# Simulation Studies on Optimal Experimental Design under Budget Constraints

William Qian

2024-12-01

## Abstract

This study investigates optimal experimental designs under budget constraints through comprehensive simulation studies, focusing on clustered study designs where treatments are assigned at the cluster level. Using the ADEMP framework, we systematically examined how the number of clusters ($G$) and observations per cluster ($R$) influence estimation accuracy under various parameter configurations and cost constraints. Our simulations explored both normal and Poisson-distributed outcomes, incorporating key parameters including treatment effect size ($\beta$), between-cluster variance ($\gamma^2$), within-cluster variance ($\sigma^2$), and the ratio of cluster to individual-level costs ($c_1/c_2$). Results show that while the mean parameters ($\alpha$, $\beta$) had negligible impact on optimal designs, variance components significantly influenced design efficiency. Higher between-cluster variance generally favored designs with more clusters and fewer observations per cluster, particularly for Poisson-distributed outcomes. The cost ratio emerged as a crucial determinant, with higher $c_1/c_2$ ratios leading to fewer but larger clusters. These findings provide practical guidance for researchers designing cluster-randomized trials under budget constraints, demonstrating how to optimize resource allocation between clusters and within-cluster observations based on distributional assumptions, variance components, and cost structures.

## Introduction

Optimal experimental design is a cornerstone of rigorous scientific research, aiming to maximize the efficiency and validity of experiments through strategic decisions on sample size, treatment allocation, and measurement strategies. However, researchers often face significant budget constraints that limit available resources, necessitating the development of experimental designs that achieve research objectives while adhering to financial limitations. Clustered study designs—where observations are grouped into clusters (such as schools, hospitals, or

communities) and treatments are assigned at the cluster level—are frequently employed across various fields, including medicine, social sciences, and agriculture. While these designs offer logistical advantages and cost-effectiveness, they introduce complexities due to intra-cluster correlations that must be accounted for in both the design and analysis phases.

This study explores optimal experimental designs under budget constraints through comprehensive simulation studies. By systematically varying key parameters and data collection costs, we aim to identify designs that minimize estimation errors within predefined budget limits. Specifically, we focus on: (1) evaluating the performance of different feasible designs under a fixed budget for normal and Poisson-distributed outcomes, examining how the number of clusters (G) and observations per cluster (R) influence bias, mean squared error (MSE), and coverage probability of estimated treatment effects; (2) investigating the relationships between data-generating parameters—such as treatment effect size, between-cluster variance, and within-cluster variance—and total costs to inform optimal resource allocation between clusters and observations within clusters; and (3) extending the simulation study to Poisson-distributed outcomes common in count data, modifying data generation and analysis methods accordingly, and assessing how optimal designs differ from those with normally distributed outcomes. To achieve these aims, we develop a suite of functions in R for data simulation, balanced treatment assignment, treatment effect estimation using generalized linear models with cluster-robust standard errors, and design performance evaluation. The insights from our simulations contribute to a deeper understanding of resource allocation in experimental planning, guiding researchers in designing studies that are both cost-effective and statistically robust.

## Simulation Design

We use ADEMP framework to conduct the simulation study.

### Aim

Our aim is to estimate the treatment effect in clustered experimental designs under budget constraints. We seek to identify the optimal allocation between the number of clusters (G) and the number of observations per cluster (R) that provides accurate and efficient estimation of the treatment effect within a fixed budget.

### Data Generation

#### Clustered Data Structure

We consider the structure of the clusters as follows:

- Clusters: $j = 1, 2, ..., G$
- Observations within clusters: $i = 1, 2, ..., R$

Then, the total number of observations is $N = G \times R$.

### Data Generation Process

Each cluster $j$ is randomly assigned to either the treatment group ($X_j = 1$) or the control group ($X_j = 0$). The treatment effect is defined as $\beta$, representing the difference in the outcome between the treatment and control groups. The data generation process is as follows:

### Normal Distribution

We assume a hierarchical linear model for $Y_{ij}$:

$$\begin{aligned} \mu_{i0} &= \alpha + \beta X_j \\ \mu_i|\epsilon_j &= \mu_{i0} + \epsilon_j \text{ with } \epsilon_j \sim N(0, \gamma^2) \\ Y_{ij}|\mu_i &= \mu_i + e_{ij} \text{ with } e_{ij} \sim N(0, \sigma^2) \end{aligned}$$

### Poisson Distribution

For Poisson-distributed outcomes, we assume a hierarchical Poisson model for $Y_{ij}$:

$$\begin{aligned} log(\mu_{i0}) &= N(\alpha + \beta X_j, \gamma^2) \\ Y_{ij}|\mu_i &= Poisson(\mu_i) \end{aligned}$$

To be more detailed, $\gamma^2$ can be understood as the between-cluster variance, $\sigma^2$ as the within-cluster variance.

### Cost Constraints

The total cost $C$, cost per cluster $c_1$ and cost per observation $c_2$ are fixed in each simulation ($c_2 << c_1$). The total cost is calculated as:

$$C = G \times (c_1 + (R - 1) \times c_2)$$

## Estimands

The estimand of interest is the treatment effect $\beta$, which represents the difference in the outcome between the treatment and control groups. We aim to estimate $\beta$ with minimal bias and mean squared error (MSE) within the budget constraints, suggesting that the optimal design should balance the number of clusters and observations per cluster to achieve accurate and efficient estimation.

## Methods

We employ generalized linear models (GLMs) to estimate the treatment effect:

- Normal distribution: A linear regression model using glm() with a Gaussian family.
- Poisson distribution: A log-linear model using glm() with a Poisson family.

## Performance Metrics

We evaluate the performance of each design based on the following metrics:

- Bias: $bias = E[\hat{\beta}] - \beta$
- Mean squared error (MSE): $MSE = E[(\hat{\beta} - \beta)^2]$
- Coverage probability: Proportion of confidence intervals that contain the true treatment effect.

Additionally, our estimator of $\beta$ is a unbiased estimator, so in a successful simulation, the bias should be close to zero. However, the MSE might be high due to the variance of the estimator. So in the following simulations, we will focus more on the MSE.

## Function Design

To implement the simulation study efficiently and flexibly, we developed a set of modular functions in R, each dedicated to a specific aspect of the simulation process. This modular design enhances code readability, reusability, and maintainability, allowing us to systematically explore different experimental designs under budget constraints.

First, we designed the calculate_feasible_designs function, which computes all possible combinations of the number of clusters ($G$) and the number of observations per cluster ($R$) that satisfy the given budget constraints. This function takes the total budget and the costs associated with recruiting participants—both the fixed cost per cluster($c_1$) and the variable cost per additional participant within a cluster($c_2$) as inputs. By iterating through feasible values of $G$ and calculating the corresponding maximum $R$ for each, the function generates a list of design

configurations that do not exceed the budget. This approach ensures that only viable designs are considered in the simulations, facilitating an efficient exploration of the design space.

