Sample Size Calculation:

Implementation of Different Scenarios

for the Online Tool SampleSizeR

Master Thesis in Biostatistics (STA495)

by

Georgy Astakhov
16-724-296

supervised by

Dr. Bernadetta Tarigna
Prof. Dr. Reinhard Furrer

Zurich, April 2024

Sample Size Calculation: Implementation of Different Scenarios for the Online Tool SampleSizeR

Georgy Astakhov

Version April 24, 2024

Contents

Preface v

Abstract vii

1 Introduction 1

2 Methods 3
2.1 *t* Tests . . . 3
2.1.1 Unpaired With Equal Variance . . . 3
2.1.2 Unpaired With Unequal Variance . . . 5
2.1.3 Paired Data . . . 6
2.2 Anova . . . 6
2.2.1 Oneway Anova . . . 6
2.2.2 Twoway Anova . . . 7
2.3 Marginal Models and Conditional Models . . . 8
2.3.1 Matrix Notation . . . 9
2.3.2 Marginal Models . . . 10
2.3.3 Conditional Models . . . 11
2.3.4 Marginal Models vs Conditional Models . . . 11
2.3.5 Sample Size Calculation for Marginal Models . . . 12
2.4 Complete Block Design . . . 15
2.5 Split-Plot Design . . . 19
2.6 CI-Width Based . . . 21

3 Shiny-App 23
3.1 Main Changes to SampleSizeR . . . 23
3.2 Password Protection . . . 28
3.3 Report Generation . . . 29

4 Conclusion 33

A Appendix 35
A.1 Positive Definite Matrix . . . 36
A.2 Plots . . . 36

Bibliography 39

Preface

I hope this Master's thesis will provide great guidance and will be very helpful for researchers and/or statisticians in planning their research. My goal was to make the report simple to understand but without loss of important information and the SampleSizeR user-firendly and bugs-free. I believe the goal was reached.

I would like to express my sincere gratitude to all those who have contributed to the completion of this master thesis.
First, I am very thankful to my supervisors Dr. Bernadetta Tarigan and Prof. Dr. Reinhard Furrer for their invaluable guidance, great support, and insightful feedback throughout the research process. Their expertise and encouragement have been instrumental in shaping this work.
I extend my appreciation to my wife, Alisa Fedorenko, and my cat, Imbir, for their endless love, encouragement, and understanding during this academic journey. Their unwavering support has been a constant source of strength and motivation.
I am grateful to my friends, especially Naumenko Anton and Rudenko Yuliia, and family for their intellectual exchange, and moral support. Their diverse perspectives and constructive criticism have enriched my research and personal growth.
To all those mentioned above and to anyone else who has played a part, however small, in this endeavor, I offer my sincerest thanks.

Georgy Astakhov
April 2024

Abstract

This Master's thesis aims to provide comprehensive guidance on selecting effective methods for sample size estimation tailored to various study designs in clinical research. It focuses on exploratory and confirmatory studies, detailing precision-based methods and power analysis for confirmatory ones. Theoretical backgrounds and practical implementations for different design settings, including completely randomized, randomized complete block, generalized complete block, and split-plot designs, are discussed. An additional approach for the sample size estimation in repeated measurements using the marginal GEE model is presented, alongside methods for both confirmatory and exploratory cases. The thesis also includes a practical component accessible online, with the final chapter briefly covering enhancements to the SampleSizeR, which is an online shiny app maintained by the Applied Statistics group at the Department of Mathematical Modeling and Machine Learning for the sample size calculation available here.

Chapter 1

Introduction

Clinical studies play a crucial part in, for example, the development of a new drug, testing new diagnostic tools, or finding new information yet not discovered information. Without a proper study design and a correct analysis, the conclusion might be not valid and the study was conducted for nothing. Hence, one of the first steps in study design planning is to understand what type of study a researcher is dealing with. Studies can be **observational**, where patients, mice, or other subjects of interest are monitored and recorded in their normal setting without interacting with them. These studies allow us to discover some new insights and use them later. Another approach is to run an **experimental** study, sometimes also known as a clinical trial. Experimental studies are typically used to assess the efficacy of new treatments or diagnostics, but this is not always the case. Sometimes, if the research matter is new and has not been studied in depth yet, we speak of **exploratory** studies, where the goal is to gather first information in the experimental setting, which will help to formulate the hypothesis for future work. On the other hand, if the knowledge already exists and the researchers have a concrete hypothesis that needs to be tested, we speak of a **confirmatory** study. Almost all studies have an important limiting factor, namely the budget of the study, which influences the number of subjects that can be recruited to participate in it. Therefore it is extremely necessary to determine ahead of time the right number of participants or subjects for the study. Fortunately, there are statistical tools, that allow us to do that. For exploratory studies, researchers can calculate sample sizes using a precision-based method, while for confirmatory ones, they can use power-analysis-based methods.

In chapter two we will tackle the primary goal of this Master's thesis, which is to offer comprehensive guidance on selecting the most effective methods for sample size estimation, tailored to the unique characteristics of a study design. The methods section will provide a theoretical background to the power analysis and its corresponding sample size calculation. It will cover most of the common models in a completely randomized design setting, such as a two-sample t test, paired t test, Welch's t test but also a oneway Anova and a twoway Anova. Additionally to completely randomized design, the thesis will cover the theoretical background of sample size calculation for other designs, such as randomized complete block design, generalized complete block design, and split-plot design. A very common scenario of repeated measurements in time will be tackled as well and the difference between marginal and conditional models will be briefly explained. A new and more efficient approach for the sample size calculation for repeated measurement will be presented and implemented. These are used in a confirmatory case and for the exploratory case, a confidence interval width-based method will be introduced.

A large part of the work involved improving the existing version of SampleSizeR, which is an online tool that helps to calculate the sample size in a very clean manner using a convenient user interface built with the help of shiny application by Chang et al. (2022) in R. The work involved not only implementing new features but also finding existing bugs and fixing them. The third

chapter of the thesis will briefly describe a selection of the changes that were made to the existing version of SampleSizeR, however, without going into every detail of the web development for the sake of readability. The complete code can be found in the git repository.

In the final chapter, we will briefly summarise the work that was done.

Chapter 2

Methods

In this chapter, different statistical models for the data analysis are discussed. They will be mentioned together with the design they can be applied to, their advantages, and most importantly the power analysis, which can be performed with these models, leading to a sample size calculation.

2.1 t Tests

The next subsections will elaborate on the simplest setting of any clinical trial, namely the comparison of means in two different groups by assessing the difference between these means. This scenario under certain assumptions can be analyzed using a t test. The main assumption of which is the normality of the data. The t test can be either paired or unpaired with and without an additional assumption of equal variance in two groups. A t test has a straightforward power analysis, hence a straightforward sample size calculation. A one-sample t test is based on the calculation of the t statistics

$$t_{obs} = \frac{\bar{Y} - \mu_{H_0}}{\sqrt{S^2/n}},$$

where $\bar{Y}$ is the estimator of the mean, μ_{H_0} is the mean according to the null hypothesis H_0, S is the estimator of the sample standard deviation and n is the sample size. Often μ_{H_0} is assumed to be equal to 0. Under the H_0 t_{obs} follows a central t distribution with $n-1$ degrees of freedom (df), whereas under H_1 it follows a non-central t distribution with $df = n-1$ and some non-centrality parameter (ncp) γ, which will be introduced properly later. It can be visualized as in Figure 2.1 and in the case of large df (degrees of freedom), a t test is very similar in results to a z test with a standard normal distribution.

2.1.1 Unpaired With Equal Variance

For the next subsections, we consider the following model

$$Y_{ij} = \mu + \alpha_i + \epsilon_{ij},$$

where α_i is the treatment effect of treatment $i = \{1, 2\}$, $j = \{1, \ldots, n\}$ and $\epsilon_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$. In a setting of clinical research, we want to analyze the difference in the average of the treated and the average of untreated patients, which we will denote with $\bar{Y}_2 - \bar{Y}_1$ for a simpler notation. With the model above we imply that

$$\bar{Y}_2 \sim \mathcal{N}(\mu_2, \frac{\sigma^2}{n}), \quad \bar{Y}_1 \sim \mathcal{N}(\mu_1, \frac{\sigma^2}{n}).$$

This leads to

$$\bar{Y}_2 - \bar{Y}_1 \sim \mathcal{N}(\mu_2 - \mu_1, \frac{2\sigma^2}{n}),$$

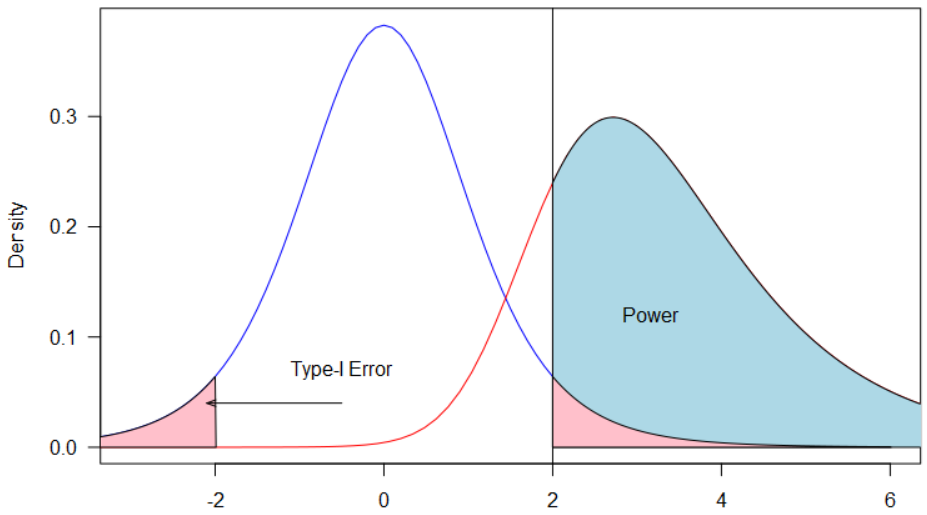

Figure 2.1: Power of a one-sample t test given $H_0 : \mu_{H_0} = 0$ (blue) and $H_1 : \mu_{H_1} = 2.5$ (red) with $t_{obs} = 2$, $df = 6$ and $\gamma = 3$ in case of H_1.

where σ^2 can be estimated with a pooled variance estimator $S_p^2 = (S_1^2 + S_2^2)/2$, where S_2^2 and S_1^2 are the the variance estimators of the treatment and control group respectively. In the case of an unpaired t test, the t statistics looks slightly different compared to the one in the previous section

$$t_{obs} = \frac{(\bar{Y}_2 - \bar{Y}_1) - D_{H_0}}{\sqrt{2S_p^2/n}},$$

where n is the sample size in each group and $df = 2n - 2$, assuming that group sizes are equal and D_{H_0} is the difference of two means $(\mu_2 - \mu_1)$ under H_0. In two-sample t test the hypotheses are defined as $H_0 : D = D_{H_0}; H_1 : D = D_{H_1}$, where D_{H_0} typically equals to 0.

The power for a difference of two random variables $\bar{Y}_2$ and $\bar{Y}_1$ is

$$Power = P\left(\left|\frac{(\bar{Y}_2 - \bar{Y}_1) - D_{H_0}}{\sqrt{2S_p^2/n}}\right| > t_{2n-2}^{-1}(1 - \alpha/2)\right),$$

where $t_{2n-2}^{-1}(1 - \alpha/2)$ is a critical value to reject the H_0 arising from the $(1 - \alpha/2)$th quantile of the t distribution with $2n - 2$ degrees of freedom, where α is the Type-I error as in the Figure 2.1. Further, we will use t_{crit} for this value. In general $t_{df}(\cdot|\cdot)$ will denote the cumulative distribution function of the t distribution with df degrees of freedom for some value and given the non-centrality parameter (ncp) γ, which is left out in cases where the ncp equals to 0 for a simpler notation.

