hw3

1

```r
dgp_p1 <- function() {
  ### Define number of schools, and number of students in the school
  G <- 50 # number of schools
  ng <- 10 # number of students in each school
  n <- G*ng # total n-size

  ### Assign students to schools
  school <-rep(seq(1,G), ng)
  school <- sort(school)

  ### Gamma (school-varying intercept for PO's)
  gamma <- rnorm(G, sd=sqrt(2))[school]

  ### Generate potential outcomes
  y0 <- gamma + rnorm(n)
  y1 <- 2 + gamma + rnorm(n)

  ### Put everything into a data-frame
  data <- data.frame("school" = school, "y0" = y0, "y1" = y1)

  return(data)
}
```

a

There are 10 students per school and 50 schools total, meaning taht there are 500 total students. The ATE here is 2, and the factors that influence the potential outcomes are some random noise from the $\mathcal{N}(0,1)$ instances added to each one, as well as the school specific random intercept called 'gamma'.

b

1) Complete: Randomly draw 250 students to be treated
2) Bernoulli: Probability of treatment for each student is 0.5
3) Cluster: Randomly draw 25 schools to be treated
4) Statified: Within each school, randomly draw 5 students to be treated

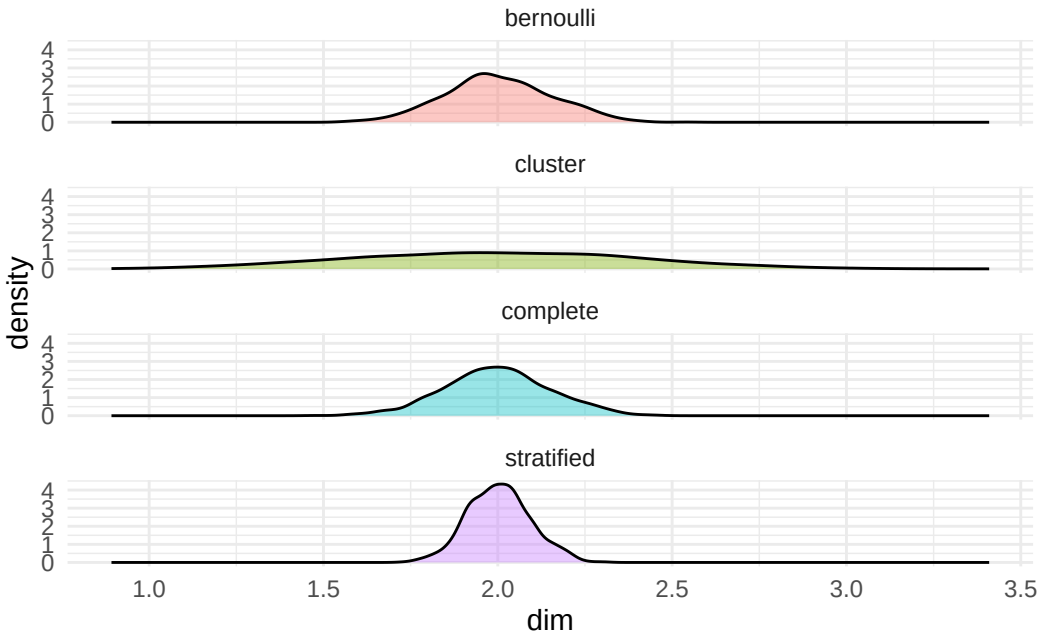
```r
# empty results df to fill
res <- tibble(
  complete = rep(0, 1000),
  bernoulli = rep(0, 1000),
  cluster = rep(0, 1000),
  stratified = rep(0, 1000)
)

for (i in 1:1000) {
  data <- dgp_p1()

  # complete
  row_ids <- sample(1:500, 250)
  complete <- data %>%
    mutate(
      id = row_number(),
      D = if_else(id %in% row_ids, 1, 0)
      ) %>%
    select(-id)

  # bernoulli
  bernoulli <- data %>%
    mutate(D = rbinom(500, 1, 0.5))

  # cluster
  school_ids <- sample(1:50, 25)
  cluster <- data %>%
    group_by(school) %>%
    mutate(
      group_id = cur_group_id(),
      D = if_else(group_id %in% school_ids, 1 , 0)
    ) %>%
    ungroup() %>%
    select(-group_id)
```

```r
  # stratified
  within_school_ids <- data %>%
    group_by(school) %>%
    mutate(chosen_id = row_number()) %>%
    slice_sample(n = 5) %>%
    ungroup() %>%
    select(school, chosen_id) %>%
    mutate(D = 1)

  stratified <- data %>%
    group_by(school) %>%
    mutate(
      id = row_number()
      ) %>%
    ungroup() %>%
    left_join(
      within_school_ids, by = c("id" = "chosen_id", "school")
      ) %>%
    mutate(D = replace_na(D, 0)) %>%
    select(-id)

  data_list <- list(complete, bernoulli, cluster, stratified)

  dim <- function(data){

    return(mean(data[data$D == 1, ]$y1) - mean(data[data$D == 0, ]$y0))

  }

  res[i, ] <- data_list %>%
    map(.f = dim)

}
```

