Microeconometrics (Causal Inference)
Week 5 - Differences-in-differences I

Joshua D. Merfeld
KDI School of Public Policy and Management

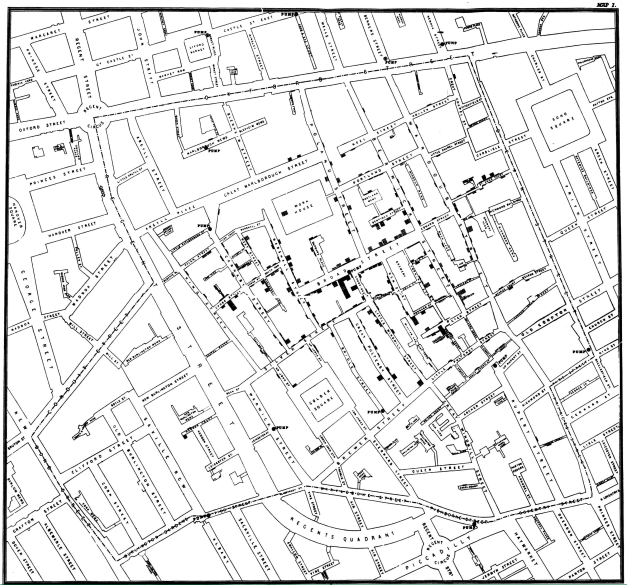
2023-09-25

What are we doing today?

- Canonical differences-in-differences
  - Inference
  - Wild cluster bootstrap
- Fixed effects vs. random effects
- Bias in two-way fixed effects

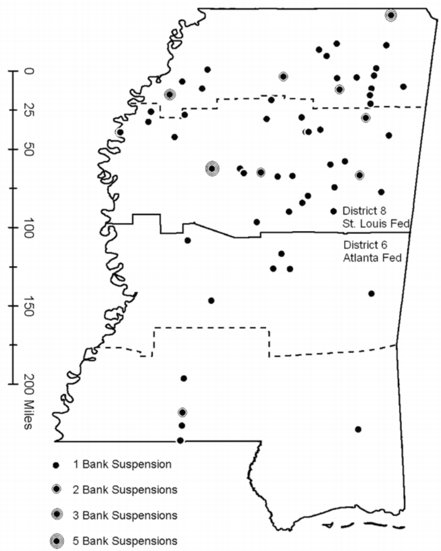
Differences-in-differences... in the 1800s?

Differences-in-differences... in the 1800s? (from Cunningham's CI)



- More commonly referred to as “DID” or “diff-in-diff”
  - Classic reference: Card and Krueger (1994)
- Most common method, likely because data requirements are least stringent
- Example in *Mostly Harmless*: offering credit to banks during the Great Depression (Richardson and Troost, 2009)
  - Set up: Two different federal reserve banks lent to neighborhood banks in Mississippi
  - Atlanta fed favored lending to banks in trouble
  - St. Louis fed favored the exact opposite

Richardson and Troost (2009) - Mississippi dividing line



Did the policy of extra lending save banks?

- Basic idea: compare what happened to Atlanta fed banks (southern Mississippi) with St. Louis fed banks (northern Mississippi)
- Could compare after lending, but what's the assumption here?

Did the policy of extra lending save banks?

- Basic idea: compare what happened to Atlanta fed banks (southern Mississippi) with St. Louis fed banks (northern Mississippi)
- Could compare after lending, but what's the assumption here?
- Assumption: same levels before intervention (very strict assumption)

In fact, pre-intervention levels are different!

TABLE 4
BANK SUSPENSIONS AND LIQUIDATIONS

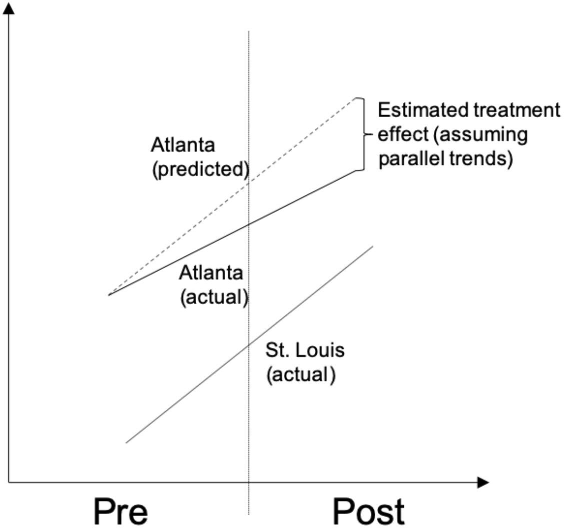
Begin July 1		End June 30	Percentage of Banks Suspending: All (1)	Federal Reserve District: 6th Atlanta (2)	Federal Reserve District: 8th St. Louis (3)	Percentage of Banks Liquidating: All (4)	Federal Reserve District: 6th Atlanta (5)	Federal Reserve District: 8th St. Louis (6)
1929	to	1930	4.8	7.1	3.0	4.5	7.1	2.4
1930	to	1931	28.9	14.2	39.5	13.6	7.1	18.6
1931	to	1932	13.2	14.9	11.8	8.0	7.9	8.1
1932	to	1933	7.7	7.5	7.9	7.3	6.5	7.9
1933	to	1934	.9	.0	1.7	.9	.0	1.7
1929	to	1934[a]	49.8	38.7	59.2	30.9	26.8	34.4

SOURCE.—*Rand McNally Bankers Directory* and National Archives and Records Administration Record Group 82. See Section II and Richardson (2006, 2007*a*, 2007*b*, 2008) for details.

[a] The last row indicates the percentage of banks operating on July 1, 1929, that either suspended or liquidated by June 30, 1933.

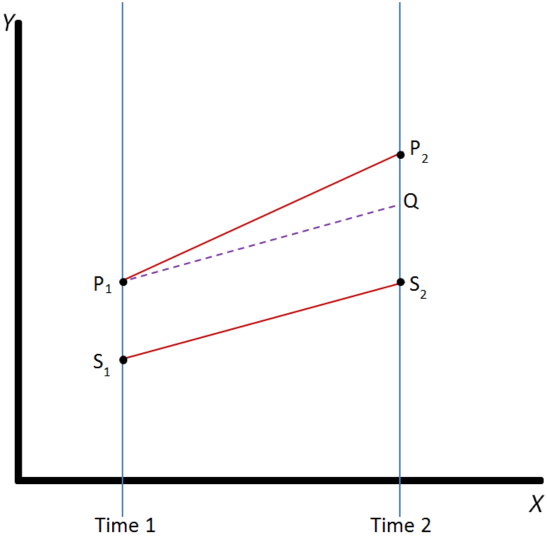
Did the policy of extra lending save banks?

- Instead, compare *changes* from before to after treatment
- Assumption: parallel trends
- If valid, the fact the districts were different prior to the treatment isn't a problem


Estimated treatment effect (assuming parallel trends)
Atlanta (predicted)
Atlanta (actual)
St. Louis (actual)
Pre
Post

- Difference 1: St. Louis post minus St. Louis pre
- Difference 2: Atlanta post minus Atlanta pre
- Difference-in-differences: Difference 2 minus difference 1

"Differences in differences" graphically



Parallel trends assumption

- The key assumption in differences-in-differences is the parallel trends assumption
  - *If the treated group had not been treated, it would have changed by the same amount ("had the same trend") as the comparison group.*
- This is a counterfactual assumption: We cannot explicitly test it
- What can we do instead?

Parallel trends assumption

- The key assumption in differences-in-differences is the parallel trends assumption
  - *If the treated group had not been treated, it would have changed by the same amount ("had the same trend") as the comparison group.*
- This is a counterfactual assumption: We cannot explicitly test it
- What can we do instead?
  - We can test trends *before* treatment
  - Or in the case of this article, *after* treatment!

Richardson and Troost (2009) - Testing the assumption

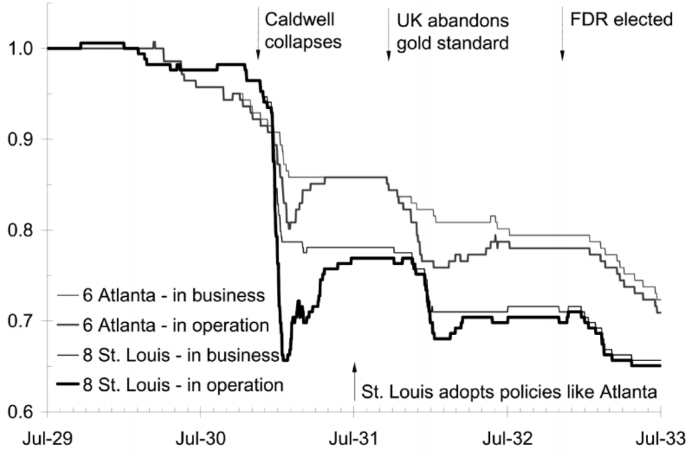


FIG. 3.—Percentage of banks in business and in operations in the 6th and 8th Federal Reserve Districts in Mississippi, July 1929 to June 1933. Source: See Section II.

