Massachusetts_Case

Karl Grindal

6/8/2020

Case 1: Massachusetts Data Security Law

```
library(plyr)
library(here)
```

```
## here() starts at C:/Users/karl_000/Documents/SpiderOak Hive/Dissertation/Dissertation_Code
```

```
##
## Attaching package: 'here'

## The following object is masked from 'package:plyr':
##
##     here
```

```
library(tidyr)
library(dplyr)
```

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

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

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

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

```
library(lubridate)
```

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

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

```
library(tseries)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
```

```
##   as.zoo.data.frame zoo
```

```
library(TTR)

AllStateClean <- read.table(here::here("Data","Other_data","AllStateClean.txt"),sep=";")

AllStateClean$MA_affected_residents <- gsub("\\,.*","",AllStateClean$MA_affected_residents) # this sele
AllStateClean$NC_affected_residents <- gsub("\\,.*","",AllStateClean$NC_affected_residents) # this sele

AllStateClean$Massachusetts[!is.na(AllStateClean$Massachusetts)]<-1
AllStateClean$New_Hampshire[!is.na(AllStateClean$New_Hampshire)]<-1
AllStateClean$North_Carolina[!is.na(AllStateClean$North_Carolina)]<-1
AllStateClean$Massachusetts[is.na(AllStateClean$Massachusetts)]<-0
AllStateClean$New_Hampshire[is.na(AllStateClean$New_Hampshire)]<-0
AllStateClean$North_Carolina[is.na(AllStateClean$North_Carolina)]<-0
```

Identifying treatment and control options

```
# Case 1: Experiment 1
massachusetts <- dplyr::filter(AllStateClean,Massachusetts==1)
massachusetts$reported_date <- substr(massachusetts$reported_date, start=1, stop=10)

new_hampshire <- dplyr::filter(AllStateClean,New_Hampshire==1)
new_hampshire$reported_date <- substr(new_hampshire$reported_date, start=1, stop=10)

# Case 1: Experiment 2
massachusetts$MA_affected_residents <- as.numeric(massachusetts$MA_affected_residents)
massachusetts1000 <- subset(massachusetts, massachusetts$MA_affected_residents > 1000)

n_carolina <- dplyr::filter(AllStateClean,North_Carolina==1)
n_carolina$NC_affected_residents <- as.numeric(n_carolina$NC_affected_residents)
n_carolina1000 <- subset(n_carolina, n_carolina$NC_affected_residents > 1000)
```

Specifying Treatment and Control Variables

```
Treat_Name <- 'Massachusetts'
Ctrl_Name <- 'North Carolina'

treatment <- massachusetts1000 # This Must Be Filled in to Work Properly!
control <- n_carolina1000  # This Must Be Filled in to Work Properly!
```

Specifying Time Variables

```
first_breach_date <- as.Date("2006-06-01")
data_start <- c(2006,6)
data_range <- c(2006,2020)
exp_start <- c(2008,4)
exp_range <- c(2008,2011)
```

```r
first_pop_date <- as.Date("2000-04-01")
no_months_out <- 174

months_prior <- 5 # months before treatment start
months_after <- 5 # months after treatment start

# Legislation H.B. 4144 signed into law August 3, 2007
# Legislation H.B. 4144 becomes effective on October 31, 2007
# OCABR finalized the regulation on September 22, 2008

treatment_start <- as.Date("09/22/2008", "%m/%d/%Y") # Legislation H.B. 4144 becomes effective
treatment_end <- as.Date("03/01/2010", "%m/%d/%Y") # post 6 months after enforcement
```

Produce Tables on Overlapping Co-occurance of Breach Incidents

```r
CoveredDays <- AllStateClean

CoveredDays$reported_date <- substr(CoveredDays$reported_date, start=1, stop=10)

CoveredDays <- subset(CoveredDays, reported_date > as.Date("2008-03-22") )
CoveredDays <- subset(CoveredDays, reported_date < as.Date("2010-09-01") )

table(CoveredDays$Massachusetts) # 0 = 313, 1 = 858
```

```
##
##   0   1
## 313 858
```

```r
table(CoveredDays$New_Hampshire) # 0 = 949, 1 = 222
```

```
##
##   0   1
## 949 222
```

```r
table(CoveredDays$North_Carolina) # 0 = 819, 1 = 352
```

```
##
##   0   1
## 819 352
```

```r
table(CoveredDays$Massachusetts,CoveredDays$New_Hampshire) # 0 = 205 + 108 ; 1 = 744 + 114
```

```
##
##       0   1
##   0 205 108
##   1 744 114
```

```r
table(CoveredDays$Massachusetts,CoveredDays$North_Carolina) # 0 = 99 + 214 ; 1 = 720 + 138
```

```
##
##       0   1
##   0  99 214
##   1 720 138
```

```r
CoveredDays$MA_affected_residents <- as.numeric(CoveredDays$MA_affected_residents)
CoveredDays$NC_affected_residents <- as.numeric(CoveredDays$NC_affected_residents)

CoveredDays$MA_BigBreach <- CoveredDays$MA_affected_residents > 1000
CoveredDays$NC_BigBreech <- CoveredDays$NC_affected_residents > 1000

# Need to resolve this R issue to get to a fixed solution

# table(CoveredDays$MA_BigBreach,CoveredDays$NC_BigBreach)
```

Create Population Time Series for Matching with Incident Frequncy

```r
# Population
pop <- read.csv(here::here("Data","Other_data","populations.csv")) # starts at 2000.04.01

datforpop <- data.frame(seq(first_pop_date, by="1 month", length.out=(length(pop)-1)))
names(datforpop) <- "yearmonth"
datforpop <- format(datforpop,  "%Y/%m")
datforpop <- rbind("yearmonth",datforpop)
row.names(datforpop) <- 1:nrow(datforpop)

pop <- cbind(datforpop,t(pop))
colnames(pop) <- pop[1,]
pop <- pop[-1,]
rownames(pop) <- seq(1:nrow(pop))
pop <- as.data.frame(pop)
```

