HW6

Erik Andersen

2023-12-01

```
here::i_am("HW6/HW6.Rmd")
```

```
## here() starts at /Users/erikandersen/Documents/Classes/ECON 587
```

```
# Load packages
pacman::p_load(tidyverse, magrittr, estimatr, broom, ri2)
```

```
# Load data
vote_df = haven::read_dta(here::here("HW6", "data", "GriffithNoonen2022_Econ587.dta"))

# Clean names
vote_df = janitor::clean_names(vote_df)

# Add variables of interest for Did.
# Post = after 2017 which is the year of interest
# Seattle = in seattle
# Treat = interaction of post and seattle
# city_cycle = interaction of city and cycle
vote_df = vote_df |>
  mutate(post = if_else(election_year>= 2017, 1, 0),
                    seattle = if_else(city == 'Seattle', 1, 0),
                    treatment = post * seattle,
                    city = as.factor(city),
                    cycle = as.factor(cycle),
                    city_cycle = city:cycle) |>
  select(post, seattle, treatment, everything())
```

Question 1

```

# did with regular SE's
did_reg_classical = lm_robust(candidates_ballot ~ post + seattle + treatment + at_large * special,
                              vote_df,
                              se_type = 'classical')
# Extract standard error of coefficient of interest
tidy(did_reg_classical) |> select(term, std.error, p.value) |> filter(term == 'treatment') # 0.958
```

a)

```
##        term std.error  p.value
## 1 treatment    0.9546 0.002916
```

```
# Rerun with HC robust SEs
did_reg_hc = lm_robust(candidates_ballot ~ post + seattle + treatment + at_large * special,
                              vote_df,
```

```
                              se_type = 'stata')
# Extract standard error again
tidy(did_reg_hc) |> select(term, std.error, p.value) |> filter(term == 'treatment') # 0.987
```

```
##        term std.error  p.value
## 1 treatment     0.985 0.003913
```

```
rm(did_reg_hc)
```

```
# Construct residuals for both regressions. The lm_robust function doesn't calculat these automatically
vote_df = vote_df |> mutate(resids = candidates_ballot - did_reg_classical$fitted.values)

# Naive test for heteroskedasticity
# resids^2 ~ treatment
reg_hetero = lm_robust(I(resids^2) ~ treatment,
                       vote_df,
                       se_type = 'classical')

tidy(reg_hetero)
```

b)

```
##          term estimate std.error statistic   p.value conf.low conf.high  df
## 1 (Intercept)    5.991    0.8667    6.9117 1.096e-11    4.289     7.692 686
## 2   treatment    1.627    7.5780    0.2147 8.300e-01  -13.252    16.506 686
##       outcome
## 1 I(resids^2)
## 2 I(resids^2)
```

```
rm(reg_hetero)
rm(did_reg_classical)
```

```
# Cluster by city_cycle
reg_city_cycle = lm_robust(candidates_ballot ~ post + seattle + treatment + at_large * special,
                    vote_df,
                    clusters = city_cycle,
                    se_type = 'CR2') # CR2 is stata standard errors

tidy(reg_city_cycle)
```

c)

```
##              term estimate std.error statistic   p.value conf.low conf.high
## 1      (Intercept)   3.4816    0.1671    20.839 5.447e-37  3.14991    3.8134
## 2            post   0.4161    0.2447     1.701 9.717e-02 -0.07919    0.9113
## 3         seattle   1.3031    0.3317     3.928 3.706e-03  0.54871    2.0576
## 4       treatment   2.8517    0.5370     5.311 4.185e-02  0.28117    5.4222
## 5        at_large  -1.2364    0.1856    -6.663 1.354e-07 -1.61390   -0.8590
## 6         special   2.3456    0.6512     3.602 1.024e-03  1.02072    3.6704
## 7 at_large:special  -2.4827    0.8282    -2.998 1.198e-02 -4.30288   -0.6624
##       df           outcome
## 1 93.911 candidates_ballot
## 2 38.066 candidates_ballot
```

```
## 3  8.698 candidates_ballot
## 4  1.803 candidates_ballot
## 5 33.173 candidates_ballot
## 6 33.041 candidates_ballot
## 7 11.135 candidates_ballot
```

```
rm(reg_city_cycle)
```

```
# Cluster by only city
reg_city = lm_robust(candidates_ballot ~ post + seattle + treatment + at_large * special,
                     vote_df,
                     clusters = city,
                     se_type = 'CR2') # CR2 is stata standard errors
tidy(reg_city)
```

