US.Level.Only.Total.OPEX.Forecast

Jason Marshall

August 7, 2018

library(forecast)  
library(tidyquant)

## Loading required package: lubridate

##   
## Attaching package: 'lubridate'

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

## Loading required package: PerformanceAnalytics

## Loading required package: xts

## Loading required package: zoo

##   
## Attaching package: 'zoo'

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

##   
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':  
##   
## legend

## Loading required package: quantmod

## Loading required package: TTR

## Version 0.4-0 included new data defaults. See ?getSymbols.

## Loading required package: tidyverse

## -- Attaching packages ---------------------------------------------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 2.2.1 v purrr 0.2.4  
## v tibble 1.4.2 v dplyr 0.7.4  
## v tidyr 0.8.0 v stringr 1.2.0  
## v readr 1.1.1 v forcats 0.2.0

## -- Conflicts ------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x lubridate::as.difftime() masks base::as.difftime()  
## x lubridate::date() masks base::date()  
## x dplyr::filter() masks stats::filter()  
## x dplyr::first() masks xts::first()  
## x lubridate::intersect() masks base::intersect()  
## x dplyr::lag() masks stats::lag()  
## x dplyr::last() masks xts::last()  
## x lubridate::setdiff() masks base::setdiff()  
## x lubridate::union() masks base::union()

##   
## Attaching package: 'tidyquant'

## The following object is masked from 'package:dplyr':  
##   
## as\_tibble

## The following object is masked from 'package:tibble':  
##   
## as\_tibble

library(timetk)  
library(sweep)  
library(tidyverse)  
library(broom)  
library(timeDate)

##   
## Attaching package: 'timeDate'

## The following objects are masked from 'package:PerformanceAnalytics':  
##   
## kurtosis, skewness

library(reshape2)

##   
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':  
##   
## smiths

library(data.table)

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:reshape2':  
##   
## dcast, melt

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

## The following object is masked from 'package:purrr':  
##   
## transpose

## The following objects are masked from 'package:xts':  
##   
## first, last

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

library(readr)  
library(DataExplorer)  
library(Hmisc)

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

##   
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':  
##   
## src, summarize

## The following object is masked from 'package:quantmod':  
##   
## Lag

## The following objects are masked from 'package:base':  
##   
## format.pval, units

library(caret)

##   
## Attaching package: 'caret'

## The following object is masked from 'package:survival':  
##   
## cluster

## The following object is masked from 'package:purrr':  
##   
## lift

library(quantreg)

## Loading required package: SparseM

##   
## Attaching package: 'SparseM'

## The following object is masked from 'package:base':  
##   
## backsolve

##   
## Attaching package: 'quantreg'

## The following object is masked from 'package:Hmisc':  
##   
## latex

## The following object is masked from 'package:survival':  
##   
## untangle.specials

# #load expense data  
# dataset.1 <- read\_csv("Y:/Sharepoint Projects/Jason Marshall/R\_Projects/OPEX\_Forecasting/Data/OpEx Expenses v2 - South.csv",  
# col\_types = cols(Amount = col\_number()))  
#   
# dataset.2 <- read\_csv("Y:/Sharepoint Projects/Jason Marshall/R\_Projects/OPEX\_Forecasting/Data/OpEx Expenses v2 - US.csv",  
# col\_types = cols(Amount = col\_number()))  
#   
# dataset.3 <- read\_csv("Y:/Sharepoint Projects/Jason Marshall/R\_Projects/OPEX\_Forecasting/Data/OpEx Expenses v2 - WCBT.csv",  
# col\_types = cols(Amount = col\_number()))  
#   
# dataset.4 <- read\_csv("Y:/Sharepoint Projects/Jason Marshall/R\_Projects/OPEX\_Forecasting/Data/OpEx Expenses v2 - West.csv",  
# col\_types = cols(Amount = col\_number()))  
#   
# dataset.5 <- read\_csv("Y:/Sharepoint Projects/Jason Marshall/R\_Projects/OPEX\_Forecasting/Data/OpEx Expenses v2 - CCBT.csv",  
# col\_types = cols(Amount = col\_number()))  
#   
# dataset.6 <- read\_csv("Y:/Sharepoint Projects/Jason Marshall/R\_Projects/OPEX\_Forecasting/Data/OpEx Expenses v2 - East.csv",  
# col\_types = cols(Amount = col\_number()))  
#   
# dataset.7 <- read\_csv("Y:/Sharepoint Projects/Jason Marshall/R\_Projects/OPEX\_Forecasting/Data/OpEx Expenses v2 - ECBT.csv",  
# col\_types = cols(Amount = col\_number()))  
#   
# #combine both datasets  
# comb.dataset <- bind\_rows(dataset.1, dataset.2, dataset.3, dataset.4, dataset.5, dataset.6, dataset.7)  
#   
# #fill NA  
# comb.dataset$Amount <- replace\_na(comb.dataset$Amount, replace = 0)  
#   
# #make unique col names  
# colsNamesShelf <-colnames(comb.dataset)  
# dataset\_col\_names <- make.names(colsNamesShelf, unique = TRUE)  
# colnames(comb.dataset) <- dataset\_col\_names  
#   
# #add date column  
# comb.dataset <- comb.dataset %>%  
# mutate(day.holder = 01) %>%  
# mutate(date.holder = paste(Year, Acct.Per, day.holder, sep = "-"))  
#   
# #set date.holder as date  
# comb.dataset$date.holder <- as.Date(comb.dataset$date.holder)  
#   
# #get last day of the month  
# comb.dataset$Date <- as.Date(timeDate::timeLastDayInMonth(comb.dataset$date.holder))  
#   
# #write to disk  
# write.csv(comb.dataset,"Y:/Sharepoint Projects/Jason Marshall/R\_Projects/OPEX\_Forecasting/Data/US.opex.test.full.csv")  
  
####################################################################################  
expense.dataset <- read\_csv("Y:/Sharepoint Projects/Jason Marshall/R\_Projects/OPEX\_Forecasting/Data/US.opex.test.full.csv")

## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:  
## cols(  
## X1 = col\_integer(),  
## GL.Level = col\_character(),  
## Summary.Account = col\_character(),  
## Parent.Account = col\_character(),  
## Account = col\_character(),  
## Geography = col\_character(),  
## Region = col\_character(),  
## Division = col\_character(),  
## Period = col\_character(),  
## Acct.Per = col\_integer(),  
## Year = col\_integer(),  
## Currency = col\_character(),  
## Amount = col\_integer(),  
## day.holder = col\_integer(),  
## date.holder = col\_date(format = ""),  
## Date = col\_date(format = "")  
## )

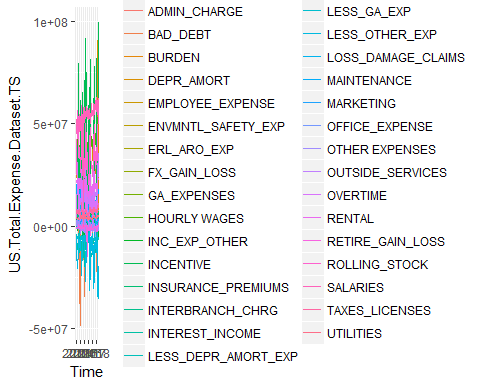
## Warning in rbind(names(probs), probs\_f): number of columns of result is not  
## a multiple of vector length (arg 1)

## Warning: 32 parsing failures.  
## row # A tibble: 5 x 5 col row col expected actual file expected <int> <chr> <chr> <chr> <chr> actual 1 1005 Amount no trailing characters e+05 'Y:/Sharepoint Projects/Jaso~ file 2 15347 Amount no trailing characters e+05 'Y:/Sharepoint Projects/Jaso~ row 3 17619 Amount no trailing characters e+05 'Y:/Sharepoint Projects/Jaso~ col 4 30257 Amount no trailing characters e+05 'Y:/Sharepoint Projects/Jaso~ expected 5 35440 Amount no trailing characters e+05 'Y:/Sharepoint Projects/Jaso~  
## ... ................. ... .......................................................................... ........ .......................................................................... ...... .......................................................................... .... .......................................................................... ... .......................................................................... ... .......................................................................... ........ ..........................................................................  
## See problems(...) for more details.

