Lab 3: Design-Based Statistical Inference for Causal Quantities

Dias Akhmetbekov

2024-02-16

Outline

- Inference for SATE and PATE
- Finite sample adjustments

Complete randomization

- Setting: **completely randomized experiment**
  - N units: N_1 treated and N_0 control
  - Random assignment: $D_i \perp (Y_{1i}, Y_{0i})$
- What we observe is $Y_i = D_i Y_{1i} + (1 - D_i) Y_{0i}$
- An intuitive estimator for the SATE:

$$\hat{\tau}_{\text{diff}} = \underbrace{\frac{1}{N_1} \sum_{i=1}^{N} D_i Y_i}_{\text{mean among treated}} - \underbrace{\frac{1}{N_0} \sum_{i=1}^{N} (1 - D_i) Y_i}_{\text{mean among control}}$$

- Conditional on the sample $\mathcal{S}$, $\hat{\tau}_{\text{diff}}$ only varies because of D_i

Complete randomization

- First, take expectations over repeated treatment assignments holding the sample $\mathcal{S}$ fixed:

$$\begin{aligned}
E_D[\hat{\tau}_{diff} \mid \mathcal{S}] &= \frac{1}{N_1}\sum_{i=1}^{N}[E_D[D_i Y_i \mid \mathcal{S}]] - \frac{1}{N_0}\sum_{i=1}^{N}[E_D[(1-D_i)Y_i \mid \mathcal{S}]] \\
&= \frac{1}{N_1}\sum_{i=1}^{N}[E_D[D_i Y_i(1) \mid \mathcal{S}]] - \frac{1}{N_0}\sum_{i=1}^{N}[E_D[(1-D_i)Y_i(0) \mid \mathcal{S}]] \\
&= \frac{1}{N_1}\sum_{i=1}^{N}[E_D[D_i \mid \mathcal{S}]Y_i(1)] - \frac{1}{N_0}\sum_{i=1}^{N}[E_D[(1-D_i) \mid \mathcal{S}]Y_i(0)] \\
&= \frac{1}{N_1}\sum_{i=1}^{N}\left(\frac{N_1}{N}\right)Y_i(1) - \frac{1}{N_0}\sum_{i=1}^{N}\left(\frac{N_0}{N}\right)Y_i(0) \\
&= \frac{1}{N}\sum_{i=1}^{N}[Y_i(1) - Y_i(0)] = SATE
\end{aligned}$$

- The unbiasedness results does NOT depend on N_0, N_1

Complete randomization

- Then take expectations over different samples:

$$\begin{aligned} E\left[E_D\left[\hat{\tau}_{diff} \mid \mathcal{S}\right]\right] &= E\left[SATE\right] \\ = \frac{1}{N}\sum_{i\in\mathcal{S}} E\left[Y_{1i} - Y_{0i}\right] &= \frac{1}{N} N E\left[Y_{1i} - Y_{0i}\right] \\ &= E\left[Y_{1i} - Y_{0i}\right] = PATE \end{aligned}$$

- $\hat{\tau}_{diff}$ is unbiased for PATE (under random sampling and random treatment assignment)

Finite-sample sampling variance

- Conditional variance for complete random assignment:

$$\begin{aligned}
\text{Var}_D[\hat{\tau}_{diff} \mid \mathcal{S}] &= \text{Var}_D[\bar{Y}_1 \mid \mathcal{S}] + \text{Var}_D[\bar{Y}_0 \mid \mathcal{S}] - 2\text{Cov}_D[\bar{Y}_1, \bar{Y}_0 \mid \mathcal{S}] \\
&= \frac{s^2_{Y_1}}{N_1}\left(\frac{N - N_1}{N}\right) + \frac{s^2_{Y_0}}{N_0}\left(\frac{N - N_0}{N}\right) - 2\left[-\frac{s_{Y_1,Y_0}}{N}\right] \\
&= \frac{s^2_{Y_1}}{N_1} + \frac{s^2_{Y_0}}{N_0} - \frac{s^2_{Y_1} + s^2_{Y_0} - 2s_{Y_1,Y_0}}{N} \\
&= \frac{s^2_{Y_1}}{N_1} + \frac{s^2_{Y_0}}{N_0} - \frac{s^2_{\tau}}{N},
\end{aligned}$$

- Last term is the in-sample variation of the individual treatment effects (not observable!)

