CDsampling: An R Package for Constrained D-Optimal Sampling in Paid Research Studies

by Yifei Huang, Liping Tong, and Jie Yang

Abstract In the context of paid research studies and clinical trials, budget considerations often require patient sampling from available populations, which comes with inherent constraints. We introduce the R package CDsampling, which is the first to our knowledge to integrate optimal design theories within the framework of constrained sampling. This package offers the possibility to find both D-optimal approximate and exact allocations for samplings with or without constraints. Additionally, it provides functions to find constrained uniform sampling as a robust sampling strategy when the model information is limited. To demonstrate its efficacy, we provide simulated examples and a real-data example with datasets embedded in the package and compare them with classical sampling methods. Furthermore, the CDsampling package revisits the theoretical results of the Fisher information matrix for generalized linear models (including the regular linear regression model) and multinomial logistic models, offering functions for its computation.

1 Introduction

Paid research studies are essential for determining the influence of new interventions or treatments and for providing quantitative evidence in various domains, such as healthcare, psychology, and politics. However, conducting such studies often involves a limited budget and a large pool of potential volunteers, which poses a challenge in selecting the best sample to meet the research objectives. Poor samples may result in biased or inaccurate estimates, low statistical power, and even misleading conclusions. Therefore, finding a good sampling strategy is crucial for researchers and practitioners.

Consider a constrained sampling problem commonly encountered in paid research studies. Suppose, in a research study to evaluate a new intervention, $N = 500$ volunteers register to participate. Upon registration, the investigators collect basic demographic information, such as gender (female or male) and age groups ($18 \sim 25$, $26 \sim 64$, 65 and above) for each volunteer. Treating gender and age as stratification factors, we obtain $m = 6$ subgroups. However, due to budget limitations, the study could only accommodate $n = 200$ participants. Let N_i denote the frequency of volunteers within the ith subgroup, where $i = 1, \ldots, m$. We call the integer number of participants sampled from each subgroup, n_i, the exact allocation, and the corresponding proportion n_i/N_i the approximate allocation. The goal is to select a sample of 200 participants $\mathbf{n} = (n_1, \ldots, n_m)$, such that $\sum_{i=1}^{m} n_i = 200$ from the pool of $N = 500$ volunteers to evaluate the intervention effect most accurately, subject to the constraint that no subgroup is oversampled beyond the number of available volunteers within that subgroup, that is, $n_i \leq N_i$. This constraint is the most commonly encountered in paid research studies (see Section 3.1 for details, and for constraints of other forms, please refer to Section 3.2).

Commonly used sampling strategies include simple random sampling, stratified sampling, and cluster sampling. Simple random sampling is the most straightforward form of probability sampling, where each element has an equal chance of being selected (Tillé, 2006). Proportionally stratified sampling involves dividing the population into homogeneous subgroups, such as gender and age groups, and applying random sampling to sample proportionally within each subgroup (Lohr, 2019). Cluster sampling, on the other hand, splits the population into heterogeneous clusters, for example, based on the locations of volunteers, and randomly selects some clusters as the sample units. However, these methods have their drawbacks. Cluster sampling is relatively low-cost but less precise than simple random sampling. Stratified sampling can produce more efficient estimators of population

means, but it requires finding well-defined and relevant subgroups that cover the entire population (Levy and Lemeshow, 2008). Moreover, these existing methods are based on assumptions that may not hold if model estimation is the primary goal of the paid research study.

In the proposed **CDsampling** package, we implement the sampling strategy based on optimal design theory (with main functions `liftone_GLM()`, `liftone_constrained_GLM()`, `liftone_MLM()`, and `liftone_constrained_MLM()`), which can improve the accuracy of the intervention effect estimation. Optimal design theory is a branch of experimental design that aims to find the best allocation of experimental units to achieve a specific optimality criterion such as minimizing the variance of estimation or equivalently maximizing the information obtained from the design. For example, D-optimality maximizes the determinant of the information matrix, which minimizes the estimators' expected volume of the joint confidence ellipsoid. A-optimality minimizes the trace of the inverse information matrix, equivalent to minimizing the average of the variances of the estimators. E-optimality minimizes the maximum eigenvalue of the inverse information matrix, which minimizes the largest expected semi-axis of the confidence ellipsoid and protects against the worst possible case (Fedorov, 1972; Atkinson et al., 2007; Yang and Stufken, 2012; Fedorov and Leonov, 2014). In this paper, we focus on D-optimality due to its overall good performance and mathematical simplicity (Zocchi and Atkinson, 1999; Atkinson et al., 2007). According to Huang et al. (2025), the constrained D-optimal sampling strategies work well for paid research studies or clinical trials. To implement the recommended sampling strategy, we develop an R package called **CDsampling** (namely, Constrained D-optimal sampling), available on CRAN at `https://cran.r-project.org/package=CDsampling`. To the best of our knowledge, **CDsampling** is the first R package offering a sampling tool with constraints in paid research studies based on optimal design theory. Package **sampling** implements random samples using different sampling schemes (Tillé and Matei, 2016). The package also provides functions to obtain (generalized) calibration weights, different estimators, as well as some variance estimators. Package **SamplingStrata** determines the best stratification for a population frame that ensures the minimum sample cost with precision thresholds (Barcaroli, 2014). On the other hand, there are existing R packages for optimal designs. Package **AlgDesign** finds exact and approximate allocations of optimal designs under D-, A-, and I-criteria (Wheeler and Braun, 2019). Package **OptimalDesign** enables users to compute D-, A-, I-, and c-efficient designs with or without replications, restricted design spaces, and under multiple linear constraints (Harman et al., 2016). Package **acebayes** finds Bayesian optimal experimental designs with approximate coordinate exchange algorithm (Overstall et al., 2020, 2018). Package **OPDOE** provides functions for optimal designs with polynomial regression models and ANOVA (Grömping, 2011). These packages provide programming tools for finding samplings or optimal designs under different criteria and models. However, they do not address the specific challenges of practical feasibility in the constrained sampling scheme of paid research studies. The proposed **CDsampling** package fills this gap by offering an efficient sampling tool to handle general constraints and common parametric models in paid research studies.

2 Method

2.1 Constrained lift-one algorithm

The lift-one algorithm was initially proposed by Yang et al. (2016) to find D-optimal designs with binary responses. This was extended to generalized linear models (GLMs) by Yang and Mandal (2015) and subsequently adapted for cumulative link models by Yang et al. (2017). The methodology was further extended to multinomial logit models (MLMs) by Bu et al. (2020).

Figure 1 provides a concise summary of the lift-one algorithm applied to general parametric models. The detailed algorithm is provided in Algorithm 3 from the Supplementary Material of Huang et al. (2025). We consider a general study or experiment involving co-

variates $\mathbf{x}_i = (x_{i1}, \ldots, x_{id})^\top$, for $i = 1, \ldots, m$, referred to as experimental settings. In paid research studies, these covariates could be the stratification factors such as the gender and age groups in our motivating example. Here, $m \geq 2$ denotes the number of experimental settings, which corresponds to $m = 6$, the number of gender and age groups in the motivating example. Suppose the responses follow a parametric model $M(\mathbf{x}_i, \boldsymbol{\theta})$, where $\boldsymbol{\theta} \subseteq \mathbb{R}^p$ with $p \geq 2$. Under regularity conditions, the Fisher information matrix of the experiment design can be written as $\mathbf{F} = \sum_{i=1}^{m} n_i \mathbf{F}_i$, where $\mathbf{F}_i, i = 1, \ldots, m$ is the Fisher information matrix at $\mathbf{x}_i$ and n_i is the number of subjects allocated to the ith experimental setting. In the setting of paid research studies, n_i corresponds to the number of subjects sampled from the ith subgroup. We usually call $\mathbf{n} = (n_1, \ldots, n_m)^\top$ the exact allocation, where $n = \sum_{i=1}^{m} n_i$, and $\mathbf{w} = (w_1, \ldots, w_m)^\top = (n_1/n, \ldots, n_m/n)^\top$ the approximate allocation, where $w_i \geq 0$ and $\sum_{i=1}^{m} w_i = 1$ (Kiefer, 1974; Pukelsheim, 2006; Atkinson et al., 2007). The approximate allocation is theoretically more tractable, while the exact allocation is practically more useful. In the statistical literature, approximate allocations are more commonly discussed (Kiefer, 1974). The D-optimal design aims to find the optimal allocation that maximizes $f(\mathbf{w}) = |\mathbf{F}(\mathbf{w})| = |\sum_{i=1}^{m} w_i \mathbf{F}_i|$. The lift-one algorithm simplifies a complex multivariate optimization problem into a sequence of simpler univariate optimization problems. This is achieved by "lifting" and optimizing one weight w_i, within the approximate allocation vector $\mathbf{w}$. Specifically, in step 3° of the algorithm depicted in Figure 1, the determinant of the Fisher information matrix function concerning the lift-one variable is expressed as $f(z) = f(\mathbf{w}_i(z))$ where the variable z substitutes for w_i in allocation $\mathbf{w}$, and the remaining weights are adjusted proportionally, denoted as $\mathbf{w}_i$. The updated weight vector is given by

$$\mathbf{w}_i(z) = \left(\frac{1-z}{1-w_i} w_1, \ldots, \frac{1-z}{1-w_i} w_{i-1}, z, \frac{1-z}{1-w_i} w_{i+1}, \ldots, \frac{1-z}{1-w_i} w_m \right)^\top.$$

The allocation that results from the convergence in step 6° of the algorithm is identified as the D-optimal approximate allocation.