d)

```
##               term estimate std.error statistic   p.value conf.low conf.high
## 1      (Intercept)   3.4816    0.3273    10.636 4.705e-07   2.7595    4.2038
## 2             post   0.4161    0.2914     1.428 1.787e-01  -0.2185    1.0507
## 3          seattle   1.3031    0.1786     7.296 1.341e-02   0.6065    1.9998
## 4        treatment   2.8517    0.2368    12.043 3.353e-07   2.3228    3.3806
## 5         at_large  -1.2364    0.3183    -3.885 3.551e-02  -2.3060   -0.1669
## 6          special   2.3456    1.0675     2.197 7.167e-02  -0.2850    4.9761
## 7 at_large:special  -2.4827    1.1434    -2.171 1.504e-01  -6.9967    2.0314
##       df           outcome
## 1 10.790 candidates_ballot
## 2 12.046 candidates_ballot
## 3  2.233 candidates_ballot
## 4  9.820 candidates_ballot
## 5  2.740 candidates_ballot
## 6  5.830 candidates_ballot
## 7  2.201 candidates_ballot
```

```
rm(reg_city)

# Cluster by only cycle
reg_cycle = lm_robust(candidates_ballot ~ post + seattle + treatment + at_large * special,
                     vote_df,
                     clusters = cycle,
                     se_type = 'CR2') # CR2 is stata standard errors
tidy(reg_cycle)
```

```
##               term estimate std.error statistic   p.value conf.low conf.high
## 1      (Intercept)   3.4816   0.09349    37.240 9.346e-10   3.2633    3.6999
## 2             post   0.4161   0.09950     4.182 8.091e-02  -0.1607    0.9929
## 3          seattle   1.3031   0.28976     4.497 4.121e-03   0.5940    2.0123
## 4        treatment   2.8517   0.42981     6.635 3.931e-02   0.4170    5.2864
## 5         at_large  -1.2364   0.09288   -13.313 7.435e-07  -1.4496   -1.0233
## 6          special   2.3456   0.91581     2.561 3.435e-02   0.2233    4.4678
## 7 at_large:special  -2.4827   1.24859    -1.988 9.765e-02  -5.5965    0.6312
##      df           outcome
## 1 7.469 candidates_ballot
## 2 1.541 candidates_ballot
## 3 5.997 candidates_ballot
```

```
## 4 1.568 candidates_ballot
## 5 8.231 candidates_ballot
## 6 7.780 candidates_ballot
## 7 5.566 candidates_ballot
```

```
rm(reg_cycle)
```

```
# Redo the earlier parts with the following new specifcation
# candidates_ballot = cycle_fixed_effct + city_fixed_effect + treatment

# First redo part a
# Classical errors
reg_two_way_classic = lm_robust(candidates_ballot ~ cycle + city + treatment + at_large * special,
                                vote_df,
                                se_type = "classical")
tidy(reg_two_way_classic)
```