Experiment 1: Create control and treatment populations with Total Incidents

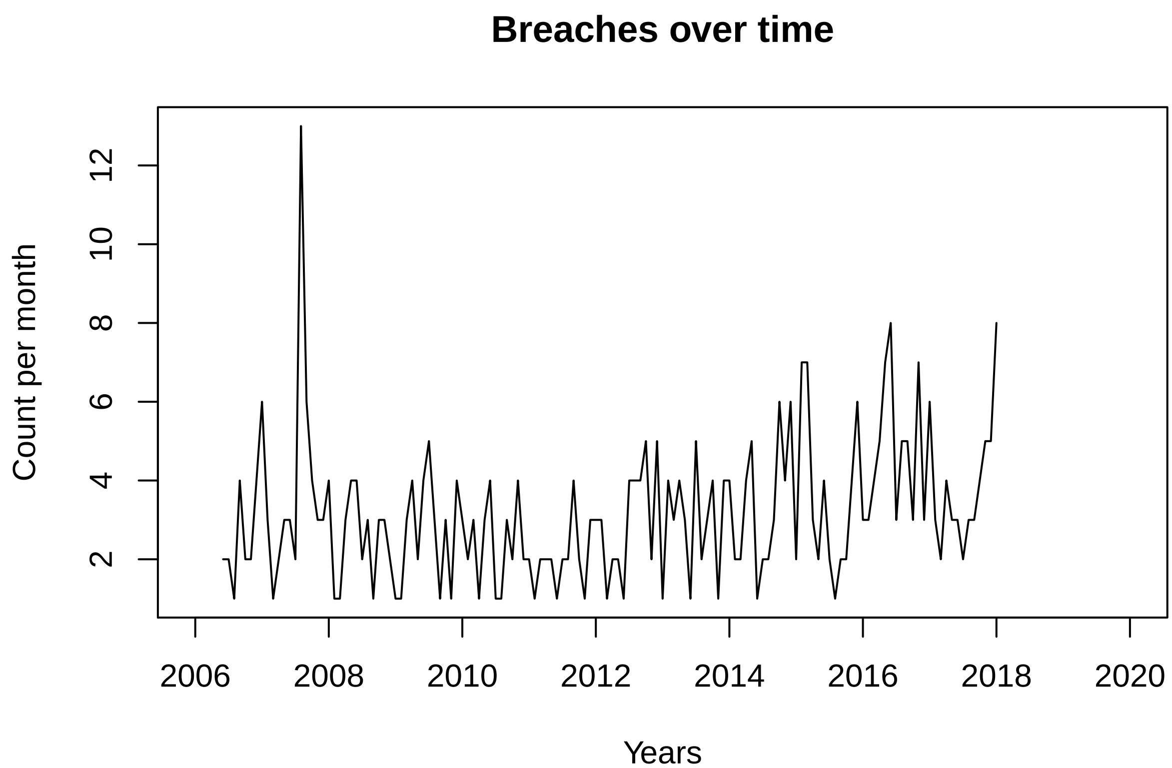
```r
# Format treatment dates into months
treatment$date_formatted <- format(as.Date(treatment$reported_date, "%Y-%m-%d"), "%Y/%m") # Alternative
treatment_freq <- treatment %>%
  dplyr::group_by(treatment$date_formatted) %>%
  dplyr::summarise(frequency = n(),)
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

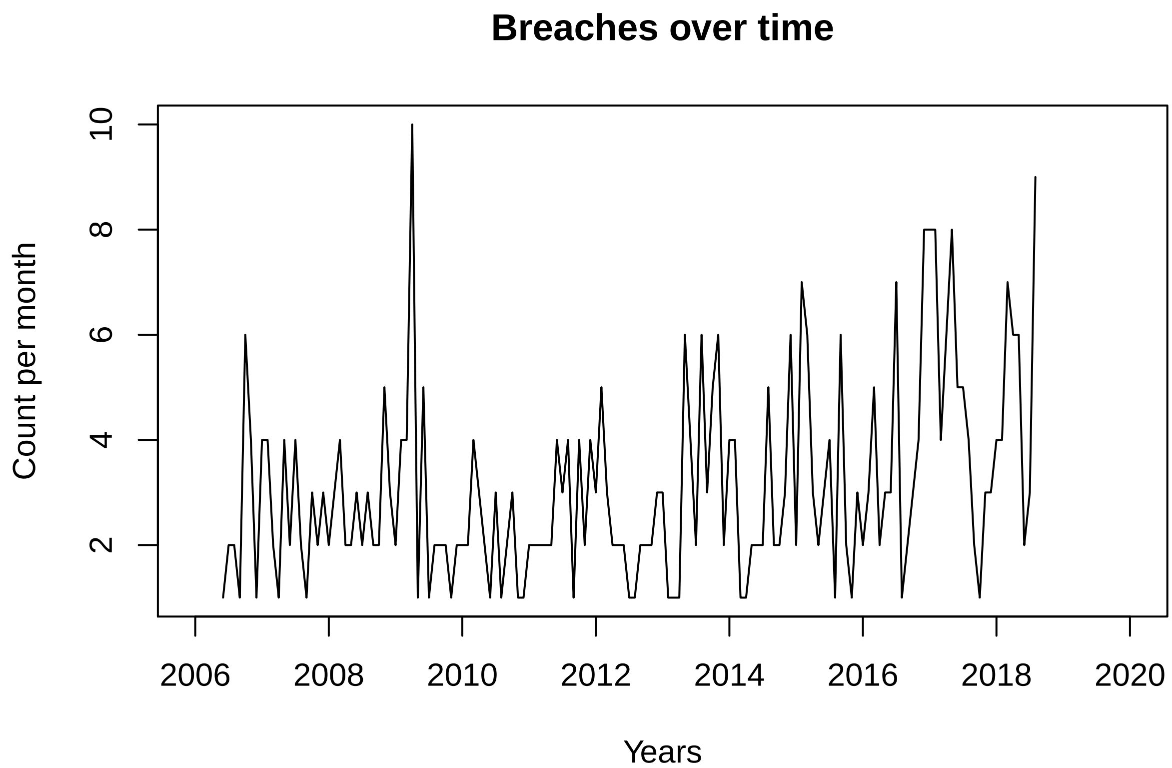
```r
names(treatment_freq)[1] <- "yearmonth"
treatment_freq$frequency[is.na(treatment_freq$frequency)]<-0

# Format control dates into months
control$date_formatted <- format(as.Date(control$reported_date, "%Y-%m-%d"), "%Y/%m") # Alternative is
control_freq <- control %>%
  dplyr::group_by(control$date_formatted) %>%
  dplyr::summarise(frequency = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

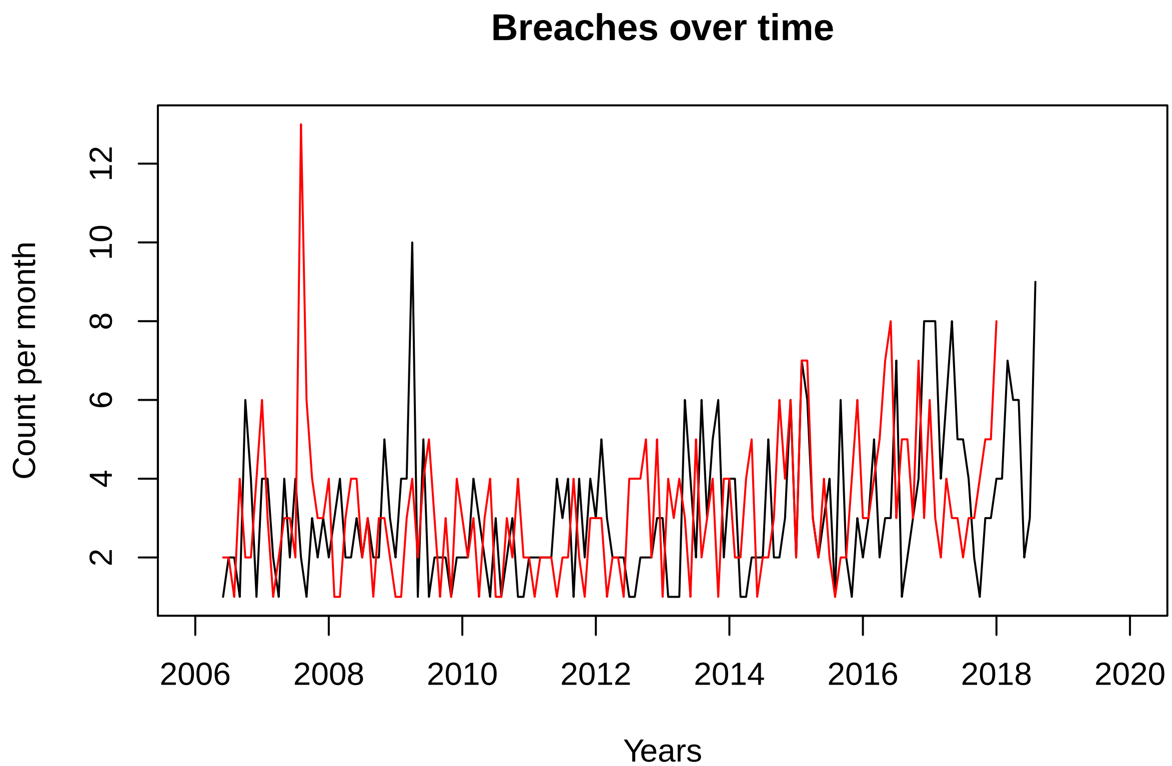
```
names(control_freq)[1] <- "yearmonth"
control_freq$frequency[is.na(control_freq$frequency)]<-0

treatment_ts <- ts(treatment_freq$frequency, frequency = 12, start = data_start)
control_ts <- ts(control_freq$frequency, frequency = 12, start = data_start)

plot.ts(treatment_ts, main = "Breaches over time", xlim=data_range, xlab = "Years", ylab = "Count per mo
```




```
plot.ts(control_ts, main = "Breaches over time", xlim=data_range, xlab = "Years", ylab = "Count per mont
```




```
ts.plot(control_ts, treatment_ts, main = "Breaches over time", xlim=data_range, gpars = list(col = c("bl
```




```
summary(treatment_ts)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.164   4.000  13.000
```