- Can be estimated in a straightforward regression:

$$Y_{it} = \beta_0 + \beta_1 TREAT_i + \beta_2 POST_t + \beta_3 (POST_t \times TREAT_i) + \varepsilon_{it} \tag{1}$$

- Can be estimated in a straightforward regression:

$$Y_{it} = \beta_0 + \beta_1 TREAT_i + \beta_2 POST_t + \beta_3 (POST_t \times TREAT_i) + \varepsilon_{it} \tag{1}$$

  - β_0: pre mean for the comparison group
  - β_1: difference in the pre mean between the treated and untreated group
  - β_2: difference in means between the pre and post period for the comparison group
  - β_3: difference-in-differences estimate
    - This is the difference in the change from pre to post for the treated group relative to the comparison group

Card and Krueger (1994) - Minimum wage and employment

```r
ckdata <- read_csv("cardkruegerlong.csv")
head(ckdata)
```

```
## # A tibble: 6 x 15
##   sheet chain co_owned state southj centralj northj   pa1   pa2 shore calls
##   <dbl> <dbl>    <dbl> <dbl>  <dbl>    <dbl>  <dbl> <dbl> <dbl> <dbl> <chr>
## 1   506     2        1     0      0        0      0     1     0     0 0
## 2    56     4        1     0      0        0      0     1     0     0 0
## 3    61     4        1     0      0        0      0     1     0     0 0
## 4    62     4        1     0      0        0      0     1     0     0 2
## 5   445     1        0     0      0        0      0     0     1     0 0
## 6   451     1        0     0      0        0      0     0     1     0 0
## # i 4 more variables: fulltime <chr>, parttime <chr>, managers <chr>,
## #   post <dbl>
```

```r
# note that state = 1 for NJ and 0 for PA.
# also note that post = 1 for 1993 and 0 for 1992
# NJ is treated group, so state = 1 means treat = 1

# looks like fulltime is a character! let's try to make it numeric
ckdata$fulltime_num <- as.numeric(ckdata$fulltime)
```

```
## Warning: NAs introduced by coercion
```

```r
ckdata$parttime_num <- as.numeric(ckdata$parttime)
```

```
## Warning: NAs introduced by coercion
```

Card and Krueger (1994) - Minimum wage and employment

```r
# said there are NAs in the data, so let's see where they are
ckdata %>% filter(is.na(fulltime_num)) %>% select(fulltime, fulltime_num)
```

```
## # A tibble: 18 x 2
##    fulltime fulltime_num
##    <chr>           <dbl>
##  1 .                  NA
##  2 .                  NA
##  3 .                  NA
##  4 .                  NA
##  5 .                  NA
##  6 .                  NA
##  7 .                  NA
##  8 .                  NA
##  9 .                  NA
## 10 .                  NA
## 11 .                  NA
## 12 .                  NA
## 13 .                  NA
## 14 .                  NA
## 15 .                  NA
## 16 .                  NA
## 17 .                  NA
## 18 .                  NA
```

```r
# ah, so they are .! those are missing values in Stata, so leave as missing.
```

```r
reg1 <- feols(fulltime_num ~ state + post + state:post, data = ckdata, vcov = "HC1")
summary(reg1)
```

```
## OLS estimation, Dep. Var.: fulltime_num
## Observations: 798
## Standard-errors: Heteroskedasticity-robust
##              Estimate Std. Error  t value   Pr(>|t|)
## (Intercept)  9.99342    1.21845  8.20172 9.5089e-16 ***
## state       -2.26949    1.29598 -1.75118 8.0301e-02 .
## post        -2.27342    1.56537 -1.45232 1.4681e-01
## state:post   2.99498    1.68446  1.77801 7.5786e-02 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 8.24468   Adj. R2: 0.002734
```

Card and Krueger (1994) - Minimum wage and employment

```r
reg1 <- feols(fulltime_num ~ state + post + state:post, data = ckdata, vcov = "HC1")
reg2 <- feols(parttime_num ~ state + post + state:post, data = ckdata, vcov = "HC1")
table <- etable(reg1, reg2,
                # standard errors, digits, fit statistics, put SE below coefficients (the norm)
                vcov = "HC1", digits = 3, fitstat = "", se.below = TRUE,
                # change significance codes to the norm
                signif.code = c("***" = 0.01, "**" = 0.05, "*" = 0.1),
                # rename the variables
                dict = c("Constant" = "Intercept", "state" = "Treat", "post" = "Post", "state:post" = "Treat x Post"))
table
```

```
##                          reg1         reg2
## Dependent Var.: fulltime_num parttime_num
##
## Constant              9.99***      19.7***
##                      (1.22)       (1.09)
## Treat                -2.27*       -1.04
##                      (1.30)       (1.23)
## Post                 -2.27         0.103
##                      (1.57)       (1.60)
## Treat x Post          2.99*       -0.405
##                      (1.68)       (1.80)
## _______________ ____________ ____________
## S.E. type       Hetero.-rob. Hetero.-rob.
## ---
## Signif. codes: 0 '***' 0.01 '**' 0.05 '*' 0.1 ' ' 1
```

```r
# drop some rows
table <- table[-c(1:2, 11:nrow(table)),]
# rename columns
colnames(table) <- c("", "Full-time", "Part-time")
```

Card and Krueger (1994) - Minimum wage and employment

```
kable(table,
      align = "lcc", booktabs = TRUE, linesep = "", escape = FALSE, row.names = FALSE) %>%
  column_spec(1,width = "2cm") %>%
  column_spec(c(2:3),width = "1.5cm") %>%
  kable_styling() %>%
  footnote("* p < 0.1, ** p < 0.05, *** p < 0.01.", general_title = "",
           footnote_as_chunk = TRUE,
           escape = FALSE
           ) %>%
  footnote("Note: Robust standard errors in parentheses.", general_title = "",
           footnote_as_chunk = TRUE,
           escape = FALSE
           )
```

	Full-time	Part-time
Constant	9.99***	19.7***
	(1.22)	(1.09)
Treat	-2.27*	-1.04
	(1.30)	(1.23)
Post	-2.27	0.103
	(1.57)	(1.60)
Treat x Post	2.99*	-0.405
	(1.68)	(1.80)

Note: Robust standard errors in parentheses.
* p < 0.1, ** p < 0.05, *** p < 0.01.

Card and Krueger (1994) - Poisson regression!

```
reg1 <- feglm(fulltime_num ~ state + post + state:post, data = ckdata, vcov = "HC1", family = "poisson")
reg2 <- feglm(parttime_num ~ state + post + state:post, data = ckdata, vcov = "HC1", family = "poisson")
table <- etable(reg1, reg2,
                # standard errors, digits, fit statistics, put SE below coefficients (the norm)
                vcov = "HC1", digits = 3, fitstat = "", se.below = TRUE,
                # change significance codes to the norm
                signif.code = c("***" = 0.01, "**" = 0.05, "*" = 0.1),
                # rename the variables
                dict = c("Constant" = "Intercept", "state" = "Treat", "post" = "Post", "state:post" = "Treat x Post"))
table
```

```
##                         reg1         reg2
## Dependent Var.: fulltime_num parttime_num
##
## Constant              2.30***      2.98***
##                      (0.122)      (0.056)
## Treat                -0.258*      -0.054
##                      (0.135)      (0.063)
## Post                 -0.258        0.005
##                      (0.176)      (0.081)
## Treat x Post          0.347*      -0.021
##                      (0.192)      (0.092)
## _______________ ____________ ____________
## S.E. type       Hetero.-rob. Hetero.-rob.
## ---
## Signif. codes: 0 '***' 0.01 '**' 0.05 '*' 0.1 ' ' 1
```

```
# drop some rows
table <- table[-c(1:2, 11:nrow(table)),]
# rename columns
colnames(table) <- c("", "Full-time", "Part-time")
```

Card and Krueger (1994) - Minimum wage and employment

```r
kable(table, caption = "Poisson regression", # adding a caption
      align = "lcc", booktabs = TRUE, linesep = "", escape = FALSE, row.names = FALSE) %>%
  column_spec(1,width = "2cm") %>%
  column_spec(c(2:3),width = "1.5cm") %>%
  kable_styling() %>%
  footnote("* p < 0.1, ** p < 0.05, *** p < 0.01.", general_title = "",
           footnote_as_chunk = TRUE,
           escape = FALSE
           ) %>%
  footnote("Note: Robust standard errors in parentheses.", general_title = "",
           footnote_as_chunk = TRUE,
           escape = FALSE
           )
```

Table 1: Poisson regression

	Full-time	Part-time
Constant	2.30***	2.98***
	(0.122)	(0.056)
Treat	-0.258*	-0.054
	(0.135)	(0.063)
Post	-0.258	0.005
	(0.176)	(0.081)
Treat x Post	0.347*	-0.021
	(0.192)	(0.092)

Note: Robust standard errors in parentheses.

* $p < 0.1$, ** $p < 0.05$, *** $p < 0.01$.

- Can add control variables

$$Y_{it} = \beta_0 + \beta_1 TREAT_i + \beta_2 POST_t + \tag{2}$$
$$\beta_3 (POST_t \times TREAT_i) + X_{it} + \varepsilon_{it} \tag{3}$$

- Adding controls can help control for differing trends ("conditional" parallel trends)
- Note: the interpretation of β_0 is no longer the same; others stay the same

Standard errors in differences-in-differences

- Card and Krueger did not cluster standard errors
  - In fact, that would have been difficult because they really only had two "clusters"!
  - But their robust standard errors are likely underestimated due to the clustering

- Classic reference: Bertrand, Duflo, and Mullainathan (2004)
  - "How Much Should We Trust Differences-in-Differences Estimates?"

