HW1_implementation

2025-01-25

```
knitr::opts_chunk$set(echo = TRUE)
library(tidyverse)
```

```
## Warning: package 'lubridate' was built under R version 4.4.2
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(here)
```

```
## here() starts at C:/Users/daisy/OneDrive - Emory/Desktop/Spring 2025/BIOS 731/Homeworks/HW1
```

```
library(tidyr)
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 4.4.2
```

```
library(dplyr)

#source(here("source", "utils.R"))
#source(here("simulations", "run_sim_wald.R"))

set.seed(50)
```

1.1: ADEMP Structure

Answer the following questions:

- How many simulation scenarios will you be running? We will be running 3 * 3 * 2 = 18 simulation scenarios
- What are the estimand(s) The estimands are the average treatment effect $\beta_{treatment}$ and the standard error of treatment effect $se(\beta_{treatment})$

- What method(s) are being evaluated/compared?

3 methods are being compared - Wald confidence intervals, nonparametric bootstrap percentile intervals, and nonparametric boostrap t-intervals.

- What are the performance measure(s)? The performance measures are bias, coverage, and computation time.

1.2: nSim

Based on desired coverage of 95% with Monte Carlo error of no more than 1%, how many simulations (n_{sim}) should we perform for each simulation scenario? Implement this number of simulations throughout your simulation study.

```
mc_err <- 0.01
cover <- 0.95
alpha <- 1 - 0.95
n_sim <- (cover * (1 - cover))/mc_err^2
n_sim
```

```
## [1] 475
```

```
num_scenarios <- 18
```

1.4 summary table

bias: once per scenario (so 18 biases) variance: once per scenario (so 18 var hat beta hats) - can do for wald i think? as beta hats r not means like boot coverage: average per scenario (so 18 coverage percentages) computation time: average per scenario (so 18 computation time rows, 3 cols for wald/boot p/boot t)

bootstrap se_hat_beta_hat = once after all bootstrap runs (i.e. nboot 50) so 475 rows ??? - think i have this already

can plot histogram of SEs for bootstrap? (facet by error type and sample size n)

```
all_wald_results <- vector("list", num_scenarios)
all_boot_p_results <- vector("list", num_scenarios)
all_boot_t_results <- vector("list", num_scenarios)



# Loop through each scenario file
for (i in 1:num_scenarios) {
  file_path <- here("results", "sim_wald", paste0("scenario_", i, ".RDA"))
  load(file_path)

  all_wald_results[[i]] <- all_wald_estim

  file_path <- here("results", "sim_boot_percentile", paste0("scenario_", i, ".RDA"))
  load(file_path)
```

```
  all_boot_p_results[[i]] <- all_boot_percent_estim

  file_path <- here("results", "sim_boot_t", paste0("scenario_", i, ".RDA"))
  load(file_path)

  all_boot_t_results[[i]] <- all_boot_t_estim

}
```

```
all_biases <- rep(NA, num_scenarios)
var_hat <- rep(NA, num_scenarios)
wald_coverage <- rep(NA, num_scenarios)
boot_p_coverage <- rep(NA, num_scenarios)
boot_t_coverage <- rep(NA, num_scenarios)
wald_time <- rep(NA, num_scenarios)
boot_p_time <- rep(NA, num_scenarios)
boot_t_time <- rep(NA, num_scenarios)
all_n <- rep(NA, num_scenarios)
all_beta_true <- rep(NA, num_scenarios)
all_err_type <- rep(NA, num_scenarios)
scenario_num <- rep(NA, num_scenarios)
mean_wald_se_beta <- rep(NA, num_scenarios)
mean_boot_p_se_beta <- rep(NA, num_scenarios)
mean_boot_t_se_beta <- rep(NA, num_scenarios)
se_hat <- rep(NA, num_scenarios)
se_hat_se <- rep(NA, num_scenarios)

for (i in 1:num_scenarios) {

  all_biases[i] <- (1/n_sim) * sum(all_wald_results[[i]]$beta_hat - all_wald_results[[i]]$beta_true)
  var_hat[i] <- sd(all_wald_results[[i]]$beta_hat)
  se_hat[i] <- mean(all_wald_results[[i]]$se_beta)
  se_hat_se[i] <- sd(all_wald_results[[i]]$se_beta)
  wald_coverage[i] <- mean(all_wald_results[[i]]$coverage == 1) * 100
  boot_p_coverage[i] <- mean(all_boot_p_results[[i]]$coverage == 1) * 100
  boot_t_coverage[i] <- mean(all_boot_t_results[[i]]$coverage == 1) * 100
  wald_time[i] <- mean(all_wald_results[[i]]$time)
  boot_p_time[i] <- mean(all_boot_p_results[[i]]$time)
  boot_t_time[i] <- mean(all_boot_t_results[[i]]$time)
  all_n[i] <- unique(all_wald_results[[i]]$n)[[1]]
  all_beta_true[i] <- unique(all_wald_results[[i]]$beta_true)
  all_err_type[i] <- unique(all_wald_results[[i]]$err_type)
  mean_wald_se_beta[i] <- mean(all_wald_results[[i]]$se_beta)
  mean_boot_p_se_beta[i] <- mean(all_boot_p_results[[i]]$se_beta)
  mean_boot_t_se_beta[i] <- mean(all_boot_t_results[[i]]$se_beta)


  scenario_num[i] <- i
}

df <- bind_cols(
  scenario_num = scenario_num,
  n = all_n,
```

```
  beta_true = all_beta_true,
  error_type = all_err_type,
  bias = all_biases,
  var = var_hat,
  se_hat = se_hat,
  se_hat_se = se_hat_se,
  wald_coverage = wald_coverage,
  wald_time = wald_time,
  wald_se = mean_wald_se_beta,
  boot_p_coverage = boot_p_coverage,
  boot_p_time = boot_p_time,
  boot_p_se = mean_boot_p_se_beta,
  boot_t_coverage = boot_t_coverage,
  boot_t_time = boot_t_time,
  boot_t_se = mean_boot_t_se_beta
  )


df <- df %>%
  mutate(
    error_type = case_when(
      error_type == 0 ~ "Lognormal",
      error_type == 1 ~ "Normal",
      TRUE ~ as.character(error_type)
    )
  )
```

```
bias_table <- df %>%
  select(n, beta_true, error_type, bias) %>%
  pivot_wider(
    names_from = error_type,
    values_from = bias,
  ) %>%
  arrange(n, beta_true)

kable(bias_table, digits = 3, caption = "Bias Summary Table")
```

