Day 10 - Temporal Models

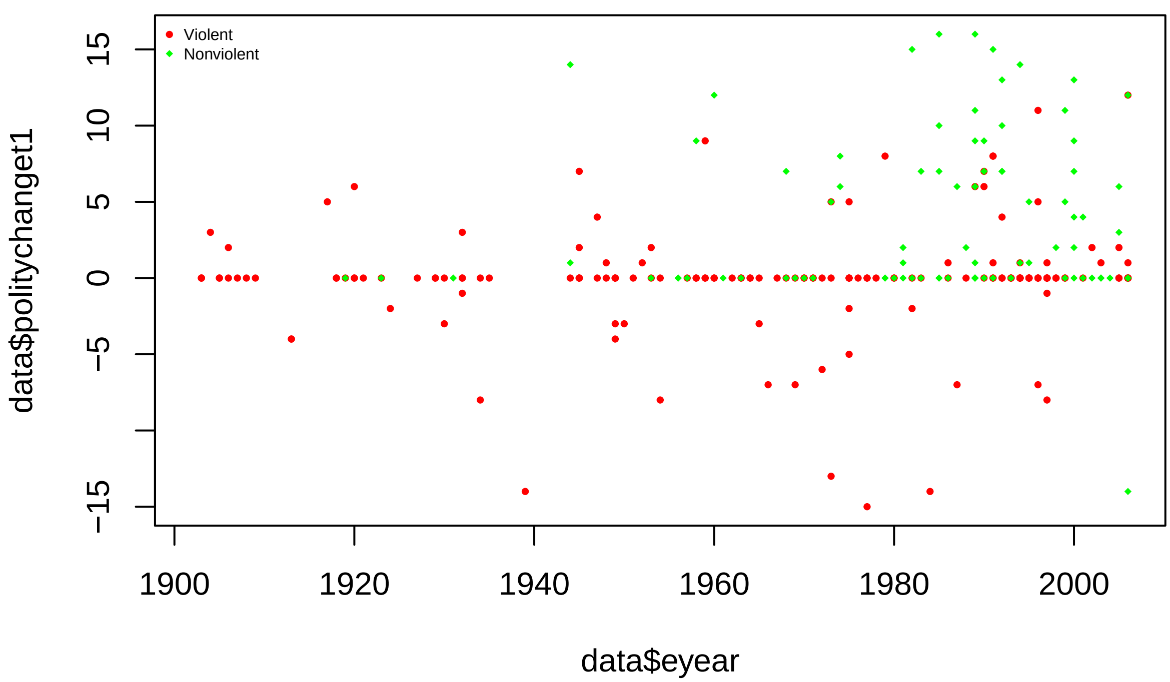
David Carlson

December 9, 2020

Temporal data analysis

- Repeated measurements over time (but either not TSCS, or no reason to theoretically account heterogeneity)
- The reading goes over standard temporal analyses, but what if there are not equally temporally spaced observations?
- We will first analyze identity link, modeling the change in polity following a violent or non-violent protest / uprising
- Let's start by looking at the data

```r
plot(data$politychanget1 ~ data$eyear, type = 'n')
points(data$eyear[as.logical(data$viol)],
       data$politychanget1[as.logical(data$viol)],
       pch = 16, col = 'red', cex = .5)
points(data$eyear[as.logical(data$nonviol)],
       data$politychanget1[as.logical(data$nonviol)],
       pch = 18, col = 'green', cex = .5)
legend('topleft',
       legend = c('Violent', 'Nonviolent'),
       pch = c(16, 18),
       col = c('red', 'green'),
       bty = 'n',
       cex = .5)
```