Inferences from sampling variance

- Usual variance estimator is the Neyman (or robust) estimator:

$$\hat{V} = \frac{\hat{s}_0^2}{N_0} + \frac{\hat{s}_1^2}{N_1}$$

- $\hat{V}$ is biased for $\text{Var}_D[\hat{\tau}_{diff} \mid \mathcal{S}]$
- It leads to conservative inferences:
  - standard errors $\sqrt{\hat{V}}$ will be at least as big as they should be;
  - CIs using $\sqrt{\hat{V}}$ will be at least wide as they should be;
  - Type I error rates will still be correct, power will be lower;
  - Both will be exactly right if treatment effects are constant.

Population variance

- From ANOVA theorem and CEF decomposition: $\mathrm{Var}[A] = E[\mathrm{Var}[A \mid B]] + \mathrm{Var}[E[A \mid B]]$. Hence,

$$\begin{aligned}
\mathrm{Var}[\hat{\tau}] &= E[\mathrm{Var}_D[\hat{\tau} \mid \mathcal{S}]] + \mathrm{Var}[E_D[\hat{\tau} \mid \mathcal{S}]] \\
&= E\left[\frac{s^2_{Y_1}}{N_1} + \frac{s^2_{Y_0}}{N_0} - \frac{s^2_{\tau}}{N}\right] + \mathrm{Var}[SATE] \\
&= E\left[\frac{s^2_{Y_1}}{N_1} + \frac{s^2_{Y_0}}{N_0} - \frac{s^2_{\tau}}{N}\right] + \mathrm{Var}\left[\frac{1}{N}\sum_{i\in\mathcal{S}} \tau\right] \\
&= E\left[\frac{s^2_{Y_1}}{N_1} + \frac{s^2_{Y_0}}{N_0} - \frac{s^2_{\tau}}{N}\right] + \frac{\sigma^2_{\tau}}{N} \\
&= \frac{\sigma^2_{Y_1}}{N_1} + \frac{\sigma^2_{Y_0}}{N_0} - \frac{\sigma^2_{\tau}}{N} + \frac{\sigma^2_{\tau}}{N} \\
&= \frac{\sigma^2_{Y_1}}{N_1} + \frac{\sigma^2_{Y_0}}{N_0}.
\end{aligned}$$

Population variance

- The Neyman estimator $\hat{V}$ is now unbiased for $\hat{V}_{\text{diff}}$;
- Consistency via the law of large numbers: $\hat{\tau}_{\text{diff}} \xrightarrow{p} \text{PATE}$
- Asymptotic normality via the Central Limit Theorem (CLT):

$$\frac{\hat{\tau}_{\text{diff}} - \text{PATE}}{\sqrt{\frac{\sigma_1^2}{N_1} + \frac{\sigma_0^2}{N_0}}} \xrightarrow{d} \mathcal{N}(0,1)$$

- Two interpretations for $\hat{V}$:
  - Unbiased estimator for sampling variance of the traditional estimator of the PATE;
  - Conservative estimator for the sampling variance of the traditional estimator of the SATE

Simulation

```r
set.seed(123) # Set seed

# Function to generate population data
generate_population <- function(N_pop) {
  Y0 <- abs(rnorm(N_pop, mean = 5, sd = 2))
  Y1 <- Y0 + rnorm(N_pop, 0, 5) + 4
  TE <- Y1 - Y0
  data.frame(Y0 = Y0, Y1 = Y1, TE = TE)
}
```

Simulation

```r
# Function to simulate the randomization process
simulate_randomization <- function(sample, Nboot) {
  Nsample <- nrow(sample)
  dim <- vars <- rep(NA, Nboot)
  for(i in 1:Nboot){
    D <- sample(c(0,1), Nsample, replace = TRUE)
    Y <- D*sample$Y1 + (1-D)*sample$Y0
    dim[i] <- mean(Y[D==1]) - mean(Y[D==0])
    vars[i] <- var(Y[D==1])/(Nsample/2) + var(Y[D==0])/(Nsample/2)
  }
  list(dim = dim, vars = vars)
}
```

