Harmonization of binomially distributed variables in individual participant data meta-analysis

Hilgers, C.M.

Award date:
2017

Harmonization of Binomially Distributed Variables in Individual Participant Data Meta-Analysis

Technische Universiteit Eindhoven

Catherine Hilgers

31 August, 2017

Contents

Chapter 1

Introduction

1.1 Context

Statisticians and epidemiologists often wish to combine multiple data sets or studies in order to extract information about a larger data set. This desire occurs because larger data sets have greater power to detect statistical effects and create models based on a sample that better represents the population the statistician wishes to model. (Siddique, Reiter, et al. 2015) The process of this combination is known as meta-analysis (MA).

Meta-analysis is divided into three distinct processes: aggregate data meta-analysis (AD-MA), which rely on results from published studies; and individual participant data meta-analysis (IPD-MA), which pools participant-level data, also called individual participant data or IPD, from the studies. (Siddique, Reiter, et al. 2015) A third approach is offered by a federated framework, where data is not pooled in a single location. Instead, analyses are sent to the data-hosting server from a central location, and summary statistics are returned to the central location for combination and further analysis. (Gaye et al. 2014)

There are several factors that can help researchers choose between these methods. Because AD-MA is reliant on published, study-level results, researchers performing AD-MA cannot adjust for covariates, perform participant subgroup analyses, apply different statistical models than those implemented by the original study researchers, or analyze different outcomes than the original study already analyzed. In particular, in order to deal with missing data using gold-standard approaches outlined in Graham (2009) and Siddique, Reiter, et al. (2015), IPD is required. IPD-MA largely surmounts the issues encountered in AD-MA, but suffers from being much more time-consuming and expensive, due to the difficulty of obtaining participant data from studies, ensuring ethical concerns about privacy and reuse of data are met, and carefully verifying compatibility of the included data sets. (L. E. Griffith, Van Den Heuvel, et al. 2015; Fortier et al. 2016) Despite these difficulties, IPD-MA is considered the best practice in meta-analysis for applications where the above issues with AD-MA obstruct analysis. (Thomas, Radji, and Benedetti 2014) It has achieved this high regard because it allows consistent outcome definitions, and consistent analyses between studies. (Resche-Rigon and White 2016) A federated framework can allow the researchers some more flexibility with data usage than IPD-MAs. A federated IPD-MA allows a researcher to form a research question, then query a centralized server. This server communicates the query to servers of individual studies on which the participant data is housed. These servers return de-identified summary statistics to the central server, and these summaries are then combined using standard AD-MA

methods and communicated to the researcher. Because data is hosted on the individual study servers, ethical concerns about participant privacy are diminished or eliminated, and issues with AD-MA that stem from only being able to work with published results cease to exist because the researcher can form their own research questions. However, all non-privacy concerns that face researchers performing a non-federated (or pooled) IPD-MA still must be addressed.

In the particular case when a researcher wishes to combine measures from different studies that have been collected differently, the researcher performs what is known as a harmonization. (L. Griffith et al. 2013) The harmonization of measures collected differently can only be performed in an IPD-MA or federated framework. Many statistical methods that are available to researchers performing IPD-MA are at least theoretically applicable to a federated setting, and so we can continue with a focus on IPD-MA with the understanding that this may be of interest in a federated setup.

In a review of IPD-MAs that validated risk prediction models, Ahmed et al. (2014) identified two chief methodological concerns for initiatives performing an IPD-MA. These are heterogeneity between studies and missing data. Heterogeneity between studies comes from many sources, not limited to: sampling frame and target population, study quality, disparate time frames, different measurement scales (ie. the Children's Depression Rating Scale (CDRS) or the Hamilton Depression Rating Scale (HDRS), both measuring child depression), instruments (ie. different instruments used to measure blood pressure or weight), or categorizations (ie. 'How many drinks do you have per day?' vs. 'How many drinks do you have per week?'). (L. Griffith et al. 2013) One potential solution is to perform prospective IPD-MAs, where researchers agree on a study protocol before data collection begins. (Ahmed et al. 2014) However, this results in an undesirable loss of previously collected data, particularly because participant data is often costly and time-consuming to collect.

In fact, the missing data and between-study heterogeneity concerns identified by Ahmed et al. (2014) are related. Data that is missing because participants did not answer the question is said to be *sporadically missing*. Nearly all cohort studies and IPD-MAs deal with this type of missing data, and methods for dealing with sporadic missingness are well-documented. (Graham 2009) However, between-study heterogeneity that occurs because certain variables were not collected (or not in the same way) by some studies can be treated as a missing data problem. In the IPD-MA literature, studies that did not collect a variable of interest are said to have *systematically missing* data. In this case, the researcher performing the IPD-MA is concerned with a certain construct such as blood pressure, depression, or cognitive ability that is collected with different instruments. The researcher has two options. They can choose a single measurement method and discard the studies for which the method is systematically missing, or they can treat the systematically missing variable as a missing data problem, and attempt to harmonize the differently collected measures using missing data methods. These issues equally face researchers wishing to operate in a federated framework, so we will specifically address statistical methods for dealing with both systematic and sporadic missingness in IPD-MAs of cohort data sets.

1.2 Problem Statement

There are many motivating factors for treating missing data in IPD-MAs. Chief among these is that most commonly used statistical models require complete data sets, that

is, data sets with no missing data at all. The generation of complete data sets from data sets with missing data can be performed using missing data techniques. This also allows the complete data sets to be used by researchers with less statistical experience, for their own analyses. There has to date not been enough research on how to perform the harmonization of measures collected on different scales or with different instruments in the epidemiological context. Adequate harmonization techniques would allow researchers to make use of participant data that was already collected at a great cost or over a long period of time, instead of discarding data sets that did not collect all the outcome and covariate variables desired for the meta-analysis. There has been an increase in the interest of the broader epidemiological research community in the combination of measures collected differently. (L. E. Griffith, Van Den Heuvel, et al. 2015) To date, however, the specific case where the researcher wishes to harmonize binomially distributed variables with the same underlying construct (such as memory) has not been studied. A focus on this context is needed.

1.3 Aims and Scope

The aim of my research is to explore methods for harmonization of binomially distributed variables in the epidemiological context, in particular for the combination of two or more IPD studies, of disjoint individuals, where some variables appear in two or more data sets. This will be accomplished by a case study of a simulated data set, applying different methods and comparing their ability to harmonize different measures.

The scope is limited by not investigating the case of harmonization of continuously distributed variables, which has been fairly well-covered in the literature. (Audigier, White, et al. 2017; Jolani et al. 2015; Resche-Rigon and White 2016; M Quartagno and JR Carpenter 2015) Only participant data from epidemiological questionnaires will be used, as opposed to data from randomized controlled trials, because the process of randomization used by the latter changes (and often facilitates) the statistical analysis, especially in treatment of bias. Randomized controlled trials do not have the same difficulty in estimating their model parameters, because randomization largely obviates the need for collection of and adjustment for covariates. The federated setting, while interesting, will also not be explored in depth. Methods that are applicable in an IPD-MA can often be adapted for a federated setting.

1.4 Significance of the Study

The field of epidemiology could be greatly advanced if combination of different scales and measures through data harmonization was widely applied. There are so many different ways of measuring people's health, habits, and lifestyle characteristics, and standardization of these methods is desirable but not eminently practical, since each study has different needs and research questions to address.

Many researchers wishing to perform an IPD-MA wish to include measures that have been collected differently by different studies, so this research has great potential for practical applications. (Siddique, Chavez, et al. 2016) The harmonization of binomially distributed outcomes is of great interest to researchers and requires special attention. Memory scores are often collected with binomially distributed variables, for example the Mini-Mental State Examination, the Abbreviated Mental Test Score, the Rey Auditory

Verbal Learning Test, the Alzheimer's Disease Assessment Scale - Cognitive Subscale, amongst others. Any test composed of binary items is of interest here.

Calls for such research have been made in the literature. Jolani et al. (2015) writes that "a method that will allow pooling of IPD from studies that use different measures for the same construct could greatly increase power for treatment effect analysis," and the authors outline methods for continuous and binary distributed variables but stop short of the binomial setting.

The intended outcome of the research is to address these calls by other researchers by demonstrating how to perform such a harmonization of differently-collected binomially distributed variables in a federated setup.

1.5 Thesis Overview

An overview of the important definitions and general methods for missing data and harmonization is presented, followed by a literature review of applicable methods. Next the research problem will be explicitly given, and the methods for addressing the research questions will be given. The methods comprise the generation of synthetic data sets, with and without missingness in differently collected binomially distributed outcomes. The harmonization of these differently collected outcomes will follow, with an appropriate IPD meta-analysis model applied to the data sets. Lastly, a discussion of the relative merits of each of the methods, with recommendations for future research will be given.

Chapter 2

Background

The background overview that follows will give the reader the context and definitions required to understand the field I am working in and the remainder of the thesis.

The aim of this chapter is to place my research in the context of the field and allow the reader to follow the subsequent sections of the thesis. It also aims to demonstrate my knowledge of the field and make clear the case for my own research.

An overview of the process of meta-analysis will be given, with a motivation for the treatment of uncollected variables in clinical studies as a missing data problem. Current and historical methods for the treatment of missing data will be detailed, in the context of IPD-MAs of clinical studies or cohort data. The specific case of IPD-MAs that wish to harmonize studies that collected different memory scales will be introduced.

2.1 Search Strategy

On 22 March 2017, a search was performed on Google Scholar for "meta analysis + missing data". Titles were scanned for papers of potential interest. References of relevant papers were scanned, and papers citing these relevant papers were found through a search on Google Scholar. These were added to the literature review as the research went on. Papers were also collected from my supervisor, and a search was performed again in Google Scholar for papers citing these relevant works. Reading and writing of this chapter continued throughout the thesis-writing process.

2.2 Meta-Analysis

Many studies are collecting data on human health and risk factors for disease. Large sample sizes are desirable for accuracy and statistical power, especially when studying rare diseases or complex interactions between risk factors in multifactorial diseases. (Magalhaes and C Wolfson 2012) The precise data required for these studies are expensive and time-consuming to collect. Desire for large sample sizes and comparison of data across studies are two main reasons for the popularity of the integration of data sets, also called meta-analysis.

Meta-analysis has been in use for over 30 years for the synthesis of multiple published studies. It is well-loved because it can substantially increase power to detect effects. (Siddique, Reiter, et al. 2015) The general process of meta-analysis is detailed below.

Performing a meta-analysis

Any meta-analysis begins by formulating the research question. Usually the researchers wish to investigate the effect of an exposure X on an outcome Y. (Matteo Quartagno 2016) In the simplest case with continuous X and Y, a linear regression can be performed with the following model:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i \tag{2.1}$$

i is the index of individual
ϵ_i is an error term $\sim N(0, \sigma_e^2)$
β_0 is the intercept of the regression line
β_1 is the slope of the regression line

The parameters of interest are usually β_0, β_1, the so-called effects or regression coefficients, and/or σ_e^2, the variance of the error. These are estimated respectively by $\hat{\beta}_0$, $\hat{\beta}_1$, and $\hat{\sigma}_e^2$.

The researcher will then perform a literature search, identifying studies that investigated the effect of X. Often this is done in a combination of ways, with web searches, reference searches, and consultation of experts in the field. (Matteo Quartagno 2016) The researchers make a list of selection criteria, and the studies found in the literature search are checked against these criteria. Studies which meet the criteria are included in the meta-analysis.

At this point, the researcher has two options. They can aggregate published, study-level data. For example in Model 2.1, supposing the researcher was interested in slope β_1, this would mean collecting estimates of effects $\hat{\beta}_{1,s}$ and their standard errors $\hat{\sigma}_{1,s}$, from each of studies $s \in \{1, \ldots, S\}$ included in the meta-analysis. This is called aggregate data meta-analysis (AD-MA). (Matteo Quartagno 2016) The other option available to the researcher is to pool individual-level data from the included studies with an individual participant data meta-analysis (IPD-MA). If each study s has N_s participants, this means that each $y_{i,s}$ and $x_{i,s}$ is collected for $i \in \{1, \ldots, N_s\}$, for each study $s \in \{1, \ldots, S\}$. This is known as individual participant data meta-analysis (IPD-MA). In the case of IPD-MA, the data is pooled and then one of a few methods are used to generate estimates of regression coefficients or errors. These methods are described in more detail later on. The correct choice between IPD-MA and AD-MA is not always clear. A comparison of their merits is required.

Comparison of IPD-MA and AD-MA

IPD-MA is distinguished from AD-MA because the former pools participant-level data, whereas AD-MA only makes use of published results. IPD-MA allows for model adjustment for covariates beyond the published results, and it allows the detection of different treatment effects across individuals. (L. E. Griffith, Van Den Heuvel, et al. 2015) It is important to note that the inferential equivalence of different measures or scales for the same hypothesized underlying construct (such as memory) can only be assessed using IPD-MA. (L. Griffith et al. 2013)

AD-MA is performed by combining published results, usually by placing the results on a common metric. Most often this is performed using a standardized mean difference or a log odds ratio. (Siddique, Reiter, et al. 2015) Two major deficiencies with AD-MA are (i) the differences in the ways the studies are collected (between-study heterogeneity) cannot be analyzed with AD-MA, and (ii) the inability to study different outcome measures than

those published by the study. These obstacles can thwart the meta-analysis altogether. Performing an AD-MA also restricts research only to topics for which data have been published. IPD-MA can overcome many of the issues associated with AD-MA. In an IPD-MA, raw, participant-level data is obtained from studies included in the meta-analysis. After obtaining the raw data, researchers can assess between-study heterogeneity, investigate research questions for which published results are lacking, investigate the mediating effect of different groups of covariates, and, importantly, address missing data.

The choice between IPD-MA or AD-MA is also influenced by data structure and the type of analysis the researcher wishes to apply. These are investigated next.

Data structure and analysis

In any meta-analysis, participants are clustered within their studies. This induces a two-level hierarchy at minimum. Higher level settings exist, such as when participants are clustered within hospitals, and hospitals are clustered within study, but we will focus on the two-level case. Multilevel data analysis must take into account correlation within clusters, using what is known as a multilevel model. These are synonymously known as hierarchical or mixed models. (Matteo Quartagno 2016) A multilevel analysis can be performed with a one-stage or two-stage analysis.

A **one-stage analysis** models all IPD from studies in the meta-analysis in one step. A one-stage analysis can only take place if the researcher has access to IPD. Clustering, or correlation between participants in a study, must be taken into account with a random effect in the model for clusters.

Model 2.1 can be translated to a hierarchical model with random (study-specific) intercept as follows:

$$y_{i,s} = (\beta_0 + u_{0,s}) + \beta_1 x_{i,s} + \epsilon_{i,s} \tag{2.2}$$

where s indicates study membership, and the term $u_{0,s} \sim N(0, \sigma_u^2)$ is a study-specific random intercept. Thus, the intercept varies between studies. (Matteo Quartagno 2016)

If the researcher wishes to consider not only intercept varying between studies (study-specific, random intercept), but also the slope varying between studies (study-specific, random slope), a hierarchical model with random slope and intercept is given by:

$$y_{i,s} = (\beta_0 + u_{0,s}) + (\beta_1 + u_{1,s}) x_{i,s} + \epsilon_{i,s} \tag{2.3}$$

Here, $u_{0,s}$ and $u_{1,s}$ are often assumed to follow a bivariate normal distribution:

$$\begin{pmatrix} u_{0,s} \\ u_{1,s} \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \mathbf{\Sigma} \right), \mathbf{\Sigma} = \begin{pmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{pmatrix}, \begin{pmatrix} \sigma_{11} & 0 \\ 0 & \sigma_{22} \end{pmatrix} \tag{2.4}$$

with a covariance matrix $\mathbf{\Sigma}$ chosen unstructured or with correlations between $u_{0,s}$ and $u_{1,s}$ set to 0, on left and right respectively in (2.4). $\sigma_{12} = \sigma_{21}$ is the covariance between $u_{0,s}$ and $u_{1,s}$, and $\sigma_{11} = \sigma_1^2, \sigma_{22} = \sigma_2^2$ are the variances of each. To allow error terms to vary between studies, the assumption $\epsilon_{i,s} \sim N(0, \sigma_{e,s}^2)$ can be made.

Estimation of these one-stage multilevel models can be performed with maximum likelihood, or restricted maximum likelihood (REML), which is more likely to produce unbiased estimates of variance components. (Matteo Quartagno 2016) These techniques essentially maximize the regression coefficients, such that the observed data is most likely to have occurred.

Two-stage analyses

Two-stage analyses can be performed on AD or IPD. If the researcher has IPD, a two-stage analysis is performed by applying the model the researcher wishes to investigate, such as Model 2.1, to each study separately. The estimated parameters are collected from each study, and then they are combined using a set of rules, most often Rubin's rules, which essentially average the parameters, with slightly more complex calculations for variances. (Rubin 1986) If the researcher does not have IPD, a two-stage analysis for AD-MA is performed by collecting the estimated parameters from published data of studies included in the meta-analysis, and then combining the parameters using Rubin's rules, as usual. (Matteo Quartagno 2016)

The combination is often performed by weighting the results from individual studies according to the size of the standard error: the Inverse Variance Weighting (IVW) approach. (DerSimonian and Laird 1986; Matteo Quartagno 2016) With IVW, parameter estimates from individual studies are weighted with the inverse of their variance. IVW is performed in one of two ways: a *fixed effects analysis*, or a *random effects analysis*. Fixed effects analysis assumes a common covariate effect across different studies, while random effects analysis assumes that the effect from each covariate, for each study, is a random draw from a distribution.

The fixed effects IVW model combines study-specific estimates $\hat{\beta}_{1,s}$ and $\hat{\sigma}_{1,s}$ as such:

$$\hat{\beta}_{1,s} = \beta_1 + \epsilon_s, \quad \epsilon_s \sim N(0, \hat{\sigma}^2_{1,s}) \tag{2.5}$$

The maximum likelihood estimate is $\hat{\beta}_{1,fixed}$, given by

$$\hat{\beta}_{1,fixed} = \frac{\sum_{s=1}^{S} \hat{\beta}_{1,s}/\hat{\sigma}^2_{1,s}}{\sum_{s=1}^{S} 1/\hat{\sigma}^2_{1,s}} \tag{2.6}$$

The weight given to the estimate is $1/\hat{\sigma}^2_{1,s}$. The estimated variance of $\hat{\beta}_{1,fixed}$ is

$$\hat{Var}(\hat{\beta}_{1,fixed}) = \frac{1}{\sum_{s=1}^{S} 1/\hat{\sigma}^2_{1,s}} \tag{2.7}$$

In contrast, the random effects model is given by

$$\hat{\beta}_{1,s} = \beta_1 + u_s + \epsilon_s, \quad \epsilon_s \sim N(0, \sigma^2_{1,s}), \quad u_s \sim N(0, \tau^2) \tag{2.8}$$

Here, u_s and ϵ_s are independent, and the u_s is considered a random draw from a distribution, so if a study was re-run, a different value u_s could be drawn for the same study. This principle is called exchangeability. (Matteo Quartagno 2016) τ^2 is the between-study variance component. In the fixed effects model in 2.6 and 2.7, the estimates have weights $1/\sigma^2_{1,s}$, whereas in the random effects model the weights are $1/(\sigma^2_{1,s} + \hat{\tau}^2)$

$\hat{\tau}^2$ is an estimate of the between-study variance component τ^2. It can be found in many ways, but the most common is the Der Simonian and Laird method (DerSimonian and Laird 1986), but this method has been criticized for underestimating standard errors of its estimates. (Matteo Quartagno 2016) A modified alternative has been proposed by Harting and Knapp, which offers a different estimate of the variance and estimates confidence intervals with a t-distribution insead of a standard normal distribution. (Hartung and Knapp 2001)

If the researcher has IPD, a choice can be made between fixed- or random-effects models. This choice can be guided by certain metrics such as the Q or I^2 statistics, and by knowledge of the data. If the researcher only has AD, then the type of model is restricted by availability.

Comparison of analysis methods

With an AD-MA, only the two-stage MA is possible, since the IPD are not pooled. With an IPD-MA, there is a choice available between one- and two-stage analyses.

Much has been written about which choice is more appropriate. Until recently, the two methods were thought to give very similar results, but work from Debray et al. (2013) highlighted a case where conclusions of contradictory clinical significance were reached, using a one-stage and two-stage analysis on the same data. Burke, Ensor, and Riley (2017) compared the two methods, in the IPD case. The authors write that one-stage analysis is recommended because it uses a "more exact likelihood specification, avoiding the assumptions of within-study normality and known within-study variances, which are especially problematic in meta-analyses with small studies and/or rare events. Yet, one-stage methods are also criticised for being computationally intensive and prone to convergence problems." Two-stage analysis is "often preferred because it uses standard MA methods that are well-documented, for example, in the Cochrane Handbook," a highly-cited reference for performance of meta-analyses. (Higgins and Green 2011) Furthermore, Tierney et al. (2015) recommend reporting results from both one-stage and two-stage analyses.

In their comparison of one-stage and two-stage methods for IPD-MA, Burke, Ensor, and Riley (2017) listed a few common reasons for different results from the two methods: the analyst makes discrepant modeling assumptions, the specification of unknown parameters is changed, or different techniques are used for model estimation or confidence interval derivation. They also note that the inverse variance weighting approach of two-stage analyses is not appropriate when dealing with time-to-event or binary outcomes that are rare or very common, or where studies are expected to be small. In these cases a one-stage approach is recommended.

Obstacles to meta-analysis

The proliferation of different measures or scales for the same hypothesized underlying construct is a major issue for meta-analyses in epidemiology and other fields. (Siddique, Chavez, et al. 2016) Every study has its own requirements and research goals, and to meet these goals, different measures and scales are used, modified, or invented all the time. Difficulty arises when meta-analysts wish to compare or combine such measures.

Missing data occur in nearly all clinical data sets, and thus missing data occur in IPD-MAs as well. Two types of missingness occur. *Sporadically missing data* occur when a variable is only partially observed in one or more studies included in the meta-analysis. Usually sporadic missingness occurs because a participant has not answered a question. On the other hand, *systematically missing data* occurs when a variable was totally unobserved or not collected in some studies included in the meta-analysis. Systematic missingness often occurs because the studies included in the meta-analysis had different study designs or budget constraints that did not allow them to collect all variables of interest. (Matteo Quartagno 2016) Both sporadic and systematic missingness present difficulties for meta-analysts.

Different terms are used for the process of combining information from studies with systematic missingness in some of the target variables. These include data fusion, statistical matching, and harmonization. The terms are often used in similar ways, which can lead to confusion. Data fusion and statistical matching specifically refer to the case where there are variables which are systematically missing for some data sets, and for these variables with systematic missingness, there is only one study which collected the vari-

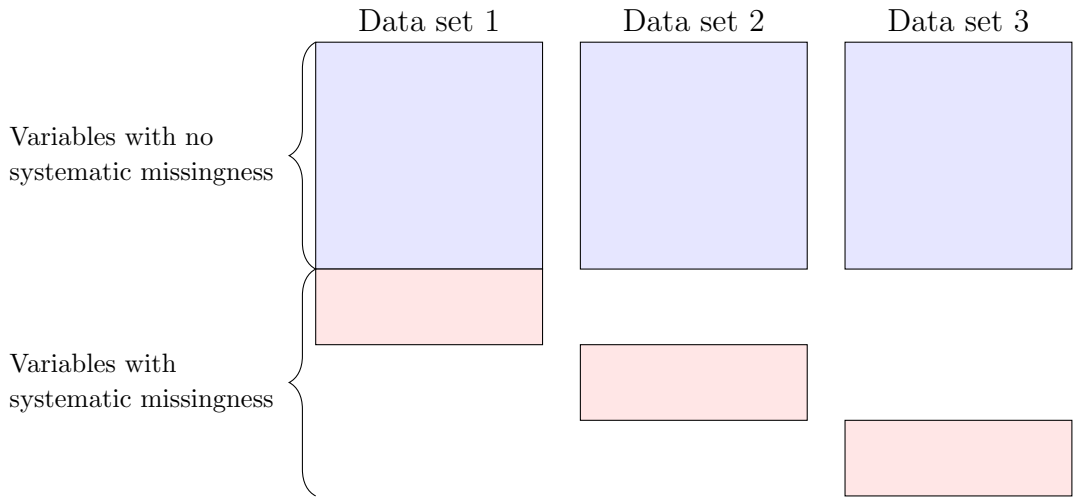

Figure 2.1: Case with no overlap of variables with systematic missingness between data sets

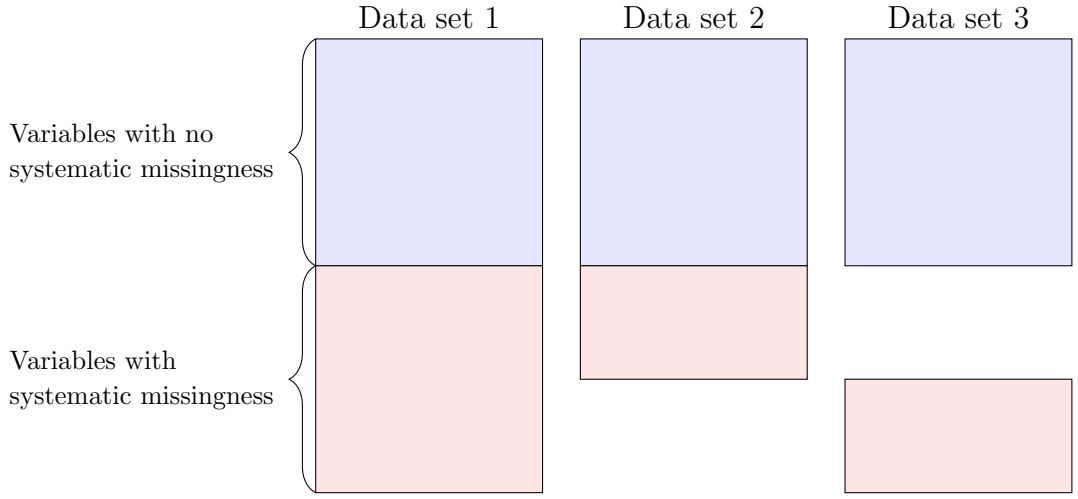

Figure 2.2: Case with overlap of variables with systematic missingness between data sets

able. This we call the *no overlap case*. (Conti, Marella, and Scanu 2017) It is illustrated in Figure 2.1. To contrast, the *overlap case* is illustrated in Figure 2.2. Harmonization refers to the more general process of achieving or improving the comparability of different survey data sets (L. E. Griffith, Van Den Heuvel, et al. 2015) or variables (Jolani et al. 2015). In L. E. Griffith, Van Den Heuvel, et al. (2015), the authors write that harmonization is "differentiated from aggregate meta-analysis in that individual participant data is pooled," which indicates that harmonization is used synonymously with IPD-MA in some cases.

Whatever the terminology used, if a researcher wishes to include a variable in their model that is not collected in one or more studies turned up in their literature review, this variable can be considered systematically missing in those studies. This allows the problem of measures collected differently for the same underlying construct to be treated with missing data methods. Next we will go over some fundamental definitions for the treatment of missing data.

2.3 Missing Data

Missing data is a problem that affects nearly all cohort and clinical data sets. If missingness is improperly handled, invalid inferences can result. (Matteo Quartagno 2016) Until 1987, there was no better mechanism for creating a complete data set from one with missing data than deleting all data for any participant that had missing values. This technique is known as complete case analysis. (Graham 2009) Many new techniques have since emerged, and these will be covered in detail in the next section. First, some definitions will be given.

Let R be a vector with an entry for each variable in the data set. The entry in R takes the value 1 if there is data for that variable, for that participant, and 0 otherwise. Missingness is then described in terms of these vectors, as in Table 2.1. Suppose we have three variables, Y, X_1 and X_2. Participants $1, 2$, and 3 have R vectors $\{1, 1, 1\}$, $\{1, 1, 0\}$, and $\{1, 0, 1\}$ respectively. Participant 1 has complete data, or no missing values. Participants 2 and 3 exhibit missingness.

Participant	Y	X_1	X_2
1	1	1	1
2	1	1	0
3	1	0	1

Table 2.1: Patterns of missingness

Three patterns of missingness were identified by Rubin (1986) to describe the way that missing data occurs in data sets structured like the cohort studies we are interested in.

- **Missing completely at random (MCAR)**: Missingness does not depend on observed or unobserved data. Cases for which data are missing can be thought of as a random sample of all cases. If the data is MCAR, any analysis of the dataset can be safely performed on a subset of the data that has no missingness - a complete case analysis. This will yield unbiased estimates (that is, close to population values) at the expense of a loss of statistical power. We will see later on that even in the case of MCAR, complete case analysis is still undesirable because of this loss of statistical power. That is, the ability to detect an effect, if the effect actually exists. In Table 2.1, a complete case analysis would proceed by deleting participants 2 and 3, and continuing as usual with the desired analysis. (Graham 2009)

- **Missing at random (MAR)**: Missingness depends on observed data, but not on unobserved data. That means the cause of missingness is described by the collected data. Once one has conditioned (or "controlled for") all the data one has, the remaining missingness is completely random. For example, perhaps participant response (or not) is related only to the age of the participant, with older participants less likely to respond to certain questions. If the variable *Age* is collected for every participant, then *Age* can be controlled for, and missingness is said to be MAR. Thus the term MAR is slightly misleading. It could be more accurately called "Conditionally missing at random." MAR missingness, when the cause of missingness is controlled for appropriately, will also yield unbiased parameter estimates. (Graham 2009) How to control for the cause of missingness is elaborated later in the text.

- **Missing not at random (MNAR)**: Missingness depends on unobserved data. MNAR is very difficult to identify because it relies on data that has not been collected. Note in the case of MAR, if the researcher does not control for the variables upon which missingness depends, then the data is effectively MNAR as well. (Graham 2009) MNAR missingness yields biased parameter estimates, and can severely inhibit the researcher's ability to perform analyses that describe real-world population values.

According to Graham (2009), missing data methods can be judged by three criteria:

1. The method should yield unbiased parameter estimates over a wide range of parameters, ie. the parameter estimate should be close to the population value.

2. There should be a method for assessing the degree of uncertainty about parameter estimates. Reasonable estimates of the standard error or confidence intervals should be obtainable.

3. Once bias and standard errors have been dealt with, the method should have good statistical power. Again, this means that the method should have the ability to detect an effect, if the effect actually exists.

There are a few commonly used methods for dealing with missing data in general, which we can check against these three criteria.

- **Mean substitution** replaces the missing variable with the mean for that variable where it was observed. This technique yields a mean for the complete variable that is close to the true parameter value, but other parameters (standard deviations, particularly) with this method can be seriously biased. Already, mean substitution fails the first criterion for unbiased parameter estimates and is not recommended. (Graham 2009)

- **Complete case analysis**, also known as listwise deletion, deletes all data for participants with missingness in any variable. This technique may yield biased parameter estimates if the groups with complete data are substantially different from those with missing data. However if the difference between these is entirely explained by the variables for which everyone has data, then the bias is often minimal from listwise deletion, especially for multiple regression models frequently applied by researchers using cohort data. There will always be some loss of power from listwise deletion however, since the partial data is unused. If the loss of cases is less than 5%, the increase in bias and loss of power are usually minimal, but it is still recommended to apply a different missing data method. Standard errors based on complete case analysis are still meaningful. (Graham 2009) Thus in certain cases, complete case analysis can meet all three criteria given by Graham (2009), but is in general not recommended.

- **Pairwise deletion** is a missing data method applied with a correlation matrix. Correlations are estimated based on cases that have data for both variables. Thus the correlations are based on different subsets of cases, which could introduce bias. In practice these tend to be small. However it is possible here that the matrix of correlations will not be positive definite. If this is the case, many multivariate statistical analyses will be unavailable to the researchers. There is also no way to obtain standard errors. (Graham 2009) Thus pairwise deletion is also not recommended.

- **Regression-based single imputation** is a missing data method that estimates a regression line from the complete data and then fills in missing values using the estimate from the regression line. Because there are no error terms, all imputed values fall precisely on the regression line and thus bias the estimates toward the regression line. For this reason single imputation is not recommended in general. (Graham 2009) Multiple imputation methods will be covered in depth later on.

- **Expectation Maximization (EM), Multiple imputation (MI), and Full Information Maximum Likelihood (FIML)** are more commonly used, especially recently, and these are discussed next in detail. Each demonstrates the desirable properties given by the criteria above.

Expectation Maximization (EM)

An EM algorithm reads in the raw data, with missing values, and reads out a maximum likelihood covariance matrix and a vector of means. There are other EM algorithms that read out different end products. EM is an iterative procedure. The following characterization of one relevant EM algorithm is given by Graham (2009):

- E-step: Each iteration goes through an E-step, where cases are read in. If a value exists, then sums, sums of squares, and sums of cross-products are incremented. If it is missing, a current estimate for the value is used instead. This estimate is generated using a regression-based single imputation, using all the other variables in the model as predictors. For sums, this estimate is used as is. For sums of squares and cross-products, if only one of the values is missing then the quantity is incremented. If both are missing, then the quantity is incremented and a correction factor is added. The correction is conceptually the same as adding a random residual error term as in MI (see next section).

- M-step: Each iteration next goes through an M-step, wherein the variances, covariances, and means (the parameters) are estimated based on current values of sums, sums of squares, and cross-products. Using the covariance matrix at this iteration, new regression equations are calculated for each variable predicted by all the other variables. These regression equations are then used to update the estimates for missing values during the E-step of the subsequent iteration.

The E and M steps are iterated until elements of the covariance matrix stop changing significantly by some pre-defined criteria. The EM algorithm is then said to have converged.

The parameter estimates for means, variances, and covariances are excellent, but the EM algorithm does not give standard errors (SEs) automatically. SEs can be estimated with bootstrap procedures. Because the SEs are not convenient to generate, EM is not particularly good for hypothesis testing that is often required for cohort analyses. That said, many standard analyses, including preliminary analyses, don't require SEs anyway. EM is recommended for reporting of means, standard deviations, and correlation matrices in published work. Certain data quality analyses such as coefficient alpha analyses can be based on the EM covariance matrix. Exploratory factor analysis with missing data can be performed using the EM covariances matrix too.

A single dataset can be imputed using EM parameters with random error. This yields good parameter estimates, close to the population average. The complete dataset can also be used in software that requires no missingness, such as SPSS. Note though that such a

dataset should not be used for hypothesis testing. SEs based on this data (from a multiple regression analysis, for example), will be too small. Hypothesis testing is better carried out with MI or FIML. The procedure in SPSS that outputs a single imputed dataset based on EM is not recommended unless random error residuals are added afterwards to each imputed value. (Graham 2009)

Multiple Imputation (MI)

Multiple imputation is very commonly used to deal with missingness in statistical analyses. In Matteo Quartagno (2016), the author describes MI as a three-step process.

1. Imputation: A distribution is specified according to the data structure, including a choice of levels or clusters. This distribution is called the posterior predictive distribution, and it defines the imputation model. $M > 1$ complete, imputed data sets are drawn from the posterior predictive distribution, taking into account the observed data. M typically varies between 3 and 10. (Stef Van Buuren 2007)

2. Analysis: The desired statistical analysis is applied to each of the M complete imputed data sets, so that M estimates of the model parameters and their variances result.

3. Combination: These M model parameter estimates and variances are pooled, usually according to Rubin's rules. (Rubin 2004)

Using the notation in Stef Van Buuren (2007), let Y_j be one of k incomplete random variables, and $Y = (Y_1, \ldots, Y_k)$. The observed part of Y_j is denoted Y_j^{obs}, and the missing part is denoted Y_j^{mis}. Let $Y_{-j} = (Y_1, \ldots, Y_{j-1}, Y_{j+1}, \ldots Y_k)$. R_j is the response indicator for Y_j, so $R_j = 1$ if Y_j is observed, and is 0 if missing. Similar to Y, $R = (R_1, \ldots, R_k)$ and $R_{-j} = (R_1, \ldots, R_{j-1}, R_{j+1}, \ldots R_k)$. Suppose further that $X = (X_1, \ldots, X_l)$ are complete variables collected on the same participants.

The creation of complete imputed data sets in Step 1 can be performed in two ways: with *joint modeling* or with *fully conditional specification*.