Next, the assign_treatment function assigns clusters to either the treatment group or the control group, ensuring that both groups are represented in the study. This function is crucial because having at least one cluster in each group is necessary for estimating the treatment effect. The function begins by assigning one cluster to each group and then randomly assigns the remaining clusters, maintaining the randomness essential for unbiased treatment effect estimation. This careful assignment prevents scenarios where all clusters might inadvertently be assigned to the same group, which would invalidate the comparative analysis.

The generate_data function is responsible for simulating the clustered data based on the specified parameters and the chosen outcome distribution (normal or Poisson). It incorporates the treatment assignments from the assign_treatment function and generates cluster-level random effects to introduce between-cluster variability. For each observation within a cluster, the function computes the outcome using the appropriate statistical model: a linear model for normally distributed outcomes or a log-linear model for Poisson-distributed outcomes. This function ensures that the simulated data accurately reflect the hierarchical structure and variability inherent in clustered experimental designs, providing a realistic basis for evaluating different design configurations.

To estimate the treatment effect from the simulated data, we developed the estimate_treatment_effect function. This function fits a generalized linear model (GLM) appropriate for the specified outcome distribution and adjusts for clustering by computing cluster-robust standard errors using sandwich estimators. The use of cluster-robust standard errors is essential to obtain valid statistical inference in the presence of intra-cluster correlation. The function extracts the estimated treatment effect, its standard error, and calculates the p-value to determine statistical significance. This standardized approach to estimation allows for consistent evaluation of treatment effects across different simulated datasets and design configurations.

The run_simulation function orchestrates the simulation process by repeatedly generating data and estimating the treatment effect for a specified design configuration. It runs a predetermined number of simulation iterations (replications) to assess the variability and reliability of the treatment effect estimates for that design. By aggregating results across simulations, this function enables us to compute performance metrics such as bias, mean squared error, coverage probability, and statistical power for each design. This iterative process is critical for understanding the statistical properties of the estimators under different design scenarios.

Finally, the evaluate_design function analyzes the simulation results to compute the performance metrics for each design configuration. It calculates the bias of the treatment effect estimator by comparing the average estimated treatment effect across simulations to the true effect size used in data generation. The function also computes the mean squared error, which reflects both the variance and the bias of the estimator, and the coverage probability of the confidence intervals, indicating how often the true treatment effect is captured within

the estimated intervals. These metrics provide a comprehensive assessment of each design's effectiveness in estimating the treatment effect accurately and efficiently within the budget constraints.

Overall, this modular function design facilitates a systematic and thorough exploration of optimal experimental designs under budget constraints. By compartmentalizing each step of the simulation process into dedicated functions, we enhance the clarity and reproducibility of our simulation study, enabling us to draw meaningful conclusions about resource allocation in experimental planning. The functions work cohesively to simulate realistic clustered data, perform valid statistical analyses, and evaluate the performance of various experimental designs, thereby contributing valuable insights into the optimization of experimental design under practical constraints.

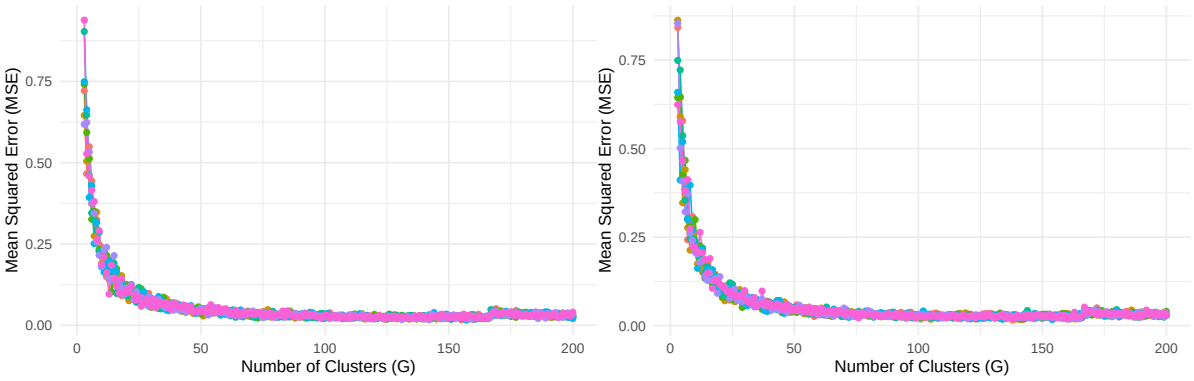
# Results

## Alpha and Beta

From the equation $\mu_{i0} = \alpha + \beta X_j$, we can boldly assume that neither of the $\alpha$ and $\beta$ will have effect on our estimands. In order to verify this, we can run the simulation with different $\alpha$ and $\beta$ values.

In Figure 1, each color represents a different $\alpha$ ($\beta$) value. We observed that in both Figure 1 (a) and Figure 1 (b), the trend of the MSE is collapsed with each other. This indicates that the $\alpha$ and $\beta$ values do not have significant effect on the estimands.

Figure 1: MSE vs Clusters for Different Alpha and Beta Values




(a) Alpha (b) Beta

Further more, we selected the optimal designs for different $\alpha$ and $\beta$ values. Table 1 shows that with different $\alpha$ and $\beta$ values, the optimal MSE is very close to each other. We did a ANOVA test on the results and find that both of the p-value for $\alpha$ and $\beta$ results are >0.9, suggesting

Table 1: Optimal Designs for Different Alpha Values and Beta Values

(a) Alpha

| alpha | OptimalMSE | G | R | Cost |
|---|---|---|---|---|
| 0.0 | 0.0568911 | 155 | 2 | 9300 |
| 0.5 | 0.0554511 | 139 | 3 | 9730 |
| 1.0 | 0.0570973 | 125 | 4 | 10000 |
| 1.5 | 0.0579654 | 143 | 2 | 8580 |
| 2.0 | 0.0575748 | 157 | 2 | 9420 |
| 2.5 | 0.0565993 | 142 | 3 | 9940 |
| 3.0 | 0.0576882 | 124 | 4 | 9920 |

