ECON 773: Assignment 2

The BLUE Team, Jeneta Ljutic (400138620), Tadhg Taylor-McGreal (400330297), Stella Till (400364649)

2026-01-19

Table of contents

Preface

Goal

The goals of this assignment are to:

- explore the properties of the OLS estimator using Monte Carlo experiments
- analyze data from an RCT using `lm_robust` and `lm_lin`

Instructions

See assignment 1.

Monte Carlo experiments

Monte Carlo 1: coin flips

This question will be demonstrated in class, as an introduction to Monte Carlo experiments.

Flip a coin once (1 for heads, 0 for tails), and store in X_1:

```
(X_1 <- sample(0:1, size = 1, replace = TRUE))
```

```
[1] 1
```

The variable n will refer to sample size. Roll a dice n times:

```
n <- 10
(X_n <- sample(0:1, size = n, replace = TRUE))
```

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

We will use a large set of packages for this assignment, loaded in the next chunk. Do you recognize them?

```
install.packages("gtsummary")
install.packages("gt")
install.packages("gtsummary")
install.packages("estimatr")
install.packages("AER")
```

```
library(tidyverse)
```

```
── Attaching core tidyverse packages ──────────────────── tidyverse 2.0.0
──
✔ dplyr     1.1.4     ✔ readr     2.1.6
```

```
✔ forcats   1.0.1     ✔ stringr   1.6.0
✔ ggplot2   4.0.1     ✔ tibble    3.3.0
✔ lubridate 1.9.4     ✔ tidyr     1.3.2
✔ purrr     1.2.0
── Conflicts ──────────────────────────────────── tidyverse_conflicts()
──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
```

```
library(gt)
library(gtsummary)
library(estimatr)
library(broom)
```

To compute the mean up to and including the ith throw, we can use the `cummean` function from `dplyr`:

```
Xbar_in <- cummean(X_n)
cbind(X_n, Xbar_in)
```

```
      X_n   Xbar_in
 [1,]   0 0.0000000
 [2,]   0 0.0000000
 [3,]   0 0.0000000
 [4,]   1 0.2500000
 [5,]   1 0.4000000
 [6,]   1 0.5000000
 [7,]   1 0.5714286
 [8,]   1 0.6250000
 [9,]   0 0.5555556
[10,]   1 0.6000000
```

If you only want the mean across all n flips, can do:

```
(Xbar_n <- mean(sample(0:1, size = n, replace = TRUE)))
```

```
[1] 0.3
```

We can use the following function to generate a tibble that keeps track across flips:

```
gen_coin_cm <- function(n, experiment) {
    X_n <- sample(0:1, size = n, replace = TRUE)
```

```
    Xbar_in <- cummean(X_n)
    return(tibble(i = 1:n, X_n = X_n, Xbar_in = Xbar_in, experiment =
experiment))
}
```

Once you define a function, you can use it as follows:

```
gen_coin_cm(20, 3)
```

```
# A tibble: 20 × 4
       i   X_n Xbar_in experiment
   <int> <int>   <dbl>      <dbl>
 1     1     1   1              3
 2     2     1   1              3
 3     3     0   0.667          3
 4     4     1   0.75           3
 5     5     1   0.8            3
 6     6     0   0.667          3
 7     7     1   0.714          3
 8     8     1   0.75           3
 9     9     1   0.778          3
10    10     1   0.8            3
11    11     1   0.818          3
12    12     0   0.75           3
13    13     0   0.692          3
14    14     1   0.714          3
15    15     0   0.667          3
16    16     1   0.688          3
17    17     0   0.647          3
18    18     0   0.611          3
19    19     1   0.632          3
20    20     0   0.6            3
```

First, generate a tibble coin_df with n = 10 and experiment = 1. Second, make a plot of coin_df with i on the horizontal and Xbar_in on the vertical. Third, set n = 1000, generate coin1_df as in part 1 of this question, and redo the plot in part 2. Fourth, generate additional data sets coin2_df and coin3_df as in part 3 of this question. Join them all together using bind_rows. Then, make a line plot with a different colour for each experiment, using colour = factor(experiment) in geom_line. Fifth, make one more plot with one experiment and n = 100000.