It can be shown that $t_{obs} = (Z + \gamma) / \left(\sqrt{V/(2n-2)}\right)$, where

$$Z = \frac{(\bar{Y}_2 - \bar{Y}_1) - D_{H_1}}{\sqrt{2\sigma^2/n}}$$
$$\gamma = \frac{(D_{H_1} - D_{H_0})}{\sqrt{2\sigma^2/n}}$$
$$V = \frac{2n-2}{\sigma^2} S_p.$$

In a setting where $D_{H_1} = D$ and $D_{H_0} = 0$ the power equals to

$$Power = P\left(\left|\frac{(\bar{Y}_2 - \bar{Y}_1)}{\sqrt{2S_p^2/n}}\right| > t_{crit}\right)$$
$$\approx 1 - t_{2n-2}\left(t_{crit}|\gamma\right),$$

where $\gamma = D/\left(\sqrt{2S_p^2/n}\right)$ and we use an approximation because we assume that $P\left(t_{obs} < -t_{crit}\right)$ is almost 0.

In other words, it means that to approximately calculate the power, we would need to calculate the probability of observing the critical value under t distribution with $2n-2$ degrees of freedom and non-centrality parameter γ and subtract it from 1.
This can be calculated manually or by using a built-in R function.

```
# Manually
pt(qt(alpha/2, 2n-2, lower.tail = FALSE), 2n-2, ncp = sqrt(n/2) *
            delta/sd, lower.tail = FALSE)

# Built-in function
power.t.test(n = n, delta = delta, sd = sd, sig.level = alpha)
```

With this calculation, we can receive the needed sample size for a given power by solving for n.

2.1.2 Unpaired With Unequal Variance

The test for unpaired data with unequal variance is also known as Welch's t test and its derivation is less straightforward. If the assumption of equal variance is not fulfilled, *df* can no longer be calculated directly from the sample size and requires an additional step to do so. The degrees of freedom can be approximated using the following expression

$$df \approx \frac{\left(\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}\right)^2}{\frac{(S_1^2/n_1)^2}{n_1 - 1} + \frac{(S_2^2/n_2)^2}{n_2 - 1}},$$

where n_1 is the group size of Y_1 and n_2 of Y_2 respectively. Since we are dealing with the approximation of the *df* the resulting t statistics is also only approximately t distributed.

There is no direct implementation of power analysis in R, however, the package MKpower by Kohl (2023) has a function for the power calculation, or it can be done manually similar to the normal two-sample t test while adjusting for a proper *df*.

```
require(MKpower)
power.welch.t.test(n = n, delta = delta, sd1 = sd1, sd2 = sd2,
                   sig.level = alpha)
```

2.1.3 Paired Data

For the paired t test we have a similar derivation as for one-sample t test and $df = n-1$, however, n would be the number of pairs and not the total sample size. The standard deviation should be the standard deviation of the difference between two groups denoted with S_d, however, if it is not known it can be calculated by adjusting for the assumed correlation ρ between groups Y_1 and Y_2

$$S_d^2 = S_1^2 + S_2^2 - 2\rho S_1 S_2.$$

Both possibilities are implemented into the SampleSizeR. There is a built-in function in R to calculate the needed sample size, but it can also be done manually in a similar way as before.

```
power.t.test(n = n, delta = delta, sd = sd,
             sig.level = sig.level, type = "paired")
```

2.2 Anova

2.2.1 Oneway Anova

Anova, which stands for analysis of variance, is somewhat similar to a t test and can reject the same $H_0 : \mu_1 = \mu_2$, where μ_i is the mean of some random value Y_i with a treatment level $i = \{1, 2\}$. For example, the H_0 in a classical scenario would be that the outcome of patients treated with a treatment is equal to the outcome of patients treated with a placebo. However, the approach with Anova on how the H_0 is rejected is different and therefore the capabilities of Anova are wider compared to a t test. In the case of oneway Anova the H_0 can have $a > 2$ levels of a factor e.g. $H_0 : \mu_1 = \cdots = \mu_a$, which is not supported by the t test. Anova is nothing else than an F test, which allows us to determine whether the ratio of two variances is greater than some critical value of an $F_{df_1,df_2,\lambda}$ distribution, where df_1 is the number of levels of a factor (a) minus one and df_2 is the total sample size $(a \times n)$ minus a number of levels of a factor (a) and λ is the non-centrality parameter of the F distribution which is equal to 0 under H_0 or to the value presented in (1) (Dean et al., 2017). In this subsection, we consider the following model

$$Y_{ij} = \mu + \alpha_i + \epsilon_{ij},$$

where $i = \{1, \ldots, a\}$ is the treatment level, $j = \{1, \ldots, n_i\}$ is the observation and $\epsilon_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$. In general, the test statistics F is defined by

$$F = \frac{\text{SS(Between)}/df_1}{\text{SS(Within)}/df_2} = \frac{\text{MS(Between)}}{\text{MS(Within)}}.$$

For the power analysis and hence for the sample size calculation, we have a similar situation as with a t test, namely to calculate the power we need to define the non-centrality parameter

$$\lambda = \frac{\sum_i n_i(\alpha_i - \bar{\alpha})^2/a}{\sigma^2}. \tag{1}$$

We can simplify the term $(\alpha_i - \bar{\alpha})^2$ to $\Delta^2/2$. It is rooted in the understanding that the most challenging scenario to evaluate is when the effects of two-factor groups, for example, the first

and last one, differ by an amount represented by Δ, while the effects of the other groups fall midway between them (Dean et al., 2017)[Chapter 3.6]. This can be written as

$$\begin{aligned} \mu + \alpha_2 = \mu + \alpha_3 = \cdots = \mu + \alpha_{a-1} = c \\ \mu + \alpha_1 = c + \Delta/2, \quad \text{and } \mu + \alpha_a = c - \Delta/2. \end{aligned} \tag{2}$$

Additionally, in case of equal group size $n_i = n,\ \forall i = 1, \ldots, a$ equation (1) can be written as

$$\lambda = \frac{n\Delta^2}{2\sigma^2}.$$

Note that Δ is a minimum difference of any two means we are interested in.
Using λ we can calculate the power also using F_{crit}, df_1, df_2. In R this can be done manually or with the help of some packages e.g. `daewr` by Lawson and Krennrich (2023). The oneway Anova table can be found in Table 2.1, where with $\bar{Y}_{i\cdot}$ we represent the mean within the group i and with $\bar{Y}_{\cdot\cdot}$ the mean within the complete sample.

```
#Manually

1 - pf(F_crit, df1, df2, lambda)

# Package

require(daewr)
Fpower1(alpha = sig.level, nlev = a, nreps = n,
        Delta = delta, sigma = sd)
```

Table 2.1: Oneway Anova table.

Source of Variation	Degrees of Freedom	Sum of Squares	Mean Square	F Statistic
Between	$a-1$	SS(Between) = $\sum_i n\left(\bar{Y}_{i\cdot} - \bar{Y}_{\cdot\cdot}\right)^2$	MS(Between) = $\frac{\text{SS(Between)}}{a-1}$	$F = \frac{\text{MS(Between)}}{\text{MS(Within)}}$
Within	$a(n-1)$	SS(Within) = $\sum_i \sum_j \left(Y_{ij} - \bar{Y}_{i\cdot}\right)^2$	MS(Within)= $\frac{\text{SS(Within)}}{a(n-1)}$	
	$an-1$	SS(Total) = $\sum\sum \left(Y_{ij} - \bar{Y}_{\cdot\cdot}\right)^2$		

2.2.2 Twoway Anova

By adding factor B with b levels we consider the following model with **no interaction** term between two treatment factors

$$Y_{ijk} = \mu + \alpha_i + \beta_j + \epsilon_{ijk}, \tag{3}$$

where $i = \{1, \ldots, a\}$ is the level of the treatment factor A and $j = \{1, \ldots, b\}$ is the level of treatment factor B, i.e., a is a number of levels of factor A and b of factor B, $k = \{1, \ldots, n_i\}$ is the observation and $\epsilon_{ijk} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2)$. In the context of the sample size calculation using twoway Anova is similar to the oneway Anova, where the former's main difference is the degrees of freedom for the denominator while still assuming $n_i = n\ \forall i$. To perform the power analysis we would need to calculate a non-centrality parameter λ for each factor

$$\lambda_A = \frac{bn\Delta_A^2}{2\sigma^2},$$

$$\lambda_B = \frac{an\Delta_B^2}{2\sigma^2}.$$

Now using λ_A or λ_B we can calculate the power for either factor using F_{crit}, df_1, df_2. However, df_2 is now different than as in oneway Anova. In the case of **no interaction** term as in model (3) the df_2 is equal to $abn - a - b + 1$ (Dean et al., 2017) and the overview of the degrees of freedom can be found in Table 2.2.

Table 2.2: Overview of degrees of freedom in a twoway Anova with no interaction term.

Source	Degrees of Freedom
A	$a-1$
B	$b-1$
Within	$abn-a-b+1$
Total	$abn-1$

The calculation can be performed manually in R or using `daewr` package.

```
# Manually
n <- ... # Some value
a <- ... # Nr. of levels of exposure a
b <- ... # Nr. of levels of exposure b

cssa <- (delta_a^2)/2
nca <- b * (n * cssa)/(sigma^2)
cssb <- (delta_b^2)/2
ncb <- a * (n * cssb)/(sigma^2)
df2 <- n - a - b + 1
power2a <- 1 - pf(Fcrit(sig.level, dfa, df2), dfa, df2, nca)
power2b <- 1 - pf(Fcrit(sig.level, dfb, df2), dfb, df2, ncb)

# With Fpower() from the package "daewr"
power2a <- Fpower(alpha = sig.level, nu1 = dfa, nu2 = df2,
                  nc = nca)
power2b <- Fpower(alpha = sig.level, nu1 = dfb, nu2 = df2,
                  nc = ncb)
```

In case of an **existing interaction** between the two factors the model (3) becomes:

$$Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \epsilon_{ijk}. \tag{4}$$

The calculation of the power can be done using the same package `daewr` as in Section 2.2.1 in R.

```
Fpower2(alpha = sig.level, nlev = c(a,b), nreps = n,
        Delta = delta, sigma = sd)
```

The overview of the degrees of freedom in a twoway Anova with interaction can be found below in Table 2.3. As we can see in it the main difference is df_2 which is now equal to $ab(n-1)$.

2.3 Marginal Models and Conditional Models

In this section, we will elaborate on the difference between marginal and conditional models.

Table 2.3: Overview of degrees of freedom in a twoway Anova with an interaction term.

Source	Degrees of Freedom
A	$a-1$
B	$b-1$
A × B	$(a-1)\times(b-1)$
Within	$ab(n-1)$
Total	$abn-1$

One of the major steps when choosing between marginal or conditional models is to make it clear and understand what the data analysis should answer by the end of the study. This question should be known in advance since the post-hoc interpretation of the coefficients is not an appropriate approach. Typically in epidemiological studies, we want to infer about the population average, e.g. treatment group vs control group as by Diggle (2002)[Chapter 7], however, it is not carved in stone and can differ from study to study. Depending on the clinical question the choice of the model becomes very crucial. In the next three sections, we will discuss marginal models and conditional models, their differences, and which model is relevant to a certain scenario.

However, before jumping into details, we shall introduce the link function $g(\cdot)$ which has the same interpretation as in generalized linear models such that

$$g(E[\boldsymbol{Y}]) = \boldsymbol{X}\boldsymbol{\beta}. \tag{5}$$

In the main design that will be discussed further $g(\cdot)$ is assumed to be an identity function, but in Section 2.3.5 a logit link function is used for a binary response.

