GHB final 2020

Sulyok

July 19, 2020

setwd("~/Rprojects/GHB")  
Sys.setenv(LANG = "en")  
sessionInfo()

## R version 3.6.3 (2020-02-29)  
## Platform: x86\_64-pc-linux-gnu (64-bit)  
## Running under: Ubuntu 18.04.3 LTS  
##   
## Matrix products: default  
## BLAS: /usr/lib/x86\_64-linux-gnu/blas/libblas.so.3.7.1  
## LAPACK: /usr/lib/x86\_64-linux-gnu/lapack/liblapack.so.3.7.1  
##   
## locale:  
## [1] LC\_CTYPE=C LC\_NUMERIC=C   
## [3] LC\_TIME=de\_DE.UTF-8 LC\_COLLATE=C   
## [5] LC\_MONETARY=de\_DE.UTF-8 LC\_MESSAGES=C   
## [7] LC\_PAPER=de\_DE.UTF-8 LC\_NAME=C   
## [9] LC\_ADDRESS=C LC\_TELEPHONE=C   
## [11] LC\_MEASUREMENT=de\_DE.UTF-8 LC\_IDENTIFICATION=C   
##   
## attached base packages:  
## [1] stats graphics grDevices utils datasets methods base   
##   
## loaded via a namespace (and not attached):  
## [1] compiler\_3.6.3 magrittr\_1.5 tools\_3.6.3 htmltools\_0.5.0  
## [5] yaml\_2.2.1 stringi\_1.4.6 rmarkdown\_2.2 knitr\_1.28   
## [9] stringr\_1.4.0 xfun\_0.14 digest\_0.6.25 rlang\_0.4.6   
## [13] evaluate\_0.14

library(haven)  
data <- read\_sav("GHB\_tox\_adat\_140711.sav")  
GTD <- read\_sav("GHB\_tox\_google.sav")  
#GTD<-GTD[,-c(6:9)]  
#GTD<-na.omit(GTD)  
GTD<-GTD[-47,]  
counts<-table(data$FELVETEL)  
counts<-as.data.frame(counts)  
  
  
GTD$date <- seq(as.Date("2009-09-14"), as.Date("2013-07-13"), "month" )  
  
library(ggplot2)  
  
  
p<-ggplot(GTD, aes(x = GTD$date)) +  
geom\_point(aes(y=GTD$tox\_eset\*5, colour="Admission numbers"), alpha= 0.4) + geom\_smooth(aes(y=GTD$tox\_eset\*5, colour="Admission numbers"), span = 0.2, se=FALSE) +  
geom\_point(aes(y=GTD$google1, col="GHB-related seach volumes"), alpha= 0.4) + geom\_smooth(aes(y=GTD$google1, col="GHB-related seach volumes"), span = 0.3, se=FALSE) +  
 geom\_point(aes(y=GTD$google2, col="Gina-related seach volumes"), alpha= 0.4) + geom\_smooth(aes(y=GTD$google2, col="Gina-related seach volumes"), span = 0.3, se=FALSE) +   
 scale\_colour\_manual("", breaks = c("Admission numbers", "GHB-related seach volumes","Gina-related seach volumes") , values = c("red", "blue", "green")) +  
scale\_x\_date(date\_minor\_breaks = "1 month") +  
xlab(NULL) +  
ylab("Google search volume") + ylim(c(0, 100)) +  
 scale\_y\_continuous(sec.axis = sec\_axis(~./5, name = "Monthly admission number")) +  
theme\_bw() + theme(legend.position = "none")

## Scale for 'y' is already present. Adding another scale for 'y', which will  
## replace the existing scale.

p

## Warning: Use of `GTD$tox\_eset` is discouraged. Use `tox\_eset` instead.

## Warning: Use of `GTD$date` is discouraged. Use `date` instead.

## Warning: Use of `GTD$tox\_eset` is discouraged. Use `tox\_eset` instead.

## Warning: Use of `GTD$date` is discouraged. Use `date` instead.

## Warning: Use of `GTD$google1` is discouraged. Use `google1` instead.

## Warning: Use of `GTD$date` is discouraged. Use `date` instead.

## Warning: Use of `GTD$google1` is discouraged. Use `google1` instead.

## Warning: Use of `GTD$date` is discouraged. Use `date` instead.

## Warning: Use of `GTD$google2` is discouraged. Use `google2` instead.

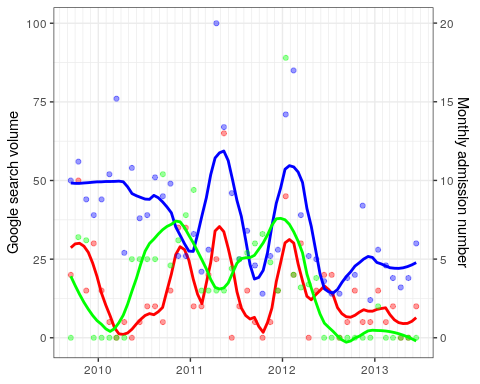
## Warning: Use of `GTD$date` is discouraged. Use `date` instead.

