Hierarchical Logistic Model for Mulit-Center Clinical Trial

P8160 - Group 3A

Kate Colvin (kac2301), Xuanyu Guo(xg2451), Dang Lin (dl3757), Boxiang Tang (bt2654)

2025-03-03

Introduction and Background

A multicenter clinical trial is a trial that enrolls participants at multiple different medical institutions. While there are a number of distinct advantages to multicenter trials, such as the possibilities for a larger sample size and greater generalizability [Jo, 2020], they are also expensive and more complex to execute and analyze. For instance, different trial sites can have different characteristics, such as patient demographics, treatment practices, or geographic factors, that impact the outcome of trial patients. Such variability between trial sites can complicate the statistical analysis, and possibly bias trial outcomes. Moreover, in some cases, multicenter clinical trials have contradicted single-center trials [Dechartres, 2011]. Therefore an important goal of statistical clinical trial research must be to create methods and guidelines for the reliable design and analysis of multicenter clinical trials.

A challenge in evaluating clinical trial methodology is the cost of conducting a trial and the accessibility of this data to external researchers. It's been long recognized that simulation studies can play a key role in designing and analyzing clinical trials by improving their efficiency and reliability [Bonate, 2000]

In this project, we designed a simulation study to determine optimal Monte Carlo sampling strategies to evaluate the population-level probability of adverse events in the context of a hypothetical multicenter clinical trial. This simulated trial includes patient-level covariates and clinic-level random effects that informs a heirarchical logistic model to predict the probability of adverse events.

Probability of an adverse event for patient i in clinic j:

$$p_{i,j} = \text{logit}^{-1}(\alpha + b_j + \beta X_i)$$

X_i: Each patient i has a continuous risk factor X_i (disease severity score)

b_j: Random effects at the clinic level

α: The baseline log-odds of an adverse event when $X_i = 0$ and there is no clinic-level random effect

β: The effect size of the patient-level covariate X_i, controlling for clinic-level random effects

We evaluated the bias and variance in the predicted probability of an adverse event of three sampling methods: simple Monte Carlo (MC), MC with control variates (CV), and MC with importance sampling (IS). We also compare the CPU time statistics for each method.

Statistical Methods

In order to evaluate our different Monte Carlo sampling stratgies, we created a synthetic data set meant to represent a hypothetical multicenter clinical trial. This data set includes patient-level covariates X_i and clinic-level random effects b_j, which represents site-specific variables such as population demographics and differences in protocol. In section 1 below, we specify the clinic random effect as $b_j \sim$ Log-Normal(-1, 0.5), which is a heavy tailed distribution (i.e. decays slower than an Exponential distribution), and the patient

level covariates were specified as $X_i \sim$ Gamma(2, 2). Distribution parameters were chosen to yield realistic estimates of adverse events [Luo, 2016].

These random variables are used to estimate the population-level probability of an adverse event across all patients and clinics using the logit function below:

$$P(AdverseEvent) = \int\int logit^{-1}(\alpha + b + \beta x) f_B(b) f_X(x) dbdx$$

We evaluated three different Monte Carlo sampling methods: simple Monte Carlo (MC), MC with control variates (CV), and MC with importance sampling (IS) for evaluating the integral above. The equations for each method are given below, where X_i and b_i are selected from the appropriate probability distributions $f_X(x)$ and $f_B(b)$, respectively, $U(x_i, b_i)$ is a control variate with a known expectation value, μ_U, and $g_X(x_i)$ and $g_B(b_i)$ are easy to sample distributions that are non-zero over the same ranges as $f_X(x_i)$ and $f_B(b_i)$, respectively.

$$P_{Simple}(AdverseEvent) = \frac{1}{N}\sum_{i=1}^{N} logit^{-1}(f(x_i, b_i))$$

$$P_{CV}(AdverseEvent) = \frac{1}{N}\sum_{i=1}^{N} (logit^{-1}(f(x_i, b_i)) - c^*(U(x_i, b_i)) - \mu_U)$$

$$P_{IS}(AdverseEvent) = \frac{1}{N}\sum_{i=1}^{N} \frac{f_X(x_i) f_B(b_i)}{g_X(x_i) g_B(b_i))} logit^{-1}(f(x_i, b_i))$$

For each method, we report the bias and variance in the predicted probability of an adverse event from each method, where the bias is calculated in comparison to the "true probability", which was estimated using a simple Monte Carlo simulation with N = 100,000,000. We also provide CPU time statistics to gauge the efficiency of each method and cumulative convergence plots to compare the convergence of each method to the "true probability".

