# HITECH_Act_Case

Karl Grindal

6/19/2020

```
HITECH <- read.csv(here::here("Data","Other_data","HITECH.csv"))
DUNS <- read.csv(here::here("Data","Other_data","DUNS_HITECH.csv"))

NAICS <- merge(HITECH, DUNS, by="clean_name", all=TRUE)

NAICS$DUNS_2 <- substr(NAICS$Primary.NAICS.Code, start=1, stop=2)
# colSums(!is.na(NAICS)) # 1740 resolved for Duns_2
# colSums(is.na(NAICS)) # 248 not resolved for Duns_2

248/1740 # = 14.2% are not included
```

```
## [1] 0.1425287
```

```
1740/(1740+248) # = 87.5% of incidents are resolved
```

```
## [1] 0.8752515
```

```
write.csv(NAICS, here::here("Data","Other_data","NAICS_Clean.csv"))
```

## Create Population Time Series for Matching with Incident Frequncy

```
# Population
pop <- read.csv(here::here("Data","Other_data","populations.csv"))
pop <- pop[c(12,20,22,30,42),] # Hawaii row 12, Maine row 20, Massachusetts row 22, New Hampshire row 3

datforpop <- data.frame(seq(as.Date("2000-04-01"), 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)

# Create a new column for the total population accross collecting states
pop[,2:6] <- sapply(pop[,2:6],as.numeric)
pop$totpop <- rowSums(pop[ ,c(2:6)], na.rm=TRUE)
```

# Create Treatment and Control Groups

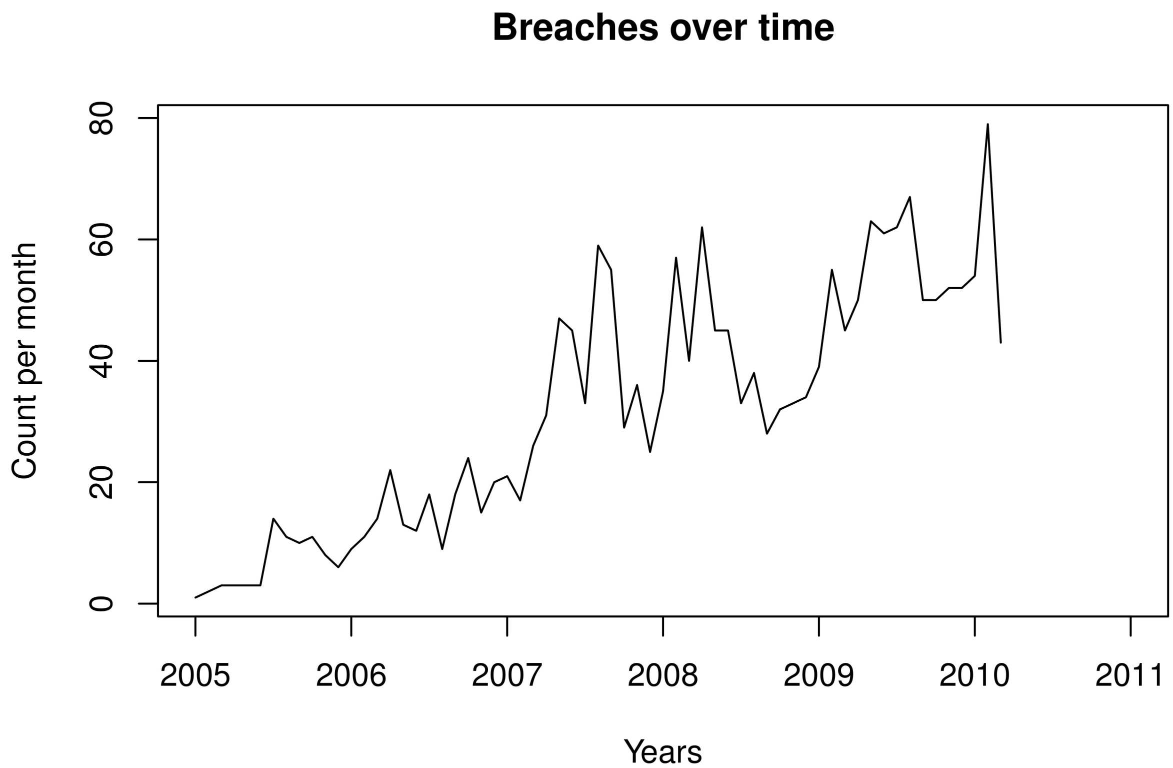
```r
# HIGH TECH Act regulations become effective on February 17, 2009
# Enforcement of HIGH TECH Act implement on May 27, 2009

experiment_start <- as.Date("02/17/2009", "%m/%d/%Y")-180 # Legislation H.B. 4144 becomes effective
experiment_end <- as.Date("05/27/2009", "%m/%d/%Y")+180 # moving avg uses 30 days post treatment

datfull <- data.frame(seq(as.Date(experiment_start), by="1 month", length.out=16))
names(datfull) <- "yearmonth"
datfull <- format(datfull,  "%Y/%m")
datfull
```

```
##    yearmonth
## 1    2008/08
## 2    2008/09
## 3    2008/10
## 4    2008/11
## 5    2008/12
## 6    2009/01
## 7    2009/02
## 8    2009/03
## 9    2009/04
## 10   2009/05
## 11   2009/06
## 12   2009/07
## 13   2009/08
## 14   2009/09
## 15   2009/10
## 16   2009/11
```

```r
NAICS$date_formatted <- format(as.Date(NAICS$firstdate, "%Y-%m-%d"), "%Y/%m") # Alternative is "%m/%d/%
NAICS_ts <- NAICS %>%
  dplyr::group_by(NAICS$date_formatted) %>%
  dplyr::summarise(frequency = n())
```

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

```r
names(NAICS_ts)[1] <- "yearmonth"
NAICS_ts <- merge(NAICS_ts, pop, by="yearmonth", all.x=TRUE)
NAICS_ts$frequency[is.na(NAICS_ts$frequency)]<-0

# Look at Raw Frequency Counts

NAICS_ts <- ts(NAICS_ts$frequency, frequency = 12, start = c(2005,1))

ts.plot(NAICS_ts, main = "Breaches over time", xlim=c(2005,2011), gpars = list(col = c("black", "red"))
```




### Create potential treatment and population groups

```
# Format control dates into months
NAICS$date_formatted <- format(as.Date(NAICS$firstdate, "%Y-%m-%d"), "%Y/%m") # Alternative is "%m/%d/%
NAICS <- NAICS[!is.na(NAICS$date_formatted),]

Health <- NAICS[grepl( "62", NAICS$DUNS_2), ]
Finance <- NAICS[grepl( "52", NAICS$DUNS_2), ]
Education <- NAICS[grepl( "61", NAICS$DUNS_2), ]
Information <- NAICS[grepl( "51", NAICS$DUNS_2), ]

Non_Health <- NAICS %>%
                   filter(DUNS_2 != 62)
```

### Experiment 1: Create control and treatment populations (Massachusetts v New Hampshire) with Total Incidents

```
treatment <-  Health # This Must Be Filled in to Work Properly!
control <-  Non_Health  # This Must Be Filled in to Work Properly!

# Format treatment dates into months
```