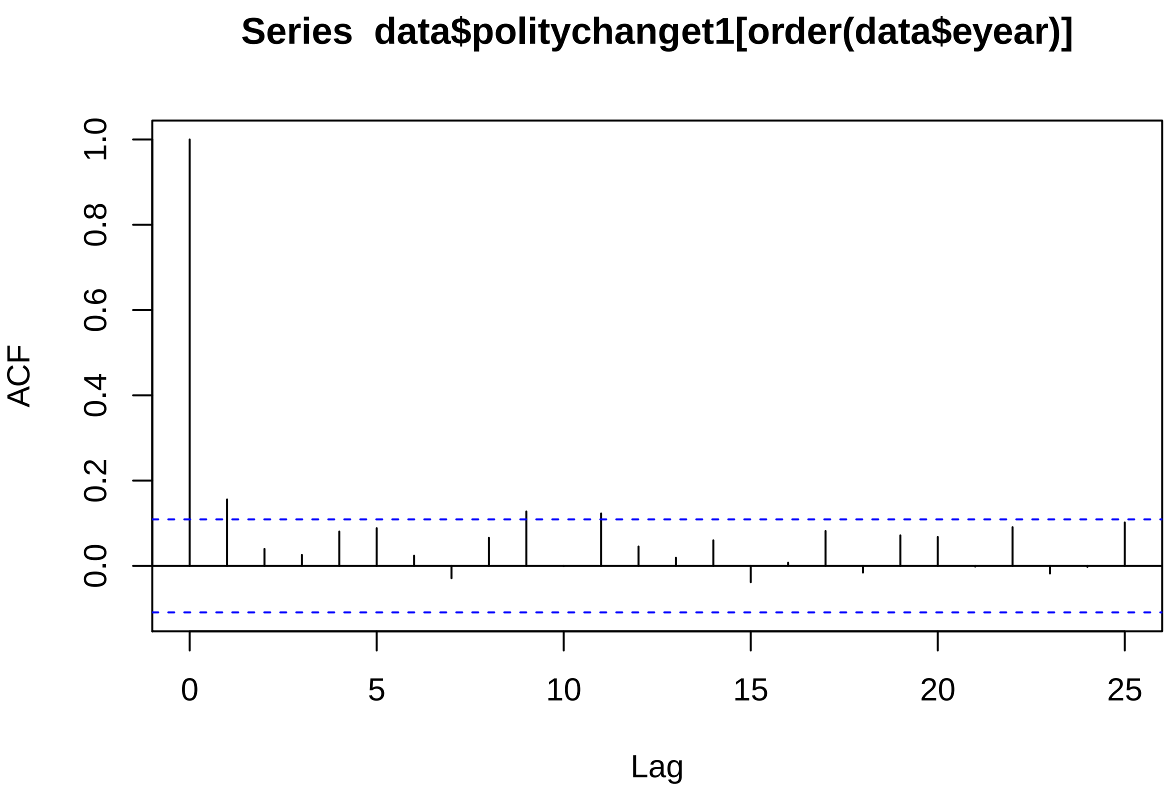
- Looking at the above plot, it seems that non-violent movements are more associated with increases in polity scores

- But what issues may arise?

```
simpleMod = lm(politychanget1 ~ nonviol, data = data)
summary(simpleMod)
```

```
##
## Call:
## lm(formula = politychanget1 ~ nonviol, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17.8295  -2.8295   0.0643   1.0643  12.1705
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.06433    0.32629  -0.197    0.844
## nonviol      3.89387    0.55977   6.956 2.89e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.267 on 257 degrees of freedom
##   (64 observations deleted due to missingness)
## Multiple R-squared:  0.1585, Adjusted R-squared:  0.1552
## F-statistic: 48.39 on 1 and 257 DF,  p-value: 2.89e-11
```

```
#let's add some controls
acf(data$politychanget1[order(data$eyear)],
    na.action = na.pass) #not really an issue, but assumes evenly spaced observations
```



```r
library(car)
```

```
## Loading required package: carData
```