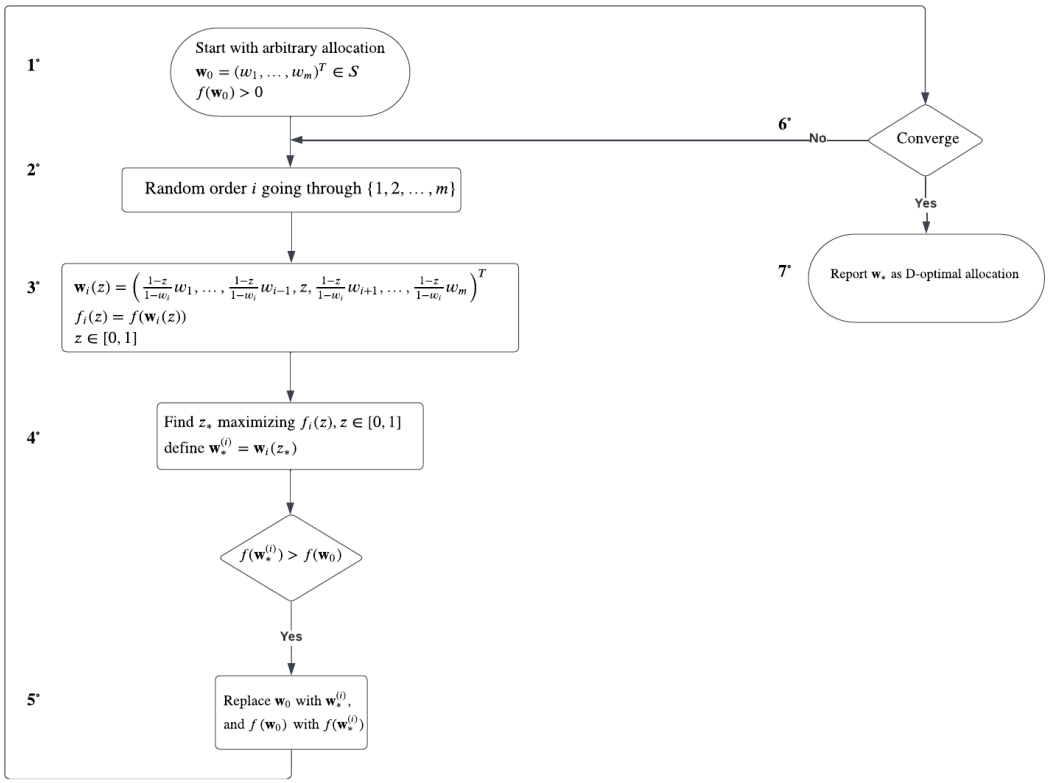

Figure 1: The framework of lift-one algorithm.

In the context of paid research studies, budgetary limitations often necessitate the se-

lection of a subset of participants. We consider the sampling of n subjects from a larger population of N, where $n < N$. A typical constraint in such studies is $n_i \leq N_i$, where N_i represents the number of available subjects from the ith experimental setting group. This effectively places an upper bound on the sample size for each subgroup (or stratum), ensuring the sampling process does not overdraw from any subgroup. Additional constraints may include $n_1 + n_2 + n_3 + n_4 \leq 392$ (see the MLM example in Section 3.2), $4n_1 \geq n_3$ (see Example S2.2 in Supplementary Material Section S2), etc. The constrained lift-one algorithm seeks the approximate allocation $\mathbf{w}$ that maximizes $|\mathbf{F}(\mathbf{w})|$, on a collection of feasible approximate allocations $S \subseteq S_0$ where

$$S_0 := \{(w_1, \ldots, w_m)^\top \in \mathbb{R}^m \mid w_i \geq 0, i = 1, \ldots, m; \sum_{i=1}^{m} w_i = 1\}.$$

The set S is presumed to be either a closed convex set or a finite union of closed convex sets. The framework of the constrained lift-one algorithm is provided in Figure 2. The details of the algorithm can be found in Algorithm 1 of Huang et al. (2025). In the constrained lift-one algorithm, the search for the optimal lift-one weight z in step 3° of Figure 2 is confined within the interval of $[r_{i1}, r_{i2}]$ (Example S2.2 in Supplementary Material Section S2 and Section S.3 in Huang et al. (2025) for details of finding $[r_{i1}, r_{i2}]$). To ensure that the resulting allocation is D-optimal, two additional decision steps, labeled steps 7° and 8°, are incorporated into the algorithm. The reported allocation in step 10° is the constrained D-optimal approximate allocation for the study. To illustrate the difference between the lift-one algorithm and the constrained lift-one algorithm, we provide examples in Supplementary Material Section S2.

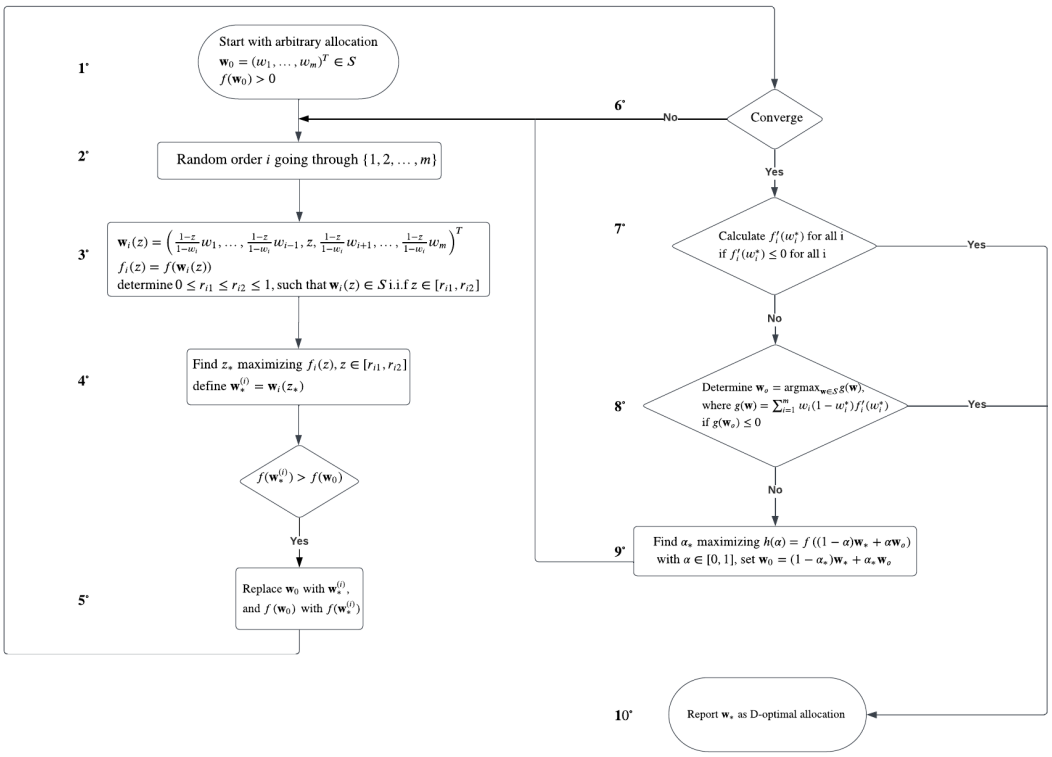

Figure 2: The framework of constrained lift-one algorithm.

Upon obtaining the approximate allocation, we may employ the constrained approximate to exact allocation algorithm outlined in Figure 3 for the conversion of a real-valued approximate allocation to an integer-valued exact allocation. The full details of the algorithm are in Algorithm 2 of Huang et al. (2025). The algorithm begins by assigning a floor integer value to all subgroups $n_i = \lfloor N w_i \rfloor$. Subsequently, each remaining subject is added to the corresponding group in a manner that maximizes the determinant of the Fisher information matrix. This transformation provides a more pragmatic application in the actual sampling

process within paid research studies.

