Scientific Research Publishing

Open Journal of Statistics, 2023, 13, 316-358
https://www.scirp.org/journal/ojs
ISSN Online: 2161-7198
ISSN Print: 2161-718X

Superiority of Bayesian Imputation to Mice in Logit Panel Data Models

Peter Otieno Opeyo[1,2], Weihu Cheng[1], Zhao Xu[1]

[1]Department of Statistics, Beijing University of Technology, Beijing, China
[2]Department of Statistics and Computational Mathematics, The Technical University of Kenya, Nairobi, Kenya
Email: opeyopeter@gmail.com, chengweihu@bjut.edu.cn, zhaox@bjut.edu.cn

How to cite this paper: Opeyo, P.O., Cheng, W.H. and Xu, Z. (2023) Superiority of Bayesian Imputation to Mice in Logit Panel Data Models. *Open Journal of Statistics*, **13**, 316-358.
https://doi.org/10.4236/ojs.2023.133017

Received: May 8, 2023
Accepted: June 13, 2023
Published: June 16, 2023

Abstract

Non-responses leading to missing data are common in most studies and causes inefficient and biased statistical inferences if ignored. When faced with missing data, many studies choose to employ complete case analysis approach to estimate the parameters of the model. This however compromises on the susceptibility of the estimates to reduced bias and minimum variance as expected. Several classical and model based techniques of imputing the missing values have been mentioned in literature. Bayesian approach to missingness is deemed superior amongst the other techniques through its natural self-lending to missing data settings where the missing values are treated as unobserved random variables that have a distribution which depends on the observed data. This paper digs up the superiority of Bayesian imputation to Multiple Imputation with Chained Equations (MICE) when estimating logistic panel data models with single fixed effects. The study validates the superiority of conditional maximum likelihood estimates for nonlinear binary choice logit panel model in the presence of missing observations. A Monte Carlo simulation was designed to determine the magnitude of bias and root mean square errors (RMSE) arising from MICE and Full Bayesian imputation. The simulation results show that the conditional maximum likelihood (ML) logit estimator presented in this paper is less biased and more efficient when Bayesian imputation is performed to curb non-responses.

Keywords

Panel Data, Imputation, Monte Carlo, Bias, Conditional Maximum Likelihood

1. Introduction

Complications in statistical analyses due to missing values in data sets have been of major concern to researchers in virtually all fields of study. If the response variable is binary, these complications are compounded by the nonlinear nature of model specification adopted to relate the continuous covariates to the categorical response. Numerous studies show that frequentist approaches to missing data, like complete case analysis, often lead to biased estimates and substantial loss of power [1] [2] [3] [4]. The logit panel data model may therefore not be an exception to this problem.

Model based techniques of imputation have proved to be more reliable than single value imputation. One customary model-based approach is to perform multiple imputations with chained equations (MICE) [4] [5] which has been widely accepted due to its good performance and ease of use. Available statistical tools designed for MICE operate in three key stages: 1) a small number (usually five or ten) of data sets are created by imputing each missing value multiple times; 2) analyzing each of the completed data sets using known complete data methods; 3) obtaining the overall result by pooling the derived estimates to take additional uncertainty created by the missing values into account. The separation of the imputation and analysis stages of MICE calls for the need to have imputation models which contain not only other covariates in the predictor function but also the outcome [6]. When the outcome is univariate and the data is complete, this can be easily performed because the outcome is just one of the variables in the data set. However, when the outcome has a multivariate nature, such as a binary outcome study, the outcome variable inclusion function becomes subjective and may take simple or complex summaries of the long trajectories, such as including only a single observed value, average value or the area under the trajectory. Evidently, it is not easy to settle for the most adequate representation for a specific analysis model, and except in very simple situations, some of the summaries adopted discard relevant information. Comparative assessment done herein shows that the inclusion of inadequate summary measures of subjects' trajectories can lead to increased bias.

A full Bayesian imputation technique has been observed to combine the analysis model with the imputation models without having to specify an appropriate summary measure for the outcome variable. The milestone difference to MICE is that by combining the imputation and analysis in one estimation procedure, the Bayesian approach obtains inferences on the posterior distribution of the parameters and missing covariates directly and jointly. Thereby, the whole trajectory of the longitudinal outcome is implicitly taken into account in the imputation of missing covariate values, and no summary representation is necessary. This study embraces the works of Ibrahim *et al.* [7], who propose a decomposition of the likelihood into a series of univariate conditional distributions which produces the sequential full Bayesian (SFB) approach that is flexible and easy to implement as an alternative to MICE. Besides, the uncertainty about the missing values is automatically taken into account and no pooling is necessary since missing values are continuously imputed in each iteration of the estimation procedure. Stubbendick and Ibrahim [8], in their study used the likelihood based approach that factorized the joint likelihood into a sequence of conditional distributions. This approach is akin to SFB except for the model used. In addition, other authors have shown how to apply weighted estimating equations for inference in settings with incomplete data [9] [10]. Generally, studies agree that model based imputation techniques are superior to single value imputations. However, not much literature is found on conditional logit models as it were.

In this study, we describe a few strategies to include a binary outcome in the MICE algorithm and compare them with the Bayesian approach. Each of the methods was evaluated using simulation from which biases and root mean square errors (RMSE) are compared to discern the better of the two methods.

We structure the rest of the paper as follows: Section 2 briefly describes the logit panel data model. In Section 3, we introduce the problem of missing data and describe and compare the two methods of interest, MICE, and the SFB approach. Both methods are applied to simulated data in Section 4. An evaluation of the bias and RMSE for both approaches is described in Section 5 where the findings are summarized and discussed.

2. Model Specification

2.1. The Logit Panel Data Model

A general panel data model is a special case of the generalized longitudinal model where a population unit *i* is observed across different time periods and takes the form

$$y_{it} = \boldsymbol{x}_{it}'\boldsymbol{\beta} + c_i + \gamma_t + u_{it},\; i = 1,\cdots,N; t = 1,\cdots,T \tag{1}$$

where y_{it} is the *t*th observation of individual *i*, measured at time *t*, $\boldsymbol{\beta}$ denotes the vector of *k* regression coefficients of the design matrix of the fixed effects $\boldsymbol{X}_i$, where $\boldsymbol{x}_{it}$ is a column vector that contains the *t*th row of that matrix. c_i is an individual-specific time-invariant parameter while γ_t is a time-specific individual-invariant parameter. u_{it} is an error term that is normally distributed with mean zero and variance σ_y^2. In many studies, individuals or subjects are assumed to change over time, and that not any two of them have the same characteristics at a time *t*. As such, we only have the individual specific effects, letting $\gamma_t = 0$, Equation (1) takes the form

$$y_{it} = \boldsymbol{x}_{it}'\boldsymbol{\beta} + c_i + u_{it} \tag{2}$$

The association between c_i and u_{it} establishes whether the relation (2) is treated as fixed or random effects model. This is to say that if c_i is correlated with $\boldsymbol{x}_{it}$ then the model has only u_{it} as the stochastic part and c_i is treated as fixed (non-random) and consequentially, we have a fixed effect (FE) panel model. On the other hand, (2) is a random effect (RE) model, if c_i becomes part of the stochastic part u_{it}. Cumulative Equation (2) has a total of $k + N$ parameters to be estimated as $\hat{\boldsymbol{\beta}}$ and $\hat{c}_i$. For FE, one does not estimate the effects of the variables that are individual specific and time-invariant but rather controls for them or “partials them out” to reduce the bias from other omitted variables.

For a continuous dependent variable y_{it}, the parameters in the panel data model can be estimated unbiasedly and efficiently. Unfortunately, however, the dependent variable may be categorical which calls for specific nonlinear functions that preserve the structure of the dependent variable to be considered. The logistic function is one such nonlinear relation that yields the logit model. Conditional maximum likelihood estimation is the most preferred method for logistic regressions. In this method, the conditional maximum likelihood "conditions" the (fixed effects) out of the likelihood function [16]. This is done by conditioning the likelihood function on the total number of events observed for each subject as was first explained by Chamberlain [11].

Most economic studies have the dependent variable as categorical with two (or more) options indicating a success or a failure of an event. Such dependent variable is normally represented by a binary choice variable $y_{it} = \begin{cases} 1 & \text{if the event happens} \\ 0 & \text{if the event fails} \end{cases}$ for individual *i* at time *t*. Then y_{it} follows a binomial distribution with probability of success for individual *i* at time *t* as p_{it} and

$$E(y_{it}) = 1 \times p_{it} + 0 \times (1 - p_i) = p_{it} \tag{3}$$

This expected value of the binary outcome is modeled as a function of some explanatory variables or covariates as

$$p_{it} = Pr(y_{it} = 1) = E(y_{it} \mid \mathbf{x}_{it}, c_i) = F(\mathbf{x}'_{it}\boldsymbol{\beta} + c_i) \tag{4}$$

where $F(\cdot)$ is a link function that relates the binary outcome to the various types of covariates in. Here, we assume strict exogeneity holds *i.e.* the residual u_{it} is uncorrelated with all *x*-variables over the entire time period spanned by the panel so that u_{it} has a zero-mean.

If *F* is the unit function for all $\mathbf{x}'_{it}\boldsymbol{\beta} + c_i$ then under random sampling, the unconditional probability that $y_{it} = 1$ is equal to the unconditional expected value of y_{it}, *i.e.* $Pr(=1 \mid \mathbf{x}_{it}, c_i) = E(y_{it} = 1 \mid \mathbf{x}_{it}, c_i; \beta)$

So if the binary response model above is correctly specified, we have

$$\left.\begin{aligned} Pr(y_{it} = 1 \mid \mathbf{x}_{it}, c_i) &= \mathbf{x}'_{it}\boldsymbol{\beta} + c_i \\ Pr(y_{it} = 0 \mid \mathbf{x}_{it}, c_i) &= 1 - (\mathbf{x}'_{it}\boldsymbol{\beta} + c_i) \end{aligned}\right\} \tag{5}$$

Model (5) is referred to as the linear probability model (LPM) since the probability of success is expressed as a linear function of the explanatory variables in the vector $\mathbf{x}$ and the parameters can be estimated by OLS or within estimator. A major shortfall of LPM when used to estimate the parameters for any discrete choice response variable is that we can get predicted "probabilities" either less than zero or greater than one which of course absurdly contravenes the definition of probability.

The problems of LPM can be addressed by choosing a monotonically increasing function *F* such that $0 < F(\mathbf{x}'_{it}\boldsymbol{\beta} + c_i) < 1$ and

$$\left.\begin{aligned} Pr(y_{it} = 1 \mid \mathbf{x}_{it}, c_i) &\to 1 \text{ as } \mathbf{x}_{it}\boldsymbol{\beta} + c_i \to \infty \\ Pr(y_{it} = 1 \mid \mathbf{x}_{it}, c_i) &\to 0 \text{ as } \mathbf{x}_{it}\boldsymbol{\beta} + c_i \to -\infty \end{aligned}\right\}. \tag{7}$$

Thus F is a nonlinear function, and hence we cannot use a linear regression model estimation techniques. Various non-linear functions for F have been suggested in the literature and the most common ones are the logistic distribution, yielding the logit model, and the standard normal distribution, yielding the probit model.

In the logit model, F takes the form,

$$F\left(\boldsymbol{x}_{it}'\boldsymbol{\beta}+c_i\right)=\frac{\mathrm{e}^{x_{it}'\boldsymbol{\beta}+c_i}}{1+\mathrm{e}^{x_{it}'\boldsymbol{\beta}+c_i}} \tag{8}$$

which is between zero and one for all values of $\boldsymbol{x}_{it}'\boldsymbol{\beta}$. This is the cumulative distribution function (CDF) for a logistic variable whose parameters can be estimated.

2.2. Implications of the Probability Function *Pr*(.) for Binary Outcomes

When the outcome variable is binary by specification, its probability function *Pr*(.) only permits two plausible values for either a success or a failure irrespective of the functional forms of the covariates in the model. Overlooking this property of the binary outcome yields a linear probability model (LPM) (5) which has quite a number of limitations. Among these limitations are: 1) we can get predicted "probabilities" either less than zero or greater than one from the LPM which is absurd since predictions outside this range are meaningless and somewhat embarrassing; 2) conceptually, it does not make sense to say that a probability is linearly related to a continuous independent variable for all possible values. If it were, then continually increasing this explanatory variable would eventually drive $P\left(y=1|x\right)$ above one or below zero; 3) the residual of the LPM is heteroskedastic and the only best way of solving this problem is to obtain estimates of the standard errors that are robust to heteroskedasticity; 4) the residual is not normally distributed hence small sample inferences cannot be based on the usual suite of normality-based distributions such as the t test.

Equation (8) now represents what is known as the (cumulative) logistic distribution function which cushions against the limitations of LPM. It is easy to verify that as $\boldsymbol{x}_{it}'\boldsymbol{\beta}+c_i$ ranges from −∞ to +∞, F ranges between 0 and 1 and that F is nonlinearly related to $\boldsymbol{x}_{it}'\boldsymbol{\beta}+c_i$ (*i.e.*, $\boldsymbol{x}_{it}'$), thus satisfying the requirements for a probability function. However, in satisfying these requirements, we have created an estimation problem because F is nonlinear not only in X but also in the βs as can be seen clearly from (8). This implies that we cannot use the familiar OLS procedure to estimate the parameters. But through linearization, this problem becomes more apparent than real. This is done by obtaining the odds ratio $\frac{Pr\left(y_{it}=1|\boldsymbol{x}_{it},c_i\right)}{Pr\left(y_{it}=0|\boldsymbol{x}_{it},c_i\right)}$ in favor of a success *i.e.* the ratio of the probability that $y_{it}=1$ to the probability that $y_{it}=0$. It is realized that the logarithm of the odds ratio, is not only linear in X, but also (from the estimation viewpoint) linear in the parameters. $\ln\left(\frac{Pr(y_{it}=1\mid \boldsymbol{x}_{it},c_i)}{Pr(y_{it}=0\mid \boldsymbol{x}_{it},c_i)}\right)=\boldsymbol{x}'_{it}\boldsymbol{\beta}+c_i$. This log of the odds ratio is called the logit, and hence the name logit model.

