Summer 8-9-2019

Beta Regression Models for Repeated-Measures Data Analysis

Nicholas A. Hein
University of Nebraska Medical Center

BETA REGRESSION MODELS FOR REPEATED-MEASURES DATA ANALYSIS

by

Nicholas Hein

A DISSERTATION

Presented to the Faculty of

the University of Nebraska Graduate College

in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy

Biostatistics
Graduate Program

Under the Supervision of Professor Christopher Wichman

University of Nebraska Medical Center
Omaha, Nebraska

August, 2019

Supervisory Committee:

Jane Meza, Ph.D.

Kendra Schmid, Ph.D.

Elizabeth Wellsandt, DPT, Ph.D.

Acknowledgments

This work was completed utilizing the Holland Computing Center of the University of Nebraska, which receives support from the Nebraska Research Initiative. Furthermore, this publication was made possible from NIH funding through the NIMH and NINDS Institutes by the following grants:

Texas NeuroAIDS Research Center: U24MH100930

California NeuroAIDS Tissue Network: U24MH100928

National Neurological AIDS Bank: U24MH100929

Manhattan HIV Brain Bank: U24MH100931

Data Coordinating Center: U24MH100925

Its contents are solely the responsibility of the authors and do not necessarily represent the official view of the NNTC or NIH.

Subsequently, throughout the writing of this dissertation, I have received a considerable amount of support and guidance. I would like to thank my supervisor, Dr. Wichman, whose expertise was invaluable during the entire process, especially during the formulating of the research topic. I would like to acknowledge my committee members, Drs. Meza, Schmid, and Wellsandt for their continued advice and feedback throughout the entire process.

In addition, I would like to thank my parents, Andrew and Cheryl, for their support and encouragement during my academic career. Finally, thank you to my wife, Elizabeth, for enduring this process with me, and always being supportive.

BETA REGRESSION MODELS FOR REPEATED-MEASURES DATA ANALYSIS

Nicholas A. Hein, Ph.D.

University of Nebraska, 2019

Supervisor: Christopher Wichman, Ph.D.

Bounded data often give rise to uncorrectable skew and heteroscedasticity. Bounded data are a relatively frequent occurrence in clinical and research settings. For example, in neuropsychology, most neurocognitive tests are bounded, and subjects are repeatedly measured over time. The statistician needs to choose a model that accounts for the correlated nature of the repeated measures. The Beta distribution is a natural choice for modeling bounded data. Currently, generalized linear mixed models (GLMM) and generalized estimating equations (GEE) are two methods that can be used to model Beta distributed data with repeated measures. However, GLMMs and GEEs have limitations, i.e., GLMMs require numerical integration and GEEs are not based on a joint likelihood making model selection more ambiguous. Therefore, we present two alternative models (LNMVB and SLMVB) that are based on a joint likelihood and do not require numerical integration for the estimation of the model parameters. We compare our proposed models to the Beta GLMM and the Beta GEE using simulated data and a real dataset from the National NeuroAIDS Tissue Consortium. Through simulation, we found the LNMVB and the Beta GEE were the only models that produced unbiased estimates of the location parameter for all scenarios considered. The LNMVB tended to have better control of the Type I error rate compared to the Beta GEE, especially for smaller sample sizes (i.e., $N \leq 30$). The coverage probabilities for both the LNMVB and the Beta GEE tended towards 95% as sample size increased with the LNMVB generally closer to the desired 95% coverage probability. Lastly, the Beta GEE was the only model that consistently had a mean bias near zero when estimating the correlation parameter. Based on simulated data, we conclude that the LNMVB is preferred for analyzing small sample (i.e., ≤ 30), repeatedly-measured proportional data. Either the LNMVB or the Beta GEE is sufficient to analyze large sample (i.e., ≥ 50), correlated Beta distributed data. Furthermore, if the correlation is the parameter of interest, the Beta GEE is the preferred model.

Table of Contents

Acknowledgments i

Abstract ii

Table of Contents iii

List of Figures vi

List of Tables xvii

List of Abbreviations xviii

1. Introductory Material 1

1.1. General Introduction 1

1.1.1. Introduction 1

1.1.2. Dissertation aims 5

1.2. Motivating Dataset 8

1.3. Literature Review 9

1.3.1. Introduction 9

1.3.2. Notation 10

1.3.3. Beta regression for independent responses 10

1.3.4. Multivariate Beta distribution through construction 12

1.3.5. Multivariate Beta distribution using copulas 17

1.3.6. The Beta Marginal Model 23

1.3.7. The Beta Generalized Linear Mixed Model 26

2. Methodological Contributions 29

2.1. Introduction 29

2.2. Notation 29

2.3. Libby and Novick Multivariate Beta (LNMVB) 30

2.4. Sarmanov-Lee Multivariate Beta (SLMVB) 39

3. Simulation Study 54

3.1. Introduction 54

3.2. Design of simulation study 54

3.3. Simulation of data 55

3.4. Model fitting 56

3.5. Metrics 57

3.6. Results 60

3.6.1. One group: $\mu = 0.05$, AR(1) correlation 60

3.6.2. One group: $\mu = 0.3$, AR(1) correlation 65

3.6.3. One group: $\mu = 0.5$, AR(1) correlation 67

3.6.4. One group: $\mu = 0.05$, CS correlation 69

3.6.5. One group: $\mu = 0.3$, CS correlation 71

3.6.6. One group: $\mu = 0.5$, CS correlation 73

3.6.7. Two groups: $\mu = 0.05$, AR(1) correlation 75

3.6.8. Two groups: $\mu = 0.3$, AR(1) correlation 77

3.6.9. Two groups: $\mu = 0.5$, AR(1) correlation 79

3.6.10. Two groups: $\mu = 0.05$, CS correlation 81

3.6.11. Two groups: $\mu = 0.3$, CS correlation 83

3.6.12. Two groups: $\mu = 0.5$, CS correlation 85

3.7. Summary 87

3.7.1. One group 87

3.7.2. Two groups 88

4. Data Analysis, National NeuroAIDS Tissue Consortium 89

4.1. Participants 89

4.2. Methods 91

4.3. Results 92

5. General Conclusions, Limitations and Future Research 95

5.1. Introduction 95

5.2. Conclusions 96

5.3. Limitations 98

5.4. Future Research 100

6. Bibliography 102

Appendix A: One Group Simulation Results 110

Appendix B: Two Group Simulation Results 150

Appendix C: NNTC Results, SLMVB AR(1) Model 190

Appendix D: R Code 192

List of Figures

Figure 3.1 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$. 61

Figure 3.2 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$....................63

Figure 3.3 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$. 64

Figure 4.1 Schematic summary of inclusion/exclusion criteria of NNTC data....90

Figure 4.2 Mean profiles plots of NNTC data using LNMVB, SLMVB CS, GLMM, and GEE models. 93

Figure A.1 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$. 110

Figure A.2 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$....................111

Figure A.3 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$. 112

Figure A.4 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1.$ 113

Figure A.5 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3.$ 114

Figure A.6 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3.$ 115

Figure A.7 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3.$ 116

Figure A.8 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3.$ 117

Figure A.9 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5.$ 118

Figure A.10 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5.$ 119

Figure A.11 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5.$ 120

Figure A.12 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5.$ 121

Figure A.13 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1.$ 122

Figure A.14 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$..123

Figure A.15 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1.$ 124

Figure A.16 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1.$ 125

Figure A.17 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3.$ 126

Figure A.18 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$..127

Figure A.19 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3.$ 128

Figure A.20 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$. 129

Figure A.21 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$. 130

Figure A.22 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$. 131

Figure A.23 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$. 132

Figure A.24 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$. 133

Figure A.25 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$. 134

Figure A.26 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$. 135

Figure A.27 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$. 136

Figure A.28 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1.$ 137

Figure A.29 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3.$ 138

Figure A.30 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3.$ 139

Figure A.31 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3.$ 140

Figure A.32 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3.$ 141

Figure A.33 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5.$ 142

Figure A.34 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5.$ 143

Figure A.35 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5.$ 144

Figure A.36 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5$. 145

Figure A.37 Summary of Type I error for one group simulations using 1000 replicates with AR(1) correlation structure. 146

Figure A.38 Summary of coverage probabilities for one group simulations using 1000 replicates with AR(1) correlation structure. 147

Figure A.39 Summary of bias of location parameter for one group simulations using 1000 replicates with AR(1) correlation structure. 148

Figure A.40 Summary of model convergence for one group simulations using 1000 replicates with AR(1) correlation structure. 149

Figure B.1 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$. 150

Figure B.2 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$...151

Figure B.3 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$. 152

Figure B.4 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$. 153

Figure B.5 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3.$ 154

Figure B.6 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3.$ 155

Figure B.7 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3.$ 156

Figure B.8 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3.$ 157

Figure B.9 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5.$ 158

Figure B.10 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5.$ 159

Figure B.11 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5.$ 160

Figure B.12 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5.$ 161

Figure B.13 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$. 162

Figure B.14 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$..163

Figure B.15 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$. 164

Figure B.16 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$. 165

Figure B.17 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$. 166

Figure B.18 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$..167

Figure B.19 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$. 168

Figure B.20 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$. 169

Figure B.21 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$. 170

Figure B.22 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$..171

Figure B.23 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$. 172

Figure B.24 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$. 173

Figure B.25 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$. 174

Figure B.26 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$..175

Figure B.27 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$. 176

Figure B.28 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$. 177

Figure B.29 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3.$ 178

Figure B.30 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3.$ 179

Figure B.31 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3.$ 180

Figure B.32 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3.$ 181

Figure B.33 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5.$ 182

Figure B.34 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5.$ 183

Figure B.35 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5.$ 184

Figure B.36 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5.$ 185

Figure B.37 Summary of Type I error for two group simulations using 1000 replicates with AR(1) correlation structure. 186

Figure B.38 Summary of coverage probabilities for two group simulations using 1000 replicates with AR(1) correlation structure. 187

Figure B.39 Summary of bias of location parameter for two group simulations using 1000 replicates with AR(1) correlation structure. 188

Figure B.40 Summary of model convergence for two group simulations using 1000 replicates with AR(1) correlation structure. 189

Figure C.1 Mean profiles plots of NNTC data using SLMVB AR(1) model.......190

List of Tables

Table 3.1 Simulation parameters..55

Table 4.1 Patient characteristics. 91

Table 4.2 Model estimates (LNMVB, SLMVB CS, GLMM, and GEE) of the NNTC data. 94

Table 4.3 Empirical pairwise correlation estimates of the NNTC data..............95

Table C.1 SLMVB AR(1) model estimates of the NNTC data.........................191

List of Abbreviations

AIC	Akaike information criterion
AR(1)	auto-regressive(1)
BIC	Bayesian information criterion
cdf	cumulative density function
CI	confidence interval
CS	compound symmetry
FGM	Farlie-Gumbel-Morgenstern
GDP	gross domestic product
GEE	generalized estimating equation
GLMM	generalized linear mixed model
HIV	human immunodeficiency virus
HVLT-R	Hopkins Verbal Learning Test-Revised
LNMVB	Libby and Novick Multivariate Beta
LR	likelihood ratio
MCC	multivariate correlation coefficient
MLE	maximum likelihood estimator
NNTC	National NeuroAIDS Tissue Consortium
pdf	probability density function
QIC	quasi-likelihood information criterion
RMSD	root mean squared deviation

SE	standard error
SLMVB	Sarmanov-Lee Multivariate Beta

1. Introductory Material

1.1. General Introduction

1.1.1. Introduction

Today’s research results in extensive amounts of data being collected in a magnitude of fields, e.g., medical research, economics, manufacturing, sports sciences, social sciences, etc. For a statistician, the measured response, not the field, characterizes the data. Additionally, the response dictates what distributional assumptions are appropriate when modeling the data. Responses that are bounded on the closed unit interval $[0,1]$ are often referred to as proportions. The Beta distribution is one distribution that is useful for modeling proportions.

Another defining characteristic of the data is the study design or how the data is collected. A single outcome/response may be recorded for each unit of interest, such as, subject, patient, household, etc. along with characteristics of the study unit, e.g., gender, age, location, treatment status, etc. The data arising from this study design is commonly termed as cross-sectional data. However, it is not uncommon for more than one observation to be recorded for each unit of study. The response of interest could be observed and recorded sequentially through time; this study is referred to as a longitudinal study design. Alternatively, the response could be recorded under different conditions, which is commonly referred to as a repeated-measures design. For brevity, we will refer to a repeated-measures design as both a repeated-measures design and a longitudinal study design.

Repeatedly measured, proportional data in the biomedical field is quite common: oxygen saturation levels as measured by pulse oximetry; the forced expiratory volume in one second and the forced vital capacity ratio as measured by spirometer; percentage of knee torque of injured limb compared to knee torque of uninjured limb; etc. Additionally, in neuropsychology, proportional responses that are often repeatedly measured include score as a percentage on the Clinical Dementia Rating, the Boston Naming Test, and Differential Ability Scales. Outside of healthcare disciplines, economics uses proportion and percentage metrics that are often measured over time such as percent of gross domestic product (GDP), percent employed/unemployed, stock market capitalization to GDP, etc.

Often, a practitioner is interested in situations where the response can be modeled as a function of exogenous variables. Such analysis requires an understanding of the mechanism that generated the data such as sampling design or study design. It is vital that the statistical methodology used to analyze the data reflects the study design so valid conclusions and inferences can be made. In repeated-measures designs, dependence exists between the responses within the same unit of study. A practitioner needs to apply a method that takes into account the dependence in statistical analyses.

When analyzing dependent proportion data, three different frameworks exist. For n-repeated measures, a practitioner may choose from marginal models (Beta Generalized Estimating Equations) or subject-specific models (Beta Generalized Linear Mixed Models). It should be noted that a practitioner could choose a normal marginal model, Linear Mixed Model, or repeated-measures ANOVA; however, predictions using these models may lie outside the closed interval $[0,1]$.[1] When there are only two-repeated measures, for example, pre- and post-measurements, a practitioner may choose from the aforementioned Beta models or bivariate Beta models.

A marginal model is one where the mean response modeled is conditioned only on the covariates. Marginal models do not specify the full joint distribution of the data.

Marginal models define a mean function, a variance function, and a dependence structure between related observations.[1] If the conditional mean is correctly specified, Generalized Estimating Equations (GEEs) as proposed by Liang and Zeger[2] yield consistent estimators of the parameters.[3] The Beta GEE has a population-averaged interpretation of the response on the transformed scale of the regression coefficients. The source of dependence is not made explicit in the marginal model. Instead, the dependence is treated as a nuisance parameter.

In a generalized linear mixed model (GLMM), dependence is imposed through an unobserved heterogeneity, i.e., random effects, in the conditional mean specification. Adding random effects to the Beta regression model (Section 1.2.3) yields the Beta GLMM.[4] Parameter estimates are obtained by maximizing the marginal likelihood which is obtained by integrating out the random effects from the likelihood function.[1] It is standard practice to assume that the random effects are distributed multivariate normal with mean $\mathbf{0}$ and variance $\mathbf{\Sigma}$; however, other distributions are possible for the random effects.[5] Due to the non-linear transformation of the link function, the Beta GLMM parameters only have a subject-specific interpretation, i.e., a given individual's response on the transformed scale for a unit within-subject change in the corresponding parameter.[1]

The bivariate Beta is an extension to the Beta regression model presented in Section 1.3.3. The bivariate Beta can be constructed using Gamma random variables with shared parameters[6] or combing univariate marginal Beta distributions using copulas.[7-9] For the discussion on copulas, this dissertation will focus on the copulas for bivariate distributions defined by Sarmanov[7] and proposed multivariate extension by Lee[8]. Bivariate distributions created using Sarmanov[7] copulas are referred to as the Sarmanov family of bivariate distributions. There are examples in the literature of each method (through construction and through copulas) or extensions of the methods being used to fit bivariate proportional data.[9-13] The bivariate Beta either through construction or copulas allows for the parameters to be estimated using the method of maximum likelihood on the joint likelihood. However, research concerning the bivariate beta regression models has recently decreased, possibly associated with the implementation of the Beta GLMM and the Beta GEE in current statistical software.

Each of the three methodologies to analyze longitudinal proportional data are not without their limitations. In the GEE method, the dependence is specified through a working correlation, as defined by Pearson[14], whose parameters are estimated by the methods of moments.[3] This could result in a misspecification of the correlation structure. However, the estimates are robust against misspecification by using the empirical variance estimator.[1,15] Shults and colleagues[16] have demonstrated that the GEE method may provide infeasible estimates (ρ can exceed 1) for the correlation parameter when using the empirical variance estimator. Additionally, since the GEE method does not rely on maximum likelihood, the likelihood ratio (LR) test, Akaike information criterion (AIC), Bayesian information criterion (BIC), etc. are not available to help with model selection. However, Pan[17] has developed and advocated for using the quasi-likelihood information criterion (QIC) in choosing a working correlation and for selecting covariates. The QIC is not without limitations; Hin and Wang[18] note that any attempt to select the true correlation structure is distorted if the mean response is incorrectly specified.

GLMMs may appear to have a distinct advantage over marginal models. GLMMs are based on likelihoods, thereby allowing for model selection that uses likelihood criteria. However, the random effects need to be marginalized out before the method of maximum likelihood can be applied.[1] Assuming the random effects are multivariate normal, there is no closed form expression (i.e., an expression that can be evaluated in a finite number of operations) for the integral.[19] Fitzmaurice et al[1] and Tuerlinckx et al[19] summarize the methods that can be used to approximate the integral and their respective limitations. Two methods (i.e., penalized quasi-likelihood and marginalized quasi-likelihood) produce biased estimates under certain conditions.[1] Additionally, the target of inference of a marginal model appears obtainable from a GLMM by averaging over the distribution of unobserved heterogeneity. However, Fitzmaurice and colleagues[1] have shown this not to be true for non-linear link functions. Fitzmaurice and colleagues[1] also note that any misspecification of the GLMM can yield biased estimates of the implied marginal means.

It appears that the bivariate Beta may overcome some of the limitations of both the GEE and GLMM methods; however, the bivariate Beta is limited to two repeated measures. It is the goal of this dissertation to address this limitation of the bivariate Beta. Specifically, the bivariate Beta will be extended to n-repeated measures.

1.1.2. Dissertation aims

Using the construction proposed by Libby and Novick[6] and the methodology of Sarmanov[7] and Lee,[8] two closed-form expressions for the multivariate Beta will be constructed. The following is a brief explanation of how each method can be used to create a bivariate Beta distribution. In this dissertation these methods are extended to a multivariate Beta distribution. For clarity, the term multivariate Beta is in reference to the joint distribution and does not imply the simultaneous observation and analysis of many variables at once.

Libby and Novick[6] developed a closed form expression by transforming Gamma random variables. Specifically, Libby and Novick[6] let $Y_1 = \frac{X_1}{X_1+X_3}$ and $Y_2 = \frac{X_2}{X_2+X_3}$, where $X_i \sim Gamma(\alpha_i, \beta_i)$ for $i = 1, 2,$ and 3. By letting $\beta_i = \beta_j$ for all i, j marginal moments are easily calculated and have simple closed forms. For n-time points, $n+1$ parameters need to be estimated for an intercept-only model. However, there is no closed form for the correlation, and a structure cannot be imposed. The multivariate Beta based on the Libby and Novick[6] construction will be referred to as the Libby and Novick Multivariate Beta (LNMVB).

The general framework of the Sarmanov[7] bivariate distribution for (x_1, x_2) with specified marginal $f_{X_1}(x_1)$ and $f_{X_2}(x_2)$ is given by

$$f_{(X_1,X_2)}(x_1, x_2) = f_{X_1}(x_1) f_{X_2}(x_2)[1 + \omega\phi_1(x_1)\phi_2(x_2)]$$

where $\phi_i(t)$ is called the mixing function. The mixing function must be a bounded non-constant function such that

$$\int \phi_i(t) f_{T_i}(t_i) dt = 0$$

Additionally, ω determines the correlation, and the following condition must be satisfied

$$1 + \omega\phi_1(x_1)\phi_2(x_2) > 0.$$

Lee[8] proposed the mixing function $\phi_i(t) = t - \mu_t$ which leads to a bivariate Beta with marginal Beta distributions. The multivariate Beta based on this methodology will be referred to as the Sarmanov-Lee Multivariate Beta (SLMVB). The SLMVB has closed form moments and correlations. For n-time points, $2n + \sum_{i=2}^{n} \binom{n}{2}$ parameters must be estimated for an intercept-only model. Imposing a correlation structure can reduce the number of parameters that must be estimated. For example, a compound symmetry (CS) or auto-regressive(1) (AR(1)) structure requires $2n+1$ parameters to be estimated for an intercept-only model.

Aim1: Develop two closed form, n-time point, multivariate Beta models ($n > 2$).

a) Using the construction proposed by Libby and Novick[6] a multivariate Beta will be constructed.

b) Using the methodology of Sarmanov[7] and Lee[8] a multivariate Beta will be developed using Lee's[8] proposed extension to the Sarmanov family of bivariate distributions.

c) LNMVB (aim 1a) and SLMVB (aim 1b) will be re-parametrized using the method by Paolino[20] and Ferrari and Cribari-Neto.[21]

d) The constraint, $\beta_i = \beta_j$ for all i, j will be imposed on the LNMVB allowing for an unstructured correlation structure.

e) Under the SLMVB, the $\omega's$ will be constrained, such that the correlation structure is either CS or AR(1).

Aim 2: Establish the efficiency, Type I and Type II error rates for the models developed in Aim 1.

a) Multivariate Beta data will be simulated with CS and AR(1) correlation structures using an algorithm developed by Vorechovsky.[22]

b) Using the simulated data, bias for parameter estimates, root mean square deviation, power, type I error, and coverage probabilities will be calculated for the models developed in Aim 1.

Aim3: Compare the performance of the proposed model to current analytical options.

a) Each simulated dataset will be fit using the proposed multivariate Beta models, a Beta GEE and a Beta GLMM.

b) The performance of each model paradigm will be analyzed and compared by examining bias for parameter estimates, root mean square deviation, power, type I error, and coverage probabilities.

Aim 4: The proposed multivariate Beta models will be used to analyze clinical repeated-measures data from the field of neuropsychology.

1.2. Motivating Dataset

We present the dataset of a motivating cohort study that is analyzed in this dissertation. The National NeuroAIDS Tissue Consortium (NNTC) was established in 1998 to collect neuromedical, neuropsychological, and psychiatric data of patients (including men, women, and minorities) with the human immunodeficiency virus (HIV) and without HIV prior to death.[23] Additionally, ante- and post-mortem biological samples (i.e., blood, urine, and cerebrospinal fluid) were collected.[23] The consortium's goals include the establishment of a network of brain banks and other system tissues in a standardized fashion to support scientific studies of NeuroAIDS disorders.[23] The NNTC project is funded under the U24 grant mechanism from the National Institute of Mental Health and the National Institute of Neurological Disorders and Stroke.

Our analysis focused on neuropsychological performance measures. An approximately 2- to 3-hour battery of neuropsychological tests is used consortium-wide.[24] The neuropsychological measures were selected for their sensitivity to HIV associated impairments.[24] For participants too ill to complete the full battery of tests, the order of the tests is prioritized to ensure that a briefer battery consisting of representative tests from each domain is administered.[24] Additionally, tests were modified to accommodate participants with sensory limitations, e.g., blindness.[24] Raw test scores from each assessment were uploaded to the Data Coordinating Center for storage and processing.[24]

We analyzed the Hopkins Verbal Learning Test-Revised (HVLT-R) delayed recall scaled score of African American women participants. The HVLT-R contains 12 nouns, four words each from one of three semantic categories to be learned over the course of three learning trials.[26] Twenty to 25 minutes after completion of the three learning trials, a delayed recall trial and recognition recall trial are completed.[26] The delayed recall requires the free recall of any word remembered during the three learning trials.[26] To minimize practice effects that may arise in cohort studies, six alternate forms of the HVLT-R are utilized.[24] The HVLT-R delayed test scores were processed, converting raw scores to scaled scores, and then T-scores were calculated based on the scaled scores.[24] T-scores are demographically-correct scores based on existing test norms.[24] We chose to analyze scales scores[25] to allow for a possible demographic by visit occurrence interaction.

1.3. Literature Review

1.3.1. Introduction

In this section, an introduction to the Beta regression model, a review of the existing methods for constructing multivariate Beta distributions, applications of the techniques, proposed extensions of the methods are presented, and a brief review of the Beta Marginal Model and Beta GLMM. Two general methods can be used to create multivariate Beta distributions. A multivariate Beta distribution can be constructed using random variables with shared parameter(s) or by combining univariate Beta distributions with copulas. Section 1.3.2 will briefly describe the notation used in subsequent sections. Section 1.3.3 will describe the parameterization of the Beta regression for independent observations. Section 1.3.4 will focus on constructing multivariate Beta distributions through random variables with shared parameters while Section 1.3.5 will focus on multivariate Beta distributions using copulas. In both sections, models using the constructed bivariate Beta distribution will be highlighted along with the proposed multivariate extension, when applicable. In Section 1.3.6 and 1.3.7 the Beta Marginal Model and Beta GLMM, respectively, will be briefly discussed.

It should be noted that the Dirichlet density is a multivariate generalization of the Beta distribution.[27] However, the Dirichlet distribution is limited to the lower dimensional simplex, i.e., the random variables sum to 1. Therefore, it is not an appropriate distribution to model data where each repeated measure can take on values in the open unit interval. Therefore, no additional time will be spent exploring the Dirichlet distribution.

1.3.2. Notation

The following is a brief description of the notation that will be used henceforth. A capital letter will represent a random variable, and a realization of that random variable will be the lower case letter. Boldface random variables or realizations of the random variables will represent the respective vectors or matrices. A capital boldface $\boldsymbol{R}$ will be reserved for the set of real numbers. The probability density function (pdf) of X and cumulative density function (cdf) X will be represented by $f_X(x)$ and $F_X(x)$, respectively. If $\boldsymbol{A}$ is an $n \times n$ matrix with entries a_{ij} for $i, j = 1, \dots, n$, then $tr(A) = \sum_{i=1}^{n} a_{ii}$, i.e., the sum of the elements of the main diagonal. Lastly, if $x = a + bi$, where $a, b \in \boldsymbol{R}$ and i is an imaginary number, then $Re(x) = a$.

1.3.3. Beta regression for independent responses

For cross-sectional proportion data, a mean-precision parameterization Beta-regression model has been developed by Paolino[20] and Ferrari and Cribari-Neto.[21] The density of the Beta distribution is given by

$$f(y; \alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} y^{\alpha-1}(1-y)^{\beta-1}, \;\; 0 < y < 1 \qquad (1.1)$$

where $\alpha, \beta > 0$ and $\Gamma(\cdot)$ is the gamma function. The mean and variance of y are, respectively

$$E(y) = \frac{\alpha}{\alpha+\beta} \tag{1.2}$$

and

$$Var(y) = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}. \tag{1.3}$$

Paolino[20] and Ferrari and Cribari-Neta[21] proposed re-parameterizing (1.1) in terms of its mean and dispersion by letting $\mu = \frac{\alpha}{\alpha+\beta}$ and $\phi = \alpha + \beta$. This re-parameterization allows for an easier interpretation of the model parameters. It follows from equations (1.2) and (1.3) that

$$E(y) = \mu$$

and

$$Var(y) = \frac{V(\mu)}{1+\phi}$$

where $V(\mu) = \mu(1-\mu)$. Therefore the density in equation (1.1) can be expressed as

$$f(y; \mu, \phi) = \frac{\Gamma(\phi)}{\Gamma(\mu\phi)\Gamma((1-\mu)\phi)} y^{\mu\phi-1}(1-y)^{(1-\mu)\phi-1}, \qquad 0 < y < 1 \tag{1.4}$$

where $0 < \mu < 1$ and $\phi > 0$.

Let $y_1, \dots, y_n$ be independent random variables that follow the density in equation (1.4) with mean μ_i and unknown precision ϕ. Ferrari and Cribari-Neta[21] obtained a regression model using the framework of McCullagh and Nelder.[28] Specifically, Ferrari and Cribari-Neta[21] assumed that the mean of y_i could be written as

$$g(\mu_i) = \sum_{j=1}^{p} x_{ij}\beta_j = \eta_i$$

where $\beta = (\beta_1, \dots, \beta_p)^T$ is a vector of unknown regression parameters and $x_{i1}, \dots, x_{ip}$ are observations on p covariates ($p < n$), which are assumed fixed and known. Additionally, $g(\cdot)$ is strictly monotonic and twice differentiable link function that maps $(0,1)$ into $\boldsymbol{R}$. Ferrari and Cribari-Neta[21] used the logit link, which leads to β being interpreted as changes in the log odds of success. Ferrari and Cribari-Neta[21] treated ϕ as a nuisance parameter.

1.3.4. Multivariate Beta distribution through construction

The multivariate Beta can be constructed using a variable-in-common method or using matrices.[27,29] The focus of this dissertation will be on the multivariate Beta constructed using the variable-in-common technique. The multivariate Beta constructed through matrices is not suitable for repeated measures as explained below. It is, however, presented for completeness. The multivariate Beta constructed using matrices will be referred to as matrix-variate Beta.

The matrix-variate Beta has been defined and studied by many authors[30-34]; Gupta[29] attempts to clarify the definitions. The random variable U with pdf given in equation (1.1) where $a > 0$ and $b > 0$, is said to have a Beta type I distribution with parameters (a, b).[35] The random variable V with pdf,

$$f_V(v) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} v^{a-1}(1+v)^{-(a+b)}, \quad v > 0 \tag{1.5}$$

where $a > 0$ and $b > 0$, is said to have a Beta type II distribution with parameters (a, b).[35] It can be shown that (1.5) can be obtained from (1.1) using the transformation $V = U/(1-U)$. Therefore, equation (1.5) is referred to as the inverted Beta distribution by some authors.[35] The matrix-variate generalizations of (1.1) and (1.5) are referred to as the matrix-variate Beta type I and matrix-variate Beta type II distributions, respectively.[29,35]

Let A and B be independent $p \times p$ symmetric positive definite matrices having a Wishart density $W_p(q, \Sigma)$ and $W_p(n, \Sigma)$, respectively, where $n = N - (q+1)$. Then the matrix $U = (A+B)^{-\frac{1}{2}} A (A+B)^{-\frac{1}{2}}$ and $V = AB^{-1}$ are distributed as matrix-variate Beta type I and matrix-variate Beta type II, respectively, with parameters $\frac{q}{2}$ and $\frac{n}{2}$.[36] The matrix-variate Beta type I is denoted as $U \sim B_p^I\left(\frac{q}{2}, \frac{n}{2}\right)$, if its pdf is given by

$$\left\{\beta_p\left(\frac{q}{2}, \frac{n}{2}\right)\right\}^{-1} \det(U)^{q-(p+1)/2} \det\left(I_p - U\right)^{n-(p+1)/2}, \quad 0 < U < I_p$$

where $q > p-1$, $n > p-1$, and $\beta_p\left(\frac{q}{2}, \frac{n}{2}\right)$ is the multivariate Beta function given by

$$\beta_p(a, b) = \int_0^{I_p} \det(A)^{a-\frac{1}{2}(p+1)} \det\left(I_p - A\right)^{b-\frac{1}{2}(p+1)} dA$$

with $Re(a) > \frac{1}{2}(p-1)$ and $Re(b) > \frac{1}{2}(p-1)$.[29,36] Similarly, the matrix-variate Beta type II is denoted as $V \sim B_p^{II}\left(\frac{q}{2}, \frac{n}{2}\right)$, if its pdf is given by

$$\left\{\beta_p\left(\frac{q}{2}, \frac{n}{2}\right)\right\}^{-1} \det(V)^{q-(p+1)/2} \det\left(I_p + V\right)^{-(q+n)/2}, \quad V > 0$$

where $q > p-1$ and $n > p-1$.[36] As in the univariate case, the matrix-variate Beta type II can be obtained by transforming the matrix-variate Beta type I, i.e., $U = \left(I_p + V\right)^{-1} V$.[35]

The distributions of trU and trV play an important role in hypothesis testing when using a multivariate linear model. Specifically, trU and trV appear as the null distribution in a one-way MANOVA model for testing whether all means are equal.[36] However the matrix-variate Beta distributions apply only to symmetric matrices, if the interest is in a vector of responses, i.e., repeated measures, the matrix-variate Betas are inappropriate.[6] Therefore, a multivariate Beta is needed for repeated measures.

Using shared parameters, the multivariate Beta can be constructed from Gamma random variables, Beta random variables, Dirichlet random variables, or using order statistics from the Uniform distribution.[27] Libby and Novick[6] were the first to construct a multivariate Beta using Gamma random variables with shared parameters.

Let $X_0, \dots, X_n$ be distributed as independent Gamma random variables with parameters α_i and β_i, $i = 0, \dots, n$. Using the transformation $Y_0 = X_0$ and $Y_i = \frac{X_i}{X_i + X_0}$ for $i = 1, \dots, n$, Libby and Novick[6] derive the joint density of $y_1, \dots, y_n$ as

$$P(y_1, \dots, y_n) = \frac{\Gamma(\sum_{i=0}^{n} \alpha_i)}{\prod_{i=0}^{n} \Gamma(\alpha_i)} \frac{\prod_{i=1}^{n} \left[\lambda_i^{\alpha_i} \left(\frac{y_i}{1-y_i} \right)^{\alpha_i - 1} \left(\frac{1}{1-y_i} \right)^2 \right]}{\left[1 + \sum_{i=1}^{n} \lambda_i \left(\frac{y_i}{1-y_i} \right) \right]^{\sum_{i=0}^{n} \alpha_i}}, \quad 0 < y_i < 1 \qquad (1.6)$$

where $\lambda_i = \frac{\beta_i}{\beta_0}$, $0 < y < 1$, and $\Gamma(\cdot)$ is the gamma function. Libby and Novick[6] describe the density in equation (1.6) as a generalized multivariate Beta of the first kind. Any marginal multivariate distribution of equation (1.6) will still be a multivariate generalized Beta of the first kind.[6] The univariate marginal distribution of equation (1.6) is given by

$$P(y_i) = \frac{\Gamma(\alpha_0 + \alpha_i)}{\Gamma(\alpha_0)\Gamma(\alpha)} \frac{\lambda_i^{\alpha_i} \left(\frac{y_i}{1-y_i} \right)^{\alpha_i - 1} \left(\frac{1}{1-y_i} \right)^2}{\left[1 + \lambda_i \left(\frac{y_i}{1-y_i} \right) \right]^{\alpha_0 + \alpha_i}}, \quad 0 < y_i < 1 \qquad (1.7)$$

which Libby and Novick[6] refer to as a generalized Beta distribution with scale parameter λ_i. Furthermore, it should be clear that density (1.7) is the univariate form of density (1.6). Libby and Novick[6] justify the naming convention by re-expressing the density in equation (1.1) as

$$P(y) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \frac{\left(\frac{y}{1-y} \right)^{\alpha - 1} \left(\frac{1}{1-y} \right)^2}{\left[1 + \left(\frac{y}{1-y} \right) \right]^{\alpha + \beta}}, \quad 0 < y < 1$$

and noting a type I Beta is a generalized Beta with parameters α, β, and $\lambda = 1$.

Both Jones[37] and Olkin and Liu[38] obtained the density in equation (1.6) independently. Jones[37] obtains the density (1.6) starting from a multivariate F distribution. While Olkin and Liu[38] obtain density (1.6) using similar construction schemes to that of Libby and Novick.[6] It should be noted that the construction of the bivariate case of equation (1.6) with $\lambda = 1$ is often credited to Olkin and Liu.[38] For clarity, the bivariate case of equation (1.6) with $\lambda = 1$ will be referred to as the Olkin and Liu[38] bivariate Beta.

The tth moment of the generalized Beta distribution equation (1.7), i.e., the univariate marginal of the generalized multivariate Beta is

$$E(y^t) = \sum_{k=t}^{\infty} \frac{\Gamma(\alpha+\beta)\Gamma(\alpha+k)}{\Gamma(\alpha+\beta+k)\Gamma(\alpha)} \frac{(\lambda+1)^{k-t}}{\lambda^k}.^6$$

As previously stated, when $\lambda = 1$ the generalized Beta is the standard Beta. The moments of the standard Beta have a closed form and do not require an iterative method to calculate the numerical value of the moment.[6]

Jones[37], Olkin and Liu[38], and Nagar and colleagues[39] calculated the expected correlation for the Olkin and Liu[38] bivariate Beta. The expected correlation requires the evaluation of a generalized Gauss hypergeometric function, which has no closed form.[38,40] Through construction, it is clear that the random variables of the Olkin and Liu[38] bivariate Beta have a positive correlation in $[0,1]$.[27] However, using simulations, Gianola and collegues[40] have demonstrated the inadequacy of the Pearson[14] correlation statistic in measuring association in random variables generated from the Olkin and Liu[38] bivariate Beta. Furthermore, Jones[37], Olkin and Liu[38], and Nagar et al. [39] expressions for the expected correlation of the Olkin and Liu[38] bivariate Beta are not in agreement.

Specifically, the parameters of the generalized Gauss hypergeometric function do not match.

Regardless of these limitations, multiple authors have used Libby and Novick's[6] construction or variations to the construction to model bivariate correlated data bounded on the interval $[0,1]$. Libby and Novick[6] present an example of fitting utilities with their generalized Beta distribution. However, the model was limited to the bivariate case and parameters had to be estimated using a Monte-Carlo iterative procedure due to the complexity of the first and second partial derivatives of the joint pdf. Other researchers[10,11,41-43] have modified the Olkin and Liu[38] bivariate Beta and have fit their proposed model to various datasets. Adell and collegues[10] fit a zero-inflated bivariate Beta for retinal image id in lambs. Adell et al[10] re-parameterized the Olkin and Liu[38] bivariate Beta using the parameterization proposed by Paolino[20] and Ferrari and Cribari-Neta.[21] Nadarajah[11,43] added additional parameters to the Olkin and Liu[38] bivariate Beta creating two additional distributions and fit the model[11] to drought data. However, the normalizing constant of the joint distribution of one of Nadarajah's[11] extension requires the evaluation of the Gauss hypergeometric function.[11] And the other extension by Nadarajah[43] is constrained to the lower dimensional simplex. Arnold and Ng[41] constructed bivariate Beta distributions by using additional Gamma random variables in the construction. Specifically, Arnold and Ng[41] used five independent Gamma random variables for construction, where three of the Gamma random variables had shared parameters. The model was evaluated using a simulation study and an eight-parameter construction for the bivariate case was proposed.[41] The Olkin and Liu[38] bivariate Beta is a particular case of the five parameter construction proposed by Arnold and Ng.[41] Arnold and Ng[41] five parameter construction allows for negative correlations, but the joint density does not have a closed form. Arnold and Ng[41] proposed a modified maximum likelihood to fit their model, while Crackel and Flegal[44] fit the model under a Bayesian framework. Lastly, Gupta[42] used the construction proposed by Libby and Novick[6] using non-central Gamma random variables limited to the bivariate case. Again, the joint pdf proposed by Gupta[42] does not have a closed form due to the inclusion of the Gauss hypergeometric function.

Using shared parameters, researchers[27,45] have proposed alternatives to using Gamma random variables for the construction of a multivariate Beta distribution. Nadarajah and Kotz[45] created three different bivariate Beta distributions starting from independent Beta random variables. However, two of the bivariate distributions are limited to the lower dimensional simplex, and the third does not have a closed form.[45] Alternatively, Olkin and Trikalinos[27] construct a bivariate Beta distribution using three independent Dirichlet random variables that allows for correlation over the range $[-1,1]$. Unfortunately, the joint pdf does not have a closed form.[27] However, they were able to estimate the parameters using methods of moments. Olkin and Trikalinos[27] also provide a construction for a bivariate Beta distribution using order statistics from a Uniform distribution on $[0,1]$; they have not followed this line of inquiry, but note it may lead to some novel results.

1.3.5. Multivariate Beta distribution using copulas

The structure of dependence between *n*-related outcomes can be defined in terms of their joint (i.e., multivariate) distribution.[9,46] Additionally, the joint distribution uniquely defines all lower dimensional marginal distributions and conditional distributions.[46] One possible way to obtain the joint distribution of known marginal distributions is through the use of copulas.[9] A copula is a function which joins univariate marginal distributions to form a multivariate distribution.[47] Sklar[48] was the first to use the terminology copula in the theorem that bears his name; however, the use of copula functions predates the use of the term.[47]

The mapping $C\colon [0,1]^n \rightarrow [0,1]$ is called a copula according to Nelson[49] if

(i) for every $\boldsymbol{u} \in [0,1]^n$, $C(\boldsymbol{u}) = 0$ if at least one coordinate of $\boldsymbol{u}$ is 0, and $C(\boldsymbol{u}) = u_i$ if all coordinates of $\boldsymbol{u}$ are 1 except u_i.

(ii) C is n-increasing, i.e. for every $\boldsymbol{a}, \boldsymbol{b} \in [0,1]^n$ such that $a_i \leq b_i$ for every i, and $V_C([\boldsymbol{a}, \boldsymbol{b}]) \geq 0$, where $[\boldsymbol{a}, \boldsymbol{b}] = [a_1, b_1] \times [a_2, b_2] \times \ldots \times [a_n, b_n]$ and $V_C([\boldsymbol{a}, \boldsymbol{b}]) = \sum(\boldsymbol{c}) C(\boldsymbol{c})$. The sum is over the vertices $\boldsymbol{c}$ of $[\boldsymbol{a}, \boldsymbol{b}]$ and $sgn(\boldsymbol{c}) = 1$ if $c_i = a_i$ for an even number of $i's$ and -1 if $c_i = a_i$ for an odd number of $i's$.

Sklar's[48] theorem can now be stated.

Sklar's[48] theorem states that a joint distribution can be expressed using its univariate marginal distributions and multivariate dependence structure. The multivariate dependence structure is referred to as a copula.[46,47,50] Specifically, Sklar's[48] theorem states if we assume a n-dimensional random vector $\boldsymbol{X}$ with marginal cumulative distribution functions $F_{X_1}, \ldots, F_{X_n}$ with domain $\boldsymbol{R}^n$ then the joint distribution $F_{\boldsymbol{X}}$ can be written as a function of its marginal distributions,

$$F_{\boldsymbol{X}} = C_{\boldsymbol{X}}\left(F_{X_1}(x_1), \ldots, F_{X_n}(x_n)\right)$$

where $C_{\boldsymbol{X}}$ is a copula as defined above. If the marginal distributions are continuous, then the copula function will be unique.[46,49,50] Copula construction is not constrained to continuous distributions.[46,49] Additionally, the marginal distributions are not required to be a common distribution (e.g., the marginal distributions could be a combination of Gaussian and Gamma distributions).[50] An important consequence of Sklar's[48] theorem is that every joint distribution can be decomposed as a product of its marginal densities and its copula density[50,51], i.e.

$$f_{\boldsymbol{X}}(\boldsymbol{x}) = f_{X_1}(x_1) \dots f_{X_n}(x_n) \cdot c_{\boldsymbol{X}}(u_1, \dots, u_n).$$

Unfortunately, there is no general or canonical way to formulate the copula and determine the associations amongst dependent outcomes.[50] However, the parametric form of copulas can be grouped into families.[50] Some significant copula families in statistical modeling are the elliptical, Archimedean, and Farlie-Gumbel-Morgenstern (FGM) copulas family.[47,50] The Sarmanov[7] family of bivariate distributions is another family of copulas that until recently has been largely ignored.[52] The focus of this dissertation will be on the extension of the Sarmanov[7] family of bivariate distributions; however, the elliptical, Archimedean, and FGM family of copulas will be briefly reviewed for comparison.

Elliptical copulas arise from elliptical distributions, e.g., Gaussian, Student-t etc.[50] Elliptical copulas can be extended to an arbitrary number of dimensions; however, a n-dimensional elliptical copula would require a minimum of $\frac{n(n-1)}{2}$ parameters.[50] An additional drawback to the elliptical copulas is that the dependence is restricted to radial symmetry and they do not necessarily exist in closed form.[50]

Archimedean copulas allow for a more flexible dependence structure, i.e., different upper and lower tail behavior and Archimedean copulas often have a closed form.[50] However, marginal distributions are exchangeable using Archimedean copulas, which is usually not practical for dimensions greater than two.[50] A mapping $F: \boldsymbol{R}^n \to \boldsymbol{R}$ is called exchangeable, if

$$F(x_1, \dots, x_n) = F\left(x_{\pi(1)}, \dots, x_{\pi(n)}\right)$$

holds for every $\boldsymbol{x} \in \boldsymbol{R}^n$ and all permutations $\pi \in S_n$, where S_n is a permutation of $\{1, \ldots, n\}$.[53] This limitation does not exclude Archimedean copulas from being used in higher dimensional cases; however, Archimedean copulas are most often applied in the bivariate case.[50] Additionally, the dependence is often governed by one parameter.[49] The Archimedean copula can be expressed as

$$C_{\boldsymbol{X}}(u_1, \ldots, u_n) = \psi^{-1}\big(\psi(u_1) + \cdots + \psi(u_n)\big)$$

where ψ is the generator function.[50] Archimedean copulas are often described in terms of their generator function, e.g., Clayton, Frank, Gumbel being some of the most commonly used.[50]

The FGM family has been studied extensively for bivariate model building.[52] The bivariate FGM family is given by

$$f_{(X_1,X_2)}(x_1, x_2) = f_{X_1}(x_1) f_{X_2}(x_2)\{1 + \alpha[1 - 2F_{X_1}(x_1)][1 - 2F_{X_2}(x_2)]\}. \qquad (1.8)$$

where $|\alpha| \leq 1$.[52] The FGM family is a fairly straightforward way to introduce dependence; however, the correlation coefficients are limited to the interval $\left(-\frac{1}{3}.\frac{1}{3}\right)$.[8] Additionally, the marginal distributions generally do not match the univariate distributions that were used to construct the joint distribution.[52] That is $\int f_{(X_1,X_2)}(x_1, x_2)\, dx_1 \neq f_{X_2}(x_2)$; however, the bivariate exponential distribution does produce exponential marginal distributions.[52] It is clear from equation (1.8) that a bivariate Beta would not produce Beta marginal distributions.

The Sarmanov[7] family of bivariate distributions is one such family where the marginal distributions match the univariate distributions used in construction.[8] The Sarmanov[7] bivariate copula remained relatively unnoticed until Lee[8] published a paper in 1996 that focused on the bivariate Beta distribution using the Sarmanov[7] family of bivariate distributions.[52] It should be noted that Danaher[54] was able to obtain a bivariate

Beta-Binomial distribution using canonical expansion such that the bivariate Beta matches that of Lee.[8]

The Sarmanov[7] family of bivariate distributions is defined in Section 1.1.2 along with Lee's[8] proposed mixing function. Lee[8] proved that the range of the correlation coefficients for the Sarmanov[7] family of bivariate distributions is a function of the marginal distributions and mixing function. Specifically, the correlation is bounded by

$$|\rho| \leq |\omega|\sqrt{E[\phi_1^2(X_1)]E[\phi_2^2(X_2)]}.$$ [8]

Shubina and Lee[55] calculated the upper and lower correlation bounds for equal Beta marginal distributions, i.e., $X_i \sim Beta(a,b)$ for $i = 1$ and 2, as

$$upper = \sigma^2 \max_{p\in[0,1]} \frac{f_X^2\left(F_X^{-1}(p;a,b);a+1,b+1\right)}{p(1-p)}$$

and

$$lower = -\sigma^2 \max_{p\in\left[0,\frac{1}{2}\right]} \frac{f_X\left(F_X^{-1}(p;a,b);a+1,b+1\right)f_X\left(F_X^{-1}(1-p;a,b);a+1,b+1\right)}{p(1-p)}$$

where f_X is the pdf of the Beta distribution with parameters a, b and F_X^{-1} is the inverse to the Beta cdf with parameters a, b. Furthermore, Shubina and Lee[55] showed for symmetric equal Beta marginal distributions (i.e. $X_i \sim Beta(a,a)$ for $i = 1,2$) as $a \rightarrow \infty$, $X \underset{d}{\rightarrow} N\left(\frac{1}{2}, \frac{1}{8a+4}\right)$ and thus the correlation range tends to $\pm\frac{2}{\pi}$. Alternatively, for symmetric equal Beta marginal distributions, as $a \rightarrow 0$ the correlation range tends to ± 1.[55]

Lee[8] further extended the Sarmanov[7] family of bivariate distributions to the multivariate case. Assume that $f_{X_i}(x_i)$ for $i = 1, \dots, n$ are univariate pdfs with supports defined on $A_i \subseteq \boldsymbol{R}$ for $i = 1, \dots, n$ and let $\phi_i(t)$, $i = 1, \dots, n$ be a set of bounded nonconstant functions such that $\int_{-\infty}^{\infty} \phi_i(t) f_{T_i}(t_i)dt = 0$ for all $1 \leq i \leq n$. Then, the function

$$f_{\boldsymbol{X}}(\boldsymbol{x}) = \{\prod_{i=1}^{n} f_{X_i}(x_i)\}\{1 + R(x_1, \dots, x_n; \phi_1, \dots, \phi_n, \Omega_n)\}$$

is a multivariate joint density with specified marginal distributions $f_{X_i}(x_i), i = 1, \dots, n$, where

$$R(x_1, \dots, x_n; \phi_1, \dots, \phi_n, \Omega_n) = \sum_{j_1}^{n-1} \sum_{j_2}^{n} \omega_{j_1,j_2} \phi_{j_1}(x_{j_1}) \phi_{j_2}(x_{j_2}) + \sum_{j_1}^{n-2} \sum_{j_2}^{n-1} \sum_{j_3}^{n} \omega_{j_1,j_2,j_3} \phi_{j_1}(x_{j_1}) \phi_{j_2}(x_{j_2}) \phi_{j_3}(x_{j_3}) + \cdots + \omega_{1,2,\dots,n} \prod_{i=1}^{n} \phi_i(x_i) \tag{1.9}$$

and $\Omega_n = \{\omega_{j_1,j_2}, \omega_{j_1,j_2,j_3}, \dots, \omega_{1,2,\dots,n}\}$. The set of real numbers Ω_n is chosen such that $1 + R(x_1, \dots, x_n; \phi_1, \dots, \phi_n, \Omega_n) \geq 0$ holds for all $x_i \in \boldsymbol{R}$, $i = 1, \dots, n$.[8] It is clear from equation (1.9) that higher order effects are included in the joint distribution. Prentice[56] has demonstrated for a multivariate Beta-binomial model using a canonical construction that the higher effects are required for the model to be sufficiently flexible.

Lee's[8] 1996 paper help rediscover the Sarmanov[7] family of bivariate distributions. However, there appear to be limited applications of the Sarmanov[7] family of bivariate distributions in the literature, particularly when the marginal distributions are Beta distributions. Chen and colleagues[57] used the Sarmanov[7] Lee[8] bivariate Beta as a prior in a Bayesian meta-analysis of adverse events in clinical trials. Danaher and Hardie[58] fit a Sarmanov[7] bivariate Beta-binomial model to two different data sets, purchasing bacon and eggs and purchasing two magazine subscriptions. Additionally, Danaher and Hardie[58] described the relationship between the Sarmanov[7] family of bivariate distributions and the canonical expansion model. The canonical expansion model with marginal Beta-binomial distributions was used to model media exposure.[54] Shoukri and colleagues[59] used the Sarmanov[7] family of bivariate distributions with Beta-binomial marginal distributions to model high blood pressure among family members. Furthermore, Shoukri and colleagues[60] developed hypotheses tests for the Sarmanov[7] family of bivariate distributions with Beta-binomial marginal distributions. Lastly, Gianola and colleagues[40] proposed a new measure of association based on logarithmic (Kullback-Leibler) and relative distances between distributions and compared their measure to Pearson's[14] correlation coefficient with different joint distributions, in particular, the Sarmanov[7] Lee[8] bivariate Beta. Additionally, there is no literature beyond Lee's[8] own description of the Sarmanov[7] family of bivariate distributions being extended to *n*-dimensions.

1.3.6. The Beta Marginal Model

Section 1.2.4 and Section 1.2.5 described two different multivariate approaches for handling the correlation among sampling units; however, two other modeling approaches are often utilized, i.e., marginal models and mixed effects models.[1] For responses that are assumed to follow the Beta distribution, the choice of a marginal model or a mixed effects model leads to different interpretations of the regression parameters.[1] A marginal model has a population-average interpretation of the parameter estimates while the mixed effects models have a subject-specific interpretation of the regression coefficients.[1] These different interpretations are the result of the assumptions about the source of within-subject associations.[1] The following is a brief description of the marginal model and the method of GEE for parameter estimates.

We begin our discussion of the marginal model by introducing notation that will be used for both the marginal model and the mixed effects model. We first assume that there are N subjects measured repeatedly. Let Y_{ij} denote the response for the j^{th} measurement on the i^{th} subject. Furthermore, we assume that there are n_i repeated measures for the i^{th} subject. Therefore, the responses for the i^{th} subject can be grouped into an $n_i \times 1$ vector, i.e., $\boldsymbol{Y}_i = \left(Y_{i1}, \dots, Y_{in_i}\right)'$ for $i = 1, \dots, N$. We assume that the vector of responses, $\boldsymbol{Y}_i$, are independent of one another but the repeated measures on the same subject are correlated. Associated with each response Y_{ij} is a known $p \times 1$ vector of explanatory covariates, i.e., $\boldsymbol{X}_{ij} = \left(X_{ij}\ , \dots, X_{ijp}\right)'$ for $i = 1, \dots, N; j = 1, \dots, n_i$. We can group the vectors of covariates into an $n_i \times p$ matrix of covariates

$$\boldsymbol{X}_i = \begin{pmatrix} \boldsymbol{X}_{i1}' \\ \vdots \\ \boldsymbol{X}_{in_i}' \end{pmatrix} = \begin{pmatrix} X_{i11} & \dots & X_{i1p} \\ \vdots & \ddots & \vdots \\ X_{in_i1} & \dots & X_{in_ip} \end{pmatrix}, i = 1, \dots, N.$$

Lastly, $\boldsymbol{\beta} = \left(\beta_1, \dots, \beta_p\right)'$ is a $p \times 1$ vector of unknown parameters.

Marginal models do not specify the full joint distribution of the data.[61] Marginal models separately model the mean response and the within-subject associations among responses.[1,3,61] In marginal models, the goal is to make inferences about the conditional mean.[1] The within-subject associations are treated as a nuisance parameter(s) that must be estimated to make correct inferences about changes in the population mean response.[1,3] Marginal models have a three-part specification:

1. The conditional expectation of each response, i.e., $E\left(Y_{ij} \middle| X_{ij}\right) = \mu_{ij}$, is assumed to depend on a vector of explanatory covariates via a known link function

   $g\left(\mu_{ij}\right) = \eta_{ij} = X_{ij}'\beta.$

2. The conditional variance of each response given the covariates is assumed to depend on the mean according to

   $var\left(Y_{ij} \middle| X_{ij}\right) = \phi v(\mu_{ij}),$

   where $v(\mu_{ij})$ is a variance function, and ϕ is a scale parameter.

3. The conditional within-subject associations given the covariates are assumed to be a function of additional parameters, α, which also depends on the means.[1]

Therefore, given specifications (2) and (3) the corresponding covariance matrix can be constructed as

$$\boldsymbol{V}_i = \boldsymbol{A}_i^{\frac{1}{2}} Corr(\boldsymbol{Y}_i) \boldsymbol{A}_i^{\frac{1}{2}},$$

where $\boldsymbol{A}_i$ is a diagonal matrix with $\phi v(\mu_{ij})$ along the diagonal and $Corr(\boldsymbol{Y}_i)$ is a correlation matrix which is a function of the $\alpha' s$.[1] If the conditional mean is correctly specified, the method of GEE[2] yields a consistent estimator $\hat{\beta}$ of β by solving the score equation $\sum_{i=1}^{N} \boldsymbol{D}_i' \boldsymbol{V}_i^{-1}(\boldsymbol{Y}_i - \boldsymbol{\mu}_i) = 0$ where $\boldsymbol{D}_i = \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}$.[1,3,61] Under the GEE methodology, $\boldsymbol{V}_i$ is often referred to as the working covariance matrix.[1,61] Specifically, $\boldsymbol{V}_i$ approximates the true underlying covariance matrix for $\boldsymbol{Y}_i$; however, $\boldsymbol{V}_i = Cov(\boldsymbol{Y}_i)$ if the variance and within-subject associations are correctly specified.[1]

The score equations have no closed form solution; therefore, an iterative algorithm is required.[1,61] The GEE methodology uses the following iterative two-stage estimation algorithm:

1. Given the current estimates of ϕ, α, and $\boldsymbol{V}_i$ an updated estimate of β is obtained as a solution to the above-defined score equation.

