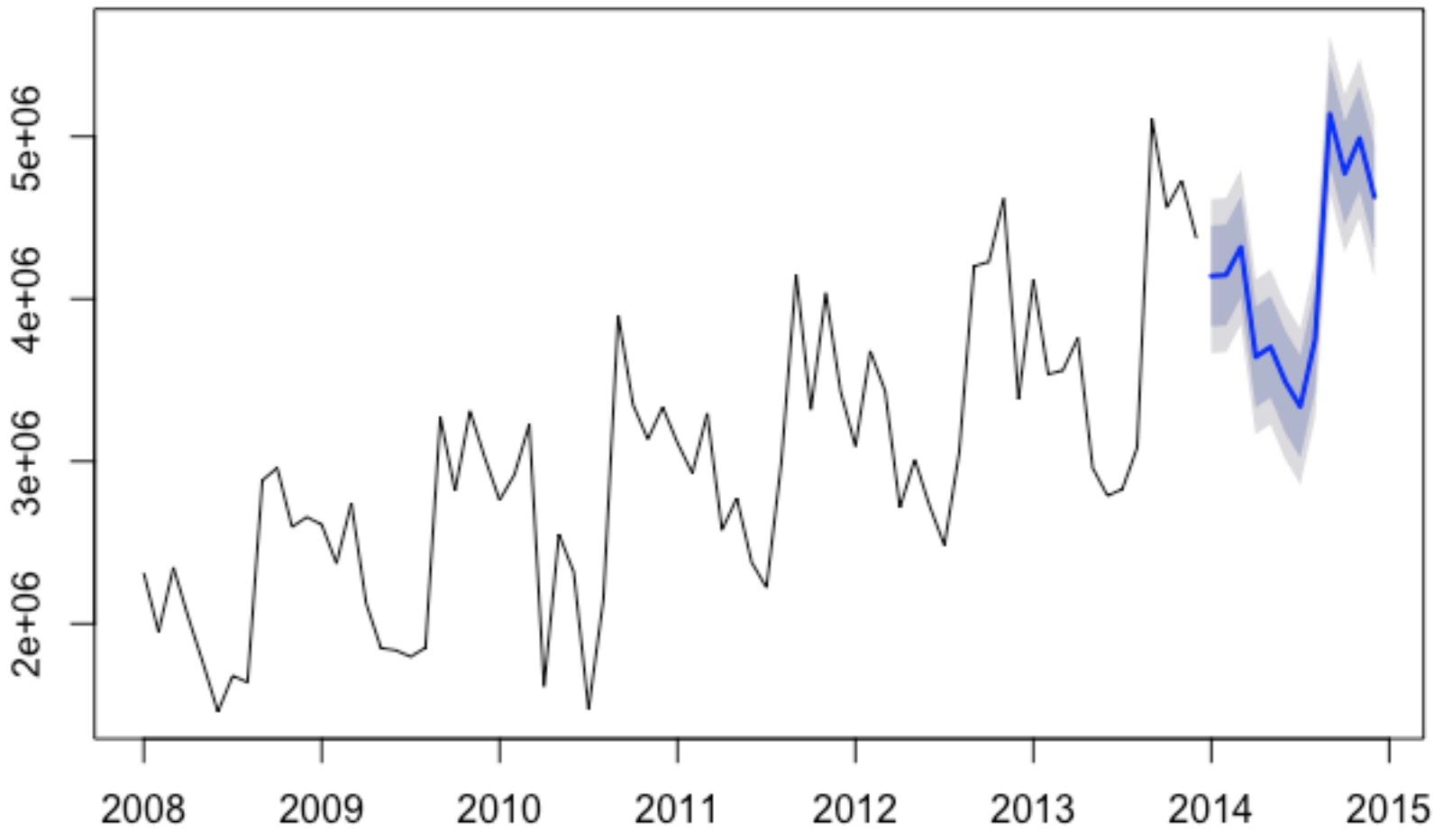
Holt-Winter’s seasonal method

Holt and Winters have expanded Holt’s model further to include the seasonality aspect. The parameter gamma, which is for smoothing the seasonality, was added to achieve this. The values are better than the models without seasonality. This is logical, since the data is strongly influenced by seasonality. In the following model, none of the parameters are given so that they will be optimised automatically. There are two models: one using an additive error model method and one using a multiplicative error model. The additive model gives slightly better results than the multiplicative model.

```
Model_hw_1 <- hw(TotalAsIs ,seasonal="additive",h=12)
summary(Model_hw_1)
```

```
##
## Forecast method: Holt-Winters' additive method
##
## Model Information:
## Holt-Winters' additive method
##
## Call:
##  hw(x = TotalAsIs, h = 12, seasonal = "additive")
##
##   Smoothing parameters:
##     alpha = 0.0087
##     beta  = 0.0087
##     gamma = 1e-04
##
##   Initial states:
##     l = 2047375.0884
##     b = 22509.7631
##     s=259168.3 654942.6 474529.8 876025.2 -475155 -852844
##            -664662.5 -412596.7 -438677.3 273215 138077.9 167976.7
##
##   sigma:  241685
##
##      AIC     AICc      BIC
## 2124.856 2134.747 2161.283
##
## Error measures:
##                     ME   RMSE      MAE         MPE     MAPE      MASE
## Training set 21615.43 241685 202218.5 -0.08252109 7.329458 0.5512016
##                     ACF1
## Training set -0.2819072
##
## Forecasts:
##          Point Forecast   Lo 80   Hi 80   Lo 95   Hi 95
## Jan 2014        4141204 3831472 4450936 3667510 4614898
## Feb 2014        4147309 3837472 4457147 3673453 4621165
## Mar 2014        4318537 4008512 4628563 3844394 4792680
## Apr 2014        3642744 3332425 3953063 3168153 4117335
## May 2014        3704865 3394124 4015605 3229628 4180102
## Jun 2014        3488859 3177546 3800173 3012746 3964973
## Jul 2014        3336738 3024677 3648799 2859482 3813994
## Aug 2014        3750478 3437474 4063482 3271780 4229176
## Sep 2014        5137771 4823607 5451935 4657298 5618244
## Oct 2014        4772337 4456775 5087900 4289726 5254949
## Nov 2014        4988809 4671591 5306028 4503665 5473953
## Dec 2014        4629097 4309943 4948252 4140992 5117202
```

