Simulation Studies on Optimal Experimental Design under Budget Constraints

William Qian

2024-12-01

Abstract

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 β, 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}:

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

Poisson Distribution

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

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

To be more detailed, γ^2 can be understood as the between-cluster variance, σ^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 β, which represents the difference in the outcome between the treatment and control groups. We aim to estimate β 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 β 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.

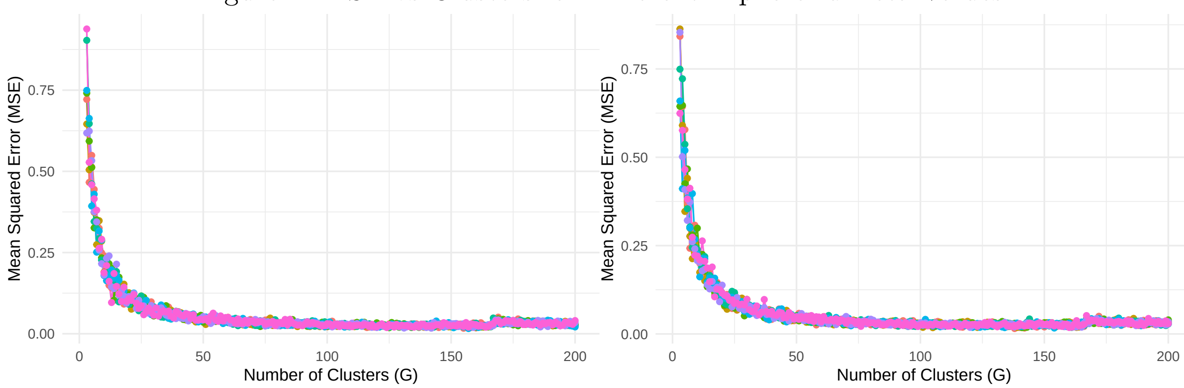
Results

Alpha and Beta

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

In Figure 1, each color represents a different α (β) 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 α and β 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 α and β values. Table 1 shows that with different α and β 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 α and β results are >0.9, suggesting that there is no significant difference between the optimal MSE for different α and β values. All of these results indicate that the α and β values do not have significant effect on our ability to estimate the treatment effect.

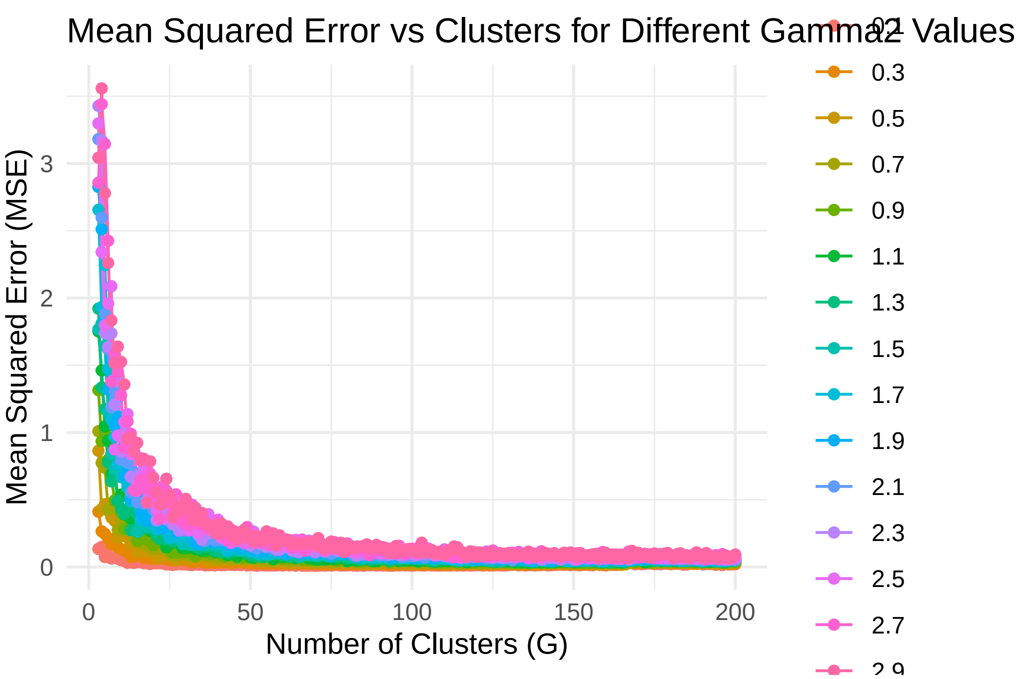
Table 1: Optimal Designs for Different Alpha Values and Beta Values

(a) Alpha

alpha	mean_mse	G	R	total_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	mean_mse	G	R	total_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