With joint modeling, Step 1 is divided into three tasks: modeling, imputation, and estimation. (Stef Van Buuren 2007) In the modeling task, a hypothetical joint distribution $P(Y, X, R)$ is specified. The imputation task derives a posterior predictive distribution $P(Y^{mis}|Y^{obs}, X, R)$. The estimation task calculates the posterior distribution of the parameters, after which random draws can be taken from the distribution. The entire joint modeling process takes place under the assumption of MAR or MCAR missing data. The difficulty lies in specifying a model $P(Y, X, R)$ in the modeling task. The joint model must be able to calculate realistic imputed values and be congenial with the analysis model applied in Step 2, ie. it must be at least as complex as the analysis model. Joint models have been developed that allow imputation under multivariate normal (when all variables are continuous and normally distributed), log-linear (for binary variables), and a general location models (for both continuous and categorical variables). According to Stef Van Buuren (2007), these methods sometimes "lack flexibility to account for important features of the data," because derived variables like sum scores and transformations require consistency between the pre- and post- transformation variables, and contradictory results can be generated.

Instead of specifying the joint model $P(Y, X, R)$ explicitly, an imputation model $P(Y^{mis}|Y^{obs}, X, R)$ can instead describe how Y^{mis} should be generated. This technique

is known as fully conditional specification. Here, the modeling step in the joint modeling method is bypassed, and instead imputations are generated variable-by-variable, specifying conditional model $P(Y_j^{mis}|Y_{-j}, X, R)$ for $j = 1, \ldots, k$. The process is iterative, with one iteration cycling through each Y_j for $j = 1, \ldots, k$. Stef Van Buuren (2007) writes that "If the joint distribution defined by the specified conditional distributions exists, then this process is called a Gibbs sampler." Fully conditional specification can be more flexible than joint modeling in creating multivariate models because they do not need to adhere to known standard multivariate models. Features of the data such as restricting to possible values or bounds can be incorporated into the specification.

So in the first step, joint modeling occurs if a multivariate joint distribution is specified as the imputation model for all variables. Alternatively, fully conditional specification occurs if a conditional distribution is defined for each incomplete variable. (Stef Van Buuren et al. 2006) Both methods usually produce unbiased estimates. (Stef Van Buuren 2007) If all variables are normally distributed, joint modeling and fully conditional specification produce identical results.

The second step simply involves applying the desired analysis model to each of the complete, imputed data sets separately. Any model that can be applied to a complete data set can be applied here.

In the third step, Rubin's rules are applied as follows (Marshall et al. 2009)

> For regression coefficient b with estimates $\hat{b}_1, \ldots, \hat{b}_M$ for each of the M imputations, these are combined into a single estimate for b using the average. $\hat{b} = \frac{1}{M}\sum_{i=1}^{M}\hat{b}_i$. The associated estimated total variance is given by combining the imputation-specific variances $\sigma^2_{b_1}, \ldots, \sigma^2_{b_M}$ to get $\hat{\sigma}^2_b = W + \left(1 + \frac{1}{M}\right)B$, for $W = \frac{1}{M}\sum_{i=1}^{M}\hat{\sigma}^2_{b_i}$ the average within-imputation variance, and $B = \frac{1}{M-1}\sum_{i=1}^{M}\left(\hat{b}_i - \hat{b}\right)^2$ is the between-imputation variance. The output is a vector of estimates of regression coefficients, and their associated covariance matrix.

Full Information Maximum Likelihood (FIML)

FIML deals with missing data, performs parameter estimation, and estimates standard errors in one step. Like EM and MI, FIML assumes that the missingness is either MAR or MCAR. The algorithm proceeds by computing a likelihood function for each participant, using only the variables that are observed for that participant. The likelihood of the observed data is then maximized using an appropriate function to match the data distribution (for example, multivariate normality). The maximized likelihood functions for each participant are combined and again maximized over the means and covariances of the parameters. (C. K. Enders and Bandalos 2001)

Because handling of missing data and the estimation of parameters and standard errors happens in a single step, regular complete-case algorithms need to be rewritten to handle missing data. In 2009 this rewriting was limited. FIML is most commonly available for SEM (Structural Equation Modeling) software such as Amos, LISREL, Mplus, ad Mx. These programs were written for SEM but can be used for any analysis using a general linear model, such as multiple regression. (Graham 2009)

Summary of available missing data methods

There are many common misconceptions surrounding the danger and impact of missing data and various types of missingness. (Graham 2009) Some researchers even recently,

such as Akl et al. (2015), are still advocating complete case analysis. There is also much speculation about the dangers of MNAR missingness. Graham (2009) addresses this by citing simulation studies that used MNAR data, that were not heavily affected by the MAR assumption common in MI analyses.

The best available methods for dealing with missingness in cohort data, according to the three criteria given at the beginning of the section, are multiple imputation, full information maximum likelihood, and expectation maximization.

If the researcher requires only one imputed dataset for analysis, Graham (2009) recommends imputation based on EM parameters along with random error (using the NORM program). All parameter estimates in this case are near the center of the parameter space, unlike in datasets imputed with DA, where parameter estimates can be anywhere in the parameter space. If hypothesis testing is not required, EM is recommended. If hypothesis testing is required, as in most applications in the health sciences, then MI or FIML are recommended. (Graham 2009)

Treatment of Missing Data in IPD-MA

There are two forms in which missing data appears in IPD-MA. Sporadic missingness refers to the situation where data is missing in spite of the variable having been collected in the study. (Audigier, White, et al. 2017) Systematic missingness occurs when some variables are totally unobserved for one or more studies. (Resche-Rigon, White, et al. 2013) In terms of cohort data, sporadic missingness occurs if the patient did not answer a question, and systematic missingness occurs if the question was not included in the survey. Audigier, White, et al. (2017) notes that this distinction between types of missingness can be applied to any two-level setting. Here, the levels are patients nested within surveys, and surveys within the meta-analysis. For treatment of missingness in IPD-MA, a failure to account for the clustering of data sets included in the analysis will usually introduce bias. Matteo Quartagno (2016)

Sporadic missingness can be effectively dealt with by the procedures recommended above, namely multiple imputation, full-information maximum likelihood, or expectation maximization algorithms. Sporadic missingness occurs in nearly all cohort studies, and is extensively covered in the literature. (Resche-Rigon, White, et al. 2013) Most authors use random effect models for their imputation model, thus preserving the multilevel structure within the data. (Audigier, White, et al. 2017)

The research question at hand can be framed as a systematically missing data problem: if two or more studies collect data on the same underlying construct using different measures, the measures that were not collected can be considered systematically missing for that study. A good method for tackling this problem should be able to handle both sporadically and systematically missing data. Since multiple imputation methods for sporadically missing data cannot in general be applied without modification to systematically missing data, a specific look at methods for systematic missingness is required. (Audigier, White, et al. 2017)

2.4 IPD-MA for Systematically Missing Variables

Any method for addressing systematic missingness in IPD-MA settings, for target variables that measure an underlying construct, should ideally meet a few criteria.

1. Addressing systematic and sporadic missingess: nearly all cohort studies experience sporadic missingness from participant non-response, so when we are looking at methods for systematically missing data we need to keep sporadic missingness in mind as well.

2. Addressing between-study heterogeneity: because we are working in a multilevel framework with IPD-MA, any analysis model must take into account clustering within study resulting from between-study heterogeneity. Any missing data method must exhibit a multi-level structure to account for this.

3. Not making overly strong assumptions about conditional independence: often the goal of the missing data method will be to estimate a joint model of systematically missing variables, ie. to find the relation between variables that exhibit systematic missingness. The method should not assume that the systematically missing variables are conditionally independent, or independent given the variables observed in both data sets. (Rässler 2003) This assumption is impossible if the systematically missing variables are highly correlated with one another, as one would expect if the target variables have an underlying latent construct. For the assumption of conditional independence to work, the target variables would need to be completely explained by the covariates collected in all studies.

4. Non-informative prior distributions: in the case of multiple imputation methods for addressing systematic missingness, it is advisable that the prior distribution not make overly strong assumptions about the eventual correlation structure.

Three common methods for harmonizing different measures collected on the same underlying construct are identified by Siddique, Reiter, et al. (2015):

- Linear or z-transformation can be used to create a common metric across data sets. This technique does not require common measures across studies. This method however does not take into account between-study heterogeneity, and assumes that underlying constructs are the same and measured equally well across studies. This assumption is often too strong. Interpretation of the transformed variable can also present difficulties for researchers.

- Latent variable methods identify underlying constructs across data sets. Siddique, Reiter, et al. (2015) argues that sometimes overly strong assumptions about latent structure must be made, particularly when there is no overlap in measurements between studies.

- Multiple imputation methods treat systematically missing measures as missing data, replacing the missing data with plausible values, creating multiple data sets. Unlike latent variable and z-transformation methods, multiple imputation does not create a new scale. The interpretation of the imputed measures is more practical. However, the multiple imputation methods commonly applied to the case of sporadically missing data cannot in general be applied directly when systematically missing data exists.

Existing Overlap Within Studies

Audigier, White, et al. (2017) performs a very recent review of methods available for multilevel data with both systematically and sporadically missing values in continuous

and binary variables, in the case where overlap exists between studies. This case, again, is illustrated above in Figure 2.2. Both the cases where only one variable has missing data, and where several variables have missing data are explored. The most recent recommended methods were multiple imputation based, and of two kinds: fully conditional specification (FCS) and joint modeling (JM) approaches. Resche-Rigon, White, et al. (2013) recommends a FCS approach for systematically missing continuous data, which was extended by Jolani et al. (2015) to include non-continuous data types. According to Audigier, White, et al. (2017), these methods were time-consuming in the common case where many variables are incomplete. Resche-Rigon and White (2016) streamlined the approach, making the estimators easier to fit. Joint modeling methods for systematically missing data have been proposed by Yucel (2011) and M Quartagno and JR Carpenter (2015). The methods explored in Audigier, White, et al. (2017) are the most up-to-date methods, namely the JM approach with the `jomo` package for R, recommended by M Quartagno and JR Carpenter (2015), and the FCS approaches in Jolani et al. (2015) and Resche-Rigon and White (2016). The authors performed a real data analysis and a simulation study before offering some recommendations. Ultimately all three methods performed relatively well. JM with `jomo` was recommended by the authors when the number of incomplete binary variables is large and the number of observed levels is large. The FCS approach from Resche-Rigon and White (2016) should be avoided if the number of levels is small, or when the proportion of sporadically missing values is large. In particular it is recommended if the number of levels with systematically missing variables is large. The FCS approach implemented by Jolani et al. (2015) is recommended when the levels contain a small number of participants. These methods will be explored in depth in the next chapter.

No Overlap Within Studies

Data fusion is a term used to describe the situation where two or more data sets, with different samples of the same population are combined, with some variables appearing in one data set or another, but no individual data set collects all variables of interest. (Siddique, Reiter, et al. 2015) This is illustrated in Figure 2.1. According to Gilula and McCulloch (2013), data fusion as a practice began with the work of Kamakura and Wedel (1997), who framed the question and introduced the earliest methodologies. The variables that are never jointly observed, that are the target of analysis, are called the target variables. For many data fusion analyses, the target variables are related in some way by an underlying construct. The researcher usually either wishes to make an inference about the joint distribution of these target variables, or to create a complete data set by treating the issue as a missing data problem. (Gilula and McCulloch 2013) In Kiesl and Rässler (2006), the authors state that the data fusion problem cannot be solved without the use of additional information, or by making some assumptions. The assumptions usually involve conditional independence: that is, the target variables are assumed independent once the jointly observed variables are accounted for. If this assumption is made, the joint distribution can be obtained from the marginals. (Gilula and McCulloch 2013) This assumption depends heavily on the existence of richly informative common variables, and as is noted in the criteria given for systematic missingness data methods, is often too strong an assumption.

Gilula and McCulloch (2013) introduces a method for the situation with two data sets, which have two non-jointly observed categorical target variables. Their method relies on

the use of a third data set which does jointly observe the target variables, sometimes known as a calibration data set. (Siddique, Reiter, et al. 2015)

Siddique, Reiter, et al. (2015) also applies a method that makes use of external calibration data. This paper does not perfectly address our research question, because the studies they harmonize are longitudinal randomized controlled trials. The method uses external data from calibration studies, but the data used to calibrate is not included in their final analyses. The calibration data helps estimate relationships between the two measures, and allows the use of diagnostics to test how well the imputed values approximate observed ones. If there was overlap within at least one study between the two measures, then the calibration data would be unnecessary but perhaps still useful. The method is as follows: The external calibration trials are concatenated with the five trials included in the meta-analysis. Missing data are imputed using MI, and then diagnostics are performed to verify that the imputed data behaves similarly to the observed data. The external calibration trials are then removed, and statistics are collected from the multiply imputed data sets of the IPD-MA. Specifically, Siddique, Reiter, et al. (2015) applied a multivariate linear mixed effects model in their multiple imputation. The model described by (Siddique, Reiter, et al. 2015) assumes the data are MAR, that the partial correlation between the the target variables does not depend on treatment group (placebo or non) or by trial, and that observations are independent within trials in the imputation model. This last assumption was made because the correlation between target variables was negative, which was unlikely to be true and more likely to be due to small sample size.

After imputation, the calibration data was then removed and the remaining data was analyzed. Reiterś method of two-stage multiple imputation, and its combining rules, were applied. (Reiter 2008) According to this method, m draws of the parameters from the imputation model are sampled, and then nested within parameter draws, n imputations are generated for each missing value. mn complete data sets result. Analyses are performed on each data set separately and then combined according to Reiter's method. Two checks on the imputation model were performed: graphical comparison of observed vs. imputed data, and posterior predictive checks with numerical summaries from test statistics.

Further use of calibration data:

- S Van Buuren et al. (2003) applies what they call response conversion to convert a few measures of activities of daily living onto a common scale. Once they have developed a conversion key, essentially a formula which transforms the activity of daily living measure onto the common scale, they write that this key only needs to be developed once, and then can be applied to other data sets

- Carrig et al. (2015) applies a non-parametric multiple imputation method that also makes use of a calibration sample, that they collect after deciding which measures they are targeting. They argue against a 'transform and recode' approach for putting the target variables onto a common scale, that does not make use of calibration data, because it "makes the untested, strong assumption that the transformed scores derived from different variables are identically related to an underlying target construct.

- Crane et al. (2008) used item response theory methods to co-calibrate four scales used to identify cases of dementia. They identified anchor items that were similar across cognitive tests.

- Schifeling, Reiter, and DeYoreo (2016) uses auxiliary data to adjust for measurement error.

Often it is impossible to use calibration data, either because data sets that do jointly observe the variables to be harmonized are inaccessible or non-existent, or because collection of new data is expensive and time-consuming. Sometimes grants do not allow for the collection of new data, only the use of existing data. For this reason we explore studies that do not make use of calibration data, and discuss the additional assumptions that must be made under this framework.

Studies that did not use calibration data:

- Curran and Hussong (2009) introduced the so-called measurement model integration (MMI) approach, which assumes there is an unobserved latent propensity underlying the observed measurements from the instrument or scale. The model uses item response theory to fit each target variable to an underlying score. These latent variable scores are then used in subsequent analysis. Carrig et al. (2015) however, questions the trust researchers must place in the equivalence underlying the scores. If the latent structures are multidimensional, or if the number of jointly observed variables is small, the misspecification of MMI models can lead to invalid estimated scale scores. This method also does not take into account between-study heterogeneity.

- Conti, Marella, and Scanu (2013) introduces a measure of uncertainty for the estimation of a joint distribution of two target variables that were not jointly observed. They advocate the choice of a class of possible distributions, and quantify uncertainty in the specification of the joint model according to their new measure. That is - they use informative priors. Later in Conti, Marella, and Scanu (2016), they introduce the notion of matching error, which is upper-bounded using their measure of uncertainty. The estimate of the joint distribution of the target variables is then constructed from a modified version of the conditional independence assumption, subject to certain logical constraints. Most recently in Conti, Marella, and Scanu (2017), the same authors gave a systematic overview of the problem, analyzing uncertainty in the data fusion problem.

- D'Orazio, Di Zio, and Scanu (2012) performed a review of methods for data fusion, or as they call it, statistical matching. They compared random hot-deck imputation with and without study weighting, and predictions from linear probability models with some weighting calibrations. They advocate taking the survey design into account by including some proxies of the sampling design, even if these are not highly correlated with the target variables. Both of these single imputation methods have been called into question earlier in this chapter.

- Fleischer and Groenitz (2013) criticized the conditional independence assumption that underlies many statistical matching techniques. The authors concluded that most statistical matching methods require overly strong assumptions.

- Rässler (2003) introduced a multiple imputation technique that is based on informative prior distributions, and does not consider between-study heterogeneity. The data sets are simply concatenated.

- Reiter (2012), and Rubin (1986) like Rässler (2003), concatenates the files and performs a multiple imputation. File concatenation does not usually take into account between-study heterogeneity, and this is no exception.

- Zhang (2015) uses proxy variables: choosing covariates that exhibit strong correlation (or predictive power) to the systematically missing variables. This technique also does not take into account between-study heterogeneity.

Overall the methods for the case of no overlap between studies are not particularly inspiring. Each of the proposed methods suffers from some issue highlighted at the beginning of the section. Whether making overly strong assumptions about conditional independence, failing to account for between-study heterogeneity, or using informative prior distributions, none of these methods exhibited all the desired qualities we mentioned.

Harmonization of Cognitive Scales

My research will focus in particular on missing data methods for IPD-MA with systematically missing cognitive scales to be harmonized, so a look at the issues particular to this subject is required.

Many memory scores are collected as ordinal scales, meaning that the scale is categorical but ordered, such as an integer score out of 30 points. It is not necessarily known what the distances between categories mean. For many scores, the outcome is calculated by addition from a series of binary correct/incorrect questions that are each considered to be equally difficult. This is the case for commonly used measures such as the Mini-Mental State Examination, the Abbreviated Mental Test Score, the Rey Auditory Verbal Learning Test, the Alzheimer's Disease Assessment Scale - cognitive subscale, found in systematic reviews of cognitive outcomes given by Vogels et al. (2007) and Harrison et al. (2016). In these systematic reviews the only other scale listed was a Trail Making Test, which is measured in time to task completion, and is thus continuous. Because ordinal scales are so common in this field, a focus on methods for such ordinal scales is warranted.

Multiple imputation is one possibility for the treatment of missing ordinal data. Two main methods present themselves: treatment of the ordinal variable as continuous and applying usual MI methods for multivariate normal data, or applying methods that treat the ordinal variable as categorical.

The method of multivariate normal MI was explored in a simulation study by Chen et al. (2005). The authors found that for imputation of missing ordinal data, sporadically missing variables under a multivariate normal model with naive rounding (using 0.5 as the threshold to round up or down to the nearest integer) produced worse results than complete case analysis alone under the MAR assumption. 10 years later, Wu, Jia, and C. Enders (2015) performed a more comprehensive review of methods for sporadically missing ordinal data. The authors tested multivariate normal models, both with and without rounding up or down to the nearest integer. They found that the multivariate normal approach without rounding performed well for missing variables with more than two levels, for different sample sizes, missing data proportions, and asymmetrical item distributions. Naive rounding under the multivariate normal model, as in Chen et al. (2005) was found to introduce substantial bias, so this approach is not recommended.

Multiple imputation with methods developed for categorical variables is another possibility. Linear discriminant analysis and logistic regression models are two such options.

Imputation is then usually performed with FCS methods such as the `mice` package for R. In these cases, imputed values belong to categories and need not be rounded.

According to Wu, Jia, and C. Enders (2015), linear discriminant analysis "assumes that predictors of group membership follow a multivariate normal distribution, with means that vary across categories but a covariance matrix that is constant over categories." Thus for each imputation, group means and a common covariance matrix are drawn from posterior distributions. These are then used to derive a linear combination of predictors (called the discriminant function) that maximizes group difference. Together with the prior probability for category membership, the discriminant function computes the posterior probability that the participant with the missing value belongs to each category. The imputed value is the category that receives the highest probability in this calculation. This technique produced good results in the simulation by Wu, Jia, and C. Enders (2015).

Logistic regression models, on the other hand, can be applied with either multinomial or proportional odds models. Multinomial logistic regression is meant for categorical data, but can be applied here. If there are K levels of the ordinal variable to be imputed, for each level j, the model calculates an equation that links predictors linearly with the log odds of falling into any of the $K-1$ other categories, vs. level j. Intercepts and regression coefficients vary accross equations. Proportional odds regression models calculate $K-1$ equations as well, linking the predictors linearly with the log odds of falling into or below each category, versus falling above it. (Wu, Jia, and C. Enders 2015) In the proportional odds model, however, the proportional odds assumption must hold: the regression coefficients are assumed constant across equations, so the influence of the covariate is assumed equal across all adjacent categories of the ordinal variable. This approach did not produce the desired results in the simulation by Wu, Jia, and C. Enders (2015).

Comparing the two categorical MI models, the proportional odds model takes into account the order of response categories for the nominal variable, while multinomial logistic regression does not. However the multinomial logistic model does not require the proportional odds assumption. Both of these models are available in the R package `mice`.

Expectation-maximization (EM) can also theoretically be applied to sporadically missing ordinal data. The assumption of multivariate normality was mentioned by Wu, Jia, and C. Enders (2015), and then they wrote that EM can proceed as usual. Using bootstrapping techniques, imputed data sets can be drawn from the EM estimate. (Wu, Jia, and C. Enders 2015) The authors did not test this model, and no specific instances of EM for missing ordinal data (whether sporadic or systematic) were found in the literature.

Wu, Jia, and C. Enders (2015) also tested latent variable models. With this strategy there is an assumed latent variable underlying each ordinal variable (even ones that display no missingness), and these latent variables are assumed to follow a multivariate normal distribution. So, imputed values are drawn from such a multivariate normal distribution, and then the result can be assigned to a discrete level based on estimated thresholds. This approach worked well for missing variables with more than two levels, for different sample sizes, missing data proportions, and asymmetrical item distributions. Thus after testing multivariate normal and categorical MI methods, with and without rounding, and expectation maximization, the authors recommended either the latent variable approach, the multivariate normal MI without rounding, or the discriminant analysis approach.

All the above methods were tested on sporadically missing ordinal variables. No papers were found that specifically address systematically missing ordinal variables. The package `jomo` for R offered by M Quartagno and JR Carpenter (2015) can perform multilevel joint

modeling on binary and categorical data using latent normal models, but does not go into ordinal methods specifically. The two-stage fully conditional specification approach offered by Resche-Rigon and White (2016) can accommodate binary and categorical data as well, and similarly does not cover the case of ordinal outcome variables specifically.

In terms of analysis models, the most common regression methods are intended for continuous, normal data. Linear regression models that accommodate different data types are known as generalized linear models. Generalized, because outcome types other than continuous are accommodated; linear, because there are no quadratic or higher powered terms; and mixed, because there are both fixed and random effects allowed within the model.

2.5 Discussion

A few important methods for treatment of missing data in general have been identified: namely multiple imputation, expectation maximization, and full information maximum likelihood. Statistical approaches for handling both systematically and sporadically missing data were investigated, and methods for data sets without overlap in the target variables were found to be lacking. A few methods for the overlap case merit further attention. All were multiple imputation methods, two of which were fully conditional specification approaches, and one of which was a joint modeling approach. The review then narrowed to consider methods for ordinal data of the type usually collected for ordinal scales. A few methods presented themselves for sporadically missing data, but none appeared for systematically missing data. A clear gap in the field exists for the consideration of systematic missingness in ordinal data. The best methods identified in the literature review, with potential for application to our context, are detailed in the next chapter.

Chapter 3

Literature Review

In the literature review section, a few of the most promising avenues mentioned in the last chapter will explored in depth. These methods are the state of the art in this field, all introduced in 2015 or later.

3.1 Methods from the Literature

Three of the most promising methods from the literature for multiple imputation are now examined in detail. These are from papers by M Quartagno and JR Carpenter (2015), Resche-Rigon and White (2016), and Jolani et al. (2015). The first uses joint modeling to impute missing values, and the last two use fully conditional specification. See Section 2.3 for more information on the distinction between these two MI methods.

To be perfectly applicable to the simulated data sets, these methods need to meet a few criteria. They must be able to handle both systematic and sporadic missingness under the MAR assumption. They also must be able to handle several value types. Sex is binary, physical activity and education are categorical, age is continuous, and the memory scores are ordinal. Ideally, a method for completing our data sets should be able to handle all these value types.

M Quartagno and JR Carpenter (2015)

Assumptions: all missing data are missing at random. Both sporadic and systematic missingness are accounted for.

The relevant analysis model given in their paper concerns a continuous outcome Y and two continuous covariates X_1 and X_2.

> **Stage one** In the first stage a within-study regression is performed, regressing $Y_{i,s}$ on $X_{1,i,s}$ and $X_{2,i,s}$ simulataneously. The regression is given for participant i in study $s \in \{1, \ldots, S\}$.
>
> $$Y_{i,s} = \beta_{0,s} + \beta_{1,s} X_{1,i,s} + \beta_{2,s} X_{2,i,s} + \epsilon_{i,s} \tag{3.1}$$
>
> where
>
> $Y_{i,s}$ is the outcome Y for participant i in study s
> $X_{1,i,s}$ is the covariate X_1 for participant i in study s
> $X_{2,i,s}$ is the covariate X_2 for participant i in study s

$\beta_{0,s}$ is the intercept of the study-specific regression line
$\beta_{1,s}$ is the study-specific effect of covariate X_1 on Y
$\beta_{2,s}$ is the study-specific effect of covariate X_2 on Y
$\epsilon_{i,s}$ is the study-specific error. It is assumed that $\epsilon_{i,s} \sim N(0, \sigma_s^2)$

Thus, estimates of $(\hat{\beta}_{j,s}, \hat{s}_{j,s})$ for $j = 1, 2$ are obtained, where $\hat{\beta}_{j,s}$ is the estimated study-specific effect of covariate X_j on Y, and $\hat{s}_{j,s}$ is the estimated standard error of $\hat{\beta}_{j,s}$.

Stage Two A random effects meta-analysis model is fitted, using DerSimonian/Laird estimate of between-study heterogeneity (DerSimonian and Laird 1986), so for $j = 1, 2$:

$$\hat{\beta}_{j,s} = \beta_{random,j} + u_{j,s} + \hat{s}_{j,s}\epsilon_s \tag{3.2}$$

where $\epsilon_s \sim N(0, 1)$ and $u_s \sim N\left(0, \tau_j^2\right)$.

Their imputation model assumes that all variables have systematic missingness, and has a random study-specific covariance matrix. Thus it is congenial with 3.1. It is given by

$$\begin{aligned} X_{2,i,s} &= \alpha_{0,s}^1 + \epsilon_{i,s}^1 & \alpha_{0,s}^1 &= \alpha_0^1 + u_s^1 \\ Y_{i,s} &= \alpha_{0,s}^2 + \epsilon_{i,s}^2 & \alpha_{0,s}^2 &= \alpha_0^2 + u_s^2 \\ X_{1,i,s} &= \alpha_{0,s}^3 + \epsilon_{i,s}^3 & \alpha_{0,s}^3 &= \alpha_0^3 + u_s^3 \end{aligned}$$

Such that

$$\begin{pmatrix} u_s^1 \\ u_s^2 \\ u_s^3 \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \Omega_u\right) \qquad \begin{pmatrix} \epsilon_{i,s}^1 \\ \epsilon_{i,s}^2 \\ \epsilon_{i,s}^3 \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \Omega_{e,s}\right)$$

Here, $\Omega_{e,s} \sim IW(a, A)$, the inverse-Wishart distribution with parameters a, A. The authors write that they choose the identity matrix with minimum scale parameter as the inverse-Wishart prior, and thus we have a random covariance matrix. They note that other distributions could be used if desired. Information on the Wishart and inverse-Wishart distributions can be found in the Appendix.

Resche-Rigon and White (2016)

Assumptions: all missingness occurs at random. All variables are continuous and normally distributed.

The relevant analysis model in their paper concerns two-level clustered data given by the following linear mixed model.

$$Y_i = \mathbf{X}_i\beta + \mathbf{Z_i}b_i + e_i \tag{3.3}$$

Y_i is an $n_i \times 1$ vector of observed outcomes on units $j \in \{1, \ldots, n_i\}$ within cluster $i \in \{1, \ldots, N\}$. $Y_i = \{y_{i1}, \ldots, y_{in_i}\}^\top$
$\mathbf{X}_i$ is an $n_i \times p$ matrix of variables associated with Y_i via β
β is a $p \times 1$ vector of fixed effects
$\mathbf{Z}_i$ is an $n_i \times q$ matrix of variables associated with Y_i via b_i
b_i is a $q \times 1$ vector of random effects, such that $b_i \sim N(0, \Psi_b)$
Ψ_b is a $q \times q$ variance-covariance matrix of random effects

e_i is an $n_i \times 1$ error vector, such that $e_i \sim N(0, \Sigma_i)$, and $\Sigma_i = \sigma_i^2 I(n_i)$, where $I(n_i)$ is an $n_i \times n_i$ identity matrix

The posterior distribution of b_i given Y_i is

$$b_i|Y_i \sim N(m(Y_i), v(Y_i)) \tag{3.4}$$

$$m(Y_i) = \Psi_b \mathbf{Z}_i^\top \left(\mathbf{Z}_i \Psi_b \mathbf{Z}_i^\top + \Sigma_i\right)^{-1} (Y_i - \mathbf{X}_i \beta) \tag{3.5}$$

$$v(Y_i) = \Psi_b - \Psi_b \mathbf{Z}_i^\top \left(\mathbf{Z}_i \Psi_b \mathbf{Z}_i^\top + \Sigma_i\right)^{-1} \mathbf{Z}_i \Psi_b \tag{3.6}$$

Exploring missingness, the authors write that $\mathbf{Y} = \left(Y_1^\top, \ldots, Y_N^\top\right)^\top$ contains missing data $\mathbf{Y}^{mis}$. The observed data is given by $\mathbf{Y}^{obs}$. If Y_i is missing entirely, then it is said to be systematically missing for cluster i. It is sporadically missing if it is partially incomplete. The goal is to generate independent draws under the missing at random assumption, from the posterior predictive distribution for missing data given by

$$P\left(\mathbf{Y}^{mis}|\mathbf{Y}^{obs}\right) = \int P\left(\mathbf{Y}^{mis}|\mathbf{Y}^{obs}, \theta\right) P\left(\theta|\mathbf{Y}^{obs}\right) d\theta \tag{3.7}$$

Here, $\theta = (\beta, \Psi_b, \{\sigma_i\})$ is a vector of parameters in Model 3.3, and $P\left(\theta|\mathbf{Y}^{obs}\right)$ is the observed data posterior density of θ.

In general, independent draws for $P\left(\mathbf{Y}^{mis}|\mathbf{Y}^{obs}\right)$ are achieved by a three step process:

1. Fit model $P(Y|\theta)$ to units with observed Y. Get estimate $\hat{\theta}$ from an MLE, with estimated variance covariance matrix S_θ

2. Draw a value of θ, θ^* from its posterior, approximated by $N\left(\hat{\theta}, S_\theta\right)$

3. Draw values $\mathbf{Y}^{mis}$ from $P(Y|\theta^*)$

The authors use multiple imputation by chained equations (MICE) to "specify the multivariate imputation model on a variable-by-variable basis by a set of conditional densities obtained by different regression models, one for each incomplete variable." (Resche-Rigon and White 2016) So, the imputation model is adapted to each variable type. In an iterative sequence, conditional distributions for the missing data given the other data for each incomplete variable are obtained, and missing values are replaced by simulated draws. It can be initiated using basic random sampling with replacement from the observed data. After imputation, the whole cycle is repeated again for a pre-specified number of imputations. This repetitions helps stabilize the posterior distribution for each variable. The posterior distributions can be gathered from maximum likelihood estimators or from Gibbs samplers.

At each step a conditional imputation must be generated. The authors propose a novel method for generating these conditional imputations. The conditional imputation model should ideally be compatible with the (unknown) overall data model. However the authors find that the problem of generating conditional distributions of a random slopes model to be intractable. Hence, the authors make some approximations that misspecify the conditional model. The performance of their approximation is investigated in a simulation and found to be good for their purposes.

Jolani et al. (2015)

Assumptions: No sporadic missingness of data. Both continuous and binary variables are allowed (unlike the other methods). Only covariates are allowed to experience systematic missingness, and these can be either binary or continuous form. Analysis models for binomial valued outcomes were also considered.

Like Resche-Rigon and White (2016), the authors use MICE. Their novel method is called multilevel multiple imputation (MLMI), and it allows systematically missing data to by imputed while still accounting for between-study heterogeneity. The authors adopt a generalized linear mixed effects model.

The analysis model is a generalized linear mixed effects model, defined in the exponential class. For study $i = 1, \ldots, N$ and participant $j = 1, \ldots, N_i$, where N_i is the number of participants in study i, it is given by

$$f_1\left(Y_{ij}|\mathbf{u_i}, \beta, \phi\right) = exp\{\phi^{-1}[Y_{ij}\zeta_{ij} - a(\zeta_{ij})] + b(Y_{ij}, \phi)\} \tag{3.8}$$

Y_{ij} is the observed outcome Y for subject j in study i
$\mathbf{x}_{ij} = (x_{ij1}, \ldots, s_{ijK})$ is the vector of $k = 1, \ldots, K$ covariates for subject j in study i. Covariates are assumed independent from each other.
β is $\mathbf{x}_{ij}$'s associated K-dimensional vector of fixed effect parameters
$\mathbf{u}_i$ is an L-dimensional vector of random effects for study i, where $L < K$ and $\mathbf{u}_i \sim MVN\left(\mathbf{0}, \mathbf{T}\right)$
$\mathbf{v}_{ij}$ is a vector of variables associated with $\mathbf{u_i}$. Usually $\mathbf{v}_{ij}$ is a subset of $\mathbf{x}_{ij}$
$\zeta_{ij} = \mathbf{x}_{ij}^\top\beta + \mathbf{v}_{ij}^\top\mathbf{u_i}$
ϕ is a scalar dispersion parameter
$a(\cdot)$ is a link function
$b(\cdot)$ together with $a(\cdot)$ defines a particular family within the exponential class, be it binomial, normal, or Poisson.

Further specification for model parameters are given for different distributions.

Normal distribution:

- $Y_{ij} \sim N\left(\mu_{ij}, \sigma^2\right)$
- $\zeta_{ij} = \mu_{ij}$
- $\phi = \sigma^2$
- $a(\zeta_{ij}) = \frac{1}{2}\mu_{ij}^2$
- $b(Y_{ij}, \phi) = \frac{Y_{ij}^2}{2\sigma^2} - \frac{1}{2}ln\left(2\pi\sigma^2\right)$
- $f_1\left(Y_{ij}|\mathbf{u_i}, \beta, \phi\right) = f_1\left(Y_{ij}|\mu_{ij}, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}}exp\left(-\frac{(Y_{ij}-\mu_{ij})^2}{2\sigma^2}\right)$

Binomial distribution:

- $Y_{ij} \sim Binom(n_{ij}, \pi_{ij})$
- $\zeta_{ij} = ln\left(\frac{\pi_{ij}}{1-\pi_{ij}}\right)$
- $\phi = 1$
- $a(\zeta_{ij}) = n_{ij}ln(1 + exp(\zeta_{ij}))$
- $b(Y_{ij}, \phi) = ln\binom{n_{ij}}{y_{ij}}$

– $f_1\left(Y_{ij}|\mathbf{u_i}, \beta, \phi\right) = f_1\left(Y_{ij}|n_{ij}, \pi_{ij}\right) = \binom{n_{ij}}{y_{ij}} \pi_{ij}^{Y_{ij}} (1-\pi_{ij})^{Y_{ij}}$

Model 3.8 is estimated using the `lme4` package for R.

