PHP 2550 Project 3: Optimizing Recruitment Designs for Cluster-Randomized Trials Under Budget Constraints

Peirong Hao

Abstract

This simulation project investigates optimal recruitment designs for cluster-randomized trials within budgetary limitations. The project examines two data-generating models: normally distributed outcomes and poisson-distributed outcomes. Hierarchical data are generated with multiple parameters, such as the number of clusters (G), the number of observations per cluster (R), and cluster-level variance (γ^2). The simulation attempts to determine an optimal balance between G and R to reduce estimation variability. The comparative costs of sampling new clusters (c_1) versus conducting additional measurements within existing clusters (c_2) are adjusted to assess their influence on estimates. Models are fitted using linear mixed-effects models for the normal scenario and generalized linear mixed-effects models for the poisson scenario, with performance assessed through metrics including the variance of estimates and average of estimates' standard errors. Results indicate that increasing the number of clusters (G) improves precision within specified budgets. Higher budgets enable the inclusion of more observations in both models. Larger β values enhance estimate stability, while higher γ^2 amplifies variability, particularly in the Poisson model. As c_1 (cost per new cluster) increases, G decreases. Higher relative cost ratio (c_1/c_2) also makes collecting additional observations within existing clusters more economical. Overall, the normal model exhibits greater stability than the Poisson model. These findings provide practical guidance for researchers designing cost-effective cluster-randomized trials.

Simulation Design Using ADEMP Framework

Aims

We aim to enhance the design of a cluster-randomized trial by accounting for budget constraints and varying parameters. This project explores two data-generating scenarios: normally dis-

tributed outcomes and poisson-distributed outcomes. The simulation focuses on identifying the optimal combination of the number of clusters (G) and the number of observations per cluster (R) that minimizes estimation variability within a fixed budget (B). Moreover, the study investigates the impact of varying the relative costs of sampling the first observation from a cluster (c_1) compared to additional measurements within the same cluster (c_2) under the condition that $c_1 > c_2$. The analysis further explores how adjustments to model parameters—such as the intercept (α), treatment effect (β), and cluster-level variance (γ^2)—influence the treatment effect estimates. Finally, it examines the effect of varying the budget (B) on the optimal design.

Data-Generating Mechanisms

We generate the data using hierarchical models. Observations j range from 1 to R, representing measurements within each cluster, while clusters i range from 1 to G (the total number of clusters). Treatment assignment (X_i) is originally determined by a Bernoulli distribution with a fixed probability of 0.5, ensuring an approximately equal allocation of clusters to treatment and control groups. However, to avoid scenarios where all clusters are assigned to a single group, the assignment is manually balanced. Observations within the same cluster share a cluster-level mean, modeled as $\mu_i = \alpha + \beta X_i + \epsilon_i$, where ϵ_i represents the cluster-level deviation and follows a distribution $N(0, \gamma^2)$. Individual-level outcomes are generated as $Y_{ij} = \mu_i + e_{ij}$, where e_{ij} reflects within-cluster variation and is drawn from $N(0, \sigma^2)$. For the poisson scenario, the cluster-level mean is modeled on the log scale as $\log(\mu_i) = \alpha + \beta X_i + \epsilon_i$, where $\epsilon_i \sim N(0, \gamma^2)$ captures the cluster-level variability. Individual-level outcomes are generated from a poisson distribution, $Y_{ij} \sim \text{Poisson}(\mu_i)$. Unlike the normal scenario, there is no σ parameter in this case because the poisson distribution models variability through its mean, with the variance being equal to the mean.

Estimands

The primary estimand is the treatment effect ($\widehat{\beta}$), contributing to the outcome difference between the treatment and control groups. In the normal scenario, β reflects the difference in cluster-level means between the two groups, while in the poisson scenario, $\widehat{\beta}$ represents the log-scale difference in cluster-level means. Importantly, β is an unbiased estimate. The precision and accuracy of $\widehat{\beta}$ are assessed by examining the variance of the estimates and the squared average standard error of the estimates.

Methods

The simulation generates hierarchical data using the `DataSim()` function, allowing for variation in parameters such as the number of clusters (G), the number of observations per cluster (R), the cluster-level variance (γ^2), intercept (α), and treatment effect (β). For each set of

parameters, two datasets—one based on normal models and the other on poisson models—are generated per iteration, with multiple iterations conducted in total to ensure robust evaluation. To optimize the recruitment design, the number of clusters (G) and the number of observations per cluster (R) are determined using the `BudgetOpt()` function, which incorporates the fixed budget (B), the cost of sampling from new clusters (c_1), and the relative costs of c_1 versus c_2 (the cost of collecting additional measurements within existing clusters). Constraints like ensuring a minimum of four clusters ($G \geq 4$) and at least two observations per cluster ($R \geq 2$) preserve hierarchical order and prevent errors during model fitting. The optimization process starts by determining the maximum feasible number of clusters (G) that can be managed within the budget while satisfying the minimum conditions for both G and R. For every feasible G value, the associated R values are calculated using the GetR() function. The ultimate sets of G and R values are chosen according to their adherence to all restrictions.

The `ParamVary()` function systematically varies a single parameter—such as α, β, γ, c_1, the relative cost of c_1 versus c_2, or B—while keeping other variables constant. The default parameters used in the simulations are $B = 5000$, $c_1 = 20$, relative cost of c_1 versus $c_2 = 2$, $\alpha = 5$, $\beta = 3$, $\gamma = 2$, and $\sigma = 0.5$. When one variable changes, the others remain constant to isolate its effect. For each parameter configuration, the function generates datasets corresponding to all possible pairs of G and R, fits hierarchical models (`lmer()` for normally distributed outcomes and `glmer()` for poisson-distributed outcomes), and saves the estimated β, standard errors of the estimates, and resource allocation parameters (G and R). The MeasureGen() function processes these results, summarizing key metrics such as the variance of the estimated β and the mean of squared standard errors for each combination of G and R. All simulated datasets and their corresponding performance metrics are stored as CSV files in specified folders for further evaluation and replication. It is important to note that the σ value in the normal scenario, indication of within-cluster variability, does not affect the estimation of the treatment effect β. Since β reflects the disparity in cluster-level means between the treated and control groups, this disparity is unaffected by the within-cluster noise σ. As a result, σ can remain constant. Likewise, the intercept parameter α in the normal setting should not influence the estimation of β, as it denotes a baseline value that shifts all cluster means, irrespective of treatment status. Nonetheless, α may influence the estimation of β in the poisson instance, since α influences the cluster mean exponentially, significantly impacting the outcome Y.

Performance Measures

The performance of the study design is assessed through multiple metrics that evaluate the reliability of the estimated treatment effect. The variance of the estimated β quantifies the variability of the estimates across iterations. The mean of the standard errors of estimates is calculated to describe the expected uncertainty in the treatment effect for a single dataset. The simulation also tracks the number of clusters (G) and the number of observations per cluster (R) within the specified budget constraints.