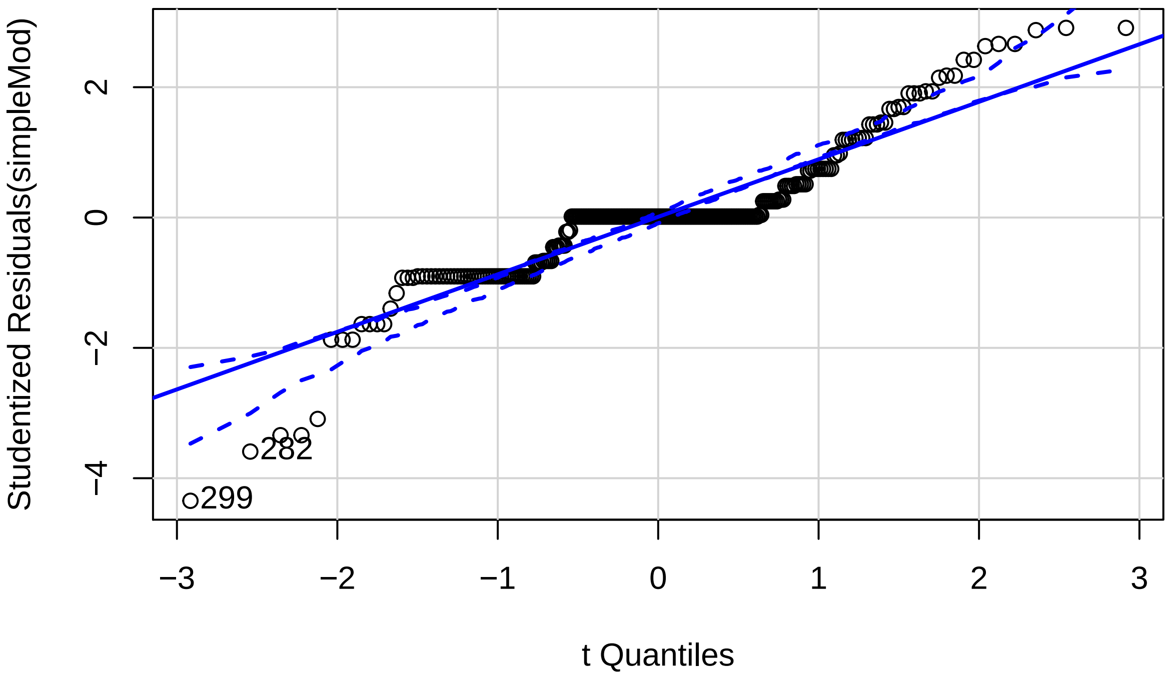
```r
qqPlot(simpleMod) #discrete makes it off, but also fat tails
```

```
## [1] 282 299
```

```r
#check for stationarity
#Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test. Here we will test the null hypothesis of trend stationa
library(tseries)
```

```
## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo
```

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




```r
kpss.test(data$politychanget1[order(data$eyear)],
          null = 'Trend') #low enough to be concerning, and again assumes even spacing
```

