Time Series Group Project

Ann Eitrheim, Alvin Haryanto, Dan Tallarico

Each of these time series represent a number of daily views of a different Wikipedia article. The page/article names contain the Wikipedia project (e.g. en.wikipedia.org), type of access/traffic (e.g. desktop) and type of agent (e.g. spid). In other words, each article name has the following format: 'name_project_access_agent' (e.g. 'AKB48_zh.wikipedia.org_all-access_spid'). Unfortunately, the data source for this dataset does not distinguish between traffic values of zero and missing values. A missing value may mean the traffic was zero or that the data is not available for that day.

Data Preprocessing

```
train_2 <- read.csv('train_2.csv') #The second stage will use training data up until September 1st, 2017.
```

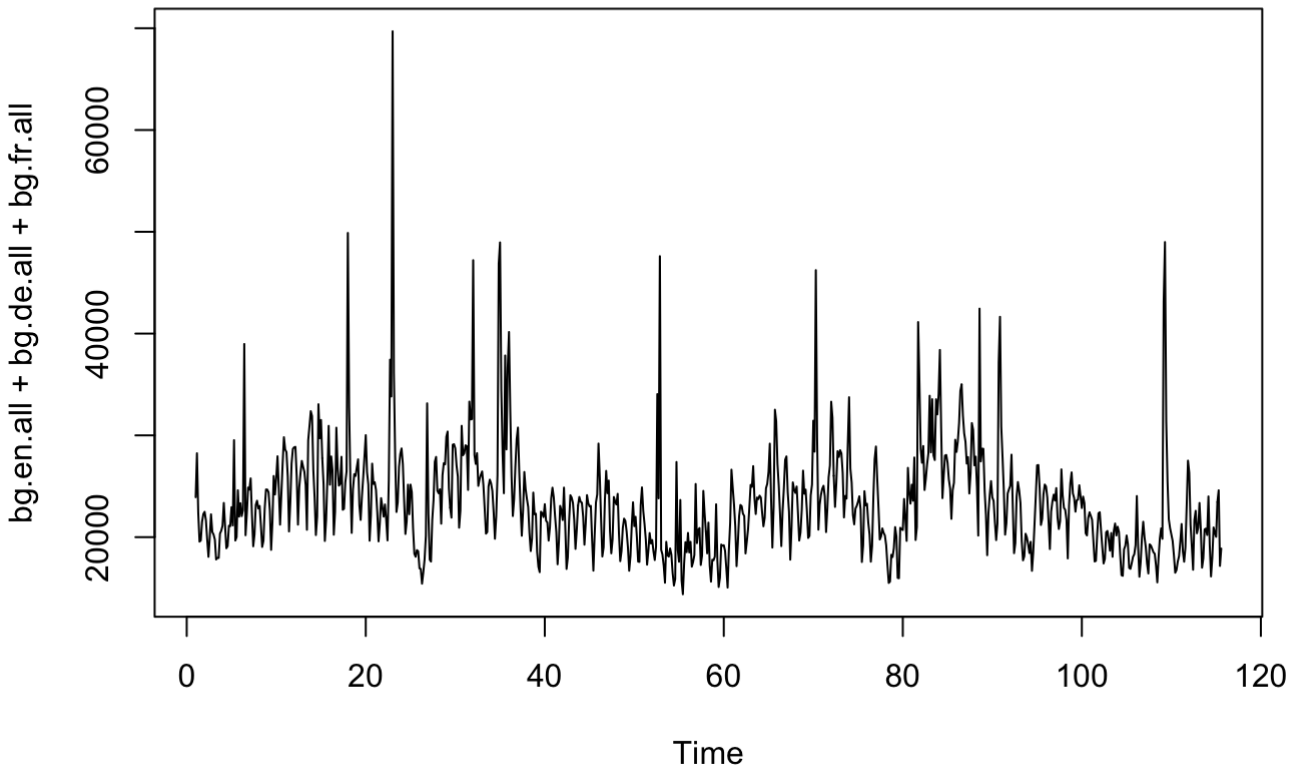
```
library(dplyr)
library(tibble)
library(tidyr)
library(stringr)
library(forecast)
library(tseries)
library(hts)
library(TSA)

train <- train_2 #[complete.cases(train_2), ]
tdates <- train %>% select(-Page) #df with no page names

#separating the mediawiki, wikimedia, and wikipedia data
foo <- train %>% select(Page) %>% rownames_to_column()
mediawiki <- foo %>% filter(str_detect(Page, "mediawiki"))
wikimedia <- foo %>% filter(str_detect(Page, "wikimedia"))
wikipedia <- foo %>% filter(str_detect(Page, "wikipedia")) %>%
  filter(!str_detect(Page, "wikimedia")) %>%
  filter(!str_detect(Page, "mediawiki"))

#separating the page name into topic, location, access, and agent
wikipedia <- wikipedia %>%
  separate(Page, into = c("foo", "bar"), sep = ".wikipedia.org_") %>%
  separate(foo, into = c("article", "locale"), sep = -3) %>%
  separate(bar, into = c("access", "agent"), sep = "_") %>%
  mutate(locale = str_sub(locale,2,3))
wikimedia <- wikimedia %>%
  separate(Page, into = c("article", "bar"), sep = "_commons.wikimedia.org_") %>%
  separate(bar, into = c("access", "agent"), sep = "_") %>%
  add_column(locale = "wikmed")
mediawiki <- mediawiki %>%
  separate(Page, into = c("article", "bar"), sep = "_www.mediawiki.org_") %>%
  separate(bar, into = c("access", "agent"), sep = "_") %>%
  add_column(locale = "medwik")

#rejoining mediawiki, wikimedia, and wikipedia into one df
tpages <- wikipedia %>%
  full_join(wikimedia, by = c("rowname", "article", "locale", "access", "agent")) %>%
  full_join(mediawiki, by = c("rowname", "article", "locale", "access", "agent"))

sample_n(tpages, 5)
```

```
##       rowname                    article locale     access      agent
## 72487   90342                    Albania     es all-access all-agents
## 80662   98517 Жарков,_Алексей_Дмитриевич     ru all-access all-agents
## 35649   42054     John_Eleuthère_du_Pont     en all-access all-agents
## 21964   28369                      張莊園     zh all-access all-agents
## 90858  108713              胭脂_(电视剧)     zh mobile-web all-agents
```

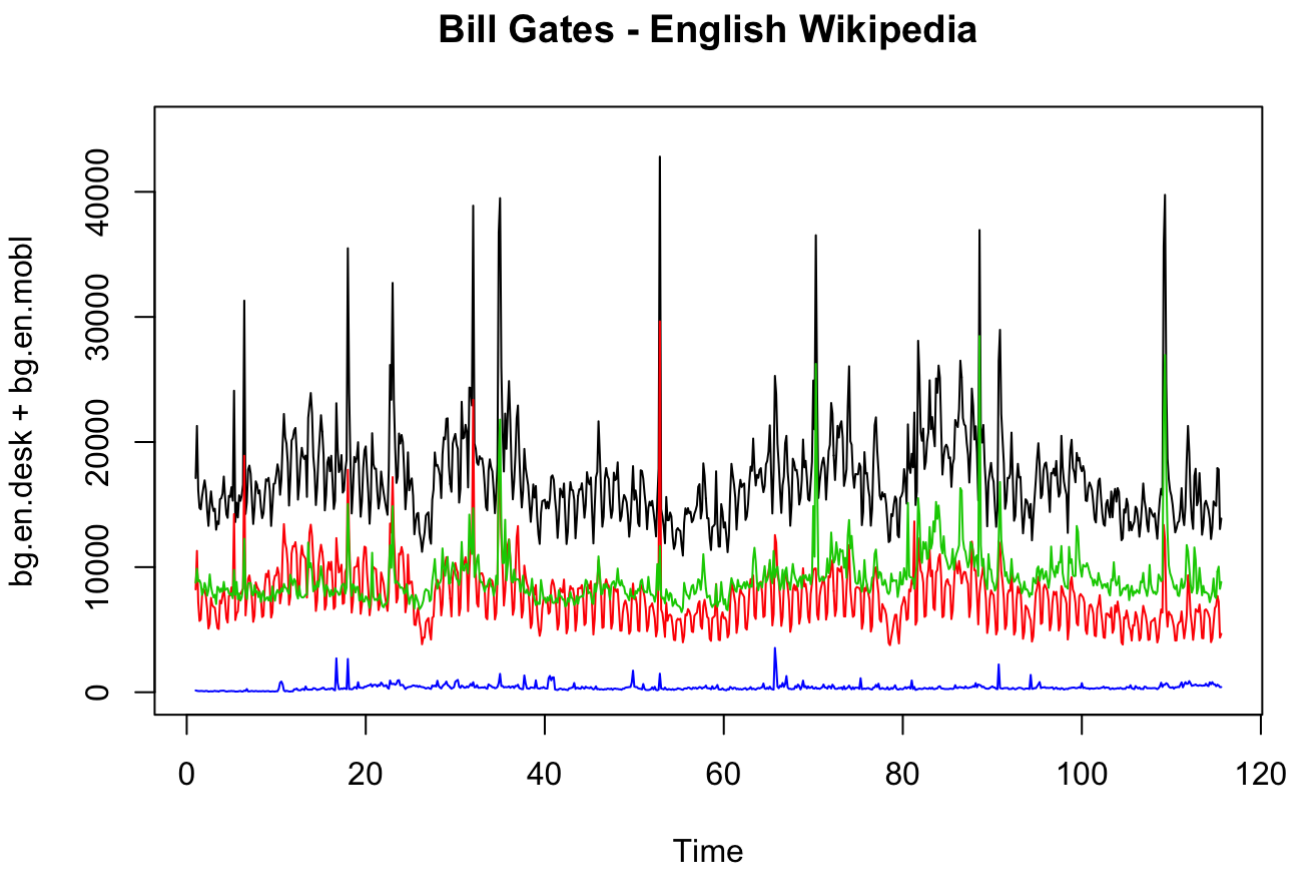
Exploratory Data Analysis

```
bg.fr.desk <- ts(t(tdates[6279,]),frequency = 7)
bg.fr.mobl <- ts(t(tdates[53362,]), frequency = 7)
bg.fr.spid <- ts(t(tdates[129691,]), frequency = 7)
bg.fr.all <- bg.fr.desk + bg.fr.mobl + bg.fr.spid

bg.en.desk <- ts(t(tdates[11057,]), frequency = 7)
bg.en.mobl <- ts(t(tdates[73051,]), frequency = 7)
bg.en.spid <- ts(t(tdates[34899,]), frequency = 7)
bg.en.all <- bg.en.desk + bg.en.mobl + bg.en.spid

bg.de.desk <- ts(t(tdates[67313,]), frequency = 7)
bg.de.mobl <- ts(t(tdates[116365,]), frequency = 7)
bg.de.spid <- ts(t(tdates[48744,]), frequency = 7)
bg.de.all <- bg.de.desk + bg.de.mobl + bg.de.spid

bg.es.desk <- ts(t(tdates[70902,]), frequency = 7)
bg.es.mobl <- ts(t(tdates[95498,]), frequency = 7)
bg.es.spid <- ts(t(tdates[143106,]), frequency = 7)
bg.es.all <- bg.es.desk + bg.es.mobl + bg.es.spid

bg.all <- bg.en.all + bg.de.all + bg.fr.all + bg.es.all
```