The imputation model can be applied to continuous, binary, categorical, or count (ordinal) data, with a subset of systematically missing predictors. If predictor k is systematically missing for study i, then all $(x_{i1k}, \ldots, x_{iN_ik})$ are unknown. The imputation model for each such $x_{ijk} \in (x_{i1k}, \ldots, x_{iN_ik})$ is given by:

$$f_2(x_{ijk}|\mathbf{b}_{ik}, \gamma_k, \varphi_k) = exp\left\{\varphi_k^{-1}[x_{ijk}\eta_{ijk} - c(\eta_{ijk})] + d(x_{ijk}, \varphi_k)\right\} \tag{3.9}$$

$\mathbf{z}_{ijk}$ is a P-dimensional vector of covariates associated with fixed effect parameters γ_k included in the imputation model
γ_k are the fixed effect parameters included in the imputation model, typically remaining predictors $x_{ijs}, s \neq k$ that may have influenced the occurrence of missing data or variables explaining variance in candidate predictors and outcome Y_{ij}
φ_k is a scale dispersion parameter associated with covariate x_{ijk}
$\mathbf{b}_{ik}$ is a Q-dimensional vector of random effects in the imputation model, associated with vector of subject-level covariates $\mathbf{w}_{ijk}$. It is assumed that $\mathbf{b}_{ik} \sim MVN\left(\mathbf{0}, \Psi_k\right)$
$\eta_{ijk} = \mathbf{z}_{ijk}^\top \gamma_k + \mathbf{w}_{ijk}^\top \mathbf{b}_{ik}$
$c(\cdot)$ and $d(\cdot)$ are defined differently depending on the value type of x_{ijk}
Ψ_k is the between-study covariance

The authors write that the composition of γ_k should be defined so that it increases the plausibility of the MAR assumption. This means that the imputation model may contain more parameters than the analysis model. For congeniality, all predictors and outcomes from 3.8 should be included in the imputation model, and Ψ_k must be equally or less restricted than $\mathbf{T}$.

The unknown parameters from Model 3.9 are denoted $\theta_k = (\gamma_k, \psi_k, \varphi_k)$. The authors note that if we are working in a binary or count setting, $\varphi_k = 1$.

The authors only go into detail on binary and continuous missing predictor settings.

Binary missing predictors:

– The logit transformation of the probability of success π_{ijk} is chosen as $\eta_{ijk} = ln(\pi_{ijk}/(1-\pi_{ijk}))$
– $x_{ijk} \sim Bernoulli(\pi_{ijk})$
– $\pi_{ijk} = \frac{1}{1+exp(-\mathbf{z}_{ijk}^\top \gamma_k - \mathbf{w}_{ijk}^\top \mathbf{b}_{ik})}$
– $\mathbf{b}_{ik} \sim MVN(\mathbf{0}, \Psi_k)$

Continuous missing predictors:

– $x_{ijk} \sim N\left(\mathbf{z}_{ijk}^\top \gamma_k + \mathbf{w}_{ijk}^\top \mathbf{b}_{ik}, \varphi_k\right)$, so $x_{ijk} = \mathbf{z}_{ijk}^\top \gamma_k + \mathbf{w}_{ijk}^\top \mathbf{b}_{ik} + \epsilon_{ijk}$
– $\mathbf{b}_{ik} \sim MVN(\mathbf{0}, \Psi_k)$
– $\epsilon_{ijk} \sim N(0, \sigma_k^2)$

The authors note that with mixed effect models, the error variance σ_k^2 and the between-study variance Ψ_k are assumed independent.

The imputation procedure consists of three steps. In the first, Model 3.9 is fitted to the M studies where x_{ijk} is observed. Secondly, random draws of θ_k^* are generated

sequentially, using $\hat{\gamma}_k$, $\hat{\Psi}_k$, and $\hat{\sigma}_k^2$ as applicable. Lastly, θ_k^* is used to generate imputations for the systematically missing predictor x_{ijk}.

In the case of two or more covariates that are systematically missing for some studies, the imputation proceeds as follows. Suppose L covariates of $\mathbf{x}_{ij}$, $2 \leq L \leq K$ are systematically missing for some studies. With MICE, each systematically missing covariate is imputed sequentially, using the most recent imputed values of other covariates. Iterations proceed until a certain convergence limit is reached.

First, for each systematically missing predictor $x_{ijl}, l = 1, \ldots, L$, a value of θ_l is drawn. Next, the missing part of x_{ijl} is imputed from the drawn value from the corresponding values of parameters θ_l. The observed and missing parts of x_{ijl} are denoted x_{ijl}^{obs} and x_{ijl}^{mis} respectively. The tth iteration successively draws from

$$\begin{aligned}
\theta_1^{*(t)} &\sim P\left(\theta_1 | \mathbf{z}_{ij1}^{(t-1)}, x_{ij1}^{obs}\right) \\
x_{ij1}^{mis(t)} &\sim P\left(x_{ij1}^{mis} | \mathbf{z}_{ij1}^{(t-1)}, x_{ij1}^{obs}, \theta_1^{*(t)}\right) \\
&\vdots \\
\theta_L^{*(t)} &\sim P\left(\theta_L | \mathbf{z}_{ijL}^{(t-1)}, x_{ijL}^{obs}\right) \\
x_{ijL}^{mis(t)} &\sim P\left(x_{ijL}^{mis} | \mathbf{z}_{ijL}^{(t-1)}, x_{ijL}^{obs}, \theta_L^{*(t)}\right)
\end{aligned}$$

Here, $\mathbf{z}_{ijl}$ consists of predictors $x_{ijs}, x \neq l$ that were imputed in previous steps, along with outcome y_{ij} and other variables that influence the occurrence of missing data or explain some variability in covariate values. Repeating this algorithm produces a set of complete data, ie. an imputation, and this is repeated for the desired number of imputations.

Post-imputation, each complete data set is analyzed with analysis Model 3.8, producing estimates of parameters β and estimates of variance of these parameters. These estimates are then combined using Rubin's rules. (Rubin 1986)

Comparison of the methods

The three listed methods above all display some drawbacks. The data sets from our simulation display both systematic and sporadic missingness. This means that the method in Jolani et al. (2015) is not equipped to handle our data, because in the form detailed above it does not allow for sporadic missingness to be handled. The remaining two methods, M Quartagno and JR Carpenter (2015) and Resche-Rigon and White (2016) assume that all variables are continuous, which does not accommodate our categorical covariates SEX, EDU, PHYS, or our binomially distributed memory scores. However, some improvements have recently been made to each of these latter two methods, and these are available as packages in the statistical software R.

3.2 Recently Available Methods

The method of Matteo Quartagno (2016) was made available very recently (on June 6, 2017) as a package for statistical software R called `jomo`. (Matteo Quartagno and James Carpenter 2017) The fully conditional specification two stage method of Resche-Rigon and White (2016) was also made available very recently (on July 9,2017) as a package for R

called `micemd` (Audigier and Resche-Rigon 2017) Both of these methods now claim to be able to handle categorical, continuous, or binary-valued systematically missing covariates, although binomial-valued systematically missing variables (whether covariate or outcome) are again notably missing. Nonetheless, these two methods are the most applicable to the setting of harmonization of memory scores, and they are tested in the following chapters.

Chapter 4

Methods

The chapter that follows will detail my research question and hypotheses, and then go into how I plan to test the research question and why I chose these testing methods. A simulated data set will be proposed, and the procedure for generating this simulation will be detailed. I will then discuss how to apply methods identified by the literature review to the simulated data sets.

4.1 Research Question and Hypotheses

The research question we wish to investigate is how to impute systematically missing outcomes for harmonization, in the case of ordinal scales such as memory scores, either with or without overlap in outcome variables. We wish to examine whether they can be imputed at all, and what the efficacy of each approach is in the context of a simulated data set that approximates one found in real life. We hypothesize that the methods will work for both the overlap and no overlap case, although we speculate that the no overlap case will not yield very good results because none of the imputation models were developed for this case. The methods also are not meant for binomially distributed data, and so we do not expect very good results even in the overlap case. Nonetheless, these are the methods found in the literature that show the most promise in handling the type of data we are interested in.

The methods follow a few steps: first a simulation of complete data sets is performed. Missingness is added to the complete data sets, both systematic and sporadic, and for different variations of systematic missingness. Next, different imputation methods are applied to the data sets with missing values, to create complete, imputed data sets. Then an analysis model is chosen and applied to the original data set without missingness, and to each of the imputed data sets. An imputation can be said to be good if the analysis model resulting from the imputed data sets gives parameters close to those that result from applying the same model to the complete, original data sets. The parameters of the analysis model are thus our outcome of interest, and the bias and mean squared error of these parameters are used to measure closeness to the true parameters. The steps are broken down in the following sections.

4.2 Simulation

The first step was the simulation of complete data sets. We specifically research the case where there are three studies to be harmonized: 1, 2, and 3. All three studies have collected a set of covariates $\mathbf{X}$, and there are three target outcome variables: Y_1, Y_2 and Y_3. These are ordinal variables, meaning they are additive scales composed of binary items.

The case study data set was based on a real harmonization initiative from L. E. Griffith, Heuvel, et al. (2016). The three studies investigated were the Canadian Study on Health and Aging (Pasture and Onkia 1994), the Canadian Community Health Survey on Healthy Aging (Canada 2008), and the Quebec Longitudinal Study on Nutrition and Aging (Gaudreau et al. 2007). The studies collected data on participants who were aged 65 or older, and tested their cognitive ability with different neuropsychological tests. These were the 15-item Rey Auditory Verbal Learning Test, the 12-item Buschke Cued Recall Procedure, and a French version of the 16-item Free and Cued Selective Reminding Test adapted from Grober and Buschke. Confounding variables were chosen by L. E. Griffith, Heuvel, et al. (2016) when they demonstrated a relationship with cognition and physical activity. The authors chose an extensive list of confounding variables. We focused on only a few of them: age, education, and sex, along with physical activity.

In our case study, the covariates $\mathbf{X}$ are:

- *age*: a continuous variable representing the age of the participant
- *sex*: a categorical variable representing sex. A value of 1 signifies male and 2 signifies female.
- *edu*: a categorical variable representing education. A value of 1 signifies low education, 2 medium education, and 3 high education
- *phys*: a categorical variable representing physical activity. A value of 1 signifies low physical activity, 2 medium physical activity, and 3 high physical activity. Age, sex, and education influence physical activity level in a linear fashion

The outcome variables Y_1, Y_2, and Y_3 represent cognitive scales, ordinal variables with a chosen number of binary items. These are influenced by all the above covariates

- Y_1 = MEM1: an ordinal cognitive scale with 15 items, like the 15-item Rey Auditory Verbal Learning Test
- Y_2 = MEM2: an ordinal cognitive scale with 12 items, like the 12-item Buschke Cued Recall Procedure
- Y_3 = MEM3: an ordinal cognitive scale with 16 items, like the 16-item Free and Cued Selective Reminding Test adapted from Grober and Buschke

The exact specification of the simulated data sets is given in the Appendix. The most important details are that the three studies exhibited heterogeneity in several ways: the study sizes were different, with 10000, 2000, and 500 participants in Studies 1,2, and 3 respectively. The mean and standard deviation of age, sex and education were also heterogeneous across studies. Age, sex and education had a confounding effect on physical activity, although the confounding effect was the same for studies 1,2, and 3. This means that although all the covariates exhibit heterogeneity across studies in their values, the *way* that age, sex, and education influence physical activity is homogeneous across studies. The same is true for the generation of the memory scores: a continuous,

underlying latent variable was hypothesized, akin to cognitive ability. This cognitive ability was influenced by the all the covariates: age, sex, education, and physical activity. It was heterogeneous across studies because of the heterogeneity in the covariates, but again the *way* the the covariates influenced cognitive ability was the same across studies. The continuous cognitive ability was then used to generate the scores in Y_1, Y_2, and Y_3.

The intended number of simulations was 1000 data sets, but because the chosen imputation and analysis models required an extraordinarily long time to run, the number of simulations was revised down to 100. The first three lines of the generated collection of simulated data sets is given here.

Simulation	Study	Subject	Age	Sex	Edu	Phys	Y_1	Y_2	Y_3
1	1	1	62.33061093	1	3	2	11	10	10
1	1	2	75.25189419	1	2	2	9	9	7
1	1	3	68.58132458	0	3	2	11	10	10

Table 4.1: Simulated data sets for case study

4.3 Missingness

The next step was the application of both sporadic and systematic missingness to the 100 complete data sets. Two settings were explored: the case with and without overlap.

In the *overlap case*, $\mathbf{X}$ is collected by all three studies, Y_1 has been collected by Studies 1 and 2, and Y_2 has been collected by Studies 1 and 3. Y_3 is not included in this case. Because there is no variable that is collected in only one study, we say we are in the case of overlap. This case is pictured in Table 4.2, where 1 indicates that the variable was collected, and 0 indicates that it was not collected. Even if it was collected, it can still exhibit sporadic missingness.

In the *no overlap case*, $\mathbf{X}$ is still collected by all three studies, but each of the outcome variables is collected in only one study. Y_1 was collected by Study 1, Y_2 by Study 2, and Y_3 by Study 3. This case is pictured in Table 4.3.

	Measure		
Study	$\mathbf{X}$	Y_1	Y_2
1	1	1	1
2	1	1	0
3	1	0	1

Table 4.2: Systematic missingness pattern for overlap case

	Measure			
Study	$\mathbf{X}$	Y_1	Y_2	Y_3
1	1	1	0	0
2	1	0	1	0
3	1	0	0	1

Table 4.3: Systematic missingness pattern for no overlap case

Sporadically missing data was created under both the MCAR assumption and the MAR assumption. MAR missingness in the memory scores was tied to age, for which there is precedent in Samtani et al. (2015), who found that age of participant had a significant effect on missingness of the cognitive outcome in their cohort data set. Age was divided into quantiles for each study, with 20% of missingness in each memory score for the highest quantile, 15% for the next-highest, 10% for the third-highest, and 5% for the lowest (youngest). For MCAR, or non-conditional missingness, it is reasonable to assume that age, sex, education, and physical activity would experience very little missingness, since the data represents studies that were conducted in a single sitting (as opposed to longitudinal), and these variables are typically collected first. Thus a modest amount of data, 5% of each variable, was set to missing. For the cognitive scales it was possible for the value to be set to missing both under MAR and MCAR missingness generation.

The percentage of missingness for each covariate was thus 5%, and the outcome variables Y_1, Y_2, and Y_3 exhibited the following percentages of missingness for each study:

Data set	Study	Y_1	Y_2	Y_3
No Overlap	1	13.6228	100	100
	2	100	19.959	100
	3	100	100	16.622
Overlap	1	13.6228	13.6464	-
	2	20.0615	100	-
	3	100	17.192	-

Table 4.4: Percentage missing after systematic and sporadic missingness implementation

4.4 Imputation

Two methods were chosen from the literature review for application to the simulated data sets, both recently available as packages for statistical software R. These were the `jomo` joint modeling package from Matteo Quartagno and James Carpenter (2017), which was the result of the paper M Quartagno and JR Carpenter (2015). The second method chosen was the `micemd` package from Audigier and Resche-Rigon (2017), the result of the paper Resche-Rigon and White (2016). The details of both imputation models are given in the preceding chapter. They were chosen because they displayed the best potential for adaptation to the simulated data sets. Both were applicable when systematic and sporadic missingness existed, unlike the approach of Jolani et al. (2015).

Both `jomo` and `micemd` were applied to the simulated data sets with and without overlap. 100 simulated data sets were prepared, and so 100 data sets have the packages applied to them separately. A relatively low number of imputations, five, was chosen because of the lengthy time requirements to generate the imputations and to apply an analysis model to the imputed data sets.

Application of R package `jomo`

`jomo` was applied to the data sets both with overlap and without overlap. For the specification of the algorithm, only two value types are allowed: categorical or continuous.

It was recommended by the authors that binary and ordinal variables be specified as categorical for the imputation.

In the case of the data set with overlap, *age* was specified as continuous, and the remainder (*sex*, *edu*, *phys*, Y_1, and Y_2) were specified as categorical. *Study* was specified as the cluster, and the imputation proceeded without error. However, in the case of the data set without overlap, specifying *age* as continuous, and the remainder (*sex*, *edu*, *phys*, Y_1, Y_2,and Y_3) as categorical resulted in an error message. Only when Y_1, Y_2,and Y_3 were specified as continuous did the algorithm execute without error. This specification was not ideal because the authors did not intend it this way, but there is precedence for the treatment of ordinal variables as continuous, as noted in the Background chapter, and so we proceeded.

The algorithm was implemented with 5 imputations. The code used is available in the Appendix.

Application of R package `micemd`

For the overlap case, *age* was specified as continuous, and the remainder (*sex*, *edu*, *phys*, Y_1, and Y_2) were specified as categorical. The algorithm proceeded without error for both the overlap and no overlap cases. The algorithm was implemented with 5 imputations. The code used is available in the Appendix.

After five imputations on each of the 100 simulated data sets, for the overlap and no overlap cases, the next step was to explore the accuracy of the imputations and apply an analysis model to the complete data sets.

4.5 Analysis

To examine how well the imputed values of the systematically missing variables matched the true values from the complete data set, tables were filled in that calculated the number of matches in imputed vs. true for each systematically missing variable. Although the fit of the analysis model is the most important result of the imputations, it is still interesting to see how accurate the imputed variables are. Next, an analysis model was chosen and applied to each imputed data set in each simulation.

The chosen analysis model must reflect the way the data were generated. There were no quadratic or higher power terms used to generate any of the data, so a natural choice is a linear form. The general class of models chosen was thus a linear regression. The analysis model must also exhibit a multilevel structure, with participants nested within study. The outcome variables, Y_1, Y_2, and Y_3, are scales composed of binary items, with each binary item assumed to exhibit the same difficulty for the same scale. This means that they exhibit a binomial distribution. The type of model which can accommodate all of these requirements is known as a generalized linear mixed model.

We suppose that there exists some latent variable or *ability* Z_{hij}, which depends on participant i's characteristics described by the covariates, the difficulty of the binary items composing scale Y_h, and on study j. This latent variable is continuous, and determines how difficult it is for the participant to attain success in each binary item that composes the memory scales. Because this latent variable underlies the outcome of the participant for each Y_h, we can describe the conditional distribution for Y_{hij} given Z_{hij}. Suppose the probability of success in a binary item of scale Y_h for participant j in study i is $\pi(z)$. Y_1 has 15 items, and so 15 possible successes, Y_2 has 12, and Y_3 has 16. Let the number of

items in scale Y_h be given by M_h. The most appropriate distribution for the behavior of these memory scales is the binomial distribution, the probability mass function of which is given by

$$P(Y_h = \ell | Z_{hij} = z) = \binom{M_h}{\ell} \pi(z)^{\ell} (1 - \pi(z))^{M_h - \ell} \tag{4.1}$$

We still need to investigate how $\pi(z)$ is defined. $\pi(z)$ is different for each memory scale h, and for each participant i in study j, because it depends on the continuous value $Z_{hij} = z$, the participant's ability to correctly answer a binary item in the memory scale h. Thus if $Z_{hij} = z$, $\pi(z)$ can be given by the logit link function, commonly used in generalized mixed models for binomially distributed variables.

$$\pi(z) = \frac{e^z}{1 + e^z}$$

The regression model is then linear in Z_{hij}, and is given by

$$Z_{hij} = \beta_{h,0} + \delta_{hj} + \beta_{h,s} sex_{ij} + \beta_{h,a} age_{ij} + \beta_{h,e_2} edu_{2,ij} + \tag{4.2}$$

$$\beta_{h,e_3} edu_{3,ij} + \beta_{h,p_2} phys_{2,ij} + \beta_{h,p_3} phys_{3,ij}, \tag{4.3}$$

with $\beta_{h,0}$ the intercept for memory test h, for participants with low education and low physical activity, $\beta_{h,s}$ the effect of sex $\beta_{h,a}$ the effect of age, β_{h,e_2} the effect of medium education, β_{h,e_3} the effect of high education, β_{h,p_2} the effect of medium physical activity, and β_{h,p_3} the effect of high physical activity. sex_{ij} is an indicator for sex of participant i in study j, age_{ij} is the age of participant i in study j, $edu_{2,ij}$ is an indicator variable for medium education of participant i in study j, $edu_{3,ij}$ is an indicator variable for high education of participant i in study j, $phys_{2,ij}$ is an indicator variable for medium physical activity of of participant i in study j, and $phys_{3,ij}$ is an indicator for high physical activity of participant i in study j. $\delta_{hj} \sim N(0, \sigma_h^2)$ is a random effect for study. In this model, the parameters $\beta_{h,0}, \beta_{h,s}, \beta_{h,a}, \beta_{h,e_2}, \beta_{h,e_3}, \beta_{h,p_2}, \beta_{h,p_3}$, and σ_h^2 are independent of study, and thus do not need to be reported on the study level. The variance σ_h^2 is important because it describes heterogeneity between studies.

The analysis model was applied to the original complete data set for 100 simulations, for the each memory score, resulting in 100 estimates for each parameter of the model. Next, the analysis model was applied to each of the imputed data sets separately, after which the parameters were combined over the imputations using Rubin's Rules. In this case Rubin's Rules only require taking the average. This resulted in 100 estimates for each parameter of the model, one for each simulation, for each model in the overlap and no overlap case.

The analysis model was estimated using maximum likelihood estimation. The output for each simulated data set, for each imputation, was an estimate of the parameters $\hat{\beta}_{h,0}, \hat{\beta}_{h,s}, \hat{\beta}_{h,a}, \hat{\beta}_{h,e_2}, \hat{\beta}_{h,e_3}, \hat{\beta}_{h,p_2}, \hat{\beta}_{h,p_3}$, and $\hat{\sigma}_h^2$, and the estimates of their standard errors $\hat{\tau}_{h,0}, \hat{\tau}_{h,s}, \hat{\tau}_{h,a}, \hat{\tau}_{h,e_2}, \hat{\tau}_{h,e_3}, \hat{\tau}_{h,p_2}, \hat{\tau}_{h,p_3}$, and $\hat{\eta}_h^2$.

The results were collected in two phases. First, the mean squared error (MSE) and bias were used to compare the accuracy of the imputation methods in their estimation of fixed effects. A smaller MSE or bias is considered better. Suppose the parameter to be estimated is β. Fitting the model to the complete data sets results in a vector of 100 parameter estimates $\beta_e, e = 1, \ldots, 100$, with one estimate for each simulation.

The analysis model is applied to each of the imputed data sets for each simulation, resulting in five estimates for each parameter, for each simulation. Suppose the simulations

are indexed by $e = 1, \ldots, 100$ and the imputations are indexed by $\ell = 1, \ldots 5$. Under Rubin's rules,

$$\hat{\beta}_e = \frac{1}{5} \sum_{\ell=1}^{5} \hat{\beta}_{e\ell}$$

The mean squared error is then given by

$$\text{MSE} = \frac{1}{100}(\hat{\beta}_e - \beta_e)^2$$

And the bias is given by

$$\text{Bias} = \frac{1}{100}(\hat{\beta}_e - \beta_e)$$

In the second phase, the MSE and bias of the random effects was calculated for methods that offered promising results in the estimates of fixed effects. A good imputation model must not only estimate fixed effects accurately, but also between-study heterogeneity. This quantity is represented by the estimate $\hat{\sigma}_h^2$, and if the estimated value was close (in terms of bias and MSE) to the true σ_h^2 estimated in the complete data set, then the method was considered good.

Chapter 5

Results

The results are given in the tables that follow. First Tables 5.1 shows the average number of matches vs. non-matches between imputed systematically missing variable values and the true values in the complete data set. Next, the fixed effect estimates for the complete and imputed data sets, averaged over imputations, are given in Tables 5.2, 5.3, and 5.4, recalling that the third memory score Y_3 was not part of the imputation for the data sets with overlap. Last, the random effects for all outcomes were calculated for the methods that achieved good results in the estimation of fixed effects. These are given in Table 5.5

		Y_1		Y_2		Y_3	
Data Set	Study	$=$	$\neq$	$=$	$\neq$	$=$	$\neq$
jomo	1	-	-	1469.714	8530.286	1218.31	8781.69
Without	2	210.084	1789.916	-	-	208.146	1791.854
Overlap	3	58.022	441.978	69.65	430.35	-	-
micemd	1	-	-	747.848	9252.152	614.812	9385.188
Without	2	182.946	1817.054	-	-	161.768	1838.232
Overlap	3	46.862	453.138	58.584	441.416	-	-
jomo With	2	-	-	162.018	1837.982	-	-
Overlap	3	37.194	462.806	-	-	-	-
micemd With	2	-	-	255.238	1744.762	-	-
Overlap	3	55.11	444.89	-	-	-	-

Table 5.1: Average exact matches between imputed systematically missing variable values and the true values in the complete data set, over 5 imputations and 100 simulations. $=$ means there was a match, $\neq$ means there was no match.

Data Set	Value	$\hat{\beta}_{h,e_2}$	$\hat{\beta}_{h,e_3}$	$\hat{\beta}_{h,s}$	$\hat{\beta}_{h,a}$	$\hat{\beta}_{h,p_2}$	$\hat{\beta}_{h,p_3}$
Complete	Average	0.0882165	0.2238988	0.1773187	-0.0241964	0.2332562	0.4635988
jomo Without Overlap	Average	0.059384	0.168232	0.1442054	-0.0230859	0.1944467	0.3581061
	Bias	-0.0288325	-0.0556667	-0.0331133	0.0011105	-0.0388095	-0.1054927
	MSE	0.0009624	0.0033062	0.0012126	0.0000029	0.0016875	0.0114134
micemd Without Overlap	Average	0.0837759	0.2209401	0.1749225	-0.02538	0.2344563	0.4666026
	Bias	-0.0044406	-0.0029587	-0.0023962	-0.0011836	0.0012001	0.0030038
	MSE	0.0002743	0.000333	0.0001568	0.0000034	0.0002525	0.0003574
jomo With Overlap	Average	0.070794	0.1766229	0.1509173	-0.0203375	0.193609	0.3719336
	Bias	-0.0174225	-0.0472758	-0.0264013	0.0038589	-0.0396472	-0.0916652
	MSE	0.0004302	0.0023781	0.000775	0.0000155	0.001706	0.0086017
micemd With Overlap	Average	0.0887257	0.2248799	0.1768725	-0.0244197	0.234458	0.4664647
	Bias	0.0005092	0.0009811	-0.0004462	-0.0002233	0.0012018	0.0028659
	MSE	0.000095	0.000112	0.0000608	0.0000005	0.0001123	0.0001343

Table 5.2: True and estimated fixed effects for outcome Y_1

Data Set	Value	$\hat{\beta}_{h,e_2}$	$\hat{\beta}_{h,e_3}$	$\hat{\beta}_{h,s}$	$\hat{\beta}_{h,a}$	$\hat{\beta}_{h,p_2}$	$\hat{\beta}_{h,p_3}$
Complete	Average	0.0896985	0.2257904	0.1773158	-0.0240823	0.2326406	0.4604072
jomo	Average	0.0286537	0.0347203	0.0441435	-0.0083294	0.0379362	0.0819322
Without	Bias	-0.0610449	-0.19107	-0.1331723	0.015753	-0.1947044	-0.378475
Overlap	MSE	0.0041431	0.0369767	0.01807	0.0002546	0.0383751	0.1437202
micemd	Average	0.1197389	0.3130112	0.2559318	-0.0353796	0.3448196	0.7012372
Without	Bias	0.0300404	0.0872208	0.078616	-0.0112973	0.1121791	0.24083
Overlap	MSE	0.0145728	0.0309336	0.0213043	0.0001735	0.0477521	0.085125
jomo	Average	0.055387	0.1606089	0.1387985	-0.0168724	0.1861662	0.3348661
With	Bias	-0.0343115	-0.0651814	-0.0385173	0.0072099	-0.0464744	-0.1255411
Overlap	MSE	0.0013838	0.0044467	0.0015864	0.0000529	0.0023053	0.0159799
micemd	Average	0.0890109	0.2274399	0.1770022	-0.0248983	0.2356981	0.4653371
With	Bias	-0.0006876	0.0016496	-0.0003136	-0.0008159	0.0030575	0.0049299
Overlap	MSE	0.0002072	0.0001889	0.0001032	0.0000019	0.0001354	0.0001493

Table 5.3: True and estimated fixed effects for outcome Y_2

Data Set	Value	$\hat{\beta}_{h,e_2}$	$\hat{\beta}_{h,e_3}$	$\hat{\beta}_{h,s}$	$\hat{\beta}_{h,a}$	$\hat{\beta}_{h,p_2}$	$\hat{\beta}_{h,p_3}$
Complete	Average	0.0899386	0.2239162	0.1771599	-0.0240945	0.2334964	0.4637751
jomo	Average	0.0024816	0.0097451	0.0152178	-0.0007479	0.0172905	0.0277062
Without	Bias	-0.087457	-0.2141711	-0.1619421	0.0233466	-0.2162059	-0.4360689
Overlap	MSE	0.0080867	0.0462598	0.0265088	0.0005467	0.0471469	0.1905667
micemd	Average	0.0754709	0.2484958	0.2664376	-0.0207232	0.2813071	0.5645516
Without	Bias	-0.0144676	0.0245796	0.0892776	0.0033713	0.0478108	0.1007765
Overlap	MSE	0.0284756	0.0405116	0.0439043	0.0001265	0.0304802	0.0570043

Table 5.4: True and estimated fixed effects for outcome Y_3

Data Set	Value	$\hat{\sigma}_1^2$	$\hat{\sigma}_2^2$
Complete	Average	9.091224×10^{-5}	1.365312×10^{-4}
jomo With Overlap	Average	0.02896762	0.02439645
	Bias	0.02887671	0.02425991
	MSE	0.0009609373	0.000627878
micemd With Overlap	Average	1.523204×10^{-4}	1.809084×10^{-4}
	Bias	6.140818×10^{-5}	4.437717×10^{-5}
	MSE	9.237626×10^{-8}	1.631998×10^{-7}

Table 5.5: True and estimated random effect for outcomes Y_1 and Y_2

Chapter 6

Discussion

Over the last few chapters we have explored the topic of data harmonization of binomially distributed measures in the context of IPD-MA. The very recentness of the best available methods (`jomo` and `micemd` were published as R packages in June and July of 2017 respectively) indicate that this a topic of current interest and research. In the context of binomial outcomes, researchers who are performing a harmonization may currently be unable to perform their desired analyses because the performance of such a harmonization has no precedent. The results of the last chapter indicate that it can be done in certain settings with low bias and mean squared error, thus approximating well a complete data set.

6.1 Analysis of Results

The first table, 5.1, does not paint an overly promising picture: the power of the imputation methods to produce an exact match is relatively low. In this regard, `jomo` outperforms `micemd` by producing more matches on average in the no overlap case, while `micemd` outperforms `jomo` in the case with overlap. However, the purpose of multiple imputation is not to reproduce the complete data set, but to produce many complete data sets that also approximate the variability naturally present in real data. As we can see in the tables that follow, some of the imputation methods perform quite well. Positive and negative effects were, at least, always estimated on the correct side of 0.

For the fixed effects in the no overlap case, neither `micemd` and `jomo` performed very well in terms of bias. (Seen in Tables 5.2, 5.3, and 5.4. Interestingly, the results were better for memory score 1 than for memory score 2 or 3, even though memory score 1 had the largest number of missing data points, because it was systematically missing in the largest study: Study 1. It is not overly surprising that the no overlap case did not produce very good results. It is difficult to explore between-study heterogeneity when a measure has only been collected for one study. The background chapter did not find any particularly promising methods developed for the no overlap case, and these were no different.

However in the fixed effects for the overlap case, for which `micemd` and `jomo` were designed, we do see some promising results! `jomo` with overlap outperforms the no overlap case, but still pales in comparison to `micemd` with overlap, which estimates parameters that are very close to the real ones - the bias and MSE are generally very low, meaning that the estimated model after imputation and the model applied to the complete data set are very similar. For memory scores Y_1 and Y_2, `micemd` with overlap is essentially

accurate to two significant digits.

The methods without overlap were singled out for investigation of their random effects. This step verified whether they were able to find between-study heterogeneity if it was present. As we can see in Table 5.5, the average value of σ_h^2 for outcomes Y_1 and Y_2 is very small, near zero. This indicates that the way that study affects memory is approximately equal across studies, and because of the parameters we chose for the model, this should indeed be the case. Although the covariates were generated with different means and standard deviations across studies, the way that the covariates influenced outcome was the same across studies. In the table we see that both methods produced estimates $\hat{\sigma}_h^2$ that were close to zero, but the `micemd` estimates were closer by two decimal places.

Overall, the `micemd` performed extremely well for the data set chosen to test our research question: in the case where a researcher wishes to harmonize binomially distributed scales such as memory scores, and where there is overlap in the outcome variables. Where there is no overlap in the outcome variables, we still have not found a method that works well.

There are several decisions which were taken that could affect the generalizability of these results to other data sets. The number of simulations was lower than the standard of about 1000 simulations. The experiment should be performed again with a faster computer or much more time (approximately one month of computation). The simulation itself is only one example and does not necessarily extend to all scenarios, however efforts were made to make it as realistic as possible. The method offered by Jolani et al. (2015) could also be of scientific interest if it can be tested for the binomial setting, which we did not manage to do. The best results were also found for imputation of the smallest data sets: in the overlap case, the largest data set of 10000 participants is the one which has no sporadic missingness, and thus it is possible that the smaller data sets of 2000 and 500 participants were more easily imputed. A possible topic for future research is examining the role of study size in this simulation.

6.2 Further Research

Two avenues present themselves for further investigation: an adaptation of the method of Jolani et al. (2015) to our setting, and potential for application to a federated setting.

Adaptation of Jolani et al. (2015) Method

The third promising method identified by the literature review came from Jolani et al. (2015). This was the only paper which explored systematic missingness in binary variables, which displayed potential for adaptation to a binomial setting.

The first limitation of the method is that it is only applicable to data that does not exhibit sporadic missingness. However, sporadic missingness is easily handled by regular multiple imputation. Thus a preliminary step can be performed, imputing each data set separately to provide m data sets for each study, with all sporadically missing values filled in. The question of how to combine the m imputed data sets for each study into into a larger data set that has clustering variable *study* could be performed in multiple ways. Perhaps the best way would be to explore all combinations of imputed data sets, but this would create a problematic proliferation of data sets. Relying on the randomness created in each imputation, it does not seem so bad to combine the first imputation of each of

the three studies, and the second of each of the three studies, and so on, to create m multilevel data sets, each with three studies, that display systematic missingness.

The relevant analysis model from the paper is still a binomial distribution as in Equation 4.1. The important parameter to estimate is again

$$\pi(z) = \frac{e^z}{1+e^z},$$

which is dependent upon $Z_{hij} = z$. In the notation of Jolani et al. (2015), the relation to the covariates is given by the logit link function, with the equation

$$Z_{hij} = \mathbf{x}_{ij}^\top \beta + \mathbf{v}_{ij}^\top \mathbf{u}_j = ln\left(\frac{\pi(z)}{1-\pi(z)}\right)$$

- $\mathbf{x}_{ij}$ is a vector of K covariates predicting y_{hij}
- β is a K-dimensional vector of fixed effects
- $\mathbf{v}_{ij}$ is an L-dimensional vector of random variables associated with $\mathbf{u}_j$
- $\mathbf{u}_j$ is an L-dimensional vector of random effects for the jth study. It is assumed that $\mathbf{u}_i \sim MVN(\mathbf{0}, \mathbf{T})$

The imputation model of Jolani assumes that a subset of covariates are systematically missing. In our case, a subset of outcomes are systematically missing. The following is adapted from Jolani's suggestions for a binary systematically missing covariate. In this case we impute a binomially distributed systematically missing outcome. Suppose outcome Y_h is missing in study j. Let N_j be the number of participants in study j. Then values $\{y_{h1j}, \ldots, y_{hN_jj}\}$ are unknown. Each y_{hij} is assumed to follow a Binomial distribution with parameters M_h and π_{hij}. We impute each y_{hij} with the following imputation model

$$f(y_{hij}|M_h, \pi_{hij}) = \binom{M_h}{y_{hij}} \pi_{hij}^{y_{hij}} (1-\pi_{hij})^{M_h - y_{hij}} \tag{6.1}$$

The relation to the covariates is given again by a logit link function, such that

$$\eta_{hij} = \mathbf{z}_{hij}^\top \gamma_h + \mathbf{w}_{hij}^\top \mathbf{b}_{jk} = ln\left(\frac{\pi_{hij}}{1-\pi_{hij}}\right) \Rightarrow \pi_{hij} = \frac{1}{1+exp(-\mathbf{z}_{hij}^\top \gamma_h - \mathbf{w}_{hij}^\top \mathbf{b}_{hj})}$$

- $\mathbf{z}_{hij}$ is a vector of P covariates predicting y_{hij}.

$$\mathbf{z}_{hij} = (age_{ij}, edu_{2,ij}, edu_{3,ij}, phys_{2,ij}, phys_{3,ij}, sex_{ij})^\top$$

- γ_h is a P-dimensional vector of fixed effects
- $\mathbf{w}_{hij}$ is a Q-dimensional vector of random variables associated with $\mathbf{u}_j$
- $\mathbf{b}_{hj}$ is a Q-dimensional vector of random effects for the jth study. It is assumed that $\mathbf{b}_{hj} \sim MVN(\mathbf{0}, \Psi_h)$

The unknowns here are γ_h and Ψ_h. These must be estimated using an imputation method, and could follow the same iterative method given by Jolani et al. (2015) and detailed in the literature review section. This method shows good potential for adaptation to our setting and is a promising avenue for future research.