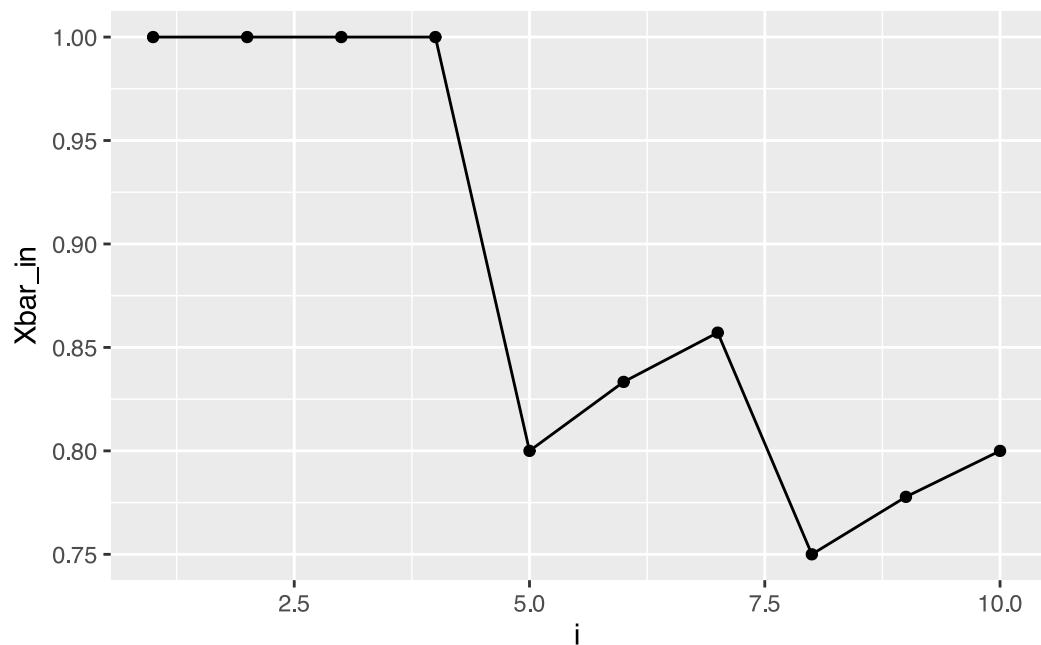Interpret your results. Is the phenomenon you observe related to the LLN or the CLT?

Answer

```
coin_df <- gen_coin_cm(10, 1)
coin1_df <- gen_coin_cm(1000, 1)
```

```
coin2_df <- gen_coin_cm(1000, 2)
coin3_df <- gen_coin_cm(1000, 3)
```
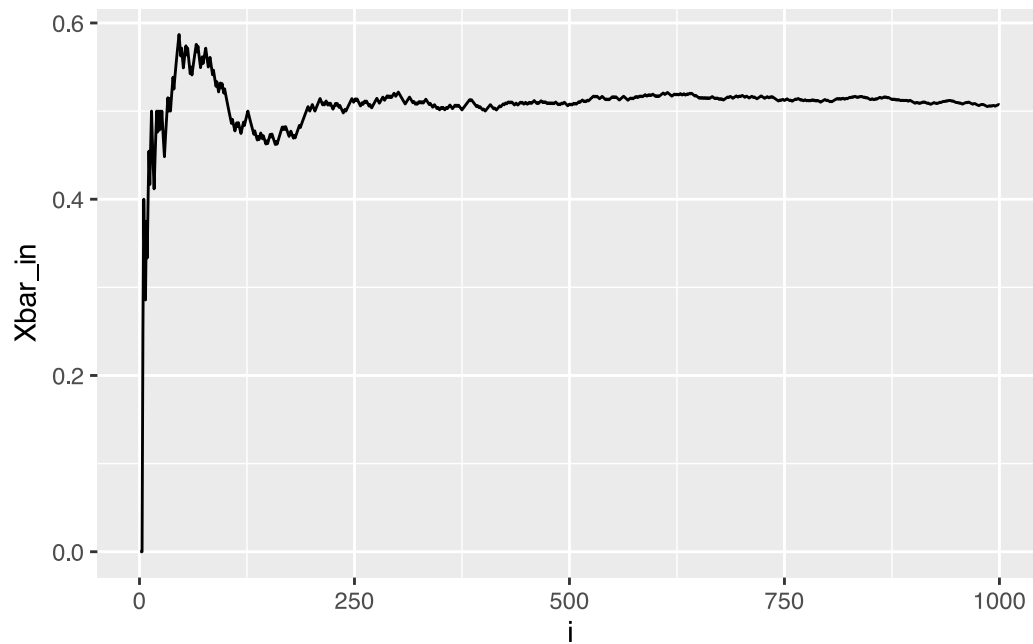
In order to see whether this worked we now make the plot requested in the second part of this question.

```
coin_df |>
    ggplot() +
        geom_point(aes(x=i, y=Xbar_in)) +
        geom_line(aes(x=i, y=Xbar_in))
```