## Warning: Use of `GTD$google2` is discouraged. Use `google2` instead.

## Warning: Use of `GTD$date` is discouraged. Use `date` instead.

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'

## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'  
## `geom\_smooth()` using method = 'loess' and formula 'y ~ x'



cortesttoxeset<-function(x){cor.test(x, GTD$tox\_eset, method="kendall")}  
contempcorrsmonthtox\_eset<-lapply( GTD[, 4:9], cortesttoxeset)

## Warning in cor.test.default(x, GTD$tox\_eset, method = "kendall"): Cannot compute  
## exact p-value with ties

## Warning in cor.test.default(x, GTD$tox\_eset, method = "kendall"): Cannot compute  
## exact p-value with ties  
  
## Warning in cor.test.default(x, GTD$tox\_eset, method = "kendall"): Cannot compute  
## exact p-value with ties  
  
## Warning in cor.test.default(x, GTD$tox\_eset, method = "kendall"): Cannot compute  
## exact p-value with ties  
  
## Warning in cor.test.default(x, GTD$tox\_eset, method = "kendall"): Cannot compute  
## exact p-value with ties  
  
## Warning in cor.test.default(x, GTD$tox\_eset, method = "kendall"): Cannot compute  
## exact p-value with ties

contempcorrsmonthtox\_eset

## $google1  
##   
## Kendall's rank correlation tau  
##   
## data: x and GTD$tox\_eset  
## z = 2.058, p-value = 0.03959  
## alternative hypothesis: true tau is not equal to 0  
## sample estimates:  
## tau   
## 0.2238888   
##   
##   
## $google2  
##   
## Kendall's rank correlation tau  
##   
## data: x and GTD$tox\_eset  
## z = 1.2315, p-value = 0.2181  
## alternative hypothesis: true tau is not equal to 0  
## sample estimates:  
## tau   
## 0.1407854   
##   
##   
## $google1\_csusz  
##   
## Kendall's rank correlation tau  
##   
## data: x and GTD$tox\_eset  
## z = 1.2031, p-value = 0.229  
## alternative hypothesis: true tau is not equal to 0  
## sample estimates:  
## tau   
## 0.1325483   
##   
##   
## $google2\_csusz  
##   
## Kendall's rank correlation tau  
##   
## data: x and GTD$tox\_eset  
## z = 1.4637, p-value = 0.1433  
## alternative hypothesis: true tau is not equal to 0  
## sample estimates:  
## tau   
## 0.1689524   
##   
##   
## $google1\_2csusz  
##   
## Kendall's rank correlation tau  
##   
## data: x and GTD$tox\_eset  
## z = 0.61742, p-value = 0.537  
## alternative hypothesis: true tau is not equal to 0  
## sample estimates:  
## tau   
## 0.06898685   
##   
##   
## $google2\_2csusz  
##   
## Kendall's rank correlation tau  
##   
## data: x and GTD$tox\_eset  
## z = 2.9679, p-value = 0.002998  
## alternative hypothesis: true tau is not equal to 0  
## sample estimates:  
## tau   
## 0.3467076

library(lubridate)

##   
## Attaching package: 'lubridate'

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

counts$Var1 <- ymd(counts$Var1)  
  
# extract week year  
  
week <- function(x)format(x, '%Y.%W')  
  
counts$weekyear<-week(counts$Var1)  
  
month<- function(x)format(x, '%Y.%B')  
  
counts$month<-month(counts$Var1)  
#create a df with 0-counts for all days  
  
n<-rep(0, 1369)  
n<-as.data.frame(n)  
n$Var1<-seq(as.Date("2009-09-14"), as.Date("2013-06-13"), "day" )  
  
#join them  
library(dplyr)

##   
## Attaching package: 'dplyr'

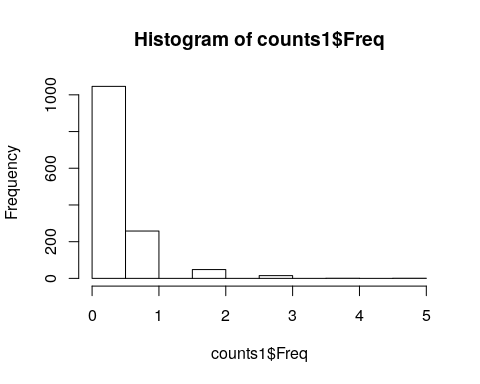
## The following objects are masked from 'package:stats':  
##   
## filter, lag

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

counts<-full\_join(n, counts, by="Var1")  
counts$Freq[is.na(counts$Freq)] <- 0   
  
# extract week year  
  
week <- function(x)format(x, '%Y.%W')  
  
counts$weekyear<-week(counts$Var1)  
summary(counts)

## n Var1 Freq weekyear   
## Min. :0 Min. :2009-09-14 Min. :0.000 Length:1369   
## 1st Qu.:0 1st Qu.:2010-08-22 1st Qu.:0.000 Class :character   
## Median :0 Median :2011-07-30 Median :0.000 Mode :character   
## Mean :0 Mean :2011-07-30 Mean :0.298   
## 3rd Qu.:0 3rd Qu.:2012-07-06 3rd Qu.:0.000   
## Max. :0 Max. :2013-06-13 Max. :5.000   
## month   
## Length:1369   
## Class :character   
## Mode :character   
##   
##   
##

counts$month<-month(counts$Var1)  
  
  
  
# extract week year  
  
GTD$weekyear<-week(GTD$date)  
GTD$month<-month(GTD$date)  
  
#join  
  
library(dplyr)  
counts1<-full\_join(counts, GTD, by="month")  
hist(counts1$Freq)



mean(counts1$Freq)

## [1] 0.2980278

var(counts1$Freq)

## [1] 0.3687169

counts$date<-NULL  
counts1$n<-NULL  
summary(counts1)