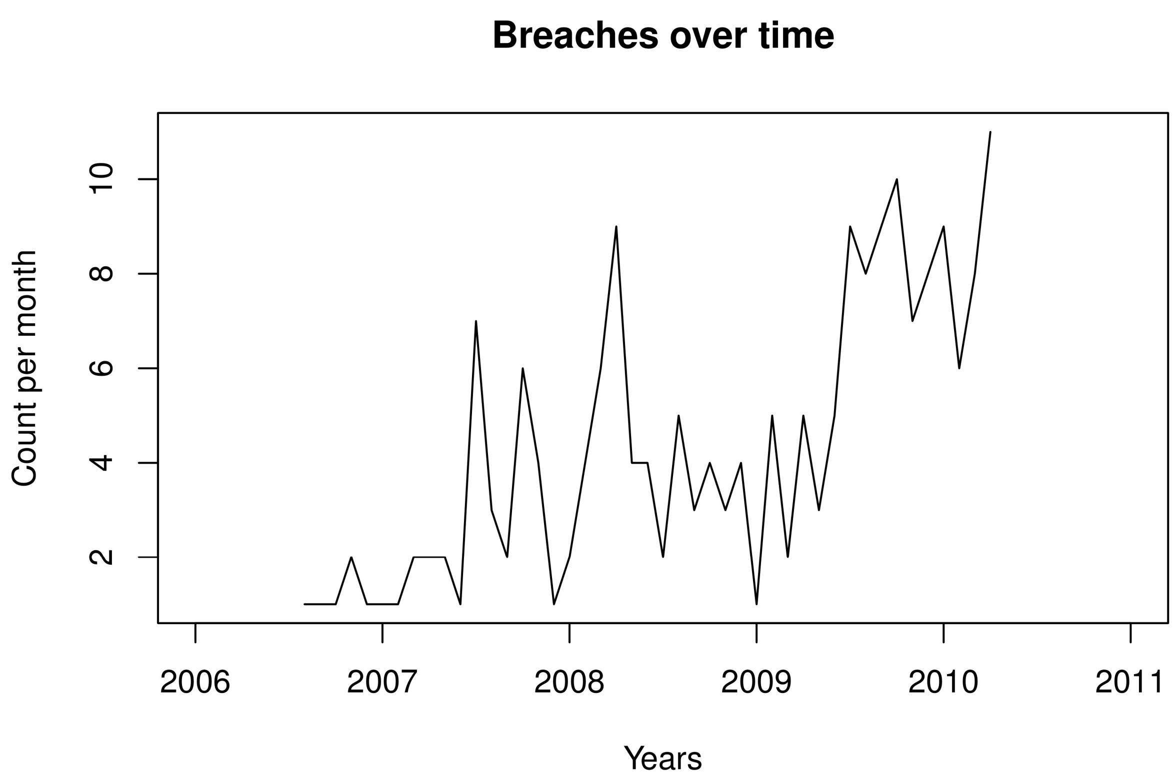
```r
treatment$date_formatted <- format(as.Date(treatment$reported_date, "%Y-%m-%d"), "%Y/%m") # Alternative
treatment_freq <- treatment %>%
  dplyr::group_by(treatment$date_formatted) %>%
  dplyr::summarise(frequency = n(),)
```

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

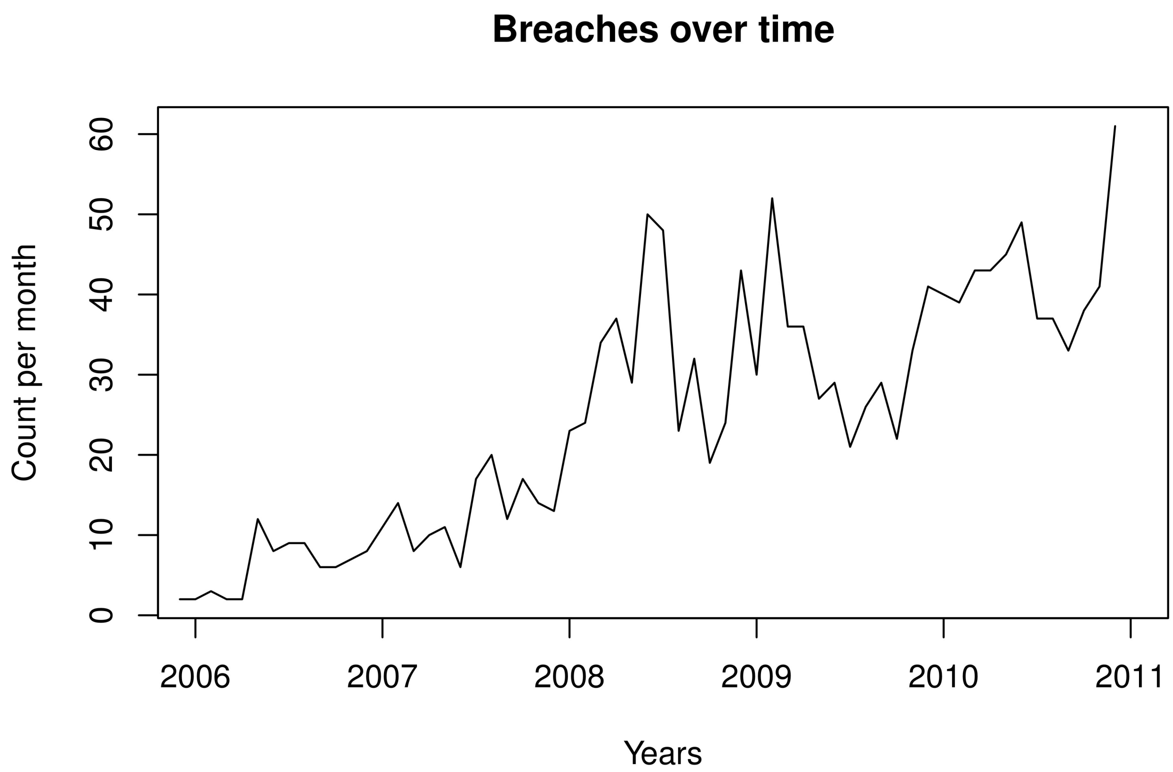
```r
names(treatment_freq)[1] <- "yearmonth"
treatment_freq$frequency[is.na(treatment_freq$frequency)]<-0
treatment_freq<- merge(datfull,treatment_freq, by="yearmonth", all=TRUE)

# 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)
```

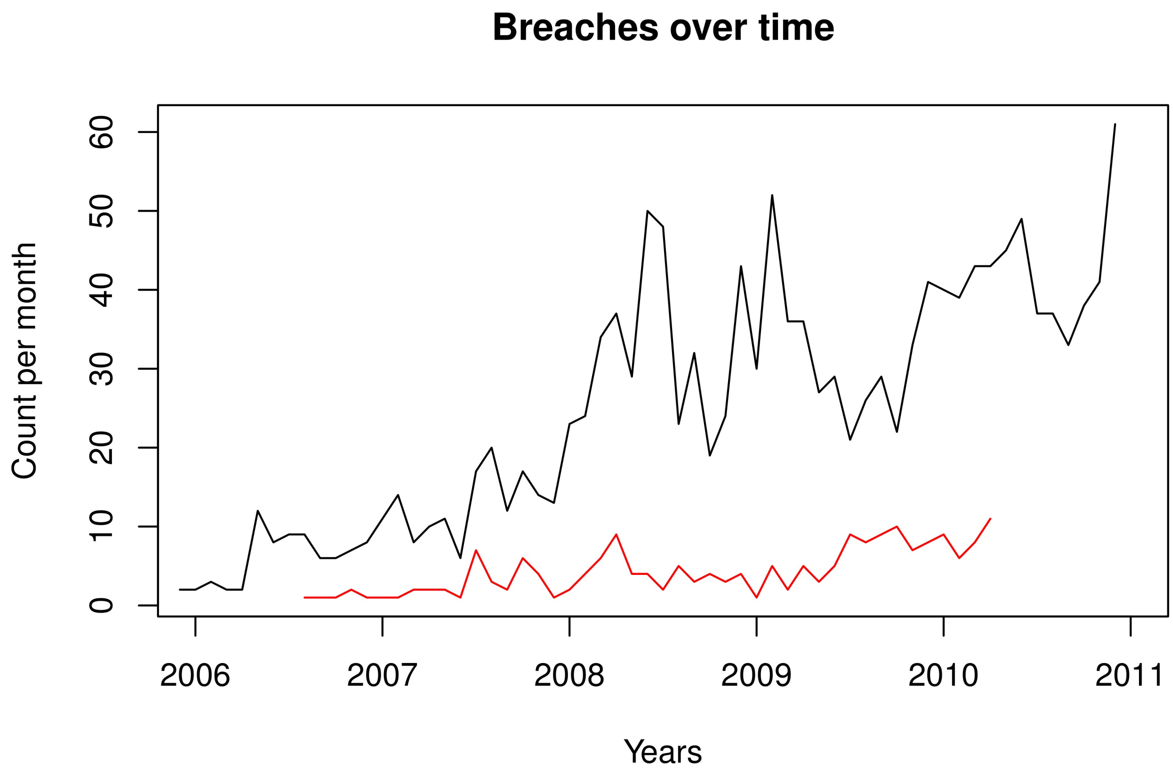
```r
names(control_freq)[1] <- "yearmonth"
control_freq$frequency[is.na(control_freq$frequency)]<-0
control_freq<- merge(datfull,control_freq, by="yearmonth", all=TRUE)

treatment_ts <- ts(treatment_freq$frequency, frequency = 12, start = c(2006,8))
control_ts <- ts(control_freq$frequency, frequency = 12, start = c(2005,12))

plot.ts(treatment_ts, main = "Breaches over time", xlim=c(2006,2011), xlab = "Years", ylab = "Count per
```



```r
plot.ts(control_ts, main = "Breaches over time", xlim=c(2006,2011), xlab = "Years", ylab = "Count per mo
```