(b) Beta

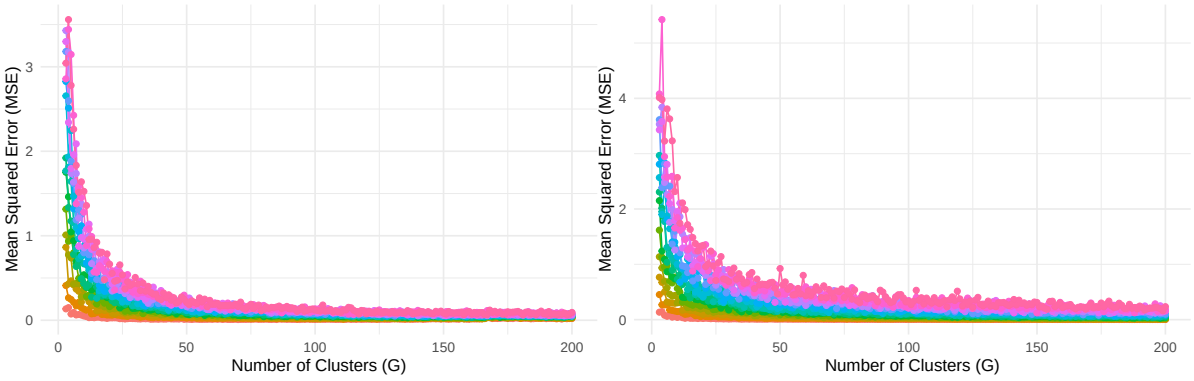
| beta | OptimalMSE | G | R | Cost |
|---|---|---|---|---|
| 0.0 | 0.0581459 | 138 | 3 | 9660 |
| 0.5 | 0.0560478 | 105 | 5 | 9450 |
| 1.0 | 0.0569062 | 141 | 3 | 9870 |
| 1.5 | 0.0582138 | 161 | 2 | 9660 |
| 2.0 | 0.0568961 | 154 | 2 | 9240 |
| 2.5 | 0.0564536 | 120 | 4 | 9600 |
| 3.0 | 0.0582648 | 160 | 2 | 9600 |

that there is no significant difference between the optimal MSE for different $\alpha$ and $\beta$ values. All of these results indicate that the $\alpha$ and $\beta$ values do not have significant effect on our ability to estimate the treatment effect.

## Gamma

$\gamma$ stands for the between-cluster variance, which is the variance of the cluster-level random effects. In the simulation, we tested different $\gamma$ values for both Normal and Poisson distributions. The results are shown in Figure 2. We observed that with the same $\gamma$ value, the MSE is more spreed out in the Poisson distribution than in the Normal distribution. And we also observed that each color in the plot stratified clearly, indicating that the $\gamma$ value has a significant effect on the estimands.

Figure 2: MSE vs Clusters for Different Gamma Values




(a) Normal (b) Poisson

An ANOVA test was conducted on the results to test if the mean MSE for differnt $\gamma$ value

Table 2: Optimal Designs for Different Gamma Values

(a) Normal

| gamma2 | OptimalMse | G | R | Cost |
|---|---|---|---|---|
| 0.1 | 0.0082524 | 70 | 10 | 9800 |
| 0.3 | 0.0119108 | 111 | 5 | 9990 |
| 0.5 | 0.0163661 | 165 | 2 | 9900 |
| 0.7 | 0.0196172 | 121 | 4 | 9680 |
| 0.9 | 0.0257904 | 136 | 3 | 9520 |
| 1.1 | 0.0329058 | 165 | 2 | 9900 |
| 1.3 | 0.0323362 | 161 | 2 | 9660 |
| 1.5 | 0.0383462 | 159 | 2 | 9540 |
| 1.7 | 0.0409794 | 138 | 3 | 9660 |
| 1.9 | 0.0440433 | 143 | 2 | 8580 |
| 2.1 | 0.0483687 | 190 | 1 | 9500 |
| 2.3 | 0.0525708 | 197 | 1 | 9850 |
| 2.5 | 0.0549993 | 151 | 2 | 9060 |
| 2.7 | 0.0565745 | 191 | 1 | 9550 |
| 2.9 | 0.0624897 | 194 | 1 | 9700 |

(b) Poisson

| gamma2 | OptimalMse | G | R | Cost |
|---|---|---|---|---|
| 0.1 | 0.0017808 | 198 | 1 | 9900 |
| 0.3 | 0.0059037 | 169 | 1 | 8450 |
| 0.5 | 0.0081995 | 195 | 1 | 9750 |
| 0.7 | 0.0164247 | 194 | 1 | 9700 |
| 0.9 | 0.0224154 | 185 | 1 | 9250 |
| 1.1 | 0.0260261 | 198 | 1 | 9900 |
| 1.3 | 0.0356490 | 195 | 1 | 9750 |
| 1.5 | 0.0469979 | 184 | 1 | 9200 |
| 1.7 | 0.0631379 | 187 | 1 | 9350 |
| 1.9 | 0.0663746 | 180 | 1 | 9000 |
| 2.1 | 0.0803046 | 174 | 1 | 8700 |
| 2.3 | 0.1118052 | 178 | 1 | 8900 |
| 2.5 | 0.1217764 | 141 | 3 | 9870 |
| 2.7 | 0.1211594 | 150 | 2 | 9000 |
| 2.9 | 0.1598825 | 199 | 1 | 9950 |

is different. The p-value in both the Poisson and Normal distribution is smaller than 0.001, suggesting that the $\gamma$ value has a significant effect on the estimands.

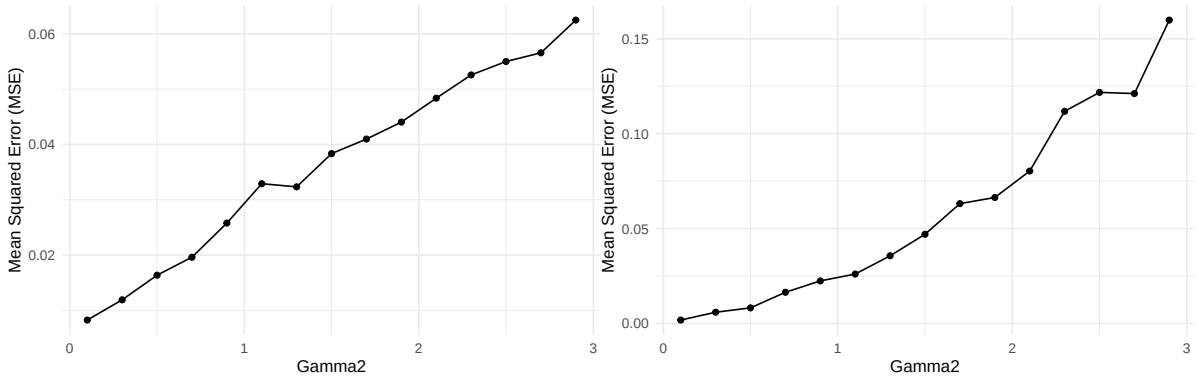
To further expore the relationship between $\gamma$ and the optimal design, we selected the optimal designs for different $\gamma$ values. Both Table 2 and Figure 3 shows that with higher $\gamma$ values, the optimal MSE is higher, indicating that the between-cluster variance has a negative effect on the estimands.

