# BIOS 731 HW2

Baijia Xu

2026-02-10

Here is the URL for my Github repo: https://github.com/BXU69/bios731_hw2_xu.

## Problem 1.1 ADEMP Structure

- **A (Aim):**

The goal is to evaluate and compare three approaches for constructing a 95% confidence interval for the treatment effect in linear regression: (i) Wald intervals, (ii) nonparametric bootstrap percentile intervals, and (iii) nonparametric bootstrap t intervals, under varying sample size, true treatment effect, and error distribution.

- **D (Data-generating mechanism):**

Data are generated from the multiple linear regression model

$$Y_i = \beta_0 + \beta_{treatment} X_{i1} + \mathbf{Z_i}^T \boldsymbol{\gamma} + \epsilon_i$$

with primary focus on $\beta_{treatment}$. It is acceptable to simulate without confounders (i.e., $\gamma = 0$). We generate $X_{i1}$ as a binary treatment indicator (e.g., $X_{i1} \sim$ Bernoulli(0.5)).

We vary three design factors in a full factorial design: (1). $n \in \{10, 50, 500\}$

(2). $\beta_{treatment} \in \{0, 0.5, 2\}$

(3). error distribution: (i). normal errors: $\epsilon_i \sim N(0, 2)$; (ii). heavy-tailed errors: $\epsilon_i \sim t_\nu$ with $\nu = 3$, scaled to have variance 2.

- **E (Estimand):**

The estimand is $\beta_{treatment}$, the average treatment effect.

- **M (Methods):**

(1). Wald confidence intervals (standard model-based approach), (2). Nonparametric bootstrap **percentile** intervals, (3). Nonparametric bootstrap $t$ intervals.

- **P (Performance measures):**

Across repeated simulations, we summarize:

(1). Bias of $\hat{\beta}_{treatment}$,

(2). Coverage of each 95% CI method,

(3). Distribution of $\hat{se}(\hat{\beta})$,

(4). Computation time for each CI method.

18 simulation scenarios will be run ($3 \times \times 3 \times 2$).

### Problem 1.2 nSim

$$\sqrt{\frac{p(1-p)}{n_{sim}}} \leq 0.01 \Rightarrow n_{sim} \geq \frac{0.95 \times 0.05}{0.01^2} = 475$$

Therefore we use 475 simulations per scenario.

### Problem 1.3 Implementation

### Problem 1.4 Summarize

```r
library(tidyverse)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
## -- 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   3.5.2     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## 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 error
```

```r
# 1) Load all replicate files
all_files <- list.files(
  path = "data/reps",
  pattern = "\\.Rds$",
  recursive = TRUE,
  full.names = TRUE
)

stopifnot(length(all_files) > 0)

raw <- map_dfr(all_files, readRDS)

# (optional) sanity checks
# Check you have 18 scenarios
raw %>% distinct(n, beta_treatment, error_type) %>% nrow()
```

```
## [1] 18
```

