A simulation study investigating the bootstrap

Jitong Jiang

2026-01-31

Github Link:

`https://github.com/jitongj/bios731_hw2_jiang.git`

1.1

Aim

To compare three methods for constructing 95% CIs for the treatment effect in a multiple linear regression model, and evaluate the CI methods performance across different sample sizes, true treatment effects, and error distributions, with a focus on bias, standard error behavior, coverage probability, and computation time.

- **Data-generating mechanism:**

For each simulation, assuming there $\gamma = 0$, we generate data from the multiple linear regression model:

$$Y_i = \beta_0 + \beta_{\text{treat}} X_{i1} + \varepsilon_i, \qquad i = 1, \dots, n,$$

where X_{i1} is generated by $Ber(0.5)$.

We consider the following factors:

- sample size: $n \in \{10, 50, 500\}$.

- True treatment effect: $\beta_{\text{treat}} \in \{0, 0.5, 2\}$.

- Error distribution: $\varepsilon_i \sim N(0, 2)$, and $\varepsilon_i \sim t_{\nu=3}$ with $\text{Var}(\varepsilon_i) = 2$.

Each combination of these factors is one simualtion scenario.

- **Estimand:**

The estimand is the β_{treat}

- **Methods:**

Wald confidence intervals, Nonparametric bootstrap percentile intervals, Nonparametric bootstrap   intervals

- **Performance measures:**

Bias, Monte Carlo SD, coverage

- **Number of scenarios:**

$$3 \times 3 \times 2 = 18.$$

1.2

$$n_{\text{sim}} = \frac{\widehat{\text{coverage}}\,(1-\widehat{\text{coverage}})}{\left(\text{SE}\left(\widehat{\text{coverage}}(\hat{\theta})\right)\right)^2}.$$

Given 95% coverage and MCSE ≤ 0.01:

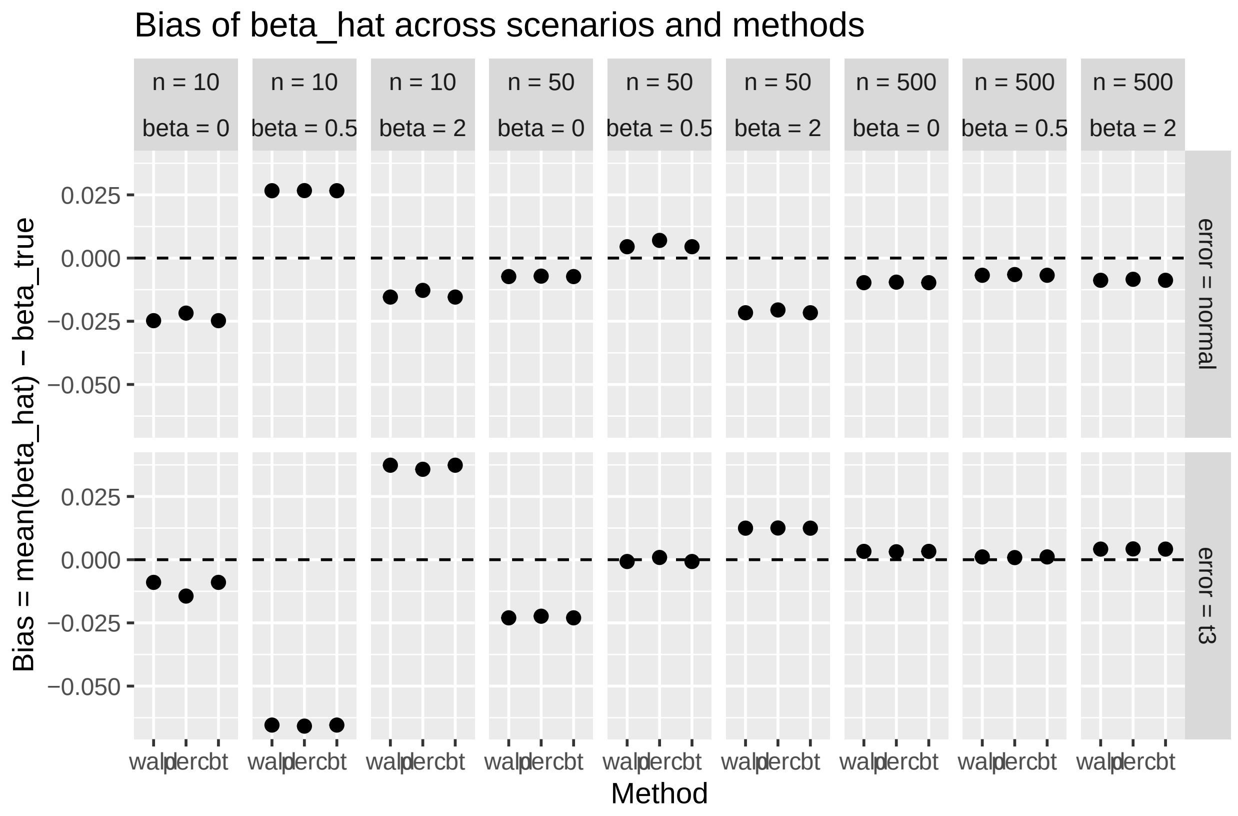
$$n_{\text{sim}} = \frac{0.95(1-0.95)}{(0.01)^2} = 475.$$