Federation

Another avenue to explore would be the federated setting. In fact the best method for the simulated data sets, `micemd`, is a two stage approach which makes it particularly adaptable to the federated setup, where pooling of each the data from each study is not possible in a single location. Overall there are many ways in which it would be interesting to expand the current research.

6.3 Conclusions

The aims of this thesis, to investigate harmonization of binomially distributed variables, were met. The simulation used to test different methods yielded one standout technique: the `micemd` approach developed by M Quartagno and JR Carpenter (2015). Indeed, it showed very little bias for a method that was not developed for the binomial setting. Thus I would recommend that a researcher performing an IPD-MA in the overlap setting, who wishes to harmonize different binomial variables collected on the same construct, use `micemd` to impute the missing data and proceed with analysis on the imputed data sets as usual.

Appendix A

A.1 Wishart Distribution

According to *Wishart Distribution* (n.d.), the Wishart distribution is a "generalization of a univariate chi-square distribution, to two or more variables." The distribution is for symmetric, positive semidefinite matrices, which have chi-square random variables on the diagonal of the matrix. In our context, these matrices are understood as covariate matrices.

The "chi-square distribution can be constructed by summing the squares of independent, identically distributed (i.i.d.), zero-mean, univariate normal random variables." (*Wishart Distribution* n.d.) Analogously, the Wishart distribution can be constructed by summing the inner products of i.i.d., zero-mean, multivariate normal random vectors. This property makes the Wishart distribution ideal for modeling the distribution of a sample covariate matrix of multivariate normal random data, after first scaling for the sample size. (*Wishart Distribution* n.d.)

The Wishart distribution has two parameters:

- Σ is a symmetric, positive semidefinite matrix
- v is a positive scalar, indicating the degrees of freedom. This is analogous to the chi-square distribution's degrees of freedom parameter

A random $d \times d$ matrix W has a d-dimensional Wishart distribution with parameter Σ, and n degrees of freedom if

$$W \overset{D}{=} \sum_{i=1}^{n} X_v X_v^T, \quad X_v \sim N_d(0, \Sigma)$$

We write $W \sim \mathcal{W}_d(n, \Sigma)$. Note that $\mathcal{W}_1(n, \sigma^2) = \sigma^2 \chi^2(n)$.
For $W \sim \mathcal{W}_d(n, \Sigma)$, the mean $\mathbb{E}(W)$ is equal to $n\Sigma$.
The probability density function of a d-dimensional Wishart distribution is

$$f(X, T, v) = \frac{|X|^{\frac{v-d-1}{2}} e^{-\frac{1}{2}\operatorname{trace}\left(\Sigma^{-1}X\right)}}{2^{\frac{vd}{2}} \pi^{\frac{d(d-1)}{4}} |\Sigma|^{\frac{v}{2}} \Gamma(v/2) \cdots \Gamma((v-(d-1))/2)} \tag{A.1}$$

where X and Σ are $d \times d$ symmetric positive definite matrices, and v is a scalar such that $v \geq d-1$. The Wishart distribution also exists for singular Σ, in a different form.

A.2 Inverse Wishart Distribution

According to *Inverse Wishart Distribution* (n.d.), the inverse Wishart distribution is based on the Wishart distribution. It is used specifically as the conjugate prior, for the covariance matrix of a multivariate normal distribution.

The probability density function of a d-dimensional inverse Wishart distribution is

$$f\left(X, \Sigma, v\right) = \frac{|T|^{\frac{v}{2}} e^{-\frac{1}{2}\text{trace}\left(TX^{-1}\right)}}{2^{\frac{vd}{2}} \pi^{\frac{d(d-1)}{4}} |X|^{\frac{v+d+1}{2}} \Gamma\left(v/2\right) \cdots \Gamma\left(\left(v-\left(d-1\right)\right)/2\right)} \tag{A.2}$$

where X and T are $d \times d$ symmetric positive definite matrices, and v is a scalar such that $v \geq d$. The inverse Wishart distribution also exists for singular T, in a different form.

For a random matrix with a Wishart distribution, and parameters T^{-1} and v, the inverse of that random matrix is said to have an inverse Wishart distribution, with parameters T and v. The mean is given by

$$\frac{1}{v-d-1}T$$

A.3 Data Set Generation

Many variables need to be defined by the user to enter into the data generation code:

- N1, N2, N3 are integers representing the number of participants in studies $1, 2, 3$. For this simulation, we chose $N1 = 10000, N2 = 2000, N3 = 500$. These sizes are very different from each other, as often occurs in real meta-analyses
- SIM is an integer representing the number of simulations desired by the user, the number of complete data sets to be created. For this simulation, we chose SIM = 100
- M_AGEi is the mean age for study $i = 1, 2, 3$. For this simulation, we chose M_AGE1 = 70, M_AGE2 = 80, M_AGE3 = 75
- S_AGEi is the standard deviation of age for study $i = 1, 2, 3$. For this simulation, we chose S_AGE1 = 6, S_AGE2 = 7, S_AGE3 = 4
- P_SEXi is the probability of being male for study $i = 1, 2, 3$. For this simulation, we chose P_SEX1 = 0.40, P_SEX2 = 0.35, P_SEX3 = 0.45
- P_EDULi is the probability of having low education level for study $i = 1, 2, 3$. For this simulation, we chose P_EDUL1 = 0.30, P_EDUL2 = 0.50, P_EDUL3 = 0.15
- P_EDUMi is the threshold for having a medium education level for study $i = 1, 2, 3$. The probability of having a medium education is given by P_EDUMi - P_EDULi. For this simulation, we chose P_EDUM1 = 0.60, P_EDUM2 = 0.85, P_EDUM3 = 0.55. Note that the probability of having a high education level for this three-level score is defined by 1 - P_EDUMi
- INTji is a real number representing the jth intercept for study $i = 1, 2, 3$, which is used in a regression line to describe the continuous latent variable underlying physical activity. $j = 1, 2$, because there are two intercepts for physical activity, defining two regression lines for two levels. The third level of physical activity is defined in terms of the other two levels. For this simulation, we chose INT1i = 1.5, and INT2i = 3.5, for $i = 1, 2, 3$
- A_AGEi is the effect of AGE on the average of the continuous latent variable underlying physical activity, for study $i = 1, 2, 3$. For this simulation, we chose A_AGEi = −0.03 for $i = 1, 2, 3$

- A_SEXi is the effect of sex on average physical activity, for study $i = 1, 2, 3$. For this simulation, we chose A_SEX$i = -0.50$ for $i = 1, 2, 3$
- A_EDULi is the effect of low education on the average of the continuous latent variable underlying physical activity, for study $i = 1, 2, 3$. For this simulation, we chose A_EDUL$i = -0.60$ for $i = 1, 2, 3$
- A_EDUMi is the effect of medium education on the average of the continuous latent variable underlying physical activity, for study $i = 1, 2, 3$. For this simulation, we chose A_EDUM$i = -0.30$ for $i = 1, 2, 3$
- Mk is the number of binary items tested in cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose M1 = 15, M2 = 12, M3 = 16
- B_0k is the intercept for the average of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose B_0$k = 2.5$, $k = 1, 2, 3$
- B_AGEk is the effect of age on the average of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose B_AGE$k = -0.025$, $k = 1, 2, 3$
- B_SEXk is the effect of sex on the average of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose B_SEX$k = 0.20$, $k = 1, 2, 3$
- B_EDULk is the effect of low education on the average of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose B_EDUL$k = -0.25$, $k = 1, 2, 3$
- B_EDUMk is the effect of medium education on the average of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose B_EDUM$k = -0.15$, $k = 1, 2, 3$
- B_PHYSLk is the effect of low physical activity on the average of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose B_PHYSL1 $= -0.50$, $k = 1, 2, 3$
- B_PHYSMk is the effect of medium physical activity on the average of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose B_PHYSM$k = -0.25$, $k = 1, 2, 3$
- L_0k is the intercept for the standard deviation of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose L_0$k = -2.00$, $k = 1, 2, 3$
- L_AGEk is the effect of age on the standard deviation of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose L_AGE$k = 0.02$, $k = 1, 2, 3$
- L_SEXk is the effect of sex on the standard deviation of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose L_SEX$k = 0.15$, $k = 1, 2, 3$
- L_EDULk is the effect of low education on the standard deviation of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose L_EDUL$k = -0.15$, $k = 1, 2, 3$
- L_EDUMk is the effect of medium education on the standard deviation of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose L_EDUM$k = -0.05$, $k = 1, 2, 3$
- L_PHYSLk is the effect of low physical activity on the standard deviation of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this

simulation, we chose L_PHYSL$k = 0$, $k = 1, 2, 3$

- L_PHYSMk is the effect of medium physical activity on the standard deviation of the continuous latent variable underlying cognitive scale MEMk, $k = 1, 2, 3$. For this simulation, we chose L_PHYSM$k = 0$, $k = 1, 2, 3$

The populations in the three studies are heterogeneous, displaying differences in mean and standard deviation of age, sex, education. Because the covariates A_AGEi, A_SEXi, A_EDULi, and A_EDUMi are non-zero, age, sex, and education have a confounding effect on the continuous latent variable for physical activity, CMEM. These confounding effects are the same for studies $i = 1, 2, 3$, which means that the *way* that age, sex, and education affect physical activity is the same across studies. However, because of the differences in mean and standard deviation of age, sex, and education across studies, the mean and standard deviation of physical activity (in CMEM) is heterogeneous across studies as well.

The effects of age, sex, education, and physical activity on the average and standard deviation of the continuous latent variable underlying memory, CMEMk, for $k = 1, 2, 3$, are the same for each memory score, indicated by the choices of B_0k, B_AGEk, B_SEXk, B_EDULk, B_EDUMk, B_PHYSLk, B_PHYSMk, L_0k, L_AGEk, L_SEXk, L_EDULk, L_EDUMk, L_PHYSLk, L_PHYSMk as equal for $k = 1, 2, 3$. L_PHYSLk and L_PHYSMk were set to 0, meaning that physical activity had no effect on the standard deviation of CMEMk.

These user-defined variables were then used to generate the covariates and outcomes for the simulated data sets.

- N = N1 + N2 + N3
- STUDY is a vector of length N, where for the first N1 particpants, STUDY = 1, for the next N2 participants, STUDY = 2, and for the last N3 participants, STUDY = 3.
- AGE is defined by M_AGEi + S_AGEi× Z_AGE, where M_AGEi and S_AGEi were defined by the user and Z_AGE is a random number generated by the standard normal distribution. $i = 1, 2, 3$ depending on which study the participant belongs to
- SEX is equal to 1 if U_SEX, a random number generated by the standard uniform distribution, is greater than the user-defined probability P_SEXi, the probability of being male. $i = 1, 2, 3$ depending on which study the participant belongs to
- EDU is *low* if U_EDU, a random number generated by the standard uniform distribution, is less than or equal to the user-defined probability P_EDULi. EDU is *medium* if P_EDULi < U_EDU ≤ P_EDUMi, and otherwise EDU is *high*
- PHYS is generated by an underlying latent variable, a continuous physical activity CPHYS. CPHYS has mean MU_PHYS, which is given by

$$\begin{aligned}\text{MU\_PHYS} = &(\text{A\_AGE}i \times \text{AGE}) + (\text{A\_SEX}i \times \text{SEX}) + (\text{A\_EDUL}i \text{ if EDU } = low) \\ &+ (\text{A\_EDUM}i \text{ if EDU } = medium)\end{aligned}$$

  $i = 1, 2, 3$ depending on which study the participant belongs to. CPHYS is then generated by taking the logistic quantile of U_PHY and subtracting MU_PHYS. The logistic quantile of U_PHY draws from a continuous distribution with mean 0, and subtracting the mean MU_PHYS adjusts the mean to the desired level. PHYS is then defined as *low* if CPHYS ≤ INT1i. PHYS is *medium* if INT1i < CPHYS ≤ INT2i. PHYS is otherwise *high*

- MEMk is generated in a similar way to PHYS, with an underlying latent variable, a continuous memory score for each k, called CMEMk. First the mean MU_MEMk is defined, given by

$$\begin{aligned}\text{MU\_MEM}k = {} & \text{B\_0}k + (\text{B\_AGE}k \times \text{AGE}) + (\text{B\_SEX}k \times \text{SEX}) \\ & + (\text{B\_EDUL}k \text{ if EDU } = low) + (\text{B\_EDUM}k \text{ if EDU } = medium) \\ & + (\text{B\_PHYSL}k \text{ if PHYS } = low) + (\text{B\_PHYSM}k \text{ if PHYS } = medium)\end{aligned}$$

  Next, the standard deviation of MEMk is defined by

$$\begin{aligned}\text{S\_MEM}k = {} & \text{L\_0}k + (\text{L\_AGE}k \times \text{AGE}) + (\text{L\_SEX}k \times \text{SEX}) \\ & + (\text{L\_EDUL}k \text{ if EDU } = low) + (\text{L\_EDUM}k \text{ if EDU } = medium) \\ & + (\text{L\_PHYSL}k \text{ if PHYS } = low) + (\text{L\_PHYSM}k \text{ if PHYS } = medium)\end{aligned}$$

  Then CMEMk is calculated using Z_MEM, a vector of length N×SIM, where each entry is a random number generated by the standard normal distribution

$$\text{CMEM}k = \text{MU\_MEM}k + \text{S\_MEM}k \times \text{Z\_MEM}$$

  The last intermediate variable needed to create MEMk is P_MEMk, which is the probability of success in one of the binary items that compose summary score MEMk. Each binary item is assumed to have the same probability of success.

$$\text{P\_MEM}k = \frac{e^{\text{CMEM}k}}{1 + e^{\text{CMEM}k}}$$

  Finally, MEMk is defined by the binomial distribution, with probability of success P_MEMk and number of trials given by the user-defined Mk. Thus MEMk is an integer, such that $\text{MEM}k \sim \text{Binomial}(\text{M}k, \text{P\_MEM}k)$

The output of these calculations is a collection of simulated data sets, with the number of data sets determined by the user-chosen integer SIM.

```
library(random)
library(simglm)
library(jomo)

# ------------------------------------------------------------------------
# SIMULATION OF DATA SET WITH THREE COVARIATES, THREE STUDIES
# ------------------------------------------------------------------------

# Initialize variables used to generate the data frame

# Seed values
SEED1 <- as.integer(0)
SEED2 <- as.integer(0)
SEED3 <- as.integer(0)
SEED4 <- as.integer(0)
SEED5 <- as.integer(0)
SEED6 <- as.integer(0)
SEED7 <- as.integer(0)
```

```
SEED8 <- as.integer(0)

# Number of simulations
SIM <- as.integer(0)

# Study-specific variables ----------------

# Ni is the size of study i
N1 <- as.integer(0)
N2 <- as.integer(0)
N3 <- as.integer(0)

# M_AGEi is the mean age for study i
M_AGE1 <- as.numeric(0)
M_AGE2 <- as.numeric(0)
M_AGE3 <- as.numeric(0)

# S_AGEi is the standard deviation of age for study i
S_AGE1 <- as.numeric(0)
S_AGE2 <- as.numeric(0)
S_AGE3 <- as.numeric(0)

# P_SEXi is the probability of being male for study i
P_SEX1 <- as.numeric(0)
P_SEX2 <- as.numeric(0)
P_SEX3 <- as.numeric(0)

# P_EDULi is the probability of having a low level of education
   in study i
P_EDUL1 <- as.numeric(0)
P_EDUL2 <- as.numeric(0)
P_EDUL3 <- as.numeric(0)

# P_EDUMi is the probability of having a medium level of
   education in study i
P_EDUM1 <- as.numeric(0)
P_EDUM2 <- as.numeric(0)
P_EDUM3 <- as.numeric(0)

# Regression variables for physical activity
   ------------------------
# INTij is the ith intercept for study j
# Two intercepts for each study, because there are three levels
   of physical activity
INT11 <- as.numeric(0)
INT12 <- as.numeric(0)
INT13 <- as.numeric(0)
```

```
INT21 <- as.numeric(0)
INT22 <- as.numeric(0)
INT23 <- as.numeric(0)

# A_AGEi is the effect of age in study i on average physical
    activity
A_AGE1 <- as.numeric(0)
A_AGE2 <- as.numeric(0)
A_AGE3 <- as.numeric(0)

# A_SEXi is the effect of sex in study i on average physical
    activity
A_SEX1 <- as.numeric(0)
A_SEX2 <- as.numeric(0)
A_SEX3 <- as.numeric(0)

# A_EDULi  is the effect of low education in study i on average
    physical activity
A_EDUL1 <- as.numeric(0)
A_EDUL2 <- as.numeric(0)
A_EDUL3 <- as.numeric(0)

# A_EDUMi  is the effect of medium education in study i on
    average physical activity
A_EDUM1 <- as.numeric(0)
A_EDUM2 <- as.numeric(0)
A_EDUM3 <- as.numeric(0)

# Test-specific variables ----------------------
# Mi is the number of words tested in test i
M1 <- as.integer(0)
M2 <- as.integer(0)
M3 <- as.integer(0)

# B_0i is the intercept for test i, used to create average of
    memory i
B_01 <- as.numeric(0)
B_02 <- as.numeric(0)
B_03 <- as.numeric(0)

# B_AGEi is the effect of age on ith memory score, used to
    create average of memory i
B_AGE1 <- as.numeric(0)
B_AGE2 <- as.numeric(0)
B_AGE3 <- as.numeric(0)

# B_SEXi is the effect of sex on ith memory score, used to
    create average of memory i
```

```
B_SEX1 <- as.numeric(0)
B_SEX2 <- as.numeric(0)
B_SEX3 <- as.numeric(0)

# B_EDULi is the effect of low education on ith memory score,
   used to create average of memory i
B_EDUL1 <- as.numeric(0)
B_EDUL2 <- as.numeric(0)
B_EDUL3 <- as.numeric(0)

# B_EDUMi is the effect of medium education on ith memory score,
    used to create average of memory i
B_EDUM1 <- as.numeric(0)
B_EDUM2 <- as.numeric(0)
B_EDUM3 <- as.numeric(0)

# B_PHYSLi is the effect of low PA on ith memory score, used to
   create average of memory i
B_PHYSL1 <- as.numeric(0)
B_PHYSL2 <- as.numeric(0)
B_PHYSL3 <- as.numeric(0)

# B_PHYSMi is the effect of medium PA on ith memory score, used
   to create average of memory i
B_PHYSM1 <- as.numeric(0)
B_PHYSM2 <- as.numeric(0)
B_PHYSM3 <- as.numeric(0)

# L_0i is used to create st.dev of memory i
L_01 <- as.numeric(0)
L_02 <- as.numeric(0)
L_03 <- as.numeric(0)

# L_AGEi is the effect of age on st.dev of ith memory score,
   used to create st.dev of memory i
L_AGE1 <- as.numeric(0)
L_AGE2 <- as.numeric(0)
L_AGE3 <- as.numeric(0)

# L_SEXi is the effect of sex on st.dev of ith memory score,
   used to create st.dev of memory i
L_SEX1 <- as.numeric(0)
L_SEX2 <- as.numeric(0)
L_SEX3 <- as.numeric(0)

# L_EDULi is the effect of low education on st.dev of ith memory
    score, used to create st.dev of memory i
L_EDUL1 <- as.numeric(0)
```

```
L_EDUL2 <- as.numeric(0)
L_EDUL3 <- as.numeric(0)

# L_EDUMi is the effect of medium education on st.dev of ith
   memory score, used to create st.dev of memory i
L_EDUM1 <- as.numeric(0)
L_EDUM2 <- as.numeric(0)
L_EDUM3 <- as.numeric(0)

# L_PHYSLi is the effect of low PA on st.dev of ith memory score
   , used to create st.dev of memory i
L_PHYSL1 <- as.numeric(0)
L_PHYSL2 <- as.numeric(0)
L_PHYSL3 <- as.numeric(0)

# L_PHYSMi is the effect of medium PA on st.dev of ith memory
   score, used to create st.dev of memory i
L_PHYSM1 <- as.numeric(0)
L_PHYSM2 <- as.numeric(0)
L_PHYSM3 <- as.numeric(0)


# Data frame ------------------------------------------

simulateDatasets <- function(
  SIM,
  N1,
  N2,
  N3,
  M1,
  M2,
  M3,
  SEED1,
  SEED2,
  SEED3,
  SEED4,
  SEED5,
  SEED6,
  SEED7,
  SEED8,
  M_AGE1,
  M_AGE2,
  M_AGE3,
  S_AGE1,
  S_AGE2,
  S_AGE3,
  P_SEX1,
  P_SEX2,
```

P_SEX3 ,
P_EDUL1 ,
P_EDUL2 ,
P_EDUL3 ,
P_EDUM1 ,
P_EDUM2 ,
P_EDUM3 ,
A_AGE1 ,
A_AGE2 ,
A_AGE3 ,
A_SEX1 ,
A_SEX2 ,
A_SEX3 ,
A_EDUL1 ,
A_EDUL2 ,
A_EDUL3 ,
A_EDUM1 ,
A_EDUM2 ,
A_EDUM3 ,
INT11 ,
INT12 ,
INT13 ,
INT21 ,
INT22 ,
INT23 ,
B_01 ,
B_02 ,
B_03 ,
B_AGE1 ,
B_AGE2 ,
B_AGE3 ,
B_SEX1 ,
B_SEX2 ,
B_SEX3 ,
B_EDUL1 ,
B_EDUL2 ,
B_EDUL3 ,
B_EDUM1 ,
B_EDUM2 ,
B_EDUM3 ,
B_PHYSL1 ,
B_PHYSL2 ,
B_PHYSL3 ,
B_PHYSM1 ,
B_PHYSM2 ,
B_PHYSM3 ,
L_01 ,
L_02 ,

```
L_03,
L_AGE1,
L_AGE2,
L_AGE3,
L_SEX1,
L_SEX2,
L_SEX3,
L_EDUL1,
L_EDUL2,
L_EDUL3,
L_EDUM1,
L_EDUM2,
L_EDUM3,
L_PHYSL1,
L_PHYSL2,
L_PHYSL3,
L_PHYSM1,
L_PHYSM2,
L_PHYSM3) {

# N is the total of the study sizes
N <- N1 + N2 + N3

# Initialize variables
Z_AGE <- numeric(N*SIM)
U_SEX <- numeric(N*SIM)
U_EDU <- numeric(N*SIM)
U_PHY <- numeric(N*SIM)
Z_MEM <- numeric(N*SIM)
AGE <- numeric(N*SIM)
SEX <- factor(N*SIM)
EDU <- factor(N*SIM)
MU_PHYS <- numeric(N*SIM)
CPHYS <- numeric(N*SIM)
PHYS <- factor(N*SIM)
MU_MEM1 <- numeric(N*SIM)
MU_MEM2 <- numeric(N*SIM)
MU_MEM3 <- numeric(N*SIM)
S_MEM1 <- numeric(N*SIM)
S_MEM2 <- numeric(N*SIM)
S_MEM3 <- numeric(N*SIM)
CMEM1 <- numeric(N*SIM)
CMEM2 <- numeric(N*SIM)
CMEM3 <- numeric(N*SIM)
P_MEM1 <- numeric(N*SIM)
P_MEM2 <- numeric(N*SIM)
P_MEM3 <- numeric(N*SIM)
MEMORY1 <- integer(N*SIM)
```

```
MEMORY2 <- integer(N*SIM)
MEMORY3 <- integer(N*SIM)

# Initialize variable for study indicator
STUDY <- integer(N*SIM)

# Generate Z_, U_ values for each person
set.seed(SEED1)
Z_AGE <- rnorm(N*SIM)

set.seed(SEED2)
U_SEX <- runif(N*SIM)

set.seed(SEED3)
U_EDU <- runif(N*SIM)

set.seed(SEED4)
U_PHY <- runif(N*SIM)

set.seed(SEED5)
Z_MEM <- rnorm(N*SIM)

# Generate covariates
# STUDY
STUDY <- rep(c(rep(1,N1),rep(2,N2),rep(3,N3)), times = SIM)

# AGE
AGE <- (M_AGE1 + S_AGE1*Z_AGE)*(STUDY == 1) +
   (M_AGE2 + S_AGE2*Z_AGE)*(STUDY == 2) +
   (M_AGE3 + S_AGE3*Z_AGE)*(STUDY == 3)

# SEX
SEX <- (U_SEX > P_SEX1)*(STUDY == 1) +
   (U_SEX > P_SEX2)*(STUDY == 2) +
   (U_SEX > P_SEX3)*(STUDY == 3)

# EDU
EDU <- ((U_EDU <= P_EDUL1) + 2*(U_EDU > P_EDUL1)*(U_EDU <=
    P_EDUM1) + 3*(U_EDU > P_EDUM1))*(STUDY == 1) +
   ((U_EDU <= P_EDUL2) + 2*(U_EDU > P_EDUL2)*(U_EDU <= P_EDUM2)
       + 3*(U_EDU > P_EDUM2))*(STUDY == 2) +
   ((U_EDU <= P_EDUL3) + 2*(U_EDU > P_EDUL3)*(U_EDU <= P_EDUM3)
       + 3*(U_EDU > P_EDUM3))*(STUDY == 3)

# Generating physical activity, dependent on other covariates
    generated until now
MU_PHYS <- (A_AGE1*AGE + A_SEX1*SEX + A_EDUL1*(EDU == 1) +
    A_EDUM1*(EDU == 2))*(STUDY == 1) +
```

```
   (A_AGE2*AGE + A_SEX2*SEX + A_EDUL2*(EDU == 1) + A_EDUM2*(EDU
       == 2))*(STUDY == 2) +
   (A_AGE3*AGE + A_SEX3*SEX + A_EDUL3*(EDU == 1) + A_EDUM3*(EDU
       == 2))*(STUDY == 3)

# Note that U_PHY is uniform, and quantile is the inverse of
    the distribution.
# The variable that comes from this quantile has mean 0.
# We then subtract the mean to get a continuous physical
    activity.
CPHYS = qlogis(U_PHY) - MU_PHYS

# Create low, med, high physical activity from the continuous
    distribution
# PA depends on the covariates through the mean
PHYS <- ((CPHYS <= INT11) + 2*(CPHYS > INT11)*(CPHYS <= INT21)
    + 3*(CPHYS > INT21))*(STUDY == 1) +
   ((CPHYS <= INT12) + 2*(CPHYS > INT12)*(CPHYS <= INT22) + 3*(
      CPHYS > INT22))*(STUDY == 2) +
   ((CPHYS <= INT13) + 2*(CPHYS > INT13)*(CPHYS <= INT23) + 3*(
      CPHYS > INT23))*(STUDY == 3)

# Generating memory
# MU_MEMi is the mean of the memory score i
MU_MEM1 <- B_01 + B_AGE1*AGE + B_SEX1*SEX + B_EDUL1*(EDU == 1)
    +
   B_EDUM1*(EDU == 2) + B_PHYSL1*(PHYS == 1) + B_PHYSM1*(PHYS
      == 2)

MU_MEM2 <- B_02 + B_AGE2*AGE + B_SEX2*SEX + B_EDUL2*(EDU == 1)
    +
   B_EDUM2*(EDU == 2) + B_PHYSL2*(PHYS == 1) + B_PHYSM2*(PHYS
      == 2)

MU_MEM3 <- B_03 + B_AGE3*AGE + B_SEX3*SEX + B_EDUL3*(EDU == 1)
    +
   B_EDUM3*(EDU == 2) + B_PHYSL3*(PHYS == 1) + B_PHYSM3*(PHYS
      == 2)

# S_MEMi is the st.dev of memory score i
S_MEM1 <- exp(L_01 + L_AGE1*AGE + L_SEX1*SEX + L_EDUL1*(EDU ==
    1) +
                              L_EDUM1*(EDU == 2) + L_PHYSL1*(PHYS
                                  == 1) +
                              L_PHYSM1*(PHYS == 2))

S_MEM2 <- exp(L_02 + L_AGE2*AGE + L_SEX2*SEX + L_EDUL2*(EDU ==
    1) +
```

```
                                    L_EDUM2*(EDU == 2) + L_PHYSL2*(PHYS
                                        == 1) +
                                    L_PHYSM2*(PHYS == 2))

S_MEM3 <- exp(L_03 + L_AGE3*AGE + L_SEX3*SEX + L_EDUL3*(EDU ==
    1) +
                                    L_EDUM3*(EDU == 2) + L_PHYSL3*(PHYS
                                        == 1) +
                                    L_PHYSM3*(PHYS == 2))

# CMEMi is the continuous memory value for the ith memory
    score, a normal latent variable
CMEM1 <- MU_MEM1 + S_MEM1*Z_MEM
CMEM2 <- MU_MEM2 + S_MEM2*Z_MEM
CMEM3 <- MU_MEM3 + S_MEM3*Z_MEM

# P_MEMi is the probability of successes, where every word is
    considered to
# have the same probability of being remembered. Change of
    normal to logistic,
# considering the sum of outcomes to be binomial
P_MEM1 <- exp(CMEM1)/(1+exp(CMEM1))
P_MEM2 <- exp(CMEM2)/(1+exp(CMEM2))
P_MEM3 <- exp(CMEM3)/(1+exp(CMEM3))

set.seed(SEED6)
MEMORY1 <- rbinom(N*SIM, M1, P_MEM1)

set.seed(SEED7)
MEMORY2 <- rbinom(N*SIM, M2, P_MEM2)

set.seed(SEED8)
MEMORY3 <- rbinom(N*SIM, M3, P_MEM3)

SIMNUM <- as.integer(rep(1:SIM, each=N))
SUBJECTNUM <- as.integer(c(1:N))
STUDY <- as.integer(STUDY)
SEX <- as.factor(SEX)
EDU <- as.factor(EDU)
PHYS <- as.factor(PHYS)
MEMORY1 <- as.integer(MEMORY1)
MEMORY2 <- as.integer(MEMORY2)
MEMORY3 <- as.integer(MEMORY3)

# df <- data.frame(as.integer(SIMNUM), as.integer(SUBJECTNUM),
    as.factor(STUDY),
#                     as.numeric(AGE), as.factor(SEX), as.factor(
    EDU), as.factor(PHYS),
```

```
  #                    as.integer(MEMORY1), as.integer(MEMORY2),
      as.factor(MEMORY3))
  df <- data.frame(SIMNUM, SUBJECTNUM, STUDY, AGE, SEX, EDU,
      PHYS, MEMORY1, MEMORY2, MEMORY3)
  return(df)
}

# Creating a dataset ------------------------------------
# CONFOUNDING OF AGE, SEX, AND EDU:
          YES
# IDENTICAL POPULATIONS ON AGE, SEX, AND EDU:          YES
# IDENTICAL EFFECTS OF AGE, SEX, AND EDU ON PA:        YES
# ID EFFECTS OF AGE, SEX, AND EDU ON MEMORY:           YES

SEED1=578576
SEED2=6673968
SEED3=98923436
SEED4=53273265
SEED5=4214621
SEED6=8347626
SEED7=8734147
SEED8=4756277
SIM=100
N1=10000
N2=2000
N3=500
M1=15
M2=12
M3=16
M_AGE1=70
S_AGE1=6
P_SEX1=0.40
P_EDUL1=0.20
P_EDUM1=0.60
M_AGE2=80
S_AGE2=7
P_SEX2=0.35
P_EDUL2=0.50
P_EDUM2=0.85
M_AGE3=75
S_AGE3=4
P_SEX3=0.45
P_EDUL3=0.15
P_EDUM3=0.55
INT11=1.5
INT21=3.5
A_AGE1=-0.03
A_SEX1=-0.50
```

A_EDUL1=−0.60
A_EDUM1=−0.30
INT12=1.5
INT22=3.5
A_AGE2=−0.03
A_SEX2=−0.50
A_EDUL2=−0.60
A_EDUM2=−0.30
INT13=1.5
INT23=3.5
A_AGE3=−0.03
A_SEX3=−0.50
A_EDUL3=−0.60
A_EDUM3=−0.30
B_01=2.5
B_02=2.5
B_03=2.5
B_AGE1=−0.025
B_SEX1=0.20
B_EDUL1=−0.25
B_EDUM1=−0.15
B_PHYSL1=−0.50
B_PHYSM1=−0.25
B_AGE2=−0.025
B_SEX2=0.20
B_EDUL2=−0.25
B_EDUM2=−0.15
B_PHYSL2=−0.50
B_PHYSM2=−0.25
B_AGE3=−0.025
B_SEX3=0.20
B_EDUL3=−0.25
B_EDUM3=−0.15
B_PHYSL3=−0.50
B_PHYSM3=−0.25
L_01=−2.00
L_02=−2.00
L_03=−2.00
L_AGE1=0.02
L_SEX1=0.15
L_EDUL1=−0.15
L_EDUM1=−0.05
L_PHYSL1=0
L_PHYSM1=0
L_AGE2=0.02
L_SEX2=0.15
L_EDUL2=−0.15
L_EDUM2=−0.05

```
L_PHYSL2=0
L_PHYSM2=0
L_AGE3=0.02
L_SEX3=0.15
L_EDUL3=-0.15
L_EDUM3=-0.05
L_PHYSL3=0
L_PHYSM3=0

simulatedDataset <- simulateDatasets(
  SIM,
  N1,
  N2,
  N3,
  M1,
  M2,
  M3,
  SEED1,
  SEED2,
  SEED3,
  SEED4,
  SEED5,
  SEED6,
  SEED7,
  SEED8,
  M_AGE1,
  M_AGE2,
  M_AGE3,
  S_AGE1,
  S_AGE2,
  S_AGE3,
  P_SEX1,
  P_SEX2,
  P_SEX3,
  P_EDUL1,
  P_EDUL2,
  P_EDUL3,
  P_EDUM1,
  P_EDUM2,
  P_EDUM3,
  A_AGE1,
  A_AGE2,
  A_AGE3,
  A_SEX1,
  A_SEX2,
  A_SEX3,
  A_EDUL1,
  A_EDUL2,
```

A_EDUL3,
A_EDUM1,
A_EDUM2,
A_EDUM3,
INT11,
INT12,
INT13,
INT21,
INT22,
INT23,
B_01,
B_02,
B_03,
B_AGE1,
B_AGE2,
B_AGE3,
B_SEX1,
B_SEX2,
B_SEX3,
B_EDUL1,
B_EDUL2,
B_EDUL3,
B_EDUM1,
B_EDUM2,
B_EDUM3,
B_PHYSL1,
B_PHYSL2,
B_PHYSL3,
B_PHYSM1,
B_PHYSM2,
B_PHYSM3,
L_01,
L_02,
L_03,
L_AGE1,
L_AGE2,
L_AGE3,
L_SEX1,
L_SEX2,
L_SEX3,
L_EDUL1,
L_EDUL2,
L_EDUL3,
L_EDUM1,
L_EDUM2,
L_EDUM3,
L_PHYSL1,
L_PHYSL2,

```
  L_PHYSL3,
  L_PHYSM1,
  L_PHYSM2,
  L_PHYSM3)

setwd("/Users/catherinehilgers/Documents/Thesis/R Code")
write.csv(simulatedDataset, "simulatedDataset.csv",row.names=
  FALSE)
```