e)

```
##                 term estimate std.error statistic   p.value conf.low conf.high
## 1        (Intercept)  2.99313    0.7685   3.89496 1.082e-04  1.48420    4.5021
## 2          cycle2003 -0.10968    0.4056  -0.27042 7.869e-01 -0.90613    0.6868
## 3          cycle2005 -0.38077    0.4036  -0.94347 3.458e-01 -1.17322    0.4117
## 4          cycle2007 -0.55282    0.4088  -1.35222 1.768e-01 -1.35557    0.2499
## 5          cycle2009  0.09001    0.4125   0.21823 8.273e-01 -0.71990    0.8999
## 6          cycle2011 -0.30115    0.4073  -0.73934 4.600e-01 -1.10097    0.4987
## 7          cycle2013 -0.26897    0.4105  -0.65516 5.126e-01 -1.07510    0.5372
## 8          cycle2015 -0.30810    0.3993  -0.77164 4.406e-01 -1.09211    0.4759
## 9          cycle2017  0.28212    0.4202   0.67142 5.022e-01 -0.54295    1.1072
## 10         cycle2019  0.09605    0.4092   0.23474 8.145e-01 -0.70742    0.8995
## 11       cityEverett -0.01794    0.5452  -0.03290 9.738e-01 -1.08842    1.0525
## 12        cityFresno  0.15972    0.8165   0.19562 8.450e-01 -1.44344    1.7629
## 13          cityKent -0.55572    0.8187  -0.67883 4.975e-01 -2.16320    1.0518
## 14    cityLong Beach  0.55547    0.8027   0.69200 4.892e-01 -1.02069    2.1316
## 15   cityLos Angeles  0.90663    0.7645   1.18596 2.361e-01 -0.59445    2.4077
## 16       cityOakland  0.84596    0.7548   1.12083 2.628e-01 -0.63606    2.3280
## 17    citySacramento -0.19787    0.8034  -0.24631 8.055e-01 -1.77532    1.3796
## 18     citySan Diego  1.61675    0.7952   2.03318 4.243e-02  0.05536    3.1781
## 19 citySan Francisco  2.51049    0.7802   3.21763 1.356e-03  0.97846    4.0425
## 20      citySan Jose  0.77741    0.7849   0.99050 3.223e-01 -0.76372    2.3185
## 21       citySeattle  1.56212    0.5507   2.83658 4.700e-03  0.48077    2.6435
## 22       citySpokane  0.94533    0.7616   1.24127 2.149e-01 -0.55009    2.4408
## 23        cityTacoma  0.02720    0.8097   0.03359 9.732e-01 -1.56277    1.6172
## 24     cityVancouver -0.15675    0.8290  -0.18909 8.501e-01 -1.78451    1.4710
## 25         treatment  3.23227    0.9819   3.29190 1.048e-03  1.30427    5.1603
## 26          at_large -0.66214    0.6007  -1.10221 2.708e-01 -1.84172    0.5174
## 27           special  2.09710    0.3986   5.26179 1.931e-07  1.31451    2.8797
## 28  at_large:special -2.13876    0.9733  -2.19747 2.833e-02 -4.04987   -0.2277
##     df           outcome
## 1  660 candidates_ballot
## 2  660 candidates_ballot
## 3  660 candidates_ballot
## 4  660 candidates_ballot
## 5  660 candidates_ballot
```

```
## 6  660 candidates_ballot
## 7  660 candidates_ballot
## 8  660 candidates_ballot
## 9  660 candidates_ballot
## 10 660 candidates_ballot
## 11 660 candidates_ballot
## 12 660 candidates_ballot
## 13 660 candidates_ballot
## 14 660 candidates_ballot
## 15 660 candidates_ballot
## 16 660 candidates_ballot
## 17 660 candidates_ballot
## 18 660 candidates_ballot
## 19 660 candidates_ballot
## 20 660 candidates_ballot
## 21 660 candidates_ballot
## 22 660 candidates_ballot
## 23 660 candidates_ballot
## 24 660 candidates_ballot
## 25 660 candidates_ballot
## 26 660 candidates_ballot
## 27 660 candidates_ballot
## 28 660 candidates_ballot
```

```r
rm(reg_two_way_classic)
# Het robust errors
reg_two_way_hc = lm_robust(candidates_ballot ~ cycle + city + treatment + at_large * special,
                           vote_df,
                           se_type = "stata")
tidy(reg_two_way_hc)
```

```
##                 term estimate std.error statistic   p.value conf.low conf.high
## 1        (Intercept)  2.99313    0.5418   5.52402 4.770e-08  1.92919    4.0571
## 2          cycle2003 -0.10968    0.4249  -0.25812 7.964e-01 -0.94408    0.7247
## 3          cycle2005 -0.38077    0.3953  -0.96332 3.357e-01 -1.15689    0.3954
## 4          cycle2007 -0.55282    0.3551  -1.55666 1.200e-01 -1.25014    0.1445
## 5          cycle2009  0.09001    0.4518   0.19924 8.421e-01 -0.79708    0.9771
## 6          cycle2011 -0.30115    0.3656  -0.82362 4.105e-01 -1.01912    0.4168
## 7          cycle2013 -0.26897    0.3794  -0.70897 4.786e-01 -1.01391    0.4760
## 8          cycle2015 -0.30810    0.3820  -0.80657 4.202e-01 -1.05816    0.4420
## 9          cycle2017  0.28212    0.4427   0.63722 5.242e-01 -0.58723    1.1515
## 10         cycle2019  0.09605    0.3746   0.25639 7.977e-01 -0.63957    0.8317
## 11       cityEverett -0.01794    0.1949  -0.09200 9.267e-01 -0.40073    0.3649
## 12        cityFresno  0.15972    0.5906   0.27045 7.869e-01 -0.99989    1.3193
## 13          cityKent -0.55572    0.5071  -1.09586 2.735e-01 -1.55148    0.4400
## 14    cityLong Beach  0.55547    0.5529   1.00473 3.154e-01 -0.53009    1.6410
## 15   cityLos Angeles  0.90663    0.6072   1.49312 1.359e-01 -0.28566    2.0989
## 16       cityOakland  0.84596    0.5111   1.65524 9.835e-02 -0.15758    1.8495
## 17    citySacramento -0.19787    0.5360  -0.36919 7.121e-01 -1.25027    0.8545
## 18     citySan Diego  1.61675    0.5977   2.70508 7.005e-03  0.44318    2.7903
## 19 citySan Francisco  2.51049    0.7289   3.44426 6.089e-04  1.07927    3.9417
## 20      citySan Jose  0.77741    0.5758   1.35007 1.775e-01 -0.35327    1.9081
## 21       citySeattle  1.56212    0.2923   5.34403 1.253e-07  0.98815    2.1361
## 22       citySpokane  0.94533    0.4581   2.06370 3.944e-02  0.04587    1.8448
## 23        cityTacoma  0.02720    0.5252   0.05179 9.587e-01 -1.00398    1.0584
```

```
## 24     cityVancouver -0.15675    0.5296  -0.29600 7.673e-01 -1.19661    0.8831
## 25         treatment  3.23227    0.9800   3.29835 1.025e-03  1.30804    5.1565
## 26          at_large -0.66214    0.4567  -1.44979 1.476e-01 -1.55892    0.2346
## 27           special  2.09710    0.6198   3.38358 7.578e-04  0.88011    3.3141
## 28  at_large:special -2.13876    0.8023  -2.66572 7.870e-03 -3.71417   -0.5634
##     df           outcome
## 1  660 candidates_ballot
## 2  660 candidates_ballot
## 3  660 candidates_ballot
## 4  660 candidates_ballot
## 5  660 candidates_ballot
## 6  660 candidates_ballot
## 7  660 candidates_ballot
## 8  660 candidates_ballot
## 9  660 candidates_ballot
## 10 660 candidates_ballot
## 11 660 candidates_ballot
## 12 660 candidates_ballot
## 13 660 candidates_ballot
## 14 660 candidates_ballot
## 15 660 candidates_ballot
## 16 660 candidates_ballot
## 17 660 candidates_ballot
## 18 660 candidates_ballot
## 19 660 candidates_ballot
## 20 660 candidates_ballot
## 21 660 candidates_ballot
## 22 660 candidates_ballot
## 23 660 candidates_ballot
## 24 660 candidates_ballot
## 25 660 candidates_ballot
## 26 660 candidates_ballot
## 27 660 candidates_ballot
## 28 660 candidates_ballot
```

```
rm(reg_two_way_hc)

# Redo part c
# Clustered errors at city and cycle level
reg_two_way_city_cycle = lm_robust(candidates_ballot ~ cycle + city + treatment + at_large * special,
                    vote_df,
                    clusters = city_cycle,
                    se_type = 'CR2') # CR2 is stata standard errors

tidy(reg_two_way_city_cycle)
```

```
##                 term estimate std.error statistic   p.value conf.low conf.high
## 1        (Intercept)  2.99313    0.6747   4.43651 0.0003004   1.5784    4.4078
## 2          cycle2003 -0.10968    0.5780  -0.18977 0.8510046  -1.2993    1.0800
## 3          cycle2005 -0.38077    0.4228  -0.90048 0.3769442  -1.2542    0.4926
## 4          cycle2007 -0.55282    0.4279  -1.29192 0.2083597  -1.4347    0.3291
## 5          cycle2009  0.09001    0.5767   0.15609 0.8772564  -1.0996    1.2796
## 6          cycle2011 -0.30115    0.4249  -0.70870 0.4851368  -1.1768    0.5745
## 7          cycle2013 -0.26897    0.4652  -0.57820 0.5685008  -1.2289    0.6910
## 8          cycle2015 -0.30810    0.4356  -0.70725 0.4858319  -1.2044    0.5882
```

```
## 9          cycle2017  0.28212    0.4389   0.64281 0.5263061  -0.6226    1.1869
## 10         cycle2019  0.09605    0.4469   0.21492 0.8315953  -0.8250    1.0171
## 11       cityEverett -0.01794    0.2653  -0.06761 0.9468600  -0.5761    0.5403
## 12        cityFresno  0.15972    0.6828   0.23392 0.8173684  -1.2624    1.5819
## 13          cityKent -0.55572    0.6408  -0.86726 0.3959808  -1.8915    0.7800
## 14    cityLong Beach  0.55547    0.7010   0.79236 0.4374621  -0.9070    2.0179
## 15   cityLos Angeles  0.90663    0.7201   1.25896 0.2245400  -0.6092    2.4224
## 16       cityOakland  0.84596    0.6162   1.37295 0.1878130  -0.4552    2.1471
## 17    citySacramento -0.19787    0.6729  -0.29407 0.7717735  -1.6026    1.2068
## 18     citySan Diego  1.61675    0.7166   2.25609 0.0357711   0.1191    3.1144
## 19 citySan Francisco  2.51049    0.9169   2.73804 0.0133611   0.5866    4.4343
## 20      citySan Jose  0.77741    0.7045   1.10343 0.2838168  -0.6987    2.2535
## 21       citySeattle  1.56212    0.4066   3.84224 0.0013598   0.7026    2.4216
## 22       citySpokane  0.94533    0.5773   1.63744 0.1196765  -0.2715    2.1622
## 23        cityTacoma  0.02720    0.6524   0.04169 0.9671543  -1.3329    1.3873
## 24     cityVancouver -0.15675    0.6241  -0.25116 0.8041179  -1.4542    1.1407
## 25         treatment  3.23227    0.7230   4.47036 0.0419910   0.2772    6.1873
## 26          at_large -0.66214    0.5701  -1.16154 0.2790835  -1.9779    0.6537
## 27           special  2.09710    0.6401   3.27644 0.0024636   0.7953    3.3989
## 28  at_large:special -2.13876    0.8301  -2.57654 0.0253148  -3.9604   -0.3171
##        df           outcome
## 1  18.493 candidates_ballot
## 2  25.307 candidates_ballot
## 3  23.653 candidates_ballot
## 4  24.665 candidates_ballot
## 5  24.231 candidates_ballot
## 6  24.735 candidates_ballot
## 7  24.079 candidates_ballot
## 8  25.483 candidates_ballot
## 9  24.537 candidates_ballot
## 10 24.662 candidates_ballot
## 11 17.619 candidates_ballot
## 12 20.473 candidates_ballot
## 13 20.207 candidates_ballot
## 14 19.969 candidates_ballot
## 15 17.542 candidates_ballot
## 16 16.804 candidates_ballot
## 17 19.757 candidates_ballot
## 18 19.423 candidates_ballot
## 19 18.328 candidates_ballot
## 20 18.729 candidates_ballot
## 21 16.570 candidates_ballot
## 22 17.225 candidates_ballot
## 23 20.186 candidates_ballot
## 24 21.139 candidates_ballot
## 25  2.114 candidates_ballot
## 26  7.956 candidates_ballot
## 27 33.247 candidates_ballot
## 28 11.270 candidates_ballot
```

```r
# Redo part d
# Cluster by city only
reg_two_way_city = lm_robust(candidates_ballot ~ cycle + city + treatment + at_large * special,
                    vote_df,
```

```
                    clusters = city,
                    se_type = 'CR2') # CR2 is stata standard errors
tidy(reg_two_way_city)
```

```
##                 term estimate std.error statistic  p.value conf.low conf.high
## 1        (Intercept)  2.99313    0.8477   3.53099 0.041997   0.2103    5.7759
## 2          cycle2003 -0.10968    0.5104  -0.21490 0.833232  -1.2142    0.9948
## 3          cycle2005 -0.38077    0.4622  -0.82379 0.426155  -1.3880    0.6265
## 4          cycle2007 -0.55282    0.5030  -1.09905 0.292491  -1.6441    0.5384
## 5          cycle2009  0.09001    0.6183   0.14558 0.886622  -1.2542    1.4343
## 6          cycle2011 -0.30115    0.4421  -0.68120 0.508200  -1.2604    0.6581
## 7          cycle2013 -0.26897    0.4439  -0.60591 0.555699  -1.2346    0.6966
## 8          cycle2015 -0.30810    0.4445  -0.69308 0.500591  -1.2696    0.6534
## 9          cycle2017  0.28212    0.3161   0.89246 0.389235  -0.4046    0.9689
## 10         cycle2019  0.09605    0.4129   0.23264 0.819809  -0.7997    0.9918
## 11       cityEverett -0.01794    0.0304  -0.58993 0.625797  -0.1793    0.1434
## 12        cityFresno  0.15972    0.7601   0.21013 0.852774  -3.0643    3.3837
## 13          cityKent -0.55572    0.7668  -0.72471 0.543381  -3.8280    2.7166
## 14    cityLong Beach  0.55547    0.7743   0.71736 0.544940  -2.6481    3.7590
## 15   cityLos Angeles  0.90663    0.7600   1.19299 0.351544  -2.2557    4.0690
## 16       cityOakland  0.84596    0.6748   1.25366 0.334843  -2.0114    3.7033
## 17    citySacramento -0.19787    0.7620  -0.25969 0.819079  -3.4313    3.0355
## 18     citySan Diego  1.61675    0.7660   2.11068 0.165373  -1.5853    4.8188
## 19 citySan Francisco  2.51049    0.7669   3.27346 0.079399  -0.7150    5.7360
## 20      citySan Jose  0.77741    0.7613   1.02122 0.412731  -2.4361    3.9910
## 21       citySeattle  1.56212    0.1568   9.96305 0.001897   1.0712    2.0530
## 22       citySpokane  0.94533    0.6577   1.43744 0.285098  -1.8371    3.7277
## 23        cityTacoma  0.02720    0.7623   0.03568 0.974731  -3.2003    3.2547
## 24     cityVancouver -0.15675    0.7618  -0.20576 0.855584  -3.3542    3.0407
## 25         treatment  3.23227    0.5093   6.34631 0.003085   1.8220    4.6425
## 26          at_large -0.66214    0.7584  -0.87309 0.474055  -3.9008    2.5765
## 27           special  2.09710    1.1008   1.90512 0.108263  -0.6351    4.8293
## 28  at_large:special -2.13876    1.1735  -1.82262 0.198568  -6.7809    2.5034
##        df           outcome
## 1   2.845 candidates_ballot
## 2  12.790 candidates_ballot
## 3  11.979 candidates_ballot
## 4  12.485 candidates_ballot
## 5  12.239 candidates_ballot
## 6  12.471 candidates_ballot
## 7  12.181 candidates_ballot
## 8  12.854 candidates_ballot
## 9  12.327 candidates_ballot
## 10 12.486 candidates_ballot
## 11  1.650 candidates_ballot
## 12  2.030 candidates_ballot
## 13  2.017 candidates_ballot
## 14  2.086 candidates_ballot
## 15  2.073 candidates_ballot
## 16  2.034 candidates_ballot
## 17  2.029 candidates_ballot
## 18  2.062 candidates_ballot
## 19  2.049 candidates_ballot
## 20  2.041 candidates_ballot
```

