Exercise: Analyzing Field Experiments

Following on from our simulation of omitted variable bias last week, we will now simulate an experimental analysis to see if we can avoid this bias. You can adapt your script from last week or start again, the first 4 steps are the same:

1. Generate data on a population of 1,000 people. Specifically, create a variable (a vector) that randomly assigns these people to be male or female (50:50). *Hint: In R, try rbinom and in Stata, try rbinomial.*

```
N <- 1000
x <- rbinom(N,1,0.5)
```

2. Now we are going to simulate the potential outcomes - a measure of attitudes - *if our units were not treated* (y_0) for our population. Create another variable of random normally-distributed values with mean of 5 and standard deviation of 1. *Hint: In R, try rnorm and in Stata, try rnormal.*

```
y0 <- rnorm(N,5,1)
```

3. One problem with observational data is that potential outcomes are often correlated with other variables such as gender. Adjust your value of y_0 to add 1 (one) for all units who are male.

```
y0 <- y0 + x
```

4. Now simulate potential outcomes *if the units receive treatment* (y_1) for all units. Define a *constant* treatment effect of $c = 2$ and create another variable $y_1 = y_0 + c$.

```
c <-2
y1 <- y0 + c
```

5. Next, let's assume a specific Treatment Assignment Mechanism – random assignment. Unlike last week, treatment assignment will *not* depends on gender (x) or potential outcomes. Create a treatment variable D that gives each unit a 50% chance of binary treatment (like a coin flip). *Hint: In R, try rbinom and in Stata, try rbinomial.*

```
data <- tibble(x,y0,y1) %>%
  mutate(D=rbinom(N,1,0.5))
```

6. Are gender and treatment correlated (they were strongly last week, remember)? Calculate the correlation between x and D.

```
cor(data$x,data$D)
```

```
## [1] -0.00798816
```

7. In practice, we only observe one outcome value: y_{obs}. Create a new variable y_{obs} which equals y_1 if $D = 1$ but which equals y_0 if $D = 0$.

```
data <- data %>% mutate(y_obs=case_when(D==1~y1,
                                        D==0~y0))
```

8. Compare the balance of observed pre-treatment covariates (gender) in the treatment and control groups using a difference-in-means t-test. Interpret the result.

```
data %>% t.test(x ~ D, data=.)
```

```
##
```

```
##  Welch Two Sample t-test
##
## data:  x by D
## t = 0.25236, df = 997.85, p-value = 0.8008
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.05412766  0.07010424
## sample estimates:
## mean in group 0 mean in group 1
##       0.5030181       0.4950298
```

9. Compare the **balance** of observed potential outcomes (y_0, y_1) in the treatment and control groups using a difference-in-means t-test for each. Interpret the results.

```
data %>% t.test(y0 ~ D, data=.)
```

```
##
##  Welch Two Sample t-test
##
## data:  y0 by D
## t = -0.87037, df = 996.97, p-value = 0.3843
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.20489870  0.07898633
## sample estimates:
## mean in group 0 mean in group 1
##        5.454504        5.517460
```

```
data %>% t.test(y1 ~ D, data=.)
```

```
##
##  Welch Two Sample t-test
##
## data:  y1 by D
## t = -0.87037, df = 996.97, p-value = 0.3843
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.20489870  0.07898633
## sample estimates:
## mean in group 0 mean in group 1
##        7.454504        7.517460
```

10. Based on the observable data, analyze the results of the experiment using a difference-in-means t-test of outcomes (y_{obs}) by treatment status (D). Interpret the result. Is this an accurate estimate of the treatment effect that we created at the start?

```
data %>% t.test(y_obs ~ D, data=.)
```

```
##
##  Welch Two Sample t-test
##
## data:  y_obs by D
## t = -28.52, df = 996.97, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.204899 -1.921014
## sample estimates:
```

```
## mean in group 0 mean in group 1
##        5.454504        7.517460
```

11. As an alternative, run the basic regression of observable outcomes (y_{obs}) on treatment (D). How does this compare to the difference-in-means test above?

```
data %>% lm(y_obs ~ D, data=.) %>%
  stargazer(header=F, keep.stat=c("n"))
```