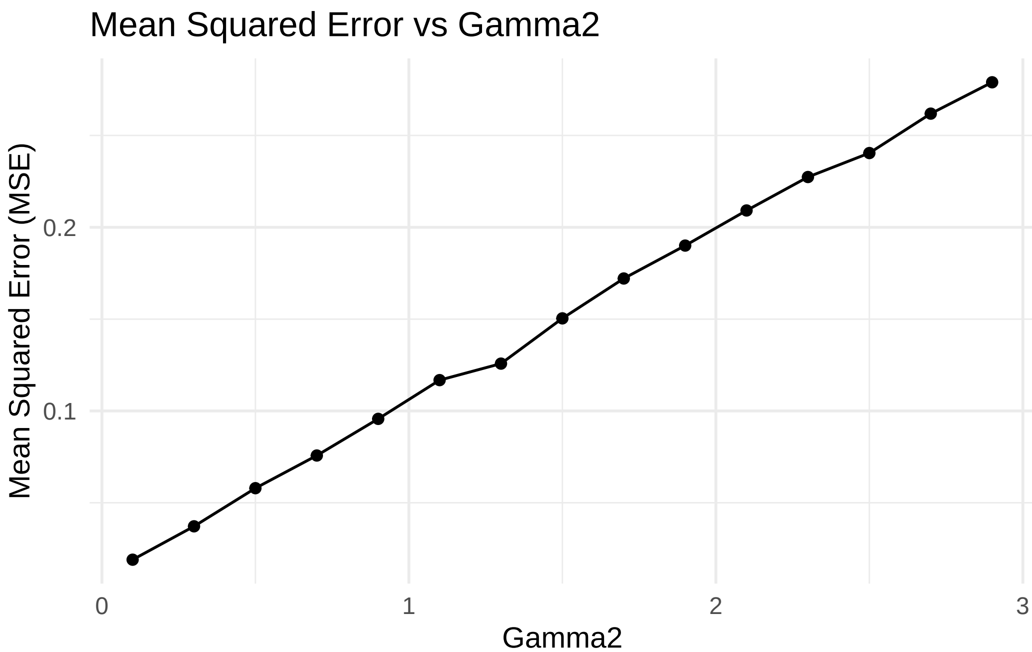


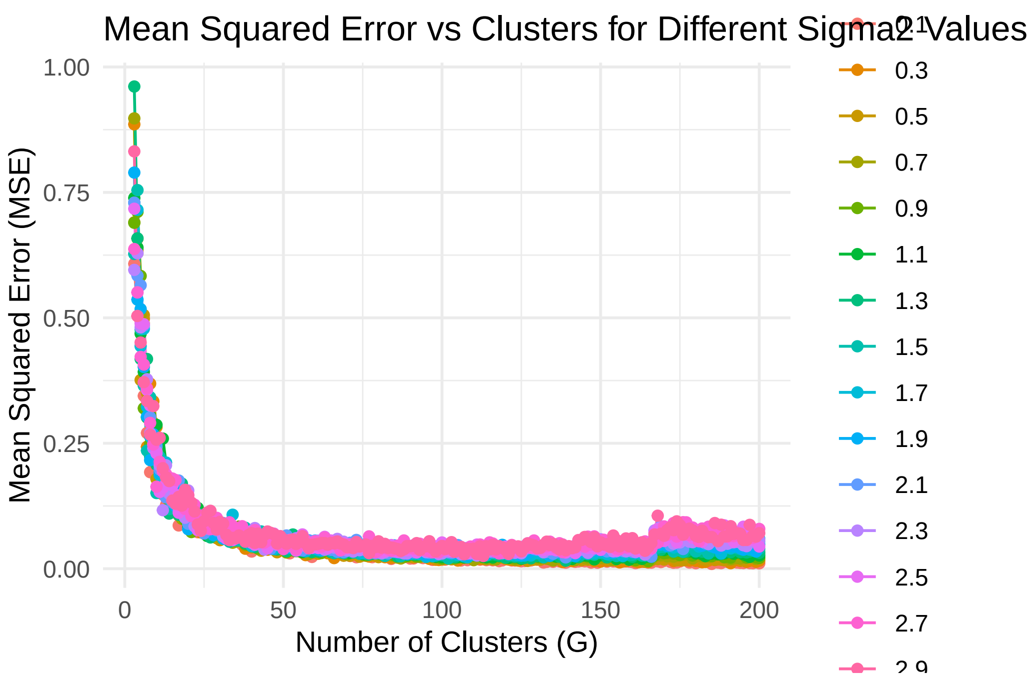
Characteristic	**0.1**, N = 198	**0.3**, N = 198	**0.5**, N = 198	**0.7**, N = 198	**0.9**, N = 198	**1.1**, N = 198	**1.3**, N = 198	**1.5**, N = 198	**1.7**, N = 198	**1.9**, N = 198	**2.1**, N = 198	**2.3**, N = 198	**2.5**, N = 198	**2.7**, N = 198	**2.9**, N = 198	**p-value**
bias	0.00 (-0.01, 0.01)	0.00 (-0.01, 0.01)	0.00 (-0.01, 0.01)	0.00 (-0.01, 0.01)	0.00 (-0.02, 0.01)	0.00 (-0.02, 0.02)	0.00 (-0.02, 0.02)	0.00 (-0.02, 0.02)	0.01 (-0.02, 0.02)	0.00 (-0.02, 0.01)	0.00 (-0.02, 0.02)	0.00 (-0.02, 0.03)	0.00 (-0.03, 0.02)	0.00 (-0.02, 0.02)	0.00 (-0.03, 0.03)	0.2

Characteristic	**0.1**, N = 198	**0.3**, N = 198	**0.5**, N = 198	**0.7**, N = 198	**0.9**, N = 198	**1.1**, N = 198	**1.3**, N = 198	**1.5**, N = 198	**1.7**, N = 198	**1.9**, N = 198	**2.1**, N = 198	**2.3**, N = 198	**2.5**, N = 198	**2.7**, N = 198	**2.9**, N = 198	**p-value**
mse	0.01 (0.01, 0.02)	0.02 (0.02, 0.03)	0.03 (0.03, 0.05)	0.04 (0.03, 0.06)	0.05 (0.04, 0.08)	0.05 (0.04, 0.09)	0.06 (0.05, 0.11)	0.07 (0.06, 0.13)	0.08 (0.06, 0.13)	0.08 (0.07, 0.15)	0.09 (0.07, 0.18)	0.10 (0.08, 0.19)	0.11 (0.08, 0.20)	0.12 (0.09, 0.20)	0.12 (0.09, 0.22)	<0.001
coverage	0.940 (0.920, 0.960)	0.940 (0.920, 0.960)	0.940 (0.920, 0.950)	0.940 (0.920, 0.960)	0.940 (0.930, 0.960)	0.940 (0.920, 0.960)	0.940 (0.930, 0.950)	0.940 (0.920, 0.960)	0.940 (0.920, 0.960)	0.940 (0.930, 0.960)	0.940 (0.920, 0.960)	0.940 (0.930, 0.960)	0.940 (0.920, 0.960)	0.950 (0.930, 0.960)	0.950 (0.930, 0.960)	0.8

Table 3: Optimal Designs for Different Gamma2 Values

gamma2	mean_mse	G	R	total_cost
0.1	0.0190063	70	10	9800
0.3	0.0371850	111	5	9990
0.5	0.0579494	165	2	9900
0.7	0.0757360	121	4	9680
0.9	0.0956974	136	3	9520
1.1	0.1167840	165	2	9900
1.3	0.1257796	161	2	9660
1.5	0.1504238	159	2	9540
1.7	0.1721281	138	3	9660
1.9	0.1900457	143	2	8580
2.1	0.2091637	190	1	9500
2.3	0.2273801	197	1	9850
2.5	0.2404325	151	2	9060
2.7	0.2618897	191	1	9550
2.9	0.2789687	194	1	9700