## Var1 Freq weekyear.x month   
## Min. :2009-09-14 Min. :0.000 Length:1369 Length:1369   
## 1st Qu.:2010-08-22 1st Qu.:0.000 Class :character Class :character   
## Median :2011-07-30 Median :0.000 Mode :character Mode :character   
## Mean :2011-07-30 Mean :0.298   
## 3rd Qu.:2012-07-06 3rd Qu.:0.000   
## Max. :2013-06-13 Max. :5.000   
##   
## datum tox\_eset tox\_sz\_eset google1   
## Length:1369 Min. : 0.000 Min. :0.000 Min. : 12.00   
## Class :character 1st Qu.: 1.000 1st Qu.:0.000 1st Qu.: 23.00   
## Mode :character Median : 2.000 Median :1.000 Median : 28.00   
## Mean : 2.866 Mean :1.897 Mean : 36.46   
## 3rd Qu.: 4.000 3rd Qu.:3.000 3rd Qu.: 46.00   
## Max. :13.000 Max. :8.000 Max. :100.00   
## NA's :30   
## google2 google1\_csusz google2\_csusz google1\_2csusz   
## Min. : 0.00 Min. : 12.00 Min. : 0.00 Min. : 12.00   
## 1st Qu.: 0.00 1st Qu.: 23.00 1st Qu.: 0.00 1st Qu.: 23.00   
## Median :15.00 Median : 28.00 Median :15.00 Median : 33.00   
## Mean :17.01 Mean : 36.86 Mean :17.04 Mean : 37.45   
## 3rd Qu.:25.00 3rd Qu.: 46.00 3rd Qu.:25.00 3rd Qu.: 49.00   
## Max. :89.00 Max. :100.00 Max. :89.00 Max. :100.00   
## NA's :17 NA's :17 NA's :48   
## google2\_2csusz date weekyear.y   
## Min. : 0.00 Min. :2009-09-14 Length:1369   
## 1st Qu.: 0.00 1st Qu.:2010-08-14 Class :character   
## Median :16.00 Median :2011-07-14 Mode :character   
## Mean :17.62 Mean :2011-07-28   
## 3rd Qu.:25.00 3rd Qu.:2012-07-14   
## Max. :89.00 Max. :2013-06-14   
## NA's :48

ts <- ts(counts1$Freq, start=c(2009, 9, 14), end=c(2013, 6, 14), frequency=365)   
counts1$weekday<-weekdays(counts1$Var1)  
  
  
kruskal.test(counts1$Freq, as.factor(counts1$weekday))

##   
## Kruskal-Wallis rank sum test  
##   
## data: counts1$Freq and as.factor(counts1$weekday)  
## Kruskal-Wallis chi-squared = 20.621, df = 6, p-value = 0.002145

library(dunn.test)  
dunn.test(counts1$Freq, as.factor(counts1$weekday), method="holm")

## Kruskal-Wallis rank sum test  
##   
## data: x and group  
## Kruskal-Wallis chi-squared = 20.621, df = 6, p-value = 0  
##   
##   
## Comparison of x by group   
## (Holm)   
## Col Mean-|  
## Row Mean | Dienstag Donnerst Freitag Mittwoch Montag Samstag  
## ---------+------------------------------------------------------------------  
## Donnerst | -1.608327  
## | 0.6466  
## |  
## Freitag | -1.891698 -0.285429  
## | 0.4390 0.3877  
## |  
## Mittwoch | -0.563022 1.045304 1.329397  
## | 0.8601 1.0000 0.9186  
## |  
## Montag | -1.277508 0.330818 0.615825 -0.714485  
## | 0.9064 0.7408 1.0000 1.0000  
## |  
## Samstag | -3.093931 -1.487662 -1.200698 -2.531629 -1.818058  
## | 0.0188\* 0.7526 0.9195 0.1022 0.4834  
## |  
## Sonntag | -3.691884 -2.085615 -1.797887 -3.129582 -2.416010 -0.597189  
## | 0.0023\* 0.2961 0.4693 0.0175\* 0.1334 1.0000  
##   
## alpha = 0.05  
## Reject Ho if p <= alpha/2

#crosscorr months  
  
ccfKendall1<- sapply( -12:12, function(l) cor.test( GTD$tox\_eset, Hmisc::Lag(GTD$google1,l),method =  
"kendall", use ="complete.obs")$estimate )

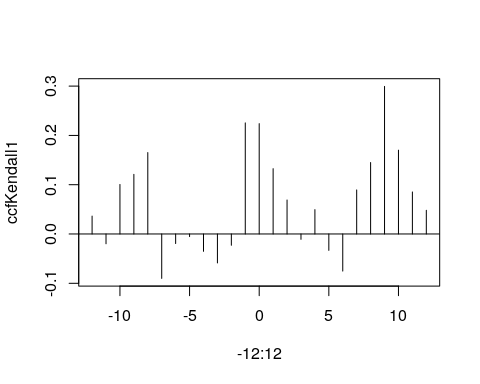
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties

## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google1, l), method =  
## "kendall", : Cannot compute exact p-value with ties

ccfKendall1

## tau tau tau tau tau tau   
## 0.036322034 -0.019818244 0.100512969 0.120989147 0.165190921 -0.090095249   
## tau tau tau tau tau tau   
## -0.019363707 -0.005267546 -0.035190406 -0.058742526 -0.022894552 0.225398852   
## tau tau tau tau tau tau   
## 0.223888816 0.132548329 0.068986848 -0.010838497 0.049411564 -0.033250326   
## tau tau tau tau tau tau   
## -0.075258492 0.089419556 0.144981647 0.299442426 0.170067700 0.085270952   
## tau   
## 0.048054092

plot(-12:12,ccfKendall1,type="h")  
abline(h=0)



ccfKendall2<- sapply( -12:12, function(l) cor.test( GTD$tox\_eset, Hmisc::Lag(GTD$google2,l),method =  
"kendall", use ="complete.obs")$estimate )

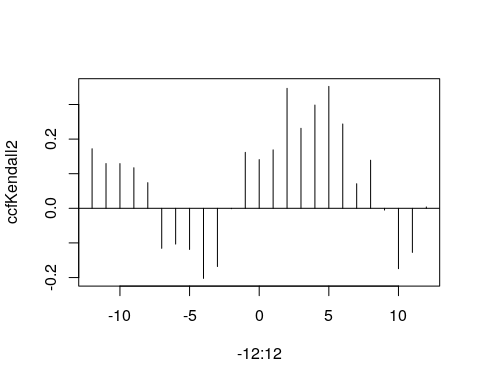
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties

## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties  
  
## Warning in cor.test.default(GTD$tox\_eset, Hmisc::Lag(GTD$google2, l), method =  
## "kendall", : Cannot compute exact p-value with ties

ccfKendall2

## tau tau tau tau tau tau   
## 0.172093340 0.129413752 0.129271414 0.117250667 0.074134942 -0.115567343   
## tau tau tau tau tau tau   
## -0.103329166 -0.118828999 -0.202487709 -0.168138164 -0.001239466 0.161675582   
## tau tau tau tau tau tau   
## 0.140785431 0.168952379 0.346707551 0.231214302 0.298393835 0.352305773   
## tau tau tau tau tau tau   
## 0.243726558 0.071046051 0.138945369 -0.005085476 -0.174410636 -0.127372989   
## tau   
## 0.003953874

plot(-12:12,ccfKendall2,type="h")  
abline(h=0)



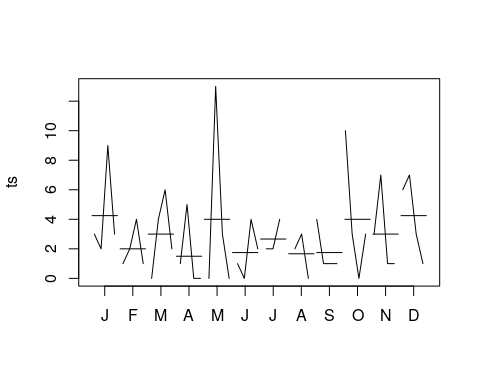
library(glarma)  
  
  
  
##monthly data  
setwd("~/Rprojects/GHB")  
library(haven)  
data <- read\_sav("GHB\_tox\_adat\_140711.sav")  
GTD <- read\_sav("GHB\_tox\_google.sav")  
GTD<-GTD[-47,]  
library(forecast)

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

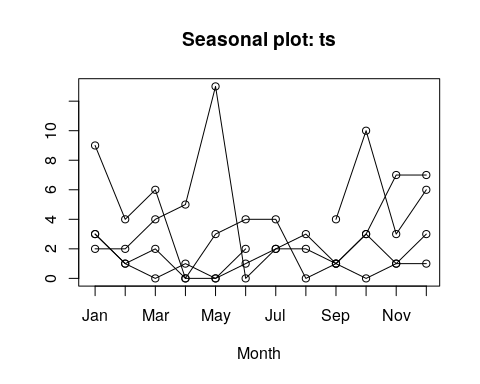
##   
## Attaching package: 'forecast'

## The following object is masked from 'package:glarma':  
##   
## forecast

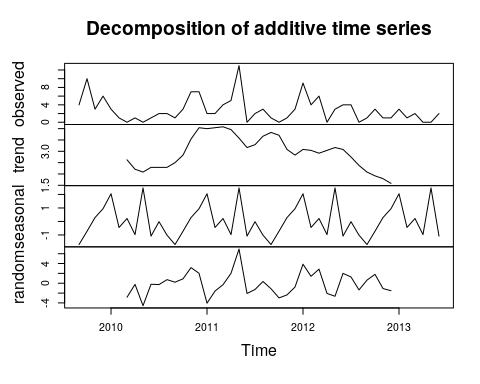
ts <- ts(GTD$tox\_eset, start=c(2009, 9, 14), end=c(2013, 6, 14), frequency=12)   
monthplot(ts)



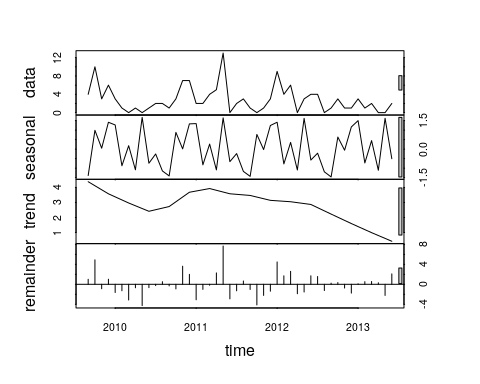
seasonplot(ts)



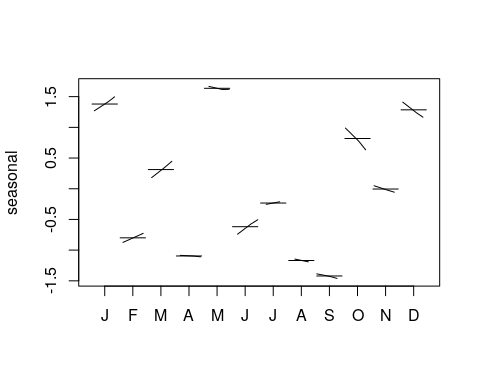
decomp<-decompose(ts)  
plot(decomp)



decomp<-stl(ts, s.window=12)  
plot(decomp)

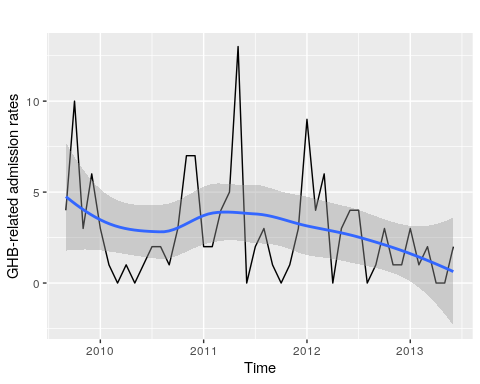


monthplot(decomp, choice="seasonal")

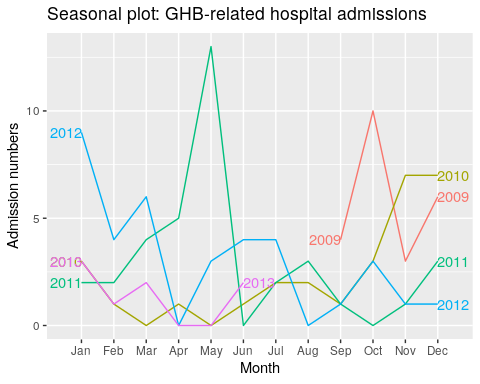


autoplot(ts)+ geom\_smooth(method="loess") + ylab("GHB-related admission rates")