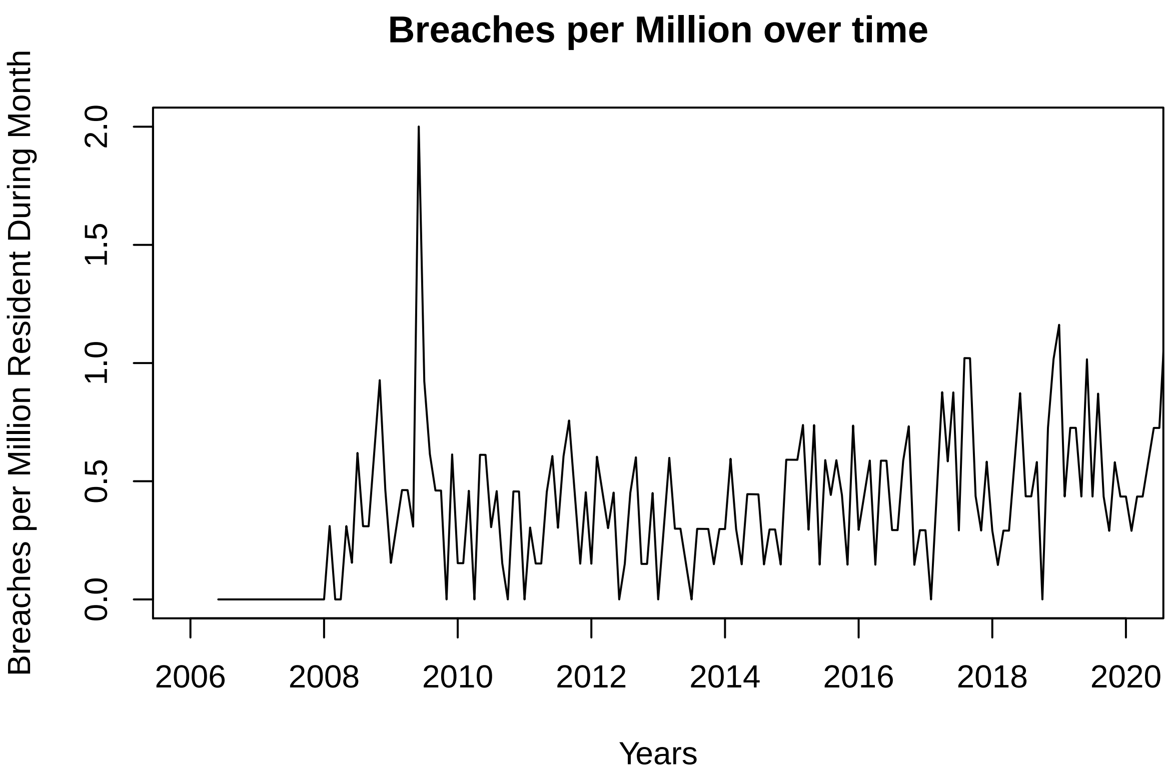
```
summary(control_ts)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   3.000   3.116   4.000  10.000
```

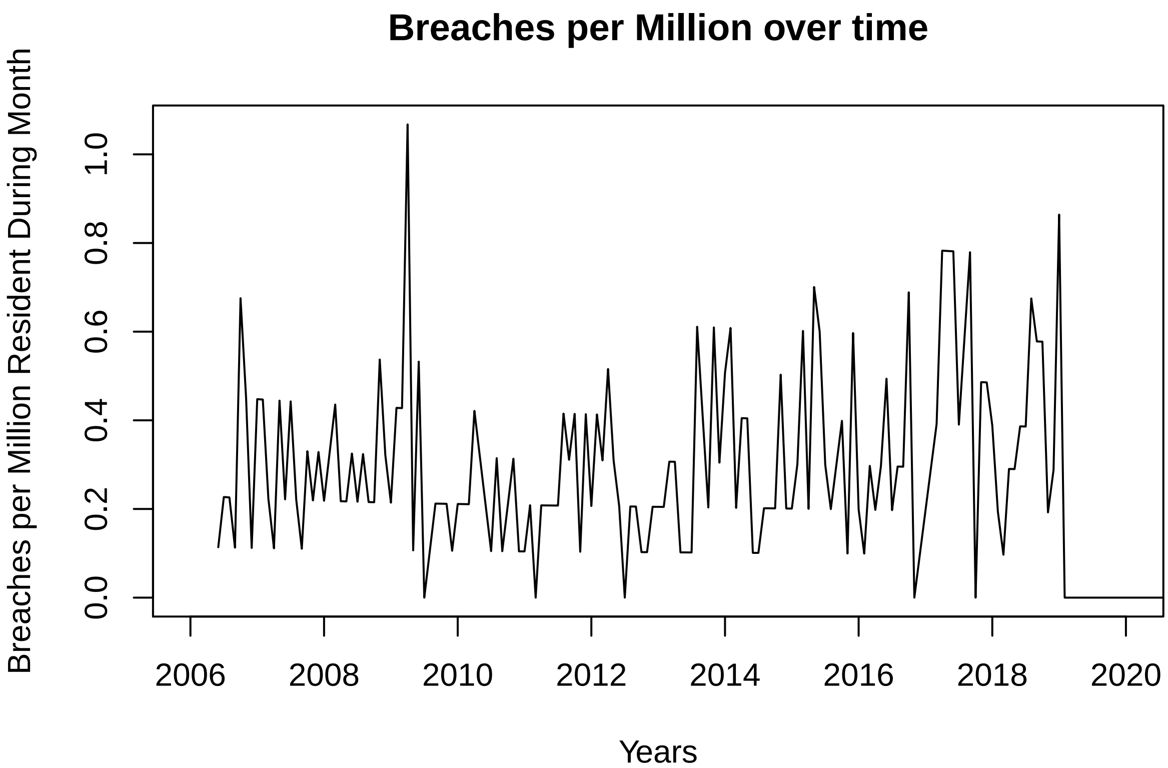
Create charts with breaches per million residents

```
# Merge Treatment and Control Together
comb_ts <- merge(treatment_freq, control_freq, by="yearmonth", all=TRUE)

# Merge Combined Treatment and Control Together with Population Statistics
comb_ts <- merge(comb_ts, pop, by='yearmonth', all.x = TRUE)
comb_ts$frequency.x[is.na(comb_ts$frequency.x)]<-0
comb_ts$frequency.y[is.na(comb_ts$frequency.y)]<-0

# change class of columns to numeric
comb_ts[2:ncol(comb_ts)] <- sapply(comb_ts[2:ncol(comb_ts)],as.numeric)

comb_ts$treatpermil <- comb_ts$frequency.x/(comb_ts[,c(Treat_Name)]/1000000)
comb_ts$controlpermil <- comb_ts$frequency.y/(comb_ts[,c(Ctrl_Name)]/1000000)
```

