Experiments and Causality: Problem Set #4 (Vaibhav Beohar/MIDS/Spring 2021)

Alex, Scott & Micah

12/9/2020

```
options(digits = 5)
```

```
# install.packages("kableExtra")
# install.packages("ivreg")
# install.packages("ri2")
# install.packages("tidyverse")
# install.packages("ivreg")
# install.packages("stargazer")
# install.packages("sandwich")
# install.packages("ghostcript")

library(ri2)
library(stats)
library(tidyverse)
require(lmtest)
require(sandwich)
library(data.table)
library(stargazer)
library(tinytex)

library(ivreg)
library(kableExtra)
library(dplyr)

library(sandwich)
library(lmtest)
library(stargazer)
library(dplyr)
```

1. Noncompliance in Recycling Experiment

Suppose that you want to conduct a study of recycling behavior. A number of undergraduate students are hired to walk door to door and provide information about the benefits of recycling to people in the treatment group. Here are some facts about how the experiment was actually carried out.

- 1,500 households are assigned to the treatment group.
- The undergrads tell you that they successfully managed to contact 700 households.
- The control group had 3,000 households (not contacted by any undergraduate students).
- The subsequent recycling rates (i.e. the outcome variable) are computed and you find that 500 households in the treatment group recycled. In the control group, 600 households recycled.

```
library(data.table)
d <- data.table(id = 1:4500)
d[, assigned := ifelse(id <=1500, 1, 0)]
d[, treated := ifelse(id <=700, 1, 0)]
d[, recycled := ifelse(id <=500, 1, ifelse(id>=3900, 1, 0))]

head(d)
```

```
##    id assigned treated recycled
## 1:  1        1       1        1
## 2:  2        1       1        1
## 3:  3        1       1        1
## 4:  4        1       1        1
## 5:  5        1       1        1
## 6:  6        1       1        1
```

1. What is the ITT? Do the work to compute it, and store it into the object `recycling_itt`.

```
# 1st way to calculate ITT (via regression)
recycling_itt <- coef(d[, lm(recycled ~ assigned)])[2]
recycling_itt
```

```
## assigned
##    0.133
```

```
# 2nd way to calculate ITT (via mean of treatment/control)
recycling_itt = d[assigned ==1, mean(recycled)] - d[assigned ==0, mean(recycled)]
recycling_itt
```

```
## [1] 0.133
```

We show two ways to compute ITT which show the results are identical to each other at: 0.133

2. What is the CACE? Do the work to compute it, and store it into the object `recycling_cace`.

```
# calculating ITT_d below
recycling_itt_d <- coef(d[, lm(treated ~ assigned)])[2]
recycling_itt_d
```

```
## assigned
##  0.46667
```

```
# calculating CACE below as a fraction of ITT and ITT_d
recycling_cace <- recycling_itt / recycling_itt_d
recycling_cace
```

```
## assigned
##    0.285
```

CACE is computed as: 0.285

There appear to be some inconsistencies regarding how the undergraduates actually carried out the instructions they were given.

- One of the students, Mike, tells you that they actually lied about the the number of contacted treatment households and that the true number was 500.
- Another student, Andy, tells you that the true number was actually 600.

3. What is the CACE if Mike is correct?

```
#We first calculate the ITT_d for the number of households which Mike contacted
d[, treated_mike := ifelse(id <=500, 1, 0)]
recycling_itt_d_mike <- coef(d[, lm(treated_mike ~ assigned)])[2]
recycling_itt_d_mike
```

```
## assigned
##  0.33333
```

```
cace_mike <- recycling_itt / recycling_itt_d_mike
cace_mike
```

```
## assigned
##    0.399
```

We see that the CACE computation for Mike's contacted households comes to: 0.399

4. What is the CACE if Andy is correct?

```
d[, treated_andy := ifelse(id <=600, 1, 0)]
recycling_itt_d_andy <- coef(d[, lm(treated_andy ~ assigned)])[2]
recycling_itt_d_andy
```

```
## assigned
##      0.4
```

```
cace_andy <- recycling_itt / recycling_itt_d_andy
cace_andy
```

```
## assigned
##   0.3325
```

We see that the CACE computation for Andy's dataset is: 0.3325

For the rest of this question, suppose that **in fact** Mike was telling the truth.

5. What was the impact of the undergraduates's false reporting on our estimates of the treatment's effectiveness?

We see that as per dataset given by Mike, CACE calculated from that (Mike's) dataset was larger (0.399 vs. 0.285). Hence if Mike was correct, we would have underestimated the treatment effect in our earlier estimates.

6. Does your answer change depending on whether you choose to focus on the ITT or the CACE?

Yes our answer changes depending on what metric (ITT or CACE) we focus on. We see that ITT is not impacted by the take-out rate. However, CACE is a fraction of ITT and ITT_d (a.k.a take-out rates). And since ITT_d is impacted by the number of compliers in experimental studies, so does CACE. Therefore, we do see that CACE gets impacted by the number of people complying in the experimentation.

```
  library(knitr)
  knit_hooks$set(crop=hook_pdfcrop)
```

2. Fun with the placebo

The table below summarizes the data from a political science experiment on voting behavior. Subjects were randomized into three groups: a baseline control group (not contacted by canvassers), a treatment group (canvassers attempted to deliver an encouragement to vote), and a placebo group (canvassers attempted to deliver a message unrelated to voting or politics).

```
##
```

```
## The downloaded binary packages are in
##  /var/folders/qj/2d4k07wn6m976hlrt3_s48nc0000gn/T//RtmphTW8ID/downloaded_packages
```

```
##
## The downloaded binary packages are in
##  /var/folders/qj/2d4k07wn6m976hlrt3_s48nc0000gn/T//RtmphTW8ID/downloaded_packages
```

Assignment	Treated?	N	Turnout
Baseline	No	2463	0.3008
Treatment	Yes	512	0.3890
Treatment	No	1898	0.3160
Placebo	Yes	476	0.3002
Placebo	No	2108	0.3145

Evaluating the Placebo Group

```
#setting seed to 9, which makes the dataset closely resemble the given summary dataset
set.seed(9)

#Following code builds small sub-datasets which are then weaved
#       into the final unified datatable.
baseline <- data.table(N = 1:2463)
baseline[, ':='(Assignment="Baseline", Treated="No")]
baseline[, Turnout:=rbinom(2463, 1, .3008)]

Treat_d1 <- data.table(N = 1:512)
Treat_d1[, ':='(Assignment="Treatment", Treated="Yes")]
Treat_d1[, Turnout:=rbinom(512, 1, .3890)]

Treat_d0 <- data.table(N = 1:1898)
Treat_d0[, ':='(Assignment="Treatment", Treated="No")]
Treat_d0[, Turnout:=rbinom(1898, 1, .3160)]

Placebo_d1 <- data.table(N = 1:476)
Placebo_d1[, ':='(Assignment="Placebo", Treated="Yes")]
Placebo_d1[, Turnout:=rbinom(476, 1, .3002)]

Placebo_d0 <- data.table(N = 1:2108)
Placebo_d0[, ':='(Assignment="Placebo", Treated="No")]
Placebo_d0[, Turnout:=rbinom(2108, 1, .3145)]

#d <- rbind(baseline, Treat_d1, Treat_d0, Placebo_d1, Placebo_d0)
# d[, .(count = .N, avg = mean(Turnout)), by=list(Assignment, Treated)]
```