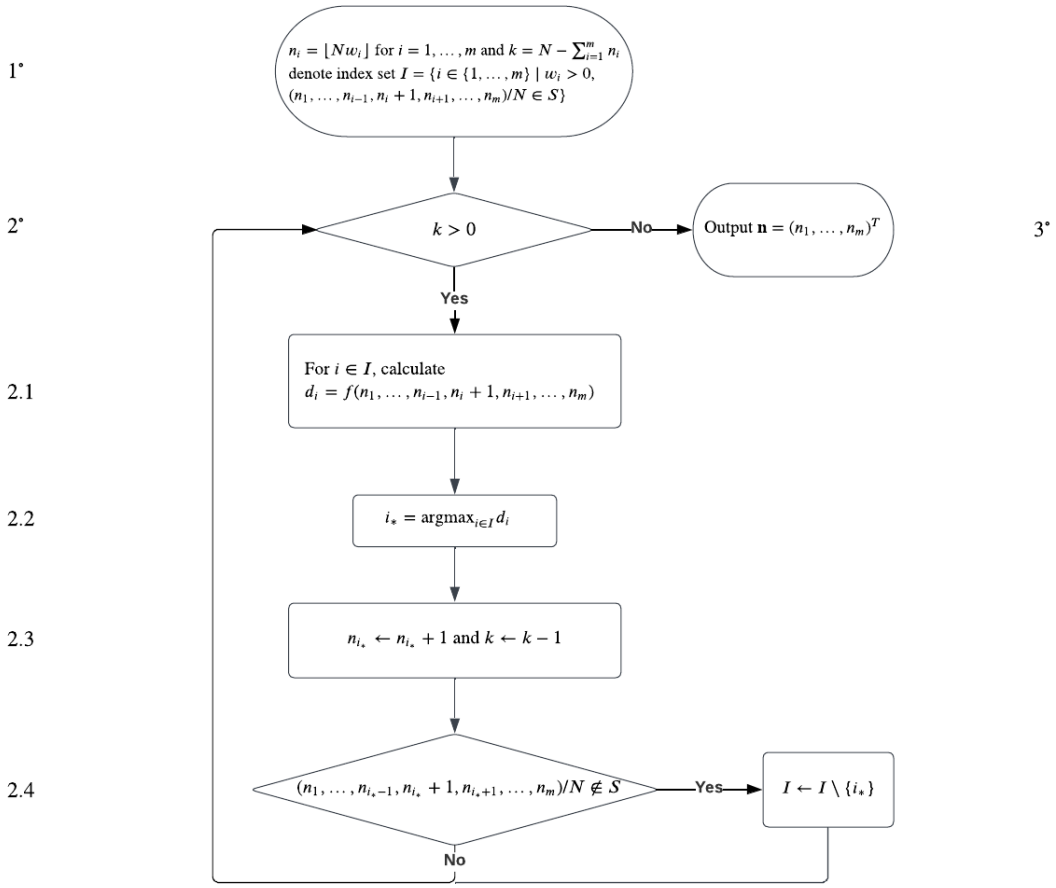

Figure 3: The framework of constrained approximate to exact algorithm.

The calculation of the Fisher information matrix $\mathbf{F}$ and the subsequent maximization of its determinant $|\mathbf{F}|$ are key steps in both the constrained and original (unconstrained) lift-one algorithms. The theoretical details and functions for the computation of $\mathbf{F}$ are provided in Sections 2.2 and Section 2.3.

2.2 Fisher information for generalized linear models

The generalized linear model (GLM) broadens the scope of the traditional linear model. In the standard approach, the response variable is expected to change in direct proportion to the covariate. Yet, this isn't always a practical assumption. Take binary outcomes, for instance, the classic linear model falls short here. Similarly, it's unsuitable for positive-only data like count data. Nelder and Wedderburn (1972) expanded the model to accommodate a wider range of applications.

For a GLM, we assume that response variables $Y_1, \ldots, Y_n$ are independent and from the exponential family. Then we have (Dobson and Barnett, 2018; McCullagh and Nelder, 1989)

$$E(Y_i \mid \mathbf{X}_i) = \mu_i, \quad g(\mu_i) = \eta_i = \mathbf{X}_i^\top \boldsymbol{\beta}$$

where g is a link function, $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)^\top$ is the parameter vector of interest, and μ_i is the conditional expectation of response Y_i given predictors $\mathbf{X}_i = \mathbf{h}(\mathbf{x}_i) = (h_1(\mathbf{x}_i), \ldots, h_p(\mathbf{x}_i))^\top$, where $i = 1, \ldots, n$ with known and deterministic predictor functions $\mathbf{h} = (h_1, \ldots, h_p)^\top$. There are various link functions that could be used, for example, logit link $\eta_i = \log \frac{\mu_i}{1-\mu_i}$; probit link $\eta_i = \Phi^{-1}(\mu_i)$, where $\Phi(\cdot)$ is the normal cumulative distribution function; and complementary log-log link $\eta_i = \log\{-\log(1 - \mu_i)\}$. Regular linear models

can be considered as GLMs with the identity link function and normal responses. Suppose we have a design with m design points $\mathbf{x}_1, \ldots, \mathbf{x}_m$ that has an exact allocation $(n_1, \ldots, n_m)$ where $\sum_i n_i = n$ and corresponding approximate allocation $(w_1, \ldots, w_m) = (\frac{n_1}{n}, \ldots, \frac{n_m}{n})$. The Fisher information matrix $\mathbf{F}$ under GLMs can be written as (McCullagh and Nelder, 1989; Khuri et al., 2006; Stufken and Yang, 2012; Yang et al., 2016):

$$\mathbf{F} = n\mathbf{X}^\top \mathbf{W}\mathbf{X} = n \sum_{i=1}^{m} w_i \nu_i \mathbf{X}_i \mathbf{X}_i^\top \tag{1}$$

where $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_m)^\top$ is the $m \times p$ design matrix with $\mathbf{X}_i = \mathbf{h}(\mathbf{x}_i)$, $\mathbf{W} = \text{diag}\{w_1\nu_1, \ldots, w_m\nu_m\}$ is a diagonal matrix with $\nu_i = \frac{1}{\text{Var}(Y_i)}(\frac{\partial \mu_i}{\partial \eta_i})^2$. Here, ν_i represents how much information the design point $\mathbf{x}_i$ contains. The explicit formats of ν_i with different response distributions and link functions can be found in Table 5 of the Supplementary Material of Huang et al. (2025). To calculate the Fisher information matrix $\mathbf{F}$ given the approximate allocation $\mathbf{w}$, we may use the `F_func_GLM()` function in the **CDsampling** package. Additionally, the `W_func_GLM()` function can be used to find the diagonal elements of the matrix $\mathbf{W}$ in the Fisher information matrix (1) of GLM. An example of finding the Fisher information matrix for GLM is provided in Example S1.1 of Supplementary Material.

2.3 Fisher information for multinomial logistic model

The multinomial logistic model (MLM) is an extension of GLM aiming to manage responses that fall into multiple categories, such as rating scales and disease severity levels (Agresti, 2013).

We assume that the responses $\mathbf{Y}_i = (Y_{i1}, \ldots, Y_{iJ})$ follow a multinomial distribution with probabilities $(\pi_{i1}, \ldots, \pi_{iJ})$, where $\pi_{ij} = \mathbb{P}(Y_i = j \mid \mathbf{x}_i)$, $Y_i \in \{1, \ldots, J\}$ is the ith original categorical response, $j = 1, \ldots, J$, and $\sum_j \pi_{ij} = 1$. If the response variables are nominal, in other words, there is no natural ordering among different levels, a commonly used MLM is the baseline-category logit model with the Jth level selected as the baseline category. If the response variable is ordinal, that is, we have a natural ordering of response levels, there are three commonly used MLMs in the literature: cumulative logit model, adjacent logit model, and continuation-ratio logit model (Bu et al., 2020; Dousti Mousavi et al., 2023; Wang and Yang).

In addition to different logit models, the proportional odds (po) assumption is an important concept in MLMs. The po assumption is a parsimonious model assumption, where a model simultaneously uses all $J-1$ logits in a single model with the same coefficients. This means the covariate effect is constant on the odds ratio among different response levels. When the assumption doesn't hold, the model is referred to as a non-proportional odds (npo) model and has more parameters in it. When the assumption only holds for part of the parameters, the model is referred to as a partial proportional odds (ppo) model. Commonly used multinomial logit models with po, npo, or ppo assumptions can be summarized in a unified matrix form (Glonek and McCullagh, 1995; Zocchi and Atkinson, 1999; Bu et al., 2020):

$$\mathbf{C}^\top \log(\mathbf{L}\boldsymbol{\pi}_i) = \mathbf{X}_i\boldsymbol{\theta}$$

where

$$\mathbf{C}^\top = \begin{pmatrix} \mathbf{I}_{J-1} & -\mathbf{I}_{J-1} & \mathbf{0}_{J-1}^\top \\ \mathbf{0}_{J-1}^\top & \mathbf{0}_{J-1}^\top & 1 \end{pmatrix}$$

is a $J \times (2J-1)$ constant matrix, $\mathbf{L}$ is a $(2J-1) \times J$ constant matrix with different formats among the four different multinomial logit models, and $\boldsymbol{\pi}_i = (\pi_{i1}, \ldots, \pi_{iJ})^\top$. The model

matrix $\mathbf{X}_i$ is defined in general as

$$\mathbf{X}_i = \begin{pmatrix} \mathbf{h}_1^\top(\mathbf{x}_i) & & & \mathbf{h}_c^\top(\mathbf{x}_i) \\ & \ddots & & \vdots \\ & & \mathbf{h}_{J-1}^\top(\mathbf{x}_i) & \mathbf{h}_c^\top(\mathbf{x}_i) \\ \mathbf{0}_{p_1}^\top & \dots & \mathbf{0}_{p_{J-1}}^\top & \mathbf{0}_{p_c}^\top \end{pmatrix}_{J\times p}$$

and the parameter $\boldsymbol{\theta} = (\boldsymbol{\beta}_1^\top, \dots, \boldsymbol{\beta}_{J-1}^\top, \boldsymbol{\zeta}^\top)^\top$ consists of $p = p_1 + \cdots + p_{J-1} + p_c$ unknown parameters. Here $\mathbf{h}_j^\top(\cdot) = (h_{j1}(\cdot), \dots, h_{jp_j}(\cdot))$ are known functions to determine p_j predictors associated with unknown parameters $\boldsymbol{\beta}_j = (\beta_{j1}, \dots, \beta_{jp_j})^\top$ in jth response category, and $\mathbf{h}_c^\top(\cdot) = (h_1(\cdot), \dots, h_{p_c}(\cdot))$ are known functions to determine p_c predictors associated with proportional odds parameters $\boldsymbol{\zeta} = (\zeta_1, \dots, \zeta_{p_c})^\top$. If $\mathbf{h}_j^\top(\mathbf{x}_i) \equiv 1$, the model is a po model; if $\mathbf{h}_c^\top(\mathbf{x}_i) \equiv 0$, the model is an npo model.