2.3.1 Matrix Notation

A brief introduction to the matrix notation of a statistical model. We define the outcome as $\boldsymbol{Y}$ of size $n \times 1$, the data as $\boldsymbol{X}$ of size $n \times (p+1)$ and the coefficients $\boldsymbol{\beta}$ of size $(p+1)\times 1$. A single entry of the matrix $\boldsymbol{X}$ would be denoted as $\boldsymbol{X}_{ik}$, where $i = \{1,\dots,n\}$ would be row and $k = \{0,\dots,p\}$ would be the column. We also define n as the total sample size of the study, which also equals the number of observations, and $p+1$ is the number of covariates including the intercept.
We can either define a model as $g(E[\boldsymbol{Y}]) = \beta_0\boldsymbol{X}_0 + \cdots + \beta_k\boldsymbol{X}_k + \cdots + \beta_p\boldsymbol{X}_p$, where $\boldsymbol{X}_k$ is kth column or by using a matrix notation in a shorter way as $g(E[\boldsymbol{Y}]) = \boldsymbol{X}\boldsymbol{\beta}$, where in the latter case it would look as following

$$\begin{bmatrix} E[Y_1] \\ E[Y_2] \\ \vdots \\ E[Y_n] \end{bmatrix} = \begin{bmatrix} 1 & X_{11} & \dots & X_{1p} \\ 1 & X_{i1} & \dots & X_{ip} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_{n1} & \dots & X_{np} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}.$$

If we are dealing with a setting that involves clustered data or repeated measurements, we have an additional index $j = \{1,\dots,r\}$, so $\boldsymbol{X}_{ijk}$ represents the jth observation within the cluster i for the covariate k. Whereas now n represents the number of clusters, which no more equals the number of total observations. We still can write it down in the same manner as the equation (5) above. But $\boldsymbol{X}$ has now a different size, namely $nr \times (p+1)$, hence it looks slightly different and can be represented as

$$\begin{bmatrix} E[Y_{11}] \\ \vdots \\ E[Y_{1r}] \\ \vdots \\ E[Y_{n1}] \\ \vdots \\ E[Y_{nr}] \end{bmatrix} = \begin{bmatrix} 1 & X_{111} & \dots & X_{11p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_{1r1} & \dots & X_{1rp} \\ \vdots & \vdots & \vdots & \\ 1 & X_{n11} & \dots & X_{n1p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_{nr1} & \dots & X_{nrp} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}. \tag{6}$$

2.3.2 Marginal Models

As mentioned above a typical research question in epidemiological studies is about the population average. Some very simple examples:

- How does the BMI change with age in children?
- What is the effect of a new ophthalmological treatment based on BCVA score on the patients?

In these two small examples, the research focus lies on the whole population whether that would be children, patients, or any population in general. There are multiple ways to model this type of research question. Some of them are based on the likelihood optimization methods, such as the Bahadur Model, however, their computation may be very tedious and for that reason, they are not used often (Molenberghs and Verbeke, 2005)[Chapter 7]. One of the most popular marginal models is the generalized estimated equations (GEE) by Liang and Zeger (1986) which is estimated via the quasi-likelihood approach. Since other models are not very well known, this chapter will focus on the GEEs. The model is defined as following, where $\boldsymbol{Y}_{ij}$ is for example a BCVA score of patient $i = \{1, \dots, n\}$ on visit $j = \{1, \dots, r\}$ and $g(\cdot)$ is an identity function

$$g(E[\boldsymbol{Y}_{ij}]) = E[\boldsymbol{Y}_{ij}] = \boldsymbol{\mu}_{ij} = \boldsymbol{X}_{ij}\boldsymbol{\beta}, \tag{7}$$

where $\boldsymbol{X}_{ij}$ is not a matrix, but a vector of size $1 \times (p+1)$ and $\boldsymbol{\beta}$ remains a $(p+1) \times 1$ vector. In the case of the marginal normal model, it is assumed that the outcome of patient i is

$$\boldsymbol{Y}_i \sim \mathcal{N}(\boldsymbol{X}_i\boldsymbol{\beta}, \boldsymbol{V}_i),$$

where $\boldsymbol{X}_i$ is a $r \times (p+1)$ matrix with all observation within the cluster of the patient i and $\boldsymbol{V}_i$ is the covariance matrix of the observations of the patient i and has the size of $r \times r$. It is also assumed that $\boldsymbol{V}_i = \boldsymbol{V} \, \forall i$, hence the index will be removed for easier notation in further equations. The covariance matrix $\boldsymbol{V}$ can be decomposed in

$$\boldsymbol{V} = \boldsymbol{D}^{1/2}\boldsymbol{R}(\rho)\boldsymbol{D}^{1/2},$$

where $\boldsymbol{D}$ is a diagonal matrix with r marginal estimated variances and $\boldsymbol{R}(\rho)$ is a correlation matrix of some type and $\rho \in (-1/r, 1)$. The correlation matrices that are typically used are the Auto Regressive (AR1) correlation structure, Compound Symmetry (CS), also called Exchangeable correlation structure, and less common during the modeling, but common during the sample size calculation is the Unstructured (UN) correlation matrix. All of them can be seen below. For an easier notation $\boldsymbol{R}(\rho)$ will be denoted with $\boldsymbol{R}$.

$$\boldsymbol{R}_{AR1} = \begin{pmatrix} 1 & \rho & \rho^2 \\ \rho & 1 & \rho \\ \rho^2 & \rho & 1 \end{pmatrix}, \quad \boldsymbol{R}_{CS} = \begin{pmatrix} 1 & \rho & \rho \\ \rho & 1 & \rho \\ \rho & \rho & 1 \end{pmatrix}, \quad \boldsymbol{R}_{UN} = \begin{pmatrix} 1 & \rho_1 & \rho_2 \\ \rho_1 & 1 & \rho_3 \\ \rho_2 & \rho_3 & 1 \end{pmatrix}. \tag{8}$$

The GEE's score function is defined as

$$S(\boldsymbol{\beta}, \boldsymbol{R}) = \sum_{i=1}^{n} \left(\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\beta}} \right)' (\boldsymbol{D}^{1/2} \boldsymbol{R} \boldsymbol{D}^{1/2})^{-1} (\boldsymbol{Y}_i - \boldsymbol{X}_i \boldsymbol{\beta}), \tag{9}$$

which is optimized for proper $\boldsymbol{R}$ and $\boldsymbol{\beta}$ by setting equation (9) to 0. In order not to deviate from the main purpose of the thesis, we will not go into details of the estimation process. However, an important feature that results from the estimation is that the initial misspecification of the correlation matrix $\boldsymbol{R}$ is allowed up to a certain degree. It is, however, still important that the chosen correlation matrix structure is in alignment with the study design (Molenberghs and Verbeke, 2005)[Chapter 8].

2.3.3 Conditional Models

Opposed to marginal models, conditional models can answer clinical questions on a subject-specific level, where the subject is defined by $i = \{1, \ldots, n\}$ and its subject-specific effect by c_i. For example, it might be interesting for a doctor to see the effect of the treatment, which is applied multiple times, for a specific patient. The model is given by

$$\overbrace{g(E[\mathbf{Y}_{ij}])}^{1\times 1} = \overbrace{\underbrace{\mathbf{X}_{ij}}_{1\times(p+1)} \underbrace{\boldsymbol{\beta}}_{(p+1)\times 1}}^{1\times 1} + \overbrace{\underbrace{\mathbf{Z}_{ij}}_{1\times q} \underbrace{\boldsymbol{c}_i}_{q\times 1}}^{1\times 1},$$

where q is the number of random effects and $\boldsymbol{c}_i$ is typically assumed to be $\overset{\text{iid}}{\sim} \mathcal{N}(0, \boldsymbol{\Sigma}_{\boldsymbol{c}})$, where $\boldsymbol{\Sigma}_{\boldsymbol{c}}$ is $\text{Cov}(\boldsymbol{c}_i)$ of size $q \times q$. Unlike the marginal quasi-likelihood models, it requires a full joint likelihood function, which can be used to estimate the parameter of interest. Some classical approaches can be used for the estimation such as MLE (Diggle, 2002)[Chapter 8].

2.3.4 Marginal Models vs Conditional Models

Now, once both models are introduced, we can discuss which model should be used. The best answer would be "it depends", however, that is not very helpful. When we use an identity link function $g(\cdot)$, we can state, that it does not matter. Because if the link function is an identity function, the marginal expectation is equivalent to the expectation of all conditional expectations, which can be shown by

$$E[\boldsymbol{Y}] = E[E[\boldsymbol{Y}_i | \boldsymbol{c}_i]].$$

However, if the link function is not identity, but e.g., a logit function, due to Jensen's inequality the above condition does not hold anymore, and not only modeling results but also estimated coefficients are no more directly comparable. Therefore for non-Gaussian data, the model choice is very crucial. There are different opinions on which model to choose. A very detailed discussion can be found in Muff et al. (2016), where they conclude that there are some properties of the conditional model that speak in favor of it. However, according to Molenberghs and Verbeke (2005)[Chapter 16] and the introduction above the interpretation of the coefficients is different, meaning it would make more sense to make a choice based on the research question and not some additional properties, because even if the model is perfect from a statistical point of view as long as it does not answer the initial question it is useless for the cause. Another factor that might influence the choice is the possibility of sample size calculation for GEE without the need to run simulations, as described in Diggle (2002)[Chapter 8].

2.3.5 Sample Size Calculation for Marginal Models

The complete procedure of the sample size derivation is described in Liu and Liang (1997). The defined model differs a bit from the one presented in equation (7)

$$g(E[\boldsymbol{Y}_{ij}]) = g(\boldsymbol{\mu}_{ij}) = \boldsymbol{X}_{ij}\boldsymbol{\beta} + \boldsymbol{Z}_{ij}\boldsymbol{\tau},$$

where $\boldsymbol{\tau}$ is now some nuisance parameter, that we are not particularly interested in, such as the intercept of the model or other covariates. Whereas $\boldsymbol{\beta}$ is a $h \times 1$ vector, which we are interested in, such as the treatment effect and $h+1$ are the treatment levels. The sample size calculation is based on quasi-score T statistics, which resembles the Wald statistics, however, differs since GEE uses a quasi-likelihood approach. As a reminder the score function $S(\cdot)$ is the same as in equation (9) and the Hessian matrix $\boldsymbol{\Sigma}$ of the score function is

$$\boldsymbol{\Sigma} = \operatorname{Cov}\left[S\left(\boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{R}\right)\right]. \tag{10}$$

Before going further let's introduce some new notations $\boldsymbol{u}_l$, $\boldsymbol{v}_l$ and π_l that will be used, where $l = \{1, \ldots, L\}$. It will be easier to understand it with a simple example. Imagine we have a situation with four repeated measurements per patient and two treatment levels (placebo coded with 0 and drug A coded with 1). L is the number of all possible allocation combinations of β and τ. If the patient gets a placebo during the study, then his/her $\boldsymbol{u}_1 = (0,0,0,0)$ and $\boldsymbol{v}_1 = (1,1,1,1)$. Here $\boldsymbol{u}_1$ indicates a possible allocation of the treatment within the cluster and $\boldsymbol{v}_1$ the allocation of the intercept within the same cluster. Notice that the length of both vectors is the same as the number of repeated measurements per patient. If the patient receives the drug his/hers $\boldsymbol{u}_2 = (1,1,1,1)$ and $\boldsymbol{v}_2 = (1,1,1,1)$. In that particular design, no other combinations are possible, hence $L = 2$. The π_l is the probability of a patient being assigned to one of the combinations. As a reminder, $\boldsymbol{R}$ is still the correlation matrix as defined in equation (8).