1. Construct a data set that would reproduce the table. (Too frequently we receive data that has been summarized up to a level that is unuseful for our analysis. Here, we're asking you to "un-summarize" the data to conduct the rest of the analysis for this question.)

```
# binding all the temp dataframes above into a unified data.table
d <- data.table(rbind(baseline, Treat_d1, Treat_d0, Placebo_d1, Placebo_d0))

# And now printing a summary to mimic the same provided to us in the problem
d[, .(N = .N, Turnout = mean(Turnout)), by=list(Assignment,Treated)]
```

```
##    Assignment Treated    N Turnout
## 1:   Baseline      No 2463 0.29760
```

```
## 2:  Treatment     Yes  512 0.39844
## 3:  Treatment      No 1898 0.31876
## 4:    Placebo     Yes  476 0.30252
## 5:    Placebo      No 2108 0.31499
```

2. Estimate the proportion of compliers by using the data on the treatment group.

```
# 1st solution: Compliance rate as % of compliers of total assigned folks to treatment
compliance_rate_t <- d[d[, Assignment == "Treatment"], .N, by=Treated]$N[1] /
  d[Assignment == "Treatment", .N]

compliance_rate_t
```

```
## [1] 0.21245
```

```
# Or other solution:
# (A) Calculate total folks assigned to treatment
total_treatment <- d[Assignment == "Treatment", .N]
# (B) Calculate total compliers among those assigned to treatment
total_treatment_complier <- filter(d, (Assignment=="Treatment" & Treated=="Yes"))[, .N]

compliance_rate_t <- total_treatment_complier / total_treatment
compliance_rate_t
```

```
## [1] 0.21245
```

We show that the total compliance rate (or proportion of compliers) as per the both approaches outlined above come to: 0.21245

3. Estimate the proportion of compliers by using the data on the placebo group.

```
total_placebo <- d[Assignment == "Placebo", .N]
total_placebo_complier <- filter(d, (Assignment=="Placebo" & Treated=="Yes"))[, .N]

compliance_rate_p <- total_placebo_complier / total_placebo
compliance_rate_p
```

```
## [1] 0.18421
```

Adopting the same approach from sub-question 2, we calculate the compliance rate among those assigned to placebo, is 0.18421.

4. Are the proportions in parts (1) and (2) statistically significantly different from each other? Provide *a test* and n description about why you chose that particular test, and why you chose that particular set of data.

```
proportions_difference_test <- prop.test(x = c(total_treatment_complier,
                                       total_placebo_complier),
                                       n = c(total_treatment, total_placebo))
proportions_difference_test$p.value
```

```
## [1] 0.013605
```

We chose the two-proportions z-test because we are comparing two observed proportions. We are interested in finding whether the observed proportion of compliers in treatment assignment group is equal to the observed proportion of compliers in placebo assignment. Here are null hypothesis becomes H0:pA = pB (where pA = compliers in treatment assignment and pB = compliers in placebo assignment).

With a p-value of 0.0136 we note that we reject the null at 5% significance level and note that both proportions are statistically significantly different.

5. What critical assumption does this comparison of the two groups' compliance rates test? Given what you learn from the test, how do you suggest moving forward with the analysis for this problem?

One of the critical asumptions in our experiments is that the perecentage of compliers in treatment is same as the perecentage of compliers in placebo. However, as seen from the solution above, the compliance rates in treatment and placebo are statistically significanlty different at the 5% confidence level. This could be due to a variety of reasons that should be analyzed as part of the experiment design, in order to move ahead.

We should check the three core assumptions of randomized control trials first (i.e perfect randomization, excludability and non-inference) and adddress if there could be breakdowns in either of these assumptions.

Most importantly, we should address if there are inherent biases that can mess with perfect randomizations. For example, did the canvassers stick to their script of not mentioning local issues with the subjects that could color their opinion on how they vote? Or we should check if the canvassers adopted any different approach while administering the non-treatment placebo to the subjects? Did the canvassers know beforehand that they were administering placebos, and hence were not motivated enough in giving the placebo to the subjects (with as much enthusiasm as they were giving treatments)?

These are some of the points we should review and possibly address before moving ahead with the experiment.

6. Estimate the CACE of receiving the placebo. Is the estimate consistent with the assumption that the placebo has no effect on turnout?

```
# 1st approach: using ivreg() to calculate using 2SLR approach:
d_baseline_placebo <- filter(d, (Assignment=="Placebo") | (Assignment=="Baseline"))
cace_estimate <- d_baseline_placebo[,
                                    ivreg(Turnout ~ Treated, ~ Assignment)]
summary(cace_estimate)$coefficients
```

```
##             Estimate Std. Error t value    Pr(>|t|)
## (Intercept) 0.297605  0.0092952 32.0171 6.2153e-205
## TreatedYes  0.081911  0.0705203  1.1615  2.4548e-01
## attr(,"df")
## [1] 5045
## attr(,"nobs")
## [1] 5047
```

```
# Or using an alternate approach to come to the same solution:
placebo_itt <- coef(d_baseline_placebo[, lm(Turnout ~ as.factor(Assignment))])[2]
placebo_itt_d <- coef(d_baseline_placebo[, lm(factor(Treated) ~ Assignment)])[2]
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored
```

```
## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
placebo_itt_d
```

```
## AssignmentPlacebo
##           0.18421
```

```
cace_estimate <- placebo_itt / placebo_itt_d
cace_estimate
```

```
## as.factor(Assignment)Placebo
##                     0.081911
```

We see that the contact by canvassers increased turnout among placebo compliers by 0.08191 percentage points. Although, this is subject to considerable sampling uncertainty. This still leads us to believe that the placebo is having some impact on ATE and is not zero (and there is perhaps something wrong with the design of the experiment that needs to be rectified).

Estimate the CACE Several Ways

7. Using a difference in means (i.e. not a linear model), compute the ITT using the appropriate groups' data. Then, divide this ITT by the appropriate compliance rate to produce an estiamte the CACE.

```
# an alternate approach (NOT ASKED IN THE QUESTION HERE):
# d_baseline_treatment <- filter(d, (Assignment=="Treatment") | (Assignment=="Baseline"))
# itt <- coef(d_baseline_treatment[, lm(Turnout ~ as.factor(Assignment))])[2]
# itt

# Approach asked for here (difference in means and then dividing by
#       take-out rate to calculate CACE:
itt        <- d[Assignment == "Treatment", mean(Turnout)] -
  d[Assignment == "Baseline", mean(Turnout)]
cace_means <- itt / compliance_rate_t
cace_means
```

```
## [1] 0.17924
```

Calculated ITT and CACE: 0.03808 and 0.17924.