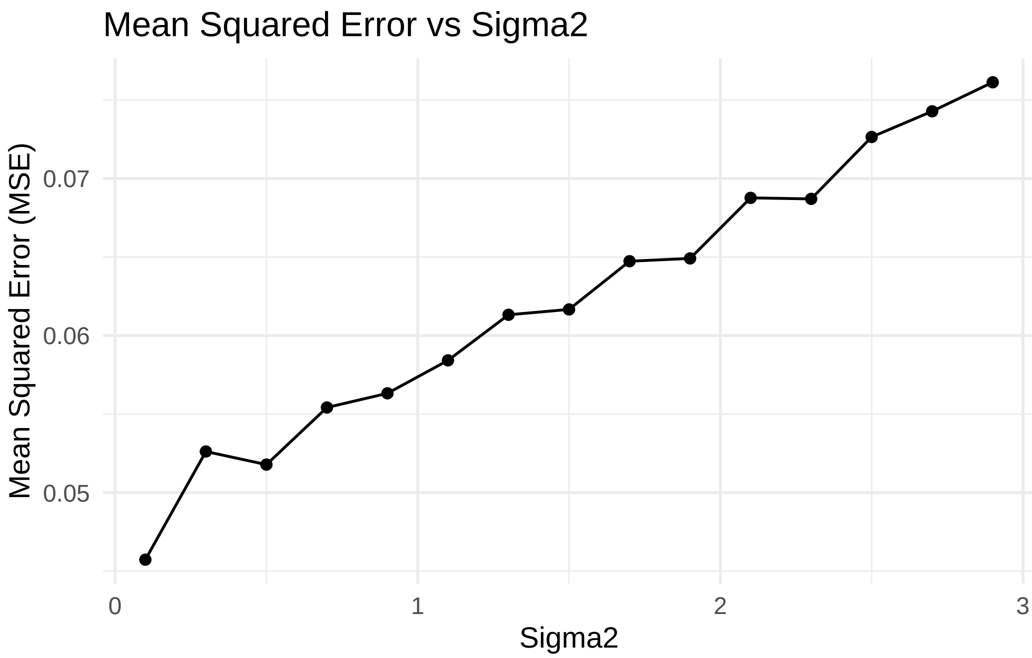

Mean Squared Error vs Gamma2
Mean Squared Error (MSE)
0.1
0.2
0
1
2
3
Gamma2

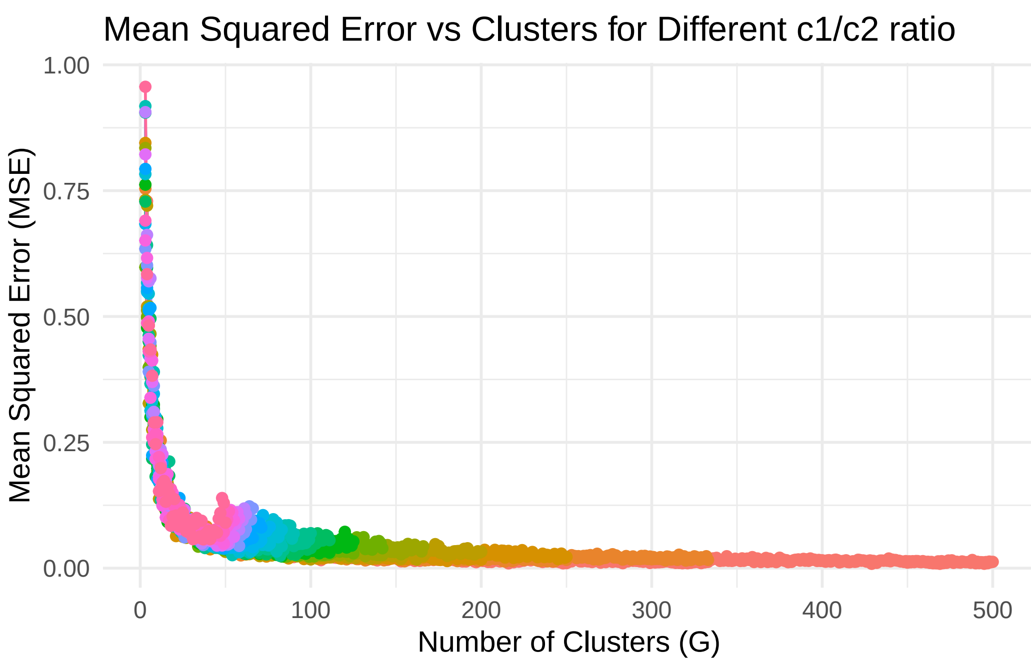

Mean Squared Error vs Clusters for Different Sigma2 Values
Mean Squared Error (MSE)
0.00
0.25
0.50
0.75
1.00
0
50
100
150
200
Number of Clusters (G)
0.1
0.3
0.5
0.7
0.9
1.1
1.3
1.5
1.7
1.9
2.1
2.3
2.5
2.7

Characteristic	**0.1**, N = 198	**0.3**, N = 198	**0.5**, N = 198	**0.7**, N = 198	**0.9**, N = 198	**1.1**, N = 198	**1.3**, N = 198	**1.5**, N = 198	**1.7**, N = 198	**1.9**, N = 198	**2.1**, N = 198	**2.3**, N = 198	**2.5**, N = 198	**2.7**, N = 198	**2.9**, N = 198	**p-value**
bias	0.002 (-0.010, 0.013)	-0.001 (-0.011, 0.012)	0.001 (-0.014, 0.011)	0.002 (-0.009, 0.013)	-0.001 (-0.013, 0.013)	-0.003 (-0.013, 0.013)	0.003 (-0.012, 0.014)	0.001 (-0.011, 0.017)	-0.001 (-0.015, 0.013)	0.001 (-0.015, 0.015)	0.000 (-0.019, 0.015)	0.001 (-0.017, 0.016)	-0.003 (-0.021, 0.015)	-0.002 (-0.017, 0.016)	0.000 (-0.014, 0.017)	0.7
mse	0.02 (0.01, 0.04)	0.02 (0.02, 0.04)	0.03 (0.02, 0.04)	0.03 (0.02, 0.04)	0.03 (0.03, 0.04)	0.03 (0.03, 0.05)	0.04 (0.03, 0.05)	0.04 (0.03, 0.05)	0.04 (0.03, 0.05)	0.04 (0.04, 0.06)	0.05 (0.04, 0.06)	0.05 (0.04, 0.06)	0.05 (0.04, 0.07)	0.05 (0.04, 0.07)	0.05 (0.04, 0.07)	<0.001
coverage	0.940 (0.920, 0.960)	0.940 (0.920, 0.960)	0.940 (0.920, 0.960)	0.950 (0.930, 0.960)	0.940 (0.930, 0.960)	0.950 (0.923, 0.960)	0.940 (0.930, 0.960)	0.945 (0.930, 0.960)	0.940 (0.920, 0.960)	0.940 (0.930, 0.960)	0.950 (0.930, 0.960)	0.940 (0.930, 0.960)	0.940 (0.920, 0.960)	0.940 (0.920, 0.960)	0.940 (0.930, 0.960)	>0.9

Table 5: Optimal Designs for Different Sigma2 Values

sigma2	mean_mse	G	R	total_cost
0.1	0.0457286	185	1	9250
0.3	0.0526188	191	1	9550
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