Tables and Figures

Table 1: (G,R) pair, vary c1

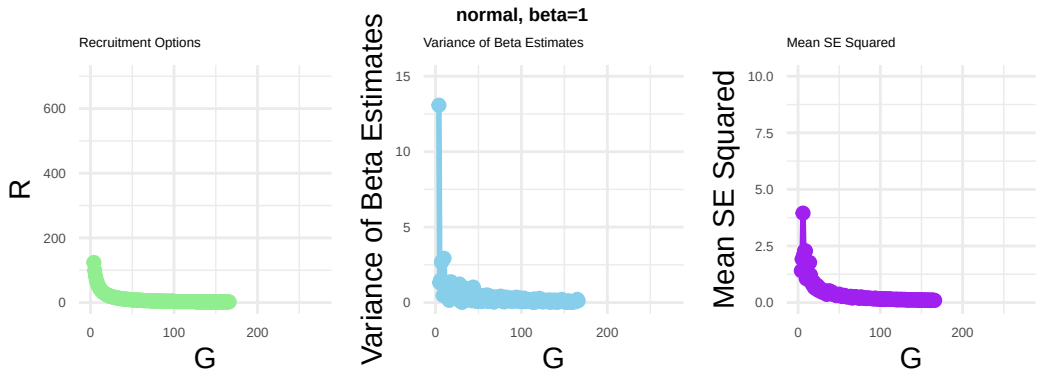
Param	Normal	Poisson
10	315, 2	259, 2
20	89, 4	122, 3
30	66, 4	36, 8

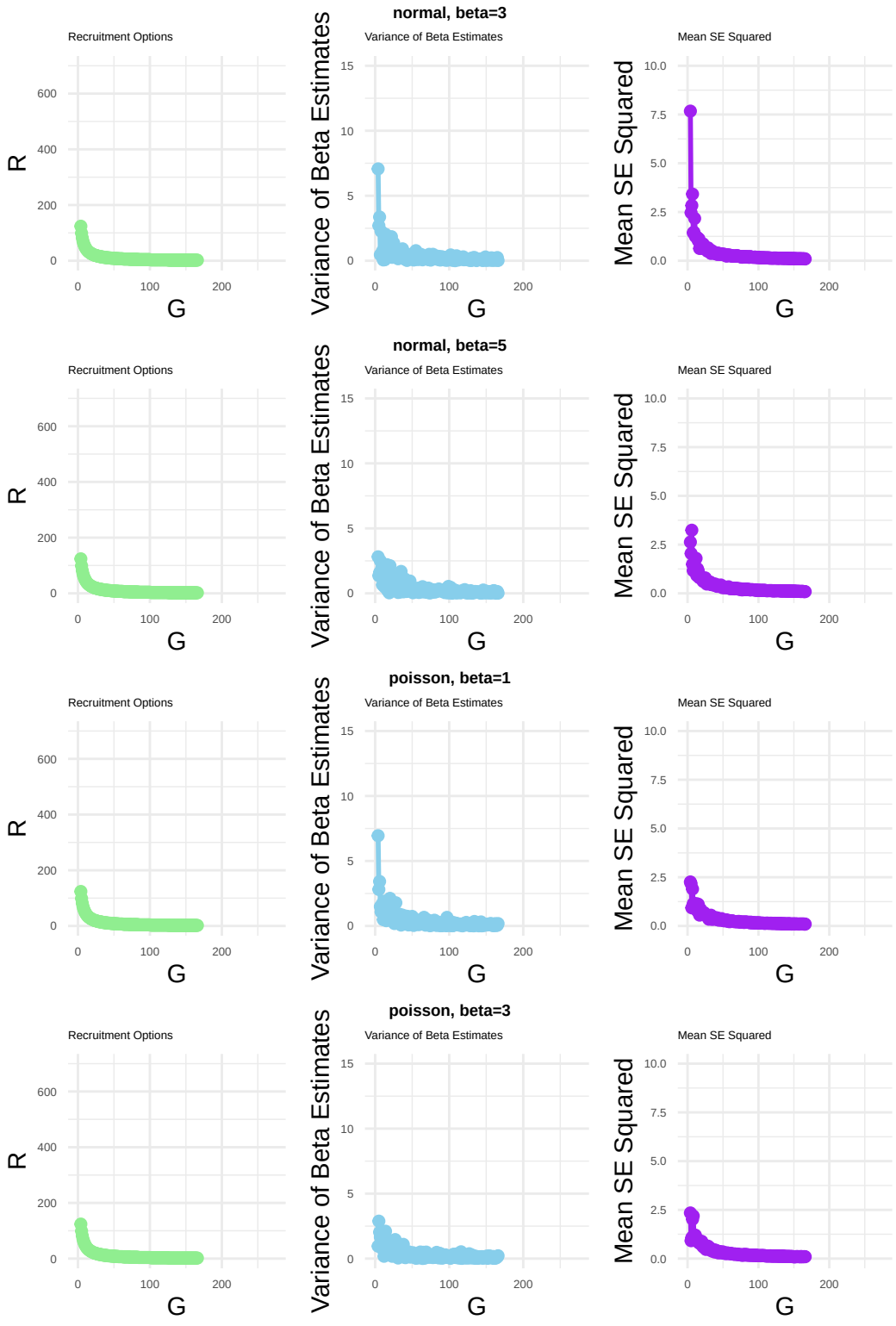
Table 2: (G,R) pair, vary relative cost

Param	Normal	Poisson
4	125, 5	170, 2
5	198, 2	134, 5
10	226, 2	195, 3

Table 3: (G,R) pair, vary alpha

Param	Normal	Poisson
5	116, 3	151, 2
6	61, 7	98, 4
7	72, 5	161, 2




normal, beta=3
Recruitment Options
Variance of Beta Estimates
Mean SE Squared
R
G
normal, beta=5
poisson, beta=1
poisson, beta=3

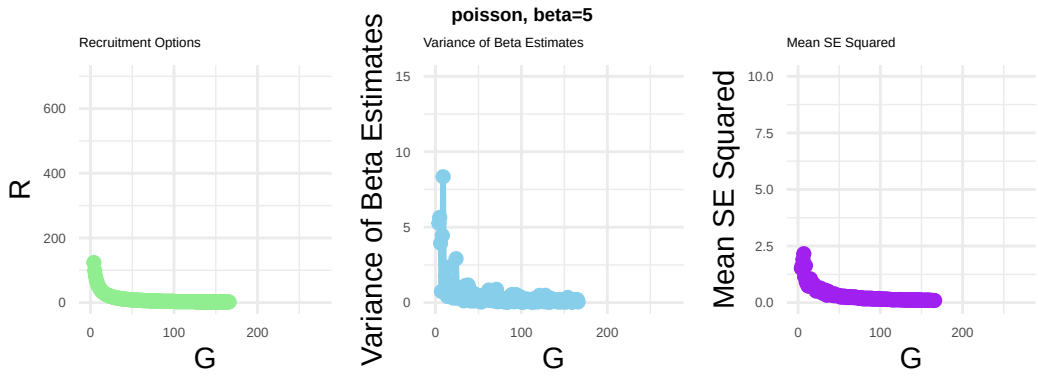
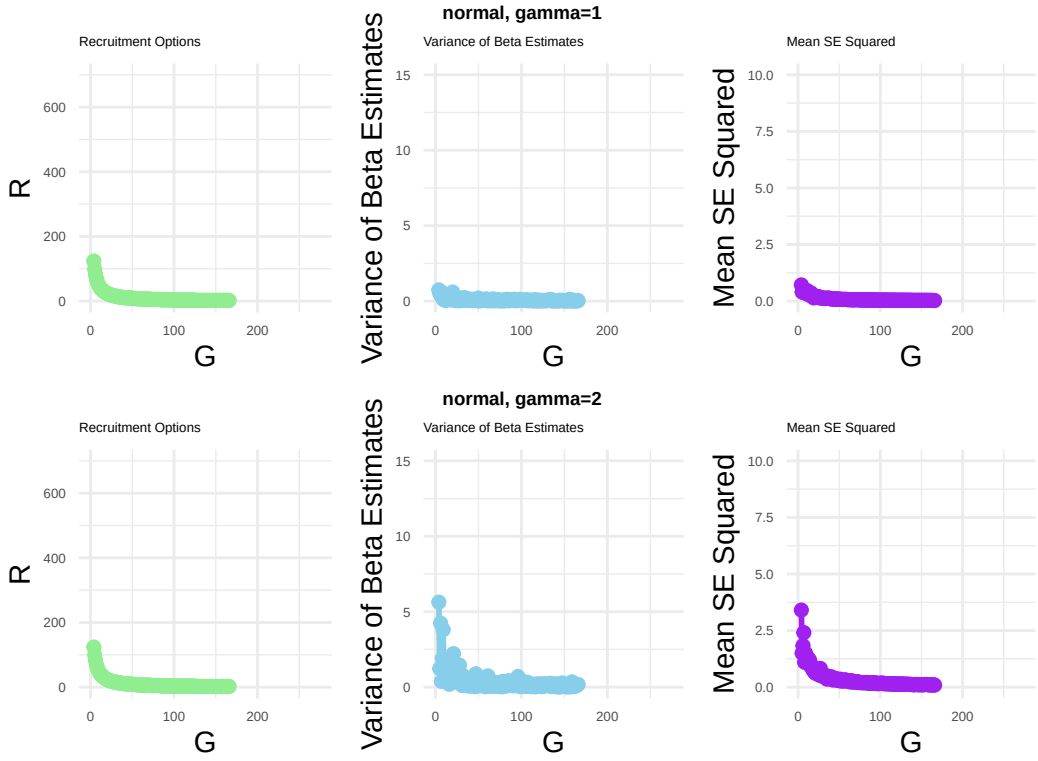