8. Use two separate linear models to estimate the CACE of receiving the treatment by first estimating the ITT and then dividing by ITT_D. Use the `coef()` extractor and in line code evaluation to write a descriptive statement about what you learn after your code.

```
d_baseline_treatment <- filter(d, (Assignment=="Treatment") | (Assignment=="Baseline"))

itt_model   <- coef(d_baseline_treatment[, lm(Turnout ~ as.factor(Assignment))])[2]
itt_model
```

```
## as.factor(Assignment)Treatment
##                        0.03808
```

```
itt_d_model <- coef(d_baseline_treatment[, lm(factor(Treated) ~ Assignment)])[2]
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored
```

```
## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
itt_d_model
```

```
## AssignmentTreatment
##             0.21245
```

```
itt_model / itt_d_model
```

```
## as.factor(Assignment)Treatment
##                        0.17924
```

This is an alternate approach of calculating Causal ATE. We see that using the 1st stage and 2nd stage linear models (which is equivalent to using the 2-stage linear regression model or a reduced form model approach) the answer is exactly same as the earlier approach of calculating CACE via a difference in means approach. In both approaches, we obtain the CACE as 0.17924.

9. When a design uses a placebo group, one additional way to estiamte the CACE is possible – subset to include only compliers in the treatment and placebo groups, and then estimate a linear model. Produce that estimate here.

```
d_treatment_placebo <- filter(d, ((Assignment=="Treatment") |
                                    (Assignment=="Placebo")) & (Treated=="Yes"))
```

```
cace_subset_model <- d_treatment_placebo[, ivreg(Turnout ~ Assignment)]
cace_subset_model
```

```
##
## Call:
## ivreg(formula = Turnout ~ Assignment)
##
## Coefficients:
##         (Intercept)  AssignmentTreatment
##              0.3025               0.0959
```

```
cace_subset_model <- d_treatment_placebo[, lm(Turnout ~ Assignment)]

summary(cace_subset_model)
```

```
##
## Call:
## lm(formula = Turnout ~ Assignment)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -0.398 -0.398 -0.302  0.602  0.698
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)            0.3025     0.0218   13.87   <2e-16 ***
## AssignmentTreatment    0.0959     0.0303    3.17   0.0016 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.476 on 986 degrees of freedom
## Multiple R-squared:  0.0101, Adjusted R-squared:  0.00906
## F-statistic:   10 on 1 and 986 DF,  p-value: 0.00159
```

10. In large samples (i.e. "in expectation") when the design is carried out correctly, we have the expectation that the results from 7, 8, and 9 should be the same. Are they? If so, does this give you confidence that these methods are working well. If not, what explains why these estimators are producing different estimates?

Step 9 is computing the average treatment effect on the compliers (which is called as ATET a.k.a ATE on the treated). Whereas, the CACE is the complier ATE with some additional noise due to inclusion on non-compliers. However, the placebo design ATET allows us to get unbiased estimates of the CACE (by increasing the precision and reducing standard errors).

In theory, both approaches should give us the similar estimates of CACE (in perfectly randomized and administered experiments where the placebo design is not impacting the treatment group and vice versa). In this case, however, we see that the ATET is much different from the treatment CACE. This was expected, as we saw from the proportions difference test that the compliance rates in treatment and placebo groups were significantly different at the 5% confidence level.

11. In class we discussed that the rate of compliance determines whether one or another design is more efficient. (You can review the textbook expectation on page 162 of *Field Experiments*)). Given the compliance rate in this study, which design *should* provide a more efficient estimate of the treatment effect?

Conventional designs are preferred when compliance rate of compliers in treatment are greater than 50%. In

this case, since we have complier rates at 21% (well below the 50% threshold), we should prefer the placebo design approach for a more efficient estimate of the treatment effrct.

12. When you apply what you've said in part (11) against the data that you are working with, does the {placebo vs. treatment} or the {control vs. treatment} comparison produce an estimate with smaller standard errors?

```
#computing std error of control vs treatment using the ivreg() model
d_baseline_treatment <- filter(d, (Assignment=="Treatment") | (Assignment=="Baseline"))

cace_estimate <- d_baseline_treatment[,
                                      ivreg(Turnout ~ Treated, ~ Assignment)]
summary(cace_estimate)
```

```
##
## Call:
## ivreg(formula = Turnout ~ Treated | Assignment)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -0.477 -0.298 -0.298  0.702  0.702
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.29760    0.00937   31.76   <2e-16 ***
## TreatedYes   0.17924    0.06273    2.86   0.0043 **
##
## Diagnostic tests:
##                   df1  df2 statistic p-value
## Weak instruments    1 4871    664.14  <2e-16 ***
## Wu-Hausman          1 4870      2.23    0.14
## Sargan              0   NA        NA      NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.465 on 4871 degrees of freedom
## Multiple R-Squared: 0.000312,    Adjusted R-squared: 0.000107
## Wald test: 8.17 on 1 and 4871 DF,  p-value: 0.00429
```

In the {control vs. treatment} design, we are calculating a regular version of CACE. In the {placebo vs. treatment} we are calculating a version of ATET. We know that the ATET provides a shrinkage in standard error due to the elimination of never-takers. Therefore, standard error of conventional CACE should be 1/(compliance rate)^(1/2) more than the standard error of ATET.

As seen in the illustrative calculations, we see precisely that (std error of CACE is 0.0627 and std error of ATET is 0.03029).

3. Turnout in Dorms

Guan and Green report the results of a canvassing experiment conduced in Beijing on the eve of a local election. Students on the campus of Peking University were randomly assigned to treatment or control groups.

- Canvassers attempted to contact students in their dorm rooms and encourage them to vote.
- No contact with the control group was attempted.
- Of the 2,688 students assigned to the treatment group, 2,380 were contacted.

- A total of 2,152 students in the treatment group voted; of the 1,334 students assigned to the control group, 892 voted.
- One aspect of this experiment threatens to violate the exclusion restriction. At every dorm room they visited, even those where no one answered, canvassers left a leaflet encouraging students to vote.

```
d <- fread('https://ucb-mids-w241.s3-us-west-1.amazonaws.com/Guan_Green_CPS_2006.csv')
head(d)
```

```
##    turnout treated  dormid treatment_group
## 1:       0       0 1010101               0
## 2:       0       0 1010101               0
## 3:       0       0 1010101               0
## 4:       0       0 1010102               0
## 5:       0       0 1010102               0
## 6:       0       1 1010103               1
```

```
# Custom function to calculate and return clustered std errors
clustered_se <- function(mod){
  mod$vcovCL_ <- NULL
  mod$vcovCL_ <- vcovCL(mod, cluster = d[, dormid])
  return(sqrt(diag(mod$vcovCL_)))
}
```

Here are definitions for what is in that data:

- `turnout` did the person turn out to vote?
- `treated` did someone at the dorm open the door?
- `dormid` a unique ID for the door of the dorm
- `treatment_group` whether the dorm door was assigned to be treated or not