1.3

We reduced the bootstrap sizes from $B = 500$ and $B_{inner} = 100$ to $B = 200$ and $B_{inner} = 25$ to make the simulation computationally feasible. Bootstrap percentile CI accuracy improves at rate $O(B^{-1/2})$, and B=200 yields reasonably stable tail quantiles. For bootstrap-t, the nested re-sampling dominates running time, so I reduced B_{inner}, accepting slightly higher variability in the standard error estimates in exchange for a large reduction in running time.

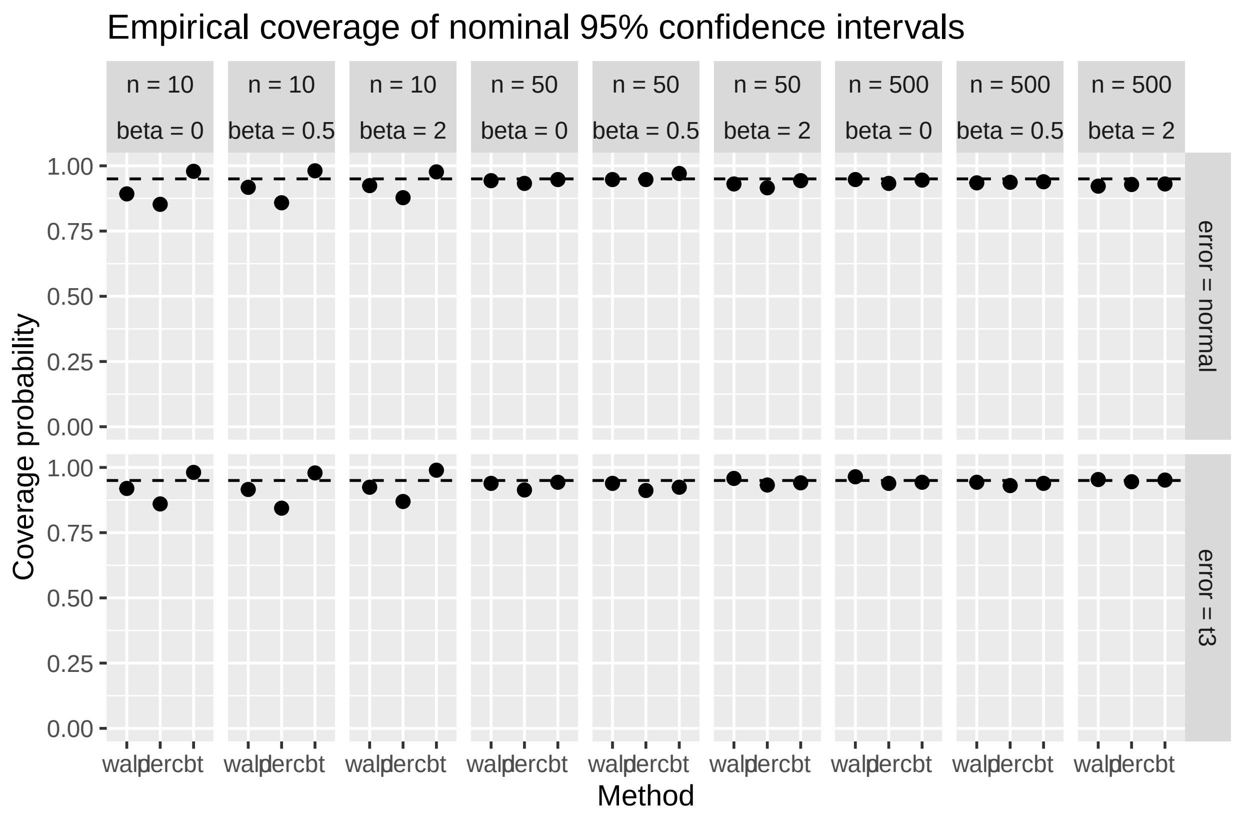
```r
library(tidyverse)
source(here::here("source", "05_compute_metric.R"))
out_dir <- here::here("data", "simulation_scenarios")

files <- list.files(out_dir, pattern = "\\.RData$", full.names = TRUE)

all_results <- purrr::map_dfr(files, function(f) {
  load(f)
  results_df
})

metrics_df <- compute_metrics(all_results)


all_results <- all_results %>%
  mutate(method = factor(method, levels = c("wald", "perc", "bt")))

metrics_df <- metrics_df %>%
  mutate(method = factor(method, levels = c("wald", "perc", "bt")))
```

```r
facet_labeller <- labeller(
  n = function(x) paste0("n = ", x),
  beta_true = function(x) paste0("beta = ", x),
  error_distr = function(x) paste0("error = ", x)
)

## Bias
p_bias <- metrics_df %>%
  ggplot(aes(x = method, y = bias)) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  geom_point(size = 2) +
  facet_grid(error_distr ~ n + beta_true, labeller = facet_labeller) +
  labs(
    title = "Bias of beta_hat across scenarios and methods",
    x = "Method",
    y = "Bias = mean(beta_hat) - beta_true",
    caption = "Each point represents the Monte Carlo bias for a given (n, beta_true, error distribution
  )

p_bias
```