A.4 Missingness Generation

```
library(dplyr)

# Load data set
simulatedDataset <- read.csv("/Users/catherinehilgers/Documents/
  Thesis/R Code/simulatedDataset.csv")

# Set appropriate value types
simulatedDataset$SIMNUM <- as.factor(simulatedDataset$SIMNUM)
simulatedDataset$STUDY <- as.factor(simulatedDataset$STUDY)
simulatedDataset$SUBJECTNUM <- as.integer(
  simulatedDataset$SUBJECTNUM)
simulatedDataset$SEX <- as.factor(simulatedDataset$SEX)
simulatedDataset$EDU <- as.factor(simulatedDataset$EDU)
simulatedDataset$PHYS <- as.factor(simulatedDataset$PHYS)
simulatedDataset$MEMORY1 <- as.integer(simulatedDataset$MEMORY1)
simulatedDataset$MEMORY2 <- as.integer(simulatedDataset$MEMORY2)
simulatedDataset$MEMORY3 <- as.integer(simulatedDataset$MEMORY3)

# Create a simulated dataset with sporadic and systematically
  missing values
# MAR: want 20% of missingness in each memory score for the
  highest quantile,
# 15% for the next-highest, 10% for the third-highest, and 5%
  for the lowest

# Find quantiles of AGE for each study in each simulation
quantiles <- by(simulatedDataset, simulatedDataset[, c("SIMNUM",
  "STUDY")],
                function(x) as.vector(quantile(x$AGE)))

# Organize the quantiles into a data frame
quantilesdf <- data.frame(matrix(unlist(quantiles), nrow = (3*
  SIM), byrow=T))
colnames(quantilesdf) <- c("q0", "q1", "q2", "q3", "q4")
quantilesdf$SIMNUM <- as.factor(rep(c(1:SIM),each = 3))
quantilesdf$STUDY <- as.factor(rep(c(1:3),SIM))
```

```
# Join quantiles data frame with the simulated data set
simulatedQuantiles <- inner_join(simulatedDataset, quantilesdf)

# Create a proportion missing variable
simulatedQuantiles$PROPMISSING <- numeric(nrow(
   simulatedQuantiles))

# 20% of missingness in each memory score for the highest
   quantile,
# 15% for the next-highest, 10% for the third-highest, and 5%
   for the lowest
simulatedQuantiles$PROPMISSING <- 0.05*(simulatedQuantiles$AGE
   <= simulatedQuantiles$q1) +
  0.1*((simulatedQuantiles$q1 < simulatedQuantiles$AGE) & (
     simulatedQuantiles$AGE <= simulatedQuantiles$q2)) +
  0.15*((simulatedQuantiles$q2 < simulatedQuantiles$AGE) & (
     simulatedQuantiles$AGE <= simulatedQuantiles$q3)) +
  0.2*(simulatedQuantiles$AGE > simulatedQuantiles$q3)

# Generate uniformly distributed vectors to check missingness
   against
set.seed(54839)
missingq_MEM1 <- runif(nrow(simulatedQuantiles), min=0, max=1)
set.seed(58734)
missingq_MEM2 <- runif(nrow(simulatedQuantiles), min=0, max=1)
set.seed(28304)
missingq_MEM3 <- runif(nrow(simulatedQuantiles), min=0, max=1)

simulatedQuantiles$MEMORY1 <- ifelse(missingq_MEM1 <
   simulatedQuantiles$PROPMISSING, NA,
   simulatedQuantiles$MEMORY1)
simulatedQuantiles$MEMORY2 <- ifelse(missingq_MEM2 <
   simulatedQuantiles$PROPMISSING, NA,
   simulatedQuantiles$MEMORY2)
simulatedQuantiles$MEMORY3 <- ifelse(missingq_MEM3 <
   simulatedQuantiles$PROPMISSING, NA,
   simulatedQuantiles$MEMORY3)

# MCAR missingness: set 5% of all variables to missing
# 5% missingness
proportion_missing <- 0.05

# Create vectors for missingness
set.seed(124)
missing_AGE <- runif(nrow(simulatedDataset), min=0, max=1)
set.seed(356)
missing_EDU <- runif(nrow(simulatedDataset), min=0, max=1)
```

```
set.seed(455)
missing_SEX <- runif(nrow(simulatedDataset), min=0, max=1)
set.seed(306)
missing_PHYS <- runif(nrow(simulatedDataset), min=0, max=1)
set.seed(654)
missing_MEM1 <- runif(nrow(simulatedDataset), min=0, max=1)
set.seed(456)
missing_MEM2 <- runif(nrow(simulatedDataset), min=0, max=1)
set.seed(789)
missing_MEM3 <- runif(nrow(simulatedDataset), min=0, max=1)

# Set to missing
simulatedQuantiles$AGE <- ifelse(missing_AGE <
    proportion_missing, NA, simulatedQuantiles$AGE)
simulatedQuantiles$EDU <- ifelse(missing_EDU <
    proportion_missing, NA, simulatedQuantiles$EDU)
simulatedQuantiles$SEX <- ifelse(missing_SEX <
    proportion_missing, NA, simulatedQuantiles$SEX)
simulatedQuantiles$PHYS <- ifelse(missing_PHYS <
    proportion_missing, NA, simulatedQuantiles$PHYS)
simulatedQuantiles$MEMORY1 <- ifelse(missing_MEM1 <
    proportion_missing, NA, simulatedQuantiles$MEMORY1)
simulatedQuantiles$MEMORY2 <- ifelse(missing_MEM2 <
    proportion_missing, NA, simulatedQuantiles$MEMORY2)
simulatedQuantiles$MEMORY3 <- ifelse(missing_MEM3 <
    proportion_missing, NA, simulatedQuantiles$MEMORY3)

# Now simulatedQuantiles is a data frame with both MAR and MCAR
    missingness

# Systematic missingness
# Case with overlap: Study 1 has MEMORY1 and MEMORY2, Study 2
    has MEMORY1 only, Study 3 has MEMORY2 only
simulatedOverlap <- simulatedQuantiles[, c("SIMNUM", "SUBJECTNUM
    ", "STUDY", "AGE", "EDU", "SEX", "PHYS", "MEMORY1",
                                                "MEMORY2")]
simulatedOverlap$MEMORY1 <- ifelse(simulatedOverlap$STUDY == 3,
    NA, simulatedOverlap$MEMORY1)
simulatedOverlap$MEMORY2 <- ifelse(simulatedOverlap$STUDY == 2,
    NA, simulatedOverlap$MEMORY2)
write.csv(simulatedOverlap, "simulatedOverlap.csv",row.names=
    FALSE)

# Case without overlap: Study 1 has MEMORY1, Study 2 has MEMORY2
    , Study 3 has MEMORY3 only
simulatedNoOverlap <- simulatedQuantiles[, c("SIMNUM", "
    SUBJECTNUM", "STUDY", "AGE", "EDU", "SEX", "PHYS", "MEMORY1",
                                            "MEMORY2", "
```

```
                                                    MEMORY3")]
simulatedNoOverlap$MEMORY1 <- ifelse(simulatedNoOverlap$STUDY ==
    1, simulatedNoOverlap$MEMORY1, NA)
simulatedNoOverlap$MEMORY2 <- ifelse(simulatedNoOverlap$STUDY ==
    2, simulatedNoOverlap$MEMORY2, NA)
simulatedNoOverlap$MEMORY3 <- ifelse(simulatedNoOverlap$STUDY ==
    3, simulatedNoOverlap$MEMORY3, NA)
write.csv(simulatedNoOverlap, "simulatedNoOverlap.csv",row.names
   =FALSE)
```

A.5 Imputation

```
# APPLY jomo ------------------------------------------------------
library(jomo)
library(dplyr)

# Load data set with overlap, systematic and sporadic
    missingness
simulatedOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/simulatedOverlap.csv")
simulatedOverlap$SIMNUM <- as.factor(simulatedOverlap$SIMNUM)
simulatedOverlap$STUDY <- as.factor(simulatedOverlap$STUDY)
simulatedOverlap$SUBJECTNUM <- as.integer(
    simulatedOverlap$SUBJECTNUM)
simulatedOverlap$SEX <- as.factor(simulatedOverlap$SEX)
simulatedOverlap$EDU <- as.factor(simulatedOverlap$EDU)
simulatedOverlap$PHYS <- as.factor(simulatedOverlap$PHYS)
simulatedOverlap$MEMORY1 <- as.factor(simulatedOverlap$MEMORY1)
simulatedOverlap$MEMORY2 <- as.factor(simulatedOverlap$MEMORY2)

# Set parameters for jomo
nburn <- as.integer(10)
nbetween <- as.integer(10)
nimp <- as.integer(5)

imp_Overlap <- by(simulatedOverlap, simulatedOverlap[,"SIMNUM"],
          function(x) jomo(x[, c("AGE", "SEX", "EDU", "PHYS", "
             MEMORY1", "MEMORY2")],
                           clus=x[, "STUDY"],
                           nburn=nburn,
                           nbetween=nbetween,
                           nimp=nimp))

imputedDatasets_jomoOverlap <- bind_rows(imp_Overlap[1:SIM])
setwd("/Users/catherinehilgers/Documents/Thesis/R Code")
write.csv(imputedDatasets_jomoOverlap, "
    imputedDatasets_jomoOverlap.csv")
```

```
# Load data set with NO overlap, with systematic and sporadic
    missingness
simulatedNoOverlap <- read.csv("/Users/catherinehilgers/
    Documents/Thesis/R Code/simulatedNoOverlap.csv")

# Set appropriate value types - important for jomo
simulatedNoOverlap$SIMNUM <- as.factor(simulatedNoOverlap$SIMNUM
    )
simulatedNoOverlap$STUDY <- as.factor(simulatedNoOverlap$STUDY)
simulatedNoOverlap$SUBJECTNUM <- as.integer(
    simulatedNoOverlap$SUBJECTNUM)
simulatedNoOverlap$SEX <- as.factor(simulatedNoOverlap$SEX)
simulatedNoOverlap$EDU <- as.factor(simulatedNoOverlap$EDU)
simulatedNoOverlap$PHYS <- as.factor(simulatedNoOverlap$PHYS)
simulatedNoOverlap$MEMORY1 <- as.numeric(
    simulatedNoOverlap$MEMORY1)
simulatedNoOverlap$MEMORY2 <- as.numeric(
    simulatedNoOverlap$MEMORY2)
simulatedNoOverlap$MEMORY3 <- as.numeric(
    simulatedNoOverlap$MEMORY3)

# Set parameters for jomo
nburn <- as.integer(10)
nbetween <- as.integer(10)
nimp <- as.integer(5)

imp_NoOverlap <- by(simulatedNoOverlap, simulatedNoOverlap[,"
    SIMNUM"],
          function(x) jomo(x[, c("AGE", "SEX", "EDU", "PHYS", "
              MEMORY1", "MEMORY2", "MEMORY3")],
                           clus=x[, "STUDY"],
                           nburn=nburn,
                           nbetween=nbetween,
                           nimp=nimp))

imputedDatasets_jomoNoOverlap <- bind_rows(imp_NoOverlap[1:SIM])
setwd("/Users/catherinehilgers/Documents/Thesis/R Code")
write.csv(imputedDatasets_jomoNoOverlap, "
    imputedDatasets_jomoNoOverlap.csv")

library(mice)
library(micemd)
library(lme4)
library(dplyr)


#-----------calculate missingness percentage in each outcome
    var ----------
```

```
i=3
  sum(is.na(simulatedNoOverlap$MEMORY1[simulatedNoOverlap$STUDY
      == i]))/500
  sum(is.na(simulatedNoOverlap$MEMORY2[simulatedNoOverlap$STUDY
      == i]))/500
  sum(is.na(simulatedNoOverlap$MEMORY3[simulatedNoOverlap$STUDY
      == i]))/500


  sum(is.na(simulatedOverlap$MEMORY1[simulatedOverlap$STUDY == i
      ]))/500
  sum(is.na(simulatedOverlap$MEMORY2[simulatedOverlap$STUDY == i
      ]))/500

# Load data set with overlap, systematic and sporadic
    missingness
simulatedOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/simulatedOverlap.csv")
simulatedOverlap$SIMNUM <- as.factor(simulatedOverlap$SIMNUM)
simulatedOverlap$STUDY <- as.integer(simulatedOverlap$STUDY)
simulatedOverlap$SUBJECTNUM <- as.integer(
    simulatedOverlap$SUBJECTNUM)
simulatedOverlap$SEX <- as.factor(simulatedOverlap$SEX)
simulatedOverlap$EDU <- as.factor(simulatedOverlap$EDU)
simulatedOverlap$PHYS <- as.factor(simulatedOverlap$PHYS)
simulatedOverlap$MEMORY1 <- as.factor(simulatedOverlap$MEMORY1)
simulatedOverlap$MEMORY2 <- as.factor(simulatedOverlap$MEMORY2)

# Set parameters
# Index of cluster variable is 3 for STUDY
ind.clust <- 3

# Initialisation of the argument predictorMatrix
temp <- by(simulatedOverlap, simulatedOverlap[,"SIMNUM"],
    function(x) mice(x, m=1, maxit=0))

for (i in 1:SIM) {
  temp[[i]]$pred[ind.clust,ind.clust] <- 0
  temp[[i]]$pred[-ind.clust,ind.clust] <- -2
  temp[[i]]$pred[temp$pred == 1] <- 2
}

predictor.matrix <- as.matrix(temp[[1]]$pred)

# Initialisation of the argument method
method_temp <- by(simulatedOverlap, simulatedOverlap[,"SIMNUM"],
              function(x) find.defaultMethod(x, ind.clust))
```

```
method <- as.vector(method_temp[[1]])

imp_FCS2stgOverlap <- by(simulatedOverlap, simulatedOverlap[,"
    SIMNUM"],
                         function(x) mice.par(x, predictorMatrix
                             = predictor.matrix))

# imputedDatasets_FCS2stgOverlap <- bind_rows(imp_FCS2stgOverlap
    )

setwd("/Users/catherinehilgers/Documents/Thesis/R Code")
save(imp_FCS2stgOverlap, file="imp_FCS2stgOverlap.RData")

#
    ------------------------------------------------------------


# Load data set with NO overlap, with systematic and sporadic
    missingness
simulatedNoOverlap <- read.csv("/Users/catherinehilgers/
    Documents/Thesis/R Code/simulatedNoOverlap.csv")

simulatedNoOverlap$SIMNUM <- as.factor(simulatedNoOverlap$SIMNUM
    )
simulatedNoOverlap$STUDY <- as.integer(simulatedNoOverlap$STUDY)
simulatedNoOverlap$SUBJECTNUM <- as.integer(
    simulatedNoOverlap$SUBJECTNUM)
simulatedNoOverlap$SEX <- as.factor(simulatedNoOverlap$SEX)
simulatedNoOverlap$EDU <- as.factor(simulatedNoOverlap$EDU)
simulatedNoOverlap$PHYS <- as.factor(simulatedNoOverlap$PHYS)
simulatedNoOverlap$MEMORY1 <- as.factor(
    simulatedNoOverlap$MEMORY1)
simulatedNoOverlap$MEMORY2 <- as.factor(
    simulatedNoOverlap$MEMORY2)
simulatedNoOverlap$MEMORY3 <- as.factor(
    simulatedNoOverlap$MEMORY3)

# Set parameters
# Index of cluster variable is 3 for STUDY
ind.clust <- 3

# Initialisation of the argument predictorMatrix
temp_NoOverlap <- by(simulatedNoOverlap, simulatedNoOverlap[,"
    SIMNUM"],
             function(x) mice(x, m=1, maxit=0))

for (i in 1:SIM) {
  temp_NoOverlap[[i]]$pred[ind.clust,ind.clust] <- 0
  temp_NoOverlap[[i]]$pred[-ind.clust,ind.clust] <- -2
```

```
  temp_NoOverlap[[i]]$pred[temp$pred == 1] <- 2
}

predictor.matrix_NoOverlap <- as.matrix(temp_NoOverlap[[1]]$pred
    )

imp_FCS2stgNoOverlap <- by(simulatedNoOverlap,
    simulatedNoOverlap[,"SIMNUM"],
                           function(x) mice.par(x, predictorMatrix
                               = predictor.matrix_NoOverlap))

setwd("/Users/catherinehilgers/Documents/Thesis/R Code")
save(imp_FCS2stgNoOverlap, file="imp_FCS2stgNoOverlap.RData")

# ------------------------------------------
# Load them up now that they are done

load("/Users/catherinehilgers/Documents/Thesis/R Code/
    imp_FCS2stgNoOverlap.RData")
load("/Users/catherinehilgers/Documents/Thesis/R Code/
    imp_FCS2stgOverlap.RData")

# Check number of simulations
SIM <- 100

FCS2stgfunction <- function(df,SIM) {
  # Combine the five imputed data sets for each simulation into
     one imputed data frame
  # 1st imputation
  dataframe1 <- complete(df[[1]], 1)
  for (i in 2:SIM) {
    dataframe1 <- bind_rows(dataframe1, complete(df[[i]], 1))
  }

  # 2nd imputation
  dataframe2 <- complete(df[[1]], 2)
  for (i in 2:SIM) {
    dataframe2 <- bind_rows(dataframe2, complete(df[[i]], 2))
  }

  # 3rd imputation
  dataframe3 <- complete(df[[1]], 3)
  for (i in 2:SIM) {
    dataframe3 <- bind_rows(dataframe3, complete(df[[i]], 3))
  }

  # 4th imputation
  dataframe4 <- complete(df[[1]], 4)
```

```
  for (i in 2:SIM) {
    dataframe4 <- bind_rows(dataframe4, complete(df[[i]], 4))
  }

  # 5th imputation
  dataframe5 <- complete(df[[1]], 5)
  for (i in 2:SIM) {
    dataframe5 <- bind_rows(dataframe5, complete(df[[i]], 5))
  }

  complete_imputed <- do.call("rbind", list(dataframe1,
     dataframe2, dataframe3, dataframe4, dataframe5))

  IMPNUM <- as.integer(rep(1:5, each=1250000))

  complete_imputed <- cbind(complete_imputed, IMPNUM)

  return(complete_imputed)
}

FCS2stgOverlap <- FCS2stgfunction(imp_FCS2stgOverlap, SIM)
FCS2stgNoOverlap <- FCS2stgfunction(imp_FCS2stgNoOverlap, SIM)

setwd("/Users/catherinehilgers/Documents/Thesis/R Code")
write.csv(FCS2stgOverlap, "FCS2stgOverlap.csv",row.names=FALSE)
write.csv(FCS2stgNoOverlap, "FCS2stgNoOverlap.csv",row.names=
   FALSE)
```

A.6 Analysis Model

```
library(dplyr)
library(lme4)

# Load the complete data set
complete <- read.csv("/Users/catherinehilgers/Documents/Thesis/R
    Code/simulatedDataset.csv")
complete$SIMNUM <- as.factor(complete$SIMNUM)
complete$STUDY <- as.factor(complete$STUDY)
complete$SUBJECTNUM <- as.integer(complete$SUBJECTNUM)
complete$SEX <- as.factor(complete$SEX)
complete$EDU <- as.factor(complete$EDU)
complete$PHYS <- as.factor(complete$PHYS)
complete$MEMORY1 <- as.integer(complete$MEMORY1)
complete$MEMORY2 <- as.integer(complete$MEMORY2)
complete$MEMORY3 <- as.integer(complete$MEMORY3)

# Load the jomo data sets
# Use for both jomo data sets
```

```
SIM=100
N1=10000
N2=2000
N3=500
N = N1 + N2 + N3
SIMNUM <- as.integer(rep(rep(1:SIM, each=N), each = 5))

# Load the jomo with overlap case
jomoOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/imputedDatasets_jomoOverlap.csv")
jomoOverlap$clus <- as.factor(jomoOverlap$clus)
colnames(jomoOverlap)[10] <- "STUDY"
jomoOverlap$id <- as.integer(jomoOverlap$id)
colnames(jomoOverlap)[11] <- "SUBJECTNUM"
jomoOverlap$SEX <- as.factor(jomoOverlap$SEX)
jomoOverlap$EDU <- as.factor(jomoOverlap$EDU)
jomoOverlap$PHYS <- as.factor(jomoOverlap$PHYS)
jomoOverlap$MEMORY1 <- as.integer(jomoOverlap$MEMORY1)
jomoOverlap$MEMORY2 <- as.integer(jomoOverlap$MEMORY2)
jomoOverlap$X <- NULL
jomoOverlap$X1 <- NULL
jomoOverlap$Z1 <- NULL
jomoOverlap <- jomoOverlap[!(jomoOverlap$Imputation == 0), ]
colnames(jomoOverlap)[9] <- "IMPNUM"
jomoOverlap <- cbind(jomoOverlap,SIMNUM)
head(jomoOverlap)

# jomo without overlap
jomoNoOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/imputedDatasets_jomoNoOverlap.csv")
jomoNoOverlap$clus <- as.factor(jomoNoOverlap$clus)
colnames(jomoNoOverlap)[11] <- "STUDY"
jomoNoOverlap$id <- as.integer(jomoNoOverlap$id)
colnames(jomoNoOverlap)[12] <- "SUBJECTNUM"
jomoNoOverlap$SEX <- as.factor(jomoNoOverlap$SEX)
jomoNoOverlap$EDU <- as.factor(jomoNoOverlap$EDU)
jomoNoOverlap$PHYS <- as.factor(jomoNoOverlap$PHYS)
# rounding MEMORY1,2,3 to integers after they have been imputed
    as numeric
# Some are out of bounds and must be changed to the highest
    level
jomoNoOverlap$MEMORY1 <- as.integer(round(jomoNoOverlap$MEMORY1,
     digits = 0))
jomoNoOverlap$MEMORY1[jomoNoOverlap$MEMORY1 == 16] <- as.integer
    (15)
jomoNoOverlap$MEMORY1[jomoNoOverlap$MEMORY1 == 17] <- as.integer
    (15)
jomoNoOverlap$MEMORY2 <- as.integer(round(jomoNoOverlap$MEMORY2,
```

```
    digits = 0))
jomoNoOverlap$MEMORY2[jomoNoOverlap$MEMORY2 == 13] <- as.integer
    (12)
jomoNoOverlap$MEMORY2[jomoNoOverlap$MEMORY2 == 14] <- as.integer
    (12)
jomoNoOverlap$MEMORY2[jomoNoOverlap$MEMORY2 == 15] <- as.integer
    (12)
jomoNoOverlap$MEMORY3 <- as.integer(round(jomoNoOverlap$MEMORY3,
    digits = 0))
jomoNoOverlap$MEMORY3[jomoNoOverlap$MEMORY3 == 18] <- as.integer
    (16)
jomoNoOverlap$MEMORY3[jomoNoOverlap$MEMORY3 == 17] <- as.integer
    (16)
jomoNoOverlap$X <- NULL
jomoNoOverlap$X1 <- NULL
jomoNoOverlap$Z1 <- NULL
jomoNoOverlap <- jomoNoOverlap[!(jomoNoOverlap$Imputation == 0),
    ]
colnames(jomoNoOverlap)[10] <- "IMPNUM"
jomoNoOverlap <- cbind(jomoNoOverlap,SIMNUM)
head(jomoNoOverlap)

# Load the FCS 2stage data sets
# FCS 2 stage without overlap
FCS2stgNoOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/FCS2stgNoOverlap.csv")
FCS2stgNoOverlap$SIMNUM <- as.factor(FCS2stgNoOverlap$SIMNUM)
FCS2stgNoOverlap$STUDY <- as.factor(FCS2stgNoOverlap$STUDY)
FCS2stgNoOverlap$SUBJECTNUM <- as.integer(
    FCS2stgNoOverlap$SUBJECTNUM)
FCS2stgNoOverlap$SEX <- as.factor(FCS2stgNoOverlap$SEX)
FCS2stgNoOverlap$EDU <- as.factor(FCS2stgNoOverlap$EDU)
FCS2stgNoOverlap$PHYS <- as.factor(FCS2stgNoOverlap$PHYS)
FCS2stgNoOverlap$MEMORY1 <- as.integer(FCS2stgNoOverlap$MEMORY1)
FCS2stgNoOverlap$MEMORY2 <- as.integer(FCS2stgNoOverlap$MEMORY2)
FCS2stgNoOverlap$MEMORY3 <- as.integer(FCS2stgNoOverlap$MEMORY3)

# FCS 2 stage with overlap
FCS2stgOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/FCS2stgOverlap.csv")
FCS2stgOverlap$SIMNUM <- as.factor(FCS2stgOverlap$SIMNUM)
FCS2stgOverlap$STUDY <- as.factor(FCS2stgOverlap$STUDY)
FCS2stgOverlap$SUBJECTNUM <- as.integer(
    FCS2stgOverlap$SUBJECTNUM)
FCS2stgOverlap$SEX <- as.factor(FCS2stgOverlap$SEX)
FCS2stgOverlap$EDU <- as.factor(FCS2stgOverlap$EDU)
FCS2stgOverlap$PHYS <- as.factor(FCS2stgOverlap$PHYS)
FCS2stgOverlap$MEMORY1 <- as.integer(FCS2stgOverlap$MEMORY1)
```

```
FCS2stgOverlap$MEMORY2 <- as.integer(FCS2stgOverlap$MEMORY2)

# ------------------------------------------

# Apply a GLM model to each of complete data sets

# Create a column for the number of items minus the items
    correct
M1=15
M2=12
M3=16

complete$M1 <- rep(M1, times = 125000)
complete$M2 <- rep(M2, times = 125000)
complete$M3 <- rep(M3, times = 125000)

FCS2stgOverlap$M1 <- rep(M1, times = 625000)
FCS2stgOverlap$M2 <- rep(M2, times = 625000)
FCS2stgOverlap$M3 <- rep(M3, times = 625000)

FCS2stgNoOverlap$M1 <- rep(M1, times = 625000)
FCS2stgNoOverlap$M2 <- rep(M2, times = 625000)
FCS2stgNoOverlap$M3 <- rep(M3, times = 625000)

jomoOverlap$M1 <- rep(M1, times = 625000)
jomoOverlap$M2 <- rep(M2, times = 625000)
jomoOverlap$M3 <- rep(M3, times = 625000)

jomoNoOverlap$M1 <- rep(M1, times = 625000)
jomoNoOverlap$M2 <- rep(M2, times = 625000)
jomoNoOverlap$M3 <- rep(M3, times = 625000)

SIM <- 100
# Fit model on complete data set using glmer
    ------------------------------------

# MEMORY 1
complete_coefsMEM1 <- data.frame("SIMNUM" = as.integer(rep(c(1:
    SIM), each = 3)),
                                 "Intercept" = numeric(300),
                                 "EDU2" = numeric(300),
                                 "EDU3" = numeric(300),
                                 "SEX1" = numeric(300),
                                 "AGE" = numeric(300),
                                 "PHYS2" = numeric(300),
                                 "PHYS3" = numeric(300))
for (i in 1:SIM) {
  temp <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
```

```
                    data = complete[complete$SIMNUM == i, ],
                    family = binomial)
  # Set values to the generated coefficient numbers
  complete_coefsMEM1[c((3*(i-1) + 1):(3*(i-1) + 3)),c(2:8)] <-
      coef(temp)$STUDY
}

setwd("/Users/catherinehilgers/Documents/Thesis/R Code")
write.csv(complete_coefsMEM1, "complete_coefsMEM1.csv")

# MEMORY2
complete_coefsMEM2 <- data.frame("SIMNUM" = as.integer(rep(c(1:
    SIM), each = 3)),
                                 "Intercept" = numeric(300),
                                 "EDU2" = numeric(300),
                                 "EDU3" = numeric(300),
                                 "SEX1" = numeric(300),
                                 "AGE" = numeric(300),
                                 "PHYS2" = numeric(300),
                                 "PHYS3" = numeric(300))
for (i in 1:SIM) {
  temp <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                    data = complete[complete$SIMNUM == i, ],
                    family = binomial)
  # Set values to the generated coefficient numbers
  complete_coefsMEM2[c((3*(i-1) + 1):(3*(i-1) + 3)),c(2:8)] <-
      coef(temp)$STUDY
}

write.csv(complete_coefsMEM2, "complete_coefsMEM2.csv")

# MEMORY3
complete_coefsMEM3 <- data.frame("SIMNUM" = as.integer(rep(c(1:
    SIM), each = 3)),
                                 "Intercept" = numeric(300),
                                 "EDU2" = numeric(300),
                                 "EDU3" = numeric(300),
                                 "SEX1" = numeric(300),
                                 "AGE" = numeric(300),
                                 "PHYS2" = numeric(300),
                                 "PHYS3" = numeric(300))
for (i in 1:SIM) {
  temp <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                    data = complete[complete$SIMNUM == i, ],
                    family = binomial)
  # Set values to the generated coefficient numbers
```

```
  complete_coefsMEM3[c((3*(i-1) + 1):(3*(i-1) + 3)),c(2:8)] <-
    coef(temp)$STUDY
}

write.csv(complete_coefsMEM3, "complete_coefsMEM3.csv")

# FCS 2 stage No Overlap
-------------------------------------------------------------------------------

# MEM1
FCS2stgNoOverlap_coefsMEM1 <- data.frame("IMPNUM" = as.integer(
    rep(c(1:5), each = 300)),
                                         "SIMNUM" = as.integer(
                                            rep(rep(c(1:SIM),
                                            each = 3)), times =
                                            3),
                                         "Intercept" = numeric
                                            (1500),
                                         "EDU2" = numeric(1500),
                                         "EDU3" = numeric(1500),
                                         "SEX1" = numeric(1500),
                                         "AGE" = numeric(1500),
                                         "PHYS2" = numeric(1500)
                                            ,
                                         "PHYS3" = numeric(1500)
                                            )

for (i in 1:SIM) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 1)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM1[c((3*(i-1) + 1):(3*(i-1) + 3)),c
     (3:9)] <- coef(temp1)$STUDY

  # Imputation 2
  temp2 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                  data = FCS2stgNoOverlap[((
                     FCS2stgNoOverlap$SIMNUM == i) & (
                     FCS2stgNoOverlap$IMPNUM == 2)), ],
                  family = binomial)
```

```
  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM1[c((3*(i-1) + 301):(3*(i-1) + 303)),
     c(3:9)] <- coef(temp2)$STUDY

  # Imputation 3
  temp3 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 3)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM1[c((3*(i-1) + 601):(3*(i-1) + 603)),
     c(3:9)] <- coef(temp3)$STUDY

  # Imputation 4
  temp4 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 4)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM1[c((3*(i-1) + 901):(3*(i-1) + 903)),
     c(3:9)] <- coef(temp4)$STUDY

  # Imputation 5
  temp5 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 5)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM1[c((3*(i-1) + 1201):(3*(i-1) + 1203)
     ),c(3:9)] <- coef(temp5)$STUDY
}

# MEMORY2
FCS2stgNoOverlap_coefsMEM2 <- data.frame("IMPNUM" = as.integer(
   rep(c(1:5), each = 300)),
                                         "SIMNUM" = as.integer(
                                            rep(rep(c(1:SIM),
                                            each = 3)), times =
                                            3),
```

```
                                        "Intercept" = numeric
                                            (1500),
                                        "EDU2" = numeric(1500),
                                        "EDU3" = numeric(1500),
                                        "SEX1" = numeric(1500),
                                        "AGE" = numeric(1500),
                                        "PHYS2" = numeric(1500)
                                            ,
                                        "PHYS3" = numeric(1500)
                                            )

for (i in 1:SIM) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
                     FCS2stgNoOverlap$SIMNUM == i) & (
                     FCS2stgNoOverlap$IMPNUM == 1)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM2[c((3*(i-1) + 1):(3*(i-1) + 3)),c
      (3:9)] <- coef(temp1)$STUDY

  # Imputation 2
  temp2 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
                     FCS2stgNoOverlap$SIMNUM == i) & (
                     FCS2stgNoOverlap$IMPNUM == 2)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM2[c((3*(i-1) + 301):(3*(i-1) + 303)),
      c(3:9)] <- coef(temp2)$STUDY

  # Imputation 3
  temp3 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
                     FCS2stgNoOverlap$SIMNUM == i) & (
                     FCS2stgNoOverlap$IMPNUM == 3)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM2[c((3*(i-1) + 601):(3*(i-1) + 603)),
      c(3:9)] <- coef(temp3)$STUDY
```

```
  # Imputation 4
  temp4 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 4)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM2[c((3*(i-1) + 901):(3*(i-1) + 903)),
     c(3:9)] <- coef(temp4)$STUDY

  # Imputation 5
  temp5 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 5)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM2[c((3*(i-1) + 1201):(3*(i-1) + 1203)
     ),c(3:9)] <- coef(temp5)$STUDY
}

# MEMORY3
FCS2stgNoOverlap_coefsMEM3 <- data.frame("IMPNUM" = as.integer(
   rep(c(1:5), each = 300)),
                                         "SIMNUM" = as.integer(
                                            rep(rep(c(1:SIM),
                                            each = 3)), times =
                                            3),
                                         "Intercept" = numeric
                                            (1500),
                                         "EDU2" = numeric(1500),
                                         "EDU3" = numeric(1500),
                                         "SEX1" = numeric(1500),
                                         "AGE" = numeric(1500),
                                         "PHYS2" = numeric(1500)
                                            ,
                                         "PHYS3" = numeric(1500)
                                            )

for (i in 1:SIM) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgNoOverlap[((
```

```
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 1)), ],
                family = binomial)

# Set values to the generated coefficient numbers
FCS2stgNoOverlap_coefsMEM3[c((3*(i-1) + 1):(3*(i-1) + 3)),c
    (3:9)] <- coef(temp1)$STUDY

# Imputation 2
temp2 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                data = FCS2stgNoOverlap[((
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 2)), ],
                family = binomial)

# Set values to the generated coefficient numbers
FCS2stgNoOverlap_coefsMEM3[c((3*(i-1) + 301):(3*(i-1) + 303)),
    c(3:9)] <- coef(temp2)$STUDY

# Imputation 3
temp3 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                data = FCS2stgNoOverlap[((
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 3)), ],
                family = binomial)

# Set values to the generated coefficient numbers
FCS2stgNoOverlap_coefsMEM3[c((3*(i-1) + 601):(3*(i-1) + 603)),
    c(3:9)] <- coef(temp3)$STUDY

# Imputation 4
temp4 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                data = FCS2stgNoOverlap[((
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 4)), ],
                family = binomial)

# Set values to the generated coefficient numbers
FCS2stgNoOverlap_coefsMEM3[c((3*(i-1) + 901):(3*(i-1) + 903)),
    c(3:9)] <- coef(temp4)$STUDY

# Imputation 5
temp5 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                data = FCS2stgNoOverlap[((
```

```
                    FCS2stgNoOverlap$SIMNUM == i) & (
                    FCS2stgNoOverlap$IMPNUM == 5)), ],
                  family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgNoOverlap_coefsMEM3[c((3*(i-1) + 1201):(3*(i-1) + 1203)
    ),c(3:9)] <- coef(temp5)$STUDY
}


# FCS 2 stage WITH overlap (no MEMORY3)
  ------------------------------------------------------

# MEM1
FCS2stgOverlap_coefsMEM1 <- data.frame("IMPNUM" = as.integer(rep
    (c(1:5), each = 300)),
                                       "SIMNUM" = as.integer(rep(
                                          rep(c(1:SIM), each = 3)
                                          ), times = 3),
                                       "Intercept" = numeric
                                          (1500),
                                       "EDU2" = numeric(1500),
                                       "EDU3" = numeric(1500),
                                       "SEX1" = numeric(1500),
                                       "AGE" = numeric(1500),
                                       "PHYS2" = numeric(1500),
                                       "PHYS3" = numeric(1500))

for (i in 1:SIM) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                    == i) & (FCS2stgOverlap$IMPNUM == 1)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM1[c((3*(i-1) + 1):(3*(i-1) + 3)),c(3:9)
     ] <- coef(temp1)$STUDY

  # Imputation 2
  temp2 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                    == i) & (FCS2stgOverlap$IMPNUM == 2)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM1[c((3*(i-1) + 301):(3*(i-1) + 303)),c
```

```
    (3:9)] <- coef(temp2)$STUDY

  # Imputation 3
  temp3 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                   == i) & (FCS2stgOverlap$IMPNUM == 3)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM1[c((3*(i-1) + 601):(3*(i-1) + 603)),c
    (3:9)] <- coef(temp3)$STUDY

  # Imputation 4
  temp4 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                   == i) & (FCS2stgOverlap$IMPNUM == 4)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM1[c((3*(i-1) + 901):(3*(i-1) + 903)),c
    (3:9)] <- coef(temp4)$STUDY

  # Imputation 5
  temp5 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                   == i) & (FCS2stgOverlap$IMPNUM == 5)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM1[c((3*(i-1) + 1201):(3*(i-1) + 1203)),
    c(3:9)] <- coef(temp5)$STUDY
}

# MEMORY2
FCS2stgOverlap_coefsMEM2 <- data.frame("IMPNUM" = as.integer(rep
  (c(1:5), each = 300)),
                                       "SIMNUM" = as.integer(rep(
                                         rep(c(1:SIM), each = 3)
                                         ), times = 3),
                                       "Intercept" = numeric
                                         (1500),
                                       "EDU2" = numeric(1500),
                                       "EDU3" = numeric(1500),
                                       "SEX1" = numeric(1500),
                                       "AGE" = numeric(1500),
```

```
                                        "PHYS2" = numeric(1500),
                                        "PHYS3" = numeric(1500))

for (i in 1:SIM) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                    == i) & (FCS2stgOverlap$IMPNUM == 1)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM2[c((3*(i-1) + 1):(3*(i-1) + 3)),c(3:9)
     ] <- coef(temp1)$STUDY

  # Imputation 2
  temp2 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                    == i) & (FCS2stgOverlap$IMPNUM == 2)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM2[c((3*(i-1) + 301):(3*(i-1) + 303)),c
     (3:9)] <- coef(temp2)$STUDY

  # Imputation 3
  temp3 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                    == i) & (FCS2stgOverlap$IMPNUM == 3)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM2[c((3*(i-1) + 601):(3*(i-1) + 603)),c
     (3:9)] <- coef(temp3)$STUDY

  # Imputation 4
  temp4 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                    == i) & (FCS2stgOverlap$IMPNUM == 4)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM2[c((3*(i-1) + 901):(3*(i-1) + 903)),c
     (3:9)] <- coef(temp4)$STUDY
```

```
  # Imputation 5
  temp5 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = FCS2stgOverlap[((FCS2stgOverlap$SIMNUM
                    == i) & (FCS2stgOverlap$IMPNUM == 5)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  FCS2stgOverlap_coefsMEM2[c((3*(i-1) + 1201):(3*(i-1) + 1203)),
     c(3:9)] <- coef(temp5)$STUDY
}

# jomo No Overlap
-------------------------------------------------------------------------------



# MEM1
jomoNoOverlap_coefsMEM1 <- data.frame("IMPNUM" = as.integer(rep(
    c(1:5), each = 300)),
                                      "SIMNUM" = as.integer(rep(
                                         rep(c(1:SIM), each = 3)
                                         ), times = 3),
                                      "Intercept" = numeric
                                         (1500),
                                      "EDU2" = numeric(1500),
                                      "EDU3" = numeric(1500),
                                      "SEX1" = numeric(1500),
                                      "AGE" = numeric(1500),
                                      "PHYS2" = numeric(1500),
                                      "PHYS3" = numeric(1500))

for (i in 1:SIM) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                    i) & (jomoNoOverlap$IMPNUM == 1)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM1[c((3*(i-1) + 1):(3*(i-1) + 3)),c(3:9)]
      <- coef(temp1)$STUDY

  # Imputation 2
  temp2 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                    i) & (jomoNoOverlap$IMPNUM == 2)), ],
```

```
                    family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM1[c((3*(i-1) + 301):(3*(i-1) + 303)),c
      (3:9)] <- coef(temp2)$STUDY

  # Imputation 3
  temp3 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                    data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                        i) & (jomoNoOverlap$IMPNUM == 3)), ],
                    family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM1[c((3*(i-1) + 601):(3*(i-1) + 603)),c
      (3:9)] <- coef(temp3)$STUDY

  # Imputation 4
  temp4 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                    data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                        i) & (jomoNoOverlap$IMPNUM == 4)), ],
                    family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM1[c((3*(i-1) + 901):(3*(i-1) + 903)),c
      (3:9)] <- coef(temp4)$STUDY

  # Imputation 5
  temp5 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                    data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                        i) & (jomoNoOverlap$IMPNUM == 5)), ],
                    family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM1[c((3*(i-1) + 1201):(3*(i-1) + 1203)),c
      (3:9)] <- coef(temp5)$STUDY
}

# MEMORY2
jomoNoOverlap_coefsMEM2 <- data.frame("IMPNUM" = as.integer(rep(
    c(1:5), each = 300)),
                                      "SIMNUM" = as.integer(rep(
                                          rep(c(1:SIM), each = 3)
                                          ), times = 3),
                                      "Intercept" = numeric
                                          (1500),
```

```
                                        "EDU2" = numeric(1500),
                                        "EDU3" = numeric(1500),
                                        "SEX1" = numeric(1500),
                                        "AGE" = numeric(1500),
                                        "PHYS2" = numeric(1500),
                                        "PHYS3" = numeric(1500))

for (i in 1:SIM) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                    i) & (jomoNoOverlap$IMPNUM == 1)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM2[c((3*(i-1) + 1):(3*(i-1) + 3)),c(3:9)]
      <- coef(temp1)$STUDY

  # Imputation 2
  temp2 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                    i) & (jomoNoOverlap$IMPNUM == 2)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM2[c((3*(i-1) + 301):(3*(i-1) + 303)),c
     (3:9)] <- coef(temp2)$STUDY

  # Imputation 3
  temp3 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                    i) & (jomoNoOverlap$IMPNUM == 3)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM2[c((3*(i-1) + 601):(3*(i-1) + 603)),c
     (3:9)] <- coef(temp3)$STUDY

  # Imputation 4
  temp4 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                 data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                    i) & (jomoNoOverlap$IMPNUM == 4)), ],
                 family = binomial)
```

```
  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM2[c((3*(i-1) + 901):(3*(i-1) + 903)),c
      (3:9)] <- coef(temp4)$STUDY

  # Imputation 5
  temp5 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                 data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                     i) & (jomoNoOverlap$IMPNUM == 5)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM2[c((3*(i-1) + 1201):(3*(i-1) + 1203)),c
      (3:9)] <- coef(temp5)$STUDY
}

# MEMORY3
jomoNoOverlap_coefsMEM3 <- data.frame("IMPNUM" = as.integer(rep(
    c(1:5), each = 300)),
                                      "SIMNUM" = as.integer(rep(
                                          rep(c(1:SIM), each = 3)
                                          ), times = 3),
                                      "Intercept" = numeric
                                          (1500),
                                      "EDU2" = numeric(1500),
                                      "EDU3" = numeric(1500),
                                      "SEX1" = numeric(1500),
                                      "AGE" = numeric(1500),
                                      "PHYS2" = numeric(1500),
                                      "PHYS3" = numeric(1500))

for (i in 1:SIM) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                 data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                     i) & (jomoNoOverlap$IMPNUM == 1)), ],
                 family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM3[c((3*(i-1) + 1):(3*(i-1) + 3)),c(3:9)]
      <- coef(temp1)$STUDY

  # Imputation 2
  temp2 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
      PHYS + (1 | STUDY),
                 data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                     i) & (jomoNoOverlap$IMPNUM == 2)), ],
```

```
                    family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM3[c((3*(i-1) + 301):(3*(i-1) + 303)),c
     (3:9)] <- coef(temp2)$STUDY

  # Imputation 3
  temp3 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                    data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                       i) & (jomoNoOverlap$IMPNUM == 3)), ],
                    family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM3[c((3*(i-1) + 601):(3*(i-1) + 603)),c
     (3:9)] <- coef(temp3)$STUDY

  # Imputation 4
  temp4 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                    data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                       i) & (jomoNoOverlap$IMPNUM == 4)), ],
                    family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM3[c((3*(i-1) + 901):(3*(i-1) + 903)),c
     (3:9)] <- coef(temp4)$STUDY

  # Imputation 5
  temp5 <- glmer(cbind(MEMORY3, M3-MEMORY3) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                    data = jomoNoOverlap[((jomoNoOverlap$SIMNUM ==
                       i) & (jomoNoOverlap$IMPNUM == 5)), ],
                    family = binomial)

  # Set values to the generated coefficient numbers
  jomoNoOverlap_coefsMEM3[c((3*(i-1) + 1201):(3*(i-1) + 1203)),c
     (3:9)] <- coef(temp5)$STUDY
}


# jomo WITH overlap
  ------------------------------------------------------------

# MEM1
jomoOverlap_coefsMEM1 <- data.frame("IMPNUM" = as.integer(rep(c
    (1:5), each = 300)),
                                    "SIMNUM" = as.integer(rep
                                       (rep(c(1:SIM), each =
```

```
                                                    3)), times = 3),
                                                "Intercept" = numeric
                                                    (1500),
                                                "EDU2" = numeric(1500),
                                                "EDU3" = numeric(1500),
                                                "SEX1" = numeric(1500),
                                                "AGE" = numeric(1500),
                                                "PHYS2" = numeric(1500),
                                                "PHYS3" = numeric(1500))

for (i in 1:9) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                  data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
                      (jomoOverlap$IMPNUM == 1)), ],
                  family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM1[c((3*(i-1) + 1):(3*(i-1) + 3)),c(3:9)]
     <- coef(temp1)$STUDY

  # Imputation 2
  temp2 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                  data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
                      (jomoOverlap$IMPNUM == 2)), ],
                  family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM1[c((3*(i-1) + 301):(3*(i-1) + 303)),c
     (3:9)] <- coef(temp2)$STUDY

  # Imputation 3
  temp3 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                  data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
                      (jomoOverlap$IMPNUM == 3)), ],
                  family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM1[c((3*(i-1) + 601):(3*(i-1) + 603)),c
     (3:9)] <- coef(temp3)$STUDY

  # Imputation 4
  temp4 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                  data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
```

```
                    (jomoOverlap$IMPNUM == 4)), ],
                  family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM1[c((3*(i-1) + 901):(3*(i-1) + 903)),c
     (3:9)] <- coef(temp4)$STUDY

  # Imputation 5
  temp5 <- glmer(cbind(MEMORY1, M1-MEMORY1) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                  data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
                      (jomoOverlap$IMPNUM == 5)), ],
                  family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM1[c((3*(i-1) + 1201):(3*(i-1) + 1203)),c
     (3:9)] <- coef(temp5)$STUDY
}

# MEMORY2
jomoOverlap_coefsMEM2 <- data.frame("IMPNUM" = as.integer(rep(c
    (1:5), each = 300)),
                                    "SIMNUM" = as.integer(rep
                                       (rep(c(1:SIM), each =
                                       3)), times = 3),
                                    "Intercept" = numeric
                                       (1500),
                                    "EDU2" = numeric(1500),
                                    "EDU3" = numeric(1500),
                                    "SEX1" = numeric(1500),
                                    "AGE" = numeric(1500),
                                    "PHYS2" = numeric(1500),
                                    "PHYS3" = numeric(1500))

for (i in 1:SIM) {
  # Imputation 1
  temp1 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
     PHYS + (1 | STUDY),
                  data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
                      (jomoOverlap$IMPNUM == 1)), ],
                  family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM2[c((3*(i-1) + 1):(3*(i-1) + 3)),c(3:9)]
     <- coef(temp1)$STUDY

  # Imputation 2
  temp2 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
```

```
    PHYS + (1 | STUDY),
                data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
                    (jomoOverlap$IMPNUM == 2)), ],
                family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM2[c((3*(i-1) + 301):(3*(i-1) + 303)),c
    (3:9)] <- coef(temp2)$STUDY

  # Imputation 3
  temp3 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
                    (jomoOverlap$IMPNUM == 3)), ],
                family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM2[c((3*(i-1) + 601):(3*(i-1) + 603)),c
    (3:9)] <- coef(temp3)$STUDY

  # Imputation 4
  temp4 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
                    (jomoOverlap$IMPNUM == 4)), ],
                family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM2[c((3*(i-1) + 901):(3*(i-1) + 903)),c
    (3:9)] <- coef(temp4)$STUDY

  # Imputation 5
  temp5 <- glmer(cbind(MEMORY2, M2-MEMORY2) ~ EDU + SEX + AGE +
    PHYS + (1 | STUDY),
                data = jomoOverlap[((jomoOverlap$SIMNUM == i) &
                    (jomoOverlap$IMPNUM == 5)), ],
                family = binomial)

  # Set values to the generated coefficient numbers
  jomoOverlap_coefsMEM2[c((3*(i-1) + 1201):(3*(i-1) + 1203)),c
    (3:9)] <- coef(temp5)$STUDY
}
```