2. Given the current estimate of β, updated estimates of ϕ and α are calculated using standardized residuals.[1,2]

The two-stage procedure iterates between steps 1 and 2 until a convergence criterion is reached.[1,2] Once convergence is achieved, $\hat{\beta}$ is a consistent estimator of β and with large samples the estimator of the

$$cov(\hat{\beta}) = \left(\sum \widehat{D_\iota}'\widehat{V_\iota}^{-1}\widehat{D_\iota}\right)^{-1}\left\{\sum \widehat{D_\iota}'\widehat{V_\iota}^{-1}(Y_i - \widehat{\mu_\iota})(Y_i - \widehat{\mu_\iota})'\widehat{V_\iota}^{-1}\widehat{D_\iota}\right\}\left(\sum \widehat{D_\iota}'\widehat{V_\iota}^{-1}\widehat{D_\iota}\right)^{-1}$$

yields correct standard errors.[1] However, limitations of this method have been shown[16,18], and we refer the reader back to Section 1.1.1 for a description of the limitations.

For responses that follow the Beta distribution, it is convenient to assume the logit, $\log\left(\frac{\mu_{ij}}{1-\mu_{ij}}\right)$, for the link function $g(\cdot)$; however, other link functions can be considered.[61] The variance function is often specified as $var(Y_{ij}|X_{ij}) = \phi\mu_{ij}(1-\mu_{ij})$.[61] For repeated measures data, CS, auto-regressive, or unstructured associations are most commonly considered.[1,61] It should be emphasized, that no distributional assumptions are required for the GEE approach.[1] However, a distribution function from the exponential family usually suggests the form of the conditional mean and conditional variance of $\boldsymbol{Y}_i$.[3]

1.3.7. The Beta Generalized Linear Mixed Model

The marginal model does not make explicit the source of within-subject association in the observed data.[3] Marginal models do not require the joint distribution to be fully specified; it was sufficient to define the marginal means, variances, and pairwise associations for estimation and prediction using the GEE approach.[1] Separately specifying the marginal means and covariance ensure that the prediction for the marginal means does not rely on the assumed model for the covariance.[1]

An alternative approach for accounting for the within-subject associations is inducing correlation through an unobserved heterogeneity, i.e., random effects, in the conditional mean specification.[1,3] GLMMs is a family of models that incorporates random effects into the conditional mean. GLMMs allow a subset of regression coefficients to vary randomly from one individual to another according to some distribution.[1] The random effects can be thought of as accounting for the heterogeneity among individuals due to unmeasured variables.[1] In general, random effects are assumed multivariate normal for mathematical and computational convenience; however, alternative distributions are possible.[1,61] In a repeated-measures design, the random effects are most commonly scalar (i.e., random intercept) or a bivariate vector (i.e., random intercept and random slope).[61] GLMMs assume that responses for any particular individual are conditionally independent observations from a distribution belonging to the exponential family.[1] Specifically, the observations are independent given the random effects.[1]

GLMM can be formulated using a three-part specification:

1. The conditional distribution of each Y_{ij} given a $q \times 1$ vector of random effects, b_i, belong to the exponential family of distributions. Additionally, $Var(Y_{ij}|b_i) = v\{E(Y_{ij}|b_i)\}\phi$, where $v(\cdot)$ is a known variance function of the conditional mean, $E(Y_{ij}|b_i)$ and the Y_{ij}'s are conditionally independent given the random effects.

2. The conditional mean is assumed to depend on fixed and random effects via the linear predictor

   $\eta_{ij} = X'_{ij}\beta + Z'_{ij}b_i,$

   with

   $g\{E(Y_{ij}|b_i)\} = \eta_{ij} = X'_{ij}\beta + Z'_{ij}b_i$

   for some link function, $g(\cdot)$.

3. The random effects are assumed to have some probability distribution, h. Additionally, the random effects, b_i, are assumed to be independent of the covariates, X_i.[1]

Therefore, the conditional likelihood of $\boldsymbol{Y}_i$ given $\boldsymbol{X}_i$ can be expressed in the form $l_i = \int f(\boldsymbol{Y}_i|\boldsymbol{X}_i, b_i = b)h(b)db$ where $f(\boldsymbol{Y}_i|\boldsymbol{X}_i, b_i = b) = \prod_{j=1}^{n_i} f(Y_{ij}|\boldsymbol{x}_{ij}, b_i = b)$.[3] Assuming that h is a multivariate normal density, some technique (e.g. adaptive Gaussian quadrature, quasi-likelihood, etc.) must be employed before evaluating l_i.[1,3,61]

Adding random effects to equation (1.4) yields the Beta GLMM given by

$$\log\left(\frac{\mu_{ij}}{1-\mu_{ij}}\right) = x'_{ij}\beta + z'_{ij}b_i, \text{ with } b_i \sim N(0, G)$$

where G denotes the positive definitie covariance matrix of the random effects.[61] In the Beta GLMM, the regression parameters have only a subject-specific interpretation because of the non-linear link function.[1,61] Specifically,

$$logit\left(E\left(Y_{ij}\middle|b_i\right)\right) = x'_{ij}\beta + z'_{ij}b_i,$$

but

$$logit\left(E\left(Y_{ij}\right)\right) \neq x'_{ij}\beta.$$[61]

The subject-specific interpretation can be regarded as the mean difference in outcome on the logit scale between an individual with said covariate and the same individual supposed not to have said covariate.[1,61]

It should be clear that there are multiple methods that can be used to analyze correlated proportional data. Each method has its limitations. In the next section, we propose two multivariate Beta densities that can be fit using the maximum likelihood method, thereby overcoming the limitations of the Beta GEE and Beta GLMM, in this regard.

2. Methodological Contributions

2.1. Introduction

Sections 1.2.4 and 1.2.5 briefly described the development of multivariate Beta densities through construction and copulas, respectively. Until now, methodology for fitting multivariate Beta densities (as described in Sections 1.2.4 and 1.2.5) was limited to the bivariate case. This chapter provides the methodology for constructing and fitting multivariate Beta densities to n-repeated measures. Specifically, in Section 2.3 we used Libby and Novick's[6] technique to construct a multivariate Beta density. Furthermore, we derived the score equations, the Hessian matrix, and expected pairwise correlation for the LNMVB. Similarly, in Section 2.4 we used Lee's[8] proposed multivariate Beta and derived the score equations, the Hessian matrix and the correlation structure. Prior to the development of the multivariate Beta densities in Sections 2.3 and 2.4, we describe the notation used throughout the chapter in Section 2.2.

2.2. Notation

We first assumed that there are N subjects measured repeatedly. Let Y_{ij} denote the response for the j^{th} measurement on the i^{th} subject. Additionally, $Y_{ij} \in (0,1)$ for all i, j. We assumed a balanced design, i.e., there are n repeated measures for every subject. Therefore, the responses for the i^{th} subject were grouped into an $n \times 1$ vector, i.e., $\boldsymbol{Y}_i = (Y_{i1}, \ldots, Y_{in})'$ for $i = 1, \ldots, N$. We assumed that the vector of responses, $\boldsymbol{Y}_i$, are independent of one another but the repeated measures on the same subject are correlated. Associated with each response Y_{ij} is a known $p \times 1$ vector of explanatory covariates, i.e., $\boldsymbol{X}_{ij} = \left(X_{ij1}, \dots, X_{ijp}\right)'$ for $i = 1, \dots, N; j = 1, \dots, n$. We grouped the vectors of covariates into an $n \times p$ matrix of covariates

$$\boldsymbol{X}_i = \begin{pmatrix} \boldsymbol{X}_{i1}' \\ \vdots \\ \boldsymbol{X}_{in}' \end{pmatrix} = \begin{pmatrix} X_{i11} & \dots & X_{i1p} \\ \vdots & \ddots & \vdots \\ X_{in1} & \dots & X_{inp} \end{pmatrix}, i = 1, \dots, N.$$

Lastly, $\boldsymbol{\beta} = \left(\beta_1, \dots, \beta_p\right)'$ is a $p \times 1$ vector of unknown regression parameters.

2.3. Libby and Novick Multivariate Beta (LNMVB)

The construction of the LNMVB began by letting $X_0, X_1, \dots, X_n$ be distributed as independent gamma random variables with parameters α_i and β_i, for $i = 0, \dots, n$. The joint pdf of $X_0, \dots, X_n$ is given by

$$f_{X_0,\dots,X_n}(x_0, \dots, x_n) = \prod_{i=0}^{n} \left[\frac{\beta_i^{\alpha_i}}{\Gamma(\alpha_i)} x_i^{\alpha_i - 1} e^{-\beta_i x_i}\right], \tag{2.1}$$

$\alpha_i,\ \beta_i,\ x_i > 0$ for $i = 0, \dots, n$.

By transforming the variables in the joint pdf (2.1) and marginalizing out Y_0 we arrived at the joint pdf described by Libby and Novick.[6] Specifically, let $Y_0 = X_0$ and $Y_i = \frac{X_i}{X_0 + X_i}$ for $i = 1, \dots, n$. Then the joint pdf of $Y_0, \dots, Y_n$ is given by

$$f_{Y_0,\dots,Y_n}(y_0, \dots, y_n) = \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} y_0^{\alpha_0 - 1} e^{-y_0 \beta_0} y_0^n \prod_{i=1}^{n} \left[\frac{\beta_i^{\alpha_i}}{\Gamma(\alpha_i)} \left(\frac{y_0 y_i}{1 - y_i}\right)^{\alpha_i - 1} e^{-\left(\frac{y_0 y_i}{1 - y_i}\right)\beta_i} (1 - y_i)^{-2}\right]$$

$$= y_0^{\sum_{i=0}^{n} \alpha_i - 1} e^{-y_0\left(\beta_0 + \sum_{i=1}^{n}\left(\frac{y_i}{1 - y_i}\right)\beta_i\right)} \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} \prod_{i=1}^{n} \left[\frac{\beta_i^{\alpha_i}}{\Gamma(\alpha_i)} \left(\frac{y_i}{1 - y_i}\right)^{\alpha_i - 1} (1 - y_i)^{-2}\right]$$

$$= \frac{\left(\beta_0 + \sum_{i=1}^{n}\left(\frac{y_i}{1 - y_i}\right)\beta_i\right)^{\sum_{i=0}^{n} \alpha_i}}{\Gamma(\sum_{i=0}^{n} \alpha_i)} y_0^{\sum_{i=0}^{n} \alpha_i - 1} e^{-y_0\left(\beta_0 + \sum_{i=1}^{n}\left(\frac{y_i}{1 - y_i}\right)\beta_i\right)} \times$$

$$\frac{\Gamma(\sum_{i=0}^{n} \alpha_i)}{\prod_{i=0}^{n} \Gamma(\alpha_i)} \frac{\beta_0^{\alpha_0} \prod_{i=1}^{n}\left[\beta_i^{\alpha_i}\left(\frac{y_i}{1 - y_i}\right)^{\alpha_i - 1}(1 - y_i)^{-2}\right]}{\left(\beta_0 + \sum_{i=1}^{n}\left(\frac{y_i}{1 - y_i}\right)\beta_i\right)^{\sum_{i=0}^{n} \alpha_i}},$$

$\alpha_i, \beta_i > 0 \; \forall \; i$, $y_0 \geq 0$, and $y_i \in (0,1)$ for $i = 1, \ldots, n$.

Marginalizing out Y_0 leads to

$$f_{Y_1,\ldots,Y_n}(y_1, \ldots, y_n) = \frac{\Gamma(\sum_{i=0}^{n} \alpha_i)}{\prod_{i=0}^{n} \Gamma(\alpha_i)} \frac{\beta_0^{\alpha_0} \prod_{i=1}^{n}\left[\beta_i^{\alpha_i}\left(\frac{y_i}{1-y_i}\right)^{\alpha_i - 1}(1-y_i)^{-2}\right]}{\left(\beta_0 + \sum_{i=1}^{n}\left(\frac{y_i}{1-y_i}\right)\beta_i\right)^{\sum_{i=0}^{n} \alpha_i}},$$

$\alpha_i, \beta_i > 0$ for $i = 0, \ldots, n$ and $y_i \in (0,1)$ for $i = 1, \ldots, n$.

Since

$$\frac{\left(\beta_0 + \sum_{i=1}^{n}\left(\frac{y_i}{1-y_i}\right)\beta_i\right)^{\sum_{i=0}^{n} \alpha_i}}{\Gamma(\sum_{i=0}^{n} \alpha_i)} y_0^{\sum_{i=0}^{n} \alpha_i - 1} e^{-y_0\left(\beta_0 + \sum_{i=1}^{n}\left(\frac{y_i}{1-y_i}\right)\beta_i\right)}$$

is a gamma random variable with parameters $\sum_{i=0}^{n} \alpha_i$ and $\left(\beta_0 + \sum_{i=1}^{n}\left(\frac{y_i}{1-y_i}\right)\beta_i\right)$. Setting $\beta_i = 1$ for $i = 0, \ldots, n$ (which allowed the pdf to be re-parameterized) results in the joint pdf

$$f_{Y_1,\ldots,Y_n}(y_1, \ldots, y_n) = \frac{\Gamma(\sum_{i=0}^{n} \alpha_i)}{\prod_{i=0}^{n} \Gamma(\alpha_i)} \frac{\prod_{i=1}^{n}\left[\left(\frac{y_i}{1-y_i}\right)^{\alpha_i - 1}\left(\frac{1}{1-y_i}\right)^{2}\right]}{\left(1 + \sum_{i=1}^{n}\left(\frac{y_i}{1-y_i}\right)\right)^{\sum_{i=0}^{n} \alpha_i}}, \tag{2.2}$$

$\alpha_i > 0$ and $y_i \in (0,1)$ for $i = 1, \ldots, n$.

The joint pdf (2.2) will be referred to as the LNMVB distribution. The univariate marginal pdf of (2.2) is as follows:

$$f_{Y_i}(y_i) = \frac{\Gamma(\alpha_0 + \alpha_i)}{\Gamma(\alpha_0)\Gamma(\alpha_i)} \frac{\left(\frac{y_i}{1-y_i}\right)^{\alpha_i - 1}\left(\frac{1}{1-y_i}\right)^{2}}{\left(1 + \frac{y_i}{1-y_i}\right)^{\alpha_0 + \alpha_i}}$$

$$= \frac{\Gamma(\alpha_0 + \alpha_i)}{\Gamma(\alpha_0)\Gamma(\alpha_i)} y_i^{\alpha_i - 1}(1 - y_i)^{\alpha_0 - 1}, \tag{2.3}$$

$\alpha_0, \alpha_i > 0 \; and \; y_i \in (0,1)$.

Thus, the marginal pdf of the LNMVB distribution as expressed by pdf (2.3) is Beta distributed with parameters α_i and α_0. Since the LNMVB distribution has Beta distributed marginal distributions, we re-parametrized the pdf (2.2) in terms of the marginal means. From pdf (2.3), $\mu_i = \frac{\alpha_i}{\alpha_i+\alpha_0}$ and therefore, $\alpha_i = \frac{\mu_i\alpha_0}{1-\mu_i}$ for $i = 1, \dots, n$. Thus, pdf (2.2) can be expressed as

$$f_{Y_1,\dots,Y_n}(y_1, \dots, y_n) = \frac{\Gamma\left(\alpha_0+\sum_{i=1}^{n}\frac{\mu_i\alpha_0}{1-\mu_i}\right)}{\Gamma(\alpha_0)\prod_{i=1}^{n}\Gamma\left(\frac{\mu_i\alpha_0}{1-\mu_i}\right)} \frac{\prod_{i=1}^{n}\left[\left(\frac{y_i}{1-y_i}\right)^{\frac{\mu_i\alpha_0}{1-\mu_i}-1}\left(\frac{1}{1-y_i}\right)^{2}\right]}{\left(1+\sum_{i=1}^{n}\left(\frac{y_i}{1-y_i}\right)\right)^{\alpha_0+\sum_{i=1}^{n}\frac{\mu_i\alpha_0}{1-\mu_i}}}, \tag{2.4}$$

$\alpha_0 > 0$ and $\mu_i, y_i \in (0,1)$ for $i = 1, \dots, n$.

Using the notation of Section 2.2 and the joint pdf (2.4) the likelihood can written as follows:

$$L(\alpha_0, \boldsymbol{\mu}; \boldsymbol{Y}) = \prod_{i=1}^{N}\left\{\frac{\Gamma\left(\alpha_0+\sum_{j=1}^{n}\frac{\mu_{ij}\alpha_0}{1-\mu_{ij}}\right)}{\Gamma(\alpha_0)\prod_{j=1}^{n}\Gamma\left(\frac{\mu_{ij}\alpha_0}{1-\mu_{ij}}\right)} \frac{\prod_{j=1}^{n}\left[\left(\frac{y_{ij}}{1-y_{ij}}\right)^{\frac{\mu_{ij}\alpha_0}{1-\mu_{ij}}-1}\left(\frac{1}{1-y_{ij}}\right)^{2}\right]}{\left(1+\sum_{j=1}^{n}\left(\frac{y_{ij}}{1-y_{ij}}\right)\right)^{\alpha_0+\sum_{j=1}^{n}\frac{\mu_{ij}\alpha_0}{1-\mu_{ij}}}}\right\}, \tag{2.5}$$

where $\boldsymbol{\mu} = \begin{pmatrix} \mu_{11} & \cdots & \mu_{1n} \\ \vdots & \ddots & \vdots \\ \mu_{N1} & \cdots & \mu_{Nn} \end{pmatrix}$ and $\boldsymbol{Y} = (\boldsymbol{Y}_1, \dots, \boldsymbol{Y}_N)'$. To add regression parameters to the likelihood (2.5), link functions were required to guarantee the regression parameters map to the domain of their respective parameter. The regression model was obtained by assuming that the mean of y_{ij} can be written as

$$g(\mu_{ij}) = \eta_{ij} = \sum_{k=1}^{p} x_{ijk}\beta_k, \quad \text{for } i = 1, \dots, N \text{ and } j = 1, \dots, n$$

and the shared parameter can be written as

$$h(\alpha_0) = \xi_{ij} = \beta_0, \text{ for } i = 1, \dots, N \text{ and } j = 1, \dots, n$$

where β_0 is a nuisance parameter ($\beta_0 \in \boldsymbol{R}$), $\boldsymbol{\beta} = (\beta_1, \dots, \beta_p)^T$ is a vector of unknown regression parameters such that $\boldsymbol{\beta} \in \boldsymbol{R}^p$ and $\boldsymbol{x}_{ij}^T$ are observations on p covariates ($p < N$), which are assumed known and fixed. $g(\cdot)$ and $h(\cdot)$ are link functions that are strictly monotonic and twice differentiable. Furthermore $g(\cdot)$ maps $(0,1)$ onto $\boldsymbol{R}$ and $h(\cdot)$ maps $(0, \infty)$ onto $\boldsymbol{R}$.

Several link functions were possible for $g(\cdot)$. Three commonly used link functions that map $(0,1)$ onto $\boldsymbol{R}$ are the logit link, the probit link, and the complementary log-log link. For further details and comparisons of the three link functions, see McCullagh and Nelder[28] (Section 4.3.1). We used the logit link function (i.e., $g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$) for computational simplicity. For $h(\cdot)$ we used the log link, i.e., $h(\alpha_0) = \log(\alpha_0)$. Using the logit link and log link, μ_{ij} and α_0 can be expressed as

$$\mu_{ij} = \frac{e^{\boldsymbol{x}_{ij}^T\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_{ij}^T\boldsymbol{\beta}}} \text{ and } \alpha_0 = e^{\beta_0}, \text{ respectively.}$$

Including regression parameters into the likelihood (2.5) using the aforementioned link functions results in the following likelihood:

$$L(\beta_0, \boldsymbol{\beta}; \boldsymbol{X}, \boldsymbol{Y}) = \prod_{i=1}^{N} \left\{ \frac{\Gamma\left(e^{\beta_0} + \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right)}{\Gamma(e^{\beta_0}) \prod_{j=1}^{n} \Gamma\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right)} \frac{\prod_{j=1}^{n}\left[\left(\frac{y_{ij}}{1-y_{ij}}\right)^{e^{\beta_0+\boldsymbol{x}_{ij}^T\boldsymbol{\beta}}-1} \left(\frac{1}{1-y_{ij}}\right)^2\right]}{\left(1+\sum_{j=1}^{n}\left(\frac{y_{ij}}{1-y_{ij}}\right)\right)^{e^{\beta_0}+\sum_{j=1}^{n} e^{\beta_0+\boldsymbol{x}_{ij}^T\boldsymbol{\beta}}}} \right\}$$

and log-likelihood

$$l(\beta_0, \boldsymbol{\beta}; \boldsymbol{X}, \boldsymbol{Y}) = \sum_{i=1}^{N} \left\{ log\Gamma\left(e^{\beta_0} + \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right) - log\Gamma\left(e^{\beta_0}\right) - \right.$$

$$\sum_{j=1}^{n}\left[log\Gamma\left(e^{\beta_0+\boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right) - \left(e^{\beta_0+\boldsymbol{x}_{ij}^T\boldsymbol{\beta}} - 1\right)\log\left(\frac{y_{ij}}{1-y_{ij}}\right) + 2\log(1-y_{ij})\right] -$$

$$\left(e^{\beta_0} + \textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right) log\left(1 + \textstyle\sum_{j=1}^{n}\left(\frac{y_{ij}}{1-y_{ij}}\right)\right)\Big\}$$

(2.6)

Taking partial derivatives of the log-likelihood (2.6) with respect to β_i $(i = 0, \dots, p)$ leads to the score equations, denoted U_i for $i = 0, \dots, p$.

$$U_0 = \frac{\partial l}{\partial \beta_0} = \textstyle\sum_{i=1}^{N}\Big\{\psi\left(e^{\beta_0} + \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right)\left(e^{\beta_0} + \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right) - \psi\left(e^{\beta_0}\right)e^{\beta_0} -$$

$$\textstyle\sum_{j=1}^{n}\left[\psi\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right)e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}} - e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\log\left(\frac{y_{ij}}{1-y_{ij}}\right)\right] - \Big(e^{\beta_0} +$$

$$\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\Big) log\left(1 + \sum_{j=1}^{n}\left(\frac{y_{ij}}{1-y_{ij}}\right)\right)\Big\}$$

and

$$U_k = \frac{\partial l}{\partial \beta_k} = \textstyle\sum_{i=1}^{N}\Big\{\psi\left(e^{\beta_0} + \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right)\left(\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\, x_{ijk}\right) -$$

$$\textstyle\sum_{j=1}^{n}\left[\psi\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\right)e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}x_{ijk} - e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}x_{ijk}\log\left(\frac{y_{ij}}{1-y_{ij}}\right)\right] -$$

$$\left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T\boldsymbol{\beta}}\, x_{ijk}\right) log\left(1 + \textstyle\sum_{j=1}^{n}\left(\frac{y_{ij}}{1-y_{ij}}\right)\right)\Big\},$$

for $k = 1, \dots, p$ and $\psi(\cdot)$ is the digamma function.

The maximum likelihood estimator (MLE) for the regression parameters can be found by setting the score equations equal to zero and solving for the respective regression parameter. Unfortunately, there is no closed form expression for the MLEs of the regression parameters. Therefore, we used an iterative procedure, i.e., the quasi-Newton-Raphson algorithm.[62]

The problem can formally be defined as follows:

$$\text{given } U\colon \boldsymbol{R}^{p+1} \to \boldsymbol{R}^{p+1}, \qquad \text{find } \beta_* \in \boldsymbol{R}^{p+1} \text{ such that } U(\beta_*) = 0 \qquad (2.7)$$

where U is assumed to be continuously differentiable. The Newton-Raphson method for problem (2.7) is derived by finding the root of an affine approximation to U at the current iterate β_c.[62] Specifically, we can express U at a perturbation p of β_c as

$$U(\beta_c + s) = U(\beta_c) + \int_{\beta_c}^{\beta_c+s} \nabla U(t)^T dt \qquad (2.8)$$

where $\nabla U(t)^T = \nabla^2 l(t)$ is referred to as the Hessian matrix of the log-likelihood whose entries are

$$\nabla^2 l(t)_{ij} = \frac{\partial^2 l(t)}{\partial t_i \partial t_j}, \qquad 0 \leq i, \ \ j \leq p.$$

Using $\nabla U(t)^T s$ to approximate the integral in (2.8) gives the affine approximation to U at a perturbation s of β_c.[62] Setting the affine approximation to U at a perturbation s of β_c and solving leads to the Newton-Raphson iteration

$$\nabla U(\beta_c)^T s = -U(\beta_c),$$

$$\beta_+ = \beta_c + s.^{62} \qquad (2.9)$$

Since β_+ is not expected to equal β_*, but only a better estimate than β_c, (2.9) can be made into an algorithm by applying it iteratively from a starting value,[62] i.e., at each iteration i, solve

$$\nabla U\left(\beta^{(i)}\right)^T s^{(i)} = -U\left(\beta^{(i)}\right),$$

$$\beta^{(i+1)} = \beta^{(i)} + s^{(i)}.$$

Furthermore, additional steps to the Newton-Raphson algorithm can be included to ensure global convergence, i.e., the algorithm will converge to a local minimum regardless of the starting value.[62] Therefore the quasi-Newton-Raphson algorithm[62] is as follows:

At each iteration (i):

1. Compute $U(\beta^{(i)})$ and decide whether to stop or continue. Stop if $\max_k U_k(\beta^{(i)}) < tol_1$ or $\frac{\max_k \left|\beta_k^{(i)} - \beta_k^{(i-1)}\right|}{\max_r |\beta_r|} < tol_2$, where tol_i is the specified tolerance, commonly 10^{-8}.
2. Approximate $\nabla U(\beta^{(i)})$ using finite differences.
3. Estimate condition number of $\nabla U(\beta^{(i)})$ using algorithm proposed by Cline and colleagues.[63] If the Hessian is ill-conditioned (nearly singular), perturb it, i.e., $\nabla U(\beta^{(i)}) = \nabla U(\beta^{(i)}) + (p+1) macheps^{\frac{1}{2}} \left|\left|\nabla U(\beta^{(i)})\right|\right|_1 I_{p+1}$ where $macheps$ is the smallest number τ such that $1 + \tau > 1$.
4. Solve $\nabla U(\beta^{(i)})^T s^{(i)} = -U(\beta^{(i)})$.
5. Decide whether to take a Newton step, $\beta^{(i+1)} = \beta^{(i)} + s^{(i)}$, or use cubic backtracking to choose $\beta^{(i+1)}$.

The quasi-Newton-Raphson algorithm[62] was implemented in R[64] version 3.4.2 using package nleqslv.[65] The initial values of the regression parameters were set such that they correspond to their respective means on the data scale. We used finite differences to estimate the Hessian matrix; however, for completeness the Hessian matrix can be calculated as follows:

$$\frac{\partial U_0}{\partial \beta_0} = \sum_{i=1}^{N} \Bigg\{ \psi_1\left(e^{\beta_0} + \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(e^{\beta_0} + \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)^2 + \psi\left(e^{\beta_0} + \right.$$

$$\left. \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(e^{\beta_0} + \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right) - \psi_1(e^{\beta_0}) e^{2\beta_0} - \psi(e^{\beta_0}) e^{\beta_0} -$$

$$\sum_{j=1}^{n} \Bigg[\psi_1\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)^2 + \psi\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right) e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} -$$

$$e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} \log\left(\frac{y_{ij}}{1 - y_{ij}}\right)\Bigg] - \left(e^{\beta_0} + \sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right) log\left(1 + \sum_{j=1}^{n}\left(\frac{y_{ij}}{1 - y_{ij}}\right)\right)\Bigg\},$$

$$\begin{aligned}\frac{\partial U_0}{\partial \beta_k} = \textstyle\sum_{i=1}^{N} \Big\{ &\psi_1\left(e^{\beta_0} + \textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(e^{\beta_0} + \textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\, x_{ijk}\right) + \\ &\psi\left(e^{\beta_0} + \textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\, x_{ijk}\right) - \\ &\textstyle\sum_{j=1}^{n} \Big[\psi_1\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)^2 x_{ijk} + \psi\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right) e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} x_{ijk} - \\ &e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} x_{ijk} \log\left(\frac{y_{ij}}{1 - y_{ij}}\right)\Big] - \left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} x_{ijk}\right) log\left(1 + \textstyle\sum_{j=1}^{n}\left(\frac{y_{ij}}{1 - y_{ij}}\right)\right)\Big\},\end{aligned}$$

for $k = 1, \ldots, p$,

$$\begin{aligned}\frac{\partial U_k}{\partial \beta_k} = \textstyle\sum_{i=1}^{N} \Big\{ &\psi_1\left(e^{\beta_0} + \textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\, x_{ijk}\right)^2 + \psi\Big(e^{\beta_0} + \\ &\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\Big)\left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\, x_{ijk}^2\right) - \textstyle\sum_{j=1}^{n} \Big[\psi_1\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} x_{ijk}\right)^2 + \\ &\psi\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right) e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} x_{ijk}^2 - e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} x_{ijk}^2 \log\left(\frac{y_{ij}}{1 - y_{ij}}\right)\Big] - \\ &\left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\, x_{ijk}^2\right) log\left(1 + \textstyle\sum_{j=1}^{n}\left(\frac{y_{ij}}{1 - y_{ij}}\right)\right)\Big\},\end{aligned}$$

for $k = 1, \ldots, p$, and

$$\begin{aligned}\frac{\partial U_k}{\partial \beta_r} = \textstyle\sum_{i=1}^{N} \Big\{ &\psi_1\left(e^{\beta_0} + \textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\, x_{ijk}\right)\left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\, x_{ijr}\right) + \\ &\psi\left(e^{\beta_0} + \textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\, x_{ijk} x_{ijr}\right) - \\ &\textstyle\sum_{j=1}^{n} \Big[\psi_1\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right)^2 x_{ijk} x_{ijr} + \psi\left(e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\right) e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} x_{ijk} x_{ijr} - \\ &e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}} x_{ijk} x_{ijr} \log\left(\frac{y_{ij}}{1 - y_{ij}}\right)\Big] - \left(\textstyle\sum_{j=1}^{n} e^{\beta_0 + \boldsymbol{x}_{ij}^T \boldsymbol{\beta}}\, x_{ijk} x_{ijr}\right) log\Big(1 + \\ &\textstyle\sum_{j=1}^{n}\left(\frac{y_{ij}}{1 - y_{ij}}\right)\Big)\Big\}\end{aligned}$$

for $k = 1, \ldots p$, $r = 0, \ldots, p$, such that $k \neq r$, $\psi(\cdot)$ is the digamma function, and $\psi_1(\cdot)$ is the trigamma function.

The negation of the Hessian matrix is referred to as the Fisher information matrix. We used the inverse of the observed Fisher information matrix as variance-covariance estimates (denoted $\widehat{\boldsymbol{\Sigma}}$) of the regression parameters.

Lastly, pairwise correlations were calculated as

$$\frac{E[Y_iY_{i+1}]-E[Y_i]E[Y_{i+1}]}{\sigma_{Y_i}\sigma_{Y_{i+1}}}, \quad \text{for } i = 1, \dots, n-1.$$

The $E[Y_i] = \mu_{Y_i} = \frac{e^{x_{ij}^T\boldsymbol{\beta}}}{1+e^{x_{ij}^T\boldsymbol{\beta}}}$ and $\sigma_{Y_i} = \frac{\mu_{Y_i}\left(1-\mu_{Y_i}\right)^2}{1+\alpha_0-\mu_{Y_i}}$ where $\alpha_0 = e^{\beta_0}$. The $E[Y_iY_{i+1}]$ requires the calculation of a double integral. Specifically,

$$E[Y_iY_{i+1}] = \int_0^1\int_0^1 y_iy_{i+1}f_{Y_i,Y_{i+1}}(y_i, y_{i+1})dy_idy_{i+1}$$

where

$$f_{Y_i,Y_{i+1}}(y_i, y_{i+1}) = \frac{\Gamma\left(\alpha_0\left[1+\frac{\mu_{Y_i}}{1-\mu_{Y_i}}+\frac{\mu_{Y_{i+1}}}{1-\mu_{Y_{i+1}}}\right]\right)}{\Gamma(\alpha_0)\Gamma\left(\frac{\mu_{Y_i}}{1-\mu_{Y_i}}\right)\Gamma\left(\frac{\mu_{Y_{i+1}}}{1-\mu_{Y_{i+1}}}\right)}\frac{\left(\frac{y_i}{1-y_i}\right)^{\frac{\alpha_0\mu_{Y_i}}{1-\mu_{Y_i}}-1}\left(\frac{1}{1-y_i}\right)^2\left(\frac{y_{i+1}}{1-y_{i+1}}\right)^{\frac{\alpha_0\mu_{Y_{i+1}}}{1-\mu_{Y_{i+1}}}-1}\left(\frac{1}{1-y_{i+1}}\right)^2}{\left(1+\frac{y_i}{1-y_i}+\frac{y_{i+1}}{1-y_{i+1}}\right)^{\alpha_0\left[1+\frac{\mu_{Y_i}}{1-\mu_{Y_i}}+\frac{\mu_{Y_{i+1}}}{1-\mu_{Y_{i+1}}}\right]}}.$$

The double integral was numerically estimated using the “TwoD” algorithm, that is Gauss-Konrod with (3,7)-nodes on 2D rectangles implemented in R[64] package pracma.[66] Using simulations, Olkin and Liu[38] were able to obtain pairwise correlations ranging from zero to one.

An alternative to constructing a joint pdf to model repeated measures data is inducing correlation on univariate marginal distributions through copulas. Lee's[8] proposed multivariate copula is a flexible copula allowing for negative correlation. However, this flexibility comes at the expense of requiring additional parameters for model fitting. Section 2.4 focuses on re-parametrizing Lee's[8] proposed multivariate copula and reducing the number of parameters required.

2.4. Sarmanov-Lee Multivariate Beta (SLMVB)

Lee[8] proposed a multivariate extension to Sarmanov's[7] bivariate family of distributions. Lee's[8] multivariate extension is as follows:

$$h_{X_1,\dots,X_n}(x_1,\dots,x_n) = \{\prod_{i=1}^{n} f_{X_i}(x_i)\}\{1 + R_{\phi_1,\dots,\phi_n,\Omega_n}(x_1,\dots,x_n)\} \tag{2.10}$$

where $f_{X_1}(x_1),\dots,f_{X_n}(x_n)$ are specified marginal distributions and

$$\begin{aligned} &R_{\phi_1,\dots,\phi_n,\Omega_n}(x_1,\dots,x_n) \\ &\qquad = \sum_{j_1<j_2}^{n-1}\sum_{j_2}^{n} \omega_{j_1,j_2}\phi_{j_1}(x_{j_1})\phi_{j_2}(x_{j_2}) \\ &\qquad + \sum_{j_1<j_2}^{n-2}\sum_{j_2<j_3}^{n-1}\sum_{j_3}^{n} \omega_{j_1,j_2,j_3}\phi_{j_1}(x_{j_1})\phi_{j_2}(x_{j_2})\phi_{j_3}(x_{j_3}) + \cdots \\ &\qquad + \omega_{1,\dots,n}\prod_{i=1}^{n}\phi_i(x_i) \end{aligned}$$

with $\Omega_n = \{\omega_{j_1,j_2},\omega_{j_1,j_2,j_3},\dots,\omega_{1,\dots,n}\}$ such that $1 + R_{\phi_1,\dots,\phi_n,\Omega_n}(x_1,\dots,x_n) \geq 0$ holds for all $x_i \in$ support of f_{X_i} for $i = 1,\dots,n$.

The multivariate Beta can be specified by letting the marginal distributions of the joint pdf (2.10) be specified as Beta distributions, namely

$$f_{X_i}(x_i) = \frac{\Gamma(\alpha_i+\beta_i)}{\Gamma(\alpha_i)\Gamma(\beta_i)} x_i^{\alpha_i-1}(1-x_i)^{\beta_i-1} \tag{2.11}$$

$\alpha_i, \beta_i > 0$ and $x_i \in (0,1)$ for $i = 1,\dots,n$.

Additionally, the mixing functions, ϕ_j, are defined using Lee's[8] proposed mixing function for the bivariate case, i.e.

$$\phi_j(x_j) = x_j - \mu_j, \quad \text{for } j = 1,\dots,n$$

where $\mu_j = \frac{\alpha_j}{\alpha_j - \beta_j}$ such that α_j, β_j are the parameters from the respective marginal Beta distributions.

We re-parameterized the ω's in terms of correlation parameters, e.g., $\omega_{i,j} = \frac{\rho_{i,j}}{\sigma_i \sigma_j}$ and $\omega_{i,j,k} = \frac{\rho_{i,j,k}}{\sigma_i \sigma_j \sigma_k}$ where $\rho_{i,j}$ is the correlation between the i^{th} and j^{th} marginal distribution and $\rho_{i,j,k}$ and $\sigma_i, \sigma_j, \sigma_k$ is the correlation and standard deviations, respectively, between the i^{th}, j^{th}, and k^{th} marginal distributions. Derivations of this re-parameterization of the $\omega's$ were limited to a tri-variate distribution for simplicity; however, all formulas presented are easily extended to an n-variate distribution. The following are the derivations of the re-parameterization of the ω's.

Define $h_{X_1, X_2, X_3}(x_1, x_2, x_3)$ using equation (2.10) with common marginal distributions and the proposed mixing functions, then

$$E[X_1 X_2 | X_3] = \int_0^1 \int_0^1 f_{X_1}(x_1) f_{X_2}(x_2)[1 + \omega_{1,2}(x_1 - \mu_1)(x_2 - \mu_2) + \omega_{1,3}(x_1 - \mu_1)(x_3 - \mu_3) + \omega_{2,3}(x_2 - \mu_2)(x_3 - \mu_3) + \omega_{1,2,3}(x_1 - \mu_1)(x_2 - \mu_2)(x_3 - \mu_3) xy dx dy$$

$$= \mu_1 \mu_2 + \omega_{1,2} \sigma_1^2 \sigma_2^2 + \omega_{1,3} \sigma_1^2 \mu_2 (x_3 - \mu_3) + \omega_{2,3} \mu_1 \sigma_2^2 (x_3 - \mu_3) + \omega_{1,2,3} \sigma_1^2 \sigma_2^2 (x_3 - \mu_3).$$

$$E\big[E[X_1 X_2 | X_3]\big] = E[X_1 X_2] = \mu_1 \mu_2 + \omega_{1,2} \sigma_1^2 \sigma_2^2.$$

Furthermore, it can be shown that

$$E\big[X_i X_j\big] = \mu_i \mu_j + \omega_{i,j} \sigma_i^2 \sigma_j^2, \quad \text{for } i \neq j. \tag{2.12}$$

The univariate expected values for X_1 is

$$E[X_1 | X_2 X_3] = \int_0^1 f_{X_1}(x_1)[1 + \omega_{1,2}(x_1 - \mu_1)(x_2 - \mu_2) + \omega_{1,3}(x_1 - \mu_1)(x_3 - \mu_3) + \omega_{2,3}(x_2 - \mu_2)(x_3 - \mu_3) + \omega_{1,2,3}(x_1 - \mu_1)(x_2 - \mu_2)(x_3 - \mu_3) x dx$$

$$= \mu_1 + \omega_{1,2}\sigma_1^2(x_2 - \mu_2) + \omega_{1,3}\sigma_1^2(x_3 - \mu_3) + \omega_{2,3}\mu_1(x_2 - \mu_2)(x_3 - \mu_3) + \omega_{1,2,3}\sigma_1^2(x_2 - \mu_3)(x_3 - \mu_3).$$

$$E\left[E[X_1|X_2X_3]\right] = E[X_1] = \mu_1.$$

Which can be generalized to any X_i

$$E[X_i] = \mu_i. \tag{2.13}$$

Therefore,

$$\rho_{1,2} = \frac{E[X_1X_2] - E[X_1]E[X_2]}{\sigma_1\sigma_2} = \omega_{1,2}\sigma_1\sigma_2.$$

Thus,

$$\omega_{1,2} = \frac{\rho_{1,2}}{\sigma_1\sigma_2}.$$

Using equations (2.12) and (2.13)

$$\omega_{i,j} = \frac{\rho_{i,j}}{\sigma_i\sigma_j} \qquad \text{for } i \neq j. \tag{2.14}$$

The correlation of three marginal distributions can be calculated as follows:

$$\rho_{1,23} = \frac{E[(X_1-\mu_1)(X_2-\mu_2)(X_3-\mu_3)]}{\sigma_1\sigma_2\sigma_3}$$

$$= \frac{E[X_1X_2X_3] - E[X_1X_2]\mu_3 - E[X_1X_3]\mu_2 - E[X_2X_3]\mu_1 + 2\mu_1\mu_2\mu_3}{\sigma_1\sigma_2\sigma_3}$$

$$= \big[\mu_1\mu_2\mu_3 + \omega_{1,2}\sigma_1^2\sigma_2^2\mu_3 + \omega_{1,3}\sigma_1^2\mu_2\sigma_3^2 + \omega_{2,3}\mu_1\sigma_2^2\sigma_3^2 + \omega_{1,2,3}\sigma_1^2\sigma_2^2\sigma_3^2 - \big(\mu_1\mu_2 + \omega_{1,2}\sigma_1^2\sigma_2^2\big)\mu_3 - \big(\mu_1\mu_3 + \omega_{1,3}\sigma_1^2\sigma_3^2\big)\mu_2 - \big(\mu_2\mu_3 + \omega_{2,3}\sigma_2^2\sigma_3^2\big)\mu_1 + 2\mu_1\mu_2\mu_3\big]/\sigma_1\sigma_2\sigma_3$$

$$= \omega_{1,2,3}\sigma_1\sigma_2\sigma_3$$

using

$$E[X_1X_2X_3] = \int_0^1\int_0^1\int_0^1 f_{X_1}(x_1)f_{X_2}(x_2)f_{X_3}(x_3)[1 + \omega_{1,2}(x_1 - \mu_1)(x_2 - \mu_2) + \omega_{1,3}(x_1 - \mu_1)(x_3 - \mu_3) + \omega_{2,3}(x_2 - \mu_2)(x_3 - \mu_3) + \omega_{1,2,3}(x_1 - \mu_1)(x_2 - \mu_2)(x_3 - \mu_3)xyzdxdydz$$

$$= \mu_1\mu_2\mu_3 + \omega_{1,2}\sigma_1^2\sigma_2^2\mu_3 + \omega_{1,3}\sigma_1^2\mu_2\sigma_3^2 + \omega_{2,3}\mu_1\sigma_2^2\sigma_3^2 + \omega_{1,2,3}\sigma_1^2\sigma_2^2\sigma_3^2.$$

Therefore,

$$\omega_{1,2,3} = \frac{\rho_{1,2,3}}{\sigma_1\sigma_2\sigma_3}$$

which can be generalized to

$$\omega_{i,j,k} = \frac{\rho_{i,j,k}}{\sigma_i\sigma_j\sigma_k} \qquad \text{for } i \neq j \neq k. \tag{2.15}$$

Equation (2.15) requires the use of higher-order correlations, i.e., the correlation between three or more variables. Wang and Zheng[67] proposed a multivariate correlation coefficient (MCC) that we used to express higher-order correlations in terms of pairwise Pearson's[14] correlation coefficients. Specifically, let the entries of matrix M be the pairwise Pearson's[14] correlation coefficients, i.e.,

$$M = \begin{bmatrix} \rho_{11} & \cdots & \rho_{1n} \\ \vdots & \ddots & \vdots \\ \rho_{n1} & \cdots & \rho_{nn} \end{bmatrix} = \begin{bmatrix} 1 & \cdots & \rho_{1n} \\ \vdots & \ddots & \vdots \\ \rho_{n1} & \cdots & 1 \end{bmatrix}$$

then

$$\rho_{1,\ldots,n} = \sqrt{1 - \det(M)}.$$

In the bivariate case, the MCC reduces to Pearson's[14] correlation coefficient. Additionally, the pairwise correlations must be restricted to $[0,1]$ for the MCC $\in [0,1]$. Letting pairwise correlations be negative produces infeasible MCCs, i.e., MCC $\notin [-1,1]$.

Using the parameterization proposed by Ferrari and Cribari-Neto[21] (i.e., $\mu_i = \frac{\alpha_i}{\alpha_i+\beta_i}$ and $\phi_i = \alpha_i + \beta_i$) and re-parametrizing the ω's in terms of (2.14) and (2.15) leads to the SLMVB:

$$f_{X_1,\dots,X_n}(x_1,\dots,x_n) = \prod_{i=1}^{n}\left\{\frac{\Gamma(\phi_i)}{\Gamma(\mu_i\phi_i)\Gamma(\phi_i(1-\mu_i))}x_i^{\mu_i\phi_i-1}(1-x_i)^{\phi_i(1-\mu_i)-1}\right\}\left\{1+\right.$$

$$\sum_{j_1<j_2}^{n-1}\sum_{j_2}^{n}\frac{\rho_{j_1,j_2}(x_{j_1}-\mu_{j_1})(x_{j_2}-\mu_{j_2})}{\left(\frac{\mu_{j_1}(1-\mu_{j_1})}{1+\phi_{j_1}}\right)^{\frac{1}{2}}\left(\frac{\mu_{j_2}(1-\mu_{j_2})}{1+\phi_{j_2}}\right)^{\frac{1}{2}}}+$$

$$\sum_{j_1<j_2}^{n-2}\sum_{j_2<j_3}^{n-1}\sum_{j_3}^{n}\frac{\rho_{j_1,j_2,j_3}(x_{j_1}-\mu_{j_1})(x_{j_2}-\mu_{j_2})(x_{j_3}-\mu_{j_3})}{\left(\frac{\mu_{j_1}(1-\mu_{j_1})}{1+\phi_{j_1}}\right)^{\frac{1}{2}}\left(\frac{\mu_{j_2}(1-\mu_{j_2})}{1+\phi_{j_2}}\right)^{\frac{1}{2}}\left(\frac{\mu_{j_3}(1-\mu_{j_3})}{1+\phi_{j_3}}\right)^{\frac{1}{2}}}+\cdots+$$

$$\left.\rho_{1,\dots,n}\prod_{i=1}^{n}\frac{x_i-\mu_i}{\left(\frac{\mu_i(1-\mu_i)}{1+\phi_i}\right)^{\frac{1}{2}}}\right\}, \tag{2.16}$$

$x_i, \mu_i \in (0,1), \rho \in [0,1]$, and $\phi_i \geq 0$ for $i = 1,\dots,n$.

For our purposes, we limited the SLMVB to 4 repeated measures for brevity. By imposing a correlation structure we were able to re-express $\rho_{1,2},\dots,\rho_{1,2,3},\dots,\rho_{1,\dots,n}$ in terms of a single parameter, ρ using Wang and Zheng's[67] MCC. Restricting the correlation to a CS structure leads to the follow parameterizations of the correlation variables:

$$\rho_{1,2} = \rho_{1,3} = \rho_{1,4} = \rho_{2,3} = \rho_{2,4} = \rho_{3,4} = \rho$$

$$\rho_{1,2,3} = \rho_{1,2,4} = \rho_{1,3,4} = \rho_{2,3,4} = (3\rho^2 - 2\rho^3)^{\frac{1}{2}}$$

$$\rho_{1,2,3,4} = (6\rho^2 - 8\rho^3 + 3\rho^4)^{\frac{1}{2}}$$

Furthermore, if we impose an AR(1) correlation structure, the correlations can be expressed as follows:

$$\rho_{i,j} = \rho^{|i-j|}$$

$$\rho_{1,2,3} = \rho_{2,3,4} = (2\rho^2 - \rho^4)^{\frac{1}{2}},\ \rho_{1,2,4} = \rho_{1,3,4} = (\rho^2 + \rho^4 - \rho^6)^{\frac{1}{2}}$$

$$\rho_{1,2,3,4} = (3\rho^2 - 3\rho^4 + \rho^6)^{\frac{1}{2}}$$

Thus, using the joint distribution (2.16), either correlation structure (CS or AR(1)), and the notation of Section 2.2 the likelihood can be written as follows:

$$\begin{aligned}
L(\boldsymbol{\mu}, \boldsymbol{\phi}, \rho; \boldsymbol{Y}) = \prod_{i=1}^{N} \Bigg[\prod_{j=1}^{n} \left\{ \frac{\Gamma(\phi_j)}{\Gamma(\mu_{ij}\phi_j)\Gamma\big(\phi_j(1-\mu_{ij})\big)} y_{ij}^{\mu_{ij}\phi_j - 1} \left(1 - y_{ij}\right)^{\phi_j(1-\mu_{ij})-1} \right\} \Bigg\{ 1 + \\
\sum_{j_1<j_2}^{n-1} \sum_{j_2}^{n} \frac{\rho_{j_1,j_2}(y_{ij_1}-\mu_{ij_1})(y_{ij_2}-\mu_{ij_2})}{\left(\frac{\mu_{ij_1}(1-\mu_{ij_1})}{1+\phi_{j_1}}\right)^{\frac{1}{2}} \left(\frac{\mu_{ij_2}(1-\mu_{ij_2})}{1+\phi_{j_2}}\right)^{\frac{1}{2}}} + \\
\sum_{j_1<j_2}^{n-2} \sum_{j_2<j_3}^{n-1} \sum_{j_3}^{n} \frac{\rho_{j_1,j_2,j_3}(y_{ij_1}-\mu_{ij_1})(y_{ij_2}-\mu_{ij_2})(y_{ij_3}-\mu_{ij_3})}{\left(\frac{\mu_{ij_1}(1-\mu_{ij_1})}{1+\phi_{j_1}}\right)^{\frac{1}{2}} \left(\frac{\mu_{ij_2}(1-\mu_{ij_2})}{1+\phi_{j_2}}\right)^{\frac{1}{2}} \left(\frac{\mu_{j_3}(1-\mu_{ij_3})}{1+\phi_{j_3}}\right)^{\frac{1}{2}}} + \cdots + \\
\rho_{1,\ldots,n} \prod_{j=1}^{n} \frac{y_{ij}-\mu_{ij}}{\left(\frac{\mu_{ij}(1-\mu_{ij})}{1+\phi_j}\right)^{\frac{1}{2}}} \Bigg\} \Bigg],
\end{aligned}$$

where $\boldsymbol{\mu} = \begin{pmatrix} \mu_{11} & \cdots & \mu_{1n} \\ \vdots & \ddots & \vdots \\ \mu_{N1} & \cdots & \mu_{Nn} \end{pmatrix}$, $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_n)'$, and $\boldsymbol{Y} = (\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_N)'$.

Similar to the LNMVB, link functions were required to guarantee the regression parameters map to the domain of their respective parameter. Therefore, the regression model is obtained by assuming that the mean of y_{ij} can be expressed as

$$g(\mu_{ij}) = \eta_{ij} = \sum_{k=1}^{p} x_{ijk}\beta_k, \quad \text{for } i = 1, \ldots, N \text{ and } j = 1, \ldots, n$$

and the precision parameter can be written as

$$h(\phi_j) = \xi_{ij} = \beta_{p+j}, \text{ for } i = 1, \dots, N \text{ and } j = 1, \dots, n.$$

As with the LNMVB we let $g(\cdot)$ be the logit-link function and $h(\cdot)$ be the log-link function. The log-likelihood of the SLMVB with regression parameters and specified link functions can be expressed as:

$$l(\boldsymbol{\beta}, \boldsymbol{\beta}_\phi; \boldsymbol{X}, \boldsymbol{Y}) = \sum_{i=1}^{N} \left[\sum_{j=1}^{4} \left\{ log\Gamma\left(e^{\beta_{\phi_j}}\right) - log\Gamma\left(\frac{e^{\beta_{\phi_j}+x_{ij}^T\boldsymbol{\beta}}}{1+e^{x_{ij}^T\boldsymbol{\beta}}}\right) - log\Gamma\left(\frac{e^{\beta_{\phi_j}}}{1+e^{x_{ij}^T\boldsymbol{\beta}}}\right) + \right.\right.$$

$$\left.\left(\frac{e^{\beta_{\phi_j}+x_{ij}^T\boldsymbol{\beta}}}{1+e^{x_{ij}^T\boldsymbol{\beta}}} - 1\right)\log(y_{ij}) + \left(\frac{e^{\beta_{\phi_j}}}{1+e^{x_{ij}^T\boldsymbol{\beta}}} - 1\right)\log(1-y_{ij})\right\} + \log\left\{1 + \right.$$

$$\rho_{1,2}\left(\frac{y_{i1}-\frac{e^{x_{i1}^T\boldsymbol{\beta}}}{1+e^{x_{i1}^T\boldsymbol{\beta}}}}{\left(1+e^{x_{i1}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{x_{i1}^T\boldsymbol{\beta}}}{1+e^{\beta_{\phi_1}}}}}\right)\left(\frac{y_{i2}-\frac{e^{x_{i2}^T\boldsymbol{\beta}}}{1+e^{x_{i2}^T\boldsymbol{\beta}}}}{\left(1+e^{x_{i2}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{x_{i2}^T\boldsymbol{\beta}}}{1+e^{\beta_{\phi_2}}}}}\right) + \cdots +$$

$$\rho_{3,4}\left(\frac{y_{i3}-\frac{e^{x_{i3}^T\boldsymbol{\beta}}}{1+e^{x_{i3}^T\boldsymbol{\beta}}}}{\left(1+e^{x_{i3}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{x_{i3}^T\boldsymbol{\beta}}}{1+e^{\beta_{\phi_3}}}}}\right)\left(\frac{y_{i4}-\frac{e^{x_{i4}^T\boldsymbol{\beta}}}{1+e^{x_{i4}^T\boldsymbol{\beta}}}}{\left(1+e^{x_{i4}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{x_{i4}^T\boldsymbol{\beta}}}{1+e^{\beta_{\phi_4}}}}}\right) +$$

$$\rho_{1,2,3}\left(\frac{y_{i1}-\frac{e^{x_{i1}^T\boldsymbol{\beta}}}{1+e^{x_{i1}^T\boldsymbol{\beta}}}}{\left(1+e^{x_{i1}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{x_{i1}^T\boldsymbol{\beta}}}{1+e^{\beta_{\phi_1}}}}}\right)\left(\frac{y_{i2}-\frac{e^{x_{i2}^T\boldsymbol{\beta}}}{1+e^{x_{i2}^T\boldsymbol{\beta}}}}{\left(1+e^{x_{i2}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{x_{i2}^T\boldsymbol{\beta}}}{1+e^{\beta_{\phi_2}}}}}\right)\left(\frac{y_{i3}-\frac{e^{x_{i3}^T\boldsymbol{\beta}}}{1+e^{x_{i3}^T\boldsymbol{\beta}}}}{\left(1+e^{x_{i3}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{x_{i3}^T\boldsymbol{\beta}}}{1+e^{\beta_{\phi_3}}}}}\right) + \cdots +$$

$$\rho_{2,3,4}\left(\frac{y_{i2}-\frac{e^{\boldsymbol{x}_{i2}^T\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_{i2}^T\boldsymbol{\beta}}}}{\left(1+e^{\boldsymbol{x}_{i2}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{\boldsymbol{x}_{i2}^T\boldsymbol{\beta}}}{1+e^{\beta}\phi_2}}}\right)\left(\frac{y_{i3}-\frac{e^{\boldsymbol{x}_{i3}^T\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_{i3}^T\boldsymbol{\beta}}}}{\left(1+e^{\boldsymbol{x}_{i3}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{\boldsymbol{x}_{i3}^T\boldsymbol{\beta}}}{1+e^{\beta}\phi_3}}}\right)\left(\frac{y_{i4}-\frac{e^{\boldsymbol{x}_{i4}^T\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_{i4}^T\boldsymbol{\beta}}}}{\left(1+e^{\boldsymbol{x}_{i4}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{\boldsymbol{x}_{i4}^T\boldsymbol{\beta}}}{1+e^{\beta}\phi_4}}}\right)+$$

$$\rho_{1,2,3,4}\left(\frac{y_{i1}-\frac{e^{\boldsymbol{x}_{i1}^T\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_{i1}^T\boldsymbol{\beta}}}}{\left(1+e^{\boldsymbol{x}_{i1}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{\boldsymbol{x}_{i1}^T\boldsymbol{\beta}}}{1+e^{\beta}\phi_1}}}\right)\left(\frac{y_{i2}-\frac{e^{\boldsymbol{x}_{i2}^T\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_{i2}^T\boldsymbol{\beta}}}}{\left(1+e^{\boldsymbol{x}_{i2}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{\boldsymbol{x}_{i2}^T\boldsymbol{\beta}}}{1+e^{\beta}\phi_2}}}\right)\left(\frac{y_{i3}-\frac{e^{\boldsymbol{x}_{i3}^T\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_{i3}^T\boldsymbol{\beta}}}}{\left(1+e^{\boldsymbol{x}_{i3}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{\boldsymbol{x}_{i3}^T\boldsymbol{\beta}}}{1+e^{\beta}\phi_3}}}\right)\left(\frac{y_{i4}-\frac{e^{\boldsymbol{x}_{i4}^T\boldsymbol{\beta}}}{1+e^{\boldsymbol{x}_{i4}^T\boldsymbol{\beta}}}}{\left(1+e^{\boldsymbol{x}_{i4}^T\boldsymbol{\beta}}\right)^{-1}\sqrt{\frac{e^{\boldsymbol{x}_{i4}^T\boldsymbol{\beta}}}{1+e^{\beta}\phi_4}}}\right)\Bigg\}$$

where $\boldsymbol{\beta} = \left(\beta_1, \dots, \beta_p\right)^T$ and $\boldsymbol{\beta}_\phi = \left(\beta_{p+1}, \dots, \beta_{p+n}\right)^T$ and ρ is defined using either the CS or AR(1) structure.