c

Each design is indeed unbiased for the ATE. The variances for the bernoulli and complete randomizations are pretty similar, while the cluster randomization has the largest variance and the stratified randomization has the smallest variance.

```r
res %>%
  pivot_longer(
    cols = everything(),
    names_to = "sampling_mechanism",
    values_to = "dim"
    ) %>%
  ggplot(aes(x = dim, fill = sampling_mechanism)) +
  geom_density(alpha = 0.4, show.legend = F) +
  facet_wrap(~ sampling_mechanism, ncol = 1) +
  theme_minimal()
```




```r
res %>%
  summarise(across(everything(), var))
```

```
# A tibble: 1 x 4
  complete bernoulli cluster stratified
     <dbl>     <dbl>   <dbl>      <dbl>
1   0.0228    0.0233   0.171    0.00808
```

d

I think it makes sense that cluster randomization would have the highest variance because there are sort of two sources of variability being incorporated: variability within each chosen cluster and the variability between all of the clusters. The data is generated such that there is already variability between the clusters and by re-randomizing in such a way that entire clusters are randomly selected to be treated we essentially enhance that source of variability.

One reason that we would expect the stratified randomization to have the lowest variance is because, unlike in the cluster randomization case, we don't have extra between-cluster variance. I think this is also reason that it has lower variance than the bernoulli and complete randomization cases, as in each re-randomization we ensure that observations from each cluster are included in the treatment. This gives us better balance across clusters and thus avoids more of that unwanted between-cluster variance.

2

a

Making sure to only perform the analysis on units that are eligible to receive the subsidy, we simply regress our response variable on the treatment category column.

```
data_raw <- read.dta13("/Users/joshuayamamoto/Downloads/BD-SAN-FINAL.dta") %>%
  mutate(r4_recode = case_when(
    r4_any_od_adults == "Yes" ~ 1L,
    r4_any_od_adults == "No" ~ 0L
  ))


data <- data_raw %>%
  filter(eligible == "Eligible") %>%
  select(r4_recode, treat_cat_3) %>%
  drop_na()

coef(summary(lm(r4_recode ~ treat_cat_3 , data)))[ -1,1] %>%
  as.data.frame() %>%
  rename('Estimate' = '.')
```

```
                          Estimate
treat_cat_3LPP Only      -0.07935631
treat_cat_3Supply Only   -0.09581019
treat_cat_3Loser (Low)   -0.09035256
treat_cat_3Loser (Med)   -0.15611116
```

```
treat_cat_3Loser (High)  -0.13388554
treat_cat_3Winner (Low)  -0.10971356
treat_cat_3Winner (Med)  -0.19489560
treat_cat_3Winner (High) -0.16496828
```

b

I think that the main reason has to do with the fact that it would spark controversy and issues within clusters if some individuals were afforded better treatments than others. Furthermore there would likely be a lot of spillover if the treatment was individually assigned since friends and family would probably let each other benefit from their treatment.