In the sections below, we defined functions used to run the integrations using simple Monte Carlo (Section 2), control variates (Section 3), and importance sampling (Section 4) methods.

Task 1: Specifying Distributions

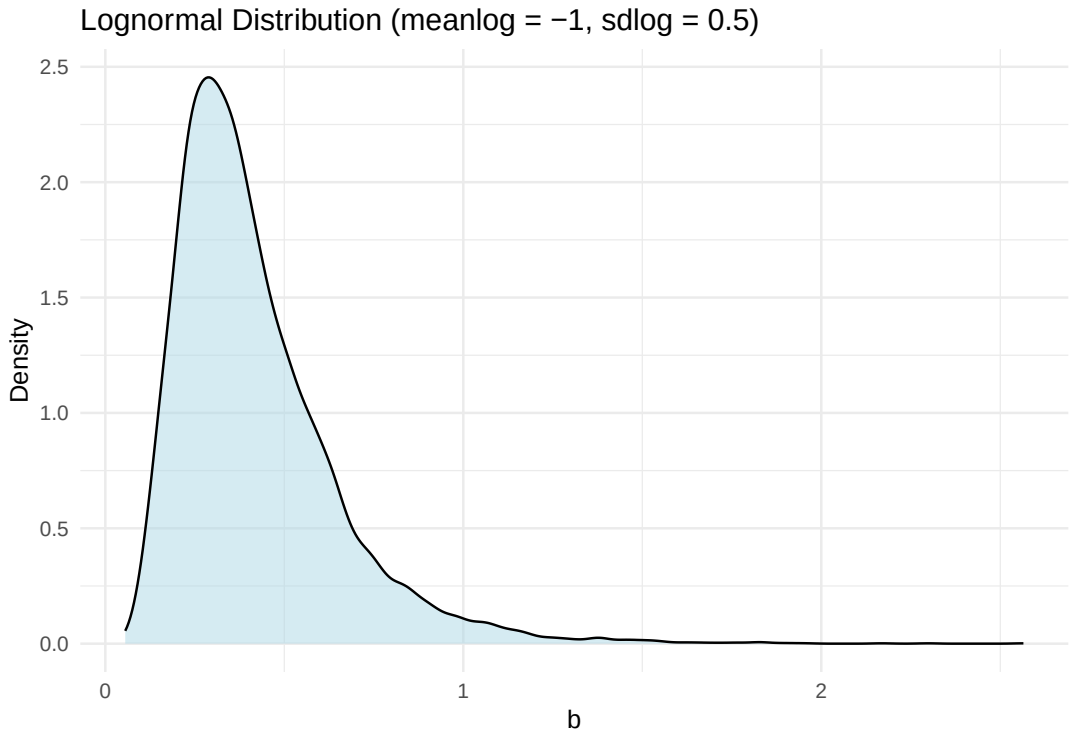
Let b follow a Log-normal distribution with parameters meanlog = -1 and sdlog = 0.5, see graph below.