```
## 21  3.090 candidates_ballot
## 22  2.036 candidates_ballot
## 23  2.034 candidates_ballot
## 24  2.054 candidates_ballot
## 25  4.028 candidates_ballot
## 26  2.016 candidates_ballot
## 27  5.668 candidates_ballot
## 28  2.195 candidates_ballot
```

```r
# Cluster by only cycle
reg_two_way_cycle = lm_robust(candidates_ballot ~ cycle + city + treatment + at_large * special,
                    vote_df,
                    clusters = cycle,
                    se_type = 'CR2') # CR2 is stata standard errors
tidy(reg_two_way_cycle)
```

```
##                 term estimate std.error statistic   p.value conf.low conf.high
## 1        (Intercept)  2.99313   0.53039   5.64331 2.153e-03  1.64434   4.34192
## 2          cycle2003 -0.10968   0.05259  -2.08556 6.846e-02 -0.22973   0.01036
## 3          cycle2005 -0.38077   0.01618 -23.53738 1.657e-06 -0.42181  -0.33972
## 4          cycle2007 -0.55282   0.06098  -9.06617 1.148e-05 -0.69191  -0.41373
## 5          cycle2009  0.09001   0.04652   1.93489 8.967e-02 -0.01759   0.19761
## 6          cycle2011 -0.30115   0.06156  -4.89196 1.020e-03 -0.44173  -0.16057
## 7          cycle2013 -0.26897   0.03646  -7.37807 1.330e-04 -0.35470  -0.18324
## 8          cycle2015 -0.30810   0.05532  -5.56939 4.524e-03 -0.45936  -0.15684
## 9          cycle2017  0.28212   0.06891   4.09404 5.748e-03  0.11550   0.44875
## 10         cycle2019  0.09605   0.03834   2.50530 1.837e-01 -0.16741   0.35952
## 11       cityEverett -0.01794   0.33557  -0.05345 9.586e-01 -0.77872   0.74285
## 12        cityFresno  0.15972   0.71201   0.22432 8.297e-01 -1.56733   1.88676
## 13          cityKent -0.55572   0.65160  -0.85286 4.266e-01 -2.15175   1.04030
## 14    cityLong Beach  0.55547   0.47865   1.16048 2.884e-01 -0.60515   1.71608
## 15   cityLos Angeles  0.90663   0.61352   1.47775 1.923e-01 -0.61282   2.42607
## 16       cityOakland  0.84596   0.63331   1.33577 2.320e-01 -0.71970   2.41162
## 17    citySacramento -0.19787   0.64983  -0.30450 7.710e-01 -1.78508   1.38934
## 18     citySan Diego  1.61675   0.62646   2.58076 4.189e-02  0.08228   3.15121
## 19 citySan Francisco  2.51049   1.00187   2.50582 4.791e-02  0.03233   4.98866
## 20      citySan Jose  0.77741   0.41295   1.88260 1.099e-01 -0.23891   1.79374
## 21       citySeattle  1.56212   0.37210   4.19807 2.716e-03  0.71062   2.41361
## 22       citySpokane  0.94533   0.56199   1.68213 1.443e-01 -0.43506   2.32573
## 23        cityTacoma  0.02720   0.63632   0.04275 9.673e-01 -1.52183   1.57623
## 24     cityVancouver -0.15675   0.53817  -0.29127 7.801e-01 -1.45687   1.14337
## 25         treatment  3.23227   0.65067   4.96764 5.136e-02 -0.05319   6.51773
## 26          at_large -0.66214   0.46316  -1.42962 2.330e-01 -2.00164   0.67737
## 27           special  2.09710   0.91738   2.28596 5.246e-02 -0.02855   4.22275
## 28  at_large:special -2.13876   1.23447  -1.73254 1.378e-01 -5.21770   0.94018
##       df           outcome
## 1  5.186 candidates_ballot
## 2  8.502 candidates_ballot
## 3  5.228 candidates_ballot
## 4  8.536 candidates_ballot
## 5  7.865 candidates_ballot
## 6  8.476 candidates_ballot
## 7  7.194 candidates_ballot
## 8  4.162 candidates_ballot
## 9  6.311 candidates_ballot
```

```
## 10 1.372 candidates_ballot
## 11 8.871 candidates_ballot
## 12 6.226 candidates_ballot
## 13 5.975 candidates_ballot
## 14 6.235 candidates_ballot
## 15 5.717 candidates_ballot
## 16 5.757 candidates_ballot
## 17 6.045 candidates_ballot
## 18 5.975 candidates_ballot
## 19 5.744 candidates_ballot
## 20 5.860 candidates_ballot
## 21 8.371 candidates_ballot
## 22 5.907 candidates_ballot
## 23 6.130 candidates_ballot
## 24 6.336 candidates_ballot
## 25 1.720 candidates_ballot
## 26 3.629 candidates_ballot
## 27 7.786 candidates_ballot
## 28 5.564 candidates_ballot
```

```r
# Cameron Miller two way clustering decomposition
# Variance for clustering only by city
var_city = reg_two_way_city$std.error["treatment"]^2

# Variance for clustering only by cycle
var_cycle = reg_two_way_cycle$std.error["treatment"]^2

# Variance for clustering at the intersection of city and cycle
var_intersection = reg_two_way_city_cycle$std.error["treatment"]^2

# Calculate two way clustering variance
se_two_way = sqrt(var_city + var_cycle - var_intersection)

rm(reg_two_way_city, reg_two_way_cycle, reg_two_way_city_cycle,
        var_city, var_cycle, var_intersection, se_two_way)
```