Due to the complexity of the log-likelihood, we limited the derivations of the score equations and the Hessian matrix to the model that was fit under simulations (Section 3) and the clinical data (Section 4). In both Section 3 and Section 4 we fit a treatment by time interaction model with two treatments and four repeated measures. Therefore, $\boldsymbol{\beta} = (\beta_1, \dots, \beta_8)^T$ and $\boldsymbol{\beta}_\phi = (\beta_9, \dots, \beta_{12})^T$ with $\eta_{i1} = \beta_1 + \beta_5 x_{i1}, \dots, \eta_{i4} = \beta_4 + \beta_8 x_{i4}$ being the regression parameters for each repeated measure such that x_{ij} for $i = 1, \dots, N$ and $j = 1, \dots, 4$ is an indicator variable for treatment group. Furthermore, $\xi_{i1} = \beta_9, \dots, \xi_{i4} = \beta_{12}$ for $i = 1, \dots, N$ are the nuisance parameters for the precision of each repeated-measure. The score equations for the treatment by time interaction model are as follows:

$$U_k = \frac{\partial l}{\partial \beta_k} = \sum_{i=1}^N \Bigg[-\psi\left(\frac{e^{\beta\phi_k+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\left(\frac{e^{\beta\phi_k+\beta_k+\beta_{k+4}x_{ik}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^2}\right)+$$

$$\psi\left(\frac{e^{\beta\phi_k}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\left(\frac{e^{\beta\phi_k+\beta_k+\beta_{k+4}x_{ik}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^2}\right)+\left(\frac{e^{\beta\phi_k+\beta_k+\beta_{k+4}x_{ik}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^2}\right)\log\left(\frac{y_{ik}}{1-y_{ik}}\right)+\frac{\frac{\partial R}{\partial \beta_k}}{R}\Bigg];$$

for $k = 1, \dots, 4$

where

$$R = 1 + \rho_{1,2}\left(\frac{y_{i1}-\frac{e^{\beta_1+\beta_5 x_{i1}}}{1+e^{\beta_1+\beta_5 x_{i1}}}}{\left(1+e^{\beta_1+\beta_5 x_{i1}}\right)^{-1}\sqrt{\frac{e^{\beta_1+\beta_5 x_{i1}}}{1+e^{\beta}\phi_1}}}\right)\left(\frac{y_{i2}-\frac{e^{\beta_2+\beta_6 x_{i2}}}{1+e^{\beta_2+\beta_6 x_{i2}}}}{\left(1+e^{\beta_2+\beta_6 x_{i2}}\right)^{-1}\sqrt{\frac{e^{\beta_2+\beta_6 x_{i2}}}{1+e^{\beta}\phi_2}}}\right)+\cdots+$$

$$\rho_{3,4}\left(\frac{y_{i3}-\frac{e^{\beta_3+\beta_7 x_{i3}}}{1+e^{\beta_3+\beta_7 x_{i3}}}}{\left(1+e^{\beta_3+\beta_7 x_{i3}}\right)^{-1}\sqrt{\frac{e^{\beta_3+\beta_7 x_{i3}}}{1+e^{\beta}\phi_3}}}\right)\left(\frac{y_{i4}-\frac{e^{\beta_4+\beta_8 x_{i4}}}{1+e^{\beta_4+\beta_8 x_{i4}}}}{\left(1+e^{\beta_4+\beta_8 x_{i4}}\right)^{-1}\sqrt{\frac{e^{\beta_4+\beta_8 x_{i4}}}{1+e^{\beta}\phi_4}}}\right)+$$

$$\rho_{1,2,3}\left(\frac{y_{i1}-\frac{e^{\beta_1+\beta_5 x_{i1}}}{1+e^{\beta_1+\beta_5 x_{i1}}}}{\left(1+e^{\beta_1+\beta_5 x_{i1}}\right)^{-1}\sqrt{\frac{e^{\beta_1+\beta_5 x_{i1}}}{1+e^{\beta}\phi_1}}}\right)\left(\frac{y_{i2}-\frac{e^{\beta_2+\beta_6 x_{i2}}}{1+e^{\beta_2+\beta_6 x_{i2}}}}{\left(1+e^{\beta_2+\beta_6 x_{i2}}\right)^{-1}\sqrt{\frac{e^{\beta_2+\beta_6 x_{i2}}}{1+e^{\beta}\phi_2}}}\right)\left(\frac{y_{i3}-\frac{e^{\beta_3+\beta_7 x_{i3}}}{1+e^{\beta_3+\beta_7 x_{i3}}}}{\left(1+e^{\beta_3+\beta_7 x_{i3}}\right)^{-1}\sqrt{\frac{e^{\beta_3+\beta_7 x_{i3}}}{1+e^{\beta}\phi_3}}}\right)+$$

$$\cdots+$$

$$\rho_{2,3,4}\left(\frac{y_{i2}-\frac{e^{\beta_2+\beta_6 x_{i2}}}{1+e^{\beta_2+\beta_6 x_{i2}}}}{\left(1+e^{\beta_2+\beta_6 x_{i2}}\right)^{-1}\sqrt{\frac{e^{\beta_2+\beta_6 x_{i2}}}{1+e^{\beta}\phi_2}}}\right)\left(\frac{y_{i3}-\frac{e^{\beta_3+\beta_7 x_{i3}}}{1+e^{\beta_3+\beta_7 x_{i3}}}}{\left(1+e^{\beta_3+\beta_7 x_{i3}}\right)^{-1}\sqrt{\frac{e^{\beta_3+\beta_7 x_{i3}}}{1+e^{\beta}\phi_3}}}\right)\left(\frac{y_{i4}-\frac{e^{\beta_4+\beta_8 x_{i4}}}{1+e^{\beta_4+\beta_8 x_{i4}}}}{\left(1+e^{\beta_4+\beta_8 x_{i4}}\right)^{-1}\sqrt{\frac{e^{\beta_4+\beta_8 x_{i4}}}{1+e^{\beta}\phi_4}}}\right)+$$

$$\rho_{1,2,3,4}\left(\frac{y_{i1}-\frac{e^{\beta_1+\beta_5 x_{i1}}}{1+e^{\beta_1+\beta_5 x_{i1}}}}{\left(1+e^{\beta_1+\beta_5 x_{i1}}\right)^{-1}\sqrt{\frac{e^{\beta_1+\beta_5 x_{i1}}}{1+e^{\beta}\phi_1}}}\right)\left(\frac{y_{i2}-\frac{e^{\beta_2+\beta_6 x_{i2}}}{1+e^{\beta_2+\beta_6 x_{i2}}}}{\left(1+e^{\beta_2+\beta_6 x_{i2}}\right)^{-1}\sqrt{\frac{e^{\beta_2+\beta_6 x_{i2}}}{1+e^{\beta}\phi_2}}}\right)\left(\frac{y_{i3}-\frac{e^{\beta_3+\beta_7 x_{i3}}}{1+e^{\beta_3+\beta_7 x_{i3}}}}{\left(1+e^{\beta_3+\beta_7 x_{i3}}\right)^{-1}\sqrt{\frac{e^{\beta_3+\beta_7 x_{i3}}}{1+e^{\beta}\phi_3}}}\right)\times$$

$$\left(\frac{y_{i4}-\frac{e^{\beta_4+\beta_8 x_{i4}}}{1+e^{\beta_4+\beta_8 x_{i4}}}}{\left(1+e^{\beta_4+\beta_8 x_{i4}}\right)^{-1}\sqrt{\frac{e^{\beta_4+\beta_8 x_{i4}}}{1+e^{\beta}\phi_4}}}\right),$$

$$\frac{\partial}{\partial\beta_k}\left(\frac{y_{ik}-\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^{-1}\sqrt{\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta}\phi_k}}}\right)=\frac{1}{2}\frac{y_{ik}\left(e^{\beta_k+\beta_{k+4}x_{ik}-1}\right)-e^{\beta_k+\beta_{k+4}x_{ik}}}{\sqrt{\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta}\phi_k}}}, \qquad (2.17)$$

and $\psi(\cdot)$ is the digamma function. Note that by substituting (2.17) into R for the appropriate terms and dropping terms that do not contain β_k yields $\frac{\partial R}{\partial\beta_k}$.

$$U_{k+4} = \frac{\partial l}{\partial \beta_k} = \sum_{i=1}^{N} \left[-\psi\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^2}\right)x_{ik} + \right.$$

$$\psi\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^2}\right)x_{ik} + \left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^2}\right)\log\left(\frac{y_{ik}}{1-y_{ik}}\right)x_{ik} +$$

$$\left. \frac{\frac{\partial R}{\partial \beta_{k+4}}}{R} \right]; \qquad \text{for } k = 1, \dots, 4,$$

where R was previously defined, and

$$\frac{\partial}{\partial \beta_{k+4}}\left(\frac{y_{ik}-\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^{-1}\sqrt{\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_{\phi_k}}}}}\right) = \frac{1}{2}\frac{y_{ik}\left(e^{\beta_k+\beta_{k+4}x_{ik}}-1\right)-e^{\beta_k+\beta_{k+4}x_{ik}}}{\sqrt{\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_{\phi_k}}}}}x_{ik}$$

was used to derive $\frac{\partial R}{\partial \beta_{k+4}}$ as previously described.

$$U_{k+8} = \frac{\partial l}{\partial \beta_{\phi_k}} = \sum_{i=1}^{N} \left[\psi\left(e^{\beta_{\phi_k}}\right)e^{\beta_{\phi_k}} - \psi\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right) - \right.$$

$$\psi\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right) + \left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\log(y_{ik}) +$$

$$\left. \left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\log(1-y_{ik}) + \frac{\frac{\partial R}{\partial \beta_{\phi_k}}}{R} \right]; \qquad \text{for } k = 1, \dots, 4,$$

and

$$\frac{\partial}{\partial \beta_{\phi_k}}\left(\frac{y_{ik}-\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^{-1}\sqrt{\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_{\phi_k}}}}}\right)$$

$$= \frac{1}{2}\left(\frac{y_{ik}-\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^{-1}\sqrt{\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_{\phi_k}}}}}\right)\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_{\phi_k}}}\right)$$

was used to derive $\frac{\partial R}{\partial \beta_{\phi_k}}$ using the aforementioned procedure.

The MLE for the parameters (excluding the correlation parameter) were found by setting the score equations equal to zero and solving for the respective parameter. As with the LNMVB, there is no closed form solution for the MLEs. Before using the quasi-Newton-Raphson algorithm[62] (Section 2.3) to estimate the MLEs of the parameters, the correlation parameter was estimated using Methods of Moments.[68] Specifically, $\rho^2_{1,2,3,4} = 1 - \det(M)$ is solved for ρ where $\rho_{1,2,3,4}$ is defined using either the CS or AR(1) structure and M is the correlation matrix of the data. ρ is replaced by $\hat{\rho}$ in the score equations and the β's are estimated using the quasi-Newton-Raphson algorithm.[62] Similar to the estimation procedure of the LNMVB, the Hessian matrix was estimated using finite differences. The analytic entries of the Hessian matrix are given for completeness.

$$\frac{\partial U_k}{\partial \beta_k} = \sum_{i=1}^{N} \Bigg[\left(\frac{e^{\beta_{\phi_k} + \beta_k + \beta_{k+4} x_{ik}}}{\left(1 + e^{\beta_k + \beta_{k+4} x_{ik}}\right)^2} \right) \left\{ \psi_1 \left(\frac{e^{\beta_{\phi_k}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) - \psi_1 \left(\frac{e^{\beta_{\phi_k} + \beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) \right\} +$$

$$\left(\frac{e^{\beta_{\phi_k} + \beta_k + \beta_{k+4} x_{ik}} \left(1 - e^{\beta_k + \beta_{k+4} x_{ik}}\right)}{\left(1 + e^{\beta_k + \beta_{k+4} x_{ik}}\right)^3} \right) \left\{ \psi \left(\frac{e^{\beta_{\phi_k}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) - \psi \left(\frac{e^{\beta_{\phi_k} + \beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) + \right.$$

$$\left. \log \left(\frac{y_{ik}}{1 - y_{ik}} \right) \right\} + \frac{\frac{\partial^2 R}{\partial \beta_k^2} R - \left(\frac{\partial R}{\partial \beta_k} \right)^2}{R^2} \Bigg]; \qquad \text{for } k = 1, \dots, 4,$$

where $\psi(\cdot)$ is the digamma function, $\psi_1(\cdot)$ is the trigamma function, R and $\frac{\partial R}{\partial \beta_k}$ are previously defined (see score equations),

$$\frac{\partial^2}{\partial \beta_k^2} \left(\frac{y_{ik} - \frac{e^{\beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}}}{\left(1 + e^{\beta_k + \beta_{k+4} x_{ik}}\right)^{-1} \sqrt{\frac{e^{\beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_{\phi_k}}}}} \right) = \frac{\frac{1}{4} y_{ik} \left(e^{\beta_k + \beta_{k+4} x_{ik}} + 1\right) - \frac{1}{2} e^{\beta_k + \beta_{k+4} x_{ik}}}{\sqrt{\frac{e^{\beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_{\phi_k}}}}}, \qquad (2.18)$$

and $\frac{\partial^2 R}{\partial \beta_k^2}$ can be calculated by substituting (2.18) into R for the appropriate terms and dropping terms that do not contain β_k.

$$\frac{\partial U_k}{\partial \beta_{k+4}} = \frac{\partial U_k}{\partial \beta_k} x_{ik}; \qquad \text{for } k = 1, \dots, 4,$$

$$\frac{\partial U_k}{\partial \beta_{\phi_k}} = \sum_{i=1}^{N} \left[\left(\frac{e^{\beta_{\phi_k} + \beta_k + \beta_{k+4} x_{ik}}}{\left(1 + e^{\beta_k + \beta_{k+4} x_{ik}}\right)^2} \right) \left\{ \psi_1 \left(\frac{e^{\beta_{\phi_k}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) \left(\frac{e^{\beta_{\phi_k}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) + \right.\right.$$

$$\psi \left(\frac{e^{\beta_{\phi_k}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) - \psi_1 \left(\frac{e^{\beta_{\phi_k} + \beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) \left(\frac{e^{\beta_{\phi_k} + \beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) -$$

$$\left.\left. \psi \left(\frac{e^{\beta_{\phi_k} + \beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}} \right) \right\} + \frac{\frac{\partial^2 R}{\partial \beta_k \partial \beta_{\phi_k}} R - \frac{\partial R}{\partial \beta_{\phi_k}} \frac{\partial R}{\partial \beta_k}}{R^2} \right]; \qquad \text{for } k = 1, \dots, 4,$$

where $\frac{\partial R}{\partial \beta_{\phi_k}}$ is previously defined,

$$\frac{\partial^2}{\partial \beta_k \partial \beta_{\phi_k}} \left(\frac{y_{ik} - \frac{e^{\beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_k + \beta_{k+4} x_{ik}}}}{\left(1 + e^{\beta_k + \beta_{k+4} x_{ik}}\right)^{-1} \sqrt{\frac{e^{\beta_k + \beta_{k+4} x_{ik}}}{1 + e^{\beta_{\phi_k}}}}} \right) = \frac{\frac{1}{2} y_{ik} \left(e^{\beta_k + \beta_{k+4} x_{ik}} - 1\right) - e^{\beta_k + \beta_{k+4} x_{ik}}}{\sqrt{e^{\beta_k + \beta_{k+4} x_{ik}}} \sqrt{1 + e^{\beta_{\phi_k}}}} \left(\frac{1}{2} e^{\beta_{\phi_k}} \right),$$

and $\frac{\partial^2 R}{\partial \beta_k \partial \beta_{\phi_k}}$ can be calculated using the aforementioned procedure.

$$\frac{\partial U_k}{\partial \beta_r} = \frac{\frac{\partial^2 R}{\partial \beta_k \partial \beta_r} R - \frac{\partial R}{\partial \beta_k} \frac{\partial R}{\partial \beta_r}}{R^2}; \quad \text{for } r \neq k, r \neq k + 4; r \neq \phi_k,$$

where $\frac{\partial^2 R}{\partial \beta_k \partial \beta_r}$ can be calculated by substituting $\frac{\partial R}{\partial \beta_r}$ into $\frac{\partial R}{\partial \beta_k}$ for the appropriate terms and dropping terms that do not contain β_r.

$$\frac{\partial U_{k+4}}{\partial \beta_k} = \sum_{i=1}^{N}\left[\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^2}\right)\left\{\psi_1\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)-\psi_1\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\right\}x_{ik}+\right.$$

$$\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}\left(1-e^{\beta_k+\beta_{k+4}x_{ik}}\right)}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^3}\right)\left\{\psi\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)-\psi\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)+\right.$$

$$\left.\left.\log\left(\frac{y_{ik}}{1-y_{ik}}\right)\right\}x_{ik}+\frac{\frac{\partial^2 R}{\partial \beta_{k+4}\partial \beta_k}R-\frac{\partial R}{\partial \beta_{k+4}}\frac{\partial R}{\partial \beta_k}}{R^2}\right];\ \text{for } k=1,\dots,4,$$

where $\frac{\partial R}{\partial \beta_{k+4}}$ was calculated for the score equations and $\frac{\partial^2 R}{\partial \beta_{k+4}\partial \beta_k}=\frac{\partial^2 R}{\partial \beta_k \partial \beta_{k+4}}$.

$$\frac{\partial U_{k+4}}{\partial \beta_{k+4}} = \frac{\partial U_{k+4}}{\partial \beta_k}x_{ik}; \qquad \text{for } k=1,\dots,4.$$

$$\frac{\partial U_{k+4}}{\partial \beta_{\phi_k}} = \frac{\partial U_k}{\partial \beta_{\phi_k}}x_{ik}; \qquad \text{for } k=1,\dots 4.$$

$$\frac{\partial U_{k+4}}{\partial \beta_r} = \frac{\partial U_k}{\partial \beta_r}x_{ik}; \qquad \text{for } r \neq k, r \neq k+4; r \neq \phi_k.$$

$$\frac{\partial U_{k+8}}{\partial \beta_k} = \sum_{i=1}^{N}\left[\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^2}\right)\left\{\psi_1\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)+\right.\right.$$

$$\psi\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)-\psi_1\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)-$$

$$\left.\left.\psi\left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}\right)+\log\left(\frac{y_{ik}}{1-y_{ik}}\right)\right\}+\frac{\frac{\partial^2 R}{\partial \beta_{\phi_k}\partial \beta_k}R-\frac{\partial R}{\partial \beta_{\phi_k}}\frac{\partial R}{\partial \beta_k}}{R^2}\right]; \qquad \text{for } k=1,\dots,4,$$

$$\frac{\partial^2}{\partial \beta_{\phi_k}\partial \beta_k}\left(\frac{y_{ik}-\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^{-1}\sqrt{\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_{\phi_k}}}}}\right)=\frac{\frac{1}{4}y_{ik}\left(e^{\beta_k+\beta_{k+4}x_{ik}-1}\right)-\frac{1}{2}e^{\beta_k+\beta_{k+4}x_{ik}}}{\sqrt{\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_{\phi_k}}}}}\left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_{\phi_k}}}\right),$$

and $\frac{\partial^2 R}{\partial \beta_{\phi_k}\partial \beta_k}$ can be calculated as previously described.

$$\frac{\partial U_{k+8}}{\partial \beta_{k+4}} = \frac{\partial U_{k+8}}{\partial \beta_k}x_{ik}; \qquad \text{for } k=1,\dots,4.$$

$$\frac{\partial U_{k+8}}{\partial \beta_{\phi_k}} = \sum_{i=1}^{N} \Bigg[e^{\beta_{\phi_k}} \{ \psi_1 (e^{\beta_{\phi_k}}) e^{\beta_{\phi_k}} + \psi (e^{\beta_{\phi_k}}) \} - \left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) \left\{ \psi_1 \left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) \left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) + \psi \left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) \right\} - \left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) \left\{ \psi \left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) \left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) + \psi \left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) \right\} + \left(\frac{e^{\beta_{\phi_k}+\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) \log(y_{ik}) + \left(\frac{e^{\beta_{\phi_k}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}} \right) \log(1 - y_{ik}) + \frac{\frac{\partial^2 R}{\partial \beta_{\phi_k}^2} - \left(\frac{\partial R}{\partial \beta_{\phi_k}} \right)^2}{R^2} \Bigg]; \quad \text{for } k = 1, \dots, 4,$$

$$\frac{\partial^2}{\partial \beta_{\phi_k}^2} \left(\frac{y_{ik} - \frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^{-1} \sqrt{\frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_{\phi_k}}}}} \right) = \frac{1}{2} \left(\frac{y_{ik} - \frac{e^{\beta_k+\beta_{k+4}x_{ik}}}{1+e^{\beta_k+\beta_{k+4}x_{ik}}}}{\left(1+e^{\beta_k+\beta_{k+4}x_{ik}}\right)^{-1} \sqrt{e^{\beta_k+\beta_{k+4}x_{ik}}}} \right) \left(\frac{e^{\beta_{\phi_k}} \left(1+\frac{1}{2} e^{\beta_{\phi_k}}\right)}{\left(1+e^{\beta_{\phi_k}}\right)^{\frac{3}{2}}} \right),$$

and $\frac{\partial^2 R}{\partial \beta_{\phi_k}^2}$ can be determined using formerly described procedure.

$$\frac{\partial U_{k+8}}{\partial \beta_r} = \frac{\frac{\partial^2 R}{\partial \beta_{\phi_k} \partial \beta_r} R - \frac{\partial R}{\partial \beta_{\phi_k}} \frac{\partial R}{\partial \beta_r}}{R^2}; \qquad \text{for } r \neq k, r \neq k+4; r \neq \phi_k,$$

where $\frac{\partial^2 R}{\partial \beta_{\phi_k} \partial \beta_r}$ can be calculated by substituting $\frac{\partial R}{\partial \beta_r}$ into $\frac{\partial R}{\partial \beta_{\phi_k}}$ for the appropriate terms and dropping terms that do not contain β_r. Similar to the LNMVB regression model, we used the inverse of the observed Fisher information matrix as variance-covariance estimates of the regression parameters.

Due to the complexity of the copula, determining the maximum obtainable correlation can be established using linear programming. Specifically, we can maximize the objective function, ρ, with the following constraints:

$$1 + R_{\phi_1,\dots,\phi_n,\Omega_n}(x_1, \dots, x_n) \geq 0$$

$$x_i \in (0,1)$$

$$a_i, b_i > 0,\ \rho \geq 0; \qquad \text{for } i = 1, \dots, n.$$

In the function $R_{\phi_1,\dots,\phi_n,\Omega_n}(x_1, \dots, x_n)$, the $\omega's$ are parametrized in terms of ρ using equations (2.14) and (2.15), $\mu_i = \frac{\alpha_i}{\alpha_i+\beta_i}$, and $\sigma_i^2 = \frac{\alpha_i\beta_i}{(\alpha_i+\beta_i)^2(\alpha_i+\beta_i+1)}$. Under these constraints, both the CS and AR(1) structures have a correlation range of $[0,1]$. The maximum correlation can be obtained either as $\alpha_i, \beta_i \to 0$ or $\alpha_i, \beta_i \to \infty$ for $i = 1, \dots, n$ using both correlation structures. Unfortunately, negative correlations were not possible because of the restriction imposed by the MCC. The above described linear programming problem was implemented in R[64] using package NLOPTR[69] using the Improved Stochastic Ranking Evolution Strategy algorithm.

In this Section, we have developed two models for Beta distributed repeated measures data whose regression parameters can be estimated using the maximum likelihood estimation method. We established that both the LNMVB and the SLMVB are limited to positive correlations. However, the SLMVB's likelihood is more complicated than the likelihood of the LNMVB, and the SLMVB requires additional parameters compared to the LNMVB. In the next Section, we determined the performance of these two models and compared their performance to the alternatives, i.e., the Beta GLMM and the Beta GEE.

3. Simulation Study

3.1. Introduction

Simulations were completed to establish and compare the performance of our proposed models (i.e., LNMVB and SLMVB) to the Beta GEE and the Beta GLMM in the case of correlated outcomes. We studied the effects of varying sample sizes, the strength of correlation amongst repeated measures, the correlation structure, location parameter, and the number of treatment groups. The type I error, power, 95% coverage probabilities, the percent samples for which convergence was reached, mean bias of the location and correlation parameters, and root mean squared deviation (RMSD) were used to quantify the behavior of the methods.

3.2. Design of simulation study

Two types of simulations were performed; we examined a single group (i.e., time effect only) and we examined two groups (i.e., a treatment by time interaction). We begin by describing the single group simulation. Assuming a balanced design with subjects being measured at evenly spaced fixed intervals, we generated data for $N = 15, 30, 50,$ and 100 subjects, using the following model:

$$\log\left(\frac{\mu_{ij}}{1-\mu_{ij}}\right) = \beta_1 + \beta_2 * time_{i2} + \beta_3 * time_{i3} + \beta_4 * time_{i4} \qquad (3.1a)$$

$$Y_{ij} \sim Beta(\mu_{ij}, \phi) \qquad (3.1b)$$

for subjects $i = 1, \dots, N$ and measurements $j = 1, \dots, 4$ such that the Beta density (3.1b) was parameterized using the form in (1.4). We allowed the mean to vary among subjects and measurements while holding the standard deviation fixed at 0.01. The parameter β_1 was varied by simulation such that $\mu_{i1} = 0.05, 0.3,$ or 0.5 (for brevity; see limitations for $\mu \to 1$) for $i = 1, \dots, N$. The parameters β_2 and β_3 were set to zero to keep the first three repeated measures stationary and the parameter β_4 was adjusted for desired effect size (adapting Cohen's[70] definition of effect size for repeated-measures) to establish type I error and power. Effect sizes ranged from 0 to 1 using an interval of 0.1. Measurements on the same subject were correlated. We varied the strength of correlation according to Cohen's[70] recommendation, i.e., small $= 0.1$, medium $= 0.3$, and large $= 0.5$. Additionally, we used two different correlation structures, AR(1) and CS.

For the two group simulation, we modified equation (3.1a) to include a treatment group. Specifically,

$$\log\left(\frac{\mu_{ij}}{1-\mu_{ij}}\right) = \beta_1 + \beta_2 * time_{i2} + \beta_3 * time_{i3} + \beta_4 * time_{i4} + \beta_5 * trt_i + \beta_6 * trt_i * time_{i2} + \beta_7 * trt_i * time_{i3} + \beta_8 * trt_i * time_{i4} \tag{3.2}$$

for subjects $i = 1, \dots, N$ and measurements $j = 1, \dots, 4$ where $N = 12, 30, 50,$ and 100 equally distributed across the two treatment groups ($trt_i = 0$ or 1). The parameter β_1 was defined as described above, the parameters $\beta_2, \dots, \beta_7$ were set to 0 (i.e., stationary means across repeated measures and treatment groups), and the parameter β_8 was adjusted for desired effect sizes (between groups) as previously described. One thousand replicates were performed using all possible combinations of the parameters (see Table 3.1).

Table 3.1 Simulation parameters.

# of treatment groups	Correlation structures	Stationary mu	ρ	Effect sizes	Total N
1	AR(1), CS	0.05, 0.3, 0.5	0.1, 0.3, 0.5	0, 0.1, …, 1	15, 30, 50, 100
2	AR(1), CS	0.05, 0.3, 0.5	0.1, 0.3, 0.5	0, 0.1, …, 1	12, 30, 50, 100

3.3. Simulation of data

To generate dependent data, correlated Gaussian data were transformed to Beta distributed data. First, a target correlation structure was specified in matrix form such that the entries of the matrix are ρ_{ij} for $i, j = 1, \dots, 4$. Using the Nataf model,[22] the correlation coefficient ρ_{ij} of each pair (i, j) of the Beta random variables were adjusted to form the correlation coefficient $\tilde{\rho}_{ij}$ of a pair of Gaussian random variables. Using a non-linear solver (package nleqslv[65]), for each $\tilde{\rho}_{ij}$ the following equation was solved:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \frac{H_i - \mu_i}{\sigma_i} \frac{H_j - \mu_j}{\sigma_j} \varphi\left(\widetilde{H}_i, \widetilde{H}_j, \tilde{\rho}_{ij}\right) d\widetilde{H}_i d\widetilde{H}_j - \rho_{ij} = 0; \qquad \text{for } i, j = 1, \dots 4$$

where the values of the Beta random variables H_i, H_j (with means μ_i, μ_j and standard deviations σ_i, σ_j) are expressed in terms of standard Gaussian variables, i.e., $H = G^{-1}[\Phi(\widetilde{H})]$ such that $G^{-1}(\cdot)$ is the inverse Beta density, $\Phi(\cdot)$ is the standard Normal cumulative density function, and $\varphi(\cdot)$ is the standard bivariate Normal density.[22] Next, a Cholesky factorization was performed on the correlation matrix, $\boldsymbol{C} = \boldsymbol{S}\boldsymbol{S}'$, whose entries are $\tilde{\rho}_{ij}$. Furthermore, $4N$ independent standard Normal random variables were generated, denoted $\widetilde{\boldsymbol{Y}}$, such that $\widetilde{\boldsymbol{Y}}$ is a matrix with N rows and 4 columns. Dependent data, $\widetilde{\boldsymbol{Y}}'$, were generated using the Cholesky factorization and standard Normal random variables, i.e., $\widetilde{\boldsymbol{Y}}' = \boldsymbol{S}\widetilde{\boldsymbol{Y}}$.[71] Lastly, $\widetilde{\boldsymbol{Y}}'$ were transformed to Beta random variables using $Y_{ij} = G^{-1}[\Phi(\tilde{Y}_{ij}{}')]$ for $i = 1, \dots, N$ and $j = 1, \dots, 4$. Data were generated separately for each treatment group.

3.4. Model fitting

Model fitting was performed using R[64] version 3.3 on a Linux high performance computer. Results were then compiled using R[64] version 3.4.2 on a Windows 10 PC. Algorithms and R[64] packages used to fit the LNMVB and SLMVB are described in detail in Section 2.3 and 2.4, respectively. The R[64] package geeM[72] was used to fit the Beta GEE and the R[64] package GLMMadaptive[73] was used to fit the Beta GLMM. The link function, inverse link function, inverse link function derivative, and variance function were user specified for the Beta GEE. These functions were established assuming a logit link function for the mean response. Default settings were used for the geem and mixed_model procedures from the package geeM[72] and GLMMadaptive[73], respectively. By default the geem procedure calculates robust standard errors and the mixed_model procedure uses 11 quadrature points to estimate the integral during model estimation. Additionally, a user defined family was specified for the mixed_model procedure. For the user defined family, the Beta density was parameterized using equation (1.4), and a logit link and log link were used for the parameters μ and ϕ, respectively.

3.5. Metrics

Type I error and power are established using the F-test. Specifically, the F-statistic is calculated as:

$$F = \frac{(\boldsymbol{L\hat{\beta}})'(\boldsymbol{L\hat{\Sigma}L})^{-1}(\boldsymbol{L\hat{\beta}})}{df},$$

where $\hat{\boldsymbol{\beta}}$ is a vector of the estimated regression parameters, $\hat{\boldsymbol{\Sigma}}$ is the estimated variance-covariance matrix, $\boldsymbol{L}$ is a contrast that corresponds to the appropriate hypothesis, and df are the degrees of freedom. Under the model framework (3.1a),

$$\boldsymbol{L} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

which corresponds to the null hypothesis $\beta_2 = \beta_3 = \beta_4 = 0$, i.e., there is no time effect. For the model (3.2)

$$\boldsymbol{L} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

which tests the hypothesis $\beta_6 = \beta_7 = \beta_8 = 0$, i.e., there is no $treatment\ x\ time$ interaction. The F-statistic was then compared to the F-distribution with 3 numerator degrees of freedom (i.e., number of repeated measures minus one times number of groups minus one (if more than one group)) and $(N - G)(j - 1)$ denominator degrees of freedom where N is the number of subjects, G is the number of groups, and j is the number of repeated measurements per subject. Type I error is the percent of F-tests < α-level when the effect size is zero. Similarly, power is the percent of F-tests < α-level for the respective effect size. If $\widehat{\boldsymbol{\Sigma}}$ was singular, then we considered the F-test > α-level.

Coverage probabilities and mean length of confidence intervals were calculated using Wald-type confidence intervals. Confidence intervals were constructed for each group's μ_j $(j = 1, \dots, 4)$ as follows:

$$\boldsymbol{L}_k \widehat{\boldsymbol{\beta}} \pm t_{1-\frac{\alpha}{2}, df} \left(\boldsymbol{L}_k \widehat{\boldsymbol{\Sigma}} \boldsymbol{L}_k{}' \right)^{\frac{1}{2}},$$

for $k = 1, \dots, 4$ (one treatment group) or $k = 1, \dots, 8$ (two treatment groups)

where t_p is the $100p^{th}$ percentile of the standard t distribution with $df = (N - G)(j - 1)$ degrees of freedom and L_k is the contrast for the respective μ_j. The confidence intervals were then back transformed to the data scale using the inverse logit function. The coverage probabilities are the percent of expected μ's that are within their respective confidence intervals. If $\widehat{\boldsymbol{\Sigma}}$ was singular, then we considered that the confidence interval did not cover the expected $\mu(s)$ for that replicate. Furthermore, percentage of missing standard errors (SEs) were calculated.

Bias and RMSD were calculated on the data scale. When calculating the bias and RMSD of the location parameter, metrics were calculated separately for each group and time point. Mean bias was calculated as

$$\frac{\sum_{i=1}^{1000} \hat{\theta}_i - \theta_i}{1000},$$

where $\hat{\theta}_i$ is the estimated parameter and θ_i is the expected value for the ith replicate. RMSD was estimated as

$$\sqrt{\frac{\sum_{i=1}^{1000}\left(\hat{\theta}_i-\theta_i\right)^2}{1000}}.$$

Both the LNMVB and GLMM required additional calculations to estimate $\hat{\theta}$ when calculating the bias of the correlation. Estimated pairwise correlations were calculated for the LNMVB as described in Section 2.3. The parameter estimate, $\hat{\theta}$, was then estimated as the average of these pairwise estimated correlations. According to Nakagawa and colleagues[74] the correlation of a GLMM can be estimated as

$$\rho = \frac{\sigma_b^2}{\sigma_b^2+\sigma_\varepsilon^2}$$

where σ_b^2 is the variance of the random effect and σ_ε^2 is the variance of the model errors. Assuming the errors are Beta distributed and the Beta density is parametrized using equation (1.4), then using the delta method σ_ε^2 is calculated as

$$\left[\frac{1-2\mu}{\mu(1-\mu)}\left(1-\frac{\mu(1-\mu)}{1+\phi}\right)\right]^{-2}\left(\frac{\mu(1-\mu)}{1+\phi}\right)$$

such that μ and ϕ are back transformed to the data scale using the inverse logit function and exponential function, respectively.

We assumed an α-level of 0.05 for all statistical tests. When reporting results we focused on small (i.e., 0.2), medium (i.e., 0.5), and large (i.e., 0.8) effect sizes. Full results (excluding bias of correlation parameter) for one group with AR(1) correlation structure are reported in Appendix A. Full results (excluding bias of correlation parameter) for two groups with AR(1) correlation structure are reported in Appendix B. Results are not reported for bias of correlation parameter in appendices because Section 3.6 sufficiently describes in detail said results. For the two-group simulation, results are reported only for the group with non-stationary location parameter in Appendix B. There appeared to be little difference in the pattern of results between the groups in the two-group simulation. For brevity, results for CS correlation structure are not included in the appendices since the correlation structure did not appear to have a substantial impact on the results. Full results can be obtained by contacting the author. For convenience and to aid in explanation, Figures A.1 through A.3 of Appendix A are reported in text of Section 3.6.1 as Figures 3.1 through 3.3.

3.6. Results

3.6.1. One group: $\mu = 0.05$, AR(1) correlation

[refer to Figures A.1 through A.12]

When the stationary mean was 0.05, and the correlation was 0.1, the LNMVB had a Type I error rate near 5% across all the sample sizes (see Figure 3.1 or Figure A.1). The Beta GLMM had an inflated Type I error when the sample size was $N = 15$ (i.e., 7.4%); however, for the sample sizes $N = 30, 50,$ and 100 the Type I error and power curves were similar to those of the LNMVB though slightly inflated in comparison (Figure 3.1 or Figure A.1). The Beta GEE did not achieve nominal Type I error and had a Type I error rate $> 10\%$ when the sample size was small, i.e., $N = 15$ (Figure 3.1 or Figure A.1). The SLMVB models never achieved nominal Type I error. For the sample size $N = 15$, the SLMVBs' Type I error was $> 20\%$ and increased as N increased for both the SLMVB CS and SLMVB AR1 (Figure 3.1 or Figure A.1).

For the correlations $\rho = 0.3$ and 0.5, both the Beta GLMM and Beta GEE had inflated Type I error rates when the sample size was small (i.e., $N \leq 30$). However, as the sample size increased, the Type I error rate approached 5% with the Beta GEE Type I error rate being more inflated than that of the Beta GLMM. When $\rho = 0.5$, neither the

Beta GLMM or Beta GEE achieved a nominal Type I error rate. The LNMVB had a Type I error rate $< 5\%$ across all sample sizes for $\rho = 0.3$ or 0.5, and the SLMVB models had highly inflated Type I error rates displaying the same trend as when the correlation was 0.1.

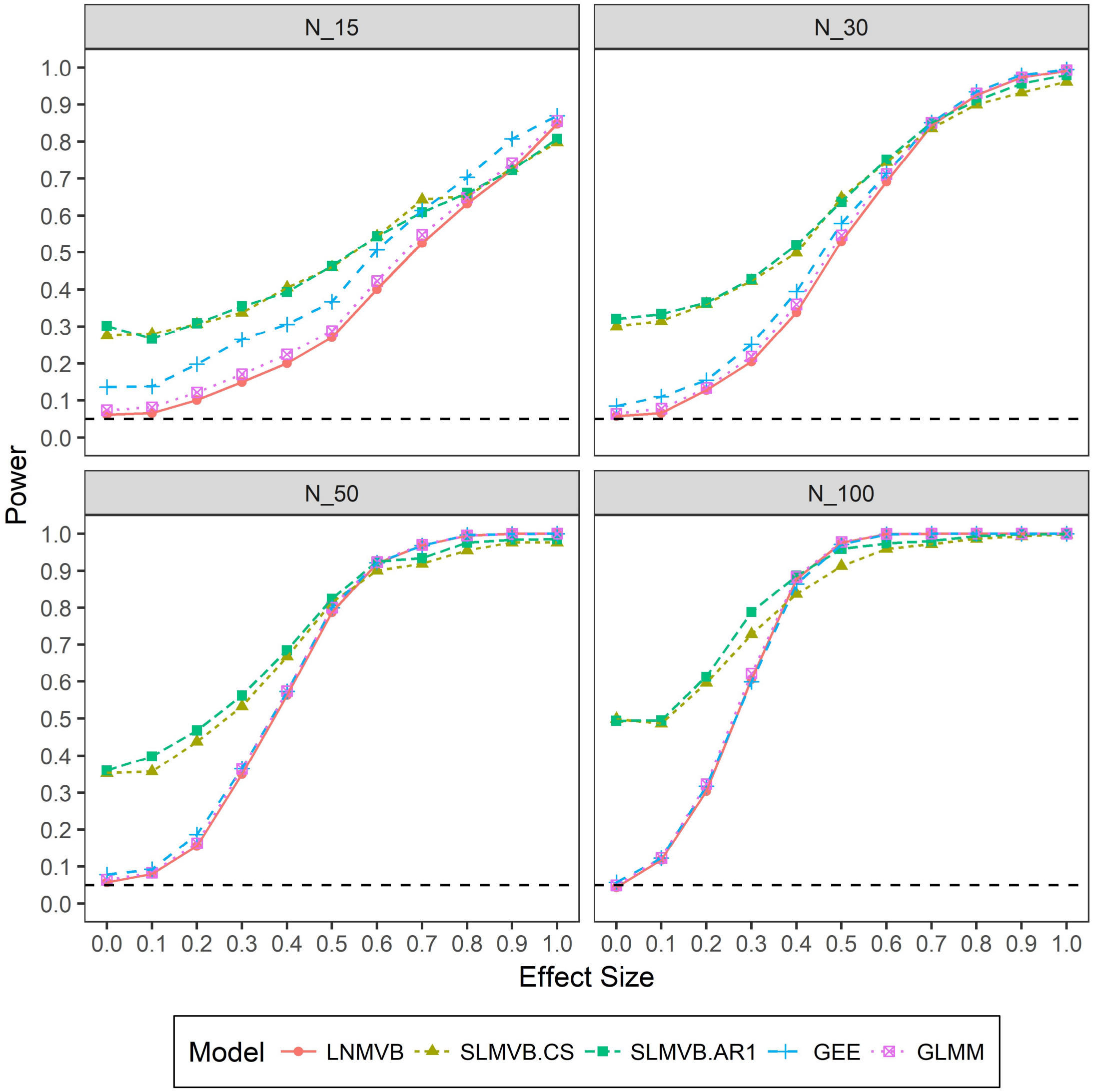


Figure 3.1 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

When the strength of correlation was 0.1, 0.3, or 0.5, the LNMVB, Beta GLMM, and Beta GEE displayed similar coverage probabilities (see Figure 3.2 or Figure A.2 for coverage probabilities when the correlation was small). Specifically, when the sample size N was small, the coverage probabilities were slightly $< 95\%$ and as $N \rightarrow 100$ the coverage probabilities approached 95%. For the SLMVB models, the coverage probabilities were $< 90\%$ for all combinations of correlations and sample sizes (Figure 3.2 or Figure A.2 excludes data points whose coverage probabilities were $< 90\%$).

The mean bias and the RMSD of the location parameter were near zero for the LNMVB, Beta GLMM, and Beta GEE across all correlations and sample sizes, while the SLMVB models had a mean bias and RMSD of the location parameter further from zero than the other three models. In many cases, the mean bias of the location parameter of the SLMVB models were > 0.02 which is at least 2 standard deviations from the mean. In Figure 3.3 (small correlation), the increased bias is most evident when the sample size was $N = 15$. Figure A.4 of Appendix A (RMSD, small correlation) displays the same general pattern as Figure 3.3 or Figure A.3; however, the scales between the mean bias and the RMSD are different.

The Beta GEE's mean bias of the correlation parameter was consistently near zero, while the Beta GLMM slightly overestimated the correlation parameter when $\rho = 0.1$ and underestimated the correlation parameter when the correlation was $\rho = 0.3$ or 0.5. The SLMVB models overestimated the correlation parameter with the amount of overestimation decreasing as ρ increased. The correlation parameter was not able to be estimated under the LNMVB model since the integral could not be approximated.

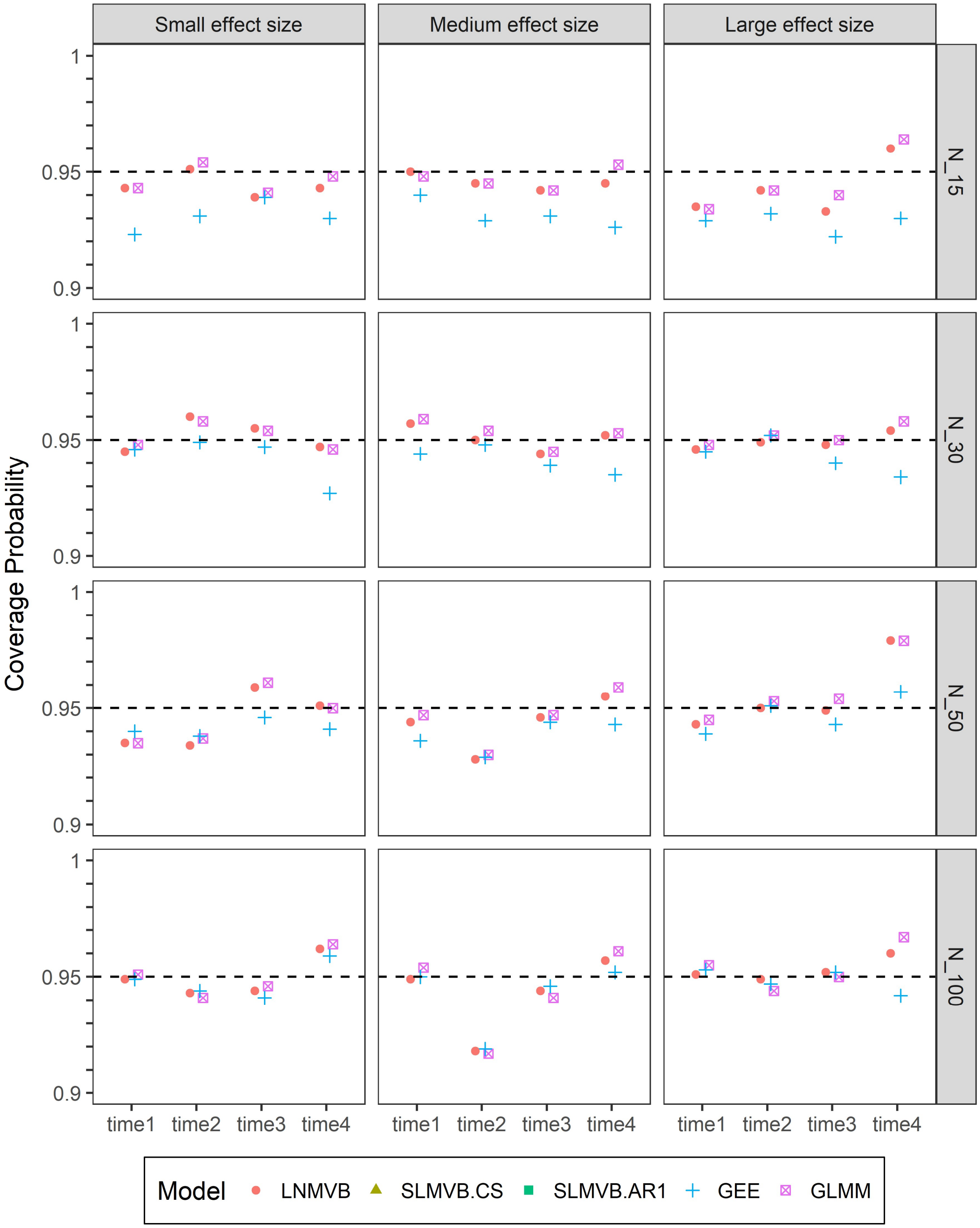


Figure 3.2 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

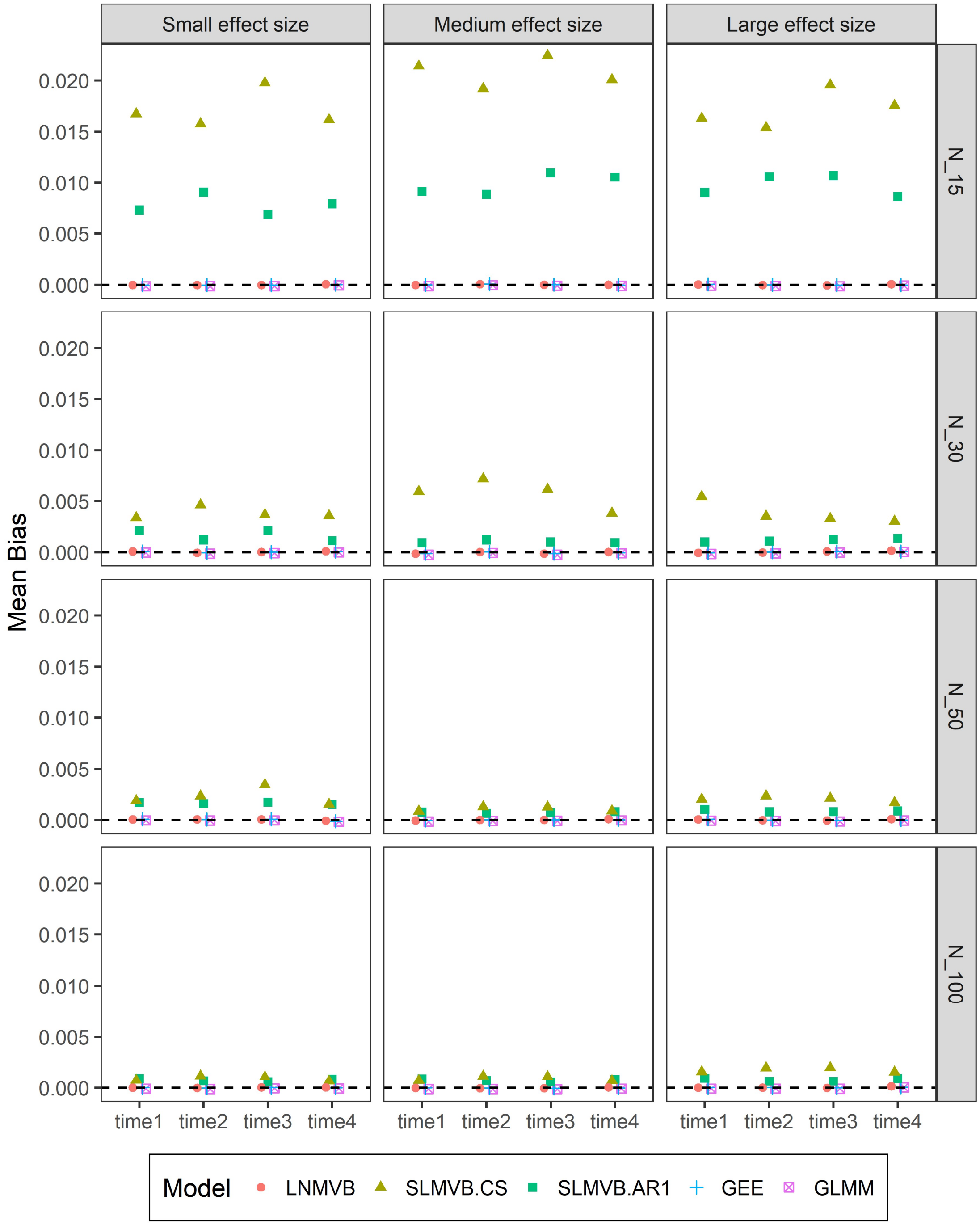


Figure 3.3 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

To summarize, the LNMVB had Type I error rates near or below 5%, coverage probabilities that approached 95% as the sample size N increased, and unbiased estimation of the location parameter. However, the correlation parameter could not be estimated. The SLMVB models had markedly inflated Type I error rates, coverage probabilities $< 90\%$, and biased estimates of the location and correlation parameters. Furthermore, the SLMVB models had convergence issues that worsened as the strength of correlation increased. The Beta GLMM performed similarly to the LNMVB, except the Beta GLMM had an inflated Type I error rate under certain conditions, and the Beta GLMM was better able to estimate the correlation parameter. Lastly, the Beta GEE had mean bias near zero for both the location and correlation parameter and coverage probabilities that approached 95% as sample size increased. However, the Beta GEE had the most inflation of the Type I error rate when compared to the LNMVB and Beta GLMM.

3.6.2. One group: $\mu = 0.3$, AR(1) correlation
[refer to Figures A.13 though A.24]

With a stationary mean of 0.3, the Beta GLMM had a Type I error rate between 5% and 8.1% for all correlations and sample sizes with the Type I error decreasing as the sample size increased. The SLMVB models had inflated Type I error rates that increased as sample size increased, regardless of the strength of the correlation. Additionally, the power of the SLMVB models was the lowest of all the models. The Beta GEE had an inflated Type I error rate across all the correlations and the sample sizes, with the amount of inflation decreasing as the sample size increased. For example, when the sample size was $N = 15$, the Beta GEE had a Type I error rate $> 10\%$ that decreased to $\approx 6.5\%$ across all correlations. The LNMVB's Type I error decreased as the sample size N increased. The power curves of the LNMVB model were similar to that of the Beta GLMM. For correlations 0.1 and 0.3, the LNMVB's Type I error was slightly inflated compared to the Beta GLMM, and for correlation 0.5 the Beta GLMM's Type I error was slightly inflated compared to the LNMVB.

The Beta GLMM and SLMVB models consistently had coverage probabilities $< 90\%$. The Beta GEE tended to underestimate the 95% coverage probabilities. The amount of underestimation tended to decrease as the sample size N increased. Specifically, for $N = 15$, the minimum coverage probability across repeated measures was approximately 91%, and for $N = 100$ the minimum coverage probability across repeated measures was about 93%. The LNMVB tended to overestimate coverage probabilities for the small and the medium correlations, regardless of the sample size. For $\rho = 0.5$ the LNMVB coverage probabilities were similar to that of the Beta GEE coverage probabilities, i.e., a slight underestimation that approached 95% as the sample size increased.

The LNMVB and Beta GEE had both a mean bias and a RMSD of the location parameter near zero for all correlations and sample sizes. The Beta GLMM overestimated the true value of the location parameter often by 0.02 or more (i.e., by 2 or more standard deviations). The SLMVB models mean bias of the location parameter was near zero for correlations $\rho = 0.1$ and 0.3; however, for the correlation $\rho = 0.5$ and the sample sizes ≤ 30 the SLMVB models produced biased estimates of the location parameter. Furthermore, unbiased estimates of the location parameter did not always result in a RMSD near 0. In the cases of biased estimates under the SLMVB models, the bias was < 0.02.

Only the Beta GEE's estimates of the correlation parameter were consistently near the true value. The SLMVB models' mean bias of the correlation parameter approached 0 as N increased with the SLMVB CS model slightly underestimating the true value for the sample size $N = 100$. The Beta GLMM significantly overestimated the correlation parameter, and the LNMVB did not produce valid estimates.

In summary, the LNMVB had Type I error rates that approached the nominal Type I error rate (but never reached 5%) as the sample size increased. Additionally, the mean bias and the RMSD of the location parameter were near zero; however, the LNMVB was not able to estimate the correlation parameter. The SLMVB models had Type I error rates that increased as N increased; however, this did not result in an increase in power. The coverage probabilities were consistently $< 90\%$ with some estimates of the location and correlation parameters being biased. Again, the SLMVB models had convergence issues that worsened as the strength of the correlation increased. The Beta GLMM had Type I error rates near 5% as the sample size increased; however, estimates of the location and correlation parameters were significantly biased, and the coverage probabilities were $< 90\%$. The Beta GLMM had 100% model convergence; however, there were issues with the estimation of the Hessian matrix. The Beta GEE had an inflated Type I error rate whose amount of inflation decreased as the sample size N increased, but never reached 5%. The coverage probabilities approached 95% as N increased with coverage probabilities tending to be slightly $< 95\%$. However, the mean bias and the RMSD of the location and the correlation parameters were invariably near zero.

3.6.3. One group: $\mu = 0.5$, AR(1) correlation
[refer to Figures A.25 through A.36]

With a stationary mean of 0.5 and a correlation of 0.1, neither the LNMVB, Beta GLMM, nor Beta GEE was able to achieve the nominal 5% Type I error rate. The Beta GLMM had the least inflated Type I error rate of the three models. The Beta GLMM and

Beta GEE had Type I error rates $> 10\%$ that decreased to 6.1% and 7%, respectively, as the sample size increased. The LNMVB Type I error rate was consistently near 10% regardless of the sample size. As ρ increased, the LNMVB was closest of the three models to the nominal 5% Type I error rate for the smaller sample sizes; however, for the larger sample sizes the Beta GLMM and Beta GEE had the lowest Type I error rates with the Beta GLMM tending to be less inflated than that of the Beta GEE. The SLMVB models had inflated Type I error rates that increased as the sample size increased; however, this did not necessarily coincide with an increase in power. The Beta GLMM also experienced a reduction in power for larger effect sizes when $\rho = 0.3$ or 0.5 due to convergence issues.