A.7 Evaluation of Imputation

```
library(dplyr)
library(lme4)
# Load each imputed data set
```

```
# FCS 2 stage without overlap
FCS2stgNoOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/FCS2stgNoOverlap.csv")
FCS2stgNoOverlap$SIMNUM <- as.factor(FCS2stgNoOverlap$SIMNUM)
FCS2stgNoOverlap$STUDY <- as.factor(FCS2stgNoOverlap$STUDY)
FCS2stgNoOverlap$SUBJECTNUM <- as.integer(
    FCS2stgNoOverlap$SUBJECTNUM)
FCS2stgNoOverlap$SEX <- as.factor(FCS2stgNoOverlap$SEX)
FCS2stgNoOverlap$EDU <- as.factor(FCS2stgNoOverlap$EDU)
FCS2stgNoOverlap$PHYS <- as.factor(FCS2stgNoOverlap$PHYS)
FCS2stgNoOverlap$MEMORY1 <- as.integer(FCS2stgNoOverlap$MEMORY1)
FCS2stgNoOverlap$MEMORY2 <- as.integer(FCS2stgNoOverlap$MEMORY2)
FCS2stgNoOverlap$MEMORY3 <- as.integer(FCS2stgNoOverlap$MEMORY3)

# FCS 2 stage with overlap
FCS2stgOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/FCS2stgOverlap.csv")
FCS2stgOverlap$SIMNUM <- as.factor(FCS2stgOverlap$SIMNUM)
FCS2stgOverlap$STUDY <- as.factor(FCS2stgOverlap$STUDY)
FCS2stgOverlap$SUBJECTNUM <- as.integer(
    FCS2stgOverlap$SUBJECTNUM)
FCS2stgOverlap$SEX <- as.factor(FCS2stgOverlap$SEX)
FCS2stgOverlap$EDU <- as.factor(FCS2stgOverlap$EDU)
FCS2stgOverlap$PHYS <- as.factor(FCS2stgOverlap$PHYS)
FCS2stgOverlap$MEMORY1 <- as.integer(FCS2stgOverlap$MEMORY1)
FCS2stgOverlap$MEMORY2 <- as.integer(FCS2stgOverlap$MEMORY2)

# ------------------------------------
# Use for both jomo data sets
SIM=100
N1=10000
N2=2000
N3=500
N = N1 + N2 + N3
SIMNUM <- as.integer(rep(rep(1:SIM, each=N), each = 5))

# jomo without overlap
jomoNoOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/imputedDatasets_jomoNoOverlap.csv")
jomoNoOverlap$clus <- as.factor(jomoNoOverlap$clus)
colnames(jomoNoOverlap)[11] <- "STUDY"
jomoNoOverlap$id <- as.integer(jomoNoOverlap$id)
colnames(jomoNoOverlap)[12] <- "SUBJECTNUM"
jomoNoOverlap$SEX <- as.factor(jomoNoOverlap$SEX)
jomoNoOverlap$EDU <- as.factor(jomoNoOverlap$EDU)
jomoNoOverlap$PHYS <- as.factor(jomoNoOverlap$PHYS)
# rounding MEMORY1,2,3 to integers after they have been imputed
```

```
    as numeric
jomoNoOverlap$MEMORY1 <- as.integer(round(jomoNoOverlap$MEMORY1,
    digits = 0))
jomoNoOverlap$MEMORY2 <- as.integer(round(jomoNoOverlap$MEMORY2,
    digits = 0))
jomoNoOverlap$MEMORY3 <- as.integer(round(jomoNoOverlap$MEMORY3,
    digits = 0))
jomoNoOverlap$X <- NULL
jomoNoOverlap$X1 <- NULL
jomoNoOverlap$Z1 <- NULL
jomoNoOverlap <- jomoNoOverlap[!(jomoNoOverlap$Imputation == 0),
    ]
colnames(jomoNoOverlap)[10] <- "IMPNUM"
jomoNoOverlap <- cbind(jomoNoOverlap,SIMNUM)
head(jomoNoOverlap)

# jomo with overlap
jomoOverlap <- read.csv("/Users/catherinehilgers/Documents/
   Thesis/R Code/imputedDatasets_jomoOverlap.csv")
jomoOverlap$clus <- as.factor(jomoOverlap$clus)
colnames(jomoOverlap)[10] <- "STUDY"
jomoOverlap$id <- as.integer(jomoOverlap$id)
colnames(jomoOverlap)[11] <- "SUBJECTNUM"
jomoOverlap$SEX <- as.factor(jomoOverlap$SEX)
jomoOverlap$EDU <- as.factor(jomoOverlap$EDU)
jomoOverlap$PHYS <- as.factor(jomoOverlap$PHYS)
jomoOverlap$MEMORY1 <- as.integer(jomoOverlap$MEMORY1)
jomoOverlap$MEMORY2 <- as.integer(jomoOverlap$MEMORY2)
jomoOverlap$X <- NULL
jomoOverlap$X1 <- NULL
jomoOverlap$Z1 <- NULL
jomoOverlap <- jomoOverlap[!(jomoOverlap$Imputation == 0), ]
colnames(jomoOverlap)[9] <- "IMPNUM"
jomoOverlap <- cbind(jomoOverlap,SIMNUM)
head(jomoOverlap)

# ------------------------------------
# Load complete data set

complete <- read.csv("/Users/catherinehilgers/Documents/Thesis/R
    Code/simulatedDataset.csv")
complete$SIMNUM <- as.factor(complete$SIMNUM)
complete$STUDY <- as.factor(complete$STUDY)
complete$SUBJECTNUM <- as.integer(complete$SUBJECTNUM)
complete$SEX <- as.factor(complete$SEX)
complete$EDU <- as.factor(complete$EDU)
complete$PHYS <- as.factor(complete$PHYS)
complete$MEMORY1 <- as.integer(complete$MEMORY1)
```

```
complete$MEMORY2 <- as.integer(complete$MEMORY2)
complete$MEMORY3 <- as.integer(complete$MEMORY3)

# ------------------------------------
# DIFFERENCE CHECKS

# Check imputed vs. true for the imputed data sets

# NO OVERLAP CASE ----------------
# FCS2stgNoOverlap ----------------
# MEMORY1 is sys missing for studies 2,3
# Average over 100 simulations, 5 imputations

# Study 2
FCS2stgNoOverlap_MEM1Study2 <- data.frame("IMP" = integer(5), "
   TRUE" = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- FCS2stgNoOverlap[((FCS2stgNoOverlap$IMPNUM == i) &
                          (FCS2stgNoOverlap$STUDY == 2)), ]$MEMORY1
                            == complete[complete$STUDY == 2, ]
                            $MEMORY1
  temp <- as.data.frame(table(temp))
  FCS2stgNoOverlap_MEM1Study2[i,1] <- i
  FCS2stgNoOverlap_MEM1Study2[i,2] <- temp[2,2]/100
  FCS2stgNoOverlap_MEM1Study2[i,3] <- temp[1,2]/100
}

mean(FCS2stgNoOverlap_MEM1Study2$TRUE.)
mean(FCS2stgNoOverlap_MEM1Study2$FALSE.)

# Study 3
FCS2stgNoOverlap_MEM1Study3 <- data.frame("IMP" = integer(5), "
   TRUE" = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- FCS2stgNoOverlap[((FCS2stgNoOverlap$IMPNUM == i) &
                                (FCS2stgNoOverlap$STUDY == 3)), ]
                                  $MEMORY1 == complete[
                                  complete$STUDY == 3, ]$MEMORY1
  temp <- as.data.frame(table(temp))
  FCS2stgNoOverlap_MEM1Study3[i,1] <- i
  FCS2stgNoOverlap_MEM1Study3[i,2] <- temp[2,2]/100
  FCS2stgNoOverlap_MEM1Study3[i,3] <- temp[1,2]/100
}

mean(FCS2stgNoOverlap_MEM1Study3$TRUE.)
mean(FCS2stgNoOverlap_MEM1Study3$FALSE.)

# MEMORY2 is sys missing for studies 1,3
```

```
# Average over 100 simulations, 5 imputations
# FCS2stgNoOverlap
# Study 1
FCS2stgNoOverlap_MEM2Study1 <- data.frame("IMP" = integer(5), "
   TRUE" = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- FCS2stgNoOverlap[((FCS2stgNoOverlap$IMPNUM == i) &
                              (FCS2stgNoOverlap$STUDY == 1)), ]
                                $MEMORY2 == complete[
                                complete$STUDY == 1, ]$MEMORY2
  temp <- as.data.frame(table(temp))
  FCS2stgNoOverlap_MEM2Study1[i,1] <- i
  FCS2stgNoOverlap_MEM2Study1[i,2] <- temp[2,2]/100
  FCS2stgNoOverlap_MEM2Study1[i,3] <- temp[1,2]/100
}

mean(FCS2stgNoOverlap_MEM2Study1$TRUE.)
mean(FCS2stgNoOverlap_MEM2Study1$FALSE.)

# Study 3
FCS2stgNoOverlap_MEM2Study3 <- data.frame("IMP" = integer(5), "
   TRUE" = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- FCS2stgNoOverlap[((FCS2stgNoOverlap$IMPNUM == i) &
                              (FCS2stgNoOverlap$STUDY == 3)), ]
                                $MEMORY2 == complete[
                                complete$STUDY == 3, ]$MEMORY2
  temp <- as.data.frame(table(temp))
  FCS2stgNoOverlap_MEM2Study3[i,1] <- i
  FCS2stgNoOverlap_MEM2Study3[i,2] <- temp[2,2]/100
  FCS2stgNoOverlap_MEM2Study3[i,3] <- temp[1,2]/100
}

mean(FCS2stgNoOverlap_MEM2Study3$TRUE.)
mean(FCS2stgNoOverlap_MEM2Study3$FALSE.)

# MEMORY3 is sys missing for studies 1,2
# Average over 100 simulations, 5 imputations
# FCS2stgNoOverlap
# Study 1
FCS2stgNoOverlap_MEM3Study1 <- data.frame("IMP" = integer(5), "
   TRUE" = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- FCS2stgNoOverlap[((FCS2stgNoOverlap$IMPNUM == i) &
                              (FCS2stgNoOverlap$STUDY == 1)), ]
                                $MEMORY3 == complete[
                                complete$STUDY == 1, ]$MEMORY3
  temp <- as.data.frame(table(temp))
```

```
  FCS2stgNoOverlap_MEM3Study1[i,1] <- i
  FCS2stgNoOverlap_MEM3Study1[i,2] <- temp[2,2]/100
  FCS2stgNoOverlap_MEM3Study1[i,3] <- temp[1,2]/100
}

mean(FCS2stgNoOverlap_MEM3Study1$TRUE.)
mean(FCS2stgNoOverlap_MEM3Study1$FALSE.)

# Study 2
FCS2stgNoOverlap_MEM3Study2 <- data.frame("IMP" = integer(5), "
   TRUE" = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- FCS2stgNoOverlap[((FCS2stgNoOverlap$IMPNUM == i) &
                            (FCS2stgNoOverlap$STUDY == 2)), ]
                              $MEMORY3 == complete[
                              complete$STUDY == 2, ]$MEMORY3
  temp <- as.data.frame(table(temp))
  FCS2stgNoOverlap_MEM3Study2[i,1] <- i
  FCS2stgNoOverlap_MEM3Study2[i,2] <- temp[2,2]/100
  FCS2stgNoOverlap_MEM3Study2[i,3] <- temp[1,2]/100
}

mean(FCS2stgNoOverlap_MEM3Study2$TRUE.)
mean(FCS2stgNoOverlap_MEM3Study2$FALSE.)

# jomoNoOverlap ----------------
# MEMORY1 is sys missing for studies 2,3
# Average over 100 simulations, 5 imputations

# Study 2
jomoNoOverlap_MEM1Study2 <- data.frame("IMP" = integer(5), "TRUE
   " = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- jomoNoOverlap[((jomoNoOverlap$IMPNUM == i) &
                         (jomoNoOverlap$STUDY == 2)), ]
                           $MEMORY1 == complete[
                           complete$STUDY == 2, ]$MEMORY1
  temp <- as.data.frame(table(temp))
  jomoNoOverlap_MEM1Study2[i,1] <- i
  jomoNoOverlap_MEM1Study2[i,2] <- temp[2,2]/100
  jomoNoOverlap_MEM1Study2[i,3] <- temp[1,2]/100
}

mean(jomoNoOverlap_MEM1Study2$TRUE.)
mean(jomoNoOverlap_MEM1Study2$FALSE.)

# Study 3
jomoNoOverlap_MEM1Study3 <- data.frame("IMP" = integer(5), "TRUE
```

```
    " = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- jomoNoOverlap[((jomoNoOverlap$IMPNUM == i) &
                                  (jomoNoOverlap$STUDY == 3)), ]
                                    $MEMORY1 == complete[
                                    complete$STUDY == 3, ]$MEMORY1
  temp <- as.data.frame(table(temp))
  jomoNoOverlap_MEM1Study3[i,1] <- i
  jomoNoOverlap_MEM1Study3[i,2] <- temp[2,2]/100
  jomoNoOverlap_MEM1Study3[i,3] <- temp[1,2]/100
}

mean(jomoNoOverlap_MEM1Study3$TRUE.)
# 58.022
mean(jomoNoOverlap_MEM1Study3$FALSE.)
# 441.978

# MEMORY2 is sys missing for studies 1,3
# Average over 100 simulations, 5 imputations
# jomoNoOverlap
# Study 1
jomoNoOverlap_MEM2Study1 <- data.frame("IMP" = integer(5), "TRUE
    " = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- jomoNoOverlap[((jomoNoOverlap$IMPNUM == i) &
                                  (jomoNoOverlap$STUDY == 1)), ]
                                    $MEMORY2 == complete[
                                    complete$STUDY == 1, ]$MEMORY2
  temp <- as.data.frame(table(temp))
  jomoNoOverlap_MEM2Study1[i,1] <- i
  jomoNoOverlap_MEM2Study1[i,2] <- temp[2,2]/100
  jomoNoOverlap_MEM2Study1[i,3] <- temp[1,2]/100
}

mean(jomoNoOverlap_MEM2Study1$TRUE.)
mean(jomoNoOverlap_MEM2Study1$FALSE.)

# Study 3
jomoNoOverlap_MEM2Study3 <- data.frame("IMP" = integer(5), "TRUE
    " = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- jomoNoOverlap[((jomoNoOverlap$IMPNUM == i) &
                                  (jomoNoOverlap$STUDY == 3)), ]
                                    $MEMORY2 == complete[
                                    complete$STUDY == 3, ]$MEMORY2
  temp <- as.data.frame(table(temp))
  jomoNoOverlap_MEM2Study3[i,1] <- i
  jomoNoOverlap_MEM2Study3[i,2] <- temp[2,2]/100
```

```
  jomoNoOverlap_MEM2Study3[i,3] <- temp[1,2]/100
}

mean(jomoNoOverlap_MEM2Study3$TRUE.)
mean(jomoNoOverlap_MEM2Study3$FALSE.)

# MEMORY3 is sys missing for studies 1,2
# Average over 100 simulations, 5 imputations
# jomoNoOverlap
# Study 1
jomoNoOverlap_MEM3Study1 <- data.frame("IMP" = integer(5), "TRUE
   " = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- jomoNoOverlap[((jomoNoOverlap$IMPNUM == i) &
                                  (jomoNoOverlap$STUDY == 1)), ]
                                     $MEMORY3 == complete[
                                     complete$STUDY == 1, ]$MEMORY3
  temp <- as.data.frame(table(temp))
  jomoNoOverlap_MEM3Study1[i,1] <- i
  jomoNoOverlap_MEM3Study1[i,2] <- temp[2,2]/100
  jomoNoOverlap_MEM3Study1[i,3] <- temp[1,2]/100
}

mean(jomoNoOverlap_MEM3Study1$TRUE.)
mean(jomoNoOverlap_MEM3Study1$FALSE.)

# Study 2
jomoNoOverlap_MEM3Study2 <- data.frame("IMP" = integer(5), "TRUE
   " = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- jomoNoOverlap[((jomoNoOverlap$IMPNUM == i) &
                                  (jomoNoOverlap$STUDY == 2)), ]
                                     $MEMORY3 == complete[
                                     complete$STUDY == 2, ]$MEMORY3
  temp <- as.data.frame(table(temp))
  jomoNoOverlap_MEM3Study2[i,1] <- i
  jomoNoOverlap_MEM3Study2[i,2] <- temp[2,2]/100
  jomoNoOverlap_MEM3Study2[i,3] <- temp[1,2]/100
}

mean(jomoNoOverlap_MEM3Study2$TRUE.)
mean(jomoNoOverlap_MEM3Study2$FALSE.)


# OVERLAP CASE ----------------
# FCS2stgOverlap ----------------
# MEMORY1 is sys missing for studies 3
# Average over 100 simulations, 5 imputations
```

```
# Study 3
FCS2stgOverlap_MEM1Study3 <- data.frame("IMP" = integer(5), "
   TRUE" = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- FCS2stgOverlap[((FCS2stgOverlap$IMPNUM == i) &
                          (FCS2stgOverlap$STUDY == 3)), ]
                            $MEMORY1 == complete[
                            complete$STUDY == 3, ]$MEMORY1
  temp <- as.data.frame(table(temp))
  FCS2stgOverlap_MEM1Study3[i,1] <- i
  FCS2stgOverlap_MEM1Study3[i,2] <- temp[2,2]/100
  FCS2stgOverlap_MEM1Study3[i,3] <- temp[1,2]/100
}

mean(FCS2stgOverlap_MEM1Study3$TRUE.)
mean(FCS2stgOverlap_MEM1Study3$FALSE.)

# MEMORY2 is sys missing for study 2
# Study 2
FCS2stgOverlap_MEM2Study2 <- data.frame("IMP" = integer(5), "
   TRUE" = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- FCS2stgOverlap[((FCS2stgOverlap$IMPNUM == i) &
                          (FCS2stgOverlap$STUDY == 2)), ]
                            $MEMORY2 == complete[
                            complete$STUDY == 2, ]$MEMORY2
  temp <- as.data.frame(table(temp))
  FCS2stgOverlap_MEM2Study2[i,1] <- i
  FCS2stgOverlap_MEM2Study2[i,2] <- temp[2,2]/100
  FCS2stgOverlap_MEM2Study2[i,3] <- temp[1,2]/100
}

mean(FCS2stgOverlap_MEM2Study2$TRUE.)
mean(FCS2stgOverlap_MEM2Study2$FALSE.)



# jomoOverlap ----------------
# MEMORY1 is sys missing for studies 3
# Average over 100 simulations, 5 imputations

# Study 3
jomoOverlap_MEM1Study3 <- data.frame("IMP" = integer(5), "TRUE"
   = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- jomoOverlap[((jomoOverlap$IMPNUM == i) &
                       (jomoOverlap$STUDY == 3)), ]$MEMORY1
```

```
                                    == complete[complete$STUDY == 3,
                                    ]$MEMORY1
  temp <- as.data.frame(table(temp))
  jomoOverlap_MEM1Study3[i,1] <- i
  jomoOverlap_MEM1Study3[i,2] <- temp[2,2]/100
  jomoOverlap_MEM1Study3[i,3] <- temp[1,2]/100
}

mean(jomoOverlap_MEM1Study3$TRUE.)
mean(jomoOverlap_MEM1Study3$FALSE.)

# MEMORY2 is sys missing for study 2
# Study 2
jomoOverlap_MEM2Study2 <- data.frame("IMP" = integer(5), "TRUE"
   = logical(5), "FALSE" = logical(5))
for (i in 1:5) {
  temp <- jomoOverlap[((jomoOverlap$IMPNUM == i) &
                         (jomoOverlap$STUDY == 2)), ]$MEMORY2
                         == complete[complete$STUDY == 2,
                         ]$MEMORY2
  temp <- as.data.frame(table(temp))
  jomoOverlap_MEM2Study2[i,1] <- i
  jomoOverlap_MEM2Study2[i,2] <- temp[2,2]/100
  jomoOverlap_MEM2Study2[i,3] <- temp[1,2]/100
}

mean(jomoOverlap_MEM2Study2$TRUE.)
mean(jomoOverlap_MEM2Study2$FALSE.)

library(dplyr)

# Load data sets

complete_coefsMEM1 <- read.csv("/Users/catherinehilgers/
   Documents/Thesis/R Code/complete_coefsMEM1.csv")
complete_coefsMEM2 <- read.csv("/Users/catherinehilgers/
   Documents/Thesis/R Code/complete_coefsMEM2.csv")
complete_coefsMEM3 <- read.csv("/Users/catherinehilgers/
   Documents/Thesis/R Code/complete_coefsMEM3.csv")
FCS2stgNoOverlap_coefsMEM1 <- read.csv("/Users/catherinehilgers/
   Documents/Thesis/R Code/FCS2stgNoOverlap_coefsMEM1.csv")
FCS2stgNoOverlap_coefsMEM2 <- read.csv("/Users/catherinehilgers/
   Documents/Thesis/R Code/FCS2stgNoOverlap_coefsMEM2.csv")
FCS2stgNoOverlap_coefsMEM3 <- read.csv("/Users/catherinehilgers/
   Documents/Thesis/R Code/FCS2stgNoOverlap_coefsMEM3.csv")
jomoNoOverlap_coefsMEM1 <- read.csv("/Users/catherinehilgers/
   Documents/Thesis/R Code/jomoNoOverlap_coefsMEM1.csv")
jomoNoOverlap_coefsMEM2 <- read.csv("/Users/catherinehilgers/
```

```
    Documents/Thesis/R Code/jomoNoOverlap_coefsMEM2.csv")
jomoNoOverlap_coefsMEM3 <- read.csv("/Users/catherinehilgers/
    Documents/Thesis/R Code/jomoNoOverlap_coefsMEM3.csv")
FCS2stgOverlap_coefsMEM1 <- read.csv("/Users/catherinehilgers/
    Documents/Thesis/R Code/FCS2stgOverlap_coefsMEM1.csv")
FCS2stgOverlap_coefsMEM2 <- read.csv("/Users/catherinehilgers/
    Documents/Thesis/R Code/FCS2stgOverlap_coefsMEM2.csv")
jomoOverlap_coefsMEM1 <- read.csv("/Users/catherinehilgers/
    Documents/Thesis/R Code/jomoOverlap_coefsMEM1.csv")
jomoOverlap_coefsMEM2 <- read.csv("/Users/catherinehilgers/
    Documents/Thesis/R Code/jomoOverlap_coefsMEM2.csv")

# Erase unnecessary columns
complete_coefsMEM1$X <- NULL
complete_coefsMEM1[, c(11:18)] <- NULL
complete_coefsMEM2$X <- NULL
complete_coefsMEM1 <- complete_coefsMEM1[-c(101,102,203,204,305)
    , ]
complete_coefsMEM2 <- complete_coefsMEM2[-c(101,102,203,204,305)
    , ]
FCS2stgNoOverlap_coefsMEM1$X <- NULL
FCS2stgNoOverlap_coefsMEM2$X <- NULL
FCS2stgNoOverlap_coefsMEM3$X <- NULL
jomoNoOverlap_coefsMEM1$X <- NULL
jomoNoOverlap_coefsMEM2$X <- NULL
jomoNoOverlap_coefsMEM3$X <- NULL
FCS2stgOverlap_coefsMEM1$X <- NULL
FCS2stgOverlap_coefsMEM2$X <- NULL
jomoOverlap_coefsMEM1$X <- NULL
jomoOverlap_coefsMEM2$X <- NULL

# Add the STUDY column
STUDY <- rep(c(1:3), times = 500)

FCS2stgNoOverlap_coefsMEM1$STUDY <- STUDY
FCS2stgNoOverlap_coefsMEM2$STUDY <- STUDY
FCS2stgNoOverlap_coefsMEM3$STUDY <- STUDY
jomoNoOverlap_coefsMEM1$STUDY <- STUDY
jomoNoOverlap_coefsMEM2$STUDY <- STUDY
jomoNoOverlap_coefsMEM3$STUDY <- STUDY
FCS2stgOverlap_coefsMEM1$STUDY <- STUDY
FCS2stgOverlap_coefsMEM2$STUDY <- STUDY
jomoOverlap_coefsMEM1$STUDY <- STUDY
jomoOverlap_coefsMEM2$STUDY <- STUDY

# ------------------------------------------------
# Averages for each STUDY, for complete data set
# MEM1
```

```
colMeans(complete_coefsMEM1[complete_coefsMEM1$STUDY == 1, c
    (3:9)])
colMeans(complete_coefsMEM1[complete_coefsMEM1$STUDY == 2, c
    (3:9)])
colMeans(complete_coefsMEM1[complete_coefsMEM1$STUDY == 3, c
    (3:9)])

# MEM2
colMeans(complete_coefsMEM2[complete_coefsMEM2$STUDY == 1, c
    (3:9)])
colMeans(complete_coefsMEM2[complete_coefsMEM2$STUDY == 2, c
    (3:9)])
colMeans(complete_coefsMEM2[complete_coefsMEM2$STUDY == 3, c
    (3:9)])

# MEM3
colMeans(complete_coefsMEM3[complete_coefsMEM3$STUDY == 1, c
    (3:9)])
colMeans(complete_coefsMEM3[complete_coefsMEM3$STUDY == 2, c
    (3:9)])
colMeans(complete_coefsMEM3[complete_coefsMEM3$STUDY == 3, c
    (3:9)])

# ------------------------------------------------------------
# Create new data frames with average over imputations

# FCS2stgNoOverlap_coefs
FCS2stgNoOverlap_coefs <- data.frame(MEM = rep(c(1:3), each =
    300),
                                     SIMNUM = rep(c(1:100), each
                                         = 3),
                                     STUDY = rep(c(1:3), times =
                                         300),
                                     Intercept = numeric(900),
                                     EDU2 = numeric(900),
                                     EDU3 = numeric(900),
                                     SEX1 = numeric(900),
                                     AGE = numeric(900),
                                     PHYS2 = numeric(900),
                                     PHYS3 = numeric(900))
head(FCS2stgNoOverlap_coefs)

for (i in 1:100) {
  FCS2stgNoOverlap_coefs[(3*(i-1)+1), c(4:10)] <-
    colMeans(FCS2stgNoOverlap_coefsMEM1[((
        FCS2stgNoOverlap_coefsMEM1$SIMNUM == i) &
                                            (
                                                FCS2stgNoOverlap_coefsMEM1$STUDY
```

```
                                                == 1)), c(3:9)])
  FCS2stgNoOverlap_coefs[(3*(i-1)+2), c(4:10)] <-
    colMeans(FCS2stgNoOverlap_coefsMEM1[((
      FCS2stgNoOverlap_coefsMEM1$SIMNUM == i) &
                                          (
                                            FCS2stgNoOverlap_coefsMEM1$S
                                             == 2)), c(3:9)])
  FCS2stgNoOverlap_coefs[(3*(i-1)+3), c(4:10)] <-
    colMeans(FCS2stgNoOverlap_coefsMEM1[((
      FCS2stgNoOverlap_coefsMEM1$SIMNUM == i) &
                                          (
                                            FCS2stgNoOverlap_coefsMEM1$S
                                             == 3)), c(3:9)])
}

for (i in 1:100) {
  FCS2stgNoOverlap_coefs[(3*(i-1)+301), c(4:10)] <-
    colMeans(FCS2stgNoOverlap_coefsMEM2[((
      FCS2stgNoOverlap_coefsMEM2$SIMNUM == i) &
                                          (
                                            FCS2stgNoOverlap_coefsMEM2$S
                                             == 1)), c(3:9)])
  FCS2stgNoOverlap_coefs[(3*(i-1)+302), c(4:10)] <-
    colMeans(FCS2stgNoOverlap_coefsMEM2[((
      FCS2stgNoOverlap_coefsMEM2$SIMNUM == i) &
                                          (
                                            FCS2stgNoOverlap_coefsMEM2$S
                                             == 2)), c(3:9)])
  FCS2stgNoOverlap_coefs[(3*(i-1)+303), c(4:10)] <-
    colMeans(FCS2stgNoOverlap_coefsMEM2[((
      FCS2stgNoOverlap_coefsMEM2$SIMNUM == i) &
                                          (
                                            FCS2stgNoOverlap_coefsMEM2$S
                                             == 3)), c(3:9)])
}

for (i in 1:100) {
  FCS2stgNoOverlap_coefs[(3*(i-1)+601), c(4:10)] <-
    colMeans(FCS2stgNoOverlap_coefsMEM3[((
      FCS2stgNoOverlap_coefsMEM3$SIMNUM == i) &
                                          (
                                            FCS2stgNoOverlap_coefsMEM3$S
                                             == 1)), c(3:9)])
  FCS2stgNoOverlap_coefs[(3*(i-1)+602), c(4:10)] <-
    colMeans(FCS2stgNoOverlap_coefsMEM3[((
      FCS2stgNoOverlap_coefsMEM3$SIMNUM == i) &
                                          (
                                            FCS2stgNoOverlap_coefsMEM3$S
```

```
                                                    == 2)), c(3:9)])
  FCS2stgNoOverlap_coefs[(3*(i-1)+603), c(4:10)] <-
    colMeans(FCS2stgNoOverlap_coefsMEM3[((
       FCS2stgNoOverlap_coefsMEM3$SIMNUM == i) &
                                          (
                                             FCS2stgNoOverlap_coefsMEM3$STUD
                                              == 3)), c(3:9)])
}

# jomoNoOverlap_coefs
jomoNoOverlap_coefs <- data.frame(MEM = rep(c(1:3), each = 300),
                                  SIMNUM = rep(c(1:100), each
                                      = 3),
                                  STUDY = rep(c(1:3), times =
                                      300),
                                  Intercept = numeric(900),
                                  EDU2 = numeric(900),
                                  EDU3 = numeric(900),
                                  SEX1 = numeric(900),
                                  AGE = numeric(900),
                                  PHYS2 = numeric(900),
                                  PHYS3 = numeric(900))
head(jomoNoOverlap_coefs)

for (i in 1:100) {
  jomoNoOverlap_coefs[(3*(i-1)+1), c(4:10)] <-
    colMeans(jomoNoOverlap_coefsMEM1[((
       jomoNoOverlap_coefsMEM1$SIMNUM == i) &
                                          (
                                             jomoNoOverlap_coefsMEM1$STUDY
                                              == 1)), c(3:9)])
  jomoNoOverlap_coefs[(3*(i-1)+2), c(4:10)] <-
    colMeans(jomoNoOverlap_coefsMEM1[((
       jomoNoOverlap_coefsMEM1$SIMNUM == i) &
                                          (
                                             jomoNoOverlap_coefsMEM1$STUDY
                                              == 2)), c(3:9)])
  jomoNoOverlap_coefs[(3*(i-1)+3), c(4:10)] <-
    colMeans(jomoNoOverlap_coefsMEM1[((
       jomoNoOverlap_coefsMEM1$SIMNUM == i) &
                                          (
                                             jomoNoOverlap_coefsMEM1$STUDY
                                              == 3)), c(3:9)])
}

for (i in 1:100) {
  jomoNoOverlap_coefs[(3*(i-1)+301), c(4:10)] <-
    colMeans(jomoNoOverlap_coefsMEM2[((
```

```
        jomoNoOverlap_coefsMEM2$SIMNUM == i) &
                                              (
                                                 jomoNoOverlap_coefsMEM2$STUI
                                                  == 1)), c(3:9)])
  jomoNoOverlap_coefs[(3*(i-1)+302), c(4:10)] <-
    colMeans(jomoNoOverlap_coefsMEM2[((
        jomoNoOverlap_coefsMEM2$SIMNUM == i) &
                                              (
                                                 jomoNoOverlap_coefsMEM2$STUI
                                                  == 2)), c(3:9)])
  jomoNoOverlap_coefs[(3*(i-1)+303), c(4:10)] <-
    colMeans(jomoNoOverlap_coefsMEM2[((
        jomoNoOverlap_coefsMEM2$SIMNUM == i) &
                                              (
                                                 jomoNoOverlap_coefsMEM2$STUI
                                                  == 3)), c(3:9)])
}

for (i in 1:100) {
  jomoNoOverlap_coefs[(3*(i-1)+601), c(4:10)] <-
    colMeans(jomoNoOverlap_coefsMEM3[((
        jomoNoOverlap_coefsMEM3$SIMNUM == i) &
                                              (
                                                 jomoNoOverlap_coefsMEM3$STUI
                                                  == 1)), c(3:9)])
  jomoNoOverlap_coefs[(3*(i-1)+602), c(4:10)] <-
    colMeans(jomoNoOverlap_coefsMEM3[((
        jomoNoOverlap_coefsMEM3$SIMNUM == i) &
                                              (
                                                 jomoNoOverlap_coefsMEM3$STUI
                                                  == 2)), c(3:9)])
  jomoNoOverlap_coefs[(3*(i-1)+603), c(4:10)] <-
    colMeans(jomoNoOverlap_coefsMEM3[((
        jomoNoOverlap_coefsMEM3$SIMNUM == i) &
                                              (
                                                 jomoNoOverlap_coefsMEM3$STUI
                                                  == 3)), c(3:9)])
}

# FCS2stgOverlap
FCS2stgOverlap_coefs <- data.frame(MEM = rep(c(1:2), each = 300)
    ,
                                   SIMNUM = rep(c(1:100), each =
                                       3),
                                   STUDY = rep(c(1:3), times =
                                       200),
                                   Intercept = numeric(600),
                                   EDU2 = numeric(600),
```

```
                                        EDU3 = numeric(600),
                                        SEX1 = numeric(600),
                                        AGE = numeric(600),
                                        PHYS2 = numeric(600),
                                        PHYS3 = numeric(600))
head(FCS2stgOverlap_coefs)

for (i in 1:100) {
  FCS2stgOverlap_coefs[(3*(i-1)+1), c(4:10)] <-
    colMeans(FCS2stgOverlap_coefsMEM1[((
       FCS2stgOverlap_coefsMEM1$SIMNUM == i) &
                                          (
                                             FCS2stgOverlap_coefsMEM1$STUDY
                                              == 1)), c(3:9)])
  FCS2stgOverlap_coefs[(3*(i-1)+2), c(4:10)] <-
    colMeans(FCS2stgOverlap_coefsMEM1[((
       FCS2stgOverlap_coefsMEM1$SIMNUM == i) &
                                          (
                                             FCS2stgOverlap_coefsMEM1$STUDY
                                              == 2)), c(3:9)])
  FCS2stgOverlap_coefs[(3*(i-1)+3), c(4:10)] <-
    colMeans(FCS2stgOverlap_coefsMEM1[((
       FCS2stgOverlap_coefsMEM1$SIMNUM == i) &
                                          (
                                             FCS2stgOverlap_coefsMEM1$STUDY
                                              == 3)), c(3:9)])
}

for (i in 1:100) {
  FCS2stgOverlap_coefs[(3*(i-1)+301), c(4:10)] <-
    colMeans(FCS2stgOverlap_coefsMEM2[((
       FCS2stgOverlap_coefsMEM2$SIMNUM == i) &
                                          (
                                             FCS2stgOverlap_coefsMEM2$STUDY
                                              == 1)), c(3:9)])
  FCS2stgOverlap_coefs[(3*(i-1)+302), c(4:10)] <-
    colMeans(FCS2stgOverlap_coefsMEM2[((
       FCS2stgOverlap_coefsMEM2$SIMNUM == i) &
                                          (
                                             FCS2stgOverlap_coefsMEM2$STUDY
                                              == 2)), c(3:9)])
  FCS2stgOverlap_coefs[(3*(i-1)+303), c(4:10)] <-
    colMeans(FCS2stgOverlap_coefsMEM2[((
       FCS2stgOverlap_coefsMEM2$SIMNUM == i) &
                                          (
                                             FCS2stgOverlap_coefsMEM2$STUDY
                                              == 3)), c(3:9)])
}
```

```
# jomoOverlap
jomoOverlap_coefs <- data.frame(MEM = rep(c(1:2), each = 300),
                                SIMNUM = rep(c(1:100), each =
                                    3),
                                STUDY = rep(c(1:3), times =
                                   200),
                                Intercept = numeric(600),
                                EDU2 = numeric(600),
                                EDU3 = numeric(600),
                                SEX1 = numeric(600),
                                AGE = numeric(600),
                                PHYS2 = numeric(600),
                                PHYS3 = numeric(600))
head(jomoOverlap_coefs)

for (i in 1:100) {
  jomoOverlap_coefs[(3*(i-1)+1), c(4:10)] <-
    colMeans(jomoOverlap_coefsMEM1[((
      jomoOverlap_coefsMEM1$SIMNUM == i) &
                                     (
                                       jomoOverlap_coefsMEM1$STUDY
                                        == 1)), c(3:9)])
  jomoOverlap_coefs[(3*(i-1)+2), c(4:10)] <-
    colMeans(jomoOverlap_coefsMEM1[((
      jomoOverlap_coefsMEM1$SIMNUM == i) &
                                     (
                                       jomoOverlap_coefsMEM1$STUDY
                                        == 2)), c(3:9)])
  jomoOverlap_coefs[(3*(i-1)+3), c(4:10)] <-
    colMeans(jomoOverlap_coefsMEM1[((
      jomoOverlap_coefsMEM1$SIMNUM == i) &
                                     (
                                       jomoOverlap_coefsMEM1$STUDY
                                        == 3)), c(3:9)])
}

for (i in 1:100) {
  jomoOverlap_coefs[(3*(i-1)+301), c(4:10)] <-
    colMeans(jomoOverlap_coefsMEM2[((
      jomoOverlap_coefsMEM2$SIMNUM == i) &
                                     (
                                       jomoOverlap_coefsMEM2$STUDY
                                        == 1)), c(3:9)])
  jomoOverlap_coefs[(3*(i-1)+302), c(4:10)] <-
    colMeans(jomoOverlap_coefsMEM2[((
      jomoOverlap_coefsMEM2$SIMNUM == i) &
                                     (
```

```
                                                    jomoOverlap_coefsMEM2$STUDY
                                                     == 2)), c(3:9)])
  jomoOverlap_coefs[(3*(i-1)+303), c(4:10)] <-
    colMeans(jomoOverlap_coefsMEM2[((
       jomoOverlap_coefsMEM2$SIMNUM == i) &
                                                   (
                                                    jomoOverlap_coefsMEM2$STUDY
                                                     == 3)), c(3:9)])
}


# ------------------------------------------

complete_coefs <- rbind(complete_coefsMEM1, complete_coefsMEM2,
   complete_coefsMEM3)
complete_coefs$MEM <- rep(c(1:3), each = 300)

# Compute averages over all simulations for each parameter
# In MEMORY1
averages_MEM1 <- data.frame(DATASET = rep(c("Complete", "
   jomoNoOverlap", "FCS2stgNoOverlap",
                                        "jomoOverlap", "
                                           FCS2stgOverlap"), each =
                                           3),
                            STUDY = rep(c(1,2,3), times = 5),
                            Intercept = numeric(15),
                            EDU2 = numeric(15),
                            EDU3 = numeric(15),
                            SEX1 = numeric(15),
                            AGE = numeric(15),
                            PHYS2 = numeric(15),
                            PHYS3 = numeric(15))

head(averages_MEM1)

for (i in 1:3) {
  averages_MEM1[i, c(3:9)] <- colMeans(complete_coefs[((
     complete_coefs$MEM == 1) &
                                                        (
                                                           complete_coefs$STU
                                                            ==
                                                           i)),
                                                            c
                                                           (3:9)
                                                           ])
}

for (i in 1:3) {
```

```
  averages_MEM1[(3+i), c(3:9)] <- colMeans(jomoNoOverlap_coefs
      [((jomoNoOverlap_coefs$MEM == 1) &
                                                    (
                                                        jomoNoOverlap
                                                        ==
                                                        i)),
                                                         c
                                                        (4:10)
                                                        ])
}

for (i in 1:3) {
  averages_MEM1[(6+i), c(3:9)] <- colMeans(
      FCS2stgNoOverlap_coefs[((FCS2stgNoOverlap_coefs$MEM == 1) &
                                                        (
                                                            FCS
                                                            ==
                                                            i
                                                            )
                                                            )
                                                            ,
                                                            c
                                                            (4:1
                                                            ])
}

for (i in 1:3) {
  averages_MEM1[(9+i), c(3:9)] <- colMeans(jomoOverlap_coefs[((
      jomoOverlap_coefs$MEM == 1) &
                                                        (
                                                            jomo
                                                            ==
                                                            i
                                                            )
                                                            )
                                                            ,
                                                            c
                                                            (4:1
                                                            ])
}
```

```
for (i in 1:3) {
  averages_MEM1[(12+i), c(3:9)] <- colMeans(FCS2stgOverlap_coefs
     [((FCS2stgOverlap_coefs$MEM == 1) &
                                                                (
                                                                   FCS2s
                                                                   ==
                                                                   i
                                                                   )
                                                                   )
                                                                   ,
                                                                   c
                                                                   (4:10
                                                                   ])

}

# IN MEMORY2
averages_MEM2 <- data.frame(DATASET = rep(c("Complete", "
    jomoNoOverlap", "FCS2stgNoOverlap",
                                              "jomoOverlap", "
                                                 FCS2stgOverlap"),
                                                  each = 3),
                            STUDY = rep(c(1,2,3), times = 5),
                            Intercept = numeric(15),
                            EDU2 = numeric(15),
                            EDU3 = numeric(15),
                            SEX1 = numeric(15),
                            AGE = numeric(15),
                            PHYS2 = numeric(15),
                            PHYS3 = numeric(15))

head(averages_MEM2)

for (i in 1:3) {
  averages_MEM2[i, c(3:9)] <- colMeans(complete_coefs[((
     complete_coefs$MEM == 2) &
                                                        (
                                                           complete_coefs$STU
                                                            ==
                                                           i)),
                                                            c
                                                           (3:9)
                                                           ])

}
```

```
for (i in 1:3) {
  averages_MEM2[(3+i), c(3:9)] <- colMeans(jomoNoOverlap_coefs
      [((jomoNoOverlap_coefs$MEM == 2) &
                                                    (
                                                      jomo
                                                      ==
                                                      i
                                                    )
                                                    )
                                                    ,
                                                    c
                                                    (4:1
                                                    ])

}

for (i in 1:3) {
  averages_MEM2[(6+i), c(3:9)] <- colMeans(
      FCS2stgNoOverlap_coefs[((FCS2stgNoOverlap_coefs$MEM == 2) &
                                                    (
                                                      F
                                                      =
                                                      i
                                                    )
                                                    )
                                                    ,
                                                    c
                                                    (
                                                    ]

}

for (i in 1:3) {
  averages_MEM2[(9+i), c(3:9)] <- colMeans(jomoOverlap_coefs[((
      jomoOverlap_coefs$MEM == 2) &
                                                    (
                                                      jomo
                                                      ==
                                                      i
```

```
                                                                    )
                                                                    )
                                                                    ,
                                                                    c
                                                                    (4:10)
                                                                    ])
}

for (i in 1:3) {
  averages_MEM2[(12+i), c(3:9)] <- colMeans(FCS2stgOverlap_coefs
      [((FCS2stgOverlap_coefs$MEM == 2) &
                                                                      (
                                                                        FCS
                                                                        ==
                                                                        i
                                                                        )
                                                                        )
                                                                        ,
                                                                        c
                                                                        (4:1
                                                                        ])
}

# IN MEMORY3
averages_MEM3 <- data.frame(DATASET = rep(c("Complete", "
    jomoNoOverlap", "FCS2stgNoOverlap",
                                                "jomoOverlap", "
                                                   FCS2stgOverlap"),
                                                    each = 3),
                            STUDY = rep(c(1,2,3), times = 5),
                            Intercept = numeric(15),
                            EDU2 = numeric(15),
                            EDU3 = numeric(15),
                            SEX1 = numeric(15),
                            AGE = numeric(15),
                            PHYS2 = numeric(15),
                            PHYS3 = numeric(15))

head(averages_MEM3)

for (i in 1:3) {
  averages_MEM3[i, c(3:9)] <- colMeans(complete_coefs[((
```

```
      complete_coefs$MEM == 3) &
                                                    (
                                                      complete_coefs$
                                                       ==
                                                      i)),
                                                       c
                                                      (3:9)
                                                      ])
}

for (i in 1:3) {
  averages_MEM3[(3+i), c(3:9)] <- colMeans(jomoNoOverlap_coefs
      [((jomoNoOverlap_coefs$MEM == 3) &
                                                          (
                                                            jomo

                                                            ==

                                                            i
                                                            )
                                                            )
                                                            ,

                                                            c
                                                            (4:1
                                                            ])

}

for (i in 1:3) {
  averages_MEM3[(6+i), c(3:9)] <- colMeans(
      FCS2stgNoOverlap_coefs[((FCS2stgNoOverlap_coefs$MEM == 3) &
                                                          (
                                                            F

                                                            =

                                                            i
                                                            )
                                                            )
                                                            ,

                                                            c
                                                            (
                                                            ]

}
```

```
for (i in 1:3) {
  averages_MEM3[(9+i), c(3:9)] <- colMeans(jomoOverlap_coefs[((
     jomoOverlap_coefs$MEM == 3) &
                                                                (
                                                                   jomoOver
                                                                   ==
                                                                   i
                                                                   )
                                                                   )
                                                                   ,
                                                                   c
                                                                   (4:10)
                                                                   ])

}

for (i in 1:3) {
  averages_MEM3[(12+i), c(3:9)] <- colMeans(FCS2stgOverlap_coefs
     [((FCS2stgOverlap_coefs$MEM == 3) &
                                                                    (
                                                                       FCS
                                                                       ==
                                                                       i
                                                                       )
                                                                       )
                                                                       ,
                                                                       c
                                                                       (4:1
                                                                       ])

}

# BIAS MEMORY 1
bias_MEM1 <- data.frame(DATASET = rep(c("Complete", "
    jomoNoOverlap", "FCS2stgNoOverlap",
                                                    "jomoOverlap", "
                                                       FCS2stgOverlap"),
                                                        each = 3),
                                  STUDY = rep(c(1,2,3), times = 5),
                                  Intercept = numeric(15),
                                  EDU2 = numeric(15),
                                  EDU3 = numeric(15),
```

```
                                SEX1 = numeric(15),
                                AGE = numeric(15),
                                PHYS2 = numeric(15),
                                PHYS3 = numeric(15))

head(bias_MEM1)

for (i in 1:3) {
  bias_MEM1[(3+i), c(3:9)] <- colMeans((jomoNoOverlap_coefs[((
      jomoNoOverlap_coefs$MEM == 1) &
                                          (jomoNoOverlap_coefs$STUDY
                                             == i)), c(4:10)]) -
                                            complete_coefs[((
                                               complete_coefs$MEM ==
                                               1) &
                                                                (
                                                   complete_coefs$
                                                    ==
                                                   i)),
                                                    c
                                                   (3:9)
                                                   ])
}

for (i in 1:3) {
  bias_MEM1[(6+i), c(3:9)] <- colMeans((FCS2stgNoOverlap_coefs
      [((FCS2stgNoOverlap_coefs$MEM == 1) &
                                                                  (
                                                                     FCS2stg

                                                                    ==

                                                                    i
                                                                    )
                                                                    )
                                                                    ,

                                                                    c
                                                                    (4:10)
                                                                    ])

                                                                    -

                                                 complete_coefs[((
                                                    complete_coefs$MEM
                                                    == 1) &
                                                                 (
                                                    complete_coe
```

```
                                                                    ==
                                                                     i
                                                                    ))
                                                                    ,
                                                                    c
                                                                    (3:9)
                                                                    ])
}

for (i in 1:3) {
  bias_MEM1[(9+i), c(3:9)] <- colMeans((jomoOverlap_coefs[((
      jomoOverlap_coefs$MEM == 1) &
                                                                        (
                                                                            jomoOver

                                                                            ==

                                                                            i
                                                                            )
                                                                            )
                                                                            ,

                                                                            c
                                                                            (4:10)
                                                                            ])

                                                                            -

                                                        complete_coefs[((
                                                            complete_coefs$MEM
                                                            == 1) &
                                                                                (
                                                                    complete_coefs$S

                                                                    ==
                                                                     i
                                                                    ))
                                                                    ,
                                                                    c
                                                                    (3:9)
                                                                    ])
}

for (i in 1:3) {
  bias_MEM1[(12+i), c(3:9)] <- colMeans((FCS2stgOverlap_coefs[((
      FCS2stgOverlap_coefs$MEM == 1) &
                                                                    (
```

```
                                                                FCS2stgO
                                                                ==
                                                                i
                                                                )
                                                                )
                                                                ,
                                                                c
                                                                (4:10)
                                                                ])
                                                                -
                                                    complete_coefs[((
                                                        complete_coefs$MEM
                                                        == 1) &
                                                                (
                                                            complete_co
                                                                ==
                                                                i
                                                                )
                                                                )
                                                                ,
                                                                c
                                                                (3:9)
                                                                ])
}


# BIAS MEMORY 2
bias_MEM2 <- data.frame(DATASET = rep(c("Complete", "
    jomoNoOverlap", "FCS2stgNoOverlap",
                                                    "jomoOverlap", "
                                                        FCS2stgOverlap"),
                                                        each = 3),
                                STUDY = rep(c(1,2,3), times = 5),
                                Intercept = numeric(15),
                                EDU2 = numeric(15),
                                EDU3 = numeric(15),
                                SEX1 = numeric(15),
                                AGE = numeric(15),
                                PHYS2 = numeric(15),
```

```
                                    PHYS3 = numeric(15))

head(bias_MEM2)

for (i in 1:3) {
  bias_MEM2[(3+i), c(3:9)] <- colMeans((jomoNoOverlap_coefs[((
      jomoNoOverlap_coefs$MEM == 2) &
                                                              (