In the case of GEE, the T statistics is defined as

$$T = S(\boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{R})\boldsymbol{\Sigma}^{-1}S(\boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{R}),$$

where T follows asymptotically a non-central $\chi^2_h(\nu)$ distribution with h degrees of freedom under H_1. The non-centrality parameter ν is

$$\nu = \boldsymbol{\xi}'\boldsymbol{\Sigma}^{-1}\boldsymbol{\xi}, \tag{11}$$

where $\boldsymbol{\xi}$ is the expected value of the score function $S(\cdot)$ under H_1 which is defined as

$$\boldsymbol{\xi} = \mathrm{E}\left[S\left(\boldsymbol{\beta}, \boldsymbol{\tau}, \boldsymbol{R}\right)\right].$$

Calculating $\boldsymbol{\xi}$ and $\boldsymbol{\Sigma}$ would be quite difficult, hence Liu and Liang (1997) discovered a way to approximate the former as following

$$\tilde{\boldsymbol{\xi}} = n\sum_{l=1}^{L} \pi_l \mathbf{P}_l \boldsymbol{V}^{-1}\left(\boldsymbol{\mu}_l^1 - \boldsymbol{\mu}_l^0\right),$$

where

$$\begin{aligned}
\boldsymbol{\mu}_l^1 &= g^{-1}(\boldsymbol{X}_{ij}\boldsymbol{\beta}_1 + \boldsymbol{Z}_{ij}\boldsymbol{\tau}_1) \quad \text{under } H_1 \\
\boldsymbol{\mu}_l^0 &= g^{-1}(\boldsymbol{X}_{ij}\boldsymbol{\beta}_0 + \boldsymbol{Z}_{ij}\boldsymbol{\tau}_0) \quad \text{under } H_0 \\
\mathbf{P}_l &= \left(\frac{\partial \boldsymbol{\mu}_l}{\partial \boldsymbol{\beta}}\right)' - \boldsymbol{I}_{\beta\tau}\boldsymbol{I}_{\tau\tau}^{-1}\left(\frac{\partial \boldsymbol{\mu}_l}{\partial \boldsymbol{\tau}}\right)' \\
\boldsymbol{I}_{\beta\tau} &= \sum_{l=1}^{L}\left(\frac{\partial \boldsymbol{\mu}_l}{\partial \boldsymbol{\beta}}\right)'\boldsymbol{V}^{-1}\left(\frac{\partial \boldsymbol{\mu}_l}{\partial \boldsymbol{\tau}}\right) \\
\boldsymbol{I}_{\tau\tau} &= \sum_{l=1}^{L}\left(\frac{\partial \boldsymbol{\mu}_l}{\partial \boldsymbol{\tau}}\right)'\boldsymbol{V}^{-1}\left(\frac{\partial \boldsymbol{\mu}_l}{\partial \boldsymbol{\tau}}\right)
\end{aligned}$$

and the Hessian matrix $\boldsymbol{\Sigma}$ as in (10) is approximated as following

$$\tilde{\boldsymbol{\Sigma}} = n \sum_{l=1}^{L} \pi_l \mathbf{P}_l \boldsymbol{V}^{-1} \operatorname{Cov}(\mathbf{Y}_l) \boldsymbol{V}^{-1} \mathbf{P}_l'.$$

Recalling the equation (11) we can extract n from the approximations of the $\tilde{\boldsymbol{\Sigma}}$ and $\tilde{\boldsymbol{\xi}}$ such that

$$\nu \approx n \tilde{\boldsymbol{\xi}}' \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\boldsymbol{\xi}}, \tag{12}$$

where both $\tilde{\boldsymbol{\Sigma}}$ and $\tilde{\boldsymbol{\xi}}$ are the same as above but without n at the beginning.

Identity link-function $g(\cdot)$
In the case of a model with an identity link function and repeated/clustered observations the score function reduces to

$$S = \sigma^{-2} \sum_{i=1}^{n} \mathbf{X}_i' \boldsymbol{R}^{-1} \left(\mathbf{Y}_i - (\mathbf{X}_i \boldsymbol{\beta} + \mathbf{Z}_i \boldsymbol{\tau}) \right).$$

Therefore, further simplifications in a linear model case can be made for

$$\tilde{\boldsymbol{\Sigma}} = \sigma^{-2} \sum_{l=1}^{L} \pi_l \left(\mathbf{u}_l' - \boldsymbol{I}_{\beta\tau} \boldsymbol{I}_{\tau\tau}^{-1} \mathbf{v}_l' \right) \boldsymbol{R}^{-1} \left(\mathbf{u}_l - \mathbf{v}_l \boldsymbol{I}_{\tau\tau}^{-1} \boldsymbol{I}_{\beta\tau}' \right)$$

and

$$\tilde{\boldsymbol{\xi}} = \tilde{\boldsymbol{\Sigma}} \boldsymbol{\beta}.$$

In that case $\boldsymbol{I}_{\beta\tau} = \sigma^{-2} \sum_{l=1}^{L} \pi_l \mathbf{u}_l' \boldsymbol{R}^{-1} \mathbf{v}_l$ and $\boldsymbol{I}_{\tau\tau} = \sigma^{-2} \sum_{l=1}^{L} \pi_l \mathbf{v}_l' \boldsymbol{R}^{-1} \mathbf{v}_l$. As a reminder, $\mathbf{u}_l$ and $\mathbf{v}_l$ are the vectors that represent a possible allocation of covariates within one cluster, as described at the beginning of this section.
Using the terms above, we can calculate the sample size by plugging all values into the equation introduced in (12)

$$n = \nu / \left(\tilde{\boldsymbol{\xi}}' \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\boldsymbol{\xi}} \right) = \nu / \left((\tilde{\boldsymbol{\Sigma}} \boldsymbol{\beta})' \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\boldsymbol{\Sigma}} \boldsymbol{\beta} \right) = \nu / \left(\boldsymbol{\beta}' \tilde{\boldsymbol{\Sigma}} \boldsymbol{\beta} \right). \tag{13}$$

We will get a proper sample size for a given power and Type-I error α, which are incorporated into the ν.

Identity link-function $g(\cdot)$ **and** $h = 1$
Let us consider the most basic model with no nuisance parameters $\boldsymbol{\tau}$ except for the intercept and only two treatment factors where $h + 1$ are the treatment levels. In that case, $h = 1$, which is quite a common situation in epidemiology in a randomized setting, the non-centrality parameter $\nu = \left(z_{1-\alpha/2} + z_{Power} \right)^2$ and the whole term (13) simplifies to

$$n = \frac{\left(z_{1-\alpha/2} + z_{\text{Power}} \right)^2 \sigma^2}{\pi_0 \pi_1 \beta^2 \left(\mathbf{1}' \boldsymbol{R}^{-1} \mathbf{1} \right)}.$$

Here $\boldsymbol{R}$ is a correlation matrix of a common type introduced in (8), β is the expected effect of the treatment, σ is a constant variance for every measurement and π_0 with $\pi_1 = 1 - \pi_0$ respectively is the probability of the allocation to one of the treatment groups. If treatment is allocated evenly π_0 would be 0.5.

Logit link-function $g(\cdot)$ **and** $h = 1$

A similar situation but with binary outcome, expression (13) simplifies to

$$n = \frac{\left(z_{1-\alpha/2} + z_{\text{Power}}\right)^2 \left(\pi_1 p_0 \left(1 - p_0\right) + \pi_0 p_1 \left(1 - p_1\right)\right)}{\pi_0 \pi_1 \left(p_1 - p_0\right)^2 \left(\mathbf{1}' \boldsymbol{R}^{-1} \mathbf{1}\right)}$$

with p_1 being a probability of the outcome in a treatment group and vice versa for p_0.

In general, it is important to keep in mind that the calculated sample size n is the **total** sample size for the study, representing a number of clusters.

As a short proof of concept we simulate a time repeated measurement design with 4 time points with an exchangeable correlation matrix and $\rho = 0.4$, $\sigma^2 = 9$, $\beta = 1.5$ and $n = 25$. Using the formula above, we would expect a power to be equal to approximately 40%.

```
set.seed(30)
p_val <- numeric()
for(i in 1:300){
  SimData <- data.frame(id = rep(1:25, each = 4))
  SimData <- SimData %>%
    group_by(id) %>%
    mutate(A = rbinom(1, 1, 0.5)) %>%
    mutate(time = 1:4) %>%
    mutate(Y = rmvnorm(n = 1, mean = 10 + A * 1.5,
                       sigma = (diag(4) * 3) %*%
                         toeplitz(c(1, 0.4, 0.4, 0.4)) %*%
                         (diag(4) * 3))) %>%
    group_by(id, A, time) %>%
    mutate(Y2 = Y[, time]) %>%
    select(-Y)

  res <- summary(geeglm(Y2 ~ factor(A), id = id,
                        data = SimData, corstr = "exchangeable"))

  p_val[i] <- res$coefficients$`Pr(>|W|)`[2]
}
power_gee <- mean(na.omit(p_val) < 0.05)
```

After running the simulation, the power we get is 0.42, which is almost identical to the expected power.

Examples for GEE

We want to test how a new vitamin influences the learning effect of students by giving them a test that provides a grade from 0 to 20. As a researcher, we know that we need 4 visits to properly measure the effect of the vitamin, and from the previous studies, we could find out that the results of the same student have a correlation of approximately 0.5 (hence exchangeable correlation matrix) and the variance of the results stays more or less constant across all visits and equals to 10. We want to see an effect of at least 3 points for the students in the vitamin group. How many students should be considered if the power should be 0.8 and the Type-I error (α) is 0.05? By plugging the values into SampleSizeR we see that at least 22 students are needed to conduct a properly randomized study to observe the treatment effect of the vitamin with a power of 80%.

For the binary case, we can think of a similar scenario, but instead of having points for the test, we would dichotomize the outcome into passed/failed. We can assume that without the vitamin

only 20% of students would pass the exam and with the vitamin 60% and the rest stays the same. Putting these values into SampleSizeR we see that we need 28 students to achieve the power of 80%.

2.4 Complete Block Design

In this section, we will introduce the block design and an application of the marginal GEE model in such design with corresponding sample size calculation.

Statistics such as many other fields has a lot of trade-offs. One of them can be also found in clinical studies, especially during the study design planning phase. A researcher wants to have an estimate as precise as possible, hence reducing the variance to its limits. This can be achieved by making the experimental units as homogeneous as they can be. On the other hand, if you do so, the final statement of the research cannot be generalized and works only for this specific group of experimental units, which is rarely the goal of the study. As per Lawson (2014) one way of (partially) combining these two goals is to use a design called **Randomized Complete Block Design**. To achieve homogeneity, but keep the conclusion generalizable the researcher should find multiple groups, that consist of homogeneous experimental units and afterwards randomize the treatment within that group. For example, a cage with four mice from the same litter, makes them homogeneous, however randomizing four different treatment levels within one cage and having multiple cages with different litters, makes the result generalizable. If we have more than one replicate of each treatment level, this design is called **Generalized Complete Block Design**. As an example imagine the same cage, but now with eight mice in it and random two receive the first treatment, the other two receive the second treatment, and so on.

The result can be modeled (in the case of an identity link function) as following:

$$Y_{ijk} = \mu + \alpha_i + \beta_j + \epsilon_{ijk},$$

where α_i is the treatment effect, β_j is the blocking effect and $\epsilon_{ijk} \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma)$ is an error. This model allows us to treat this type of design with a twoway Anova and use methods, that are described in Section 2.2.2. However, some adjustments should be made. Since we are only interested in a treatment effect, when computing the sample size, we are always calculating the sample size based on the treatment effect only, regardless of what the blocking effect is, while adjusting the degrees of freedom accordingly. We can think of that also as of oneway Anova but with a different way of calculating df_2.