The LNMVB had coverage probabilities $> 95\%$ for all the correlations and the sample sizes with the amount of overestimation decreasing as the correlation increased. Neither the SLMVB AR(1) or CS reached 90% coverage regardless of the simulation scenario. Rarely did the Beta GLMM have coverage probabilities $> 90\%$. For the Beta GEE, the coverage probabilities approached 95% as N increased with a minimum coverage probability near 91%.

The mean bias of the location parameter was near 0 under the LNMVB modeling framework. For the SLMVB models, an increase in the correlation caused an overestimation of the location parameter on average with the overestimation approaching 0.006 (i.e., 3/5th of a standard deviation) for $\rho = 0.5$. The Beta GLMM tended to slightly overestimate the value of the location parameter when $\rho = 0.1$ or 0.3 and scarcely underestimate when $\rho = 0.5$ (up to 1/10th of a standard deviation in either direction). The Beta GEE produced unbiased estimates of the location parameter on average across all the correlations and the sample sizes. The RMSD correlated with the mean bias results of the location parameters, i.e., biased estimates were associated with increases in RMSD.

The Beta GEE was the only model that was able to estimate the true value of the correlation parameter consistently. The SLMVB models correctly estimated the correlation parameter as N and or ρ increased with the SLMVB CS being less biased than the SLMVB AR(1) on average. The Beta GLMM had extremely biased estimates (≈ 0.5), and the LNMVB did not produce estimates across all combinations of correlations and sample sizes.

In conclusion, the LNMVB tended to have the lowest Type I error rates for the smaller sample sizes, while the Beta GLMM and Beta GEE had Type I error rates closest to nominal for the larger sample sizes. The Beta GLMM and SLMVB models had the least power, primarily as the correlation increased which appeared to be related to convergence issues. The LNMVB and Beta GEE were the only models able to produce unbiased estimates of the location parameter. The Beta GEE was the single model able to provide unbiased estimates of the correlation parameter consistently; however, the SLMVB models did produce unbiased estimates of the correlation parameter as the sample size and or the correlation increased. Furthermore, the Beta GEE had coverage probabilities closet to the desired 95%; while the LNMVB tended to have coverage probabilities $> 95\%$ that approached 95% as the strength of the correlation increased. The Beta GLMM and SLMVB models generally had coverage probabilities $< 90\%$.

3.6.4. One group: $\mu = 0.05$, CS correlation

The LNMVB had a Type I error rate slightly less than nominal (i.e., $\approx 4\%$) when the correlation was 0.1, had a Type I error rate near 2% when $\rho = 0.3$, and approached a Type I error rate of 0% when $\rho = 0.5$. Under the LNMVB model, there was a noticeable decrease in power, as expected, for Type I error rates $< 3\%$. The SLMVB models Type I error rate ranged between 10% and 70% across all the correlations and the sample sizes, thereby artificially inflating the power. Both the Beta GLMM and Beta GEE approached a Type I error rate of 5% as the sample size increased; however, the Type I error rate of the Beta GEE was more inflated compared to the Beta GLMM, e.g., at $\rho = 0.1$, $N = 15$ the Beta GEE had a Type I error of 13.2%, and the Beta GLMM had a Type I error of 5.7%.

The coverage probabilities of the LNMVB, Beta GLMM, and Beta GEE were similar across all combinations of parameters, except in one scenario ($\rho = 0.5$, $N = 100$) were the Beta GLMM had noticeably lower coverage than the LNMVB and Beta GEE. All three aforementioned models had coverage probabilities $< 95\%$ when the sample size was small ($N = 15$ or 30) and approached 95% coverage as the sample size increased (excluding the previously mentioned exception for the Beta GLMM). The SLMVB models had coverage probabilities $< 90\%$ for all combinations of parameters.

The mean bias of the location parameter was near zero when estimated by the LNMVB, Beta GEE, and Beta GLMM. The SLMVB produced biased estimates of the location parameter, with the mean bias > 0.06 (i.e., > 6 standard deviations) under certain conditions. Not surprisingly, the results for the RMSD of the location parameter were similar to that of the results for the mean bias of the location parameter. Specifically, the LNMVB, Beta GLMM, and Beta GEE had RMSD of the location parameter near zero, and the SLMVB had increased RMSD of the location parameter (e.g., > 0.15 for large effect size when $\rho = 0.5$, $N = 100$).

Only the Beta GEE was able to produce unbiased estimates of the correlation parameter regularly. The Beta GLMM produced slightly biased estimates of the correlation parameter with the bias being approximately 0.05 units greater than the true value. Lastly, the SLMVB CS came close to producing unbiased estimates of the correlation parameter as the sample size approached 100.

The LNMVB, Beta GLMM, and Beta GEE had similar performance as measured by coverage probabilities, mean bias and RMSD of the location parameter. The Type I error and power curves differed between the three models. The LNMVB tended to have Type I error closet to nominal with an emphasis to underestimate Type I error as the correlation increased. The Beta GLMM approached a Type I error rate of 5% as the sample size increased to $N = 50$ and 100, while it required a sample size of $N = 100$ for the Beta GEE. However, both the Beta GLMM and Beta GEE Type I error rates were more inflated compared to the LNMVB with the Beta GEE having the most inflated Type I error rates of the three models. Furthermore, the Beta GEE was the only model whose mean bias of the correlation parameter was near zero across all scenarios. Both SLMVB models estimates of the location and correlation parameters were biased, Type I error rates were inflated, and had convergence issues that worsened as ρ increased.

3.6.5. One group: $\mu = 0.3$, CS correlation

The LNMVB Type I error rate was 8.6% when the sample size was small (i.e., $N = 15$) and approached 6.6% as sample size increased for correlation 0.1. The LNMVB had a Type I error near nominal (4.6% to 5.8%) when $\rho = 0.3$. When the correlation was 0.5, the LNMVB Type I error rate was $< 5\%$ for all sample sizes. The SLMVB models had greatly inflated Type I error rates across all combinations of parameters, excluding $\rho = 0.5$. Additionally, the SLMVB models' power was significantly less than the other three models for medium and large correlations. The Beta GEE had inflated Type I error rates ($> 10\%$ when $N = 15$) that approached $\approx 6\%$ as the sample size N approached 100. There was no pattern to the Beta GLMM's Type I error rates (i.e., the Type I error did not monotonically decrease as the sample size increased). However, the Beta GLMM's Type I error rates were consistently between the Type I error rates of the LNMVB and Beta GEE.

Only the LNMVB and Beta GEE had coverage probabilities $> 90\%$ across all scenarios. The LNMVB tended to have coverage probabilities slightly above 95% (when $\rho = 0.1$) that decreased to slightly below 95% as ρ increased to 0.5. Whereas the Beta GEE tended to have coverage probabilities $< 95\%$ at the smaller sample sizes (i.e., $N = 15$ and 30) and coverage probabilities near 95% at the remaining sample sizes (i.e., $N = 50$ and 100) regardless of the correlation. Rarely did the remaining models (SLMVB and Beta GLMM) have coverage probabilities $> 90\%$.

Furthermore, only the LNMVB and Beta GEE were able to produce unbiased estimates of the location parameter. On average, the Beta GLMM overestimated the true value of the location parameter (by as much as 0.02 or 2 standard deviations), and the SLMVB both overestimated and underestimated the value of the location parameter depending on the correlation and sample size. As expected, the LNMVB and Beta GEE had RMSD of the location parameter consistently near zero. Amongst the Beta GLMM and SLMVB models, no model consistently had a RMSD of the location parameter nearer to zero.

The mean bias of the correlation parameter was repeatedly near zero when estimated by the Beta GEE. Under the SLMVB models, estimates of the correlation parameter became unbiased as the sample size N approached 100 with the SLMVB CS producing less biased estimates, on average than the SLMVB AR(1). Neither the LNMVB nor Beta GLMM was able to yield unbiased estimates of the correlation parameter.

The LNMVB's Type I error rate decreased as the correlation increased, with small correlations being inflated and medium and large correlations being at or below nominal. The SLMVB models tended to have highly inflated Type I error rates that did not always correspond to an increase in power. The Beta GEE had Type I error rates $> 10\%$ when N was small and approached (but never reached) 5% when N was large. The Beta GLMM did not present a clear pattern to its Type I error rates; however its Type I error rates tended to fall between that of the LNMVB and Beta GEE Type I error rates. The LNMVB and the Beta GEE were the only models able to produce unbiased estimates of the location parameter, and the Beta GEE was the lone model able to produce unbiased estimates of the correlation parameter. Additionally, only the LNMVB and Beta GEE had coverage probabilities $> 90\%$ across all simulations. For small and medium correlations, the LNMVB tended to have coverage probabilities closer to 95% than those of the Beta GEE; however, for large correlations the opposite was true.

3.6.6. One group: $\mu = 0.5$, CS correlation

The LNMVB, Beta GLMM, and Beta GEE had inflated Type I error rates that converged towards 5% as the sample size increased regardless of the strength of the correlation. When $\rho = 0.1$, the Beta GLMM showed the least amount of inflation of its Type I error rate, followed by either the LNMVB or Beta GEE depending on sample size. Specifically, the LNMVB had less inflated Type I error rates for smaller sample sizes compared to the Beta GEE. As the correlation increased, the performance of the LNMVB and Beta GEE improved, and the performance of the Beta GLMM worsened as measured by Type I error and power. For $\rho = 0.3$ and 0.5 the LNMVB had the least inflated Type I error rates for smaller sample sizes (i.e., $N \leq 30$) and there was very little difference in Type I error rates between the LNMVB, Beta GLMM, and Beta GEE when $N \geq 50$. However, the Beta GLMM did have convergence issues that markedly decreased its power. The SLMVB models had inflated Type I error rates that increased as the sample size increased for small and medium correlations. For large correlations, the SLMVB had flat power curves that corresponded to low convergence rates.

The LNMVB and Beta GEE consistently had coverage probabilities $> 90\%$ whereas it was rare for the remaining models to have coverage probabilities $> 90\%$. Only for $\rho = 0.1$ did the Beta GLMM have coverage probabilities $> 90\%$. The LNMVB coverage probabilities were usually $> 97\%$ when $\rho = 0.1$ and approached 95% as the strength of the correlation increased; sample size had virtually no effect on the coverage probabilities of the LNMVB model. The Beta GEE had coverage probabilities $< 95\%$ that approached 95% as the sample size increased.

The LNMVB, Beta GLMM, and Beta GEE tended to give unbiased estimates of the location parameter, with the Beta GLMM showing some slight bias for a few scenarios (< 0.001 or 1/10th of a standard deviation). The SLMVB models' estimates of the location parameter were biased with neither the SLMVB CS nor the SLMVB AR(1) consistently performing better than the other. The SLMVB models had the most pronounced bias of the location parameter when the correlation was 0.5 with an approximate bias of 0.01 or 1 standard deviation. The LNMVB and Beta GEE tended to have the lowest RMSD when estimating the location parameter, and the Beta GLMM generally had the highest RMSD when estimating the location parameter.

As has been the trend, the Beta GEE was the only model that was able to produce unbiased estimates of the correlation parameter across all correlation values and sample sizes. The SLMVB models correctly estimated the correlation parameter as N increased with the SLMVB CS being less biased than the SLMVB AR(1). Neither the LNMVB nor Beta GLMM was able to produce estimates of the correlation parameter near the true value.

To summarize, the strength of the correlation and the sample size determined whether the LNMVB, Beta GLMM, or Beta GEE had the lowest Type I error rate amongst the three. Generally, the LNMVB was the preferred choice for small sample sizes as determined by the Type I error. For the larger sample sizes (i.e., $N \geq 50$) there was little difference in the Type I error rates of the three models; however, the Beta GLMM had convergence issues that caused a decrease in power. The SLMVB models had inflated Type I error rates when $\rho = 0.1$ or 0.3 and substantial convergence issues when $\rho = 0.5$. The LNMVB and Beta GEE had the lowest mean bias and RMSD when estimating the location parameter, and the Beta GEE was the only model whose estimates of the correlation parameter where near the true value. Additionally, the LNMVB and Beta GEE were the sole models with coverage probabilities $> 90\%$. The Beta GEE behaved as expected, with coverage probabilities approaching 95% as sample size increased; however, the LNMVB had coverage probabilities that approached 95% as the correlation increased and was largely unaffected by the sample size.

3.6.7. Two groups: $\mu = 0.05$, AR(1) correlation [refer to Figures B.1 through B.12]

When $\rho = 0.1$ the LNMVB and Beta GLMM had similar power curves across all sample sizes. Both models had Type I error rates near 5% for the sample sizes $N \geq 30$ and inflated $(< 10\%)$ for the sample size $N = 12$. The Beta GEE had noticeably higher Type I error rates than the LNMVB and Beta GLMM for the sample sizes $N = 12$ and 30; however, for the sample sizes $N \geq 50$, the power curve of the Beta GEE was almost identical to that of the LNMVB and Beta GLMM. When $\rho = 0.3$, the LNMVB had Type I error rates between 2.8% and 7.1% and when $\rho = 0.5$ the Type I error rates were $< 4\%$. When $\rho = 0.3$ or 0.5 both the Beta GLMM and Beta GEE had inflated Type I error rates across all sample sizes (excluding $\rho = 0.3$ with $N = 100$) with the Beta GEE having larger Type I error rates than the Beta GLMM. The Beta GLMM had Type I error rates that ranged between 6.3% and 10.6% while the Beta GEE rates varied between 6.7% and 22.3% (excluding $\rho = 0.3$ with $N = 100$). The SLMVB models had markedly inflated Type I error rates, i.e., $> 20\%$ and convergence issues that increased as strength of correlation increased.

The LNMVB and Beta GLMM were the only models whose coverage probabilities were $> 90\%$ for the sample size $N = 12$. Additionally, both models coverage probabilities approached 95% as the sample size increased. For the sample sizes $N = 30, 50,$ and 100 the Beta GEE had coverage probabilities $> 90\%$. The Beta GEE coverage probabilities converged to 95% as N approached 100, tending to require larger sample sizes than the LNMVB or Beta GLMM to achieve 95% coverage. The SLMVB models never had coverage probabilities $> 90\%$.

The LNMVB, Beta GLMM, and Beta GEE estimates of the location parameter were unbiased. The SLMVB models' estimates of the location parameter were rarely unbiased; the bias was > 0.04 (or 4 standard deviations) at times. The SLMVB AR(1) tended to produce less biased estimates of the location parameter than the SLMVB CS.

The RMSD of the estimates of the location parameters were near zero for the LNMVB, Beta GLMM, and Beta GEE. The SLMVB AR(1) tended to have lower RMSD when estimating the location parameter than the SLMVB CS; however, rarely did either model produced metrics near zero.

Estimates of the correlation parameter were unbiased when estimated by the Beta GEE. The Beta GLMM and SLMVB models estimates of the correlation parameter were slightly biased. The SLMVB AR(1) always overestimated the true correlation. Both the Beta GLMM and SLMVB CS over and underestimated the true correlation dependent upon the true correlation value and sample size. The LNMVB was unable to produce valid estimates of the correlation parameter.

In summary, the LNMVB tended to have Type I error rates nearest the nominal 5%, while the SLMVB models Type I error rates were the most inflated. The Beta GLMM and Beta GEE Type I error rate approached 5% as the sample size N increased; however, in all but one scenario the Type I error rate remained $> 5\%$. Estimates of the location parameter were unbiased under the LNMVB, Beta GLMM, and Beta GEE models and biased under the SLMVB models. Additionally, the LNMVB and Beta GLMM had coverage probabilities nearest the expected 95%. However, the Beta GEE model was the only model whose estimates of the correlation parameter were unbiased.

3.6.8. Two groups: $\mu = 0.3$, AR(1) correlation [refer to Figures B.13 through B.24]

The LNMVB never achieved the nominal Type I error rate, with a typical Type I error rate near or slightly higher than 8%. Additionally, the LNMVB Type I error rates decreased as the sample size and or correlation increased. Both the Beta GLMM and Beta GEE had inflated Type I error rates that approached $\approx 6\%$ as the sample size increased. The Beta GEE had a significantly inflated Type I error rate for the smaller sample sizes, i.e., near 20%, for the sample size $N = 12$ compared to the LNMVB and Beta GLMM whose rates were near 10%. The SLMVB models tended to have Type I error rates above 20%. Power curves among the LNMVB, Beta GLMM, and Beta GEE tended to be similar for the sample sizes $N = 30, 50,$ and 100, while the power of the SLMVB models were generally inflated. Furthermore, the SLMVB had worsening convergence issues as ρ increased.

The LNMVB was the only model who had coverage probabilities $> 90\%$ across all scenarios. Additionally, the LNMVB behaved as expected, with coverage probabilities approaching 95% as the sample size increased. The Beta GEE displayed similar behavior, i.e., coverage probabilities that approached 95% as the sample size increased; however, the coverage probabilities of the Beta GEE tended to be $< 95\%$. The Beta GLMM rarely had coverage probabilities $> 90\%$ and the SLMVB never had coverage probabilities $> 90\%$.

The LNMVB and Beta GEE estimates of the location parameter were unbiased across all the correlations and the sample sizes. The Beta GLMM estimates of the location parameter were biased, showing the least bias when $\rho = 0.5$ and $N = 100$. The Beta GLMM estimates of the location parameter were the most biased when $\rho = 0.1$ with bias > 0.02 (or 2 standard deviations). The SLMVB AR(1) estimates of the location parameter were unbiased at $\rho = 0.1$, slightly biased at $\rho = 0.3$, and biased at $\rho = 0.5$. The SLMVB CS estimates of the location parameter were similar to that of the SLMVB AR(1); however, the SLMVB CS estimates were more biased.

The RMSD of the estimates of the location parameter were consistently near zero when estimated by the LNMVB and Beta GEE. The SLMVB models and Beta GLMM tended to have RMSD for estimates of the location parameter > 0.02 with no model persistently outperforming the other two models.

The Beta GEE's estimates of the correlation parameter were consistently unbiased, and the SLMVB models estimates of the correlation parameter became unbiased as the correlation and or the sample size increased. The Beta GLMM never produced unbiased estimates of the correlation parameter while the LNMVB was unable to calculate the correlation parameter.

To summarize, none of the models were able to achieve Type I error rates of 5%; however, there were two cases where the LNMVB had a Type I error rate of 5.3%. When considering the LNMVB and Beta GEE (the only models whose estimates of the location parameter were unbiased), the LNMVB had lower Type I error rates for the sample size $N = 12$ while the Beta GEE had lower Type I error rates for the sample sizes $N = 50$ and 100 when $\rho = 0.1$ or 0.3. When $\rho = 0.5$, the LNMVB Type I error rate was always lower than that of the Beta GEE. Furthermore, the LNMVB had coverage probabilities that were nearer to 95% than that of the Beta GEE across all the correlations and the sample sizes. However, the Beta GEE was the only model where the estimates of the correlation parameter were unbiased among all scenarios.

3.6.9. Two groups: $\mu = 0.5$, AR(1) correlation
[refer to Figures B.25 through B.36]

The LNMVB, Beta GLMM, and Beta GEE had Type I error rates that were inflated regardless of the correlation and or the sample size. The Type I error on these three models all decreased as the sample size N increased. Of the three models, the LNMVB had the least inflated Type I error rates for the sample size $N = 12$. For the remaining sample sizes ($N = 30, 50,$ and 100), in general, the Beta GLMM had the lowest Type I error rate, followed by the Beta GEE, and lastly the LNMVB. However, the Beta GLMM had convergence issues that worsened as ρ increased that caused a reduction in power. The SLMVB models had Type I error rates that were significantly inflated when $\rho = 0.1$ or 0.3. When $\rho = 0.5$ the SLMVB CS had Type I error rates $< 5\%$ for the sample sizes $N \leq 30$ and Type I error rates $> 5\%$ for the sample sizes $N \geq 50$. Whereas the SLMVB AR(1) had a Type I error rate $< 5\%$ for the sample size $N = 12$ that increased as N increased with a maximum Type I error rate $> 20\%$ when $N = 100$.

The LNMVB was the only model whose coverage probabilities were $> 90\%$ for the sample size $N = 12$ across all correlations. Specifically, the LNMVB tended to have coverage probabilities between 96% and 97%. The Beta GEE had coverage probabilities near 95% for the sample size $N = 100$; otherwise, the coverage probabilities decreased as the sample size decreased. There were instances where the Beta GLMM had coverage probabilities between 90% and 95% ($N \geq 50$ with $\rho = 0.1$ and $N = 100$ with $\rho = 0.3$); however, the Beta GLMM commonly had coverage probabilities $< 90\%$. The SLMVB models coverage probabilities were always $< 90\%$.

The LNMVB and Beta GEE estimates of the location parameter were unbiased. The Beta GLMM estimates of the location parameter were unbiased except for the smaller sample sizes, i.e., $N = 12$ and 30, with a maximum mean bias of 0.002 (1/5th of a standard deviation). The SLMVB models' estimates of the location parameter were slightly biased across all scenarios with the SLMVB AR(1) being less biased than the SLMVB CS. The LNMVB and Beta GEE consistently had the lowest RMSD of the location parameter with the Beta GLMM having similar RMSD except at the smaller sample sizes. The SLMVB models had the highest RMSD of the location parameter across all scenarios, with a maximum RMSD > 0.04.

The estimates of the correlation parameter were unbiased when estimated by the Beta GEE. Additionally, the SLMVB models produced unbiased estimates of the correlation parameter at the larger sample sizes, i.e., $N = 50$ and 100. Neither the LNMVB or Beta GLMM estimates of the correlation parameter were unbiased. The Beta GLMM overestimated the true value of the correlation parameter by > 0.4 under all scenarios, and the LNMVB was unable to estimate the correlation parameter.

In conclusion, the LNMVB, Beta GLMM, and Beta GEE were the models with unbiased estimators of the location parameter. None of these three models were able to control the Type I error rate. The LNMVB had the lowest Type I error rate for the small sample size ($N = 12$) while the Beta GLMM tended to have the lowest Type I error for the remaining sample sizes ($N = 30, 50,$ and 100). However, the Beta GLMM had convergence issues that affected its power. Furthermore, only the LNMVB had coverage probabilities $> 90\%$ for all scenarios with a tendency to have coverage probabilities between 96% and 97%. The Beta GEE approached the desired 95% coverage probability as the sample size increased to $N = 100$ and the Beta GLMM rarely had coverage probabilities above 90%. The Beta GEE was the only model that produced unbiased estimates of the correlation parameter across all correlations and sample sizes.

3.6.10. Two groups: $\mu = 0.05$, CS correlation

When $\rho = 0.1$ the LNMVB had the best control of the Type I error rate (near 5%), with slight inflation when the sample size $N = 12$ (i.e., 6.9%). For $\rho = 0.3$ and 0.5 the LNMVB Type I error rate was always $< 4\%$ which caused a decrease in power. Both the Beta GLMM and Beta GEE had inflated Type I error rates that approached 5% as the sample size increased. However, the Beta GEE had Type I error rates near 20% for the sample size $N = 12$, regardless of the correlation. The SLMVB tended to have Type I error rates $> 20\%$ which did not necessarily equate to inflated power at larger effect sizes likely caused by convergence issues.

The LNMVB and Beta GLMM had similar coverage probabilities across simulations with the coverage probabilities of the Beta GEE tending to be less than that of the LNMVB or Beta GLMM. The LNMVB and Beta GLMM were the only models who had coverage probabilities $> 90\%$ for the sample size $N = 12$. Additionally, their coverage probabilities converged to 95% as the sample size increased. For the sample size $N = 100$, the coverage probabilities of the LNMVB, Beta GLMM, and Beta GEE were nearly identical. The SLMVB coverage probabilities were always below 90%.

The LNMVB, Beta GLMM, and Beta GEE estimates of the location parameter were unbiased with the mean bias near 0 for all simulations. The SLMVB models overestimated the true value of the location parameter with mean biases typically > 0.02 (i.e., 2 standard deviations) and reaching a maximum of > 0.06 (i.e., 6 standard deviations). The RMSD of the location parameter was as expected, with the LNMVB, Beta GLMM, and Beta GEE having RMSD near 0 while the RMSD for the SLMVB models were generally > 0.10.

The Beta GEE was the only model whose estimates of the correlation parameter were consistently near the true value with a slight emphasis to underestimate, primarily when the sample size was smaller. The Beta GLMM overestimated the correlation parameter by 0.05 across all correlations and sample sizes. Of the remaining models, only the SLMVB CS had estimates of the correlation parameter that was near the true value; however, this occurred as both correlation and sample size increased.

In summary, the LNMVB, Beta GLMM, and Beta GEE estimates of the location parameter were unbiased while the SLMVB overestimated the true value of the location parameter. The LNMVB was the only model that could consistently control the Type I error rate; however, the rate was often below 5% causing a loss of power. Both the LNMVB and Beta GLMM had similar coverage probabilities that converged to 95% as the sample size increased, with the Beta GEE tending to have coverage probabilities less than the LNMVB or Beta GLMM. Lastly, both the Beta GLMM and Beta GEE had acceptable estimates of the correlation parameter, with the Beta GEE having estimates nearer the true value.

3.6.11. Two groups: $\mu = 0.3$, CS correlation

When $\rho = 0.1$, none of the models were able to control the Type I error rate. The Beta GLMM was nearest the desired Type I error rate of 5%, especially as the sample size increased, followed by the Beta GEE (except for the sample size $N = 12$). When compared to the Beta GLMM and Beta GEE excluding the sample size $N = 12$, the LNMVB had the least control of the Type I error rate (with Type I error rates between 7.1% and 12%) when $\rho = 0.1$. The SLMVB models Type I error rate increased from approximately 20% to 40% as the sample size increased when $\rho = 0.1$. When $\rho = 0.3$ the power curves of the LNMVB and Beta GLMM were virtually indistinguishable with a Type I error rate of approximately 9% for the sample size $N = 12$, converging to $\approx 6\%$ for the sample size $N = 100$. The Beta GEE Type I error rate remained inflated (i.e., $>$ 20% when $N = 12$ and 7% when $N = 100$) for $\rho = 0.3$. Again, the SLMVB models Type I error rate increased as the sample size increased when $\rho = 0.3$. For $\rho = 0.5$ the LNMVB Type I error rate decreased from 4% to 3% as the sample size increased. Both the Beta GLMM and Beta GEE had inflated Type I error rate that approached 5.3% and 6.3%, respectively, as the sample size increased, with the Beta GLMM always nearer to the nominal 5% than that of the Beta GEE. Lastly, the SLMVB Type I error rates increased from $< 2\%$ to $> 10\%$ as the sample size increased.

The LNMVB coverage probabilities approached 95% as the sample size increased and was the only model with coverage probabilities $> 90\%$ for the sample size $N = 12$. The Beta GEE had coverage probabilities near 95% when the sample size $N = 100$; however, the coverage probabilities at the medium sample sizes ($N = 30$ and 50) were $< 95\%$ and $< 90\%$ for the sample size $N = 12$. When $\rho = 0.5$ the Beta GLMM behaved as expected with coverage probabilities converging to 95% as the sample size increased; otherwise the coverage probabilities tended to stay below 90%. The SLMVB models never had coverage probabilities $> 90\%$.

The mean bias of the estimates of the location parameter was near zero for the LNMVB and Beta GEE across all scenarios. The Beta GLMM produced unbiased estimates of the location parameter only when $\rho = 0.5$ and $N = 50$ or 100, with a maximum bias of 0.015 or approximately 1.5 standard deviations whereas the SLMVB model estimates became more biased as the correlation increased corresponding to decreased model convergence. The values of RMSD of the location parameter correlated with the values of the mean bias of the location parameter, i.e., when the mean bias of the location parameter was near zero so was the RMSD of the location parameter.

The Beta GEE estimates of the correlation parameter were unbiased for all correlations and sample sizes. The SLMVB CS produced unbiased estimates of the correlation parameter as N increased for all correlations. The remaining models' estimates of the correlation parameter were biased.

To summarize, only the LNMVB and Beta GEE estimates of the location parameter were unbiased. The Correlation and the sample size determined which model (LNMVB or Beta GEE) had better control of the Type I error rate with the Beta GEE having better control when $\rho = 0.1$ (excluding $N = 12$), the LNMVB when $\rho = 0.3$ or 0.5; however, when $\rho = 0.5$ the LNMVB had Type I error rates below nominal. The LNMVB had coverage probabilities that approached 95% as the sample size increased and was the only model with coverage probabilities $> 90\%$ for the sample size $N = 12$. The Beta GEE coverage probabilities also converged to 95%; however, the coverage probabilities of the Beta GEE tended to be less than that of the LNMVB. Lastly, the Beta GEE was the only model able to estimate the true value of the correlation parameter across all scenarios.

3.6.12. Two groups: $\mu = 0.5$, CS correlation

The SLMVB models Type I error and power were significantly inflated when $\rho = 0.1$ or 0.3 and were never above 5% when $\rho = 0.5$. Therefore, we will limit the discussion of Type I error and power to the LNMVB, Beta GLMM, and Beta GEE. The LNMVB had the best control of the Type I error for the sample size $N = 12$ with a Type I error of 13.6% when $\rho = 0.1$ decreasing to 7.8% when $\rho = 0.5$. The Beta GLMM and Beta GEE had Type I errors around 20% for the sample size $N = 12$. We will further limit our discussion to the sample sizes $N = 30, 50,$ and 100. When $\rho = 0.1$, the Beta GLMM had the most control of the Type I error rate with values between 5.4% and 6.9%, followed by the Beta GEE with rates between 6.4% and 8.1%, and lastly by the LNMVB with rates between 8.9% and 9.4%. When $\rho = 0.3$, the sample size dictated which model had the most control of the Type I error rate. Furthermore, the Beta GLMM was the only model able to achieve the nominal 5% Type I error rate. When $\rho = 0.5$ and $N = 30, 50$, the LNMVB had the lowest Type I error rates with rates of 7.8% and 5.8%, respectively. Both the Beta GLMM and Beta GEE had Type I error rates $> 10\%$ for $N = 30$. At $N = 100$, the Type I error rates were 6.5%, 6.7%, and 5.5% for the LNMVB, Beta Gee, and Beta GLMM, respectively. However, the Beta GLMM experienced convergence issues that affected its power.

The LNMVB was the only model whose coverage probabilities were $> 90\%$ for all scenarios. Specifically, the coverage probabilities tended to be $> 95\%$ at the small and the medium correlations (i.e., $\rho = 0.1$ or 0.3) and near 95% when the correlation was large. The Beta GEE coverage probabilities were $> 90\%$ but $< 95\%$ when $N = 30, 50,$ or

100 and $< 90\%$ for the sample size $N = 12$. Rarely were the coverage probabilities of the Beta GLMM $> 90\%$ and the SLMVB models coverage probabilities were never $> 90\%$.

The LNMVB, Beta GLMM, and Beta GEE estimates of the location parameter were near the true value on average, with the Beta GLMM showing a slight bias (< 0.001 or 1/10th of a standard deviation) in a few scenarios. The SLMVB models mean bias of the location parameter diverged from zero as the strength of the correlation increased. The LNMVB and Beta GEE estimates of the location parameter had the lowest RMSD. When the sample size was $N = 50$ or 100, the RMSD of the Beta GLMM matched that of the LNMVB and Beta GEE; however when the sample size $N = 12$ or 30, the Beta GLMM RMSD of the location parameter were higher than that of the LNMVB and Beta GEE. Lastly, the SLMVB tended to have the highest RMSD for estimates of the location parameter.

The Beta GEE estimates of the correlation parameter were unbiased. Estimates of the correlation parameter using the SLMVB models became unbiased as the sample size increased, with the SLMVB CS estimates being less biased than the SLMVB AR(1) estimates. The LNMVB was unable to estimate the correlation parameter, and the Beta GLMM estimates were significantly biased, i.e., > 0.5.

In conclusion, the SLMVB models’ estimates of the location parameter were biased, and the SLMVB models were unable to control the Type I error rates. The LNMVB, Beta GLMM, and Beta GEE produced unbiased estimates of the location parameter; however, when examining RMSD of the location parameter, the Beta GLMM performed worse than the LNMVB and Beta GEE. Additionally, none of these three models were able to control the Type I error rates properly (excluding one scenario for the Beta GLMM). The Beta GLMM had the lowest Type I error rates when the correlation was small, and the LNMVB had the lowest rates when the correlation was large. When the strength of the correlation was medium, no model had consistently lower Type I error rates than the others. Furthermore, the Beta GLMM had convergence issues for the medium and the high correlations. The LNMVB and Beta GEE had coverage probabilities closet to the expected 95% compared to the other three models whose coverage probabilities remained $< 90\%$. Additionally, the LNMVB was the only model with coverage probabilities $> 90\%$ when the sample size was $N = 12$. Lastly, the Beta GEE was the only model whose estimates of the correlation parameter were unbiased.

3.7. Summary

3.7.1. One group

The LNMVB and Beta GEE were the only models whose estimates of the location parameter were unbiased across all scenarios. The Beta GLMM estimates of the location parameter were the most biased when $\mu = 0.3$, and the SLMVB models' estimates of the location parameter were rarely unbiased. Therefore, we will limit the discussion to the LNMVB and Beta GEE. The RMSD of the estimates of the location parameter were nearly identical and near zero for both the LNMVB and Beta GEE. In general, the LNMVB had Type I error rates that were closer to nominal compared to the Beta GEE. This was most pronounced at the smaller sample sizes (i.e., $N = 15$ and 30) where the Type I error rates of the Beta GEE were often $> 10\%$. However, there were instances where the LNMVB had Type I error rates $< 5\%$ which were often associated with a decrease in power. For $\mu = 0.05$ the coverage probabilities of the LNMVB and Beta GEE were similar, with both models' coverage probabilities approaching 95% as the sample size N approached 100. When $\mu = 0.3$ the LNMVB tended to have coverage probabilities closer to 95% for smaller the sample sizes (i.e., $N \leq 30$) compared to the

Beta GEE. At $N \geq 50$, the LNMVB tended slightly overestimate the coverage probabilities. Furthermore, when $\mu = 0.5$ the LNMVB tended to have coverage probabilities $> 97\%$ for small correlations that converged to 95% as strength of correlation increased; whereas the Beta GEE had coverage probabilities that converged to 95% as sample size increased regardless of the correlation. Lastly, the Beta GEE's estimates of the correlation parameter where near the true value on average; as opposed to the LNMVB, which was unable to estimate the correlation parameter.

3.7.2. Two groups

As in the single group case, we will limit our discussion to models whose estimators of the location parameter were unbiased across all scenarios, i.e., the LNMVB and Beta GEE. Again, the RMSD of the estimates of the location parameter were similar and near zero for both models. When $\mu = 0.05$ the LNMVB was generally able to control the Type I error rate while the Beta GEE was unable to control the Type I error rate. For $\mu = 0.3$ both the LNMVB and Beta GEE had inflated Type I error rates that tended towards 5% as the sample size N increased with the LNMVB tending to nominal at a faster rate, in general. When $\mu = 0.5$ and the sample size was $N = 30, 50,$ or 100 the Beta GEE usually had a lower Type I error rate than the LNMVB; however, for the sample size $N = 12$ the LNMVB had better control of the Type I error rate. Neither model had a Type I error rate of 5% when $\mu = 0.5$. Furthermore, both models coverage probabilities tended to approach 95% as sample size increased. However, the LNVMB coverage probabilities were closer to the desired 95% than that of the Beta GEE. Additionally, the Beta GEE had coverage probabilities $< 90\%$ for the small sample size, while the LNMVB coverage probabilities were always $> 90\%$. However, there were instances such that the coverage probabilities of the LNMVB were $> 95\%$ (generally when $\mu = 0.5$ and $\rho = 0.1$). Lastly, the Beta GEE produced unbiased estimates of the correlation parameter as opposed to the LNMVB which was unable to estimate the correlation parameter.

4. Data Analysis, National NeuroAIDS Tissue Consortium

4.1. Participants

The 35 study participants were derived from the NNTC database, a longitudinal observational study that includes biannual neurologic, neuropsychologic, and psychiatric examinations of participants with HIV and without HIV. As of November 1, 2018, $3{,}150$ participants have enrolled in the NNTC study.[24] For details of the NNTC study and inclusion criteria see Section 1.2 and Morgello et al.[23] We restricted the $3{,}150$ participants to African American females with complete neuropsychological exam data (specifically, the HVLT-R delayed scaled score) for visits $0, 1, 2,$ and 3; which resulted in 35 participants (see Figure 4.1). Study participants were then dichotomized into < 12 years of education ($N = 17$) and ≥ 12 years of education ($N = 18$). We required the amount of education to remain constant during the 4 study visits for the 35 participants. Therefore, the inclusion/exclusion criteria and dichotomization of education resulted in a small sample fairly balanced design that we were able to analyze using a two-way interaction model.

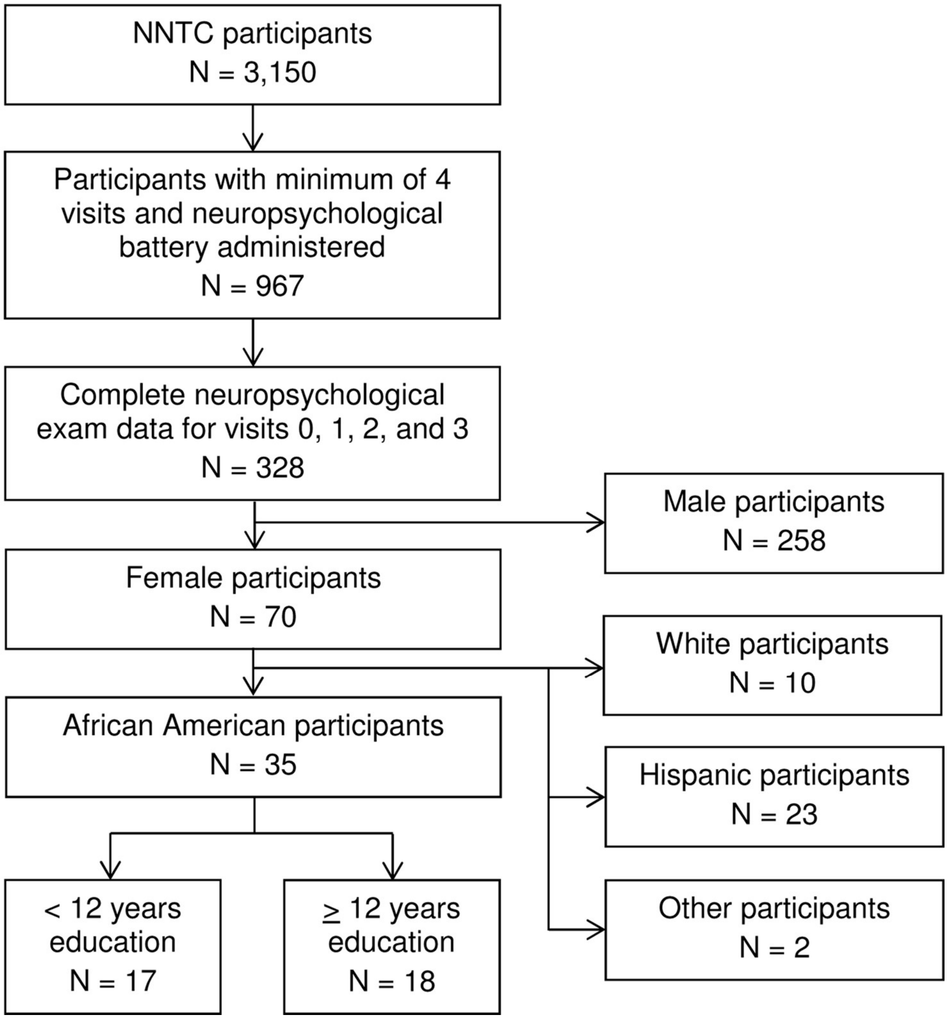


Figure 4.1 Schematic summary of inclusion/exclusion criteria of NNTC data.

The patient characteristics between education groups (< 12 years and ≥ 12 years) were reasonably balanced (Table 4.1). The average time between visits was slightly more than 6 months, regardless of education group and or visit number. The minimum baseline visit for each education group occurred in 1999. The < 12 years of education group had a longer time span of recruitment with the last baseline visit occurring in 2015, compared to the ≥ 12 years of education group whose last baseline visit occurred in 2010. The mean age was 40.6 years and 48.3 years for the < 12 years of education group and ≥ 12 years of education group, respectively.

Table 4.1 Patient characteristics.

	< 12 years of education (n = 17)	$\geq$ 12 years of education (n =18)
Mean age (SD), years	40.6 (7.5)	48.3 (8.9)
Minimum baseline visit, year	1999	1999
Maximum baseline visit, year	2015	2010
Mean time between visit 0 & 1 (SD), days	186 (30)	190 (18)
Mean time between visit 1 & 2 (SD), days	196 (38)	194 (29)
Mean time between visit 2 & 3 (SD), days	190 (27)	183 (40)

4.2. Methods

Prior to analyzing the NNTC data, the responses (HVLT-R delayed scaled score) were converted to proportions. The minimum and maximum theoretical HVLT-R delayed scaled score is 0 and 19, respectively. Therefore, HVLT-R delayed scaled score (denoted y') can be computed as

$$y' = \frac{HVLT- \quad delayed\ scaled\ score - 0}{19-0} \ .$$

This conversion bounded y' on the open interval $(0,1)$ since no individual scores were on the lower or upper bound.

We then considered the following covariates when analyzing the NNTC data: education group, visit number, and education group by visit number interaction. The following model was fitted under each paradigm (i.e., LNMVB, SLMVB, Beta GLMM, and Beta GEE)

$$\log\left(\frac{\mu_{ij}}{1-\mu_{ij}}\right) = \beta_1 + \beta_2 * visit_{i1} + \beta_3 * visit_{i2} + \beta_4 * visit_{i3} + \beta_5 * educ_i + \beta_6 * educ_i * visit_{i1} + \beta_7 * educ_i * visit_{i2} + \beta_8 * educ_i * visit_{i3}$$

$$Y'_{ij} \sim Beta(\mu_{ij}, \phi)$$

for participants $i = 1, \ldots, 35$, visits $j = 0, 1, 2$ and 3, and such that $Beta(\mu_{ij}, \phi)$ is parameterized as density (1.4). For the Beta GEE, we assumed an AR(1) correlation structure. Details of the model fitting procedures can be found in Section 2.3, Section 2.4, and Section 3.4 for the LNMVB, SLNMVB, and Beta GLMM and Beta GEE, respectively. All analyses were performed using R[64] version 3.4.2 on a Windows 10 PC.

We report coefficients, standard errors, Wald type confidence intervals, and p-values for all parameters and models in Section 4.3 and Appendix C. Additionally, expected means (calculated from model parameters) for each cell of the interaction are plotted along with cell means calculated using the data. Overall significance of the education group by visit number interaction is reported using the F-test described in Section 3.5. Lastly, correlation is reported for each model and compared to the empirical correlation of the data.

4.3. Results

The SLMVB AR(1) and GEE model did not converge; however, the estimates of the GEE model are comparable to those of the LNMVB and GLMM. Therefore, we will limit the reporting of results to the LNMVB, SLMVB CS, GLMM, and GEE. Refer to Appendix C for results of the SLMVB AR(1) model.

From Figure 4.2, the interaction between education and visit number is significant under the modeling framework of the GLMM and GEE (p-values 0.0308 and 0.0173, respectively). The LNMVB and SLMVB CS F-test p-values for the education by visit number interaction is non-significant (p-values 0.0776 and 0.0920, respectively) even though the mean profiles suggest an interaction (Figure 4.2). However, this is not unexpected based on the simulation results. From the simulation results, for sample size $N = 12$ and 30 and AR(1) correlation structure with $\rho = 0.5$, the GLMM and GEE had the most inflated Type I error rates amongst the four models with the GEE being more inflated than that of the GLMM. Therefore, the significant F-tests under the GLMM and GEE could be a symptom of the inflated Type I error rate.

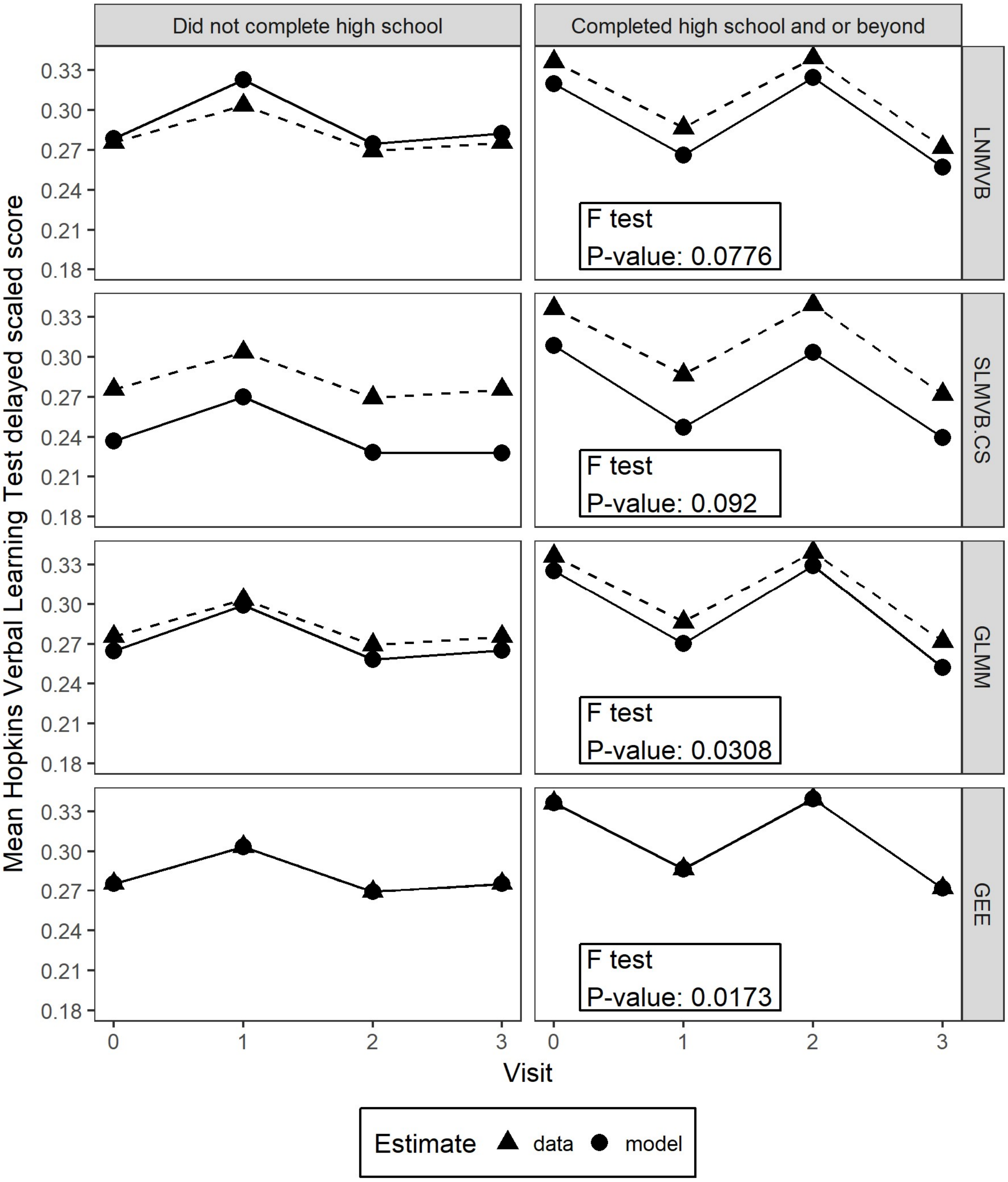


Figure 4.2 Mean profiles plots of NNTC data using LNMVB, SLMVB CS, GLMM, and GEE models.

Model estimates, standard errors, and 95% confidence intervals between the LNMVB, SLMVB CS, GLMM, and GEE models were varied (Table 4.2). Using the same aforementioned simulation results, the SLMVB CS and Beta GLMM showed a slight emphasis to overestimate the location parameters. This overestimation can be seen in the mean profile plots (Figure 4.2) and could possibly explain the varied estimates between models. However, statistical significance of parameter estimates was consistent among the four models.

Table 4.2 Model estimates (LNMVB, SLMVB CS, GLMM, and GEE) of the NNTC data.

	Estimate	**SE**	**95% CI**	**P-Value**	**Converged**
LNMVB					Yes
Intercept	-0.953	0.140	(-1.231, -0.674)	0.000	
Visit 1	0.211	0.163	(-0.112, 0.534)	0.198	
Visit 2	-0.019	0.171	(-0.358, 0.319)	0.910	
Visit 3	0.020	0.169	(-0.316, 0.356)	0.906	
Educ	0.196	0.185	(-0.171, 0.564)	0.292	
Visit 1*Educ	-0.471	0.229	(-0.925, -0.016)	0.043	
Visit 2*Educ	0.041	0.228	(-0.411, 0.493)	0.858	
Visit 3*Educ	-0.325	0.235	(-0.791, 0.141)	0.169	
SLMVB CS					Yes
Intercept	-1.171	0.143	(-1.454, -0.888)	0.000	
Visit 1	0.176	0.172	(-0.165, 0.518)	0.308	
Visit 2	-0.048	0.183	(-0.412, 0.316)	0.794	
Visit 3	-0.050	0.178	(-0.404, 0.304)	0.778	
Educ	0.363	0.197	(-0.028, 0.754)	0.068	
Visit 1*Educ	-0.483	0.234	(-0.947, -0.020)	0.041	
Visit 2*Educ	0.024	0.244	(-0.459, 0.508)	0.920	
Visit 3*Educ	-0.297	0.245	(-0.782, 0.189)	0.229	
GLMM					Yes
Intercept	-1.021	0.166	(-1.351, -0.692)	0.000	
Visit 1	0.169	0.142	(-0.114, 0.451)	0.238	
Visit 2	-0.033	0.144	(-0.318, 0.252)	0.819	
Visit 3	0.001	0.143	(-0.283, 0.285)	0.994	
Educ	0.291	0.230	(-0.166, 0.748)	0.209	
Visit 1*Educ	-0.433	0.197	(-0.824, -0.042)	0.030	
Visit 2*Educ	0.05	0.196	(-0.339, 0.439)	0.798	
Visit 3*Educ	-0.357	0.200	(-0.754, 0.039)	0.077	

GEE					No
Intercept	-0.967	0.136	(-1.237, -0.697)	0.000	
Visit 1	0.136	0.116	(-0.095, 0.366)	0.246	
Visit 2	-0.031	0.132	(-0.294, 0.231)	0.814	
Visit 3	0.000	0.16	(-0.317, 0.317)	1.000	
Educ	0.287	0.208	(-0.126, 0.700)	0.171	
Visit 1*Educ	-0.368	0.159	(-0.684, -0.051)	0.023	
Visit 2*Educ	0.044	0.169	(-0.291, 0.380)	0.794	
Visit 3*Educ	-0.305	0.221	(-0.744, 0.134)	0.171	

Pairwise correlations calculated using the data are displayed in Table 4.3. The data generally follows an AR(1) correlation structure, with $\rho \approx 0.7$. The mean pairwise correlation coefficients under the LNMVB model was 0.300. The SLMVB model with CS specified correlation structure estimate of ρ was 0.657. The GLMM model with random intercept has an exchangeable correlation structure whose value was estimated to be 0.881. Lastly, we specified an AR(1) correlation structure for the GEE model whose estimate was 0.710.

Table 4.3 Empirical pairwise correlation estimates of the NNTC data.

	Visit 0	**Visit 1**	**Visit 2**	**Visit 3**
Visit 0	1	0.683	0.728	0.456
Visit 1		1	0.746	0.524
Visit 2			1	0.591
Visit 3				1

5. General Conclusions, Limitations and Future Research

5.1. Introduction

This dissertation makes a contribution to the statistical methodology for the analysis of repeatedly-measured proportional data (applicable to the medical field, economics, social sciences, etc.) as well as provides R[64] code for the application of the proposed methods (see Appendix D). The focus was on the limitations of existing methods, and two alternative methods were introduced. This chapter provides a general discussion of our findings, lists some limitations to our proposed methods, and suggestions for areas of future research.

5.2. Conclusions

Methods for the analysis of correlated proportional data have been presented and described in detail. Applications of the methods in this dissertation have been specific to neuropsychological data; however, these proposed methods can be applied to other fields of research producing data with similar characteristics. Specifically, in Chapter 1 we described the current models available to handle repeatedly-measured proportional data and their shortcomings, e.g., joint likelihood Beta models are limited to two repeated measures, marginal models do not use full joint likelihood, GLMMs require numerical estimation of integrals, etc. In Chapter 2 we proposed two classes of models (the LNMVB and the SLMVB) to address the limitations of the models currently available. Both the LNMVB and the SLMVB are based on a full joint likelihood with no limit on the number of repeated measures that the models can handle and are parameterized such that the parameters have a marginal interpretation. Furthermore, the maximum likelihood estimates can be calculated using an iterative procedure that does not involve integrals. Therefore, the LR test, AIC, BIC, etc. can be used for model selection.

To study the performance of the models in Chapter 1 and our proposed models in Chapter 2, a simulation study was conducted in Chapter 3. Four conclusions can be drawn from the simulation study. First, the LNMVB and Beta GEE were the only models that produced unbiased estimates of the location parameter for all scenarios simulated. Generally, the location parameter and inference about the location primary is of primary importance to the investigator; therefore, further conclusions will be limited to the LNMVB and Beta GEE. Second, the LNMVB tended to have better control of the Type I error rate, which was especially evident for the smaller sample sizes. Third, both the LNMVB and Beta GEE coverage probabilities tended towards 95% as the sample size increased; however, the LNMVB had coverage probabilities closer to 95% than that of the Beta GEE which was most pronounced when the sample size was small. Lastly, the Beta GEE was the only model whose mean bias of the correlation parameter was consistently near zero for all simulation scenarios. These four conclusions imply that the LNMVB is preferred for analyzing small sample (i.e., ≤ 30) repeatedly-measured proportional data and either the LNMVB or Beta GEE works well for analyzing large sample (i.e., ≥ 50) correlated Beta distributed data. Furthermore, if the correlation is the parameter of interest and or the location estimates approach the upper bound, then the Beta GEE is the preferred model.

Chapter 4 compares the estimates of the LNMVB, SLMVB, Beta GLMM, and Beta GEE using neuropsychological data. Sample size was < 20 per group. The SLMVB AR(1) model and Beta GEE model did not converge. However, estimates of the Beta GEE model were similar to those of the LNMVB and GLMM models. Mean profile plots revealed that the Beta GEE had the least bias in estimating the means of the location parameters when compared to the empirical means. Additionally, the Beta GEE had the most significant p-value for the F-test of the interaction. Furthermore, the Beta GLMM produced a significant F-test, while the LNMVB and SLMVB CS did not produce a significant F-test. However, it is unclear if these significant F-tests are a result of a possible inflation of the Type I error rates as shown in the simulations. Lastly, the Beta GEE and SLMVB models estimates of the correlation were closet to the empirical estimates.

5.3. Limitations

The majority of the limitations are a result of the simulation study and the constraints that are inherent in any simulation study. As with any simulation study, the results cannot be generalized and are limited to the scenarios and or parameters tested. Our simulation study was a balanced design, which often is not the case when analyzing real data. Furthermore, we limited ourselves to three location parameters ($\mu = 0.05, 0.3,$ and 0.5) assuming that the models would have a symmetric behavior on either side of $\mu = 0.5$ and that there would be a pattern to how the models behaved as we moved the location parameter away from 0.5 towards either 0 or 1. This behavior was verified under the two-way interaction model with AR(1) correlation structure; however, the LNMVB results are not symmetric but are predictable (see below). To quantify Type I error and power, the effect size was used as a scale which was calculated as the difference of means divided by the standard deviation. We fixed the standard deviation at 0.01 and adjusted the mean accordingly. It remains unclear how different standard deviations would affect the results. Additionally, these models do not have theoretical power calculations, so we relied on the empirical results. Our models were limited to a two-way interaction model and including additional covariates could alter the results. Moreover, we only used an AR(1) and CS correlation structure with small, medium, and large correlations, one or two treatment groups, and fixed sample sizes.

For the simulation study, it was necessary to generate correlated Beta distributed data. Current simulations procedures of correlated Beta distributed data are limited to two repeated measures or n-repeated measures with stationary means. Therefore, correlated normally distributed data were transformed to be Beta distributed. This transformation added in an extra layer of uncertainty to the simulation. Additionally, the data were simulated at the marginal level (i.e., marginal means and correlations) thereby possibly favoring the GEE model.