Simulation

```r
# Generate population
N_pop <- 100000
pop <- generate_population(N_pop)
(PATE <- mean(pop$TE))
```

```
## [1] 4.026077
```

```r
# Single sample simulation
Nsample <- 1000
sample <- pop[sample(N_pop, Nsample),]
(SATE <- mean(sample$TE))
```

```
## [1] 3.910847
```

Simulation

```r
# Randomization distribution simulation
Nboot <- 1000
randomization_results <- simulate_randomization(sample, Nboot)
(mean_dim <- mean(randomization_results$dim))
```

```
## [1] 3.919121
```

```r
(var_dim <- var(randomization_results$dim))
```

```
## [1] 0.03871511
```

```r
(expected_vars <- mean(randomization_results$vars))
```

```
## [1] 0.06289298
```

Simulation

```
# Two-level simulation: sampling and randomization
Nsampling <- 100
dim_sampling <- matrix(NA, Nsampling, Nboot)
vars_sampling <- matrix(NA, Nsampling, Nboot)

for(j in 1:Nsampling){
  sample <- pop[sample(N_pop, Nsample),]
  results <- simulate_randomization(sample, Nboot)
  dim_sampling[j,] <- results$dim
  vars_sampling[j,] <- results$vars
}
```

Simulation

```r
(mean_dim_sampling <- mean(dim_sampling))
```

```
## [1] 4.006696
```

```r
(var_dim_sampling <- var(as.vector(dim_sampling)))
```

```
## [1] 0.06385131
```

```r
(expected_vars_sampling <- mean(vars_sampling))
```

```
## [1] 0.06597705
```

```r
# Theoretical variance
(theoretical_var <-var(pop$Y1)/(Nsample/2) + var(pop$Y0)/(Nsample/2))
```

```
## [1] 0.06595247
```

Finite samples

- Suppose the goal is robust inference for $\hat{\tau} = \bar{Y}_1 - \bar{Y}_0$ in a small sample size (30 units, for example)
- Suppose N_1 is very large but N_0 is very small
- Then, $\bar{Y}_1$ will be very precisely estimated, but $\bar{Y}_0$ will be imprecisely estimated
- That being the case, using $N - k = (N_1 + N_0) - k$ as the degrees of freedom adjustment overstates stability
- Correct degrees of freedom adjustment ought to be closer to $N_0 - k$

Simulation (Imbens and Kolesar, 2016)

```r
set.seed(123)
library(sandwich)
library(lmtest)
library(MASS)
library(ggplot2)
```

```r
# Simulation parameters
N <- 30
N1 <- 3
N0 <- N - N1
beta <- c(0, 0)
sigma_1 <- 1
sigma_0_values <- c(0.5, 0.85, 1, 1.18, 2)
num_replications <- 10000
true_effect <- beta[2]
```

Simulation (Imbens and Kolesar, 2016)

```r
# Function to simulate data
simulate_data <- function(N, N1, sigma_0, sigma_1) {
  D <- c(rep(1, N1), rep(0, N - N1))
  Y <- rep(NA, N)
  epsilon <- rnorm(N, mean = 0, sd = ifelse(D == 1, sigma_1, sigma_0))
  Y <- beta[1] + beta[2] * D + epsilon
  data.frame(Y = Y, D = D)
}

# Function to calculate and evaluate CI coverage
calculate_coverage <- function(sigma_0, estimator_function) {
  coverage_count <- 0
  for (i in 1:num_replications) {
    data <- simulate_data(N, N1, sigma_0, sigma_1)
    model <- lm(Y ~ D, data = data)
    ci <- estimator_function(model)
    if (true_effect >= ci[1] && true_effect <= ci[2]) {
      coverage_count <- coverage_count + 1
    }
  }
  coverage_probability <- coverage_count/num_replications
  return(coverage_probability)
}
```