According to Theorem 2.1 in Bu et al. (2020), the Fisher information matrix under a multinomial logistic regression model with independent observations can be written as

$$\mathbf{F} = \sum_{i=1}^{m} n_i \mathbf{F}_i \tag{2}$$

where $\mathbf{F}_i = (\frac{\partial \boldsymbol{\pi}_i}{\partial \boldsymbol{\theta}^\top})^\top \mathrm{diag}(\boldsymbol{\pi}_i)^{-1} \frac{\partial \boldsymbol{\pi}_i}{\partial \boldsymbol{\theta}^\top}$ with $\frac{\partial \boldsymbol{\pi}_i}{\partial \boldsymbol{\theta}^\top} = (\mathbf{C}^\top \mathbf{D}_i^{-1} \mathbf{L})^{-1}$ and $\mathbf{D}_i = \mathrm{diag}(\mathbf{L}\boldsymbol{\pi}_i)$. Explicit forms of $(\mathbf{C}^\top \mathbf{D}_i^{-1} \mathbf{L})^{-1}$ can be found in Section S.3 of the Supplementary Material of Bu et al. (2020). To calculate the Fisher information matrix $\mathbf{F}$ for the MLM, one may use the function `F_func_MLM()` in the **CDsampling** package. An example of finding the Fisher information matrix for MLM is provided in Example S1.2 of the Supplementary Material.

3 Examples

The methods described in Section 2 are implemented in the proposed R package **CDsampling**. The **CDsampling** package comprises 16 functions, as detailed in Table 1. The primary functions of the **CDsampling** package are `liftone_constrained_GLM()` and `liftone_constrained_MLM()`. Additionally, the package includes the original (unconstrained) lift-one algorithm for general experimental designs, accessible via the `liftone_GLM()` and `liftone_MLM()` functions. Two datasets `trial_data` and `trauma_data` are provided for illustration purposes.

In the remainder of this section, we present two examples to illustrate the usage of the **CDsampling** package for sampling problems in paid research studies, using datasets provided by **CDsampling**. All results in this section were generated using R version 4.3.2 on a macOS Sonoma 14.6.1 system.

3.1 Applications in paid research study: `trial_data` example

The `trial_data` dataset is simulated data for a toy example of paid research studies. This study includes a cohort of 500 patients for a clinical trial, with gender and age as stratification factors. A logistic regression model incorporates these factors as covariates: `gender` (coded as 0 for female and 1 for male) and `age` (coded as two dummy variables *age*_1 and *age*_2 with $(age\_1, age\_2) = (0,0)$ for age group $18 \sim 25$; $(1,0)$ for age group $26 \sim 64$, and $(0,1)$ for age group 65 and above). For simplicity, the study assumes binary gender options and a tripartite age categorization. In total, there are $m = 6$ combinations of covariate factors. In practice, non-binary gender options and a "prefer not to answer" choice may be included to respect gender diversity and protect patient confidentiality. The response Y denotes the treatment's efficacy (0 indicating ineffectiveness, 1 indicating effectiveness). The data is

	Usage	Function
Model	Calculating **W** matrix diagonal elements of generalized linear model (see Section 2.2); providing input for function `liftone_GLM()` and `liftone_constrained_GLM()`.	`W_func_GLM()`
	Calculating Fisher information matrix and its determinant of generalized linear model (see Example S1.1 in Supplementary Material).	`F_func_GLM()`, `Fdet_func_GLM()`
	Calculating Fisher information matrix of multinomial logit model at a specific design point (see Section 2.3); using as input of `liftone_MLM()` and `liftone_constrained_MLM()`.	`Fi_func_MLM()`
	Calculating Fisher information matrix and its determinant of multinomial logit model (see Example S1.2 in Supplementary Material).	`F_func_MLM()`, `Fdet_func_MLM()`
	Using in `approxtoexact_constrained_func()` to find constrained uniform exact allocation.	`Fdet_func_unif()`
Sampling	Finding unconstrained D-optimal approximate allocation for generalized linear model and multinomial logit model (see Section S2 in Supplementary Material).	`liftone_GLM()`, `liftone_MLM()`
	Finding constrained D-optimal approximate allocation for generalized linear model and multinomial logit model (see Section 3).	`liftone_constrained_GLM()`, `liftone_constrained_MLM()`
	Transferring approximate allocation to exact allocation (see Section 3).	`approxtoexact_constrained_func()` `approxtoexact_func()`
	Finding constrained uniform exact allocation for bounded constraint (see Section 3.1).	`bounded_uniform()`
Example Specific	Finding I set for exact allocation conversion in `trial_data` and `trauma_data` examples (see Section 3 and Section S4 in Supplementary Material).	`iset_func_trauma()`, `iset_func_trial()`

Table 1: Functions and corresponding usages in the CDsampling package.

generated by the logistic regression model

$$\text{logit}\{P(Y_{ij}=1 \mid x_{gender\_i}, x_{age\_1i}, x_{age\_2i})\} = \beta_0 + \beta_1 x_{gender\_i} + \beta_{21} x_{age\_1i} + \beta_{22} x_{age\_2i} \quad (3)$$

with $(\beta_0, \beta_1, \beta_{21}, \beta_{22}) = (0, 3, 3, 3)$, where $i = 1, \ldots, 6$ stands for the ith covariate combination, and $j = 1, \ldots, n_i$ is an index of patients who fall into the ith covariate combination or sampling subgroups. Figure 4 illustrates the distribution of treatment efficacy across different gender and age groups.

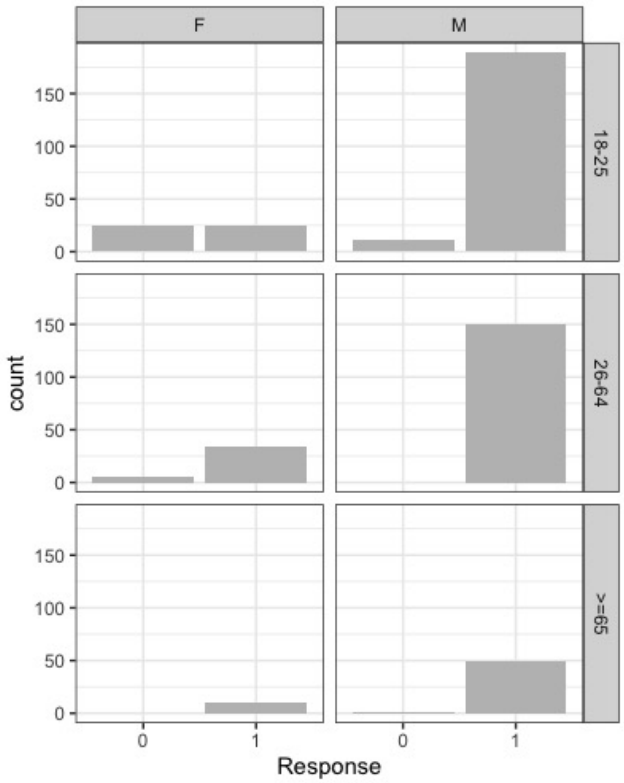

Figure 4: The number of patients from different gender (F, M) and age groups ($18-25$, $26-64$, and ≥ 65) and their responses (0 indicating ineffectiveness, 1 indicating effectiveness) to treatment in `trial_data` of CDsampling package.

In this example, it is posited that a sample of $n = 200$ participants is desired from the population of $N = 500$ volunteers due to budget constraints. The objective is to examine the variation in efficacy rates across gender and age demographics. Should a pilot study or relevant literature provide approximate values for the model parameters, a constrained lift-one algorithm may be employed to find a locally D-optimal design. Conversely, if only partial parameter information is available, the expectation can be deduced from some prior distributions, and the constrained lift-one algorithm can be utilized to determine an EW D-optimal allocation (substituting $\mathbf{W}$ in the Fisher information matrix (1) with $E(\mathbf{W})$).

There are $m = 6$ design points, corresponding to gender and age group combinations $(x_{gender\_i}, x_{age\_1i}, x_{age\_2i}) = (0,0,0)$, $(0,1,0)$, $(0,0,1)$, $(1,0,0)$, $(1,1,0)$, and $(1,0,1)$, respectively. The numbers of available volunteers in the six categories are $(N_1, N_2, \ldots, N_6) = (50, 40, 10, 200, 150, 50)$. Our goal is to find the constrained D-optimal allocation $(w_1, w_2, \ldots, w_6)$ in the set of all feasible allocations $S = \{(w_1, \ldots, w_m)^T \in S_0 \mid nw_i \leq N_i, i = 1, \ldots, m\}$.