Standard errors in differences-in-differences

- Bertrand, Duflo, and Mullainathan (2004) suggest three possibilities:
  1. Cluster at the group level
  2. Block bootstrap (not going to discuss)
  3. Aggregating data into one pre and one post period (event studies later)

- Let's go through these

Clustering

- The most common approach: cluster standard errors
- Cameron, Gelbach, and Miller (2008) show that this is problematic with few clusters
  - "Bootstrap-based improvements for inference with clustered errors"
- The authors look at many possible approaches and find that the "wild cluster bootstrap" seems to perform best, on average

Wild cluster bootstrap

- The wild cluster bootstrap is a “non-parametric” bootstrap
  - I’ll do the non-cluster as an example. Software makes this easy!
- Suppose we are interested in the following regression:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

- We want to test whether $\beta_1 = 0$ and we have relatively few clusters (say between 5 and 30)

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i \tag{4}$$

1. Estimate above regression and obtain $\hat{\beta}$, $\hat{\varepsilon}$
2. Impose the null hypothesis ($\beta_1 = 0$) and estimate the restricted regression:

$$\tilde{y}_i = \tilde{\beta}_0 + \tilde{\varepsilon}_i \tag{5}$$

3. Bootstrap replications:

- Use equation 5 to generate $\tilde{y}_i^b$, where $\tilde{y}_i^b = \tilde{\beta}_0^b + \tilde{\varepsilon}_i^b$
  - Rademacher weights: The randomness comes from either adding $\hat{\varepsilon}_i$ or $-\hat{\varepsilon}_i$ with equal probability
- Estimate:

$$\tilde{y}_i^b = \tilde{\beta}_0^{b*} + \tilde{\beta}_1^{b*} x_i + \tilde{\varepsilon}_i^{b*} \tag{6}$$

- Calculate the t-statistic for the bootstrap replication:

$$t^{b*} = \frac{\tilde{\beta}_1^{b*}}{\sqrt{\tilde{V}^{b*}}} \tag{7}$$

Two-tailed test:

- Reject the null hypothesis if

$$|t^{b*}| > |t^*| \text{ for } b = 1, \ldots, B,$$

  where t^* is the t-statistic from the *original regression*.

- P-value across B bootstrap samples is:

$$\frac{1}{B}\sum_{b=1}^{B} \mathbb{I}(|t^{b*}| > |t^*|), \tag{8}$$

where $\mathbb{I}$ is the indicator function.

Implementing WCB in R

- Thankfully there's a package that allows us to do this!
  - `fwildclusterboot` (Friedrich, 2019)
- This package works with `fixest` objects!
- Let's use the `castle.dta` data in the GitHub repo to test this

Implementing WCB in R

```r
library(haven) # to load .dta files
df <- read_dta("castle.dta")
head(df)
```

```
## # A tibble: 6 x 185
##   state   year   sid   cdl pre2_cdl caselaw anywhere assumption civil homicide_c
##   <chr>  <dbl> <dbl> <dbl>    <dbl>   <dbl>    <dbl>      <dbl> <dbl>      <dbl>
## 1 Alaba~  2000     1     0        0       0        0          0     0        329
## 2 Alaba~  2001     1     0        0       0        0          0     0        379
## 3 Alaba~  2002     1     0        0       0        0          0     0        303
## 4 Alaba~  2003     1     0        0       0        0          0     0        299
## 5 Alaba~  2004     1     0        1       0        0          0     0        254
## 6 Alaba~  2005     1     0        1       0        0          0     0        374
## # i 175 more variables: robbery_gun_r <dbl>, jhcitizen_c <dbl>,
## #   jhpolice_c <dbl>, homicide <dbl>, robbery <dbl>, assault <dbl>,
## #   burglary <dbl>, larceny <dbl>, motor <dbl>, murder <dbl>,
## #   hc_felonywsus <dbl>, jhcitizen <dbl>, jhpolice <dbl>, population <dbl>,
## #   police <dbl>, unemployrt <dbl>, income <dbl>, blackm_15_24 <dbl>,
## #   whitem_15_24 <dbl>, blackm_25_44 <dbl>, whitem_25_44 <dbl>, prisoner <dbl>,
## #   lagprisoner <dbl>, poverty <dbl>, exp_subsidy <dbl>, ...
```

```r
# key variables: state, year, cdl ("treatment"), and homicide_c (outcome)
# homicide_c to rate (per 100,000 people)
df$homicide_c <- (df$homicide_c/df$population)*100000
# and log
df$homicide_c <- log(df$homicide_c)
```

Implementing WCB in R

```r
# Note: this is not differences-in-differences.
# Just an example of the wild cluster bootstrap with fwildclusterboot
reg1 <- feols(homicide_c ~ cdl, data = df, cluster = "state")
summary(reg1)
```

```
## OLS estimation, Dep. Var.: homicide_c
## Observations: 550
## Standard-errors: Clustered (state)
##             Estimate Std. Error  t value   Pr(>|t|)
## (Intercept) 1.360284   0.082267 16.53498  < 2.2e-16 ***
## cdl         0.338002   0.092532  3.65279 0.00063065 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 0.578226   Adj. R2: 0.036511
```

Implementing WCB in R

```r
reg1 <- feols(homicide_c ~ cdl, data = df, cluster = "state")
boot_reg <- boottest(
                    reg1,
                    clustid = c("state"),
                    param = "cdl",
                    B = 10000,
                    type = "rademacher" # default weighting, by the way
                    )
boot_reg
```

```
## boottest.fixest(object = reg1, param = "cdl", B = 10000, clustid = c("state"),
##     type = "rademacher")
##
## p value: 4e-04
## confidence interval: 0.1541 0.5227
## test statistic 3.6528
```

Add controls

```r
reg1 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt, data = df, cluster = "state")
reg1
```

```
## OLS estimation, Dep. Var.: homicide_c
## Observations: 550
## Standard-errors: Clustered (state)
##                  Estimate Std. Error   t value   Pr(>|t|)
## (Intercept)      8.328111   5.001764  1.665035 1.0229e-01
## cdl              0.156808   0.088948  1.762923 8.4149e-02 .
## log(population)  0.304923   0.062867  4.850276 1.2902e-05 ***
## log(income)     -1.061536   0.477547 -2.222894 3.0868e-02 *
## unemployrt      -0.006588   0.021765 -0.302699 7.6340e-01
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 0.471373   Adj. R2: 0.35618
```

```r
boot_reg <- boottest(
                    reg1,
                    clustid = c("state"),
                    param = "cdl",
                    B = 10000,
                    type = "rademacher" # default weighting, by the way
                    )
boot_reg
```

```
## boottest.fixest(object = reg1, param = "cdl", B = 10000, clustid = c("state"),
##     type = "rademacher")
##
## p value: 0.0979
## confidence interval: -0.0305 0.3417
## test statistic 1.7629
```

Multi-way clustering, too!

```
reg1 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt, data = df, cluster = c("state", "year"))
reg1
```

```
## OLS estimation, Dep. Var.: homicide_c
## Observations: 550
## Standard-errors: Clustered (state & year)
##                  Estimate Std. Error   t value   Pr(>|t|)
## (Intercept)      8.328111   5.024169  1.657609 0.12838986
## cdl              0.156808   0.100698  1.557209 0.15047711
## log(population)  0.304923   0.062555  4.874462 0.00064729 ***
## log(income)     -1.061536   0.476525 -2.227661 0.05004055 .
## unemployrt      -0.006588   0.024931 -0.264260 0.79694594
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 0.471373   Adj. R2: 0.35618
```

```
boot_reg <- boottest(
                    reg1,
                    clustid = c("state"),
                    param = "cdl",
                    B = 10000,
                    type = "rademacher" # default weighting, by the way
                    )
boot_reg
```

```
## boottest.fixest(object = reg1, param = "cdl", B = 10000, clustid = c("state"),
##     type = "rademacher")
##
## p value: 0.0993
## confidence interval: -0.0311 0.3419
## test statistic 1.7629
```

Finally, Webb weights (Webb, 2023) – but using Rademacher weights is the norm

```r
reg1 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt, data = df, cluster = c("state", "year"))
reg1
```

```
## OLS estimation, Dep. Var.: homicide_c
## Observations: 550
## Standard-errors: Clustered (state & year)
##                  Estimate Std. Error   t value   Pr(>|t|)
## (Intercept)      8.328111   5.024169  1.657609 0.12838986
## cdl              0.156808   0.100698  1.557209 0.15047711
## log(population)  0.304923   0.062555  4.874462 0.00064729 ***
## log(income)     -1.061536   0.476525 -2.227661 0.05004055 .
## unemployrt      -0.006588   0.024931 -0.264260 0.79694594
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 0.471373   Adj. R2: 0.35618
```

```r
boot_reg <- boottest(
                    reg1,
                    clustid = c("state"),
                    param = "cdl",
                    B = 10000,
                    type = "webb"
                    )
boot_reg
```

```
## boottest.fixest(object = reg1, param = "cdl", B = 10000, clustid = c("state"),
##     type = "webb")
##
## p value: 0.103
## confidence interval: -0.0326 0.3431
## test statistic 1.7629
```

Some thoughts on clustering

- If you have more than ~30 clusters, you can probably just cluster at the group level
  - We see that the standard errors are very similar in the state examples above
- Otherwise, consider using an alternative approach
  - Also important when clusters have wildly different sample sizes, or where the treated clusters are relatively few (Moulton, 1990)
- Alternative approach with only one treated cluster: randomization inference

- They are interested in the change in insurance coverage in Hawaii relative to other states:

$$Y_{ist} = X_{ist}\beta^t + Z_{st}\gamma^t + H_{it}\delta^t + \phi_{st} + \eta_{it} \tag{9}$$

- They calculate the change as: $\Delta = \delta^1 - \delta^0$
- The idea: see where the Hawaii effect sits in the distribution of the same effect across *all US states*
  - "Placebo" effects
  - Note that this is not true "randomization" inference
    - I'll show you an example with one of my papers in a minute

Randomization inference in Buchmueller, DiNardo, and Valletta (2011)