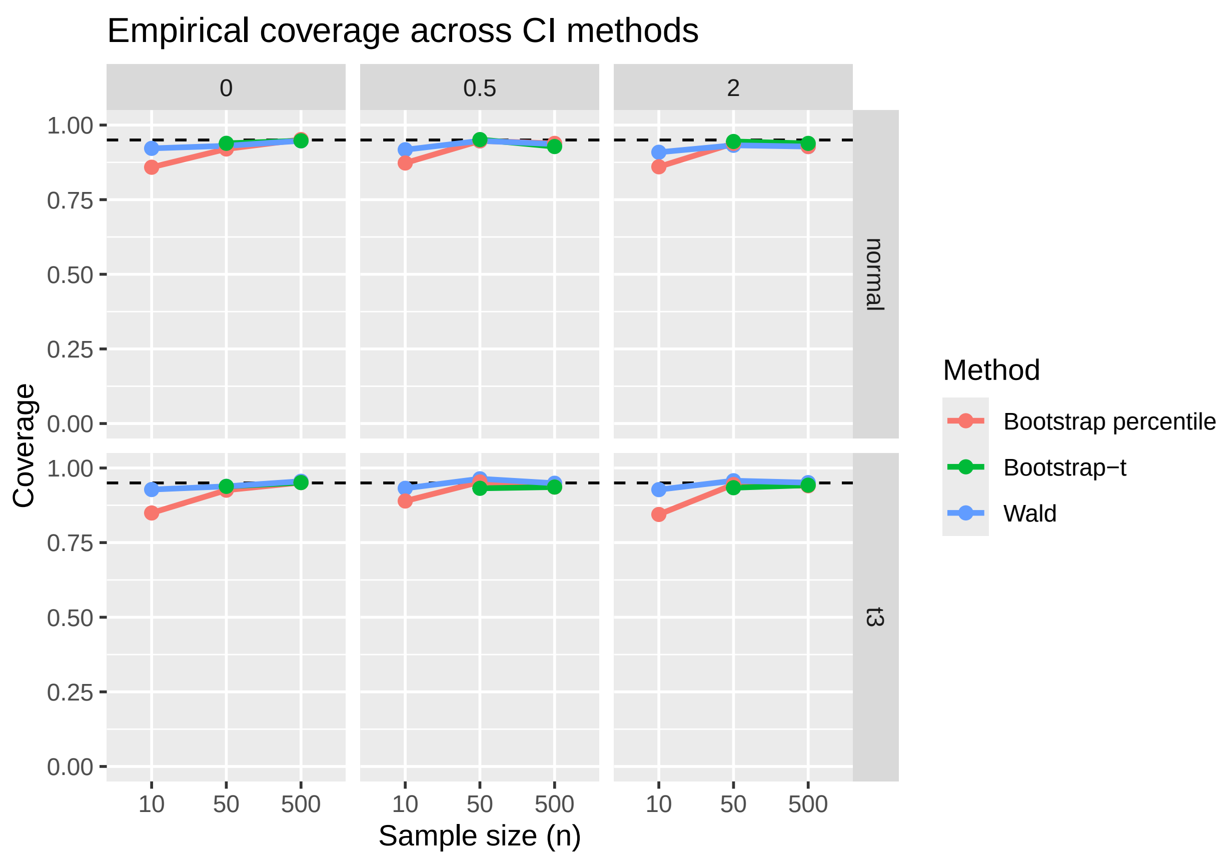
```r
# Check counts per scenario (should be 475 if fully complete)
counts <- raw %>%
  count(n, beta_treatment, error_type, name = "n_reps") %>%
  arrange(n, beta_treatment, error_type)

#print(counts)
```

```r
# 2) Scenario-level summary table
summary_tbl <- raw %>%
  group_by(n, beta_treatment, error_type) %>%
  summarise(
    n_reps = n(),
    bias_mean = mean(bias),
    bias_sd   = sd(bias),

    cover_wald = mean(cover_wald),
    cover_perc = mean(cover_perc),
    cover_t    = mean(cover_t),

    se_mean = mean(se_val),
    se_sd   = sd(se_val),

    time_wald_mean   = mean(time_wald),
    time_perc_mean   = mean(time_perc),
    time_boot_t_mean = mean(time_boot_t),
    .groups = "drop"
  ) %>%
  arrange(error_type, n, beta_treatment)
summary_tbl
```