Table 4: (G,R) pair, vary beta

Param	Normal	Poisson
1	159, 2	140, 2
3	108, 3	64, 6
5	104, 3	133, 2



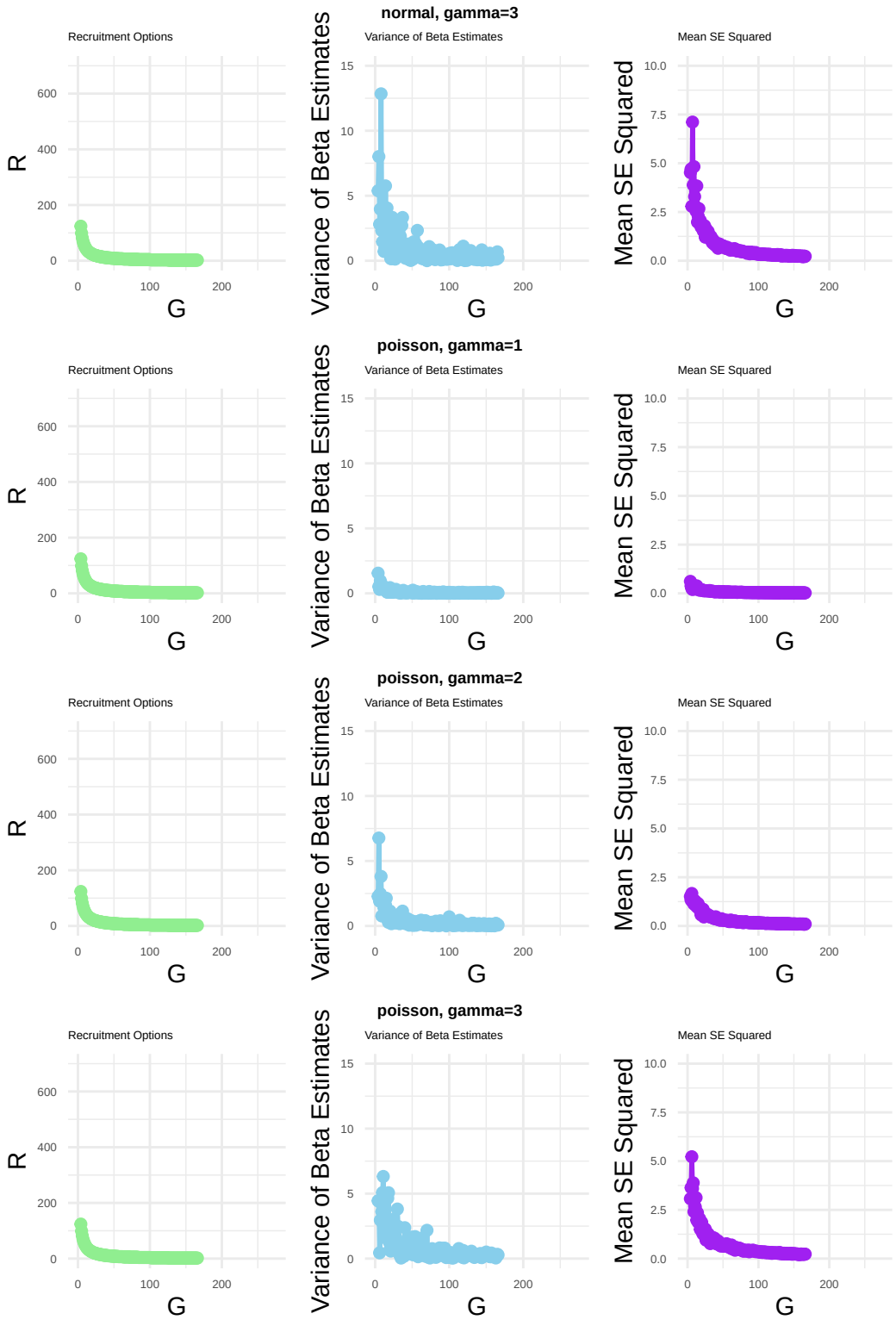

normal, gamma=3
poisson, gamma=1
poisson, gamma=2
poisson, gamma=3
Recruitment Options
Variance of Beta Estimates
Mean SE Squared
R
G
Variance of Beta Estimates
Mean SE Squared
0
200
400
600
5
10
15
2.5
5.0
7.5
10.0
0
100
200

Table 5: (G,R) pair, vary gamma

Param	Normal	Poisson
1	121, 3	83, 5
2	156, 2	96, 4
3	70, 6	104, 3

Table 6: (G,R) pair, vary B

Param	Normal	Poisson
3000	85, 2	62, 3
4000	63, 5	98, 3
5000	109, 3	114, 3

Results

Comparisons are made by analyzing all the generated plots, though most are excluded from the report to save space. Simulations demonstrated that increasing the number of clusters (G) improves the reliability of treatment effect estimates, as both the variance of beta estimates and the mean squared standard errors decrease. Unsurprisingly, due to budget constraints, the number of observations per cluster (R) decreases as G increases. Increasing the budget (B) allows for more observations to be included. When c_1 (the cost per new cluster) increases, the number of clusters (G) decreases because recruiting new clusters becomes more expensive. Similarly, as the relative cost increases, it becomes more economical to continue collecting observations within existing clusters, increasing R. Increasing α and β values reduces the variance of β estimates by improving the signal-to-noise ratio. This effect is more consistent in the normal model. In contrast, the poisson model exhibits greater variability, particularly for small clusters, resulting in less stable estimates. Higher values of (γ^2) amplify the variability of estimates, with considerably greater impact on the poisson model compared to the normal model. This leads to greater variance and higher mean squared standard errors in the poisson model.

The summary tables are generated by selecting the optimal (G, R) pair that minimizes the variance of the estimated beta. Consistent with observations from the plots, as c_1 (cost per new cluster) increases, the number of clusters (G) decreases. Similarly, as the relative cost ratio c_1/c_2 grows, acquiring additional observations within existing clusters is expected to become more cost-effective, potentially leading to an increase in R. However, no clear trend of increasing R is observed in either the normal or poisson settings. In the poisson model, increasing α intensifies the exponential relationship, influencing both G and R, though no obvious pattern of increase or decrease is evident. Higher β values improve the signal-to-noise

ratio, thereby stabilizing the estimates. Larger γ^2 amplifies variability, particularly in the Poisson model, thereby reducing stability. In response to higher γ^2, the poisson model tends to select larger G values. As the budget (B) increases, the Poisson model prioritizes increasing the number of clusters (G), whereas the normal model shows a more balanced allocation between increasing G and the number of observations per cluster (R), with no consistent trend.