3. Dealing with Missing Data—Model-Based Approach for Logit Panel Data Models

The relationship between a binary outcome $\boldsymbol{y}$ and predictor variables $\boldsymbol{X}$ is a linear mixed model expressed in a standard modeling framework as Equation (1).

If the data setting is complete and balanced, the probability density function of interest is $p(y_{it}\mid \boldsymbol{x}_{it},\boldsymbol{\theta}_{Y|X})$ where $\boldsymbol{\theta}_{Y|X}$ denotes the vector of all parameters of the model. Conversely, when missingness results in some of the covariates being incomplete, $\boldsymbol{X}$ is partitioned into two parts, $\boldsymbol{X}=[\boldsymbol{X}_{obs}\ \boldsymbol{X}_{mis}]$**:** the completely observed variables $\boldsymbol{X}_{obs}$ and those variables containing missing value $\boldsymbol{X}_{mis}$. We then have a measurement model that depends on unobserved data expressed as $p(y_{it}\mid \boldsymbol{x}_{it,obs},\boldsymbol{x}_{it,mis},\boldsymbol{\theta}_{Y|X})$, which cannot be handled by the standard complete data methods efficiently.

Two key assumptions now need to be made so that we may not require any digression from standard approaches of handling mixed effects models. These assumptions are (a) that we only have cross-sectional covariates to impute and (b) that the missing data mechanism of the outcome is ignorable, that is, missing at random (MAR) or missing completely at random (MCAR) [12].

As it were, despite having various frequentist techniques of imputing on missing data, we narrow our imputation to model-based techniques and give in-depth discussion on multiple imputation using chained equations (MICE) and sequential full Bayesian approach (SFB). This concept of replacing a missing datum with multiple values was hatched by Donald B. Rubin [13]. He stated that

"[…] *of course* (1) *imputing one value for missing datum can't be correct in general, and* (2) *in order to insert sensible values for a missing datum we must rely more or less on some model relating unobserved values to observed values.*"

1) *Multiple imputations using chained equations* (*MICE*)

The underlying principle of MI is to divide the analysis of incomplete data into three steps: MICE has become a popular imputation technique by the fact that it allows for multivariate missing data and does not require a specific missingness pattern. Since the core principle of multiple imputations is divided into three stages (imputation, analysis, and pooling), MICE becomes very vital in addressing the initial stage of imputation.

Performing MICE under certain regularity conditions, the multivariate distribution

$$p(x_{i,mis}\mid y_i,x_{i,obs},\boldsymbol{\theta}_X) \tag{9}$$

with $x_{i,mis}=(x_{i,mis_1},\cdots,x_{i,mis_p})'$ and $x_{i,obs}=(x_{i,obs_1},\cdots,x_{i,obs_q})'$ can be uniquely determined by its full-conditional distributions. Van Buuren *et al.* [14] assert that through this unique determination, Gibbs sampling of the conditional distributions can be used to produce a sample from (9). Though the MICE procedure does not actually start from a specification of (9), it directly defines a series of conditional, predictive models of the form

$$p\left(x_{i,mis_\ell} \mid x_{i,mis_{-\ell}}, x_{i,obs}, y_i, \boldsymbol{\theta}_{X_\ell}\right) \tag{10}$$

which links each incomplete predictor variable x_{i,mis_ℓ}, $\ell = 1, \cdots, p$, with other incomplete and complete predictors, $\boldsymbol{x}_{i,mis_{-\ell}}$ and $\boldsymbol{x}_{i,obs}$, respectively, without forgetting the response y_i. The predictive distributions (1.2) are drawn from the extended exponential family with linear predictor expressed as

$$g_\ell\left\{E\left(x_{i,mis_\ell} \mid x_{i,mis_{-\ell}}, x_{i,obs}, \boldsymbol{y}_i, \boldsymbol{\gamma}_\ell, \boldsymbol{\xi}_\ell, \boldsymbol{\alpha}_\ell\right)\right\} = \boldsymbol{\gamma}'_\ell \boldsymbol{x}_{i,obs} + \boldsymbol{\xi}'_\ell \boldsymbol{x}_{i,mis_{-\ell}} + \boldsymbol{\alpha}'_\ell h\left(\boldsymbol{y}_i\right)$$

where $g_\ell(\cdot)$ is the one-to-one monotonic link function for the ℓth covariate and $\boldsymbol{\gamma}_\ell$, $\boldsymbol{\xi}_\ell$ and $\boldsymbol{\alpha}_\ell$ are vectors of parameters relating the complete and missing covariates and the outcome to x_{i,mis_ℓ}.

The function $h(\cdot)$ specifies how the outcome enters the linear predictor. In the univariate case, the default choice for $h(\boldsymbol{y}_i)$ is simply the identity function. However, when we have a multivariate y_i, such as a binary outcome, we cannot always simply specify $\boldsymbol{\alpha}'\boldsymbol{y}_i$ because $\boldsymbol{y}_i$ may take on two different values for an individual *i* at time *t*. Hence, it is not meaningful to use the same regression coefficient $\boldsymbol{\alpha}_{\ell j}$ to connect outcomes of different individuals with x_{i,mis_ℓ}, and a representation needs to be found that summarizes $\boldsymbol{y}_i$ and that has the same number of elements that also have the same interpretation for all individuals.

We choose to proposed some relations for $h(\boldsymbol{y}_i)$ in this study as below

$$h\left(\boldsymbol{y}_i\right) = 0, \tag{11}$$

$$h\left(\boldsymbol{y}_i\right) = y_{it}, \tag{12}$$

$$h\left(\boldsymbol{y}_i\right) = \frac{1}{T}\sum_{t=1}^{T} y_{it}, \tag{13}$$

$$h\left(\boldsymbol{y}_i\right) = \sum_{t=1}^{T-1}\left(s_{it+1} - s_{it}\right)\frac{y_{it} + y_{it+1}}{2}. \tag{14}$$

From the above relations, it can be deduced that $h(\boldsymbol{y}_i)$ from:

i) (11) omits the response completely from the relation $g_\ell(\cdot)$.

ii) (12) uses only one observation chosen from the outcomes for i^{th} individual.

ii) (13) uses the average of the observed outcomes.

iv) (14) uses the trapezoidal area under the curve of $\boldsymbol{y}_i$ around the observation times *t* and *t* + 1 as exemplified in Figure 1 below.

Linear combinations of the above relations may still suffice and so there can be infinitely many options of specifying the predictor function *g*(·). It is not therefore a simple task to determine an appropriate $\boldsymbol{y}_i$ to use unless the setting is too simple as well. Only in very simple settings is it possible to determine which function of $\boldsymbol{y}_i$ is appropriate. Under a random intercept model for $\boldsymbol{y}_i$, (13) is the appropriate summary function in the imputation model for a normal

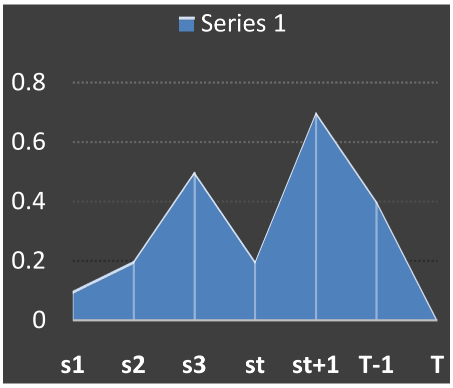

Figure 1. Shows the area under the trajectory used as a summary of the response variable y over the study periods.

cross-sectional covariate [15] whereas for more complex analysis models or discrete covariates, it is not straightforward to derive the appropriate summary functions.

The splendid of all inclusions of the outcome variable into the predictor function would be to one in which all available outcome variables are captured in the function $h(\boldsymbol{y}_i)$. This however can only be possible in settings where the outcome is balanced or close to balanced and does not have a large number of repeated measurements. Alternatively, we could impute missing outcome values so that all individuals have the same number of measurements at approximately the same time points.

In order to achieve high validity of imputations using MICE, we need to ensure that (a) the imputation models are correctly specified and (b) the missing data mechanism is ignorable—MAR or MCAR [12] [16].

2) ***Full Bayesian imputation***

Since MICE may be faced by the challenge of choosing the best summary representation of a multivariate outcome, we can settle for a full Bayesian approach. In this approach, the complete data posterior is obtained by combining the complete data likelihood with prior information to yield

$$\begin{aligned} & p\left(\theta_{Y|X}, \theta_X, \boldsymbol{x}_{i,mis} \mid y_{it}, \boldsymbol{x}_{i,obs}\right) \\ & \propto p\left(y_{it} \mid \boldsymbol{x}_{i,mis}, \boldsymbol{x}_{i,obs}, \theta_{Y|X}\right) p\left(\boldsymbol{x}_{i,mis} \mid \boldsymbol{x}_{i,obs}, \theta_X\right) \pi\left(\theta_{Y|X}\right) \pi\left(\theta_X\right) \end{aligned}$$

where θ_X is a vector containing parameters that are associated with the likelihood of the partially observed covariates $\boldsymbol{X}_{mis}$, and $\pi\left(\theta_{Y|X}\right)$ and $\pi\left(\theta_X\right)$ are prior distributions. Ibrahim *et al.*, [7] explain that a convenient way to specify the joint likelihood of the missing covariates $p\left(\boldsymbol{x}_{i,mis} \mid \boldsymbol{x}_{i,obs}, \theta_X\right)$ is to use a sequence of conditional univariate distributions

$$\begin{aligned} & p\left(x_{i,mis_i}, \cdots, x_{i,mis_p} \mid \boldsymbol{x}_{i,obs}, \theta_X\right) \\ & = p\left(x_{i,mis_1} \mid \boldsymbol{x}_{i,obs}, \theta_{X_1}\right) \prod_{\ell=2}^{p} p\left(x_{i,mis_\ell} \mid \boldsymbol{x}_{i,obs}, x_{i,mis_1}, \cdots, x_{i,mis_{l-1}}, \theta_{X_\ell}\right) \end{aligned} \tag{15}$$

with $\theta_X = \left(\theta_{X_1}, \cdots, \theta_{X_p}\right)'$. It is similarly assumed that each of these distributions is again chosen from the extended exponential family and in accordance with the type of the respective variable. An advantage of this sequential way of writing the joint distribution of the covariates is that it provides a straightforward way to specify the joint distribution even when the covariates are of mixed type.

After specifying the prior distributions $\pi\left(\theta_{Y|X}\right)$ and $\pi\left(\theta_X\right)$, Markov chain Monte Carlo (MCMC) methods, such as Gibbs sampling, can be used to draw samples from the joint posterior distribution of all parameters and missing values. Because all missing values are imputed in each iteration of the Gibbs sampler, the additional uncertainty created by the missing values is automatically taken into account, and no pooling is necessary.

We opt to adopt the full likelihood instead of the series of predictive models (10) due to one key advantage: that we can choose how to factorize this full likelihood. Precisely, by factorizing the joint distribution $p\left(y_{it}, \boldsymbol{x}_{i,mis}, \boldsymbol{x}_{i,obs} \mid \theta_{Y|X}, \theta_X\right)$ in to the conditional distribution $p\left(y_{it} \mid \boldsymbol{x}_{i,mis}, \boldsymbol{x}_{i,obs}, \theta_{Y|X}\right)$ and the marginal distribution $p\left(\boldsymbol{x}_{i,mis} \mid \boldsymbol{x}_{i,obs}, \theta_X\right)$, the joint posterior distribution can be specified without having to include the outcome into any predictor, and no summary representation $h\left(\boldsymbol{y}_i\right)$ is needed. This becomes clear when writing out the full conditional distribution of the incomplete covariates, used by the Gibbs sampler:

$$\begin{aligned}
& p\left(x_{i,mis_l} \mid y_i, \boldsymbol{x}_{i,obs}, \boldsymbol{x}_{i,mis_{-l}}, \boldsymbol{\theta}\right) \\
& \propto \left\{\prod_{t=1}^{T} p\left(y_{it} \mid \boldsymbol{x}_{i,mis}, \boldsymbol{x}_{i,obs}, \boldsymbol{\theta}_{Y|X}\right)\right\} p\left(\boldsymbol{x}_{i,mis} \mid \boldsymbol{x}_{i,obs}, \theta_X\right) \pi\left(\boldsymbol{\theta}_{Y|X}\right) \pi\left(\boldsymbol{\theta}_X\right) \\
& \propto \left\{\prod_{t=1}^{T} p\left(y_{it} \mid \boldsymbol{x}_{i,mis}, \boldsymbol{x}_{i,obs}, \boldsymbol{\theta}_{Y|X}\right)\right\} p\left(\boldsymbol{x}_{i,mis_l} \mid \boldsymbol{x}_{i,obs}, \boldsymbol{x}_{i,mis_{<l}}, \theta_{X_l}\right) \\
& \quad \left\{\prod_{k=l+1}^{p} p\left(\boldsymbol{x}_{i,mis_k} \mid \boldsymbol{x}_{i,obs}, \boldsymbol{x}_{i,mis_{<k}}, \theta_{X_k}\right)\right\} \pi\left(\boldsymbol{\theta}_{Y|X}\right) \pi\left(\boldsymbol{\theta}_{X_l}\right) \prod_{k=l+1}^{p} \pi\left(\boldsymbol{\theta}_{X_k}\right)
\end{aligned}$$

where densities $p\left(y_{it} \mid \boldsymbol{x}_{i,mis}, \boldsymbol{x}_{i,obs}, \boldsymbol{\theta}_{Y|X}\right)$, $p\left(\boldsymbol{x}_{i,mis_l} \mid \boldsymbol{x}_{i,obs}, \boldsymbol{x}_{i,mis_{<l}}, \theta_{X_l}\right)$ and $p\left(\boldsymbol{x}_{i,mis_k} \mid \boldsymbol{x}_{i,obs}, \boldsymbol{x}_{i,mis_{<k}}, \theta_{X_k}\right)$ are members of the extended exponential family with linear predictors expressed as