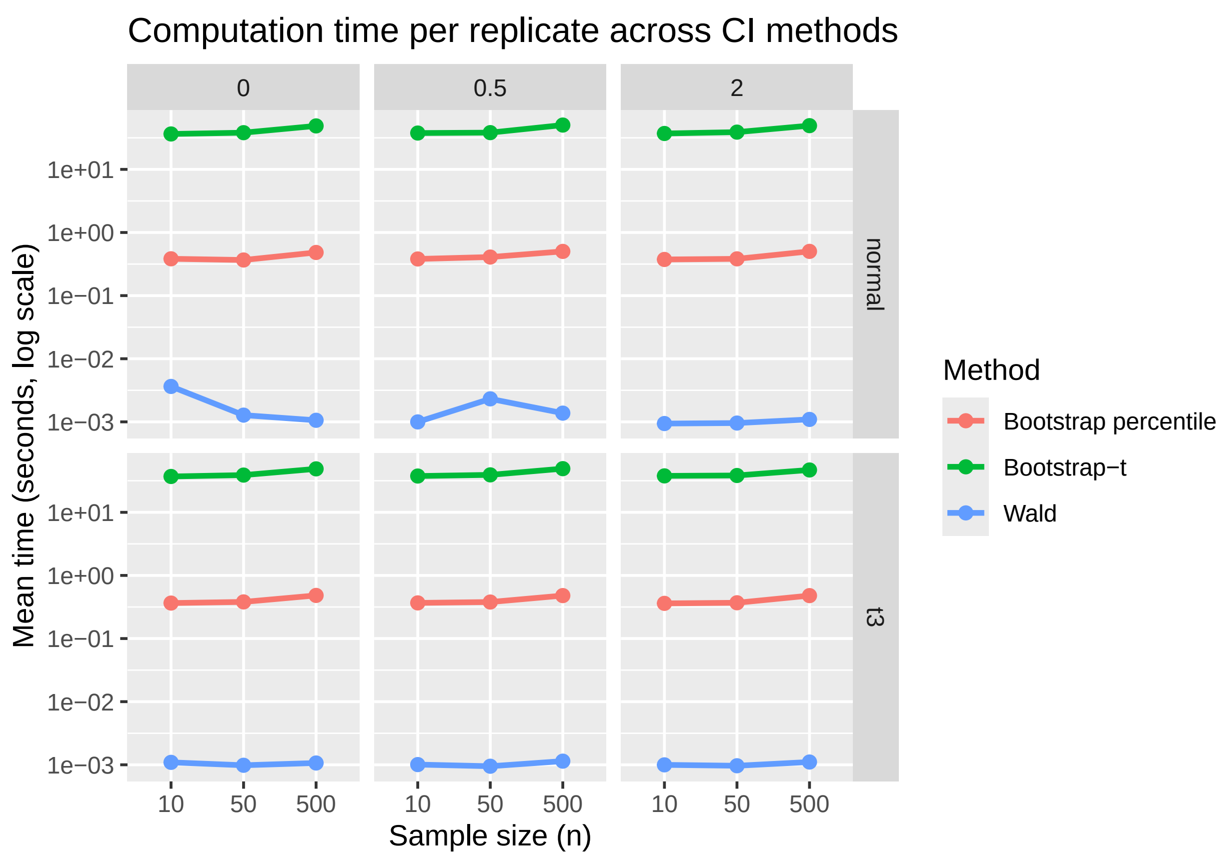
```
## # A tibble: 18 x 14
##        n beta_treatment error_type n_reps bias_mean bias_sd cover_wald
##    <dbl>          <dbl> <chr>       <int>     <dbl>   <dbl>      <dbl>
##  1    10            0   heavy         471 -0.0190     0.945      0.928
##  2    10            0.5 heavy         470 -0.0578     0.895      0.932
##  3    10            2   heavy         469 -0.0329     1.02       0.928
##  4    50            0   heavy         471  0.0268     0.383      0.938
##  5    50            0.5 heavy         470  0.0154     0.404      0.964
##  6    50            2   heavy         469 -0.0190     0.388      0.957
##  7   500            0   heavy         471 -0.00255    0.125      0.955
##  8   500            0.5 heavy         470 -0.00467    0.130      0.949
##  9   500            2   heavy         469 -0.00173    0.133      0.951
## 10    10            0   normal        474 -0.0251     0.963      0.922
## 11    10            0.5 normal        472  0.00963    0.953      0.917
## 12    10            2   normal        472 -0.0157     0.966      0.909
## 13    50            0   normal        474  0.0178     0.428      0.930
## 14    50            0.5 normal        472  0.000417   0.402      0.947
## 15    50            2   normal        472 -0.0132     0.403      0.932
## 16   500            0   normal        474 -0.0133     0.128      0.947
## 17   500            0.5 normal        472  0.00738    0.133      0.936
## 18   500            2   normal        472 -0.00434    0.136      0.928
## # i 7 more variables: cover_perc <dbl>, cover_t <dbl>, se_mean <dbl>,
## #   se_sd <dbl>, time_wald_mean <dbl>, time_perc_mean <dbl>,
## #   time_boot_t_mean <dbl>
```

```r
library(tidyverse)

summary_tbl2 <- summary_tbl %>%
  mutate(error_type = recode(error_type, heavy = "t3", normal = "normal"),
         n = factor(n, levels = c(10, 50, 500)),
         beta_treatment = factor(beta_treatment, levels = c(0, 0.5, 2)))

cov_long <- summary_tbl2 %>%
  select(n, beta_treatment, error_type, cover_wald, cover_perc, cover_t) %>%
  pivot_longer(cols = starts_with("cover_"),
               names_to = "method", values_to = "coverage") %>%
  mutate(method = recode(method,
                         cover_wald = "Wald",
                         cover_perc = "Bootstrap percentile",
                         cover_t    = "Bootstrap-t"))

ggplot(cov_long, aes(x = n, y = coverage, color = method, group = method)) +
  geom_hline(yintercept = 0.95, linetype = "dashed") +
  geom_line(linewidth = 1) +
  geom_point(size = 2) +
  facet_grid(error_type ~ beta_treatment) +
  ylim(0, 1) +
  labs(title = "Empirical coverage across CI methods",
       x = "Sample size (n)",
       y = "Coverage",
       color = "Method")
```