f)

Question 2

```r
# Define randomization procedure
# This is all we need to do since we're doing a naive estimate
randomization_naive = declare_ra(N = nrow(vote_df),
                                 m = 2)

# Do randomization inference
conduct_ri(candidates_ballot ~ cycle + city + treatment + at_large * special,
           declaration = randomization_naive,
           assignment = "treatment",
           data = vote_df,
           sharp_hypothesis = 0,
           progress_bar = TRUE)
```

a)

```
##        term estimate two_tailed_p_value
## 1 treatment     3.29              0.051
```

```
rm(randomization_naive)
```

```
# Now define randomization procedure based on clustering structure
randomization_intersection = declare_ra(N = nrow(vote_df),
                                        m = 2,
                                        clusters = vote_df$city_cycle)

# Inference
conduct_ri(candidates_ballot ~ cycle + city + treatment + at_large * special,
           declaration = randomization_intersection,
           assignment = "treatment",
           data = vote_df,
           sharp_hypothesis = 0,
           progress_bar = TRUE)
```

b)

```
##        term estimate two_tailed_p_value
## 1 treatment    3.286              0.001
```

```
rm(randomization_intersection)
```

```
# Here we're manually doing the permutation. So first we're going to group gy city cycle and only retur
# We also drop seattle so we only  have untreated units
city_cycle_df = vote_df |>
  group_by(city_cycle) |>
  summarise(treatment = mean(treatment),
            city = unique(city),
            cycle = unique(cycle),
            candidates_ballot = mean(candidates_ballot, na.rm = T),
            at_large = mean(at_large, na.rm = T),
            special = mean(special, na.rm = T)) |>
  filter(treatment == 0)

# Create empty 148 by 148 matrix to put all permutations into
mean_effect = matrix(NA, nrow(city_cycle_df), nrow(city_cycle_df))

# Calculate mean effect for every permutation of two city-cycle pairs being treated
for (i in 1:(nrow(city_cycle_df) - 1)) {
  # this only goes to 147 so we don't double count

  # Outer loop. Two loops just make all the permutations of treatments
  city_cycle_df$treatment[i] = 1

  for (j in (i + 1):nrow(city_cycle_df )) {
    # This starts at i+1 to avoid double counting

    # Inner loop
    city_cycle_df$treatment[j] = 1
```

```
    # Calculate the treatment effect for treated and untreated groups
    effect = lm_robust( # I could use lm_robust_fit but I'm lazy so we can have a slow loop
      candidates_ballot ~ cycle + city + treatment + at_large * special,
      city_cycle_df,
      clusters = city,
      se_type = 'CR2'
    )

    # Calculate the treatment effect for each permutation and store it in output matrix
    mean_effect[i, j] = coef(effect)["treatment"]

    # Reset treatment for inner loop. We need to do this so we only have two treated units at at time
    city_cycle_df$treatment[j] = 0
  }

  # Reset outer loop so we only have two treated units at at time
  city_cycle_df$treatment[i] = 0
}

# Now, to calculate the p value, we see what proportion of the treatment effects we generated above are
greater = c(mean_effect) |> as_tibble() |> abs() |>  filter(value > 3.23227)

(p_value = nrow(greater)/length(mean_effect))
```