An additional way to model the blocking factor is to use a random effect for blocking, which is preferred by many statisticians as per Dixon (2016). In reality, according to Casella (2008)[Chapter 4] it does not matter which approach is chosen since it makes no difference in the estimate of the treatment effect and its variance. The equation will look similar, but the blocking effect will become an additional error term, and not a fixed coefficient

$$Y_{ijk} = \mu + \alpha_i + \beta_j + \epsilon_{ijk}. \tag{14}$$

Here $\beta_j \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2_\beta)$ and it is independent of ϵ_{ijk}. This can be modeled once again using conditional models such as linear mixed-effects models or marginal models such as GEE. On a side note, in the case of GEE, we do not model the blocking factor β per se, but rather using it as an indication for the cluster to model $\boldsymbol{R}$. Using GEE for that purpose, results in a similar advantage as described in Section 2.3.5, namely a way to calculate the sample size using a close-end solution. However, before jumping into the process of power analysis, some additional information should be discussed. First, the correlation matrix described in the mentioned section should be slightly

adjusted. The AR correlation matrix does not make intuitive sense in the case of a complete block design, it can be omitted in further discussion. The exchangeable correlation matrix makes more intuitive sense and we can either assume a classical $2m \times 2m$ exchangeable matrix $\boldsymbol{R}$, e.g., for $m = 2$, where m is the number of replicates with the same treatment level within one block

$$\mathbf{R} = \begin{pmatrix} 1 & \rho & \rho & \rho \\ \rho & 1 & \rho & \rho \\ \rho & \rho & 1 & \rho \\ \rho & \rho & \rho & 1 \end{pmatrix}, \quad \rho \in \left(-\frac{1}{2m-1}, 1 \right)$$

Or we can assume an additional level of correlation between replicates within the same treatment level which we define as ϕ. Since the treatment level is randomized within the block, the matrix below is an example if the treatment is randomized as $\boldsymbol{u}_1 = (1, 1, 0, 0)$, where $\boldsymbol{u}_1$ is one of the possible allocations of the treatment within the cluster/block as explained in Section 2.3.5.

$$\mathbf{R}'_l = \begin{pmatrix} 1 & \rho+\phi & \rho & \rho \\ \rho+\phi & 1 & \rho & \rho \\ \rho & \rho & 1 & \rho+\phi \\ \rho & \rho & \rho+\phi & 1 \end{pmatrix}, \tag{15}$$

where the conditions for the matrix $\mathbf{R}'_l$ to be positive definite can be found in the Appendix A.1. During the calculation of the sample size, $\mathbf{R}'_l$ will be adjusted accordingly for each $l = \{1, \ldots, L\}$

$$\mathbf{R}'_l = \phi\Big(\boldsymbol{u}_l\boldsymbol{u}'_l + (\mathbf{1} - \boldsymbol{u}_l)(\mathbf{1} - \boldsymbol{u}_l)'\Big) + \mathbf{R},$$

where $\mathbf{1}$ is a $2m \times 1$ vector.
Whereas π_l is the probability of each possible combination of $\boldsymbol{u}_l$ and is calculated as

$$\pi_l = \binom{2m}{m}^{-1}.$$

There might be a way to have a more simplified formula, similar to the continuous outcome in the two-sample scenario shown in Section 2.3.5, however, the general way of calculating the sample size works very well. To show that, we will run simulations with randomly picked parameters. We use simulations since it is one of the possible and quite reliable ways to calculate the power if no close-end solution is available (Arnold et al., 2011), hence we can use it as a comparison for the suggested method.

The proposed closed-end algorithm does not always converge, which typically happens for scenarios where the simulated power is lower than 20%. Assuming that scenarios where such low power is barely needed, the algorithm is suitable for the most common situations.

This passage refers mainly to the code, which can be found below. The first step is to create a function (`GeeSimualtion`), which will run the simulation-based power analysis. The datasets are simulated using known parameters and the above-described formula with a random intercept per block. Each simulation-based power calculation will generate 500 datasets and run through `geeglm()` function of the package `geepack` by Yan (2002) to model a GEE. Afterwards, we calculate how many results for the treatment coefficient (A) have been significant, and the number of all significant results given $\alpha = 5\%$ is divided by the number of simulations. The result represents the simulated power for the provided parameters. The second function (`GCBSampleSizeR`) is an `R` implementation of the method by Liu and Liang (1997) as described in Section 2.3.5 with adjustments discussed in this section. Part of it was taken from the package `longpower` by Hu et al. (2021) and adjusted to the current needs. Once the functions are defined we generate lists for different parameters such as Δ, ρ, σ, n and m. Here we replace the notation of the treatment

effect α defined in the model (14) with Δ in order not to confuse it with a Type-I error α. In each simulation iteration, a random parameter is picked from the list. Here we set the Type-I error to 5% and ϕ to 0. Afterward, the power based on two approaches is calculated.

```
# This part defines a function to simulate the dataset and calculate the power
# based on the simulation

GeeSimulation <- function(delta, n, m, sigma, cs_param, silent = TRUE){
  Sigma1 <- (diag(m*2) * sigma) %*% toeplitz(c(1, rep(cs_param, m*2-1))) %*%
    (diag(m*2) * sigma)
  Sigma_fin <- as.matrix(bdiag(rep(list(Sigma1), n)))
  p_val <- numeric()
  for(i in 1:500){
    SimData <- data.frame(block = rep(1:n, each = m*2))
    A <- matrix(replicate(n, sample(rep(c(0,1), each = m ), m*2)),ncol = 1)
    SimData$A <- A
    Y <- rmvnorm(n = 1, mean = 10 + A * delta,
                 sigma = Sigma_fin)
    SimData$Y <- t(Y) + matrix(replicate(n, rep(rnorm(1, 0, sd = 1.5),
                                                m*2)),ncol = 1)
    res <- summary(geeglm(Y ~ A, id = block, data = SimData,
                          corstr = "exchangeable"))
    p_val[i] <- res$coefficients$`Pr(>|W|)`[2]
    if(i %% 50 == 0 & silent) print(i)
  }
  return(mean(na.omit(p_val) <= 0.05))
}

# This part defines the function which applies the method introduced in the
# previous section.

GCBSampleSizeR <-  function(delta, u, v, sigma, prob, corStr, n = NULL,
                            m, power = NULL, phi = NULL,  alpha = 0.05){

  if(length(sigma) == 1) sigma <- rep(sigma, 2 * m)
  corStr.list <- lapply(1:length(u), function(i) corStr)
  corStrInv <- lapply(1:length(corStr.list),
                      function(i) {
                        corStr <- corStr.list[[i]]
                        if(!is.null(phi)){
                          ctrl_matrix <- (ifelse(u[[i]] == 0, 1, 0) %*%
                                            t(ifelse(u[[i]] == 0, 1, 0)))
                          trt_matrix <- (u[[i]] %*% t(u[[i]]))

                          corStr <- trt_matrix * phi + corStr + ctrl_matrix * phi
                          diag(corStr) <- 1
                        }
                        corStr <- (diag(2*m) * sigma) %*%
                          corStr %*% (diag(2*m) * sigma)
                        return(solve(corStr))
                        }
```

```
                      )
  n.body <- quote({
    IBetaLambda <- 0
    for (i in 1:length(u)) IBetaLambda <- IBetaLambda + prob[i] * t(u[[i]]) %*%
        corStrInv[[i]] %*% v[[i]]
    ILambdaLambda <- 0
    for (i in 1:length(u)) ILambdaLambda <- ILambdaLambda +
      prob[i] * t(v[[i]]) %*%
      corStrInv[[i]] %*% v[[i]]
    ILambdaLambdaInv <- solve(ILambdaLambda)
    Sigma <- 0
    for (i in 1:length(u)) Sigma <- Sigma + prob[i] *
      (t(u[[i]]) - IBetaLambda %*% ILambdaLambdaInv %*%
         t(v[[i]])) %*%
      corStrInv[[i]] %*% (u[[i]] - v[[i]] %*%
                            ILambdaLambdaInv %*% t(IBetaLambda))
    (qnorm(1 - alpha/2) + qnorm(power))^2/(delta %*% Sigma %*% delta)[1, 1]
  })
  if (is.null(n))
    n <- return(eval(n.body))
  else if (is.null(power))
    power <- return(uniroot(function(power) eval(n.body) - n,
                    c(0.01, 1), tol = .Machine$double.eps^2,
                    extendInt = "yes")$root)
}

tests <- 2500
n_test <- sample(7:15, tests, replace = TRUE)
m_test <- sample(1:4, tests, replace = TRUE)
delta_test <- sample(seq(1, 2.5,0.25), tests, replace = TRUE)
cs_param_test <- sample(seq(0, .9, 0.1), tests, replace = TRUE)
sigma_test <- sample(seq(2, 4, 0.25), tests, replace = TRUE)
res_sim <- res_formula  <- numeric()

set.seed(25)

for(i in 1:tests){
  n <- n_test[i]
  m <- m_test[i] # patients per factor level in block
  delta <- delta_test[i]
  cs_param <- cs_param_test[i]
  sigma <- sigma_test[i] # A constant
  res_sim[i] <- GeeSimulation(delta, n, m, sigma, cs_param, FALSE)
  corStr <- matrix(cs_param, nrow = m*2, ncol = m*2)
  diag(corStr) <- 1
  combis <- expand.grid(replicate(m*2, list(0:1)))
  sums <- rowSums(combis)
  combis <- as.matrix(combis[which(sums == m),])
  u <- split(combis,1:nrow(combis))
```

```
  v <- replicate(choose(m*2,m), list(rep(1, m*2)))
  prob <- rep(1/choose(m*2,m), choose(m*2,m))
  d <- try(GCBSampleSizeR(delta = delta, u = u, v = v,
                                        sigma = sigma, prob = prob,
                                        corStr = corStr, n = n, m = m))

  res_formula[i]  <- ifelse("try-error" %in% class(d), NA,
                            GCBSampleSizeR(delta = delta, u = u, v = v,
                                        sigma = sigma, prob = prob,
                                        corStr = corStr, n = n,
                                        m = m))
  res_formula[i] <- ifelse(res_formula[i] < 0.01 , 1, res_formula[i] )
  if(i %% 250 == 0) print(i)
}

results <- data.frame(n = n_test, m = n_test, delta = delta_test,
                      cs_param = cs_param_test,
           sigma = sigma_test, res_sim = res_sim,
             res_formula = round(res_formula,2),
           diff = round(res_sim - res_formula,2))
```

Looking at the results in Figure 2.2, which shows all points from the simulation against all points from the formula approach. Ideally, we would expect them to be perfectly aligned and sit on the light blue line, however, since a simulation has a stochastic element in it, it will never be perfect. It is possible to increase the number of replicates in the simulation, however, this will cost a lot of time to compile and will not change the results drastically. As we can see in that figure values are very close to each other, especially for typically used power of 80%. The same can be confirmed looking at $\mu_{\text{diff}} = 0.03$ and $\sigma_{\text{diff}} = 0.03$. Both are very small meaning that the difference between these two methods is minimal, however, the computational efficiency of simulated power is much lower and it requires a lot of time to compile.

2.5 Split-Plot Design

As per Lawson (2014)[Chapter 8], the split-plot design is very powerful and provides great help for researchers. It can increase power drastically for the main effect with hidden replications. Besides that, it also helps to detect interactions with other factors, which occur quite often in the real world. One of the reasons for the split-plot design existence is the fact that not all factors can be easily randomized, meaning that we cannot randomly assign a certain level to that factor. These hard-to-vary factors are called "whole-plot". Let's make an example. Imagine you are producing wine and want to see how the angle of the hill, where grapes grow and the level of irrigation influence the yield. Randomly setting the level of the irrigation is rather the easy-to-vary factor also known as ("sub-plot" or "split-plot") and the steepness of the hill is not something one can adjust for. This type of design also has its disadvantages. While the easy-to-vary factor gains power from this design, the hard-to-vary factor loses it.

The data coming from that design with complete randomization (CR) on the whole-plot level can be modeled according to Casella (2008)[Chapter 5] with

$$Y_{ijk} = \mu + \alpha_i + \omega_{(k)i} + \beta_j + (\alpha\beta)_{ij} + \epsilon_{k(ij)}, \tag{16}$$

where α_i and $i = \{1, \ldots, a\}$ is the whole-plot factor , β_j and $j = \{1, \ldots, b\}$ is the sub-plot

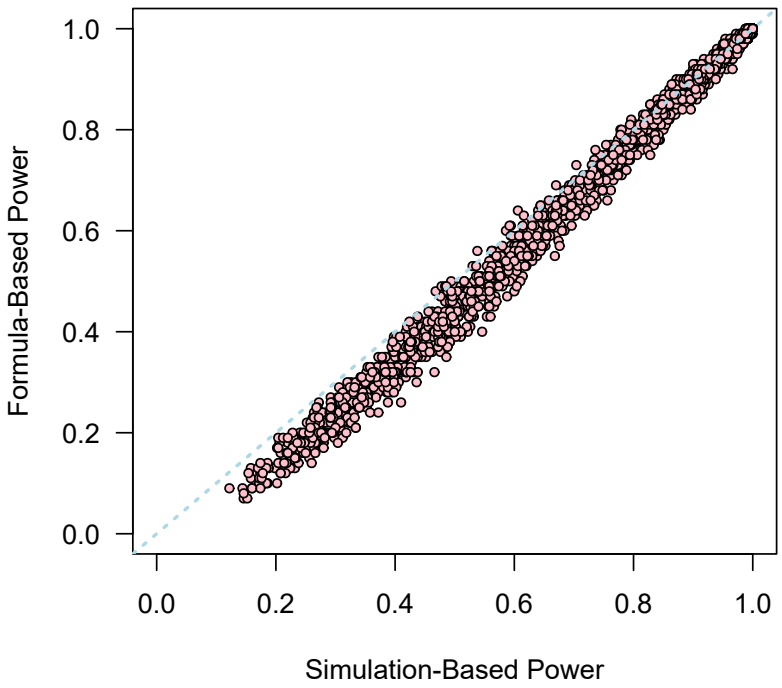

Figure 2.2: Simulation vs formula-based power.

factor, $\omega_{(k)i}$ is the kth replicate within ith whole-plot error, where $k = \{1, \ldots, r\}$, $(\alpha\beta)_{ij}$ is the interaction between whole-plot level i and sub-plot j, finalizing with $\epsilon_{k(ij)}$ which is the split-plot error. The way the data layout looks can be seen in Figure 2.3.