The SLMVB had singular Hessian matrices and convergence issues. It is likely that these convergence issues caused bias in the estimation of the location parameter and under coverage of the confidence intervals. Increasing the number of iterations may alleviate some of the convergence issues; however, the log likelihood of the copula was complicated, and non-convergence may have been caused by the precision of the floating numbers determined by the operating system. If the latter were the cause of non-convergence, no amount of increase in the number of iterations in the quasi-Newton-Raphson process would help the model converge. Additionally, the Hessian matrices were estimated using numerical methods. Using second derivatives of the log likelihood could have made the Hessian matrices non-singular; however, this was not a feasible option due to the complexity of the second derivatives. Lastly, it remains unclear the importance that the higher order correlations play in parameter estimation and whether they affected the convergence of the SLMVB models. Therefore, under the scenarios tested, the true performance of the SLMVB models cannot be determined.

It should be noted, that the Beta GLMM had some minor convergence issues that appeared to affect the power of the model for larger effect sizes for a limited number of scenarios. It seems that the default initial values of the dispersion parameter were the cause of the non-convergence. That would suggest that manually setting the initial value of the dispersion parameter of the Beta GLMM would solve the convergence issues; unfortunately, the initial value tends to be data specific.

Furthermore, with the LNMVB we were unable to calculate the correlation. The calculation of the correlation requires the evaluation of a double integral or a generalized Gauss hypergeometric function both of which require numerical methods to solve. The

Gauss hypergeometric function can be estimated using an iterative procedure; however, the computational cost of evaluating this function can be high due to the number of iterations required for the desired accuracy. Therefore, we opted for the double integral that can be solved using the R package pracma.[66] During testing of the simulation, this package performed well. Unfortunately, during the implementation of the full simulation, the double integral was unable to be estimated. However, Gianola and collegues[40] have demonstrated that the Pearson[14] correlation statistic does not adequately measure the association for the LNMVB when there are two repeated measures. Additionally, the LNMVB did not display symmetric results as the location parameter moved away from 0.5. Specifically, the LNMVB was not able to handle overdispersion as the location parameter, μ, approached 1. There was a mean-variance relationship that systematically affected the estimation of the parameters as $\mu \to 1$ and or the variance increased such that $\mu_{ij} \to 1$ for some i, j. This behavior was an effect of the variance being a function of both the mean and the shared parameter, α_0. Clearly, if we assume that $\mu \to 1$ then $\alpha_0 \to 0$ by equation (1.2) which implies that $\sigma^2 \to 0$ by equation (1.3). However, if $\mu \ll 1$ then the parameter space of α_0 is not constrained. Thus, this limitation will only present itself if there is over-inflation on the upper boundary.

5.4. Future Research

Future research should be focused in two areas: 1) the performance of the models with unbalanced data and 2) the handling of observations that are either zero or one.

First, our simulation assumed a balanced design which often is not the case. Therefore, future research should establish the performance of these models with unbalanced data. Unbalanced data could consist of repeated measures not being equally spaced, missing observations, or a combination of the two.

Second, the data were Beta distributed with no observations on the boundary. There are two options to handle zero/one observations; either perform a transformation on the data or use a zero/one inflated model. Additionally, multiple transformations can be applied. It would be of interest to determine how the various strategies perform. Regarding model interpretability, the transformation tends to be the models that are easiest to describe to non-statistical researchers as opposed to zero/one inflated models. Therefore, it is prudent to determine whether transformations to the data bias the results.

6. Bibliography

1. Fitzmaurice GM, Laird NM, Ware JH. *Applied longitudinal analysis.* Vol 998. John Wiley & Sons; 2012.

2. Liang K, Zeger SL. Longitudinal data analysis using generalized linear models. *Biometrika*. 1986;73(1):13-22.

3. Gardiner JC, Luo Z, Roman LA. Fixed effects, random effects and GEE: What are the differences? *Stat Med*. 2009;28(2):221-239.

4. Zimprich D. Modeling change in skewed variables using mixed beta regression models. *Research in Human Development*. 2010;7(1):9-26.

5. Chen J, Zhang D, Davidian M. A monte carlo EM algorithm for generalized linear mixed models with flexible random effects distribution. *Biostatistics*. 2002;3(3):347-360.

6. Libby DL, Novick MR. Multivariate generalized beta distributions with applications to utility assessment. *Journal of Educational Statistics*. 1982;7(4):271-294.

7. Sarmanov O. Generalized normal correlation and 2-dimensional frechet-classes. *Dokl Akad Nauk SSSR*. 1966;168(1):32-&.

8. Ting Lee M. Properties and applications of the sarmanov family of bivariate distributions. *Communications in Statistics-Theory and Methods*. 1996;25(6):1207-1222.

9. de Souza DF, da Silva Moura, Fernando Antônio. Multivariate beta regression. .

10. Adell N, Puig P, Rojas-Olivares A, Caja G, Carné S, Salama AA. A bivariate model for retinal image identification in lambs. *Comput Electron Agric*. 2012;87:108-112.

11. Nadarajah S. A new bivariate beta distribution with application to drought data. *Metron*. 2007;65(2):153-174.

12. Cepeda-Cuervo E, Achcar JA, Lopera LG. Bivariate beta regression models: Joint modeling of the mean, dispersion and association parameters. *Journal of Applied Statistics*. 2014;41(3):677-687.

13. Wang M, Rennolls K. Bivariate distribution modeling with tree diameter and height data. *For Sci*. 2007;53(1):16-24.

14. Pearson K. Mathematical contributions to the theory of evolution. III. regression, heredity, and panmixia. *Philosophical Transactions of the Royal Society of London.Series A, containing papers of a mathematical or physical character*. 1896;187:253-318.

15. Heagerty P, Liang K, Zeger S, Diggle P. *Analysis of longitudinal data.* New York: Oxford University Press; 2002.

16. Shults J, Chaganty NR. Analysis of serially correlated data using quasi-least squares. *Biometrics*. 1998:1622-1630.

17. Pan W. Model selection in estimating equations. *Biometrics*. 2001;57(2):529-534.

18. Hin L, Wang Y. Working-correlation-structure identification in generalized estimating equations. *Stat Med*. 2009;28(4):642-658.

19. Tuerlinckx F, Rijmen F, Verbeke G, Boeck P. Statistical inference in generalized linear mixed models: A review. *Br J Math Stat Psychol*. 2006;59(2):225-255.

20. Paolino P. Maximum likelihood estimation of models with beta-distributed dependent variables. *Political Analysis*. 2001;9(4):325-346.

21. Ferrari S, Cribari-Neto F. Beta regression for modelling rates and proportions. *Journal of Applied Statistics*. 2004;31(7):799-815.

22. Vořechovský M. Simulation of simply cross correlated random fields by series expansion methods. *Struct Saf*. 2008;30(4):337-363.

23. Morgello S, Gelman B, Kozlowski P, et al. The national NeuroAIDS tissue consortium: A new paradigm in brain banking with an emphasis on infectious disease. *Neuropathol Appl Neurobiol*. 2001;27(4):326-335.

24. National NeuroAIDS tissue consortium. https://nntc.org. Updated n.d. Accessed March 11, 2019.

25. Heaton RK, Grant I, Matthews CG, Fastenau PS, Adams KM. Heaton, grant, and matthews' comprehensive norms: An overzealous attempt. *Journal of Clinical and Experimental Neuropsychology*. 1996;18(3):444-448.

26. Benedict RH, Schretlen D, Groninger L, Brandt J. Hopkins verbal learning Test–Revised: Normative data and analysis of inter-form and test-retest reliability. *Clin Neuropsychol*. 1998;12(1):43-55.

27. Olkin I, Trikalinos TA. Constructions for a bivariate beta distribution. *Statistics & Probability Letters*. 2015;96:54-60.

28. McCullagh P, Nelder J. *Generalized linear models (2nd edn) monographs on statistics and applied probability 37.* London, Chapman & Hall; 1989.

29. Gupta R. Multivariate beta distribution. In: *A modern course on statistical distributions in scientific work.* Springer; 1975:337-344.

30. Khattree R, Gupta RD. Some probability distributions connected with beta and gamma matrices. *Communications in Statistics-Theory and Methods*. 1992;21(2):369-390.

31. Mitra SK. A density-free approach to the matrix variate beta distribution. *Sankhyā: The Indian Journal of Statistics, Series A*. 1970:81-88.

32. Konno Y. Exact moments of the multivariate F and beta distributions. *日本統計学会誌*. 1988;18(2):123-130.

33. Olkin I, Rubin H. Multivariate beta distributions and independence properties of the wishart distribution. *The Annals of Mathematical Statistics*. 1964:261-269.

34. Tan W. Note on the multivariate and the generalized multivariate beta distributions. *Journal of the American Statistical Association*. 1969;64(325):230-241.

35. Gupta AK, Nagar DK. Matrix-variate beta distribution. *International Journal of Mathematics and Mathematical Sciences*. 2000;24(7):449-459.

36. Sakurai T. Limiting distributions of high-dimensional multivariate beta-type distributions. *Journal of Multivariate Analysis*. 2012;111:110-119.

37. Jones M. Multivariate t and beta distributions associated with the multivariate F distribution. *Metrika*. 2002;54(3):215-231.

38. Olkin I, Liu R. A bivariate beta distribution. *Statistics & Probability Letters*. 2003;62(4):407-412. doi: https://doi.org/10.1016/S0167-7152(03)00048-8.

39. Nagar DK, Orozco-Castañeda JM, Gupta AK. Product and quotient of correlated beta variables. *Applied Mathematics Letters*. 2009;22(1):105-109.

40. Gianola D, Manfredi E, Simianer H. On measures of association among genetic variables. *Anim Genet*. 2012;43(s1):19-35.

41. Arnold BC, Ng HKT. Flexible bivariate beta distributions. *Journal of Multivariate Analysis*. 2011;102(8):1194-1202.

42. Gupta AK, Orozco-Castañeda JM, Nagar DK. Non-central bivariate beta distribution. *Statistical papers*. 2011;52(1):139-152.

43. Nadarajah S. The bivariate F 3-beta distribution. *Communications of the Korean Mathematical Society*. 2006;21(2):363-374.

44. Crackel R, Flegal J. Bayesian inference for a flexible class of bivariate beta distributions. *Journal of Statistical Computation and Simulation*. 2017;87(2):295-312.

45. Nadarajah S, Kotz S. Some bivariate beta distributions. *Statistics*. 2005;39(5):457-466.

46. Zhu Y, Ghosh SK, Goodwin BK. Modeling dependence in the design of whole farm insurance contract,| A copula-based model approach. . 2008:27-29.

47. Nelsen RB. Properties and applications of copulas: A brief survey. . 2003:10.

48. Sklar A. **Fonctions de répartition À N dimensions et leurs marges**. *Publications de l'Institut de Statistique de L'Université de Paris*. 1959;8:229-231.

49. Nelsen RB. *An introduction to copulas.* Springer Science & Business Media; 2007.

50. Schoelzel C, Friederichs P. Multivariate non-normally distributed random variables in climate research–introduction to the copula approach. *Nonlinear Processes in Geophysics*. 2008;15(5):761-772.

51. Kolev N, Anjos Ud, Mendes, Beatriz Vaz de M. Copulas: A review and recent developments. *Stochastic Models*. 2006;22(4):617-660.

52. Park Y, Fader PS. Modeling browsing behavior at multiple websites. *Marketing Science*. 2004;23(3):280-303.

53. Shaked M. A concept of positive dependence for exchangeable random variables. *The Annals of Statistics*. 1977;5(3):505-515.

54. Danaher PJ. A canonical expansion model for multivariate media exposure distributions: A generalization of the" duplication of viewing law". *J Market Res*. 1991:361-367.

55. Shubina M, Lee MT. On maximum attainable correlation and other measures of dependence for the sarmanov family of bivariate distributions. *Communications in Statistics-Theory and Methods*. 2004;33(5):1031-1052.

56. Prentice RL. Correlated binary regression with covariates specific to each binary observation. *Biometrics*. 1988:1033-1048.

57. Chen Y, Luo S, Chu H, Wei P. Bayesian inference on risk differences: An application to multivariate meta-analysis of adverse events in clinical trials. *Statistics in biopharmaceutical research*. 2013;5(2):142-155.

58. Danaher PJ, Hardie BGS. Bacon with your eggs? applications of a new bivariate beta-binomial distribution. *The American Statistician*. 2005;59(4):282-286.

59. Shoukri MM, ElDali A, Donner A. Measures of family resemblance for binary traits: Likelihood based inference. *The international journal of biostatistics*. 2012;8(1).

60. Shoukri M, Collison K, Al-Mohanna F. Testing of gender differences on sib-sib correlations for binary traits: Likelihood based inference with application to arterial blood pressures data. *J Biomet Biostat*. 2014;5(186):2.

61. Hunger M, Döring A, Holle R. Longitudinal beta regression models for analyzing health-related quality of life scores over time. *BMC medical research methodology*. 2012;12(1):144.

62. Dennis Jr JE, Schnabel RB. *Numerical methods for unconstrained optimization and nonlinear equations.* Vol 16. Siam; 1996.

63. Cline AK, Moler CB, Stewart GW, Wilkinson JH. An estimate for the condition number of a matrix. *SIAM Journal on Numerical Analysis*. 1979;16(2):368-375.

64. R Core Team. R: A language and environment for statistical computing. . 2017. https://www.R-project.org/.

65. Hasselman B. Nleqslv: Solve systems of nonlinear equations. . 2017. R package version 3.3.1. https://CRAN.R-project.org/package=nleqslv;.

66. Borchers HW. Pracma: Practical numerical math functions. . 2018. R package version 2.1.5. https://CRAN.R-project.org/package=pracma;.

67. Wang J, Zheng N. Measures of correlation for multiple variables. *arXiv preprint arXiv:1401.4827*. 2014.

68. Pearson K. Contribution to the mathematical theory of evolution: I disection of frequency curves. *Philosophical Transactions of the Royal Society of London*. 1894.

69. Johnson SG. The NLopt nonlinear-optimization package. . . http://ab-initio.mit.edu/nlopt.

70. Cohen J. *Statistical power analysis for the behavioral sciences. 2nd.* Hillsdale, NJ: erlbaum; 1988.

71. Ross SM. *Simulation.* Fifth ed. Elsevier; 2013.

72. McDaniel LS, Henderson NC, Rathouz PJ. Fast pure {R} implementation of {GEE}: Application of the {M}atrix package. *{The {R} Journal*. 2013;5(1):181-187. https://journal.r-project.org/archive/2013-1/mcdaniel-henderson-rathouz.pdf.

73. Rizopoulos D. GLMMadaptive: Generalized linear mixed models using adaptive gaussian. . 2018. R package version 0.2-0. https://CRAN.R-project.org/package=GLMMadaptive;.

74. Nakagawa S, Johnson PCD, Schielzeth H. The coefficient of determination R(2) and intra-class correlation coefficient from generalized linear mixed-effects models revisited and expanded. *J R Soc Interface*. 2017;14(134):10.1098/rsif.2017.0213. Epub 2017 Sep 13.

Appendix A: One Group Simulation Results

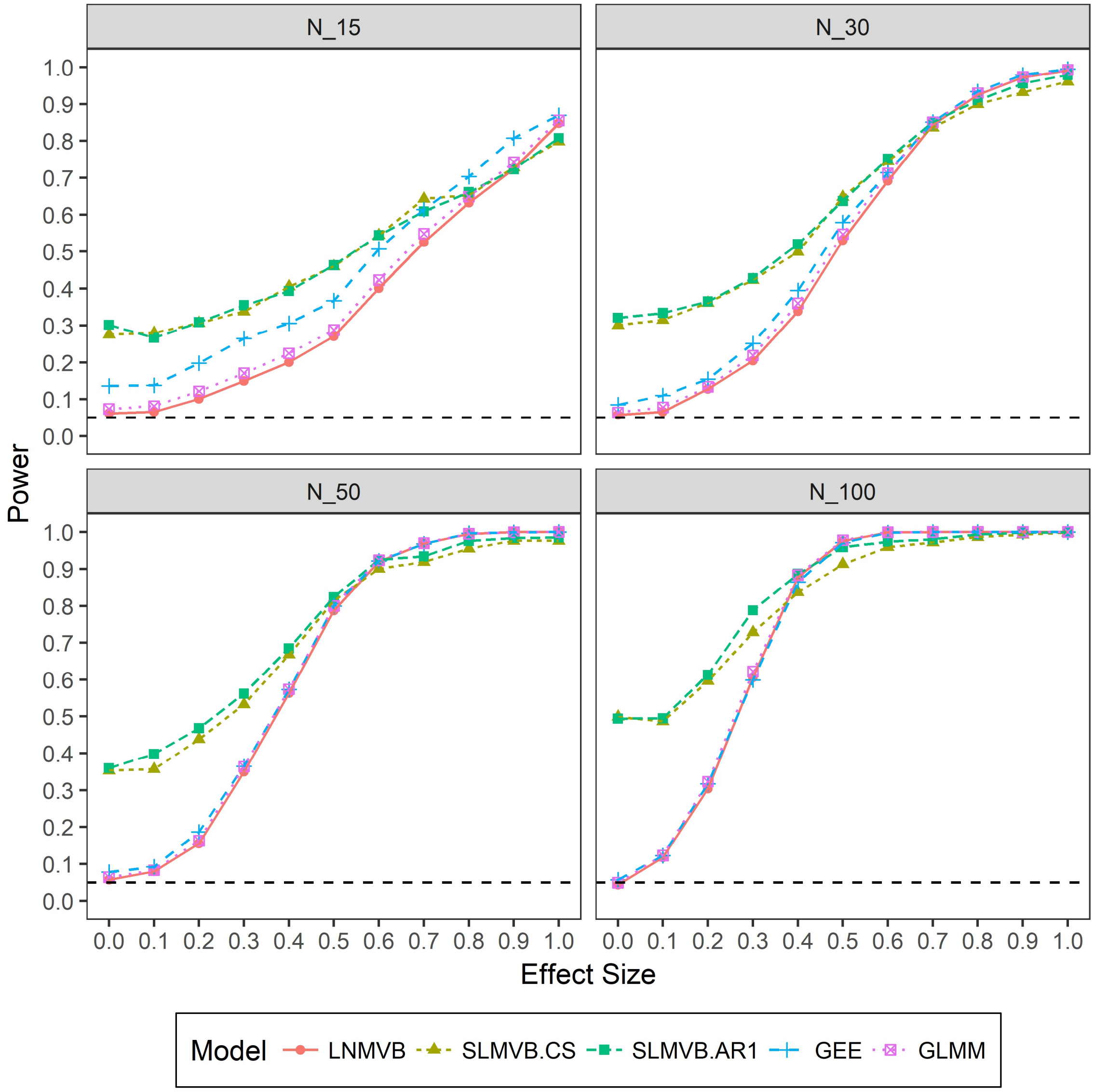


Figure A.1 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

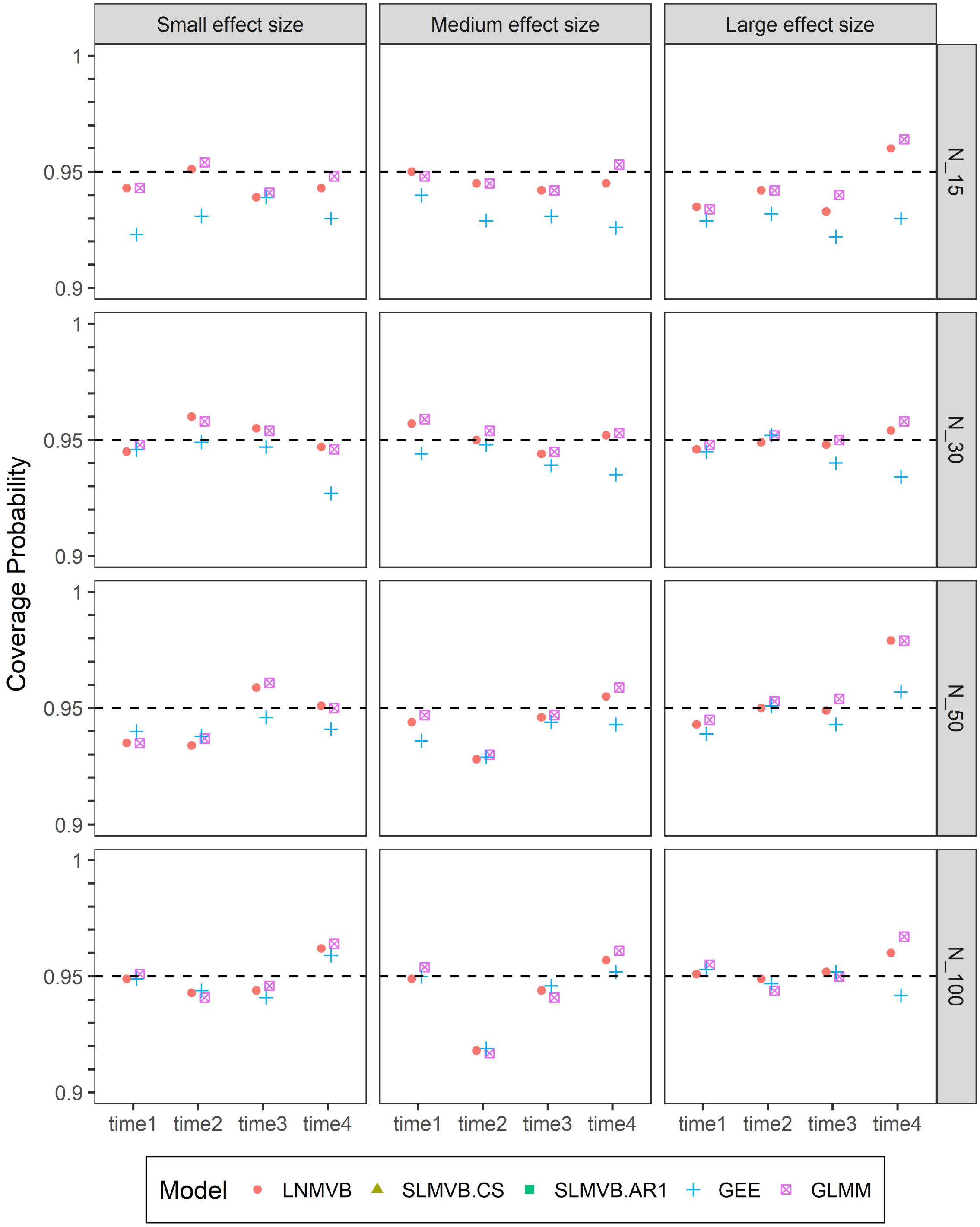


Figure A.2 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

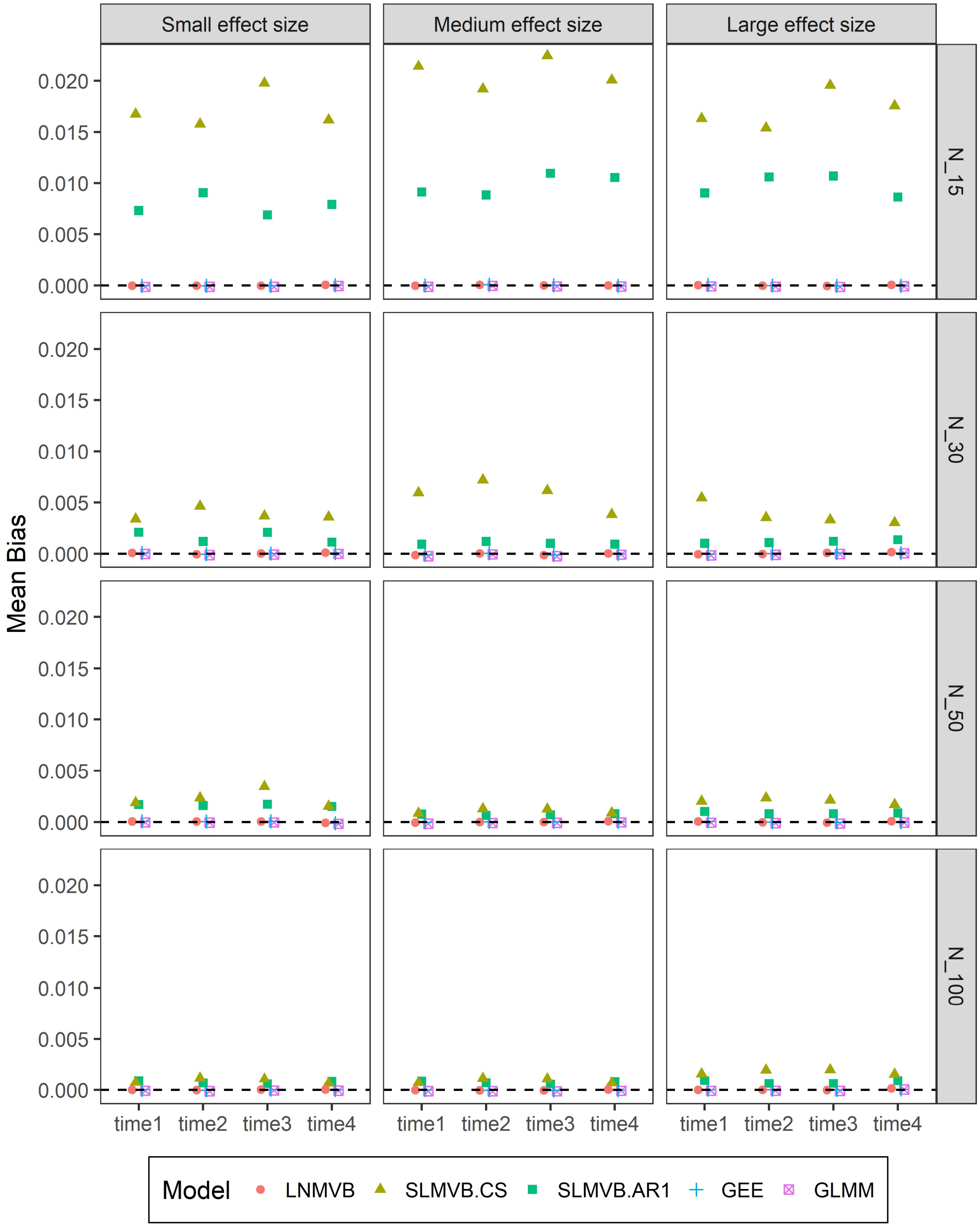


Figure A.3 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

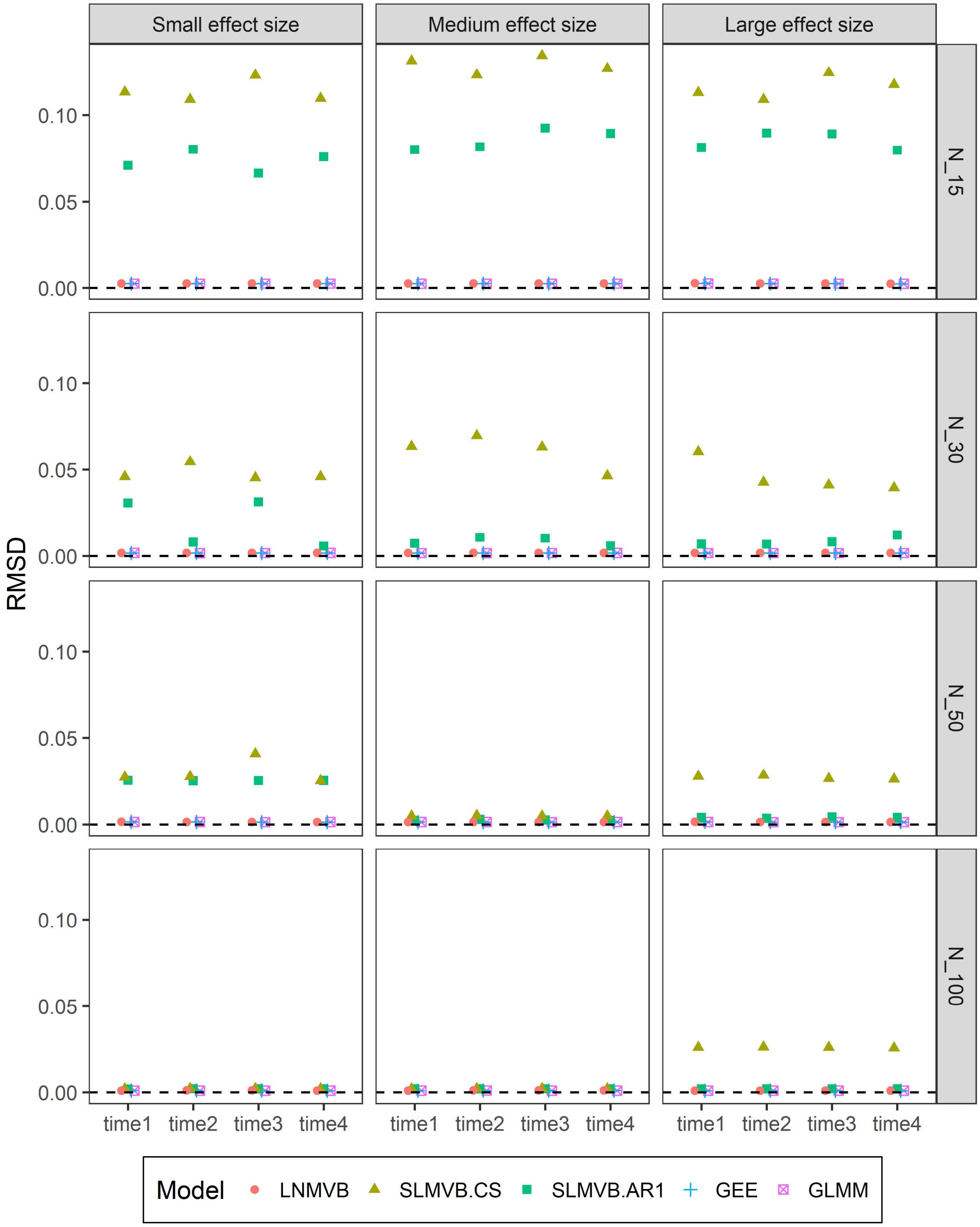


Figure A.4 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

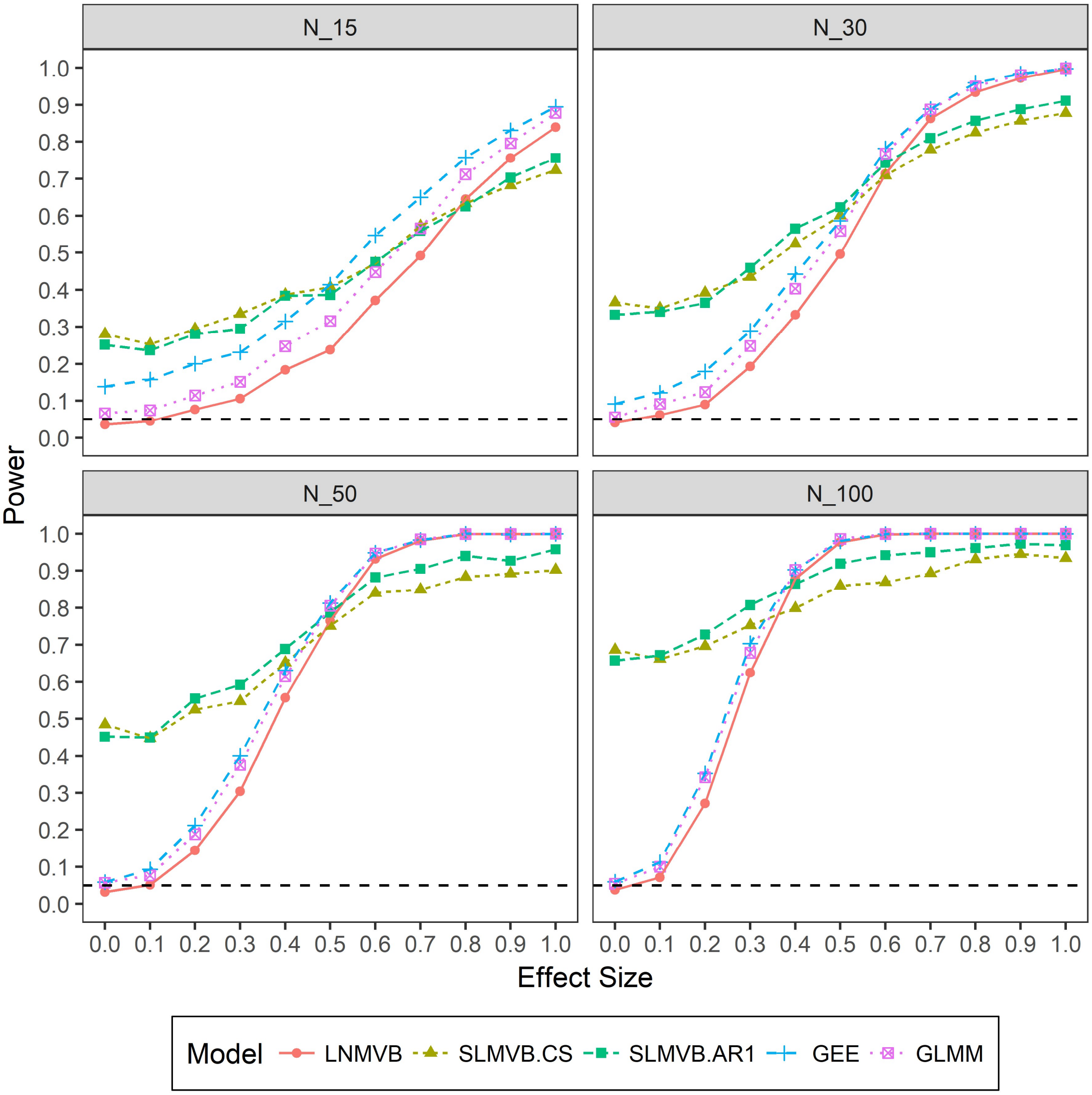


Figure A.5 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

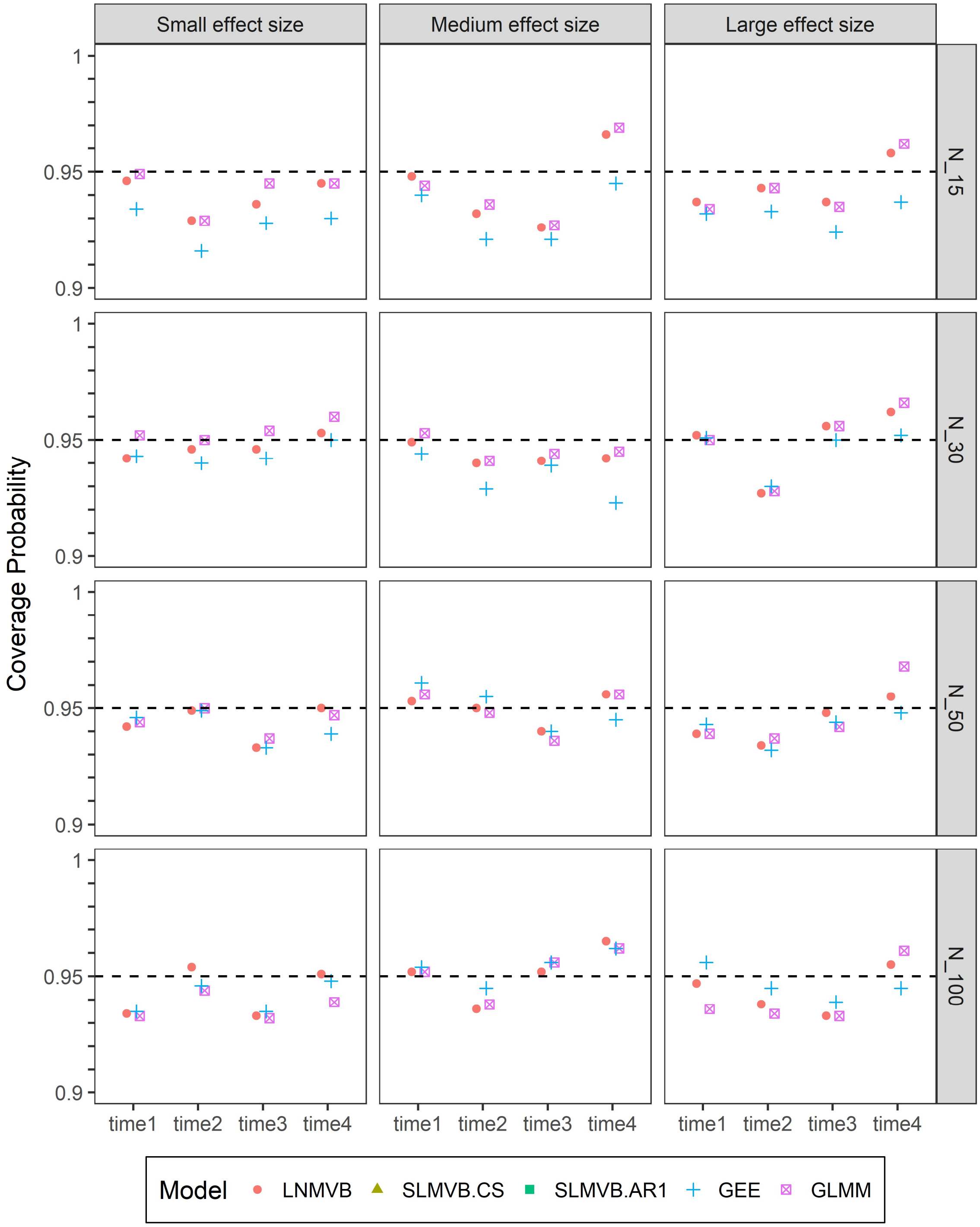


Figure A.6 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

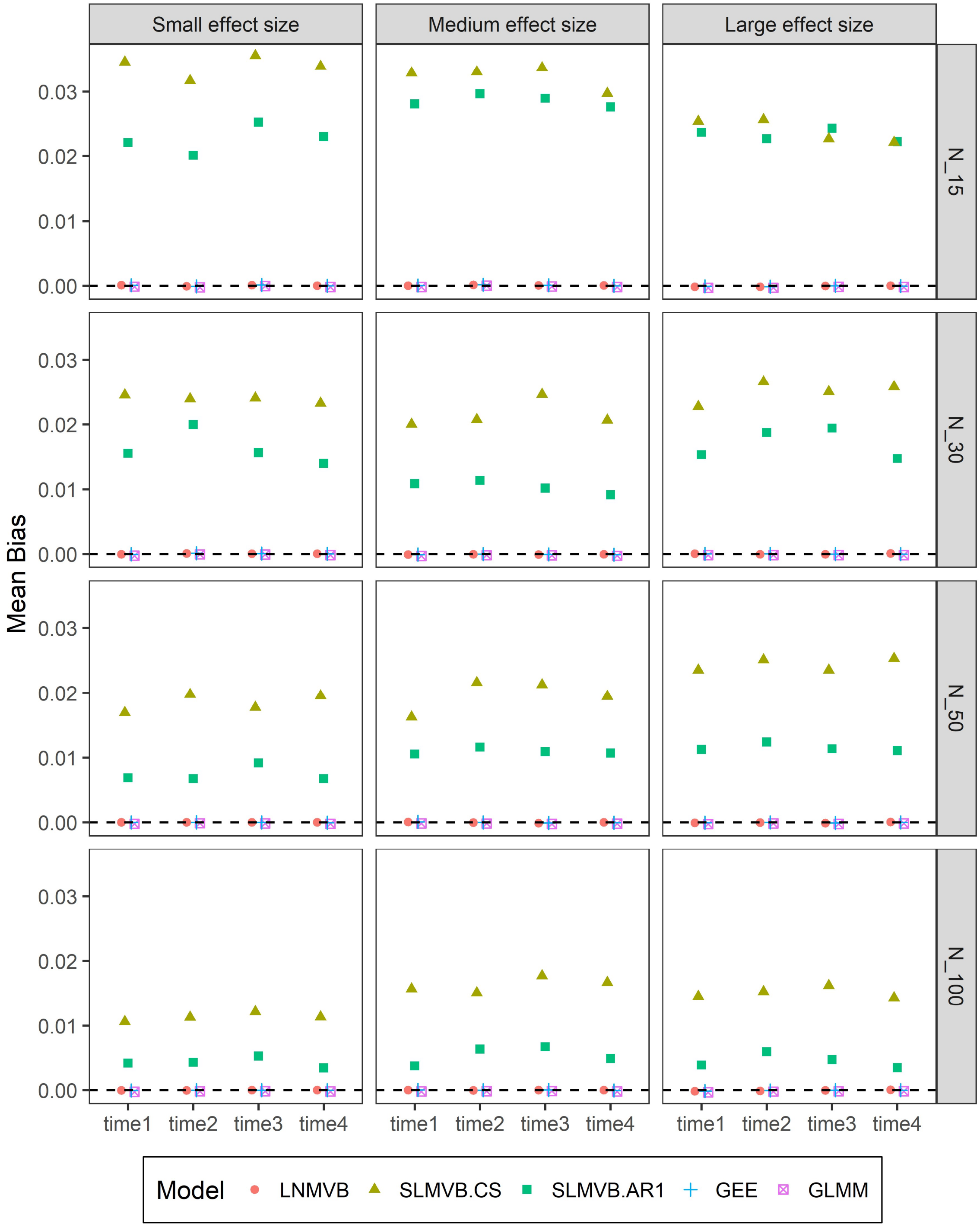


Figure A.7 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3$.

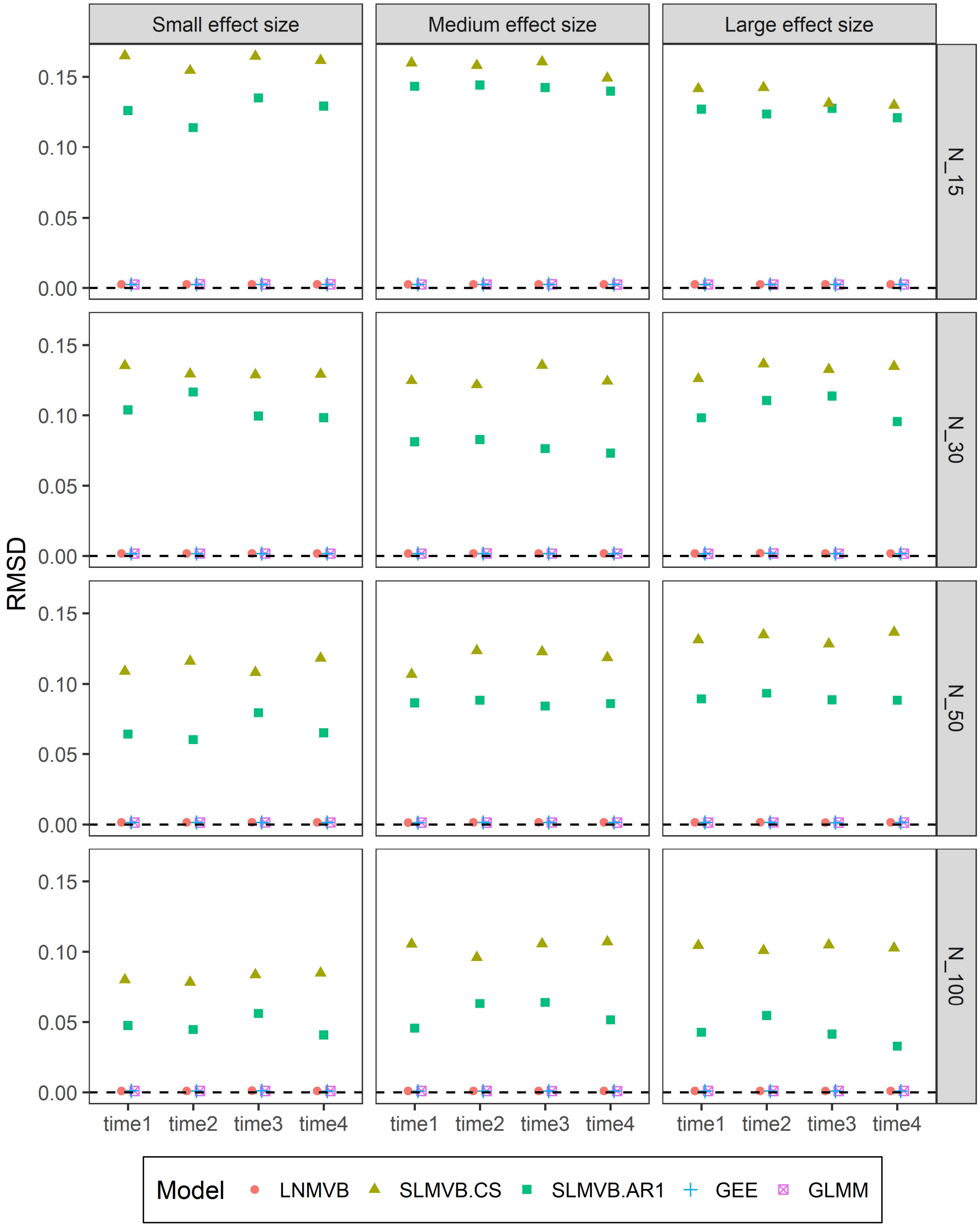


Figure A.8 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3$.

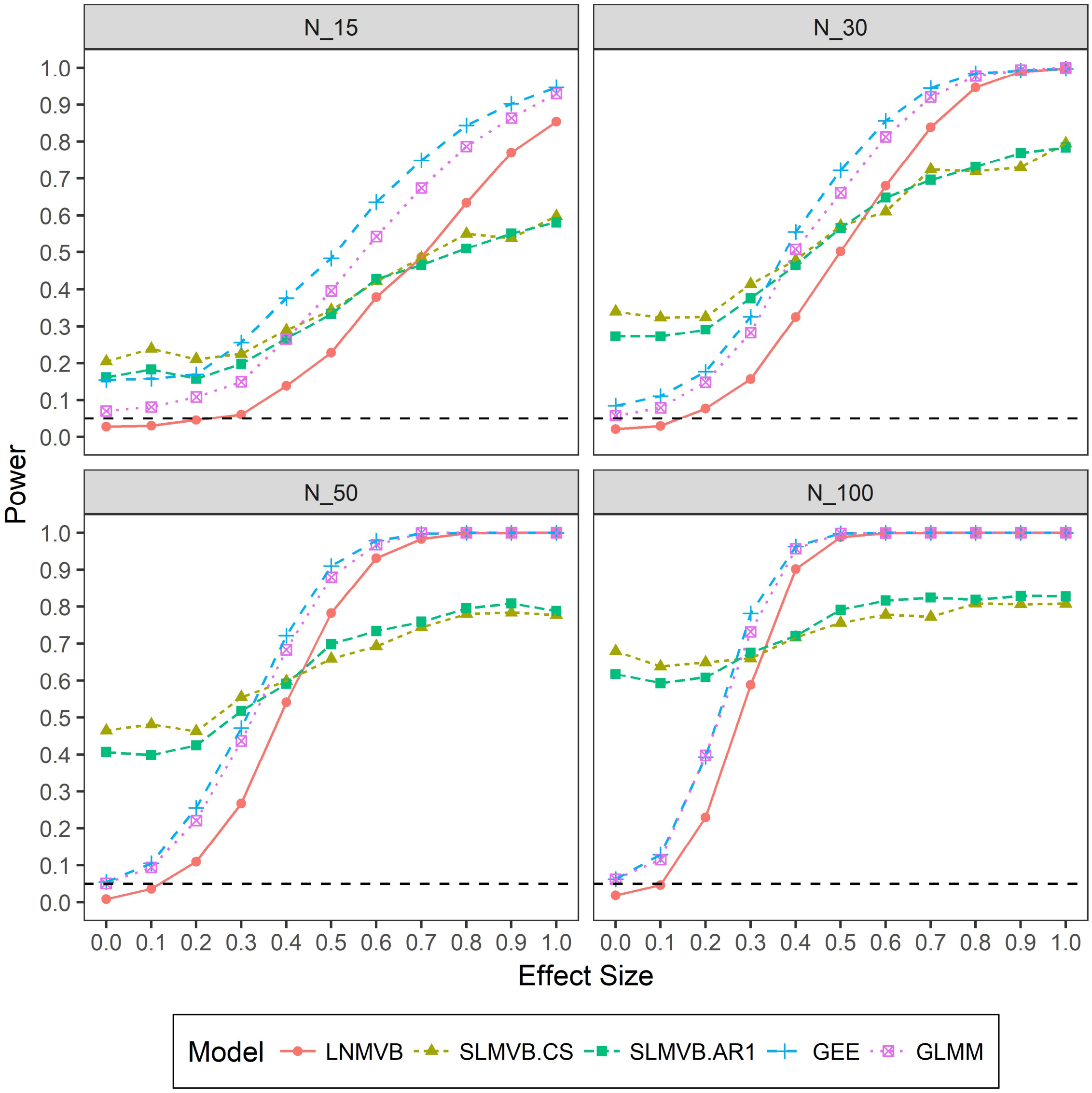


Figure A.9 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

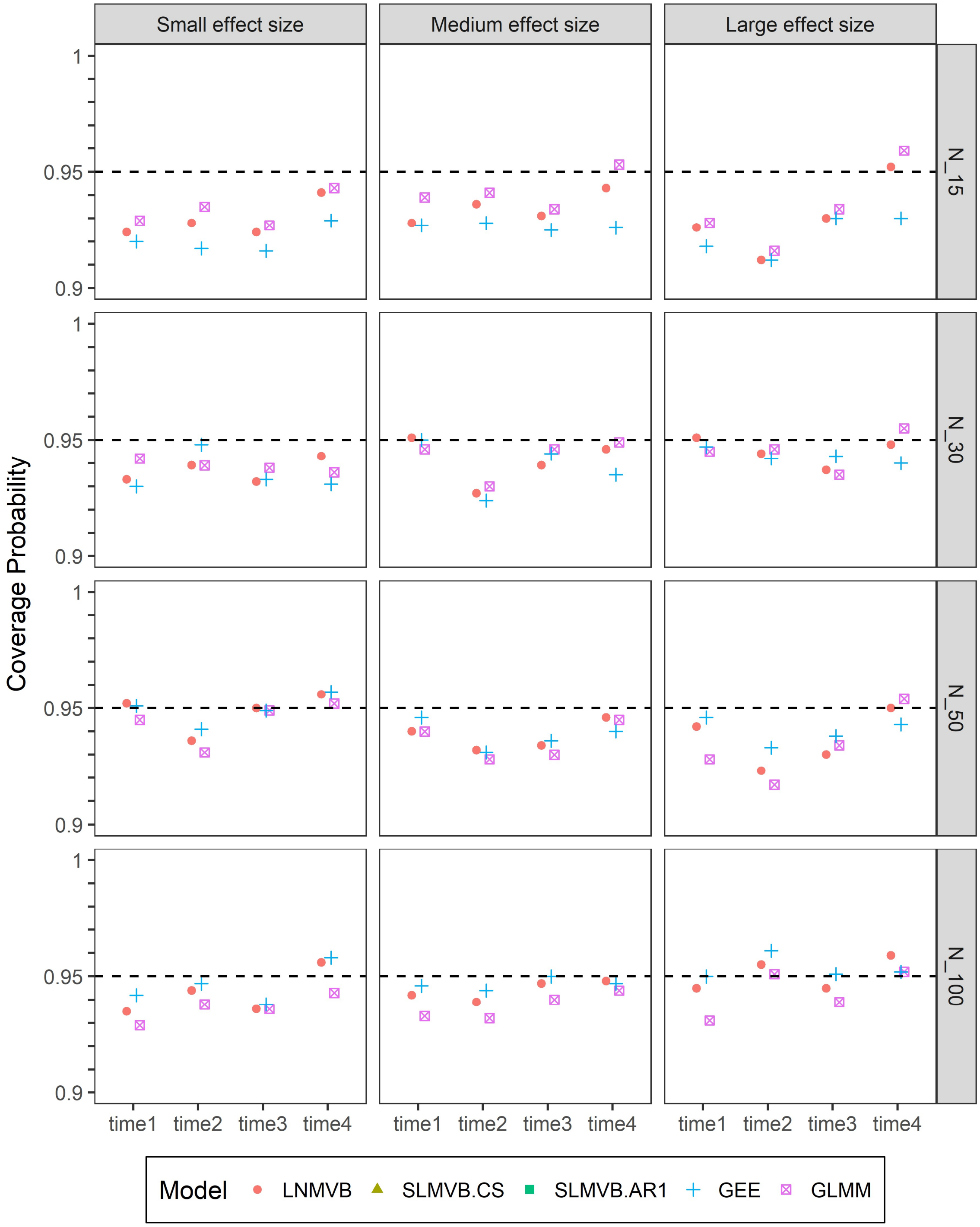


Figure A.10 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

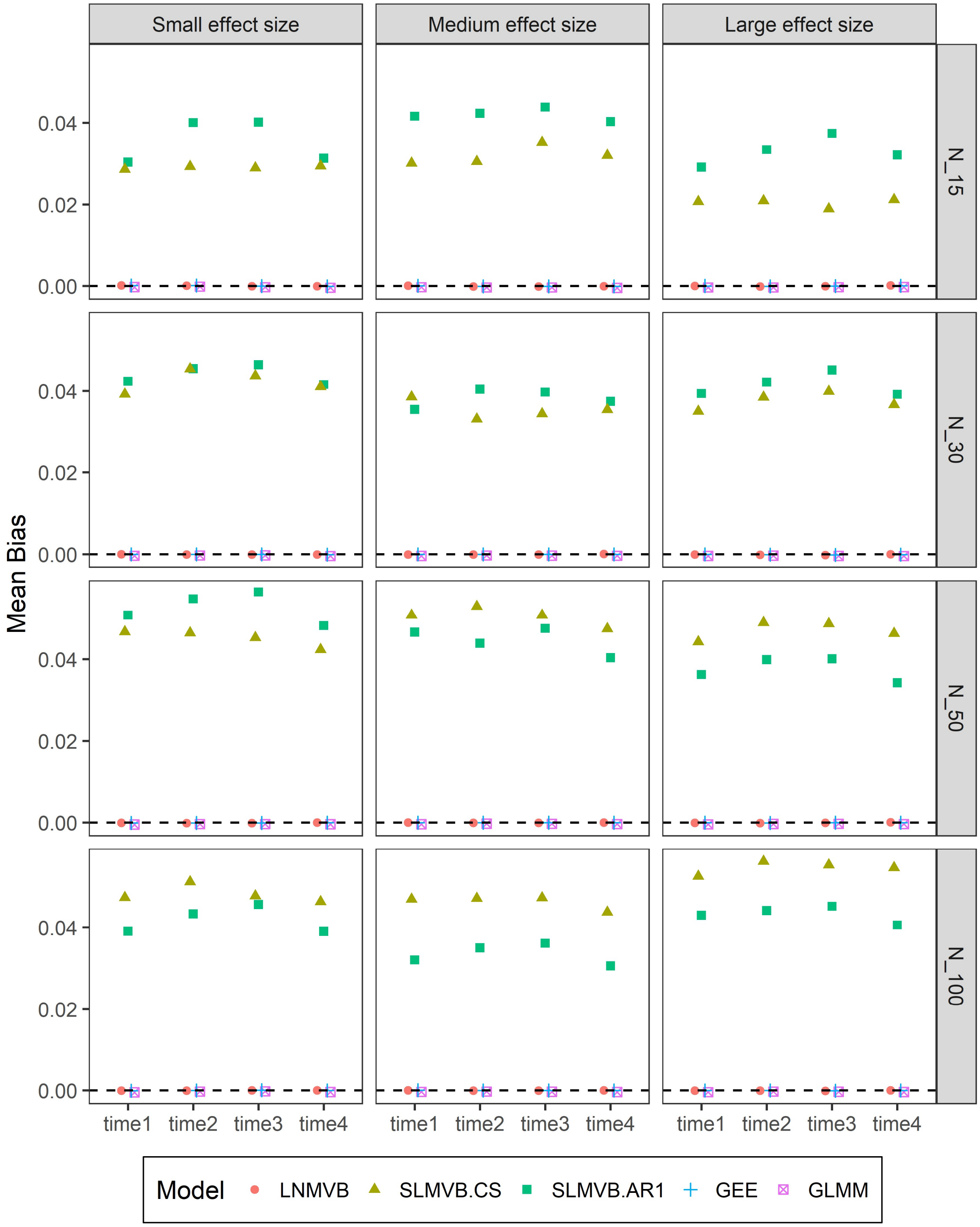


Figure A.11 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5$.