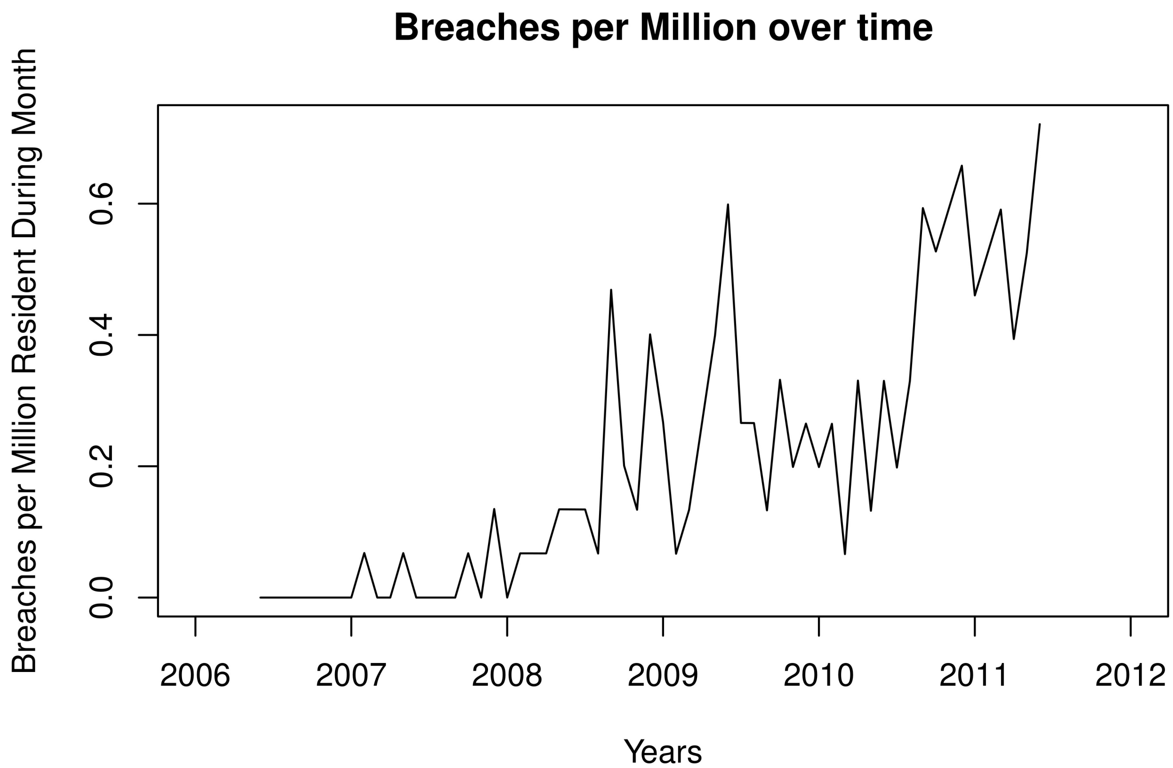



```r
ts.plot(control_ts, treatment_ts, main = "Breaches over time", xlim=c(2006,2011), gpars = list(col = c(
```