We consider the logistic regression model (3), to find the locally D-optimal design, we assume, for illustrative purposes, that the model parameters $(\beta_0, \beta_1, \beta_{21}, \beta_{22}) = (0, 3, 3, 3)$. We may define the parameters and the model matrix as follows. Subsequently, we find the $\mathbf{W}$ matrix in (1) for the Fisher information matrix $\mathbf{F}$.

```
> beta = c(0, 3, 3, 3) #coefficients
> #design matrix
> X=matrix(data=c(1,0,0,0,1,0,1,0,1,0,0,1,1,1,0,0,1,1,1,0,1,1,0,1), ncol=4, byrow=TRUE)
> W=W_func_GLM(X=X, b=beta, link="logit") #find W as input of constrained liftone
```

To define the number of design points, sample size, and constraints with S, we use the following R codes (see Section S3 in the Supplementary Material of Huang et al. (2025) for details on finding r_{i1} and r_{i2} in step 3 of the constrained liftone algorithm in Figure 2):

```
> rc = c(50, 40, 10, 200, 150, 50)/200 #available volunteers/sample size
> m = 6 #design points

> g.con = matrix(0,nrow=(2*m+1), ncol=m) #constraints
> g.con[1,] = rep(1, m)
> g.con[2:(m+1),] = diag(m)
> g.con[(m+2):(2*m+1), ] = diag(m)
> g.dir = c("==", rep("<=", m), rep(">=", m)) #direction
> g.rhs = c(1, rc, rep(0, m)) #righ-hand side

> #lower bound in step 3 of constrained liftone
> lower.bound=function(i, w){
+   rc = c(50, 40, 10, 200, 150, 50)/200
+   m=length(w)
+   temp = rep(0,m)
+   temp[w>0]=1-pmin(1,rc[w>0])*(1-w[i])/w[w>0];
+   temp[i]=0;
+   max(0,temp);
+   }

> #upper bound in step 3 of constrained liftone
> upper.bound=function(i, w){
+   rc = c(50, 40, 10, 200, 150, 50)/200
+   min(1,rc[i]);
+   }
```

To identify the subgroups of the output D-optimal allocations, we may add an optional label for each of the $m = 6$ covariatres combination or subgroups as "F, 18-25", "F, 26-64", "F, >=65", "M, 18-2", "M, 26-6", "M, >=65" using the following codes:

```
> label = c("F, 18-25", "F, 26-64", "F, >=65", "M, 18-25", "M, 26-64", "M, >=65")
```

Then, we run the constrained lift-one algorithm with `liftone_constrained_GLM()` to find the constrained D-optimal approximate allocation.

```
> set.seed(092)
> approximate_design = liftone_constrained_GLM(X=X, W=W, g.con=g.con, g.dir=g.dir,
+ g.rhs=g.rhs, lower.bound=lower.bound, upper.bound=upper.bound, reltol=1e-10,
+ maxit=100, random=TRUE, nram=4, w00=NULL, epsilon=1e-8, label=label)
```

The design output is presented below:

```
> print(approximate_design)

Optimal Sampling Results:
================================================================================
Optimal approximate allocation:
   F, 18-25 F, 26-64 F, >=65 M, 18-25 M, 26-64 M, >=65
w  0.25     0.2      0.05    0.5      0.0      0.0
w0 0.25     0.0      0.05    0.7      0.0      0.0
--------------------------------------------------------------------------------
maximum :
2.8813e-08
--------------------------------------------------------------------------------
convergence :
TRUE
--------------------------------------------------------------------------------
itmax :
9.0
--------------------------------------------------------------------------------
deriv.ans :
0.0, 3.6017e-08, 4.8528e-07, -1.1525e-07, -1.0310e-07, -7.9507e-08
--------------------------------------------------------------------------------
gmax :
0.0
--------------------------------------------------------------------------------
reason :
"gmax <= 0"
--------------------------------------------------------------------------------
```

The output includes several key components:

- **w**: reports the D-optimal approximate allocation.
- $\mathbf{w}_0$: reports the random initial approximate allocation used to initialize optimization.
- **maximum**: reports the maximum determinant of Fisher information matrix.
- **reason**: reports the termination criterion for the constrained lift-one algorithm including either "all derivative ≤ 0" or "gmax ≤ 0", which corresponds to step 7° and step 8° in the constrained lift-one algorithm in Figure 2.

In practical terms, exact allocations are more beneficial. One may use the constrained approximate to exact allocation algorithm depicted in Figure 3, which is implemented as the `approxtoexact_constrained_func()` function.

```
> exact_design = approxtoexact_constrained_func(n=200, w=approximate_design$w, m=6,
+ beta=beta, link='logit', X=X, Fdet_func=Fdet_func_GLM, iset_func=iset_func_trial,
```

```
+ label=label)
> print(exact_design)

Optimal Sampling Results:
================================================================================
Optimal exact allocation:
               F, 18-25 F, 26-64 F, >=65 M, 18-25 M, 26-64 M, >=65
allocation      50.0     40.0     10.0    100.0    0.0      0.0
allocation.real 0.25     0.2      0.05    0.5      0.0      0.0
--------------------------------------------------------------------------------
det.maximum :
46.1012
--------------------------------------------------------------------------------
```

The output provides three key components for the sampling results:

- **allocation**: reports the exact allocation of D-optimal sampling, specifying the number of subjects to sample from each subgroup.
- **allocation.real**: reports the real-number approximate allocation used prior to integer conversion.
- **det.maximum**: reports the maximum determinant of the Fisher information matrix by the optimal design.

In this example, the D-optimal exact allocation is to sample 50 subjects from the "female, $18-25$" subgroup, 40 subjects from the "female, $26-64$" subgroup, 10 subjects from the "female, ≥ 65" subgroup, and 100 subjects from the "male, $18-25$" subgroup. Such a design may not explore all the design space and may lead to extreme design cases. In practice, allocating some subjects to the omitted subgroups "male, $26-64$" and "male, ≥ 65" could improve robustness and reduce the risk of overfitting.

Alternatively, one may aim for EW D-optimal allocations when partial coefficient information is available with the **W** matrix substituted by the expectation (EW stands for the expectation of the **W** matrix). To calculate these expectations, one may define prior distributions for the parameters based on available information. For instance, in this scenario, we assume the following independent prior distributions: $\beta_0 \sim \text{uniform}(-2,2)$, $\beta_1 \sim \text{uniform}(-1,5)$, $\beta_{21} \sim \text{uniform}(-1,5)$, and $\beta_{22} \sim$ uniform(-1,5). Subsequently, the diagonal elements of **W** are determined through integration. For $i = 1,\ldots,m$ and $\eta_i = \beta_0 + x_{gender\_i}\beta_1 + x_{age\_i1}\beta_{21} + x_{age\_i2}\beta_{22}$, we calculate the key component $E(v_i)$ of the ith diagonal element of **W** through:

$$\int_{-1}^{5}\int_{-1}^{5}\int_{-1}^{5}\int_{-2}^{2} \frac{\exp(\eta_i)}{(1+\exp(\eta_i))^2}\Pr(\beta_0)\Pr(\beta_1)\Pr(\beta_{21})\Pr(\beta_{22})\, d\beta_0 d\beta_1 d\beta_{21} d\beta_{22}$$

where $\Pr(\cdot)$ stands for the corresponding probability density function. We use the `hcubature()` function in the **cubature** package to calculate the integration as illustrated by the R codes below.

```
> unif.prior <- rbind(c(-2, -1, -1, -1), c(2,  5,  5, 5)) #prior parameters

> #find expectation of W matrix given priors
> W.EW.unif = matrix(rep(0,6))
> for (i in 1:6){
+  x = X[i,]
+  W.EW.unif[i] = hcubature(function(beta) dunif(beta[1], min=unif.prior[1,1],
max=unif.prior[2,1])*dunif(beta[2], min=unif.prior[1,2], max=unif.prior[2,2])*
dunif(beta[3], min=unif.prior[1,3], max=unif.prior[2,3])*dunif(beta[4],
min=unif.prior[1,4], max=unif.prior[2,4])*(exp(x[1]*beta[1]+x[2]*beta[2]+x[3]*
```

```
beta[3]+x[4]*beta[4])/(1+exp(x[1]*beta[1]+x[2]*beta[2]+x[3]*beta[3]+x[4]*beta[4]))^2),
lowerLimit = unif.prior[1,], upperLimit  = unif.prior[2,])$integral
+  }
```

Given the expectation of **W**, the functions `liftone_constrained_GLM()` and `approxtoexact_constrained_func()` are used for deriving the constrained EW D-optimal approximate allocation and the corresponding exact allocation, respectively. This process follows a similar procedure to that used for local D-optimal approximate allocation.

```
> set.seed(123)
> approximate_design_EW = liftone_constrained_GLM(X=X, W=W.EW.unif,  g.con=g.con,
+ g.dir=g.dir, g.rhs=g.rhs, lower.bound=lower.bound, upper.bound=upper.bound,
+ reltol=1e-12, maxit=100, random=TRUE, nram=4, w00=NULL, epsilon=1e-10, label=label)

> exact_design = approxtoexact_constrained_func(n=200,
+ w=approximate_design_EW$w, m=6, beta=beta, link='logit', X=X, Fdet_func=Fdet_func_GLM,
+ iset_func=iset_func_trial, label=label)
```

The output is summarized with `print()` function and presented below:

```
> print(exact_design_EW)

Optimal Sampling Results:
================================================================================
Optimal exact allocation:
                F, 18-25 F, 26-64 F, >=65 M, 18-25 M, 26-64 M, >=65
allocation      48.0     40.0     10.0    43.0     19.0     40.0
allocation.real 0.2406   0.2      0.05    0.2102   0.0991   0.2001
--------------------------------------------------------------------------------
det.maximum :
25.59
--------------------------------------------------------------------------------
```