We also noticed that Poisson distribution has a higher optimal MSE and also more sensitive than the Normal distribution. This is because in the Normal distribution, we generate $\mu_i \sim N(\mu_{i0}, \gamma^2)$, however, in the Poisson distribution, we generate $log(\mu_i) \sim N(\mu_{i0}, \gamma^2)$. It is the log transformation makes the Poisson distribution more sensitive to the $\gamma$ value.

## Sigma

The $\sigma$ is the within-cluster variance, which is the variance of the individual-level random effects. In the simulation, since the $\sigma$ has nothing to do with the Poisson distribution here, we tested different $\sigma$ values only for Normal distributions. In Figure 4 (a), we noticed that the results collapsed with each other at begining and then tend to stratified for normal distribution. This indicates that the $\sigma$ value has a significant effect on the estimands in normal distribution. The

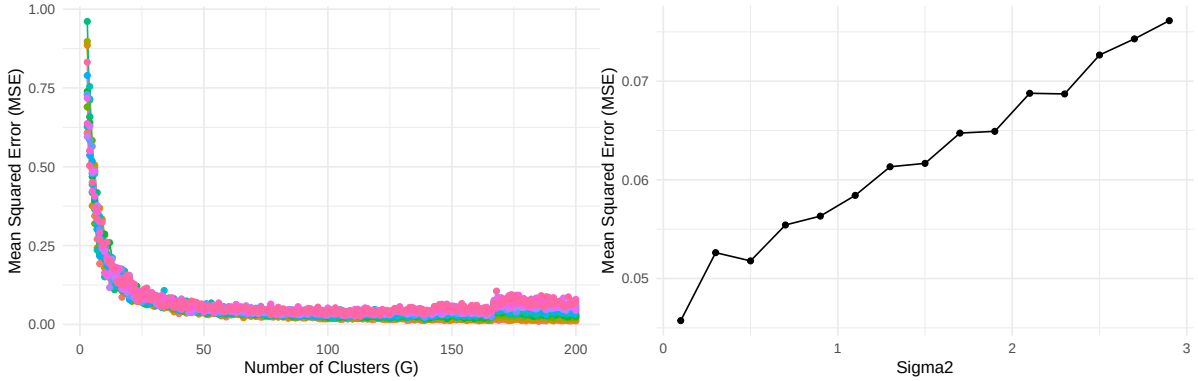
Figure 3: Optimal Designs for Different Gamma Values




(a) Normal (b) Poisson

ANOVA test further confirmed that the $\sigma$ value has a significant effect on the estimands in the Normal distribution (P-value <0.001).

Figure 4: Simulation Results for Different Sigma Values




(a) MSE vs Clusters for Different Sigma Values (b) Optimal Designs for Different Sigma Values

More specifically, we selected the optimal designs for different $\sigma$ values. Table 3 and Figure 4 (b) shows that, for normal distribution, with higher $\sigma$ values, the optimal MSE is higher, indicating that the within-cluster variance has a negative effect on the estimands in the Normal distribution.

Table 3: Optimal Designs for Different Sigma Values

| sigma2 | OptimalMSE | G | R | Cost |
|---|---|---|---|---|
| 0.1 | 0.0457286 | 185 | 1 | 9250 |
| 0.3 | 0.0526188 | 191 | 1 | 9550 |

Table 3: Optimal Designs for Different Sigma Values

| sigma2 | OptimalMSE | G | R | Cost |
|---|---|---|---|---|
| 0.5 | 0.0517844 | 164 | 2 | 9840 |
| 0.7 | 0.0554224 | 135 | 3 | 9450 |
| 0.9 | 0.0563240 | 165 | 2 | 9900 |
| 1.1 | 0.0584226 | 159 | 2 | 9540 |
| 1.3 | 0.0613256 | 139 | 3 | 9730 |
| 1.5 | 0.0616671 | 102 | 5 | 9180 |
| 1.7 | 0.0647367 | 108 | 5 | 9720 |
| 1.9 | 0.0649154 | 142 | 3 | 9940 |
| 2.1 | 0.0687715 | 139 | 3 | 9730 |
| 2.3 | 0.0687062 | 111 | 5 | 9990 |
| 2.5 | 0.0726491 | 108 | 5 | 9720 |
| 2.7 | 0.0742857 | 107 | 5 | 9630 |
| 2.9 | 0.0761328 | 113 | 4 | 9040 |

### C1/C2 Ratio

The $C1/C2$ ratio is the ratio of the cluster sample cost and the individual sample cost. In the simulation, we tested different $C1/C2$ values for both Normal and Poisson distributions. In order to make the computation more easier, we set the $C2$ value to 1, meaning that the value of $C1$ is the $C1/C2$ ratio.

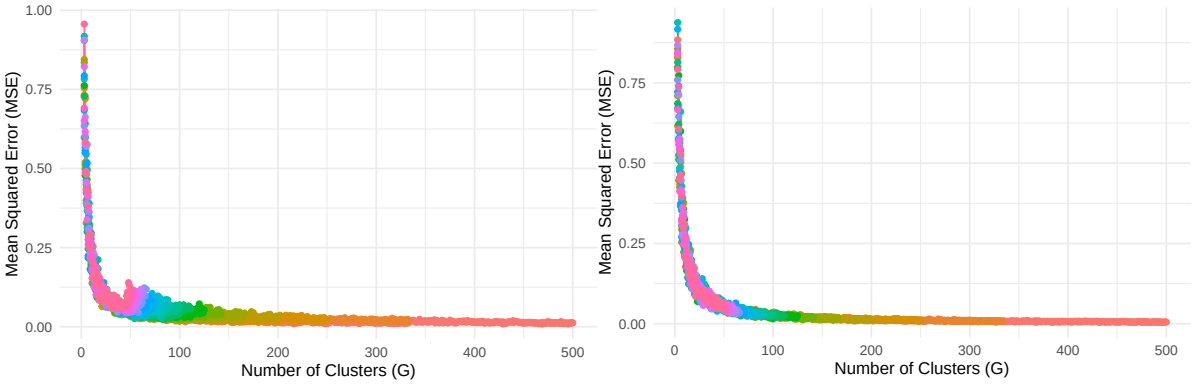
The results are shown in Figure 6. Very interestingly, we observed that each set of $C1/C2$ values has a different range of MSE. To further investigate if the simulation performance is affected by the $C1/C2$ ratio, we conducted an ANOVA test on the results. The p-value in both the Poisson and Normal distribution is smaller than 0.001, suggesting that the $C1/C2$ ratio has a significant effect on the estimands.

We also selected the optimal designs for different $C1/C2$ values. Both Table 4 and Figure 7 shows that with higher $C1/C2$ values, the optimal MSE is higher, indicating that the $C1/C2$ ratio has a negative effect on the estimands.