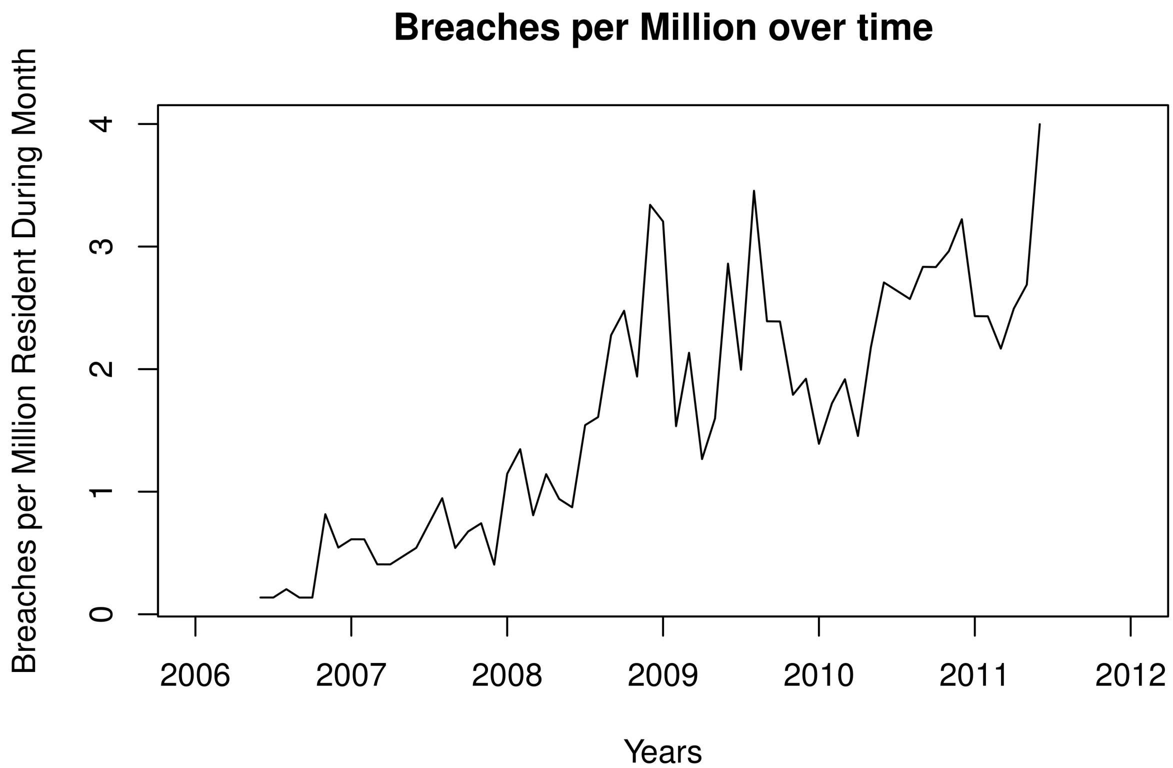



### Create charts with breaches per million residents

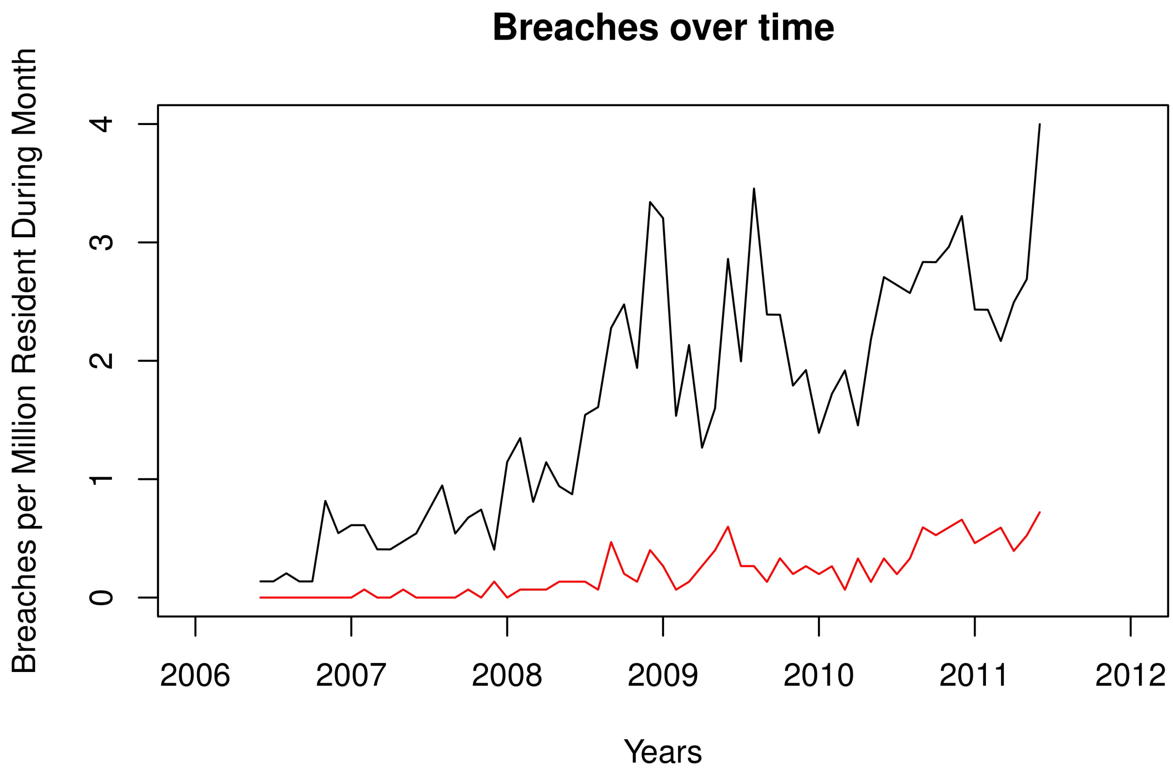
```
# Merge Treatment and Control Together
comb_ts <- merge(treatment_freq, control_freq, by="yearmonth", all=TRUE)

# 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

comb_ts$totpop <- as.numeric(as.character(comb_ts$totpop))
comb_ts$treatpermil <- comb_ts$frequency.x/(comb_ts$totpop/1000000)
class(comb_ts$frequency.y)<-"numeric"
comb_ts$controlpermil <- comb_ts$frequency.y/(comb_ts$totpop/1000000)

treatment_tsM <- ts(comb_ts$treatpermil, frequency = 12, start = c(2006,6))
control_tsM <- ts(comb_ts$controlpermil, frequency = 12, start = c(2006,6))

plot.ts(treatment_tsM, main = "Breaches per Million over time", xlim=c(2006,2012), xlab = "Years", ylab
```




```
plot.ts(control_tsM, main = "Breaches per Million over time", xlim=c(2006,2012), xlab = "Years", ylab =
```




```
ts.plot(control_tsM, treatment_tsM, main = "Breaches over time", xlim=c(2006,2012), gpars = list(col = c
```

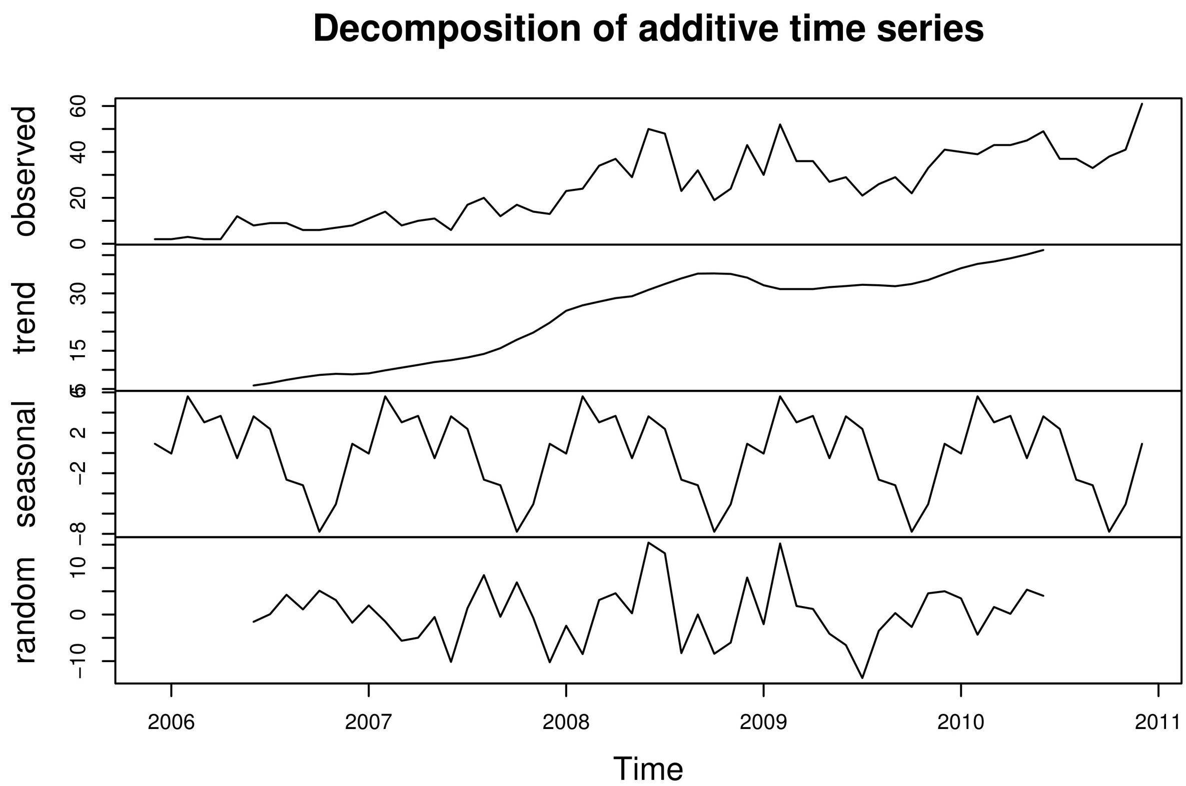



### Decompose the Control to Find Seasonal Patterns

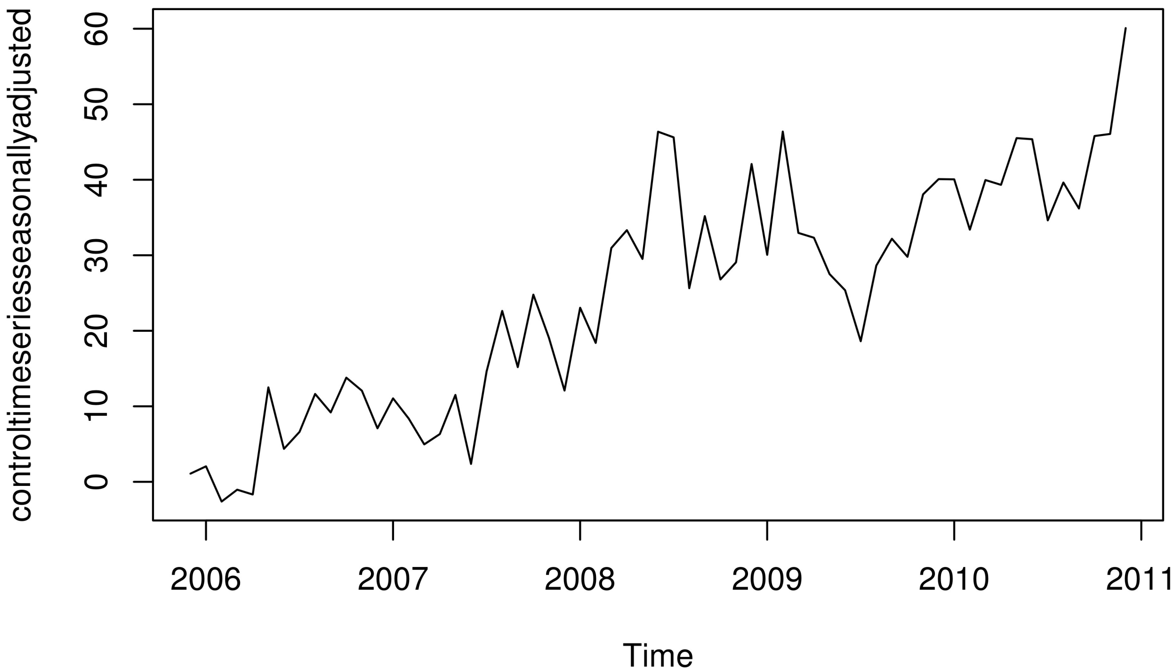
```
control_ts
```

```
##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 2005                                                2
## 2006   2   3   2   2  12   8   9   9   6   6   7   8
## 2007  11  14   8  10  11   6  17  20  12  17  14  13
## 2008  23  24  34  37  29  50  48  23  32  19  24  43
## 2009  30  52  36  36  27  29  21  26  29  22  33  41
## 2010  40  39  43  43  45  49  37  37  33  38  41  61
```