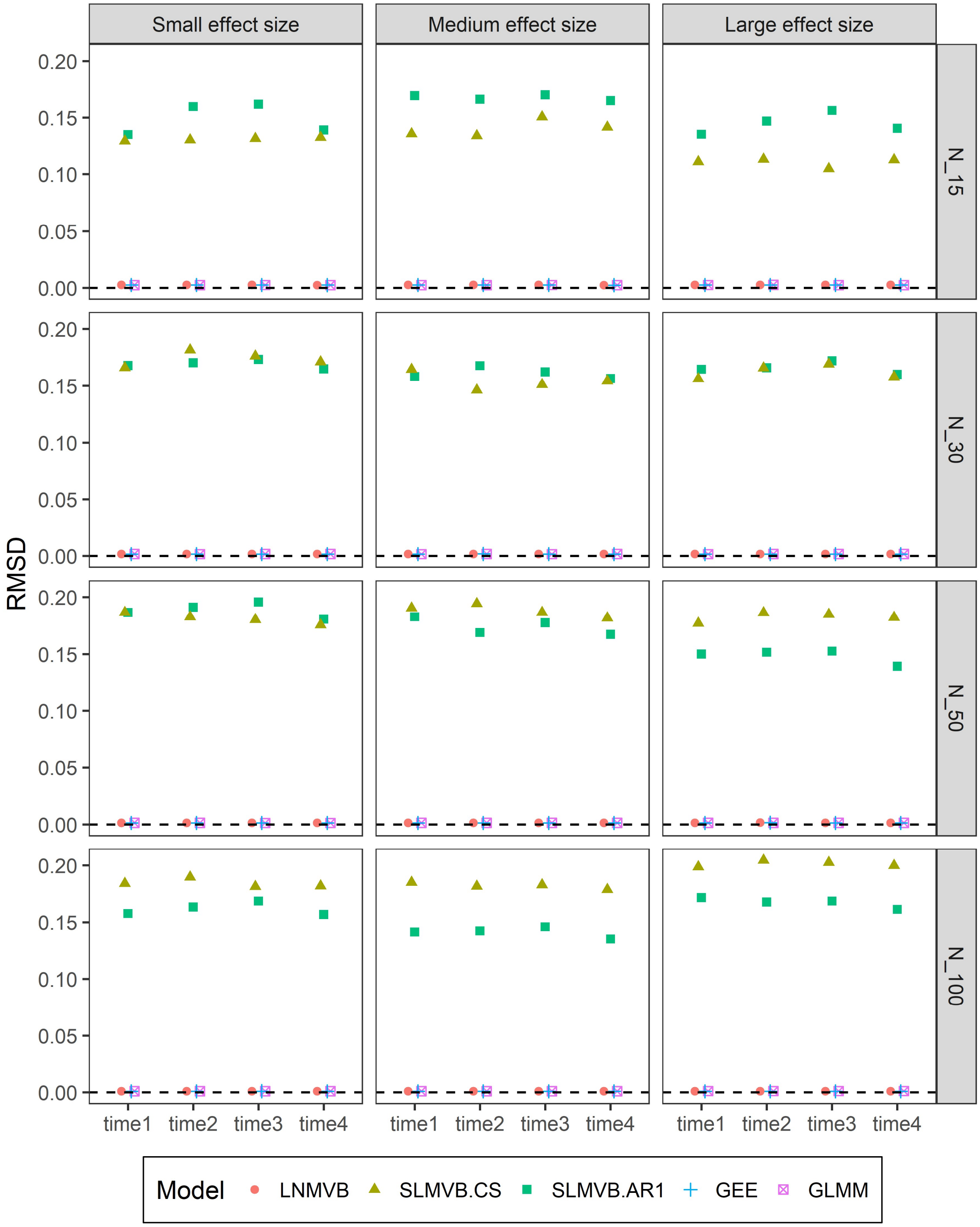


Figure A.12 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5$.

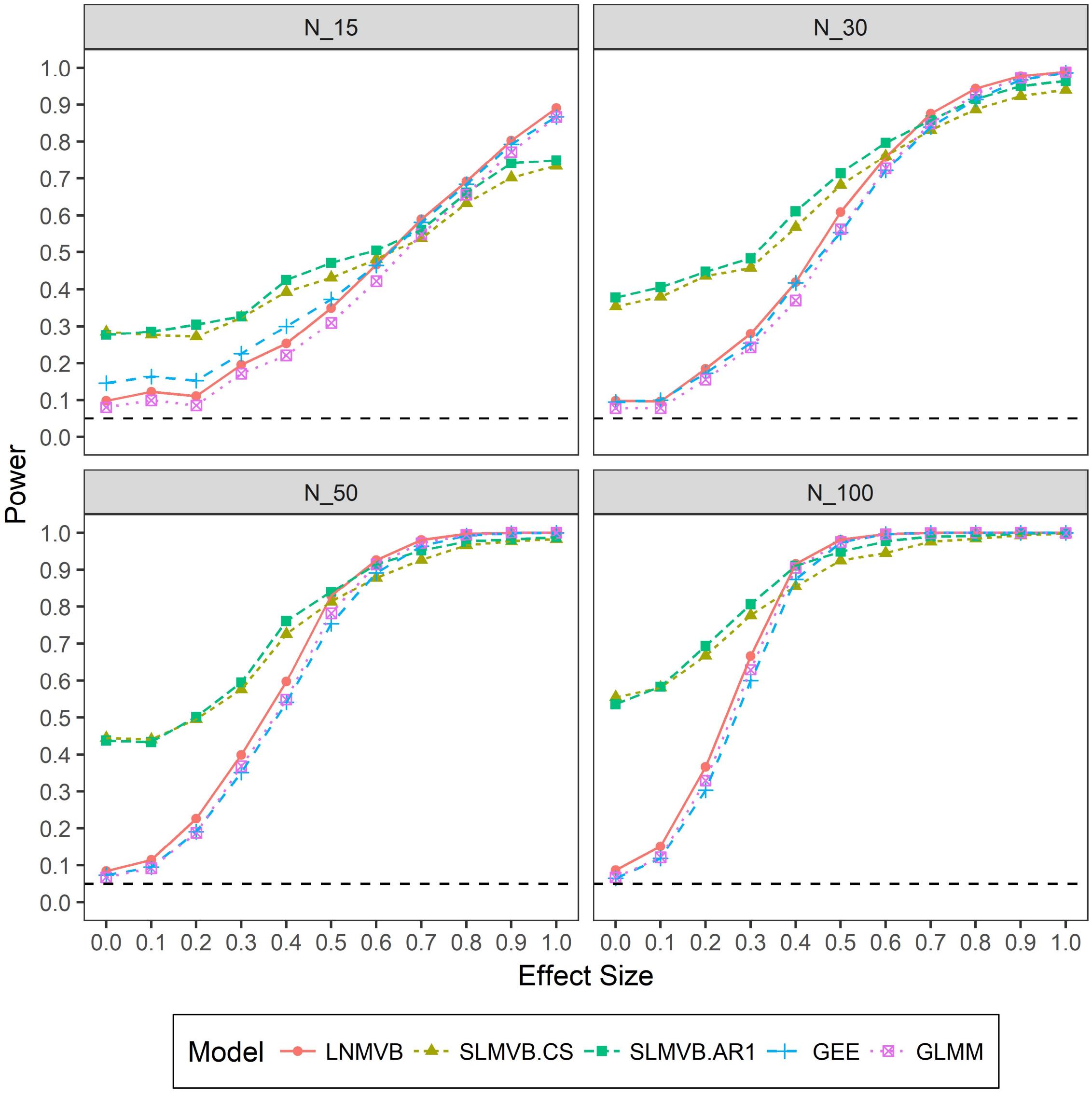


Figure A.13 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

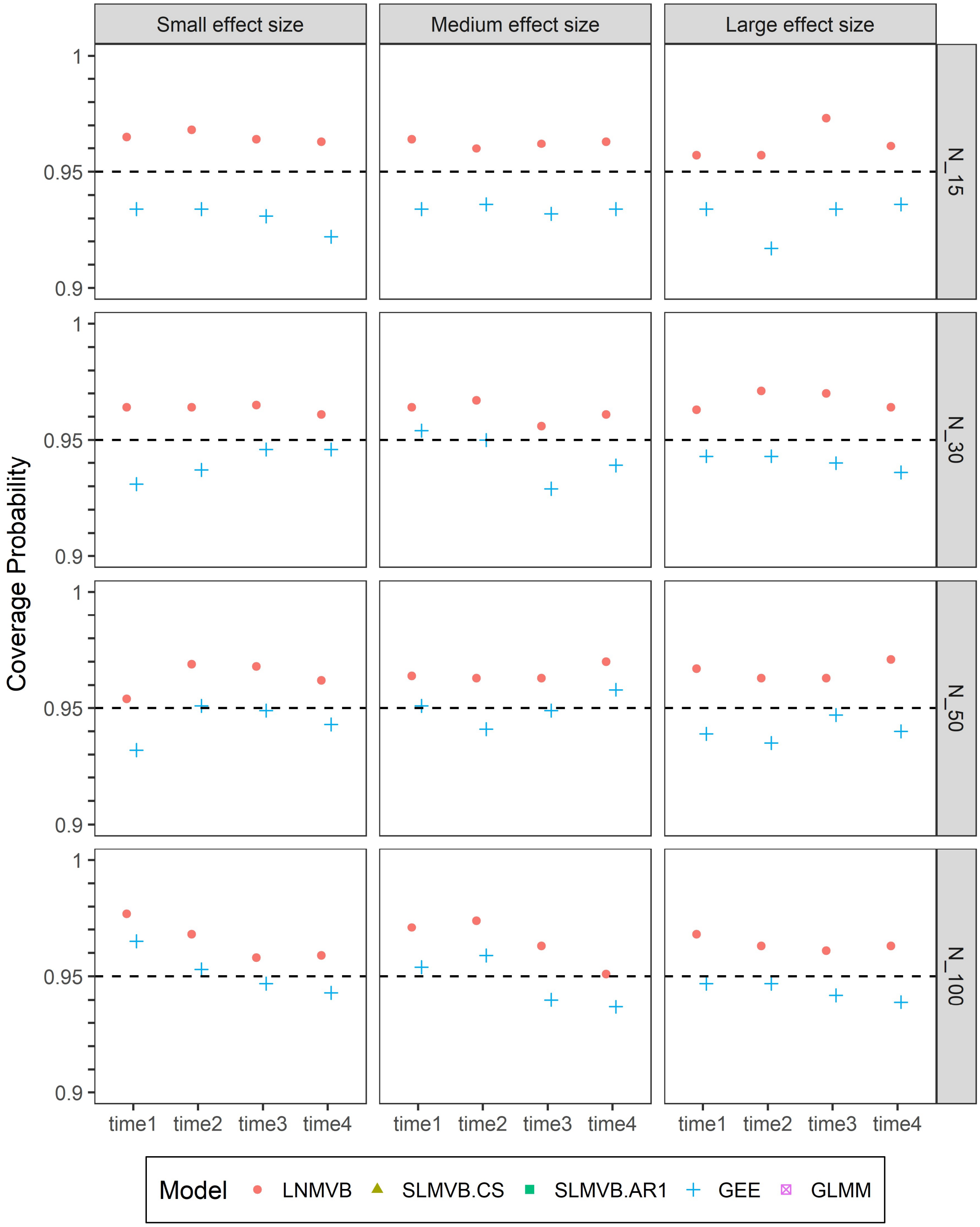


Figure A.14 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

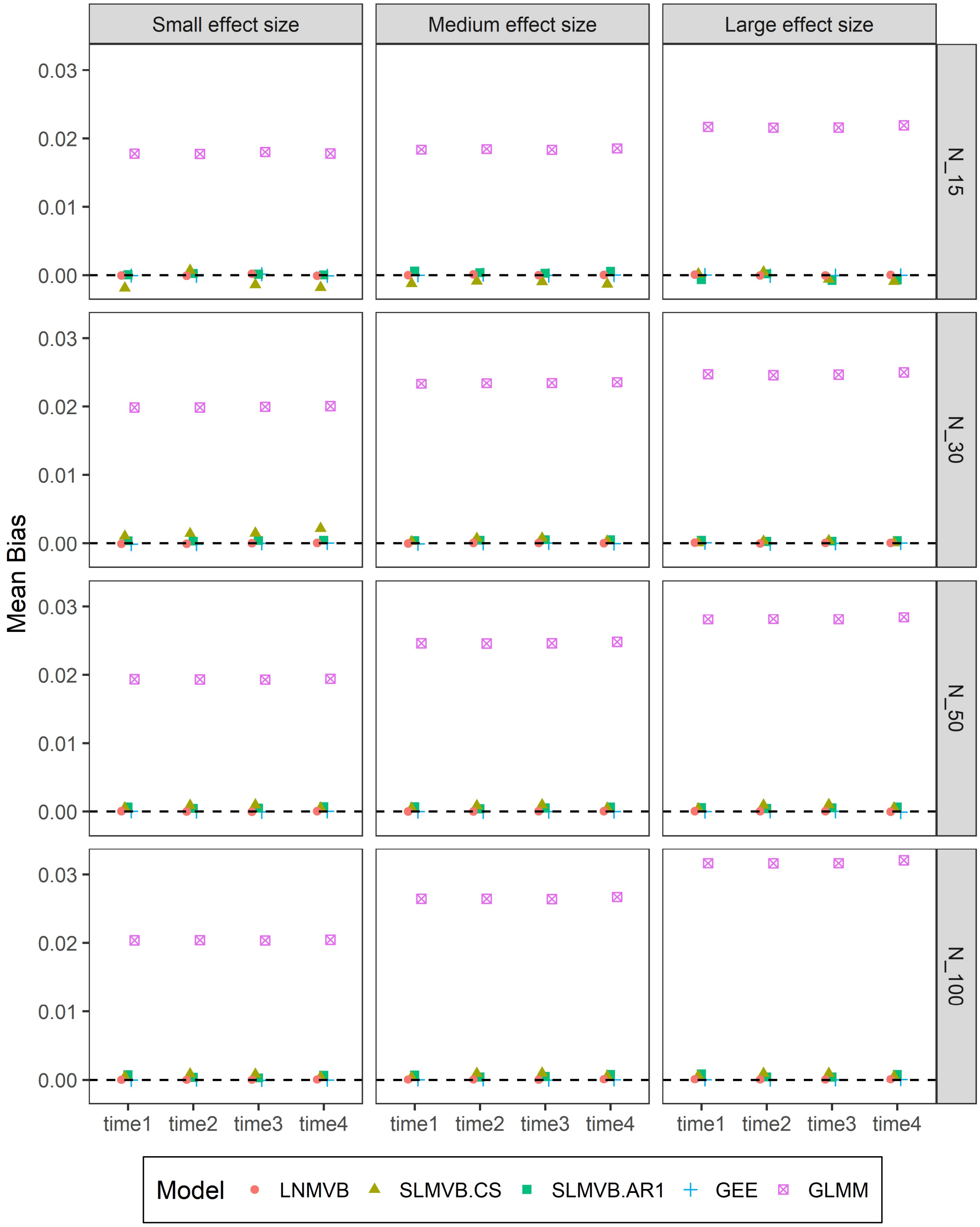


Figure A.15 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$.

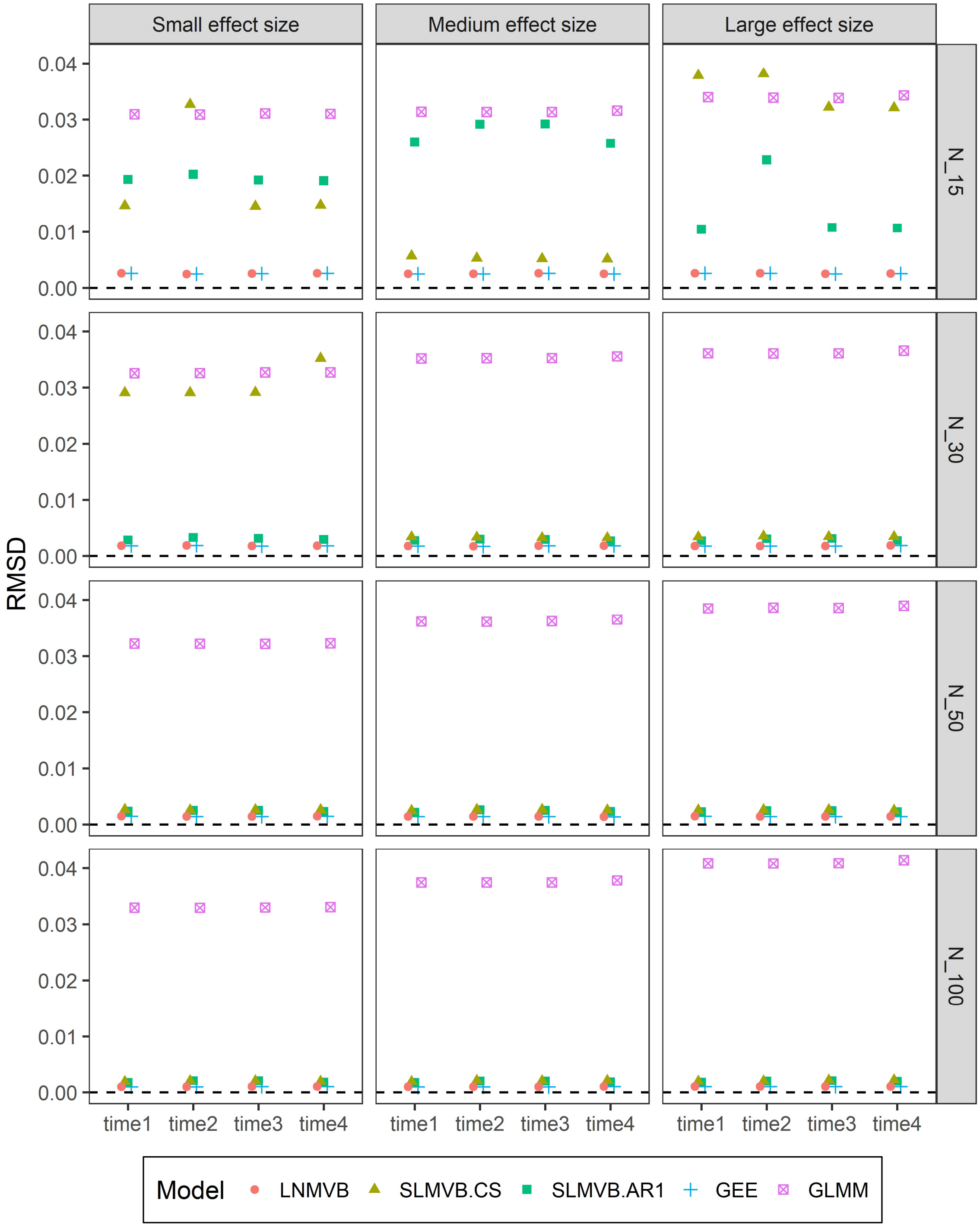


Figure A.16 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$.

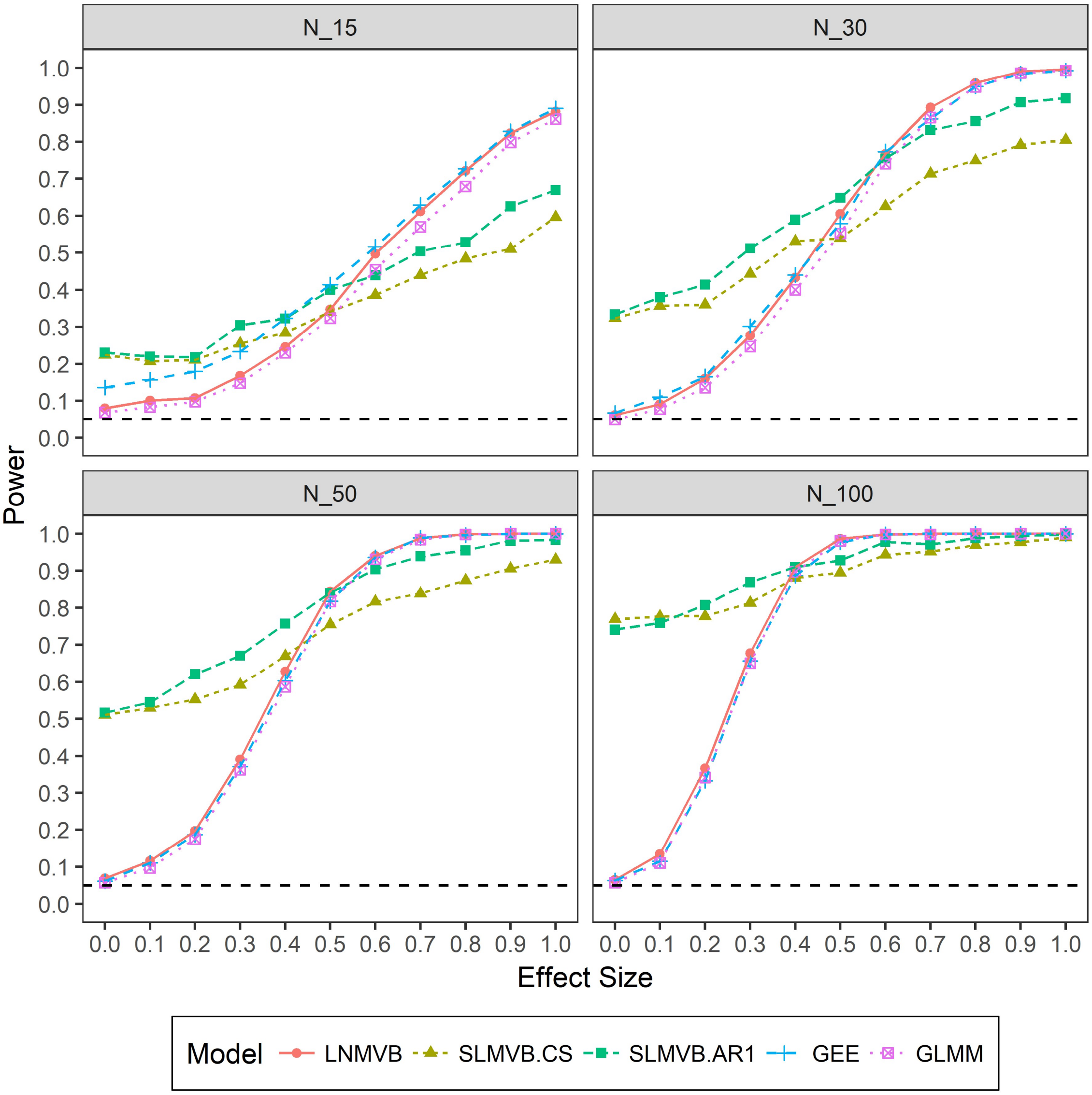


Figure A.17 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

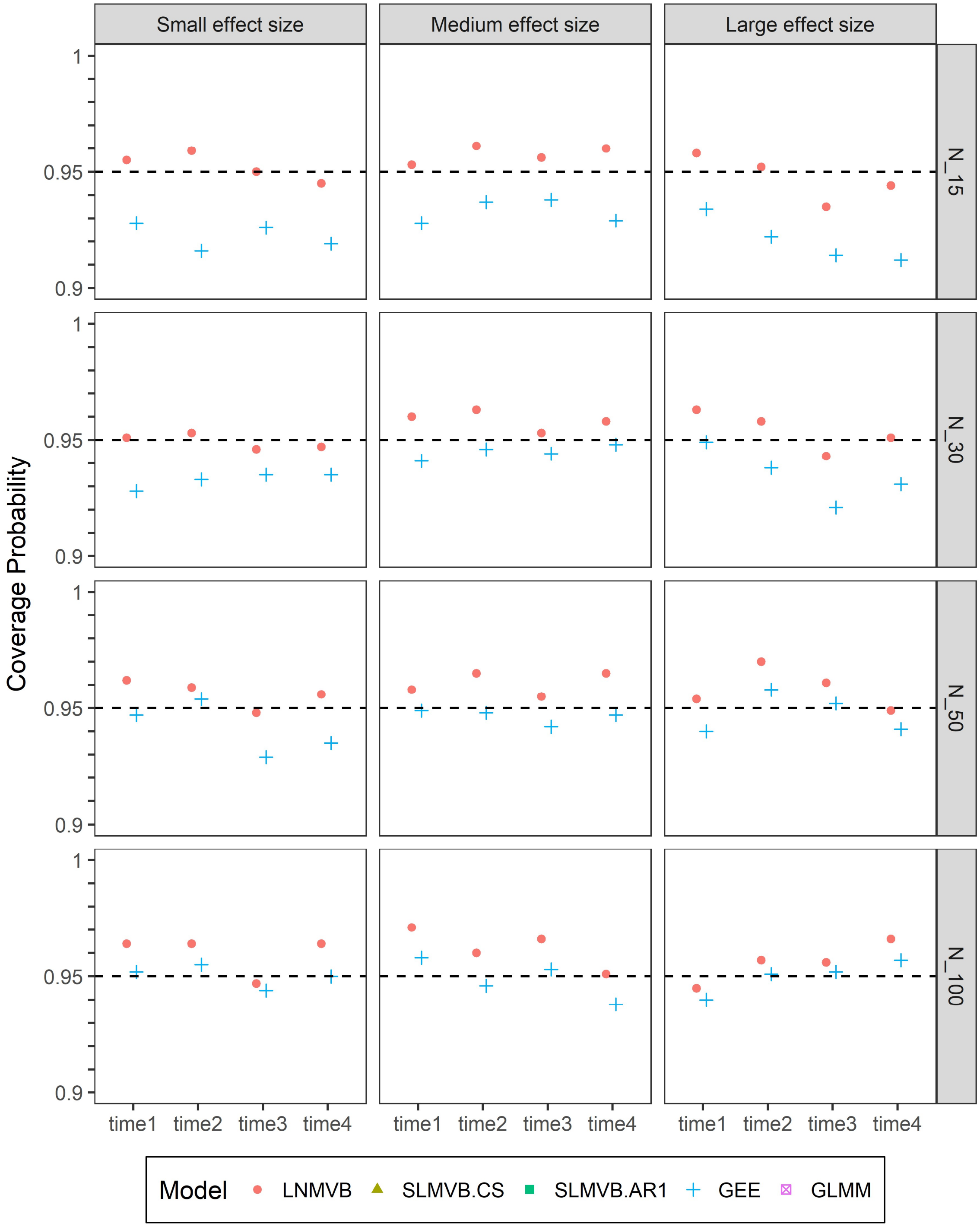


Figure A.18 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

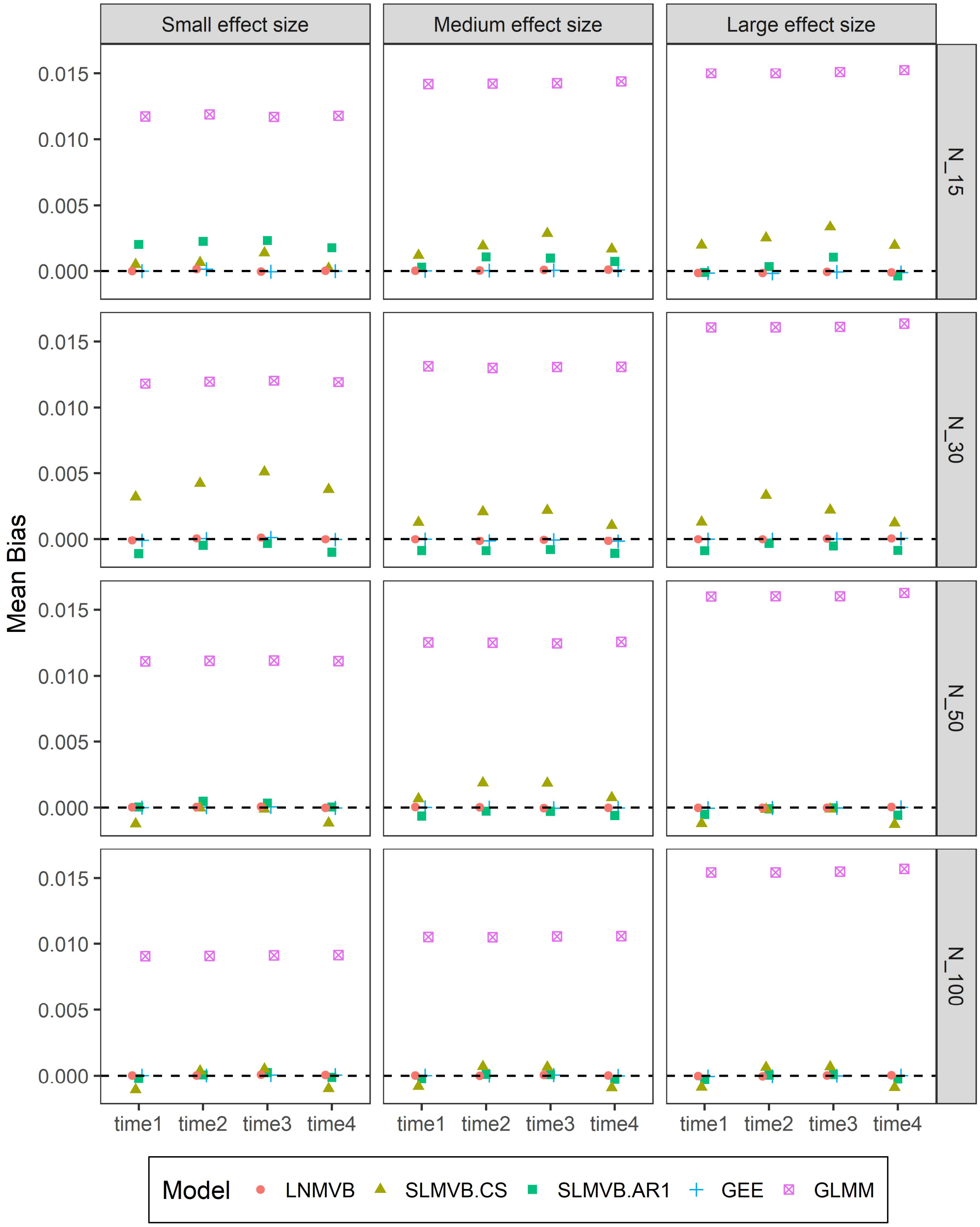


Figure A.19 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$.

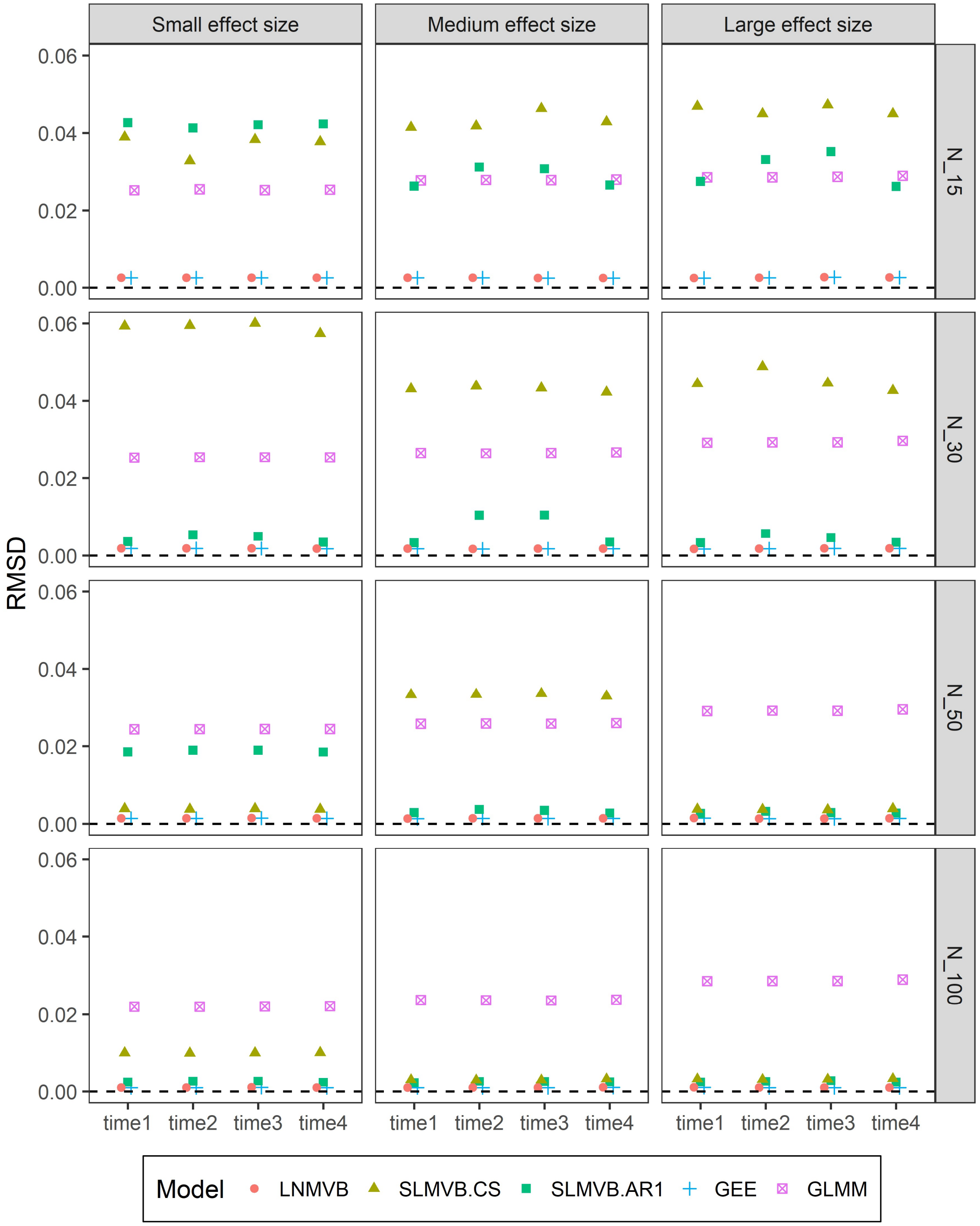


Figure A.20 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$.

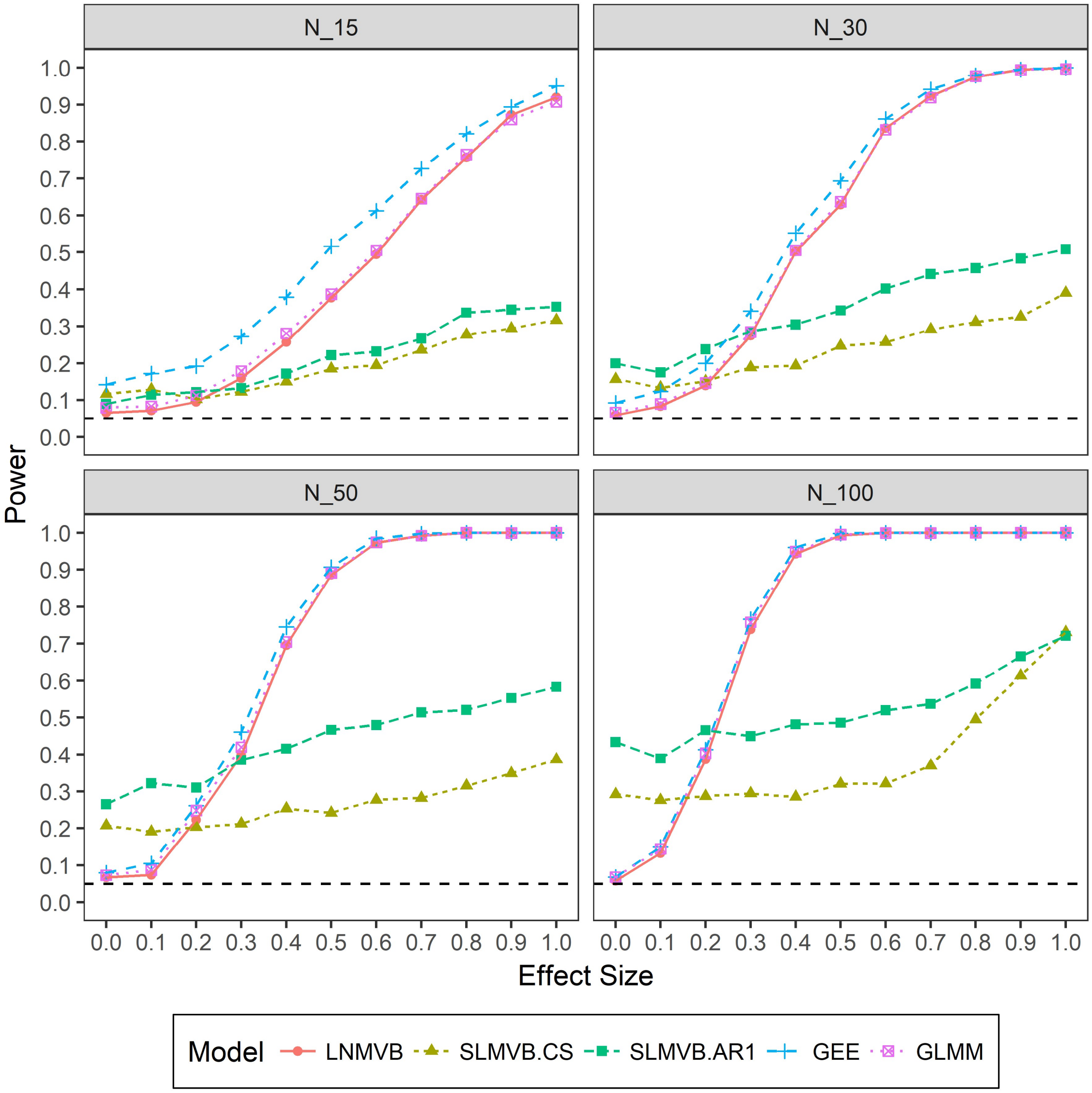


Figure A.21 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

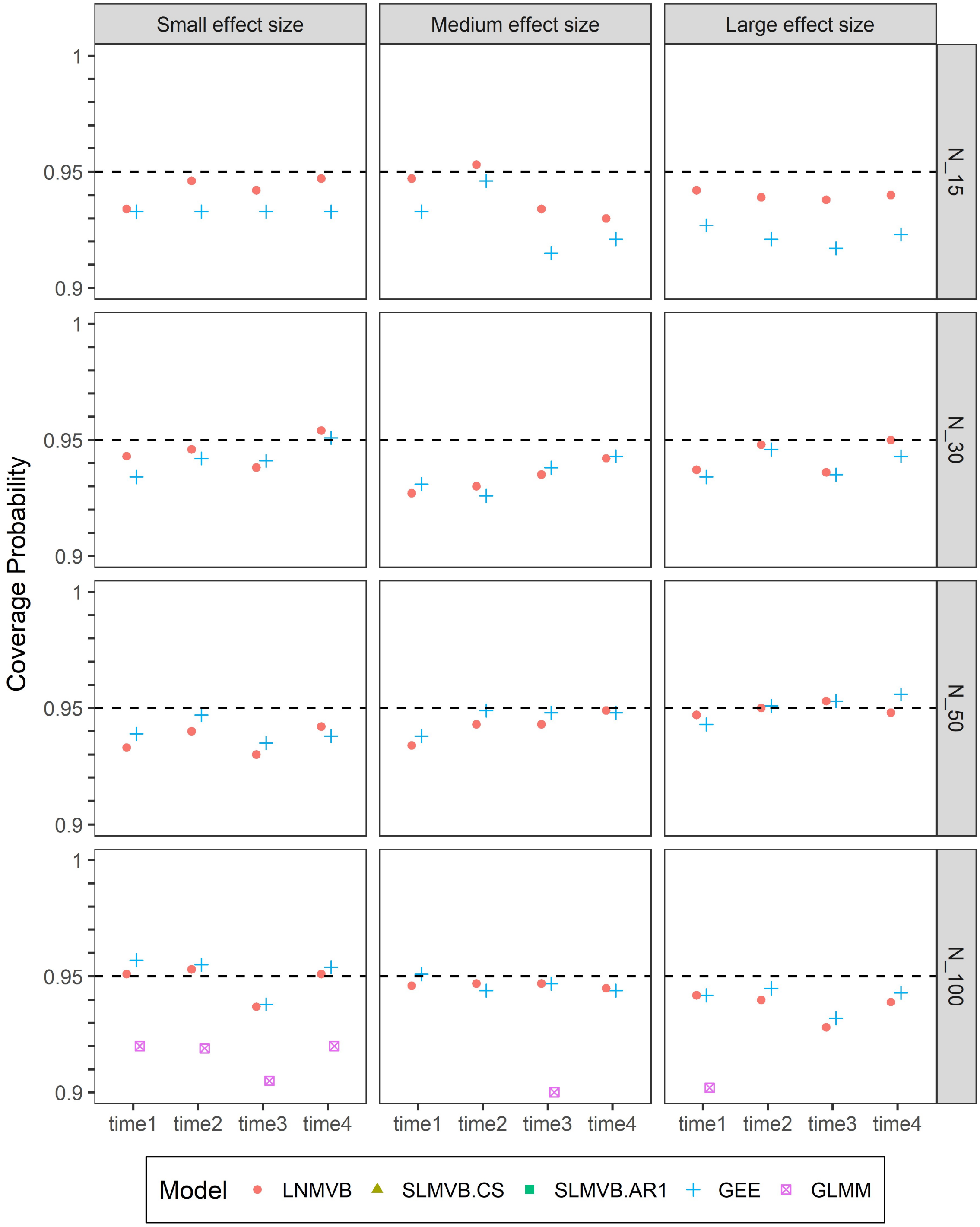


Figure A.22 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

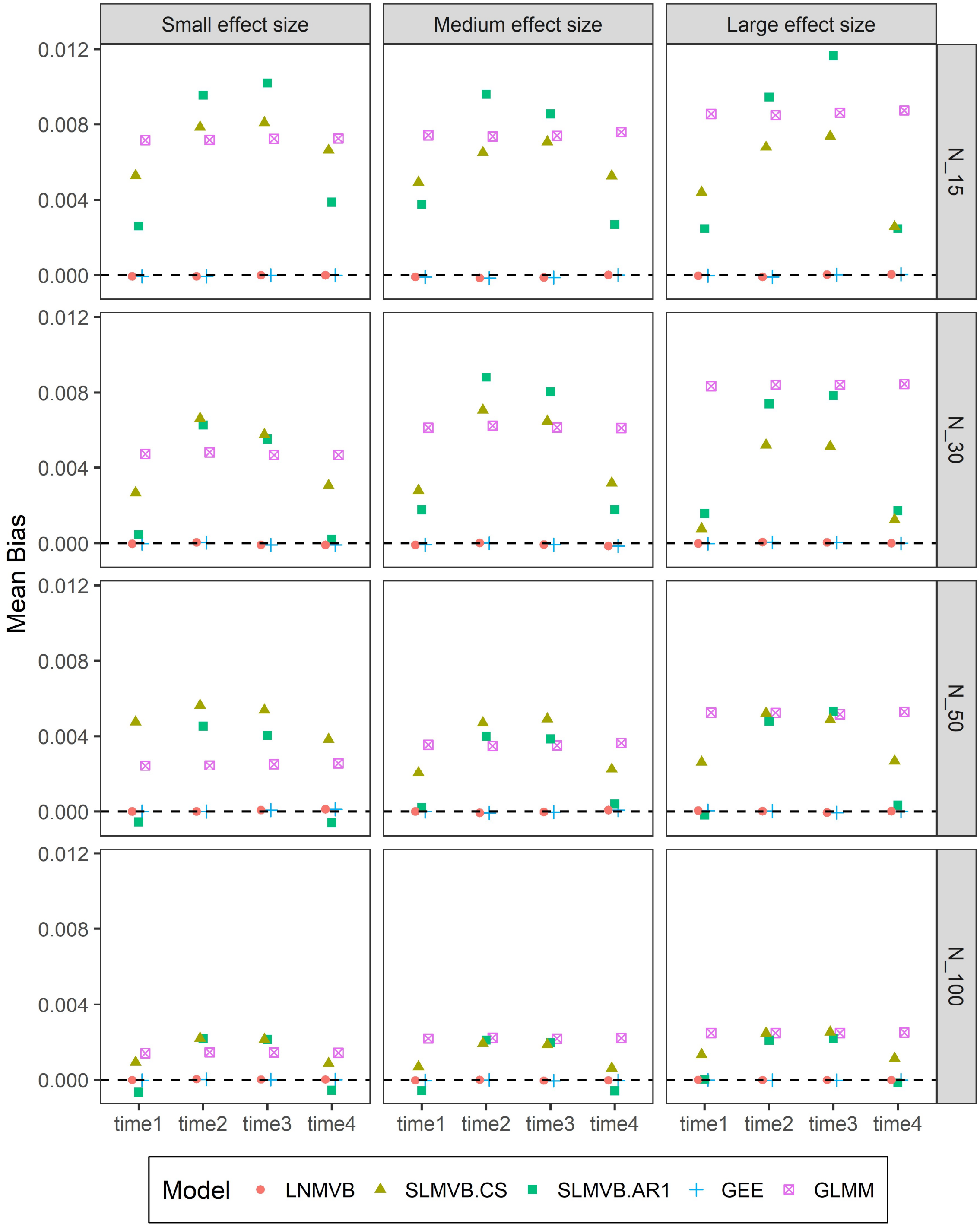


Figure A.23 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$.

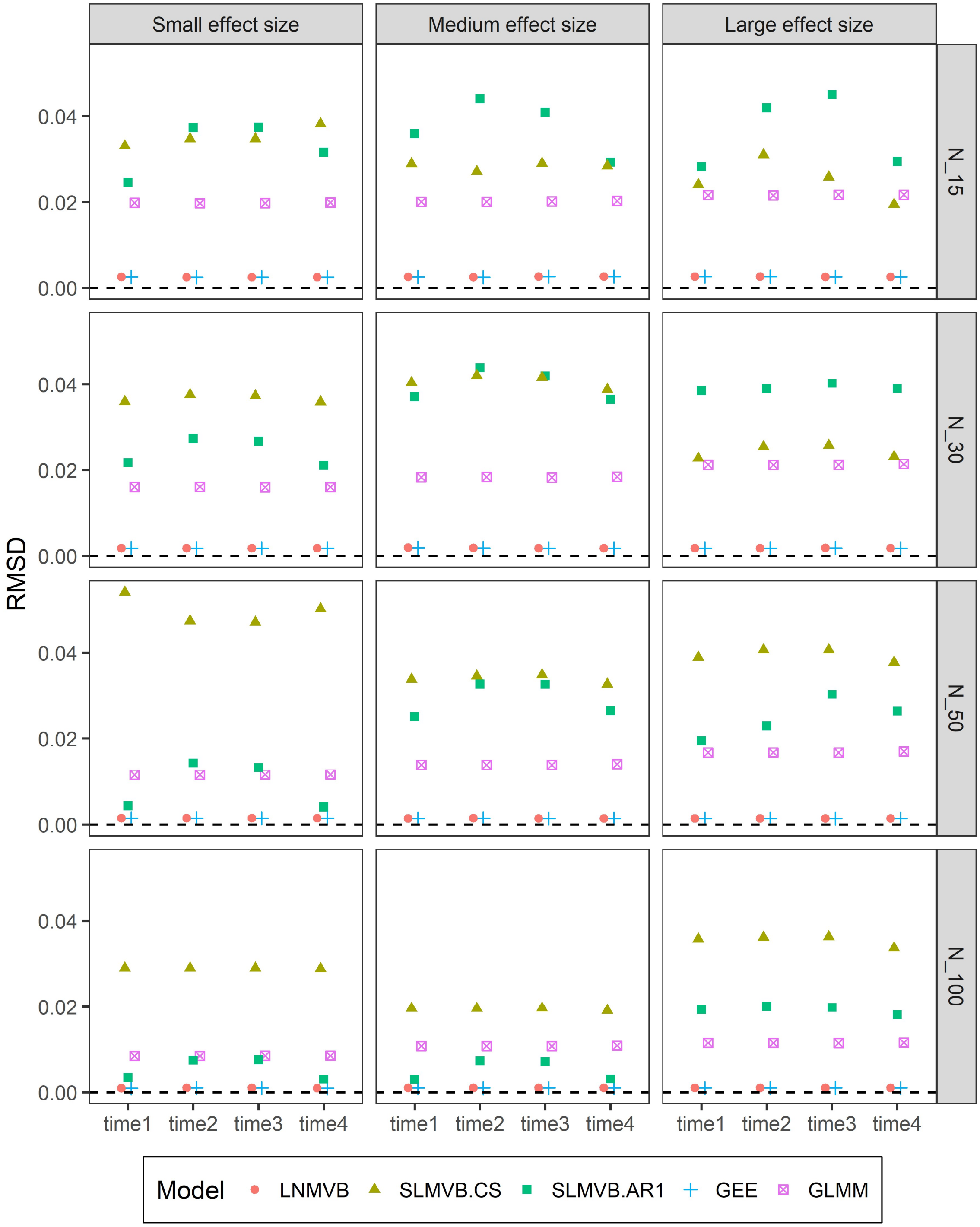


Figure A.24 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$.

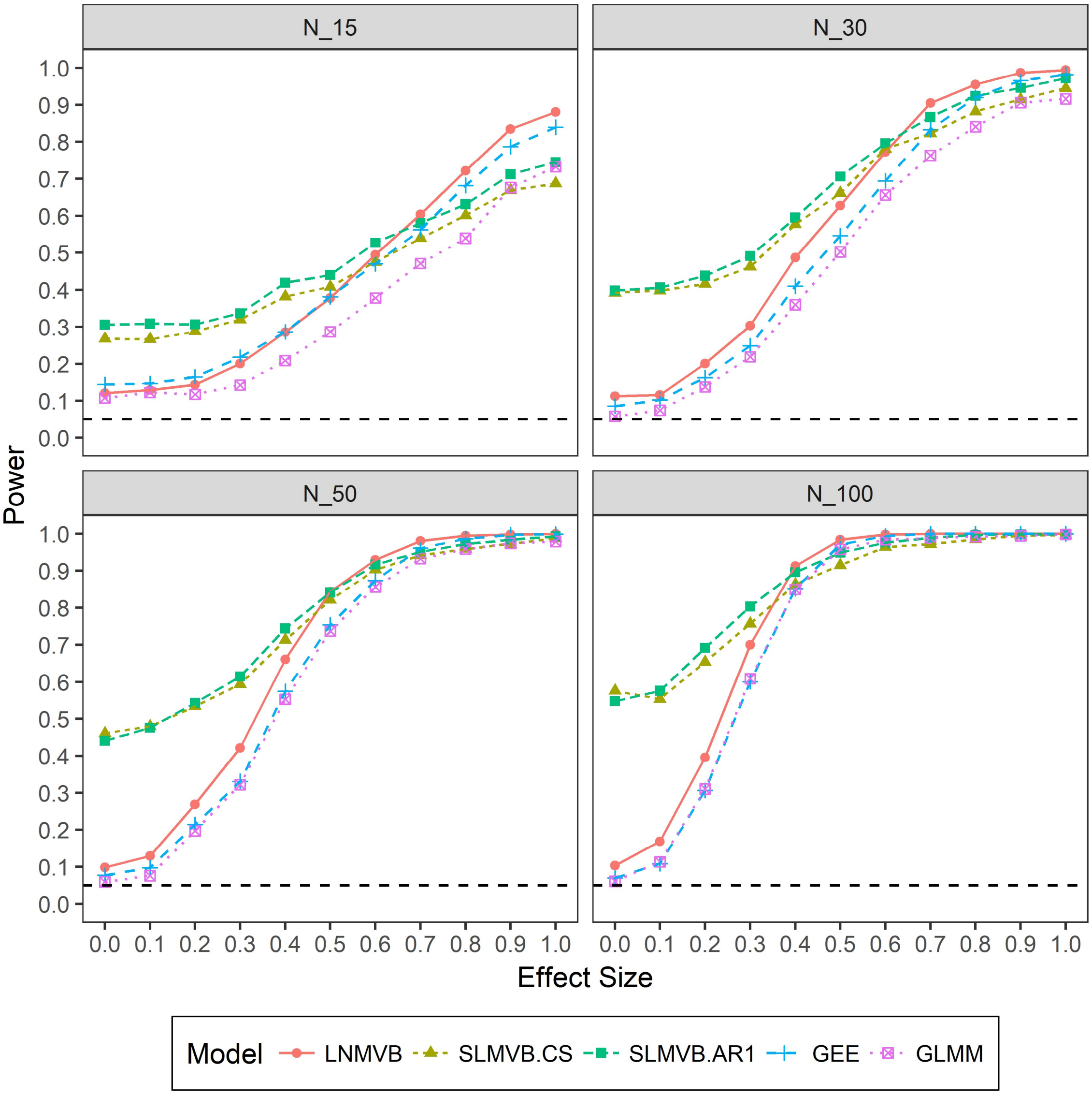


Figure A.25 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. Beta GLMM and SLMVB models did not converge for all replicates; estimates of the power may be biased.

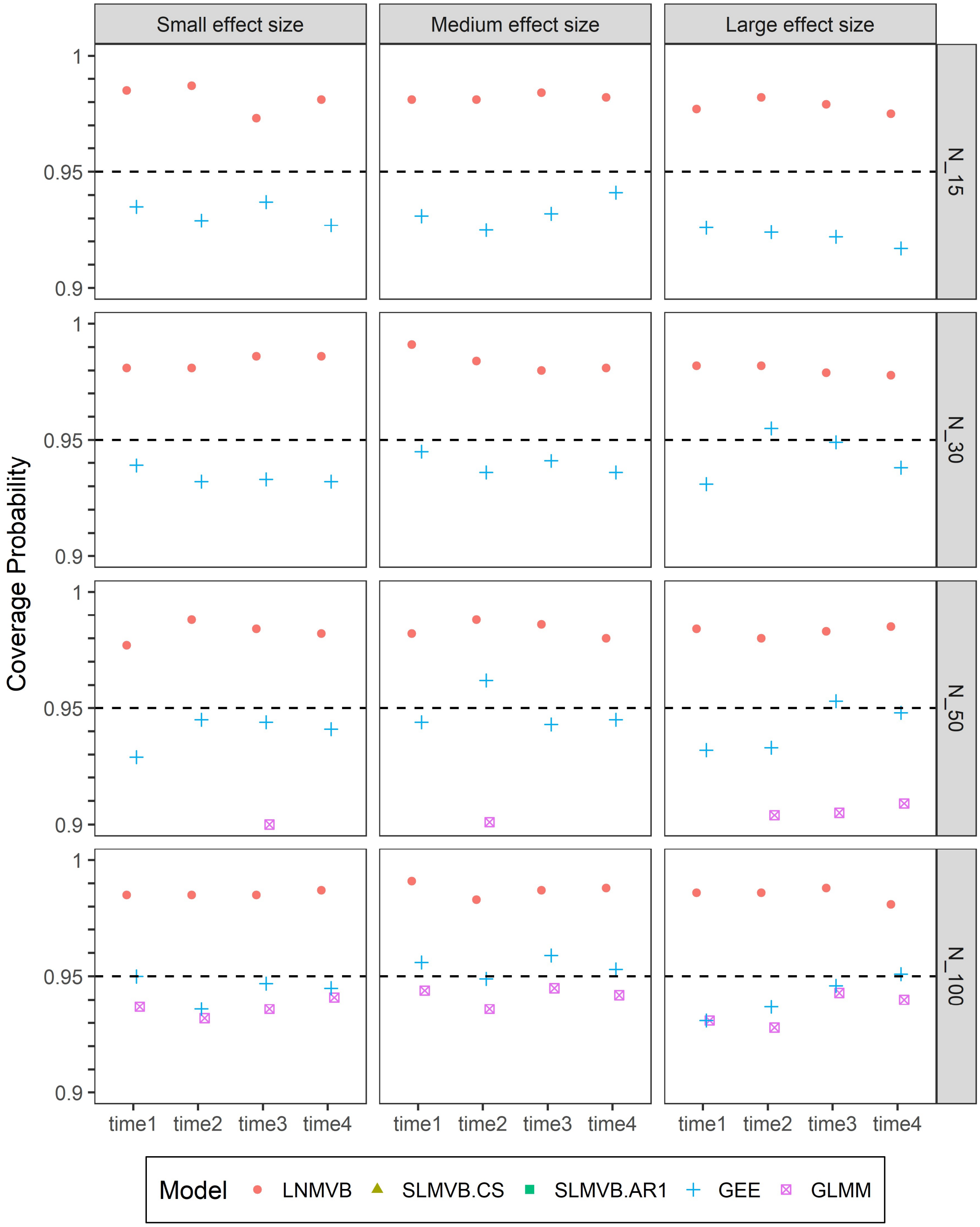


Figure A.26 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

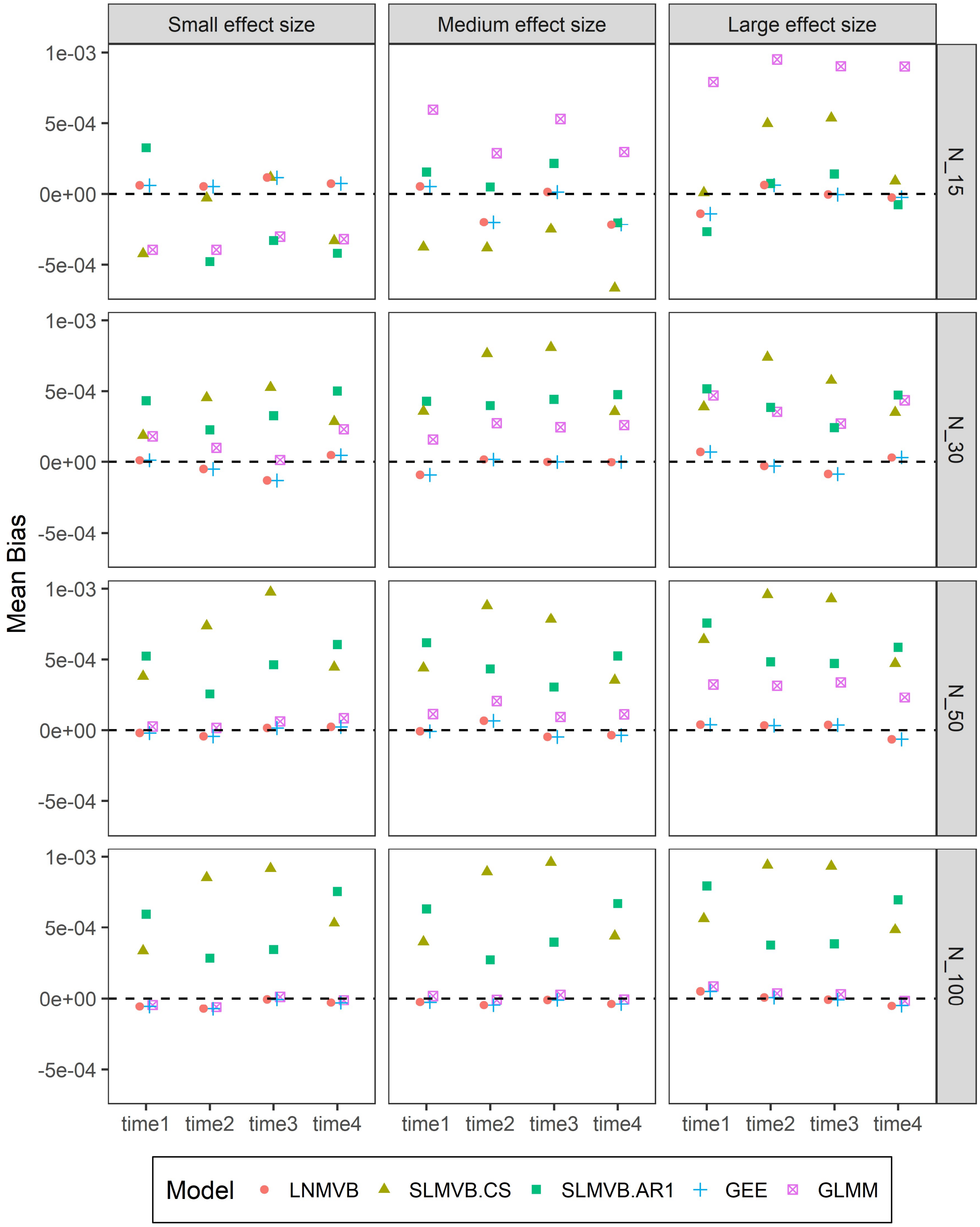


Figure A.27 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$.