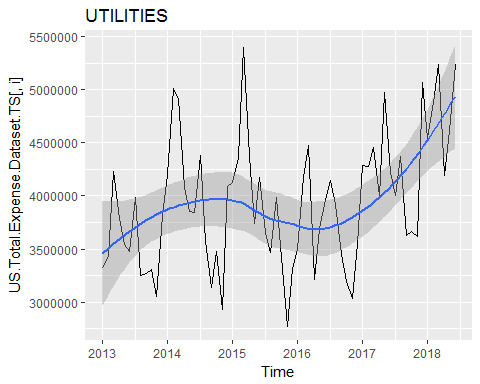
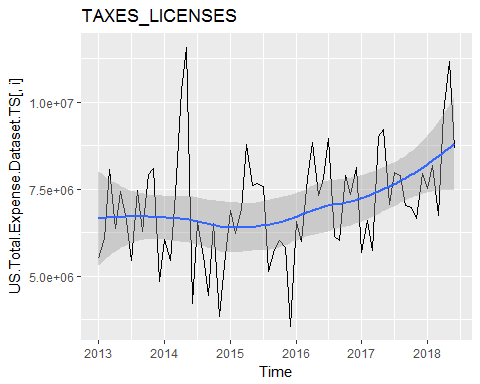
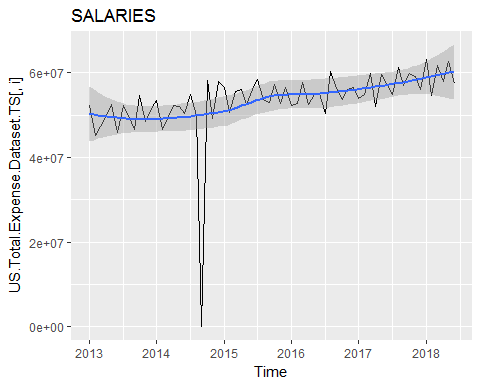
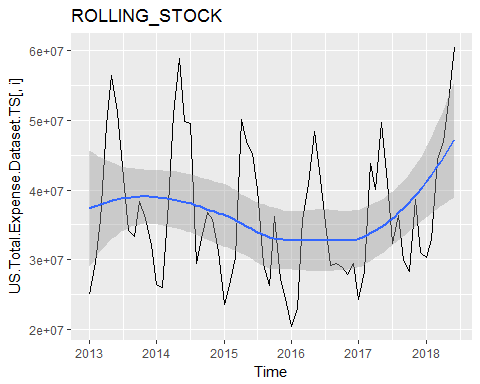
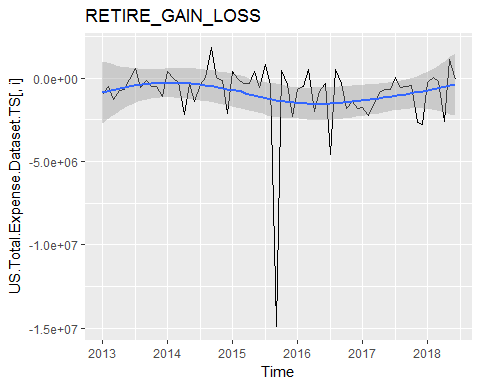
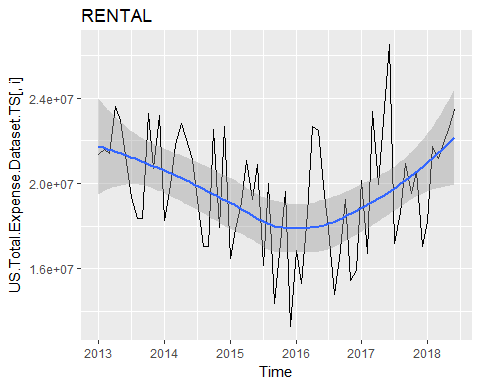
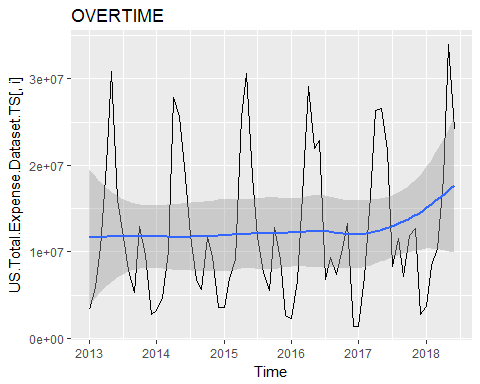
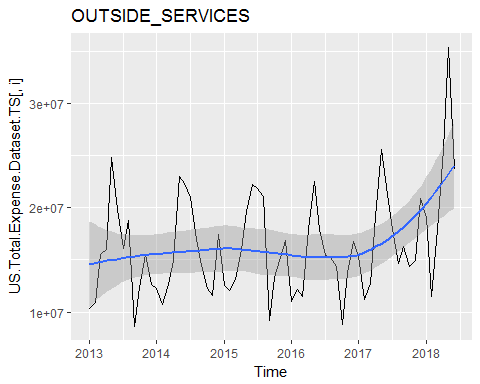
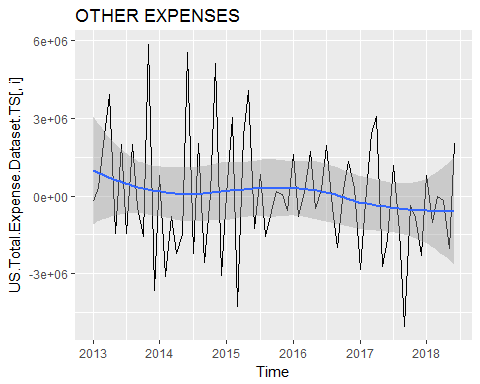
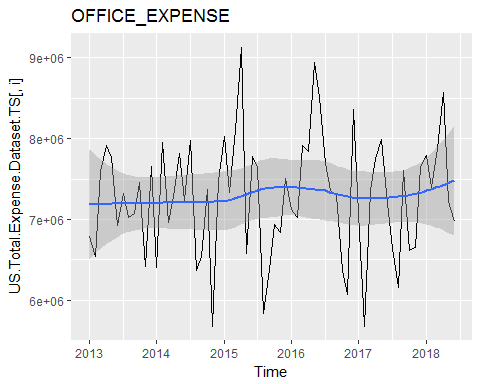
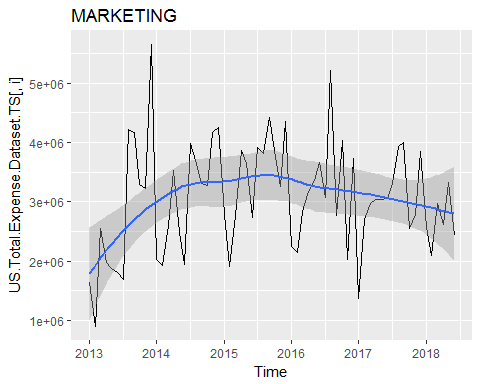
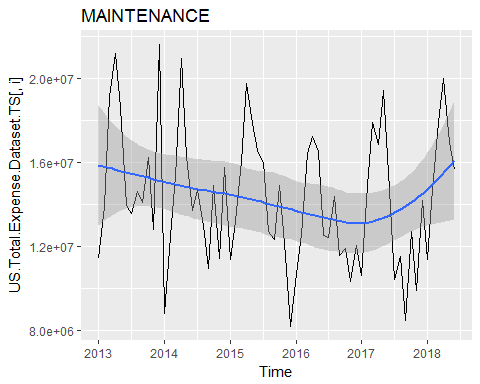
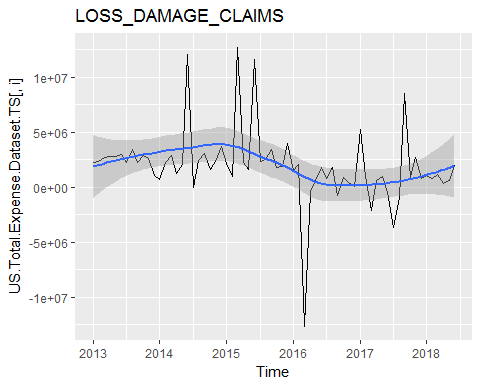
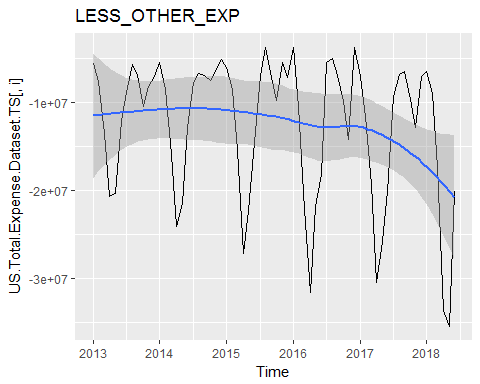
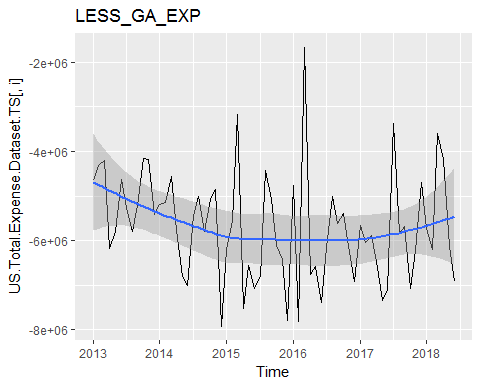
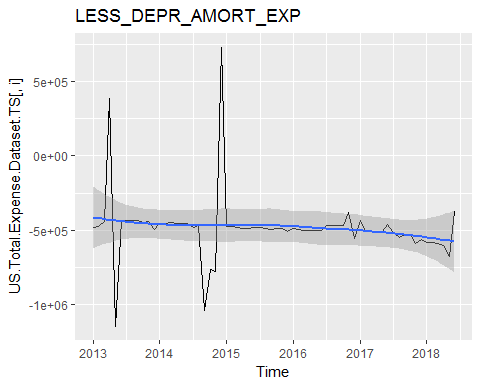
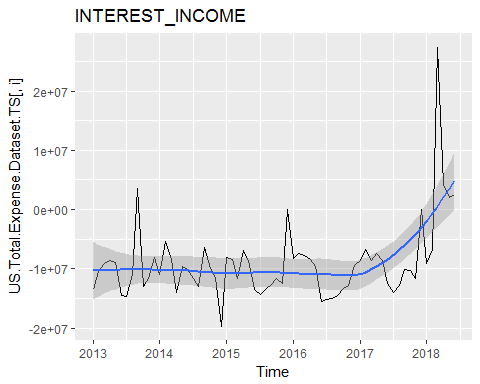
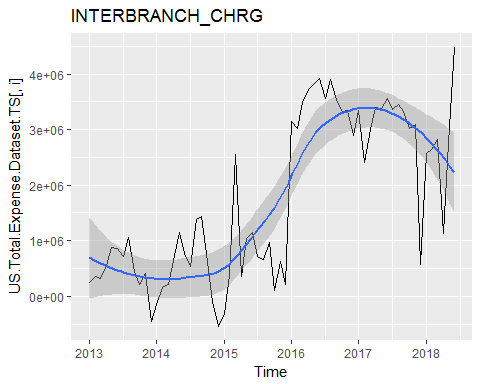
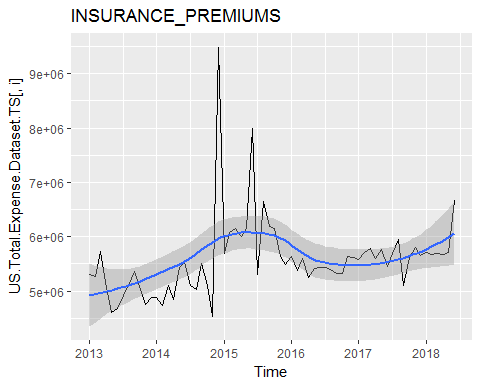
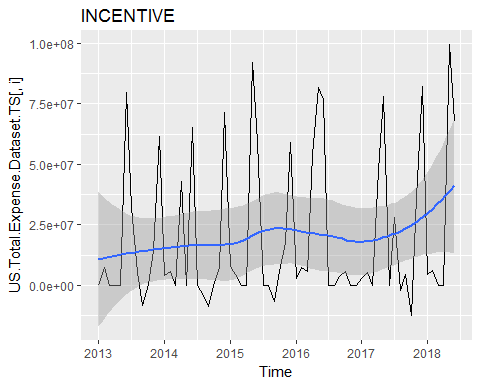
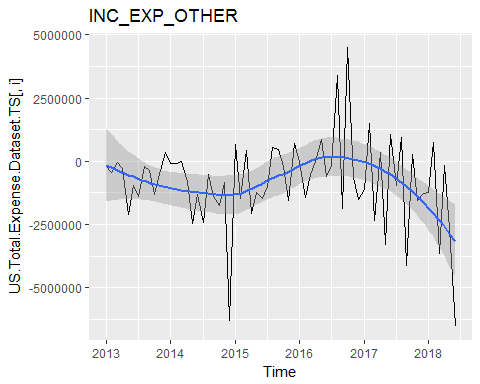
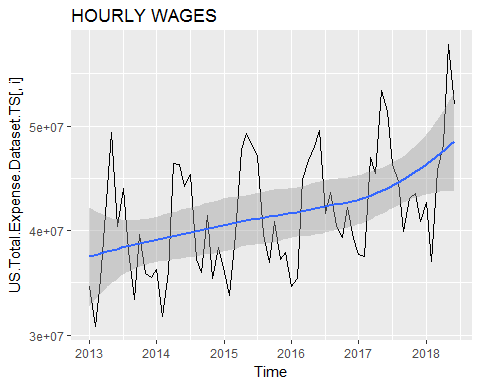
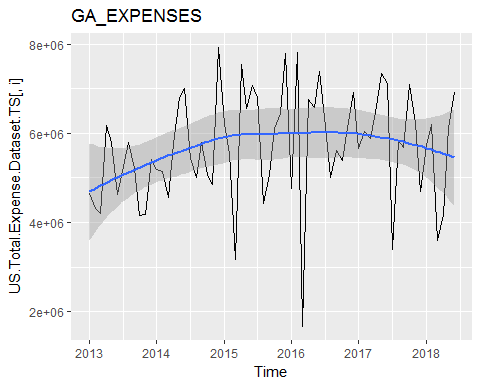
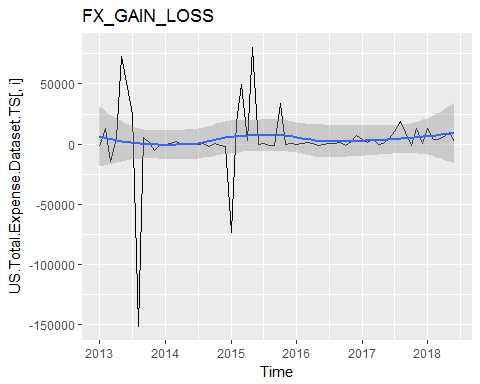
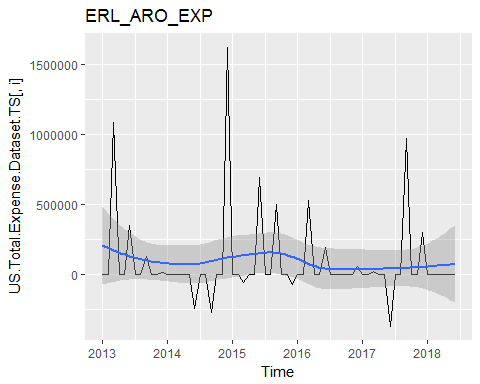
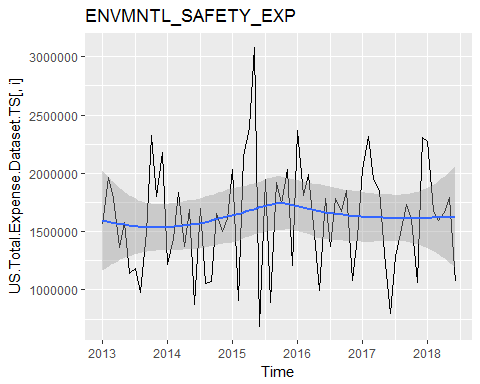
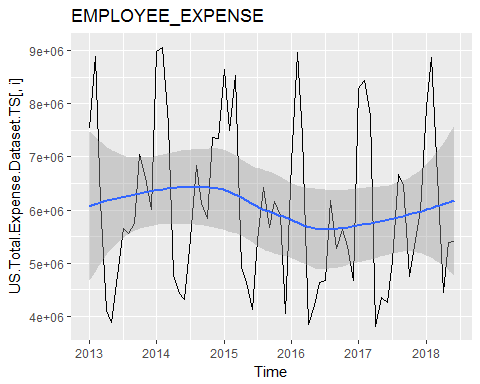
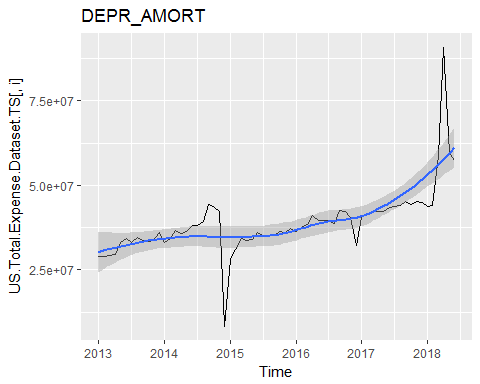
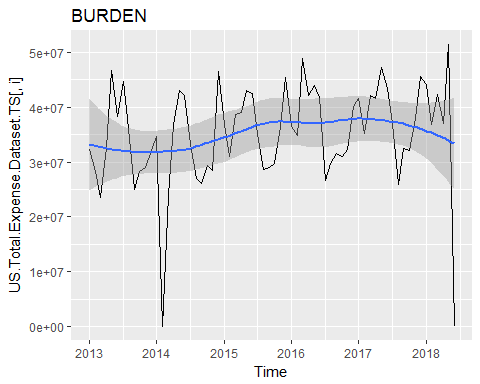
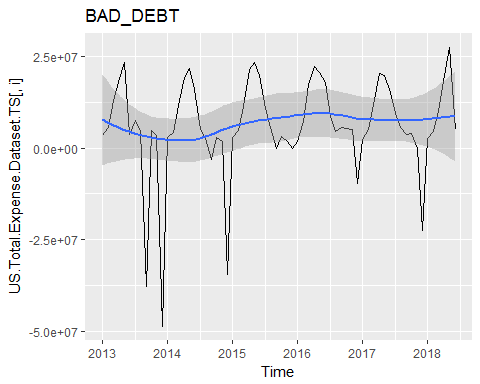
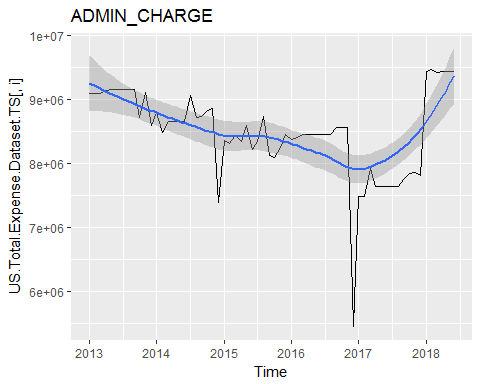
#change strings to factors  
expense.dataset <- expense.dataset %>% mutate\_if(is.character, as.factor)