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




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




### Identifying and subsetting relevant dates

```
# Identifying relevant dates

# HIGH TECH Act regulations become effective on February 17, 2009
# Enforcement of HIGH TECH Act implement on May 27, 2009

treatment_start <- as.Date("02/17/2009", "%m/%d/%Y") # Legislation H.B. 4144 becomes effective
treatment_start<- format(as.Date(as.character(treatment_start), origin = "1970-01-01"), "%Y/%m")

treatment_end <- as.Date("05/27/2009", "%m/%d/%Y") # post 6 months after enforcement
treatment_end<- format(as.Date(as.character(treatment_end), origin = "1970-01-01"), "%Y/%m")

pretreat <- comb_ts[(which(comb_ts$yearmonth==treatment_start)-6):which(comb_ts$yearmonth==treatment_sta
pretreat$type <- "pretest"

posttreat <- comb_ts[which(comb_ts$yearmonth==treatment_end):(which(comb_ts$yearmonth==treatment_end)+6)
posttreat$type <- "posttest"

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

```
## [1] -0.07721643
```

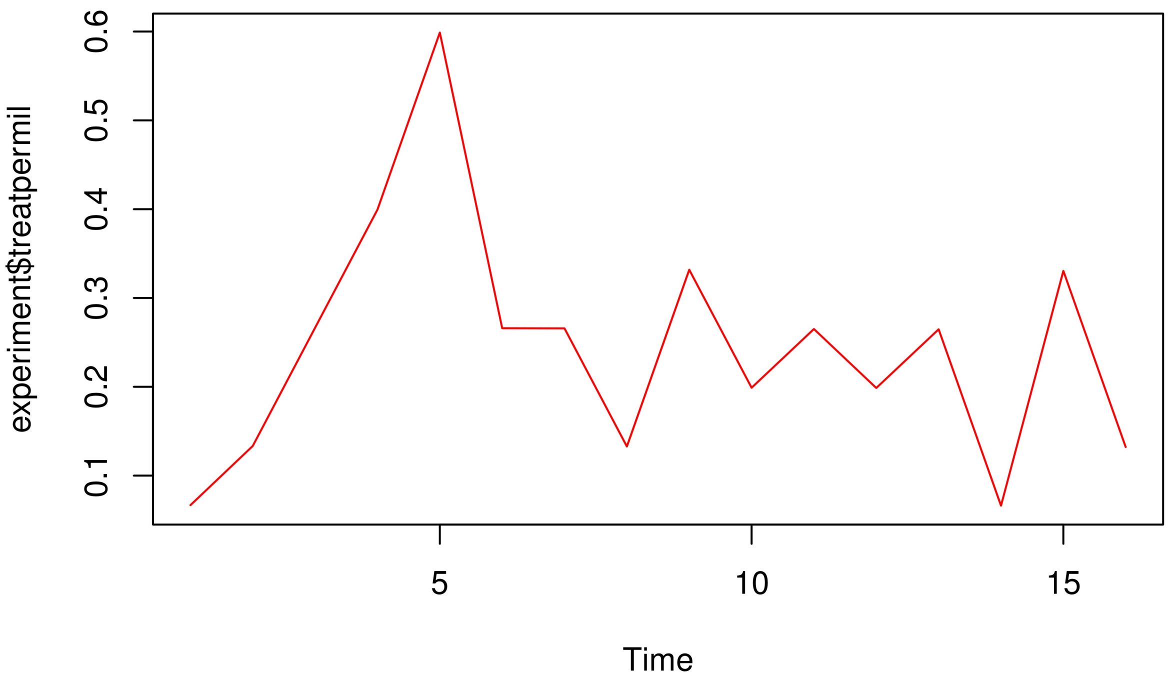
```
mean(posttreat$controlpermil) - mean(pretreat$controlpermil)
```

```
## [1] -0.3524432
```

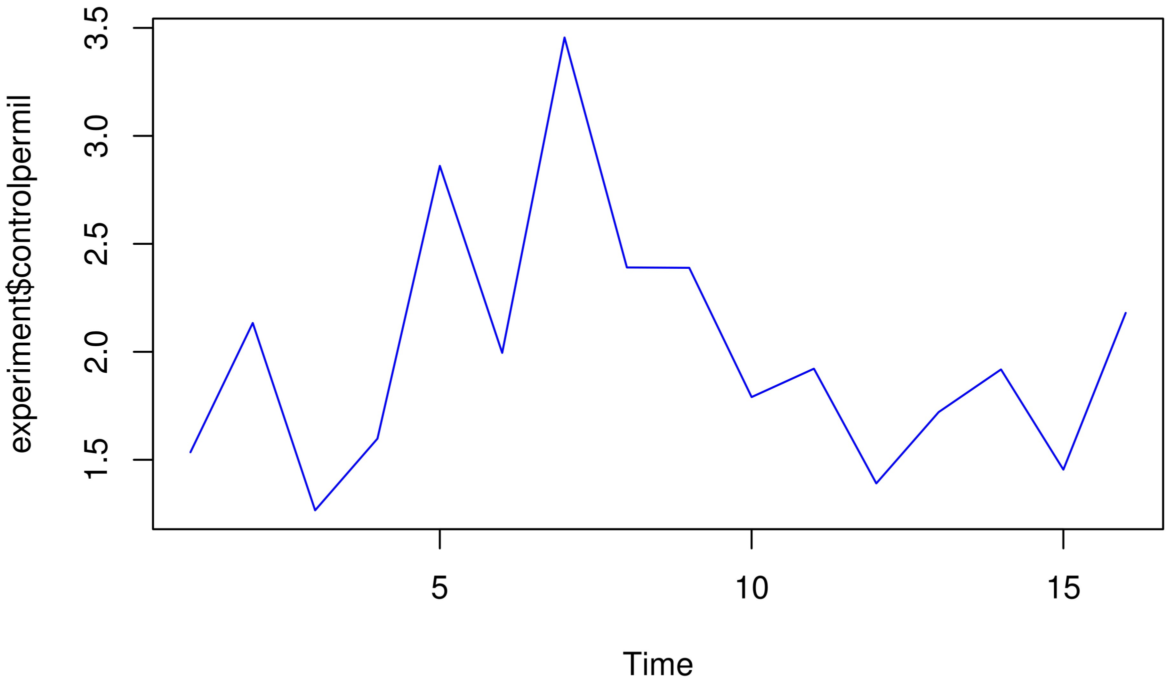
```r
treatment_range <- comb_ts[(which(comb_ts$yearmonth==treatment_start)+1):(which(comb_ts$yearmonth==trea
treatment_range$type <- "test"