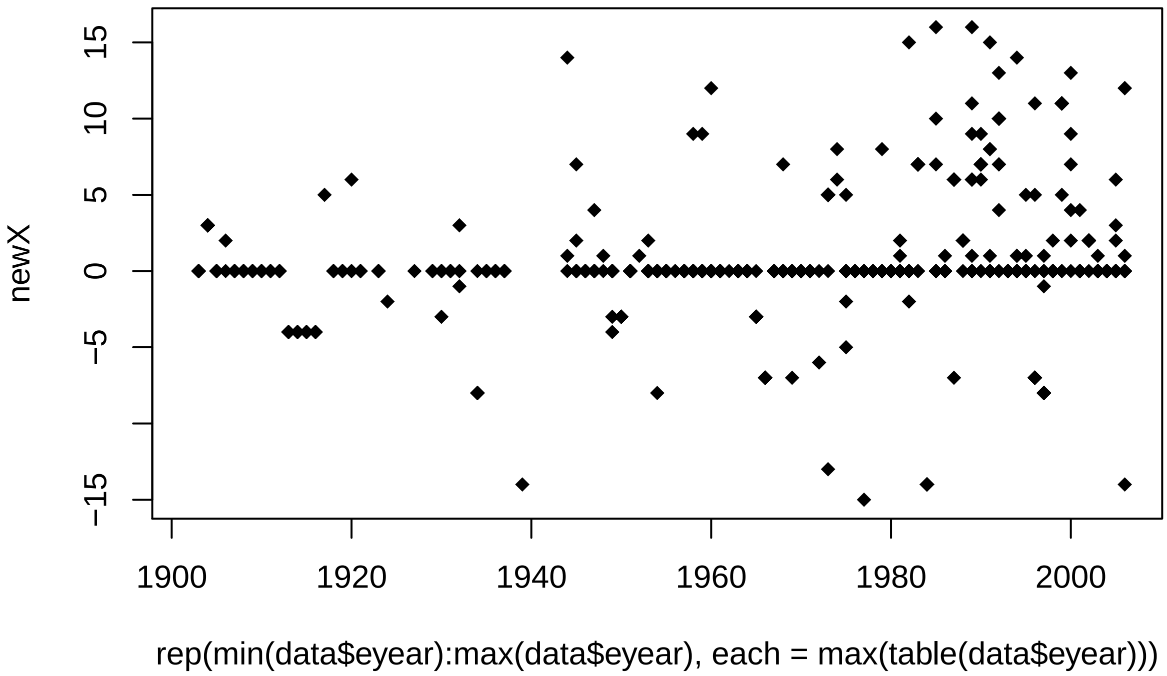
```
##
##  KPSS Test for Trend Stationarity
##
## data:  data$politychanget1[order(data$eyear)]
## KPSS Trend = 0.12267, Truncation lag parameter = 5, p-value =
## 0.0932
```

```r
#we could first difference to alleviate the trend, but again, these are not evenly spaced, and the expl
#devtools::install_github("andreas50/uts", build_vignettes=TRUE)
library(uts)
```

```
## Loading required package: lubridate
```

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

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

##
## Attaching package: 'uts'

## The following object is masked from 'package:stats':
##
##     sd

## The following objects are masked from 'package:base':
##
##     which, which.max, which.min
```

```
# uts(data$politychanget1[order(data$eyear)],
#     as.POSIXct(as.character(data$eyear[order(data$eyear)]),
#                format = '%Y')) #fails, because multiple observations per year
#let's hack it by making them a minute apart
times = c()
for(i in 1:length(unique(data$eyear))) times = c(times, 1:table(data$eyear)[i])
tser = uts(data$politychanget1[order(data$eyear)],
    as.POSIXct(paste(as.character(data$eyear[order(data$eyear)]),
                 1,
                 times,
                 sep = ':'),
               format = '%Y:%H:%M'))
#now linearly interpolate
newX = sample_values(tser,
              as.POSIXct(paste(as.character(rep(min(data$eyear):max(data$eyear), each = max(table(data$e
                 1,
                 1:max(table(data$eyear)),
                 sep = ':'),
               format = '%Y:%H:%M'))
plot(newX ~ rep(min(data$eyear):max(data$eyear),
                each = max(table(data$eyear))),
     pch = 18)
```




```
kpss.test(na.omit(newX),
          null = 'Trend') #not much change
```

```
##
##  KPSS Test for Trend Stationarity
##
## data:  na.omit(newX)
## KPSS Trend = 0.12151, Truncation lag parameter = 7, p-value =
## 0.09536
```