                                                                 jomoNoOverl
                                                                 ==
                                                                 i
                                                                 )
                                                                 )
                                                                 ,
                                                                 c
                                                                 (4:10)
                                                                 ])
                                                                 -
                                                  complete_coefs[((
                                                     complete_coefs$MEM
                                                     == 2) &
                                                                    (
                                                                       complete_coefs$ST
                                                                       ==
                                                                        i
                                                                       ))
                                                                       ,
                                                                       c
                                                                       (3:9)
                                                                       ])
}

for (i in 1:3) {
  bias_MEM2[(6+i), c(3:9)] <- colMeans((FCS2stgNoOverlap_coefs
      [((FCS2stgNoOverlap_coefs$MEM == 2) &
                                                                 (
                                                                    FCS2stgN
                                                                    ==
                                                                    i
                                                                    )
```

```
                                                                    )
                                                                    ,
                                                                    c
                                                                    (4:1
                                                                    ])
                                                                    -
                                                    complete_coefs[((
                                                        complete_coefs$MEM
                                                        == 2) &
                                                                (
                                                                    complete_coe
                                                                    ==
                                                                     i
                                                                    ))
                                                                    ,
                                                                    c
                                                                    (3:9)
                                                                    ])
}

for (i in 1:3) {
  bias_MEM2[(9+i), c(3:9)] <- colMeans((jomoOverlap_coefs[((
      jomoOverlap_coefs$MEM == 2) &
                                                            (
                                                                jomoOverl
                                                                ==
                                                                i
                                                                )
                                                                )
                                                                ,
                                                                c
                                                                (4:10)
                                                                ])
                                                                -
                                                    complete_coefs[((
                                                        complete_coefs$MEM
                                                        == 2) &
                                                                (
                                                                    complete_coe
```

```
                                                                    ==
                                                                     i
                                                                    ))
                                                                    ,
                                                                    c
                                                                    (3:9)
                                                                    ])
}

for (i in 1:3) {
  bias_MEM2[(12+i), c(3:9)] <- colMeans((FCS2stgOverlap_coefs[((
      FCS2stgOverlap_coefs$MEM == 2) &
                                                                (
                                                                    FCS2stgO

                                                                    ==

                                                                    i
                                                                    )
                                                                    )
                                                                    ,

                                                                    c
                                                                    (4:10)
                                                                    ])

                                                                    -

                                                  complete_coefs[((
                                                      complete_coefs$MEM
                                                      == 2) &
                                                                        (
                                                                    complete_coefs$$

                                                                    ==

                                                                    i
                                                                    )
                                                                    )
                                                                    ,

                                                                    c
                                                                    (3:9)
                                                                    ])

}
```

```
# BIAS MEMORY 3
bias_MEM3 <- data.frame(DATASET = rep(c("Complete", "
    jomoNoOverlap", "FCS2stgNoOverlap",
                                          "jomoOverlap", "
                                             FCS2stgOverlap"),
                                             each = 3),
                            STUDY = rep(c(1,2,3), times = 5),
                            Intercept = numeric(15),
                            EDU2 = numeric(15),
                            EDU3 = numeric(15),
                            SEX1 = numeric(15),
                            AGE = numeric(15),
                            PHYS2 = numeric(15),
                            PHYS3 = numeric(15))

head(bias_MEM3)

for (i in 1:3) {
  bias_MEM3[(3+i), c(3:9)] <- colMeans((jomoNoOverlap_coefs[((
      jomoNoOverlap_coefs$MEM == 3) &
                                                              (
                                                                 jomoNo

                                                                 ==

                                                                 i
                                                                 )
                                                                 )
                                                                 ,

                                                                 c
                                                                 (4:10)
                                                                 ])

                                                                 -

                                                    complete_coefs[((
                                                       complete_coefs$MEM
                                                       == 3) &
                                                                 (
                                                                    complete_coe

                                                                 ==
                                                                  i
                                                                 ))
                                                                 ,
                                                                 c
                                                                 (3:9)
```

```
                                                                    ])
}

for (i in 1:3) {
  bias_MEM3[(6+i), c(3:9)] <- colMeans((FCS2stgNoOverlap_coefs
      [((FCS2stgNoOverlap_coefs$MEM == 3) &
                                                                        (
                                                                          FCS2stgN
                                                                          ==
                                                                          i
                                                                          )
                                                                          )
                                                                          ,
                                                                          c
                                                                          (4:10)
                                                                          ])
                                                                          -
                                                    complete_coefs[((
                                                      complete_coefs$MEM
                                                      == 3) &
                                                                  (
                                                                    complete_coefs$S
                                                                    ==
                                                                     i
                                                                    ))
                                                                    ,
                                                                    c
                                                                    (3:9)
                                                                    ])
}

# MSE MEMORY 1
mse_MEM1 <- data.frame(DATASET = rep(c("Complete", "
    jomoNoOverlap", "FCS2stgNoOverlap",
                                                  "jomoOverlap", "
                                                     FCS2stgOverlap"),
                                                     each = 3),
                                 STUDY = rep(c(1,2,3), times = 5),
                                 Intercept = numeric(15),
                                 EDU2 = numeric(15),
                                 EDU3 = numeric(15),
                                 SEX1 = numeric(15),
```

```
                    AGE = numeric(15),
                    PHYS2 = numeric(15),
                    PHYS3 = numeric(15))

head(mse_MEM1)

for (i in 1:3) {
  mse_MEM1[(3+i), c(3:9)] <- colMeans(((jomoNoOverlap_coefs[((
      jomoNoOverlap_coefs$MEM == 1) &
                                                              (
                                                                 jomoNo
                                                                 ==
                                                                 i
                                                                 )
                                                                 )
                                                                 ,
                                                                 c
                                                                 (4:10)
                                                                 ])
                                                                 -
                                                  complete_coefs[((
                                                     complete_coefs$MEM
                                                     == 1) &
                                                                  (
                                                                     complete_coe
                                                                     ==
                                                                      i
                                                                     ))
                                                                     ,
                                                                     c
                                                                     (3:9)
                                                                     ])
                                                                     ^2)

}

for (i in 1:3) {
  mse_MEM1[(6+i), c(3:9)] <- colMeans(((FCS2stgNoOverlap_coefs
      [((FCS2stgNoOverlap_coefs$MEM == 1) &
                                                                  (
                                                                     FCS
```

```
                                                                    ==

                                                                    i
                                                                    )
                                                                    )
                                                                    ,

                                                                    c
                                                                    (4:10)
                                                                    ])

                                                                    -

                                                  complete_coefs[((
                                                      complete_coefs$MEM
                                                     == 1) &
                                                                 (
                                                                    complete_coefs$S

                                                                    ==
                                                                     i
                                                                    ))
                                                                    ,
                                                                    c
                                                                    (3:9)
                                                                    ])
                                                                    ^2)

}

for (i in 1:3) {
  mse_MEM1[(9+i), c(3:9)] <- colMeans(((jomoOverlap_coefs[((
      jomoOverlap_coefs$MEM == 1) &
                                                                (
                                                                   jomoOverlap_co

                                                                   ==

                                                                   i
                                                                   )
                                                                   )
                                                                   ,

                                                                   c
                                                                   (4:10)
                                                                   ])

                                                                   -
```

```
                                                    complete_coefs[((
                                                        complete_coefs$MEM
                                                        == 1) &
                                                                        (
                                                                            complete_coe

                                                                        ==
                                                                          i
                                                                        ))
                                                                        ,
                                                                        c
                                                                        (3:9)
                                                                        ])
                                                                        ^2)

}

for (i in 1:3) {
  mse_MEM1[(12+i), c(3:9)] <- colMeans(((FCS2stgOverlap_coefs[((
      FCS2stgOverlap_coefs$MEM == 1) &
                                                                    (
                                                                        FCS2s

                                                                    ==

                                                                    i
                                                                    )
                                                                    )
                                                                    ,

                                                                    c
                                                                    (4:10
                                                                    ])

                                                                    -

                                                    complete_coefs[((
                                                        complete_coefs$MEM
                                                        == 1) &
                                                                        (
                                                                            complete_co

                                                                        ==

                                                                        i
                                                                        )
                                                                        )
```

```
                                                                    ,

                                                                    c
                                                                    (3:9)
                                                                    ])
                                                                    ^2)

}

# MSE MEMORY 2
mse_MEM2 <- data.frame(DATASET = rep(c("Complete", "
    jomoNoOverlap", "FCS2stgNoOverlap",
                                                    "jomoOverlap", "
                                                       FCS2stgOverlap"), each
                                                        = 3),
                                STUDY = rep(c(1,2,3), times = 5),
                                Intercept = numeric(15),
                                EDU2 = numeric(15),
                                EDU3 = numeric(15),
                                SEX1 = numeric(15),
                                AGE = numeric(15),
                                PHYS2 = numeric(15),
                                PHYS3 = numeric(15))

head(mse_MEM2)

for (i in 1:3) {
  mse_MEM2[(3+i), c(3:9)] <- colMeans(((jomoNoOverlap_coefs[((
      jomoNoOverlap_coefs$MEM == 2) &
                                                                    (
                                                                       jomoNoOverl

                                                                       ==

                                                                       i
                                                                       )
                                                                       )
                                                                       ,

                                                                       c
                                                                       (4:10)
                                                                       ])

                                                                       -

                                                          complete_coefs[((
                                                             complete_coefs$MEM
                                                             == 2) &
```

```
                                                                    (
                                                                      complete_coe

                                                                      ==
                                                                       i
                                                                      ))
                                                                      ,
                                                                      c
                                                                      (3:9)
                                                                      ])
                                                                      ^2)

}

for (i in 1:3) {
  mse_MEM2[(6+i), c(3:9)] <- colMeans(((FCS2stgNoOverlap_coefs
      [((FCS2stgNoOverlap_coefs$MEM == 2) &
                                                                 (
                                                                   FCS

                                                                   ==

                                                                   i
                                                                   )
                                                                   )
                                                                   ,

                                                                   c
                                                                   (4:1
                                                                   ])

                                                                   -

                                                    complete_coefs[((
                                                       complete_coefs$MEM
                                                       == 2) &
                                                                    (
                                                                      complete_coe

                                                                      ==
                                                                       i
                                                                      ))
                                                                      ,
                                                                      c
                                                                      (3:9)
                                                                      ])
                                                                      ^2)
```

```
}

for (i in 1:3) {
  mse_MEM2[(9+i), c(3:9)] <- colMeans(((jomoOverlap_coefs[((
      jomoOverlap_coefs$MEM == 2) &
                                                          (
                                                              jomoOverlap_co
                                                              ==
                                                              i
                                                              )
                                                              )
                                                              ,
                                                              c
                                                              (4:10)
                                                              ])
                                                              -
                                                  complete_coefs[((
                                                      complete_coefs$MEM
                                                      == 2) &
                                                                 (
                                                                     complete_coefs$S
                                                                     ==
                                                                      i
                                                                     ))
                                                                     ,
                                                                     c
                                                                     (3:9)
                                                                     ])
                                                                     ^2)

}

for (i in 1:3) {
  mse_MEM2[(12+i), c(3:9)] <- colMeans(((FCS2stgOverlap_coefs[((
      FCS2stgOverlap_coefs$MEM == 2) &
                                                                (
                                                                    FCS2stgO
                                                                    ==
                                                                    i
                                                                    )
```

```
                                                                    )
                                                                    ,
                                                                    c
                                                                    (4:10
                                                                    ])
                                                                    -
                                                    complete_coefs[((
                                                        complete_coefs$MEM
                                                        == 2) &
                                                                (
                                                                    complete_co
                                                                    ==
                                                                    i
                                                                    )
                                                                    )
                                                                    ,
                                                                    c
                                                                    (3:9)
                                                                    ])
                                                                    ^2)
}

# MSE MEMORY 3
mse_MEM3 <- data.frame(DATASET = rep(c("Complete", "
    jomoNoOverlap", "FCS2stgNoOverlap",
                                                    "jomoOverlap", "
                                                        FCS2stgOverlap"), each
                                                         = 3),
                           STUDY = rep(c(1,2,3), times = 5),
                           Intercept = numeric(15),
                           EDU2 = numeric(15),
                           EDU3 = numeric(15),
                           SEX1 = numeric(15),
                           AGE = numeric(15),
                           PHYS2 = numeric(15),
                           PHYS3 = numeric(15))

head(mse_MEM3)

for (i in 1:3) {
  mse_MEM3[(3+i), c(3:9)] <- colMeans(((jomoNoOverlap_coefs[((
```

```
      jomoNoOverlap_coefs$MEM == 3) &
                                                    (
                                                        jomoNoOverl
                                                        ==
                                                        i
                                                        )
                                                        )
                                                        ,
                                                        c
                                                        (4:10)
                                                        ])
                                                        -
                                                complete_coefs[((
                                                    complete_coefs$MEM
                                                    == 3) &
                                                                (
                                                                    complete_coefs$S
                                                                    ==
                                                                     i
                                                                ))
                                                                ,
                                                                c
                                                                (3:9)
                                                                ])
                                                                ^2)

}

for (i in 1:3) {
  mse_MEM3[(6+i), c(3:9)] <- colMeans(((FCS2stgNoOverlap_coefs
      [((FCS2stgNoOverlap_coefs$MEM == 3) &
                                                    (
                                                        FCS2stgN
                                                        ==
                                                        i
                                                        )
                                                        )
                                                        ,
                                                        c
```

```
                                                                    (4:1
                                                                    ])

                                                                    -

                                                complete_coefs[((
                                                    complete_coefs$MEM
                                                    == 3) &
                                                                (
                                                                    complete_coe

                                                                    ==
                                                                     i
                                                                    ))
                                                                    ,
                                                                    c
                                                                    (3:9)
                                                                    ])
                                                                    ^2)

}

for (i in 1:3) {
  mse_MEM3[(9+i), c(3:9)] <- colMeans(((jomoOverlap_coefs[((
      jomoOverlap_coefs$MEM == 3) &
                                                              (
                                                                  jomoOverl

                                                                  ==

                                                                  i
                                                                  )
                                                                  )
                                                                  ,

                                                                  c
                                                                  (4:10)
                                                                  ])

                                                                  -

                                                complete_coefs[((
                                                    complete_coefs$MEM
                                                    == 3) &
                                                                (
                                                                    complete_coe

                                                                    ==
```

```
                                                                  i
                                                                ))
                                                                ,
                                                                c
                                                                (3:9)
                                                                ])
                                                                ^2)

}

for (i in 1:3) {
  mse_MEM3[(12+i), c(3:9)] <- colMeans(((FCS2stgOverlap_coefs[((
      FCS2stgOverlap_coefs$MEM == 3) &
                                                                    (
                                                                      FCS2stgO

                                                                      ==

                                                                      i
                                                                      )
                                                                      )
                                                                      ,

                                                                      c
                                                                      (4:10)
                                                                      ])

                                                                      -

                                                    complete_coefs[((
                                                      complete_coefs$MEM
                                                      == 3) &
                                                                  (
                                                                    complete_coefs$$

                                                                    ==

                                                                    i
                                                                    )
                                                                    )
                                                                    ,

                                                                    c
                                                                    (3:9)
                                                                    ])
                                                                    ^2)

}
```

```
write.csv(averages_MEM1, "averages_MEM1.csv")
write.csv(averages_MEM2, "averages_MEM2.csv")
write.csv(averages_MEM3, "averages_MEM3.csv")

write.csv(bias_MEM1, "bias_MEM1.csv")
write.csv(bias_MEM2, "bias_MEM2.csv")
write.csv(bias_MEM3, "bias_MEM3.csv")

write.csv(mse_MEM1, "mse_MEM1.csv")
write.csv(mse_MEM2, "mse_MEM2.csv")
write.csv(mse_MEM3, "mse_MEM3.csv")

library(dplyr)
library(lme4)

# Load the complete data set
complete <- read.csv("/Users/catherinehilgers/Documents/Thesis/R
    Code/simulatedDataset.csv")
complete$SIMNUM <- as.factor(complete$SIMNUM)
complete$STUDY <- as.factor(complete$STUDY)
complete$SUBJECTNUM <- as.integer(complete$SUBJECTNUM)
complete$SEX <- as.factor(complete$SEX)
complete$EDU <- as.factor(complete$EDU)
complete$PHYS <- as.factor(complete$PHYS)
complete$MEMORY1 <- as.integer(complete$MEMORY1)
complete$MEMORY2 <- as.integer(complete$MEMORY2)
complete$MEMORY3 <- as.integer(complete$MEMORY3)

# Load the jomo data sets
# Use for both jomo data sets
SIM=100
N1=10000
N2=2000
N3=500
N = N1 + N2 + N3
SIMNUM <- as.factor(rep(rep(1:SIM, each=N), each = 5))

# Load the jomo with overlap case
jomoOverlap <- read.csv("/Users/catherinehilgers/Documents/
   Thesis/R Code/imputedDatasets_jomoOverlap.csv")
jomoOverlap$clus <- as.factor(jomoOverlap$clus)
colnames(jomoOverlap)[10] <- "STUDY"
jomoOverlap$id <- as.integer(jomoOverlap$id)
colnames(jomoOverlap)[11] <- "SUBJECTNUM"
jomoOverlap$SEX <- as.factor(jomoOverlap$SEX)
jomoOverlap$EDU <- as.factor(jomoOverlap$EDU)
jomoOverlap$PHYS <- as.factor(jomoOverlap$PHYS)
```

```
jomoOverlap$MEMORY1 <- as.integer(jomoOverlap$MEMORY1)
jomoOverlap$MEMORY2 <- as.integer(jomoOverlap$MEMORY2)
jomoOverlap$X <- NULL
jomoOverlap$X1 <- NULL
jomoOverlap$Z1 <- NULL
jomoOverlap <- jomoOverlap[!(jomoOverlap$Imputation == 0), ]
colnames(jomoOverlap)[9] <- "IMPNUM"
jomoOverlap <- cbind(jomoOverlap,SIMNUM)
head(jomoOverlap)

# jomo without overlap
jomoNoOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/imputedDatasets_jomoNoOverlap.csv")
jomoNoOverlap$clus <- as.factor(jomoNoOverlap$clus)
colnames(jomoNoOverlap)[11] <- "STUDY"
jomoNoOverlap$id <- as.integer(jomoNoOverlap$id)
colnames(jomoNoOverlap)[12] <- "SUBJECTNUM"
jomoNoOverlap$SEX <- as.factor(jomoNoOverlap$SEX)
jomoNoOverlap$EDU <- as.factor(jomoNoOverlap$EDU)
jomoNoOverlap$PHYS <- as.factor(jomoNoOverlap$PHYS)
# rounding MEMORY1,2,3 to integers after they have been imputed
    as numeric
# Some are out of bounds and must be changed to the highest
    level
jomoNoOverlap$MEMORY1 <- as.integer(round(jomoNoOverlap$MEMORY1,
     digits = 0))
jomoNoOverlap$MEMORY1[jomoNoOverlap$MEMORY1 == 16] <- as.integer
    (15)
jomoNoOverlap$MEMORY1[jomoNoOverlap$MEMORY1 == 17] <- as.integer
    (15)
jomoNoOverlap$MEMORY2 <- as.integer(round(jomoNoOverlap$MEMORY2,
     digits = 0))
jomoNoOverlap$MEMORY2[jomoNoOverlap$MEMORY2 == 13] <- as.integer
    (12)
jomoNoOverlap$MEMORY2[jomoNoOverlap$MEMORY2 == 14] <- as.integer
    (12)
jomoNoOverlap$MEMORY2[jomoNoOverlap$MEMORY2 == 15] <- as.integer
    (12)
jomoNoOverlap$MEMORY3 <- as.integer(round(jomoNoOverlap$MEMORY3,
     digits = 0))
jomoNoOverlap$MEMORY3[jomoNoOverlap$MEMORY3 == 18] <- as.integer
    (16)
jomoNoOverlap$MEMORY3[jomoNoOverlap$MEMORY3 == 17] <- as.integer
    (16)
jomoNoOverlap$X <- NULL
jomoNoOverlap$X1 <- NULL
jomoNoOverlap$Z1 <- NULL
jomoNoOverlap <- jomoNoOverlap[!(jomoNoOverlap$Imputation == 0),
```

```
    ]
colnames(jomoNoOverlap)[10] <- "IMPNUM"
jomoNoOverlap <- cbind(jomoNoOverlap,SIMNUM)
head(jomoNoOverlap)

# Load the FCS 2stage data sets
# FCS 2 stage without overlap
FCS2stgNoOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/FCS2stgNoOverlap.csv")
FCS2stgNoOverlap$SIMNUM <- as.factor(FCS2stgNoOverlap$SIMNUM)
FCS2stgNoOverlap$STUDY <- as.factor(FCS2stgNoOverlap$STUDY)
FCS2stgNoOverlap$SUBJECTNUM <- as.integer(
    FCS2stgNoOverlap$SUBJECTNUM)
FCS2stgNoOverlap$SEX <- as.factor(FCS2stgNoOverlap$SEX)
FCS2stgNoOverlap$EDU <- as.factor(FCS2stgNoOverlap$EDU)
FCS2stgNoOverlap$PHYS <- as.factor(FCS2stgNoOverlap$PHYS)
FCS2stgNoOverlap$MEMORY1 <- as.integer(FCS2stgNoOverlap$MEMORY1)
FCS2stgNoOverlap$MEMORY2 <- as.integer(FCS2stgNoOverlap$MEMORY2)
FCS2stgNoOverlap$MEMORY3 <- as.integer(FCS2stgNoOverlap$MEMORY3)

# FCS 2 stage with overlap
FCS2stgOverlap <- read.csv("/Users/catherinehilgers/Documents/
    Thesis/R Code/FCS2stgOverlap.csv")
FCS2stgOverlap$SIMNUM <- as.factor(FCS2stgOverlap$SIMNUM)
FCS2stgOverlap$STUDY <- as.factor(FCS2stgOverlap$STUDY)
FCS2stgOverlap$SUBJECTNUM <- as.integer(
    FCS2stgOverlap$SUBJECTNUM)
FCS2stgOverlap$SEX <- as.factor(FCS2stgOverlap$SEX)
FCS2stgOverlap$EDU <- as.factor(FCS2stgOverlap$EDU)
FCS2stgOverlap$PHYS <- as.factor(FCS2stgOverlap$PHYS)
FCS2stgOverlap$MEMORY1 <- as.integer(FCS2stgOverlap$MEMORY1)
FCS2stgOverlap$MEMORY2 <- as.integer(FCS2stgOverlap$MEMORY2)

# ---------------------------------------

# Apply a GLM model to each of complete data sets

# Create a column for the number of items minus the items
    correct
M1=15
M2=12
M3=16

complete$M1 <- rep(M1, times = 125000)
complete$M2 <- rep(M2, times = 125000)
complete$M3 <- rep(M3, times = 125000)

FCS2stgOverlap$M1 <- rep(M1, times = 625000)
```

```
FCS2stgOverlap$M2 <- rep(M2, times = 625000)
FCS2stgOverlap$M3 <- rep(M3, times = 625000)

FCS2stgNoOverlap$M1 <- rep(M1, times = 625000)
FCS2stgNoOverlap$M2 <- rep(M2, times = 625000)
FCS2stgNoOverlap$M3 <- rep(M3, times = 625000)

jomoOverlap$M1 <- rep(M1, times = 625000)
jomoOverlap$M2 <- rep(M2, times = 625000)
jomoOverlap$M3 <- rep(M3, times = 625000)

jomoNoOverlap$M1 <- rep(M1, times = 625000)
jomoNoOverlap$M2 <- rep(M2, times = 625000)
jomoNoOverlap$M3 <- rep(M3, times = 625000)

SIM <- 100
# Fit model on complete data set using glmer
  ------------------------------------------

# MEMORY 1

complete_coefsMEM1 <- by(complete,
                         complete$SIMNUM,
                         function(x) glmer(cbind(MEMORY1, M1-
                            MEMORY1) ~ EDU + SEX + AGE + PHYS +
                            (1 | STUDY),
                                           data = x,
                                           family = binomial))

# MEMORY 2

complete_coefsMEM2 <- by(complete,
                         complete$SIMNUM,
                         function(x) glmer(cbind(MEMORY2, M2-
                            MEMORY2) ~ EDU + SEX + AGE + PHYS +
                            (1 | STUDY),
                                           data = x,
                                           family = binomial))

# MEMORY 3

complete_coefsMEM3 <- by(complete,
                         complete$SIMNUM,
                         function(x) glmer(cbind(MEMORY3, M3-
                            MEMORY3) ~ EDU + SEX + AGE + PHYS +
                            (1 | STUDY),
                                           data = x,
                                           family = binomial))
```

```
# FCS 2 stage No Overlap
```

```
# MEM1

FCS2stgNoOverlap_coefsMEM1 <- by(FCS2stgNoOverlap,
                                 list(FCS2stgNoOverlap$SIMNUM,
                                    FCS2stgNoOverlap$IMPNUM),
                                 function(x) glmer(cbind(MEMORY1
                                    , M1-MEMORY1) ~ EDU + SEX +
                                    AGE + PHYS + (1 | STUDY),
                                          data = x,
                                          family = binomial))


# MEMORY2
FCS2stgNoOverlap_coefsMEM2 <- by(FCS2stgNoOverlap,
                                 list(FCS2stgNoOverlap$SIMNUM,
                                    FCS2stgNoOverlap$IMPNUM),
                                 function(x) glmer(cbind(MEMORY2
                                    , M2-MEMORY2) ~ EDU + SEX +
                                    AGE + PHYS + (1 | STUDY),
                                                data = x,
                                                family =
                                                   binomial))


# MEMORY3
FCS2stgNoOverlap_coefsMEM3 <- by(FCS2stgNoOverlap,
                                 list(FCS2stgNoOverlap$SIMNUM,
                                    FCS2stgNoOverlap$IMPNUM),
                                 function(x) glmer(cbind(MEMORY3
                                    , M3-MEMORY3) ~ EDU + SEX +
                                    AGE + PHYS + (1 | STUDY),
                                                data = x,
                                                family =
                                                   binomial))
```

```
# jomo No Overlap
```

```
jomoNoOverlap_coefsMEM1 <- by(jomoNoOverlap,
                              list(jomoNoOverlap$SIMNUM,
                                   jomoNoOverlap$IMPNUM),
                              function(x) glmer(cbind(MEMORY1
                                  , M1-MEMORY1) ~ EDU + SEX +
                                  AGE + PHYS + (1 | STUDY),
                                                data = x,
                                                family =
                                                    binomial))


# MEMORY2
jomoNoOverlap_coefsMEM2 <- by(jomoNoOverlap,
                              list(jomoNoOverlap$SIMNUM,
                                   jomoNoOverlap$IMPNUM),
                              function(x) glmer(cbind(MEMORY2
                                  , M2-MEMORY2) ~ EDU + SEX +
                                  AGE + PHYS + (1 | STUDY),
                                                data = x,
                                                family =
                                                    binomial))


# MEMORY3
jomoNoOverlap_coefsMEM3 <- by(jomoNoOverlap,
                              list(jomoNoOverlap$SIMNUM,
                                   jomoNoOverlap$IMPNUM),
                              function(x) glmer(cbind(MEMORY3
                                  , M3-MEMORY3) ~ EDU + SEX +
                                  AGE + PHYS + (1 | STUDY),
                                                data = x,
                                                family =
                                                    binomial))


# jomo WITH overlap
-------------------------------------------------------

# MEM1

jomoOverlap_coefsMEM1 <- by(jomoOverlap,
                            list(jomoOverlap$SIMNUM,
                                 jomoOverlap$IMPNUM),
                            function(x) glmer(cbind(MEMORY1,
                                M1-MEMORY1) ~ EDU + SEX + AGE
                                + PHYS + (1 | STUDY),
                                              data = x,
```

```
                                                    family =
                                                        binomial))


# MEMORY2
jomoOverlap_coefsMEM2 <- by(jomoOverlap,
                                    list(jomoOverlap$SIMNUM,
                                        jomoOverlap$IMPNUM),
                                    function(x) glmer(cbind(MEMORY2,
                                        M2-MEMORY2) ~ EDU + SEX + AGE
                                        + PHYS + (1 | STUDY),
                                                    data = x,
                                                    family =
                                                        binomial))

# FCS 2 stage WITH overlap (no MEMORY3)
    ------------------------------------------------------------

# MEM1

FCS2stgOverlap_coefsMEM1 <- by(FCS2stgOverlap,
                                    list(FCS2stgOverlap$SIMNUM,
                                        FCS2stgOverlap$IMPNUM),
                                    function(x) glmer(cbind(MEMORY1,
                                        M1-MEMORY1) ~ EDU + SEX + AGE
                                        + PHYS + (1 | STUDY),
                                                    data = x,
                                                    family =
                                                        binomial))


# MEMORY2
FCS2stgOverlap_coefsMEM2 <- by(FCS2stgOverlap,
                                    list(FCS2stgOverlap$SIMNUM,
                                        FCS2stgOverlap$IMPNUM),
                                    function(x) glmer(cbind(MEMORY2,
                                        M2-MEMORY2) ~ EDU + SEX + AGE
                                        + PHYS + (1 | STUDY),
                                                    data = x,
                                                    family =
                                                        binomial))


setwd("/Users/catherinehilgers/Documents/Thesis/R Code")
save(FCS2stgOverlap_coefsMEM1, file="FCS2stgOverlap_coefsMEM1.
    RData")
save(FCS2stgOverlap_coefsMEM2, file="FCS2stgOverlap_coefsMEM2.
    RData")
save(jomoOverlap_coefsMEM1, file="jomoOverlap_coefsMEM1.RData")
```

```
save(jomoOverlap_coefsMEM2, file="jomoOverlap_coefsMEM2.RData")


# ------------------------------


save(complete_coefsMEM1, file="complete_coefsMEM1.RData")
save(complete_coefsMEM2, file="complete_coefsMEM2.RData")
save(complete_coefsMEM3, file="complete_coefsMEM3.RData")
save(FCS2stgNoOverlap_coefsMEM1, file="
    FCS2stgNoOverlap_coefsMEM1.RData")
save(FCS2stgNoOverlap_coefsMEM2, file="
    FCS2stgNoOverlap_coefsMEM2.RData")
save(FCS2stgNoOverlap_coefsMEM3, file="
    FCS2stgNoOverlap_coefsMEM3.RData")
save(jomoNoOverlap_coefsMEM1, file="jomoNoOverlap_coefsMEM1.
    RData")
save(jomoNoOverlap_coefsMEM2, file="jomoNoOverlap_coefsMEM2.
    RData")
save(jomoNoOverlap_coefsMEM3, file="jomoNoOverlap_coefsMEM3.
    RData")
save(FCS2stgOverlap_coefsMEM1, file="FCS2stgOverlap_coefsMEM1.
    RData")
save(FCS2stgOverlap_coefsMEM2, file="FCS2stgOverlap_coefsMEM2.
    RData")
save(jomoOverlap_coefsMEM1, file="jomoOverlap_coefsMEM1.RData")
save(jomoOverlap_coefsMEM2, file="jomoOverlap_coefsMEM2.RData")

# ------------------------------------------------------------
# ------------ Calculate random effect for study ---

randomeff <- data.frame("SIM" = c(1:100))

randomeff$completeMEM1 <- numeric(100)

for (i in 1:100) {
  randomeff$completeMEM1[i] <- VarCorr(complete_coefsMEM1[[i]])
     $STUDY[1]
}

randomeff$completeMEM2 <- numeric(100)

for (i in 1:100) {
  randomeff$completeMEM2[i] <- VarCorr(complete_coefsMEM2[[i]])
     $STUDY[1]
}

randomeff$FCSMEM1 <- numeric(100)
```

```
for (i in 1:100) {
  randomeff$FCSMEM1[i] <-
    (VarCorr(FCS2stgOverlap_coefsMEM1[[(5*(i-1))+1]])$STUDY[1] +
       VarCorr(FCS2stgOverlap_coefsMEM1[[(5*(i-1))+2]])$STUDY[1]
           +
       VarCorr(FCS2stgOverlap_coefsMEM1[[(5*(i-1))+3]])$STUDY[1]
           +
       VarCorr(FCS2stgOverlap_coefsMEM1[[(5*(i-1))+4]])$STUDY[1]
           +
       VarCorr(FCS2stgOverlap_coefsMEM1[[(5*(i-1))+5]])$STUDY
          [1])*0.2
}

randomeff$FCSMEM2 <- numeric(100)

for (i in 1:100) {
  randomeff$FCSMEM2[i] <-
    (VarCorr(FCS2stgOverlap_coefsMEM2[[(5*(i-1))+1]])$STUDY[1] +
       VarCorr(FCS2stgOverlap_coefsMEM2[[(5*(i-1))+2]])$STUDY[1]
           +
       VarCorr(FCS2stgOverlap_coefsMEM2[[(5*(i-1))+3]])$STUDY[1]
           +
       VarCorr(FCS2stgOverlap_coefsMEM2[[(5*(i-1))+4]])$STUDY[1]
           +
       VarCorr(FCS2stgOverlap_coefsMEM2[[(5*(i-1))+5]])$STUDY
          [1])*0.2
}

randomeff$jomoMEM1 <- numeric(100)

for (i in 1:100) {
  randomeff$jomoMEM1[i] <-
    (VarCorr(jomoOverlap_coefsMEM1[[(5*(i-1))+1]])$STUDY[1] +
       VarCorr(jomoOverlap_coefsMEM1[[(5*(i-1))+2]])$STUDY[1] +
       VarCorr(jomoOverlap_coefsMEM1[[(5*(i-1))+3]])$STUDY[1] +
       VarCorr(jomoOverlap_coefsMEM1[[(5*(i-1))+4]])$STUDY[1] +
       VarCorr(jomoOverlap_coefsMEM1[[(5*(i-1))+5]])$STUDY[1])
          *0.2
}

randomeff$jomoMEM2 <- numeric(100)

for (i in 1:100) {
  randomeff$jomoMEM2[i] <-
    (VarCorr(jomoOverlap_coefsMEM2[[(5*(i-1))+1]])$STUDY[1] +
       VarCorr(jomoOverlap_coefsMEM2[[(5*(i-1))+2]])$STUDY[1] +
       VarCorr(jomoOverlap_coefsMEM2[[(5*(i-1))+3]])$STUDY[1] +
```

```
        VarCorr(jomoOverlap_coefsMEM2[[(5*(i-1))+4]])$STUDY[1] +
        VarCorr(jomoOverlap_coefsMEM2[[(5*(i-1))+5]])$STUDY[1])
            *0.2
}


# Calculate averages
colMeans(randomeff)

#              SIM completeMEM1 completeMEM2        FCSMEM1
    FCSMEM2       jomoMEM1       jomoMEM2
# 5.050000e+01 9.091224e-05 1.365312e-04 1.523204e-04 1.809084e
    -04 2.896762e-02 2.439645e-02

# Calculate bias
# MEM1, FCS
mean(randomeff$completeMEM1 - randomeff$FCSMEM1)
# -6.140818e-05

# MEM2, FCS
mean(randomeff$completeMEM2 - randomeff$FCSMEM2)
# -4.437717e-05

# MEM1, jomo
mean(randomeff$completeMEM1 - randomeff$jomoMEM1)
# 0.02887671

# MEM2, jomo
mean(randomeff$completeMEM2 - randomeff$jomoMEM2)
# 0.02425991

# Calculate MSE
# MEM1, FCS
mean((randomeff$completeMEM1 - randomeff$FCSMEM1)^2)
# 9.237626e-08

# MEM2, FCS
mean((randomeff$completeMEM2 - randomeff$FCSMEM2)^2)
# 1.631998e-07

# MEM1, jomo
mean((randomeff$completeMEM1 - randomeff$jomoMEM1)^2)
# 0.0009609373

# MEM2, jomo
mean((randomeff$completeMEM2 - randomeff$jomoMEM2)^2)
# 0.000627878
```