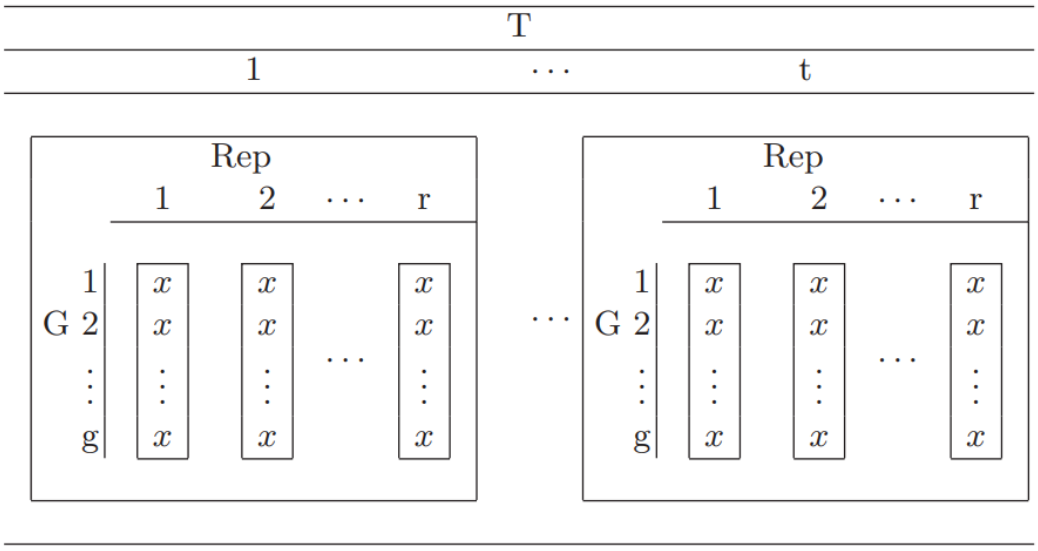

Figure 2.3: Split plot design with CR design on whole plot treatments from Casella (2008). T = whole-plot treatment, G = split-plot treatment, Rep = replication

If the data comes from the split-plot design but with a randomized complete block (RCB) design on the whole-plot level, we have a different model according to Casella (2008)[Chapter 5].

$$Y_{ijk} = \mu + \alpha_i + \theta_k + \omega_{(k)i} + \beta_j + (\alpha\beta)_{ij} + (\theta\beta)_{jk} + \epsilon_{k(ij)}, \tag{17}$$

where now we have an additional blocking effect θ_k of the kth block and the interaction $(\theta\beta)_{jk}$ of block k with the split-plot factor j. The data layout can be also seen in Figure 2.4.

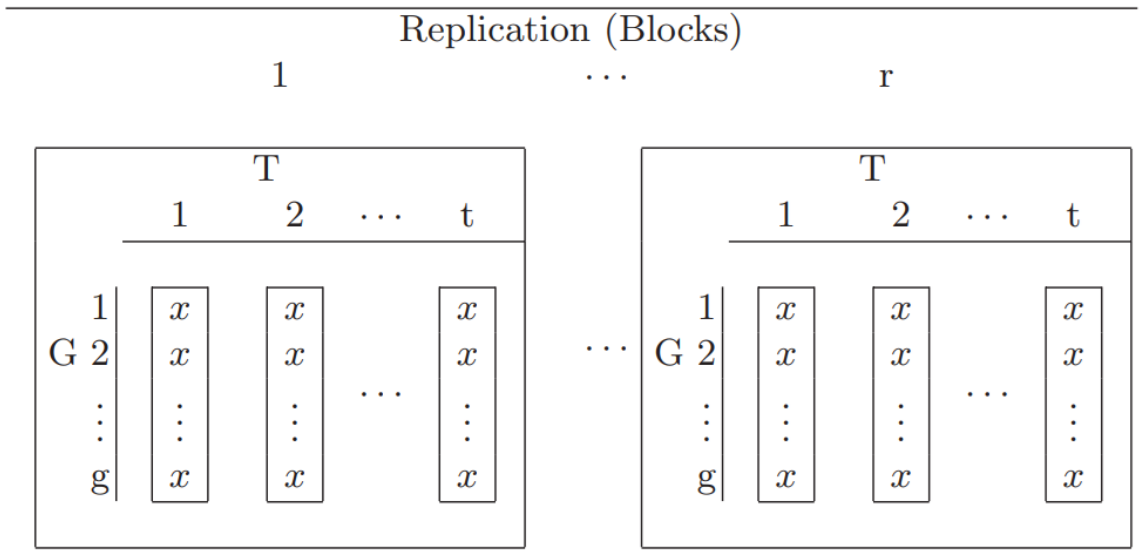

Figure 2.4: Split plot design with RCB design on whole plot treatments from Casella (2008). T = whole-plot treatment, G = split-plot treatment

Knowing that the loss in power comes mostly from the whole-plot factor, the focus of the power analysis and hence the sample size calculation should be lying on the whole-plot factor. According to Lawson (2014), it depends on what kind of design is used on the whole-plot level. If it is a completely randomized design as in equation (16) within the whole plot then the power analysis can be performed using the oneway Anova on the whole-plot level described in Section 2.2.2. In that case, we need to correctly define degrees of freedom (df), hence df_1 would be equal to $a-1$ and df_2 would be equal to $a(r-1)$.

However, if there is a randomized complete block design on the whole-plot level as in the (17) the power analysis described in Section 2.4 should be performed. This leads to $df_1 = a-1$ and $df_2 = (r-1) \times (a-1)$ when power calculation is performed on the whole-plot level.

2.6 CI-Width Based

If we are dealing with an exploratory study the sample size can be computed based on the width of the confidence interval. It allows the researcher to set a certain level of precision for the parameters of interest, such as the effect of some new treatments. All needed values are similar to those presented in Section 2.1 and an additional variable k which represents the number of comparisons that will be made between a treatments in order to adjust α in a proper manner for the family-wise error. The confidence interval (CI), for the case where σ^2 is known, is given by

$$\text{CI} = (\bar{Y}_i - \bar{Y}_j) \pm z_{1-\alpha/2k} \cdot \sqrt{2\sigma^2/n},$$

where $i \neq j$ and both are $i, j = \{1, \ldots, a\}$ and $(z_{1-\alpha/2k})$ is $1 - \alpha/2k$th quantile of the standard normal distribution.

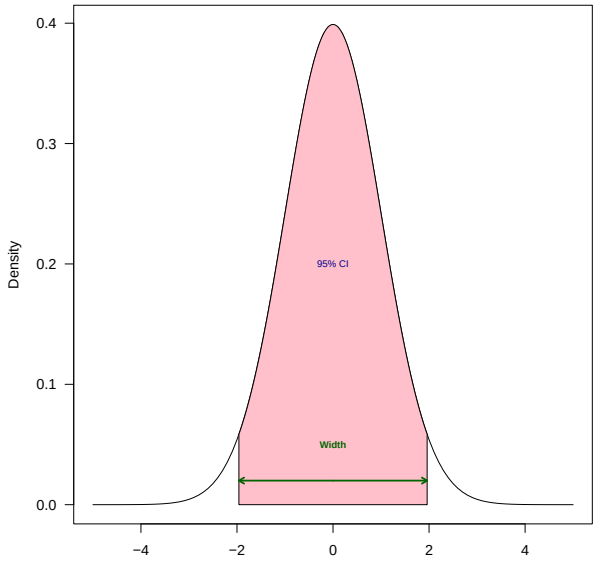

Figure 2.5: Example of width with standartized normal distribution.

So the width w can be defined as

$$w = 2z_{1-\alpha/2k} \cdot \sqrt{2\sigma^2/n}.$$

An example of such width can be observed in Figure 2.5. Solving that for n results in

$$n = \left(\frac{2z_{1-\alpha/2k} \cdot 2\sigma}{w} \right)^2.$$

If we are dealing with a small sample size we can replace $z_{1-\alpha/2k}$ with $t^{-1}_{a(n-1)}(1 - \alpha/2k)$ for a better accuracy and σ^2 with a variance estimator S^2. In that case, the width would be defined as

$$w = 2t^{-1}_{a(n-1)}(1 - \alpha/2k) \cdot \sqrt{2S^2/n}.$$

But what is the exact meaning of the width? When we, for example, run a t test we want our result to be significant meaning that the confidence interval should not contain a 0. This happens if some minimum effect D we want to detect, which can be for example some sample mean $\bar{Y}$ or the difference of two means $\bar{Y}_i$ and $\bar{Y}_j$, should be greater than the half of the width. Therefore this statement should always hold

$$D > w/2.$$

Hence, to have a significant result we need to ensure that D is greater than the half of the width (w) that we can calculate for given parameters and a given n.

Chapter 3

Shiny-App

Shiny is an R package developed by Chang et al. (2022). It allows to build interactive web-based applications and tools straight from R, while combining classical tools for web development and R capabilities.

3.1 Main Changes to SampleSizeR

This section will summarize some of the changes that were made to the previous version of SampleSizeR. We will skip some very minor fixes such as moving some buttons 2 pixels to the left but also the implementation of new methods. The most visible change is the design of the website including its color palette and position of certain tiles. In Figure 3.1 the old design can be observed and in Figure 3.2 the new design. It was decided to remove a wide variety of colors and focus on a grey-blue scale. Of course, this is nothing but a question of taste. This was achieved by adjusting the style setting of each button within an HTML div. Another change that was done on the front page was the adjustment of margins, such that all the buttons are not fixed and attached to a certain point on the screen, but rather have a relative position. This allows the website to be more flexible on different screen sizes and, hence more robust to different devices. Besides that the home button was moved to a slightly different position and the "Impressum" tile is fixed to the bottom of the website. In the "Continuous Outcome" tile another small design change was made by moving tiles to a more suitable position.

On the user input sides, the input for the α was adjusted and now only values between 0 and 1 are allowed and the step of input has been reduced from 1 to 0.01, which makes more sense in the context of α. In general, in order not to repeat the same things multiple times, this change was made for all other categories.

```
sidebarPanel(width = 5,
             numericInput("alpha7", "Choose \u03B1",
                                    min = 0.01,
                                    max = 0.5,
                                    value = 0.05,
                                    step = 0.01))
```

As described in the previous sections more features were implemented, which allow to calculate the sample size in scenarios with paired design or when the equal variance assumption is not holding. For this purpose two inputs were implemented that allow to specify what type of test is needed for the user.