$$b \sim \text{Lognormal}(-1, 0.5)$$

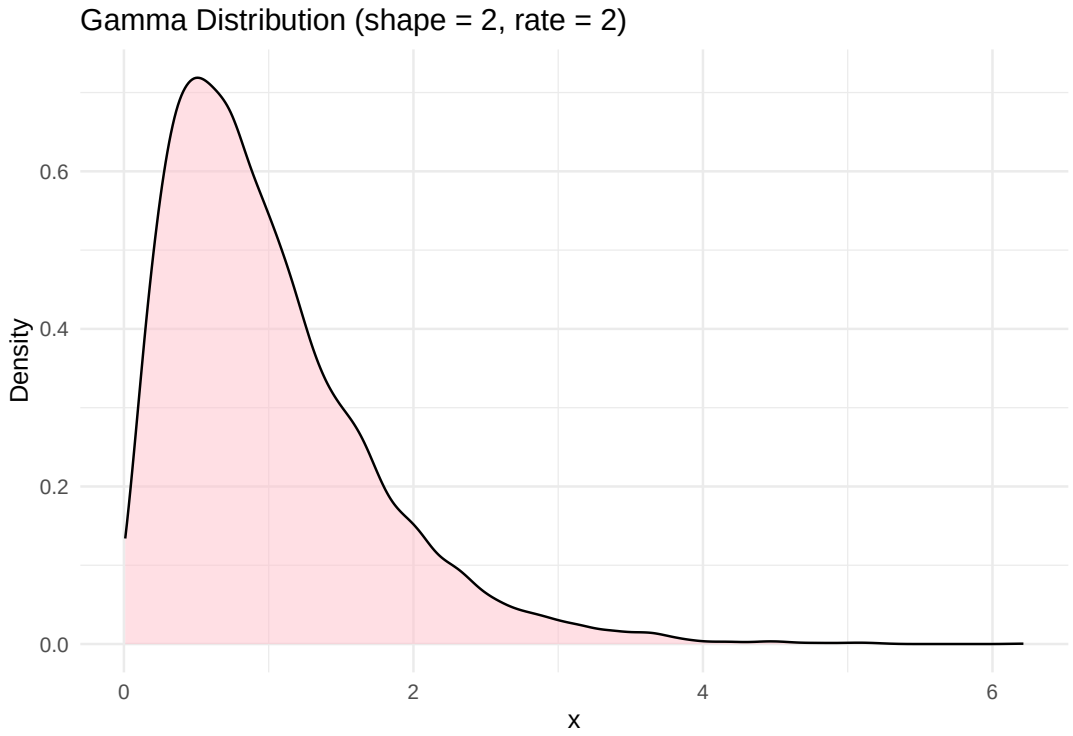
Let x follows a Gamma distribution with parameters shape = 2 and rate = 2, see graph below.

$$x \sim \text{Gamma}(2, 2)$$

```
set.seed(8160)
n <- 10000

## Pre-calculating b samples and x samples for use in simple MC and CV
b_samples <- rlnorm(n, meanlog = -1, sdlog = 0.5)
x_samples <- rgamma(n, shape = 2, rate = 2)

lognormal_df <- data.frame(values = b_samples,
                           distribution = "Lognormal(-1, 0.5)")
gamma_df <- data.frame(values = x_samples,
                       distribution = "Gamma(2, 2)")

ggplot(lognormal_df, aes(x = values)) +
  geom_density(fill = "lightblue", alpha = 0.5) +
  labs(title = "Lognormal Distribution (meanlog = -1, sdlog = 0.5)",
       x = "b", y = "Density") +
  theme_minimal()
```




```r
ggplot(gamma_df, aes(x = values)) +
  geom_density(fill = "pink", alpha = 0.5) +
  labs(title = "Gamma Distribution (shape = 2, rate = 2)",
       x = "x", y = "Density") +
  theme_minimal()
```



Task 2: Sampling using Simple Monte Carlo (SMC)

In simple Monte Carlo sampling, random samples `b_samples` and `x_samples` are generated to estimate the overall probability of an adverse event. Firstly, the parameters α and β are defined, and the logit inverse function is applied to compute the probability of an adverse event for each individual sample. The overall probability estimate is then obtained by averaging these probabilities, while the variance of the method is calculated based on the individual probability values from each sample.

Using simple Monte Carlo with n = 10000: $P_{Simple}(AdverseEvent) = \frac{1}{n}\sum_{i=1}^{n} logit^{-1}(f(x_i, b_i))$

```
start_time_smc <- Sys.time()

alpha <- -2
beta <- 0.5

logit_inverse_function <- function(x) {
  return(exp(x) / (1 + exp(x)))
}

## Calculating the probability of an adverse event based on pre-sampled b and x
p_ij <- logit_inverse_function(alpha + b_samples + beta * x_samples)

p_adverse_event_smc <- mean(p_ij)
var_smc <- var(p_ij)

cpu_time_smc <- Sys.time() - start_time_smc
```

Task 3: Sampling using Control Variate (CV)

The Control Variates method is used to improve the efficiency of Monte Carlo estimation by reducing variance through the use of a correlated control variate. Firstly, the probability of an adverse event is estimated using the logit inverse function applied to the Patient-Level sampled covariates and Clinic-Level random effects. Then, a control variate U is computed as the linear predictor before the logit transformation. The optimal correction coefficient c^* is calculated based on the covariance method. Finally, the overall probability estimate is obtained by applying the control variate correction, reducing variance without introducing bias. The efficiency of this method is evaluated by comparing the mean and variance of the estimates before and after the correction.