Limitations

The simulations provide valuable, data-driven insights into the design of cluster-randomized trials; however, several limitations should be considered. The models are based on simplified hierarchical frameworks that assume independent clusters, whereas real-world clusters may exhibit interdependencies. Additionally, the simulations do not account for common challenges such as missing data and imbalanced treatment allocation. Assumptions of no measurement error and the absence of unobserved confounding may oversimplify complex real-world scenarios. Due to time constraints, the maximum number of iterations was limited to five, which may affect the robustness of the results; increasing the number of iterations could yield more reliable findings. Furthermore, only three values were tested for each parameter, restricting the scope of the analysis. Expanding the range of parameter values and incorporating additional variations would enhance the study's comprehensiveness. Lastly, simultaneously varying multiple parameters in a nested framework could provide deeper insights into their interactions and combined effects. Addressing these limitations in future research will offer a more complete understanding of the trade-offs involved in designing cost-effective cluster-randomized trials.

Code Appendix

```
set.seed(123456)
library(dplyr)
library(tidyr)
library(ggplot2)
library(gridExtra)
library(grid)
library(purrr)
library(lme4)
library(knitr)
library(Rlab)
#data generation
#G clusters, each cluster has R members
#clusters are independent
#members in each cluster share same cluster mean (correlated)
DataSim <- function(G, R, alpha, beta, gamma, method, sigma=0.5,
 ↪  p.trt=0.5){
  data <-data.frame(matrix(ncol = 7, nrow = G*R))
  colnames(data) <- c('G', 'R', 'X', 'Y', 'alpha', 'beta', 'gamma')
  # data[,"G"] <- rep(1:G, each=R)
  # data[,"R"] <- rep(1:R, G)

  # Generate X: 0 for ctrl, 1 for trt
  #x <- rbern(n=G, prob=p.trt)#not use it because sometimes all xs are 0
  x <- c(rep(0, floor(G / 2)), rep(1, ceiling(G / 2)))
  x <- sample(x)

  mu.0 <- alpha + beta * x

  epsilon <- rnorm(G, mean=0, sd=gamma)
  mu <- mu.0 + epsilon
  if(method == "poisson"){
      mu <- exp(mu)
    }
  #mu <- ifelse(method=="poisson", exp(mu.0 + epsilon), mu.0 + epsilon)

  for(i in 1:G){
    if(method == "poisson"){
      y <- rpois(R, lambda = mu[i])
    }
```

```
    else{#normal
      eps <- rnorm(n=R, mean=0, sd=sigma)
      y <- mu[i] + eps
    }

    for(j in 1:R){
      row.num <- i*R-(R-j)
      data[row.num, ] <- c(i, j, x[i], y[j],
                           alpha, beta, gamma)
    }
  }
  return(data)
}

#alpha refers to intercept
#method = "poisson" or "normal"
DataGen <- function(folder.data, filename, alpha, beta, gamma, B, C1,
 ↪  relative.cost, method){
  if (!dir.exists(folder.data)) {
      dir.create(folder.data)}

  B.Opt <- BudgetOpt(B, C1, relative.cost)
  R <- B.Opt$R
  G <- B.Opt$G

  for(k in 1:length(G)){
    # simulate data
    data <- DataSim(G[k], R[k], alpha, beta, gamma, method)
    write.csv(data, paste0(folder.data,
     ↪  filename,'_',G[k],'_',R[k],"_data.csv"), row.names=FALSE)
  }
  return(B.Opt)
}

# ##########################################test data generation
# #method="normal"
# method="poisson"
# for(i in 1:2){
#   DataGen(folder.data =
 ↪  "~/Documents/GitHub/PHP2550-PDA-project3/Data/",
#           filename = paste0("sim",'_',i,'_',method), method = method)
```

```
# }
#
 ↪  #test1<-read.csv("~/Documents/GitHub/PHP2550-PDA-project3/Data/sim_1_data.csv")
#
# #method="normal"
# method="poisson"
# folder.data = "~/Documents/GitHub/PHP2550-PDA-project3/Data/"
# filename.data = paste0("sim",'_',i,'_',method)
# data.path = paste0(folder.data, filename.data, "_data.csv")
# #ModelFit(data.path, method=method)
# ##########################################test data generation

#fit hierarchical model
#method = "poisson" or "normal"
ModelFit <- function(data.path, method, true.beta=0.5){

  data <- read.csv(data.path)

  if(method == "poisson"){
      mdl <- glmer(Y ~ X + (1 | G), data=data, family="poisson")
    }
  else{
      mdl <- lmer(Y ~ X + (1 | G), data=data, control =
 ↪  lmerControl(optimizer = "Nelder_Mead"))
    }

  #do not need to worry about p-value < or > alpha=0.05
  #do not need coverage, i.e. 95% CI
  summ <- summary(mdl)
  est.beta <- summ$coefficients["X", "Estimate"]
  se <- summ$coefficients["X", "Std. Error"]

  measure <- c(est.beta, se)
  return(measure)
}


#constraint: G>=4, R>=2 to maintain hierarchical order, o/w error occurs
 ↪  when fitting model
#make sure C1 > C2
BudgetOpt <- function(B, C1, relative.cost){
```

```
  C2 <- C1/relative.cost
  G.min <- 4
  R.min <- 2
  B.remain <- B-G.min*C1-(R.min-1)*C2 #remaining budget after satisfying
 ↪  the minimum requirement for G,R
  G.max <- floor(B.remain/(C1+(R.min-1)*C2))+G.min
  #each new cluster has to at least two observation, max.num = potential
   ↪  (integer) + min.num

  G.all <- G.min:G.max
  R.all <- sapply(G.all, GetR, B=B, C1=C1, C2=C2)#find the corresponding
 ↪  R for each G
  indices <- which(R.all >= R.min)#have to satisfy this requirement

  G.final <- G.all[indices]
  R.final <- R.all[indices]

  return(list(G=G.final, R=R.final))
}

GetR <- function(G, B, C1, C2){
  #constraint: C1*G + C2*G*(R-1) <= B
  R <- floor(1 + (B-C1*G)/(C2*G))
  return(R)
}

#param: C1, relative.cost, beta, gamma, (change alpha mainly for poisson
 ↪  case)
#do not change p.trt, sigma
#could change B as well
ParamVary <- function(param.ls, param.name, method, max.iter = 5,
                      folder.data =
                       ↪  "~/Documents/GitHub/PHP2550-PDA-project3/Data/",
                      folder.perf =
                       ↪  "~/Documents/GitHub/PHP2550-PDA-project3/Perf/",
                      C1=20, relative.cost=2, alpha=5, beta=3, gamma=2,
                       ↪  B=5000){

  if (!dir.exists(folder.perf)) {
            dir.create(folder.perf)}
```

```
 for(i in 1:length(param.ls)){
     param = param.ls[i]

     for(j in 1:max.iter){

       if(param.name=="C1"){
         filename.data =
↪  paste0("sim",'_',method,'_',param.name,'_',param,'_',j)
         B.Opt <- DataGen(folder.data, filename.data, C1=param,
↪  method=method,
                  relative.cost=relative.cost, alpha=alpha, beta=beta,
                   ↪  gamma=gamma, B=B)
         R <- B.Opt$R
         G <- B.Opt$G
         K <- length(G)

         if(j==1){
           perf.measure <- data.frame(matrix(ncol = 4, nrow =
↪  max.iter*K))
           colnames(perf.measure) <- c("est.beta", "se", "G", "R")
           }

         for(k in 1:K){
           filename.data =
↪  paste0("sim",'_',method,'_',param.name,'_',param,'_',j,'_',G[k],'_',R[k])
           data.path = paste0(folder.data, filename.data, "_data.csv")
           measure <- ModelFit(data.path, method=method)

           row.num <- j*K-(K-k)
           perf.measure[row.num, ] <- c(measure[1], measure[2], G[k],
↪  R[k])
           }

         if(j==max.iter){
           filename.perf =
↪  paste0("sim",'_',method,'_',param.name,'_',param)
           write.csv(perf.measure, paste0(folder.perf, filename.perf,
            ↪  "_perf.csv"), row.names=FALSE)}
       }

       else if(param.name=="relative.cost"){
```

```
      filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j)
      B.Opt <- DataGen(folder.data, filename.data,
↪ relative.cost=param, method=method,
              C1=C1, alpha=alpha, beta=beta, gamma=gamma, B=B)
      R <- B.Opt$R
      G <- B.Opt$G
      K <- length(G)

      if(j==1){
        perf.measure <- data.frame(matrix(ncol = 4, nrow =
↪ max.iter*K))
        colnames(perf.measure) <- c("est.beta", "se", "G", "R")
        }

      for(k in 1:K){
        filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j,'_',G[k],'_',R[k])
        data.path = paste0(folder.data, filename.data, "_data.csv")
        measure <- ModelFit(data.path, method=method)

        row.num <- j*K-(K-k)
        perf.measure[row.num, ] <- c(measure[1], measure[2], G[k],
↪ R[k])
        }

      if(j==max.iter){
        filename.perf =
↪ paste0("sim",'_',method,'_',param.name,'_',param)
        write.csv(perf.measure, paste0(folder.perf, filename.perf,
          ↪  "_perf.csv"), row.names=FALSE)}
    }
    else if(param.name=="alpha"){

      filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j)
      B.Opt <- DataGen(folder.data, filename.data, alpha=param,
↪ method=method,
              relative.cost=relative.cost, C1=C1, beta=beta,
               ↪  gamma=gamma, B=B)
```

```
        R <- B.Opt$R
        G <- B.Opt$G
        K <- length(G)

        if(j==1){
          perf.measure <- data.frame(matrix(ncol = 4, nrow =
↪ max.iter*K))
          colnames(perf.measure) <- c("est.beta", "se", "G", "R")
          }

        for(k in 1:K){
          filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j,'_',G[k],'_',R[k])
          data.path = paste0(folder.data, filename.data, "_data.csv")
          measure <- ModelFit(data.path, method=method)

          row.num <- j*K-(K-k)
          perf.measure[row.num, ] <- c(measure[1], measure[2], G[k],
↪ R[k])
          }

        if(j==max.iter){
          # Create csv file
          filename.perf =
↪ paste0("sim",'_',method,'_',param.name,'_',param)
          write.csv(perf.measure, paste0(folder.perf, filename.perf,
           ↪  "_perf.csv"), row.names=FALSE)}

      }
      else if(param.name=="beta"){

        filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j)
        B.Opt <- DataGen(folder.data, filename.data, beta=param,
↪ method=method,
                 relative.cost=relative.cost, alpha=alpha, C1=C1,
                  ↪  gamma=gamma, B=B)
        R <- B.Opt$R
        G <- B.Opt$G
        K <- length(G)
```

```
        if(j==1){
          perf.measure <- data.frame(matrix(ncol = 4, nrow =
↪ max.iter*K))
          colnames(perf.measure) <- c("est.beta", "se", "G", "R")
          }

        for(k in 1:K){
          filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j,'_',G[k],'_',R[k])
          data.path = paste0(folder.data, filename.data, "_data.csv")
          measure <- ModelFit(data.path, method=method)

          row.num <- j*K-(K-k)
          perf.measure[row.num, ] <- c(measure[1], measure[2], G[k],
↪ R[k])
          }

        if(j==max.iter){
          filename.perf =
↪ paste0("sim",'_',method,'_',param.name,'_',param)
          write.csv(perf.measure, paste0(folder.perf, filename.perf,
           ↪  "_perf.csv"), row.names=FALSE)}
      }
      else if(param.name=="gamma"){

        filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j)
        B.Opt <- DataGen(folder.data, filename.data, gamma=param,
↪ method=method,
                 relative.cost=relative.cost, alpha=alpha, beta=beta,
                  ↪  C1=C1, B=B)
        R <- B.Opt$R
        G <- B.Opt$G
        K <- length(G)

        if(j==1){
          perf.measure <- data.frame(matrix(ncol = 4, nrow =
↪ max.iter*K))
          colnames(perf.measure) <- c("est.beta", "se", "G", "R")
          }
```

```
        for(k in 1:K){
          filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j,'_',G[k],'_',R[k])
          data.path = paste0(folder.data, filename.data, "_data.csv")
          measure <- ModelFit(data.path, method=method)

          row.num <- j*K-(K-k)
          perf.measure[row.num, ] <- c(measure[1], measure[2], G[k],
↪ R[k])
          }

        if(j==max.iter){
          filename.perf =
↪ paste0("sim",'_',method,'_',param.name,'_',param)
          write.csv(perf.measure, paste0(folder.perf, filename.perf,
           ↪  "_perf.csv"), row.names=FALSE)
          }
      }
      else if(param.name=="B"){
        filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j)
        B.Opt <- DataGen(folder.data, filename.data, B=param,
↪ method=method,
                relative.cost=relative.cost, alpha=alpha, beta=beta,
                 ↪  gamma=gamma, C1=C1)
        R <- B.Opt$R
        G <- B.Opt$G
        K <- length(G)

        if(j==1){
          perf.measure <- data.frame(matrix(ncol = 4, nrow =
↪ max.iter*K))
          colnames(perf.measure) <- c("est.beta", "se", "G", "R")
          }

        for(k in 1:K){
          filename.data =
↪ paste0("sim",'_',method,'_',param.name,'_',param,'_',j,'_',G[k],'_',R[k])
          data.path = paste0(folder.data, filename.data, "_data.csv")
          measure <- ModelFit(data.path, method=method)
```

```
            row.num <- j*K-(K-k)
            perf.measure[row.num, ] <- c(measure[1], measure[2], G[k],
↪  R[k])
            }

          if(j==max.iter){
            filename.perf =
↪  paste0("sim",'_',method,'_',param.name,'_',param)
            write.csv(perf.measure, paste0(folder.perf, filename.perf,
             ↪  "_perf.csv"), row.names=FALSE)}
        }

      }
  }
}


MeasureGen <- function(param.ls, param.name, method,
                       folder.perf =
                        ↪  "~/Documents/GitHub/PHP2550-PDA-project3/Perf"){

  param.len <- length(param.ls)
  result.ls <- vector("list", param.len)

  for(i in 1:param.len){
    param = param.ls[i]
    file.i <- list.files(path = folder.perf,
                                  pattern =
                                   ↪  paste0("sim",'_',method,'_',param.name,'_',param),
                                   ↪
                                  full.names = T)
    file.df <- read.csv(file.i, stringsAsFactors = FALSE, header = TRUE)

    result <- file.df %>%
      group_by(G, R) %>%
      summarise(
        Var_Beta_Est = var(est.beta, na.rm = TRUE),
        Mean_SE_Sq = (mean(se, na.rm = TRUE))^2,
        .groups = "drop")

    result.ls[[i]] <- result
```

```
  }
  return(result.ls)
}


GRGen <- function(param.ls, param.name, method,
                        folder.perf =
                         ↪  "~/Documents/GitHub/PHP2550-PDA-project3/Perf"){

  param.len <- length(param.ls)
  result.ls <- vector("list", param.len)

  for(i in 1:param.len){
    param = param.ls[i]
    file.i <- list.files(path = folder.perf,
                                  pattern =
                                   ↪  paste0("sim",'_',method,'_',param.name,'_',param),
                                   ↪
                                  full.names = T)
    file.df <- read.csv(file.i, stringsAsFactors = FALSE, header = TRUE)

    result <- file.df %>%
        group_by(G, R) %>%
        summarise(
          Var_Beta_Est = var(est.beta, na.rm = TRUE),
          .groups = "drop") %>%
        slice_min(order_by = Var_Beta_Est, n = 1)%>%
          unite("combined", G:R, sep = ", ") %>%
          pull(combined)#character

    result.ls[[i]] <- result

  }
  return(result.ls)
}
method = "normal"
param.name = "C1"
param.ls = c(10,20,30)
ParamVary(param.ls, param.name, method,
          relative.cost=2, alpha=5, beta=3, gamma=2, B=5000)
```

```
#result.C1.normal <- MeasureGen(param.ls, param.name, method)
GR.C1.normal <- GRGen(param.ls, param.name, method)

# for(i in 1:length(param.ls)){
#   param = param.ls[i]
  # tbl <- result.C1.normal[[i]]
  # plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
  #             geom_line(color = "lightgreen", linewidth = 1) +
  #             geom_point(color = "lightgreen", size = 2) +
  #             labs(title = "Recruitment Options", x = "G", y = "R") +
  #             theme_minimal() + ylim(0, 700) + xlim(0, 275)+
  #             theme(plot.title = element_text(size = 5),
  #                   axis.text.y = element_text(size = 5),
  #                   axis.text.x = element_text(size = 5))
  #
  # plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
  #             geom_line(color = "skyblue", linewidth = 1) +
  #             geom_point(color = "skyblue", size = 2) +
  #             labs(title = "Variance of Beta Estimates",x = "G",y =
   ↪  "Variance of Beta Estimates") +