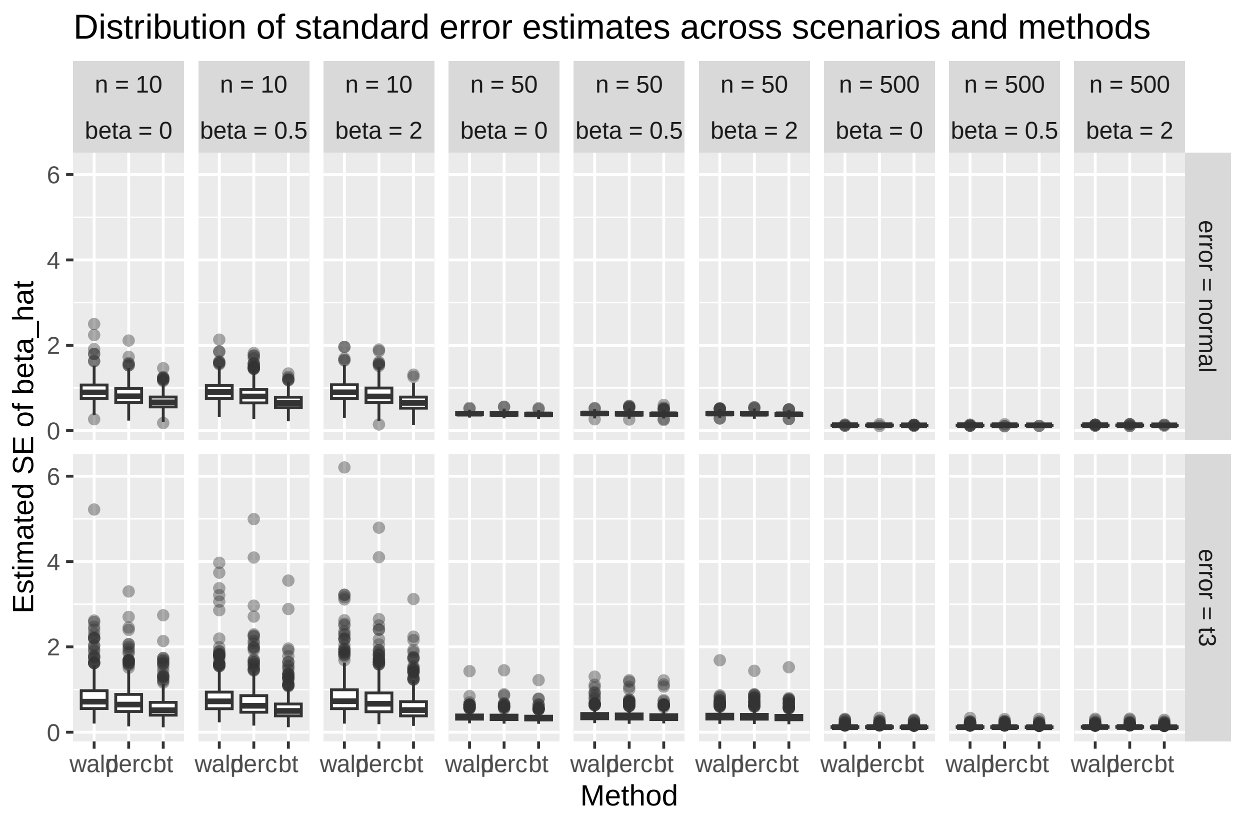
sents the Monte Carlo bias for a given (n, beta_true, error distribution) scenario. Dashed line indicates zero bias.

```
## CI
p_cov <- metrics_df %>%
  ggplot(aes(x = method, y = coverage)) +
  geom_hline(yintercept = 0.95, linetype = "dashed") +
  geom_point(size = 2) +
  facet_grid(error_distr ~ n + beta_true, labeller = facet_labeller) +
  coord_cartesian(ylim = c(0, 1)) +
  labs(
    title = "Empirical coverage of nominal 95% confidence intervals",
    x = "Method",
    y = "Coverage probability",
    caption = "Coverage is the empirical probability that the CI contains beta_true for each (n, beta_t
  )

p_cov
```