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

```
##    yearmonth frequency.x frequency.y  Hawaii   Maine Massachusetts
## 33   2008/08           1          23 1333422 1330432       6473021
## 34   2008/09           2          32 1334630 1330356       6477075
## 35   2008/10           4          19 1335839 1330279       6481128
## 36   2008/11           6          24 1337048 1330203       6485182
## 37   2008/12           9          43 1338256 1330126       6489236
## 38   2009/01           4          30 1339465 1330050       6493290
## 39   2009/02           4          52 1340674 1329973       6497344
## 40   2009/03           2          36 1341882 1329896       6501398
## 41   2009/04           5          36 1343091 1329820       6505452
## 42   2009/05           3          27 1344300 1329743       6509505
## 43   2009/06           4          29 1345508 1329667       6513559
## 44   2009/07           3          21 1346717 1329590       6517613
## 45   2009/08           4          26 1348226 1329453       6520948
## 46   2009/09           1          29 1349736 1329317       6524283
## 47   2009/10           5          22 1351245 1329180       6527618
## 48   2009/11           2          33 1352754 1329044       6530953
##    New Hampshire South Carolina   totpop treatpermil controlpermil     type
## 33       1315922        4534069 14986866  0.06672509      1.534677  pretest
## 34       1315939        4539142 14997142  0.13335874      2.133740  pretest
## 35       1315955        4544215 15007416  0.26653489      1.266041  pretest
## 36       1315971        4549288 15017692  0.39952877      1.598115  pretest
## 37       1315988        4554361 15027967  0.59888340      2.861332  pretest
## 38       1316004        4559434 15038243  0.26598852      1.994914  pretest
## 39       1316020        4564507 15048518  0.26580691      3.455490  pretest
## 40       1316037        4569580 15058793  0.13281277      2.390630     test
## 41       1316053        4574653 15069069  0.33180550      2.389000     test
## 42       1316069        4579726 15079343  0.19894766      1.790529 posttest
## 43       1316086        4584799 15089619  0.26508290      1.921851 posttest
## 44       1316102        4589872 15099894  0.19867689      1.390738 posttest
## 45       1316143        4593816 15108586  0.26475012      1.720876 posttest
## 46       1316184        4597759 15117279  0.06614947      1.918335 posttest
## 47       1316225        4601703 15125971  0.33055729      1.454452 posttest
## 48       1316266        4605646 15134663  0.13214698      2.180425 posttest
```

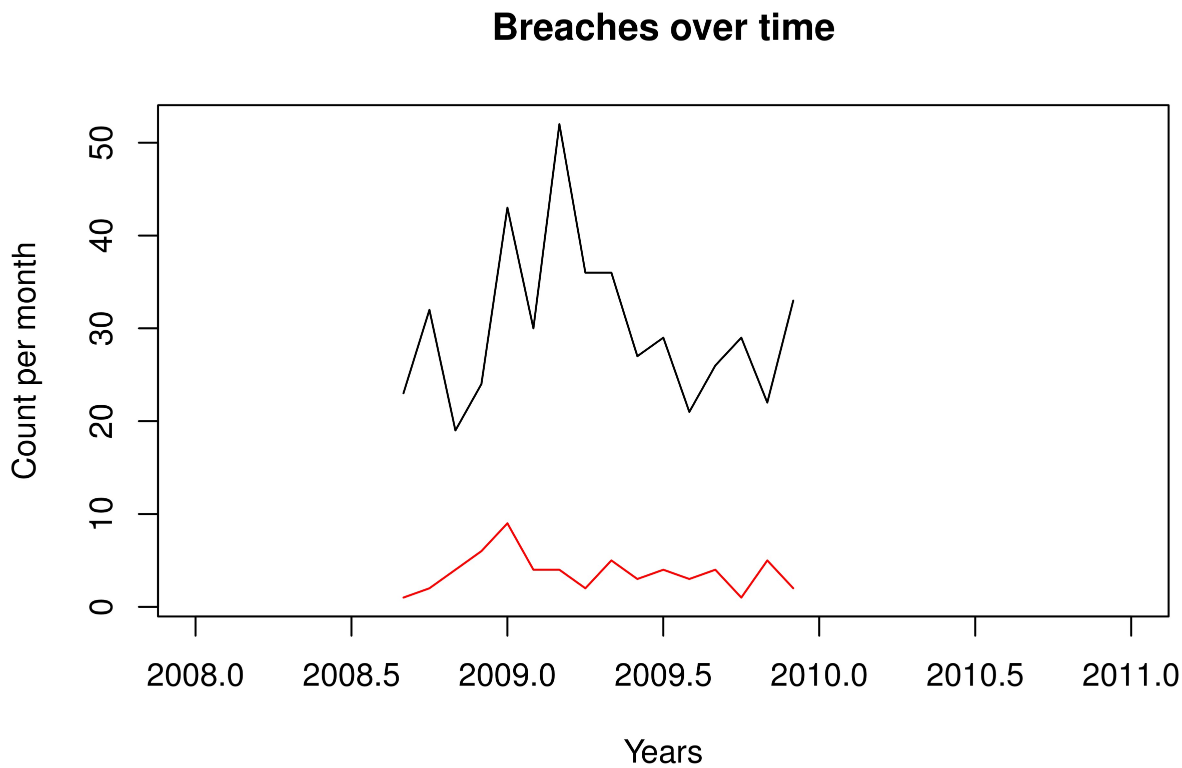
```r
ts.plot(experiment$treatpermil, col = "red")
```




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




```r
# Look at Raw Frequency Counts

treatment_ts <- ts(experiment$frequency.x, frequency = 12, start = c(2008,9))
control_ts <- ts(experiment$frequency.y, frequency = 12, start = c(2008,9))
ts.plot(control_ts, treatment_ts, main = "Breaches over time", xlim=c(2008,2011), gpars = list(col = c(
```




```
# Look at Treatment and Control per Million

treatment_tsM <- ts(experiment$treatpermil, frequency = 12, start = c(2008,6))
mean(treatment_tsM)
```

```
## [1] 0.2448597
```

```
sd(treatment_tsM)
```

```
## [1] 0.1346402
```

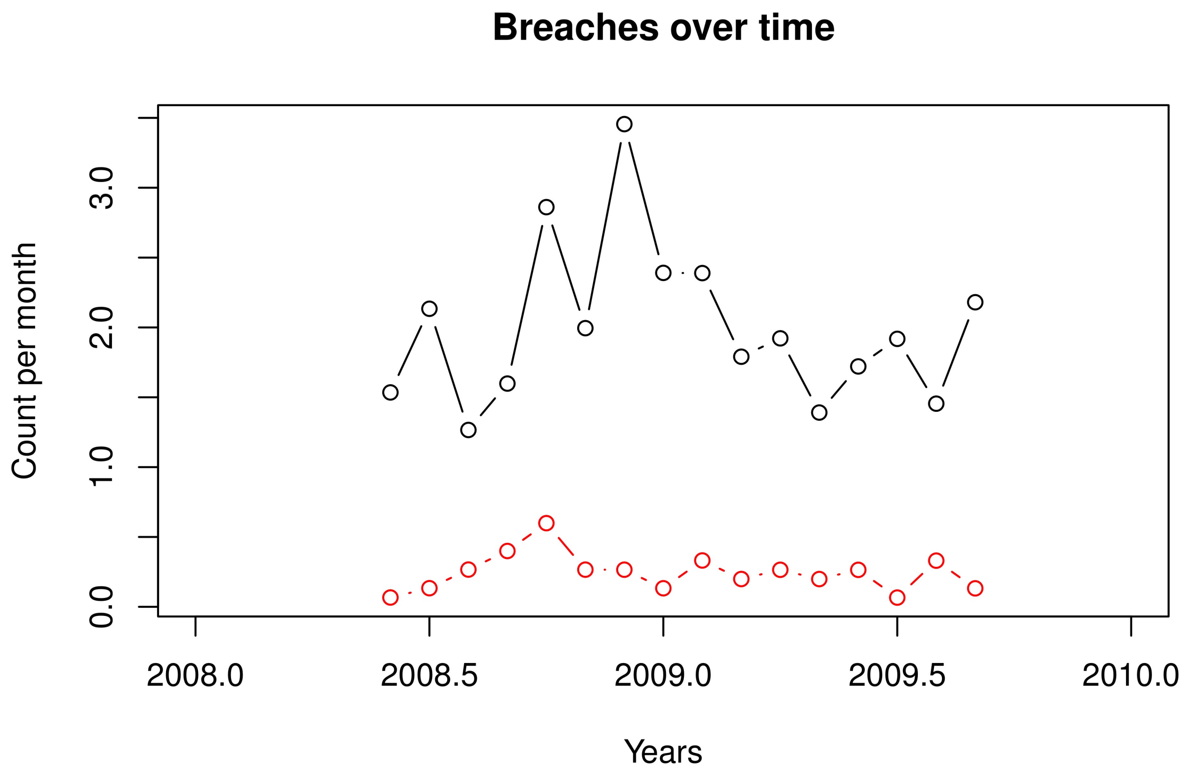
```
control_tsM <- ts(experiment$controlpermil, frequency = 12, start = c(2008,6))
mean(control_tsM)
```

```
## [1] 2.000071
```

```
sd(control_tsM)
```