In situations when the model parameters are unknown, the constrained uniform allocation is applicable. This method entails sampling an equal number of patients from each category within the given constraints. The selection criterion is $n_i = \min\{k, N_i\}$ or $\min\{k, N_i\} + 1$ with k satisfying $\sum_{i=1}^{m} \min\{k, N_i\} \leq n < \sum_{i=1}^{m} \min\{k+1, N_i\}$, where N_i represents the maximum allowable number for each category. This is an example of a bounded design problem, where each category has an upper boundary. The function `bounded_uniform()` can be used to find the constrained uniform allocation with the `trial_data` example and "**allocation**" in the output representing the constrained uniform allocation.

```
> bounded_uniform(Ni=c(50, 40, 10, 200, 150, 50), nsample=200)

Optimal Sampling Results:
================================================================================
Optimal exact allocation:
           F, 18-25 F, 26-64 F, >=65 M, 18-25 M, 26-64 M, >=65
allocation 38.0     38.0     10.0    38.0     38.0     38.0
--------------------------------------------------------------------------------
```

Alternatively, we may also use `approxtoexact_constrained_func()` to find the same constrained uniform exact allocation. This function can be used under fairly general constraints. To find the constrained uniform exact allocation using `approxtoexact_constrained_func()`, we suggest starting with one subject in each stratum or subgroup, which corresponds to the approximate allocation $\mathbf{w_{00}} = (1/200, 1/200, 1/200, 1/200, 1/200, 1/200)$ in this case.

```
> w00 = rep(1/200, 6) #initial approximate allocation
> unif_design = approxtoexact_constrained_func(n=200, w=w00, m=6, beta=NULL,
+ link=NULL, X=NULL, Fdet_func=Fdet_func_unif, iset_func=iset_func_trial, label=label)

> print(unif_design)

Optimal Sampling Results:
================================================================================
Optimal exact allocation:
                F, 18-25 F, 26-64 F, >=65 M, 18-25 M, 26-64 M, >=65
allocation      38.0     38.0     10.0    38.0     38.0     38.0
allocation.real 0.005    0.005    0.005   0.005    0.005    0.005
--------------------------------------------------------------------------------
det.maximum :
792351680.0
--------------------------------------------------------------------------------
```

The `iset_func_trial()` function in the **CDsampling** package is specifically designed for the `trial_data` example, which defines the set I in step 1° and step 2.4 in the constrained approximate to exact algorithm depicted by Figure 3. This function serves as a template that users can adapt to their specific constraints by modifying the codes. The package includes two such template functions: `iset_func_trial()` and `iset_func_trauma()` with details provided in Section S4 of Supplementary Material.

To perform a comparison analysis on different sampling strategies, including the constrained D-optimal allocation, the constrained EW D-optimal allocation with uniform priors, the constrained uniform allocation, simple random sample without replacement (SRSWOR), as well as the full data (all the 500 patients enrolled), we simulate their responses Y_{ij}'s based on model (3) with parameter values $(0, 3, 3, 3)$. We apply each sampling strategy to obtain a sample of $n = 200$ observations out of 500, and estimate the parameters using the 200 observations. The exception is the full data method, where estimation is performed using all 500 patients. We use the root mean square error (RMSE) to measure the accuracy of the estimates (see Section 4 of Huang et al. (2025) for more theoretical and technical details). We repeat the procedure 100 times and display the corresponding RMSEs in Figure 5 and simulation codes in Supplementary Material Section S3 (Wickham, 2016; Wickham et al., 2023). Obviously, if we use the full data (all 500 patients) to fit the model, its RMSE attains the lowest. Besides that, the constrained locally D-optimal allocation and the constrained EW D-optimal allocation have a little higher RMSEs than the full data estimates but outperform SRSWOR and the constrained uniform allocation. The sampling strategy based on the constrained uniform allocation doesn't need any model information and is a more robust sampling scheme, which is still better than SRSWOR.

3.2 Applications in paid research study: `trauma_data` example

In the **CDsampling** package, `trauma_data` is a dataset of $N = 802$ trauma patients from Chuang-Stein and Agresti (1997), stratified according to the trauma severity at the entry time of the study with 392 mild and 410 moderate/severe patients enrolled. The study involved four treatment groups determined by the `dose` level, $x_{i1} = 1$ (`Placebo`), 2 (`Low dose`), 3 (`Medium dose`), and 4 (`High dose`). Combining with `severity` grade ($x_{i2} = 0$ for `mild` or 1 for `moderate/severe`), there are $m = 8$ distinct experimental settings with $(x_{i1}, x_{i2}) = (1,0), (2,0), (3,0), (4,0), (1,1), (2,1), (3,1), (4,1)$, respectively. The responses belong to five ordered categories, `Death` (1), `Vegetative state` (2), `Major disability` (3), `Minor disability` (4) and `Good recovery` (5), known as the Glasgow Outcome Scale (Jennett and Bond, 1975). Figure 6 shows the distribution of outcomes over `severity` grades and `dose` levels. In this example, we have $m = 8$ subgroups, which are combinations of the two covariates categories: `dose` levels and `severity` grades. We aim to enroll $n = 600$

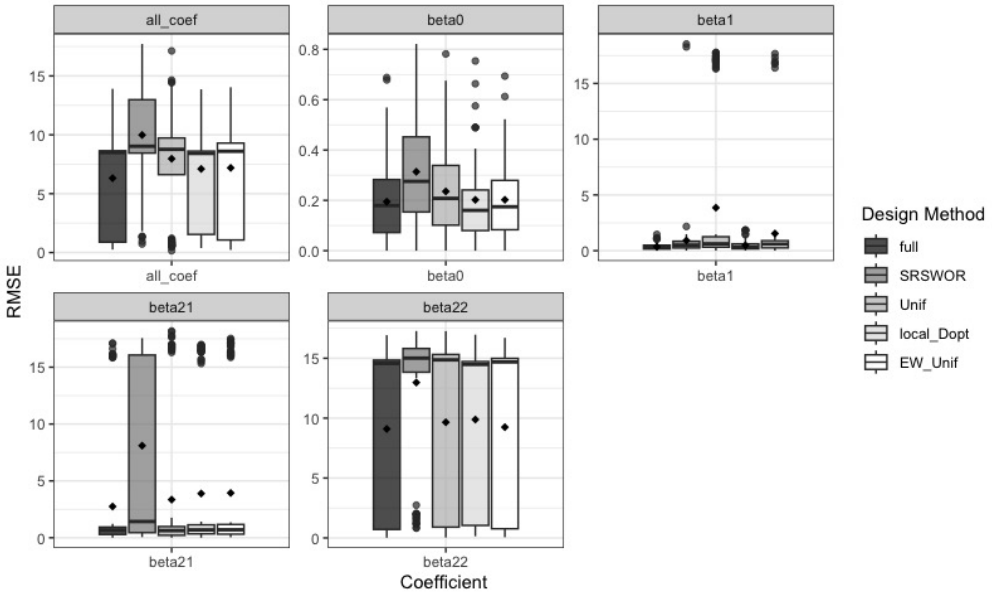

Figure 5: Boxplots of RMSEs obtained from 100 simulations using full data (full), SRSWOR, constrained uniform design (Unif), the constrained locally D-optimal allocation (local_Dopt), and constrained EW D-optimal allocation with uniform priors (EW_Unif), with black diamonds representing average RMSE.

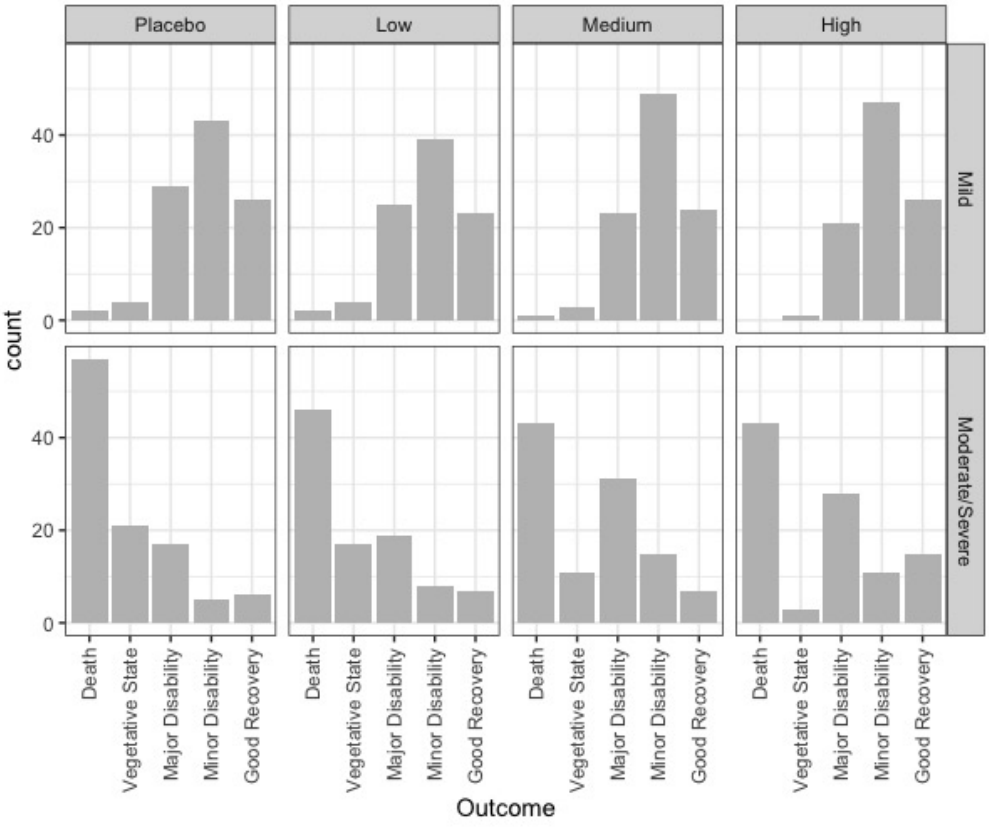

Figure 6: The number of patients from different `dose` levels (Placebo, Low, Medium, and High) and `severity` grades (Mild, Moderate/Severe) groups and their treatment outcomes in `trauma_data` of **CDsampling** package.

patients from the 802 available patients. The collection of feasible allocations is inherently

constrained by the number of patients in different severity grades, defined as

$$S = (w_1, \ldots, w_8)^\top \in S_0 \mid n(w_1 + w_2 + w_3 + w_4) \leq 392, \qquad n(w_5 + w_6 + w_7 + w_8) \leq 410.$$

The constraints specify that in the sample, the number of patients with mild symptoms must not exceed 392 across all dose levels, while those with moderate/severe symptoms must not exceed 410.