  #             theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
  #             theme(plot.title = element_text(size = 5),
  #                   axis.text.y = element_text(size = 5),
  #                   axis.text.x = element_text(size = 5))
  #
  # plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
  #             geom_line(color = "purple", linewidth = 1) +
  #             geom_point(color = "purple", size = 2) +
  #             labs(title = "Mean SE Squared",x = "G",y = "Mean SE
   ↪  Squared") +
  #             theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
  #             theme(plot.title = element_text(size = 5),
  #                   axis.text.x = element_text(size = 5),
  #                   axis.text.y = element_text(size = 5))
  #
  # header <- textGrob(paste0(method,', ',param.name,'=',param),
  #                    gp = gpar(fontsize = 7, fontface = "bold"))
  #
  # grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow =
   ↪  1),heights = c(0.05, 1))
#}
```

```r
method ="poisson"
ParamVary(param.ls, param.name, method,
          relative.cost=2, alpha=5, beta=3, gamma=2, B=5000)
#result.C1.poisson <- MeasureGen(param.ls, param.name, method)
GR.C1.poisson <- GRGen(param.ls, param.name, method)

GR.pair.C1 <- data.frame(
  Param = param.ls,
  Normal = unlist(GR.C1.normal),
  Poisson = unlist(GR.C1.poisson))
knitr::kable(GR.pair.C1,caption = "(G,R) pair, vary c1")

# for(i in 1:length(param.ls)){
#   param = param.ls[i]
  # tbl <- result.C1.poisson[[i]]
  #
  # plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
  #             geom_line(color = "lightgreen", linewidth = 1) +
  #             geom_point(color = "lightgreen", size = 2) +
  #             labs(title = "Recruitment Options", x = "G", y = "R") +
  #             theme_minimal() + ylim(0, 700) + xlim(0, 275)+
  #             theme(plot.title = element_text(size = 5),
  #                   axis.text.x = element_text(size = 5),
  #                   axis.text.y = element_text(size = 5))
  #
  # plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
  #             geom_line(color = "skyblue", linewidth = 1) +
  #             geom_point(color = "skyblue", size = 2) +
  #             labs(title = "Variance of Beta Estimates",x = "G",y =
   ↪  "Variance of Beta Estimates") +
  #             theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
  #             theme(plot.title = element_text(size = 5),
  #                   axis.text.x = element_text(size = 5),
  #                   axis.text.y = element_text(size = 5))
  #
  # plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
  #             geom_line(color = "purple", linewidth = 1) +
  #             geom_point(color = "purple", size = 2) +
  #             labs(title = "Mean SE Squared",x = "G",y = "Mean SE
   ↪  Squared") +
```

```
  #             theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
  #             theme(plot.title = element_text(size = 5),
  #                   axis.text.x = element_text(size = 5),
  #                   axis.text.y = element_text(size = 5))
  #
  # header <- textGrob(paste0(method,', ',param.name,'=',param),
  #                    gp = gpar(fontsize = 7, fontface = "bold"))
  #
  # grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow =
   ↪  1),heights = c(0.05, 1))
#}
param.name = "relative.cost"
param.ls = c(4, 5, 10)

method = "normal"
ParamVary(param.ls, param.name, method,
          C1=20, alpha=5, beta=3, gamma=2, B=5000)
GR.rc.normal <- GRGen(param.ls, param.name, method)
#result.rc.normal <- MeasureGen(param.ls, param.name, method)
# for(i in 1:length(param.ls)){
#   param = param.ls[i]
#   tbl <- result.rc.normal[[i]]
#
#    plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
#             geom_line(color = "lightgreen", linewidth = 1) +
#             geom_point(color = "lightgreen", size = 2) +
#             labs(title = "Recruitment Options", x = "G", y = "R") +
#             theme_minimal() + ylim(0, 700) + xlim(0, 275)+
#             theme(plot.title = element_text(size = 5),
#                   axis.text.x = element_text(size = 5),
#                   axis.text.y = element_text(size = 5))
#
#   plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
#             geom_line(color = "skyblue", linewidth = 1) +
#             geom_point(color = "skyblue", size = 2) +
#             labs(title = "Variance of Beta Estimates",x = "G",y =
 ↪  "Variance of Beta Estimates") +
#             theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
#             theme(plot.title = element_text(size = 5),
#                   axis.text.x = element_text(size = 5),
#                   axis.text.y = element_text(size = 5))
```

```
#
#   plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
#               geom_line(color = "purple", linewidth = 1) +
#               geom_point(color = "purple", size = 2) +
#               labs(title = "Mean SE Squared",x = "G",y = "Mean SE
 ↪  Squared") +
#               theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   header <- textGrob(paste0(method,', ',param.name,'=',param),
#                      gp = gpar(fontsize = 7, fontface = "bold"))
#
#   grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow =
 ↪  1),heights = c(0.05, 1))
# }

method ="poisson"
ParamVary(param.ls, param.name, method,
          C1=20, alpha=5, beta=3, gamma=2, B=5000)
GR.rc.poisson <- GRGen(param.ls, param.name, method)
GR.pair.rc <- data.frame(
  Param = param.ls,
  Normal = unlist(GR.rc.normal),
  Poisson = unlist(GR.rc.poisson))
knitr::kable(GR.pair.rc, caption = "(G,R) pair, vary relative cost")
#result.rc.poisson <- MeasureGen(param.ls, param.name, method)
# for(i in 1:length(param.ls)){
#   param = param.ls[i]
#   tbl <- result.rc.poisson[[i]]
#
#     plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
#               geom_line(color = "lightgreen", linewidth = 1) +
#               geom_point(color = "lightgreen", size = 2) +
#               labs(title = "Recruitment Options", x = "G", y = "R") +
#               theme_minimal() + ylim(0, 700) + xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
```

```
#   plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
#               geom_line(color = "skyblue", linewidth = 1) +
#               geom_point(color = "skyblue", size = 2) +
#               labs(title = "Variance of Beta Estimates",x = "G",y =
 ↪  "Variance of Beta Estimates") +
#               theme_minimal() + ylim(0, 15) + xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
#               geom_line(color = "purple", linewidth = 1) +
#               geom_point(color = "purple", size = 2) +
#               labs(title = "Mean SE Squared",x = "G",y = "Mean SE
 ↪  Squared") +
#               theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   header <- textGrob(paste0(method,', ',param.name,'=',param),
#                      gp = gpar(fontsize = 7, fontface = "bold"))
#
#   grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow =
 ↪  1),heights = c(0.05, 1))
# }
param.name = "alpha"
param.ls = c(5,6,7)

method = "normal"#should not matter
ParamVary(param.ls, param.name, method,
          relative.cost=2, C1=20, beta=3, gamma=2, B=5000)
GR.alpha.normal <- GRGen(param.ls, param.name, method)
#result.alpha.normal <- MeasureGen(param.ls, param.name, method)