To complete the third part of this question we will duplicate our code from above for the new data.

```
coin1_df |>
    ggplot() +
        geom_line(aes(x=i, y=Xbar_in))
```

Next, we repeated this experiment twice more; three times total.

```
Coin <- bind_rows(coin1_df, coin2_df, coin3_df)
Coin |>
    ggplot() +
        geom_line(aes(x=i, y=Xbar_in, colour=factor(experiment))) +
        geom_line(aes(x=i, y=Xbar_in, colour=factor(experiment))) +
        geom_line(aes(x=i, y=Xbar_in, colour=factor(experiment)))
```
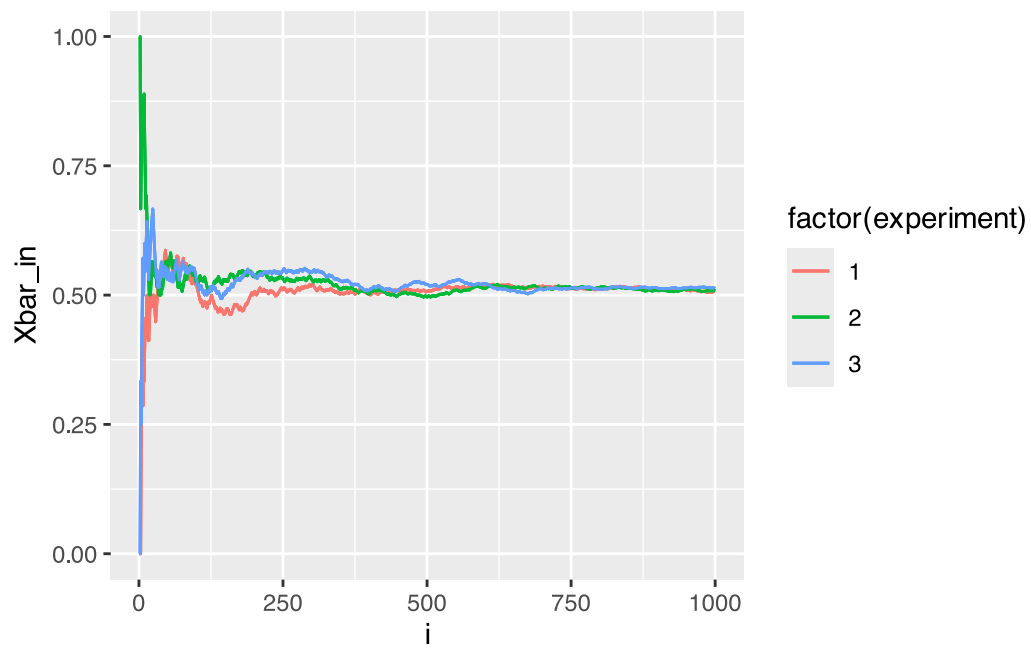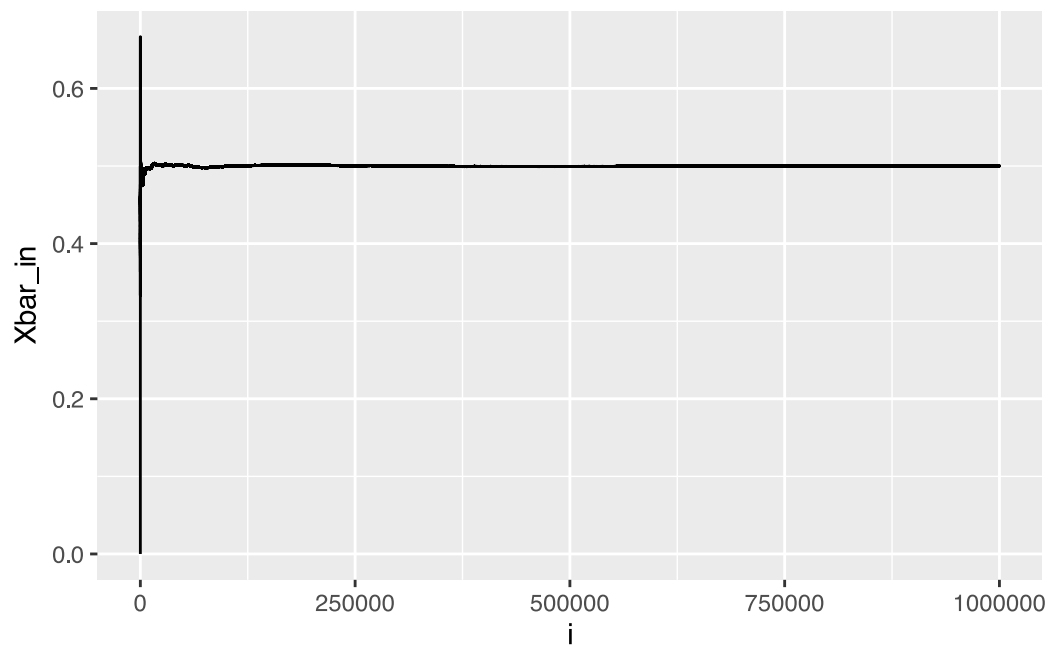
```
bigger_coin <- gen_coin_cm(1000000, 4)

bigger_coin |>
    ggplot() +
        geom_line(aes(x=i, y=Xbar_in))
```