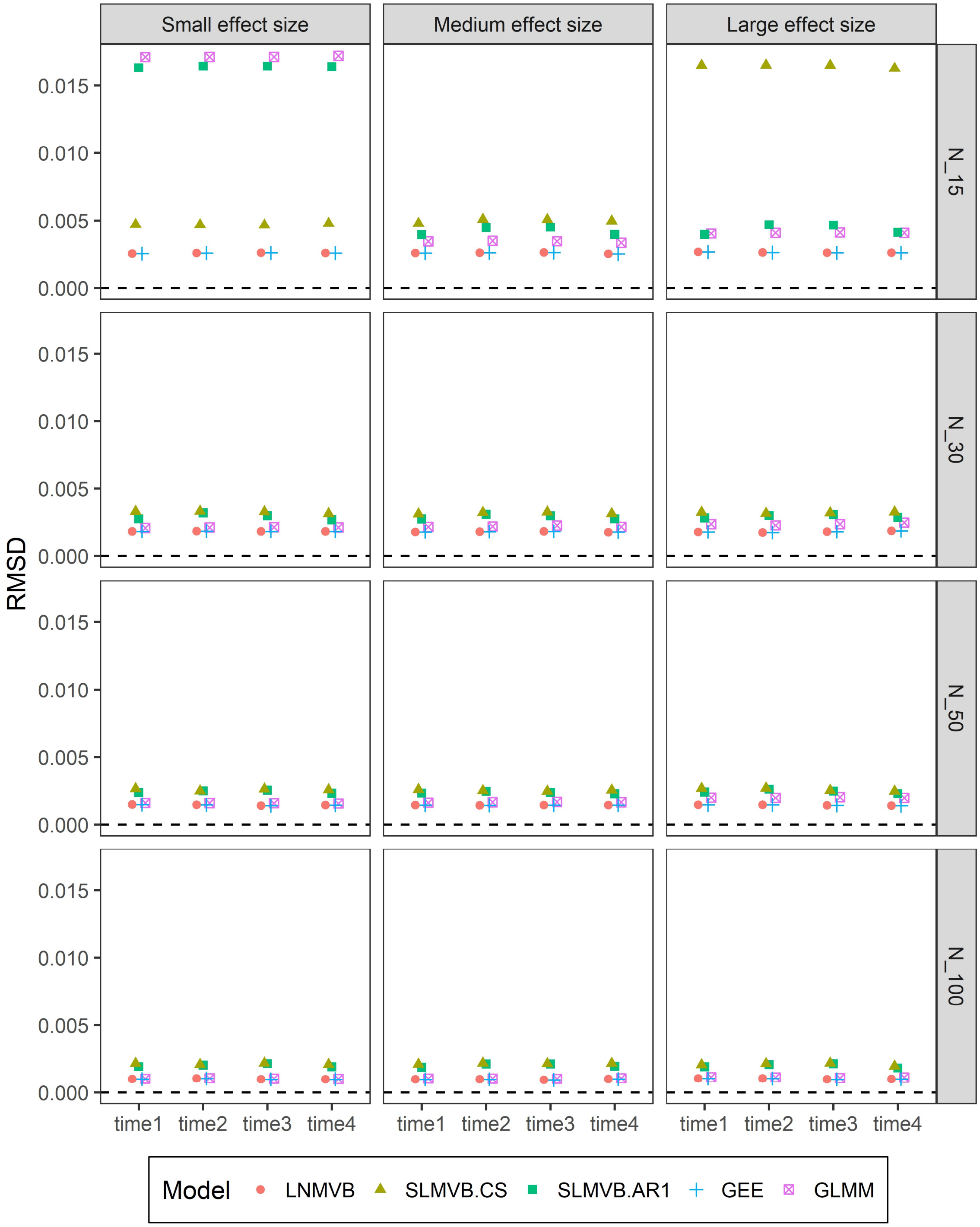


Figure A.28 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$.

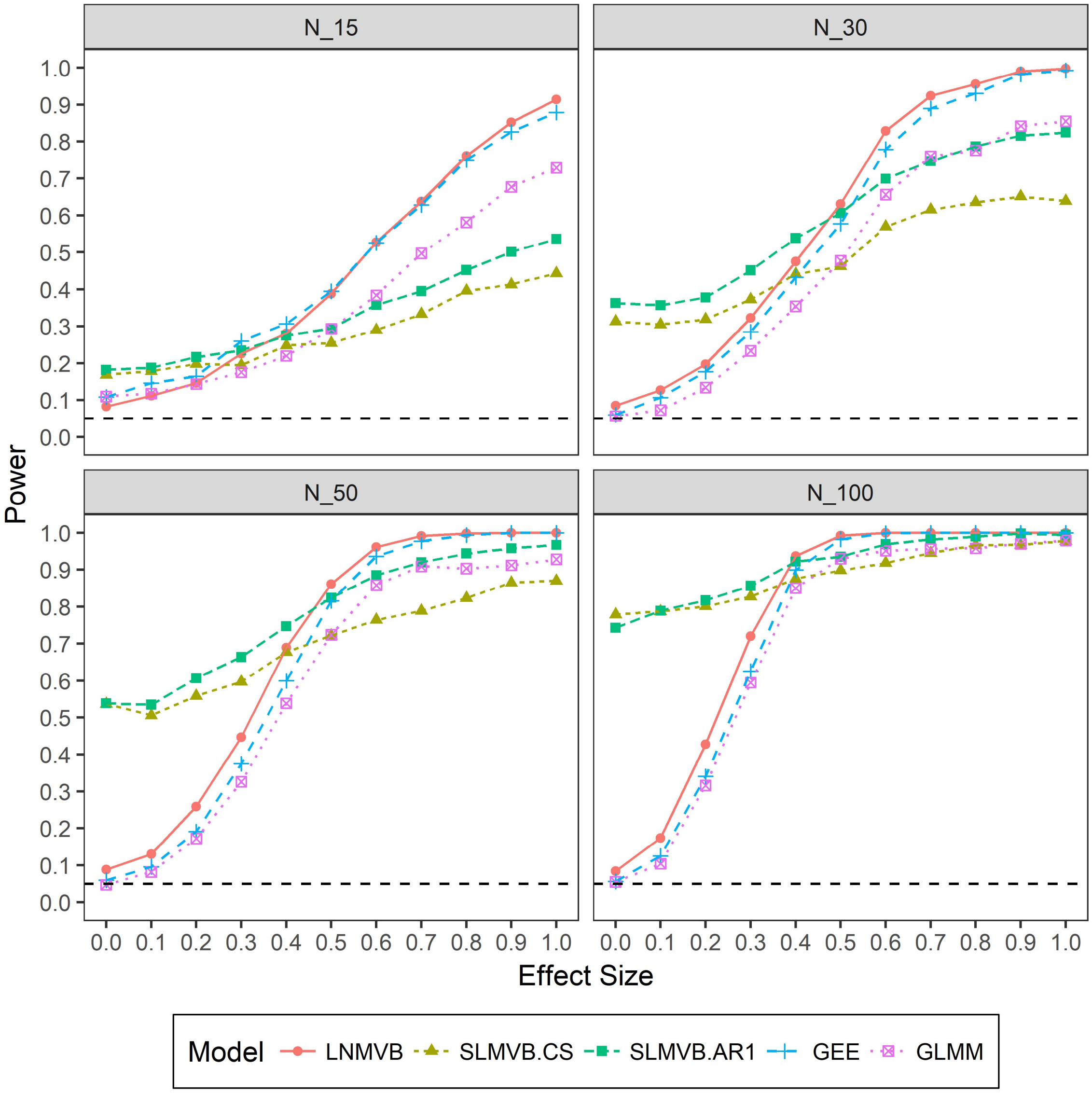


Figure A.29 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. The Beta GLMM and SLMVB models did not converge for all replicates; estimates of the power may be biased.

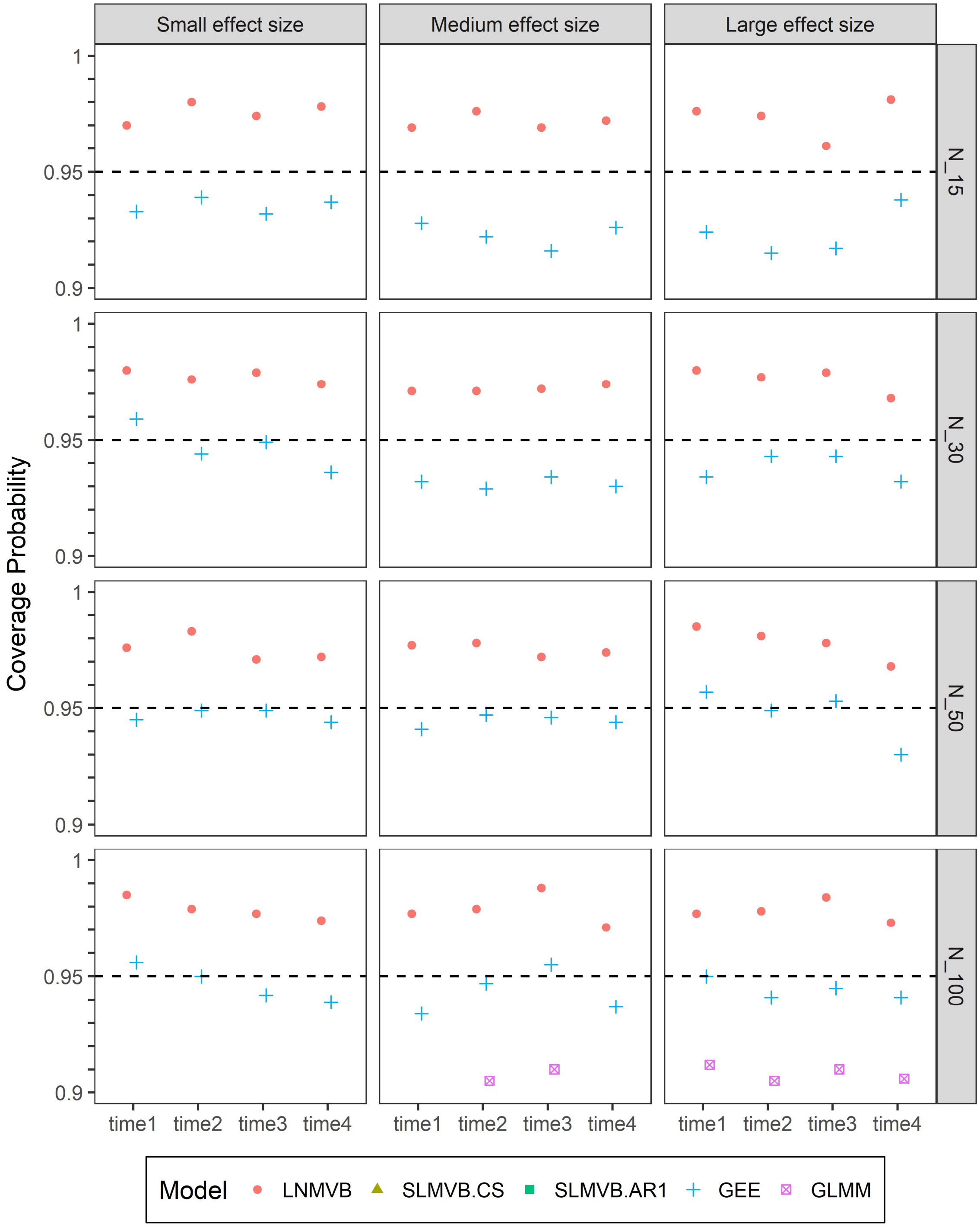


Figure A.30 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

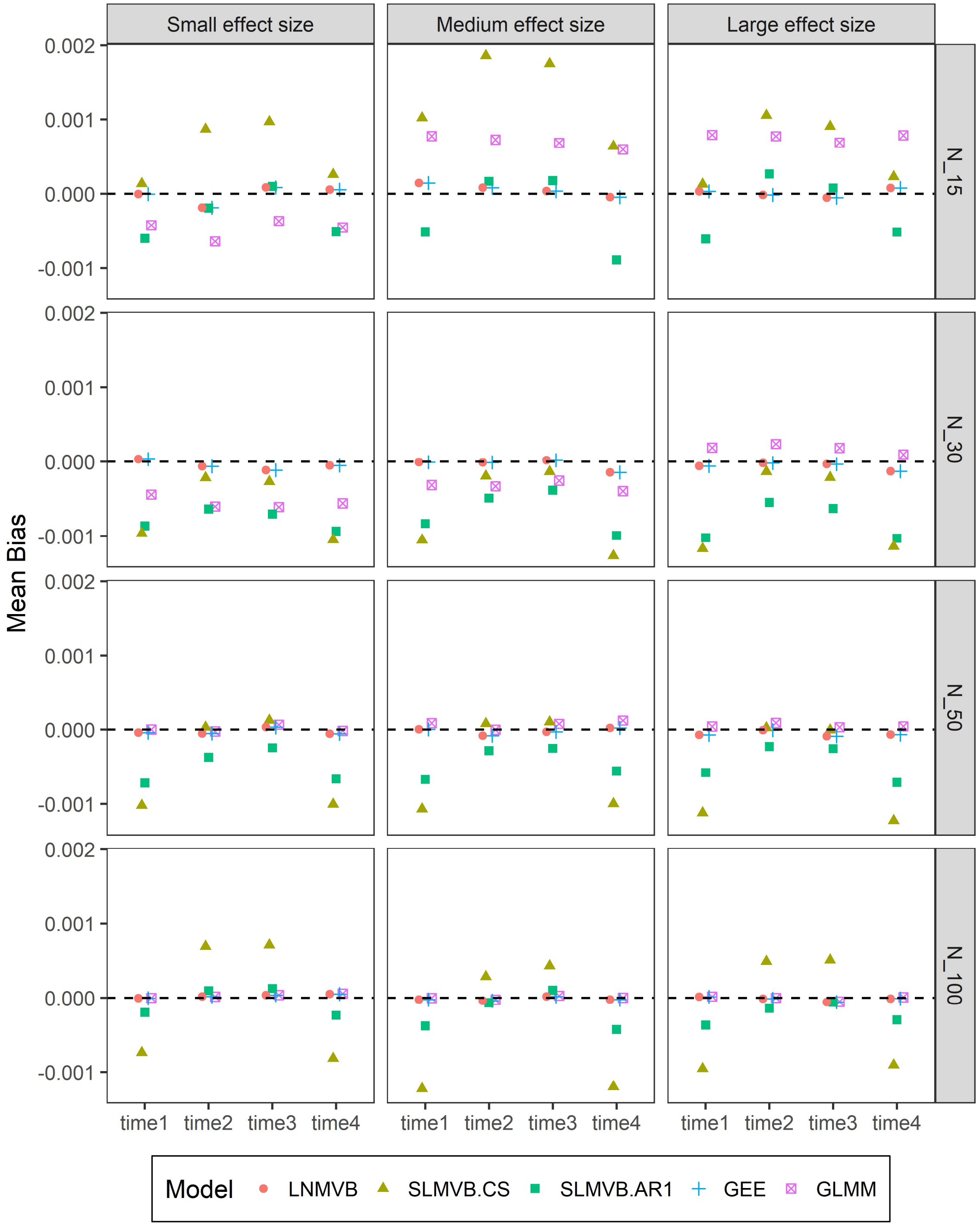


Figure A.31 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3$.

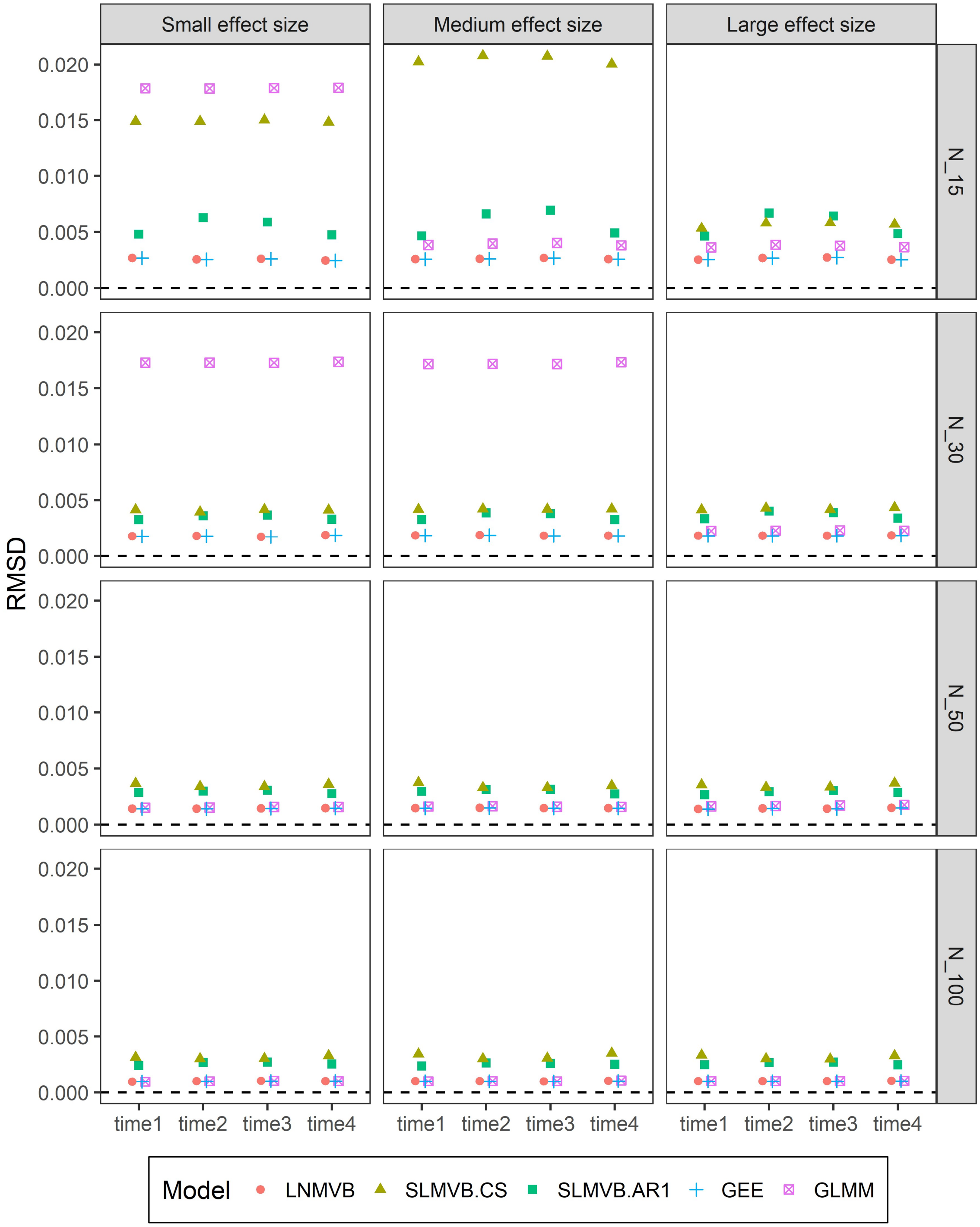


Figure A.32 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3$.

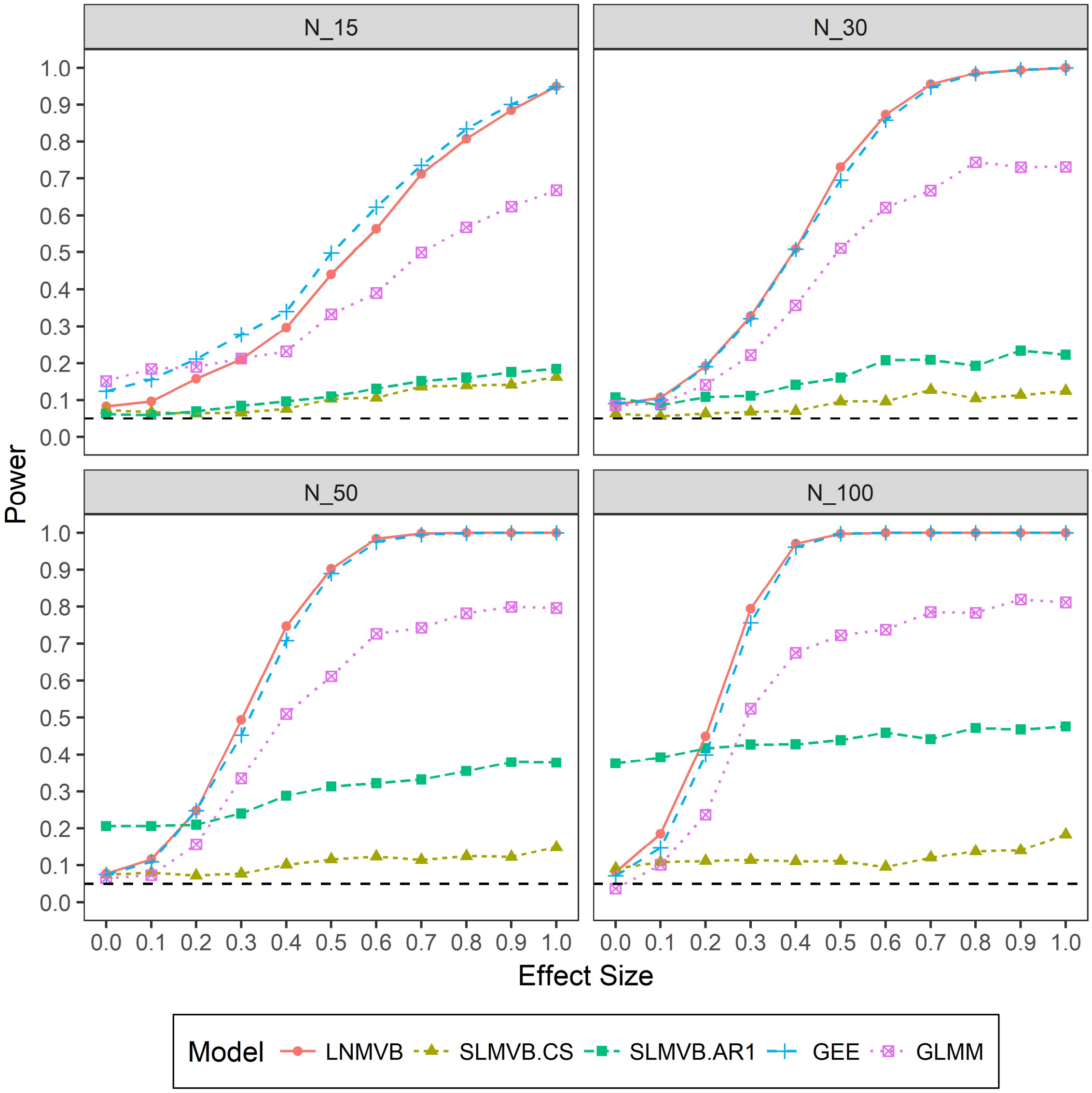


Figure A.33 Empirical power for the time effect of one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. The Beta GLMM and SLMVB models did not converge for all replicates; estimates of the power may be biased.

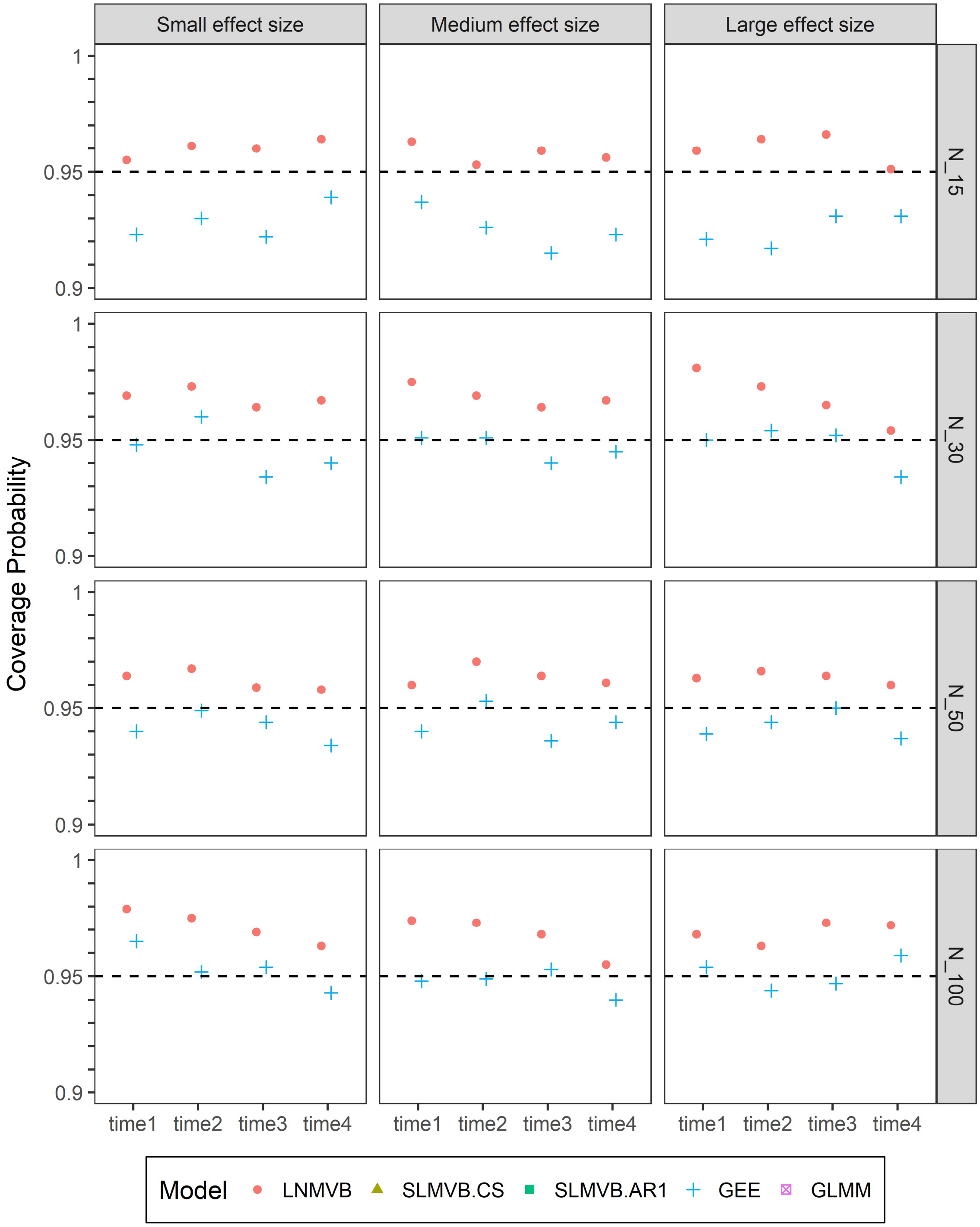


Figure A.34 Coverage probabilities of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5$.

If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

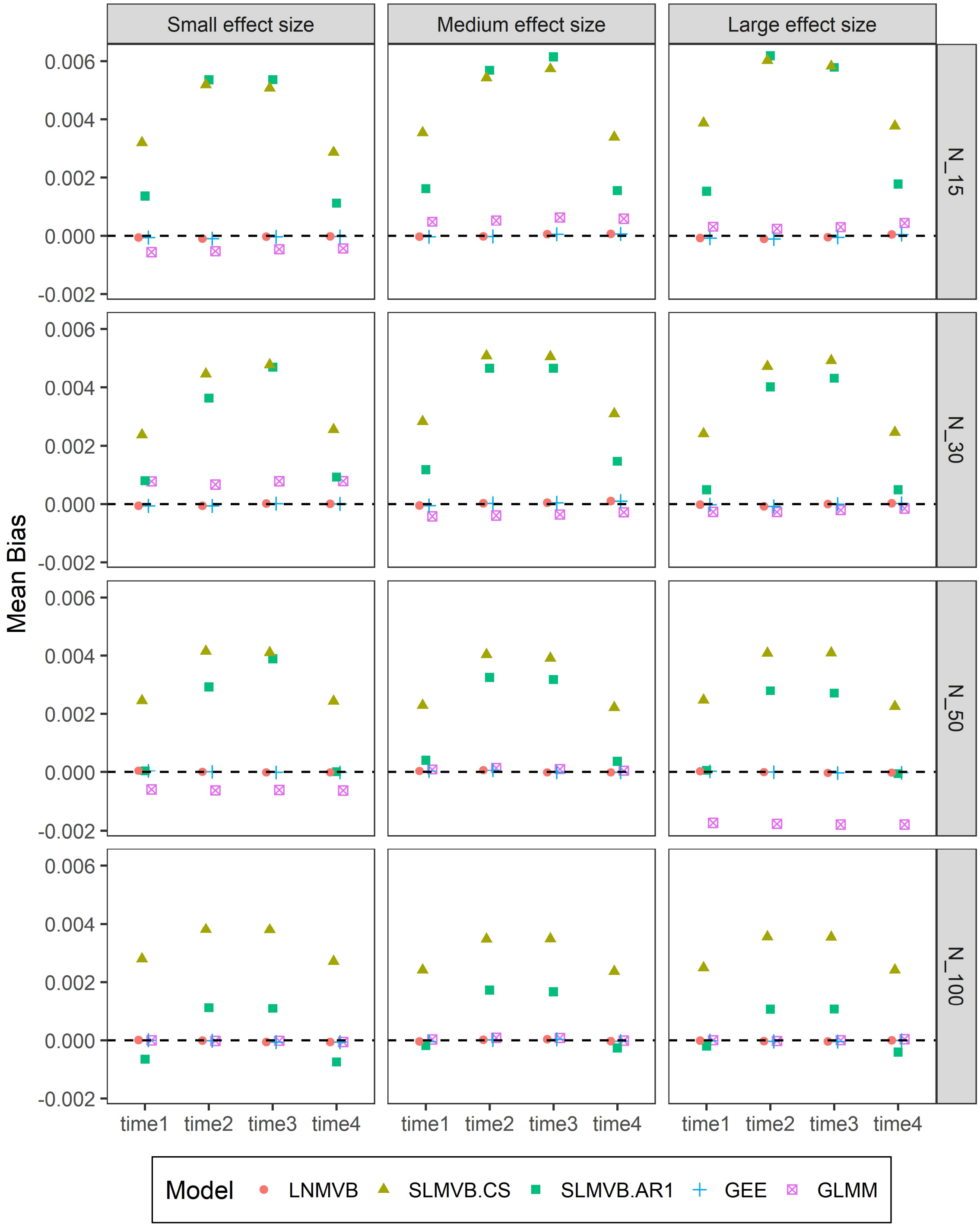


Figure A.35 Mean bias of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5$.

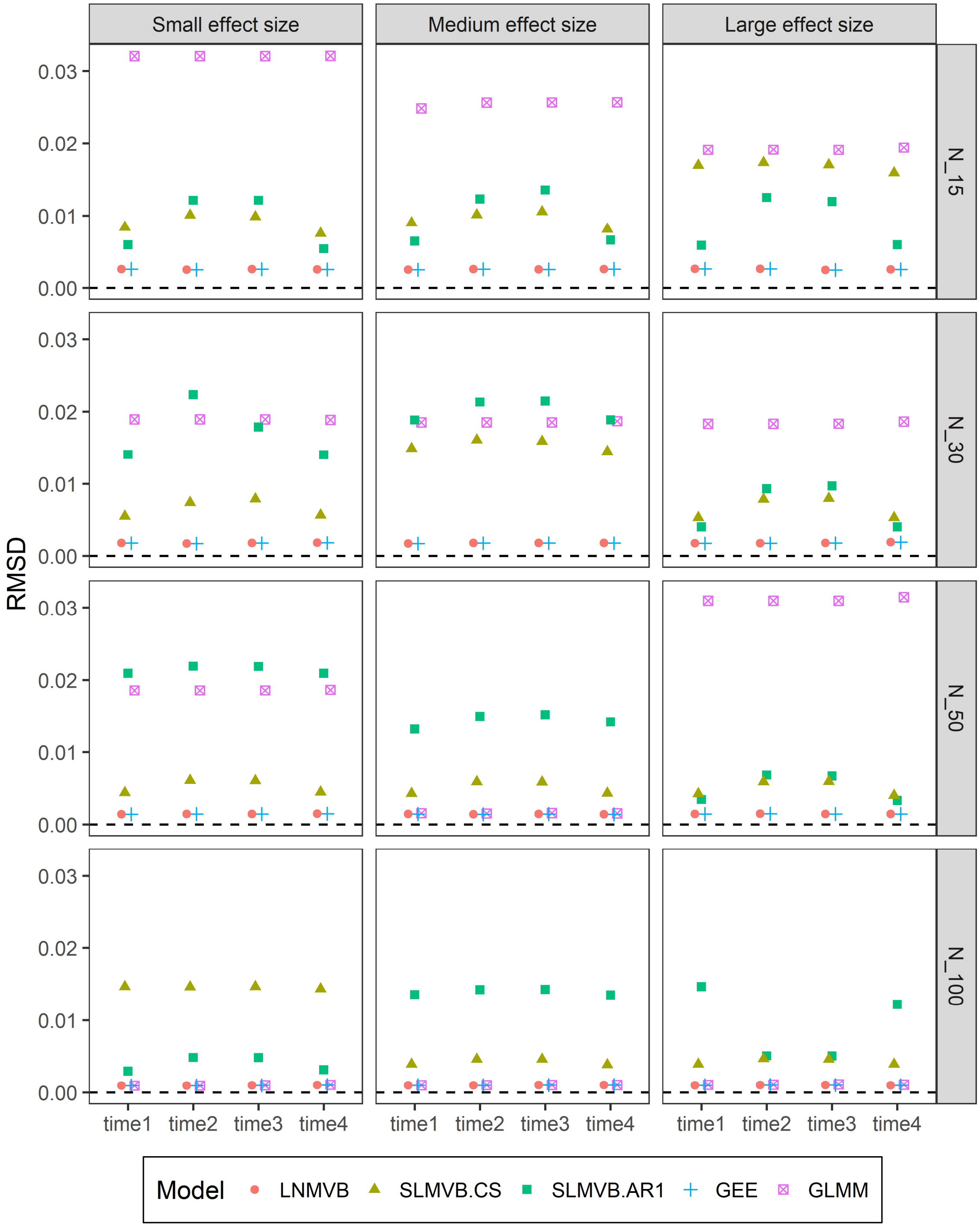


Figure A.36 Root mean squared deviation of the location estimate for one group, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5$.

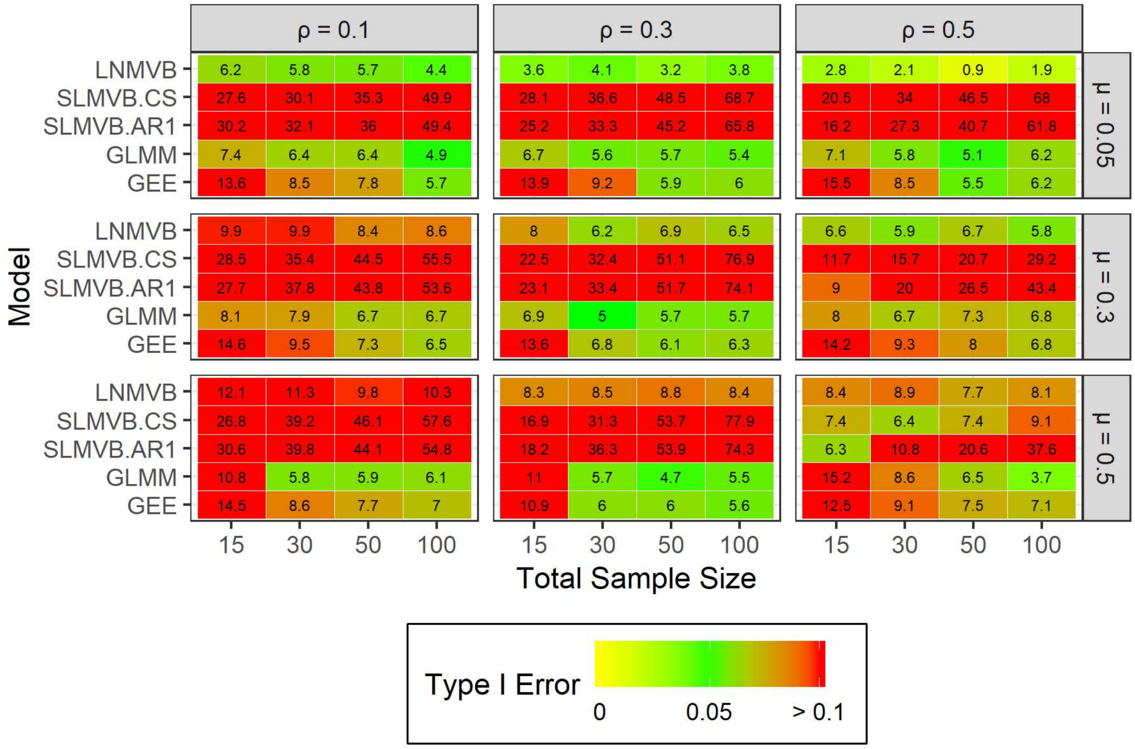


Figure A.37 Summary of Type I error for one group simulations using 1000 replicates with AR(1) correlation structure.

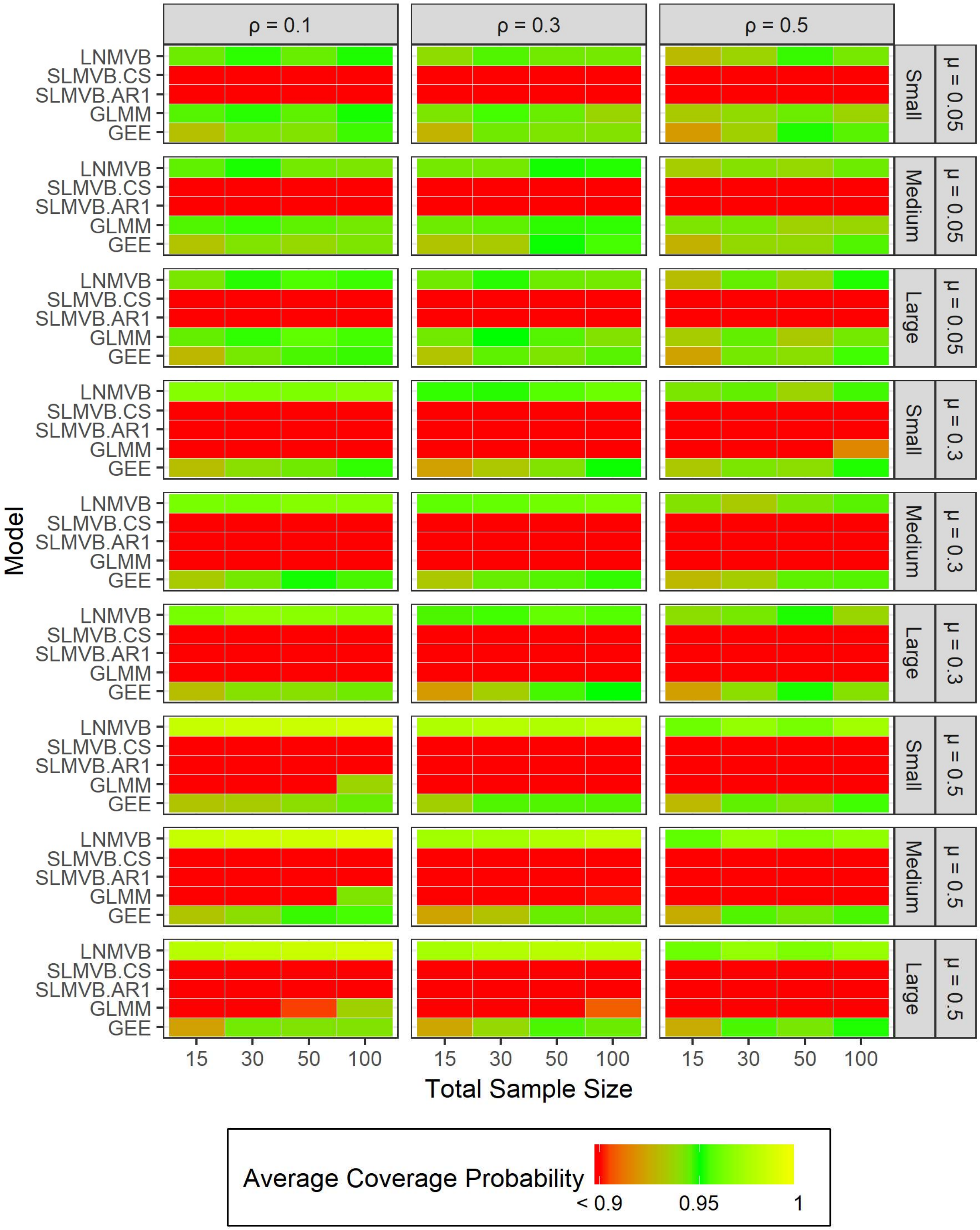


Figure A.38 Summary of coverage probabilities for one group simulations using 1000 replicates with AR(1) correlation structure.

Average coverage probabilities are the mean of the coverage probabilities of the 4 repeated-measures. Small, medium, and large represent the effect sizes.

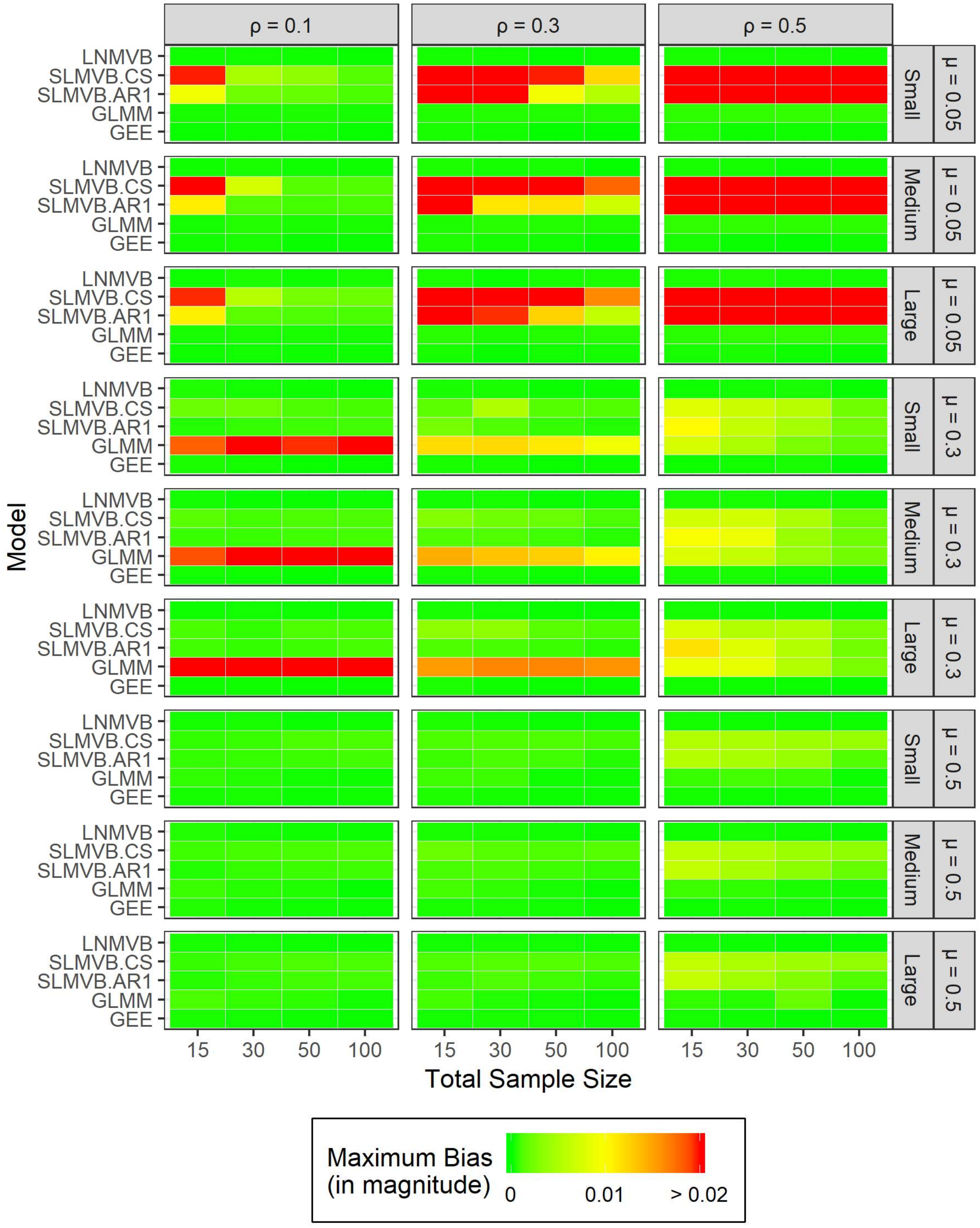


Figure A.39 Summary of bias of location parameter for one group simulations using 1000 replicates with AR(1) correlation structure.

Maximum bias is the bias that is furthest from zero of the 4 repeated-measures. Small, medium, and large represent the effect sizes.

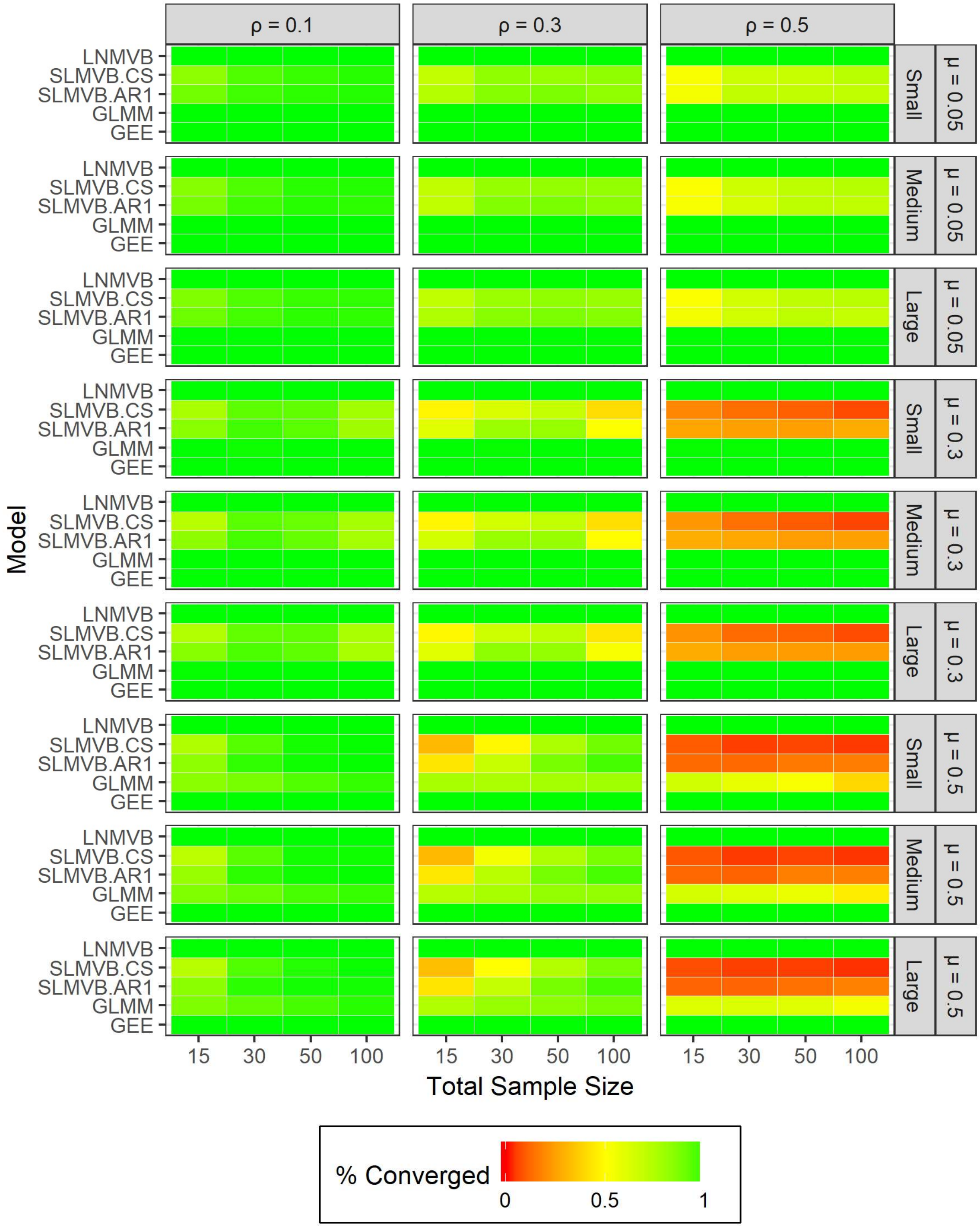


Figure A.40 Summary of model convergence for one group simulations using 1000 replicates with AR(1) correlation structure.