# for(i in 1:length(param.ls)){
#   param = param.ls[i]
#   tbl <- result.alpha.normal[[i]]
#
#     plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
#               geom_line(color = "lightgreen", linewidth = 1) +
#               geom_point(color = "lightgreen", size = 2) +
```

```
#               labs(title = "Recruitment Options", x = "G", y = "R") +
#               theme_minimal() + ylim(0, 700) + xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
#               geom_line(color = "skyblue", linewidth = 1) +
#               geom_point(color = "skyblue", size = 2) +
#               labs(title = "Variance of Beta Estimates",x = "G",y =
 ↪  "Variance of Beta Estimates") +
#               theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
#               geom_line(color = "purple", linewidth = 1) +
#               geom_point(color = "purple", size = 2) +
#               labs(title = "Mean SE Squared",x = "G",y = "Mean SE
 ↪  Squared") +
#               theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   header <- textGrob(paste0(method,', ',param.name,'=',param),
#                      gp = gpar(fontsize = 7, fontface = "bold"))
#
#   grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow =
 ↪  1),heights = c(0.05, 1))
# }

method ="poisson"
ParamVary(param.ls, param.name, method,
          relative.cost=2, C1=20, beta=3, gamma=2, B=5000)
GR.alpha.poisson <- GRGen(param.ls, param.name, method)
GR.pair.alpha <- data.frame(
  Param = param.ls,
  Normal = unlist(GR.alpha.normal),
  Poisson = unlist(GR.alpha.poisson))
```

```
knitr::kable(GR.pair.alpha, caption = "(G,R) pair, vary alpha")
#result.alpha.poisson <- MeasureGen(param.ls, param.name, method)
# for(i in 1:length(param.ls)){
#   param = param.ls[i]
#   tbl <- result.alpha.poisson[[i]]
#
#   plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
#               geom_line(color = "lightgreen", linewidth = 1) +
#               geom_point(color = "lightgreen", size = 2) +
#               labs(title = "Recruitment Options", x = "G", y = "R") +
#               theme_minimal() + ylim(0, 700) + xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
#               geom_line(color = "skyblue", linewidth = 1) +
#               geom_point(color = "skyblue", size = 2) +
#               labs(title = "Variance of Beta Estimates",x = "G",y =
 ↪  "Variance of Beta Estimates") +
#               theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
#               geom_line(color = "purple", linewidth = 1) +
#               geom_point(color = "purple", size = 2) +
#               labs(title = "Mean SE Squared",x = "G",y = "Mean SE
 ↪  Squared") +
#               theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   header <- textGrob(paste0(method,', ',param.name,'=',param),
#                      gp = gpar(fontsize = 7, fontface = "bold"))
#
#   grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow =
 ↪  1),heights = c(0.05, 1))
# }
```

```r
param.name = "beta"
param.ls = c(1,3,5)

method = "normal"
ParamVary(param.ls, param.name, method,
          relative.cost=2, C1=20, alpha=5, gamma=2, B=5000)
result.beta.normal <- MeasureGen(param.ls, param.name, method)
GR.beta.normal <- GRGen(param.ls, param.name, method)

for(i in 1:length(param.ls)){
  param = param.ls[i]
  tbl <- result.beta.normal[[i]]

    plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
              geom_line(color = "lightgreen", linewidth = 1) +
              geom_point(color = "lightgreen", size = 2) +
              labs(title = "Recruitment Options", x = "G", y = "R") +
              theme_minimal() + ylim(0, 700) + xlim(0, 275)+
              theme(plot.title = element_text(size = 5),
                    axis.text.x = element_text(size = 5),
                    axis.text.y = element_text(size = 5))

  plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
              geom_line(color = "skyblue", linewidth = 1) +
              geom_point(color = "skyblue", size = 2) +
              labs(title = "Variance of Beta Estimates",x = "G",y =
               ↪  "Variance of Beta Estimates") +
              theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
              theme(plot.title = element_text(size = 5),
                    axis.text.x = element_text(size = 5),
                    axis.text.y = element_text(size = 5))

  plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
              geom_line(color = "purple", linewidth = 1) +
              geom_point(color = "purple", size = 2) +
              labs(title = "Mean SE Squared",x = "G",y = "Mean SE
               ↪  Squared") +
              theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
              theme(plot.title = element_text(size = 5),
                    axis.text.y = element_text(size = 5),
                    axis.text.x = element_text(size = 5))
```

```
  header <- textGrob(paste0(method,', ',param.name,'=',param),
                     gp = gpar(fontsize = 7, fontface = "bold"))

  grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow = 1),heights
   ↪  = c(0.05, 1))
}

method ="poisson"
ParamVary(param.ls, param.name, method,
          relative.cost=2, C1=20, alpha=5, gamma=2, B=5000)
result.beta.poisson <- MeasureGen(param.ls, param.name, method)
GR.beta.poisson <- GRGen(param.ls, param.name, method)

for(i in 1:length(param.ls)){
  param = param.ls[i]
  tbl <- result.beta.poisson[[i]]

    plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
             geom_line(color = "lightgreen", linewidth = 1) +
             geom_point(color = "lightgreen", size = 2) +
             labs(title = "Recruitment Options", x = "G", y = "R") +
             theme_minimal() + ylim(0, 700) + xlim(0, 275)+
             theme(plot.title = element_text(size = 5),
                   axis.text.x = element_text(size = 5),
                   axis.text.y = element_text(size = 5))

  plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
             geom_line(color = "skyblue", linewidth = 1) +
             geom_point(color = "skyblue", size = 2) +
             labs(title = "Variance of Beta Estimates",x = "G",y =
              ↪  "Variance of Beta Estimates") +
             theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
             theme(plot.title = element_text(size = 5),
                   axis.text.x = element_text(size = 5),
                   axis.text.y = element_text(size = 5))

  plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
             geom_line(color = "purple", linewidth = 1) +
             geom_point(color = "purple", size = 2) +
```

```
              labs(title = "Mean SE Squared",x = "G",y = "Mean SE
               ↪  Squared") +
              theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
              theme(plot.title = element_text(size = 5),
                    axis.text.x = element_text(size = 5),
                    axis.text.y = element_text(size = 5))

  header <- textGrob(paste0(method,', ',param.name,'=',param),
                     gp = gpar(fontsize = 7, fontface = "bold"))

  grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow = 1),heights
   ↪  = c(0.05, 1))
}

GR.pair.beta <- data.frame(
  Param = param.ls,
  Normal = unlist(GR.beta.normal),
  Poisson = unlist(GR.beta.poisson))
knitr::kable(GR.pair.beta, caption = "(G,R) pair, vary beta")
param.name = "gamma"
param.ls = c(1,2,3)
method = "normal"
ParamVary(param.ls, param.name, method,
          relative.cost=2, C1=20, alpha=5, beta=3, B=5000)
result.gamma.normal <- MeasureGen(param.ls, param.name, method)
GR.gamma.normal <- GRGen(param.ls, param.name, method)

for(i in 1:length(param.ls)){
  param = param.ls[i]
  tbl <- result.gamma.normal[[i]]

    plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