#subset down to US Total Sum  
US.Total.Expense.Dataset <- expense.dataset %>%  
 group\_by(Date, Parent.Account, Geography) %>%  
 summarise(Expense.Sum = sum(Amount))  
  
#make wide version  
US.Total.Expense.Dataset <- dcast(setDT(US.Total.Expense.Dataset),   
 formula = Date + Geography ~ Parent.Account,  
 value.var = c("Expense.Sum"),  
 fun.aggregate = sum)  
#fill NAs   
US.Total.Expense.Dataset[is.na(US.Total.Expense.Dataset)] <- 0  
  
#sort oldest to newest  
US.Total.Expense.Dataset <- US.Total.Expense.Dataset[order(US.Total.Expense.Dataset$Date, decreasing = FALSE)]  
  
#create ts object drop first two cols (non-numeric)  
US.Total.Expense.Dataset.TS <- ts(data = US.Total.Expense.Dataset[,-c(1,2)],   
 start = c(2013,01), frequency = 12)

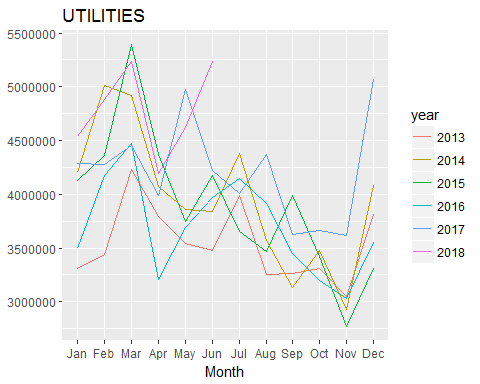
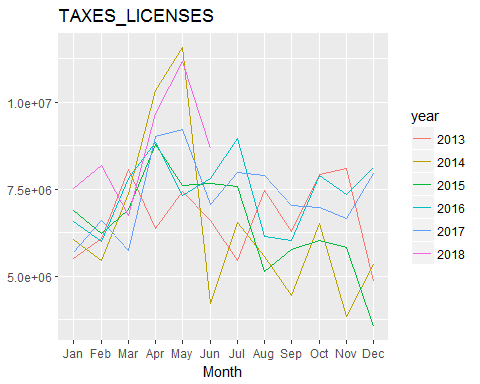
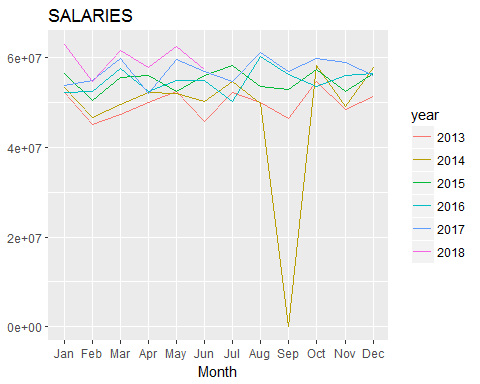
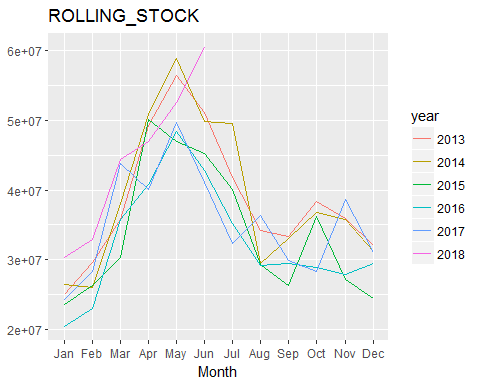
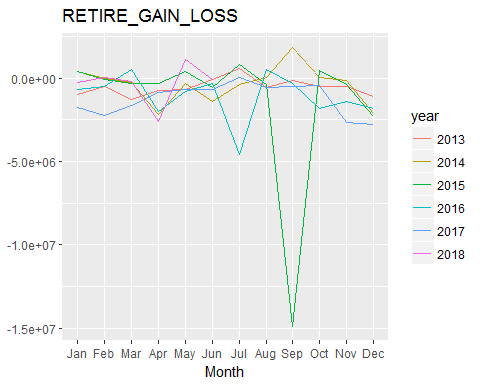
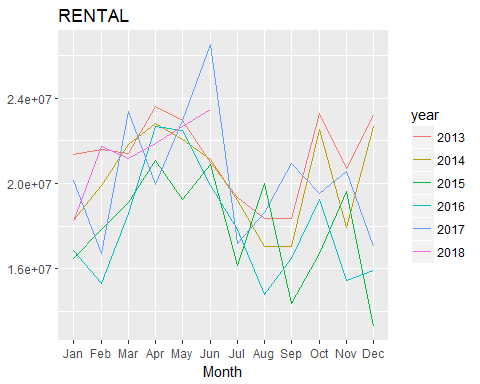
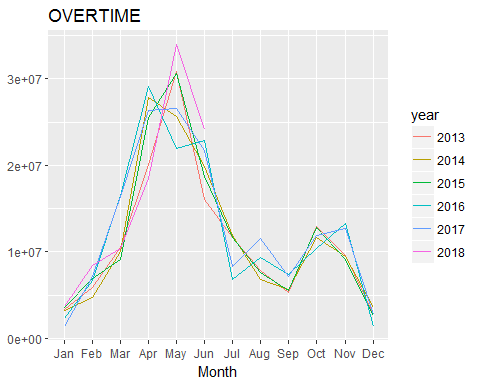
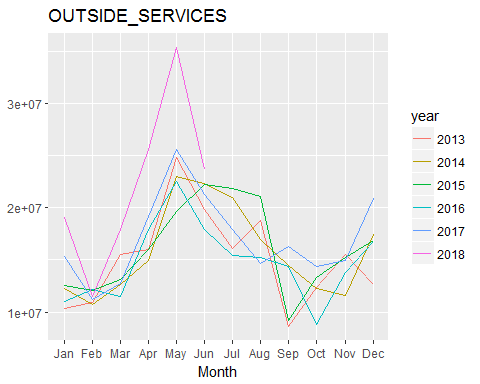
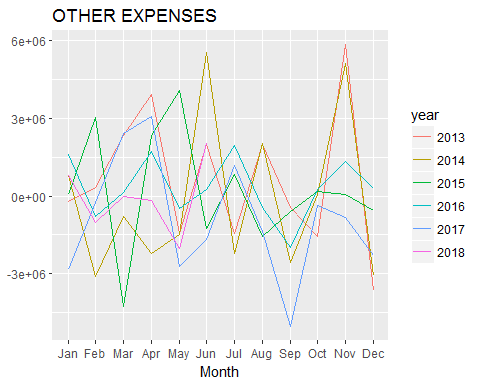
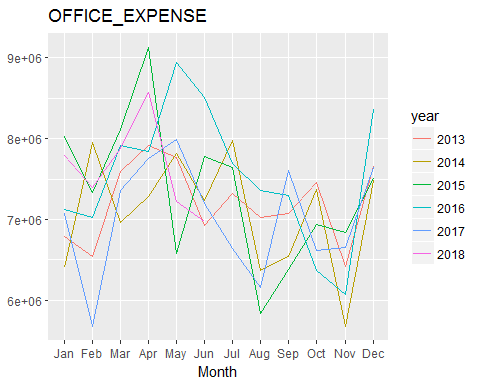
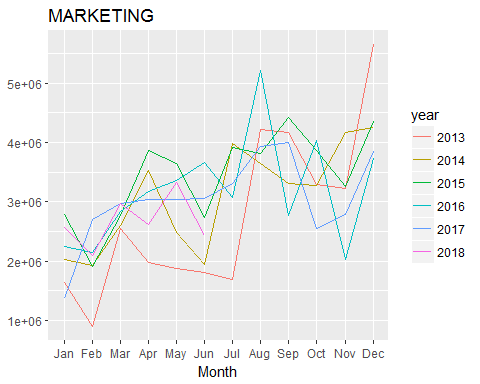
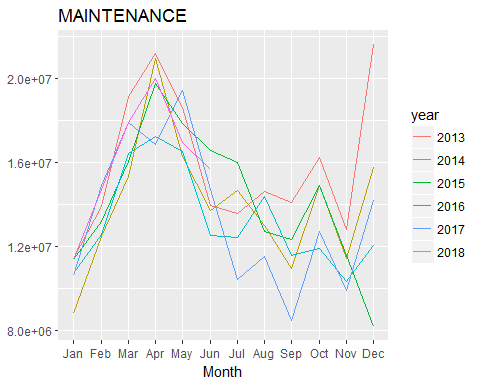
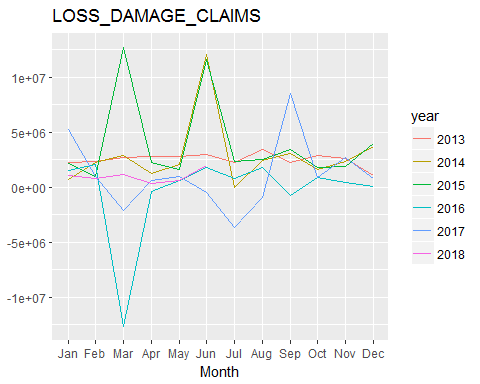
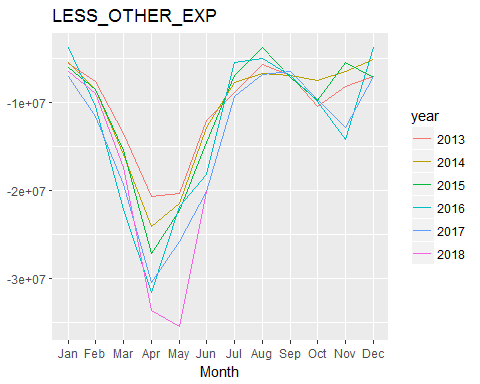
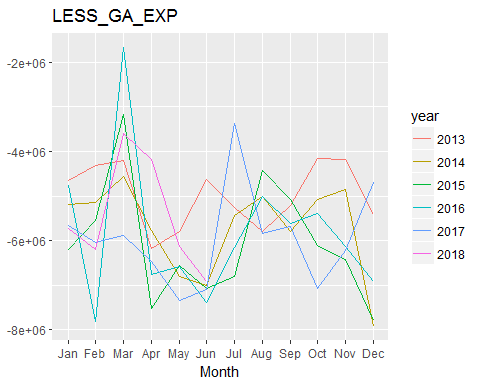
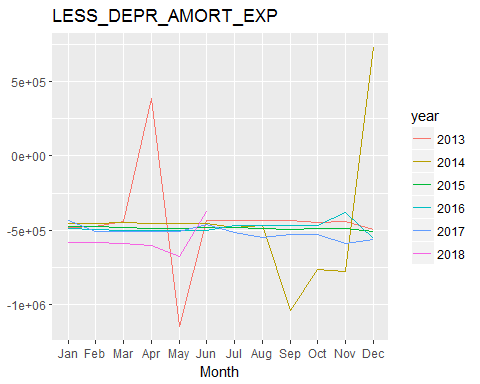
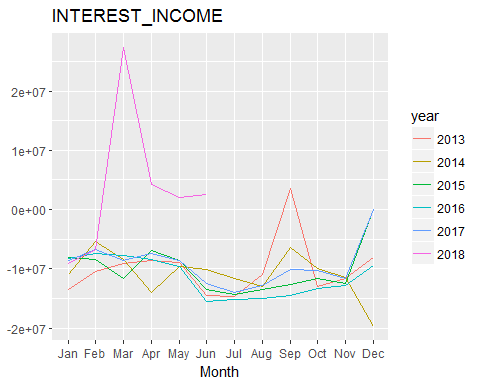
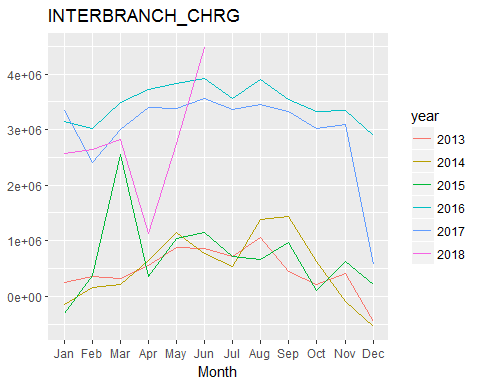
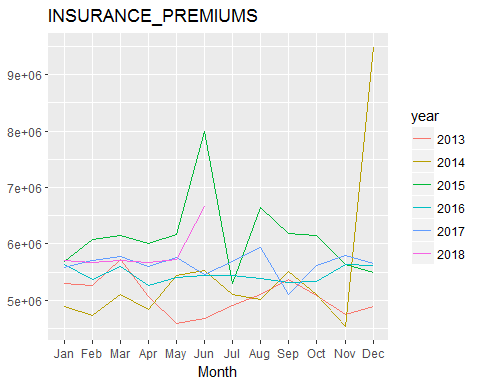
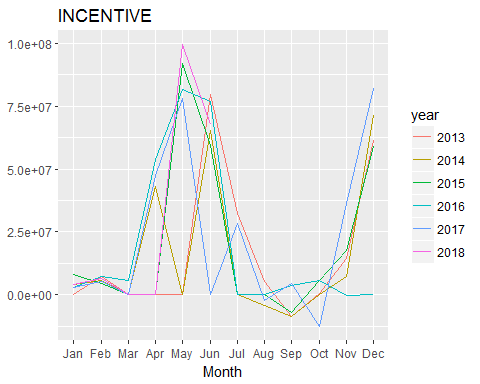
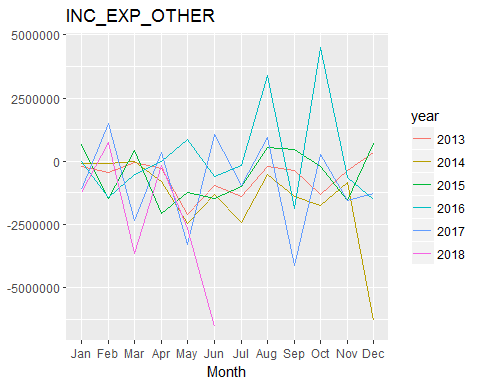
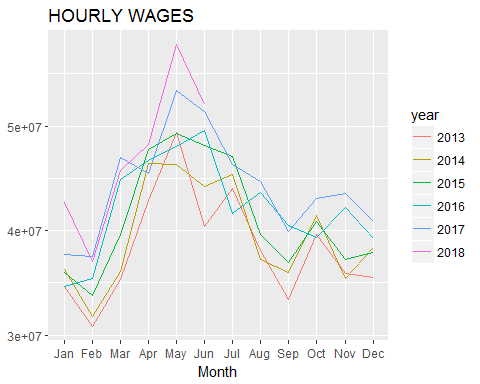
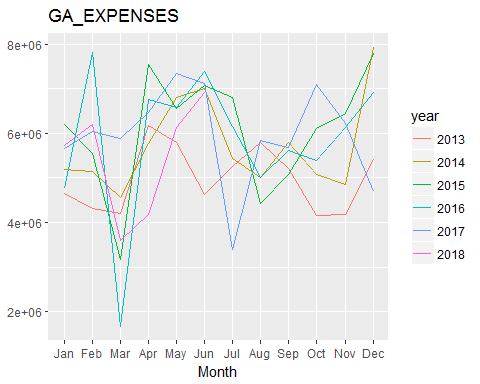
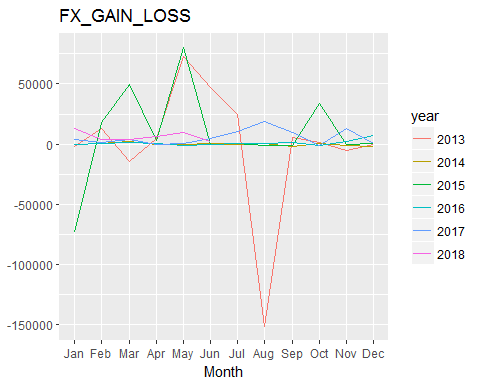
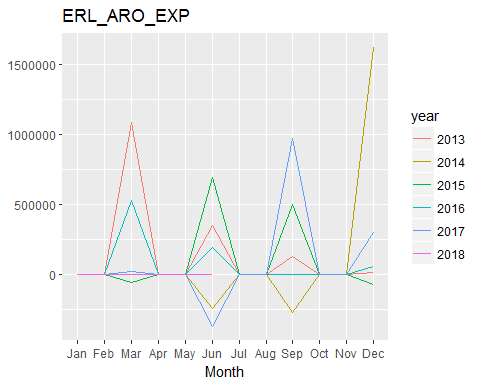
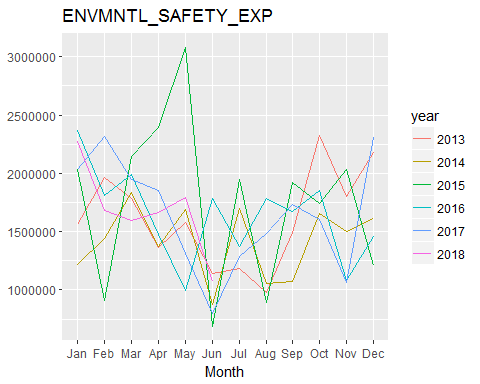
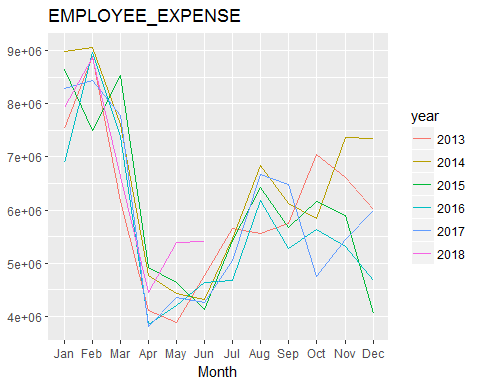
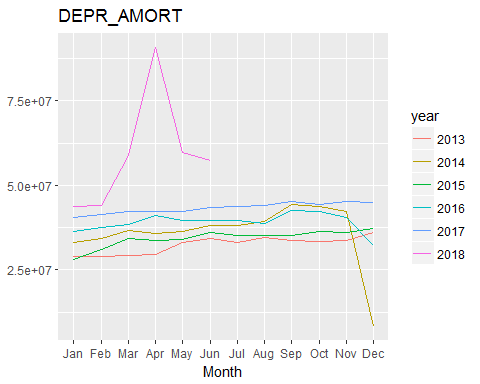
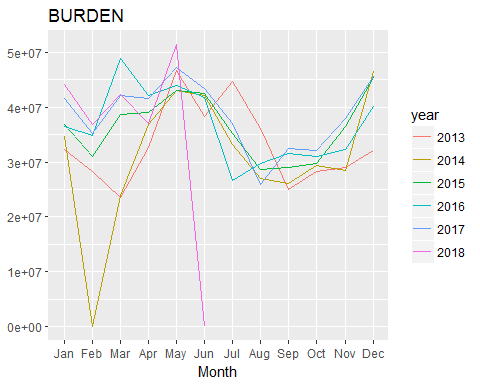
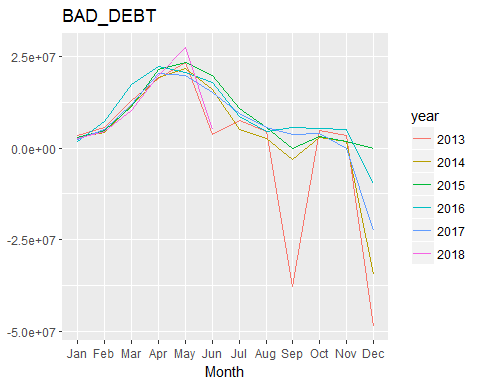
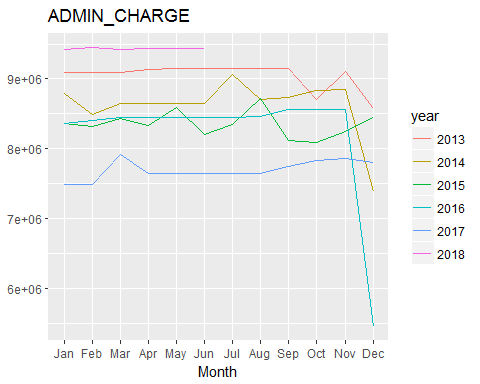
autoplot(US.Total.Expense.Dataset.TS)