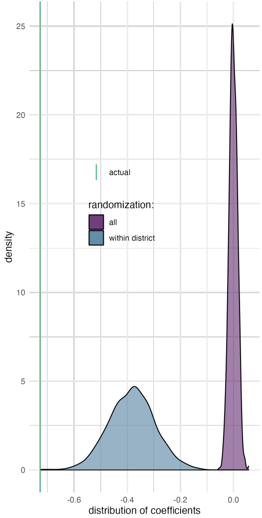
Table 1—State Effects (*percentages*), Own ESI Coverage (*by quintile*)
(*Placebo tests, Hawaii versus all other states*)

	Full sample	ESI quintiles				
		1st	2nd	3rd	4th	5th
Panel A. 1979–1982						
Hawaii effect	0.139*	0.209*	0.133*	0.132**	0.085**	0.002
Placebo effects (other states)						
95th percentile	0.046	0.066	0.065	0.055	0.045	0.031
5th percentile	−0.046	−0.062	−0.070	−0.070	−0.052	−0.046
Panel B. 2002–2005						
Hawaii effect	0.181**	0.281**	0.232**	0.142**	0.101**	0.031
Placebo tests (other states)						
95th percentile	0.043	0.045	0.062	0.054	0.045	0.039
5th percentile	−0.057	−0.067	−0.069	−0.070	−0.054	−0.052
Panel C. Difference (*2002–2005 minus 1979–1982*)						
Hawaii effect	0.042	0.071*	0.099*	0.011	0.016	0.030
Placebo tests (other states)						
95th percentile	0.051	0.067	0.087	0.077	0.052	0.049
5th percentile	−0.046	−0.076	−0.065	−0.066	−0.068	−0.040

Randomization inference in Merfeld (2023)

- I'm interested in the effects of pollution on agricultural productivity in India
- I have villages, which are nested within districts
  - I cluster on villages
- Alternative: randomly assign pollution to villages *within the same district* and compare my effects to the distribution of effects

Randomization inference in Merfeld (2023)



Placebo tests for the parallel trends assumption

- Above, we looked at the parallel trends assumption graphically in Richardson and Troost (2009)
- Another common way is to look at *leads* of treatment
  - In my paper, for example, pollution next year should not affect agricultural productivity this year

Leads of pollution in Merfeld (2023)

	(1)	(2)
particulate matter (one-year lead)	-0.033	
	(0.067)	
particulate matter (two-year lead)		-0.070
		(0.052)
weather (expanded)	No	No
fixed effects:		
village	Yes	Yes
year	Yes	Yes
F	592	783
observations	1,161,265	1,055,562

Note: Standard errors are in parentheses and are clustered at the village level. * $p<0.10$ ** $p<0.05$ *** $p<0.01$

Convincing the reader is like writing a good story

- When you're writing a diff-in-diff paper, think about the possible threats to your identification strategy
- Then, think about how you can convince the reader that your strategy is valid
  - Use placebos: is there somewhere we shouldn't expect an effect?
  - In the case of my paper, the leads convinced some seminar participants!
- You can likewise think of heterogeneity we would *expect* to see, and test for that!

Before moving on to some of the new literature...

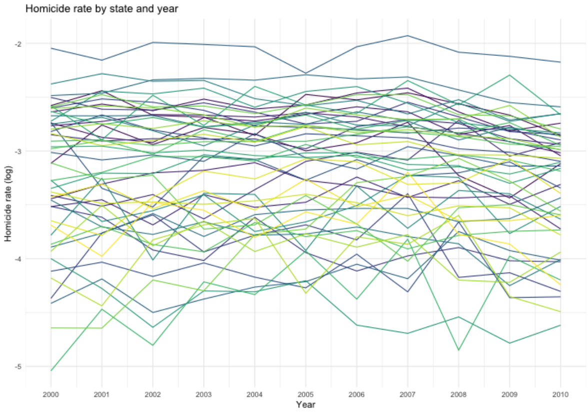
- Let's talk about fixed and random effects
  - Fixed effects will be important for the upcoming discussions

- Some nice (but old) slides from Oscar Tores-Reyna here.

Panel data

- Both fixed and random effects are used in panel data
  - Panel data: data with multiple observations for each unit
  - Examples: individuals, firms, countries, etc.
- In our previous example of homicide and castle doctrine laws: unit is the state!

Panel data


Homicide rate by state and year
Homicide rate (log)
-2
-3
-4
-5
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
Year

Fixed effects

- Fixed effects are a way to control for omitted variables
  - However there is a key assumption: the omitted variable is time-invariant

- Fixed effects are also called "within" effects
  - Why? Because we are looking at the variation within each unit

- The regression is of the form:

$$y_{it} = \alpha_i + \beta x_{it} + \varepsilon_{it}, \tag{10}$$

where α_i is the fixed effect for unit i. Note the subscript! No t!

Fixed effects, empirically

- Empirically, what are fixed effects doing?
  - They are subtracting the mean of **each unit** from the outcome variable
- In a regression, we add a dummy variable for each unit
  - We have to leave out one dummy variable, though
  - Software will do this for us!
  - Note that the intercept is usually meaningless in this case
- Cannot include time-invariant variables in the regression
  - Why? Because the fixed effect will absorb them!

Fixed effects with feols

- `feols` makes this easy on us. Let's return to our previous example.

```
reg1 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt, data = df, cluster = c("state"))
reg2 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt | state, data = df, cluster = c("state"))
etable(reg1, reg2,
       # standard errors, digits, fit statistics, put SE below coefficients (the norm)
       digits = 3, fitstat = "", se.below = TRUE,
       # change significance codes to the norm
       signif.code = c("***" = 0.01, "**" = 0.05, "*" = 0.1))
```

```
##                         reg1       reg2
## Dependent Var.: homicide_c homicide_c
##
## Constant           3.72
##                   (5.00)
## cdl                0.157*     0.059
##                   (0.089)    (0.049)
## log(population)    0.305***  -0.151
##                   (0.063)    (0.341)
## log(income)       -1.06**     0.242
##                   (0.478)    (0.335)
## unemployrt        -0.007     -0.024***
##                   (0.022)    (0.006)
## Fixed-Effects:  ---------- ----------
## state                   No        Yes
## _______________ __________ __________
## S.E.: Clustered  by: state  by: state
## ---
## Signif. codes: 0 '***' 0.01 '**' 0.05 '*' 0.1 ' ' 1
```

Just a quick note: wild cluster bootstrap still works!

```r
# need to use a numeric variable for the bootstrap. sid is in our data, thankfully.
reg1 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt | sid, data = df, cluster = c("sid"))
reg1
```

```
## OLS estimation, Dep. Var.: homicide_c
## Observations: 550
## Fixed-effects: sid: 50
## Standard-errors: Clustered (sid)
##                  Estimate Std. Error   t value   Pr(>|t|)
## cdl              0.058781   0.049370  1.190624 0.23953723
## log(population) -0.151406   0.341426 -0.443451 0.65939131
## log(income)      0.241548   0.335153  0.720711 0.47451204
## unemployrt      -0.023468   0.006455 -3.635678 0.00066447 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 0.179187     Adj. R2: 0.897774
##                  Within R2: 0.058749
```

```r
boot_reg <- boottest(
                    reg1,
                    clustid = c("sid"), # note that it requires a numeric variable!
                    param = "cdl",
                    B = 10000
                    )
boot_reg
```

```
## boottest.fixest(object = reg1, param = "cdl", B = 10000, clustid = c("sid"))
##
## p value: 0.283
## confidence interval: -0.0345 0.1615
## test statistic 1.1917
```

Fixed effects are the norm with “differences-in-differences”

- It’s not quite the same as the canonical differences-in-differences model

- We redefine treatment for the same unit
  - Before treatment, the value is zero, and after it is One
  - Note that this is different from the canonical model, where the value is zero for the comparison group and one for the treated group

- In fact, the regression we just saw is a differences-in-differences model of this form!
  - In practice, we often tend to add time fixed effects, too:

$$y_{it} = \alpha_i + \delta_t + \beta x_{it} + \varepsilon_{it}, \tag{11}$$

The “effect” of castle doctrine laws, two-way fixed effects

```r
reg1 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt | state, data = df, cluster = c("state"))
reg2 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt | state + year, data = df, cluster = c("state"))
etable(reg1, reg2,
       # standard errors, digits, fit statistics, put SE below coefficients (the norm)
       digits = 3, fitstat = "", se.below = TRUE,
       # change significance codes to the norm
       signif.code = c("***" = 0.01, "**" = 0.05, "*" = 0.1))
```

```
##                       reg1       reg2
## Dependent Var.: homicide_c homicide_c
##
## cdl                  0.059      0.076
##                    (0.049)    (0.056)
## log(population)   -0.151     -0.974**
##                    (0.341)    (0.484)
## log(income)          0.242      0.283
##                    (0.335)    (0.308)
## unemployrt        -0.024***  -0.009
##                    (0.006)    (0.012)
## Fixed-Effects:  ---------- ----------
## state                  Yes        Yes
## year                    No        Yes
## _______________ __________ __________
## S.E.: Clustered  by: state  by: state
## ---
## Signif. codes: 0 '***' 0.01 '**' 0.05 '*' 0.1 ' ' 1
```

Random effects

- Before turning to recent issues discovered with the two-way fixed effects estimator, let's talk about random effects
- Random effects are a way to capture the heterogeneity across units
  - The key is that this heterogeneity is *random* and uncorrelated with the predictors in the model
- This is really a way to capture the *variance* across units
  - In practice, this absorbs some of the variance, increasing precision (but at the cost of the assumption above)