$$E\left(y_{it} \mid \boldsymbol{x}_{i,mis}, \boldsymbol{x}_{i,obs}, \boldsymbol{\theta}_{Y|X}\right) = \boldsymbol{\gamma}'_y \boldsymbol{x}_{i,obs} + \boldsymbol{\xi}'_y \boldsymbol{x}_{i,mis} \tag{16}$$

$$g_l\left\{E\left(\boldsymbol{x}_{i,mis_l} \mid \boldsymbol{x}_{i,obs}, \boldsymbol{x}_{i,mis_{<l}}, \theta_{X_l}\right)\right\} = \boldsymbol{\gamma}'_y \boldsymbol{x}_{i,obs} + \sum_{s=1}^{l-1} \boldsymbol{\xi}'_{l_s} \boldsymbol{x}_{i,mis_s} \tag{17}$$

$$g_k\left\{E\left(\boldsymbol{x}_{i,mis_k} \mid \boldsymbol{x}_{i,obs}, \boldsymbol{x}_{i,mis_{<k}}, \theta_{X_k}\right)\right\} = \boldsymbol{\gamma}'_k \boldsymbol{x}_{i,obs} + \sum_{s=1}^{k-1} \boldsymbol{\xi}'_{k_s} \boldsymbol{x}_{i,mis_s},\ k = l+1, \cdots, p \tag{18}$$

with T denoting the number of repeated measurements of individual *i*, $\boldsymbol{x}_{i,mis_{<l}} = \left(x_{i,mis_1}, \cdots, x_{i,mis_{l-1}}\right)'$, $\boldsymbol{x}_{i,mis_{<k}} = \left(x_{i,mis_1}, \cdots, x_{i,mis_{k-1}}\right)'$, $\boldsymbol{\theta}'_{Y|X} = \left(\boldsymbol{\gamma}'_y, \boldsymbol{\xi}'_y\right)$, $\boldsymbol{\theta}'_{X_l} = \left(\boldsymbol{\gamma}'_l, \boldsymbol{\xi}'_l\right)$ and $\boldsymbol{\theta}'_{X_k} = \left(\boldsymbol{\gamma}'_k, \boldsymbol{\xi}'_k\right)$.

Equation (16) represents the predictor of the linear mixed model, Equation (17) the predictor of the imputation model of x_{mis_ℓ} with link function g_ℓ from the extended exponential family, and Equation (18) represents the predictors of the covariates that have x_{mis_ℓ} as a predictive variable, with $g_k(.)$ being the corresponding link function. Clearly, looking at Equations (16)-(18) it is noted that none of them is dependent on the outcome variable as was the case in MICE. This puts SFB as a better option over MICE at the specification stage. Synonymously, just like MICE, the SFB approach assumes ignorable missing data mechanisms and correctly specified conditional distributions of the covariates.

It has been mentioned before that it is not obvious how the imputation models in the sequence should be ordered [17] and, from a theoretical point of view, different orderings may result in different joint distributions, leading to different results. Chen and Ibrahim [18] suggest to condition the categorical imputation models on the continuous covariates. In the context of MI, it has been suggested to impute variables in a sequence so that the missing pattern is close to monotone. Our convention is to order the imputation models in (15) according to the number of missing values, starting with the variable with the least missing values. It has been shown, however, that sequential specifications, as used in the Bayesian approach, are quite robust against changes in the ordering [18] and results may be unbiased even when the order of the sequence is misspecified as long as the imputation models fit the data well enough.

4. Monte Carlo Simulation Study

4.1. Relative Theory of Monte Carlo Study

A Monte Carlo simulation (also known as multiple probability simulation) is used to model the probability of different outcomes in a process that cannot easily be predicted due to the intervention of random variables. Monte Carlo studies, or Monte Carlo experiments, encompass a loop of computational algorithms that base on repeated and randomized sampling procedures to obtain numerical results. The major goal is therefore to use this randomness to solve modeling problems. Use of Monte Carlo studies allows us to understand the impact of risk and uncertainty. Universally, Monte Carlo simulation studies follow five key steps:

1) Setting up a probability distribution for important variables.

2) Building a cumulative probability distribution for each variable.

3) Establishing an interval of random numbers for each variable.

4) Generating random numbers.

5) Actually simulating a series of trials.

4.2. Design

In this section, we subject the theoretical analysis of MICE and SFB to hypothetical binary outcome panel data to support the hypothesis that SFB may be superior to MICE. To this end we focus on the logit estimator. This simulations aims at comparing bias and RMSE of the parameter estimates obtained by the unconditional logit estimator and the conditional logit estimator in the presence of imputed missing covariates.

To take care of the different possible features of the data, this comparison will be made for two sets of data, one complete (balanced) and the other incomplete (unbalanced) due to intermittent nonresponses. The latter data set is then balanced by imputing the missing observations using the two model based approaches described in Section 3. We considered a total of four data sets: one complete set, two imputed sets by MICE using the specifications (11) and (13) of how the response variable relates to missingness and one imputed set by SFB.

We specify all the panel sets fitted to the estimation of the following model:

$$y_{it} = 1\left(\boldsymbol{x}'_{it}\boldsymbol{\beta} + c_i + u_{it} \geq 0\right), i = 1, 2, \cdots, N; t = 1, 2$$

where $\boldsymbol{x}'_{it}$ is a vector of five explanatory variables drawn from uniform, binomial and normal distributions and the error term u_{it} is drawn from a normal distribution. The variables' descriptions are in **Table 1**. All the specified predictor variables have parameters, β_1 to β_5, of the model which are used to define the dependent variable y. These parameters were fixed as β_1 = 1, β_2 = −1, β_3 = 1, β_4 = 1 and β_5 = 1. The dependent variable, y is then calculated from the relation

$$y_{it} = 1\left(c_i + \beta_1 x_{it}^{(1)} + \beta_2 x_{it}^{(2)} + \beta_3 x_{it}^{(3)} + \beta_4 x_{it}^{(4)} + \beta_5 x_{it}^{(5)} + v_{it} \geq 0\right), i = 1, 2, \cdots, n; t = 1, 2$$

where v_{it} is a logistic variable given by $v_{it} = \ln\left|\frac{u_{it}}{1 + u_{it}}\right|$ with u_{it} being a standard normal random variable. The fixed effects c_i are obtained as functions of $x^{(1)}$ and T by the relation $c_i = \frac{\sqrt{T}\sum x^{(1)}}{n} + a_i$ with a_i being a standard normal random variable as well. By simulation, the probability densities of the five variables are as shown in **Figure 2** where the blue density represents the complete data and the magenta densities are from the few imputed data sets by MICE and SFB.

For in-depth comparisons, we also assess the impact of sample size on the parameter estimates obtained by these approaches herein. In order to factor in small, medium and large sample size possibilities, we mainly settled for three different values of N that were used for all sets of data fitted into the models (N = 50, N = 100 and N = 250). This was established by a subjective imposition of the expected probability of success as $Pr\left(y_{it} = 1 \mid \boldsymbol{x}_{it}, c_i\right) = 0.5$ and plausible coefficients of variation (CoV) as 0.2, 0.14 and 0.09 respectively in the relation $N \cong \frac{1}{pr\left(y\right) \times \left(CoV\right)^2}$. Further, we evaluate the impact of the proportion of

Table 1. Description of variables.

Variable	Type	
$\boldsymbol{x}^{(1)}$	continuous	N~(0, 1)
$\boldsymbol{x}^{(2)}$	continuous	U~(0, 1)
$\boldsymbol{x}^{(3)}$	continuous	N~(0.5, 0.5)
$\boldsymbol{x}^{(4)}$	discrete	B~(NT, 2, 0.65)
$\boldsymbol{x}^{(5)}$	discrete	binomial

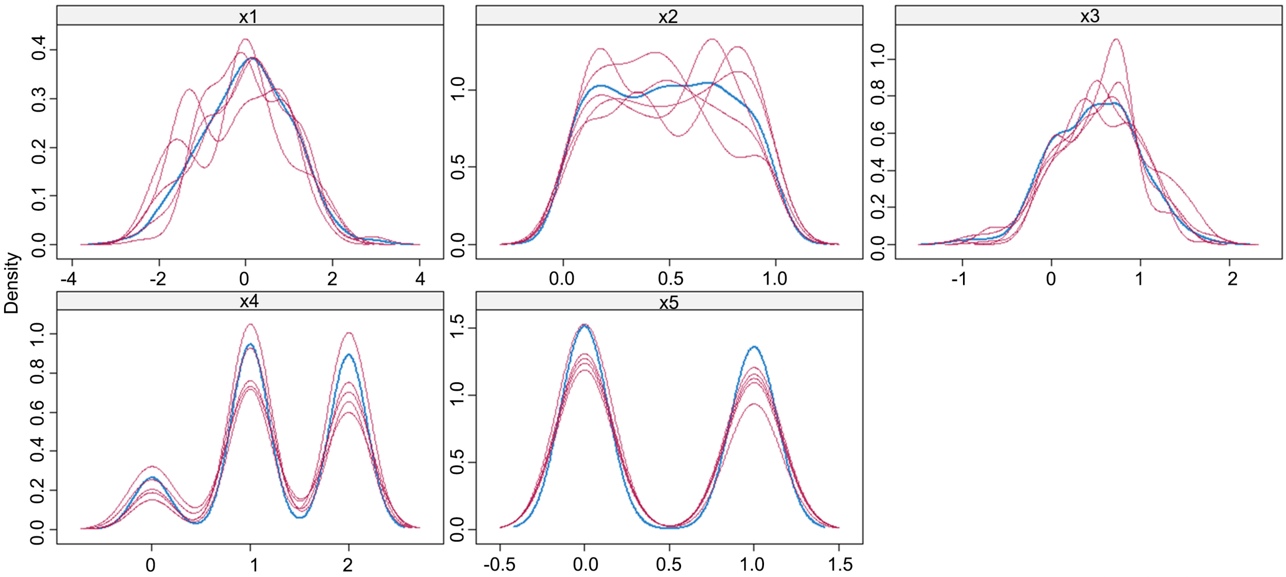

Figure 2. Covariate densities of complete data (blue) and imputed data (magenta).

missingness from 10% to 30% by randomly deleting the desired proportion of observations from the data set and imputing them back for each sample size.

Since we estimate a fixed effect model, the coefficients are truncated in order to ensure convergence. After 1000 iterations, the summarized results are given in tabulated as in Tables 2-4 where for both estimators (unconditional logit and conditional logit) considered, we report the mean bias, and the root mean squared error (RMSE) for all the parameter estimates. R-Studio has several inbuilt packages that make use of the Bayesian framework for imputation [14] [19] [20] [21]. However for longitudinal or panel data sets, the use of WinBUGS [21] or JAGS [22] provides a much more straightforward platform to easily handle SFB.

For each of the variables $\boldsymbol{x}^{(1)}$ through $\boldsymbol{x}^{(5)}$ five different samples were generated in MICE with the densities as shown. The simulated densities tend to follow the trend of the observed (complete) density in blue for each of the study variables.

4.3. Simulation Results

Table 2. Sample size $n = 50$, percentage of missingness = 10%; 30%.

			$n = 50$; Missingness Proportion = 0.1					$n = 50$; Missingness Proportion = 0.3				
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
COMPLETE	MEAN BIAS	Unconditional MLE	−0.01948	0.9563067	0.076651	−0.20602	−0.34778	−0.29096	−0.82596	0.292075	0.069342	0.596753
		Conditional MLE	−0.00628	0.9315125	0.089306	−0.19033	−0.32975	−0.27317	−0.82661	0.301872	0.082179	0.601159
	RMSE	Unconditional MLE	0.100592	0.9802211	0.663606	0.325063	0.601057	0.318272	1.288141	0.343586	0.392112	0.743191
		Conditional MLE	0.097009	0.9553502	0.65524	0.311907	0.584319	0.300971	1.271672	0.350927	0.388685	0.743756
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
MICE Scenario S1	MEAN BIAS	Unconditional MLE	0.260258	0.315379	−0.04065	0.077421	−0.32268	0.072915	−0.7043	0.437056	0.740201	0.961871
		Conditional MLE	0.269598	0.2999076	−0.02794	0.088921	−0.30704	0.084884	−0.70791	0.443984	0.743824	0.964775
	RMSE	Unconditional MLE	0.442549	0.7217758	0.448681	0.318831	0.423346	0.26687	0.717729	0.51911	0.854839	1.13552
		Conditional MLE	0.444147	0.708346	0.440965	0.317421	0.409483	0.266817	0.721077	0.522946	0.854584	1.089705

Continued

			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
MICE Scenario S2	MEAN BIAS	Unconditional MLE	0.172859	0.1821702	0.485272	0.046385	−0.22299	−0.21347	−0.84115	0.54682	0.455508	1.017005
		Conditional MLE	0.182857	0.168391	0.491847	0.05799	−0.20836	−0.19687	−0.8445	0.553073	0.463835	1.017086
	RMSE	Unconditional MLE	0.244211	0.4820251	0.578371	0.222941	0.559952	0.568807	0.923257	0.5683	0.888667	1.158646
		Conditional MLE	0.24961	0.4719167	0.581505	0.222594	0.548313	0.554065	0.923173	0.573325	0.882903	1.155659
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
SFB	MEAN BIAS	Unconditional MLE	0.026126	0.555159	−0.09357	−0.18414	−0.25881	−0.41306	−1.46703	1.312149	0.470652	1.607761
		Conditional MLE	0.038599	0.5361739	−0.07908	−0.16905	−0.24241	−0.39283	−1.45872	1.305119	0.47851	1.597364
	RMSE	Unconditional MLE	0.29805	0.6776598	0.736881	0.349123	0.486108	0.780638	1.741855	1.710453	0.761253	2.172038
		Conditional MLE	0.294392	0.6596904	0.723509	0.33722	0.47286	0.759586	1.72698	1.695001	0.75959	2.150167

Table 3. Sample size $n = 100$, percentage of missingness = 10%; 30%.