## Discussion

Our simulation study revealed several key insights about optimal experimental design under budget constraints for both normally distributed and Poisson-distributed outcomes. Through systematic investigation of various parameters and their effects on design efficiency, we have identified critical factors that influence optimal resource allocation in experimental research.

Figure 5: MSE vs Clusters for Different C1/C2 Ratio




(a) Normal

(b) Poisson

Table 4: Optimal Designs for Different C1/C2 Ratio

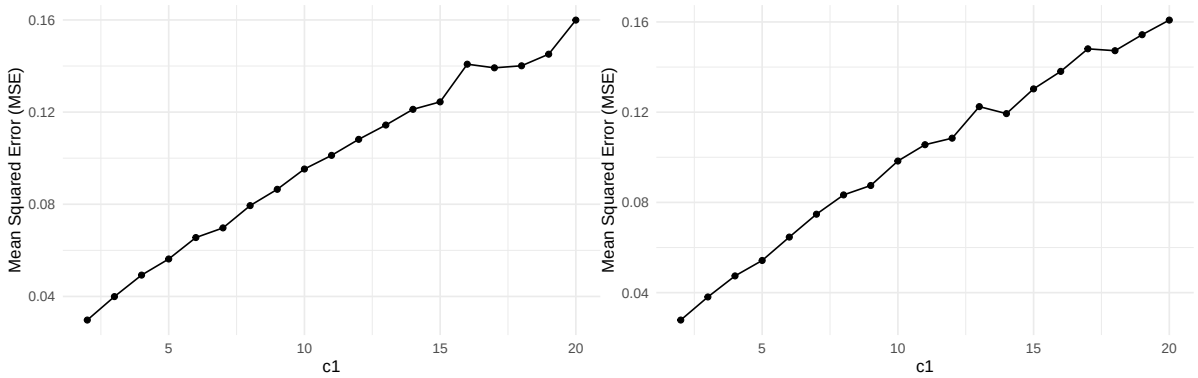
(a) Normal

| c1 | OptimalMSE | G | R | Cost |
|---|---|---|---|---|
| 2 | 0.0297566 | 429 | 1 | 858 |
| 3 | 0.0399331 | 214 | 2 | 856 |
| 4 | 0.0492680 | 180 | 2 | 900 |
| 5 | 0.0562573 | 162 | 2 | 972 |
| 6 | 0.0655591 | 85 | 6 | 935 |
| 7 | 0.0697480 | 99 | 4 | 990 |
| 8 | 0.0794216 | 83 | 5 | 996 |
| 9 | 0.0864997 | 91 | 2 | 910 |
| 10 | 0.0953008 | 82 | 3 | 984 |
| 11 | 0.1012274 | 54 | 8 | 972 |
| 12 | 0.1081789 | 58 | 6 | 986 |
| 13 | 0.1143881 | 54 | 6 | 972 |
| 14 | 0.1212100 | 49 | 7 | 980 |
| 15 | 0.1244390 | 58 | 3 | 986 |
| 16 | 0.1407954 | 48 | 5 | 960 |
| 17 | 0.1392162 | 45 | 6 | 990 |
| 18 | 0.1400984 | 49 | 3 | 980 |
| 19 | 0.1451432 | 40 | 7 | 1000 |
| 20 | 0.1599335 | 37 | 8 | 999 |

(b) Poisson

| c1 | OptimalMSE | G | R | Cost |
|---|---|---|---|---|
| 2 | 0.0278507 | 497 | 1 | 994 |
| 3 | 0.0380881 | 326 | 1 | 978 |
| 4 | 0.0474146 | 248 | 1 | 992 |
| 5 | 0.0542624 | 197 | 1 | 985 |
| 6 | 0.0646235 | 161 | 1 | 966 |
| 7 | 0.0747754 | 130 | 1 | 910 |
| 8 | 0.0833207 | 100 | 3 | 1000 |
| 9 | 0.0874934 | 102 | 1 | 918 |
| 10 | 0.0983416 | 95 | 1 | 950 |
| 11 | 0.1055769 | 88 | 1 | 968 |
| 12 | 0.1084739 | 81 | 1 | 972 |
| 13 | 0.1224630 | 74 | 1 | 962 |
| 14 | 0.1193571 | 68 | 1 | 952 |
| 15 | 0.1303207 | 63 | 1 | 945 |
| 16 | 0.1380758 | 60 | 1 | 960 |
| 17 | 0.1480772 | 56 | 1 | 952 |
| 18 | 0.1472453 | 53 | 1 | 954 |
| 19 | 0.1543376 | 52 | 1 | 988 |
| 20 | 0.1608175 | 47 | 2 | 987 |

Figure 6: Optimal Designs for Different C1/C2 Ratio




(a) Normal (b) Poisson

A notable finding is that neither the intercept ($\alpha$) nor the treatment effect size ($\beta$) significantly influenced the optimal design configurations. The ANOVA tests for both parameters yielded p-values $> 0.9$, indicating that the optimal allocation of resources between clusters and observations within clusters remains consistent regardless of the expected baseline outcomes or anticipated treatment effects. This finding simplifies the design process, as researchers can focus on other parameters when determining optimal designs.

The between-cluster variance ($\gamma^2$) emerged as a crucial factor affecting design efficiency. For both normal and Poisson distributions, higher $\gamma^2$ values led to increased Mean Squared Error (MSE), indicating reduced precision in treatment effect estimation. The effect was notably more pronounced in the Poisson distribution, likely due to the log-link function amplifying the variance structure. As between-cluster variance increased, the optimal design generally favored more clusters with fewer observations per cluster, aligning with statistical theory that between-cluster variance can only be addressed by increasing the number of clusters.

For normally distributed outcomes, the within-cluster variance ($\sigma^2$) showed a different pattern of influence. As $\sigma^2$ increased, the optimal designs tended to favor more observations per cluster rather than additional clusters. This finding reflects the ability to improve precision by increasing within-cluster sample sizes when within-cluster variance is high. The absence of $\sigma^2$ in the Poisson model highlights a fundamental difference between the two distributions in terms of variance structure.

Across our simulations, we observed a consistent pattern where optimal designs generally favored a larger number of clusters with relatively few observations per cluster. This tendency was particularly evident in scenarios with high between-cluster variance or Poisson-distributed outcomes. This pattern likely emerges because increasing the number of clusters provides more independent units of randomization, which is crucial for estimating treatment effects in cluster-randomized trials. Additionally, with the constraint of a fixed budget, the trade-off between

cluster size and number of clusters often tips in favor of more clusters because they provide more information about population-level effects than additional within-cluster observations.