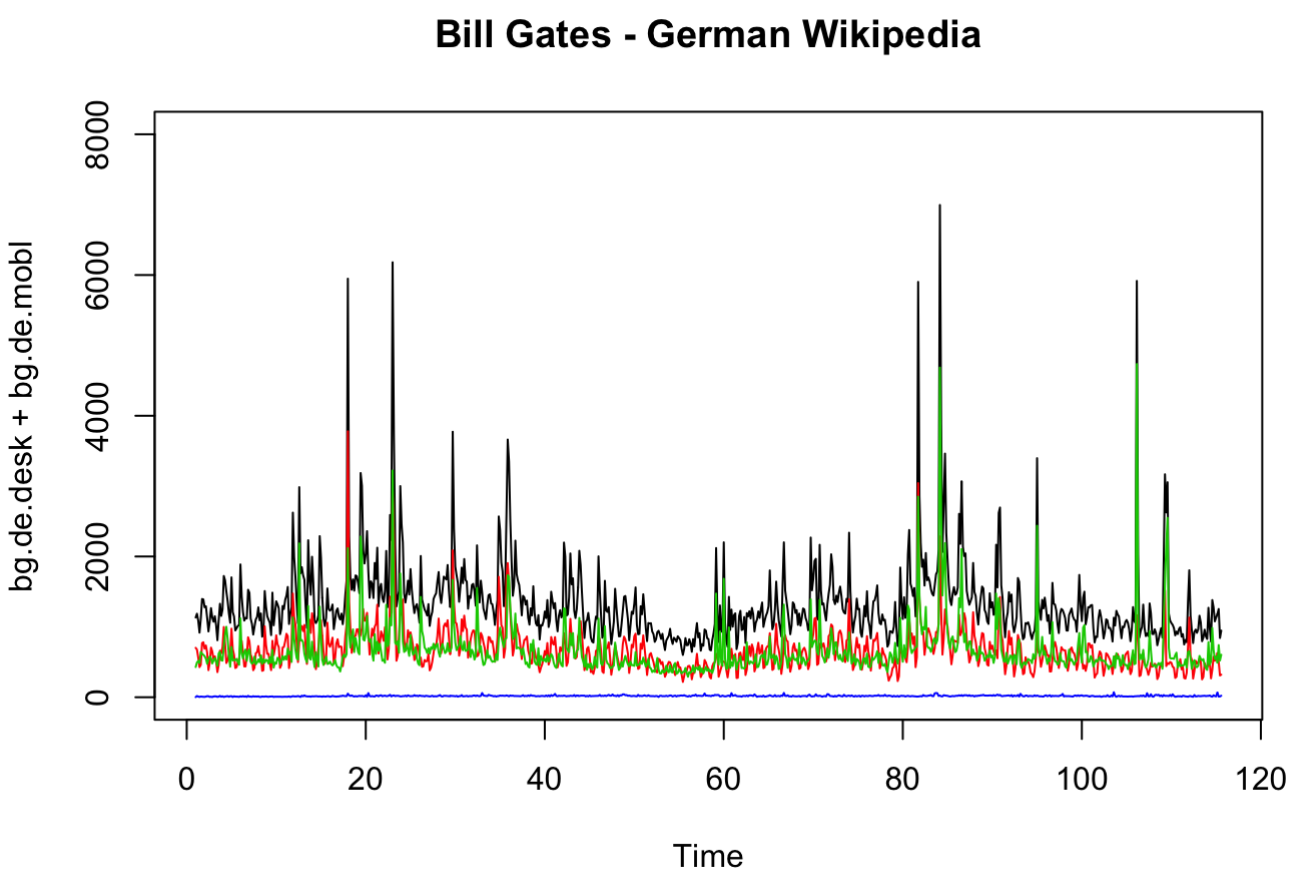
```
plot.ts(bg.all)
```



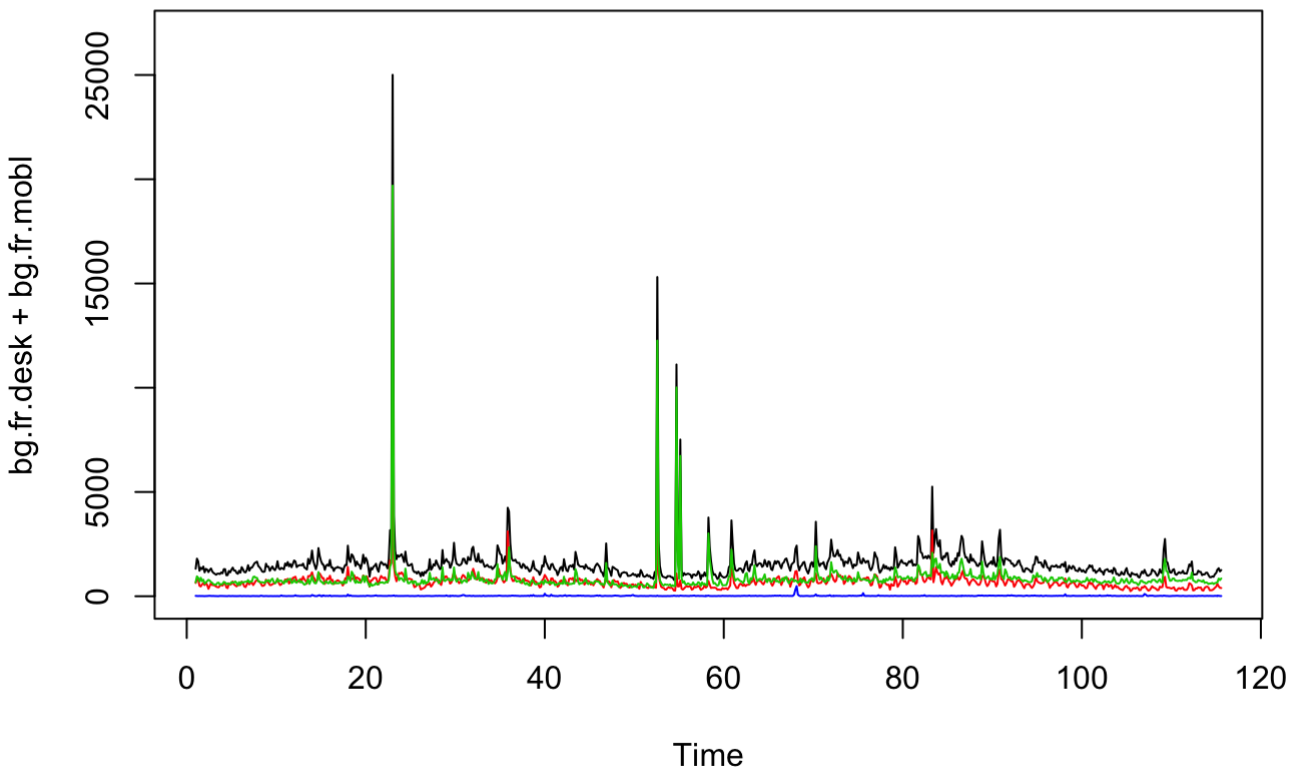
```
plot(bg.en.all, ylim = c(0,45000), main = "Bill Gates - English Wikipedia")
lines(bg.en.desk, col = 2)
lines(bg.en.mobl, col = 3)
lines(bg.en.spid, col = 4)
```



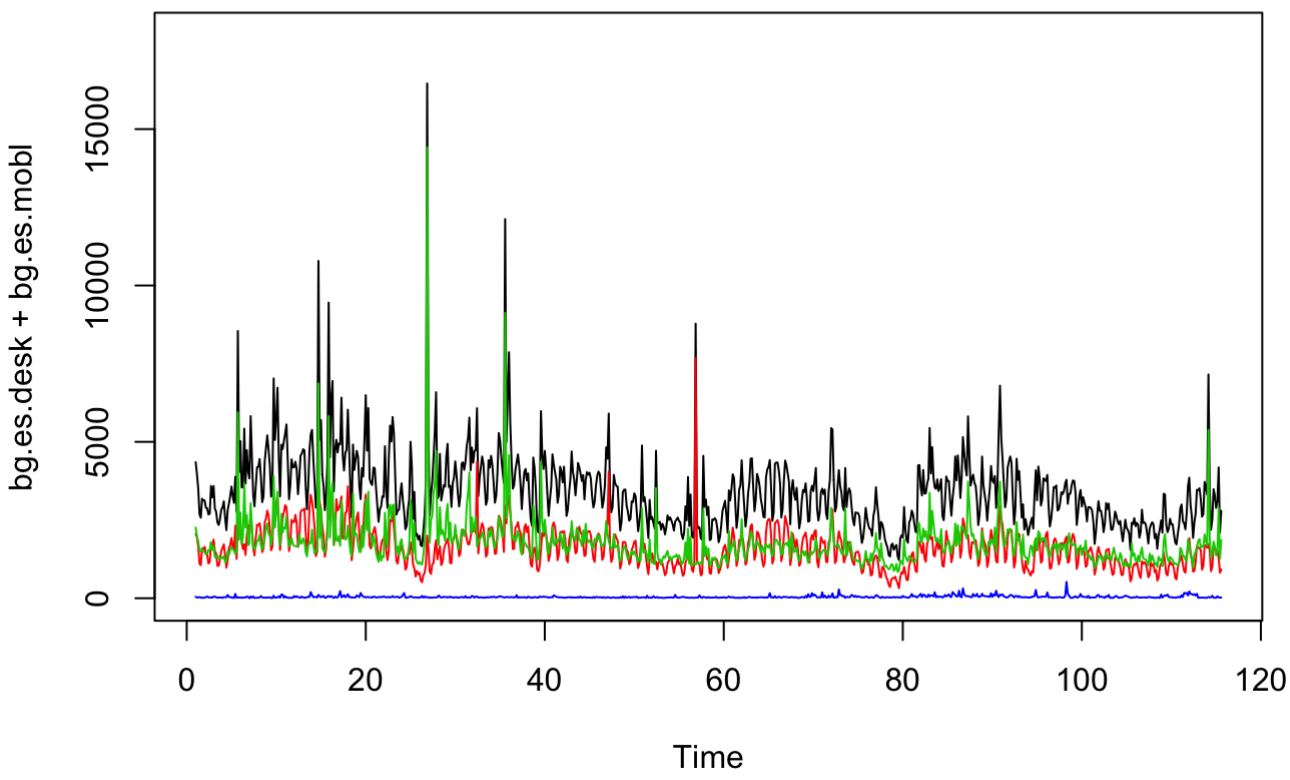

```
plot(bg.de.all, ylim = c(0,8000), main = "Bill Gates - German Wikipedia")
lines(bg.de.desk, col = 2)
lines(bg.de.mobl, col = 3)
lines(bg.de.spid, col = 4)
```