Use Linear Regressions

1. Estimate the ITT using a linear regression on the appropriate subset of data. Notice that there are two `NA` in the data. Just na.omit to remove these rows so that we are all working with the same data. Given the ways that randomization was conducted, what is the appropriate way to construct the standard errors?

```
d <- na.omit(d)
dorm_model <- lm(turnout ~ treatment_group, data = d)

#Print out the model summary with clustered std errors
coeftest(dorm_model, vcov = vcovCL(dorm_model, d$dormid))[2:1,]
```

```
##                 Estimate Std. Error t value    Pr(>|t|)
## treatment_group  0.13193   0.023271  5.6692  1.5355e-08
## (Intercept)      0.66867   0.020241 33.0349 6.0223e-212
```

```
#Print out descriptive model summary with clustered std errors
stargazer(
          dorm_model,
          se = list(clustered_se(dorm_model)[2]),
          type='text',
          add.lines =list(c('SE Flavor','Clustered')),
          column.labels = c("dorm_model"),
          header = F
          )
```

```
##
```

```
## ================================================
##                          Dependent variable:
##                      ---------------------------
##                                turnout
##                                 dorm
## ------------------------------------------------
## treatment_group                 0.132
##                                (0.023)
##
## Constant                       0.669***
##
##
## ------------------------------------------------
## SE Flavor                      Clustered
## Observations                     4,022
## R2                               0.021
## Adjusted R2                      0.021
## Residual Std. Error       0.425 (df = 4020)
## F Statistic           86.082*** (df = 1; 4020)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

We see that the treatment assignment was done at the student level, with students at each dorm room being assigned to the treatment or control (and hence clustering happening at the dorm room level and randomization assignment happening at the clustered dorm room level). Therefore it is appropriate to have clustered standard errors here, as a measure of standard error.

Use Randomization Inference

1. How many people are in treatment and control? Does this give you insight into how the scientists might have randomized? As ususal, include a narrative setence after your code.

```
# Here we calculate absolute number of people in treatment and control
n_treatment <- d[treatment_group == 1, .N]
n_control <- d[treatment_group == 0, .N]

n_treatment
```

```
## [1] 2688
```

```
n_control
```

```
## [1] 1334
```

Total number of people in treatment: 2688

Total number of people in control : 1334

We see a roughly 2:1 randomization ratio for treatment and control (2.01499 to be precise using formula `n_treatment/ n_control`)

2. Write an algorithm to conduct the Randomization Inference. Be sure to take into account the fact that random assignment was clustered by dorm room.

```
# We use ri2 package as an efficient way of implemeting randomization inference.

# STEPS:
# First we define the experiment in the declaration
# Then we pass on hyperparameters along with the experiment design (declaration)
```

```r
# in the 2nd step (ri2_out_ate)

# We also illustrate a method of running custom formula (code commented) to
# show how ri2 package can be used to provide various statistics on
# a custom formula

# Here we get total number of clusters for treatment and control
n_treatment_cluster <- d[treatment_group == 1, .N, by=dormid][, .N]
n_control_cluster <- d[treatment_group == 0, .N, by=dormid][, .N]

# Declare the type of design we need for our experiment in the ri2 package
declaration <-
  with(d,{
    declare_ra(
      clusters = dormid,
      m_each = c(n_treatment_cluster, n_control_cluster))
  })

# Print out the design of the experiment here (FOR DEBUGGING)
# note the probability of assignment of clusters for treatment and control
declaration
```

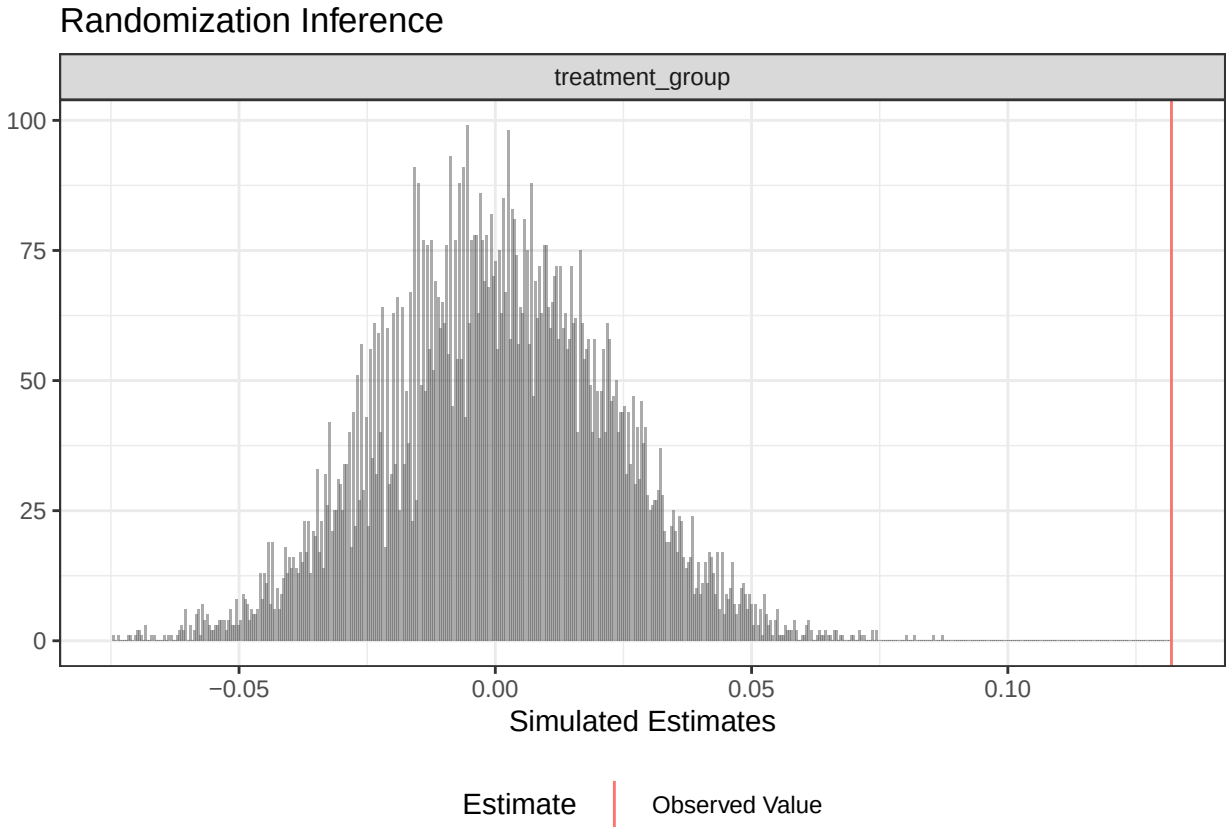
```
## Random assignment procedure: Cluster random assignment
## Number of units: 4022
## Number of clusters: 1004
## Number of treatment arms: 2
## The possible treatment categories are 0 and 1.
## The number of possible random assignments is approximately infinite.
## The probabilities of assignment are constant across units:
##  prob_0  prob_1
## 0.66733 0.33267
```

```r
# Finally running randomization inference for 10000 steps using sharp_null_hyp = 0
ri2_out_ate <- conduct_ri(
  formula =  d[, lm(turnout ~ treatment_group)],
  assignment = "treatment_group",
  sharp_hypothesis = 0, # means we assume there is no effect of treatment
  declaration = declaration, # passing model design here
  data = d,
  sims = 10000, # running for 10000 simulations,
  progress_bar = TRUE
)

# Below code commented: but could be used (IN A SIMILAR PROJECT SETTING)
# for returning a custom statistic (such as standard error)
# for a given formula (using ri2)
# stderr_fun <- function(data) {
#       model <- lm(turnout ~ treatment_group, data = d)
#       se <- sqrt(diag(vcovCL(model)))[2]
#       names(se) <- NULL
#       return(se)
# }
#
# stderr_fun(d)
```

```
#
# ri2_out_stderr <- conduct_ri(
#   test_function = stderr_fun,
#   assignment = "treatment_group",
#   sharp_hypothesis = 0,
#   declaration = declaration,
#   data = d,
#   sims = 1000,
#   progress_bar = TRUE
# )

plot(ri2_out_ate)
```




```
summary(ri2_out_ate, p = "two-tailed")
```

```
##              term estimate two_tailed_p_value
## 1 treatment_group  0.13193                   0
```

3. What is the value that you estimate for the treatment effect?