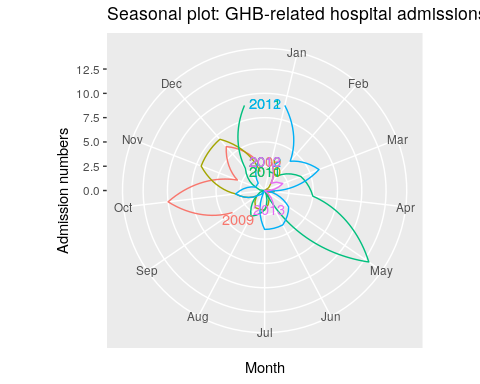
## `geom\_smooth()` using formula 'y ~ x'



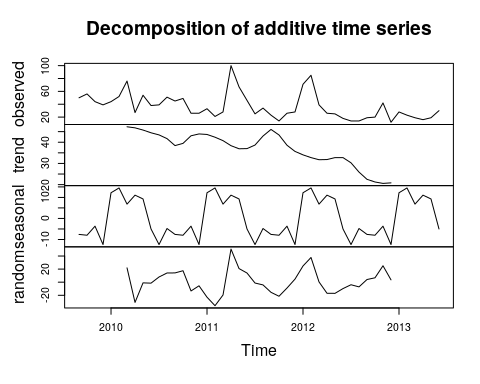
ggseasonplot(ts, year.labels=TRUE, year.labels.left=TRUE) +  
 ylab("Admission numbers") +  
 ggtitle("Seasonal plot: GHB-related hospital admissions")



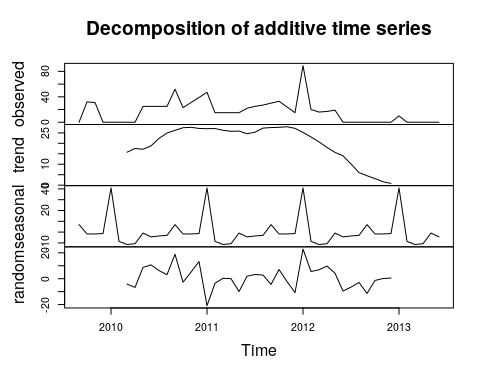
ggseasonplot(ts, year.labels=TRUE, year.labels.left=TRUE, polar=TRUE) +  
 ylab("Admission numbers") +  
 ggtitle("Seasonal plot: GHB-related hospital admissions")



tsg1 <- ts(GTD$google1, start=c(2009, 9, 14), end=c(2013, 6, 14), frequency=12)   
decomp<-decompose(tsg1)  
plot(decomp)



tsg2 <- ts(GTD$google2, start=c(2009, 9, 14), end=c(2013, 6, 14), frequency=12)   
decomp<-decompose(tsg2)  
plot(decomp)



library(MASS)

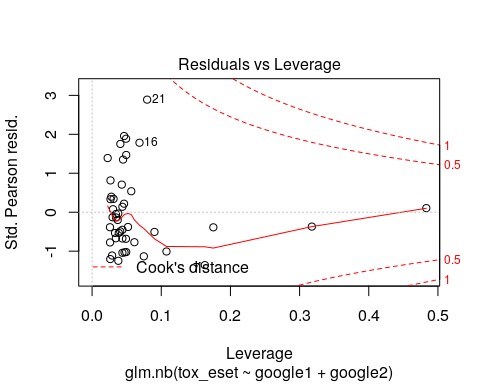
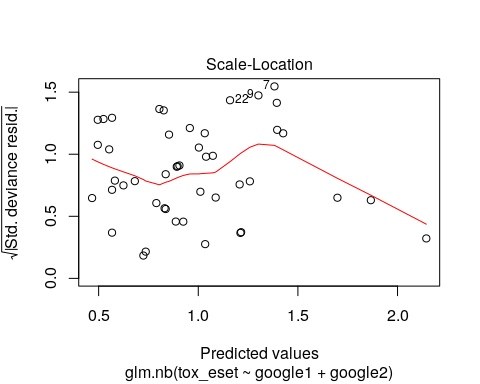
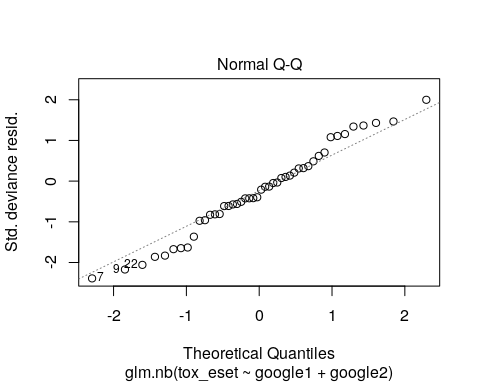
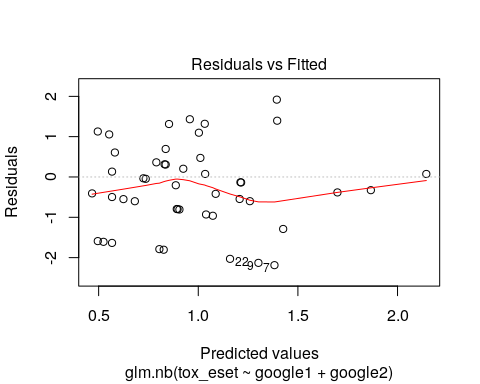
##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

fit<- glm.nb(tox\_eset ~ google1 + google2, data=GTD)  
summary(fit)