c

```r
randomization_scheme <- function() {

  ## 1: Cluster randomization of villages to treatment conditions.
  ## note: stratify by number of neighborhoods per village (1-2 and 3+)

  ## Control, LPP Only, LPP + Subsidy, Supply only.

  # first figure out the strata that each village belongs to
  strata_labels <- data_raw %>%
    distinct(vid, cid) %>%
    count(vid) %>%
    mutate(strata = case_when(
      n < 3 ~ "s1",
      n >= 3 ~ "s2"
    )) %>%
    select(-n)

  # extract strata labels for each village
  s1_villages <- strata_labels[strata_labels$strata == "s1", ]$vid
  s2_villages <- strata_labels[strata_labels$strata == "s2", ]$vid

  ## s1 has 50 villages
  ## s2 has 57 villages

  ## 22 in control, 12 LPP only, 63 in LPP + Subsidy, 10 in Supply Only
  ## (following SM from paper)
```

```r
## they overweight the LPP + Subsidy group since it will be further
## partitioned in the next stage

# randomly assign the first level of treatment conditions to entire villages by strata
s1_clusters <- sample(
  c(
    rep("Control", 11), rep("LPP_Only", 6),
    rep("LPP+Subsidy", 29), rep("Supply_Only", 4)
    ),
  size = 50, replace = F
  )

s2_clusters <- sample(
  c(
    rep("Control", 11), rep("LPP_Only", 6),
    rep("LPP+Subsidy", 34), rep("Supply_Only", 6))
  ,
  size = 57, replace = F
  )

# create new tibble with the treatment that each village belongs to
cluster_labels <- tibble(
  vid = c(s1_villages, s2_villages),
  treat1 = c(s1_clusters, s2_clusters)
)

# rejoin strata and treatment labels to original data
data_s1 <- data_raw %>%
  left_join(strata_labels, by = "vid") %>%
  left_join(cluster_labels, by = "vid")

## 2. Cluster randomization of neighborhoods in the "LPP + Subsidy" condition
## into a 2x3 set of subtreatments indicating if they get just the subsidy,
## or the subsidy + supply, and the level of the intensity of the vouchers.

## first cluster randomize into LPP + Subsidy and LPP + Subsidy + Supply

## this is done at the neighborhood level so first get a list of the
## distinct neighborhoods who received "LPP + Subsidy"
n_cid <- data_s1 %>%
  filter(treat1 == "LPP+Subsidy") %>%
```

```
  distinct(cid) %>%
  nrow()

# recover how many cid should be given LPP + Subsidy and LPP + Subsidy + Supply
# I chose to just balance this as best as possible

# this code just deals with the fact that sometimes the number of neighborhoods
# that need to be reassigned is an odd number
if (n_cid %% 2 == 1) {
  n1 <- floor(n_cid/2)
  n2 <- floor(n_cid/2) + 1
} else {
  n1 <- n_cid/2
  n2 <- n_cid/2
}

# randomly assign neighborhoods within LPP + Subsidy to either LPP + Subsidy
# or LPP + Subsidy + Supply
stage2_labels <- data_s1 %>%