              geom_line(color = "lightgreen", linewidth = 1) +
              geom_point(color = "lightgreen", size = 2) +
              labs(title = "Recruitment Options", x = "G", y = "R") +
              theme_minimal() + ylim(0, 700) + xlim(0, 275)+
              theme(plot.title = element_text(size = 5),
                    axis.text.x = element_text(size = 5),
                    axis.text.y = element_text(size = 5))

  plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
```

```
              geom_line(color = "skyblue", linewidth = 1) +
              geom_point(color = "skyblue", size = 2) +
              labs(title = "Variance of Beta Estimates",x = "G",y =
               ↪  "Variance of Beta Estimates") +
              theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
              theme(plot.title = element_text(size = 5),
                    axis.text.x = element_text(size = 5),
                    axis.text.y = element_text(size = 5))

  plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
              geom_line(color = "purple", linewidth = 1) +
              geom_point(color = "purple", size = 2) +
              labs(title = "Mean SE Squared",x = "G",y = "Mean SE
               ↪  Squared") +
              theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
              theme(plot.title = element_text(size = 5),
                    axis.text.x = element_text(size = 5),
                    axis.text.y = element_text(size = 5))

  header <- textGrob(paste0(method,', ',param.name,'=',param),
                     gp = gpar(fontsize = 7, fontface = "bold"))

  grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow = 1),heights
   ↪  = c(0.05, 1))
}

method ="poisson"
ParamVary(param.ls, param.name, method,
          relative.cost=2, C1=20, alpha=5, beta=3, B=5000)
result.gamma.poisson <- MeasureGen(param.ls, param.name, method)
GR.gamma.poisson <- GRGen(param.ls, param.name, method)

for(i in 1:length(param.ls)){
  param = param.ls[i]
  tbl <- result.gamma.poisson[[i]]

  plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
              geom_line(color = "lightgreen", linewidth = 1) +
              geom_point(color = "lightgreen", size = 2) +
              labs(title = "Recruitment Options", x = "G", y = "R") +
              theme_minimal() + ylim(0, 700) + xlim(0, 275)+
```

```
            theme(plot.title = element_text(size = 5),
                  axis.text.x = element_text(size = 5),
                  axis.text.y = element_text(size = 5))

  plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
            geom_line(color = "skyblue", linewidth = 1) +
            geom_point(color = "skyblue", size = 2) +
            labs(title = "Variance of Beta Estimates",x = "G",y =
             ↪  "Variance of Beta Estimates") +
            theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
            theme(plot.title = element_text(size = 5),
                  axis.text.x = element_text(size = 5),
                  axis.text.y = element_text(size = 5))

  plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
            geom_line(color = "purple", linewidth = 1) +
            geom_point(color = "purple", size = 2) +
            labs(title = "Mean SE Squared",x = "G",y = "Mean SE
             ↪  Squared") +
            theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
            theme(plot.title = element_text(size = 5),
                  axis.text.x = element_text(size = 5),
                  axis.text.y = element_text(size = 5))

  header <- textGrob(paste0(method,', ',param.name,'=',param),
                     gp = gpar(fontsize = 7, fontface = "bold"))

  grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow = 1),heights
   ↪  = c(0.05, 1))
}

GR.pair.gamma <- data.frame(
  Param = param.ls,
  Normal = unlist(GR.gamma.normal),
  Poisson = unlist(GR.gamma.poisson))
knitr::kable(GR.pair.gamma, caption = "(G,R) pair, vary gamma")
param.name = "B"
param.ls = c(3000, 4000, 5000)

method = "normal"
ParamVary(param.ls, param.name, method,
```

```
          relative.cost=2, C1=20, alpha=5, beta=3, gamma=2)
GR.B.normal <- GRGen(param.ls, param.name, method)
#result.B.normal <- MeasureGen(param.ls, param.name, method)
# for(i in 1:length(param.ls)){
#   param = param.ls[i]
#   tbl <- result.B.normal[[i]]
#
#   plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
#               geom_line(color = "lightgreen", linewidth = 1) +
#               geom_point(color = "lightgreen", size = 2) +
#               labs(title = "Recruitment Options", x = "G", y = "R") +
#               theme_minimal() + ylim(0, 700) + xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
#               geom_line(color = "skyblue", linewidth = 1) +
#               geom_point(color = "skyblue", size = 2) +
#               labs(title = "Variance of Beta Estimates",x = "G",y =
 ↪  "Variance of Beta Estimates") +
#               theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
#               geom_line(color = "purple", linewidth = 1) +
#               geom_point(color = "purple", size = 2) +
#               labs(title = "Mean SE Squared",x = "G",y = "Mean SE
 ↪  Squared") +
#               theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.x = element_text(size = 5),
#                     axis.text.y = element_text(size = 5))
#
#   header <- textGrob(paste0(method,', ',param.name,'=',param),
#                      gp = gpar(fontsize = 7, fontface = "bold"))
#
#   grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow =
 ↪  1),heights = c(0.05, 1))
```

```r
# }

method ="poisson"
ParamVary(param.ls, param.name, method,
          relative.cost=2, C1=20, alpha=5, beta=3, gamma=2)
GR.B.poisson <- GRGen(param.ls, param.name, method)
GR.pair.B <- data.frame(
  Param = param.ls,
  Normal = unlist(GR.B.normal),
  Poisson = unlist(GR.B.poisson))
knitr::kable(GR.pair.B, caption = "(G,R) pair, vary B")

# result.B.poisson <- MeasureGen(param.ls, param.name, method)
# for(i in 1:length(param.ls)){
#   param = param.ls[i]
#   tbl <- result.B.poisson[[i]]
#
#   plot1 <- ggplot(tbl, aes(x = G, y = R, group = 1)) +
#               geom_line(color = "lightgreen", linewidth = 1) +
#               geom_point(color = "lightgreen", size = 2) +
#               labs(title = "Recruitment Options", x = "G", y = "R") +
#               theme_minimal() + ylim(0, 700) + xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.y = element_text(size = 5),
#                     axis.text.x = element_text(size = 5))
#
#   plot2 <- ggplot(tbl, aes(x = G, y = Var_Beta_Est, group = 1)) +
#               geom_line(color = "skyblue", linewidth = 1) +
#               geom_point(color = "skyblue", size = 2) +
#               labs(title = "Variance of Beta Estimates",x = "G",y =
 ↪  "Variance of Beta Estimates") +
#               theme_minimal() + ylim(0, 15)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.y = element_text(size = 5),
#                     axis.text.x = element_text(size = 5))
#
#   plot3 <- ggplot(tbl, aes(x = G, y = Mean_SE_Sq, group = 1)) +
#               geom_line(color = "purple", linewidth = 1) +
#               geom_point(color = "purple", size = 2) +
#               labs(title = "Mean SE Squared",x = "G",y = "Mean SE
 ↪  Squared") +
```

```
#               theme_minimal() + ylim(0, 10)+ xlim(0, 275)+
#               theme(plot.title = element_text(size = 5),
#                     axis.text.y = element_text(size = 5),
#                     axis.text.x = element_text(size = 5))
#
#   header <- textGrob(paste0(method,', ',param.name,'=',param),
#                      gp = gpar(fontsize = 7, fontface = "bold"))
#
#   grid.arrange(header,arrangeGrob(plot1, plot2, plot3, nrow =
 ↪  1),heights = c(0.05, 1))
# }
```