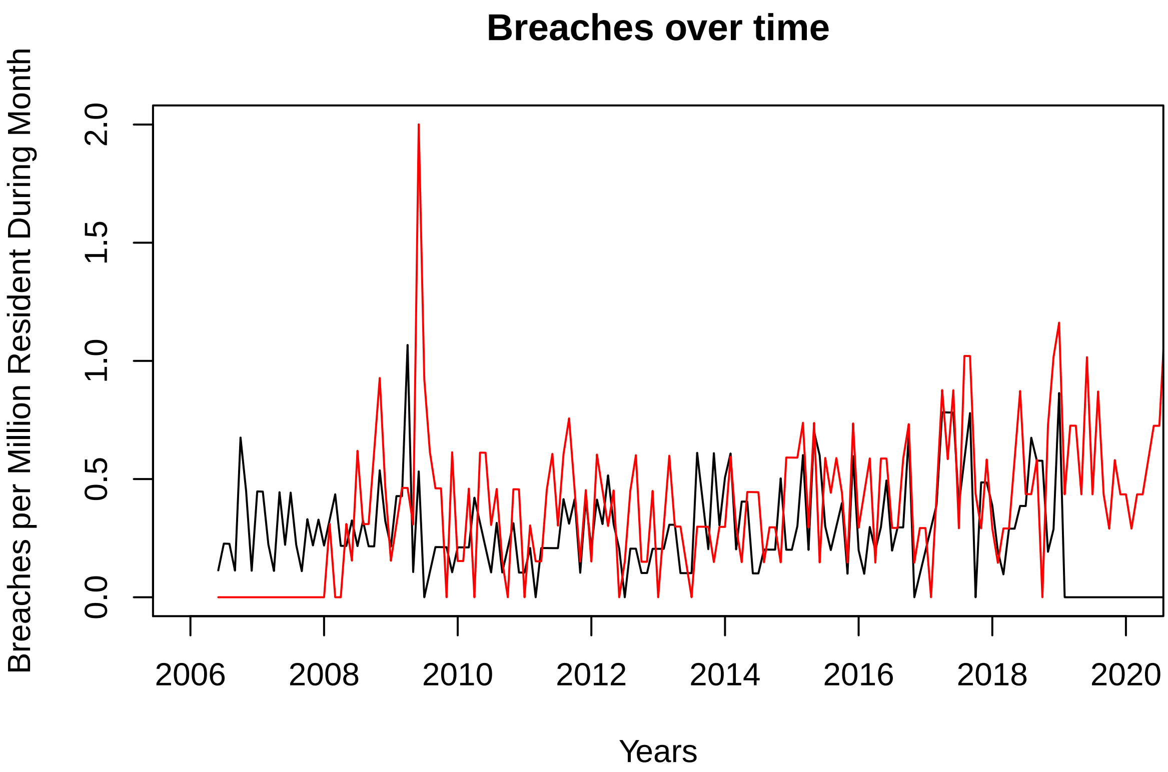
```
treatment_tsM <- ts(comb_ts$treatpermil, frequency = 12, start = data_start)
control_tsM <- ts(comb_ts$controlpermil, frequency = 12, start = data_start)

plot.ts(treatment_tsM, main = "Breaches per Million over time", xlim=data_range, xlab = "Years", ylab =
```




```
plot.ts(control_tsM, main = "Breaches per Million over time", xlim=data_range, xlab = "Years", ylab = "
```




```
ts.plot(control_tsM, treatment_tsM, main = "Breaches over time", xlim=data_range, gpars = list(col = c(
```



Identifying and subsetting relevant dates

```
treatment_start<- format(as.Date(as.character(treatment_start), origin = "1970-01-01"), "%Y/%m")
treatment_end<- format(as.Date(as.character(treatment_end), origin = "1970-01-01"), "%Y/%m")

pretreat <- comb_ts[(which(comb_ts$yearmonth==treatment_start)-months_prior):which(comb_ts$yearmonth==tr
pretreat$type <- "pretest"

posttreat <- comb_ts[which(comb_ts$yearmonth==treatment_end):(which(comb_ts$yearmonth==treatment_end)+mo
posttreat$type <- "posttest"

mean(posttreat$treatpermil) - mean(pretreat$treatpermil)
```

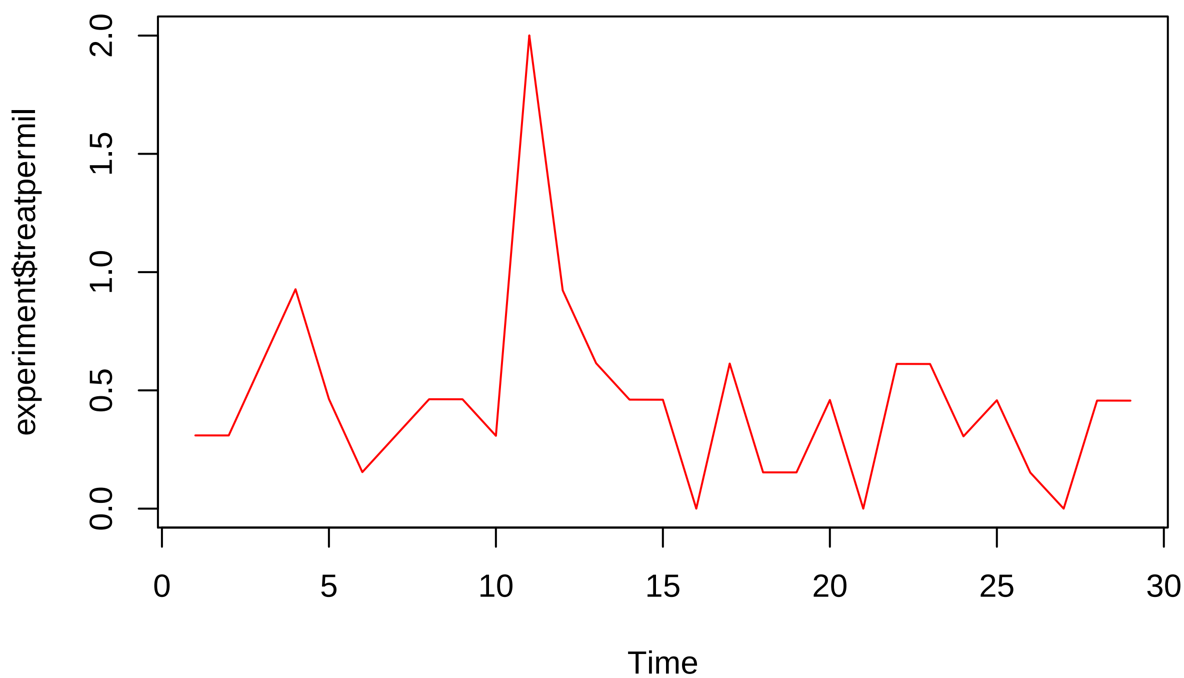
```
## [1] -0.1588701
```

```
mean(posttreat$controlpermil) - mean(pretreat$controlpermil)
```