  filter(treat1 == "LPP+Subsidy") %>%
  distinct(cid) %>%
  mutate(
    stage2grp = sample(
      c(rep("LPP+Subsidy", n1), rep("LPP+Subsidy+Supply", n2)), n1 + n2
      )
    )

# within LPP + Subsidy neighborhoods do a bernoulli random assignment of intensity
p1 <- stage2_labels %>%
  filter(stage2grp == "LPP+Subsidy") %>%
  mutate(intensity = sample(c("High", "Medium", "Low"), n1, replace = T))


# within LPP + Subsidy + Supply neighborhoods do a bernoulli random
# assignment of intensity
p2 <- stage2_labels %>%
  filter(stage2grp == "LPP+Subsidy+Supply") %>%
  mutate(intensity = sample(c("High", "Medium", "Low"), n2, replace = T))

stage2_labels_final <- rbind(p1, p2)
```

```r
# join the intensity labels to the original data by neighborhood
data_s2 <- data_s1 %>%
  left_join(stage2_labels_final, by = "cid") %>%
  mutate(treat1 = case_when(
    is.na(stage2grp) ~ treat1,
    T ~ stage2grp
  )) %>%
  select(-stage2grp) %>%
  mutate(treat2 = case_when(
    is.na(intensity) ~ treat1,
    T ~ intensity
  )) %>%
  select(-intensity)

## 3: Randomization of the vouchers for households in the "LPP + Subsidy"
## conditions, given their intensity assignment.
## note: by household

# high- 60%
# medium- 50%
# low- 40%
# I looked at roughly how many winners there were per category in the randomization
# in the original data set. I ran the commented out code below to get this
# data_raw %>%
#   group_by(treat_cat_3) %>%
#   summarise(n = n_distinct(hhid))

# use bernoulli randomization to assign winners of the lottery by
# household to low intensity neighborhoods using the varying
# success probabilities from above

s2_1 <- data_s2 %>%
  filter(treat2 == "Low") %>%
  mutate(lottery = sample(
    c("Winner", "Loser"), n(),
    replace = T, prob = c(0.4, 0.6)
    ))

s2_2 <- data_s2 %>%
  filter(treat2 == "Medium") %>%
  mutate(lottery = sample(
```

```
      c("Winner", "Loser"), n(),
      replace = T, prob = c(0.5, 0.5)
      ))

  s2_3 <- data_s2 %>%
    filter(treat2 == "High") %>%
    mutate(lottery = sample(
      c("Winner", "Loser"), n(),
      replace = T, prob = c(0.6, 0.4)
      ))

  # formatting
  lottery_labels <- rbind(s2_1, s2_2, s2_3) %>%
    mutate(lottery = paste0(lottery, " (", treat2, ")")) %>%
    select(hhid, lottery)

  # rejoin everything together
  data_s3 <- data_s2 %>%
    left_join(lottery_labels, by = "hhid") %>%
    mutate(treat3 = case_when(
      is.na(lottery) ~ treat2,
      T ~ lottery
    )) %>%
    select(-lottery) %>%
    select(hhid, cid, vid, r4_recode, treat3, treat_cat_3)

  return(data_s3)

}
```