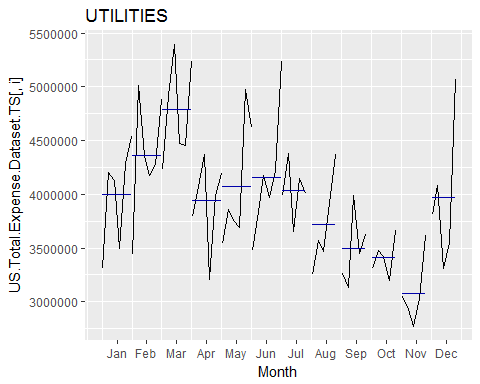
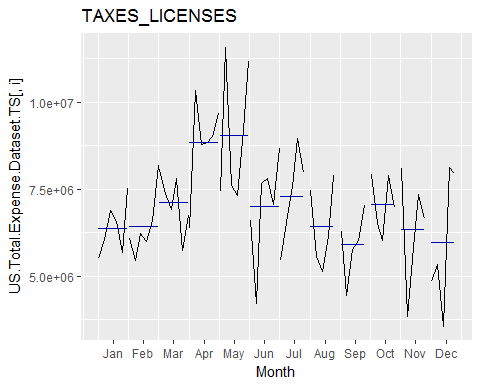
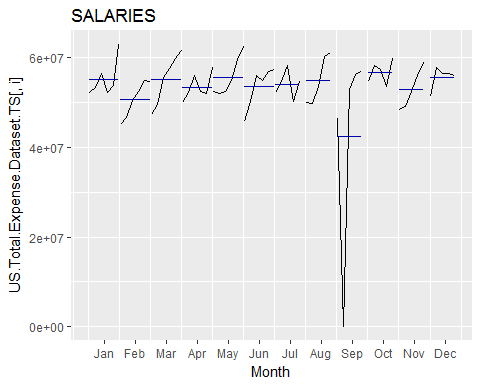
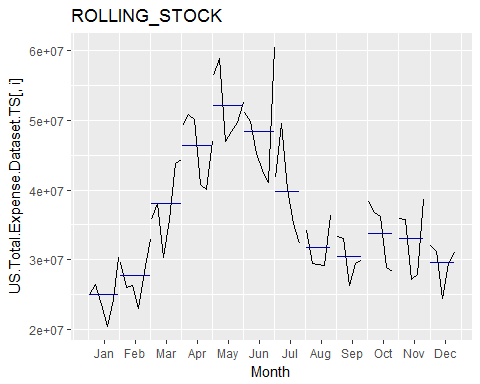
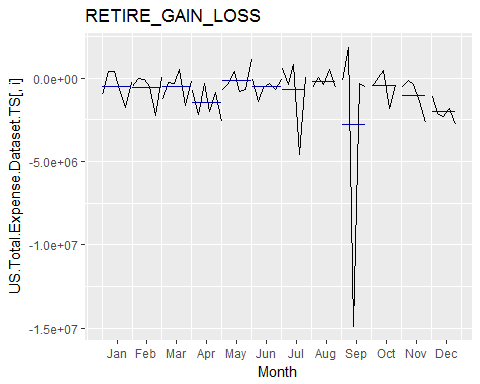
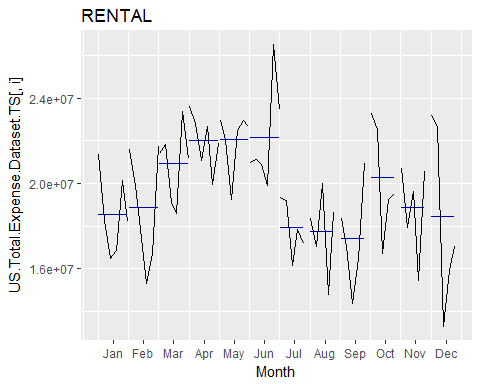
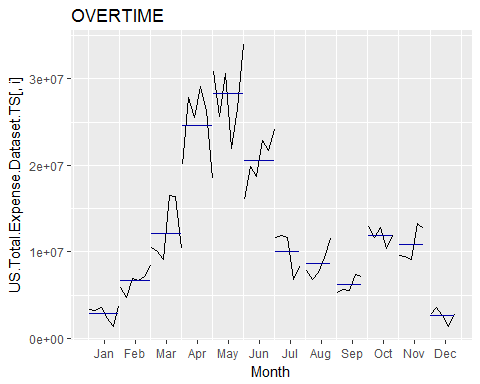
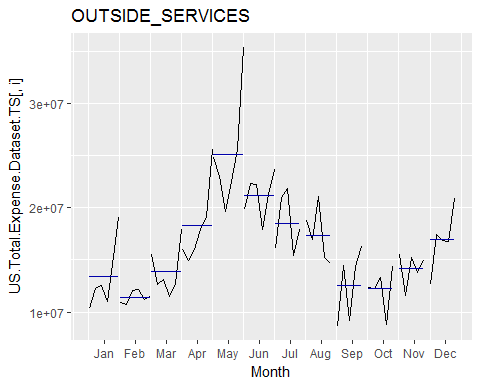
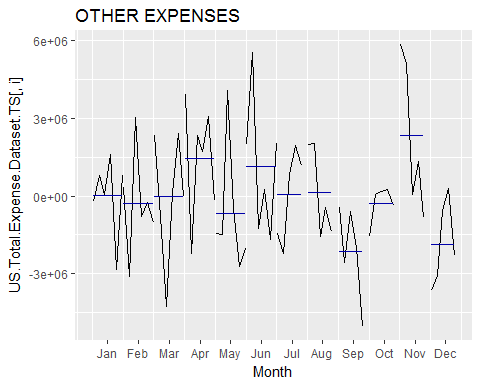
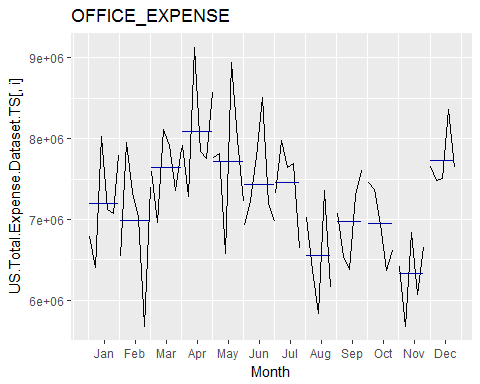
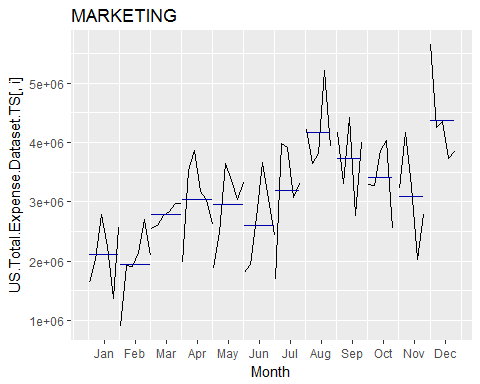
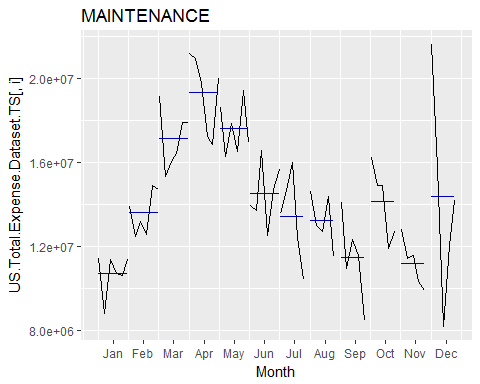
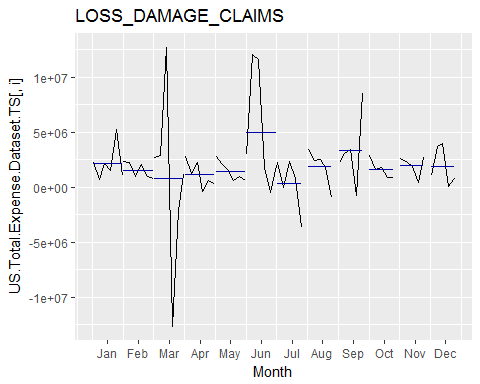
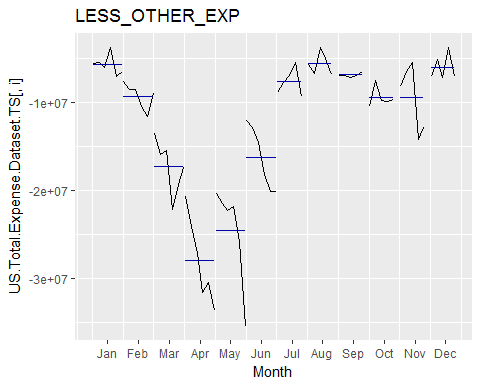
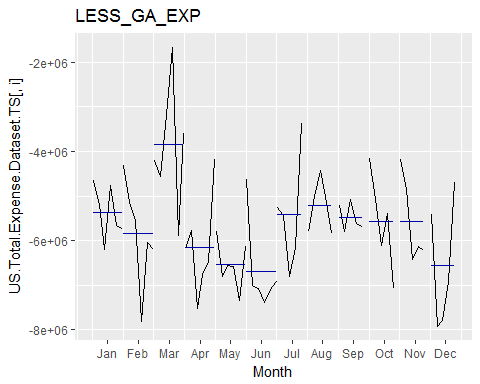
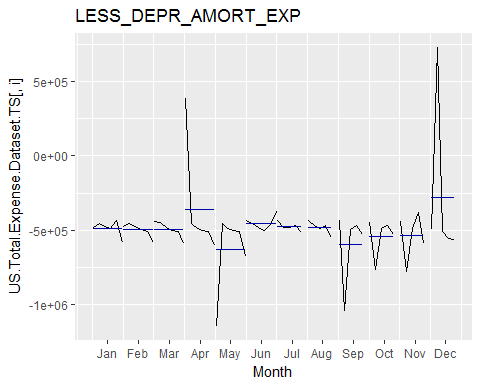
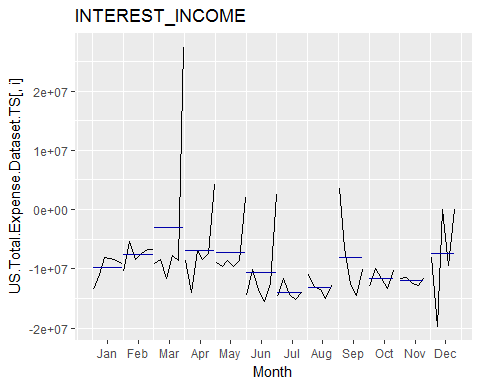
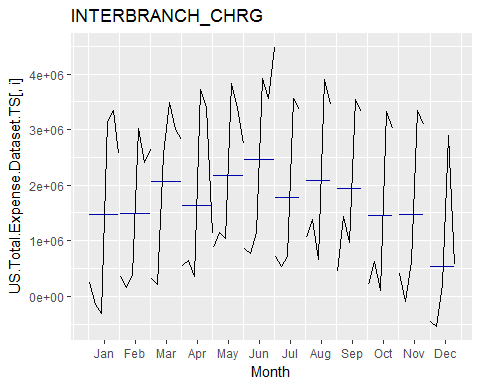
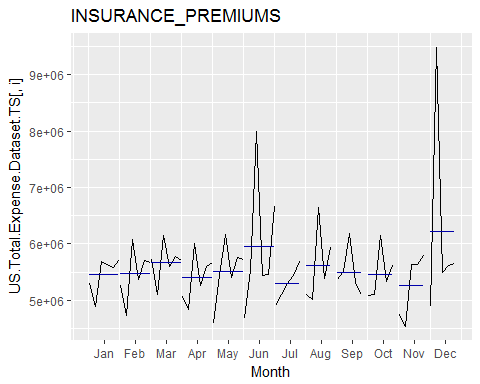
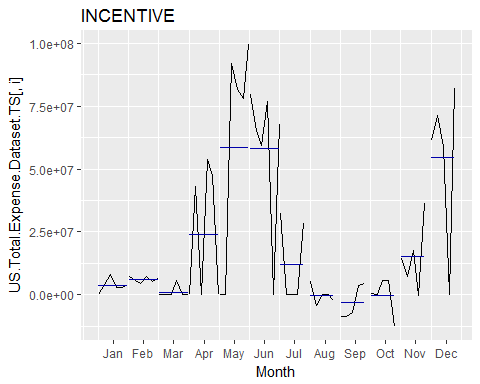
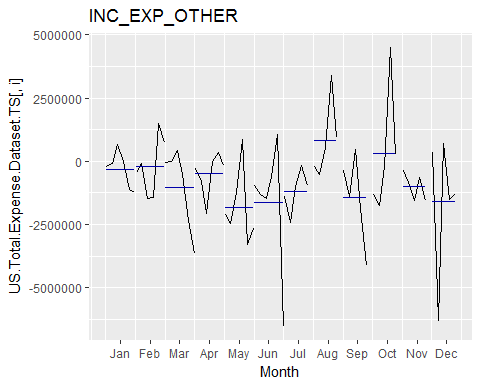
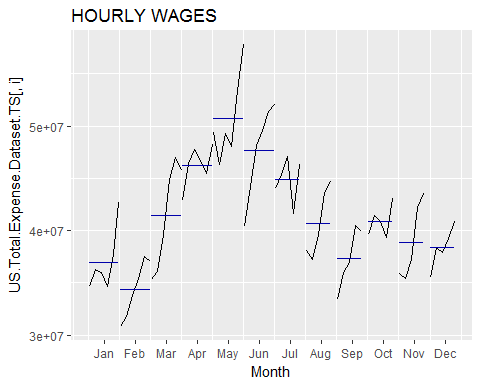
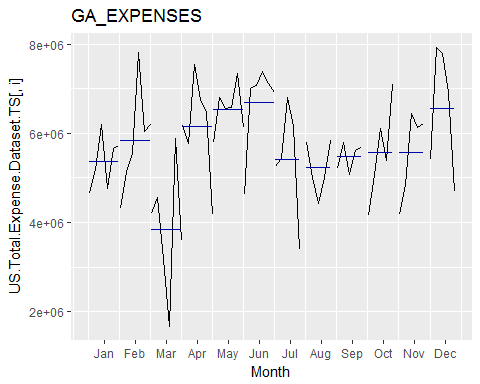
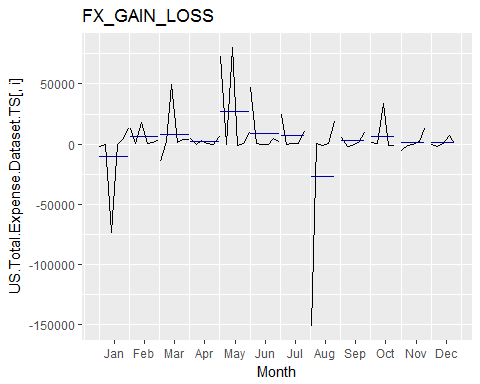
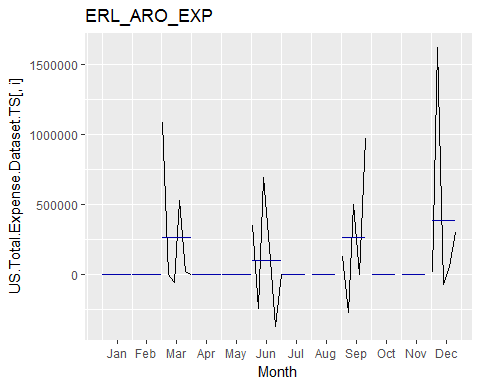
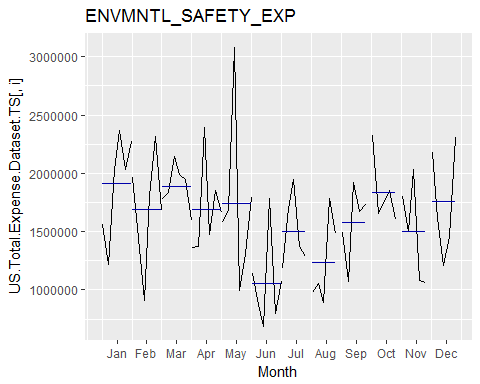
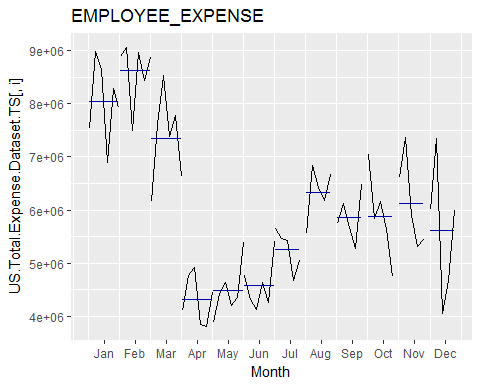
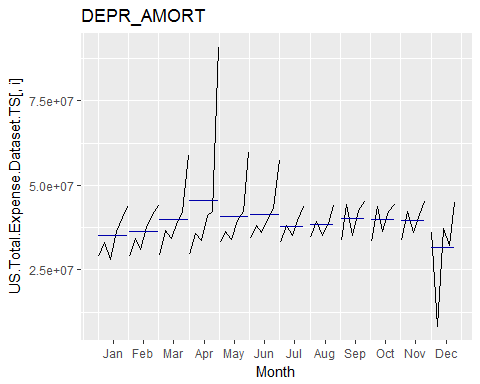
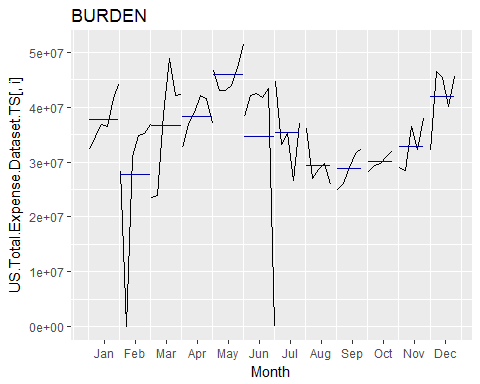
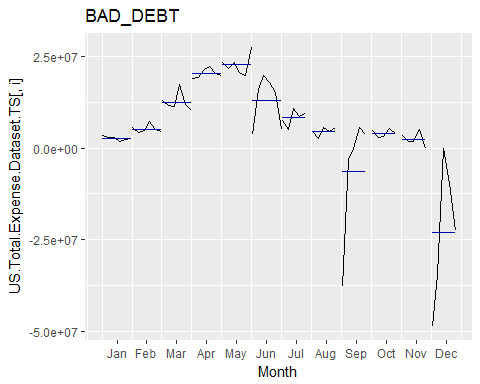
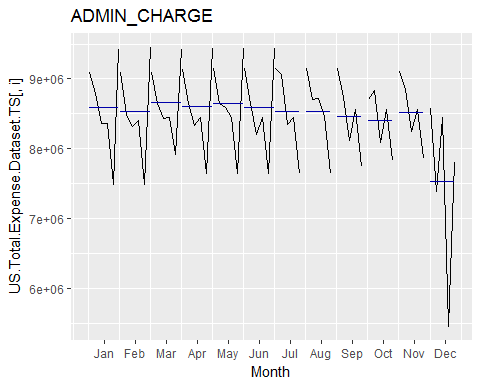
cols <- colnames(US.Total.Expense.Dataset.TS)  
  