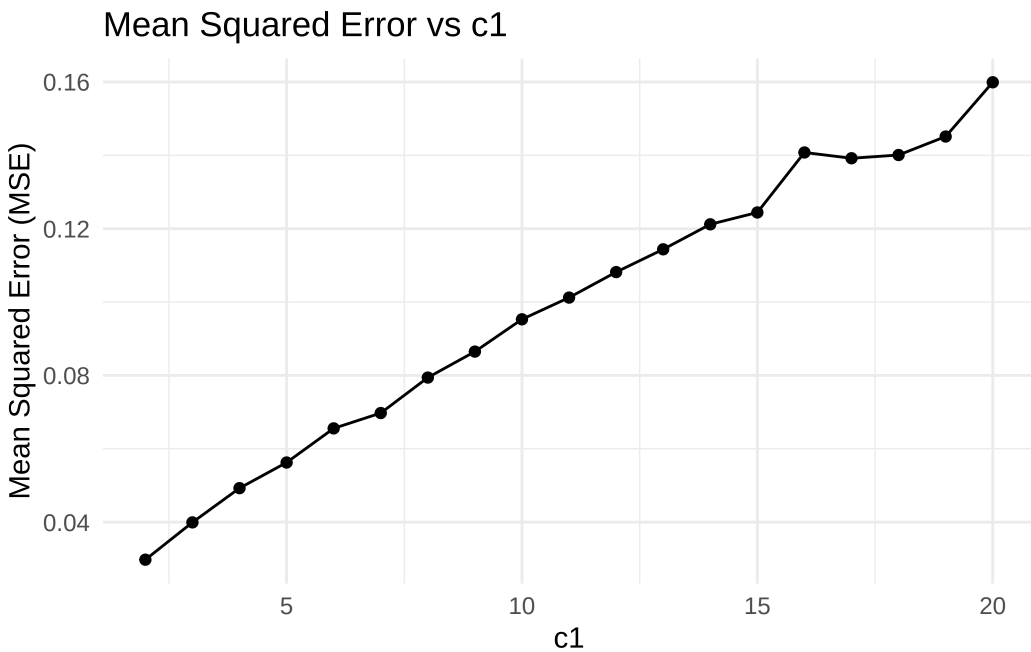

Mean Squared Error vs Sigma2
Mean Squared Error (MSE)
0.05
0.06
0.07
0
1
2
3
Sigma2

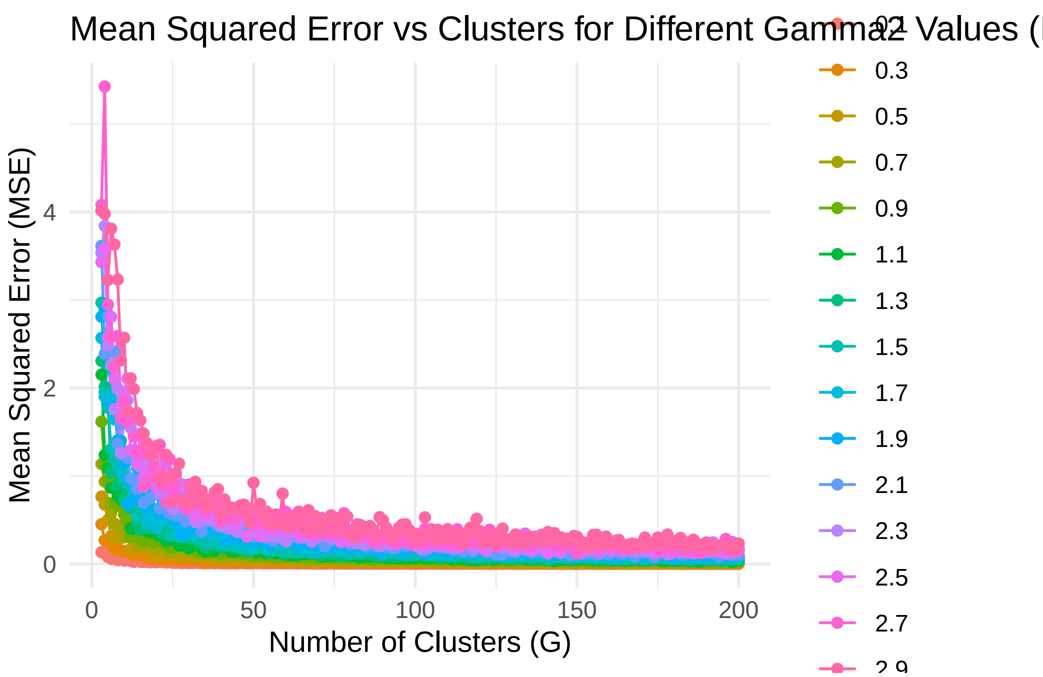

Mean Squared Error vs Clusters for Different c1/c2 ratio
Mean Squared Error (MSE)
0.00
0.25
0.50
0.75
1.00
0
100
200
300
400
500
Number of Clusters (G)

Characteristic	**2**, N = 498	**3**, N = 331	**4**, N = 248	**5**, N = 198	**6**, N = 164	**7**, N = 140	**8**, N = 123	**9**, N = 109	**10**, N = 98	**11**, N = 88	**12**, N = 81	**13**, N = 74	**14**, N = 69	**15**, N = 64	**16**, N = 60	**17**, N = 56	**18**, N = 53	**19**, N = 50	**20**, N = 48	**p-value**
bias	0.001 (-0.009, 0.010)	0.000 (-0.010, 0.010)	-0.002 (-0.013, 0.011)	-0.002 (-0.015, 0.011)	0.002 (-0.016, 0.018)	-0.001 (-0.015, 0.013)	-0.001 (-0.018, 0.014)	0.000 (-0.016, 0.023)	0.004 (-0.012, 0.018)	0.001 (-0.016, 0.017)	0.002 (-0.012, 0.021)	0.001 (-0.018, 0.022)	0.008 (-0.014, 0.023)	-0.002 (-0.015, 0.020)	-0.001 (-0.015, 0.017)	0.000 (-0.014, 0.020)	0.002 (-0.023, 0.012)	-0.007 (-0.024, 0.024)	-0.001 (-0.029, 0.025)	0.8
mse	0.02 (0.01, 0.02)	0.02 (0.02, 0.03)	0.03 (0.02, 0.04)	0.03 (0.03, 0.04)	0.04 (0.03, 0.05)	0.04 (0.03, 0.06)	0.05 (0.04, 0.07)	0.05 (0.04, 0.08)	0.05 (0.05, 0.08)	0.06 (0.05, 0.09)	0.06 (0.05, 0.09)	0.07 (0.06, 0.10)	0.08 (0.06, 0.11)	0.07 (0.06, 0.12)	0.08 (0.06, 0.12)	0.09 (0.06, 0.13)	0.09 (0.07, 0.13)	0.10 (0.07, 0.16)	0.10 (0.07, 0.16)	<0.001
coverage	0.95 (0.93, 0.96)	0.95 (0.93, 0.96)	0.95 (0.93, 0.96)	0.94 (0.92, 0.96)	0.94 (0.92, 0.96)	0.94 (0.92, 0.96)	0.94 (0.92, 0.96)	0.94 (0.92, 0.96)	0.94 (0.92, 0.96)	0.93 (0.91, 0.95)	0.94 (0.92, 0.95)	0.93 (0.91, 0.95)	0.93 (0.91, 0.95)	0.93 (0.90, 0.95)	0.93 (0.91, 0.95)	0.93 (0.90, 0.95)	0.93 (0.90, 0.95)	0.93 (0.89, 0.94)	0.92 (0.90, 0.95)	<0.001