d

The results from the randomization inference are saved and reloaded in for the analysis portion so that the code doesn't have to be re-run every single time

```
null_dist <- data.frame(dim = rep(0, 5000))

for(i in 1:5000) {
  iter <- randomization_scheme()
  iter <- iter %>%
```

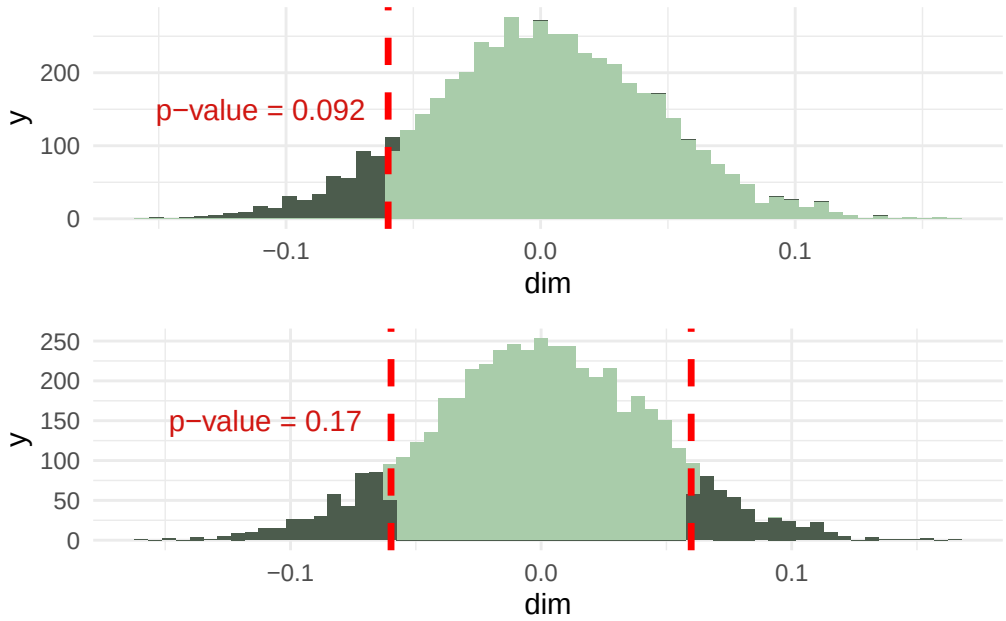
```
    filter(treat3 == "Winner (Low)" | treat3 == "Control") %>%
    drop_na()

  null_dist[i, ] <- coef(lm(r4_recode ~ treat3, iter))[2]
}
```

```
null_dist <- read_csv(here("data", "hw3_p2_null_dist.csv"))

data_sub <- data_raw %>%
  filter(treat_cat_3 == "Winner (Low)" | treat_cat_3 == "Control") %>%
   mutate(r4_recode = case_when(
      r4_any_od_adults == "Yes" ~ 1L,
      r4_any_od_adults == "No" ~ 0L
    )) %>%
  drop_na()

true_dim <- coef(lm(r4_recode ~ treat_cat_3, data_sub))[2]

p_val_one_sided <- mean(null_dist$dim < true_dim)
p_val_two_sided <- mean(null_dist$dim < -abs(true_dim)) +
  mean(null_dist$dim > abs(true_dim))

one_sided <- null_dist %>%
  mutate(filler = ifelse(dim > true_dim, "more", "less")) %>%
  ggplot(aes(x = dim, fill = filler)) +
  geom_histogram(binwidth = 0.0058, show.legend = F) +
  scale_fill_manual(values = c("#4c5c4d", "#a9ccaa")) +
  geom_vline(
    xintercept = true_dim, color = "red",
    linetype = "dashed", linewidth = 1.2
    ) +
  theme_minimal() +
  annotate(
    geom = "text", x = -0.11, y = 150,
    label = "p-value = 0.092", color = "#d11611"
      )



two_sided <- null_dist %>%
  mutate(filler = ifelse(
```

```
    dim < -abs(true_dim) | dim > abs(true_dim), "shade", "no-shade"
    )) %>%
  ggplot(aes(x = dim, fill = filler)) +
  geom_histogram(binwidth = 0.0055, show.legend = F) +
  geom_vline(
    xintercept = -abs(true_dim), color = "red",
    linetype = "dashed", linewidth = 1.2
    ) +
  geom_vline(
    xintercept = abs(true_dim), color = "red",
    linetype = "dashed", linewidth = 1.2
    ) +
  scale_fill_manual(values = c("#a9ccaa", "#4c5c4d")) +
  theme_minimal() +
  annotate(
    geom = "text", x = -0.11, y = 150,
    label = "p-value = 0.17", color = "#d11611"
      )
```

We can interpret the p-value as follows: the probability of observing a similar treatment effect under the null hypothesis (which says that the treatment effect is zero for all units) is 0.092. And the probability of observing a similar absolute treatment effect under the sharp null is 0.17.

```
one_sided / two_sided
```



Because these p-values are not significant (not even borderline) we cannot reject the sharp null. Essentially we have shown that across many hypothetical treatment randomizations under the sharp null, we would still sometimes expect to see the observed treatment effect.