##   
## Call:  
## glm.nb(formula = tox\_eset ~ google1 + google2, data = GTD, init.theta = 2.527927252,   
## link = log)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.1873 -0.8047 -0.2666 0.3507 1.9180   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.295492 0.285043 1.037 0.2999   
## google1 0.014301 0.006452 2.217 0.0266 \*  
## google2 0.009366 0.007001 1.338 0.1809   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for Negative Binomial(2.5279) family taken to be 1)  
##   
## Null deviance: 60.886 on 45 degrees of freedom  
## Residual deviance: 51.524 on 43 degrees of freedom  
## AIC: 199.91  
##   
## Number of Fisher Scoring iterations: 1  
##   
##   
## Theta: 2.53   
## Std. Err.: 1.10   
##   
## 2 x log-likelihood: -191.908

plot(fit)



estim<-cbind(fit$coefficients, confint(fit))

## Waiting for profiling to be done...

estim

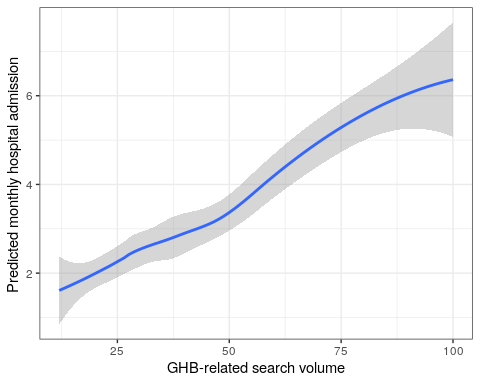
## 2.5 % 97.5 %  
## (Intercept) 0.295491754 -0.280905846 0.85675186  
## google1 0.014301476 0.001369025 0.02787862  
## google2 0.009366058 -0.004277957 0.02387441

exp(estim)

## 2.5 % 97.5 %  
## (Intercept) 1.343787 0.7550994 2.355497  
## google1 1.014404 1.0013700 1.028271  
## google2 1.009410 0.9957312 1.024162

GTD$predicted<-predict(fit)  
GTD$predictedcounts<-exp(GTD$predicted)  
library(ggplot2)  
p<-ggplot(GTD, aes(google1, predictedcounts)) + geom\_smooth(method="loess") + xlab("GHB-related search volume") + ylab("Predicted monthly hospital admission ") + theme\_bw()  
p

## `geom\_smooth()` using formula 'y ~ x'



m2 <- glm(tox\_eset ~ google1 + google2, data=GTD, family = "poisson")  
pchisq(2 \* (logLik(fit) - logLik(m2)), df = 1, lower.tail = FALSE)

## 'log Lik.' 7.376732e-05 (df=4)

library(pscl) #ceck zero inflation

## Classes and Methods for R developed in the  
## Political Science Computational Laboratory  
## Department of Political Science  
## Stanford University  
## Simon Jackman  
## hurdle and zeroinfl functions by Achim Zeileis

mod2 <- zeroinfl(tox\_eset ~ google1 + google2, data=GTD, dist="negbin")  
summary(mod2)

##   
## Call:  
## zeroinfl(formula = tox\_eset ~ google1 + google2, data = GTD, dist = "negbin")  
##   
## Pearson residuals:  
## Min 1Q Median 3Q Max   
## -1.2585 -0.6949 -0.1920 0.3426 2.9537   
##   
## Count model coefficients (negbin with log link):  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.357762 0.285278 1.254 0.2098   
## google1 0.012615 0.006603 1.911 0.0561 .  
## google2 0.010721 0.007047 1.521 0.1282   
## Log(theta) 1.012353 0.445219 2.274 0.0230 \*  
##   
## Zero-inflation model coefficients (binomial with logit link):  
## Estimate Std. Error z value Pr(>|z|)  
## (Intercept) 5.576 5273.585 0.001 0.999  
## google1 -2.468 104.492 -0.024 0.981  
## google2 1.219 200.838 0.006 0.995  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1   
##   
## Theta = 2.7521   
## Number of iterations in BFGS optimization: 31   
## Log-likelihood: -94.28 on 7 Df

aic<-AIC(fit, mod2, m2) #zeoinf is slightly better than poisson, but it is not justified (no alterative known route to count 0 generation)- negbin performs the best  
aic

## df AIC  
## fit 4 199.9079  
## mod2 7 202.5678  
## m2 3 213.6196

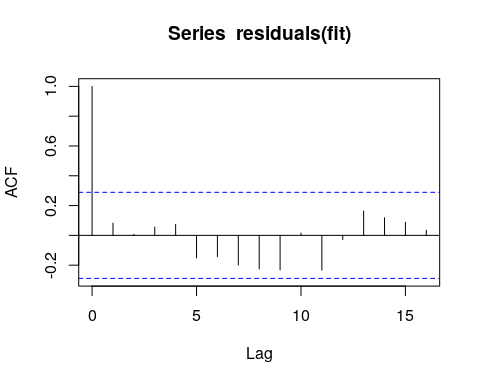
mean(GTD$tox\_eset)

## [1] 2.847826

var(GTD$tox\_eset)

## [1] 8.042995

GTD<-as.data.frame(GTD)  
y<-as.vector(GTD[,2])  
x<-as.matrix(GTD[,c(4,5)])  
acf(residuals(fit))