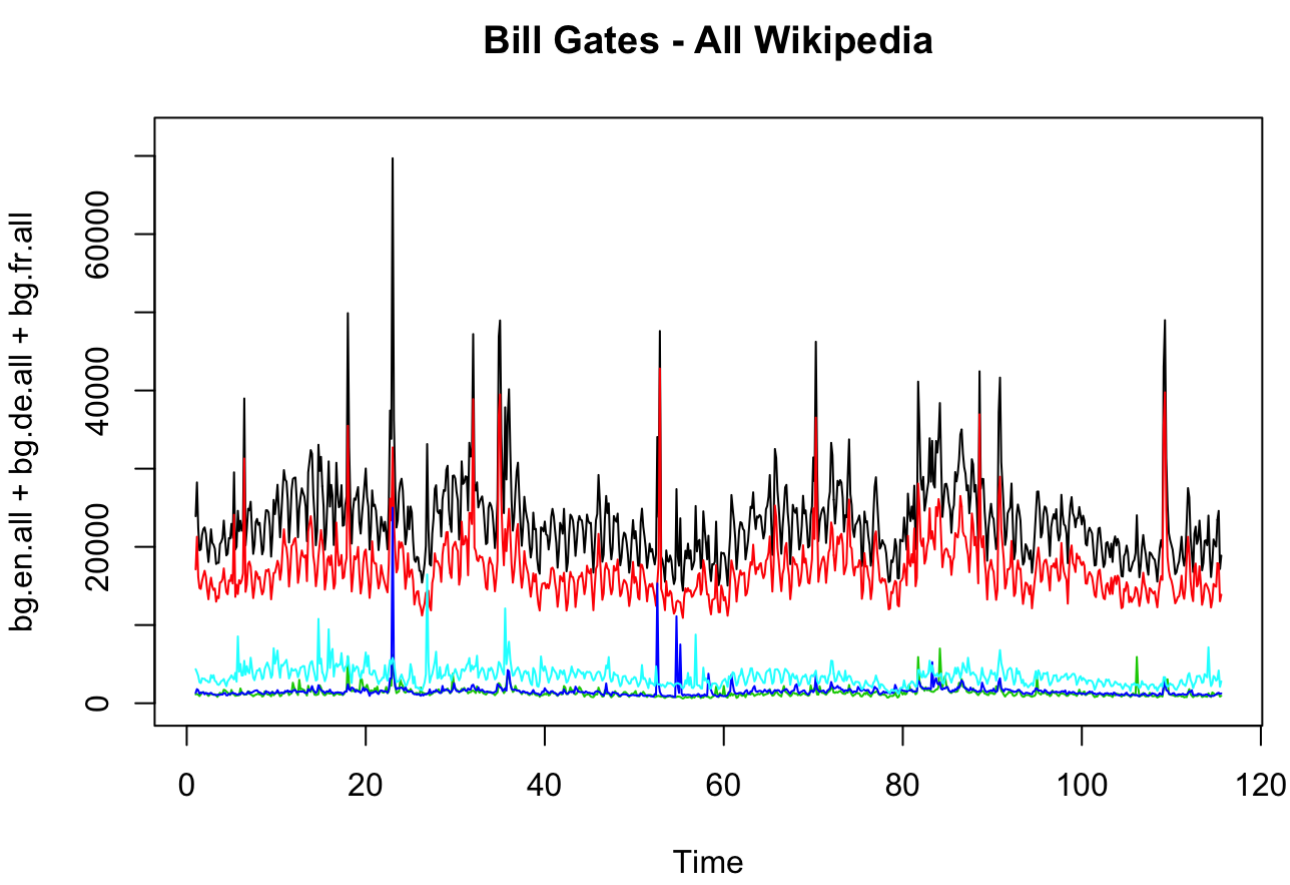

```
plot(bg.fr.all, ylim = c(0,27000), main = "Bill Gates - French Wikipedia")
lines(bg.fr.desk, col = 2)
lines(bg.fr.mobl, col = 3)
lines(bg.fr.spid, col = 4)
```




```
plot(bg.es.all, ylim = c(0,18000), main = "Bill Gates - Spanish Wikipedia")
lines(bg.es.desk, col = 2)
lines(bg.es.mobl, col = 3)
lines(bg.es.spid, col = 4)
```

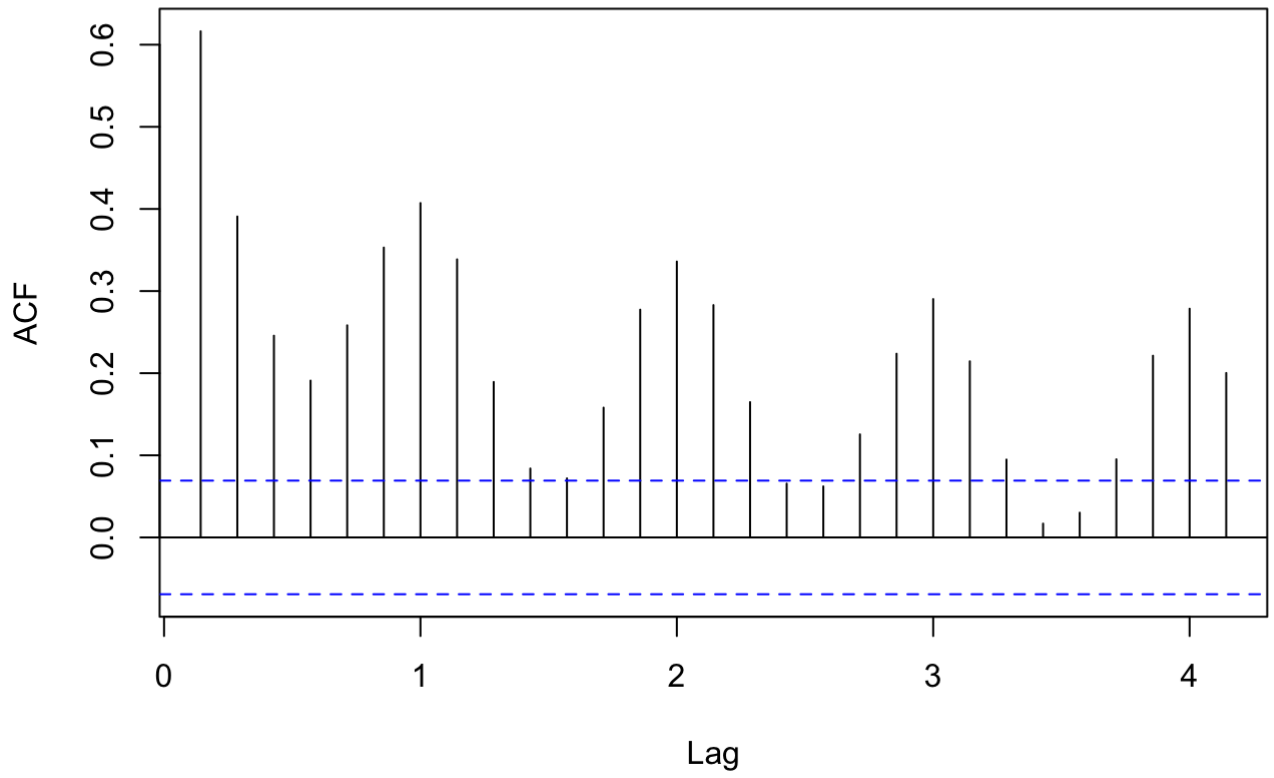



```
plot(bg.all, ylim = c(0,72000), main = "Bill Gates - All Wikipedia")
lines(bg.en.all, col = 2)
lines(bg.de.all, col = 3)
lines(bg.fr.all, col = 4)
lines(bg.es.all, col = 5)
```


Bill Gates - All Wikipedia
bg.en.all + bg.de.all + bg.fr.all
0
20000
40000
60000
0
20
40
60
80
100
120
Time

```
#to help with plotting
plot_rownr <- function(rownr){
  art <- tpages %>% filter(rowname == rownr) %>% .$article
  loc <- tpages %>% filter(rowname == rownr) %>% .$locale
  acc <- tpages %>% filter(rowname == rownr) %>% .$access
  extract_ts(rownr) %>%
    ggplot(aes(dates, views)) +
    geom_line() +
    geom_smooth(method = "loess", color = "blue", span = 1/5) +
    labs(title = str_c(art, " - ", loc, " - ", acc))
}

plot_rownr_log <- function(rownr){
  art <- tpages %>% filter(rowname == rownr) %>% .$article
  loc <- tpages %>% filter(rowname == rownr) %>% .$locale
  acc <- tpages %>% filter(rowname == rownr) %>% .$access
  extract_ts_nrm(rownr) %>%
    ggplot(aes(dates, views)) +
    geom_line() +
    geom_smooth(method = "loess", color = "blue", span = 1/5) +
    labs(title = str_c(art, " - ", loc, " - ", acc)) +
    scale_y_log10() + labs(y = "log views")
}

plot_rownr_zoom <- function(rownr, start, end){
  art <- tpages %>% filter(rowname == rownr) %>% .$article
  loc <- tpages %>% filter(rowname == rownr) %>% .$locale
  acc <- tpages %>% filter(rowname == rownr) %>% .$access
  extract_ts(rownr) %>%
    filter(dates > ymd(start) & dates <= ymd(end)) %>%
    ggplot(aes(dates, views)) +
    geom_line() +
    #geom_smooth(method = "loess", color = "blue", span = 1/5) +
    #coord_cartesian(xlim = ymd(c(start,end))) +
    labs(title = str_c(art, " - ", loc, " - ", acc))
}
```

```
acf(bg.all)
```




```
kpss.test(bg.all)
```

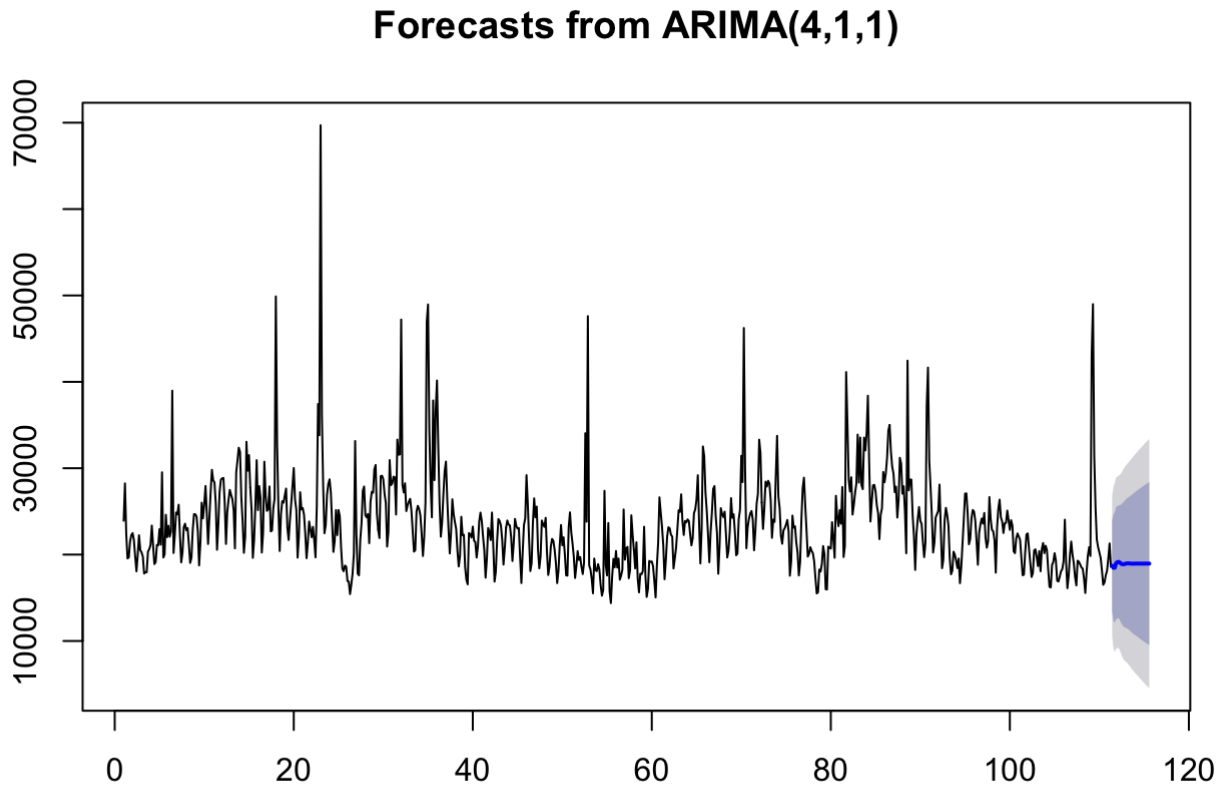
```
##
##  KPSS Test for Level Stationarity
##
## data:  bg.all
## KPSS Level = 0.71556, Truncation lag parameter = 6, p-value =
## 0.01213
```