			$n = 100$; Missingness Proportion = 0.1					$n = 100$; Missingness Proportion = 0.3				
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
COMPLETE	MEAN BIAS	Unconditional MLE	−0.026545	0.24095968	−0.514189	0.0331084	0.0151653	−0.121217	−0.042608	−0.017225	−0.10057	−0.007494
		Conditional MLE	−0.020124	0.2331836	−0.504822	0.0392261	0.0213423	−0.113572	−0.049014	−0.010339	−0.093025	−0.000911
	RMSE	Unconditional MLE	0.1903491	0.6246426	0.7734177	0.2703348	0.2503745	0.290803	0.7891205	0.3798074	0.3904801	0.4347148
		Conditional MLE	0.1880342	0.6183119	0.7646331	0.2694574	0.2492443	0.2857267	0.7839511	0.3763995	0.3858615	0.4316825
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
MICE Scenario S1	MEAN BIAS	Unconditional MLE	0.2516552	−0.1866761	0.0728258	0.198762	0.1076809	0.3508914	−0.344938	0.3547572	0.6271577	0.0110344
		Conditional MLE	0.2560313	−0.1912559	0.0781992	0.2034946	0.1129867	0.3547193	−0.348851	0.3585348	0.6293101	−0.005292
	RMSE	Unconditional MLE	0.2683889	0.5899517	0.4640846	0.241681	0.4630161	0.4446144	0.6090167	0.4263495	0.6553556	0.4012151
		Conditional MLE	0.2722989	0.5882402	0.4620065	0.2450236	0.4615298	0.4465738	0.608885	0.4287747	0.6571148	0.3986212
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
MICE Scenario S2	MEAN BIAS	Unconditional MLE	0.2494054	−0.2407727	0.1914832	0.1928121	0.0773211	0.167047	−0.600057	0.3503351	0.6186948	0.3321584
		Conditional MLE	0.2538035	−0.2450332	0.1961337	0.1976235	0.4885013	0.1719428	−0.602205	0.3539004	0.6209644	0.3360003
	RMSE	Unconditional MLE	0.2671482	0.7220803	0.4666194	0.2830729	0.0828817	0.295708	0.6825274	0.4383071	0.7377291	0.5565751
		Conditional MLE	0.2711105	0.7198476	0.4660972	0.2853809	0.4864855	0.2971254	0.6835531	0.4403339	0.7383632	0.5568191
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
SFB	MEAN BIAS	Unconditional MLE	0.2067982	−0.2604658	−0.02578	0.0550596	−0.022355	0.0145467	−0.628817	−0.111558	0.2182149	0.1355094
		Conditional MLE	0.2114806	−0.2646177	−0.019777	0.0607217	−0.016193	0.0204732	0.6309121	−0.105140	0.2230226	0.1407483
	RMSE	Unconditional MLE	0.2349832	0.7415248	0.387904	0.1675892	0.4943286	0.3062883	0.958299	0.3021251	0.4577116	0.595593
		Conditional MLE	0.2388453	0.7391579	0.3850371	0.168586	0.4909875	0.3044948	0.9566361	0.2984249	0.4577738	0.5935283

Table 4. Sample size $n = 250$, percentage of missingness = 10%; 30%.

			$n = 250$; Missingness Proportion = 0.1					$n = 250$; Missingness Proportion = 0.3				
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
COMPLETE	MEAN BIAS	Unconditional MLE	−0.033774	−0.338095	0.1962007	0.0212008	0.3798605	−0.106748	−0.109786	0.0455062	0.0197634	0.1412802
		Conditional MLE	−0.031226	−0.339734	0.1981616	0.0236182	0.3813705	−0.104002	−0.111927	0.0478644	0.0221668	0.1433672
	RMSE	Unconditional MLE	0.1686476	0.6090461	0.3114705	0.2138064	0.4685767	0.1755561	0.4067052	0.13352	0.1034421	0.2222687
		Conditional MLE	0.1677631	0.6088706	0.3122318	0.2135	0.4694655	0.1735325	0.4063504	0.1340535	0.1036447	0.2233176
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
MICE Scenario S1	MEAN BIAS	Unconditional MLE	0.1028953	−0.505525	0.4068535	0.0889948	0.3423036	0.2704451	−0.494219	0.458424	0.3653706	0.2217551
		Conditional MLE	0.1050316	−0.506706	0.4082471	0.0911703	0.3438477	0.272087	−0.495332	0.4596349	0.3667752	0.2234763
	RMSE	Unconditional MLE	0.1958269	0.6939537	0.4962603	0.1979744	0.42408	0.2815245	0.5410733	0.4780227	0.3752274	0.2853882
		Conditional MLE	0.1966131	0.6940114	0.4970059	0.1985627	0.4250075	0.2830576	0.5419056	0.4790938	0.3765496	0.286484
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
MICE Scenario S2	MEAN BIAS	Unconditional MLE	0.1291901	−0.492033	0.4363771	0.1278418	0.302608	0.22545	−0.307334	0.5329577	0.3443494	0.3621379
		Conditional MLE	0.1312643	−0.493249	0.4376999	0.1299188	0.3042428	0.2271924	−0.308922	0.5339932	0.3458157	0.3635396
	RMSE	Unconditional MLE	0.2324665	0.6988569	0.4736992	0.2033485	0.4062817	0.2693463	0.437152	0.5613183	0.3627137	0.3620414
		Conditional MLE	0.2332313	0.6988575	0.4747516	0.2043669	0.407102	0.2706187	0.4377978	0.5621783	0.3640218	0.3733636
			β_1	β_2	β_3	β_4	β_5	β_1	β_2	β_3	β_4	β_5
SFB	MEAN BIAS	Unconditional MLE	0.0109037	−0.380619	0.2890443	0.0321049	0.3811126	−0.018110	−0.249828	0.1437004	0.032043	0.2542767
		Conditional MLE	0.0132911	−0.382121	0.290744	0.0344481	0.3825821	−0.015741	−0.251541	0.1457413	0.0342991	0.2560121
	RMSE	Unconditional MLE	0.1713281	0.6413936	0.4161804	0.2355166	0.464298	0.1907921	0.8706938	0.1615806	0.1253551	0.3311532
		Conditional MLE	0.1710827	0.6412323	0.4168407	0.235269	0.4651764	0.1901571	0.869403	0.16331	0.1256505	0.3321965

5. Discussion, Conclusions and Recommendation

Through the variation of the simulated sample sizes, the study confirms the asymptomatic properties of parameter bias. Undeniably, all the reported measures (mean bias and the root mean square errors) are observed to have an inverse relationship with increasing sample size across all the models considered. We also have established as much as MICE performs well in most standard situations, SFB yields relatively stable estimates that are robust to the proportion of missingness in a binary response panel data frame. In addition, MICE does not

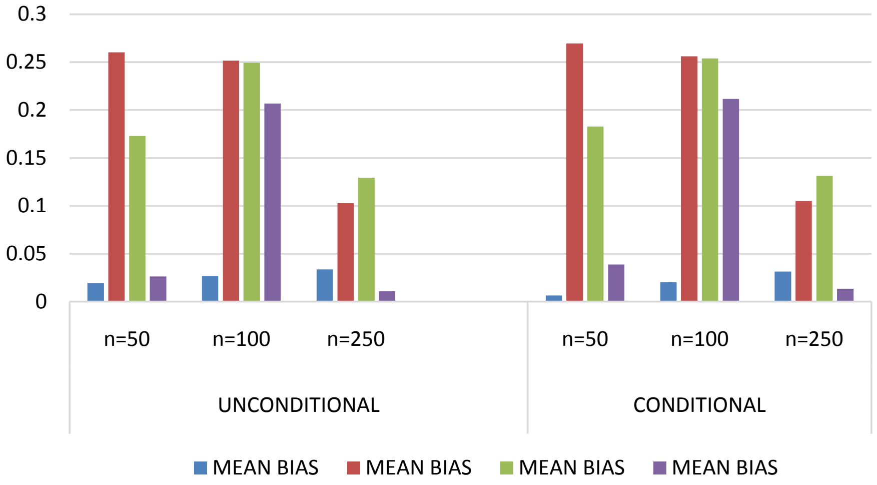

Figure 3. Average bias for parameter estimates.

respond consistently with categorical or discrete data but yields consistent parameter estimates for the continuous regressor. This is also observed for SFB but improves asymptomatically. Uniquely, the magnitude of the median bias produces estimates with increased biases for the conditional logit estimator compared to the unconditional logit estimator when all three imputation techniques are performed more so when the sample size is large [23]. **Figure 3** shows the superiority of SFB over MICE for both scenarios considered when handling the logistic panel data model.

Since the SFB approach uses a sequence of univariate conditional distributions to specify the joint pdf, it is advantageous because the analysis model is part of the conditional distributions and the parameters of interest are estimated simultaneously with the imputation of the missing values hence no further pooling is done as is the case in MICE. This means that specification of the univariate conditional imputation model permits a much straightforward and flexible imputation.

In summary, this paper has discussed brief estimation methods and procedures for estimating nonlinear (binary choice logit) panel data regression models with missing covariates.

The key objective of this study was singled out to be an assessment of the performances of MICE and Bayesian imputation to effect of non-responses (missingness) in the parameter estimates for logit panel data models. Although studies show that MICE performs relatively well in many standard situations, this study has demonstrated the advantage of working with full likelihood in more complex specifications of the response variable. In addition, situations where imputation with MICE does not take into account the actual structure of the measurement model may yield poorly imputed values which affect the consistency of the analysis.

A key importance of deriving the estimators is to increase the theoretical understanding of the estimators and also reduce the computational complexity while estimating logit panel models. As observed from the Monte Carlo results, unbalancedness in a data set biases the parameter estimates and the different imputation techniques employed in this study respond differently to the bias and efficiency of the estimates.

As a recommendation, further developments can be done on this study by considering other imputation techniques in MICE where the response variable enters the imputation model in other different specifications.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Donders, A.R.T., van der Heijden, G.J., Stijnen, T. and Moons, K.G.M. (2006) Review: A Gentle Introduction to Imputation of Missing Values. *Journal of Clinical Epidemiology*, **59**, 1087-1091. https://doi.org/10.1016/j.jclinepi.2006.01.014

[2] Janssen, K.J.M., Donders, A.R.T., Harrell Jr., F.E., Vergouwe, Y., Chen, Q., Grobbee, D.E. and Moons, K.G. (2010) Missing Covariate Data in Medical Research: To Impute Is Better than to Ignore. *Journal of Clinical Epidemiology*, **63**, 721-727. https://doi.org/10.1016/j.jclinepi.2009.12.008

[3] Knol, M.J., Janssen, K.J., Donders, A.R.T., Egberts, A.C., Heerdink, E.R., Grobbee, D.E., Moons, K.G. and Geerlings, M.I. (2010) Unpredictable Bias When Using the Missing Indicator Method or Complete Case Analysis for Missing Confounder Values: An Empirical Example. *Journal of Clinical Epidemiology*, **63**, 728-736. https://doi.org/10.1016/j.jclinepi.2009.08.028

[4] van Buuren, S. (2012) Flexible Imputation of Missing Data. Chapman & Hall/CRC Interdisciplinary Statistics, CRC Press Taylor & Francis Group, Boca Raton.

[5] Rubin, D. (1987) Multiple Imputation for Nonresponse in Surveys. John Wiley & Sons, Inc., Hoboken. https://doi.org/10.1002/9780470316696

[6] Moons, K.G.M., Donders, R.A., Stijnen, T. and Harrell Jr., F.E. (2006) Using the Outcome for Imputation of Missing Predictor Values Was Preferred. *Journal of Clinical Epidemiology*, **59**, 1092-1101. https://doi.org/10.1016/j.jclinepi.2006.01.009

[7] Ibrahim, J.G., Chen, M.-H. and Lipsitz, S.R. (2002) Bayesian Methods for Generalized Linear Models with Covariates Missing at Random. *Canadian Journal of Statistics*, **30**, 55-78. https://doi.org/10.2307/3315865

[8] Stubbendick, A.L. and Ibrahim, J.G. (2003) Maximum Likelihood Methods for Nonignorable Missing Responses and Covariates in Random Effects Models. *Biometrics*, **59**, 1140-1150. https://doi.org/10.1111/j.0006-341X.2003.00131.x

[9] Chen, B., Grace, Y.Y. and Cook, R.J. (2010) Weighted Generalized Estimating Functions for Longitudinal Response and Covariate Data That Are Missing at Random. *Journal of the American Statistical Association*, **105**, 336-353. https://doi.org/10.1198/jasa.2010.tm08551

[10] Chen, B. and Zhou, X.-H. (2011) Doubly Robust Estimates for Binary Longitudinal Data Analysis with Missing Response and Missing Covariates. *Biometrics*, **67**, 830-842. https://doi.org/10.1111/j.1541-0420.2010.01541.x

[11] Chamberlain, G. (1984) Panel Data. In: Chamberlai, G., Ed., *Handbook of Econometrics*, Vol. 2, Elsevier, Amsterdam.

[12] Seaman, S., Galati, J., Jackson, D. and Carlin, J. (2013) What Is Meant by "Missing at Random"? *Statistical Science*, **28**, 257-268. https://doi.org/10.1214/13-STS415

[13] Rubin, D.B. (2004) Multiple Imputation for Nonresponse in Surveys. John Wiley & Sons Inc., New York.