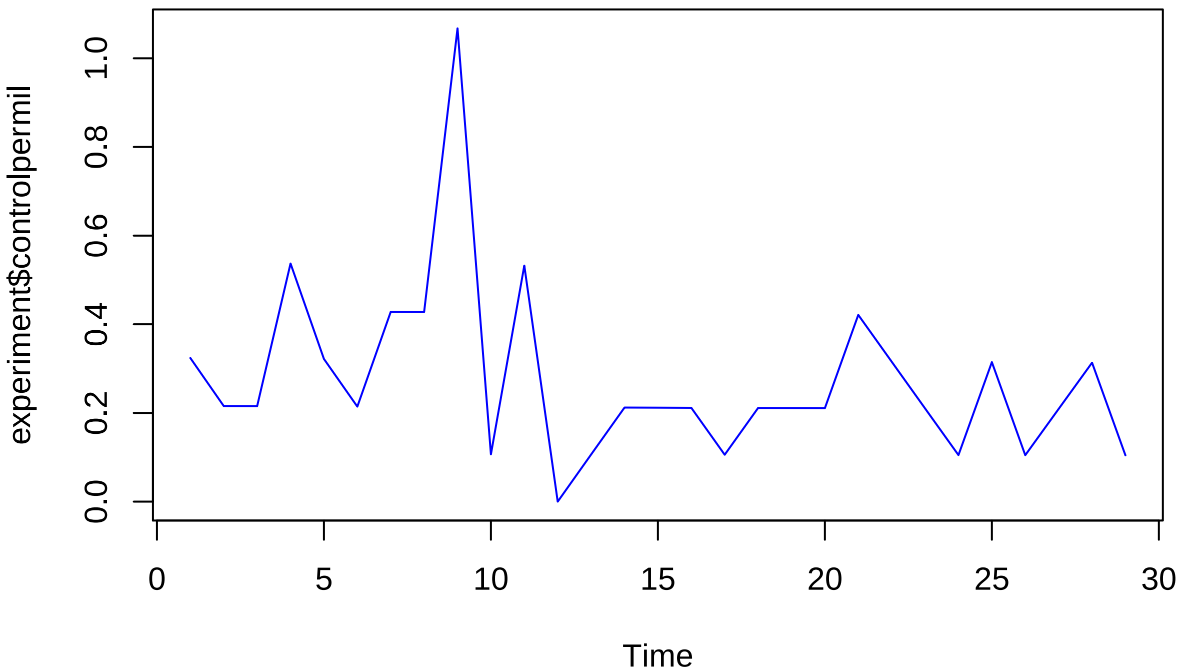
```
## [1] -0.1127882
```

```
treatment_range <- comb_ts[(which(comb_ts$yearmonth==treatment_start)+1):(which(comb_ts$yearmonth==treat
treatment_range$type <- "test"

experiment <- rbind(pretreat,treatment_range,posttreat)
experiment$treatpermil[is.na(experiment$treatpermil)]<-0
experiment$controlpermil[is.na(experiment$controlpermil)]<-0
```

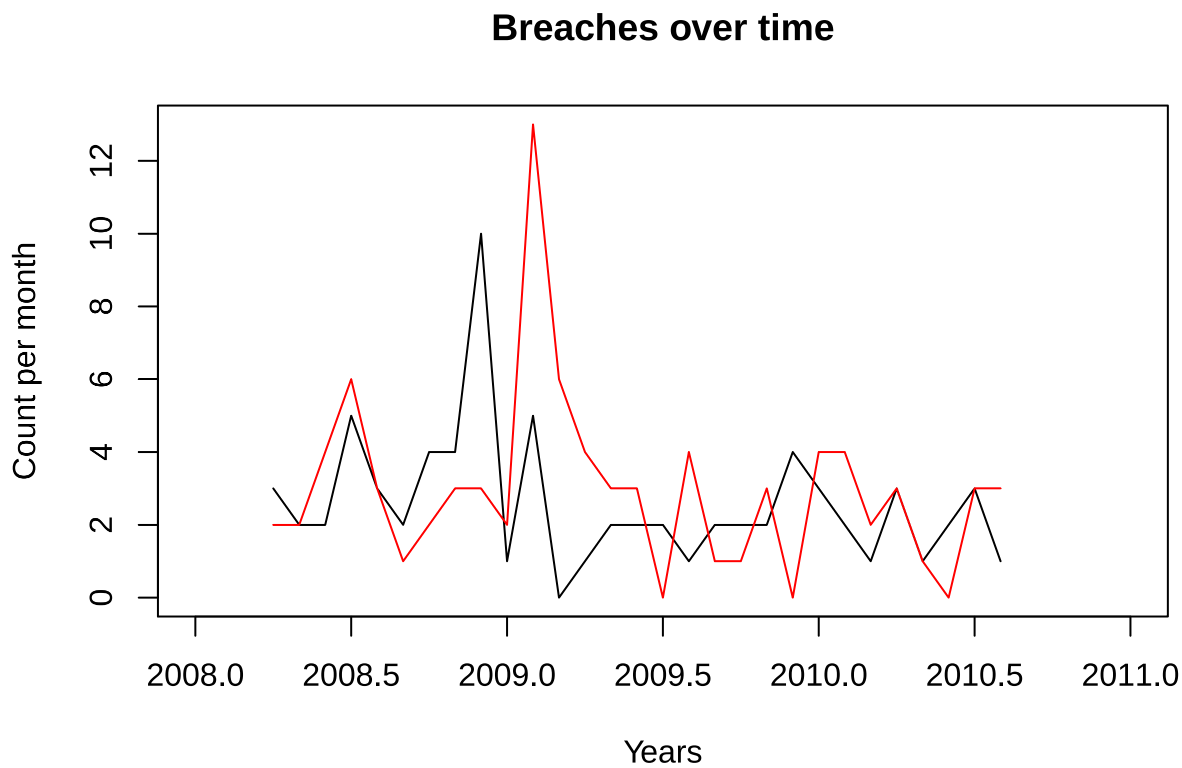
```r
ts.plot(experiment$treatpermil, col = "red")
```




```r
ts.plot(experiment$controlpermil, col = "blue")
```




```r
# Look at Raw Frequency Counts

treatment_ts <- ts(experiment$frequency.x, frequency = 12, start = exp_start)
control_ts <- ts(experiment$frequency.y, frequency = 12, start = exp_start)
ts.plot(control_ts, treatment_ts, main = "Breaches over time", xlim=exp_range, gpars = list(col = c("bl
```




```r
# Look at Treatment and Control per Million

treatment_tsM <- ts(experiment$treatpermil, frequency = 12, start = exp_start)
mean(treatment_tsM)
```

```
## [1] 0.4557598
```

```r
sd(treatment_tsM)
```

```
## [1] 0.3803766
```

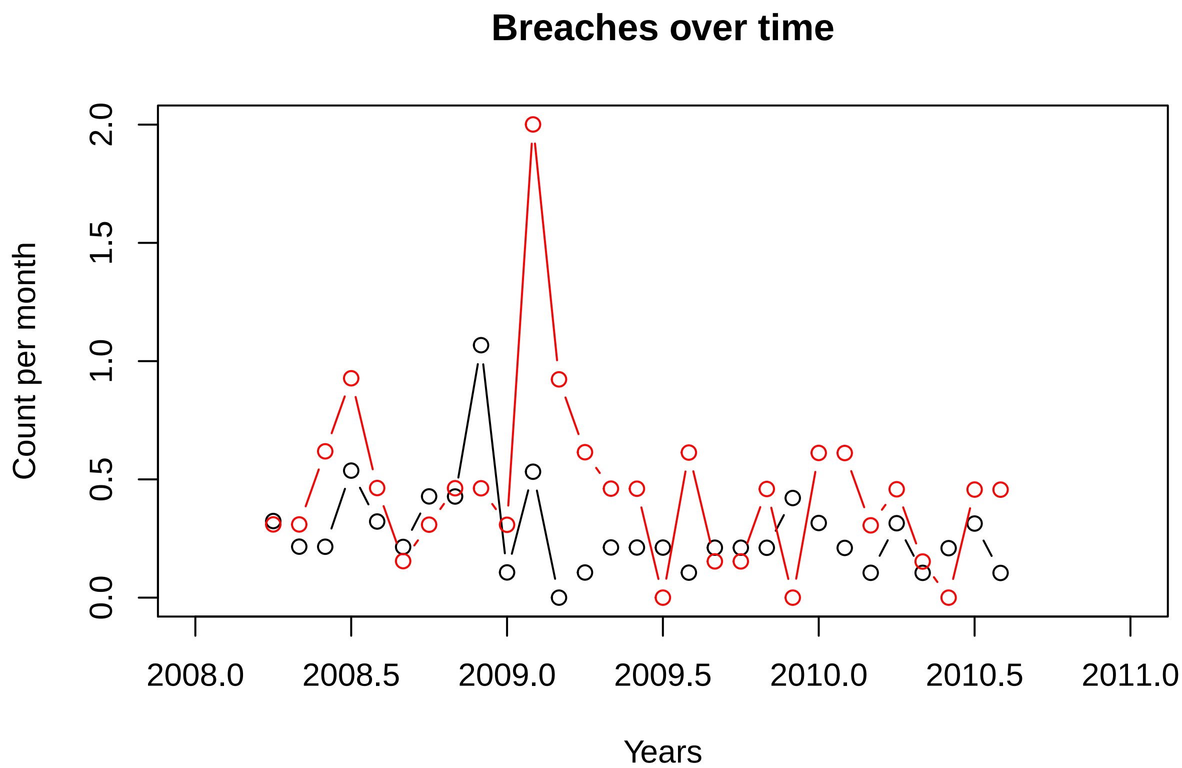
```r
control_tsM <- ts(experiment$controlpermil, frequency = 12, start = exp_start)
mean(control_tsM)
```

```
## [1] 0.2747875
```

```r
sd(control_tsM)
```