Nonseasonal Arima

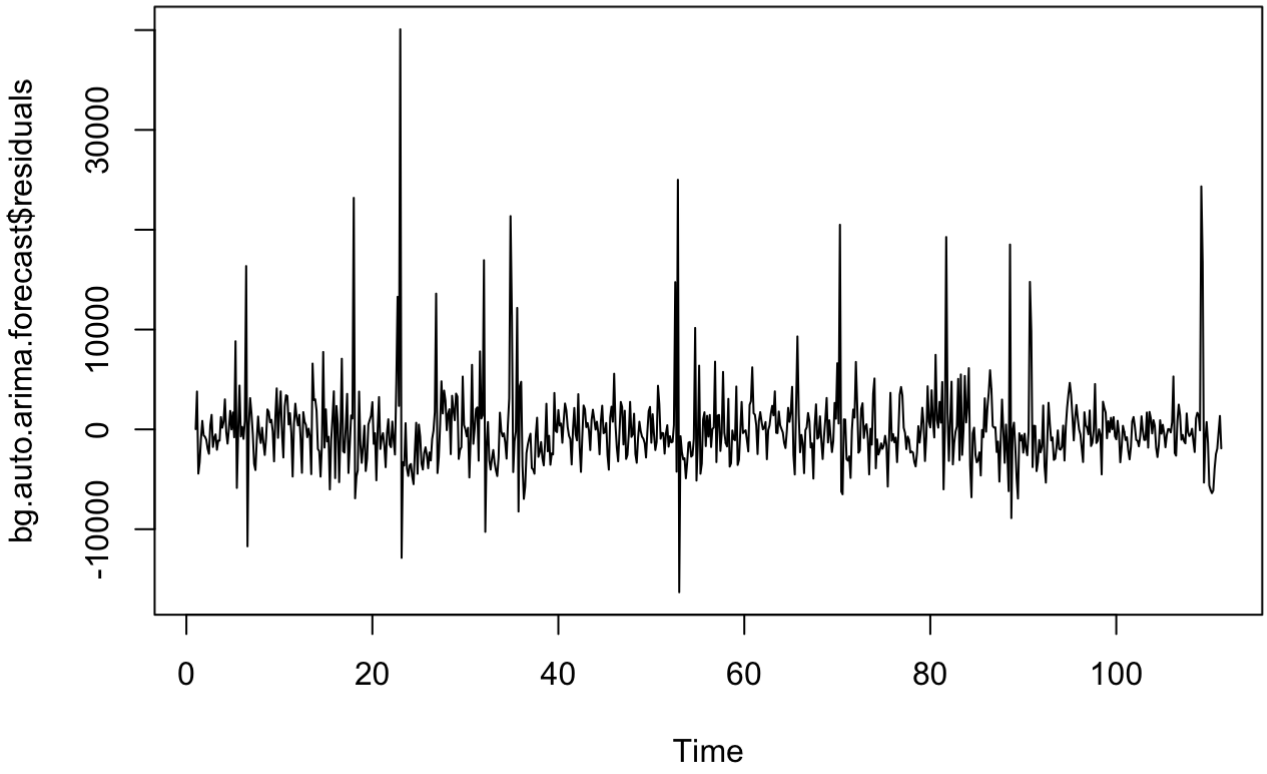
```
train <- window(bg.all, end = c(111,3))
test <- window(bg.all, start = c(111,4))

bg.auto.arima <- auto.arima(train, seasonal = F, stepwise = F)
bg.auto.arima #ARIMA(4,1,1)  #AIC=15,029.9
```

```
## Series: train
## ARIMA(4,1,1)
##
## Coefficients:
##          ar1      ar2      ar3      ar4      ma1
##       0.2824  -0.0852  -0.1457  -0.1763  -0.7081
## s.e.  0.0856   0.0441   0.0421   0.0531   0.0857
##
## sigma^2 estimated as 16933916:  log likelihood=-7518.3
## AIC=15048.59   AICc=15048.7   BIC=15076.48
```

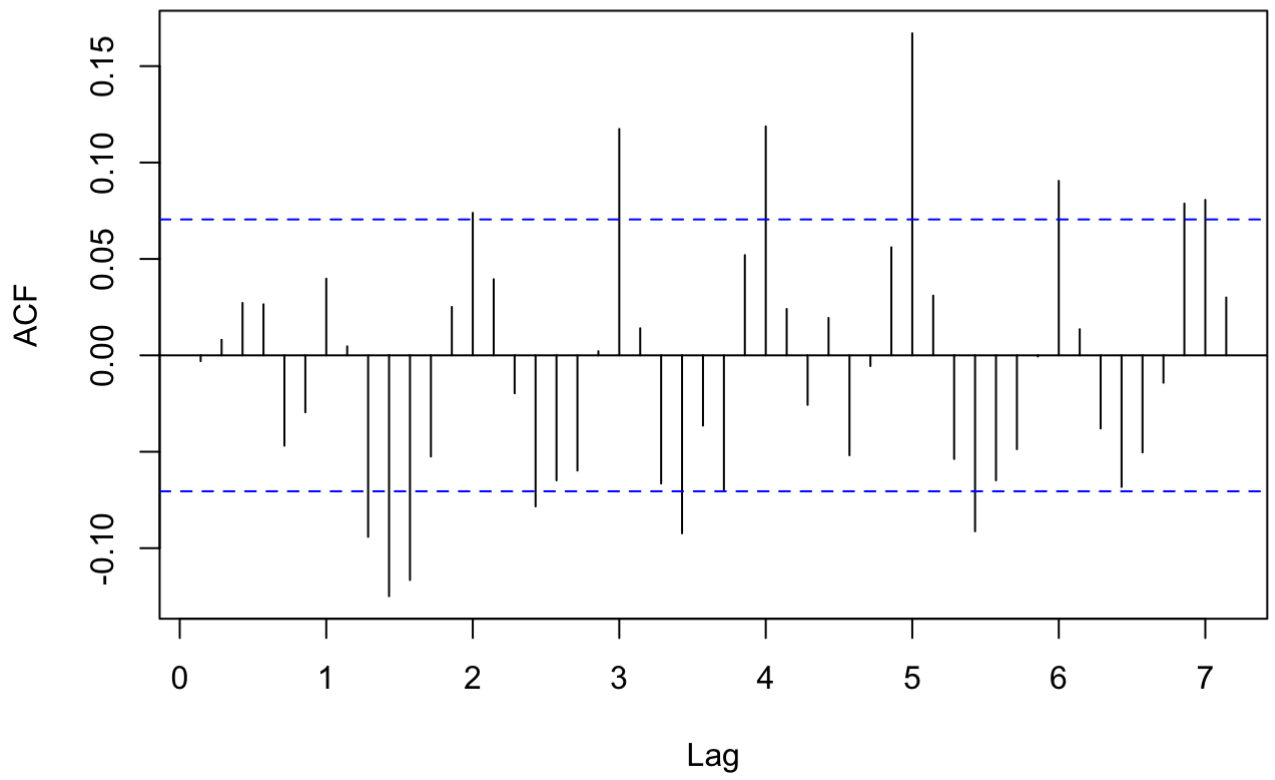
```
bg.auto.arima.forecast <- forecast(bg.auto.arima, h=30)
plot(bg.auto.arima.forecast)
```




```
plot(bg.auto.arima.forecast$residuals)
```




```
acf(bg.auto.arima.forecast$residuals, 50)
```




```
accuracy(bg.auto.arima.forecast, test)
```

```
##                      ME     RMSE      MAE       MPE      MAPE      MASE
## Training set  -22.45622 4099.082 2476.969 -1.931765  9.959495 0.7392464
## Test set     1734.44132 3250.966 2530.823  6.809611 11.496360 0.7553189
##                      ACF1 Theil's U
## Training set -0.003013109        NA
## Test set      0.381103704  1.038294
```

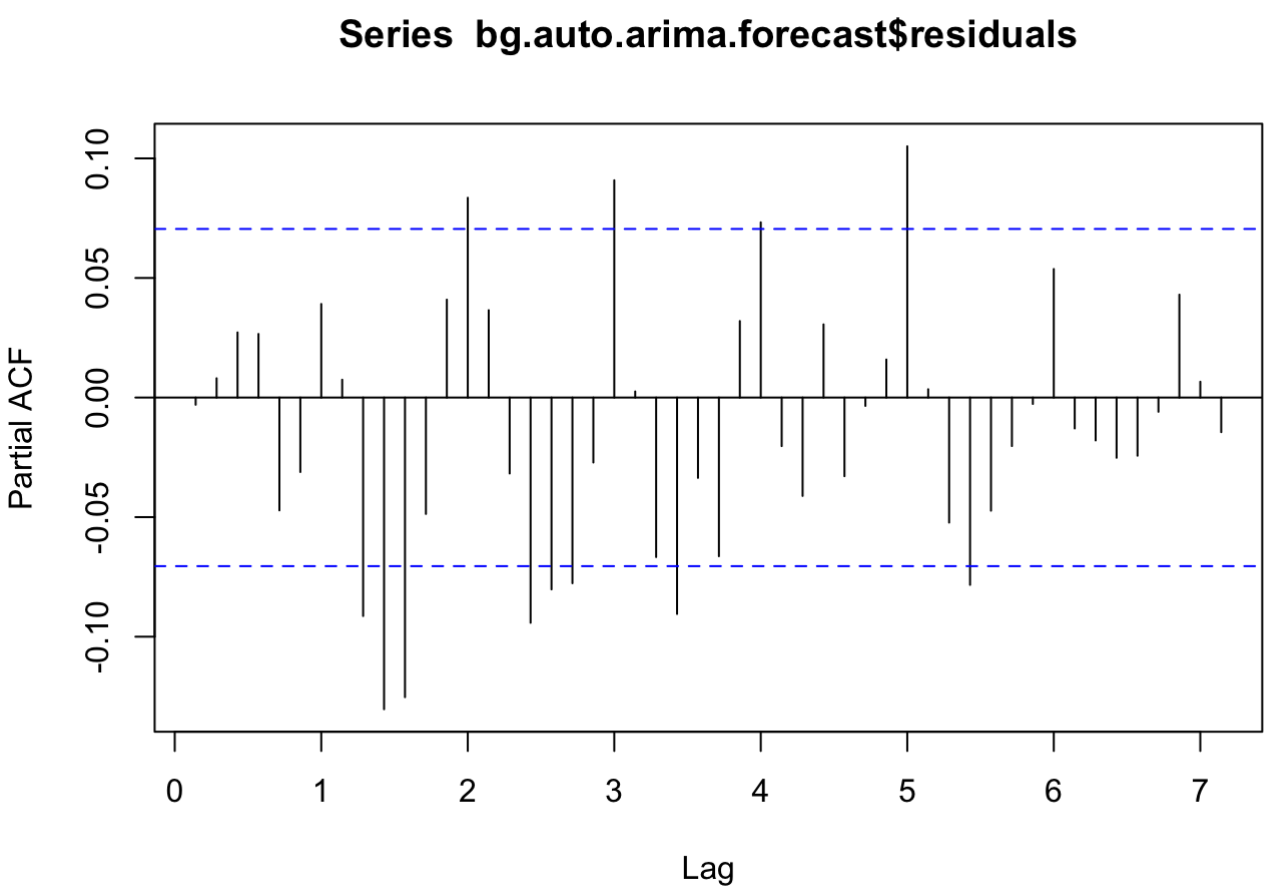
```
mean(bg.auto.arima.forecast$residuals)
```

```
## [1] -22.45622
```

```
Box.test(bg.auto.arima.forecast$residuals, type = "Ljung-Box", lag = 14)
```

```
##
##  Box-Ljung test
##
## data:  bg.auto.arima.forecast$residuals
## X-squared = 41.682, df = 14, p-value = 0.0001388
```

```
pacf(bg.auto.arima.forecast$residuals, 50)
```




```
kpss.test(bg.auto.arima.forecast$residuals)
```

```
## Warning in kpss.test(bg.auto.arima.forecast$residuals): p-value greater
## than printed p-value
```