[14] van Buuren, S. and Groothuis-Oudshoorn, K. (2011) MICE: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software*, **45**, 1-67. https://doi.org/10.18637/jss.v045.i03

[15] Carpenter, J.R. and Kenward, M.G. (2013) Multiple Imputation and Its Application. John Wiley & Sons, Ltd., Chichester. https://doi.org/10.1002/9781119942283

[16] Little, R. and Rubin, D. (1987) Statistical Analysis with Missing Data. John Wiley & Sons, Inc., Hoboken.

[17] Bartlett, J.W., Seaman, S.R., White, I.R. and Carpenter, J.R. (2015) Multiple Imputation of Covariates by Fully Conditional Specification: Accommodating the Substantive Model. *Statistical Methods in Medical Research*, **24**, 462-487. https://doi.org/10.1177/0962280214521348

[18] Chen, M.-H. and Ibrahim, J.G. (2001) Maximum Likelihood methods for Cure Rate Models with Missing Covariates. *Biometrics*, **57**, 43-52. https://doi.org/10.1111/j.0006-341X.2001.00043.x

[19] Zhao, J.H. and Schafer, J.L. (2015) Pan: Multiple Imputation for Multivariate Panel or Clustered Data. https://cran.r-project.org/web/packages/pan/pan.pdf

[20] Bartlett, J.W. and Morris, T.P. (2015) smcfcs: Multiple Imputation of Covariates by Substantive-Model Compatible Fully Conditional Specification. *The Stata Journal: Promoting communications on statistics and Stata*, **15**, 437-456. https://doi.org/10.1177/1536867X1501500206

[21] Lunn, D.J., Thomas, A., Best, N. and Spiegelhalter, D. (2000) WinBUGS—A Bayesian Modelling Framework: Concepts, Structure, and Extensibility. *Statistics and Computing*, **10**, 325-337. https://doi.org/10.1023/A:1008929526011

[22] Plummer, M. (2003) JAGS: A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling. *Proceedings of the* 3*rd International Workshop on Distributed Statistical Computing* (*DSC* 2003), Vienna, 20-22 March 2003.

[23] Opeyo, P.O., Olubusoye, O.E. and Odongo, L.O. (2014) Conditional Maximum Likelihood Estimation for Logit Panel Models with Non-Responses. *International Journal of Science and Research*, **3**, 2242-2254.

Appendix

R CODES FOR MONTE CARLO SIMULATION

n = 50 with 10% missingness

```
set.seed(12345)          # Use this to make the randomly generated data the same each time you run the simulation#
N.iter = 1000
j = 1:5
A <- array(0, dim=c(N.iter,5))
B <- array(0, dim=c(N.iter,5))
C <- array(0, dim=c(N.iter,5))
D <- array(0, dim=c(N.iter,5))
E <- array(0, dim=c(N.iter,5))
F <- array(0, dim=c(N.iter,5))
G <- array(0, dim=c(N.iter,5))
H <- array(0, dim=c(N.iter,5))
for(i in 1:N.iter){
n=50                                        # vary n
t=2           # vary t and change time in pData
nt=n*t
b1=1
b2=-1
b3=1
b4=1
b5=1
ai=rnorm(n,0,1)
x1=rnorm(nt,0,1)
x2 <- runif(nt,0,1)
x3 <- rnorm(nt,0.5,0.5)
x4 <- rbinom(nt,2,0.65)                      # nt different such numbers
hit=rnorm(nt,0,1)
uit=rnorm(nt,0,1)
vit=log(abs(uit/(1+uit)))
x5 <- ifelse(x1+hit>0, 1, 0)
alphai <- sqrt(t)*sum(x1)/n+ai #alphai simulation
ci <- kronecker(alphai ,matrix(1,t,1))#kronecker of alphai
wit=ci+b1*x1+b2*x2+b3*x3+b4*x4+b5*x5
zit= wit+vit
y <- ifelse(zit>0, 1, 0)
pData <- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,zit,ci,x1,x2,x3,x4,x5)
# Randomly Insert A Certain Proportion Of NAs Into A Dataframe #
pData2<- cbind(x1,x2,x3,x4,x5)
##################################
NAins <- NAinsert <- function(df, prop = .1){
  n <- nrow(df)
```

```
 m <- ncol(df)
 num.to.na <- ceiling(prop*n*m)
 id <- sample(0:(m*n-1), num.to.na, replace = FALSE)
 rows <- id %/% m + 1
 cols <- id %% m + 1
 sapply(seq(num.to.na), function(x){
 df[rows[x], cols[x]] <<- NA
 }
 )
 return(df)
}
##############
# TRY IT OUT #
##############
XX<-NAins(pData2, .1)
XX
pData3<- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,XX)
pData3
#Scenario 1 S1 when f(yi)=0
yS1<-y*0
XXX<-data.frame(yS1,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply (XXX,2,pMiss)
apply(XXX,1,pMiss)
#Performing MICE in XXX
library(mice)
md.pattern(XXX)
library(VIM)
aggr_plot <- aggr(XXX, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX), cex.axis=.7,
gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData <- mice(XXX,m=5,maxit=50,meth='pmm',seed=500)
summary(tempData)
#to see the new imputed covariate values, say X1
tempData$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData)
#getting back the completed data and binding with original y
completedData <- complete(tempData,1)
completedDataS1<- subset(completedData, select=c(x1,x2,x3,x4,x5))
XXXS1<-data.frame(y,completedDataS1)
XXXS1
#parameter estimation, unconditional logit and conditional logit for Scenario S1
```

```
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#Scenario 3 S3 when f(yi)=E(yit)=p
S3=rep(mean(y),times=nt)
yS3<-data.frame(S3)
XXX3<-data.frame(yS3,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX3,2,pMiss)
apply(XXX3,1,pMiss)
#Performing MICE in XXX3
library(mice)
md.pattern(XXX3)
library(VIM)
aggr_plot <- aggr(XXX3, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX3), cex.axis=.7,
gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData3 <- mice(XXX3,m=5,maxit=50,meth='norm',seed=500)
summary(tempData3)
#to see the new imputed covariate values, say X1
tempData3$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData3)
#getting back the completed data and binding with original y
completedData3 <- complete(tempData3,1)
completedDataS3<- subset(completedData3, select=c(x1,x2,x3,x4,x5))
XXXS3<-data.frame(y,completedDataS3)
XXXS3
# Impute missing values using Bayesian imputation
imputed_data <- mice(XX[, c("x1", "x2", "x3", "x4", "x5")], method = "norm.predict")    # Use correct variable names
and select only the predictor variables
# Extract the completed dataset from the imputed_data object
XXX4 <- complete(imputed_data)
BayesData <- data.frame(y,XXX4)
#parameter estimation, unconditional logit and conditional logit for Complete Original Data
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#parameter estimation, unconditional logit and conditional logit for Scenario S3
glm.out3 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS3)
```

```
clogit3=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS3)
#parameter estimation, unconditional logit and conditional logit for BayesData
glm.outBayes = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=BayesData)
clogitBayes=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = BayesData)
#Calling out coefficients
A[i,] <- glm.out1$coef[j+1]
B[i,] <- clogit1$coef[j]
C[i,] <- glm.out2$coef[j+1]
D[i,] <- clogit2$coef[j]
E[i,] <- glm.out3$coef[j+1]
F[i,] <- clogit3$coef[j]
G[i,] <- glm.outBayes$coef[j+1]
H[i,] <- clogitBayes$coef[j] }
#mean bias#
mean.biasA<- c(b1-mean(A[,1]), b2-mean(A[,2]),b3-mean(A[,3]),b4-mean(A[,4]),b5-mean(A[,5]))
mean.biasB<- c(b1-mean(B[,1]), b2-mean(B[,2]),b3-mean(B[,3]),b4-mean(B[,4]),b5-mean(B[,5]))
mean.biasC<- c(b1-mean(C[,1]), b2-mean(C[,2]),b3-mean(C[,3]),b4-mean(C[,4]),b5-mean(C[,5]))
mean.biasD<- c(b1-mean(D[,1]), b2-mean(D[,2]),b3-mean(D[,3]),b4-mean(D[,4]),b5-mean(D[,5]))
mean.biasE<- c(b1-mean(E[,1]), b2-mean(E[,2]),b3-mean(E[,3]),b4-mean(E[,4]),b5-mean(E[,5]))
mean.biasF<- c(b1-mean(F[,1]), b2-mean(F[,2]),b3-mean(F[,3]),b4-mean(F[,4]),b5-mean(F[,5]))
mean.biasG<- c(b1-mean(G[,1]), b2-mean(G[,2]),b3-mean(G[,3]),b4-mean(G[,4]),b5-mean(G[,5]))
mean.biasH<- c(b1-mean(H[,1]), b2-mean(H[,2]),b3-mean(H[,3]),b4-mean(H[,4]),b5-mean(H[,5]))
#residual errors#
rmseA<-c(sqrt(mean((b1-A[,1])^2)), sqrt(mean((b2-A[,2])^2)), sqrt(mean((b3-A[,3])^2)), sqrt(mean((b4-A[,4])^2)), sqrt(mean((b5-A[,5])^2)))
rmseB<-c(sqrt(mean((b1-B[,1])^2)), sqrt(mean((b2-B[,2])^2)), sqrt(mean((b3-B[,3])^2)), sqrt(mean((b4-B[,4])^2)), sqrt(mean((b5-B[,5])^2)))
rmseC<-c(sqrt(mean((b1-C[,1])^2)), sqrt(mean((b2-C[,2])^2)), sqrt(mean((b3-C[,3])^2)), sqrt(mean((b4-C[,4])^2)), sqrt(mean((b5-C[,5])^2)))
rmseD<-c(sqrt(mean((b1-D[,1])^2)), sqrt(mean((b2-D[,2])^2)), sqrt(mean((b3-D[,3])^2)), sqrt(mean((b4-D[,4])^2)), sqrt(mean((b5-D[,5])^2)))
rmseE<-c(sqrt(mean((b1-E[,1])^2)), sqrt(mean((b2-E[,2])^2)), sqrt(mean((b3-E[,3])^2)), sqrt(mean((b4-E[,4])^2)), sqrt(mean((b5-E[,5])^2)))
rmseF<-c(sqrt(mean((b1-F[,1])^2)), sqrt(mean((b2-F[,2])^2)), sqrt(mean((b3-F[,3])^2)), sqrt(mean((b4-F[,4])^2)), sqrt(mean((b5-F[,5])^2)))
rmseG<-c(sqrt(mean((b1-G[,1])^2)), sqrt(mean((b2-G[,2])^2)), sqrt(mean((b3-G[,3])^2)), sqrt(mean((b4-G[,4])^2)), sqrt(mean((b5-G[,5])^2)))
rmseH<-c(sqrt(mean((b1-H[,1])^2)), sqrt(mean((b2-H[,2])^2)), sqrt(mean((b3-H[,3])^2)), sqrt(mean((b4-H[,4])^2)), sqrt(mean((b5-H[,5])^2)))
mean.biasA
rmseA
mean.biasB
rmseB
```

```
mean.biasC
rmseC
mean.biasD
rmseD
mean.biasE
rmseE
mean.biasF
rmseF
mean.biasG
rmseG
mean.biasH
rmseH
```