Suprisngly the biggest sample has ~fully converged to the population mean. ...

Monte Carlo 2: dice throws

In the previous exercise, we fixed S and let n grow large. Let us now slowly grow n and see what happens, letting S be very large. We start with n = 1, corresponding to one roll of the die. (For this second exercise, we will use a die instead of a coin.)

One roll:

```
S <- 100
x <- sample(1:6, S, replace = TRUE)
die_df <- tibble(s = 1:S, Xbar = x)
die_df |> ggplot(aes(x = Xbar)) +
    geom_histogram(binwidth = 1)
```
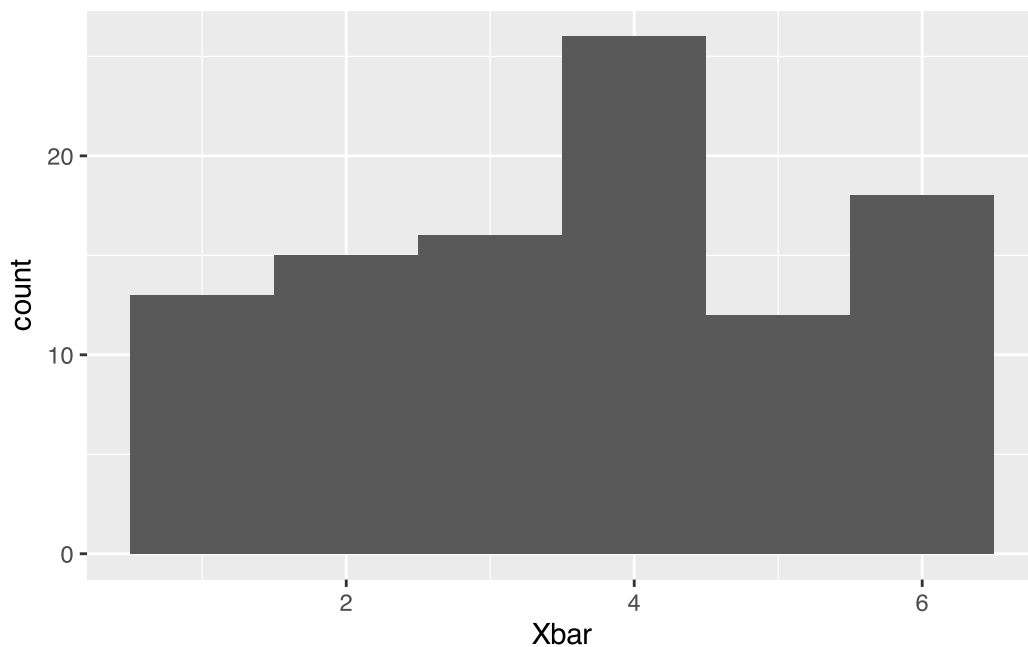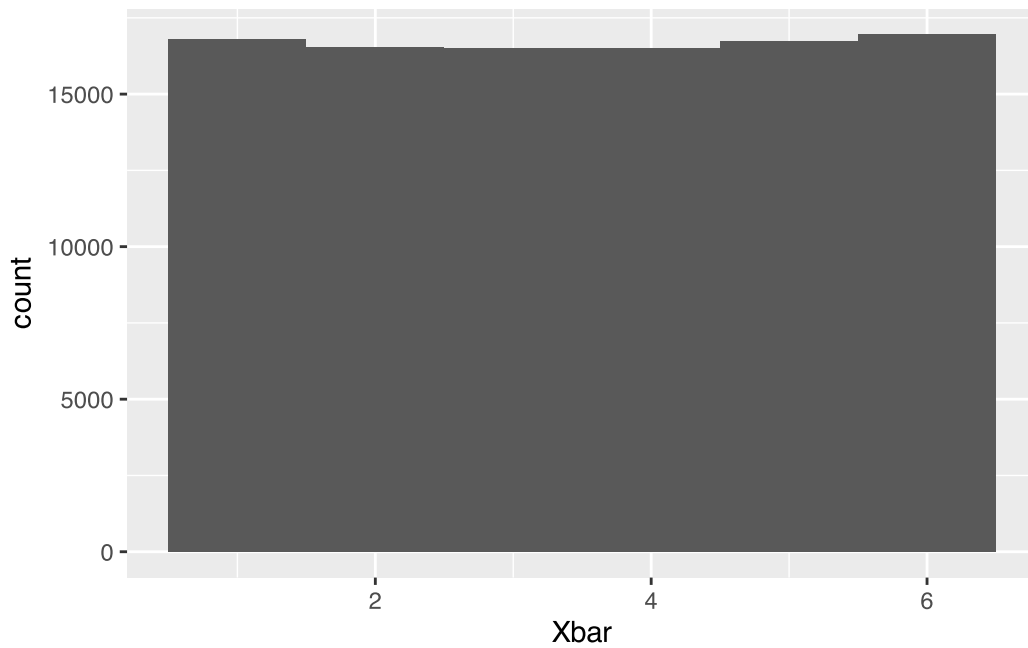