```
dorm_room_ate <- summary(ri2_out_ate, p = "two-tailed")[2]
dorm_room_ate
```

```
##   estimate
## 1  0.13193
```

We get an estimate of the treatment effect from the RI process as: 0.1319295709 (using the formula: `summary(ri2_out_ate, p = "two-tailed")[2]`)

4. What are the 2.5% and 97.5% quantiles of this distribution?

```
dorm_room_ci <- quantile(ri2_out_ate$sims_df[, 1], prob = c(0.025, 0.975))
dorm_room_ci
```

```
##      2.5%     97.5%
## -0.043506  0.043721
```

5. What is the p-value that you generate for the test: How likely is this treatment effect to have been generated if the sharp null hypothesis were true.

```
p_value <- summary(ri2_out_ate, p = "two-tailed")[3]
p_value
```

```
##   two_tailed_p_value
## 1                  0
```

The likelihood of treatment effect to have been generated if the sharp null hypothesis were true is the p-value of 0 calculated in the above section.

6. Assume that the leaflet (which was left in case nobody answered the door) had no effect on turnout. Estimate the CACE either using ITT and ITT_d or using a set of linear models. What is the CACE, the estimated standard error of the CACE, and the p-value of the test you conduct?

```
#Calculating CACE using 2SLS (ivreg) and displaying results:
dorm_room_cace <- d[, ivreg(turnout ~ treated, ~ treatment_group)]
dorm_room_cace$coefficients[2]
```

```
## treated
## 0.14894
```

```
#Clustered standard errors of the CACE:
clustered_se(dorm_room_cace)[2]
```

```
##  treated
## 0.026308
```

```
#p-value of the CACE
summary(dorm_room_cace)$coefficients[2,4]
```

```
## [1] 3.3559e-20
```

```
# Finally, displaying model summary:
stargazer(
          dorm_room_cace,
          se = list(clustered_se(dorm_room_cace)[2]
            #sqrt(diag(vcovHC(dorm_room_cace, type = 'HC0')))
            ),
          type='text',
          add.lines =list(c('SE Flavor','Clustered')),
          column.labels = c("dorm_room_cace"),
          header = F
          )
```

```
##
## ================================================
##                          Dependent variable:
##                      ---------------------------
##                                turnout
##                                 dorm
## ------------------------------------------------
```

```
## treated                          0.149
##                                 (0.026)
##
## Constant                        0.669***
##
##
## ------------------------------------------------
## SE Flavor                       Clustered
## Observations                      4,022
## R2                                0.016
## Adjusted R2                       0.016
## Residual Std. Error       0.426 (df = 4020)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

We conduct a two-stage least square regression test to find the CACE of our experiment.

CACE: 0.14894

Clustered Standard Error of the CACE: 0.02631

P-value: 3.35588×10^{-20}

7. What if the leaflet that was left actually *did* have an effect? Is it possible to estimate a CACE in this case? Why or why not?

If dropping a leaflet had an effect, this would mean that dropping a leaflet was equal to providing treatment to those subjects where we dropped those leaflets. Therefore, we would have essentially provided treatment to all those who were assigned treatment. This would imply that there was no one with left without compliance (and equal to knocking on all doors and finding all those assigned to treatment and giving them the same treatment).

Thus in this case, with 100% compliance, CACE (or the complier ATE) would equal to ITT. Therefore calculating CACE in this case would be equal to calculating ITT.

4. Another Turnout Question

We're sorry; it is just that the outcome and treatment spaces are so clear!

Hill and Kousser (2015) report that it is possible to increase the probability that someone votes in the California *Primary Election* simply by sending them a letter in the mail. This is kind of surprising, because who even reads the mail anymore anyways? (Actually, if you talk with folks who work in the space, they'll say, "We know that everybody throws our mail away; we just hope they see it on the way to the garbage.")

Can you replicate their findings? Let's walk through them.

```
# number_rows <- 100 # you should change this for your answer. full points for full data
#
# d <- data.table::fread(
#   input = 'https://ucb-mids-w241.s3-us-west-1.amazonaws.com/hill_kousser_analysis_file.csv',
#   nrows = number_rows)
```

(As an aside, you'll note that this takes some time to download. One idea is to save a copy locally, rather than continuing to read from the internet. One problem with this idea is that you might be tempted to make changes to this cannonical data; changes that wouldn't be reflected if you were to ever pull a new copy from the source tables. One method of dealing with this is proposed by Cookiecutter data science.)

Here's what is in that data.

- `age.bin` a bucketed, descriptive, version of the `age.in.14` variable

- `party.bin` a bucketed version of the `Party` variable
- `in.toss.up.dist` whether the voter lives in a district that often has close races
- `minority.dist` whether the voter lives in a majority minority district, i.e. a majority black, latino or other racial/ethnic minority district
- `Gender` voter file reported gender
- `Dist1-8` congressional and data districts
- `reg.date.pre.08` whether the voter has been registered since before 2008
- `vote.xx.gen` whether the voter voted in the `xx` general election
- `vote.xx.gen.pri` whether the voter voted in the `xx` general primary election
- `vote.xx.pre.pri` whether the voter voted in the `xx` presidential primary election
- `block.num` a block indicator for blocked random assignment.
- `treatment.assign` either "Control", "Election Info", "Partisan Cue", or "Top-Two Info"
- `yvar` the outcome variable: did the voter vote in the 2014 primary election

These variable names are horrible. Do two things:

- Rename the smallest set of variables that you think you might use to something more useful. (You can use `data.table::setnames` to do this.)
- For the variables that you think you might use; check that the data makes sense;

When you make these changes, take care to make these changes in a way that is reproducible. In doing so, ensure that nothing is positional indexed, since the orders of columns might change in the source data).