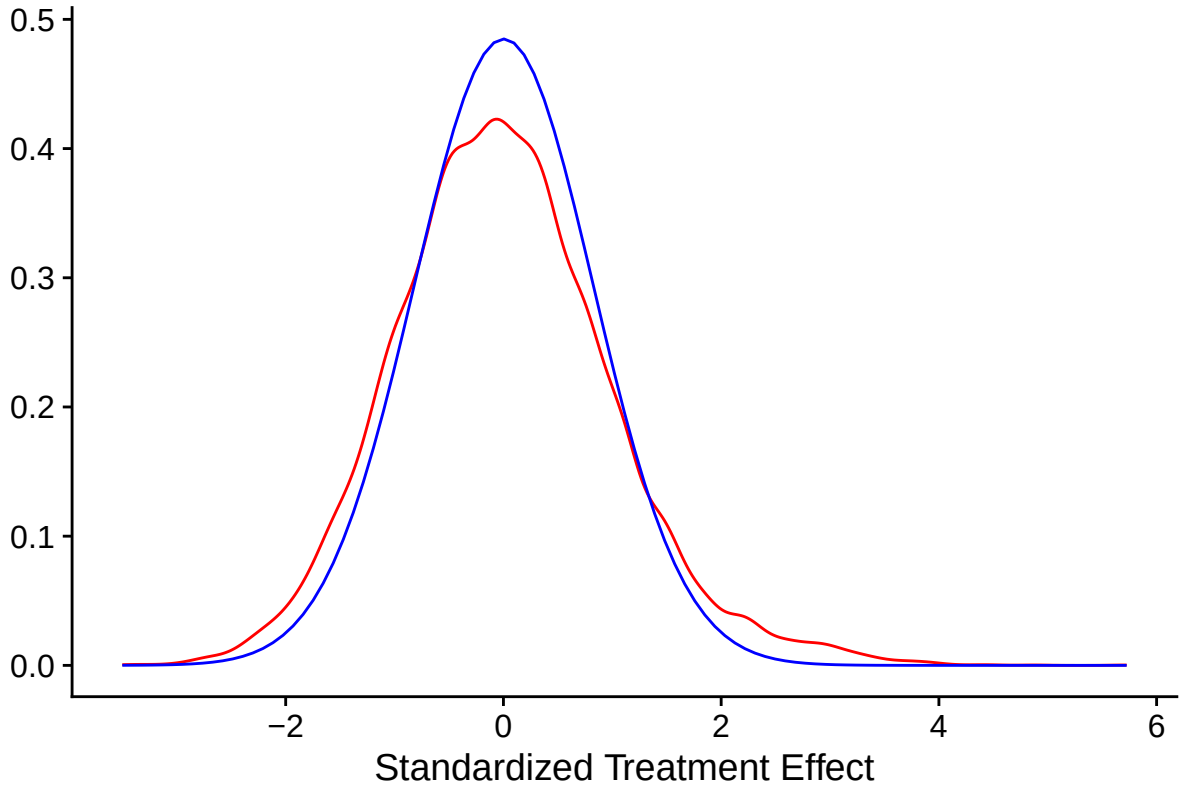
c)

```
## [1] 0.0004109
```

```
rm(greater)
rm(p_value)
```

```
# Standardize generated treatment effects
mean_effect = mean_effect[!is.na(mean_effect)] |> c()
mean_effect_standardized = (mean_effect - mean(mean_effect)) / sd(mean_effect)

# plot the density
mean_effect_standardized |>
  as_tibble() |>
  ggplot(aes(value)) +
    geom_density(color = 'red') +
    stat_function(fun = dnorm, args = list(mean = mean(mean_effect), sd = sd(mean_effect)), color = 'blu
    cowplot::theme_cowplot() +
    xlab("Standardized Treatment Effect") + ylab("") +
    labs(caption = "Red is data, blue is simulated distribution")
```



d) Red is data, blue is simulated distribution

```
ggsave(here::here("HW6", "plots", "ks.test.jpg"))
```

```
## Saving 6.5 x 4.5 in image
```

```
# Test formally if densities are the same with ks density
ks.test(mean_effect_standardized, "pnorm", mean = 0, sd = 1)
```

```
## Warning in ks.test.default(mean_effect_standardized, "pnorm", mean = 0, : ties
## should not be present for the Kolmogorov-Smirnov test
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  mean_effect_standardized
## D = 0.029, p-value = 4e-08
## alternative hypothesis: two-sided
```

```
rm(city_cycle_df)
```

```
# This is just gonna be copy pasted code from earlier but only collapsed by city. I should write a func
city_vote_df = vote_df |>
  group_by(city) |>
  summarise(treatment = mean(treatment),
            city = unique(city),
            candidates_ballot = mean(candidates_ballot, na.rm = T),
            at_large = mean(at_large, na.rm = T),
            special = mean(special, na.rm = T)) |>
  filter(treatment == 0)
```

```r
# Create empty 148 by 148 matrix to put all permutations into
mean_effect = matrix(NA, nrow(city_vote_df), nrow(city_vote_df))

# Calculate mean effect for every permutation of two city-cycle pairs being treated
for (i in 1:(nrow(city_vote_df) - 1)) {
  # this only goes to 147 so we don't double count

  # Outer loop. Two loops just make all the permutations of treatments
  city_vote_df$treatment[i] = 1

  for (j in (i + 1):nrow(city_vote_df)) {
    # This starts at i+1 to avoid double counting

    # Inner loop
    city_vote_df$treatment[j] = 1

    # Calculate the treatment effect for treated and untreated groups
    effect = lm_robust( # I could use lm_robust_fit but I'm lazy so we can have a slow loop
      candidates_ballot ~  city + treatment + at_large * special,
      city_vote_df,
      clusters = city,
      se_type = 'CR2'
    )

    # Calculate the treatment effect for each permutation and store it in output matrix
    mean_effect[i, j] = coef(effect)["treatment"]

    # Reset treatment for inner loop. We need to do this so we only have two treated units at at time
    city_vote_df$treatment[j] = 0
  }

  # Reset outer loop so we only have two treated units at at time
  city_vote_df$treatment[i] = 0
}

# Now, to calculate the p value, we see what proportion of the treatment effects we generated above are
greater = c(mean_effect) |> as_tibble() |> abs() |>  filter(value > 3.23227)

(p_value = nrow(greater)/length(mean_effect))
```

e)

```
## [1] 0
```

```r
rm(greater)
rm(p_value)
```

Quesiton 3