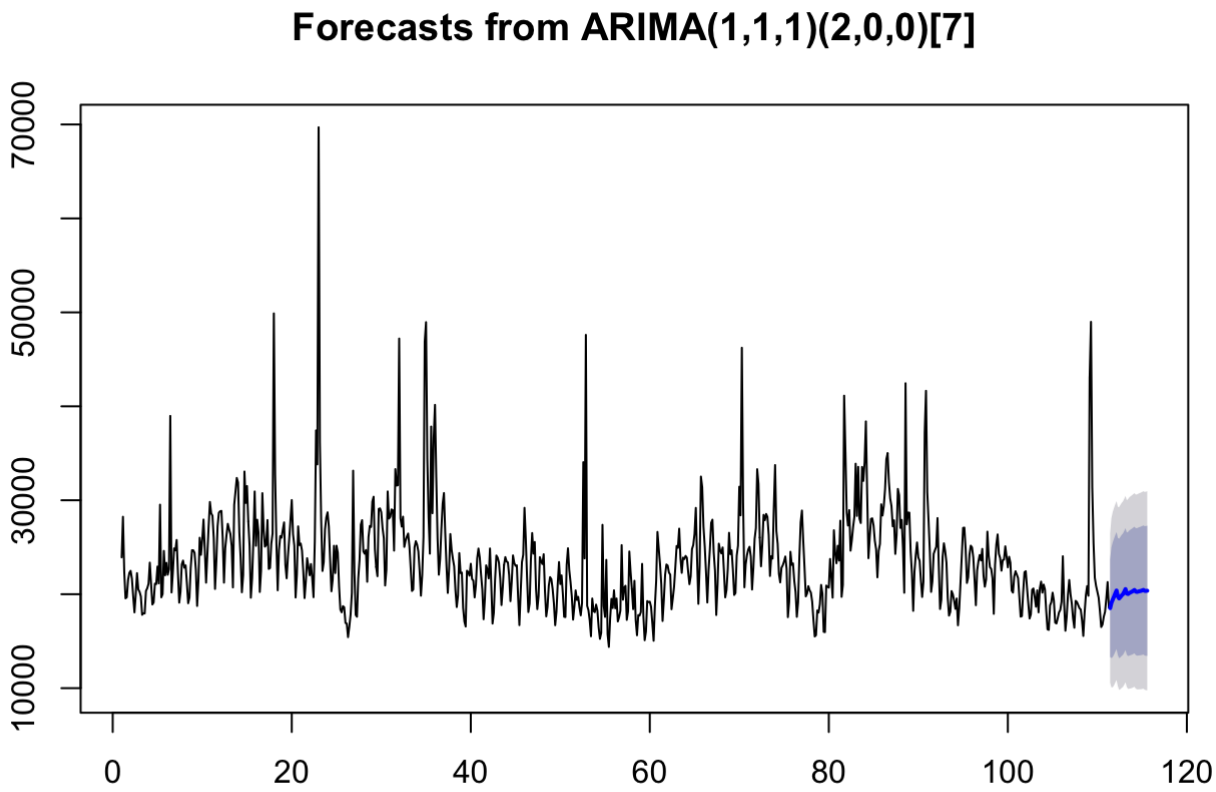
```
##
##  KPSS Test for Level Stationarity
##
## data:  bg.auto.arima.forecast$residuals
## KPSS Level = 0.019797, Truncation lag parameter = 6, p-value = 0.1
```

Seasonal Arima

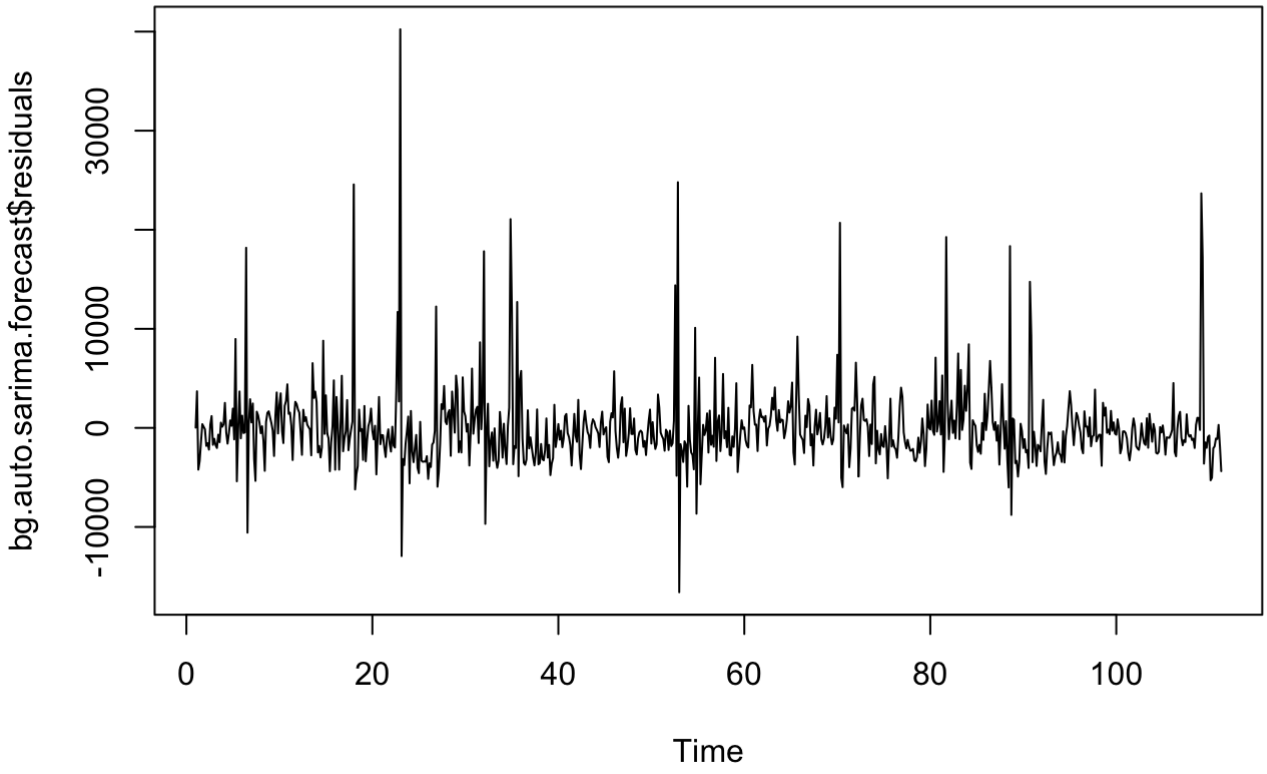
```
train <- window(bg.all, end = c(111,3))
test <- window(bg.all, start = c(111,4))

bg.auto.sarima <- auto.arima(train, stepwise = F)
bg.auto.sarima #ARIMA(1,1,1)(2,0,0)[7]  #AIC=15,003.24
```

```
## Series: train
## ARIMA(1,1,1)(2,0,0)[7]
##
## Coefficients:
##          ar1      ma1    sar1   sar2
##       0.5089  -0.9712  0.1769  0.134
## s.e.  0.0354   0.0125  0.0367  0.036
##
## sigma^2 estimated as 16381873:  log likelihood=-7506.44
## AIC=15022.88   AICc=15022.96   BIC=15046.13
```

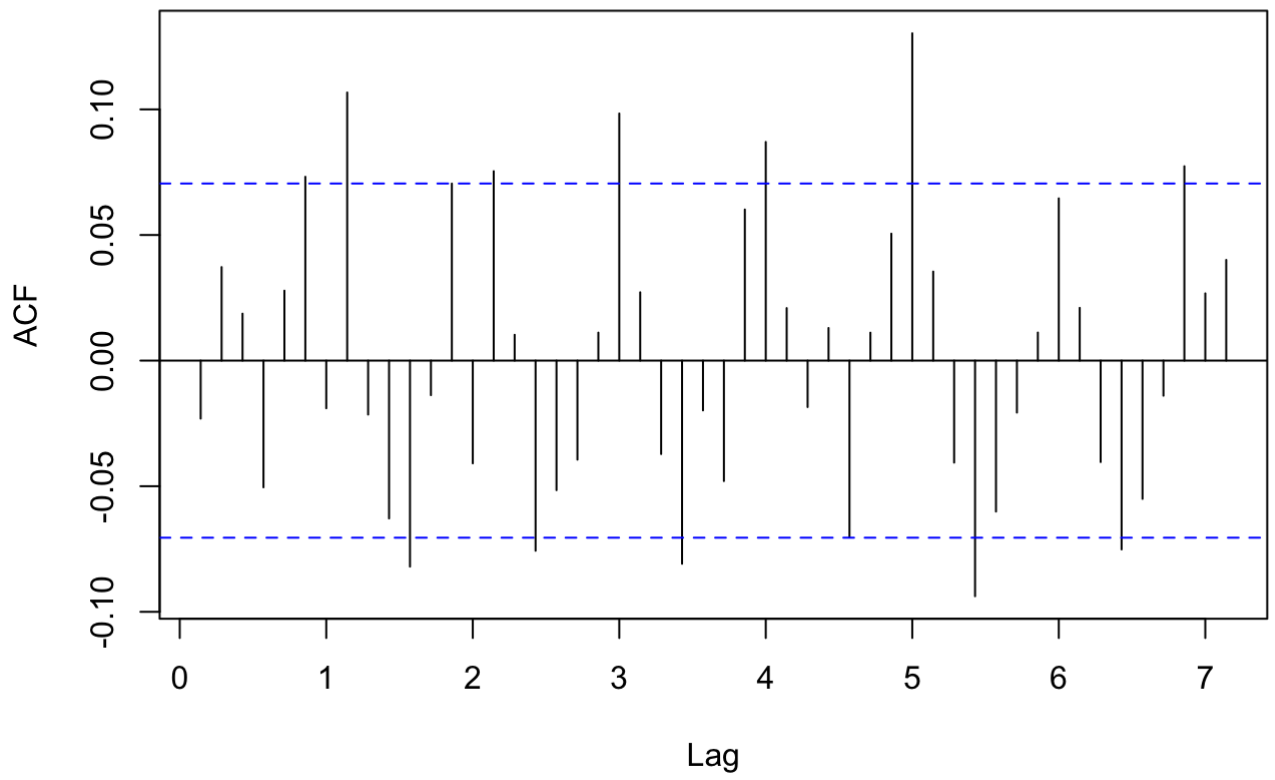
```
bg.auto.sarima.forecast <- forecast(bg.auto.sarima, h=30)
plot(bg.auto.sarima.forecast)
```




```
plot(bg.auto.sarima.forecast$residuals)
```