e

I don't think that this difference in results necessarily is due to differences in the randomization scheme since I tried to very closely follow

f

```
grid <- true_dim + seq(from = -0.25, to = 0.25, length.out = 300)

Null_Dist_df <- matrix(data = rep(0, 300000), ncol = 300)

for(i in 1:1000){
  d_new <- randomization_scheme()$treat3
  New_DIM <- c(rep(0, 300))
  for(v in 1:length(grid)){
    new_po <- data_raw %>%
      mutate(D_old = case_when(
        treat_cat_3 == "Control" ~ 0,
        T ~ 1
      )) %>%
      mutate(D_new_raw = d_new) %>%
      mutate(D_new = case_when(
        D_new_raw == "Control" ~ 0,
        T ~ 1
      )) %>%
      select(r4_recode, D_old, D_new) %>%
      drop_na() %>%
      mutate(
        Y_0 = r4_recode - D_old*grid[v],
        Y_1 = r4_recode + (1-D_old)*grid[v],
        Y_new = D_new*Y_1 + (1 - D_new)*Y_0
        )

    fit <- lm(Y_new ~ D_new, new_po)
    dim_new <- fit$coefficients["D_new"]
    New_DIM[v] <- dim_new
```

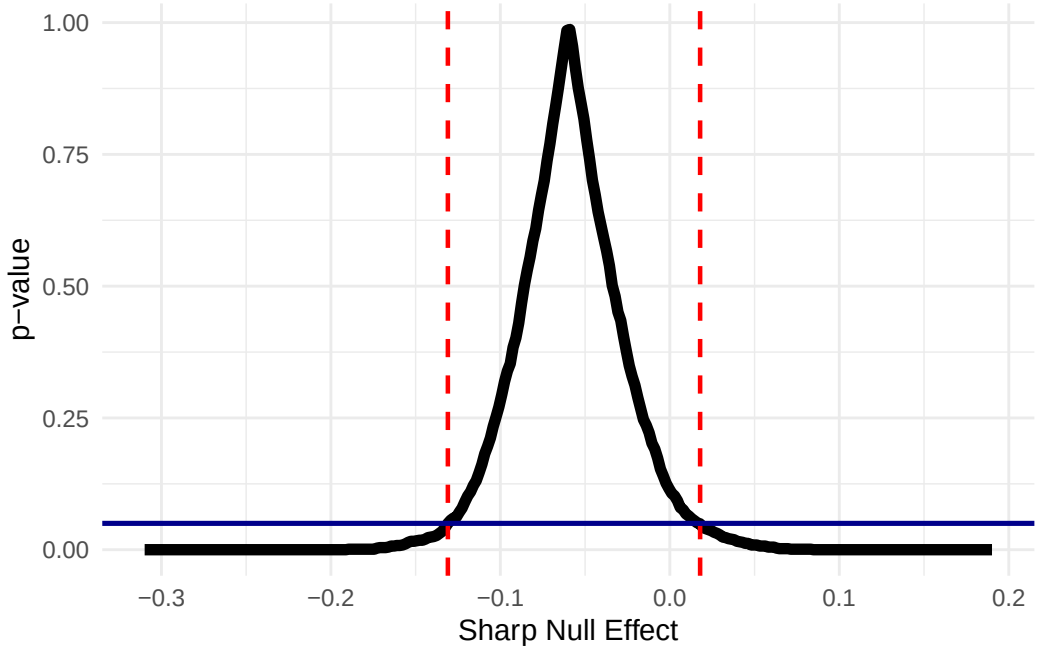
```
  }
  Null_Dist_df[i, ] <- New_DIM



}
```

```
grid <- true_dim + seq(from = -0.25, to = 0.25, length.out = 300)
Null_Dist_df <- read_csv(here("data", "hw3_p2_fisher_ci.csv"))

p_value_vec <- rep(0, 300)

for(i in 1:length(grid)){
  # Get corresponding null distribution and null value
  temp_null_dist <- Null_Dist_df[ ,i]
  v <- grid[i]
  # absolute difference between observed and assumed
  diff <- abs(true_dim-v)
  # Get adjusted p-value -- probability of as far or further away from assumed
  temp_p <-
    mean(temp_null_dist >= v + diff) +
    mean(temp_null_dist <= v - diff)
  # Add to p-value vector
  p_value_vec[i] <- temp_p
}
```

```
over <- which(p_value_vec>0.05)
index_lwr <- over[1]-1
index_upr <- over[length(over)]+1
ci <- c(grid[index_lwr], grid[index_upr])
```

```
tibble(
  p_value = p_value_vec,
  sharp_null_effect = grid
) %>%
  ggplot(aes(x = sharp_null_effect, y = p_value)) +
  geom_line(linewidth = 2) +
  geom_vline(
    xintercept = ci[1], linetype = "dashed",
    linewidth = 0.8, color = "red"
```

```
    ) +
geom_vline(
  xintercept = ci[2], linetype = "dashed",
  linewidth = 0.8, color = "red"
    ) +
geom_hline(
  yintercept = 0.05, color = "darkblue",
  linewidth = 0.9
  ) +
theme_minimal() +
labs(
  x = "Sharp Null Effect",
  y = "p-value"
)
```