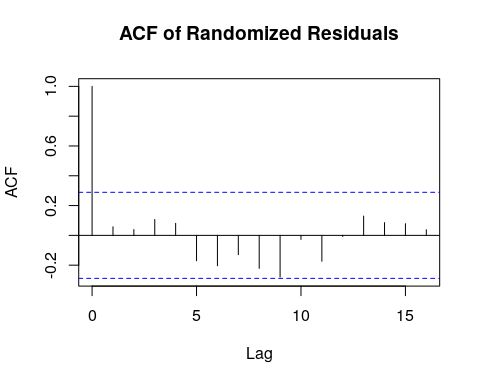
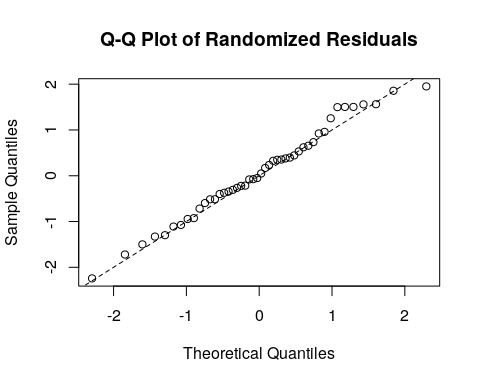
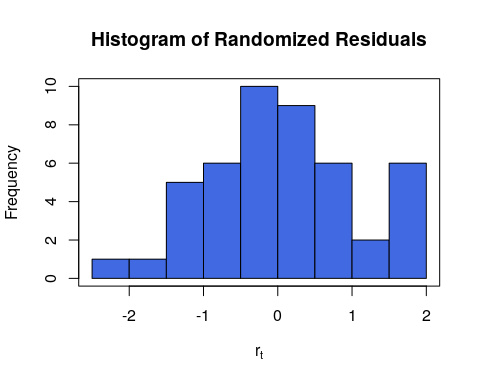
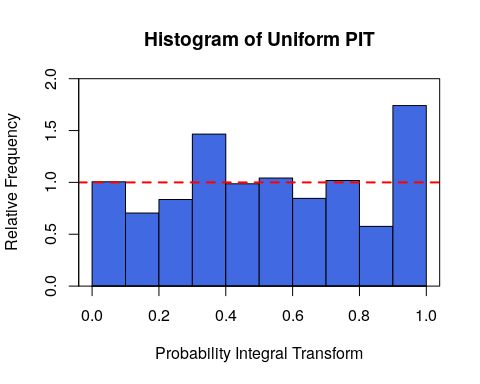
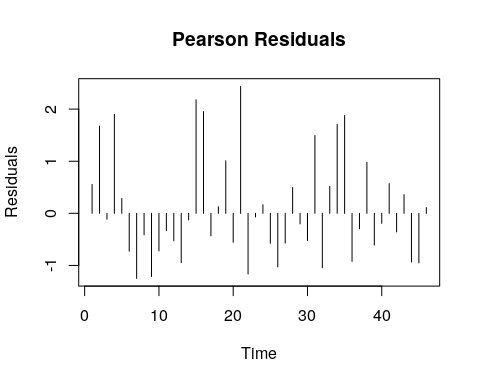
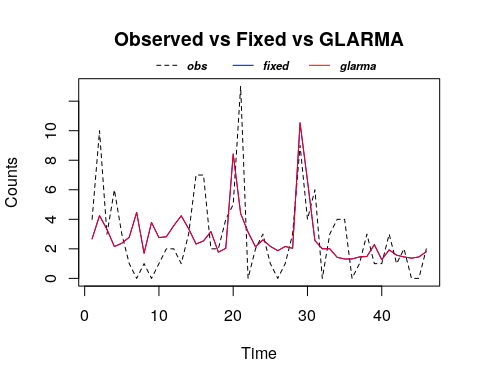
glarma<-glarma(y,x, type="NegBin") #I cannot add both phi and theta lags at the same time- this has the lowest aic  
glarma

##   
## Call: glarma(y = y, X = x, type = "NegBin")  
##   
## Negative Binomial Parameter:  
## alpha   
## 2.392726   
##   
## Linear Model Coefficients:  
## google1 google2   
## 0.01970080 0.01073953   
##   
## Degrees of Freedom: 45 Total (i.e. Null); 43 Residual  
## Null Deviance: 60.88565   
## Residual Deviance: 46.72501   
## AIC: 196.9176

summary(glarma)

##   
## Call: glarma(y = y, X = x, type = "NegBin")  
##   
## Pearson Residuals:  
## Min 1Q Median 3Q Max   
## -1.2484 -0.6004 -0.1975 0.5455 2.4351   
##   
## Negative Binomial Parameter:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## alpha 2.393 1.150 2.081 0.0374 \*  
##   
## Linear Model Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## google1 0.019701 0.004032 4.886 1.03e-06 \*\*\*  
## google2 0.010740 0.008014 1.340 0.18   
##   
## Null deviance: 60.886 on 45 degrees of freedom  
## Residual deviance: 46.725 on 43 degrees of freedom  
## AIC: 196.9176   
##   
## Number of Fisher Scoring iterations: 30  
##   
## LRT and Wald Test:  
## Alternative hypothesis: model is a GLARMA process  
## Null hypothesis: model is a GLM with the same regression structure  
## Statistic p-value   
## LR Test 2 <2e-16 \*\*\*  
## Wald Test 0 1   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

plot(glarma)



#adding differnt MA and AR orders cannot wont improve aic. LRT and WALd indicates a GLM not a glarma process  
mySolve2 <- function (A) {  
 ErrCode <- 0  
 Ainv <- try(solve(A), TRUE)  
 if (is.character(Ainv)) {  
 ErrCode <- 1  
 Ainv <- A  
 }  
 list(Ainv = Ainv, ErrCode = ErrCode)  
}  
assignInNamespace("mySolve",mySolve2,"glarma")  
  
  
glarma2<-glarma(y,x, phiLags=c(1), type="NegBin")  
summary(glarma2)