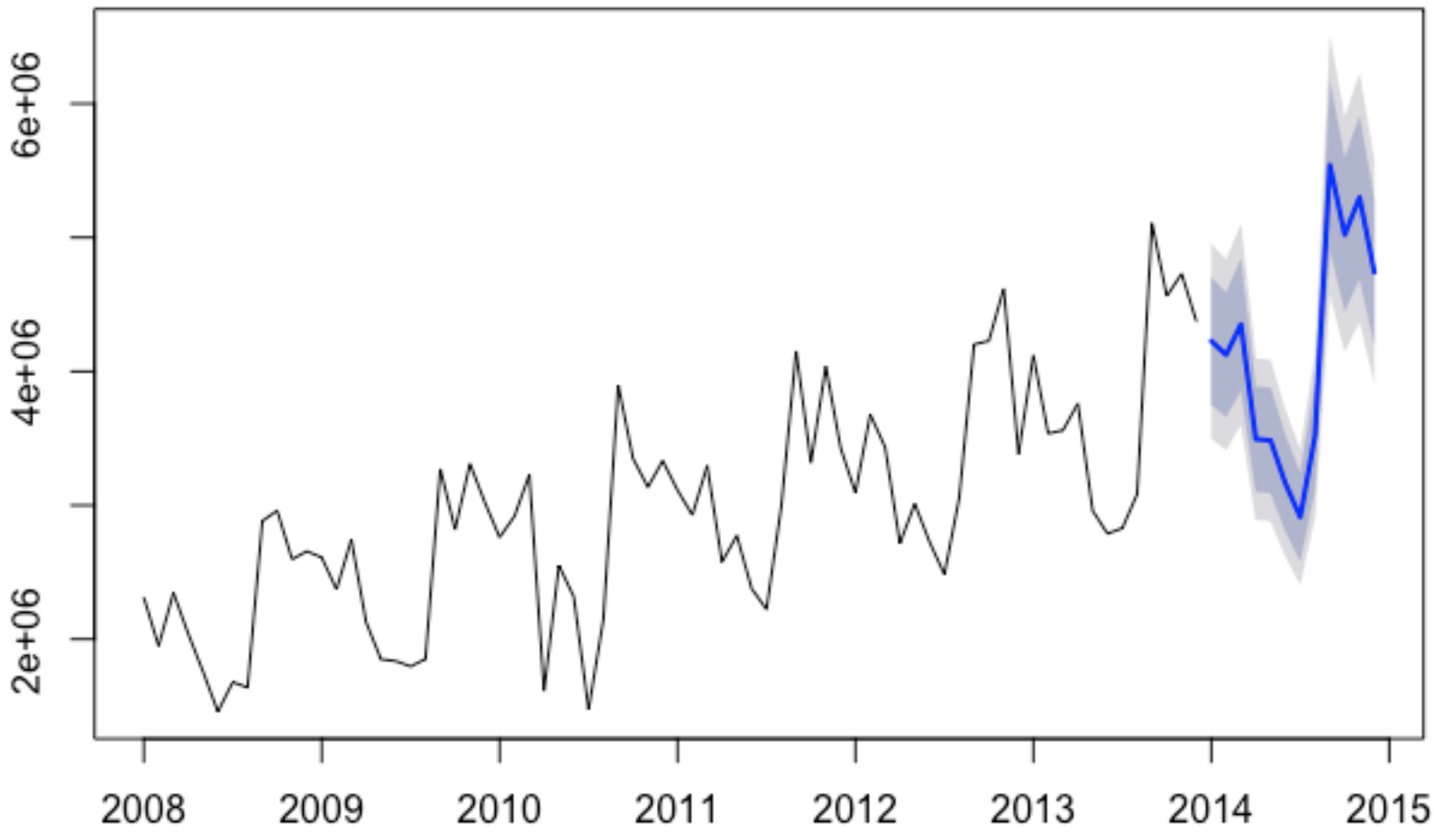
```
plot(Model_hw_1)
```



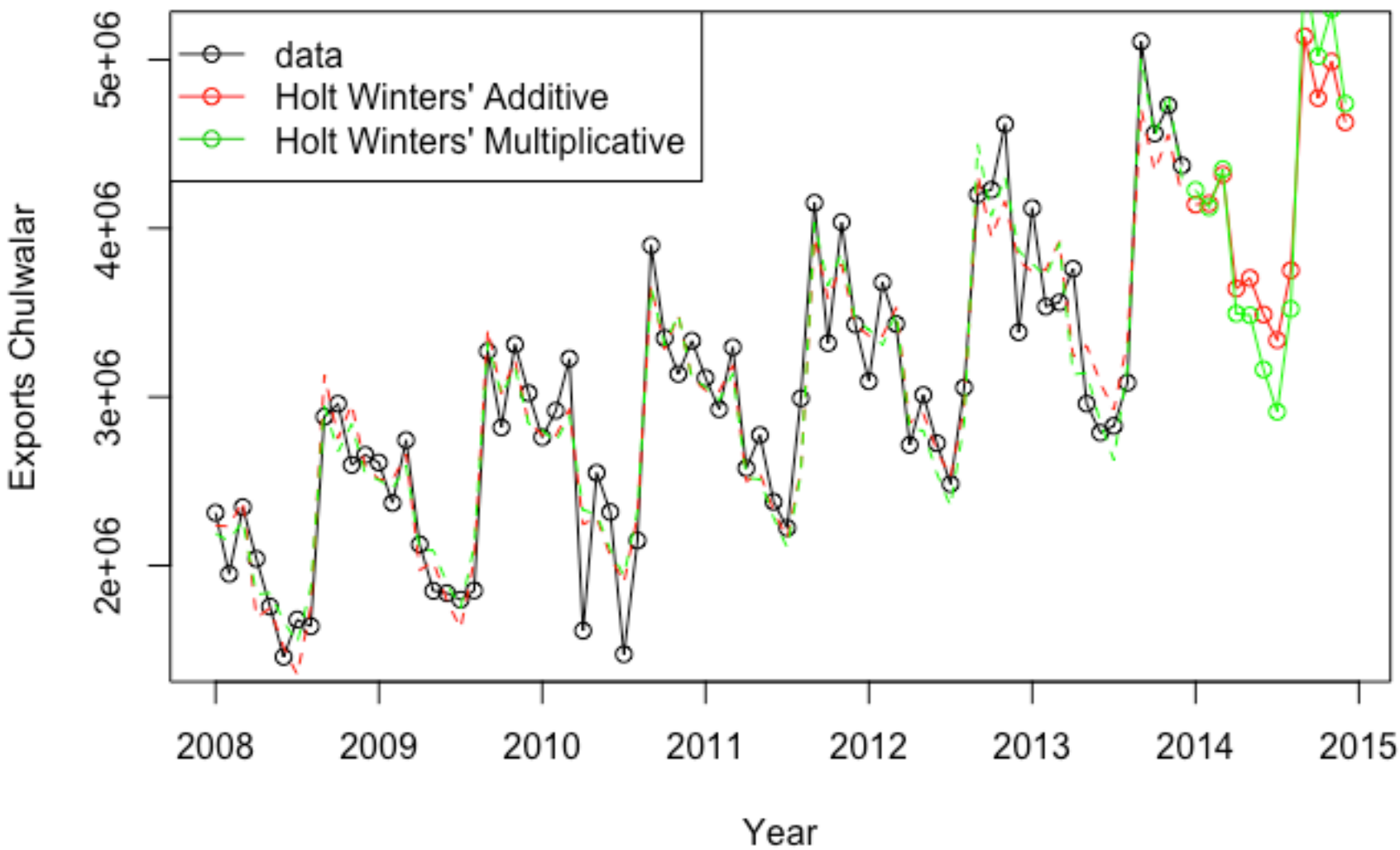

```
Model_hw_2 <- hw(TotalAsIs ,seasonal="multiplicative",h=12)
summary(Model_hw_2)
```

```
##
## Forecast method: Holt-Winters' multiplicative method
##
## Model Information:
## Holt-Winters' multiplicative method
##
## Call:
##  hw(x = TotalAsIs, h = 12, seasonal = "multiplicative")
##
##   Smoothing parameters:
##     alpha = 0.025
##     beta  = 0.0062
##     gamma = 1e-04
##
##   Initial states:
##     l = 2026247.531
##     b = 25395.1259
##     s=1.0933 1.232 1.1763 1.3086 0.8384 0.699
##            0.7653 0.8502 0.8596 1.0793 1.0316 1.0665
##
##   sigma:  0.0877
##
##      AIC     AICc      BIC
## 2128.303 2138.194 2164.729
##
## Error measures:
##                     ME     RMSE      MAE         MPE     MAPE      MASE
## Training set 17434.11 235296.6 191805.3 -0.3292809 7.213472 0.5228175
##                     ACF1
## Training set -0.3514421
##
## Forecasts:
##          Point Forecast   Lo 80   Hi 80   Lo 95   Hi 95
## Jan 2014        4226941 3751624 4702258 3500006 4953876
## Feb 2014        4123665 3659738 4587591 3414151 4833179
## Mar 2014        4350808 3860995 4840620 3601704 5099911
## Apr 2014        3494208 3100476 3887940 2892046 4096370
## May 2014        3484738 3091618 3877858 2883513 4085963
## Jun 2014        