While you're at it, you might as well also modify your `.gitignore` to ignore the data folder. Because you're definitely going to have the data rejected when you try to push it to github. And every time that happens, it is a 30 minute rabbit hole to try and re-write git history.

```
# METHODOLOGY:
# download the entire file in local storage
# load the file in R studio
# convert the file into RDS and save it locally (this ensures much smaller disk
# size and easy to load for future,
# in case it needs to be reloaded due to a mishap)
# reopen the file in rds

# df <- read.csv('/Users/Vaibhav_Beohar/Documents/VB_Mck_Docs/MIDS/W241
#       /ps4/problem-set-4-vbeohar/data/hill_kousser_analysis_file.csv')
# df <- read.csv('/home/rstudio/problem-set-4-vbeohar/
#       data/hill_kousser_analysis_file.csv')
# saveRDS(df, file = "/home/rstudio/problem-set-4-vbeohar/
#    data/hill_kousser_analysis_file_original.rds")
# rm(df)
#df_hill_kousser_orignial <- readRDS(file = "/home/rstudio/
#   problem-set-4-vbeohar/data/hill_kousser_analysis_file_original.rds")

path_to_file <- "/Users/Vaibhav_Beohar/Documents/VB_Mck_Docs/MIDS/W241/ps4/problem-set-4-vbeohar/data/hi
df_hill_kousser_orignial <- readRDS(file = path_to_file)
head(df_hill_kousser_orignial)
```

```
##   LocalityCode age.bin party.bin in.toss.up.dist minority.dist vote.10.gen
## 1            1       2         1               0             0           0
## 2            1       5         3               0             0           0
## 3            1       6         2               0             0           0
## 4            1       5         1               1             1           0
## 5            1       5         2               0             0           1
## 6            1       6         3               0             0           1
##   vote.08.gen Party age.in.14 Gender Dist1 Dist2 Dist3 Dist4 Dist5 Dist6 Dist7
```

```
## 1           0   REP        32      F CG015 SA020 SE002 SS010    NA    NA    NA
## 2           1   NPP        61      F CG013 SA018 SE002 SS009    NA    NA    NA
## 3           1   DEM        77      M CG013 SA015 SE002 SS009    NA    NA    NA
## 4           1   REP        62      M CG017 SA025 SE002 SS010    NA    NA    NA
## 5           1   DEM        60      M CG015 SA020 SE002 SS010    NA    NA    NA
## 6           1   NPP        81        CG015 SA020 SE002 SS010    NA    NA    NA
##   Dist8 reg.date.pre.08 reg.date.pre.10 vote.12.gen vote.12.pre.pri
## 1    NA               1               1           1               0
## 2    NA               0               0           1               0
## 3    NA               1               1           1               0
## 4    NA               0               0           1               0
## 5    NA               1               1           1               0
## 6    NA               1               1           1               0
##   vote.10.gen.pri vote.08.pre.pri vote.08.gen.pri block.num leftover.case
## 1               0               0               0        91             0
## 2               0               0               0       316             0
## 3               0               0               0       364             0
## 4               0               0               0       296             0
## 5               0               0               0       302             0
## 6               0               0               0       382             0
##   treatment.assign yvar matched.to.post vote.14.gen
## 1          Control    0               1           0
## 2          Control    0               1           0
## 3          Control    1               1           1
## 4          Control    0               1           0
## 5          Control    0               1           1
## 6          Control    0               1           0
```

```
d <- data.table(df_hill_kousser_orignial[, c("block.num", "treatment.assign", "yvar")])
rm(df_hill_kousser_orignial)
```

```
# Adding an additional column that makes control=0 and every other letter=1
d <- d[, Baseline_Assignment := ifelse(treatment.assign=="Control", 0, 1)]
```

Some questions!

1. **A Simple Treatment Effect**: Load the data and estimate a `lm` model that compares the rates of turnout in the control group to the rate of turnout among anybody who received *any* letter. This model combines all the letters into a single condition – "treatment" compared to a single condition "control". Report robust standard errors, and include a narrative sentence or two after your code.

```
# How else we could do this alternately:
d[Baseline_Assignment ==1, mean(yvar)] - d[Baseline_Assignment ==0, mean(yvar)]
```

```
## [1] 0.0048992
```

```
# some additional crosschecks of means:
d[Baseline_Assignment ==1, mean(yvar)]
```

```
## [1] 0.098024
```

```
d[Baseline_Assignment ==0, mean(yvar)]
```

```
## [1] 0.093125
```

```
# ITT of treatment vs control
# How it is asked in this assignment
```

```r
model_simple <-  d[, lm(yvar ~ Baseline_Assignment)]
summary(model_simple)$coefficients
```

```
##                       Estimate Std. Error  t value  Pr(>|t|)
## (Intercept)         0.0931248 0.00015076 617.7217 0.000e+00
## Baseline_Assignment 0.0048992 0.00076700   6.3875 1.686e-10
```

```r
# calculate Robust se either usign sqrt of diag of var-cov matrix or coeftest()
#rse <- sqrt(diag(vcovHC(model_simple, type = 'HC0')))
rse <- coeftest(model_simple, vcov = vcovHC(model_simple, type="HC1"))
rse
```

```
##
## t test of coefficients:
##
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         0.093125   0.000151  618.28   <2e-16 ***
## Baseline_Assignment 0.004899   0.000783    6.25    4e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
stargazer(
          model_simple,
          se = rse,
          type='text',
          add.lines =list(c('SE Flavor','Robust')),
          column.labels = c("model_simple"),
          header = F
          )
```

```
##
## ================================================
##                          Dependent variable:
##                      ---------------------------
##                                   yvar
##                                   model
## ------------------------------------------------
## Baseline_Assignment               0.005
##
##
## Constant                          0.093
##                                  (0.093)
##
## ------------------------------------------------
## SE Flavor                         Robust
## Observations                    3,872,268
## R2                                0.00001
## Adjusted R2                       0.00001
## Residual Std. Error    0.291 (df = 3872266)
## F Statistic         40.801*** (df = 1; 3872266)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

ATE of control: 0.09312

ATE of treatment: 0.09802

Total treatment effect (ITT): 0.0049

Robust standard errors: 0.09312, 0.0049, 1.50619×10^{-4}, 7.83399×10^{-4}, 618.28132, 6.25382, 0, 4.00575×10^{-10}

We see that the impact of providing any letter (vs no letter) has an ATE of 0.09802 (with a statistically significant p-value of 1.68603×10^{-10}). This means that the treatment does work! Although we dont yet know if any specific letter type has any more effect than any of the other letter types.

2. **Specific Treatment Effects**: Suppose that you want to know whether different letters have different effects. To begin, what are the effects of each of the letters, as compared to control? Estimate an appropriate linear model and use robust standard errors.