##   
## Call: glarma(y = y, X = x, type = "NegBin", phiLags = c(1))  
##   
## Pearson Residuals:  
## Min 1Q Median 3Q Max   
## -1.2357 -0.6344 -0.1949 0.5886 2.5793   
##   
## Negative Binomial Parameter:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## alpha 2.382 1.134 2.1 0.0357 \*  
##   
## GLARMA Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)  
## phi\_1 0.06397 0.14256 0.449 0.654  
##   
## Linear Model Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## google1 0.019660 0.004313 4.559 5.15e-06 \*\*\*  
## google2 0.010212 0.009049 1.129 0.259   
##   
## Null deviance: 60.886 on 45 degrees of freedom  
## Residual deviance: 45.980 on 42 degrees of freedom  
## AIC: 198.7417   
##   
## Number of Fisher Scoring iterations: 30  
##   
## LRT and Wald Test:  
## Alternative hypothesis: model is a GLARMA process  
## Null hypothesis: model is a GLM with the same regression structure  
## Statistic p-value  
## LR Test 2.176 0.140  
## Wald Test 0.201 0.654

AIC(glarma2)

## numeric(0)

glarma2<-glarma(y,x, phiLags=c(1,2), type="NegBin")  
summary(glarma2)

##   
## Call: glarma(y = y, X = x, type = "NegBin", phiLags = c(1, 2))  
##   
## Pearson Residuals:  
## Min 1Q Median 3Q Max   
## -1.2275 -0.6718 -0.2047 0.6747 2.7376   
##   
## Negative Binomial Parameter:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## alpha 2.364 1.167 2.025 0.0429 \*  
##   
## GLARMA Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)  
## phi\_1 0.10898 0.15520 0.702 0.483  
## phi\_2 0.05128 0.16612 0.309 0.758  
##   
## Linear Model Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## google1 0.020646 0.005626 3.670 0.000243 \*\*\*  
## google2 0.006385 0.010926 0.584 0.558973   
##   
## Null deviance: 60.886 on 45 degrees of freedom  
## Residual deviance: 44.671 on 41 degrees of freedom  
## AIC: 200.9186   
##   
## Number of Fisher Scoring iterations: 30  
##   
## LRT and Wald Test:  
## Alternative hypothesis: model is a GLARMA process  
## Null hypothesis: model is a GLM with the same regression structure  
## Statistic p-value  
## LR Test 1.999 0.368  
## Wald Test 0.660 0.719

glarma2<-glarma(y,x, phiLags=c(1,2,3), type="NegBin")  
summary(glarma2)

##   
## Call: glarma(y = y, X = x, type = "NegBin", phiLags = c(1, 2, 3))  
##   
## Pearson Residuals:  
## Min 1Q Median 3Q Max   
## -1.3862 -0.6774 -0.2389 0.7996 3.3307   
##   
## Negative Binomial Parameter:  
## Estimate Std.Error z-ratio Pr(>|z|)  
## alpha 2.212 1.831 1.208 0.227  
##   
## GLARMA Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## phi\_1 0.19282 0.11584 1.664 0.096 .  
## phi\_2 0.08582 0.10740 0.799 0.424   
## phi\_3 0.11137 0.16220 0.687 0.492   
##   
## Linear Model Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## google1 0.021345 0.005360 3.982 6.83e-05 \*\*\*  
## google2 0.003588 0.009809 0.366 0.715   
##   
## Null deviance: 60.886 on 45 degrees of freedom  
## Residual deviance: 55.043 on 40 degrees of freedom  
## AIC: 205.1847   
##   
## Number of Fisher Scoring iterations: 30  
##   
## LRT and Wald Test:  
## Alternative hypothesis: model is a GLARMA process  
## Null hypothesis: model is a GLM with the same regression structure  
## Statistic p-value  
## LR Test -0.267 1.000  
## Wald Test 3.638 0.303

glarma2<-glarma(y,x, thetaLags=c(1), type="NegBin")  
summary(glarma2)

##   
## Call: glarma(y = y, X = x, type = "NegBin", thetaLags = c(1))  
##   
## Pearson Residuals:  
## Min 1Q Median 3Q Max   
## -1.2366 -0.6311 -0.1946 0.5834 2.5768   
##   
## Negative Binomial Parameter:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## alpha 2.382 1.142 2.086 0.037 \*  
##   
## GLARMA Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)  
## theta\_1 0.05831 0.14361 0.406 0.685  
##   
## Linear Model Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## google1 0.019658 0.004348 4.521 6.14e-06 \*\*\*  
## google2 0.010306 0.009147 1.127 0.26   
##   
## Null deviance: 60.886 on 45 degrees of freedom  
## Residual deviance: 46.022 on 42 degrees of freedom  
## AIC: 198.7553   
##   
## Number of Fisher Scoring iterations: 30  
##   
## LRT and Wald Test:  
## Alternative hypothesis: model is a GLARMA process  
## Null hypothesis: model is a GLM with the same regression structure  
## Statistic p-value  
## LR Test 2.162 0.141  
## Wald Test 0.165 0.685

glarma2<-glarma(y,x, thetaLags=c(3), type="NegBin")  
summary(glarma2)

##   
## Call: glarma(y = y, X = x, type = "NegBin", thetaLags = c(3))  
##   
## Pearson Residuals:  
## Min 1Q Median 3Q Max   
## -1.1667 -0.5129 -0.2423 0.4758 2.8353   
##   
## Negative Binomial Parameter:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## alpha 2.6454 0.8352 3.167 0.00154 \*\*  
##   
## GLARMA Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)  
## theta\_3 0.01438 0.24569 0.059 0.953  
##   
## Linear Model Coefficients:  
## Estimate Std.Error z-ratio Pr(>|z|)   
## google1 0.017992 0.004934 3.647 0.000266 \*\*\*  
## google2 0.012549 0.010110 1.241 0.214491   
##   
## Null deviance: 60.886 on 45 degrees of freedom  
## Residual deviance: 42.365 on 42 degrees of freedom  
## AIC: 198.733   
##   
## Number of Fisher Scoring iterations: 30  
##   
## LRT and Wald Test:  
## Alternative hypothesis: model is a GLARMA process  
## Null hypothesis: model is a GLM with the same regression structure  
## Statistic p-value  
## LR Test 2.185 0.139  
## Wald Test 0.003 0.953