Table 1:

	Dependent variable:
	y_obs
D	2.063***
	(0.072)
Constant	5.455***
	(0.051)
Observations	1,000
Note:	*p<0.1; **p<0.05; ***p<0.01

12. Now add the control variable x to the regression. How do the results change? Why might we want to include this control variable?

```
data %>% lm(y_obs ~ D + x, data=.) %>%
  stargazer(header=F, keep.stat=c("n"))
```

Table 2:

	Dependent variable:
	y_obs
D	2.071***
	(0.066)
x	0.972***
	(0.066)
Constant	4.966***
	(0.057)
Observations	1,000
Note:	*p<0.1; **p<0.05; ***p<0.01

13. Now let's try to understand how things change when treatment is clustered at a higher level, between 20 groups (so each cluster contains about 50 people):

a. First, we want these groups' members to be similar to each other, so we are going to group people by the value of their y_0. In your dataset, add an additional column (*cluster*) which transforms ('bins') the y_0 variable into a categorical variable with 20 categories, where category '1' represents the lowest values of y_0 and category '20' contains the highest values of y_0. *Hint: Try the ntile function in R*

	Model 1
(Intercept)	5.47***
	(0.05)
D_cluster	2.03***
	(0.07)
R^2	0.44
Adj. R^2	0.44
Num. obs.	1000
RMSE	1.14

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table 3: Statistical models

	Model 1
(Intercept)	5.47***
	(0.34)
D_cluster	2.03**
	(0.54)
R^2	0.44
Adj. R^2	0.44
Num. obs.	1000
RMSE	1.14

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$

Table 4: Statistical models

```
data <- data %>% mutate(cluster=as.factor(ntile(y0,20)))
```

b. Add a new binary treatment variable, *D_cluster*, that randomly assigns each cluster to treatment or control (so treatment is at the level of the cluster). *Hint: Try the cluster_ra function in the randomizr package in R and Stata*

```
library(randomizr)
```

```
## Warning: package 'randomizr' was built under R version 3.6.1
```

```
data <- data  %>% mutate(D_cluster=cluster_ra(cluster, m=10))
```

c. Recalculate y_{obs} based on the new treatment variable, *D_cluster*.

```
data <- data %>% mutate(y_obs_cluster=case_when(D_cluster==1~y1,
                                                D_cluster==0~y0))
```

d. Now run the regression of observable outcomes (y_{obs}) on clustered treatment (*D_cluster*). What is the confidence interval around the size of the effect *D_cluster*? Is this accurate?

```
data %>% lm(y_obs_cluster ~ D_cluster, data=.) %>%
  texreg(include.ci=F)
```

e. Now run the same regression again, but this time cluster the standard errors by the *cluster* variable. How does the confidence interval on *D_cluster* differ from the previous analysis without clustered standard errors? *Try lm_robust in the estimatr package for clustered standard errors*

```
data %>% lm_robust(y_obs_cluster ~ D_cluster, data=., clusters=cluster) %>%
  texreg(include.ci=F)
```

14. (Advanced) Repeat the experiment and regression from Question 11 100 times with the same x, y_0 and

y_1 every time, but allow treatment assignment D to be re-randomized each time. For each experiment, perform the basic regression and calculate the 95% confidence interval on the treatment variable. Finally, calculate how many of the 100 confidence intervals contain (cover) the real treatment effect ($c = 2$).

```
out <- list()
for (i in 1:100) {
  out[[i]] <- data %>% mutate(D=rbinom(N,1,0.5),
                              y_obs=case_when(D==1~y1,
                                              D==0~y0)) %>%
    lm(y_obs ~ D + x, data=.) %>%
    tidy() %>%
    filter(term=="D") %>%
    mutate(conf.lo=estimate-1.96*std.error,
           conf.hi=estimate+1.96*std.error) %>%
    select(conf.lo, conf.hi)
}

out %>% bind_rows() %>%
  mutate(covers_2=ifelse(conf.lo>2|conf.hi<2,0,1)) %>%
  tally(covers_2)
```

```
## # A tibble: 1 x 1
##       n
##   <dbl>
## 1    96
```