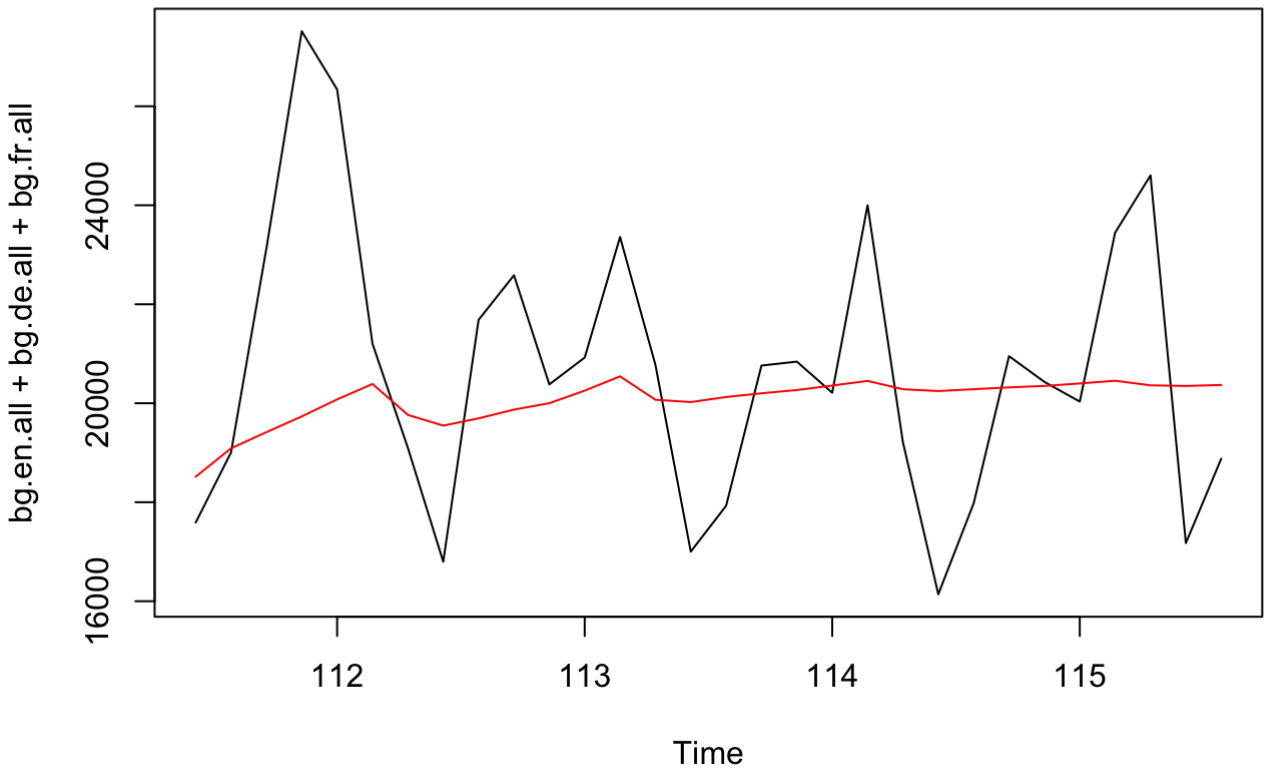

```
acf(bg.auto.sarima.forecast$residuals, 50)
```




```
accuracy(bg.auto.sarima.forecast, test)
```

```
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -31.12515 4034.341 2401.153 -2.148731 9.653707 0.7166192
## Test set     605.44325 2791.224 2092.539  1.297238 9.855258 0.6245138
##                    ACF1 Theil's U
## Training set -0.02309289        NA
## Test set      0.40633026 0.8993862
```

```
plot(test) #the forecast is less volatile/tame
lines(bg.auto.sarima.forecast$mean,col=2)
```




```
mean(bg.auto.sarima.forecast$residuals)
```

```
## [1] -31.12515
```

```
Box.test(bg.auto.sarima.forecast$residuals, type = "Ljung-Box", lag = 14)
```

```
##
##  Box-Ljung test
##
## data:  bg.auto.sarima.forecast$residuals
## X-squared = 31.867, df = 14, p-value = 0.004184
```

```
kpss.test(bg.auto.sarima.forecast$residuals)
```

```
## Warning in kpss.test(bg.auto.sarima.forecast$residuals): p-value greater
## than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  bg.auto.sarima.forecast$residuals
## KPSS Level = 0.098975, Truncation lag parameter = 6, p-value = 0.1
```

Hierarchical Model

```
#nodes <- list(4, c(3, 3, 3, 3))
#group into language, then into total
abc <- cbind(bg.en.desk, bg.en.mobl, bg.en.spid,
             bg.de.desk, bg.de.mobl, bg.de.spid,
             bg.fr.desk, bg.fr.mobl, bg.fr.spid,
             bg.es.desk, bg.es.mobl, bg.es.spid)

train <- hts(window(abc, end = c(111,3)), characters = c(6,4))
test <- hts(window(abc, start = c(111,4)), characters = c(6,4))

summary(train)
```

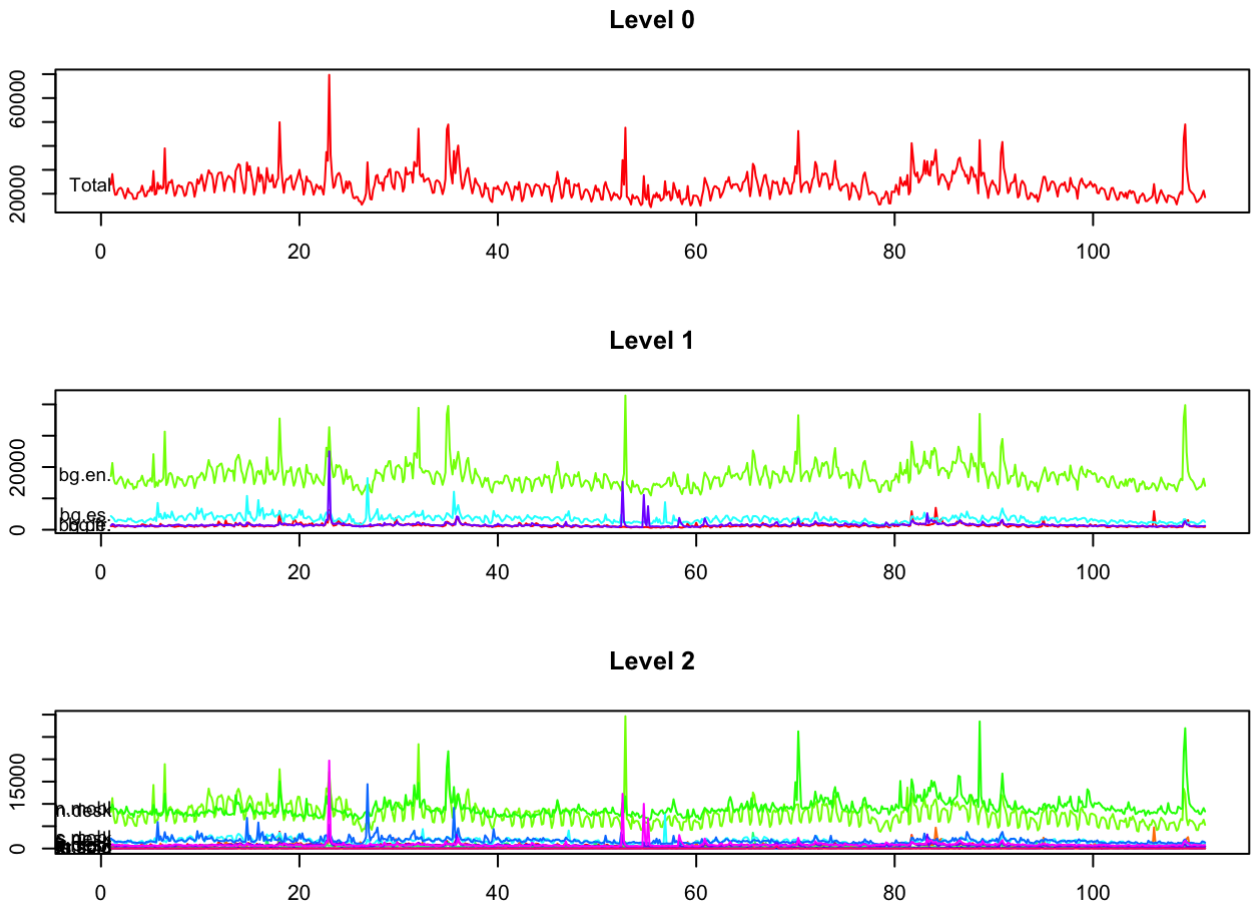
```
## Hierarchical Time Series
## 3 Levels
## Number of nodes at each level: 1 4 12
## Total number of series: 17
## Number of observations per series: 773
## Top level series:
## Time Series:
## Start = c(1, 1)
## End = c(111, 3)
## Frequency = 7
##   [1] 23923 28241 22400 19559 19674 21514 22271 22490 21671 19774 18068
##  [12] 19978 22242 20510 20256 19760 17807 17966 17957 20347 20632 21117
##  [23] 23361 21099 18907 19121 21124 21074 22942 21151 29529 19669 20009
##  [34] 24606 22017 23335 22071 22657 38977 20189 22286 24875 24662 25773
##  [45] 22238 19102 20257 23179 23580 22817 23066 21166 19041 19643 22844
##  [56] 24684 24674 24404 22618 18752 21439 26009 24191 25825 27945 24751
##  [67] 21229 24196 27224 29818 28608 28354 25965 20562 23411 27239 28670
##  [78] 28829 28869 25997 21231 24904 26306 27489 26913 26425 25144 20725
##  [89] 29527 30923 32375 31900 26589 23894 20206 22033 33058 29713 31511
## [100] 27753 25195 19627 21687 25732 30932 25153 27925 26577 20240 22483
## [111] 30754 26855 25062 25235 27891 22690 22789 25363 26469 49889 32688
## [122] 25391 20418 24710 26186 26010 26865 27643 23127 21696 23994 25876
## [133] 27932 30021 26375 25175 19652 22092 27220 25223 25394 24649 22457
## [144] 19588 21366 23831 22936 22018 23208 21677 19683 25499 37437 33821
## [155] 69700 36128 27844 22474 23416 26913 28170 28714 27185 23624 20313
## [166] 21554 25152 22254 25160 24420 20829 18374 18073 18727 18612 16938
## [177] 16877 15431 16532 17816 20133 33147 22239 17795 17601 21708 23640
## [188] 27404 27882 24562 24309 24702 21324 26080 27266 27155 29859 30373
## [199] 24804 22900 21917 29083 29132 28730 27031 25963 20930 22598 30937
## [210] 28046 28235 29024 28946 24640 33315 31562 31601 47211 28077 27196
## [221] 28252 25038 25705 26128 26457 25098 22781 20345 20505 24991 25647
## [232] 25283 24538 22430 19842 22010 26057 46925 48957 34381 28165 24316
## [243] 37844 28621 36690 40155 32403 25974 22076 23734 26994 29640 30767
## [254] 26524 22998 20138 22622 26402 24722 23763 22971 20416 18636 19856
## [265] 24376 22245 22303 18542 17091 16562 22498 22352 21972 23239 21611
## [276] 21446 19662 20973 23806 24860 23914 22109 20507 17336 19492 23070
## [287] 22613 21685 24866 21037 16876 18010 21596 24119 23890 23364 21692
## [298] 18846 20708 23103 23949 23481 23384 22005 19261 21329 24128 23321
## [309] 23018 23074 19677 16710 20240 23417 24201 29193 25917 21763 18102
## [320] 18995 23697 26510 24319 25556 20570 18412 19782 23945 23493 23210
## [331] 24254 20315 17643 18635 20906 21819 21603 20551 19009 16700 18127
## [342] 21132 23422 21082 22001 20010 17599 17567 23157 24888 22592 21295
## [353] 19751 17298 18665 20403 19383 19717 18587 17754 19058 34051 23814
## [364] 47604 18764 18305 17208 15525 19540 18203 18089 18890 18288 16467
## [375] 15248 15886 27382 18749 17587 23652 15731 14392 17445 19506 18527
## [386] 20413 18502 19540 17104 17560 18245 25222 19416 20706 20896 17269
## [397] 18183 24544 22348 20558 18412 21426 17213 15631 17784 17726 18089
## [408] 23226 17673 15105 16023 19273 19146 19207 18661 16909 15046 18753
## [419] 21399 26628 24881 23227 21055 17151 19369 22072 23163 23047 22301
## [430] 22085 18401 19550 20792 22914 25136 24935 26976 23833 22274 23894
## [441] 23735 24077 23887 22398 21067 21897 24802 25237 26422 29186 23712
## [452] 18977 22991 32521 31395 27188 25066 22306 19121 22534 24808 27603
## [463] 27932 24448 22516 17789 20961 25378 24678 24344 24973 22421 19666
```

```
## [474] 20450 22680 26534 24263 24671 22614 19920 20123 24111 25166 31439
## [485] 28400 46233 29684 20736 23168 24152 24755 25047 24021 20499 22792
## [496] 25892 27039 33306 31741 26030 22973 25265 28444 27878 28542 28230
## [507] 26296 21940 24052 23788 29183 33746 26730 25378 22166 21259 22808
## [518] 23074 23490 24011 22703 17560 19196 24515 23122 23282 21424 20241
## [529] 17591 19135 24036 27669 28905 25724 22857 19770 20072 20857 20425
## [540] 20165 19204 17248 15501 15640 18266 18043 18915 20988 20069 15995
## [551] 15962 20919 20787 20698 23751 21784 19633 26807 24418 23295 25152
## [562] 23578 27825 19721 20987 41132 35141 28659 27310 28950 24634 25731
## [573] 27155 28247 33890 28332 33551 28036 27578 33546 32060 34034 38394
## [584] 30649 23834 26020 28018 28074 27315 25672 24793 21798 24614 25347
## [595] 29567 28375 29391 31196 34369 34998 31952 30194 29386 27229 27864
## [606] 24290 26284 31206 30548 27057 27908 24932 20156 42453 27441 28693
## [617] 28678 24547 22203 18229 22428 24178 25506 23906 23561 21508 19710
## [628] 21330 37227 41649 30832 27833 24546 20247 21364 24297 24690 25043
## [639] 28102 22975 18426 19431 23976 25416 24583 23361 19589 17723 18123
## [650] 20332 19983 19310 18444 19540 16692 18508 21270 24106 27058 27087
## [661] 24812 21199 21791 24216 25168 24977 23887 21477 18830 22511 23514
## [672] 24175 23591 24821 21934 20802 21559 26657 24122 22879 22686 21695
## [683] 17912 23139 25301 26361 24296 23837 22505 23628 23684 25096 24003
## [694] 22862 23981 23351 20384 20170 21889 22420 22095 21713 19700 17611
## [705] 17672 20279 22337 22449 20670 19006 17417 17862 20476 20611 19701
## [716] 18671 20410 18073 20620 21338 20167 21021 20518 18248 16266 16201
## [727] 18808 19161 20160 19317 16977 16884 17437 18066 18317 19210 24039
## [738] 19061 16112 17559 19972 21512 19998 19011 17410 16433 19294 19172
## [749] 18850 18511 18366 17203 15558 17976 19880 20861 19839 43171 48991
## [760] 31356 25441 21751 20878 20261 19631 18203 16506 16762 17600 18126
## [771] 19552 21291 18583
##
## Labels:
## [1] "Level 0" "Level 1" "Level 2"
```