Random effects in R

```r
library(lme4)
reg1 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt, data = df, cluster = "state")
# note iid standard errors for simplicity for random effects
# (can use other packages to change the vcov calculation if we think assumptions aren't exactly true...)
reg2 <- lmer(homicide_c ~ cdl + log(population) + log(income) + unemployrt + (1 | state), data = df)
reg3 <- feols(homicide_c ~ cdl + log(population) + log(income) + unemployrt | state, data = df, cluster = "state")
modelsummary(list("None" = reg1, "RE" = reg2, "FE" = reg3), gof_omit = ".*",
             output = "markdown", coef_omit = c(-2,-3,-4,-5),
             coef_rename = c("cdl" = "Castle laws", "log(population)" = "Population (log)",
                             "log(income)" = "Income (log)", "unemployrt" = "Unemployment rate"))
```

	None	RE	FE
Castle laws	0.157	0.043	0.059
	(0.089)	(0.029)	(0.049)
Population (log)	0.305	0.276	-0.151
	(0.063)	(0.066)	(0.341)
Income (log)	-1.062	0.001	0.242
	(0.478)	(0.191)	(0.335)
Unemployment rate	-0.007	-0.029	-0.023
	(0.022)	(0.006)	(0.006)

Some things about random effects relative to vanilla OLS (not fixed effects):

- The random effects estimator is asymptotically more efficient than OLS if there is unit-level heterogeneity: $\mathbb{E}(V_{RE}) < \mathbb{E}(V_{OLS})$
  - In practice with finite samples, not necessarily
- In expectation, coefficients are the same: $\mathbb{E}(\beta_{RE}) = \mathbb{E}(\beta_{OLS})$
  - Random effects is estimated using (feasible) generalized least squares, which essentially reweights the observations

Bias in TWFE

- Recently, a number of papers have shown that the two-way fixed effects estimator can be... problematic

- We have been discussing differences-in-differences with TWFE of the following form:

$$y_{it} = \alpha_i + \delta_t + \beta D_{it} + \gamma x_{it} + \varepsilon_{it}, \tag{12}$$

  where D_{it} is a dummy variable for treatment.

- If there are only two time periods and one group receives treatment in only one period, this is not a problem!
  - The Card and Krueger setup is not an issue with TWFE

TWFE and differential treatment timing

- The real issue is when treatment is staggered across time
  - For example, if treatment is introduced in different years for different states
- It turns out this is the case with the castle doctrine law!

year	treated
2000	0.00
2001	0.00
2002	0.00
2003	0.00
2004	0.00
2005	0.00
2006	0.02
2007	0.28
2008	0.36
2009	0.40
2010	0.42

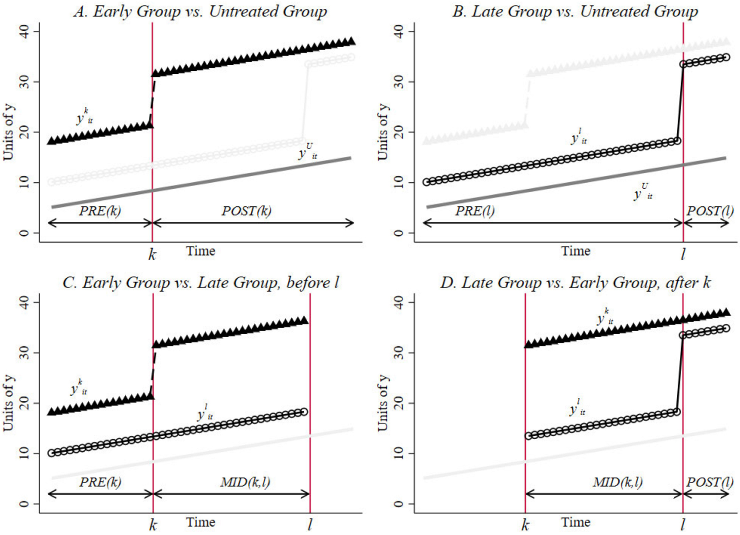
- Goodman-Bacon lays out the problem (as does Scott in *CI*)

- Suppose we have three groups and three time periods
  - Group 1 is treated before period 2 (Goodman-Bacon calls this group k)
  - Group 2 is treated before period 3 (Goodman-Bacon calls this group l)
  - Group 3 is never treated (Goodman-Bacon calls this group U)

- If we are willing to assume treatment effect *homogeneity*, then we have no problems!
  - Are you willing to assume this?

- Goodman-Bacon shows that the overall treatment effect is a *weighted average* of treatment effects from every possible 2x2 comparison where treatment status doesn't change:
  - Group 1 vs group 2
  - Group 1 vs group 3
  - Group 2 vs group 1
  - Group 2 vs group 3
- These weights are a function of two things:
  - Group sizes
  - Variance in treatment

Goodman-Bacon (2021)



- He shows there are three relevant comparisons:

$$\hat{\beta}^{2x2}_{kU} \equiv \left(\bar{y}_k^{POST(k)} - \bar{y}_k^{PRE(k)}\right) - \left(\bar{y}_U^{POST(k)} - \bar{y}_U^{PRE(k)}\right) \tag{13}$$

$$\hat{\beta}^{2x2,k}_{kl} \equiv \left(\bar{y}_k^{MID(k,l)} - \bar{y}_k^{PRE(k)}\right) - \left(\bar{y}_l^{MID(k,l)} - \bar{y}_l^{PRE(k)}\right) \tag{14}$$

$$\hat{\beta}^{2x2,l}_{kl} \equiv \left(\bar{y}_l^{POST(l)} - \bar{y}_l^{MID(k,l)}\right) - \left(\bar{y}_k^{POST(l)} - \bar{y}_k^{MID(k,l)}\right), \tag{15}$$

where k and l are treated groups, and U is the untreated group.

- The DD estimator is a weighted average of all these comparisons.

- Generalizing to K time periods:

$$\hat{\beta}^{DD} = \sum_{k \neq U} s_{kU} \hat{\beta}_{kU}^{2x2} + \sum_{k \neq U} \sum_{l > k} \left[s_{kl}^{k} \hat{\beta}_{kl}^{2x2,k} + s_{kl}^{l} \hat{\beta}_{kl}^{2x2,l} \right], \tag{16}$$

where s_{ij} is the weight for the comparison between groups i and j.

The weights:

$$s_{kU} = \frac{(n_k + n_U)^2 n_{kU}(1 - n_{kU})\bar{D}_k(1 - \bar{D}_k)}{\hat{V}^D} \tag{17}$$

$$s_{kl}^k = \frac{\left((n_k + n_l)(1 - \bar{D}_l)\right)^2 n_{kl}(1 - n_{kl})\frac{\bar{D}_k - \bar{D}_l}{1 - \bar{D}_l}\frac{1 - \bar{D}_k}{1 - \bar{D}_k}}{\hat{V}^D} \tag{18}$$

$$s_{kl}^l = \frac{\left((n_k + n_l)(\bar{D}_k)\right)^2 n_{kl}(1 - n_{kl})\frac{\bar{D}_l}{\bar{D}_k}\frac{\bar{D}_k - \bar{D}_l}{\bar{D}_k}}{\hat{V}^D} \tag{19}$$

- Note how the variance of treatment affects the weights! "Changing the number or spacing of time periods changes the weights" (Goodman-Bacon).
  - Even if the treatment effect is constant, changing the length of the panel can change the weighted average if different groups have different treatment effects.

- de Chaisemartin and D'Haultfœuille (2020) more explicitly show the problem with weights.

- Let $w_{g,t}$ represent the weight for group g at time t, $\Delta_{g,t}$ represent the average treatment effect for the same, and N_1 represent the total number of treated units at the time:

$$\beta_{FE} = \mathbb{E}\left[\sum_{(g,t):D_{g,t}=1} \frac{N_{g,t}}{N_1} w_{g,t}\Delta_{g,t}\right] \tag{20}$$

- The fixed effect estimator is a weighted average of these treatment effects.
  - But how do we get the weights?

- Let $D_{g,t}$ represent treatment status of group g at time t:

$$D_{g,t} = \alpha + \gamma_g + \lambda_t + \varepsilon_{g,t}, \tag{21}$$

  where γ_g is the group fixed effect and λ_t is the time fixed effect.

- It turns out that the average residual *is not zero*, so let's rescale it by its mean:

$$w_{g,t} = \frac{\varepsilon_{g,t}}{\sum_{(g,t):D_{g,t}=1} \frac{N_{g,t}}{N_1} \varepsilon_{g,t}} \tag{22}$$

- Since $\varepsilon_{g,t}$ can be positive or negative, this means that the weights can be negative!

- The problem is an *extrapolation* problem
  - Essentially, "the regression predicts a treatment probability larger than the one in that cell" (de Chaisemartin and D'Haultfœuille)
- Note that if the treatment effect is constant, then the weighted average is always the same, no matter the weights
  - But if the treatment effect is not constant, then the weighted average can be different from the true average

Back to Bacon-Goodman

- Let's return to Bacon-Goodman's formulation
- What happens if we have a "control" group in a later period that is treated in an earlier period?

$$\begin{aligned}\hat{\delta}_{lk}^{2x2} =& ATT_{l,POST(l)} \\ &+\Delta Y_l^0(Post(l), MID) - \Delta Y_k^0(Post(l), MID) \\ &-(ATT_k(Post) - ATT_k(Mid))\end{aligned} \tag{23}$$

- The first line is *what we want*
- The second line is parallel-trends bias
- The third line is bias due to heterogeneity in time!
  - Even with parallel trends, this third line can cause deviations from the true ATT

Enough math, what to do?

- That's enough math. Let's talk about what we can actually do!