#plot each column  
for (i in cols){  
 ts.plot <- autoplot(US.Total.Expense.Dataset.TS[,i]) +  
 geom\_smooth(method = "loess") +  
 ggtitle(i)  
 print(ts.plot)  
}



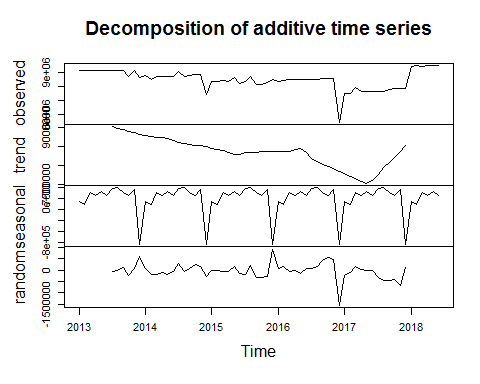
#seasonplot each column  
for (i in cols){  
 season.plot <- ggseasonplot(US.Total.Expense.Dataset.TS[,i]) +   
 ggtitle(i)  
 print(season.plot)  
}



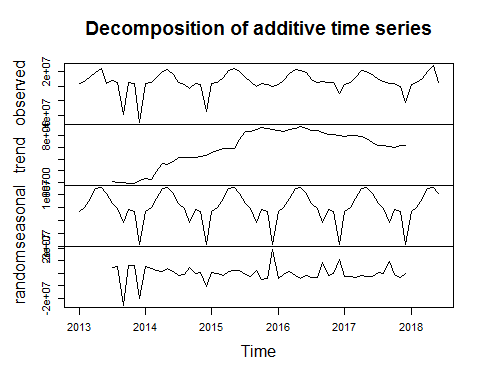
#season subseries plot each column  
for (i in cols){  
 sub.season.plot <- ggsubseriesplot(US.Total.Expense.Dataset.TS[,i]) +   
 ggtitle(i)  
 print(sub.season.plot)  
}



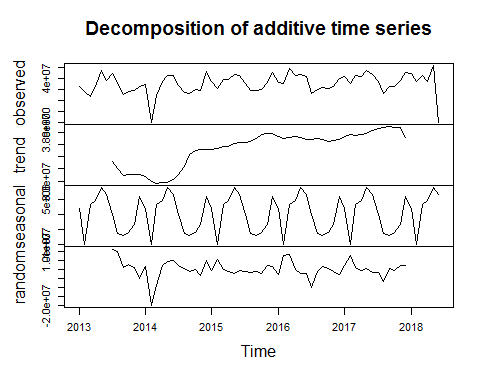
cols <- colnames(US.Total.Expense.Dataset.TS)  
for (i in cols){  
 decomp.plot <- plot(decompose(US.Total.Expense.Dataset.TS[,i]))   
 print(decomp.plot)  
}