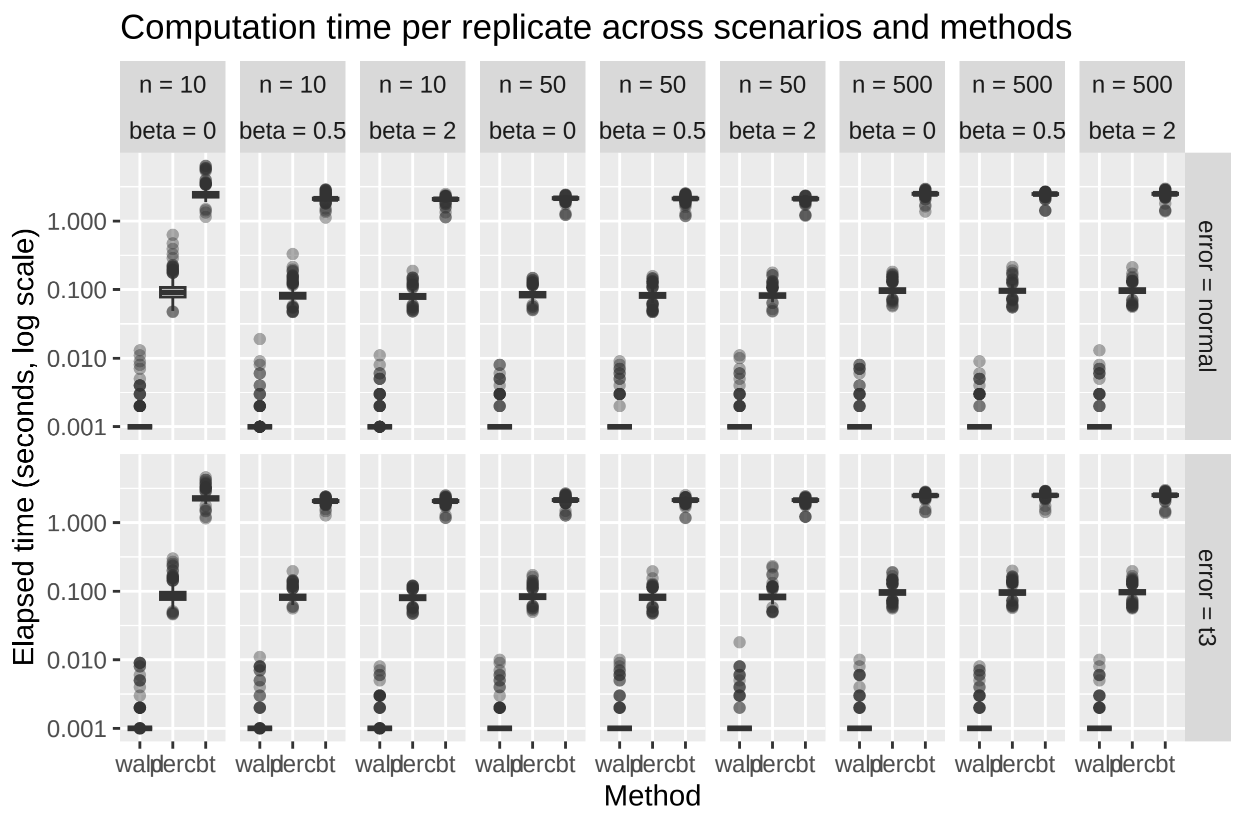
the CI contains beta_true for each (n, beta_true, error distribution) scenario. Dashed line indicates nominal 0.95.

```
## distribution

p_se <- all_results %>%
  ggplot(aes(x = method, y = se_hat)) +
  geom_boxplot(outlier.alpha = 0.4) +
  facet_grid(error_distr ~ n + beta_true, labeller = facet_labeller) +
  labs(
    title = "Distribution of standard error estimates across scenarios and methods",
    x = "Method",
    y = "Estimated SE of beta_hat",
    caption = "Boxplots show Monte Carlo distributions of method-specific SE estimates for each (n, beta
  )

p_se
```