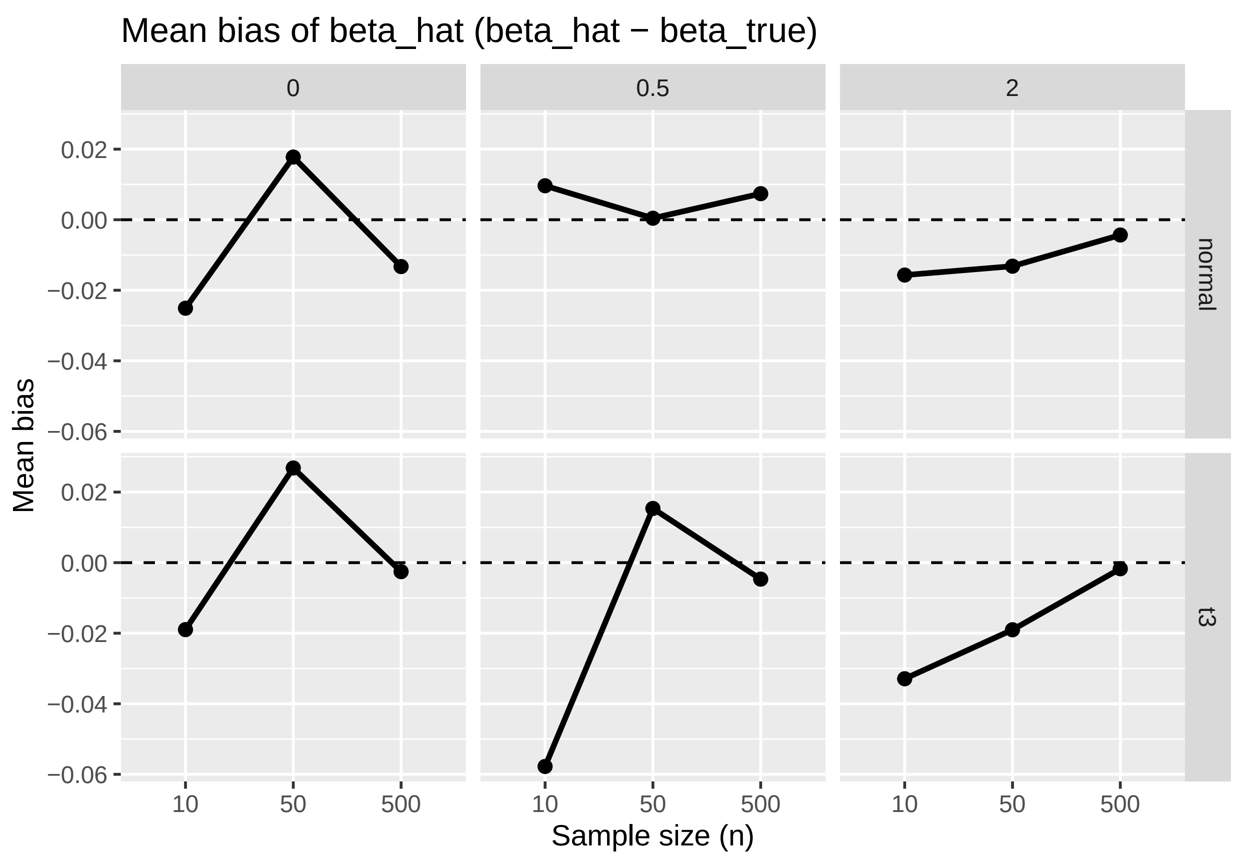
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (‘geom_line()‘).