The ratio of cluster-level to individual-level costs proved to be a critical determinant of optimal design. Higher $c_1/c_2$ ratios led to designs with fewer, larger clusters, reflecting the need to balance statistical efficiency with economic constraints. This relationship was observed in both distribution types, though with different patterns. For normal distributions, the optimal number of clusters decreased more gradually with increasing cost ratios, while in Poisson distributions, the transition to fewer clusters was more abrupt.

Several limitations of our study suggest directions for future research, including the assumption of fixed costs per cluster and individual, the focus on balanced designs, and the absence of considerations for missing data and dropout. Despite these limitations, our findings provide valuable guidance for researchers designing cluster-randomized trials under budget constraints, particularly in understanding how various parameters influence the optimal balance between number of clusters and cluster size.

## Code Appendix

```
knitr::opts_chunk$set(echo = FALSE)
knitr::opts_chunk$set(message = FALSE)
knitr::opts_chunk$set(warning = FALSE)

library(ggplot2)
library(dplyr)
library(tidyr)
library(knitr)
library(kableExtra)
library(gridExtra)
library(sandwich)
library(lmtest)
library(lme4)
library(lmerTest)
library(gtsummary)
# Function to calculate feasible designs
calculate_feasible_designs <- function(budget, c1, c2) {
  feasible_designs <- list()
  max_clusters <- floor(budget / c1)

  for (G in 2:max_clusters) {  # At least 2 clusters needed for comparison
    max_R <- floor((budget - G * c1) / (G * c2)) + 1
    if (max_R >= 1) {
      feasible_designs[[length(feasible_designs) + 1]] <- list(G = G, R = max_R)
    }
  }

  return(feasible_designs)
}

# Function to assign treatments ensuring both groups are represented
assign_treatment <- function(G) {
  if (G < 2) {
    stop("The number of clusters (G) must be at least 2.")
  }

  # Start with one cluster in each group
  Treatment <- c(0, 1)
```

```r
  if (G > 2) {
    # Assign remaining clusters randomly
    remaining_treatments <- sample(0:1, G - 2, replace = TRUE)
    Treatment <- c(Treatment, remaining_treatments)
  }

  # Shuffle the treatments to randomize cluster assignments
  Treatment <- sample(Treatment)

  return(Treatment)
}

# Function to generate data in long format
generate_data <- function(G, R, alpha, beta, gamma2, sigma2, distribution = 'normal') {
  Treatment <- assign_treatment(G)

  # Generate cluster-level random effects
  if (distribution == 'normal') {
    epsilon <- rnorm(G, mean = 0, sd = sqrt(gamma2))
  } else if (distribution == 'poisson') {
    epsilon <- rnorm(G, mean = 0, sd = sqrt(gamma2))
  }

  # Generate cluster means
  cluster_means <- alpha + beta * Treatment + epsilon

  # Create a data frame with all observations
  data_long <- data.frame(
    Cluster = rep(1:G, each = R),
    Treatment = rep(Treatment, each = R),
    Y = NA
  )

  # Generate observations based on distribution
  if (distribution == 'normal') {
    data_long$Y <- rnorm(n = G * R, mean = rep(cluster_means, each = R), sd = sqrt(sigma2))
  } else if (distribution == 'poisson') {
    mu <- exp(rep(cluster_means, each = R))
    data_long$Y <- rpois(n = G * R, lambda = mu)
  }

  return(data_long)
```

```r
}


# Function to estimate treatment effect using mixed-effects model
estimate_treatment_effect <- function(data_long, distribution = 'normal') {

  data_long$Cluster <- as.factor(data_long$Cluster)

  if (distribution == 'normal') {
    model <- glm(Y ~ Treatment, data = data_long, family = gaussian())
  } else if (distribution == 'poisson') {
    model <- glm(Y ~ Treatment, data = data_long, family = poisson())
  }

  cluster_vcov <- vcovCL(model, cluster = data_long$Cluster)
  beta_hat <- coef(model)["Treatment"]
  se <- sqrt(cluster_vcov["Treatment", "Treatment"])

  z_value <- beta_hat / se
  p_value <- 2 * (1 - pnorm(abs(z_value)))

  power <- p_value < 0.05

  return(list(
    estimate = beta_hat,
    se = se,
    p_value = p_value,
    power = power
  ))
}

# Function to run simulation for a design
run_simulation <- function(G, R, alpha, beta, gamma2, sigma2, c1, c2, n_sims = 1000, distribu
  results <- data.frame(
    estimate = numeric(n_sims),
    se = numeric(n_sims),
    p_value = numeric(n_sims),
    power = logical(n_sims)
  )

  for (sim in 1:n_sims) {
    data_long <- generate_data(G, R, alpha, beta, gamma2, sigma2, distribution)
```

```
    estimates <- estimate_treatment_effect(data_long, distribution)

    results$estimate[sim] <- estimates$estimate
    results$se[sim] <- estimates$se
    results$p_value[sim] <- estimates$p_value
    results$power[sim] <- estimates$power
  }

  total_cost <- G * (c1 + (R - 1) * c2)

  return(list(
    results = results,
    G = G,
    R = R,
    total_cost = total_cost
  ))
}


# Function to evaluate design performance
evaluate_design <- function(simulation_results, beta) {
  results <- simulation_results$results
  G <- simulation_results$G
  R <- simulation_results$R
  total_cost <- simulation_results$total_cost

  bias <- mean(results$estimate, na.rm = TRUE) - beta
  mse <- mean((results$estimate - beta)^2, na.rm = TRUE)
  coverage <- mean(abs(results$estimate - beta) <= 1.96 * results$se, na.rm = TRUE)

  return(list(
    G = G,
    R = R,
    total_cost = total_cost,
    bias = bias,
    mse = mse,
    coverage = coverage
  ))
}
budget <- 10000
c1 <- 50
c2 <- 10
```

```r
beta <- 0.5
gamma2 <- 0.5
sigma2 <- 1
n_sims <- 100

alpha_values <- seq(0, 3, by = 0.5)

feasible_designs <- calculate_feasible_designs(budget, c1, c2)
design_performance_normal_alpha <- data.frame()

for (alpha in alpha_values) {
  for (design in feasible_designs) {
    G <- design$G
    R <- design$R

    if (G < 3) next

    sim_results <- run_simulation(G, R, alpha, beta, gamma2, sigma2, c1, c2, n_sims)
    performance <- evaluate_design(sim_results, beta)

    design_performance_normal_alpha <- rbind(design_performance_normal_alpha, data.frame(
      G = performance$G,
      R = performance$R,
      total_cost = performance$total_cost,
      alpha = alpha,
      bias = performance$bias,
      mse = performance$mse,
      coverage = performance$coverage
    ))
  }
}

write.csv(design_performance_normal_alpha, "../Results/design_performance_normal_alpha.csv")

budget <- 10000
c1 <- 50
