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.

```r
# 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)
```

```r
library(sandwich)
library(lmtest)
library(stargazer)
library(dplyr)
library(ri2)
library(stats)
library(tidyverse)
require(lmtest)
require(sandwich)
library(ivreg)
library(data.table)
library(stargazer)
library(tinytex)
```

(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")

df_hill_kousser_orignial <- readRDS(file = "/Users/Vaibhav_Beohar/Documents/VB_Mck_Docs/MIDS/W241/ps4/p
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] 0.004899234
```

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

```
## [1] 0.09802401
```

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

```
## [1] 0.09312478
```

```
# ITT of treatment vs control
# How it is asked in this assignment
model_simple <-  d[, lm(yvar ~ Baseline_Assignment)]
summary(model_simple)$coefficients
```

```
##                        Estimate    Std. Error    t value     Pr(>|t|)
## (Intercept)         0.093124777 0.0001507552 617.721690 0.000000e+00
## Baseline_Assignment 0.004899234 0.0007669995   6.387532 1.686031e-10
```

```
# 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)         0.09312478 0.00015062 618.2813 < 2.2e-16 ***
## Baseline_Assignment 0.00489923 0.00078340   6.2538 4.006e-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: 0.0931248

ATE of treatment: 0.098024

Total treatment effect (ITT): 0.0048992

Robust standard errors: 0.0931248, 0.0048992, 1.5061878×10^{-4}, 7.833987×10^{-4}, 618.2813249, 6.2538198, 0, $4.0057465 \times 10^{-10}$

We see that the impact of providing any letter (vs no letter) has an ATE of 0.098024 (with a statistically significant p-value of $1.6860314 \times 10^{-10}$). 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.

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

```
##                                            Estimate   Std. Error    t value
## (Intercept)                             0.093124777 0.0001507553 617.721547
## as.factor(treatment.assign)Election info 0.004984642 0.0016893106   2.950696
## as.factor(treatment.assign)Partisan      0.005259706 0.0011984120   4.388896
## as.factor(treatment.assign)Top-two info  0.004496100 0.0011984416   3.751623
##                                              Pr(>|t|)
## (Intercept)                              0.000000e+00
## as.factor(treatment.assign)Election info 3.170605e-03
## as.factor(treatment.assign)Partisan      1.139304e-05
## as.factor(treatment.assign)Top-two info  1.756964e-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
## (Intercept)                              0.09312478 0.00015062 618.2812
## as.factor(treatment.assign)Election info 0.00498464 0.00172728   2.8858
## as.factor(treatment.assign)Partisan      0.00525971 0.00122664   4.2879
## as.factor(treatment.assign)Top-two info  0.00449610 0.00122248   3.6779
##                                           Pr(>|t|)
## (Intercept)                              < 2.2e-16 ***
## as.factor(treatment.assign)Election info 0.0039039 **
## as.factor(treatment.assign)Partisan      1.804e-05 ***
## as.factor(treatment.assign)Top-two info  0.0002352 ***
## ---
## 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?

```r
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  0.017723 0.1047 0.9006
```

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"))
```

```r
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)                        0.09810942 0.00172072 57.0165   <2e-16 ***
## as.factor(treatment.assign)Partisan 0.00027506 0.00210781  0.1305   0.8962
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
rse_2
```

```
##
## t test of coefficients:
##
##                                           Estimate  Std. Error t value
## (Intercept)                             0.09810942  0.00172072  57.017
## as.factor(treatment.assign)Top-two info -0.00048854  0.00210539  -0.232
##                                         Pr(>|t|)
## (Intercept)                               <2e-16 ***
## as.factor(treatment.assign)Top-two info   0.8165
## ---
## 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: 0.0021078

Robust standard error for *Top-Two Info* treatment indicator: 0.0021054

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 0.8961733 and 0.816505 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?

```r
#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?

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

```
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
## -0.38074 -0.10342 -0.07671 -0.04433  0.97938
##
## Coefficients:
##                                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                             -0.0001904  0.0002812  -0.677 0.498504
## as.factor(treatment.assign)Election info  0.0049996  0.0016571   3.017 0.002552
## as.factor(treatment.assign)Partisan       0.0052850  0.0011755   4.496 6.93e-06
## as.factor(treatment.assign)Top-two info   0.0045248  0.0011756   3.849 0.000119
## y_dm                                      1.0000014  0.0025635 390.090  < 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: 0.2853 on 3872263 degrees of freedom
## Multiple R-squared:  0.03782,    Adjusted R-squared:  0.03782
## F-statistic: 3.805e+04 on 4 and 3872263 DF,  p-value: < 2.2e-16
```

```
# 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 < 2.2e-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 1.0000014 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 0.03872303
##  2:       316 0.03861852
##  3:       364 0.03862179
##  4:       296 0.03846154
##  5:       302 0.03868684
##  6:       382 0.03846154
##  7:       250 0.03846154
##  8:       238 0.03868386
##  9:       285 0.03846154
## 10:       253 0.03846154
## 11:       206 0.03846154
## 12:       188 0.03868590
## 13:       183 0.03846154
## 14:       172 0.03871035
## 15:       200 0.03846154
## 16:       300 0.03866397
## 17:       190 0.03869067
## 18:       108 0.03869642
## 19:       334 0.03846154
## 20:       266 0.03846154
```

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.