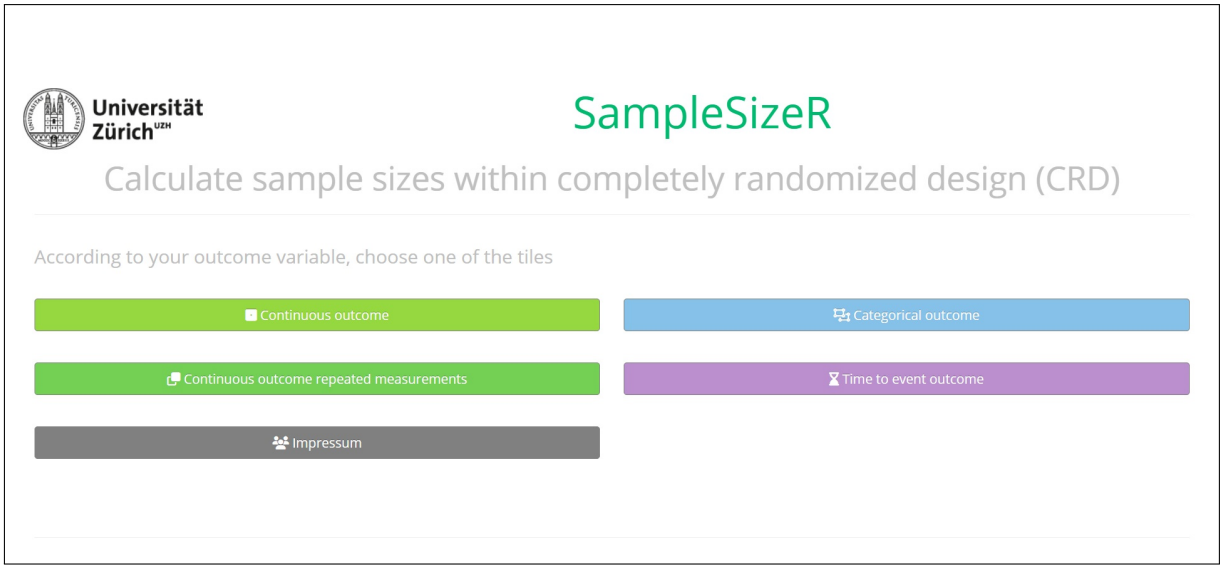

Figure 3.1: Old design of SampleSizeR.

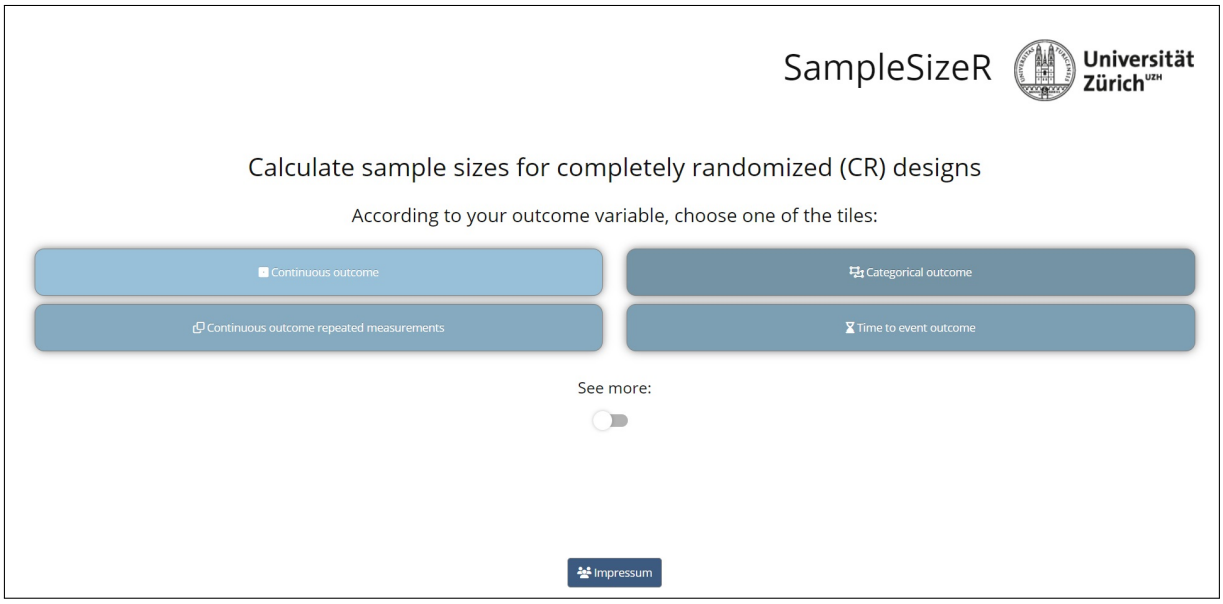

Figure 3.2: New design of SamplesizeR.

```
fluidRow(
  column(6, selectInput("paired", "Paired or unpaired study",
                        choices = c("Paired", "Unpaired"),
                        selected = "Unpaired")),
  column(6, selectInput("eqvar", "Equality of variance",
                        choices = c("Equal variance", "Unequal variance"),
                        selected = "Equal variance")))
```

One issue was discovered with a previously implemented method for the "Repeated measurement". The user was allowed to input any correlation matrix of the user's choice. However, according to Casella (2008), this type of power analysis has an assumption of sphericity for the correlation matrix, which is quite difficult to fulfill. One type of correlation matrix that does not break this assumption is an exchangeable correlation matrix. For that reason the input was restricted to

only this type of matrix and the user can simply adjust the slider to pick the right coefficient.

```
fluidRow(column(11, numericInput("cs", "Choose the coefficient (< |1|):",
                                 min = -1, max = 1, value = 0, step = 0.1 )),
         column(1, bsButton("qcs1", label = "", icon = icon("circle-info"),
                            size = "12px"),
                bsPopover(id = "qcs1", title = "Exchangeable",
                          placement = "right",
                content = "Constant correlation across all responses within the cluster",
                trigger = "click"), style = "padding: 0; margin-top: 25px;"),
                style = "margin: 0"),
tableOutput("matrix_cs"),

# Server commands

observeEvent(input$cs, {
  req(input$measures)
  x <- input$measures
  coef <- input$cs
  output$matrix_cs <- renderTable(toeplitz(c(1, rep(coef, x-1))),
              rownames = FALSE, colnames = FALSE, width = "100%",
              align = "c")
}, priority = 1)

observeEvent(input$measures, {
  req(input$measures)
  x <- input$measures
  coef <- input$cs
  output$matrix_cs <- renderTable(toeplitz(c(1, rep(coef, x-1))),
                            rownames = FALSE, colnames = FALSE, width = "100%",
                            align = "c")
}, priority = 1)
```

Four additional tiles were created that implement a user interface for the sample size calculation based on marginal models, RCB design, and CI-Width-based calculation. They do not differ a lot from the old "Repeated measurement" block, but since they do not have strong assumptions for the correlation structure, they allow a user to choose different types of these structures such as auto-regressive or exchangeable matrix but also an unstructured matrix, where a user can make its own correlation matrix, which is checked in the background whether it is a positive semi-definite matrix.

```
selectInput(
  "CorMatrixType1GEE",
  "Select the correlation matrix type",
  choices = c("AR1", "Exchangeable", "Unstructured")
),
conditionalPanel(
  "input.CorMatrixType1GEE == 'Unstructured'",
  matrixInput(
    "corTime1GEE",
```

```
    label = "Correlation matrix across time levels - only upper triangular part",
    value = matrix(
      c(
        1.00,0.64,0.41,0.26,
        0.64,1.00,0.64,0.41,
        0.41,0.64,1.00,0.64,
        0.26,0.41,0.64,1.00),4,4,byrow = TRUE),
    rows = list(names = FALSE),
    cols = list(names = FALSE)
  )
),
conditionalPanel(
  "input.CorMatrixType1GEE == 'AR1'",
  fluidRow(
    column(
      11,
      numericInput(
        "ar1gee",
        "Choose AR1 coefficient  (< |1|):",
        min = -.99,
        max = .99,
        value = 0,
        step = 0.1
      ),
      style = "padding: 0px; padding-right: 10px"
    ),
    column(
      1,
      bsButton(
        "qar1",
        label = "",
        icon = icon("circle-info"),
        size = "12px"
      ),
      bsPopover(
        id = "qar1",
        title = "AR 1",
        placement = "right",
        content = "Exponentially decaying correlation within the cluster",
        trigger = "click"
      ),
      style = "padding: 0; margin-top: 25px;"
    ),
    style = "margin: 0"
  ),
  tableOutput("matrix_ar1")
),
conditionalPanel(
  "input.CorMatrixType1GEE == 'Exchangeable'",
  fluidRow(
```

```
    column(
      11,
      numericInput(
        "cs1",
        "Choose the coefficient (< |1|):",
        min = -1,
        max = 0.99,
        value = 0,
        step = 0.1
      )
    ),
    column(
      1,
      bsButton(
        "qcs1",
        label = "",
        icon = icon("circle-info"),
        size = "12px"
      ),
      bsPopover(
        id = "qcs1",
        title = "Exchangable",
        placement = "right",
        content = "Constant correlation across all responses within the cluster",
        trigger = "click"
      ),
      style = "padding: 0; margin-top: 25px;"
    ),
    style = "margin: 0"
  ),
  tableOutput("matrix_cs")
)@

# Server commands for AR1 (similar to CS and Unstructured)

observeEvent(input$measures1, {
  req(input$measures1)
  x <- input$measures1
  coef <- input$ar1gee
  neg <- ifelse(input$ar1gee < 0, -1, 1)
  output$matrix_ar1 <- renderTable(toeplitz(c(1, neg * (abs(coef)^(1:(x-1))))),
                                   rownames = FALSE, colnames = FALSE,
                                   width = "100%",
                                   align = "c")
}, priority = 1)

observeEvent(input$ar1gee, {
  req(input$measures1)
  x <- input$measures1
```

```
  coef <- input$ar1gee
  neg <- ifelse(input$ar1gee < 0, -1, 1)
  output$matrix_ar1 <- renderTable(toeplitz(c(1, neg * (abs(coef)^(1:(x-1))))),
                                   rownames = FALSE, colnames = FALSE,
                                   width = "100%",
                                   align = "c")
}, priority = 1)
```

3.2 Password Protection

An additional feature that was implemented is the "advance user" mode. This mode involves more methods that are not supposed to be visible to the public due to the complex structure of the research described in Section 2.4 and Section 2.5. While the mode is switched off, these methods are not visible. However, to turn the mode on one should activate the switch visible in Figure 3.2. The mode is password protected, but there is no authentication system hidden in the back-end and no possibility of account creation, which would involve a lot of additional steps such as the implementation of a secured database, which goes far beyond the goal of needed protection. The way the password protection works is very simple. Once the user switches on the mode, a new field for the password entry will appear. In the background, the typed string is checked with the password defined by the administrator in the back-end. If the password is correct the corresponding message will pop up and hidden sections will become visible.

```
fluidRow(
  column(
    12,
    conditionalPanel('input.num < 1', "See more:"),
    align = "center",
    style = "color: #121212; font-size: 20px"
  )
),
br(),
fluidRow(column(
  width = 12,
  align = "left",
  conditionalPanel('input.num < 1',
                   tags$div(
                     style = "width: 100%;",
                     tags$div(
                       style = "margin-left: 47.5%; width: 120px; height:0px",
                       materialSwitch(
                         inputId = "promode",
                         label = "",
                         value = FALSE,
                         status = "success"
                       )
                     )
                   ))
)),
```

```
br(),
fluidRow(column(
  12,
  conditionalPanel(
    'input.num < 1 && input.promode === true',
    textInput("password", "Password:")
  )
), align = "center"),
fluidRow(column(
  12,
  align = "center",
  conditionalPanel(
    'input.num < 1 && input.promode === true',
    textOutput("password_check")
  )
),
style = "font-size: 10px; text-align: left; color: grey;"),
```