n= 50 with 30% missingness

```
set.seed(123456)          # Use this to make the randomly generated data the same each time you run the simulation#
N.iter=1000
j= 1:5
A <- array(0, dim=c(N.iter,5))
B <- array(0, dim=c(N.iter,5))
C <- array(0, dim=c(N.iter,5))
D <- array(0, dim=c(N.iter,5))
E <- array(0, dim=c(N.iter,5))
F <- array(0, dim=c(N.iter,5))
G <- array(0, dim=c(N.iter,5))
H <- array(0, dim=c(N.iter,5))
for(i in 1:N.iter){
n=50                                        # vary n
t=2          # vary t and change time in pData
nt=n*t
b1=1
b2=-1
b3=1
b4=1
b5=1
ai=rnorm(n,0,1)
x1=rnorm(nt,0,1)
x2 <- runif(nt,0,1)
x3 <- rnorm(nt,0.5,0.5)
x4 <- rbinom(nt,2,0.65)                  # nt different such numbers
hit=rnorm(nt,0,1)
uit=rnorm(nt,0,1)
vit=log(abs(uit/(1+uit)))
x5 <- ifelse(x1+hit>0, 1, 0)
alphai <- sqrt(t)*sum(x1)/n+ai #alphai simulation
```

```
ci <- kronecker(alphai ,matrix(1,t,1))#kronecker of alphai
wit=ci+b1*x1+b2*x2+b3*x3+b4*x4+b5*x5
zit= wit+vit
y <- ifelse(zit>0, 1, 0)
pData <- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,zit,ci,x1,x2,x3,x4,x5)
# Randomly Insert A Certain Proportion Of NAs Into A Dataframe #
pData2<- cbind(x1,x2,x3,x4,x5)
##################################
NAins <- NAinsert <- function(df, prop = .3){
  n <- nrow(df)
  m <- ncol(df)
  num.to.na <- ceiling(prop*n*m)
  id <- sample(0:(m*n-1), num.to.na, replace = FALSE)
  rows <- id %/% m + 1
  cols <- id %% m + 1
  sapply(seq(num.to.na), function(x){
  df[rows[x], cols[x]] <<- NA
  }
  )
  return(df)
}
##############
# TRY IT OUT #
##############
XX<-NAins(pData2, .3)
XX
pData3<- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,XX)
pData3
#Scenario 1 S1 when f(yi)=0
yS1<-y*0
XXX<-data.frame(yS1,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX,2,pMiss)
apply(XXX,1,pMiss)
#Performing MICE in XXX
library(mice)
md.pattern(XXX)
library(VIM)
aggr_plot <- aggr(XXX, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX), cex.axis=.7,
gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData <- mice(XXX,m=5,maxit=50,meth='pmm',seed=500)
summary(tempData)
```

```
#to see the new imputed covariate values, say X1
tempData$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData)
#getting back the completed data and binding with original y
completedData <- complete(tempData,1)
completedDataS1<- subset(completedData, select=c(x1,x2,x3,x4,x5))
XXXS1<-data.frame(y,completedDataS1)
XXXS1
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#Scenario 3 S3 when f(yi)=E(yit)=p
S3=rep(mean(y),times=nt)
yS3<-data.frame(S3)
XXX3<-data.frame(yS3,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX3,2,pMiss)
apply(XXX3,1,pMiss)
#Performing MICE in XXX3
library(mice)
md.pattern(XXX3)
library(VIM)
aggr_plot <- aggr(XXX3, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX3), cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData3 <- mice(XXX3,m=5,maxit=50,meth='norm',seed=500)
summary(tempData3)
#to see the new imputed covariate values, say X1
tempData3$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData3)
#getting back the completed data and binding with original y
completedData3 <- complete(tempData3,1)
completedDataS3<- subset(completedData3, select=c(x1,x2,x3,x4,x5))
XXXS3<-data.frame(y,completedDataS3)
XXXS3
# Impute missing values using Bayesian imputation
imputed_data <- mice(XX[, c("x1", "x2", "x3", "x4", "x5")], method = "norm.predict")    # Use correct variable names and select only the predictor variables
# Extract the completed dataset from the imputed_data object
```

```
XXX4 <- complete(imputed_data)
BayesData <- data.frame(y,XXX4)
#parameter estimation, unconditional logit and conditional logit for Complete Original Data
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#parameter estimation, unconditional logit and conditional logit for Scenario S3
glm.out3 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS3)
clogit3=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS3)
#parameter estimation, unconditional logit and conditional logit for BayesData
glm.outBayes = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=BayesData)
clogitBayes=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = BayesData)
#Calling out coefficients
A[i,] <- glm.out1$coef[j+1]
B[i,] <- clogit1$coef[j]
C[i,] <- glm.out2$coef[j+1]
D[i,] <- clogit2$coef[j]
E[i,] <- glm.out3$coef[j+1]
F[i,] <- clogit3$coef[j]
G[i,] <- glm.outBayes$coef[j+1]
H[i,] <- clogitBayes$coef[j] }
#mean bias#
mean.biasA<- c(b1-mean(A[,1]), b2-mean(A[,2]),b3-mean(A[,3]),b4-mean(A[,4]),b5-mean(A[,5]))
mean.biasB<- c(b1-mean(B[,1]), b2-mean(B[,2]),b3-mean(B[,3]),b4-mean(B[,4]),b5-mean(B[,5]))
mean.biasC<- c(b1-mean(C[,1]), b2-mean(C[,2]),b3-mean(C[,3]),b4-mean(C[,4]),b5-mean(C[,5]))
mean.biasD<- c(b1-mean(D[,1]), b2-mean(D[,2]),b3-mean(D[,3]),b4-mean(D[,4]),b5-mean(D[,5]))
mean.biasE<- c(b1-mean(E[,1]), b2-mean(E[,2]),b3-mean(E[,3]),b4-mean(E[,4]),b5-mean(E[,5]))
mean.biasF<- c(b1-mean(F[,1]), b2-mean(F[,2]),b3-mean(F[,3]),b4-mean(F[,4]),b5-mean(F[,5]))
mean.biasG<- c(b1-mean(G[,1]), b2-mean(G[,2]),b3-mean(G[,3]),b4-mean(G[,4]),b5-mean(G[,5]))
mean.biasH<- c(b1-mean(H[,1]), b2-mean(H[,2]),b3-mean(H[,3]),b4-mean(H[,4]),b5-mean(H[,5]))
#residual errors#
rmseA<-c(sqrt(mean((b1-A[,1])^2)), sqrt(mean((b2-A[,2])^2)), sqrt(mean((b3-A[,3])^2)), sqrt(mean((b4-A[,4])^2)),
sqrt(mean((b5-A[,5])^2)))
rmseB<-c(sqrt(mean((b1-B[,1])^2)), sqrt(mean((b2-B[,2])^2)), sqrt(mean((b3-B[,3])^2)), sqrt(mean((b4-B[,4])^2)),
sqrt(mean((b5-B[,5])^2)))
rmseC<-c(sqrt(mean((b1-C[,1])^2)), sqrt(mean((b2-C[,2])^2)), sqrt(mean((b3-C[,3])^2)), sqrt(mean((b4-C[,4])^2)),
sqrt(mean((b5-C[,5])^2)))
rmseD<-c(sqrt(mean((b1-D[,1])^2)), sqrt(mean((b2-D[,2])^2)), sqrt(mean((b3-D[,3])^2)), sqrt(mean((b4-D[,4])^2)),
sqrt(mean((b5-D[,5])^2)))
rmseE<-c(sqrt(mean((b1-E[,1])^2)), sqrt(mean((b2-E[,2])^2)), sqrt(mean((b3-E[,3])^2)), sqrt(mean((b4-E[,4])^2)),
sqrt(mean((b5-E[,5])^2)))
```

```
rmseF<-c(sqrt(mean((b1-F[,1])^2)), sqrt(mean((b2-F[,2])^2)), sqrt(mean((b3-F[,3])^2)), sqrt(mean((b4-F[,4])^2)), sqrt(mean((b5-F[,5])^2)))
rmseG<-c(sqrt(mean((b1-G[,1])^2)), sqrt(mean((b2-G[,2])^2)), sqrt(mean((b3-G[,3])^2)), sqrt(mean((b4-G[,4])^2)), sqrt(mean((b5-G[,5])^2)))
rmseH<-c(sqrt(mean((b1-H[,1])^2)), sqrt(mean((b2-H[,2])^2)), sqrt(mean((b3-H[,3])^2)), sqrt(mean((b4-H[,4])^2)), sqrt(mean((b5-H[,5])^2)))
mean.biasA
rmseA
mean.biasB
rmseB
mean.biasC
rmseC
mean.biasD
rmseD
mean.biasE
rmseE
mean.biasF
rmseF
mean.biasG
rmseG
mean.biasH
rmseH
```

n= 100 with 10% missingness

```
set.seed(1234567)        # Use this to make the randomly generated data the same each time you run the simulation#
N.iter=1000
j= 1:5
A <- array(0, dim=c(N.iter,5))
B <- array(0, dim=c(N.iter,5))
C <- array(0, dim=c(N.iter,5))
D <- array(0, dim=c(N.iter,5))
E <- array(0, dim=c(N.iter,5))
F <- array(0, dim=c(N.iter,5))
G <- array(0, dim=c(N.iter,5))
H <- array(0, dim=c(N.iter,5))
for(i in 1:N.iter){
n=100                                  # vary n
t=2          # vary t and change time in pData
nt=n*t
b1=1
b2=-1
b3=1
b4=1
b5=1
```

```
ai=rnorm(n,0,1)
x1=rnorm(nt,0,1)
x2 <- runif(nt,0,1)
x3 <- rnorm(nt,0.5,0.5)
x4 <- rbinom(nt,2,0.65)                        # nt different such numbers
hit=rnorm(nt,0,1)
uit=rnorm(nt,0,1)
vit=log(abs(uit/(1+uit)))
x5 <- ifelse(x1+hit>0, 1, 0)
alphai <- sqrt(t)*sum(x1)/n+ai #alphai simulation
ci <- kronecker(alphai ,matrix(1,t,1))#kronecker of alphai
wit=ci+b1*x1+b2*x2+b3*x3+b4*x4+b5*x5
zit= wit+vit
y <- ifelse(zit>0, 1, 0)
pData <- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,zit,ci,x1,x2,x3,x4,x5)
# Randomly Insert A Certain Proportion Of NAs Into A Dataframe #
pData2<- cbind(x1,x2,x3,x4,x5)
##################################
NAins <- NAinsert <- function(df, prop = .1){
  n <- nrow(df)
  m <- ncol(df)
  num.to.na <- ceiling(prop*n*m)
  id <- sample(0:(m*n-1), num.to.na, replace = FALSE)
  rows <- id %/% m + 1
  cols <- id %% m + 1
  sapply(seq(num.to.na), function(x){
  df[rows[x], cols[x]] <<- NA
  }
  )
  return(df)
}
##############
# TRY IT OUT #
##############
XX<-NAins(pData2, .1)
XX
pData3<- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,XX)
pData3
#Scenario 1 S1 when f(yi)=0
yS1<-y*0
XXX<-data.frame(yS1,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
```

```
apply(XXX,2,pMiss)
apply(XXX,1,pMiss)
#Performing MICE in XXX
library(mice)
md.pattern(XXX)
library(VIM)
aggr_plot <- aggr(XXX, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX), cex.axis=.7,
gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData <- mice(XXX,m=5,maxit=50,meth='pmm',seed=500)
summary(tempData)
#to see the new imputed covariate values, say X1
tempData$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData)
#getting back the completed data and binding with original y
completedData <- complete(tempData,1)
completedDataS1<- subset(completedData, select=c(x1,x2,x3,x4,x5))
XXXS1<-data.frame(y,completedDataS1)
XXXS1
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#Scenario 3 S3 when f(yi)=E(yit)=p
S3=rep(mean(y),times=nt)
yS3<-data.frame(S3)
XXX3<-data.frame(yS3,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX3,2,pMiss)
apply(XXX3,1,pMiss)
#Performing MICE in XXX3
library(mice)
md.pattern(XXX3)
library(VIM)
aggr_plot <- aggr(XXX3, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX3), cex.axis=.7,
gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData3 <- mice(XXX3,m=5,maxit=50,meth='norm',seed=500)
summary(tempData3)
#to see the new imputed covariate values, say X1
tempData3$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
```

```
densityplot(tempData3)
#getting back the completed data and binding with original y
completedData3 <- complete(tempData3,1)
completedDataS3<- subset(completedData3, select=c(x1,x2,x3,x4,x5))
XXXS3<-data.frame(y,completedDataS3)
XXXS3
# Impute missing values using Bayesian imputation
imputed_data <- mice(XX[, c("x1", "x2", "x3", "x4", "x5")], method = "norm.predict")    # Use correct variable names and select only the predictor variables
# Extract the completed dataset from the imputed_data object
XXX4 <- complete(imputed_data)
BayesData <- data.frame(y,XXX4)
#parameter estimation, unconditional logit and conditional logit for Complete Original Data
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#parameter estimation, unconditional logit and conditional logit for Scenario S3
glm.out3 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS3)
clogit3=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS3)
#parameter estimation, unconditional logit and conditional logit for BayesData
glm.outBayes = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=BayesData)
clogitBayes=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = BayesData)
#Calling out coefficients
A[i,] <- glm.out1$coef[j+1]
B[i,] <- clogit1$coef[j]
C[i,] <- glm.out2$coef[j+1]
D[i,] <- clogit2$coef[j]
E[i,] <- glm.out3$coef[j+1]
F[i,] <- clogit3$coef[j]
G[i,] <- glm.outBayes$coef[j+1]
H[i,] <- clogitBayes$coef[j] }
#mean bias#
mean.biasA<- c(b1-mean(A[,1]), b2-mean(A[,2]),b3-mean(A[,3]),b4-mean(A[,4]),b5-mean(A[,5]))
mean.biasB<- c(b1-mean(B[,1]), b2-mean(B[,2]),b3-mean(B[,3]),b4-mean(B[,4]),b5-mean(B[,5]))
mean.biasC<- c(b1-mean(C[,1]), b2-mean(C[,2]),b3-mean(C[,3]),b4-mean(C[,4]),b5-mean(C[,5]))
mean.biasD<- c(b1-mean(D[,1]), b2-mean(D[,2]),b3-mean(D[,3]),b4-mean(D[,4]),b5-mean(D[,5]))
mean.biasE<- c(b1-mean(E[,1]), b2-mean(E[,2]),b3-mean(E[,3]),b4-mean(E[,4]),b5-mean(E[,5]))
mean.biasF<- c(b1-mean(F[,1]), b2-mean(F[,2]),b3-mean(F[,3]),b4-mean(F[,4]),b5-mean(F[,5]))
mean.biasG<- c(b1-mean(G[,1]), b2-mean(G[,2]),b3-mean(G[,3]),b4-mean(G[,4]),b5-mean(G[,5]))
mean.biasH<- c(b1-mean(H[,1]), b2-mean(H[,2]),b3-mean(H[,3]),b4-mean(H[,4]),b5-mean(H[,5]))
#residual errors#
```

```
rmseA<-c(sqrt(mean((b1-A[,1])^2)), sqrt(mean((b2-A[,2])^2)), sqrt(mean((b3-A[,3])^2)), sqrt(mean((b4-A[,4])^2)),
sqrt(mean((b5-A[,5])^2)))
rmseB<-c(sqrt(mean((b1-B[,1])^2)), sqrt(mean((b2-B[,2])^2)), sqrt(mean((b3-B[,3])^2)), sqrt(mean((b4-B[,4])^2)),
sqrt(mean((b5-B[,5])^2)))
rmseC<-c(sqrt(mean((b1-C[,1])^2)), sqrt(mean((b2-C[,2])^2)), sqrt(mean((b3-C[,3])^2)), sqrt(mean((b4-C[,4])^2)),
sqrt(mean((b5-C[,5])^2)))
rmseD<-c(sqrt(mean((b1-D[,1])^2)), sqrt(mean((b2-D[,2])^2)), sqrt(mean((b3-D[,3])^2)), sqrt(mean((b4-D[,4])^2)),
sqrt(mean((b5-D[,5])^2)))
rmseE<-c(sqrt(mean((b1-E[,1])^2)), sqrt(mean((b2-E[,2])^2)), sqrt(mean((b3-E[,3])^2)), sqrt(mean((b4-E[,4])^2)),
sqrt(mean((b5-E[,5])^2)))
rmseF<-c(sqrt(mean((b1-F[,1])^2)), sqrt(mean((b2-F[,2])^2)), sqrt(mean((b3-F[,3])^2)), sqrt(mean((b4-F[,4])^2)),
sqrt(mean((b5-F[,5])^2)))
rmseG<-c(sqrt(mean((b1-G[,1])^2)), sqrt(mean((b2-G[,2])^2)), sqrt(mean((b3-G[,3])^2)), sqrt(mean((b4-G[,4])^2)),
sqrt(mean((b5-G[,5])^2)))
rmseH<-c(sqrt(mean((b1-H[,1])^2)), sqrt(mean((b2-H[,2])^2)), sqrt(mean((b3-H[,3])^2)), sqrt(mean((b4-H[,4])^2)),
sqrt(mean((b5-H[,5])^2)))
mean.biasA
rmseA
mean.biasB
rmseB
mean.biasC
rmseC
mean.biasD
rmseD
mean.biasE
rmseE
mean.biasF
rmseF
mean.biasG
rmseG
mean.biasH
rmseH
```