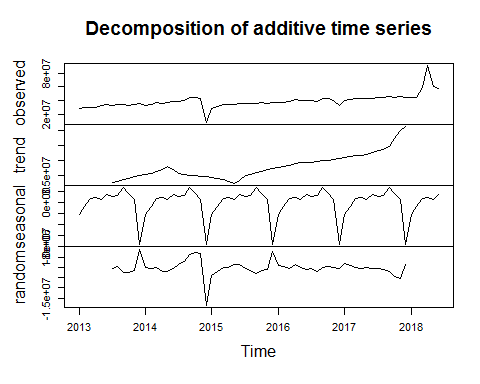
## NULL



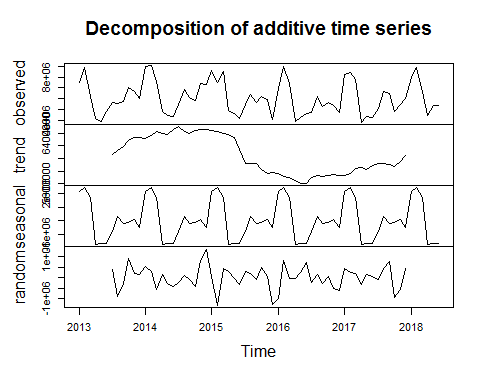
## NULL



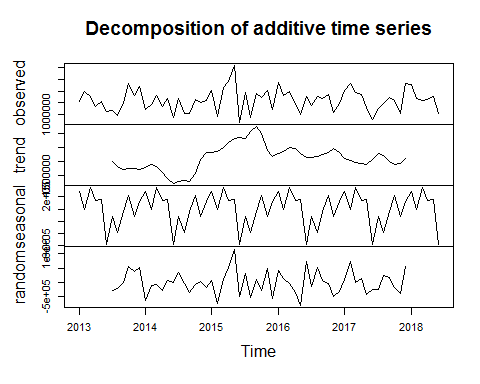
## NULL



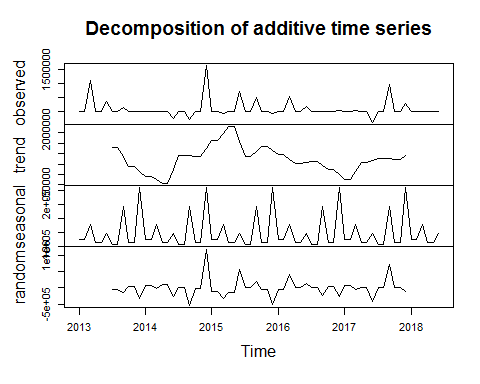
## NULL



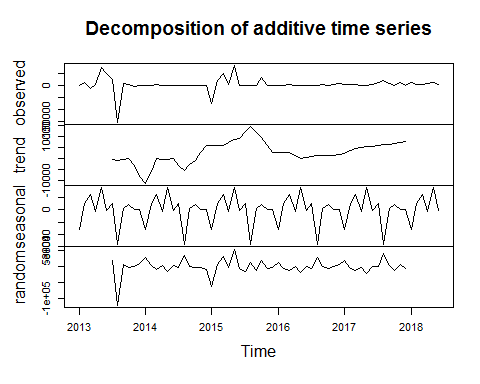
## NULL



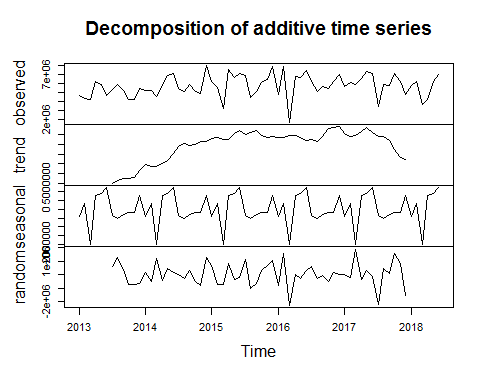
## NULL



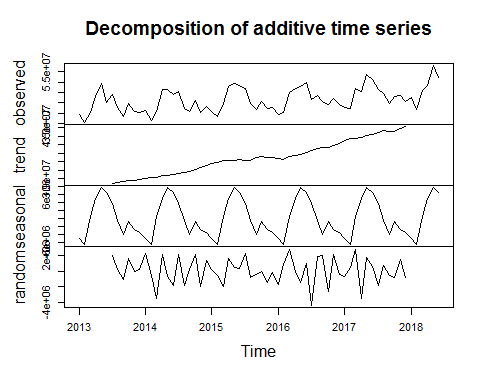
## NULL



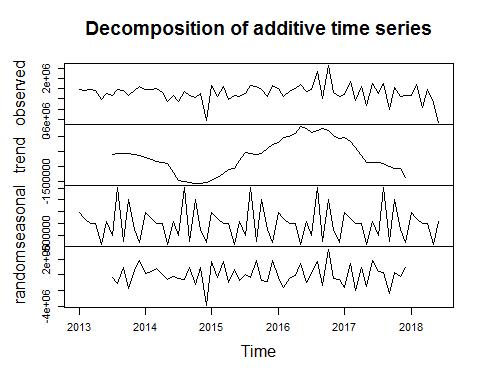
## NULL



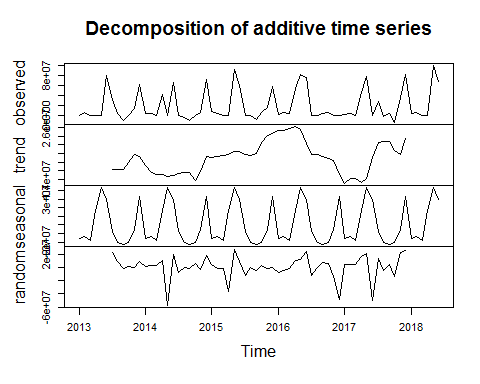
## NULL



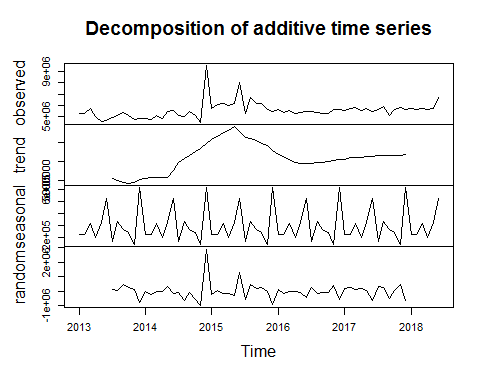
## NULL



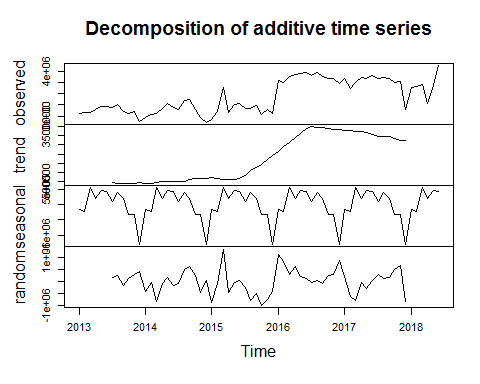
## NULL



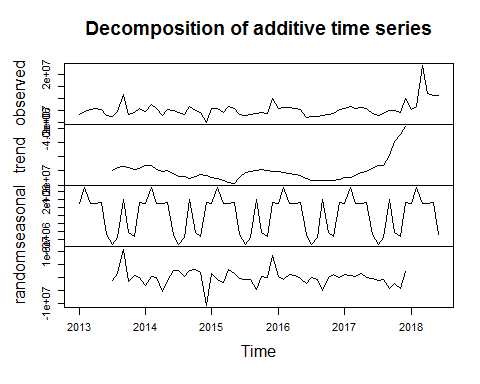
## NULL



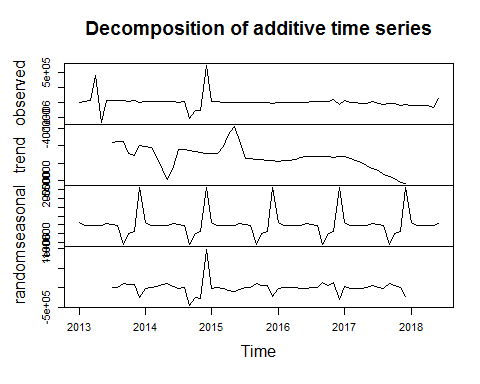
## NULL



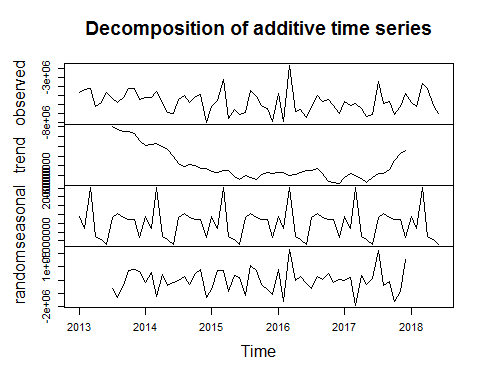
## NULL



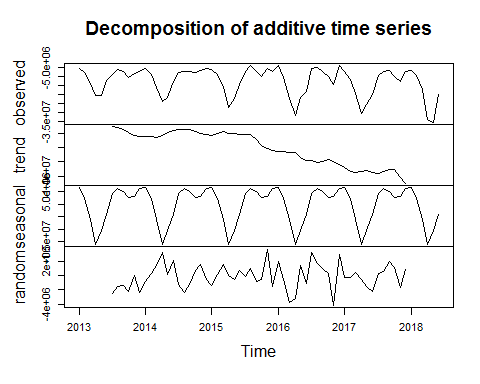
## NULL



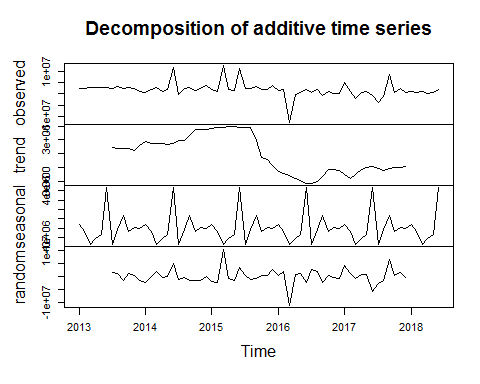
## NULL



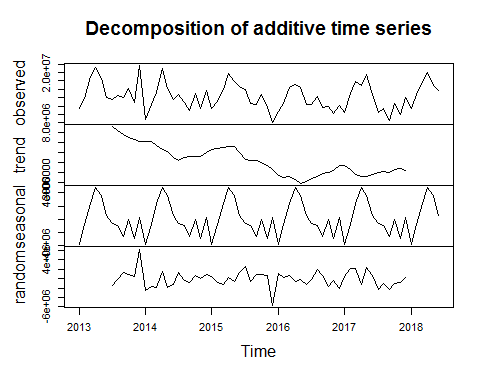
## NULL



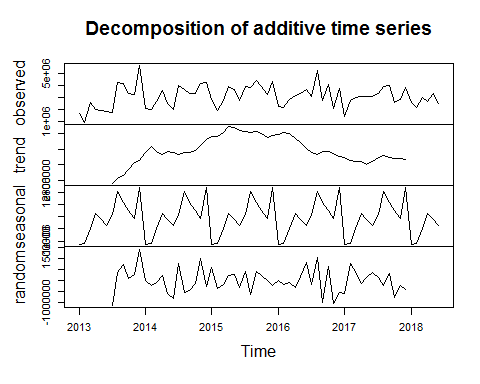
## NULL



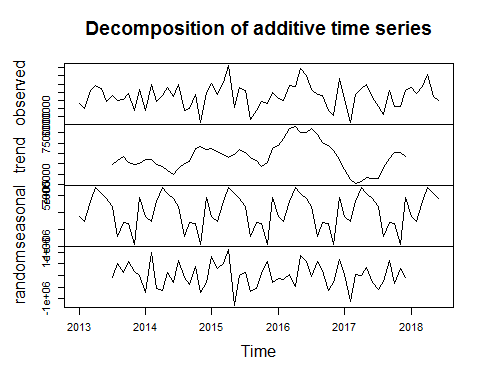
## NULL



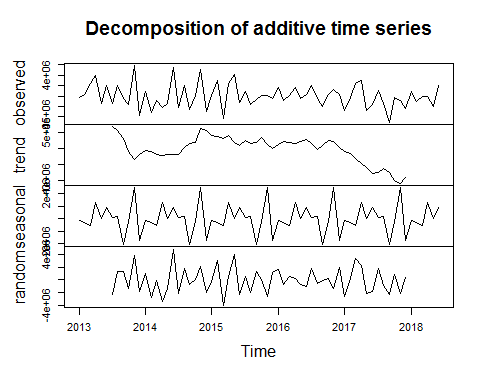
## NULL



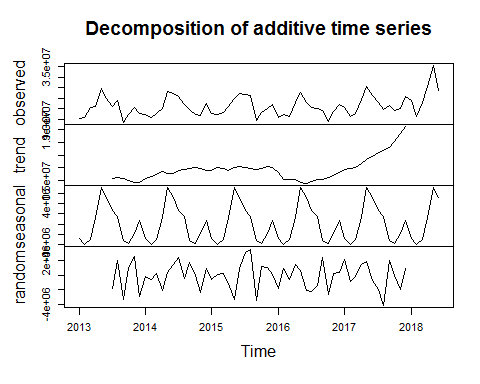
## NULL



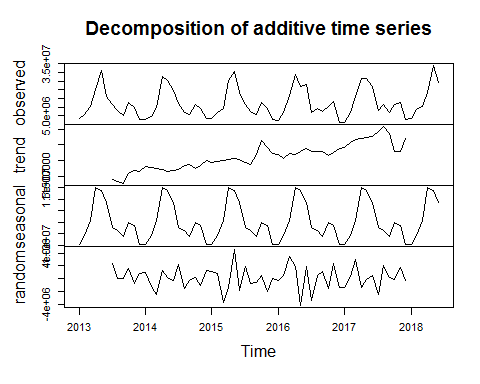
## NULL



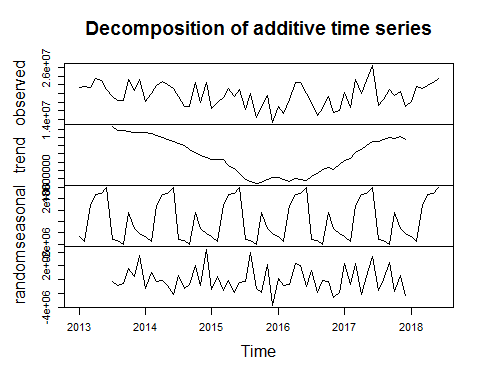
## NULL



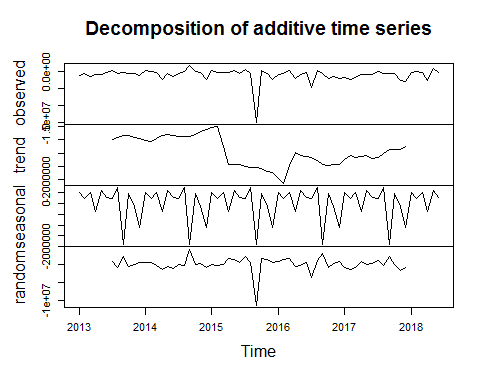
## NULL



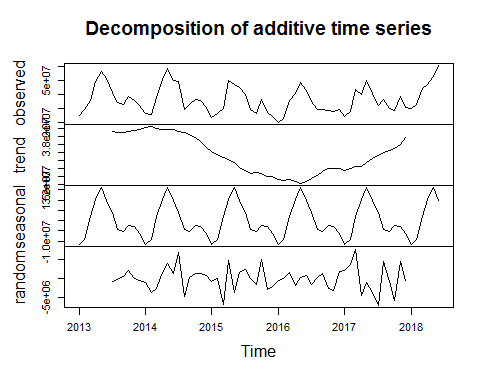