Table 1: Bias Summary Table

n	beta_true	Lognormal	Normal
10	0.0	-0.051	-0.093
10	0.5	-0.051	-0.093
10	2.0	-0.051	-0.093
50	0.0	-0.024	-0.006
50	0.5	-0.024	-0.006
50	2.0	-0.024	-0.006
500	0.0	-0.009	-0.009
500	0.5	-0.009	-0.009
500	2.0	-0.009	-0.009

```r
coverage_table <- df %>%
  select(n, beta_true, error_type, wald_coverage, boot_p_coverage, boot_t_coverage) %>%
  arrange(n, beta_true, error_type) %>%
  pivot_longer(
    cols = c(wald_coverage, boot_p_coverage, boot_t_coverage),
    names_to = "Method",
    names_pattern = "(.*)_coverage", # Extract the method name (wald, boot_p, boot_t)
    values_to = "Coverage"
  ) %>%
  pivot_wider(
    names_from = Method, # Put wald, boot_p, boot_t as column headers
    values_from = Coverage
  )

kable(coverage_table, digits = 3, caption = "Coverage Summary Table")
```

Table 2: Coverage Summary Table

n	beta_true	error_type	wald	boot_p	boot_t
10	0.0	Lognormal	95.368	82.737	53.263
10	0.0	Normal	94.105	83.368	54.947
10	0.5	Lognormal	95.368	82.737	53.263
10	0.5	Normal	94.105	83.368	54.947
10	2.0	Lognormal	95.368	82.737	53.263
10	2.0	Normal	94.105	83.368	54.947
50	0.0	Lognormal	95.368	89.263	88.842
50	0.0	Normal	94.316	90.316	91.579
50	0.5	Lognormal	95.368	89.263	88.842
50	0.5	Normal	94.316	90.316	91.579
50	2.0	Lognormal	95.368	89.263	88.842
50	2.0	Normal	94.316	90.316	91.579
500	0.0	Lognormal	93.474	90.737	92.000
500	0.0	Normal	92.842	89.895	92.211
500	0.5	Lognormal	93.474	90.737	92.000
500	0.5	Normal	92.842	89.895	92.211
500	2.0	Lognormal	93.474	90.737	92.000
500	2.0	Normal	92.842	89.895	92.211

```r
time_table <- df %>%
  select(n, beta_true, error_type, wald_time, boot_p_time, boot_t_time) %>%
  arrange(n, beta_true, error_type) %>%
  pivot_longer(
    cols = c(wald_time, boot_p_time, boot_t_time),
    names_to = "Method",
    names_pattern = "(.*)_time",
    values_to = "Time"
  ) %>%
  pivot_wider(
    names_from = Method, # Put wald, boot_p, boot_t as column headers
    values_from = Time
  )
```

```r
kable(time_table, digits = 3, caption = "CI Time Summary Table")
```

Table 3: CI Time Summary Table

n	beta_true	error_type	wald	boot_p	boot_t
10	0.0	Lognormal	0.030	1.669	18.328
10	0.0	Normal	0.025	1.440	15.947
10	0.5	Lognormal	0.024	1.419	15.662
10	0.5	Normal	0.025	1.444	15.941
10	2.0	Lognormal	0.025	1.453	16.050
10	2.0	Normal	0.025	1.437	15.828
50	0.0	Lognormal	0.024	1.415	15.462
50	0.0	Normal	0.025	1.437	15.723
50	0.5	Lognormal	0.024	1.420	15.550
50	0.5	Normal	0.025	1.432	15.639
50	2.0	Lognormal	0.025	1.441	15.814
50	2.0	Normal	0.025	1.436	15.709
500	0.0	Lognormal	0.024	1.423	15.645
500	0.0	Normal	0.025	1.458	15.944
500	0.5	Lognormal	0.025	1.446	15.876
500	0.5	Normal	0.026	1.447	15.886
500	2.0	Lognormal	0.025	1.467	16.014
500	2.0	Normal	0.024	1.460	16.005

power?