## Warning: Removed 6 rows containing missing values or values outside the scale range
## (‘geom_point()‘).
```



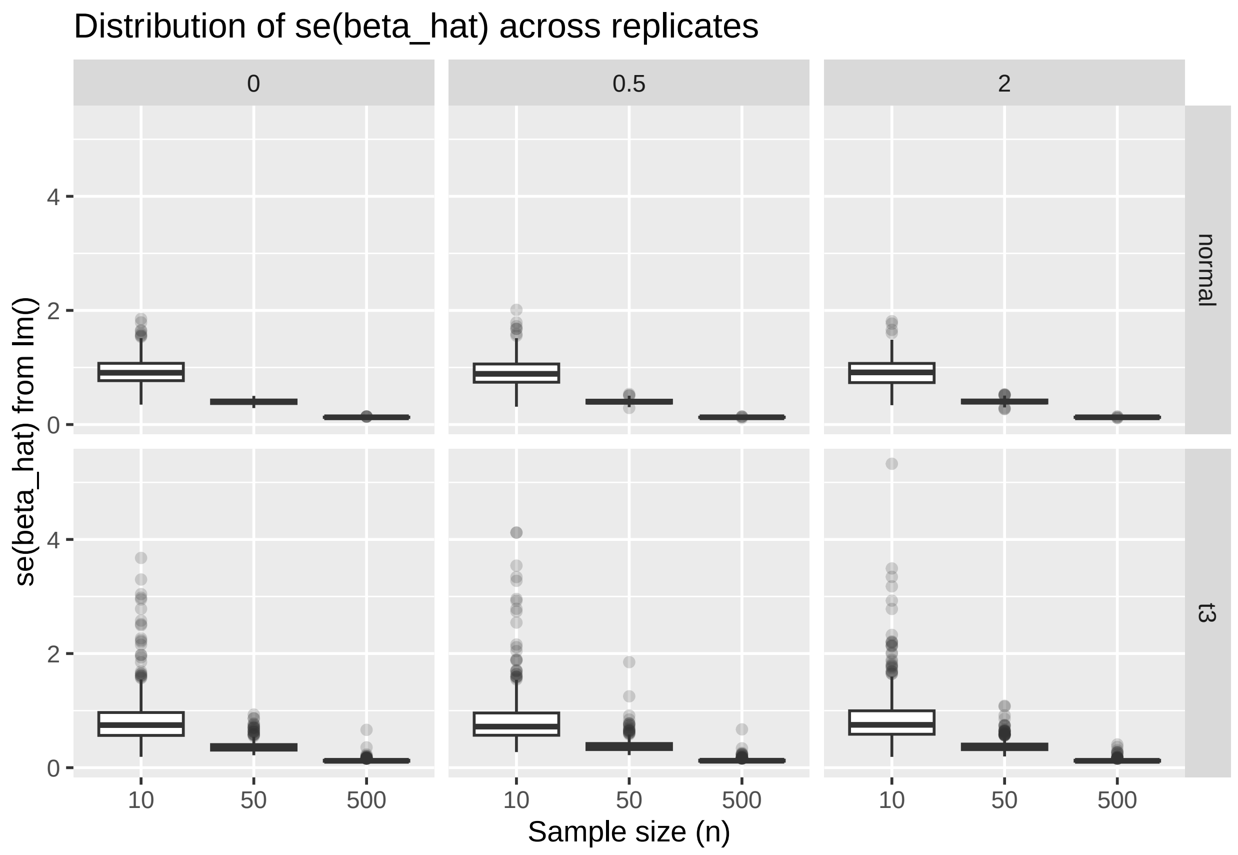
```
time_long <- summary_tbl2 %>%
  select(n, beta_treatment, error_type,
         time_wald_mean, time_perc_mean, time_boot_t_mean) %>%
  pivot_longer(cols = starts_with("time_"),
               names_to = "method", values_to = "mean_time") %>%
  mutate(method = recode(method,
                         time_wald_mean   = "Wald",
                         time_perc_mean   = "Bootstrap percentile",
                         time_boot_t_mean = "Bootstrap-t"))

ggplot(time_long, aes(x = n, y = mean_time, color = method, group = method)) +
  geom_line(linewidth = 1) +
  geom_point(size = 2) +
  scale_y_log10() +
  facet_grid(error_type ~ beta_treatment) +
  labs(title = "Computation time per replicate across CI methods",
       x = "Sample size (n)",
       y = "Mean time (seconds, log scale)",
       color = "Method")
```