## NULL



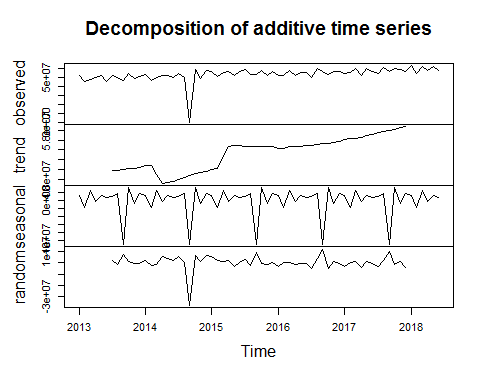
## NULL



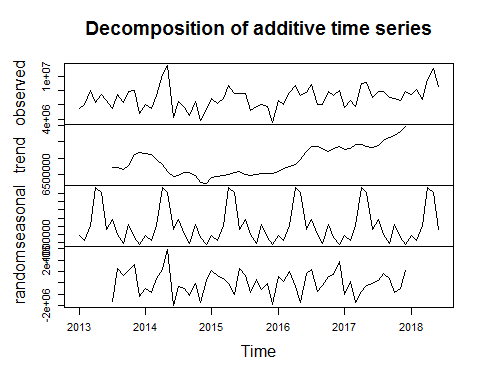
## NULL



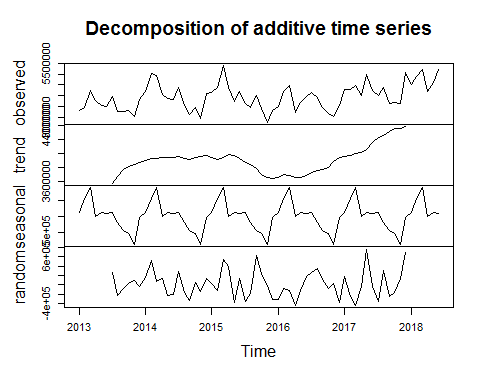
## NULL



## NULL

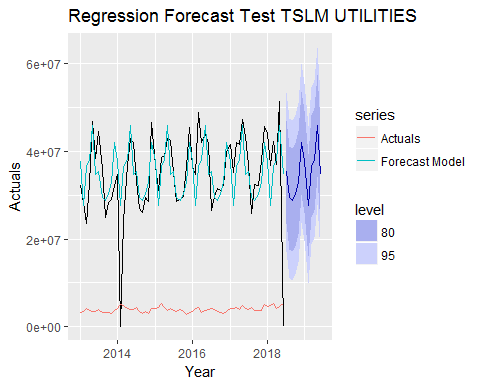
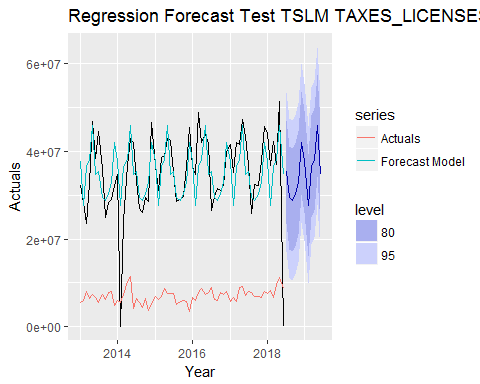
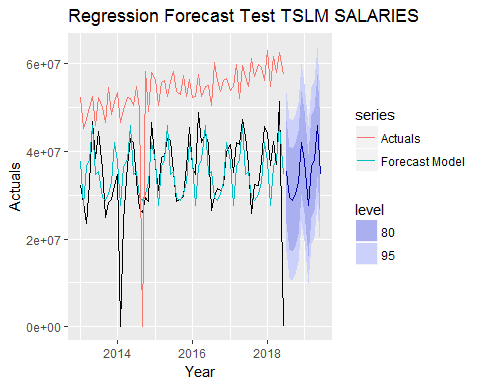
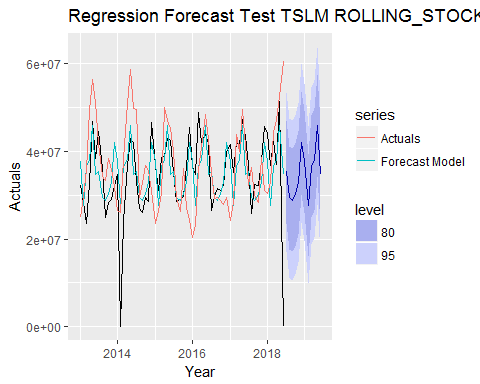
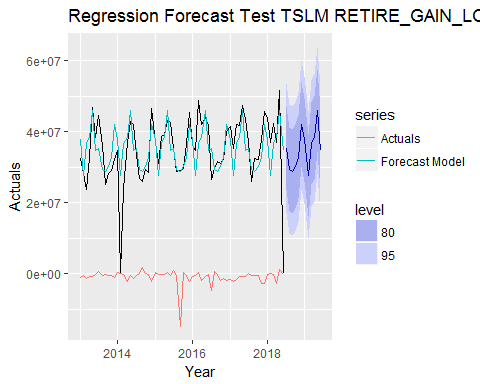
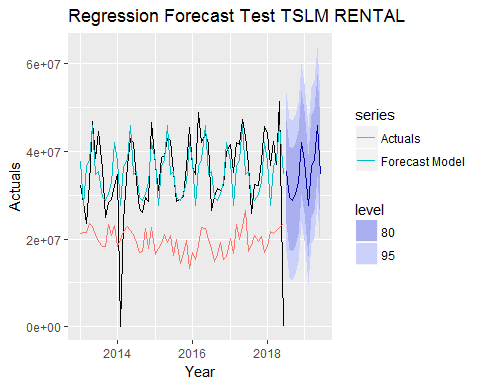
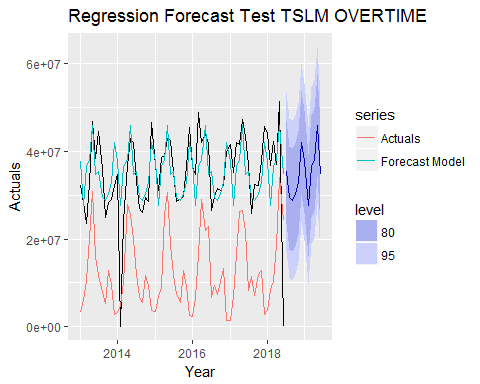
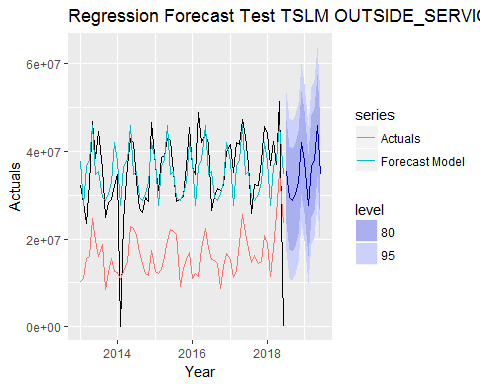
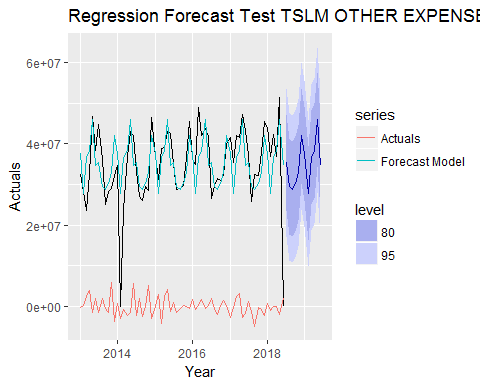
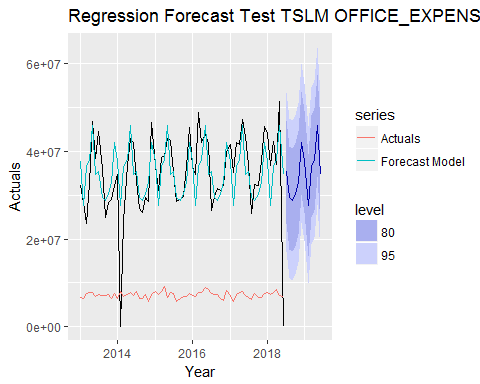
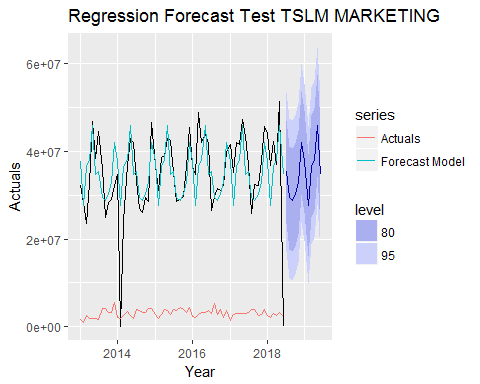
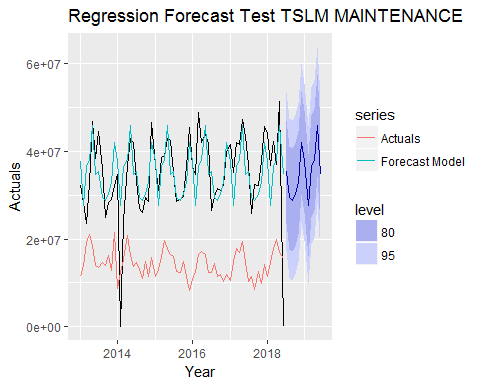
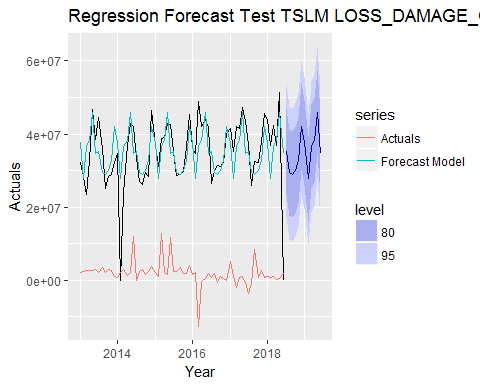
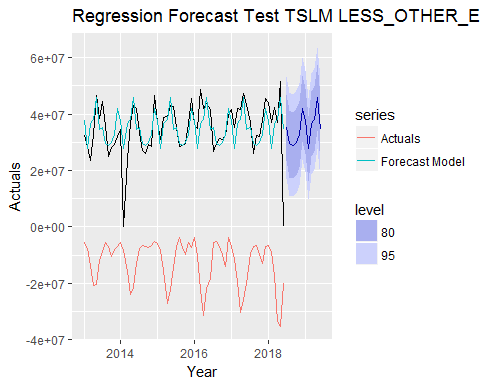
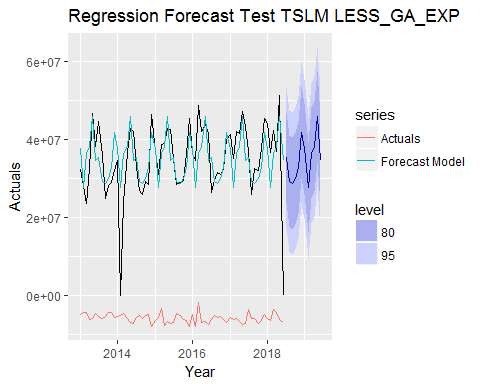
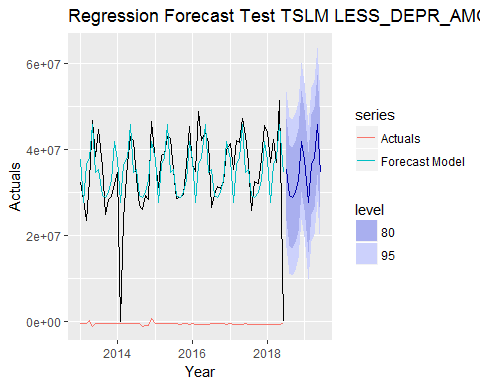
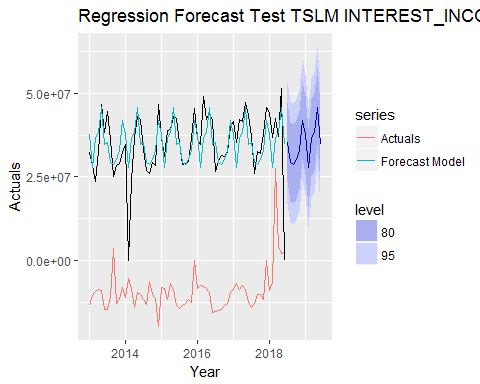
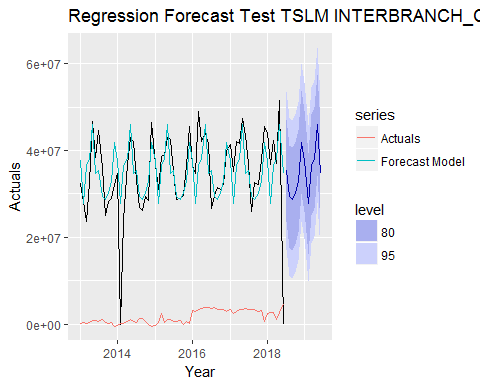
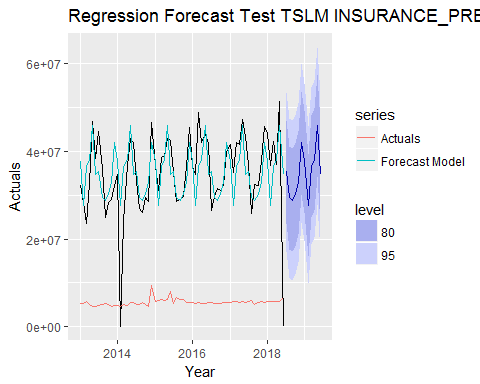
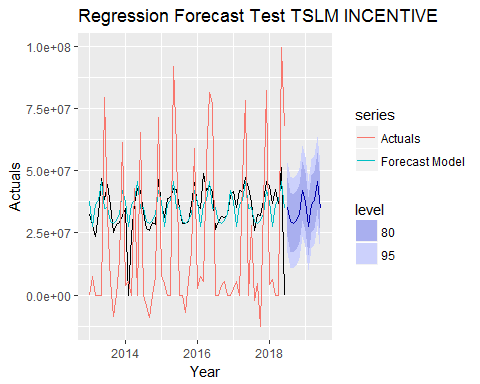
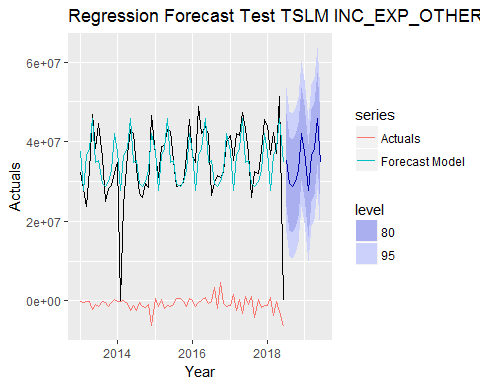
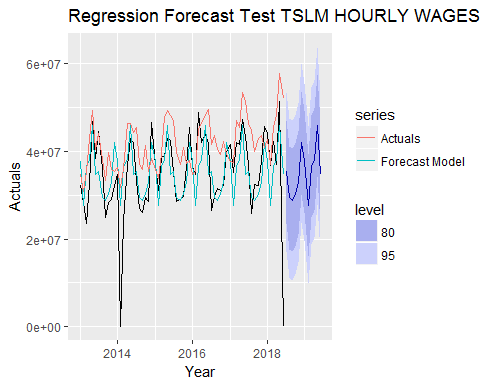
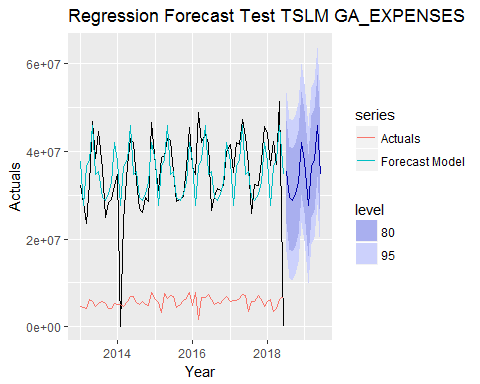
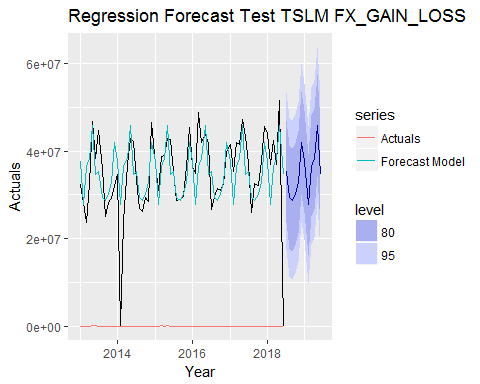
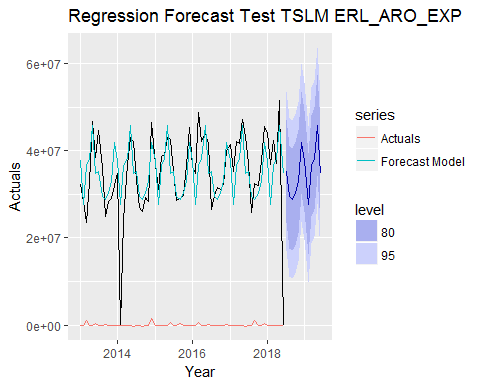
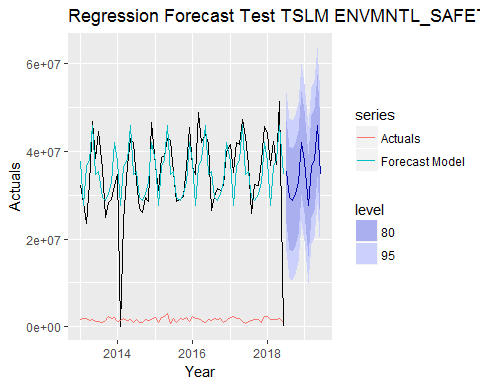
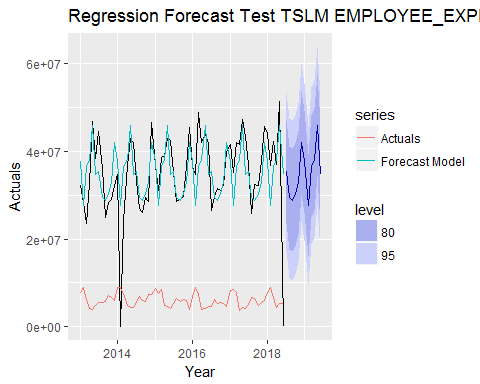
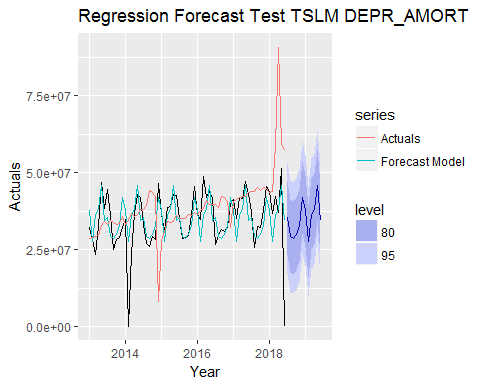