Using Monte Carlo with Control Variate: $P_{CV}(AdverseEvent) = \frac{1}{N}\sum_{i=1}^{N}(logit^{-1}(f(x_i, b_i)) - c^*(U(x_i, b_i)) - \mu_U)$, where the control variate (U) is the argument to the inverse logit function, $\alpha + b + \beta x$, where we can analytically determine the means for variables b and X.

```
start_time_cv <- Sys.time()

Y <- logit_inverse_function(alpha + b_samples + beta * x_samples)
E_Y <- mean(Y)
var_Y <- var(Y)

## U is the logit transformed Y (i.e. the original argument to the inverse logit)
U <- alpha + b_samples + beta * x_samples
E_U <- mean(U)
var_U <- var(U)

## E(Gamma(2,2)) = 1, E(Log-Normal(-1, 0.5)) = exp(-1 + (0.5^2)/2)
E_U_exact <- alpha + exp(-1 + (0.5^2)/2) + beta*1
```

```r
cov_Y_U <- cov(Y, U)
c_star <- cov_Y_U / var_U

## Applying covariate correction
Y_cv <- Y - c_star * (U - E_U_exact)
E_Y_cv <- mean(Y_cv)
var_Y_cv <- var_Y - (cov_Y_U ^ 2) / var_U

cpu_time_cv <- Sys.time() - start_time_cv

control_variate_df <- data.frame(
  Estimates = c("Mean", "Variance"),
  Without_Control_Variate = c(E_Y, var_Y),
  With_Control_Variate = c(E_Y_cv, var_Y_cv)
)
```

Task 4: Sampling using Importance Sampling (IS)

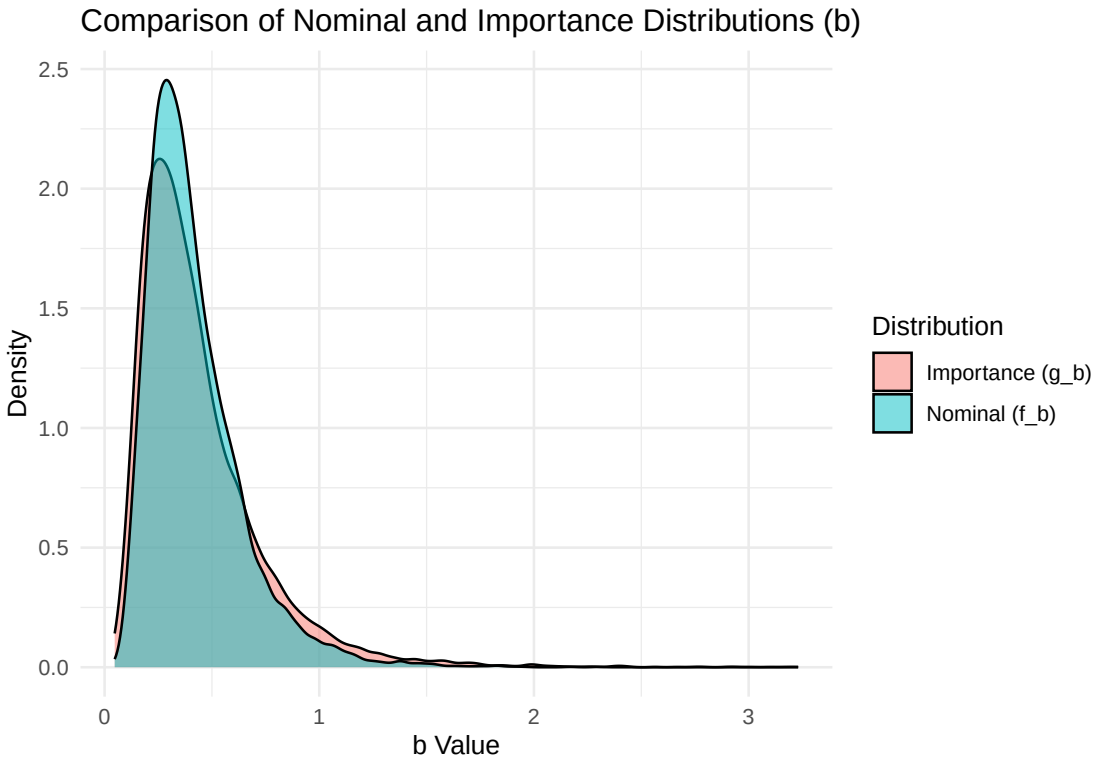
The Importance Sampling method is applied to estimate the overall probability of an adverse event by drawing samples from a carefully chosen importance distribution instead of the nominal distribution. The nominal distributions for the clinic-level random effects and patient-level covariates are modeled as Log-Normal and Gamma distributions, respectively. To improve efficiency, we selected importance distributions that closely approximate the nominal distributions, covering their domains while having slightly heavier tails to ensure sufficient coverage of high-probability regions.