Let us redo this with larger S.

```
S <- 100000
x <- sample(1:6, S, replace = TRUE)
die_df <- tibble(s = 1:S, Xbar = x)
die_df |> ggplot(aes(x = Xbar)) +
    geom_histogram(binwidth = 1)
```

A large S seems to be closer to the population probabilities of $P(X = x) = 1/6$. Because our goal is to explore the theoretical properties of random variables, we will stick with large S.

Roll twice and average.

```
n <- 2
x <- 1 / n * (sample(1:6, S, replace = TRUE) + sample(1:6, S, replace = TRUE))
die_df <- tibble(s = 1:S, Xbar = x)
die_df |> ggplot(aes(x = Xbar)) +
    geom_histogram(binwidth = 1 / n)
```
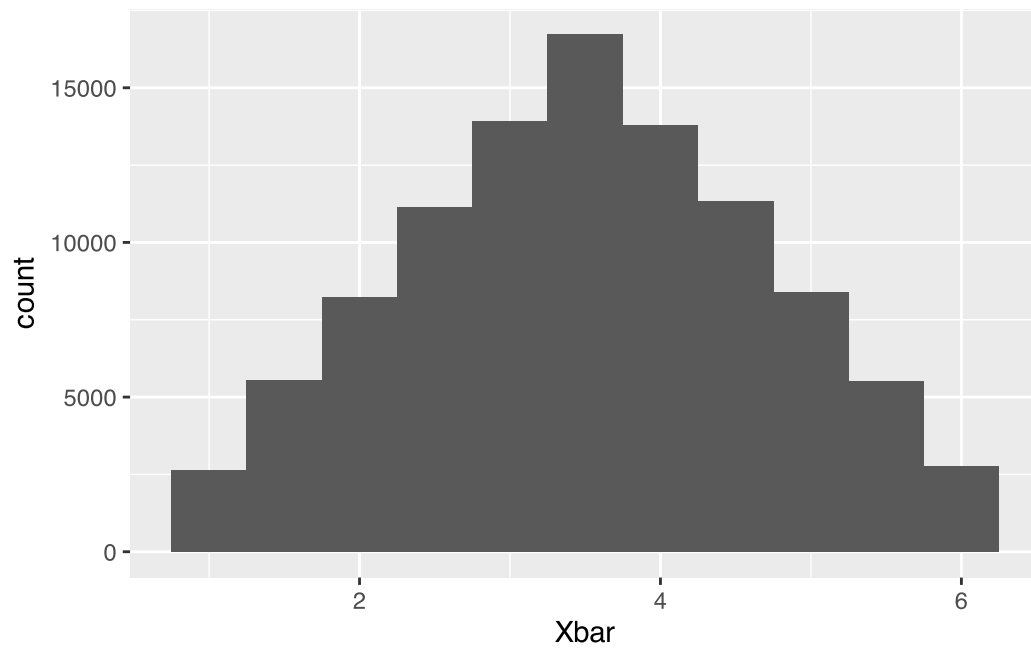
Roll three times and average.

```r
n <- 3
x <- 1 / n * sample(1:6, S, replace = TRUE)
for (k in 2:n) {
    x <- x + 1 / n * sample(1:6, S, replace = TRUE)
}
die_df <- tibble(s = 1:S, Xbar = x)
die_df |> ggplot(aes(x = Xbar)) +
    geom_histogram(binwidth = 1 / n)
```
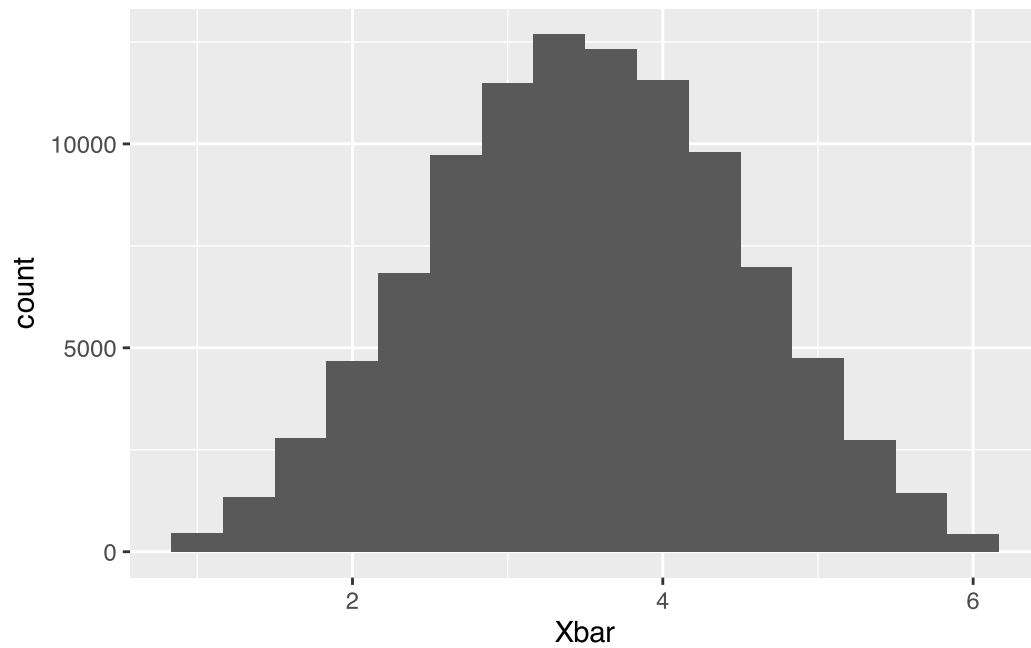
Experiment with a few more values of n (let n = 50 be your largest one.)

First: what happens to the variance of the mean as n increases? Second: what happens to the shape of the distribution as n increases? Relate your answer to the Central Limit Theorem (CLT).

Answer

...

Monte Carlo 3: Inference in RCTs

This question will be solved in class.

We want to understand the properties of different estimators in an RCT. We consider a setup where the treatment effect varies with a covariate X, and treatment assignment is unbalanced.

- $X_i \in \{0, 1\}$ with $P(X_i = 1) = 0.5$
- $D_i \in \{0, 1\}$ with $P(D_i = 1 \mid X_i) = 0.2$
- Outcomes are generated as:

$$Y_i = 0 + 1 \cdot D_i + 5 \cdot X_i + 5 \cdot (D_i \times X_i) + \epsilon_i$$

where $\epsilon_i \sim N\left(0, (1 + 2X_i)^2\right)$.

Calculate the ATE and the CATEs for $X = 0$ and $X = 1$ based on these parameters.

We want to see if the reported standard errors from simple linear regressions are trustworthy. We compare two common approaches:

1. DIM: `lm(Y ~ D)`.
2. Additive: `lm(Y ~ D + X)`.

We run a simulation to check:

1. Bias: Is the average estimate close to the true ATE?
2. Efficiency: What is the standard deviation of the estimates?
3. Coverage Probability: How often does the 95% Confidence Interval contain the true ATE?