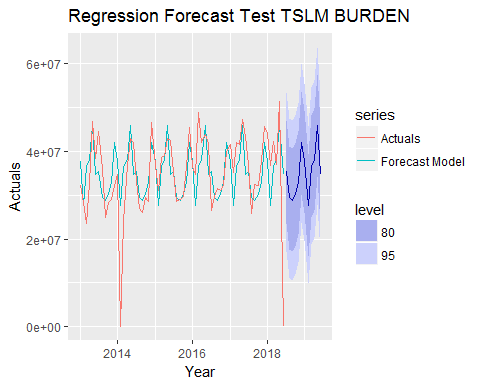
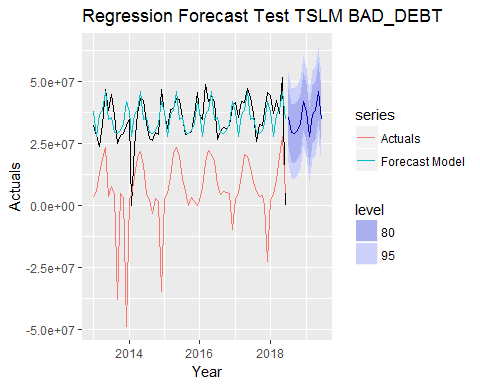
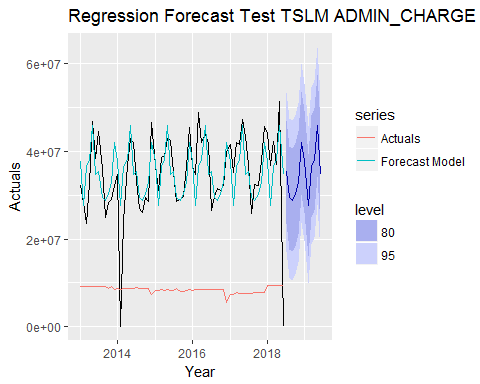


## NULL



## NULL

# tslm.1 <- tslm(US.Total.Expense.Dataset.TS[,1] ~ trend + season)  
# tslm.1  
#   
# summary(tslm.1)  
# checkresiduals(tslm.1)  
  
cols <- colnames(US.Total.Expense.Dataset.TS)  
#loop through cols with linear model  
for (i in cols){  
 tslm.mod <- tslm(US.Total.Expense.Dataset.TS[,i] ~ season)  
 forecast <- forecast(tslm.mod, h = 12)  
   
 forecast.plot <- autoplot(forecast , series="Forecast") +  
 autolayer(tslm.mod$fitted, series="Forecast Model") +  
 autolayer(US.Total.Expense.Dataset.TS[,i], series = "Actuals") +   
 xlab("Year") +   
 ylab("Actuals") +  
 ggtitle(paste("Regression Forecast Test TSLM", i, sep = " "))  
   
 print(forecast.plot)  
}



cols <- colnames(US.Total.Expense.Dataset.TS)  
for (i in cols){  
 ets.1 <- ets(US.Total.Expense.Dataset.TS[,i])  
 #ets.1  
  
 fx.ets.1 <- forecast(ets.1, h = 12)  
  
 plot <- autoplot(fx.ets.1, series="Forecast") +  
 autolayer(ets.1$fitted, series="Forecast Model") +  
 autolayer(US.Total.Expense.Dataset.TS[,i], series = "Actuals") +   
 xlab("Year") +   
 ylab("Actuals") +  
 ggtitle(paste("Regression Forecast Test ETS", i, sep = " "))  
   
 print(plot)  
}