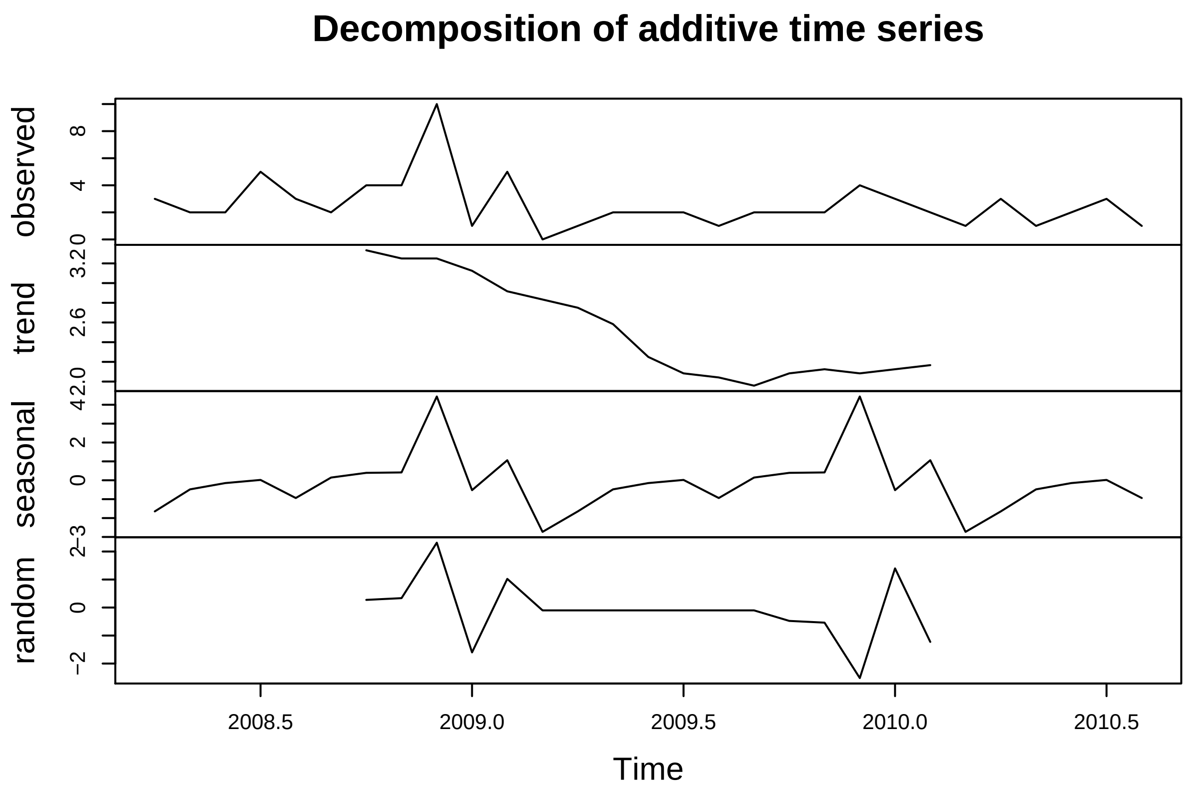
```
## [1] 0.2010686
```

```r
ts.plot(control_tsM, treatment_tsM, main = "Breaches over time", xlim=exp_range,
                    gpars = list(col = c("black", "red")), type = "b", xlab = "Years", ylab = "Count pe
```

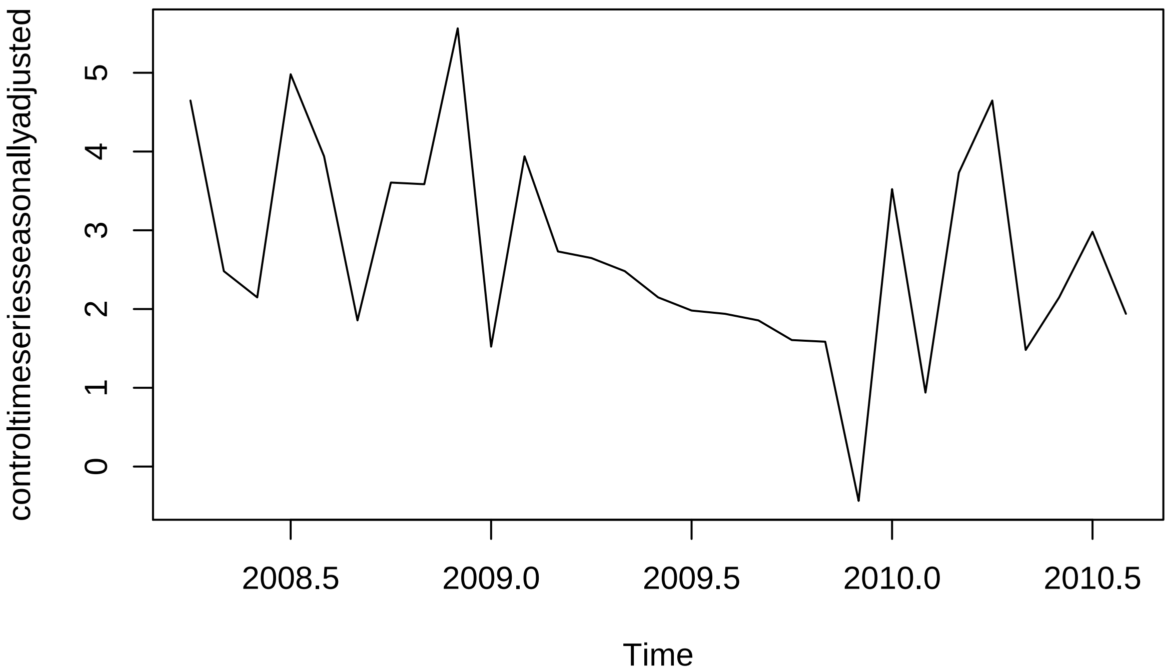


Decompose the data to find seasonal patterns

```
controltimeseriescomponents <- decompose(control_ts)
plot(controltimeseriescomponents)
```