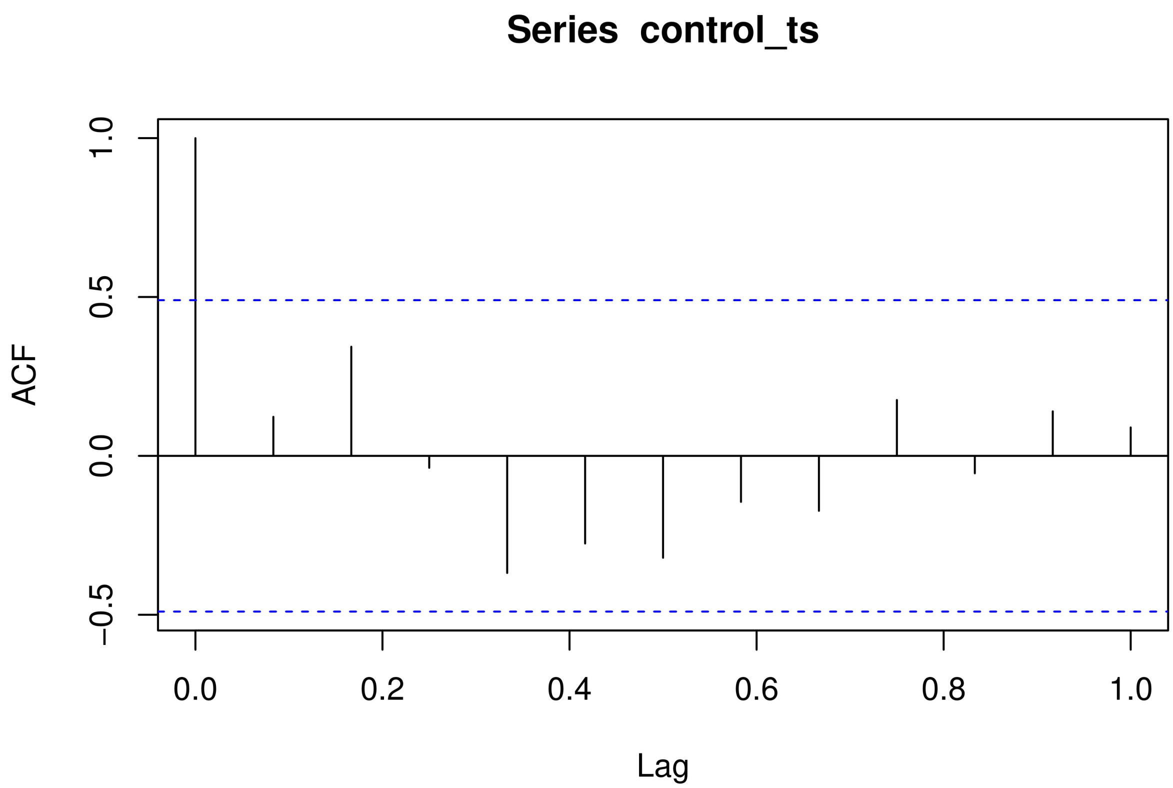
```
## [1] 0.573576
```

```
ts.plot(control_tsM, treatment_tsM, main = "Breaches over time", xlim=c(2008,2010),
                    gpars = list(col = c("black", "red")), type = "b", xlab = "Years", ylab = "Count pe
```

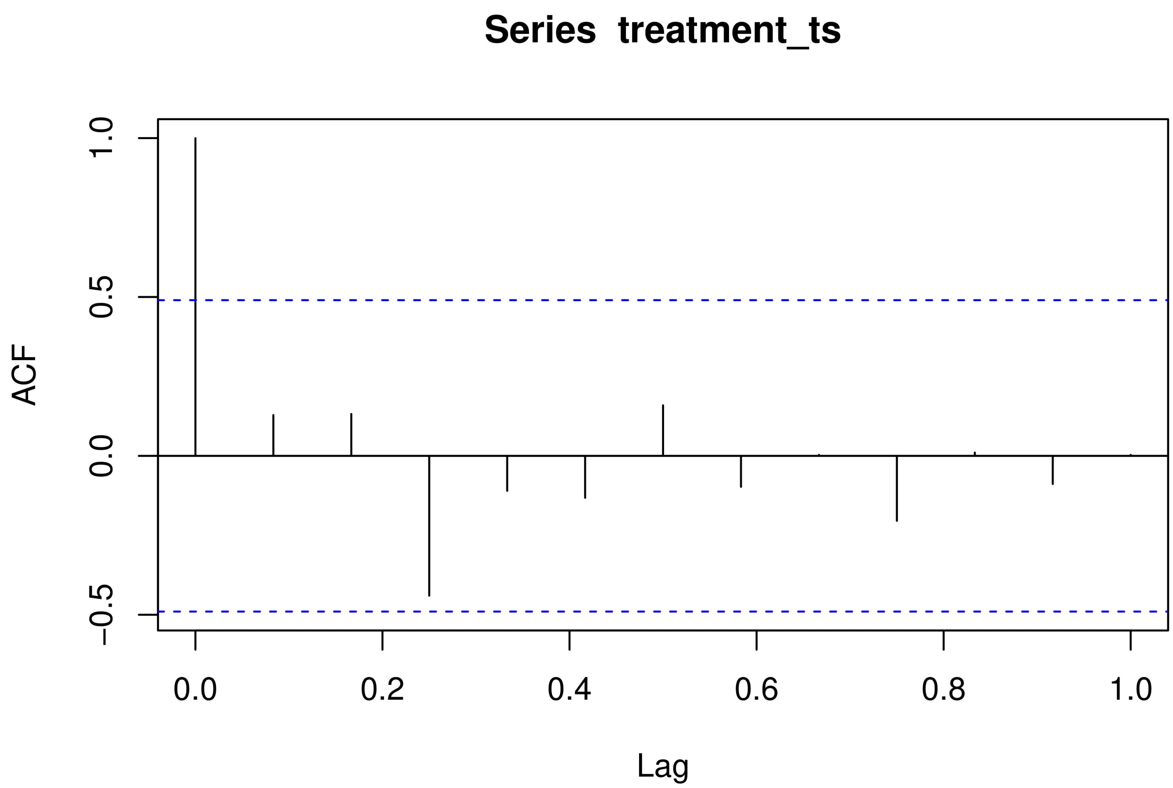



### Run Statistical Tests on Time Series for Stationarity

```
# 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)
```




```r
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","acttreatmentNH.png"))
plot(acftreatment)

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

plot(pacfcontrolMA)

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

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

```
##
## Autocorrelations of series 'X', by lag
##
## -0.7500 -0.6667 -0.5833 -0.5000 -0.4167 -0.3333 -0.2500 -0.1667 -0.0833  0.0000
##  -0.231   0.051  -0.190  -0.029  -0.118   0.052  -0.206  -0.020  -0.209   0.251
##  0.0833  0.1667  0.2500  0.3333  0.4167  0.5000  0.5833  0.6667  0.7500
##   0.235   0.623   0.291   0.085  -0.111  -0.363  -0.236  -0.235  -0.037
```