```r
ggplot(summary_tbl2, aes(x = n, y = bias_mean, group = 1)) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  geom_line(linewidth = 1) +
  geom_point(size = 2) +
  facet_grid(error_type ~ beta_treatment) +
  labs(title = "Mean bias of beta_hat (beta_hat - beta_true)",
       x = "Sample size (n)",
       y = "Mean bias")
```



```r
## distribution of SE
se_raw3 <- raw %>%
  mutate(
    n = factor(n, levels = c(10, 50, 500)),
    beta_treatment = factor(beta_treatment, levels = c(0, 0.5, 2)),
    error_type = recode(error_type, normal = "normal", heavy = "t3")
  )

ggplot(se_raw3, aes(x = n, y = se_val)) +
  geom_boxplot(outlier.alpha = 0.2) +
  facet_grid(error_type ~ beta_treatment) +
  labs(
    title = "Distribution of se(beta_hat) across replicates",
    x = "Sample size (n)",
    y = "se(beta_hat) from lm()"
  )
```



**Problem 1.5 Discussion**

**Overall findings** Across all scenarios, estimator bias was small and decreased as sample size increased. Coverage and computation time varied substantially across CI methods, particularly under small sample sizes and heavy-tailed errors.

**Bias** The OLS estimator of $\beta$ was approximately unbiased across all scenarios. For $n = 10$, moderate variability in bias was observed, particularly under heavy-tailed (t3) errors, but bias approached zero as $n$ increased to 50 and 500.

This behavior is consistent with the unbiasedness of OLS and the effect of small-sample variability under non-normal errors.

**Coverage** Coverage patterns show clear differences across methods:

Under normal errors, all methods approached the nominal 0.95 level as $n$ increased.

For $n = 10$, the percentile bootstrap tended to under-cover ($\approx 0.85\check{}0.90$), especially when $\beta = 0$.

The Wald and bootstrap-t intervals were generally closer to the nominal level, particularly for moderate and large $n$.

Under heavy-tailed (t3) errors, differences were more pronounced:

At small sample sizes, coverage deteriorated for all methods.

Bootstrap percentile showed noticeable undercoverage at $n = 10$.

Bootstrap-t and Wald were generally more stable as $n$ increased.

By $n = 500$, all three methods achieved coverage close to 0.95.

Overall, sample size had a stronger impact on coverage than the true value of $\beta$.

**Standard Errors** The model-based standard error decreased substantially with increasing sample size, as expected from asymptotic theory.

Under heavy-tailed errors, the standard error was slightly inflated at small $n$, reflecting increased variability. However, differences across error types diminished as $n$ increased.

**Computation Time** Computation time differed dramatically across methods:

Wald CI was extremely fast (on the order of $10^{-3}$ seconds).

Bootstrap percentile required substantially more time ($\approx 0.3$˘0.5 seconds).

Bootstrap-t was by far the most computationally expensive (tens of seconds per replicate).

Bootstrap-t was roughly two orders of magnitude slower than percentile, due to the nested bootstrap required to estimate the studentized standard error.

Importantly, increases in sample size had only a modest effect on computation time relative to the dramatic differences between methods.

**Practical Implications** For moderate to large sample sizes and approximately normal errors, the Wald CI provides accurate coverage at negligible computational cost, making it highly efficient.

Under heavy-tailed errors and small samples, bootstrap methods can offer modest improvements in coverage, but the computational burden—particularly for bootstrap-t—is substantial.

Given the minor coverage differences observed for larger $n$, bootstrap-t may not justify its computational expense in practical applications unless small-sample robustness is critical.

**Interaction Effects** The primary interaction observed was between sample size and error distribution:

Small $n$ combined with heavy-tailed errors produced the largest deviations from nominal coverage.

As $n$ increased, differences between error distributions and CI methods diminished.

This confirms that asymptotic approximations perform well in moderate to large samples, even under non-normal errors.