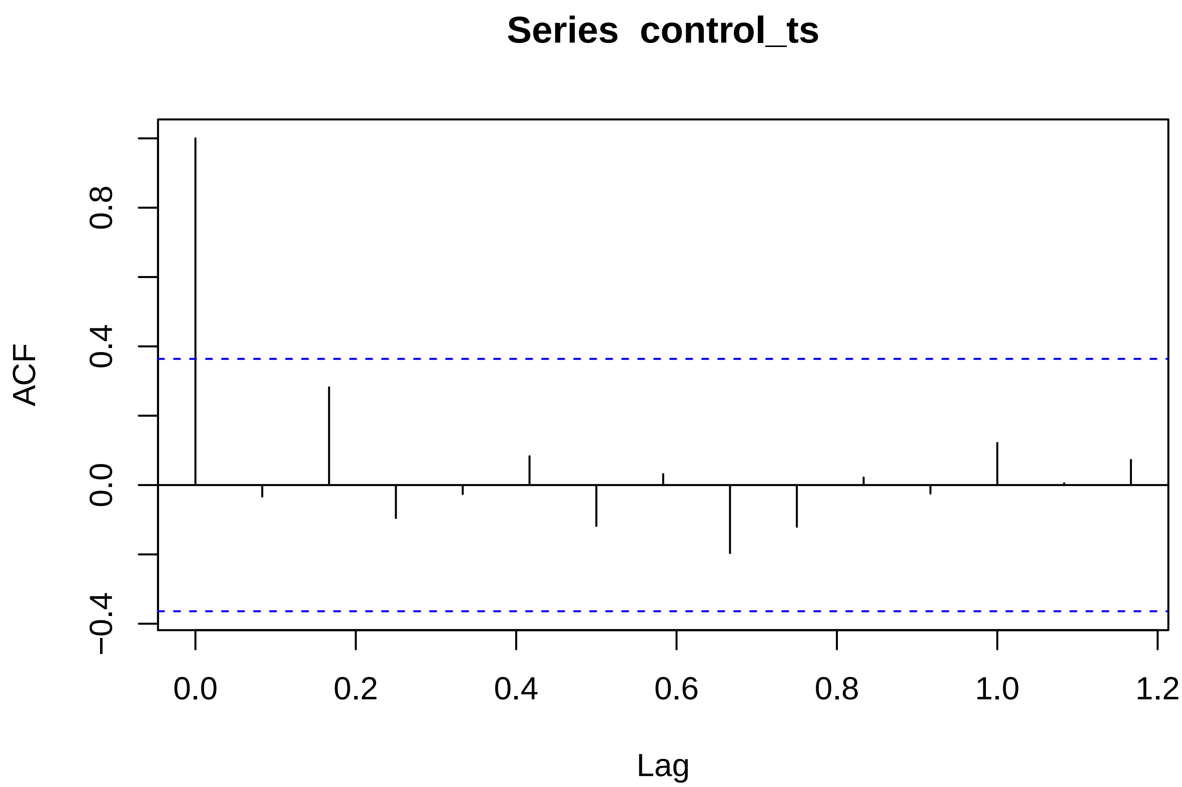



```r
controltimeseriesseasonallyadjusted <- control_ts - controltimeseriescomponents$seasonal
plot(controltimeseriesseasonallyadjusted)
```

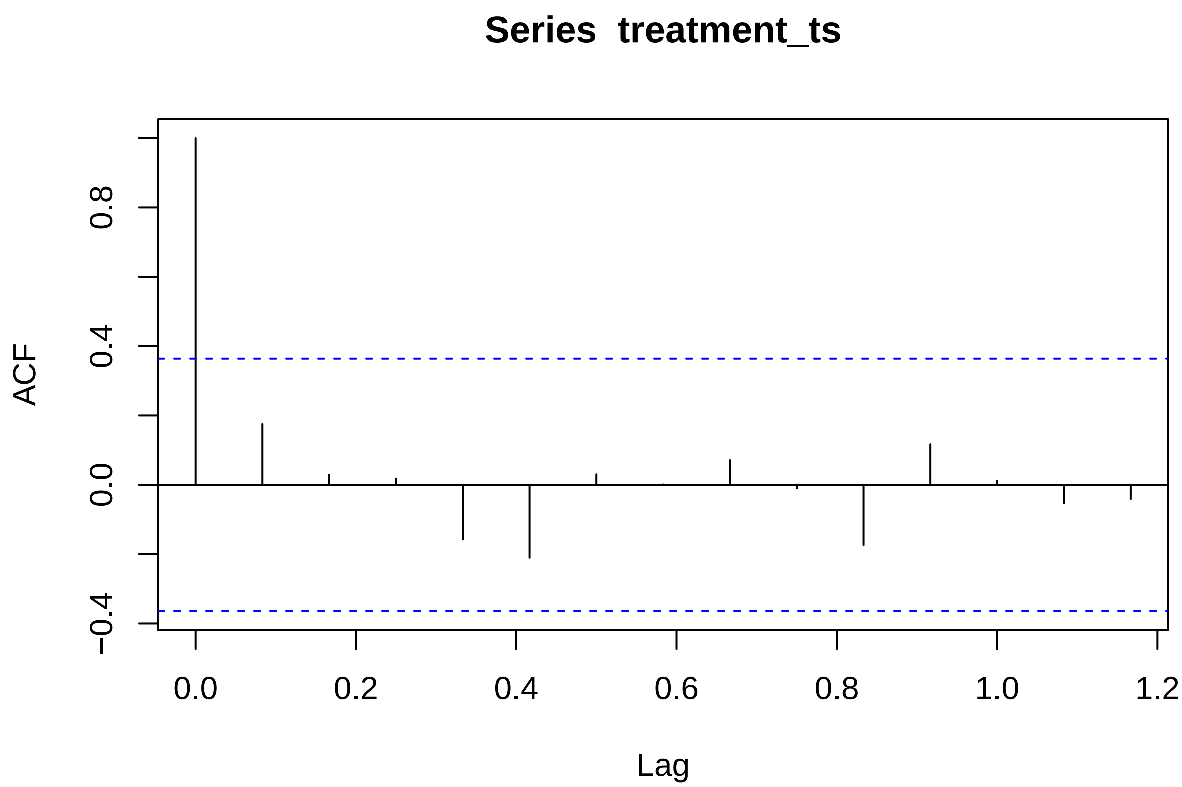


Run Statistical Tests on Time Series for Stationarity

```
# source of statistical tests http://r-statistics.co/Time-Series-Analysis-With-R.html

acfcontrol <- acf(control_ts) # autocorrelation (i.e. a Time Series with lags of itself)
```




```
acftreatment <- acf(treatment_ts)
```




```r
# shows that the control time series is a "stationary time series"

png(here::here("Output","acfcontrol.png"))
plot(acfcontrol)

png(here::here("Output","acftreatmentNH.png"))
plot(acftreatment)

pacfcontrolNH <- pacf(control_ts)  # partial autocorrelation (i.e. correlation of the time series with 
pacftreatmentMA <- pacf(treatment_ts)  # partial autocorrelation (i.e. correlation of the time series w

png(here::here("Output","pacfcontrolNH.png"))
plot(pacfcontrolNH)

png(here::here("Output","pacftreatmentMA.png"))
plot(pacftreatmentMA)

ccfRes <- ccf(control_ts, treatment_ts, ylab = "cross-correlation")
ccfRes
```

```
##
## Autocorrelations of series 'X', by lag
##
## -0.9167 -0.8333 -0.7500 -0.6667 -0.5833 -0.5000 -0.4167 -0.3333 -0.2500 -0.1667
##   0.019  -0.134  -0.068   0.086   0.055   0.106  -0.078   0.322   0.346   0.565
## -0.0833  0.0000  0.0833  0.1667  0.2500  0.3333  0.4167  0.5000  0.5833  0.6667
##  -0.061   0.212  -0.234  -0.180  -0.109  -0.018   0.127  -0.077  -0.050  -0.007
```

```
## 0.7500 0.8333 0.9167
## -0.076 0.162 0.076
```

```r
# adf test is an Augmented Dickey-Fuller Test
adf.test(control_ts) # p-value < 0.05 indicates the TS is stationary
```

```
##
## Augmented Dickey-Fuller Test
##
## data: control_ts
## Dickey-Fuller = -2.8267, Lag order = 3, p-value = 0.2557
## alternative hypothesis: stationary
```

```r
adf.test(treatment_ts)
```

```
##
## Augmented Dickey-Fuller Test
##
## data: treatment_ts
## Dickey-Fuller = -2.6375, Lag order = 3, p-value = 0.3282
## alternative hypothesis: stationary
```

```r
kpss.test(control_ts) # Kwiatkowski-Phillips-Schmidt-Shin (KPSS) testz
```

```
## Warning in kpss.test(control_ts): p-value greater than printed p-value
```

```
##
## KPSS Test for Level Stationarity
##
## data: control_ts
## KPSS Level = 0.2822, Truncation lag parameter = 2, p-value = 0.1
```

```r
kpss.test(treatment_ts)
```

```
## Warning in kpss.test(treatment_ts): p-value greater than printed p-value
```

```
##
## KPSS Test for Level Stationarity
##
## data: treatment_ts
## KPSS Level = 0.18972, Truncation lag parameter = 2, p-value = 0.1
```

```r
# https://www.sas.com/content/dam/SAS/en_ca/User%20Group%20Presentations/Health-User-Groups/ITS_SAS.pdf
```