Table 7: Optimal Designs for Different c1 Values

c1	mean_mse	G	R	total_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

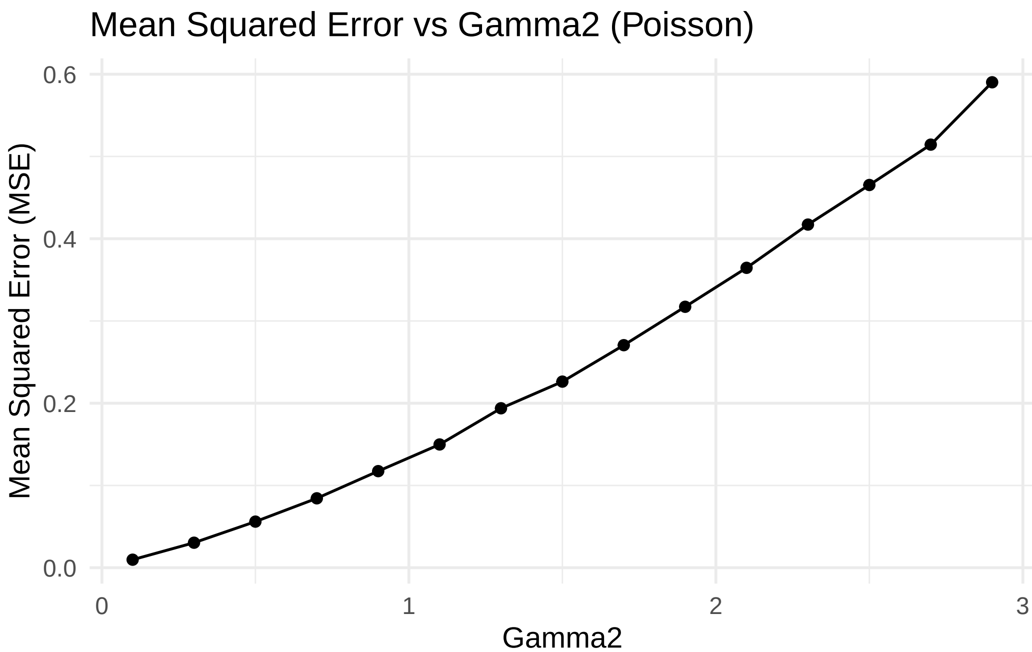

Mean Squared Error vs c1
Mean Squared Error (MSE)
0.16
0.12
0.08
0.04
5
10
15
20
c1


Mean Squared Error vs Clusters for Different Gamma2 Values (F
Mean Squared Error (MSE)
4
2
0
0
50
100
150
200
Number of Clusters (G)
0.1
0.3
0.5
0.7
0.9
1.1
1.3
1.5
1.7
1.9
2.1
2.3
2.5
2.7

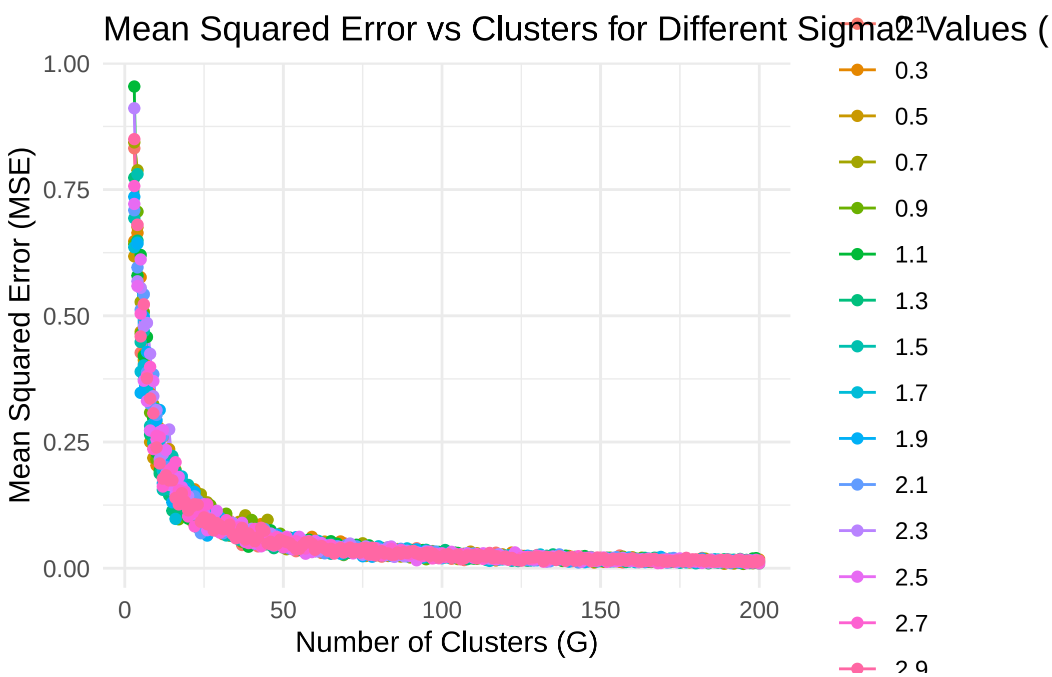
Characteristic	**0.1**, N = 198	**0.3**, N = 198	**0.5**, N = 198	**0.7**, N = 198	**0.9**, N = 198	**1.1**, N = 198	**1.3**, N = 198	**1.5**, N = 198	**1.7**, N = 198	**1.9**, N = 198	**2.1**, N = 198	**2.3**, N = 198	**2.5**, N = 198	**2.7**, N = 198	**2.9**, N = 198	**p-value**
bias	0.00 (0.00, 0.00)	0.00 (-0.01, 0.01)	0.00 (-0.01, 0.01)	0.00 (-0.02, 0.02)	0.00 (-0.02, 0.01)	0.00 (-0.02, 0.03)	0.00 (-0.02, 0.02)	0.00 (-0.03, 0.02)	0.00 (-0.03, 0.02)	0.00 (-0.03, 0.03)	0.00 (-0.03, 0.04)	-0.01 (-0.04, 0.03)	0.00 (-0.03, 0.03)	0.00 (-0.04, 0.04)	-0.01 (-0.05, 0.04)	0.8
mse	0.00 (0.00, 0.01)	0.01 (0.01, 0.03)	0.03 (0.02, 0.05)	0.04 (0.03, 0.07)	0.06 (0.04, 0.11)	0.07 (0.05, 0.14)	0.10 (0.07, 0.19)	0.12 (0.08, 0.20)	0.14 (0.10, 0.26)	0.17 (0.13, 0.30)	0.21 (0.14, 0.37)	0.25 (0.17, 0.40)	0.27 (0.20, 0.49)	0.32 (0.23, 0.54)	0.36 (0.27, 0.59)	<0.001
coverage	0.94 (0.92, 0.96)	0.94 (0.93, 0.95)	0.94 (0.91, 0.95)	0.93 (0.90, 0.95)	0.92 (0.90, 0.95)	0.92 (0.90, 0.94)	0.91 (0.89, 0.93)	0.91 (0.88, 0.94)	0.91 (0.88, 0.93)	0.90 (0.88, 0.92)	0.89 (0.85, 0.92)	0.89 (0.85, 0.91)	0.88 (0.85, 0.91)	0.87 (0.84, 0.90)	0.87 (0.84, 0.90)	<0.001