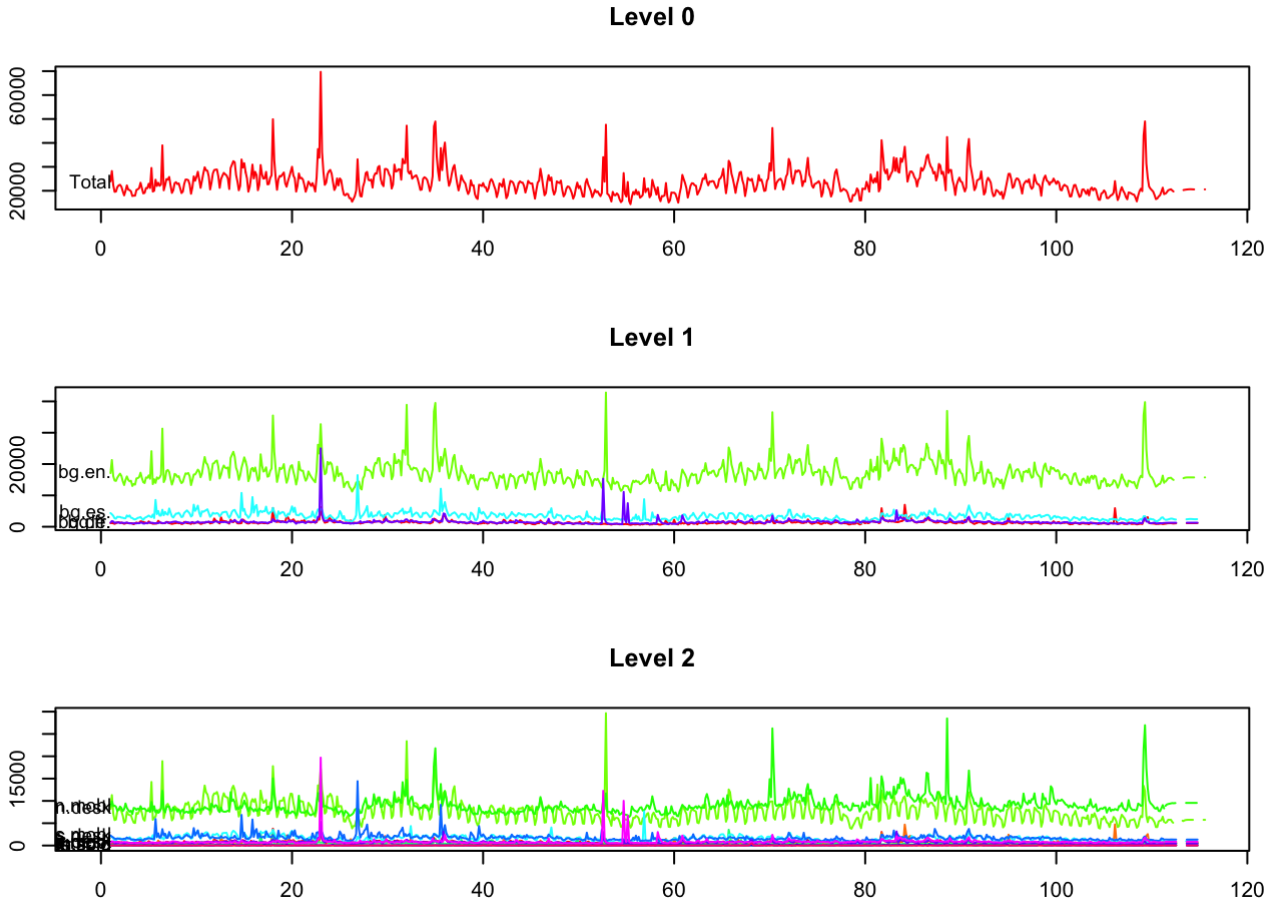
```
smatrix(train)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
##  [1,]    1    1    1    1    1    1    1    1    1     1     1     1
##  [2,]    1    1    1    0    0    0    0    0    0     0     0     0
##  [3,]    0    0    0    1    1    1    0    0    0     0     0     0
##  [4,]    0    0    0    0    0    0    1    1    1     0     0     0
##  [5,]    0    0    0    0    0    0    0    0    0     1     1     1
##  [6,]    1    0    0    0    0    0    0    0    0     0     0     0
##  [7,]    0    1    0    0    0    0    0    0    0     0     0     0
##  [8,]    0    0    1    0    0    0    0    0    0     0     0     0
##  [9,]    0    0    0    1    0    0    0    0    0     0     0     0
## [10,]    0    0    0    0    1    0    0    0    0     0     0     0
## [11,]    0    0    0    0    0    1    0    0    0     0     0     0
## [12,]    0    0    0    0    0    0    1    0    0     0     0     0
## [13,]    0    0    0    0    0    0    0    1    0     0     0     0
## [14,]    0    0    0    0    0    0    0    0    1     0     0     0
## [15,]    0    0    0    0    0    0    0    0    0     1     0     0
## [16,]    0    0    0    0    0    0    0    0    0     0     1     0
## [17,]    0    0    0    0    0    0    0    0    0     0     0     1
```

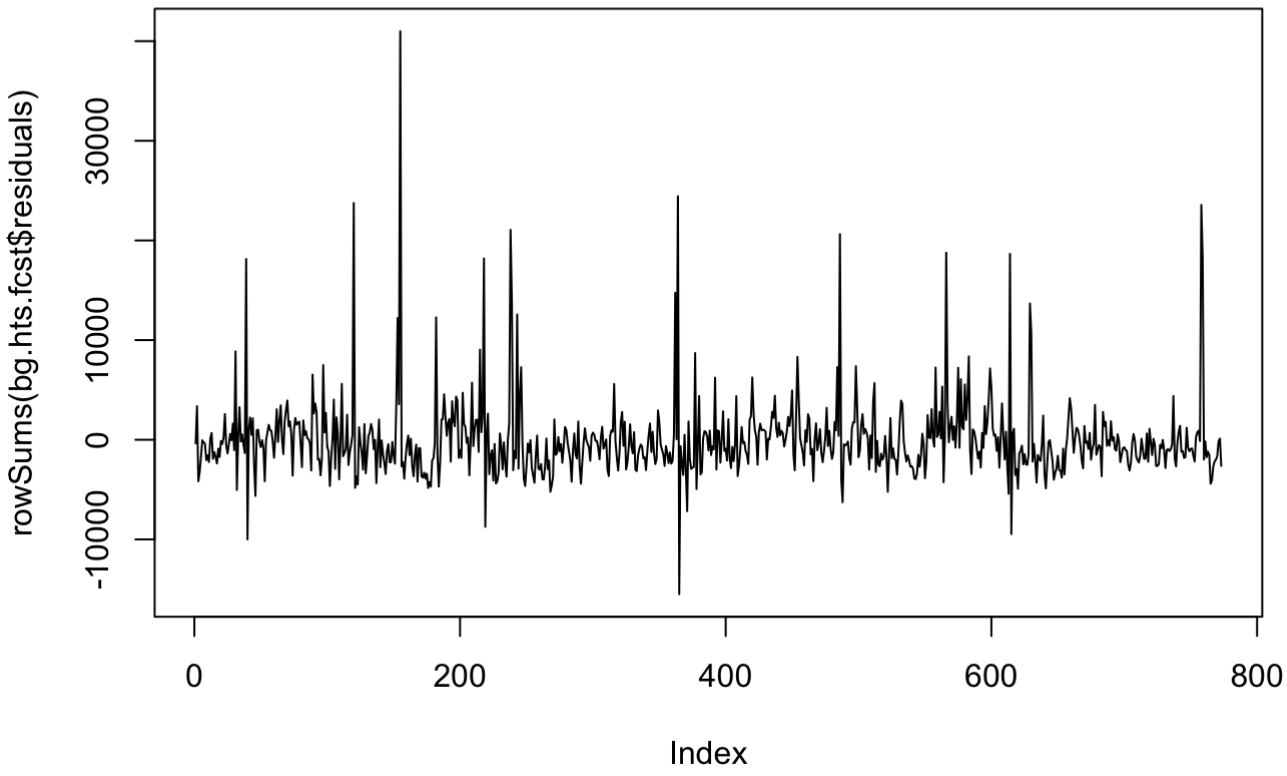
```
plot(train) #aggregates some of them in the middle, then all at the to
```