3162774 2805463 3520085 2616314 3709234
## Jul 2014        2912399 2582802 3241996 2408324 3416474
## Aug 2014        3521645 3122278 3921013 2910865 4132425
## Sep 2014        5540988 4911109 6170867 4577671 6504304
## Oct 2014        5020487 4448200 5592775 4145249 5895725
## Nov 2014        5299729 4693715 5905743 4372911 6226547
## Dec 2014        4740169 4196230 5284108 3908286 5572052
```

```
plot(Model_hw_2)
```




```
plot(Model_hw_1, ylab="Exports Chulwalar", plot.conf=FALSE, type="o", fcol="white", x
lab="Year",main="Holt-Winters' Seasonal Methods")
lines(fitted(Model_hw_1), col="red", lty=2)
lines(fitted(Model_hw_2), col="green", lty=2)
lines(Model_hw_1$mean, type="o", col="red")
lines(Model_hw_2$mean, type="o", col="green")
legend("topleft",lty=1, pch=1, col=1:3, c("data","Holt Winters' Additive","Holt Winte
rs' Multiplicative"))
```




```
# In order to use the results later, they need to be converted into point forcasts.
Model_hw_1_df <-as.data.frame(Model_hw_1)
Model_hw_1_PointForecast <- ts(Model_hw_1_df$"Point Forecast", start=c(2014,1), end=c
(2014,12), frequency=12)
Model_hw_1_PointForecast
```

```
##          Jan     Feb     Mar     Apr     May     Jun     Jul     Aug
## 2014 4141204 4147309 4318537 3642744 3704865 3488859 3336738 3750478
##          Sep     Oct     Nov     Dec
## 2014 5137771 4772337 4988809 4629097
```

```