Bibliography

Siddique, Juned, Jerome P Reiter, et al. (2015). "Multiple imputation for harmonizing longitudinal non-commensurate measures in individual participant data meta-analysis". In: *Statistics in medicine* 34.26, pp. 3399–3414.

Gaye, Amadou et al. (2014). "DataSHIELD: taking the analysis to the data, not the data to the analysis". In: *International journal of epidemiology* 43.6, pp. 1929–1944.

Graham, John W (2009). "Missing data analysis: Making it work in the real world". In: *Annual review of psychology* 60, pp. 549–576.

Griffith, Lauren E, Edwin Van Den Heuvel, et al. (2015). "Statistical approaches to harmonize data on cognitive measures in systematic reviews are rarely reported". In: *Journal of clinical epidemiology* 68.2, pp. 154–162.

Fortier, Isabel et al. (2016). "Maelstrom Research guidelines for rigorous retrospective data harmonization". In: *International journal of epidemiology*, dyw075.

Thomas, Doneal, Sanyath Radji, and Andrea Benedetti (2014). "Systematic review of methods for individual patient data meta-analysis with binary outcomes". In: *BMC medical research methodology* 14.1, p. 79.

Resche-Rigon, Matthieu and Ian R White (2016). "Multiple imputation by chained equations for systematically and sporadically missing multilevel data". In: *Statistical Methods in Medical Research*, p. 0962280216666564.

Griffith, Lauren et al. (2013). "Harmonization of cognitive measures in individual participant data and aggregate data meta-analysis". In:

Ahmed, Ikhlaaq et al. (2014). "Developing and validating risk prediction models in an individual participant data meta-analysis". In: *BMC medical research methodology* 14.1, p. 3.

Audigier, Vincent, Ian R White, et al. (2017). "Multiple imputation for multilevel data with continuous and binary variables". In: *arXiv preprint arXiv:1702.00971*.

Jolani, Shahab et al. (2015). "Imputation of systematically missing predictors in an individual participant data meta-analysis: a generalized approach using MICE". In: *Statistics in medicine* 34.11, pp. 1841–1863.

Quartagno, M and JR Carpenter (2015). "Multiple imputation for IPD meta-analysis: allowing for heterogeneity and studies with missing covariates". In: *Statistics in medicine*.

Siddique, Juned, Peter J de Chavez, et al. (2016). "Limitations in using multiple imputation to harmonize individual participant data for meta-analysis". In: *Prevention Science*, pp. 1–14.

Magalhaes, S and C Wolfson (2012). "Harmonization: a methodology for advancing research in multiple sclerosis". In: *Acta Neurologica Scandinavica* 126.s195, pp. 31–35.

Quartagno, Matteo (2016). "Multiple Imputation for Individual Patient Data Meta-Analyses." PhD thesis. London School of Hygiene & Tropical Medicine.

Rubin, Donald B (1986). "Statistical matching using file concatenation with adjusted weights and multiple imputations". In: *Journal of Business & Economic Statistics* 4.1, pp. 87–94.
DerSimonian, Rebecca and Nan Laird (1986). "Meta-analysis in clinical trials". In: *Controlled clinical trials* 7.3, pp. 177–188.
Hartung, Joachim and Guido Knapp (2001). "A refined method for the meta-analysis of controlled clinical trials with binary outcome". In: *Statistics in Medicine* 20.24, pp. 3875–3889.
Debray, Thomas PA et al. (2013). "Individual participant data meta-analysis for a binary outcome: one-stage or two-stage?" In: *PloS one* 8.4, e60650.
Burke, Danielle L, Joie Ensor, and Richard D Riley (2017). "Meta-analysis using individual participant data: one-stage and two-stage approaches, and why they may differ". In: *Statistics in medicine* 36.5, pp. 855–875.
Higgins, Julian PT and Sally Green (2011). *Cochrane handbook for systematic reviews of interventions*. Vol. 4. John Wiley & Sons.
Tierney, Jayne F et al. (2015). "Individual participant data (IPD) meta-analyses of randomised controlled trials: guidance on their use". In: *PLoS Med* 12.7, e1001855.
Conti, Pier Luigi, Daniela Marella, and Mauro Scanu (2017). "How far from identifiability? A systematic overview of the statistical matching problem in a non parametric framework". In: *Communications in Statistics-Theory and Methods* 46.2, pp. 967–994.
Van Buuren, Stef (2007). "Multiple imputation of discrete and continuous data by fully conditional specification". In: *Statistical methods in medical research* 16.3, pp. 219–242.
Rubin, Donald B (2004). *Multiple imputation for nonresponse in surveys*. Vol. 81. John Wiley & Sons.
Van Buuren, Stef et al. (2006). "Fully conditional specification in multivariate imputation". In: *Journal of statistical computation and simulation* 76.12, pp. 1049–1064.
Marshall, Andrea et al. (2009). "Combining estimates of interest in prognostic modelling studies after multiple imputation: current practice and guidelines". In: *BMC medical research methodology* 9.1, p. 57.
Enders, Craig K and Deborah L Bandalos (2001). "The relative performance of full information maximum likelihood estimation for missing data in structural equation models". In: *Structural equation modeling* 8.3, pp. 430–457.
Akl, Elie A et al. (2015). "Handling trial participants with missing outcome data when conducting a meta-analysis: a systematic survey of proposed approaches". In: *Systematic reviews* 4.1, p. 98.
Resche-Rigon, Matthieu, Ian R White, et al. (2013). "Multiple imputation for handling systematically missing confounders in meta-analysis of individual participant data". In: *Statistics in Medicine* 32.28, pp. 4890–4905.
Rässler, Susanne (2003). "A Non-Iterative Bayesian Approach to Statistical Matching". In: *Statistica Neerlandica* 57.1, pp. 58–74.
Yucel, Recai M (2011). "Random covariances and mixed-effects models for imputing multivariate multilevel continuous data". In: *Statistical modelling* 11.4, pp. 351–370.
Gilula, Zvi and Robert McCulloch (2013). "Multi level categorical data fusion using partially fused data". In: *Quantitative Marketing and Economics* 11.3, pp. 353–377.
Kamakura, Wagner A and Michel Wedel (1997). "Statistical data fusion for cross-tabulation". In: *Journal of Marketing Research*, pp. 485–498.
Kiesl, Hans and Susanne Rässler (2006). "How valid can data fusion be". In:

Reiter, Jerome P (2008). "Multiple imputation when records used for imputation are not used or disseminated for analysis". In: *Biometrika* 95.4, pp. 933–946.

Van Buuren, S et al. (2003). "Assessing comparability of dressing disability in different countries by response conversion". In: *The European Journal of Public Health* 13.suppl 3, pp. 15–19.

Carrig, Madeline M et al. (2015). "A nonparametric, multiple imputation-based method for the retrospective integration of data sets". In: *Multivariate behavioral research* 50.4, pp. 383–397.

Crane, Paul K et al. (2008). "Item response theory facilitated cocalibrating cognitive tests and reduced bias in estimated rates of decline". In: *Journal of clinical epidemiology* 61.10, pp. 1018–1027.

Schifeling, Tracy, Jerome P Reiter, and Maria DeYoreo (2016). "Data Fusion for Correcting Measurement Errors". In: *arXiv preprint arXiv:1610.00147*.

Curran, Patrick J and Andrea M Hussong (2009). "Integrative data analysis: the simultaneous analysis of multiple data sets." In: *Psychological methods* 14.2, p. 81.

Conti, Pier Luigi, Daniela Marella, and Mauro Scanu (2013). "Uncertainty analysis for statistical matching of ordered categorical variables". In: *Computational Statistics & Data Analysis* 68, pp. 311–325.

— (2016). "Statistical matching analysis for complex survey data with applications". In: *Journal of the American Statistical Association* 111.516, pp. 1715–1725.

D'Orazio, Marcello, Marco Di Zio, and Mauro Scanu (2012). "Statistical Matching of Data from Complex Sample Surveys". In: *Proceedings of the European Conference on Quality in Official Statistics-Q2012*. Vol. 29.

Fleischer, Karlheinz and Heiko Groenitz (2013). "Discussion Papers on Statistics and Quantitative Methods". In:

Reiter, Jerome P (2012). "Bayesian finite population imputation for data fusion". In: *Statistica Sinica*, pp. 795–811.

Zhang, Li-Chun (2015). "On Proxy Variables and Categorical Data Fusion". In: *Journal of Official Statistics* 31.4, pp. 783–807.

Vogels, Raymond LC et al. (2007). "Cognitive impairment in heart failure: a systematic review of the literature". In: *European Journal of Heart Failure* 9.5, pp. 440–449.

Harrison, Jennifer Kirsty et al. (2016). "Outcomes measures in a decade of dementia and mild cognitive impairment trials". In: *Alzheimer's research & therapy* 8.1, p. 48.

Chen, Ling et al. (2005). "Multiple imputation for missing ordinal data". In: *Journal of Modern Applied Statistical Methods* 4.1, p. 26.

Wu, Wei, Fan Jia, and Craig Enders (2015). "A comparison of imputation strategies for ordinal missing data on Likert scale variables". In: *Multivariate behavioral research* 50.5, pp. 484–503.

Quartagno, Matteo and James Carpenter (2017). *jomo: A package for Multilevel Joint Modelling Multiple Imputation*. URL: `https://CRAN.R-project.org/package=jomo`.

Audigier, Vincent and Matthieu Resche-Rigon (2017). *micemd: Multiple Imputation by Chained Equations with Multilevel Data*. URL: `https://cran.r-project.org/web/packages/micemd/`.

Griffith, Lauren E, Edwin van den Heuvel, et al. (2016). "Comparison of Standardization Methods for the Harmonization of Phenotype Data: An Application to Cognitive Measures". In: *American journal of epidemiology*, pp. 1–9.

Pasture, Ottawa and O Onkia (1994). "Canadian Study of Health and Aging: study methods and prevalence of dementia". In: *Canadian Medical Assocociation Journal* 150.6, pp. 899–913.

Canada, Statistics (2008). *Canadian Community Health Survey – Healthy aging (CCHS)*. `http://www23.statcan.gc.ca/imdb/p2SV.pl?Function=getSurvey&SDDS=5146`. Accessed: 2017-08-24.

Gaudreau, Pierrette et al. (2007). "Nutrition as a determinant of successful aging: description of the Quebec longitudinal study Nuage and results from cross-sectional pilot studies". In: *Rejuvenation research* 10.3, pp. 377–386.

Samtani, Mahesh N et al. (2015). "Alzheimer's disease assessment scale-cognitive 11-item progression model in mild-to-moderate Alzheimer's disease trials of bapineuzumab". In: *Alzheimer's & Dementia: Translational Research & Clinical Interventions* 1.3, pp. 157–169.

Wishart Distribution. `https://www.mathworks.com/help/stats/wishart-distribution.html?requestedDomain=www.mathworks.com`. Accessed: 2017-05-30.

Inverse Wishart Distribution. `https://www.mathworks.com/help/stats/inverse-wishart-distribution.html`. Accessed: 2017-05-30.