```
model_letters <- d[, lm(yvar ~ as.factor(treatment.assign))]
summary(model_letters)$coefficients
```

```
##                                          Estimate Std. Error  t value
## (Intercept)                             0.0931248 0.00015076 617.7215
## as.factor(treatment.assign)Election info 0.0049846 0.00168931   2.9507
## as.factor(treatment.assign)Partisan      0.0052597 0.00119841   4.3889
## as.factor(treatment.assign)Top-two info  0.0044961 0.00119844   3.7516
##                                            Pr(>|t|)
## (Intercept)                             0.0000e+00
## as.factor(treatment.assign)Election info 3.1706e-03
## as.factor(treatment.assign)Partisan      1.1393e-05
## as.factor(treatment.assign)Top-two info  1.7570e-04
```

```
#calculating robust standard errors here and displaying for all factors
rse <- coeftest(model_letters, vcov = vcovHC(model_letters, type="HC1"))
rse
```

```
##
## t test of coefficients:
##
##                                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             0.093125   0.000151  618.28  < 2e-16
## as.factor(treatment.assign)Election info 0.004985   0.001727    2.89  0.00390
## as.factor(treatment.assign)Partisan      0.005260   0.001227    4.29  1.8e-05
## as.factor(treatment.assign)Top-two info  0.004496   0.001222    3.68  0.00024
##
## (Intercept)                             ***
## as.factor(treatment.assign)Election info **
## as.factor(treatment.assign)Partisan      ***
## as.factor(treatment.assign)Top-two info  ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
stargazer(
          model_letters,
          se = rse,
          type='text',
          add.lines =list(c('SE Flavor','Robust')),
          column.labels = c("model_simple"),
          header = F
          )
```

```
##
## ===============================================================
##                                         Dependent variable:
```

```
##                                          ---------------------------
##                                                      yvar
##                                                      model
## ------------------------------------------------------------------------
## as.factor(treatment.assign)Election info              0.005
##
##
## as.factor(treatment.assign)Partisan                   0.005
##
##
## as.factor(treatment.assign)Top-two info               0.004
##
##
## Constant                                              0.093
##                                                      (0.093)
##
## ------------------------------------------------------------------------
## SE Flavor                                             Robust
## Observations                                        3,872,268
## R2                                                    0.00001
## Adjusted R2                                           0.00001
## Residual Std. Error                        0.291 (df = 3872264)
## F Statistic                              13.670*** (df = 3; 3872264)
## ========================================================================
## Note:                                    *p<0.1; **p<0.05; ***p<0.01
```

All letter treatments are used as categorical indicator variables in our new model. After looking at the results, we can confirm our prior observation that the treatment of sending letters to prospective voters is effective. However, we still dont know if any specific letter is much more effective than any other (as the similar coefficients and p-values indicate).

3. Does the increased flexibilitiy of a different treatment effect for each of the letters improve the performance of the model? Test, using an F-test. What does the evidence suggest, and what does this mean about whether there **are** or **are not** different treatment effects for the different letters?

```
model_anova <- anova(model_simple, model_letters, test='F')
model_anova
```

```
## Analysis of Variance Table
##
## Model 1: yvar ~ Baseline_Assignment
## Model 2: yvar ~ as.factor(treatment.assign)
##    Res.Df    RSS Df Sum of Sq   F Pr(>F)
## 1 3872266 327616
## 2 3872264 327616  2    0.0177 0.1    0.9
```

Looking at the p-value that is NOT statistically significant at the 5% confidence level, we cannot deduce that the increased flexibilitiy of a different treatment effect for each of the letters improves the performance of the model at all. Both models in our comparative F-test have the same statistical power and are not showing statistically significantly different results.

4. **More Specific Treatment Effects** Is one message more effective than the others? The authors have drawn up this design as a full-factorial design. Write a *specific* test for the difference between the *Partisan* message and the *Election Info* message. Write a *specific* test for the difference between *Top-Two Info* and the *Election Info* message. Report robust standard errors on both tests and include a short narrative statement after your estimates.

```r
# we use dplyr package to filter on data.tables for our 2-letter codes
d_partisan_electioninfo <- filter(d,
                          (treatment.assign =="Partisan" |
                             treatment.assign=="Election info"))
d_toptwoinfo_electioninfo <- filter(d,
                           (treatment.assign =="Top-two info" |
                              treatment.assign=="Election info"))


# creating respective models
model_partisan_vs_info <- d_partisan_electioninfo[,
                                    lm(yvar ~ as.factor(treatment.assign))]
model_top_two_vs_info  <- d_toptwoinfo_electioninfo[,
                                    lm(yvar ~ as.factor(treatment.assign))]

# calculating robust standard errors here
rse_1 <- coeftest(model_partisan_vs_info,
                  vcov = vcovHC(model_partisan_vs_info, type="HC1"))
rse_2 <- coeftest(model_top_two_vs_info,
                  vcov = vcovHC(model_top_two_vs_info, type="HC1"))

# reporting coefficients and robust std errors for both our 2-letter models
rse_1
```

```
##
## t test of coefficients:
##
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        0.098109   0.001721   57.02   <2e-16 ***
## as.factor(treatment.assign)Partisan 0.000275   0.002108    0.13      0.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
rse_2
```

```
##
## t test of coefficients:
##
##                                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             0.098109   0.001721   57.02   <2e-16
## as.factor(treatment.assign)Top-two info -0.000489   0.002105   -0.23     0.82
##
## (Intercept)                            ***
## as.factor(treatment.assign)Top-two info
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
stargazer(
          model_partisan_vs_info, model_top_two_vs_info,
          se = list(rse_1, rse_2),
          type='text',
          add.lines =list(c('SE Flavor','Robust', 'Robust')),
          column.labels = c("model1", "model2"),
          header = F
          )
```

```
##
## ================================================================================
##                                                  Dependent variable:
##                                              ---------------------------
##                                                         yvar
##                                                 model1          model2
##                                                   (1)             (2)
## --------------------------------------------------------------------------------
## as.factor(treatment.assign)Partisan              0.0003
##                                                 (0.0003)
##
## as.factor(treatment.assign)Top-two info                          -0.0005
##                                                                 (-0.0005)
##
## Constant                                          0.098           0.098
##                                                  (0.098)         (0.098)
##
## --------------------------------------------------------------------------------
## SE Flavor                                        Robust          Robust
## Observations                                     89,742          89,739
## R2                                               0.00000         0.00000
## Adjusted R2                                     -0.00001        -0.00001
## Residual Std. Error                  0.298 (df = 89740)  0.297 (df = 89737)
## F Statistic                         0.017 (df = 1; 89740) 0.054 (df = 1; 89737)
## ================================================================================
## Note:                                              *p<0.1; **p<0.05; ***p<0.01
```

Robust standard error for *Partisan* treatment indicator: 0.00211

Robust standard error for *Top-Two Info* treatment indicator: 0.00211

In these aforementioned models, we are treating "election info" as the baseline indicator variable (omitted variable), while the other two letter types *Top-Two Info* and the *Partisan* are treatment variables.

As seen by the coefficients of both models, none of these models are not showing statistically significantly different results of either of the new treatment indicator being superior to the baseline "election info" letter.

In both cases, the p-values of the treatment indicators are not significant at the 5% confidence level (values being 0.89617 and 0.81651 respectively).

5. **Blocks? We don't need no stinking blocks?** The blocks in this data are defined in the `block.num` variable (which you may have renamed). There are a *many* of blocks in this data, none of them are numerical – they're all category indicators. How many blocks are there?

```
#d[, uniqueN(block.num)]
num_blocks <- d[, .N, by=block.num][, .N]
num_blocks
```

```
## [1] 382
```

There are 382 blocks in the data.

6. **SAVE YOUR CODE FIRST** but then try to estimate a `lm` that evaluates the effect of receiving *any letter*, and includes this block-level information. What happens? Why do you think this happens? If this estimate *would have worked* (that's a hint that we don't think it will), what would the block fixed effects have accomplished?