- Let's go back to the `castle.dta` dataset
  - Cheng and Hoekstra (2013), *Journal of Human Resources*

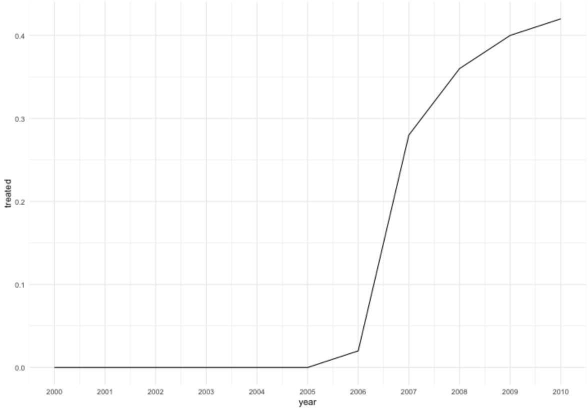
- Let's use the information in Cunningham's book

- The "castle doctrine" laws essentially make lethal force "more" legal
- Recall the changes we made: turn homicide into a rate (per 100,000 people) and take the log:

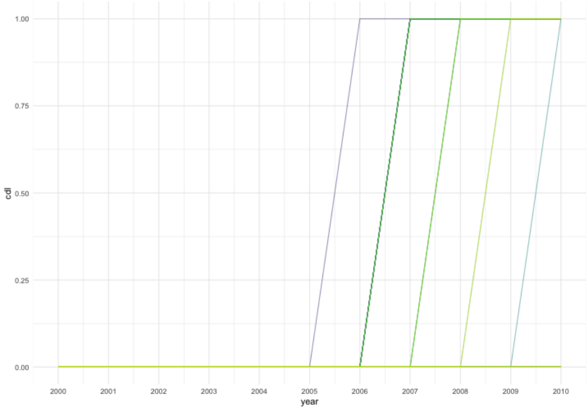
```
library(haven) # to load .dta files
df <- read_dta("castle.dta")
head(df)
# key variables: state, year, cdl ("treatment"), and homicide_c (outcome)
# homicide_c to rate (per 100,000 people)
df$homicide_c <- (df$homicide_c/df$population)*100000
# and log
df$homicide_c <- log(df$homicide_c)
```

- I made some changes to the data, as well
  - I've turned the "treatment" variable (`cdl`) into a dummy variable
- We have the issue we ran into above:
  - Treatment is staggered across time
  - This means that some already-treated units will serve as controls for later-treated units!

Treatment timing



Treatment timing by individual state



Two-way fixed effects with fixest, as simple as possible

```
# state fe, note the weights!
reg1 <- feols(homicide_c ~ cdl + log(population) + unemployrt | state, data = df,
              cluster = c("state"), weights = df$population)
# state and year fe
reg2 <- feols(homicide_c ~ cdl + log(population) + unemployrt | state + year, data = df,
              cluster = c("state"), weights = df$population)
# with state linear trends, note the syntax
reg3 <- feols(homicide_c ~ cdl + log(population) + unemployrt | state + year + state[year], data = df,
              cluster = c("state"), weights = df$population)
# put together
tablech <- etable(reg1, reg2, reg3,
                # standard errors, digits, fit statistics, put SE below coefficients (the norm)
                digits = 3, fitstat = "n", se.below = TRUE,
                depvar = FALSE,
                # rename the variables
                dict = c("cdl" = "Treat", "log(population)" = "Pop. (log)", "unemployrt" = "Unemp. rate"))
tablech <- tablech[-c(12,13),]
tablech[c(7,10),2:4] <- ""
```

Two-way fixed effects with `fixest`

	reg1	reg2	reg3
Treat	0.057	0.089**	0.097*
	(0.034)	(0.033)	(0.036)
Pop. (log)	-0.544**	-0.768*	-0.543
	(0.168)	(0.370)	(0.960)
Unemp. rate	-0.024***	-0.009	-0.003
	(0.004)	(0.010)	(0.011)
Fixed-Effects:			
state	Yes	Yes	Yes
year	No	Yes	Yes
Varying Slopes:			
year (state)	No	No	Yes
Observations	550	550	550

Note: Standard errors clustered at the state level are in parentheses.

* $p < 0.1$, ** $p < 0.05$, *** $p < 0.01$.

- Event studies are a way to look at the effect of treatment over time
  - We can see if the effect is immediate, or if it takes time to “kick in”
  - We can also see whether there are any pre-trends

- Effectively, what we want to do is redefine the time period to be relative to treatment
  - For example, if treatment is introduced in 2005, we want to redefine 2005 as year 0, 2004 as year -1, etc.
  - Let’s do this nrow

```r
# first, find the year of treatment by state
df <- df %>%
      # group by state ("panel" identifier)
      group_by(state) %>%
      mutate(year_treat = ifelse(cdl==1, year, NA),
              # first year with treatment
              year_treat = min(year_treat, na.rm = TRUE),
              # create new time variable called event_year
              event_year = year - year_treat) %>%
              # ungroup
              ungroup()
# note that states NEVER treated are missing
table(df$event_year)
```

```
##
## -Inf  -10   -9   -8   -7   -6   -5   -4   -3   -2   -1    0    1    2    3    4
##  319    1    3    7   20   21   21   21   21   21   21   21   20   18   14    1
```

```r
# now find the average homicide rate by event_year
df_event <- df %>%
            # drop the missings
            filter(event_year>-11) %>%
            group_by(event_year) %>%
            summarize(homicide_c = weighted.mean(homicide_c, weights = population, na.rm = TRUE))
```

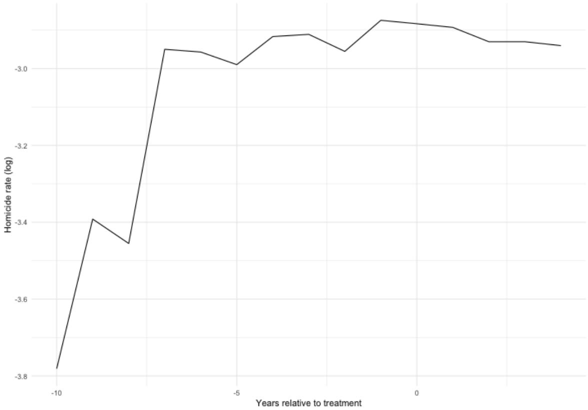
check it looks okay

```
## # A tibble: 11 x 4
##    state    year event_year   cdl
##    <chr>   <dbl>      <dbl> <dbl>
##  1 Alabama  2000         -7     0
##  2 Alabama  2001         -6     0
##  3 Alabama  2002         -5     0
##  4 Alabama  2003         -4     0
##  5 Alabama  2004         -3     0
##  6 Alabama  2005         -2     0
##  7 Alabama  2006         -1     0
##  8 Alabama  2007          0     1
##  9 Alabama  2008          1     1
## 10 Alabama  2009          2     1
## 11 Alabama  2010          3     1
```

```
## # A tibble: 11 x 4
##    state   year event_year   cdl
##    <chr>  <dbl>      <dbl> <dbl>
##  1 Alaska  2000         -7     0
##  2 Alaska  2001         -6     0
##  3 Alaska  2002         -5     0
##  4 Alaska  2003         -4     0
##  5 Alaska  2004         -3     0
##  6 Alaska  2005         -2     0
##  7 Alaska  2006         -1     0
##  8 Alaska  2007          0     1
##  9 Alaska  2008          1     1
## 10 Alaska  2009          2     1
## 11 Alaska  2010          3     1
```

Plot the pure, *means*

```
ggplot(df_event) +
  geom_line(aes(x = event_year, y = homicide_c)) +
  theme_minimal() +
  labs(x = "Years relative to treatment", y = "Homicide rate (log)")
```



What do we really want to see?

- We don't really want the means, though
- What do we want instead?

What do we really want to see?

- We don't really want the means, though
- What do we want instead?
  - We want the *effect* of treatment over time
  - We want to essentially plot *coefficients*

Calculating year-specific coefficients

```
# note that the year BEFORE treatment, -1, IS THE OMITTED CATEGORY
# i() is a fixest-specific way to create factors/dummies
reg1 <- feols(homicide_c ~ i(event_year, ref = -1) + log(population) + unemployrt | state + year, data = df,
              cluster = c("state"), weights = df$population)
reg1$coefficients
```

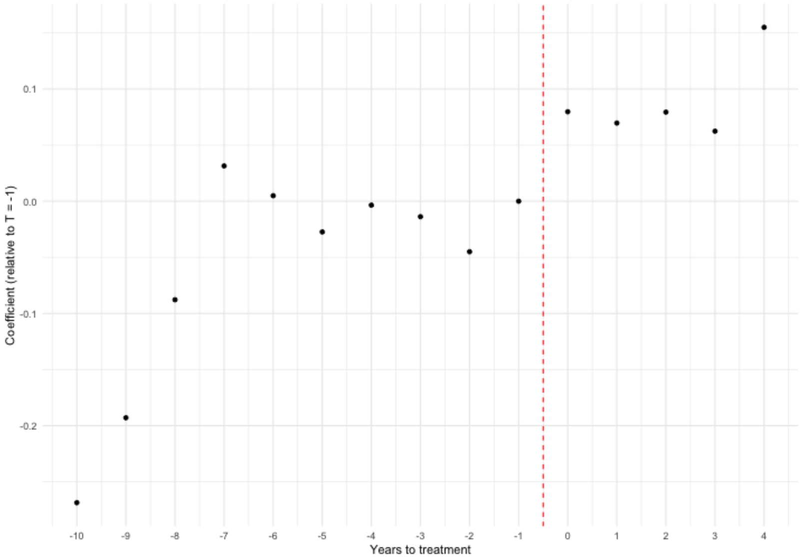
```
## event_year::-10  event_year::-9  event_year::-8  event_year::-7  event_year::-6
##    -0.268714642    -0.193012613    -0.087832067     0.031459588     0.004916894
##  event_year::-5  event_year::-4  event_year::-3  event_year::-2   event_year::0
##    -0.027364799    -0.003498961    -0.013767586    -0.045036279     0.079746416
##   event_year::1   event_year::2   event_year::3   event_year::4 log(population)
##     0.069676912     0.079426337     0.062440521     0.155021385    -0.724793998
##      unemployrt
##    -0.009803301
```

```
# It's a vector. We can extract the coefficients we want by subsetting with []
# get coefficients
coef <- c(reg1$coefficients[1:9], 0, reg1$coefficients[10:14])
# confidence intervals
lower <- c(confint(reg1)[1:9,1], 0, confint(reg1)[10:14,1])
upper <- c(confint(reg1)[1:9,2], 0, confint(reg1)[10:14,2])
# create minimum/maximum
years <- c(-10:4)
```