The parameters fitted from the `trauma_data` are

$$\begin{aligned}\boldsymbol{\beta} &= (\hat{\beta}_{11}, \hat{\beta}_{12}, \hat{\beta}_{13}, \hat{\beta}_{21}, \hat{\beta}_{22}, \hat{\beta}_{23}, \hat{\beta}_{31}, \hat{\beta}_{32}, \hat{\beta}_{33}, \hat{\beta}_{41}, \hat{\beta}_{42}, \hat{\beta}_{43})^\top \\ &= (-4.047, -0.131, 4.214, -2.225, -0.376, 3.519, -0.302, -0.237, 2.420, 1.386, -0.120, 1.284)^\top.\end{aligned}$$

The model can be written in the following format:

$$\begin{aligned}\log\left(\frac{\pi_{i1}}{\pi_{i2} + \cdots + \pi_{i5}}\right) &= \beta_{11} + \beta_{12}x_{i1} + \beta_{13}x_{i2} \\ \log\left(\frac{\pi_{i1} + \pi_{i2}}{\pi_{i3} + \pi_{i4} + \pi_{i5}}\right) &= \beta_{21} + \beta_{22}x_{i1} + \beta_{23}x_{i2} \\ \log\left(\frac{\pi_{i1} + \pi_{i2} + \pi_{i3}}{\pi_{i4} + \pi_{i5}}\right) &= \beta_{31} + \beta_{32}x_{i1} + \beta_{33}x_{i2} \\ \log\left(\frac{\pi_{i1} + \cdots + \pi_{i4}}{\pi_{i5}}\right) &= \beta_{41} + \beta_{42}x_{i1} + \beta_{43}x_{i2}\end{aligned}$$

where $i = 1, \ldots, 8$.

We use the R codes below to define the model matrix and coefficients.

```
> J=5; p=12; m=8; #response levels; parameters; subgroups

> #coefficients
> beta = c(-4.047, -0.131, 4.214, -2.225, -0.376, 3.519, -0.302, -0.237,  2.420, 1.386,
+ -0.120,  1.284)

> #define design matrix of 8 subgroups
> Xi=rep(0,J*p*m); dim(Xi)=c(J,p,m);
> Xi[,,1] = rbind(c( 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),
+           c( 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0), c( 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0),
+           c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0), c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))

> Xi[,,2] = rbind(c( 1, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),
+           c( 0, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0), c( 0, 0, 0, 0, 0, 0, 1, 2, 0, 0, 0, 0),
+           c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 0), c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))

> Xi[,,3] = rbind(c( 1, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),
+           c( 0, 0, 0, 1, 3, 0, 0, 0, 0, 0, 0, 0), c( 0, 0, 0, 0, 0, 0, 1, 3, 0, 0, 0, 0),
+           c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 3, 0), c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))

> Xi[,,4] = rbind(c( 1, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),
+           c( 0, 0, 0, 1, 4, 0, 0, 0, 0, 0, 0, 0), c( 0, 0, 0, 0, 0, 0, 1, 4, 0, 0, 0, 0),
+           c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 4, 0), c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))

> Xi[,,5] = rbind(c( 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0),
+           c( 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0), c( 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0),
+           c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1), c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))

> Xi[,,6] = rbind(c( 1, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0),
+           c( 0, 0, 0, 1, 2, 1, 0, 0, 0, 0, 0, 0), c( 0, 0, 0, 0, 0, 0, 1, 2, 1, 0, 0, 0),
```

```
+          c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 1), c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))

> Xi[,,7] = rbind(c( 1, 3, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0),
+          c( 0, 0, 0, 1, 3, 1, 0, 0, 0, 0, 0, 0), c( 0, 0, 0, 0, 0, 0, 1, 3, 1, 0, 0, 0),
+          c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 3, 1), c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))

> Xi[,,8] = rbind(c( 1, 4, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0),
+          c( 0, 0, 0, 1, 4, 1, 0, 0, 0, 0, 0, 0), c( 0, 0, 0, 0, 0, 0, 1, 4, 1, 0, 0, 0),
+          c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 4, 1), c( 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
```

To define the sample size, the constraints, and the functions of lower and upper boundaries r_{i1} and r_{i2}, we may use the following R codes (see Section S3 in the Supplementary Material of Huang et al. (2025) for details on finding r_{i1} and r_{i2}):

```
> nsample=600 #sample size
> constraint = c(392, 410)  #mild:severe

> #lower bound function in step 3 of constrained liftone
> lower.bound <- function(i, w0){
+   n = 600
+   constraint = c(392,410)
+   if(i <= 4){
+     a.lower <- (sum(w0[5:8])-(constraint[2]/n)*(1-w0[i]))/(sum(w0[5:8]))}
+   else{
+     a.lower <- (sum(w0[1:4])-(constraint[1]/n)*(1-w0[i]))/(sum(w0[1:4]))}
+    a.lower}

> #upper bound function in step 3 of constrained liftone
> upper.bound <- function(i, w0){
+  n = 600
+  constraint = c(392,410)
+  if(i <= 4){
+    b.upper <- ((constraint[1]/n)*(1-w0[i]) - (sum(w0[1:4])-w0[i]))/(1-sum(w0[1:4]))}
+  else{
+    b.upper <- ((constraint[2]/n)*(1-w0[i]) - (sum(w0[5:8])-w0[i]))/(1-sum(w0[5:8]))}
+    b.upper}

> #define constraints
> g.con = matrix(0,nrow=length(constraint)+1+m, ncol=m)
> g.con[2:3,] = matrix(data=c(1,1,1,1,0,0,0,0,0,0,0,0,1,1,1,1), ncol = m, byrow=TRUE)
> g.con[1,] = rep(1, m)
> g.con[4:(length(constraint)+1+m), ] = diag(1, nrow=m)
> g.dir = c("==", "<=","<=", rep(">=",m))
> g.rhs = c(1, ifelse((constraint/nsample<1),constraint/nsample,1), rep(0, m))
```

Then, we may define an optional label of the sampling subgroups that corresponds to each of the $m = 8$ subgroups using the following code:

```
> label=label = c("Placebo-Mild", "Low-Mild", "Medium-Mild", "High-Mild", "Placebo-Severe",
+ "Low-Severe", "Medium-Severe", "High-Severe")
```

We then run the constrained lift-one algorithm to find the constrained D-optimal approximate allocation using `liftone_constrained_MLM()` function and convert the approximate allocation to an exact allocation with `approxtoexact_constrained_func` function.

```
> set.seed(123)
> approx_design = liftone_constrained_MLM(m=m, p=p, Xi=Xi, J=J, beta=beta,
```

```
+ lower.bound=lower.bound, upper.bound=upper.bound, g.con=g.con, g.dir=g.dir,
+ g.rhs=g.rhs, w00=NULL, link='cumulative',  Fi.func=Fi_func_MLM, reltol=1e-5,
+ maxit=500, delta=1e-6, epsilon=1e-8, random=TRUE, nram=3, label=label)

> exact_design = approxtoexact_constrained_func(n=600, w=approx_design$w, m=8,
+ beta=beta, link='cumulative', X=Xi, Fdet_func=Fdet_func_MLM,
+ iset_func=iset_func_trauma, label=label)

> print(exact_design)

Optimal Sampling Results:
================================================================================
Optimal exact allocation:
             Placebo-Mild Low-Mild Medium-Mild High-Mild Placebo-Severe
allocation      155.0        0.0      0.0         100.0     168.0
allocation.real 0.2593       0.0      0.0         0.1667    0.2796
              Low-Severe Medium-Severe High-Severe
allocation      0.0        0.0           177.0
allocation.real 0.0        0.0           0.2944
--------------------------------------------------------------------------------
det.maximum :
1.63163827059162e+23
--------------------------------------------------------------------------------
```

The **allocation** output provides the exact allocation of the sampling across different treatment-severity subgroups, representing the implementable sample sizes for each subgroup. The result is derived by converting the **allocation.real**, which is the D-optimal approximate allocation outcome from `liftone_constrained_MLM()`.