n= 100 with 30% missingness

```
set.seed(12345678)        # Use this to make the randomly generated data the same each time you run the simulation#
N.iter=1000
j= 1:5
A <- array(0, dim=c(N.iter,5))
B <- array(0, dim=c(N.iter,5))
C <- array(0, dim=c(N.iter,5))
D <- array(0, dim=c(N.iter,5))
E <- array(0, dim=c(N.iter,5))
F <- array(0, dim=c(N.iter,5))
G <- array(0, dim=c(N.iter,5))
```

```
H <- array(0, dim=c(N.iter,5))
for(i in 1:N.iter){
n=100                                      # vary n
t=2          # vary t and change time in pData
nt=n*t
b1=1
b2=-1
b3=1
b4=1
b5=1
ai=rnorm(n,0,1)
x1=rnorm(nt,0,1)
x2 <- runif(nt,0,1)
x3 <- rnorm(nt,0.5,0.5)
x4 <- rbinom(nt,2,0.65)                   # nt different such numbers
hit=rnorm(nt,0,1)
uit=rnorm(nt,0,1)
vit=log(abs(uit/(1+uit)))
x5 <- ifelse(x1+hit>0, 1, 0)
alphai <- sqrt(t)*sum(x1)/n+ai #alphai simulation
ci <- kronecker(alphai ,matrix(1,t,1))#kronecker of alphai
wit=ci+b1*x1+b2*x2+b3*x3+b4*x4+b5*x5
zit= wit+vit
y <- ifelse(zit>0, 1, 0)
pData <- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,zit,ci,x1,x2,x3,x4,x5)
# Randomly Insert A Certain Proportion Of NAs Into A Dataframe #
pData2<- cbind(x1,x2,x3,x4,x5)
######################################
NAins <- NAinsert <- function(df, prop = .3){
 n <- nrow(df)
 m <- ncol(df)
 num.to.na <- ceiling(prop*n*m)
 id <- sample(0:(m*n-1), num.to.na, replace = FALSE)
 rows <- id %/% m + 1
 cols <- id %% m + 1
 sapply(seq(num.to.na), function(x){
 df[rows[x], cols[x]] <<- NA
 }
 )
 return(df)
}
##############
# TRY IT OUT #
```

```
##############
XX<-NAins(pData2, .3)
XX
pData3<- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,XX)
pData3
#Scenario 1 S1 when f(yi)=0
yS1<-y*0
XXX<-data.frame(yS1,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX,2,pMiss)
apply(XXX,1,pMiss)
#Performing MICE in XXX
library(mice)
md.pattern(XXX)
library(VIM)
aggr_plot <- aggr(XXX, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX), cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData <- mice(XXX,m=5,maxit=50,meth='pmm',seed=500)
summary(tempData)
#to see the new imputed covariate values, say X1
tempData$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData)
#getting back the completed data and binding with original y
completedData <- complete(tempData,1)
completedDataS1<- subset(completedData, select=c(x1,x2,x3,x4,x5))
XXXS1<-data.frame(y,completedDataS1)
XXXS1
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#Scenario 3 S3 when f(yi)=E(yit)=p
S3=rep(mean(y),times=nt)
yS3<-data.frame(S3)
XXX3<-data.frame(yS3,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX3,2,pMiss)
apply(XXX3,1,pMiss)
#Performing MICE in XXX3
```

```
library(mice)
md.pattern(XXX3)
library(VIM)
aggr_plot <- aggr(XXX3, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX3), cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData3 <- mice(XXX3,m=5,maxit=50,meth='norm',seed=500)
summary(tempData3)
#to see the new imputed covariate values, say X1
tempData3$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData3)
#getting back the completed data and binding with original y
completedData3 <- complete(tempData3,1)
completedDataS3<- subset(completedData3, select=c(x1,x2,x3,x4,x5))
XXXS3<-data.frame(y,completedDataS3)
XXXS3
# Impute missing values using Bayesian imputation
imputed_data <- mice(XX[, c("x1", "x2", "x3", "x4", "x5")], method = "norm.predict")   # Use correct variable names and select only the predictor variables
# Extract the completed dataset from the imputed_data object
XXX4 <- complete(imputed_data)
BayesData <- data.frame(y,XXX4)
#parameter estimation, unconditional logit and conditional logit for Complete Original Data
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#parameter estimation, unconditional logit and conditional logit for Scenario S3
glm.out3 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS3)
clogit3=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS3)
#parameter estimation, unconditional logit and conditional logit for BayesData
glm.outBayes = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=BayesData)
clogitBayes=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = BayesData)
#Calling out coefficients
A[i,] <- glm.out1$coef[j+1]
B[i,] <- clogit1$coef[j]
C[i,] <- glm.out2$coef[j+1]
D[i,] <- clogit2$coef[j]
E[i,] <- glm.out3$coef[j+1]
F[i,] <- clogit3$coef[j]
G[i,] <- glm.outBayes$coef[j+1]
H[i,] <- clogitBayes$coef[j] }
```

```
#mean bias#
mean.biasA<- c(b1-mean(A[,1]), b2-mean(A[,2]),b3-mean(A[,3]),b4-mean(A[,4]),b5-mean(A[,5]))
mean.biasB<- c(b1-mean(B[,1]), b2-mean(B[,2]),b3-mean(B[,3]),b4-mean(B[,4]),b5-mean(B[,5]))
mean.biasC<- c(b1-mean(C[,1]), b2-mean(C[,2]),b3-mean(C[,3]),b4-mean(C[,4]),b5-mean(C[,5]))
mean.biasD<- c(b1-mean(D[,1]), b2-mean(D[,2]),b3-mean(D[,3]),b4-mean(D[,4]),b5-mean(D[,5]))
mean.biasE<- c(b1-mean(E[,1]), b2-mean(E[,2]),b3-mean(E[,3]),b4-mean(E[,4]),b5-mean(E[,5]))
mean.biasF<- c(b1-mean(F[,1]), b2-mean(F[,2]),b3-mean(F[,3]),b4-mean(F[,4]),b5-mean(F[,5]))
mean.biasG<- c(b1-mean(G[,1]), b2-mean(G[,2]),b3-mean(G[,3]),b4-mean(G[,4]),b5-mean(G[,5]))
mean.biasH<- c(b1-mean(H[,1]), b2-mean(H[,2]),b3-mean(H[,3]),b4-mean(H[,4]),b5-mean(H[,5]))
#residual errors#
rmseA<-c(sqrt(mean((b1-A[,1])^2)), sqrt(mean((b2-A[,2])^2)), sqrt(mean((b3-A[,3])^2)), sqrt(mean((b4-A[,4])^2)),
sqrt(mean((b5-A[,5])^2)))
rmseB<-c(sqrt(mean((b1-B[,1])^2)), sqrt(mean((b2-B[,2])^2)), sqrt(mean((b3-B[,3])^2)), sqrt(mean((b4-B[,4])^2)),
sqrt(mean((b5-B[,5])^2)))
rmseC<-c(sqrt(mean((b1-C[,1])^2)), sqrt(mean((b2-C[,2])^2)), sqrt(mean((b3-C[,3])^2)), sqrt(mean((b4-C[,4])^2)),
sqrt(mean((b5-C[,5])^2)))
rmseD<-c(sqrt(mean((b1-D[,1])^2)), sqrt(mean((b2-D[,2])^2)), sqrt(mean((b3-D[,3])^2)), sqrt(mean((b4-D[,4])^2)),
sqrt(mean((b5-D[,5])^2)))
rmseE<-c(sqrt(mean((b1-E[,1])^2)), sqrt(mean((b2-E[,2])^2)), sqrt(mean((b3-E[,3])^2)), sqrt(mean((b4-E[,4])^2)),
sqrt(mean((b5-E[,5])^2)))
rmseF<-c(sqrt(mean((b1-F[,1])^2)), sqrt(mean((b2-F[,2])^2)), sqrt(mean((b3-F[,3])^2)), sqrt(mean((b4-F[,4])^2)),
sqrt(mean((b5-F[,5])^2)))
rmseG<-c(sqrt(mean((b1-G[,1])^2)), sqrt(mean((b2-G[,2])^2)), sqrt(mean((b3-G[,3])^2)), sqrt(mean((b4-G[,4])^2)),
sqrt(mean((b5-G[,5])^2)))
rmseH<-c(sqrt(mean((b1-H[,1])^2)), sqrt(mean((b2-H[,2])^2)), sqrt(mean((b3-H[,3])^2)), sqrt(mean((b4-H[,4])^2)),
sqrt(mean((b5-H[,5])^2)))
mean.biasA
rmseA
mean.biasB
rmseB
mean.biasC
rmseC
mean.biasD
rmseD
mean.biasE
rmseE
mean.biasF
rmseF
mean.biasG
rmseG
mean.biasH
rmseH
```

n= 250 with 10% missingness

```
set.seed(123456789)     # Use this to make the randomly generated data the same each time you run the simulation#
N.iter=1000
j= 1:5
A <- array(0, dim=c(N.iter,5))
B <- array(0, dim=c(N.iter,5))
C <- array(0, dim=c(N.iter,5))
D <- array(0, dim=c(N.iter,5))
E <- array(0, dim=c(N.iter,5))
F <- array(0, dim=c(N.iter,5))
G <- array(0, dim=c(N.iter,5))
H <- array(0, dim=c(N.iter,5))
for(i in 1:N.iter){
n=250                                   # vary n
t=2           # vary t and change time in pData
nt=n*t
b1=1
b2=-1
b3=1
b4=1
b5=1
ai=rnorm(n,0,1)
x1=rnorm(nt,0,1)
x2 <- runif(nt,0,1)
x3 <- rnorm(nt,0.5,0.5)
x4 <- rbinom(nt,2,0.65)                 # nt different such numbers
hit=rnorm(nt,0,1)
uit=rnorm(nt,0,1)
vit=log(abs(uit/(1+uit)))
x5 <- ifelse(x1+hit>0, 1, 0)
alphai <- sqrt(t)*sum(x1)/n+ai #alphai simulation
ci <- kronecker(alphai ,matrix(1,t,1))#kronecker of alphai
wit=ci+b1*x1+b2*x2+b3*x3+b4*x4+b5*x5
zit= wit+vit
y <- ifelse(zit>0, 1, 0)
pData <- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,zit,ci,x1,x2,x3,x4,x5)
# Randomly Insert A Certain Proportion Of NAs Into A Dataframe #
pData2<- cbind(x1,x2,x3,x4,x5)
##################################
NAins <- NAinsert <- function(df, prop = .1){
  n <- nrow(df)
  m <- ncol(df)
  num.to.na <- ceiling(prop*n*m)
  id <- sample(0:(m*n-1), num.to.na, replace = FALSE)
```

```
 rows <- id %/% m + 1
 cols <- id %% m + 1
 sapply(seq(num.to.na), function(x){
 df[rows[x], cols[x]] <<- NA
 }
 )
 return(df)
}
##############
# TRY IT OUT #
##############
XX<-NAins(pData2, .1)
XX
pData3<- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,XX)
pData3

#Scenario 1 S1 when f(yi)=0
yS1<-y*0
XXX<-data.frame(yS1,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX,2,pMiss)
apply(XXX,1,pMiss)
#Performing MICE in XXX
library(mice)
md.pattern(XXX)
library(VIM)
aggr_plot <- aggr(XXX, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX), cex.axis=.7,
gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData <- mice(XXX,m=5,maxit=50,meth='pmm',seed=500)
summary(tempData)
#to see the new imputed covariate values, say X1
tempData$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData)
#getting back the completed data and binding with original y
completedData <- complete(tempData,1)
completedDataS1<- subset(completedData, select=c(x1,x2,x3,x4,x5))
XXXS1<-data.frame(y,completedDataS1)
XXXS1
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
```

```
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#Scenario 3 S3 when f(yi)=E(yit)=p
S3=rep(mean(y),times=nt)
yS3<-data.frame(S3)
XXX3<-data.frame(yS3,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX3,2,pMiss)
apply(XXX3,1,pMiss)
#Performing MICE in XXX3
library(mice)
md.pattern(XXX3)
library(VIM)
aggr_plot <- aggr(XXX3, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX3), cex.axis=.7, gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData3 <- mice(XXX3,m=5,maxit=50,meth='norm',seed=500)
summary(tempData3)
#to see the new imputed covariate values, say X1
tempData3$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData3)
#getting back the completed data and binding with original y
completedData3 <- complete(tempData3,1)
completedDataS3<- subset(completedData3, select=c(x1,x2,x3,x4,x5))
XXXS3<-data.frame(y,completedDataS3)
XXXS3
# Impute missing values using Bayesian imputation
imputed_data <- mice(XX[, c("x1", "x2", "x3", "x4", "x5")], method = "norm.predict")   # Use correct variable names and select only the predictor variables
# Extract the completed dataset from the imputed_data object
XXX4 <- complete(imputed_data)
BayesData <- data.frame(y,XXX4)
#parameter estimation, unconditional logit and conditional logit for Complete Original Data
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#parameter estimation, unconditional logit and conditional logit for Scenario S3
glm.out3 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS3)
clogit3=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS3)
#parameter estimation, unconditional logit and conditional logit for BayesData
```

```
glm.outBayes = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=BayesData)
clogitBayes=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = BayesData)
#Calling out coefficients
A[i,] <- glm.out1$coef[j+1]
B[i,] <- clogit1$coef[j]
C[i,] <- glm.out2$coef[j+1]
D[i,] <- clogit2$coef[j]
E[i,] <- glm.out3$coef[j+1]
F[i,] <- clogit3$coef[j]
G[i,] <- glm.outBayes$coef[j+1]
H[i,] <- clogitBayes$coef[j] }
#mean bias#
mean.biasA<- c(b1-mean(A[,1]), b2-mean(A[,2]),b3-mean(A[,3]),b4-mean(A[,4]),b5-mean(A[,5]))
mean.biasB<- c(b1-mean(B[,1]), b2-mean(B[,2]),b3-mean(B[,3]),b4-mean(B[,4]),b5-mean(B[,5]))
mean.biasC<- c(b1-mean(C[,1]), b2-mean(C[,2]),b3-mean(C[,3]),b4-mean(C[,4]),b5-mean(C[,5]))
mean.biasD<- c(b1-mean(D[,1]), b2-mean(D[,2]),b3-mean(D[,3]),b4-mean(D[,4]),b5-mean(D[,5]))
mean.biasE<- c(b1-mean(E[,1]), b2-mean(E[,2]),b3-mean(E[,3]),b4-mean(E[,4]),b5-mean(E[,5]))
mean.biasF<- c(b1-mean(F[,1]), b2-mean(F[,2]),b3-mean(F[,3]),b4-mean(F[,4]),b5-mean(F[,5]))
mean.biasG<- c(b1-mean(G[,1]), b2-mean(G[,2]),b3-mean(G[,3]),b4-mean(G[,4]),b5-mean(G[,5]))
mean.biasH<- c(b1-mean(H[,1]), b2-mean(H[,2]),b3-mean(H[,3]),b4-mean(H[,4]),b5-mean(H[,5]))
#residual errors#
rmseA<-c(sqrt(mean((b1-A[,1])^2)), sqrt(mean((b2-A[,2])^2)), sqrt(mean((b3-A[,3])^2)), sqrt(mean((b4-A[,4])^2)),
sqrt(mean((b5-A[,5])^2)))
rmseB<-c(sqrt(mean((b1-B[,1])^2)), sqrt(mean((b2-B[,2])^2)), sqrt(mean((b3-B[,3])^2)), sqrt(mean((b4-B[,4])^2)),
sqrt(mean((b5-B[,5])^2)))
rmseC<-c(sqrt(mean((b1-C[,1])^2)), sqrt(mean((b2-C[,2])^2)), sqrt(mean((b3-C[,3])^2)), sqrt(mean((b4-C[,4])^2)),
sqrt(mean((b5-C[,5])^2)))
rmseD<-c(sqrt(mean((b1-D[,1])^2)), sqrt(mean((b2-D[,2])^2)), sqrt(mean((b3-D[,3])^2)), sqrt(mean((b4-D[,4])^2)),
sqrt(mean((b5-D[,5])^2)))
rmseE<-c(sqrt(mean((b1-E[,1])^2)), sqrt(mean((b2-E[,2])^2)), sqrt(mean((b3-E[,3])^2)), sqrt(mean((b4-E[,4])^2)),
sqrt(mean((b5-E[,5])^2)))
rmseF<-c(sqrt(mean((b1-F[,1])^2)), sqrt(mean((b2-F[,2])^2)), sqrt(mean((b3-F[,3])^2)), sqrt(mean((b4-F[,4])^2)),
sqrt(mean((b5-F[,5])^2)))
rmseG<-c(sqrt(mean((b1-G[,1])^2)), sqrt(mean((b2-G[,2])^2)), sqrt(mean((b3-G[,3])^2)), sqrt(mean((b4-G[,4])^2)),
sqrt(mean((b5-G[,5])^2)))
rmseH<-c(sqrt(mean((b1-H[,1])^2)), sqrt(mean((b2-H[,2])^2)), sqrt(mean((b3-H[,3])^2)), sqrt(mean((b4-H[,4])^2)),
sqrt(mean((b5-H[,5])^2)))
mean.biasA
rmseA
mean.biasB
rmseB
mean.biasC
rmseC
```

```
mean.biasD
rmseD
mean.biasE
rmseE
mean.biasF
rmseF
mean.biasG
rmseG
mean.biasH
rmseH
```