Simulation (Imbens and Kolesar, 2016)

```r
# Estimator functions
homoskedastic_estimator <- function(model) {
  ci <- confint(model, level = 0.95)["D",]
  return(ci)
}

heteroskedastic_estimator <- function(model) {
  se <- sqrt(diag(vcovHC(model, type = "HC1")))
  coef <- coef(model)["D"]
  ci <- coef + qt(c(0.025, 0.975), df = N - 2) * se[2]
  return(ci)
}
```

Simulation (Imbens and Kolesar, 2016)

```r
hc2_estimator <- function(model) {
  se <- sqrt(diag(vcovHC(model, type = "HC2")))
  coef <- coef(model)["D"]
  ci <- coef + qt(c(0.025, 0.975), df = N - 2) * se[2]
  return(ci)
}

bell_mccaffrey_estimator <-function(model) {
  se <- sqrt(diag(vcovHC(model, type = "HC2")))
  coef <- coef(model)["D"]
  ci <- coef + qt(c(0.025, 0.975), df = N1 - 1) * se[2]
  return(ci)
}
```

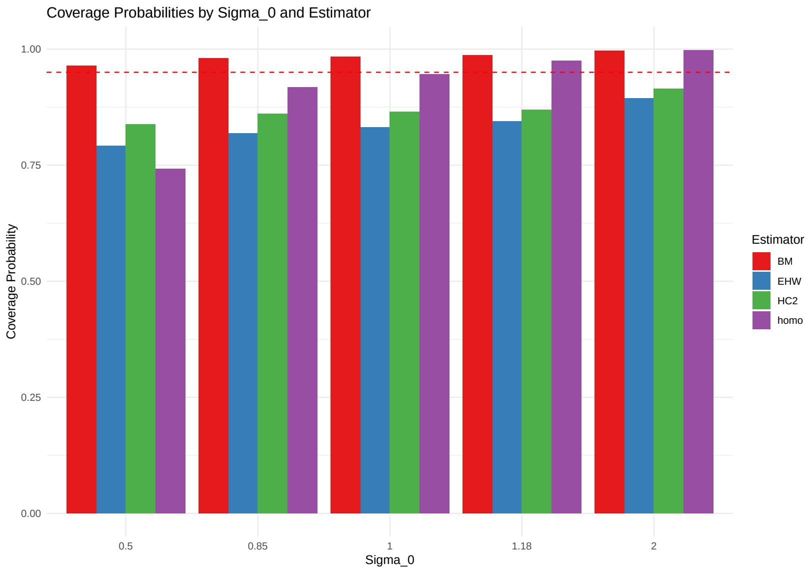
Simulation (Imbens and Kolesar, 2016)

```r
# Run simulations
coverage_results <- list()
for (sigma_0 in sigma_0_values) {
  coverage_homoskedastic <- calculate_coverage(sigma_0, homoskedastic_estimator)
  coverage_heteroskedastic <- calculate_coverage(sigma_0, heteroskedastic_estimator)
  coverage_HC2 <- calculate_coverage(sigma_0, hc2_estimator)
  coverage_BM <- calculate_coverage(sigma_0, bell_mccaffrey_estimator)
  coverage_results[[as.character(sigma_0)]] <- list(
    homo = coverage_homoskedastic,
    EHW = coverage_heteroskedastic,
    HC2 = coverage_HC2,
    BM = coverage_BM
  )
}
```

Simulation (Imbens and Kolesar, 2016)

```r
# Initialize an empty data frame
coverage_df <- data.frame(Sigma_0 = character(),
                          Estimator = character(),
                          Coverage = numeric(),
                          stringsAsFactors = FALSE)

# Fill the data frame with coverage results
for (sigma_0 in names(coverage_results)) {
  for (estimator in names(coverage_results[[sigma_0]])) {
    new_row <- data.frame(Sigma_0 = sigma_0,
                          Estimator = estimator,
                          Coverage = coverage_results[[sigma_0]][[estimator]],
                          stringsAsFactors = FALSE)
    coverage_df <- rbind(coverage_df, new_row)
  }
}
```


Coverage Probabilities by Sigma_0 and Estimator
Coverage Probability
1.00
0.75
0.50
0.25
0.00
0.5
0.85
1
1.18
2
Sigma_0
Estimator
BM
EHW
HC2
homo