To be more specific, the Log-Normal importance distribution is adjusted by increasing the standard deviation, while the Gamma importance distribution is modified by slightly increasing the shape parameter. After generating samples from these distributions, we computed the probability of an adverse event and adjusted each sample's contribution using importance weights, derived as the ratio of the nominal to the importance density. The final estimate is obtained as a weighted average, and the variance of the Importance Sampling estimator is calculated to evaluate its efficiency compared to other sampling methods.

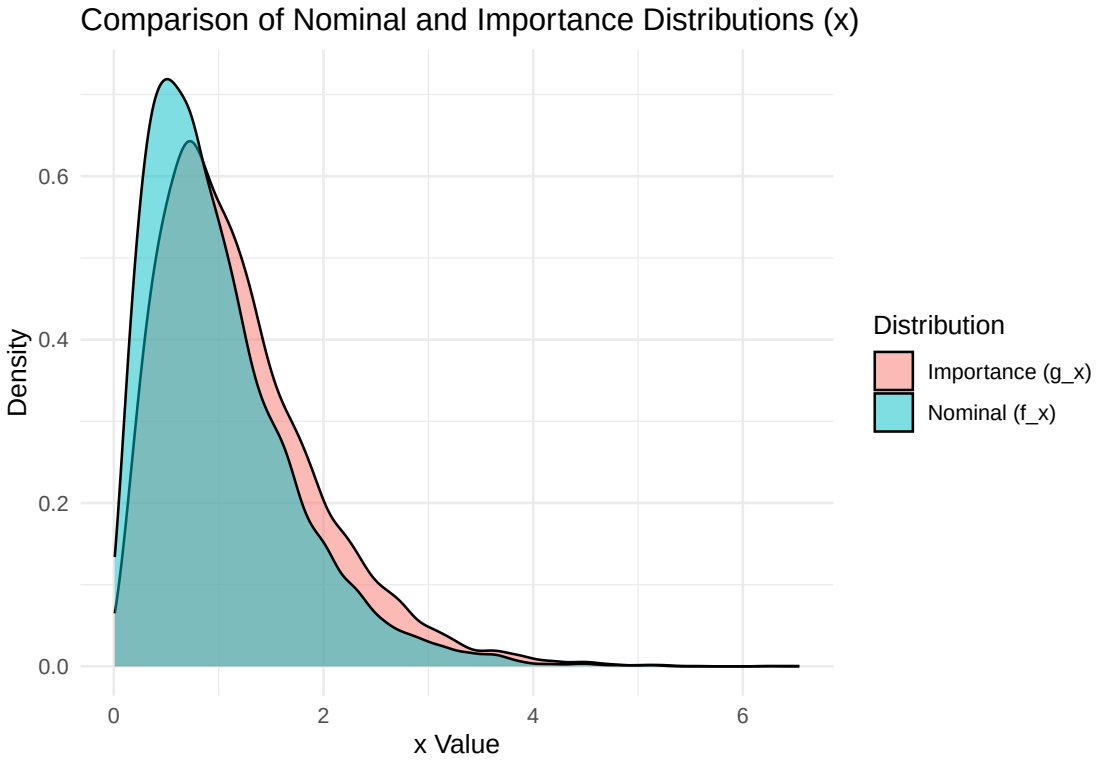
Using importance sampling: $P_{IS}(AdverseEvent) = \frac{1}{N}\sum_{i=1}^{N} \frac{f_X(x_i)f_B(b_i)}{g_X(x_i)g_B(b_i))} logit^{-1}(f(x_i, b_i))$, where $g_X(x_i) \sim$ Gamma(2.4, 2) and $g_B(b_i) \sim$ Log-Normal(-1, 0.6).