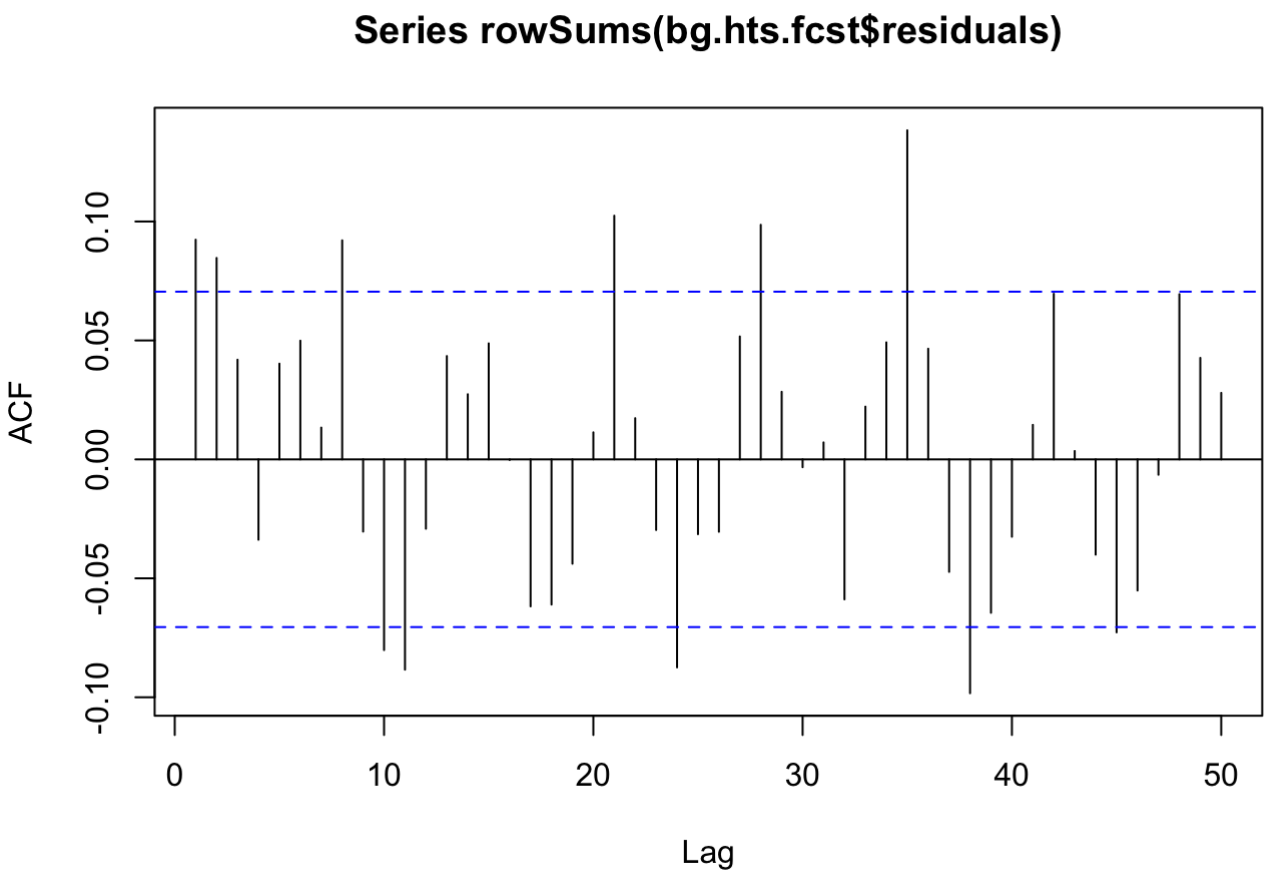

```
bg.hts.fcst <- forecast(train, method="bu", fmethod = "arima", h=30, keep.resid = T, wei
ghts = 'ols')
# bg.hts.fcst <- forecast(train, method="mo", fmethod = "arima", weights = 'ols', keep.f
itted = TRUE, keep.resid = TRUE, h=30, level = 1) # RMSE 2848.9230544
    #Forecasts are distributed in the hierarchy using bottom-up, top-down, middle-out, a
nd optimal combination methods.
    #"comb", "bu", "mo", "tdgsa", "tdgsf", "tdfp"
plot(bg.hts.fcst)
```




```
plot(rowSums(bg.hts.fcst$residuals), type = 'l')
```




```
acf(rowSums(bg.hts.fcst$residuals),50)
```




```
accuracy.gts(bg.hts.fcst, test, level = 0)
```

```
##             Total
## ME    311.8174560
## RMSE 2686.0089123
## MAE  2041.3629309
## MAPE    9.7727548
## MPE    -0.1215421
## MASE    0.6092405
```

ARIMAX Model

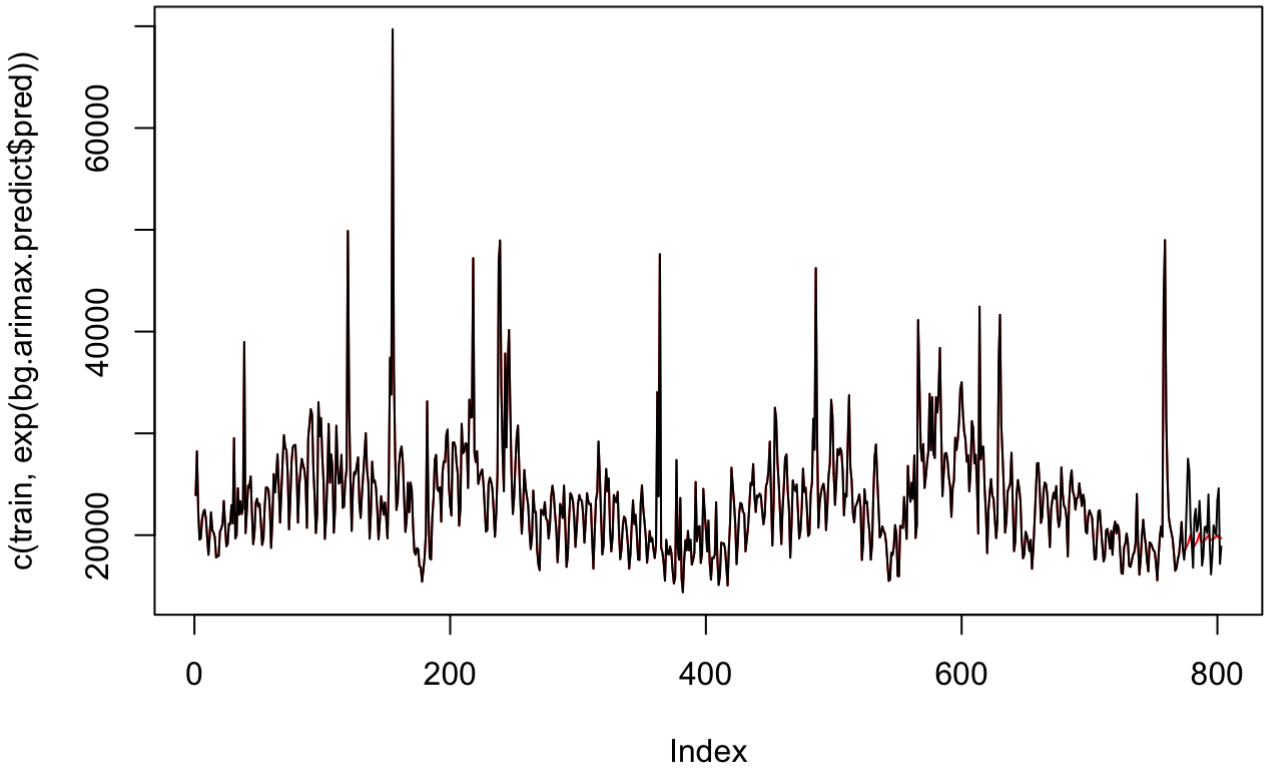
```
train <- window(bg.all, end = c(111,3))
test <- window(bg.all, start = c(111,4))

which(bg.auto.sarima.forecast$residuals >= 25000)
```

```
## [1] 155
```

```
pulses <- c(39, 120, 153, 155, 182, 218, 238, 243, 362, 364, 377, 486, 566, 614, 629, 75
8)
pulses <- 1*(seq(train) %in% pulses)

bg.pulse <- arimax(log(train),
                   order=c(1,1,1),
                   seasonal=list(order=c(2,0,0), period=7),
                   transfer=list(c(1,0)),
                   xtransf=data.frame(pulses),
                   method='ML')

bg.pulse
```

```
##
## Call:
## arimax(x = log(train), order = c(1, 1, 1), seasonal = list(order = c(2, 0, 0),
##     period = 7), method = "ML", xtransf = data.frame(pulses), transfer = list(c(1,
##     0)))
##
## Coefficients:
##          ar1      ma1    sar1    sar2  pulses-AR1  pulses-MA0
##       0.6494  -0.9868  0.3225  0.2673      0.4094      0.5755
## s.e.  0.0356   0.0151  0.0364  0.0370      0.0460      0.0238
##
## sigma^2 estimated as 0.01051:  log likelihood = 661.44,  aic = -1310.89
```

```
tf<-stats::filter(1*(seq(1:(length(train)+30))==155),filter=0.4094, method='recursive',s
ide=1) * (0.5755)
forecast.arima<-Arima(log(train), order=c(1,1,1),
                      seasonal = list(order=c(2,0,0), period=7),
                      xreg=tf[1:(length(tf)-30)])

bg.arimax.predict <- predict(forecast.arima, n.ahead = 30, newxreg=tf[774:length(tf)])

plot(c(train,exp(bg.arimax.predict$pred)),type='l', col =2)
lines(ts(bg.all), col = 1)
```




```
#MAE
mean(abs(exp(bg.arimax.predict$pred)-test))
```

```
## [1] 2211.819
```

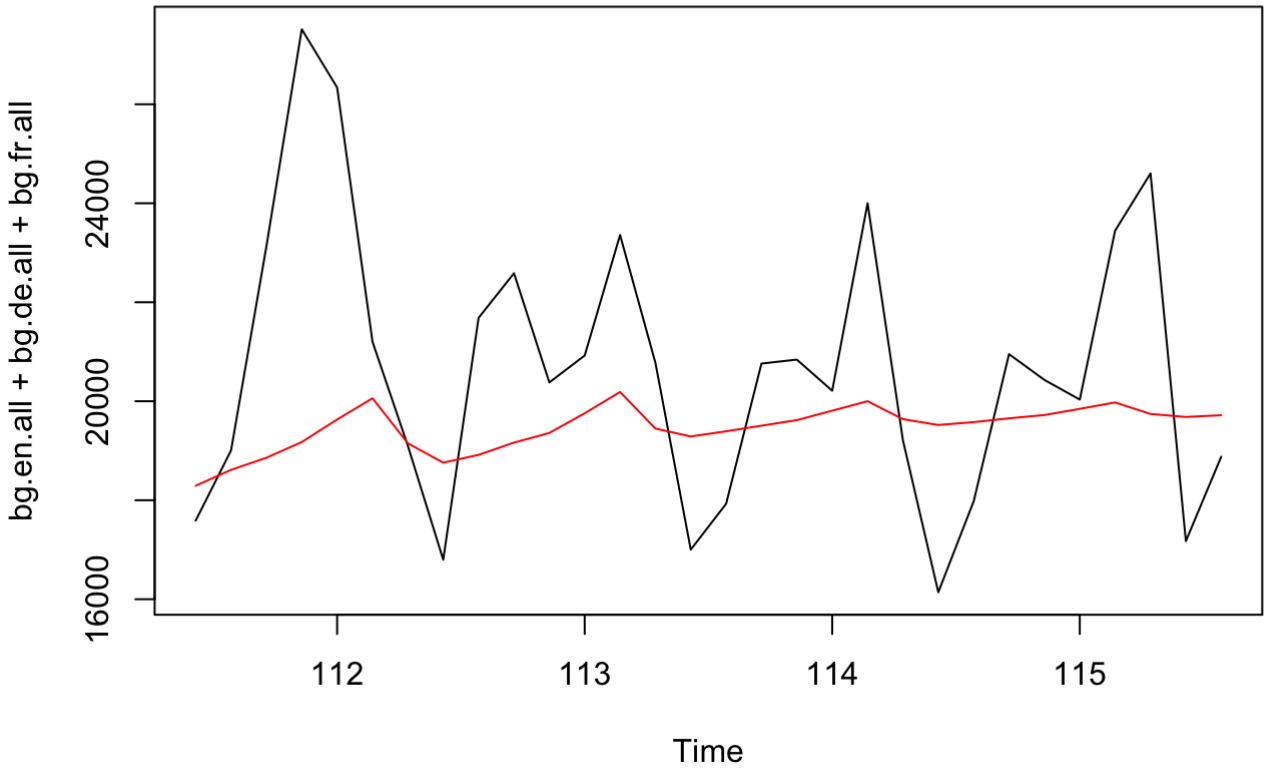
```
#MPE
sum((test-exp(bg.arimax.predict$pred))/test)*100/(length(test)-1)
```

```
## [1] 4.386575
```

```
#RMSE
sqrt(mean((exp(bg.arimax.predict$pred)-test)^2))
```

```
## [1] 2931.564
```

```
plot(test) #like the seasonal arima, it doesn't have as extreme swings in the data
lines(exp(bg.arimax.predict$pred),col=2)
```


bg.en.all + bg.de.all + bg.fr.all
16000
20000
24000
112
113
114
115
Time