ɔw Monte Carlo distributions of method−specific SE estimates for each (n, beta_true, error distribution) scenario.

```
## time
p_time <- all_results %>%
  ggplot(aes(x = method, y = time_sec)) +
  geom_boxplot(outlier.alpha = 0.4) +
  scale_y_log10() +
  facet_grid(error_distr ~ n + beta_true, labeller = facet_labeller) +
  labs(
    title = "Computation time per replicate across scenarios and methods",
    x = "Method",
    y = "Elapsed time (seconds, log scale)",
    caption = "Times are per Monte Carlo replicate; bootstrap-t is slower due to nested resampling. Fac
  )

p_time
```



bootstrap−t is slower due to nested resampling. Facets correspond to (n, beta_true, error distribution) scenarios.

(1)

Across all simulation scenarios, the point estimator $\hat{\beta}$ shows small bias that decreases with increasing sample size, regardless of the confidence interval construction method. Differences among methods arise primarily from interval construction rather than point estimation. For normally distributed errors, all three methods approach the nominal 95% coverage as n increases, while in small samples the bootstrap percentile interval tends to undercover more than Wald interval and bootstrap t intervals will overcover when n is small. For heavy-tailed errors, it shows generally the same trend. For the distribution of se, the results of two error distribution look generally the similar, with the variation decline as n gets bigger. For the computation time, the bootstrap t intervals method is the slowest since it requires the nested resampling scheme, while Wald intervals method is the fastest one.

(2)

- The Wald method is the fastest across all scenarios, with computation time largely insensitive to sample size or error distribution because it requires only a single model fit. The percentile bootstrap is slower due to repeated resampling but remains computationally feasible for moderate to large sample sizes. In contrast, the bootstrap-t method is substantially more expensive, especially as n increases, because it relies on a nested bootstrap to estimate t standard errors. The log-scale time plots show that bootstrap-t can be one to two orders of magnitude slower than the other methods, illustrating a clear accuracy–computation trade-off.
- When the errors are normally distributed, the Wald and bootstrap-based methods all achieve coverage close to the nominal 95% level for moderate to large sample sizes. However, in small samples (n=10), the Wald interval exhibits mild undercoverage which is better than bootstrap percentile method, while bootstrap-t intervals perform a little bit overcover. As sample size increases to n=50 and n=500, coverage differences among methods largely disappear, indicating that asymptotic normal approximations underlying the Wald interval become adequate in this setting. But since in the bootstrap setting, the B

and B_{inner} are set to be small to save computational time, I think when the $B = 500$ and $B_{inner} = 100$, the bootstrap t interval method would perform the best.

- For heavy-tailed errors, the bootstrap-t method provides the most reliable coverage across sample sizes, particularly in small samples. The percentile bootstrap improves upon the Wald interval but is somewhat less stable than bootstrap-t. Wald intervals consistently under-cover when errors are heavy-tailed and n is small or moderate, reflecting the failure of normality-based approximations. Overall, the bootstrap-t method is best suited for capturing the additional variability induced by heavy-tailed error distributions

(3)

Method performance depends strongly on both sample size and error distribution. Differences across methods are most pronounced in small samples and under heavy-tailed errors, while all methods converge as n increases. In practice, Wald intervals are suitable for large samples with approximately normal errors due to their efficiency. Bootstrap-based methods are preferable for small samples or non-normal errors, with the percentile bootstrap offering a reasonable balance between improved coverage and computational cost, and bootstrap-t providing the most robust coverage when computational resources allow.