Consider the following function that performs one simulation replication and returns a tibble with the results for both estimators:

```r
dgp_het <- function(n) {
    D <- rbinom(n, 1, 0.2) # Unbalanced assignment
    X <- rbinom(n, 1, 0.5)
    e <- rnorm(n, 0, 1 + 2 * X) # Heteroskedastic errors
    Y <- 0 + 1 * D + 5 * X + 5 * D * X + e
    return(tibble(D = D, X = X, Y = Y))
}

sim_one <- function(s, n) {
    df <- dgp_het(n)
    truth <- 3.5

    # 1. DIM
    fit_dim <- lm(Y ~ D, data = df)
    res_dim <- tidy(fit_dim, conf.int = TRUE) |>
        filter(term == "D") |>
        mutate(estimator = "DIM", covered = (conf.low < truth & conf.high >
truth))
```

```
    # 2. Additive
    fit_add <- lm(Y ~ D + X, data = df)
    res_add <- tidy(fit_add, conf.int = TRUE) |>
        filter(term == "D") |>
        mutate(
            estimator = "Additive",
            covered = (conf.low < truth & conf.high > truth)
        )

    bind_rows(res_dim, res_add) |>
        mutate(s = s)
}

sim_one(1, 200)
```

```
# A tibble: 2 × 10
  term  estimate std.error statistic  p.value conf.low conf.high estimator
  <chr>    <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl> <chr>
1 D         2.21     0.691      3.20 1.59e- 3    0.849      3.57 DIM
2 D         2.89     0.417      6.92 6.07e-11    2.06       3.71 Additive
# i 2 more variables: covered <lgl>, s <dbl>
```

Your task is to:

1. Run this simulation $S = 1000$ times with $n = 200$.
2. Summarize the results: calculate the mean estimate, standard deviation of the estimate, and mean coverage probability for each estimator.
3. Interpret the findings related to bias and efficiency of the estimators: which estimator do you prefer?
4. Interpret the findings related to coverage probability: do you trust the confidence intervals from these estimators?

Answer

```
S <- 1000
test_df <- 1:S |> map_df(sim_one, n=200)

test_df |> group_by(estimator) |>
        summarize(
            mean_ATE_hat = mean(estimate),
            sd_ATE_hat = sd(estimate),
            coverage = mean(covered)
        )
```

```
# A tibble: 2 × 4
  estimator mean_ATE_hat sd_ATE_hat coverage
```

```
   <chr>            <dbl>      <dbl>    <dbl>
1 Additive          3.49      0.514    0.91
2 DIM               3.48      0.912    0.861
```

Commentary on the ater part of the questions. ...

Monte Carlo 4: Lin's estimator

Now it is your turn.

Replicate the simulation from Monte Carlo 3 but add a third estimator: Lin's Estimator. Use `lm_lin(Y ~ D, covariates = ~ X)`. You should then be able to extract the estimator information using something like:

```
est_lin <- lm_lin(Y ~ D, covariates = ~X, data = df)
res_lin <- tidy(est_lin) |>
    filter(term == "D") |>
    mutate(estimator = "Lin", covered = (conf.low < truth & conf.high >
truth))
```

Questions:

1. Write a function `sim_lin_ext(s, n)` that returns a tibble with results for DIM, Additive, and Lin.
2. Calculate the mean estimate, standard deviation, and coverage probability for all three.
3. Does Lin's estimator achieve the correct coverage?
4. Which estimator has the lowest variance?

Use $S = 100$ and $n = 1000$.

Answer

...

Monte Carlo 5, Bonus: Selection Bias

Imagine a scenario where the probability of treatment depends on X.

- $X \in \{0, 1\}$ with $P(X = 1) = 0.5$.
- Stratified Assignment:
  - If $X = 0$, $P(D = 1) = 0.2$.
  - If $X = 1$, $P(D = 1) = 0.8$.
- Outcome: $Y = 0 + 1 \cdot D + 5 \cdot X + 2 \cdot D \cdot X + \epsilon$.

So $X = 1$ makes you more likely to be treated AND increases your outcome. Treatment effect is also larger for $X = 1$.

Questions:

1. Write a function `dgp_selection(n)` for this design.

2. Run a simulation ($S = 1000, n = 200$) comparing:

   - DIM: `lm(Y ~ D)`
   - Lin: `lm_lin(Y ~ D, covariates = ~ X)`

3. Show that DIM is biased. Calculate the bias.

4. Show that Lin recovers the true ATE.

5. Why does DIM fail here?

Answer

...