ITS analyses use regression-based techniques

```r
quasiexp <- experiment[experiment$type != "test",]

# Added dummy variables for ITS
control <- as.data.frame(t(rbind(quasiexp$yearmonth,quasiexp$controlpermil)))
control$treat <- as.vector(rep(0,nrow(control)))                  # Create example vector
time <- 1:nrow(control)
control$time <- as.vector(time)
control$z <- c(rep(0,6),1:(nrow(control)-6))

treatment <- as.data.frame(t(rbind(quasiexp$yearmonth,quasiexp$treatpermil)))
treatment$treat <- as.vector(rep(1,nrow(control)))                # Create example vector
```

```r
time <- 1:nrow(control)
treatment$time <- as.vector(time)
treatment$z <- c(rep(0,6),1:(nrow(control)-6))
treatment
```

```
##         V1                V2 treat time z
## 1  2008/04 0.309615975565107     1    1 0
## 2  2008/05 0.309466646613228     1    2 0
## 3  2008/06 0.618634827594207     1    3 0
## 4  2008/07 0.927505117895948     1    4 0
## 5  2008/08  0.46346211452118     1    5 0
## 6  2008/09 0.154390677890869     1    6 0
## 7  2010/03 0.305609741860619     1    7 1
## 8  2010/04 0.458181121746513     1    8 2
## 9  2010/05 0.152581953674593     1    9 3
## 10 2010/06                 0     1   10 4
## 11 2010/07 0.456877815795088     1   11 5
## 12 2010/08 0.456603838485829     1   12 6
```

```r
AppendITS <- rbind(treatment,control)
names(AppendITS) <- c("yearmonth","incident_permil","treat","time","z")
AppendITS$incident_permil <- as.numeric(as.character(AppendITS$incident_permil))
AppendITS$time <- as.numeric(as.character(AppendITS$time))
AppendITS$z <- as.numeric(as.character(AppendITS$z))
AppendITS
```

```
##    yearmonth incident_permil treat time z
## 1    2008/04       0.3096160     1    1 0
## 2    2008/05       0.3094666     1    2 0
## 3    2008/06       0.6186348     1    3 0
## 4    2008/07       0.9275051     1    4 0
## 5    2008/08       0.4634621     1    5 0
## 6    2008/09       0.1543907     1    6 0
## 7    2010/03       0.3056097     1    7 1
## 8    2010/04       0.4581811     1    8 2
## 9    2010/05       0.1525820     1    9 3
## 10   2010/06       0.0000000     1   10 4
## 11   2010/07       0.4568778     1   11 5
## 12   2010/08       0.4566038     1   12 6
## 13   2008/04       0.3239183     0    1 0
## 14   2008/05       0.2155742     0    2 0
## 15   2008/06       0.2152042     0    3 0
## 16   2008/07       0.5370887     0    4 0
## 17   2008/08       0.3218496     0    5 0
## 18   2008/09       0.2142979     0    6 0
## 19   2010/03       0.1049765     0    7 1
## 20   2010/04       0.3146144     0    8 2
## 21   2010/05       0.1047293     0    9 3
## 22   2010/06       0.2091749     0   10 4
## 23   2010/07       0.3133381     0   11 5
## 24   2010/08       0.1043704     0   12 6
```

```r
factor_cols <- c("treat","time","z")

sapply(AppendITS, class)
```

```
##       yearmonth incident_permil           treat            time               z
##     "character"       "numeric"       "numeric"       "numeric"       "numeric"
```

```r
regTest <- lm(incident_permil ~ time + treat + z, AppendITS)
summary(regTest)
```

```
##
## Call:
## lm(formula = incident_permil ~ time + treat + z, data = AppendITS)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.32980 -0.11196 -0.02624  0.10740  0.50431
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.353462   0.127121   2.781   0.0115 *
## time        -0.016603   0.028084  -0.591   0.5610
## treat        0.136149   0.078682   1.730   0.0990 .
## z            0.001556   0.045596   0.034   0.9731
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1927 on 20 degrees of freedom
## Multiple R-squared:  0.1968, Adjusted R-squared:  0.0763
## F-statistic: 1.633 on 3 and 20 DF,  p-value: 0.2134
```

```r
regTest2 <- lm(incident_permil ~ time + treat + time*treat + z + z*time + z*treat + z*treat*time, Appen
summary(regTest2)
```

```
##
## Call:
## lm(formula = incident_permil ~ time + treat + time * treat +
##     z + z * time + z * treat + z * treat * time, data = AppendITS)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.33195 -0.09059 -0.02198  0.10374  0.45334
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.319441   0.184434   1.732    0.103
## time         -0.007660   0.045826  -0.167    0.869
## treat         0.130376   0.260830   0.500    0.624
## z            -0.044169   0.263181  -0.168    0.869
## time:treat    0.013747   0.064808   0.212    0.835
## time:z        0.003091   0.020163   0.153    0.880
## treat:z      -0.324775   0.372194  -0.873    0.396
## time:treat:z  0.026808   0.028515   0.940    0.361
##
## Residual standard error: 0.2016 on 16 degrees of freedom
## Multiple R-squared:  0.2968, Adjusted R-squared:  -0.01082
## F-statistic: 0.9648 on 7 and 16 DF,  p-value: 0.488
```

```r
AppendITS
```

```
##    yearmonth incident_permil treat time z
## 1    2008/04       0.3096160     1    1 0
## 2    2008/05       0.3094666     1    2 0
## 3    2008/06       0.6186348     1    3 0
## 4    2008/07       0.9275051     1    4 0
## 5    2008/08       0.4634621     1    5 0
## 6    2008/09       0.1543907     1    6 0
## 7    2010/03       0.3056097     1    7 1
## 8    2010/04       0.4581811     1    8 2
## 9    2010/05       0.1525820     1    9 3
## 10   2010/06       0.0000000     1   10 4
## 11   2010/07       0.4568778     1   11 5
## 12   2010/08       0.4566038     1   12 6
## 13   2008/04       0.3239183     0    1 0
## 14   2008/05       0.2155742     0    2 0
## 15   2008/06       0.2152042     0    3 0
## 16   2008/07       0.5370887     0    4 0
## 17   2008/08       0.3218496     0    5 0
## 18   2008/09       0.2142979     0    6 0
## 19   2010/03       0.1049765     0    7 1
## 20   2010/04       0.3146144     0    8 2
## 21   2010/05       0.1047293     0    9 3
## 22   2010/06       0.2091749     0   10 4
## 23   2010/07       0.3133381     0   11 5
## 24   2010/08       0.1043704     0   12 6
```

```r
# View(AppendITS)
```