First, we compare the nominal and importance distributions in the plots below:

```r
set.seed(8160)

b_samples_nominal <- rlnorm(n, meanlog = -1, sdlog = 0.5)
x_samples_nominal <- rgamma(n, shape = 2, rate = 2)
b_samples_importance <- rlnorm(n, meanlog = -1, sdlog = 0.6)
x_samples_importance <- rgamma(n, shape = 2.4, rate = 2)

b_df <- data.frame(Value = c(b_samples_nominal, b_samples_importance),
                   Distribution = rep(c("Nominal (f_b)", "Importance (g_b)"), each = n))

x_df <- data.frame(Value = c(x_samples_nominal, x_samples_importance ),
                   Distribution = rep(c("Nominal (f_x)", "Importance (g_x)"), each = n))

ggplot(b_df, aes(x = Value, fill = Distribution)) +
  geom_density(alpha = 0.5) +
  labs(title = "Comparison of Nominal and Importance Distributions (b)",
       x = "b Value", y = "Density") +
  theme_minimal()
```




```
ggplot(x_df, aes(x = Value, fill = Distribution)) +
  geom_density(alpha = 0.5) +
  labs(title = "Comparison of Nominal and Importance Distributions (x)",
       x = "x Value", y = "Density") +
  theme_minimal()
```



Here, we implement the importance sampling:

```
set.seed(8160)

start_time_is <- Sys.time()

# Nominal Distribution
f_b <- function(b) {
  return(dlnorm(b, meanlog = -1, sdlog = 0.5))
  }
f_x <- function(x) {
  return(dgamma(x, shape = 2, rate = 2))
  }

# Importance Distribution
g_b <- function(b) {
  return(dlnorm(b, meanlog = -1, sdlog = 0.6))
}
g_x <- function(x) {
  return(dgamma(x, shape = 2.4, rate = 2))
}

weights <- function(b, x) {
  return((f_b(b) * f_x(x)) / (g_b(b) * g_x(x)))
}

b_samples_importance <- rlnorm(n, meanlog = -1, sdlog = 0.6)
x_samples_importance <- rgamma(n, shape = 2.4, rate = 2)

p_ij_is <- logit_inverse_function(alpha + b_samples_importance + beta * x_samples_importance)
w <- weights(b_samples_importance, x_samples_importance)

E_is <- sum(w * p_ij_is) / sum(w)
## Variance expression from lecture notes
var_is <- 1/n * (mean(w^2) - mean(w)^2)

cpu_time_is <- Sys.time() - start_time_is

importance_sampling_results <- data.frame(
  Estimates = c("Mean", "Variance"),
  Value = c(E_is, var_is)
)
```

Results

Task 5: Comparison of Sampling Methods

```
set.seed(8160)

N_true <- 100000000

b_samples_N <- rlnorm(N_true, meanlog = -1, sdlog = 0.5)
x_samples_N <- rgamma(N_true, shape = 2, rate = 2)
```

```r
p_adverse_event_true <- mean(logit_inverse_function(alpha + b_samples_N + beta * x_samples_N))

bias_smc <- p_adverse_event_smc - p_adverse_event_true
bias_cv <- E_Y_cv - p_adverse_event_true
bias_is <- E_is - p_adverse_event_true

comparison_df <- data.frame(
  Method = c("Simple Monte Carlo", "Control Variates", "Importance Sampling"),
  Estimated_Probability = c(p_adverse_event_smc, E_Y_cv, E_is),
  Bias = c(bias_smc, bias_cv, bias_is),
  Variance = c(var_smc, var_Y_cv, var_is),
  CPU_Time = c(cpu_time_smc, cpu_time_cv, cpu_time_is))

knitr::kable(comparison_df, digits = 10,
      caption = "Comparison of Sampling Methods (Bias, Variance, and CPU Time)")
```

Table 1: Comparison of Sampling Methods (Bias, Variance, and CPU Time)