c2 <- 10
alpha <- 5
gamma2 <- 0.5
sigma2 <- 1
n_sims <- 100
```

```r
beta_values <- seq(0, 3, by = 0.5)

feasible_designs <- calculate_feasible_designs(budget, c1, c2)
design_performance_normal_beta <- data.frame()

for (beta in beta_values) {
  for (design in feasible_designs) {
    G <- design$G
    R <- design$R

    if (G < 3) next

    sim_results <- run_simulation(G, R, alpha, beta, gamma2, sigma2, c1, c2, n_sims)
    performance <- evaluate_design(sim_results, beta)

    design_performance_normal_beta <- rbind(design_performance_normal_beta, data.frame(
      G = performance$G,
      R = performance$R,
      total_cost = performance$total_cost,
      beta = beta,
      bias = performance$bias,
      mse = performance$mse,
      coverage = performance$coverage
    ))
  }
}

write.csv(design_performance_normal_beta, "../Results/design_performance_normal_beta.csv")

# read results
design_performance_normal_alpha <- read.csv("../Results/design_performance_normal_alpha.csv")

normal_alpha_results_plot <- ggplot(design_performance_normal_alpha, aes(x = G, y = mse, colo
  geom_line() +
  geom_point() +
  labs(x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  theme(legend.position="none")

normal_alpha_results_table <- tbl_summary(design_performance_normal_alpha %>% select(c(bias,
  add_p()
```

```r
normal_alpha_results_optimal_table <- kable(design_performance_normal_alpha %>%
  group_by(alpha) %>%
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota

normal_alpha_results_optimal_plot <- design_performance_normal_alpha %>%
  group_by(alpha) %>%
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota
  ggplot(aes(x = alpha, y = OptimalMSE)) +
  geom_line() +
  geom_point() +
  labs(x = "Alpha",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()
# read results
design_performance_normal_beta <- read.csv("../Results/design_performance_normal_beta.csv")

normal_beta_results_plot <- ggplot(design_performance_normal_beta, aes(x = G, y = mse, color
  geom_line() +
  geom_point() +
  labs(x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  theme(legend.position="none")

normal_beta_results_table <- tbl_summary(design_performance_normal_beta %>% select(c(bias, ms
  add_p()

normal_beta_results_optimal_table <- kable(design_performance_normal_beta %>%
  group_by(beta) %>%
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota

normal_beta_results_optimal_plot <- design_performance_normal_beta %>%
  group_by(beta) %>%
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota
  ggplot(aes(x = beta, y = OptimalMSE)) +
  geom_line() +
  geom_point() +
  labs(x = "Beta",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()

normal_alpha_results_plot
```

```
normal_beta_results_plot
normal_alpha_results_optimal_table
normal_beta_results_optimal_table
normal_alpha_results_table
normal_beta_results_table
budget <- 10000
c1 <- 50
c2 <- 10
alpha <- 5
beta <- 0.5
sigma2 <- 1
n_sims <- 100

gamma2_values <- seq(0.1, 3, by = 0.2)

feasible_designs <- calculate_feasible_designs(budget, c1, c2)
design_performance_normal_gamma2 <- data.frame()

for (gamma2 in gamma2_values) {
  for (design in feasible_designs) {
    G <- design$G
    R <- design$R

    if (G < 3) next

    sim_results <- run_simulation(G, R, alpha, beta, gamma2, sigma2, c1, c2, n_sims)
    performance <- evaluate_design(sim_results, beta)

    design_performance_normal_gamma2 <- rbind(design_performance_normal_gamma2, data.frame(
      G = performance$G,
      R = performance$R,
      total_cost = performance$total_cost,
      gamma2 = gamma2,
      bias = performance$bias,
      mse = performance$mse,
      coverage = performance$coverage
    ))
  }
}

write.csv(design_performance_normal_gamma2, "../Results/design_performance_normal_gamma2.csv"
```

```r
budget <- 10000
c1 <- 50
c2 <- 10
alpha <- 5
beta <- 0.5
sigma2 <- 1
n_sims <- 100

gamma2_values <- seq(0.1, 3, by = 0.2)

feasible_designs <- calculate_feasible_designs(budget, c1, c2)
design_performance_poisson_gamma2 <- data.frame()

for (gamma2 in gamma2_values) {
  for (design in feasible_designs) {
    G <- design$G
    R <- design$R

    if (G < 3) next

    sim_results <- run_simulation(G, R, alpha, beta, gamma2, sigma2, c1, c2, n_sims, distribu
    performance <- evaluate_design(sim_results, beta)

    design_performance_poisson_gamma2  <- rbind(design_performance_poisson_gamma2, data.frame
      G = performance$G,
      R = performance$R,
      total_cost = performance$total_cost,
      gamma2 = gamma2,
      bias = performance$bias,
      mse = performance$mse,
      coverage = performance$coverage
    ))
  }
}

write.csv(design_performance_poisson_gamma2, "../Results/design_performance_poisson_gamma2.cs

# read results
design_performance_poisson_gamma2 <- read.csv("../Results/design_performance_poisson_gamma2.c

poisson_gamma_results_plot <- ggplot(design_performance_poisson_gamma2, aes(x = G, y = mse, c
  geom_line() +
```

```
  geom_point() +
  labs(x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  theme(legend.position="none")

poisson_gamma_results_table <- tbl_summary(design_performance_poisson_gamma2 %>% select(c(bia
  add_p()

poisson_gamma_results_optimal_table <- kable(design_performance_poisson_gamma2 %>%
  group_by(gamma2) %>%
  summarize(OptimalMse = min(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = total

poisson_gamma_results_optimal_plot <- design_performance_poisson_gamma2 %>%
  group_by(gamma2) %>%
  summarize(OptimalMse = min(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = total
  ggplot(aes(x = gamma2, y = OptimalMse)) +
  geom_line() +
  geom_point() +
  labs(x = "Gamma2",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()
# read results
design_performance_normal_gamma2 <- read.csv("../Results/design_performance_normal_gamma2.csv

normal_gamma_results_plot <- ggplot(design_performance_normal_gamma2, aes(x = G, y = mse, col
  geom_line() +
  geom_point() +
  labs(x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  theme(legend.position="none")

normal_gamma_results_table <- tbl_summary(design_performance_normal_gamma2 %>% select(c(bias
  add_p()

normal_gamma_results_optimal_table <-kable(design_performance_normal_gamma2 %>%
  group_by(gamma2) %>%
  summarize(OptimalMse = min(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = total

normal_gamma_results_optimal_plot <- design_performance_normal_gamma2 %>%