This confidence interval (-0.131, 0.0179) tells us the range of observed treatment effect within which we would not reject the sharp null hypothesis.

3

a

$$
\begin{aligned}
-B_n - E_n &= -\left(\frac{1}{n_1}\sum_{D_i=1} f_1(X_i) - \frac{1}{n}\sum_{i=1}^{n} f_1(X_i)\right) + \left(\frac{1}{n_0}\sum_{D_i=0} f_0(X_i) - \frac{1}{n}\sum_{i=1}^{n} f_0(X_i)\right) - \\
&\quad \left(\frac{1}{n_1}\sum_{D_i=1} \varepsilon_i(1) - \frac{1}{n}\sum_{i=1}^{n} \varepsilon_i(1)\right) + \left(\frac{1}{n_0}\sum_{D_i=0} \varepsilon_i(0) - \frac{1}{n}\sum_{i=1}^{n} \varepsilon_i(0)\right) \\
&= -\frac{1}{n_1}\sum_{D_i=1} f_1(X_i) + \frac{1}{n_0}\sum_{D_i=0} f_0(X_i) - \frac{1}{n_1}\sum_{D_i=1} \varepsilon_i(1) + \frac{1}{n_0}\sum_{D_i=0} \varepsilon_i(0) + \text{SATE} \\
&= -\frac{1}{n_1}\sum_{D_i=1} \left[f_1(X_i) + \varepsilon_i(1)\right] + \frac{1}{n_0}\sum_{D_i=0} \left[f_0(X_i) + \varepsilon_i(0)\right] + \text{SATE} \\
&= -\frac{1}{n_1}\sum_{D_i=1} Y_i(1) + \frac{1}{n_0}\sum_{D_i=0} Y_i(0) + \text{SATE}
\end{aligned}
$$

so now we can say that

$$
\begin{aligned}
\hat{\tau}_{dim} - B_n - E_n &= \frac{1}{n_1}\sum_{D_i=1} Y_i(1) - \frac{1}{n_0}\sum_{D_i=0} Y_i(0) - \frac{1}{n_1}\sum_{D_i=1} Y_i(1) + \frac{1}{n_0}\sum_{D_i=0} Y_i(0) + \text{SATE} \\
&= \text{SATE}
\end{aligned}
$$

b

First recall that we can express $\hat{\tau}_{ols}$ as follows by regressing $(Y - X^T\hat{\beta}_{ols})$ on D. If we simplify the expression down we get the following:

$$
\begin{aligned}
\hat{\tau}_{ols} &= \left(\frac{1}{n_1}\sum_{D_i=1} Y_i - \frac{1}{n_0}\sum_{D_i=1} Y_i\right) - \left(\frac{1}{n_1}\sum_{D_i=1} X_i^T\hat{\beta}_{ols} - \frac{1}{n_0}\sum_{D_i=0} X_i^T\hat{\beta}_{ols}\right) \\
&= \hat{\tau}_{dim} - \left(\frac{1}{n_1}\sum_{D_i=1} X_i^T\hat{\beta}_{ols} - \frac{1}{n_0}\sum_{D_i=0} X_i^T\hat{\beta}_{ols}\right)
\end{aligned}
$$