```
#what about heteroskedasdicity?
#let's do the same for violent or not
tser = uts(data$nonviol[order(data$eyear)],
    as.POSIXct(paste(as.character(data$eyear[order(data$eyear)]),
                 1,
                 times,
                 sep = ':'),
               format = '%Y:%H:%M'))
#now linearly interpolate
newV = sample_values(tser,
          as.POSIXct(paste(as.character(rep(min(data$eyear):max(data$eyear), each = max(table(data$
                 1,
                 1:max(table(data$eyear)),
                 sep = ':'),
               format = '%Y:%H:%M'))
interMod = lm(newX ~ newV)
summary(simpleMod)
```

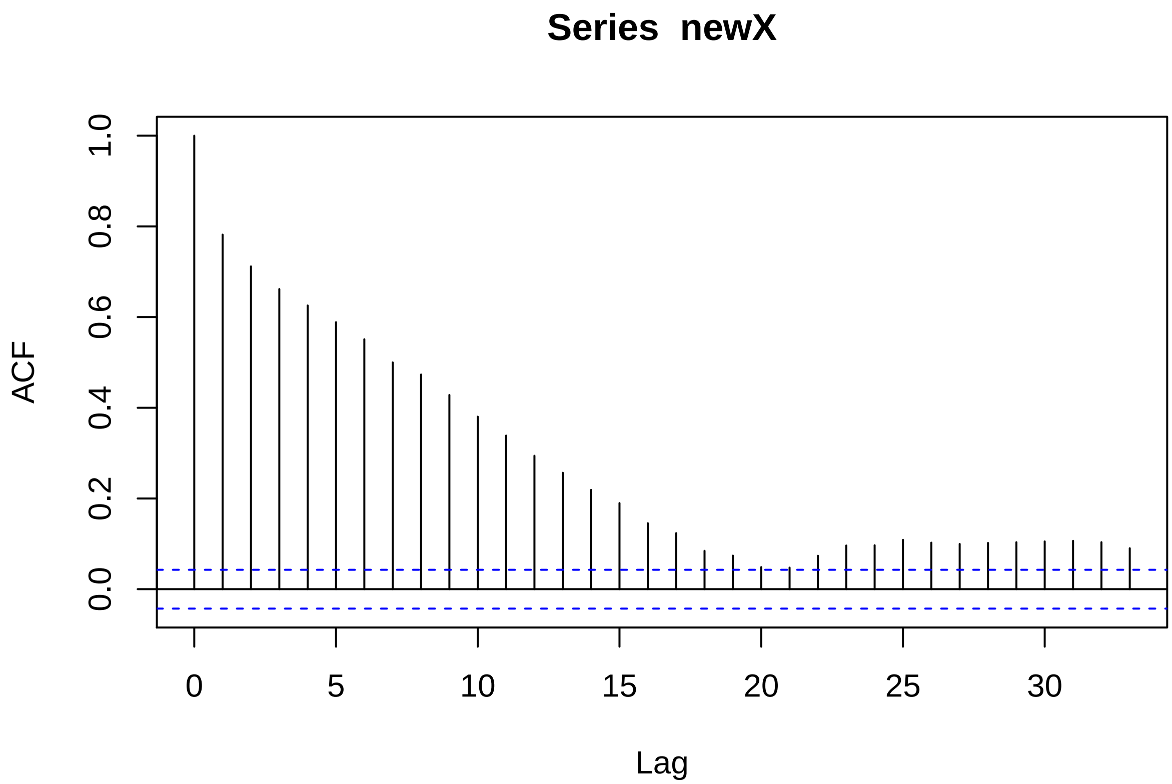
```
##
## Call:
## lm(formula = politychanget1 ~ nonviol, data = data)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -17.8295  -2.8295   0.0643   1.0643  12.1705
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.06433    0.32629  -0.197    0.844
## nonviol      3.89387    0.55977   6.956 2.89e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.267 on 257 degrees of freedom
##   (64 observations deleted due to missingness)
## Multiple R-squared:  0.1585, Adjusted R-squared:  0.1552
## F-statistic: 48.39 on 1 and 257 DF,  p-value: 2.89e-11
```

```
summary(interMod) #coefficient does not change much (why?), but SEs decrease (why?)
```