As the `trauma_data` example doesn't have bounded constraints, to find the constrained uniform sampling allocation, we use `approxtoexact_constrained_func()` with one subject in each stratum or subgroup as the input, that is, the approximate allocation $\mathbf{w} = (1/600, 1/600, 1/600, 1/600, 1/600, 1/600, 1/600, 1/600)$. The corresponding I set function is provided in the **CDsampling** package, and it can be easily defined according to other constraints, see Section S4 of Supplementary Material. Note that the determinant provided by `approxtoexact_constrained_func()` for different designs is not comparable, as the criteria `Fdet_func` differs.

```
> unif_design = approxtoexact_constrained_func(n=600, w=rep(1/600,8), m=8,
+ beta=NULL, link=NULL, X=NULL, Fdet_func=Fdet_func_unif, iset_func=iset_func_trauma)

> print(unif_design)

Optimal Sampling Results:
================================================================================
Optimal exact allocation:
                Placebo-Mild Low-Mild Medium-Mild High-Mild Placebo-Severe Low-Severe
allocation      75.0         75.0     75.0        75.0      75.0           75.0
allocation.real 0.0017       0.0017   0.0017      0.0017    0.0017         0.0017
                Medium-Severe High-Severe
allocation      75.0          75.0
allocation.real 0.0017        0.0017
--------------------------------------------------------------------------------
det.maximum :
1001129150390625
--------------------------------------------------------------------------------
```

4 Summary

The current version of **CDsampling** implements D-optimal allocations within both paid research sampling and general study frameworks with or without constraints. Its primary objective is to optimize sampling allocations for better model estimation accuracy in the studies. The package includes `F_func_GLM()` and `F_func_MLM()` for the computation of the Fisher information matrix of GLMs and MLMs, respectively. It is noteworthy that standard linear regression models are special GLMs with an identity link function and Gaussian-distributed responses, which is also supported by our package. Theoretical results are summarized in Section 2.2 and Section 2.3 while illustrative examples are provided in Supplementary Section S1.

To find standard or unconstrained D-optimal allocations, our package implements the lift-one algorithm through functions `liftone_GLM()` and `liftone_MLM()`. Paid research studies often impose sampling constraints. To address this, the constrained lift-one algorithm can be applied using functions `liftone_constrained_GLM()` and `liftone_constrained_MLM()`. An example illustrating the difference between the lift-one algorithm and the constrained lift-one algorithm is provided in Supplementary Section S2 while Section 3 presents two application examples from paid research studies.

In the absence of model information, `constrained_uniform()` function is available to find a robust constrained uniform allocation with bounded constraints, while the `approxtoexact_constrained_func()` function can be used to find constrained uniform allocation with more general constraints. For transitioning from approximate to exact allocations, the package provides `approxtoexact_constrained_func()` for constrained cases and `approxtoexact_func()` for unconstrained cases. Detailed applications for both GLMs and MLMs are provided in Sections 3.1 and 3.2.

Future enhancements of the package may aim to incorporate a broader spectrum of optimality criteria, such as A-optimality and E-optimality, as well as some models beyond GLMs and MLMs to expand its applicability.

References

A. Agresti. *Categorical Data Analysis*. Wiley, 3 edition, 2013. doi: 10.1007/s00362-015-0733-8. [p6]

A. Atkinson, A. Donev, and R. Tobias. *Optimum Experimental Designs, with SAS*. Oxford University Press, 2007. doi: 10.1093/oso/9780199296590.001.0001. [p2, 3]

G. Barcaroli. Samplingstrata: An R package for the optimization of stratified sampling. *Journal of Statistical Software*, 61:1–24, 2014. doi: 10.18637/jss.v061.i04. [p2]

X. Bu, D. Majumdar, and J. Yang. D-optimal designs for multinomial logistic models. *Annals of Statistics*, 48(2):983–1000, 2020. doi: 10.1214/19-AOS1834. [p2, 6, 7]

C. Chuang-Stein and A. Agresti. Tutorial in biostatistics-a review of tests for detecting a monotone dose-response relationship with ordinal response data. *Statistics in Medicine*, 16:2599–2618, 1997. [p13]

A. Dobson and A. Barnett. *An Introduction to Generalized Linear Models*. Chapman & Hall/CRC, 4 edition, 2018. doi: 10.1201/9781315182780. [p5]

N. Dousti Mousavi, H. Aldirawi, and J. Yang. Categorical data analysis for high-dimensional sparse gene expression data. *BioTech*, 12(3):52, 2023. doi: 10.3390/biotech12030052. [p6]

V. Fedorov. *Theory of Optimal Experiments*. Academic Press, 1972. ISBN 0122507509. [p2]

V. Fedorov and S. Leonov. *Optimal Design for Nonlinear Response Models*. Chapman & Hall/CRC, 2014. doi: 10.1201/b15054. [p2]

G. Glonek and P. McCullagh. Multivariate logistic models. *Journal of the Royal Statistical Society, Series B*, 57:533–546, 1995. URL https://www.jstor.org/stable/2346155. [p6]

U. Grömping. Optimal experimental design with R. *Journal of Statistical Software*, 43:1–4, 2011. doi: 10.18637/jss.v043.b05. [p2]

R. Harman, L. Filova, and M. L. Filova. Package 'optimaldesign', 2016. URL https://cran.r-project.org/package=OptimalDesign. [p2]

Y. Huang, L. Tong, and J. Yang. Constrained d-optimal design for paid research study. *Statistica Sinica*, 2025. doi: 10.5705/ss.202022.0414. To appear. [p2, 4, 6, 9, 13, 16]

B. Jennett and M. Bond. Assessment of outcome after severe brain damage. *Lancet*, 305: 480–484, 1975. doi: 10.1016/S0140-6736(75)92830-5. [p13]

A. Khuri, B. Mukherjee, B. Sinha, and M. Ghosh. Design issues for generalized linear models: A review. *Statistical Science*, 21:376–399, 2006. doi: 10.1214/088342306000000105. [p6]

J. Kiefer. General equivalence theory for optimum designs (approximate theory). *Annals of Statistics*, 2:849–879, 1974. doi: 10.1214/aos/1176342810. [p3]

P. S. Levy and S. Lemeshow. *Sampling of populations: methods and applications*. Wiley, Hoboken, N.J, 4th ed. edition, 2008. ISBN 9780470374597. doi: 10.1002/9780470374597. [p2]

S. Lohr. *Sampling: Design and Analysis*. Chapman and Hall/CRC, 2019. doi: 10.1201/9780429298899. [p1]

P. McCullagh and J. Nelder. *Generalized Linear Models*. Chapman and Hall/CRC, 2 edition, 1989. doi: 10.1007/978-1-4899-3242-6. [p5, 6]

J. A. Nelder and R. W. Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society Series A: Statistics in Society*, 135(3):370–384, 1972. doi: 10.2307/2344614. [p5]

A. M. Overstall, D. C. Woods, M. Adamou, and D. Michaelides. Package 'acebayes'. 2018. URL https://cran.r-project.org/package=acebayes. [p2]

A. M. Overstall, D. C. Woods, and M. Adamou. acebayes: An R package for Bayesian optimal design of experiments via approximate coordinate exchange. *Journal of Statistical Software*, 95(13):1–33, 2020. doi: 10.18637/jss.v095.i13. URL https://www.jstatsoft.org/index.php/jss/article/view/v095i13. [p2]

F. Pukelsheim. *Optimal Design of Experiments*. SIAM, Philadelphia, 2006. doi: 10.1137/1.9780898719109. [p3]

J. Stufken and M. Yang. Optimal designs for generalized linear models. In K. Hinkelmann, editor, *Design and Analysis of Experiments, Volume 3: Special Designs and Applications*, chapter 4, pages 137–164. Wiley, 2012. doi: 10.1002/9781118147634. [p6]

Y. Tillé. *Sampling Algorithms*. Springer, 2006. doi: 10.1007/0-387-34240-0. [p1]

Y. Tillé and A. Matei. Package 'sampling'. *Surv. Sampling. Kasutatud*, 23:2017, 2016. URL https://cran.r-project.org/package=sampling. [p2]

T. Wang and J. Yang. Identifying the most appropriate order for categorical responses. *Statistica Sinica*, 35:411–430. doi: 10.25417/uic.25599399.v1. [p6]

B. Wheeler and M. J. Braun. Package 'algdesign'. *R Proj. Stat. Comput*, 1(0):1–25, 2019. URL https://cloud.r-project.org/web/packages/AlgDesign/AlgDesign.pdf. [p2]

H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org. [p13]

H. Wickham, R. François, L. Henry, K. Müller, and D. Vaughan. *dplyr: A Grammar of Data Manipulation*, 2023. URL https://dplyr.tidyverse.org. R package version 1.1.4, https://github.com/tidyverse/dplyr. [p13]

J. Yang and A. Mandal. D-optimal factorial designs under generalized linear models. *Communications in Statistics - Simulation and Computation*, 44:2264–2277, 2015. doi: 10.1080/03610918.2013.815773. [p2]

J. Yang, A. Mandal, and D. Majumdar. Optimal designs for 2^k factorial experiments with binary response. *Statistica Sinica*, 26:385–411, 2016. doi: 10.5705/ss.2013.265. [p2, 6]

J. Yang, L. Tong, and A. Mandal. D-optimal designs with ordered categorical data. *Statistica Sinica*, 27:1879–1902, 2017. doi: 10.5705/ss.202016.0210. [p2]

M. Yang and J. Stufken. Identifying locally optimal designs for nonlinear models: A simple extension with profound consequences. *Annals of Statistics*, 40:1665–1681, 2012. doi: 10.1214/12-AOS992. [p2]

S. Zocchi and A. Atkinson. Optimum experimental designs for multinomial logistic models. *Biometrics*, 55:437–444, 1999. doi: 10.1111/j.0006-341X.1999.00437.x. [p2, 6]

Yifei Huang
Department of Mathematics, Statistics, and Computer Science
University of Illinois at Chicago
E-mail: yhuan39@uic.edu

Liping Tong
Advocate Aurora Health
E-mail: liping.tong@aah.org

Jie Yang
Department of Mathematics, Statistics, and Computer Science
University of Illinois at Chicago
E-mail: jyang06@uic.edu