```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 = -1.834, Lag order = 2, p-value = 0.6356
## alternative hypothesis: stationary
```

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

```
## Warning in adf.test(treatment_ts): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  treatment_ts
## Dickey-Fuller = -4.727, Lag order = 2, p-value = 0.01
## 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 = 0.124, 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 = 0.12395, 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",]
quasiexp
```

```
##    yearmonth frequency.x frequency.y  Hawaii   Maine Massachusetts
## 33   2008/08           1          23 1333422 1330432       6473021
## 34   2008/09           2          32 1334630 1330356       6477075
## 35   2008/10           4          19 1335839 1330279       6481128
## 36   2008/11           6          24 1337048 1330203       6485182
## 37   2008/12           9          43 1338256 1330126       6489236
## 38   2009/01           4          30 1339465 1330050       6493290
## 39   2009/02           4          52 1340674 1329973       6497344
## 42   2009/05           3          27 1344300 1329743       6509505
```

```
## 43   2009/06          4          29 1345508 1329667       6513559
## 44   2009/07          3          21 1346717 1329590       6517613
## 45   2009/08          4          26 1348226 1329453       6520948
## 46   2009/09          1          29 1349736 1329317       6524283
## 47   2009/10          5          22 1351245 1329180       6527618
## 48   2009/11          2          33 1352754 1329044       6530953
##    New Hampshire South Carolina   totpop treatpermil controlpermil     type
## 33       1315922        4534069 14986866  0.06672509      1.534677  pretest
## 34       1315939        4539142 14997142  0.13335874      2.133740  pretest
## 35       1315955        4544215 15007416  0.26653489      1.266041  pretest
## 36       1315971        4549288 15017692  0.39952877      1.598115  pretest
## 37       1315988        4554361 15027967  0.59888340      2.861332  pretest
## 38       1316004        4559434 15038243  0.26598852      1.994914  pretest
## 39       1316020        4564507 15048518  0.26580691      3.455490  pretest
## 42       1316069        4579726 15079343  0.19894766      1.790529 posttest
## 43       1316086        4584799 15089619  0.26508290      1.921851 posttest
## 44       1316102        4589872 15099894  0.19867689      1.390738 posttest
## 45       1316143        4593816 15108586  0.26475012      1.720876 posttest
## 46       1316184        4597759 15117279  0.06614947      1.918335 posttest
## 47       1316225        4601703 15125971  0.33055729      1.454452 posttest
## 48       1316266        4605646 15134663  0.13214698      2.180425 posttest
```

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

```
##         V1                  V2 treat time z
## 1  2008/08 0.0667250911564833     1    1 0
## 2  2008/09  0.133358742619094     1    2 0
## 3  2008/10  0.266534891816153     1    3 0
## 4  2008/11  0.399528769134432     1    4 0
## 5  2008/12  0.598883401860012     1    5 0
## 6  2009/01  0.265988520068468     1    6 0
## 7  2009/02  0.265806905371014     1    7 1
## 8  2009/05   0.19894765972231     1    8 2
## 9  2009/06  0.265082902358237     1    9 3
## 10 2009/07  0.198676891374204     1   10 4
## 11 2009/08  0.264750122877151     1   11 5
## 12 2009/09 0.0661494704172623     1   12 6
## 13 2009/10  0.330557291164977     1   13 7
## 14 2009/11  0.132146979420685     1   14 8
```

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

```r
AppendITS$time <- as.numeric(as.character(AppendITS$time))
AppendITS$z <- as.numeric(as.character(AppendITS$z))
AppendITS
```

```
##    yearmonth incident_permil treat time z
## 1    2008/08      0.06672509     1    1 0
## 2    2008/09      0.13335874     1    2 0
## 3    2008/10      0.26653489     1    3 0
## 4    2008/11      0.39952877     1    4 0
## 5    2008/12      0.59888340     1    5 0
## 6    2009/01      0.26598852     1    6 0
## 7    2009/02      0.26580691     1    7 1
## 8    2009/05      0.19894766     1    8 2
## 9    2009/06      0.26508290     1    9 3
## 10   2009/07      0.19867689     1   10 4
## 11   2009/08      0.26475012     1   11 5
## 12   2009/09      0.06614947     1   12 6
## 13   2009/10      0.33055729     1   13 7
## 14   2009/11      0.13214698     1   14 8
## 15   2008/08      1.53467710     0    1 0
## 16   2008/09      2.13373988     0    2 0
## 17   2008/10      1.26604074     0    3 0
## 18   2008/11      1.59811508     0    4 0
## 19   2008/12      2.86133181     0    5 0
## 20   2009/01      1.99491390     0    6 0
## 21   2009/02      3.45548977     0    7 1
## 22   2009/05      1.79052894     0    8 2
## 23   2009/06      1.92185104     0    9 3
## 24   2009/07      1.39073824     0   10 4
## 25   2009/08      1.72087580     0   11 5
## 26   2009/09      1.91833464     0   12 6
## 27   2009/10      1.45445208     0   13 7
## 28   2009/11      2.18042516     0   14 8
```

```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
## -0.62698 -0.20759 -0.07450  0.09745  1.29547
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.56365    0.26753   5.845 5.00e-06 ***
```

```
## time         0.10979    0.05772   1.902   0.0692 .
## treat       -1.69774    0.15604 -10.880 9.23e-11 ***
## z           -0.17215    0.08248  -2.087   0.0477 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4128 on 24 degrees of freedom
## Multiple R-squared:  0.8365, Adjusted R-squared:  0.816
## F-statistic: 40.92 on 3 and 24 DF,  p-value: 1.357e-09
```

```
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
## -0.62816 -0.12120  0.00696  0.11867  1.16329
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.18950    0.37672   3.158  0.00495 **
## time         0.23490    0.09188   2.557  0.01881 *
## treat       -1.12526    0.53276  -2.112  0.04745 *
## z           -0.76666    0.35564  -2.156  0.04347 *
## time:treat  -0.17487    0.12994  -1.346  0.19343
## time:z       0.03215    0.02169   1.482  0.15393
## treat:z      0.57368    0.50295   1.141  0.26751
## time:treat:z -0.02454    0.03068  -0.800  0.43327
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4156 on 20 degrees of freedom
## Multiple R-squared:  0.8619, Adjusted R-squared:  0.8136
## F-statistic: 17.83 on 7 and 20 DF,  p-value: 2.557e-07
```

```
AppendITS
```

```
##    yearmonth incident_permil treat time z
## 1    2008/08      0.06672509     1    1 0
## 2    2008/09      0.13335874     1    2 0
## 3    2008/10      0.26653489     1    3 0
## 4    2008/11      0.39952877     1    4 0
## 5    2008/12      0.59888340     1    5 0
## 6    2009/01      0.26598852     1    6 0
## 7    2009/02      0.26580691     1    7 1
## 8    2009/05      0.19894766     1    8 2
## 9    2009/06      0.26508290     1    9 3
## 10   2009/07      0.19867689     1   10 4
## 11   2009/08      0.26475012     1   11 5
## 12   2009/09      0.06614947     1   12 6
## 13   2009/10      0.33055729     1   13 7
```

```
## 14   2009/11      0.13214698     1   14 8
## 15   2008/08      1.53467710     0    1 0
## 16   2008/09      2.13373988     0    2 0
## 17   2008/10      1.26604074     0    3 0
## 18   2008/11      1.59811508     0    4 0
## 19   2008/12      2.86133181     0    5 0
## 20   2009/01      1.99491390     0    6 0
## 21   2009/02      3.45548977     0    7 1
## 22   2009/05      1.79052894     0    8 2
## 23   2009/06      1.92185104     0    9 3
## 24   2009/07      1.39073824     0   10 4
## 25   2009/08      1.72087580     0   11 5
## 26   2009/09      1.91833464     0   12 6
## 27   2009/10      1.45445208     0   13 7
## 28   2009/11      2.18042516     0   14 8
```

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