```
##
## Call:
## lm(formula = newX ~ newV)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.5636  -2.5636   0.9126   0.9126  13.4364
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.91256    0.09518  -9.588   <2e-16 ***
## newV         3.47615    0.18887  18.405   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.267 on 1577 degrees of freedom
##   (521 observations deleted due to missingness)
## Multiple R-squared:  0.1768, Adjusted R-squared:  0.1763
## F-statistic: 338.8 on 1 and 1577 DF,  p-value: < 2.2e-16
```

```
acf(newX, na.action = na.pass) #now we clearly see there is a problem
```




```r
#what do we do?
#the explanatory is binary, so we cannot first difference or lag or anything typical
#we can adjust the standard errors (Newey-West)
library(sandwich)
library(lmtest)
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

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

```r
coeftest(interMod,
         vcov. = NeweyWest(interMod)) #we see an increase in the SE
```

```
##
## t test of coefficients:
##
##              Estimate Std. Error t value  Pr(>|t|)
## (Intercept) -0.91256    0.30681 -2.9744   0.00298 **
## newV         3.47615    0.68401  5.0820 4.179e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
data2 = na.omit(data[, c('politychanget1',
                         'viol',
                         'eyear')])
simpleMod2 = lm(politychanget1 ~ viol, data2)
coeftest(simpleMod2,
```

```
        vcov. = NeweyWest(simpleMod2,
                          order.by = ~data2$eyear))
```