Method	Estimated_Probability	Bias	Variance	CPU_Time
Simple Monte Carlo	0.2606976	0.0006902084	0.0072318122	0.001332998 secs
Control Variates	0.2599719	-0.0000355375	0.0000800636	0.003443956 secs
Importance Sampling	0.2604511	0.0004437220	0.0000174761	0.004683971 secs

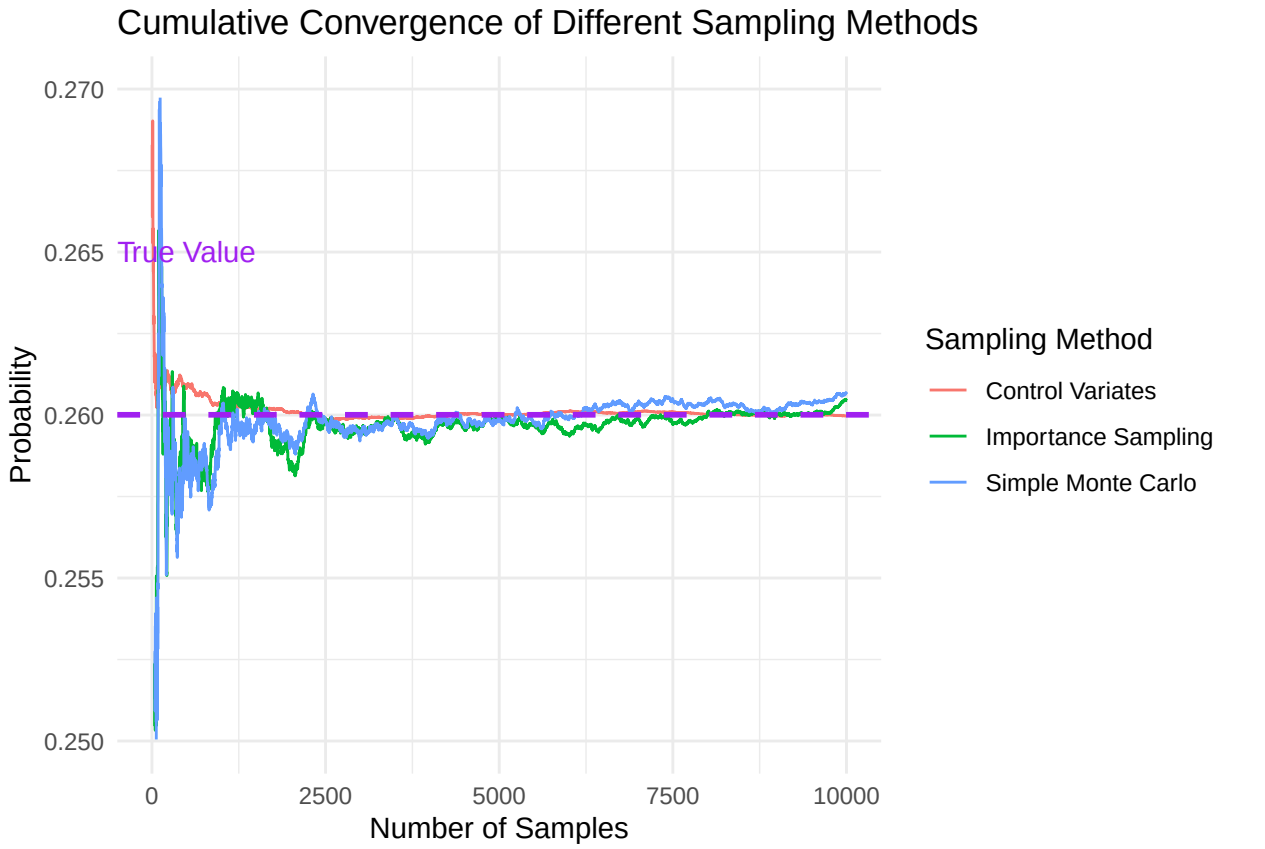
All three sampling methods produce nearly unbiased estimates, with the control variate method demonstrating a slightly smaller bias compared to the other two methods. The simple Monte Carlo method has the highest variance among the three because it relies solely on independent random sampling without any variance reduction techniques. The Control Variates method reduces variance by incorporating a correlated variable to adjust the estimate. Importance Sampling achieves the lowest variance as it samples from a more relevant distribution, focusing on high-probability regions and thereby improving estimation efficiency.