Model_hw_2_df <-as.data.frame(Model_hw_2)
Model_hw_2_PointForecast <- ts(Model_hw_2_df$"Point Forecast", start=c(2014,1), end=c
(2014,12), frequency=12)
Model_hw_2_PointForecast
```

```
##          Jan     Feb     Mar     Apr     May     Jun     Jul     Aug
## 2014 4226941 4123665 4350808 3494208 3484738 3162774 2912399 3521645
##          Sep     Oct     Nov     Dec
## 2014 5540988 5020487 5299729 4740169
```

```
# Output instruction for the data export of the results for further use in Excel.
write.csv(Model_hw_1_PointForecast,file='./Data/Model_hw_1_PointForecast.csv')
write.csv(Model_hw_2_PointForecast,file='./Data/Model_hw_2_PointForecast.csv')
```

Use accuracy measurements to determine which method is best.

The additive seasonal method was better for AIC, AICc, BIC, and MPE while the multiplicative method was better for RMSE, MAE, MAPE, and MASE.

Summarize results.

After performing the analysis, the Holt-Winter's multiplicative seasonal method was found to be best suited for predicting Chulwalar's exports. While the multiplicative seasonal method was more accurate by some measurements (RMSE, MAE, MAPE, and MASE), preference is given to RMSE, MAE, MAPE, and MASE, which the multiplicative seasonal method was found to be more accurate for. Advice to the Prime Minister of Chulwalar would be to use that method as guidance when planning exports.