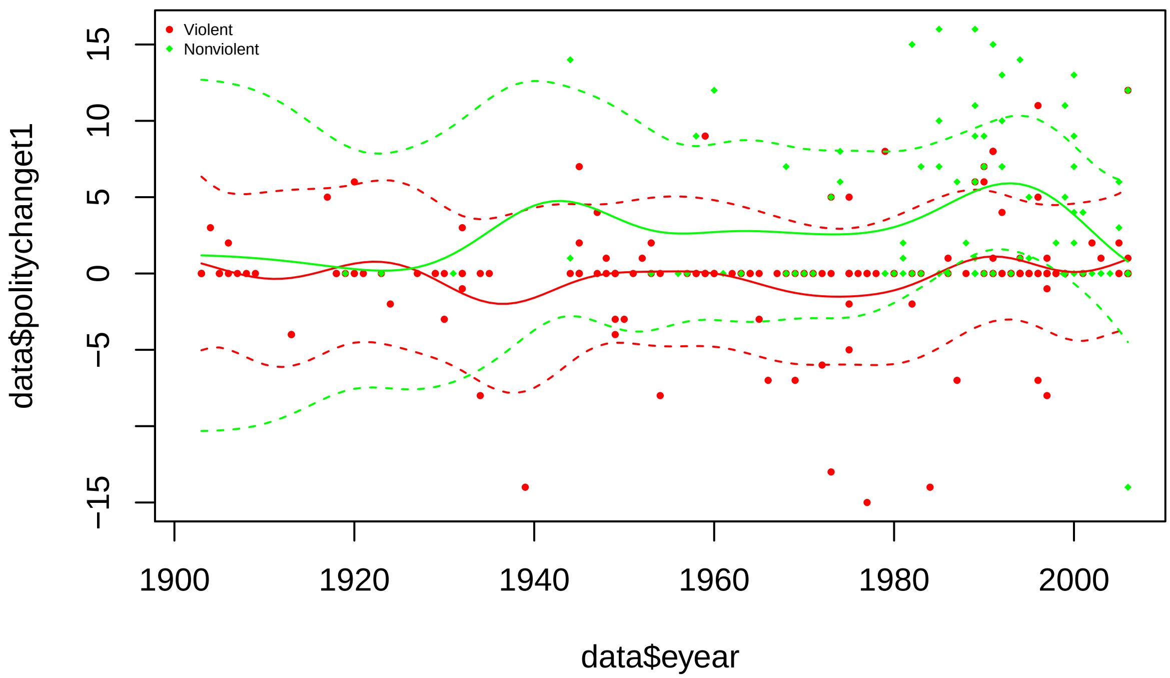
```
##
## t test of coefficients:
##
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  3.82955    0.58548  6.5409 3.285e-10 ***
## viol        -3.89387    0.67122 -5.8012 1.929e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#the above is improper, because again assumes evenly spaced observations
#Newey-West also makes strong parametric assumptions
#this is when we can turn to GPs (more details next week)
library(kernlab)
modGP = gausspr(politychanget1 ~ viol + eyear,
        data = data2,
        kernel = 'rbfdot',
        type = 'regression',
        variance.model = T) #we use dot product for the non-stationarity
```

```
## Using automatic sigma estimation (sigest) for RBF or laplace kernel
```

```
#when we code our own model, we can specify different inputs to mean and kernel, and make inferences on
#for now we just plot to make inferences
xtestViol = data.frame('viol' = 1,
                       'eyear' = min(data2$eyear):max(data2$eyear))
xtestNonViol = data.frame('viol' = 0,
                          'eyear' = min(data2$eyear):max(data2$eyear))
ysViol = predict(modGP, xtestViol)
ysViolLower = ysViol - 1.96*predict(modGP,
                                    xtestViol,
                                    type = 'sdeviation')
ysViolUpper = ysViol + 1.96*predict(modGP,
                                    xtestViol,
                                    type = 'sdeviation')
ysNonViol = predict(modGP, xtestNonViol)
ysNonViolLower = ysNonViol - 1.96*predict(modGP,
                                    xtestNonViol,
                                    type = 'sdeviation')
ysNonViolUpper = ysNonViol + 1.96*predict(modGP,
                                    xtestNonViol,
                                    type = 'sdeviation')

plot(data$politychanget1 ~ data$eyear, type = 'n')
points(data$eyear[as.logical(data$viol)],
       data$politychanget1[as.logical(data$viol)],
       pch = 16, col = 'red', cex = .5)
points(data$eyear[as.logical(data$nonviol)],
       data$politychanget1[as.logical(data$nonviol)],
       pch = 18, col = 'green', cex = .5)
lines(min(data2$eyear):max(data2$eyear),
      ysViol, col = 'red')
lines(min(data2$eyear):max(data2$eyear),
```

```r
       ysViolUpper, col = 'red', lty = 2)
lines(min(data2$eyear):max(data2$eyear),
       ysViolLower, col = 'red', lty = 2)
lines(min(data2$eyear):max(data2$eyear),
       ysNonViol, col = 'green')
lines(min(data2$eyear):max(data2$eyear),
       ysNonViolUpper, col = 'green', lty = 2)
lines(min(data2$eyear):max(data2$eyear),
       ysNonViolLower, col = 'green', lty = 2)

legend('topleft',
        legend = c('Violent', 'Nonviolent'),
        pch = c(16, 18),
        col = c('red', 'green'),
        bty = 'n',
        cex = .5)
```




```r
#we see that once temporal trends are accounted for, we cannot reject the null (but of course we are no
```

Estimation of Growth Rates

- There are many types of growth models
- They generally fall into two categories: Population dynamics in demography, and economic growth
- We will model the growth rate of protest movements, violent and non-violent, over time, and compare the growth rates

```r
library(growthrates)
```

```
## Loading required package: lattice

## Loading required package: deSolve
```

```r
dataViol = data[as.logical(data$viol), ]
dataNonViol = data[!as.logical(data$viol), ]
#linear growth model
growthLinViol = fit_easylinear(as.numeric(names(table(dataViol$eyear))),
                               as.numeric(table(dataViol$eyear)))
growthLinNonViol = fit_easylinear(as.numeric(names(table(dataNonViol$eyear))),
                               as.numeric(table(dataNonViol$eyear)))
summary(growthLinViol)
```