In terms of computational efficiency, Simple Monte Carlo is the fastest since it involves only direct sampling and averaging. Control Variates requires additional computations to estimate correlations and adjust the variance, making it slightly more computationally expensive than Simple Monte Carlo. Importance Sampling has the highest computational cost as it requires evaluating the importance weights, normalizing them, and computing a weighted average. Note that being twice as computationally expensive as simple Monte Carlo, the control variate method converges much faster, and therefore should require fewer Monte Carlo trials and reducing this difference in cost.

Task 6: Cumulative Convergence Plots (Extra Credit)

```r
cumulative_mc <- cumsum(p_ij) / seq_along(p_ij)
cumulative_cv <- cumsum(Y_cv) / seq_along(Y_cv)
cumulative_is <- cumsum(w * p_ij_is) / cumsum(w)

cumulative_convergence_df <- data.frame(
  Sample = rep(1 : 10000, 3),
  Value = c(cumulative_mc, cumulative_cv, cumulative_is),
  Method = rep(c("Simple Monte Carlo", "Control Variates", "Importance Sampling"),
               each = n)
)
```

```r
ggplot(cumulative_convergence_df, aes(x = Sample, y = Value, color = Method)) +
  geom_line(size = 0.5) +
  geom_hline(yintercept = p_adverse_event_true, linetype = "dashed",
             color = "purple", size = 1) +
  annotate("text", x = 500, y = p_adverse_event_true + 0.005,
           label = "True Value", size = 4, color = "purple") +
  labs(title = "Cumulative Convergence of Different Sampling Methods",
       x = "Number of Samples",
       y = "Probability",
       color = "Sampling Method") +
  theme_minimal() +
  ylim(c(0.25, 0.27))
```



The cumulative convergence plot illustrates how the estimated probability of an adverse event converges over increasing sample sizes for three different sampling methods. First, Simple Monte Carlo (blue line) and Importance Sampling (green line) demonstrate high fluctuations within the first 2500 samples. As the sample size continues to increase, the estimates becomes more stable, but towards the end, simple Monte Carlo remains more volatile compared to the other two methods. The Control Variate method (red line) remains very smooth and close to the true value after the first 1000 samples. All three methods converge close to the true probability (purple dashed line), illustrating their unbiased nature. However, the differences in their convergence speed show the advantages of variance reduction techniques, and overall it seems that the control variate method is the most consistent across all samples.

Discussion and Practical Implications

This study has shown that the use of improved sampling methods in Monte Carlo simulations of even relatively simple models of multicenter cinical trials significantly improves the convergence of predicted outcome values. This is at a cost of more complex coding and computation time, as well as the need to identify additional appropriate distributions for importance sampling and to be control variates.

Simulation studies allow investigators clinical trials to evaluate the effect of different interventions on health outcomes. For examples like the application studied in this project, the use of simulation allows us to test how changes in the distributions of the demographics of patients or clinic random effects can impact the probability of adverse events overall. Going forward, this information could be used to determine which patients are select for procedures, or which clinics different patients are sent to. The ability to effectively simulate multicenter trials also has the potential to allow us to optimize the time and monetary costs of conducting a clinical trial, speeding up the process of conducting clinical research and improving their efficiency and reliability [Bonate, 2000]. Additionally, outside of clinical trials, these methods could be generalized for wider applications in healthcare, such as for resource allocation.

Conclusion

Of the methods evaluated, both the control variate method and importance sampling reducing the variance compared to simple Monte Carlo sampling, but at 2x and 4x the computational cost. Of the improved sampling methods, the control variate method converges much more quickly to the known true value, and therefore may be the most accurate and efficient.

References

Bonate, Peter L. “Clinical trial simulation in drug development.” Pharmaceutical research 17 (2000): 252-256.

Dechartres, Agnes, et al. “Single-center trials show larger treatment effects than multicenter trials: evidence from a meta-epidemiologic study.” Annals of internal medicine 155.1 (2011): 39-51.

Jo, Daehyun. “The interpretation bias and trap of multicenter clinical research.” The Korean journal of pain vol. 33,3 (2020): 199-200. doi:10.3344/kjp.2020.33.3.199

Luo, Jake, et al. “Population analysis of adverse events in different age groups using big clinical trials data.” JMIR medical informatics 4.4 (2016): e6437