n= 250 with 30% missingness

```
set.seed(1234567899)     # Use this to make the randomly generated data the same each time you run the simulation#
N.iter=1000
j= 1:5
A <- array(0, dim=c(N.iter,5))
B <- array(0, dim=c(N.iter,5))
C <- array(0, dim=c(N.iter,5))
D <- array(0, dim=c(N.iter,5))
E <- array(0, dim=c(N.iter,5))
F <- array(0, dim=c(N.iter,5))
G <- array(0, dim=c(N.iter,5))
H <- array(0, dim=c(N.iter,5))
for(i in 1:N.iter){
n=250                                        # vary n
t=2          # vary t and change time in pData
nt=n*t
b1=1
b2=-1
b3=1
b4=1
b5=1
ai=rnorm(n,0,1)
x1=rnorm(nt,0,1)
x2 <- runif(nt,0,1)
x3 <- rnorm(nt,0.5,0.5)
x4 <- rbinom(nt,2,0.65)                     # nt different such numbers
hit=rnorm(nt,0,1)
uit=rnorm(nt,0,1)
vit=log(abs(uit/(1+uit)))
x5 <- ifelse(x1+hit>0, 1, 0)
alphai <- sqrt(t)*sum(x1)/n+ai #alphai simulation    ci <- kronecker(alphai ,matrix(1,t,1))#kronecker of alphai
wit=ci+b1*x1+b2*x2+b3*x3+b4*x4+b5*x5
zit= wit+vit
```

```
y <- ifelse(zit>0, 1, 0)
pData <- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,zit,ci,x1,x2,x3,x4,x5)
# Randomly Insert A Certain Proportion Of NAs Into A Dataframe #
pData2<- cbind(x1,x2,x3,x4,x5)
##################################
NAins <- NAinsert <- function(df, prop = .3){
 n <- nrow(df)
 m <- ncol(df)
 num.to.na <- ceiling(prop*n*m)
 id <- sample(0:(m*n-1), num.to.na, replace = FALSE)
 rows <- id %/% m + 1
 cols <- id %% m + 1
 sapply(seq(num.to.na), function(x){
 df[rows[x], cols[x]] <<- NA
 }
 )
 return(df)
}
##############
# TRY IT OUT #
##############
XX<-NAins(pData2, .3)
XX
pData3<- data.frame(id = rep(paste("stdnt", 1:n, sep = "_"), each = t),time = rep(1:2, n), y,XX)
pData3
#Scenario 1 S1 when f(yi)=0
yS1<-y*0
XXX<-data.frame(yS1,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX,2,pMiss)
apply(XXX,1,pMiss)
#Performing MICE in XXX
library(mice)
md.pattern(XXX)
library(VIM)
aggr_plot <- aggr(XXX, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX), cex.axis=.7,
gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData <- mice(XXX,m=5,maxit=50,meth='pmm',seed=500)
summary(tempData)
#to see the new imputed covariate values, say X1
tempData$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
```

```
densityplot(tempData)
#getting back the completed data and binding with original y
completedData <- complete(tempData,1)
completedDataS1<- subset(completedData, select=c(x1,x2,x3,x4,x5))
XXXS1<-data.frame(y,completedDataS1)
XXXS1
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#Scenario 3 S3 when f(yi)=E(yit)=p
S3=rep(mean(y),times=nt)
yS3<-data.frame(S3)
XXX3<-data.frame(yS3,XX)
#check missingness proportions by covariate
pMiss <- function(x){sum(is.na(x))/length(x)*100}
apply(XXX3,2,pMiss)
apply(XXX3,1,pMiss)
#Performing MICE in XXX3
library(mice)
md.pattern(XXX3)
library(VIM)
aggr_plot <- aggr(XXX3, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE, labels=names(XXX3), cex.axis=.7,
gap=3, ylab=c("Histogram of missing data","Pattern"))
tempData3 <- mice(XXX3,m=5,maxit=50,meth='norm',seed=500)
summary(tempData3)
#to see the new imputed covariate values, say X1
tempData3$imp$x1
#showing covariate densities of observed data(blue) and imputed data( magenta)
densityplot(tempData3)
#getting back the completed data and binding with original y
completedData3 <- complete(tempData3,1)
completedDataS3<- subset(completedData3, select=c(x1,x2,x3,x4,x5))
XXXS3<-data.frame(y,completedDataS3)
XXXS3
# Impute missing values using Bayesian imputation
imputed_data <- mice(XX[, c("x1", "x2", "x3", "x4", "x5")], method = "norm.predict")   # Use correct variable names
and select only the predictor variables
# Extract the completed dataset from the imputed_data object
XXX4 <- complete(imputed_data)
BayesData <- data.frame(y,XXX4)
#parameter estimation, unconditional logit and conditional logit for Complete Original Data
```

```
glm.out1 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=pData)
clogit1=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = pData)
#parameter estimation, unconditional logit and conditional logit for Scenario S1
glm.out2 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS1)
clogit2=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS1)
#parameter estimation, unconditional logit and conditional logit for Scenario S3
glm.out3 = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=XXXS3)
clogit3=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = XXXS3)
#parameter estimation, unconditional logit and conditional logit for BayesData
glm.outBayes = glm(y ~x1+x2+x3+x4+x5,family=binomial(logit), data=BayesData)
clogitBayes=clogit((y ~ (x1 +x2+x3+x4+x5)), strata (id), data = BayesData)
#Calling out coefficients
A[i,] <- glm.out1$coef[j+1]
B[i,] <- clogit1$coef[j]
C[i,] <- glm.out2$coef[j+1]
D[i,] <- clogit2$coef[j]
E[i,] <- glm.out3$coef[j+1]
F[i,] <- clogit3$coef[j]
G[i,] <- glm.outBayes$coef[j+1]
H[i,] <- clogitBayes$coef[j] }
#mean bias#
mean.biasA<- c(b1-mean(A[,1]), b2-mean(A[,2]),b3-mean(A[,3]),b4-mean(A[,4]),b5-mean(A[,5]))
mean.biasB<- c(b1-mean(B[,1]), b2-mean(B[,2]),b3-mean(B[,3]),b4-mean(B[,4]),b5-mean(B[,5]))
mean.biasC<- c(b1-mean(C[,1]), b2-mean(C[,2]),b3-mean(C[,3]),b4-mean(C[,4]),b5-mean(C[,5]))
mean.biasD<- c(b1-mean(D[,1]), b2-mean(D[,2]),b3-mean(D[,3]),b4-mean(D[,4]),b5-mean(D[,5]))
mean.biasE<- c(b1-mean(E[,1]), b2-mean(E[,2]),b3-mean(E[,3]),b4-mean(E[,4]),b5-mean(E[,5]))
mean.biasF<- c(b1-mean(F[,1]), b2-mean(F[,2]),b3-mean(F[,3]),b4-mean(F[,4]),b5-mean(F[,5]))
mean.biasG<- c(b1-mean(G[,1]), b2-mean(G[,2]),b3-mean(G[,3]),b4-mean(G[,4]),b5-mean(G[,5]))
mean.biasH<- c(b1-mean(H[,1]), b2-mean(H[,2]),b3-mean(H[,3]),b4-mean(H[,4]),b5-mean(H[,5]))
#residual errors#
rmseA<-c(sqrt(mean((b1-A[,1])^2)), sqrt(mean((b2-A[,2])^2)), sqrt(mean((b3-A[,3])^2)), sqrt(mean((b4-A[,4])^2)),
sqrt(mean((b5-A[,5])^2)))
rmseB<-c(sqrt(mean((b1-B[,1])^2)), sqrt(mean((b2-B[,2])^2)), sqrt(mean((b3-B[,3])^2)), sqrt(mean((b4-B[,4])^2)),
sqrt(mean((b5-B[,5])^2)))
rmseC<-c(sqrt(mean((b1-C[,1])^2)), sqrt(mean((b2-C[,2])^2)), sqrt(mean((b3-C[,3])^2)), sqrt(mean((b4-C[,4])^2)),
sqrt(mean((b5-C[,5])^2)))
rmseD<-c(sqrt(mean((b1-D[,1])^2)), sqrt(mean((b2-D[,2])^2)), sqrt(mean((b3-D[,3])^2)), sqrt(mean((b4-D[,4])^2)),
sqrt(mean((b5-D[,5])^2)))
rmseE<-c(sqrt(mean((b1-E[,1])^2)), sqrt(mean((b2-E[,2])^2)), sqrt(mean((b3-E[,3])^2)), sqrt(mean((b4-E[,4])^2)),
sqrt(mean((b5-E[,5])^2)))
rmseF<-c(sqrt(mean((b1-F[,1])^2)), sqrt(mean((b2-F[,2])^2)), sqrt(mean((b3-F[,3])^2)), sqrt(mean((b4-F[,4])^2)),
sqrt(mean((b5-F[,5])^2)))
rmseG<-c(sqrt(mean((b1-G[,1])^2)), sqrt(mean((b2-G[,2])^2)), sqrt(mean((b3-G[,3])^2)), sqrt(mean((b4-G[,4])^2)),
sqrt(mean((b5-G[,5])^2)))
rmseH<-c(sqrt(mean((b1-H[,1])^2)), sqrt(mean((b2-H[,2])^2)), sqrt(mean((b3-H[,3])^2)), sqrt(mean((b4-H[,4])^2)),
sqrt(mean((b5-H[,5])^2)))
mean.biasA
rmseA
mean.biasB
rmseB
mean.biasC
rmseC
mean.biasD
rmseD
mean.biasE
rmseE
mean.biasF
rmseF
mean.biasG
rmseG
mean.biasH
rmseH
```