3.3 Report Generation

Another important feature is the possibility to generate a complete report with all user inputs and the application's output. To make things easier for the end-user a "Copy to clipboard" button by using the package by Bhalla (2024) was added which allows to copy the report with a click. The plugin that is used for that purpose does not support the desired implementation, so it was slightly adjusted to the desired needs. The report is a stitched-together `paste()` function with all the inputs. Because the copy button was not designed for this specific type of interaction, it is required to press the button twice. On the first click it updates the report on the back-end and with the second click it copies it.

```
# UI

fluidRow(textOutput("title1"),
         style = "font-size: 20px; text-align: left;
         color: black; font-weight: bold"),
fluidRow(textOutput("report1"), style = "font-size: 16px; text-align:
         left; color: black"),
hr(),
fluidRow(CopyButton(
  "copybtn",
  label = HTML("Copy to clipboard <br> Press twice!"),
  text = ""
)),
)

# Back-end

# For the copy to clipboard button

output$report1 <-
  renderText({
    if (input$vardiff == "No" & input$paired == "Paired") {
```

```
      paste0(
        "This experiment is a completely randomized balanced design with",
        "1 treatment factor with 2 groups.
        The type I error is ",
        input$alpha,
        ", the type II error is ",
        1 - input$beta1,
        ", and we use two-sided test.
        We need sample size of ",
        power()$nreps[power()$power > input$beta1][1],
        " pairs to detect an effect size of ",
        input$delta,
        " with estimated standard deviation of ",
        round(
          sqrt(
            input$sigma1 ^ 2 + input$sigma2 ^ 2 - 2 * input$sigma1 *
              input$sigma2 * input$cor1
          ),
          2
        ),
        ". The output was generated using power.t.test() embedded in SampleSizeR",
        "available at https://shiny.math.uzh.ch/git/reinhard.furrer/SampleSizeR/."
      )
    } else if (input$vardiff == "Yes" & input$paired == "Paired") {
      paste0(
        "This experiment is a completely randomized balanced design with",
        "1 treatment factor with 2 groups.
        The type I error is ",
        input$alpha,
        ", the type II error is ",
        1 - input$beta1,
        ", and we use two-sided test.
        We need sample size of ",
        power()$nreps[power()$power > input$beta1][1],
        " pairs to detect an effect size of ",
        input$delta,
        " with estimated standard deviation of ",
        input$sigmaDiff,
        ". The output was generated using power.t.test() embedded in SampleSizeR",
        "available at https://shiny.math.uzh.ch/git/reinhard.furrer/SampleSizeR/."
      )

    } else if (input$eqvar == "Unequal variance" &
               input$paired == "Unpaired") {
      paste0(
        "This experiment is a completely randomized balanced design with 1",
        "treatment factor with 2 groups.
        The type I error is ",
        input$alpha,
        ", the type II error is ",
        1 - input$beta1,
        ", and we use two-sided test.
        We need sample size of ",
        power()$nreps[power()$power > input$beta1][1],
        " in each group to detect an effect size of ",
        input$delta,
```

```
        " between the two means
        in the outcome, with estimated standard deviation of ",
        input$sigma1,
        " in the first group and ",
        input$sigma2,
        " in the second group.",
        " The output was generated using MKpower::power.welch.t.test() embedded,"
        "in SampleSizeR available at",
        "https://shiny.math.uzh.ch/git/reinhard.furrer/SampleSizeR/."
      )

    } else if (input$eqvar == "Equal variance" &
               input$paired == "Unpaired") {
      paste0(
        "This experiment is a completely randomized balanced design with 1", "
        treatment factor with 2 groups.
        The type I error is ",
        input$alpha,
        ", the type II error is ",
        1 - input$beta1,
        ", and we use two-sided test.
        We need sample size of ",
        power()$nreps[power()$power > input$beta1][1],
        " in each group to detect an effect size of ",
        input$delta,
        " between the two means
        in the outcome, with estimated standard deviation of ",
        input$sigma,
        ". The output was generated using power.t.test() embedded in SampleSizeR",
        "available at https://shiny.math.uzh.ch/git/reinhard.furrer/SampleSizeR/."
      )

    }


  })
```

Chapter 4

Conclusion

In this Master's thesis, we summarize and elucidate the most common methods for sample size calculation in a complete randomized design, randomized complete block design, split-plot design and repeated measurements in time using power-based methods, but also precision-based methods in case of an exploratory study. Beyond the classical and simplest approaches such as the t test and Anova, we delve into more complex designs like split-plot design or repeated measurements. Of particular interest is the implementation of the method by Liu and Liang (1997), showcasing its application for the complete block design with additional features not originally discussed in the original paper. In addition to power analysis, we explore methods based on precision, such as confidence-interval-width sample size calculation.

Furthermore, in addition to the theoretical exploration, we have extensively reworked the existing version of SampleSizeR, an online shiny app maintained by the Applied Statistics group at the Department of Mathematical Modeling and Machine Learning. We have not only implemented new methods but also reworked existing ones, enhancing their functionality. For instance, we have revamped the implementation of a two-way Anova without an interaction term. Recognizing the complexity of certain methods, we have chosen to discreetly hide them under the password-protected "More" mode, ensuring accessibility for advanced users while maintaining simplicity for others. Another noteworthy addition is the dynamic report generation feature, allowing users to effortlessly generate detailed reports tailored to their inputs. Alongside these significant updates, we have fine-tuned the design to enhance application functionality across different devices and rectified minor bugs and user-unfriendly elements, ensuring an optimal user experience.

Appendix A

Appendix

```
sessionInfo()
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 22.04.1 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.10.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.10.0
##
## locale:
##  [1] LC_CTYPE=C.UTF-8       LC_NUMERIC=C           LC_TIME=C.UTF-8
##  [4] LC_COLLATE=C.UTF-8     LC_MONETARY=C.UTF-8    LC_MESSAGES=C.UTF-8
##  [7] LC_PAPER=C.UTF-8       LC_NAME=C              LC_ADDRESS=C
## [10] LC_TELEPHONE=C         LC_MEASUREMENT=C.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] geepack_1.3.9 mvtnorm_1.2-4 dplyr_1.1.4   geeM_0.10.1   Matrix_1.6-5
## [6] knitr_1.45
##
## loaded via a namespace (and not attached):
##  [1] magrittr_2.0.3   MASS_7.3-55      tidyselect_1.2.0 lattice_0.20-45
##  [5] R6_2.5.1         rlang_1.1.2      fansi_1.0.6      highr_0.10
##  [9] tools_4.1.2      grid_4.1.2       broom_1.0.5      xfun_0.41
## [13] utf8_1.2.4       cli_3.6.1        digest_0.6.33    tibble_3.2.1
## [17] lifecycle_1.0.4  tidyr_1.3.0      purrr_1.0.2      vctrs_0.6.5
## [21] glue_1.6.2       evaluate_0.23    compiler_4.1.2   pillar_1.9.0
## [25] backports_1.4.1  generics_0.1.3   pkgconfig_2.0.3
```

A.1 Positive Definite Matrix

The matrix in the equation (15) is positive definite if it fulfills the following criteria:

$$\phi \geq \begin{cases} \frac{1}{m-1}(\rho - 1), & \text{if } \rho \geq 0 \text{ and } \rho + \phi < 1, \\ -\frac{1}{m-1}(\rho + 1) - 2\rho, & \text{if } \rho \in (-\frac{m}{2m-1}, 0) \text{ and } \rho + \phi > -1. \end{cases}$$

A.2 Plots

The following code allows to recreate the Figure 2.1.

```
tobs = 2
curve(
  dt(x,  df = 6, ncp = 0),
  from = -4,
  to = 6,
  col = "blue",
  ylab = "Density",
  las = 1,
  xlab = "",
  xlim = c(-3, 6)
)
curve(
  dt(x,  df = 6, ncp = 3),
  from = -4,
  to = 10,
  add = TRUE,
  col = "red"
)
abline(v = tobs)

polygon(
  x = c(tobs, seq(tobs, 10, 0.01), 10.01),
  y = c(0, dt(
    seq(tobs, 10, 0.01), df = 6, ncp = 3
  ), 0),
  col = "lightblue"
)
polygon(
  x = c(tobs, seq(tobs, 6, 0.01), 6.01),
  y = c(0, dt(
    seq(tobs, 6, 0.01), df = 6, ncp = 0
  ), 0),
  col = "pink"
)
polygon(
  x = c(-4, seq(-4,-tobs, 0.01),-tobs + 0.01),
  y = c(0, dt(
    seq(-4,-tobs, 0.01), df = 6, ncp = 0
  ), 0),
  col = "pink"
```

```
)
text(x = 3.15,  y = 0.12, "Power")
text(x = -0.5,  y = 0.07, "Type-I Error")
arrows(
  x0 = -0.5,
  y0 = 0.04,
  x1 = -tobs + (-0.1),
  y1 = 0.04,
  length = 0.08,
  angle = 20
)
```

The following code allows to recreate the Figure 2.2.

```
plot(results$res_sim, results$res_formula, pch = 21, col = "black",
     bg = "pink", cex = 0.7, las = 1,
     xlab = "Simulation-Based Power",
     ylab = "Formula-Based Power", xlim = c(0,1), ylim = c(0,1), main = "")
abline(a = 0, b = 1, lwd = 2, lty = 3, col = "lightblue")
```

The following code allows to recreate the Figure 2.5.

```
par(mfrow = c(1, 1),
    mar = c(2.5, 4, 2, 1),
    pty = "s")
curve(
  dnorm,
  from = -5,
  to = 5,
  ylab = "Density",
  xlab = "",
  las = 1
)
polygon(
  x = c(-1.96, seq(-1.96, 1.96, 0.01), 1.96),
  y = c(0, dnorm(seq(-1.96, 1.96, 0.01)), 0),
  col = "pink"
)
text(
  x = 0,
  y = 0.05,
  substitute(paste(bold('Width'))),
  cex = 0.7,
  col = "darkgreen"
)
text(
  x = 0,
  y = 0.2,
  "95% CI",
  cex = 0.7,
```

```
  col = "darkblue"
)

arrows(
  0,
  0.02,
  -1.96,
  0.02,
  length = .07,
  col = "darkgreen",
  lwd = 2
)
arrows(
  0,
  0.02,
  1.96,
  0.02,
  length = .07,
  col = "darkgreen",
  lwd = 2
)
```

Bibliography

Arnold, B. F., Hogan, D. R., Colford, J. M., and Hubbard, A. E. (2011). Simulation methods to estimate design power: an overview for applied research. *BMC Medical Research Methodology*, 11. 16

Bhalla, D. (2024). *shinyCopy2clipboard: Copying text to the clipboard*. R package version 0.1.0, https://github.com/deepanshu88/shinyCopy2clipboard/. 29

Casella, G. (2008). *Statistical Design*. Springer New York. 15, 19, 20, 21, 24

Chang, W., Cheng, J., Allaire, J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., and Borges, B. (2022). *shiny: Web Application Framework for R*. R package version 1.7.4, https://CRAN.R-project.org/package=shiny. 1, 23

Dean, A., Voss, D., and Draguljić, D. (2017). *Design and Analysis of Experiments*. Springer International Publishing. 6, 7, 8

Diggle, P. (2002). *Analysis of Longitudinal Data*. Oxford Statistical Science Series. OUP Oxford. 9, 11

Dixon, P. (2016). Should blocks be fixed or random? New Prairie Press. 15

Hu, N., Howard, M., and Thomas, R. (2021). Power and sample size for random coefficient regression models in randomized experiments with monotone missing data. *Biometrical Journal*, 63 806–824. 16

Kohl, M. (2023). *MKpower: Power Analysis and Sample Size Calculation*. R package version 0.7, https://github.com/stamats/MKpower. 5

Lawson, J. (2014). *Design and Analysis of Experiments with R*. Chapman and Hall/CRC. 15, 19, 21

Lawson, J. and Krennrich, G. (2023). *daewr: Design and Analysis of Experiments with R*. R package version 1.2-11, https://CRAN.R-project.org/package=daewr. 7

Liang, K.-Y. and Zeger, S. L. (1986). Longitudinal data analysis using generalized linear models. *Biometrika*, 73 13–22. 10

Liu, G. and Liang, K.-Y. (1997). Sample size calculations for studies with correlated observations. *Biometrics*, 53 937. 12, 16, 33

Molenberghs, G. and Verbeke, G. (2005). *Models for Discrete Longitudinal Data*. Springer Series in Statistics. Springer New York. 10, 11

Muff, S., Held, L., and Keller, L. F. (2016). Marginal or conditional regression models for correlated non-normal data? *Methods in Ecology and Evolution*, 7 1514–1524. 11

Yan, J. (2002). geepack: Yet another package for generalized estimating equations. *R-News*, 2/3 12–14. 16