```
# I was able to run this fixed effects model on a
# Docker image with 12 CPUs, 35 GB RAM and a swap
```

```r
# allocation of 3 GB (on a 16 inch Macbook Pro with
# 16 CPUs, 1TB SSD, 64 GB RAM and 8 GB NVidia GPU).
# Time taken: ~10 minutes

model_block_fx  <- d[, lm(yvar ~ as.factor(treatment.assign) + as.factor(block.num))]

# stargazer cannot run because of an attempt to
# calculate robust standard errors.
#stargazer(model_block_fx, se=
#    sqrt(diag(vcovHC(model_block_fx, type = 'HC0'))), type='text')

# instead print out the top coefficients
coef(summary(model_block_fx))[1:10]
```

6. Even though we can't estimate this fixed effects model directly, we can get the same information and model improvement if we're *just a little bit clever*. Create a new variable that is the *average turnout within a block* and attach this back to the data.table. Use this new variable in a regression that regresses voting on `any_letter` and this new `block_average`. Then, using an F-test, does the increased information from all these blocks improve the performance of the *causal* model? Use an F-test to check.

```r
d[, ':='(y_dm = mean(yvar)), by=block.num]

# Taking as.factor(treatment.assign) is same as taking any letter assignment
model_block_average <-  d[, lm(yvar ~ as.factor(treatment.assign) + y_dm)]
summary(model_block_average)
```

```
##
## Call:
## lm(formula = yvar ~ as.factor(treatment.assign) + y_dm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.3807 -0.1034 -0.0767 -0.0443  0.9794
##
## Coefficients:
##                                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             -0.000190   0.000281   -0.68  0.49850
## as.factor(treatment.assign)Election info  0.005000   0.001657    3.02  0.00255
## as.factor(treatment.assign)Partisan       0.005285   0.001176    4.50  6.9e-06
## as.factor(treatment.assign)Top-two info   0.004525   0.001176    3.85  0.00012
## y_dm                                      1.000001   0.002564  390.09  < 2e-16
##
## (Intercept)
## as.factor(treatment.assign)Election info **
## as.factor(treatment.assign)Partisan      ***
## as.factor(treatment.assign)Top-two info  ***
## y_dm                                     ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.285 on 3872263 degrees of freedom
## Multiple R-squared:  0.0378, Adjusted R-squared:  0.0378
## F-statistic: 3.81e+04 on 4 and 3872263 DF,  p-value: <2e-16
```

```r
# comparing the simple any letter model with this new model with block averages
f_test_results      <-  anova(model_letters, model_block_average, test='F')
f_test_results
```

```
## Analysis of Variance Table
##
## Model 1: yvar ~ as.factor(treatment.assign)
## Model 2: yvar ~ as.factor(treatment.assign) + y_dm
##    Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1 3872264 327616
## 2 3872263 315228  1     12388 152170 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Including the randomized block level effects does improve the model, as seen in the f-test comparison of the earlier model of *with any letters* with the current model (which includes the *any letters (as factor)* along with the mean of the blocks (that act as blocked fixed effects)).

The coefficient of group means and outcome is appearing as very strong (as seen from the `y_dm` coefficient of 1 and a very small p-value of 0).

7. Doesn't this feel like using a bad-control in your regression? Has the treatment coefficient changed from when you didn't include the `block_average` measure to when you did? Have the standard errors on the treatment coefficient changed from when you didn't include the `block_average` measure to when you did? Why is this OK to do?

The regression of `yvar` on the randomized blocked grouped mean is not a bad control in this case, because of the randomized block assignments.

Basically, when we regress the `yvar` on treatment assignment and block fixed effects, we are essentially telling the model that the means of those blocks are impacting the model in a cetain manner.

Now, it may feel a bad control because the mean of those y-variable blocks can be only be seen as a post-treatment outcome. However, because of the power of randomization, we know that the link between treatment assignment and those blocks (and hence their respective means) has been disconnected.

Further to our point, we illustrate the mean of each of those randomized block outputs below:

```r
head(d[, .(mean(Baseline_Assignment)), by=.(block.num)], 20)
```

```
##     block.num       V1
##  1:        91 0.038723
##  2:       316 0.038619
##  3:       364 0.038622
##  4:       296 0.038462
##  5:       302 0.038687
##  6:       382 0.038462
##  7:       250 0.038462
##  8:       238 0.038684
##  9:       285 0.038462
## 10:       253 0.038462
## 11:       206 0.038462
## 12:       188 0.038686
## 13:       183 0.038462
## 14:       172 0.038710
## 15:       200 0.038462
## 16:       300 0.038664
## 17:       190 0.038691
```

```
## 18:       108 0.038696
## 19:       334 0.038462
## 20:       266 0.038462
```

As seen from the output of means of the first 20 blocks, we see that the means of those randomizly assigned blocks are almost the same. This means that the randomization percentages are same within each of those blocks. Therefore taking the mean of these blocks and regressing the model on those means is same as taking fixed blocks effects – and because of the power of randomization, we know that those blocks do not interfere (or co-vary) with treatment assignments.

Consider Designs

Determine the direction of bias in estimating the ATE for each of the following situations when we randomize at the individual level. Do we over-estimate, or underestimate? Briefly but clearly explain your reasoning.

1. Suppose that you're advertising games – Among Us? – to try and increase sales, and you individually randomly-assign people into treatment and control. After you randomize, you learn that some treatment-group members are friends with control-group members IRL.

Here we have a situation of violation of non-intereference assumtpion, leading to spillover effects between treatment and control.

Even though the treatment group only receives advertisement, the effects of those ads from treated folks spill over to the control group folks via word of mouth. Therefore, this will be a spillover to control and result underestimation (reduction) of ATE resulting in negative bias (because treatment and control both become similar).

2. As we're writing this question, end-of-year bonuses are being given out in people's companies. (This is not a concept we have in the program – each day with your smiling faces is reward enough – and who needs money anyways?) Suppose that you're interested in knowing whether this is a good idea from the point of view of worker productivity and so you agree to randomly assign bonuses to some people. *What might happen to your estimated treatment effects if people learn about the bonuses that others have received?*

In this instance, an argument can be made in both ways. However we argue that this type experiment can causes positive spillovers with underestimation of ATE.

Overestimation of ATE: The leakage of HR bonus information is taken as an incentive based bonus (by the other employees, who have not received this bonus), this will likely deincentivize the workers in the control group (those who havent received bonuses), causing them to be less-motivated and less-productive; thus further widening the gap between control and treatment worker productivity. This creates a further positive direction bias for the estimated ATE leading to overestimation of ATE (difference grows between treatment and control worker productivity).

Underestimate of ATE: Control group members learn about the bonus incentive treatment and think that this must be perhaps because of the higher productive nature of those folks who received the bonuses (in the treatment group). This could lead to control group members being extra motivated. At the same time, treatment group members are now extra motivated because of their new bonus. Leading to higher worker producitivities in both ends, leading to underestimation of ATE (because, now both treatment and control groups are closer to each other in worker productivity levels).