  group_by(gamma2) %>%
```

```
  summarize(OptimalMse = min(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = total
  ggplot(aes(x = gamma2, y = OptimalMse)) +
  geom_line() +
  geom_point() +
  labs(x = "Gamma2",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()

normal_gamma_results_plot
poisson_gamma_results_plot
poisson_gamma_results_table
normal_gamma_results_table
normal_gamma_results_optimal_table
poisson_gamma_results_optimal_table
normal_gamma_results_optimal_plot
poisson_gamma_results_optimal_plot
budget <- 10000
c1 <- 50
c2 <- 10
alpha <- 5
beta <- 0.5
gamma2 <- 0.5
n_sims <- 100

sigma2_values <- seq(0.1, 3, by = 0.2)

feasible_designs <- calculate_feasible_designs(budget, c1, c2)
design_performance_normal_sigma2 <- data.frame()

for (sigma2 in sigma2_values) {
  for (design in feasible_designs) {
    G <- design$G
    R <- design$R

    if (G < 3) next

    sim_results <- run_simulation(G, R, alpha, beta, gamma2, sigma2, c1, c2, n_sims)
    performance <- evaluate_design(sim_results, beta)

    design_performance_normal_sigma2 <- rbind(design_performance_normal_sigma2, data.frame(
      G = performance$G,
      R = performance$R,
```

```
      total_cost = performance$total_cost,
      sigma2 = sigma2,
      bias = performance$bias,
      mse = performance$mse,
      coverage = performance$coverage
    ))
  }
}

write.csv(design_performance_normal_sigma2, "../Results/design_performance_normal_sigma2.csv"

# read results
design_performance_normal_sigma2 <- read.csv("../Results/design_performance_normal_sigma2.csv

normal_sigma_results_plot <- ggplot(design_performance_normal_sigma2, aes(x = G, y = mse, col
  geom_line() +
  geom_point() +
  labs(x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  theme(legend.position="none")

normal_sigma_results_table <- tbl_summary(design_performance_normal_sigma2 %>% select(c(bias
  add_p()

normal_sigma_results_optimal_table <- kable(design_performance_normal_sigma2 %>%
  group_by(sigma2) %>%
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota

normal_sigma_results_optimal_plot <- design_performance_normal_sigma2 %>%
  group_by(sigma2) %>%
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota
  ggplot(aes(x = sigma2, y = OptimalMSE)) +
  geom_line() +
  geom_point() +
  labs(x = "Sigma2",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()
normal_sigma_results_plot
normal_sigma_results_optimal_plot
normal_sigma_results_table
normal_sigma_results_optimal_table
```

```r
budget <- 1000
alpha <- 5
beta <- 0.5
gamma2 <- 0.5
sigma2 <- 1
n_sims <- 100
c2 <- 1

c1_values <- seq(2, 20, by = 1)

design_performance_normal_c1_c2 <- data.frame()

for (c1 in c1_values) {
  feasible_designs <- calculate_feasible_designs(budget, c1, c2)

  for (design in feasible_designs) {
    G <- design$G
    R <- design$R

    if (G < 3) next

    sim_results <- run_simulation(G, R, alpha, beta, gamma2, sigma2, c1, c2, n_sims)
    performance <- evaluate_design(sim_results, beta)

    design_performance_normal_c1_c2 <- rbind(design_performance_normal_c1_c2, data.frame(
      G = performance$G,
      R = performance$R,
      total_cost = performance$total_cost,
      c1 = c1,
      c2 = c2,
      bias = performance$bias,
      mse = performance$mse,
      coverage = performance$coverage
    ))
  }
}

write.csv(design_performance_normal_c1_c2, "../Results/design_performance_normal_c1_c2.csv")

budget <- 1000
alpha <- 5
beta <- 0.5
```

```r
gamma2 <- 0.5
sigma2 <- 1
n_sims <- 100
c2 <- 1

c1_values <- seq(2, 20, by = 1)

design_performance_poisson_c1_c2 <- data.frame()

for (c1 in c1_values) {
  feasible_designs <- calculate_feasible_designs(budget, c1, c2)

  for (design in feasible_designs) {
    G <- design$G
    R <- design$R

    if (G < 3) next

    sim_results <- run_simulation(G, R, alpha, beta, gamma2, sigma2, c1, c2, n_sims, distribu
    performance <- evaluate_design(sim_results, beta)

    design_performance_poisson_c1_c2 <- rbind(design_performance_poisson_c1_c2, data.frame(
      G = performance$G,
      R = performance$R,
      total_cost = performance$total_cost,
      c1 = c1,
      c2 = c2,
      bias = performance$bias,
      mse = performance$mse,
      coverage = performance$coverage
    ))
  }
}

write.csv(design_performance_poisson_c1_c2, "../Results/design_performance_poisson_c1_c2.csv"

# read results
design_performance_poisson_c1_c2 <- read.csv("../Results/design_performance_poisson_c1_c2.csv

poisson_ratio_results_plot <- ggplot(design_performance_poisson_c1_c2, aes(x = G, y = mse, co
  geom_line() +
  geom_point() +
```

```
  labs(x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  theme(legend.position="none")

poisson_ratio_results_table <- tbl_summary(design_performance_poisson_c1_c2 %>% select(c(bias
  add_p()

poisson_ratio_results_optimal_table <- kable(design_performance_poisson_c1_c2 %>%
  group_by(c1) %>%
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota

poisson_ratio_results_optimal_plot <- design_performance_poisson_c1_c2 %>%
  group_by(c1) %>%
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota
  ggplot(aes(x = c1, y = OptimalMSE)) +
  geom_line() +
  geom_point() +
  labs(x = "c1",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()

# read results
design_performance_normal_c1_c2 <- read.csv("../Results/design_performance_normal_c1_c2.csv")

normal_ratio_results_plot <- ggplot(design_performance_normal_c1_c2, aes(x = G, y = mse, colo
  geom_line() +
  geom_point() +
  labs(x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  theme(legend.position="none")

normal_ratio_results_table <- tbl_summary(design_performance_normal_c1_c2 %>% select(c(bias,
  add_p()

normal_ratio_results_optimal_table <- kable(design_performance_normal_c1_c2 %>%
  group_by(c1) %>%
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota

normal_ratio_results_optimal_plot <- design_performance_normal_c1_c2 %>%
  group_by(c1) %>%
```

```
  summarize(OptimalMSE = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], Cost = tota
  ggplot(aes(x = c1, y = OptimalMSE)) +
  geom_line() +
  geom_point() +
  labs(x = "c1",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()

normal_ratio_results_plot
poisson_ratio_results_plot
normal_ratio_results_table
poisson_ratio_results_table
normal_ratio_results_optimal_table
poisson_ratio_results_optimal_table
normal_ratio_results_optimal_plot
poisson_ratio_results_optimal_plot
```