Linear regression for experiments

Job corps, reanalysis

This question will be demonstrated in class.

We will use the same data as in the "Job corps" question in Assignment 1:

```
library(causalweight)
```

```
Loading required package: ranger
```

```
data(JC)
JC <- as_tibble(JC)
```

Here is one way to compute the difference in means of the treatment and control group. compare it to the result on H, p. 21.

```
JC |> filter(assignment == 1) -> JC_short_TG
JC |> filter(assignment == 0) -> JC_short_CG
treat_mean <- mean(JC_short_TG$earny4)
control_mean <- mean(JC_short_CG$earny4)
(ATE_hat <- treat_mean - control_mean)
```

```
[1] 16.05513
```

That is identical to the estimated effect in H3.1.

Questions:

1. Estimate the ATE using `lm_robust` and present the results in a regression table using `tbl_regression`.
2. Run a regression that reveals whether the effect is different for men and women. Use `lm_lin` with the `covariates` argument.

Interpret all your results.

Answer

…

HIV RCT, reanalysis

We will analyze the effect of the HIV information campaign in Thornton (2008).

After you complete the data analysis below, please also answer the following substantive questions:

1. What exactly is the specific policy intervention studied in this paper?
2. Can you think of alternative policy interventions that might achieve similar goals?
3. Based on the results, is this an effective policy? What would you recommend to the health authority in Malawi?
4. Would you recommend this policy to the Canadian health authorities?
5. What is your main criticism of Thornton's study?

We first need to conduct the empirical analysis. Use the same steps as in the previous exercise to analyze the effect of the campaign (`any`). Use `lm_robust` and `tbl_regression`. Explore whether the treatment effect differs by age (heterogeneity), by creating a binary variable for age (e.g. median split or `age >= 33`) and using `lm_lin`.

Remember to use `drop_na` after loading the data to remove the missing data.

Answer

...

Project STAR

When loading libraries, we sometimes want to suppress messages and warnings to keep our document clean. We can do this by setting `#| message: false` and `#| warning: false` in the chunk options.

Load the `AER` library (install first, if necessary), and the `STAR` data that comes with it:

```
library(AER)
data("STAR")
STAR <- as_tibble(STAR) |> drop_na()
```

From `?STAR` (which you should look at, to find the variable descriptions):

> Project STAR (Student/Teacher Achievement Ratio) was a four-year longitudinal class-size study funded by the Tennessee General Assembly and conducted in the late 1980s by the State Department of Education. Over 7,000 students in 79 schools were randomly assigned into one of three interventions: small class (13 to 17 students per teacher), regular class (22 to 25 students per teacher), and regular-with-aide class (22 to 25 students with a full-time teacher's aide). Classroom teachers were also randomly assigned to the classes they would teach. The interventions were initiated as the students entered school in kindergarten and continued through third grade.

We will focus on 3rd grade effects. We summarize the treatment variable using `tbl_summary` from the `gtsummary` package for a nicer look:

```
STAR |>
    select(star3) |>
    tbl_summary()
```

Characteristic	N = 2,161[1]
star3	
regular	548 (25%)
small	858 (40%)
regular+aide	755 (35%)

[1] n (%)

We first define the treatment variable as 0 if class is "regular", 1 if small:

```
STAR |>
    filter(star3 %in% c("regular", "small")) |>
    mutate(D3 = ifelse(star3 == "small", 1, 0)) -> STAR_binary
```

First, compute the treatment effect of D3 on total reading scaled score in 3rd grade. Make sure to use heteroskedasticity-robust standard errors throughout. Interpret the results. Second, compute the effect on total math scaled score in 3rd grade. Interpret the results.

Third, run the regression for math using "multiple treatments", so using star3 as the treatment variable. Make sure to use STAR, not STAR_binary. Do you find the same results for the small treatment? Interpret your results for the regular+aide treatment.

Bonus points: can you find evidence of heterogeneity of the treatment effect?

Answer

…

For troubleshooting: do not edit or remove

```
sysname

"Darwin"

release

"25.2.0"

version
"Darwin Kernel Version 25.2.0: Tue Nov 18 21:09:49 PST 2025;
root:xnu-12377.61.12~1/RELEASE_ARM64_T8142"

nodename

"tadhgs-M5-MacBook-Pro.local"

machine

"arm64"

login

"root"

user

"tadhg"

effective_user

"tadhg"
```

```
[1] "2026-01-19 14:14:00 EST"
```