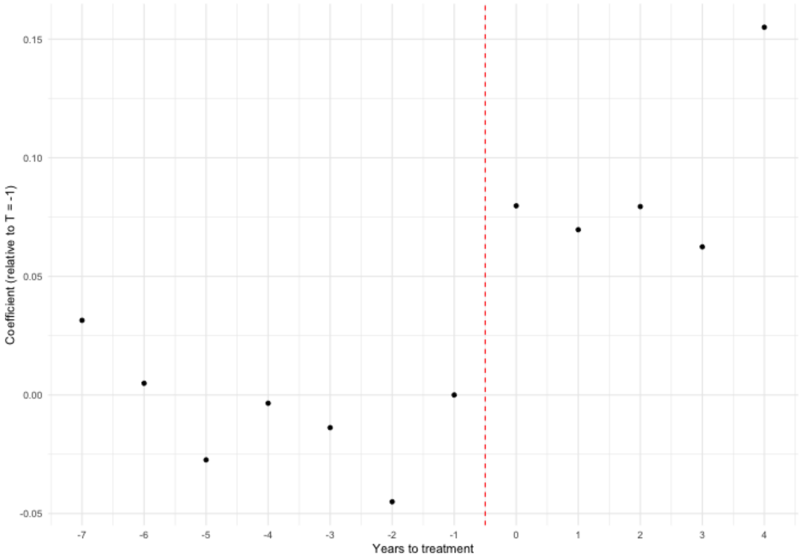
Plot these coefficients using geom_point

```
ggplot() +
  geom_point(aes(x = years, y = coef)) +
  geom_vline(xintercept = -0.5, linetype = "dashed", color = "red") +
  labs(x = "Years to treatment", y = "Coefficient (relative to T = -1)") +
  # change x axis to be every year
  scale_x_continuous(breaks = years) +
  theme_minimal()
```

Plot these coefficients using geom_point



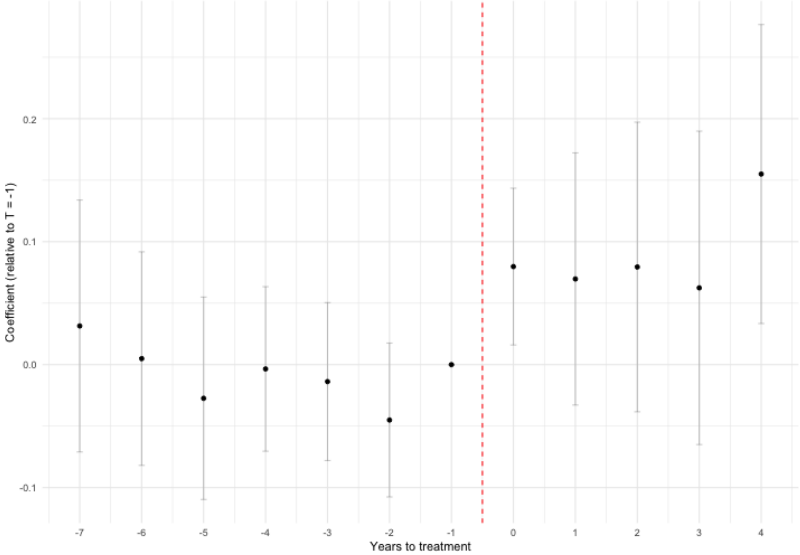
Let's remove the first three years because of small sample sizes



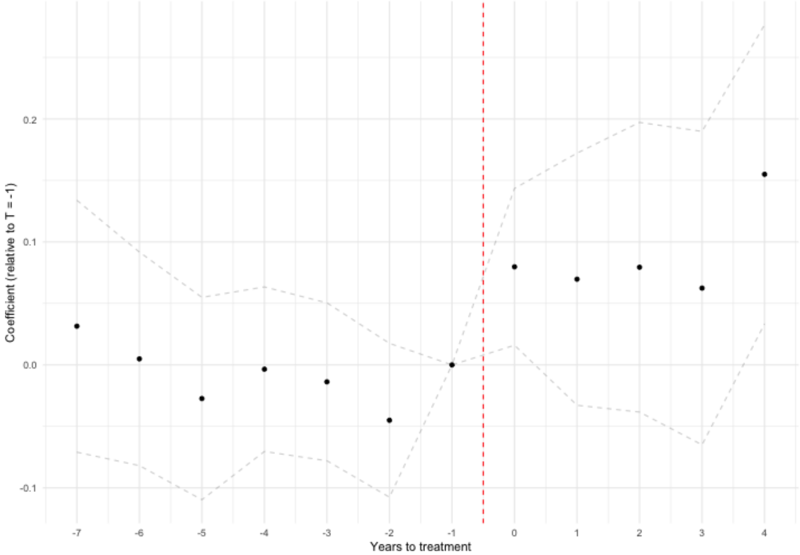
Add the confidence intervals using geom_errorbar

```
ggplot() +
  geom_point(aes(x = years, y = coef)) +
  geom_errorbar(aes(x = years, ymin = lower, ymax = upper), alpha = 0.2, width = 0.1) +
  geom_vline(xintercept = -0.5, linetype = "dashed", color = "red") +
  labs(x = "Years to treatment", y = "Coefficient (relative to T = -1)") +
  # change x axis to be every year
  scale_x_continuous(breaks = years) +
  theme_minimal()
```

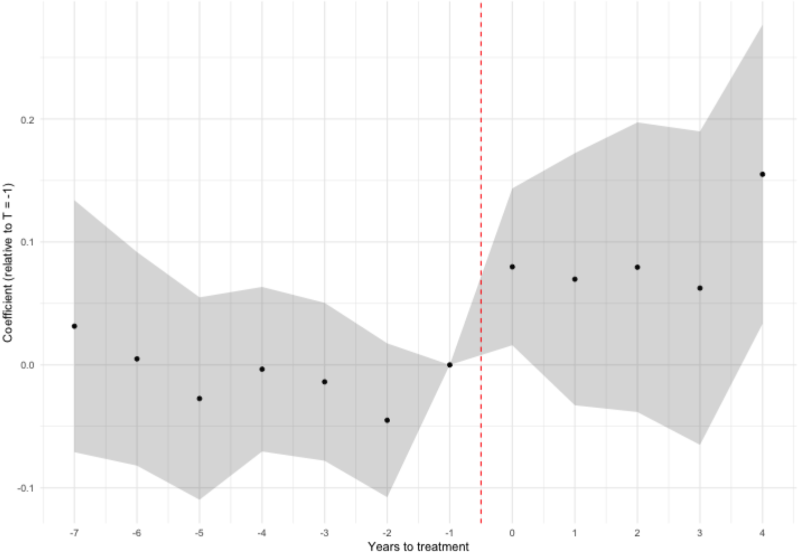
Add the confidence intervals using geom_errorbar



Add the confidence intervals using geom_line, if you prefer



Add the confidence intervals using geom_ribbon, if you prefer



Let's use the "Bacon decomposition" to look at weighting

- This won't allow us to calculate standard errors
- This of this as "diagnostics"
- We can see how different 2x2 cells have different weights, sometimes markedly so
  - We can also see that different cells have different treatment estimates

Let's try the "Bacon decomposition" - Note the treatment variable *must* be binary

```r
library(bacondecomp)

# syntax:
# bacon(formula, data, id_var, time_var, quietly = F)
bacon <- bacon(homicide_c ~ cdl + log(population) + unemployrt, data = df, id_var = "state", time_var = "year", quietly = F)
```

```
##                    type  weight  avg_est
## 1          Both Treated 0.10043 -0.00824
## 2 Treated vs Untreated 0.89957  0.07906
```

```r
bacon$two_by_twos
```

```
##    treated untreated      weight     estimate                 type
## 1     2007     99999 0.611262032  0.059301141 Treated vs Untreated
## 2     2007      2008 0.030235799 -0.001376830         Both Treated
## 3     2007      2010 0.017953382 -0.061066749         Both Treated
## 4     2007      2009 0.026493259  0.053198223         Both Treated
## 5     2006      2007 0.007511240  0.004557765         Both Treated
## 6     2006     99999 0.048373691  0.149043206 Treated vs Untreated
## 7     2006      2008 0.004157732  0.082718843         Both Treated
## 8     2006      2010 0.001570967 -0.048012581         Both Treated
## 9     2006      2009 0.002684252  0.086501842         Both Treated
## 10    2008     99999 0.160751757  0.092373673 Treated vs Untreated
## 11    2008      2010 0.004142796 -0.219504731         Both Treated
## 12    2008      2009 0.004503055 -0.151035583         Both Treated
## 13    2010     99999 0.016782728  0.076684232 Treated vs Untreated
## 14    2009     99999 0.062402696  0.184717537 Treated vs Untreated
## 15    2009      2010 0.001174611 -0.037899937         Both Treated
```