Table 9: Optimal Designs for Different Gamma2 Values (Poisson)

gamma2	mean_mse	G	R	total_cost
0.1	0.0097438	198	1	9900
0.3	0.0304002	169	1	8450
0.5	0.0560577	195	1	9750
0.7	0.0843814	194	1	9700
0.9	0.1174094	185	1	9250
1.1	0.1498408	198	1	9900
1.3	0.1938283	195	1	9750
1.5	0.2262600	184	1	9200
1.7	0.2705672	187	1	9350
1.9	0.3172452	180	1	9000
2.1	0.3645954	174	1	8700
2.3	0.4171857	178	1	8900
2.5	0.4652958	141	3	9870
2.7	0.5143786	150	2	9000
2.9	0.5901964	199	1	9950



Mean Squared Error vs Gamma2 (Poisson)
Mean Squared Error (MSE)
0.6
0.4
0.2
0.0
0
1
2
3
Gamma2

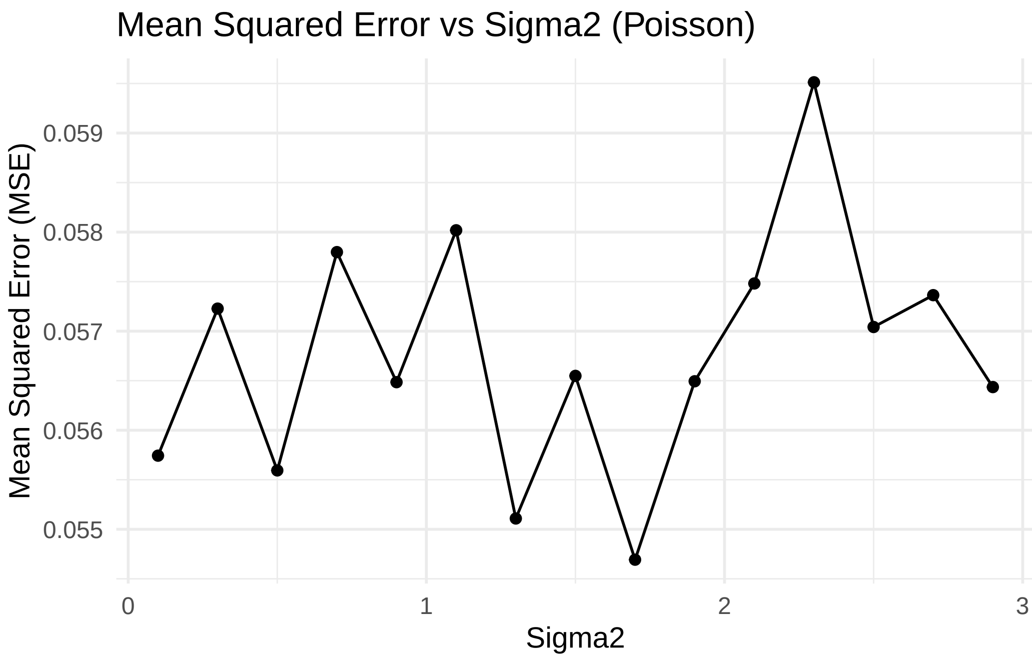


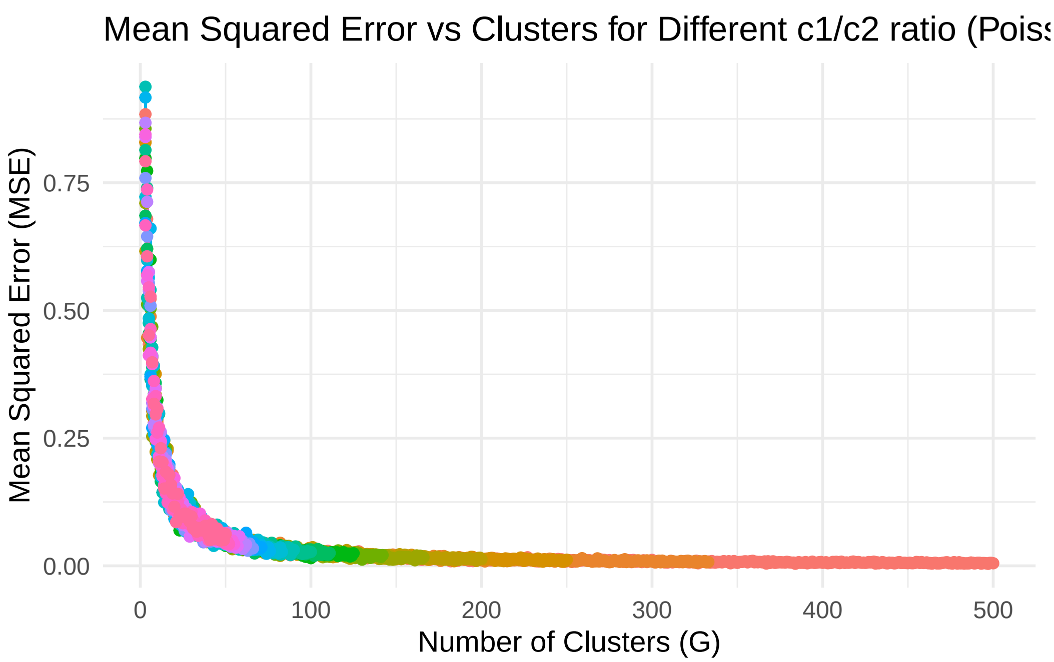
Mean Squared Error vs Clusters for Different Sigma2 Values (
Mean Squared Error (MSE)
1.00
0.75
0.50
0.25
0.00
0
50
100
150
200
Number of Clusters (G)
0.1
0.3
0.5
0.7
0.9
1.1
1.3
1.5
1.7
1.9
2.1
2.3
2.5
2.7