Next we can simplify the term $-\hat{\beta}_{n,ols}$ as follows:

$$
\begin{aligned}
-\hat{\beta}_{n,ols} &= -\left(\frac{n_1\hat{\tau}_{ols}}{n_1} + \frac{1}{n_1}\sum_{D_i=1} X_i^T\hat{\beta}_{ols} - \frac{n\hat{\tau}_{ols}}{n} - \frac{1}{n}\sum_{i=1}^{n} X_i^T\hat{\beta}_{ols}\right) + \left(\frac{1}{n_0}\sum_{D_i=0} X_i^T\hat{\beta}_{ols} - \frac{1}{n}\sum_{i=1}^{n} X_i^T\hat{\beta}_{ols}\right) \\
&= -\left(\frac{1}{n_1}\sum_{D_i=1} X_i^T\hat{\beta}_{ols} - \frac{1}{n}\sum_{i=1}^{n} X_i^T\hat{\beta}_{ols}\right) + \left(\frac{1}{n_0}\sum_{D_i=0} X_i^T\hat{\beta}_{ols} - \frac{1}{n}\sum_{i=1}^{n} X_i^T\hat{\beta}_{ols}\right) \\
&= -\frac{1}{n_1}\sum_{D_i=1} X_i^T\hat{\beta}_{ols} + \frac{1}{n_0}\sum_{D_i=0} X_i^T\hat{\beta}_{ols} \\
&= -\left(\frac{1}{n_1}\sum_{D_i=1} X_i^T\hat{\beta}_{ols} - \frac{1}{n_0}\sum_{D_i=0} X_i^T\hat{\beta}_{ols}\right)
\end{aligned}
$$

Finally if we take $\hat{\tau}_{dim} - \hat{\beta}_{n,ols}$ along with the first equality that we recovered, we get

$$
\begin{aligned}
\hat{\tau}_{dim} - \hat{\beta}_{n,ols} &= \hat{\tau}_{dim} - \left(\frac{1}{n_1}\sum_{D_i=1} X_i^T\hat{\beta}_{ols} - \frac{1}{n_0}\sum_{D_i=0} X_i^T\hat{\beta}_{ols}\right) \\
&= \hat{\tau}_{ols}
\end{aligned}
$$

as desired

c

$$
\begin{aligned}
\hat{\tau}_{Lin} &= \underset{\tau}{\text{argmin}} \sum_{i=1}^{n}\left[Y_i - \left(\tau D_i + (X_i - \bar{X})^T\hat{\beta}_{Lin} + D_i(X_i - \bar{X})^T\hat{\gamma}_{Lin}\right)\right]^2 \\
&= \underset{\tau}{\text{argmin}} \sum_{i=1}^{n}\left[\left(Y_i - (X_i - \bar{X})^T\hat{\beta}_{Lin} - D_i(X_i - \bar{X})^T\hat{\gamma}_{Lin}\right) - \tau D_i\right]^2 \\
&= \frac{1}{n_1}\sum_{D_i=1}\left(Y_i - (X_i - \bar{X})^T\hat{\beta}_{Lin} - (X_i - \bar{X})^T\hat{\gamma}_{Lin}\right) - \frac{1}{n_0}\sum_{D_i=0}\left(Y_i - (X_i - \bar{X})^T\hat{\beta}_{Lin}\right) \\
&= \underbrace{\left[\frac{1}{n_1}\sum_{D_i=1} Y_i - \frac{1}{n_0}\sum_{D_i=0} Y_i\right]}_{\hat{\tau}_{dim}} - \frac{1}{n_1}\sum_{D_i=1}(X_i - \bar{X})^T\hat{\beta}_{Lin} - \frac{1}{n_1}\sum_{D_i=1}(X_i - \bar{X})^T\hat{\gamma}_{Lin} + \frac{1}{n_0}\sum_{D_i=0}(X_i - \bar{X})^T\hat{\beta}_{L} \\
&= \hat{\tau}_{dim}
\end{aligned}
$$