Get the overall average effect by multiplying weights by estimates

```r
bacon <- bacon(homicide_c ~ cdl + log(population) + unemployrt, data = df, id_var = "state", time_var = "year", quietly = F)
```

```
##                    type  weight  avg_est
## 1         Both Treated 0.10043 -0.00824
## 2 Treated vs Untreated 0.89957  0.07906
```

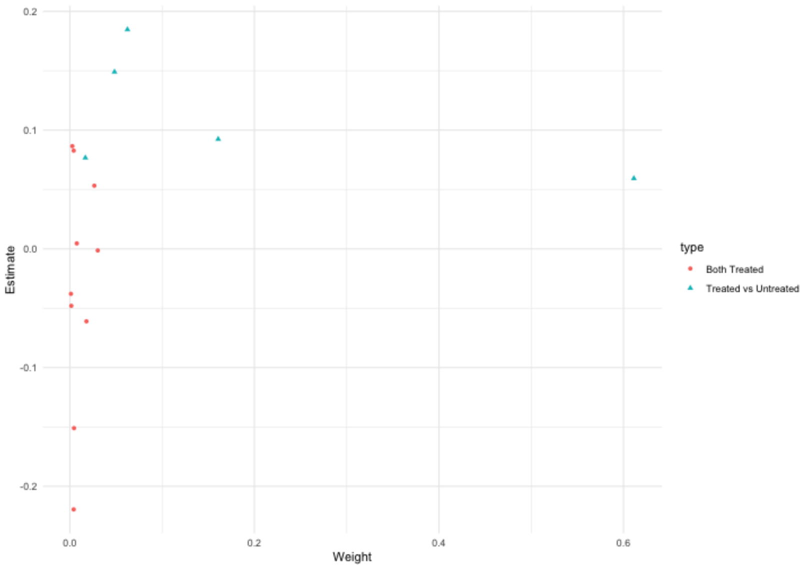
```r
weighted.mean(bacon$two_by_twos$estimate, bacon$two_by_twos$weight)
```

```
## [1] 0.07029371
```

```r
# compare to TWFE estimate
feols(homicide_c ~ cdl + log(population) + unemployrt | state + year, data = df,
              cluster = c("state")) # No weights since Bacon decomp doesn't allow them
```

```
## OLS estimation, Dep. Var.: homicide_c
## Observations: 550
## Fixed-effects: state: 50,  year: 11
## Standard-errors: Clustered (state)
##                  Estimate Std. Error   t value Pr(>|t|)
## cdl               0.073200   0.055921  1.308986 0.196645
## log(population) -1.018959   0.513389 -1.984769 0.052784 .
## unemployrt      -0.013406   0.013728 -0.976562 0.333583
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 0.175397     Adj. R2: 0.900242
##                  Within R2: 0.023689
```

We can plot the average effects for the different groups



Let's estimate effects using the `did2s` function from Kyle Butts

- bacondecomp is good for diagnostics, but we really want to estimate the ATT with standard errors

```r
library(did2s)
```

- yname: the outcome variable
- first_stage: formula for first stage, can include fixed effects and covariates, but do not include treatment variable(s)!
- second_stage: This should be the treatment variable or in the case of event studies, treatment variables.
- treatment: This has to be the 0/1 treatment variable that marks when treatment turns on for a unit. If you suspect anticipation, see note above for accounting for this.
- cluster_var: Which variables to cluster on

Let's estimate effects using the did2s function from Kyle Butts

```
library(did2s)
# note that we can use fixest syntax, with FE and with i()!
# can also add weights
did2s <- did2s(data = df, yname = "homicide_c", first_stage = "log(population) + unemployrt | state + year",
                second_stage = "cdl", treatment = "cdl", cluster_var = "state", weights = "population")
# let's compare it to the vanilla TWFE
twfe <- feols(homicide_c ~ cdl + log(population) + unemployrt | state + year, data = df,
              cluster = c("state"), weights = df$population)
```

And the results

```r
# The "correct" results
did2s
```

```
## OLS estimation, Dep. Var.: homicide_c
## Observations: 550
## Weights: weights_vector
## Standard-errors: Custom
##     Estimate Std. Error t value Pr(>|t|)
## cdl 0.094871   0.037989  2.4973 0.012806 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 262.4   Adj. R2: 0.085672
```

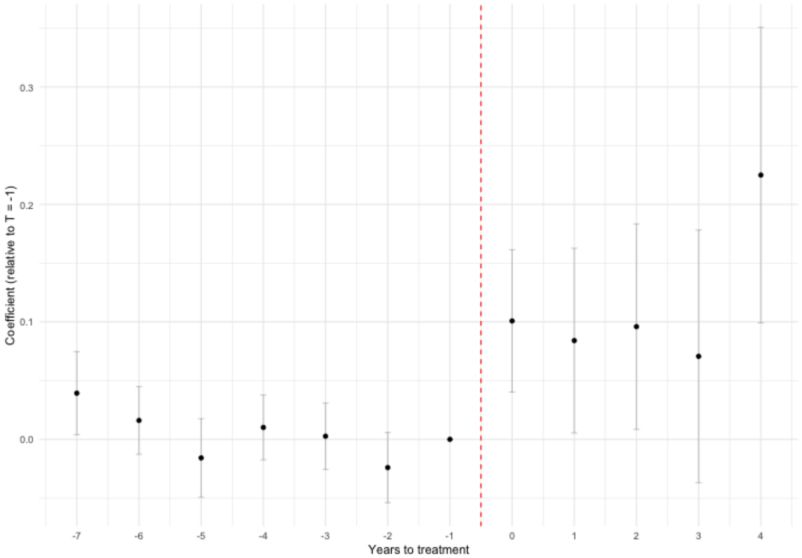
```r
# and the TWFE results?
twfe
```

```
## OLS estimation, Dep. Var.: homicide_c
## Observations: 550
## Weights: df$population
## Fixed-effects: state: 50,  year: 11
## Standard-errors: Clustered (state)
##                  Estimate Std. Error   t value  Pr(>|t|)
## cdl              0.088939   0.032653  2.723783 0.0089189 **
## log(population) -0.767701   0.370173 -2.073895 0.0433658 *
## unemployrt      -0.009254   0.010207 -0.906587 0.3690621
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## RMSE: 252.9     Adj. R2: 0.927248
##               Within R2: 0.048831
```

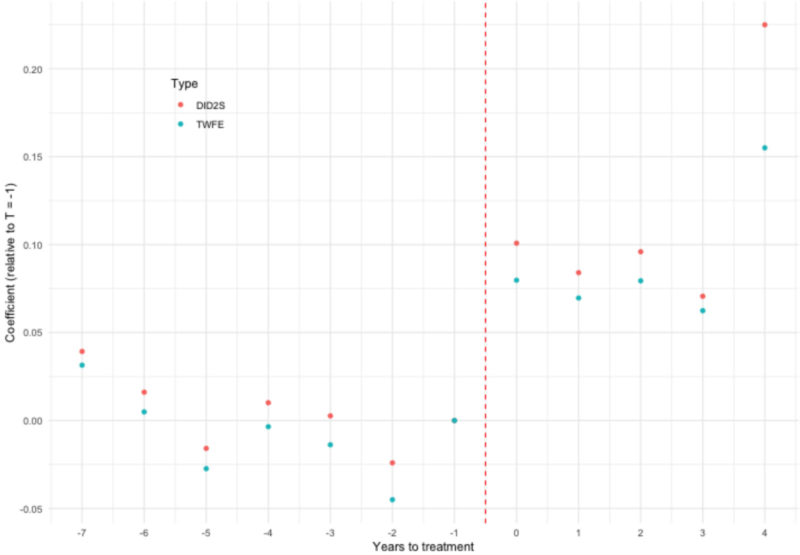
We can also estimate the event study this way!

```r
# Estimate
did2s <- did2s(data = df, yname = "homicide_c", first_stage = "log(population) + unemployrt | state + year",
               second_stage = "i(event_year, ref = -1)", treatment = "cdl", cluster_var = "state",
               weights = "population")
coefficients <- c(did2s$coefficients[2:10], 0, did2s$coefficients[11:15])
# confidence intervals
lower <- c(confint(did2s)[2:10,1], 0, confint(did2s)[11:15,1])
upper <- c(confint(did2s)[2:10,2], 0, confint(did2s)[11:15,2])
# plot estimates
ggplot() +
  geom_point(aes(x = c(-10:4), y = coefficients)) +
  geom_errorbar(aes(x = c(-10:4), ymin = lower, ymax = upper), alpha = 0.2, width = 0.1) +
  geom_vline(xintercept = -0.5, linetype = "dashed", color = "red") +
  labs(x = "Years to treatment", y = "Coefficient (relative to T = -1)") +
  # change x axis to be every year
  scale_x_continuous(breaks = years) +
  theme_minimal()
```

Some small changes to remove first three years



Compared to TWFE?



Wrapping up TWFE

- We've learned how to estimate two-way fixed effects models with `fixest`
  - We've also learned how they can be biased if treatment is staggered
- We saw how to use `did2s` to estimate the ATT
  - We also saw how to use `bacondecomp` to look at the weights
- A lingering question: TWFE with continuous treatment variables
  - Callaway et al. (2021) and Chaisemartin and D'Haultfœuille (2023) have some ideas
  - I haven't seen reliable packages for these, though
- IVs with TWFE?

Increasing cigarette taxes in California (in 1988)