Characteristic	**0.1**, N = 198	**0.3**, N = 198	**0.5**, N = 198	**0.7**, N = 198	**0.9**, N = 198	**1.1**, N = 198	**1.3**, N = 198	**1.5**, N = 198	**1.7**, N = 198	**1.9**, N = 198	**2.1**, N = 198	**2.3**, N = 198	**2.5**, N = 198	**2.7**, N = 198	**2.9**, N = 198	**p-value**
bias	0.000 (-0.011, 0.009)	-0.002 (-0.012, 0.013)	-0.001 (-0.012, 0.012)	0.005 (-0.009, 0.019)	0.001 (-0.011, 0.013)	-0.002 (-0.012, 0.011)	0.000 (-0.011, 0.012)	0.001 (-0.012, 0.015)	0.001 (-0.009, 0.012)	0.001 (-0.010, 0.010)	0.002 (-0.014, 0.013)	-0.001 (-0.013, 0.011)	0.000 (-0.013, 0.011)	0.000 (-0.011, 0.011)	-0.001 (-0.011, 0.013)	0.5
mse	0.03 (0.02, 0.05)	0.02 (0.02, 0.05)	0.03 (0.02, 0.05)	0.02 (0.02, 0.05)	0.02 (0.02, 0.05)	0.03 (0.02, 0.05)	0.03 (0.02, 0.05)	0.03 (0.02, 0.05)	0.02 (0.02, 0.05)	0.03 (0.02, 0.05)	0.03 (0.02, 0.05)	0.02 (0.02, 0.05)	0.03 (0.02, 0.05)	0.03 (0.02, 0.05)	0.03 (0.02, 0.05)	>0.9
coverage	0.93 (0.91, 0.95)	0.94 (0.91, 0.95)	0.94 (0.91, 0.96)	0.94 (0.91, 0.95)	0.94 (0.91, 0.95)	0.93 (0.91, 0.95)	0.93 (0.91, 0.95)	0.93 (0.92, 0.95)	0.93 (0.91, 0.95)	0.93 (0.91, 0.96)	0.93 (0.91, 0.95)	0.93 (0.91, 0.95)	0.93 (0.91, 0.95)	0.93 (0.91, 0.95)	0.93 (0.91, 0.95)	>0.9

Table 11: Optimal Designs for Different Sigma2 Values (Poisson)

sigma2	mean_mse	G	R	total_cost
0.1	0.0557430	197	1	9850
0.3	0.0572277	193	1	9650
0.5	0.0555942	189	1	9450
0.7	0.0577980	192	1	9600
0.9	0.0564851	195	1	9750
1.1	0.0580186	184	1	9200
1.3	0.0551089	200	1	10000
1.5	0.0565486	180	1	9000
1.7	0.0546928	194	1	9700
1.9	0.0564942	191	1	9550
2.1	0.0574814	200	1	10000
2.3	0.0595126	184	1	9200
2.5	0.0570420	196	1	9800
2.7	0.0573634	200	1	10000
2.9	0.0564354	197	1	9850

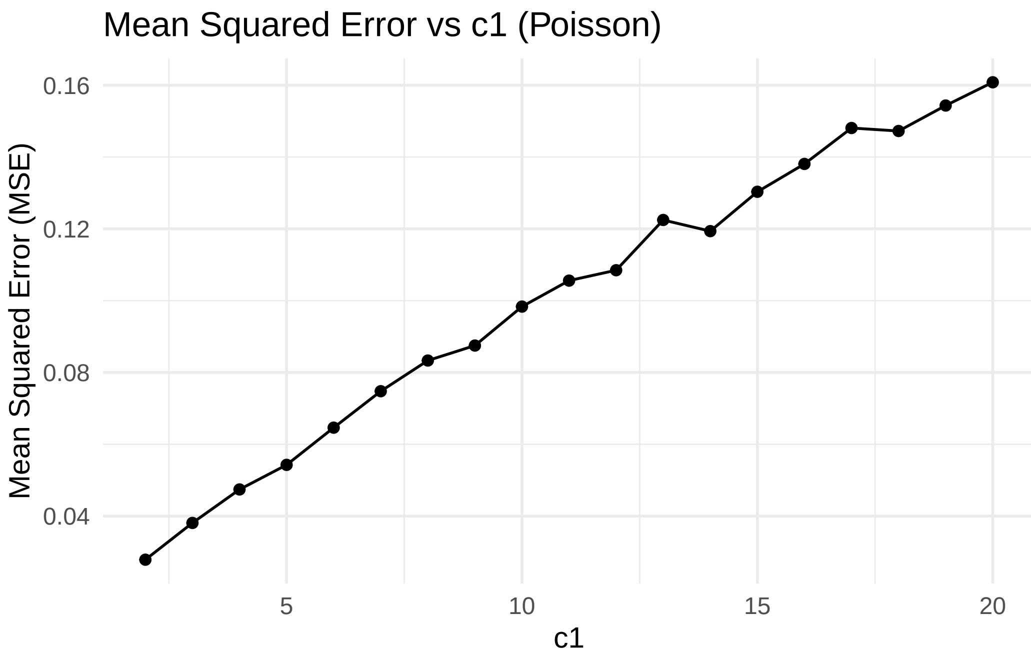

Mean Squared Error vs Sigma2 (Poisson)
Mean Squared Error (MSE)
0.059
0.058
0.057
0.056
0.055
0
1
2
3
Sigma2


Mean Squared Error vs Clusters for Different c1/c2 ratio (Poiss
Mean Squared Error (MSE)
0.75
0.50
0.25
0.00
0
100
200
300
400
500
Number of Clusters (G)

Characteristic	2, N = 498	3, N = 331	4, N = 248	5, N = 198	6, N = 164	7, N = 140	8, N = 123	9, N = 109	10, N = 98	11, N = 88	12, N = 81	13, N = 74	14, N = 69	15, N = 64	16, N = 60	17, N = 56	18, N = 53	19, N = 50	20, N = 48	p-value
bias	0.000 (-0.009, 0.008)	0.000 (-0.009, 0.008)	-0.001 (-0.011, 0.008)	0.001 (-0.010, 0.013)	0.002 (-0.010, 0.013)	0.001 (-0.012, 0.014)	-0.005 (-0.016, 0.007)	0.003 (-0.011, 0.019)	-0.002 (-0.012, 0.017)	-0.001 (-0.019, 0.016)	0.000 (-0.016, 0.016)	0.004 (-0.011, 0.019)	0.002 (-0.021, 0.017)	-0.004 (-0.029, 0.014)	-0.001 (-0.019, 0.025)	-0.005 (-0.028, 0.015)	0.007 (-0.008, 0.022)	-0.004 (-0.020, 0.019)	0.001 (-0.028, 0.022)	0.066
mse	0.01 (0.01, 0.02)	0.02 (0.01, 0.03)	0.02 (0.01, 0.04)	0.02 (0.02, 0.05)	0.03 (0.02, 0.06)	0.04 (0.02, 0.07)	0.04 (0.03, 0.07)	0.05 (0.03, 0.08)	0.05 (0.04, 0.11)	0.06 (0.04, 0.10)	0.06 (0.04, 0.12)	0.07 (0.05, 0.13)	0.07 (0.05, 0.13)	0.07 (0.05, 0.14)	0.08 (0.05, 0.14)	0.08 (0.06, 0.17)	0.09 (0.06, 0.18)	0.09 (0.06, 0.16)	0.10 (0.07, 0.18)	<0.001
coverage	0.94 (0.93, 0.96)	0.94 (0.92, 0.96)	0.94 (0.92, 0.95)	0.94 (0.91, 0.95)	0.93 (0.91, 0.95)	0.93 (0.91, 0.95)	0.93 (0.90, 0.95)	0.92 (0.90, 0.94)	0.92 (0.89, 0.95)	0.92 (0.89, 0.94)	0.93 (0.88, 0.94)	0.92 (0.88, 0.94)	0.92 (0.88, 0.93)	0.91 (0.87, 0.93)	0.91 (0.87, 0.94)	0.91 (0.86, 0.94)	0.90 (0.87, 0.93)	0.91 (0.88, 0.93)	0.91 (0.89, 0.93)	<0.001