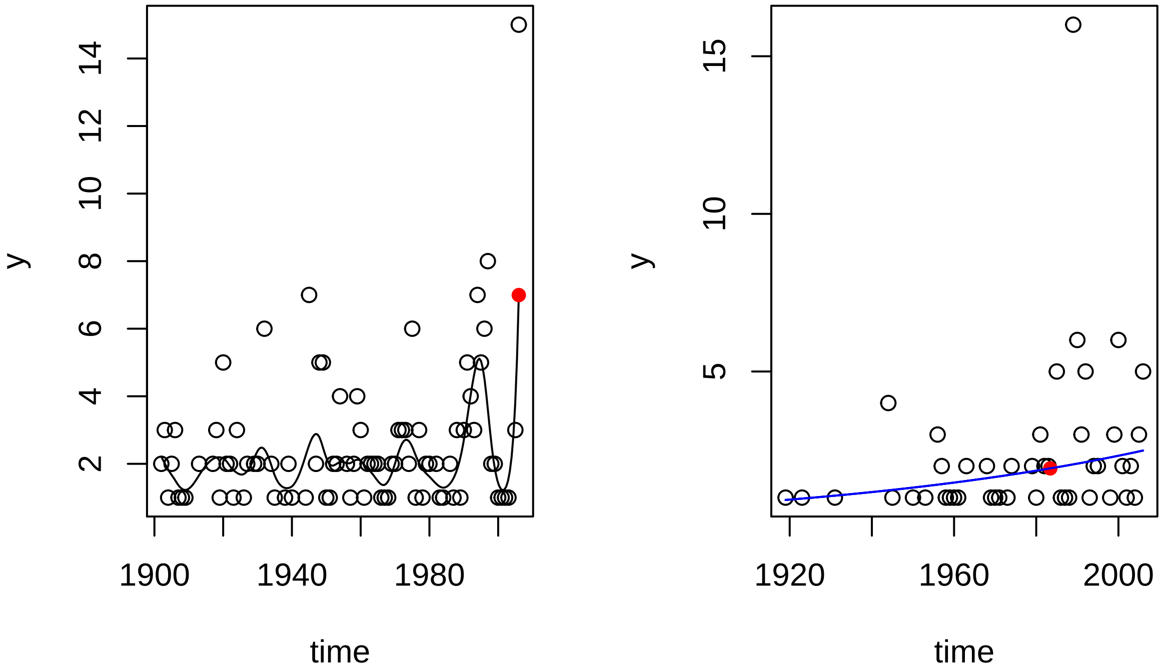
```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##       1       2       3       4       5
##  0.1386 -0.2079  0.1386 -0.2079  0.1386
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -675.95713  135.30237  -4.996   0.0154 *
## x              0.34657    0.06931   5.000   0.0154 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2192 on 3 degrees of freedom
## Multiple R-squared:  0.8929, Adjusted R-squared:  0.8571
## F-statistic:    25 on 1 and 3 DF,  p-value: 0.01539
```

```r
summary(growthLinNonViol) #growing much faster
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##       1       2       3       4       5
##  0.3584 -0.2773 -0.9129  1.2241 -0.3923
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1262.6813   595.4036  -2.121    0.124
## x               0.6356     0.2995   2.122    0.124
##
## Residual standard error: 0.9471 on 3 degrees of freedom
## Multiple R-squared:  0.6002, Adjusted R-squared:  0.4669
## F-statistic: 4.504 on 1 and 3 DF,  p-value: 0.1239
```

```r
#Nonparametric smoothing splines
#Smoothing splines are a quick method to estimate maximum growth. The method is called nonparametric, b
growthNPViol = fit_spline(as.numeric(names(table(dataViol$eyear))),
                               as.numeric(table(dataViol$eyear)))
growthNPNonViol = fit_spline(as.numeric(names(table(dataNonViol$eyear))),
                               as.numeric(table(dataNonViol$eyear)))
par(mfrow = c(1, 2))
plot(growthNPViol)
```

```r
plot(growthNPNonViol)
```




```r
#we see that non-violent growth is estimated to be consistent, while violent growth is estimated to be
```