Table 13: Optimal Designs for Different c1 Values (Poisson)

c1	mean_mse	G	R	total_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



Discussion

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) {
```

```r
  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)

  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') {
```

```r
    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)
}


# 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
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

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()

normal_alpha_results_optimal_table <- kable(design_performance_normal_alpha %>%
  group_by(alpha) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =

normal_alpha_results_optimal_plot <- design_performance_normal_alpha %>%
  group_by(alpha) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  ggplot(aes(x = alpha, y = mean_mse)) +
  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(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =

normal_beta_results_optimal_plot <- design_performance_normal_beta %>%
  group_by(beta) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  ggplot(aes(x = beta, y = mean_mse)) +
  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'

# read results
design_performance_normal_gamma2 <- read.csv("../Results/design_performance_normal_gamma2.csv

ggplot(design_performance_normal_gamma2, aes(x = G, y = mse, color = factor(gamma2))) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs Clusters for Different Gamma2 Values",
       x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()

tbl_summary(design_performance_normal_gamma2 %>% select(c(bias, mse, coverage, gamma2)), by =
  add_p()

kable(design_performance_normal_gamma2 %>%
  group_by(gamma2) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  caption = "Optimal Designs for Different Gamma2 Values")

design_performance_normal_gamma2 %>%
```

```r
  group_by(gamma2) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  ggplot(aes(x = gamma2, y = mean_mse)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs Gamma2",
       x = "Gamma2",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()
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

ggplot(design_performance_normal_sigma2, aes(x = G, y = mse, color = factor(sigma2))) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs Clusters for Different Sigma2 Values",
       x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()

tbl_summary(design_performance_normal_sigma2 %>% select(c(bias, mse, coverage, sigma2)), by =
  add_p()

kable(design_performance_normal_sigma2 %>%
  group_by(sigma2) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  caption = "Optimal Designs for Different Sigma2 Values")

design_performance_normal_sigma2 %>%
  group_by(sigma2) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  ggplot(aes(x = sigma2, y = mean_mse)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs Sigma2",
       x = "Sigma2",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()
budget <- 1000
alpha <- 5
beta <- 0.5
gamma2 <- 0.5
sigma2 <- 1
n_sims <- 100
c2 <- 1

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

```r
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")

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

ggplot(design_performance_normal_c1_c2, aes(x = G, y = mse, color = factor(c1))) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs Clusters for Different c1/c2 ratio",
       x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  theme(legend.position="none")

tbl_summary(design_performance_normal_c1_c2 %>% select(c(bias, mse, coverage, c1)), by = c1)
```

```
  add_p()

kable(design_performance_normal_c1_c2 %>%
  group_by(c1) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  caption = "Optimal Designs for Different c1 Values")

design_performance_normal_c1_c2 %>%
  group_by(c1) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  ggplot(aes(x = c1, y = mean_mse)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs c1",
       x = "c1",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()

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.

ggplot(design_performance_poisson_gamma2, aes(x = G, y = mse, color = factor(gamma2))) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs Clusters for Different Gamma2 Values (Poisson)",
       x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()

tbl_summary(design_performance_poisson_gamma2 %>% select(c(bias, mse, coverage, gamma2)), by
  add_p()

kable(design_performance_poisson_gamma2 %>%
  group_by(gamma2) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  caption = "Optimal Designs for Different Gamma2 Values (Poisson)")

design_performance_poisson_gamma2 %>%
  group_by(gamma2) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  ggplot(aes(x = gamma2, y = mean_mse)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs Gamma2 (Poisson)",
       x = "Gamma2",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()
```

```
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_poisson_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, distribu
    performance <- evaluate_design(sim_results, beta)

    design_performance_poisson_sigma2 <- rbind(design_performance_poisson_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_poisson_sigma2, "../Results/design_performance_poisson_sigma2.cs

# read results
design_performance_poisson_sigma2 <- read.csv("../Results/design_performance_poisson_sigma2.c

ggplot(design_performance_poisson_sigma2, aes(x = G, y = mse, color = factor(sigma2))) +
  geom_line() +
```

```
  geom_point() +
  labs(title = "Mean Squared Error vs Clusters for Different Sigma2 Values (Poisson)",
       x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()

tbl_summary(design_performance_poisson_sigma2 %>% select(c(bias, mse, coverage, sigma2)), by
  add_p()

kable(design_performance_poisson_sigma2 %>%
  group_by(sigma2) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  caption = "Optimal Designs for Different Sigma2 Values (Poisson)")

design_performance_poisson_sigma2 %>%
  group_by(sigma2) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  ggplot(aes(x = sigma2, y = mean_mse)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs Sigma2 (Poisson)",
       x = "Sigma2",
       y = "Mean Squared Error (MSE)") +
  theme_minimal()
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_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

ggplot(design_performance_poisson_c1_c2, aes(x = G, y = mse, color = factor(c1))) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Squared Error vs Clusters for Different c1/c2 ratio (Poisson)",
       x = "Number of Clusters (G)",
       y = "Mean Squared Error (MSE)") +
  theme_minimal() +
  theme(legend.position="none")

tbl_summary(design_performance_poisson_c1_c2 %>% select(c(bias, mse, coverage, c1)), by = c1)
  add_p()

kable(design_performance_poisson_c1_c2 %>%
  group_by(c1) %>%
  summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
  caption = "Optimal Designs for Different c1 Values (Poisson)")

design_performance_poisson_c1_c2 %>%
  group_by(c1) %>%
```

```
summarize(mean_mse = mean(mse), G = G[which.min(mse)], R = R[which.min(mse)], total_cost =
ggplot(aes(x = c1, y = mean_mse)) +
geom_line() +
geom_point() +
labs(title = "Mean Squared Error vs c1 (Poisson)",
     x = "c1",
     y = "Mean Squared Error (MSE)") +
theme_minimal()
```