Appendix B: Two Group Simulation Results

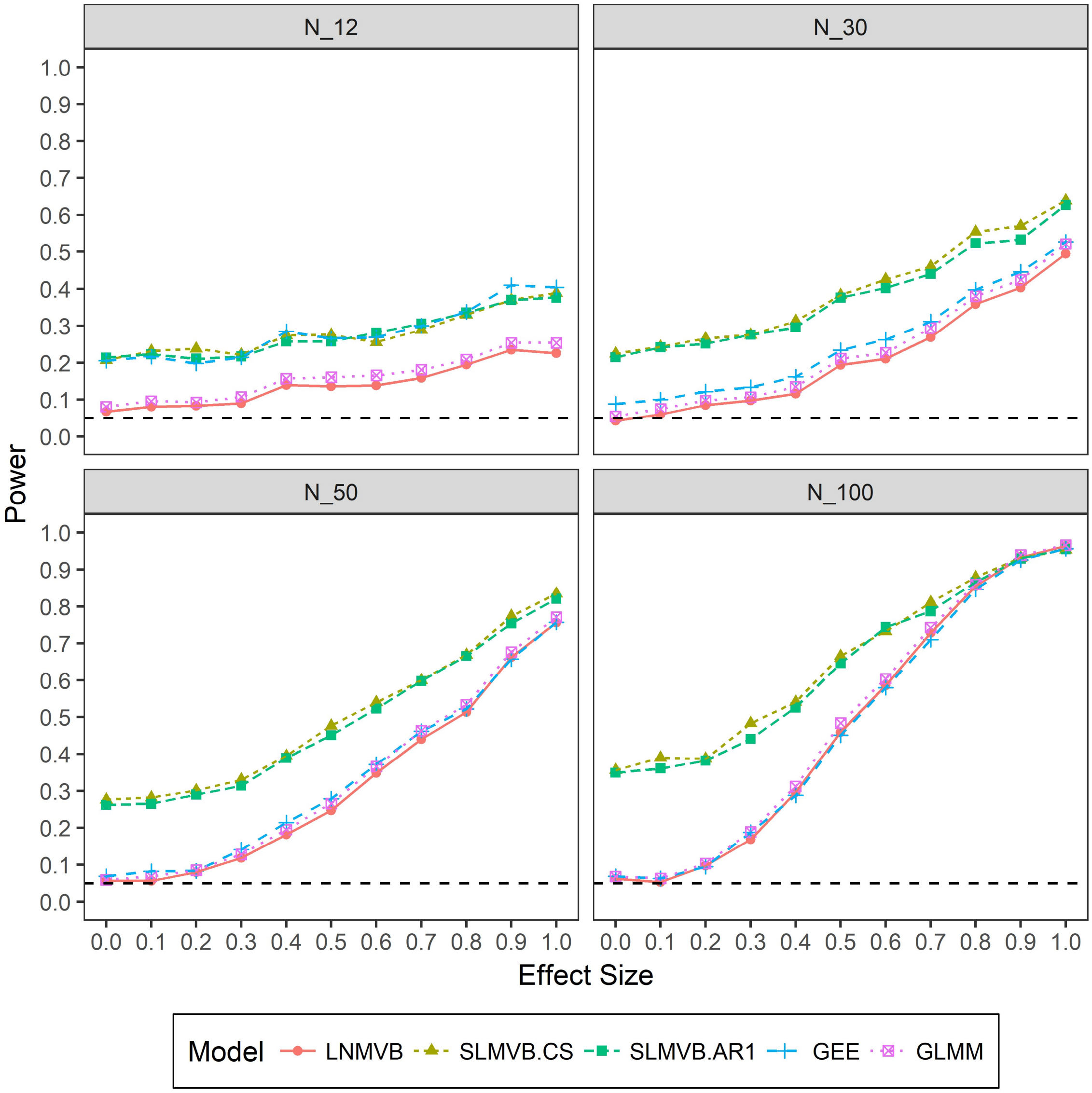


Figure B.1 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

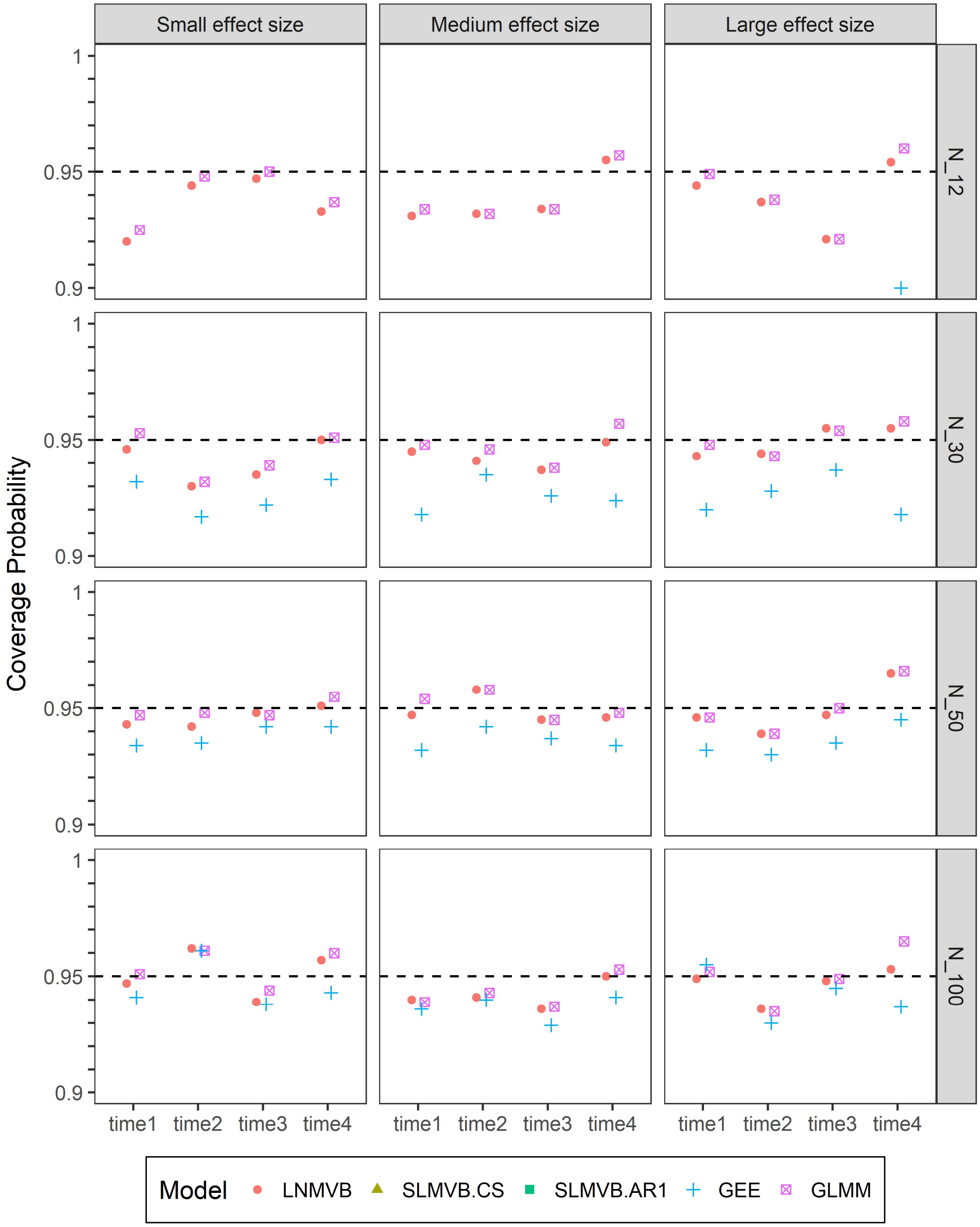


Figure B.2 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

Group with non-stationary location parameter displayed. If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

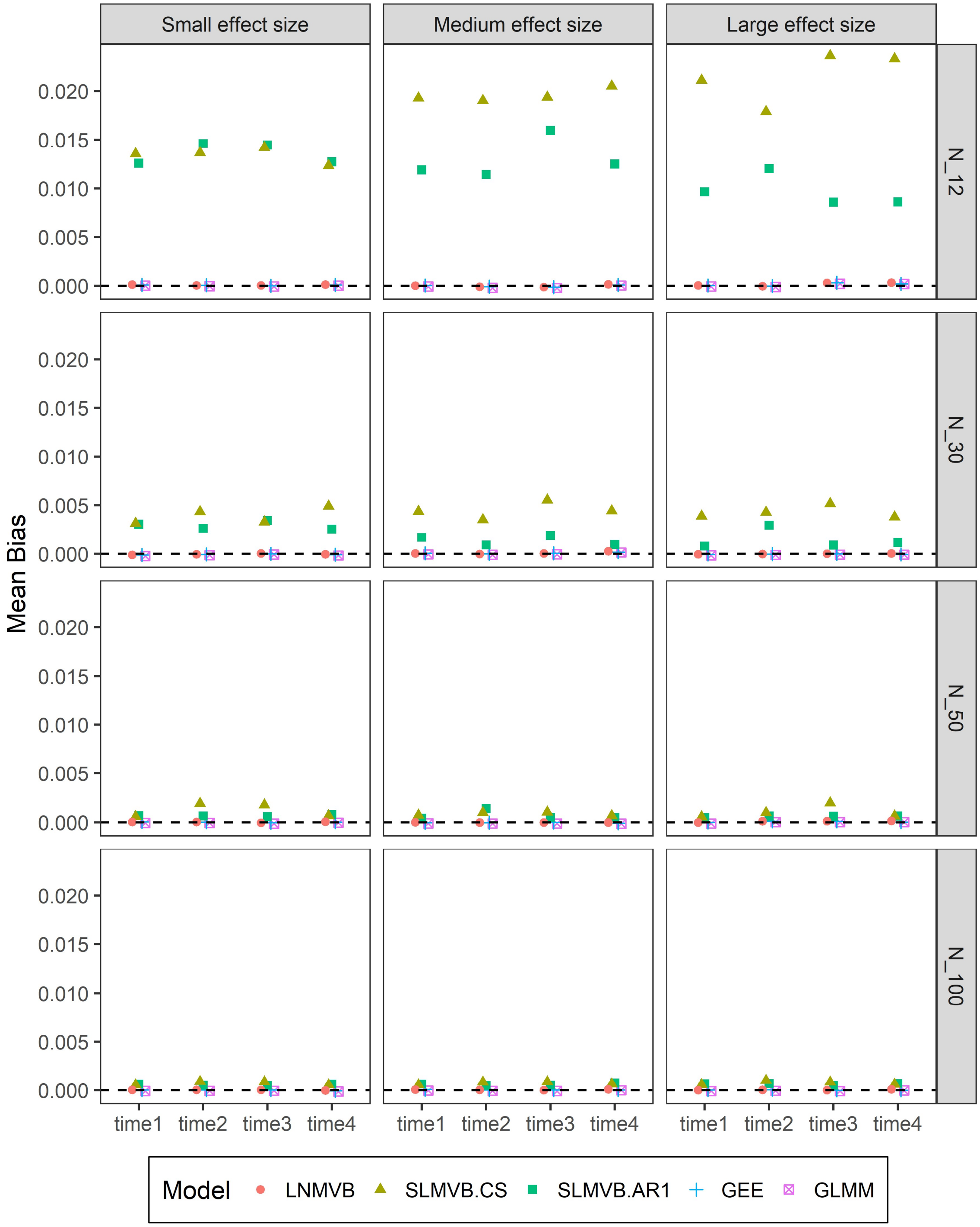


Figure B.3 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$. Group with non-stationary location parameter displayed.

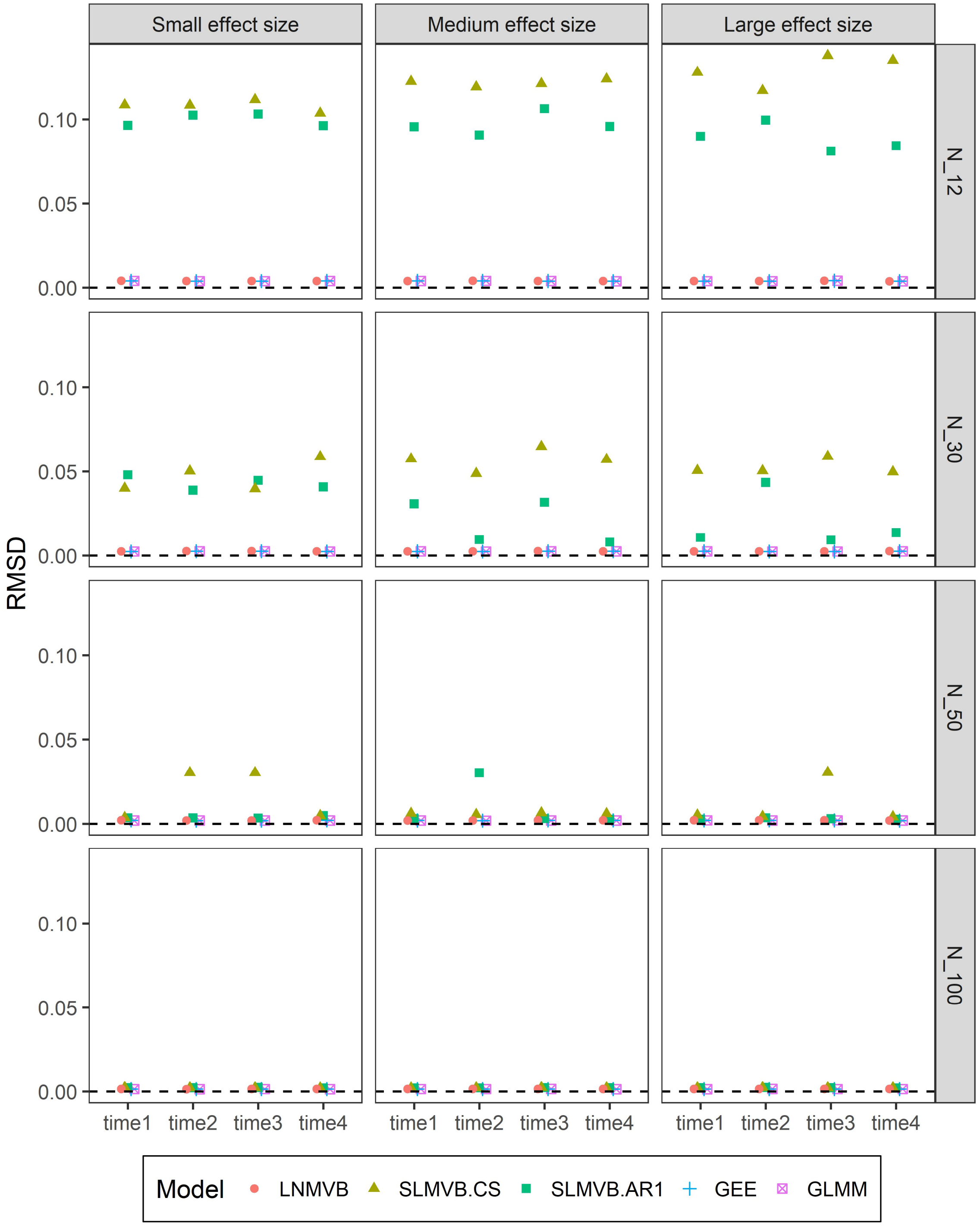


Figure B.4 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.1$.

Group with non-stationary location parameter displayed.

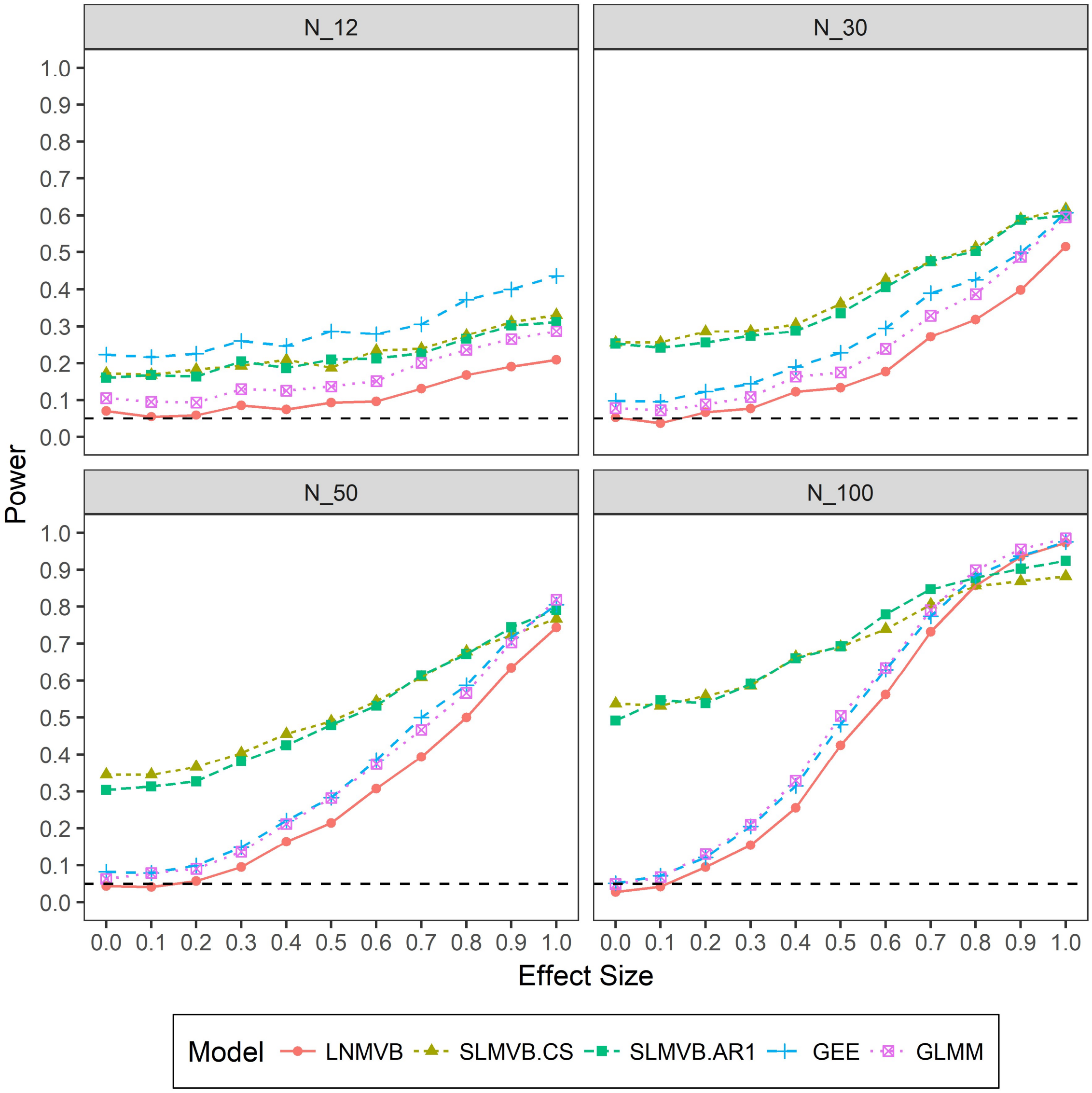


Figure B.5 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

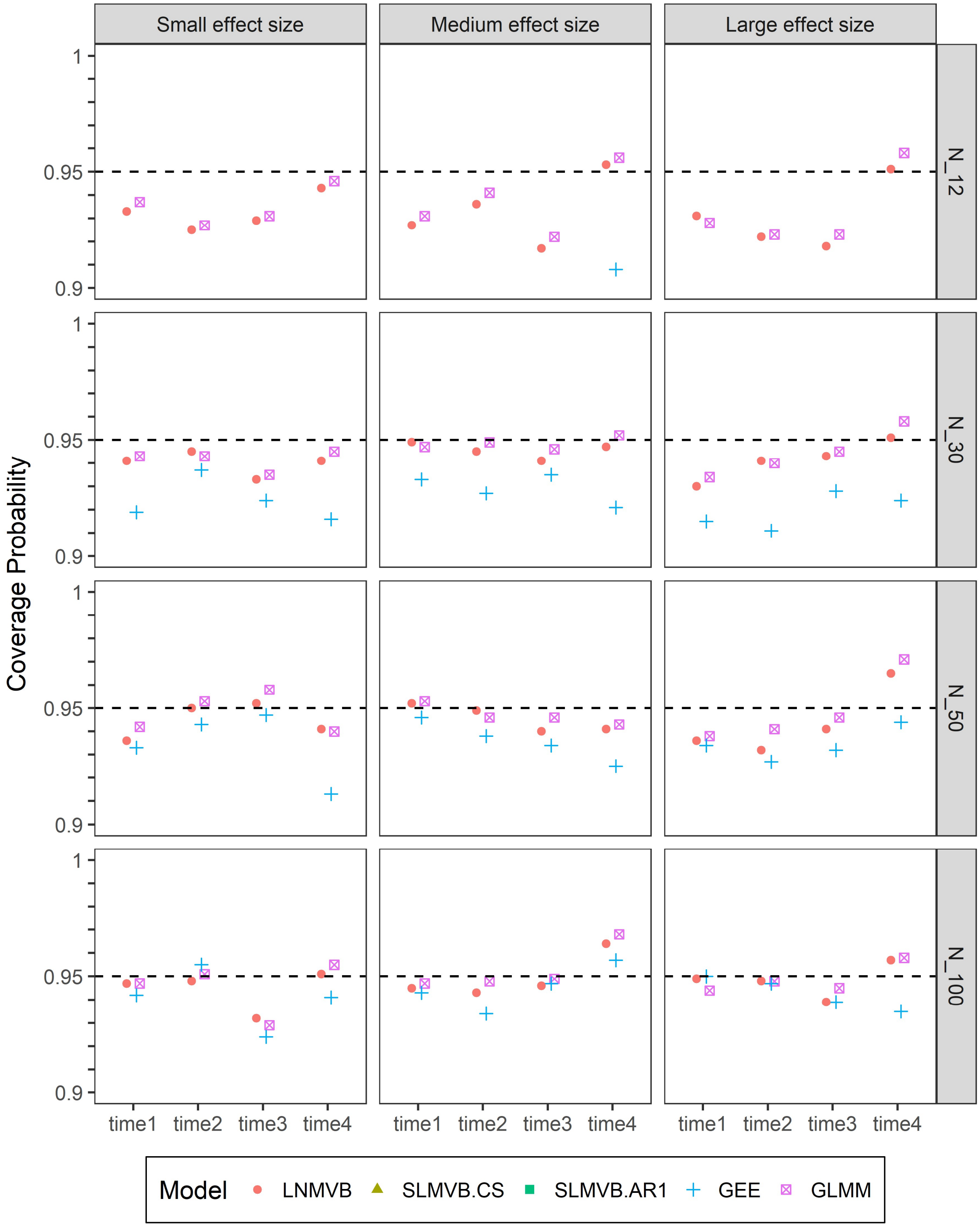


Figure B.6 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3$.

Group with non-stationary location parameter displayed. If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

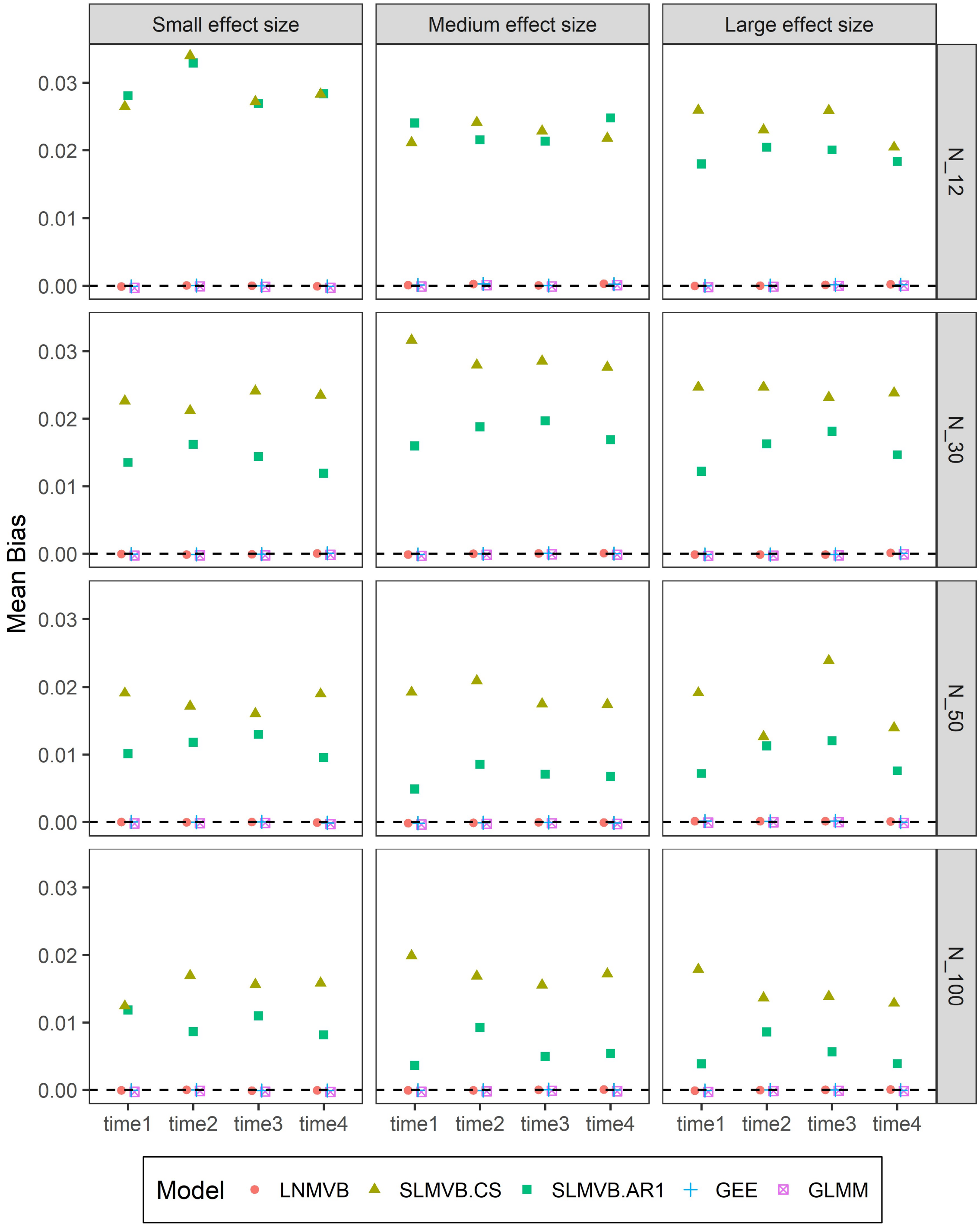


Figure B.7 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3$. Group with non-stationary location parameter displayed.

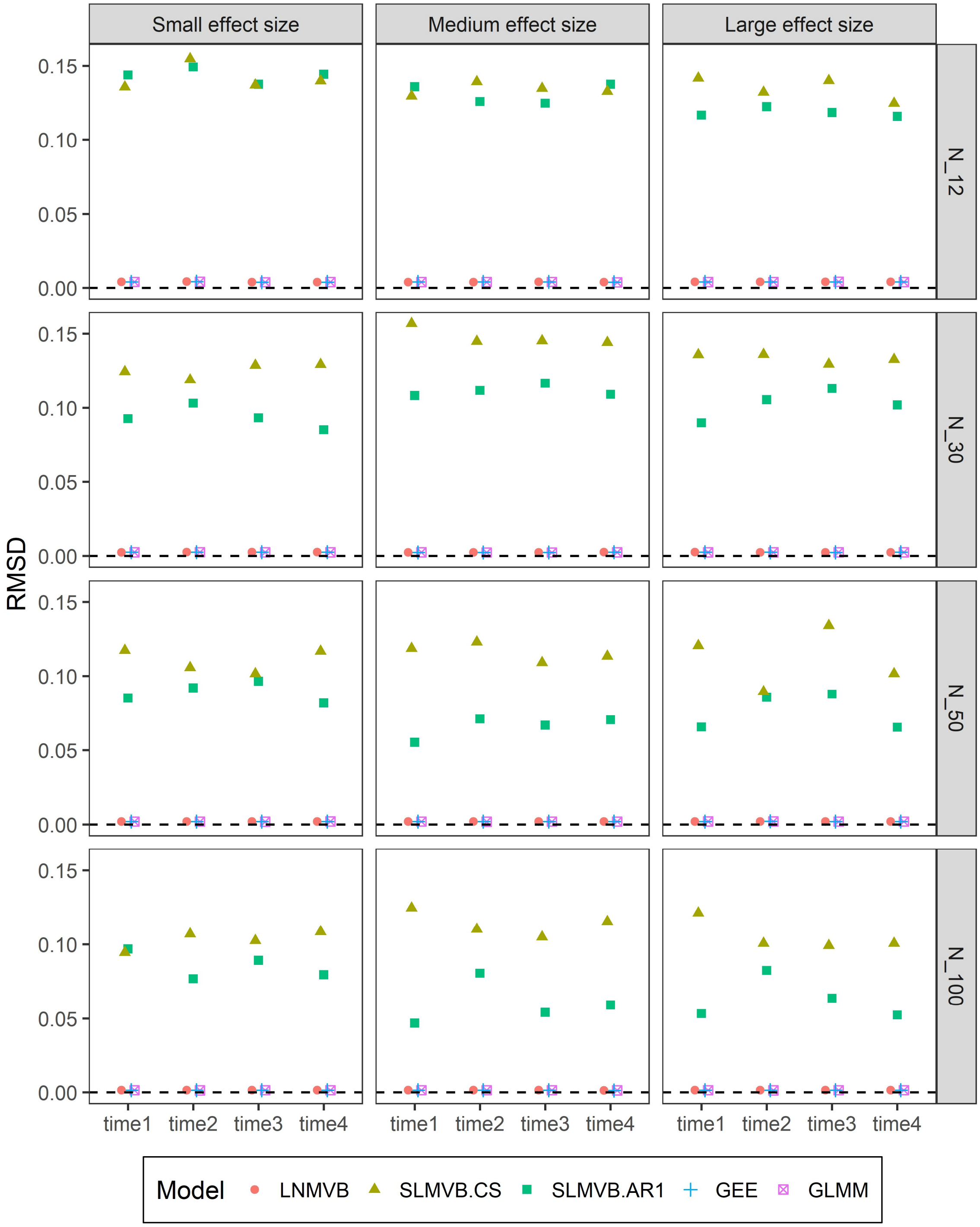


Figure B.8 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.3$.

Group with non-stationary location parameter displayed.

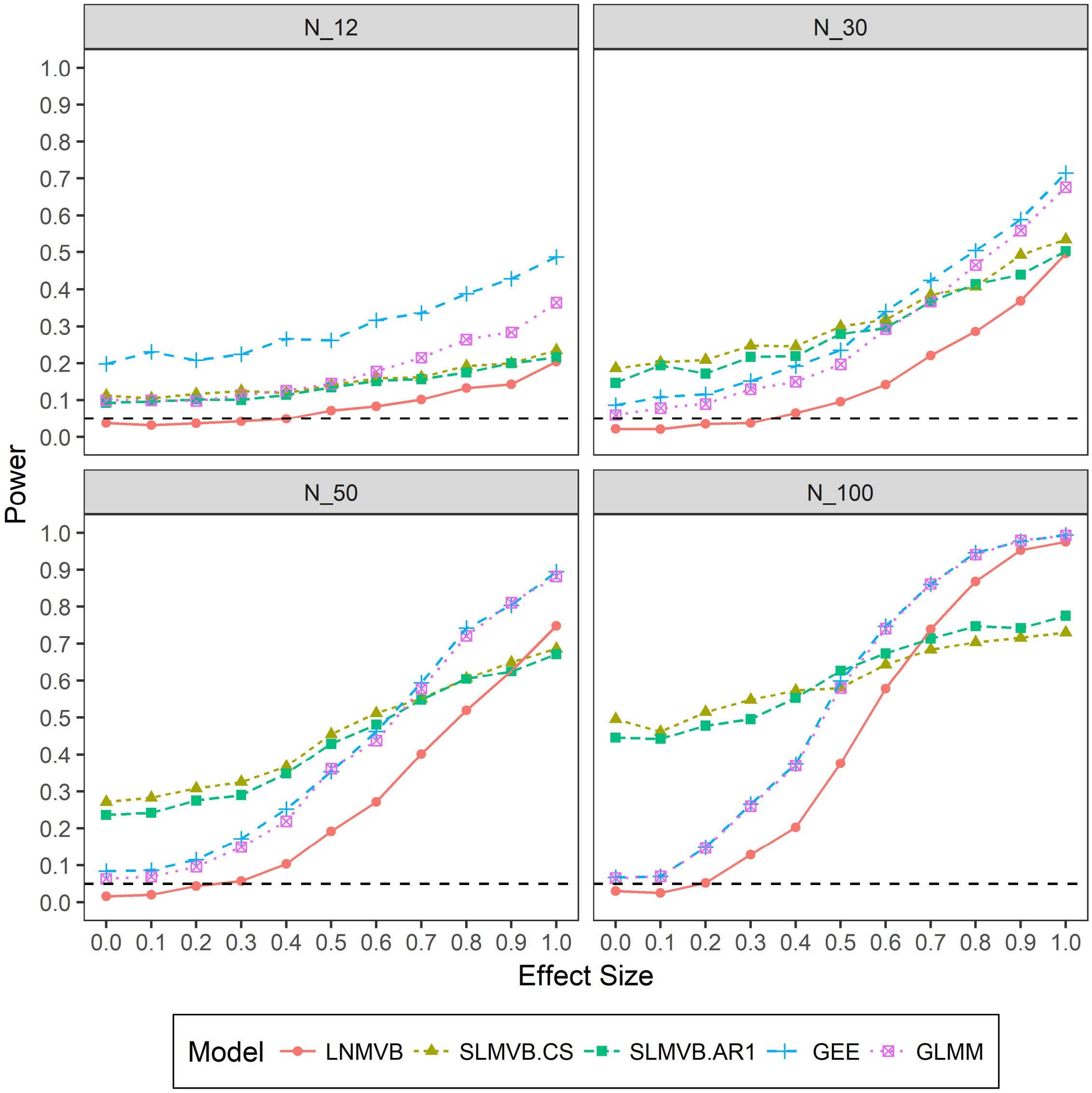


Figure B.9 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

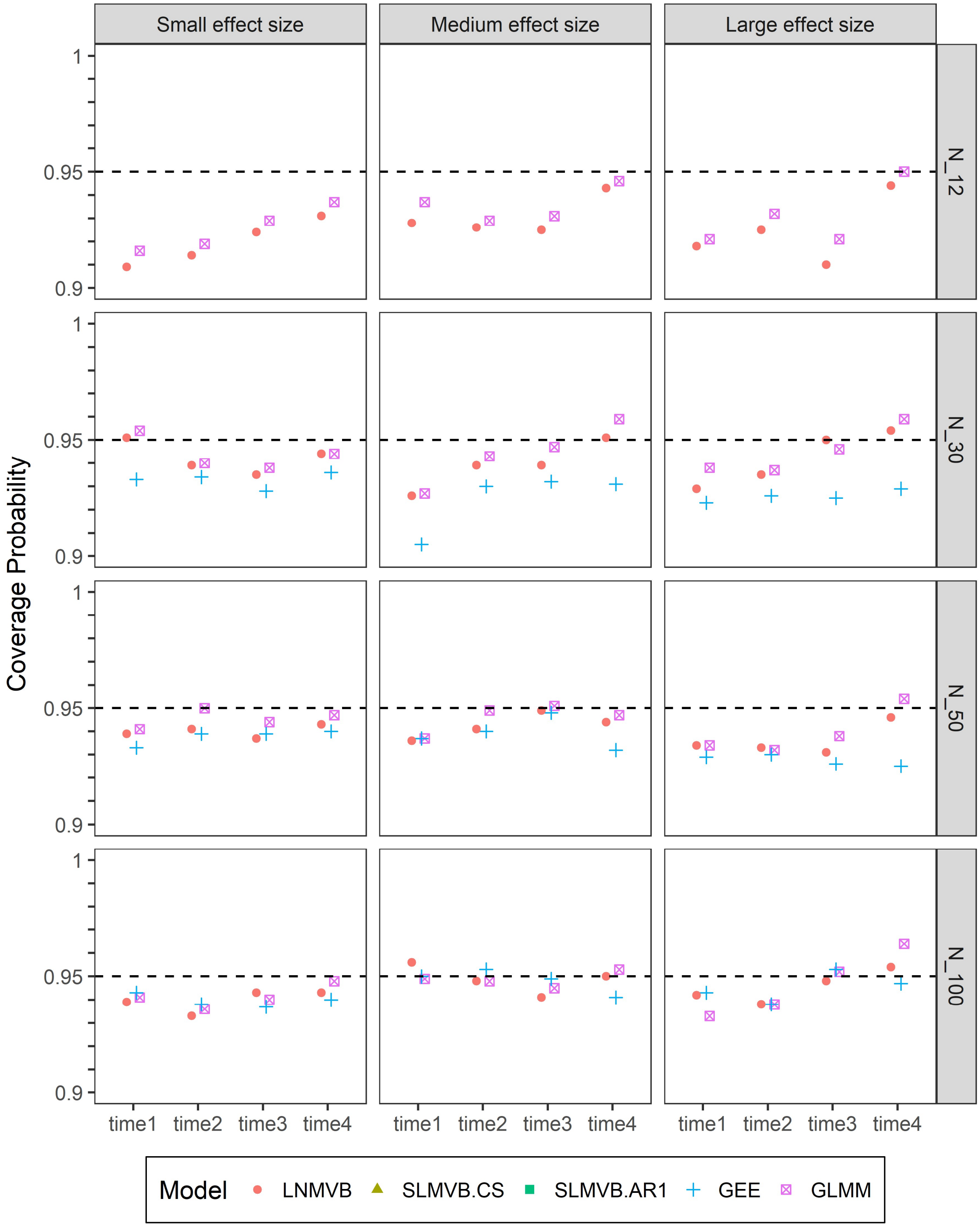


Figure B.10 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5$.

Group with non-stationary location parameter displayed. If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

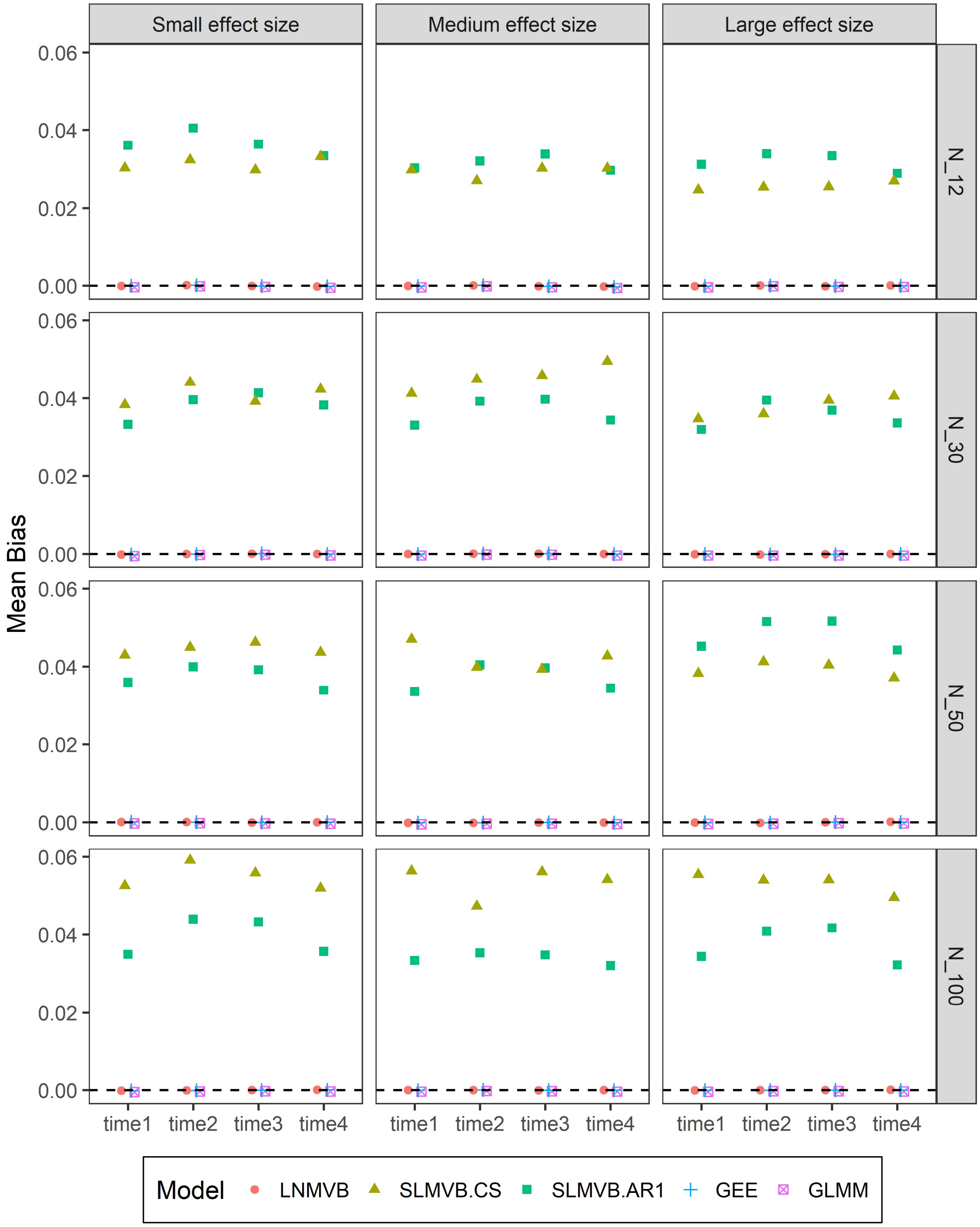


Figure B.11 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5$. Group with non-stationary location parameter displayed.

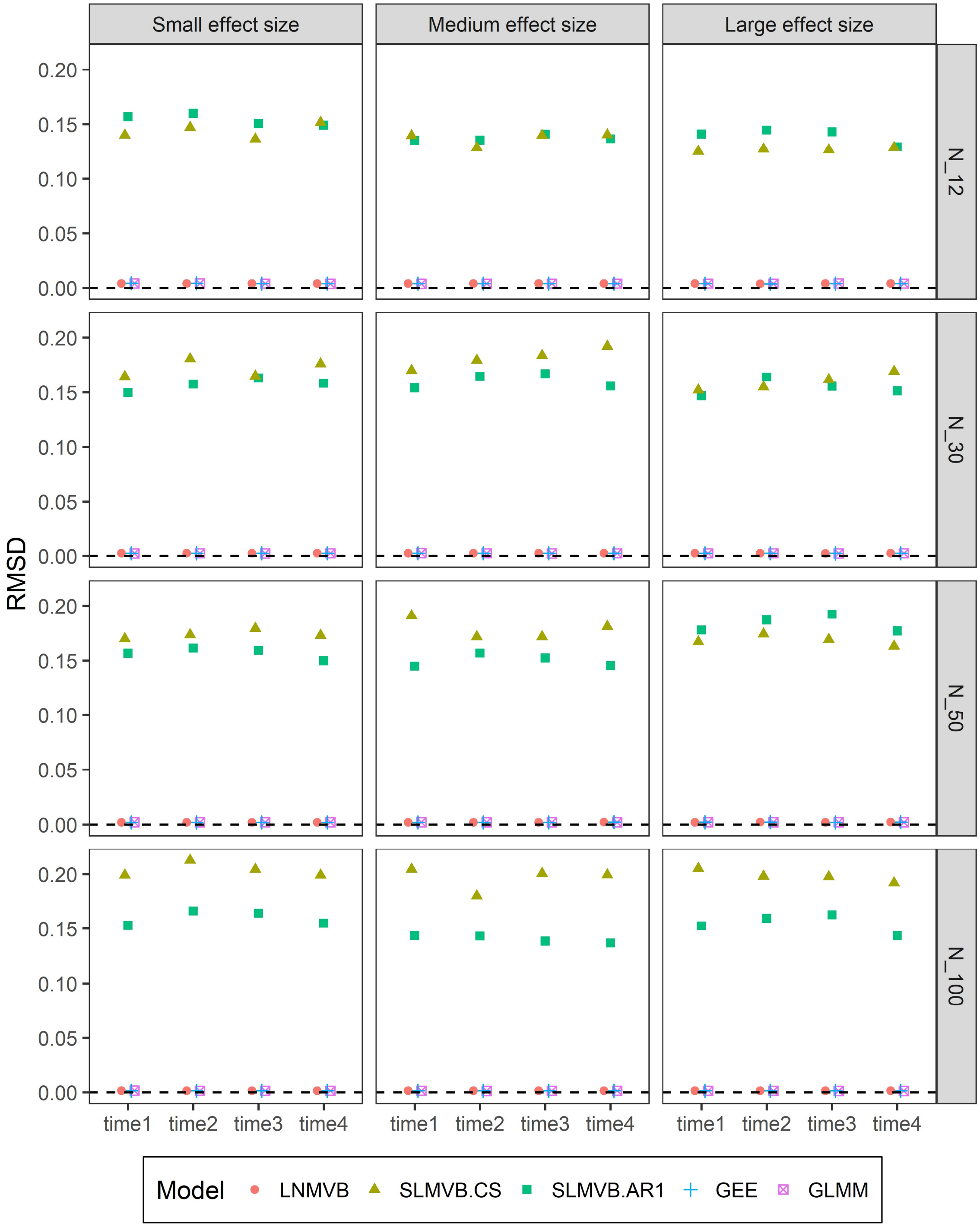


Figure B.12 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.05$ with AR(1) correlation structure, $\rho = 0.5$.

Group with non-stationary location parameter displayed.

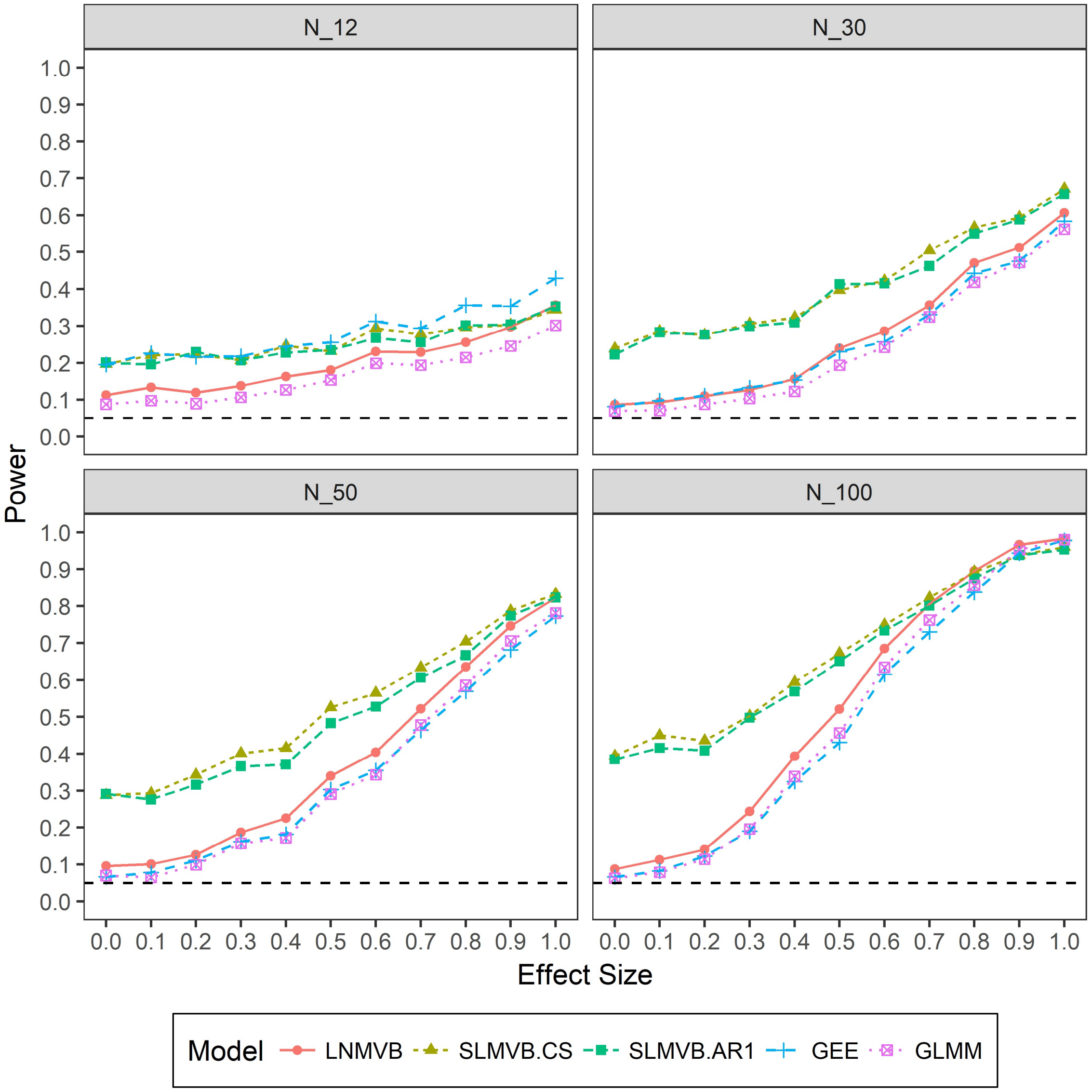


Figure B.13 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

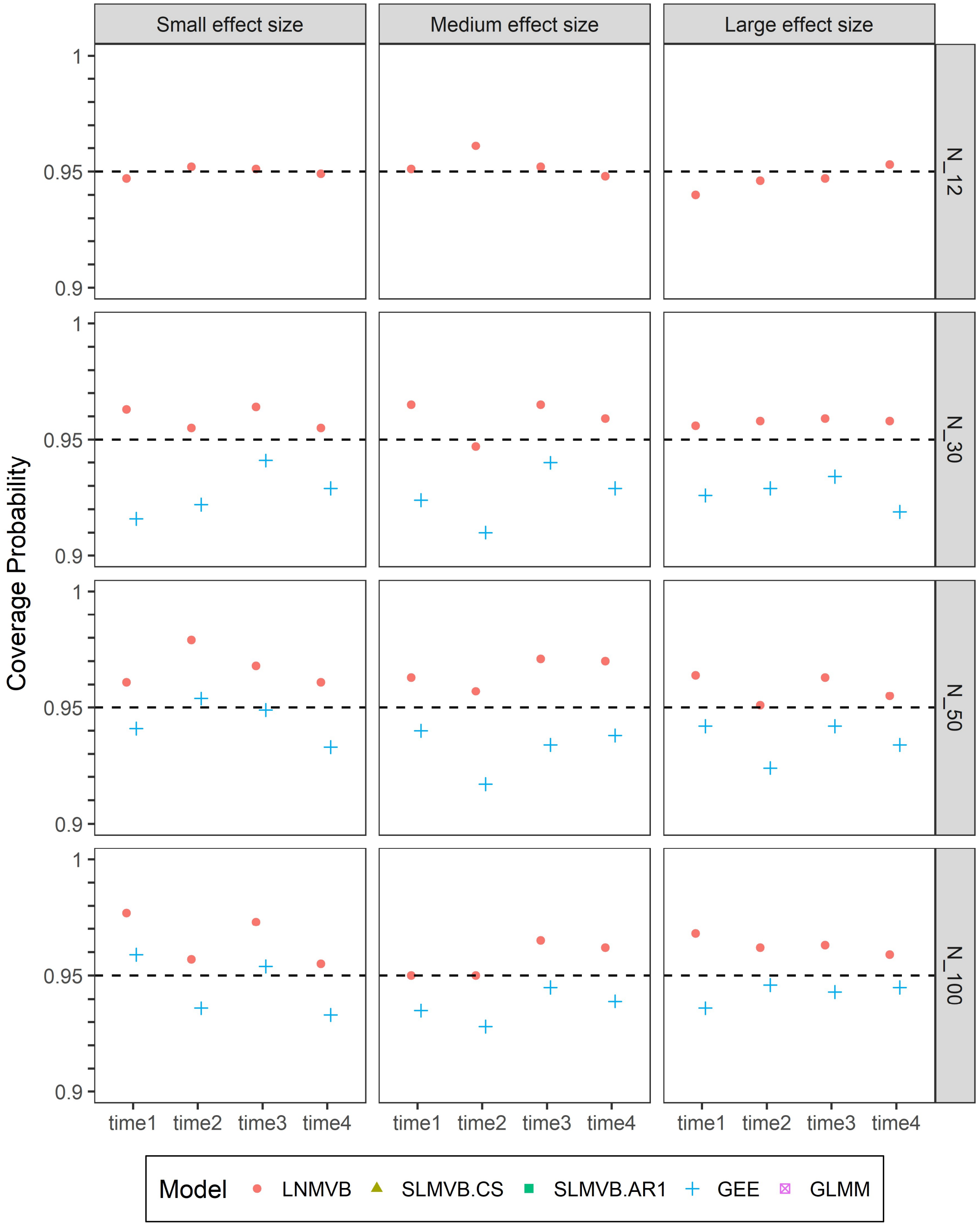


Figure B.14 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$.

Group with non-stationary location parameter displayed. If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

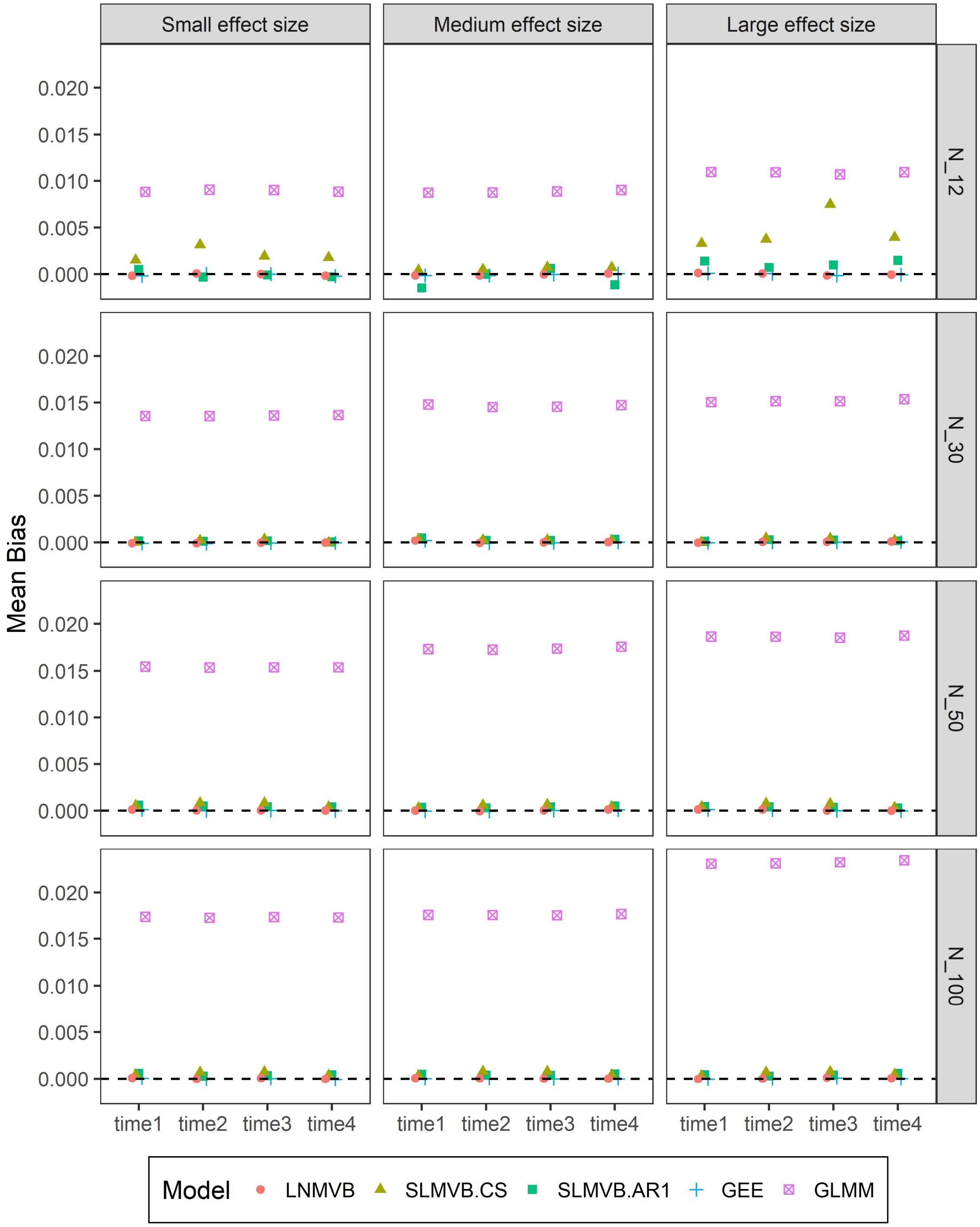


Figure B.15 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$. Group with non-stationary location parameter displayed.

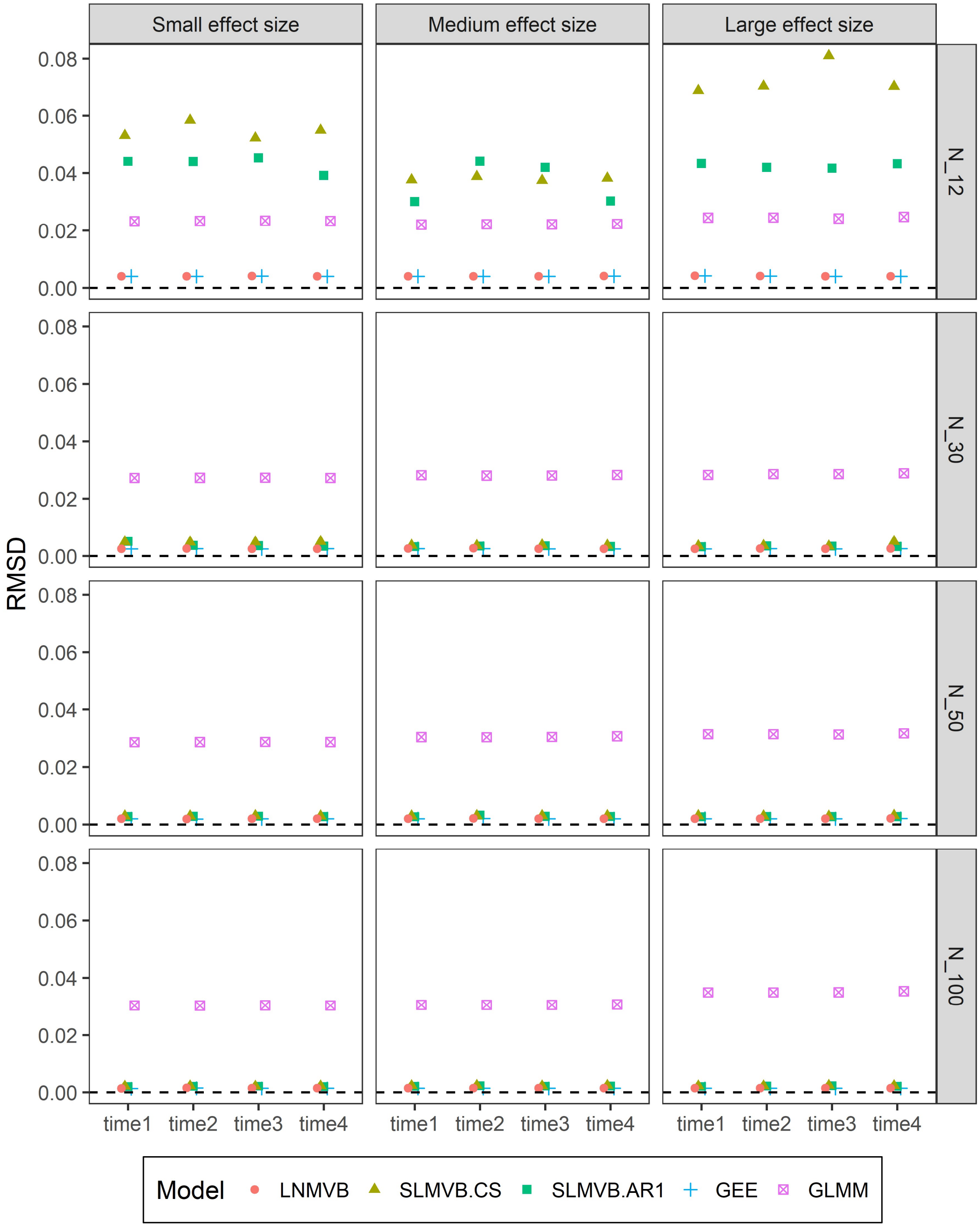


Figure B.16 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.1$.

Group with non-stationary location parameter displayed.

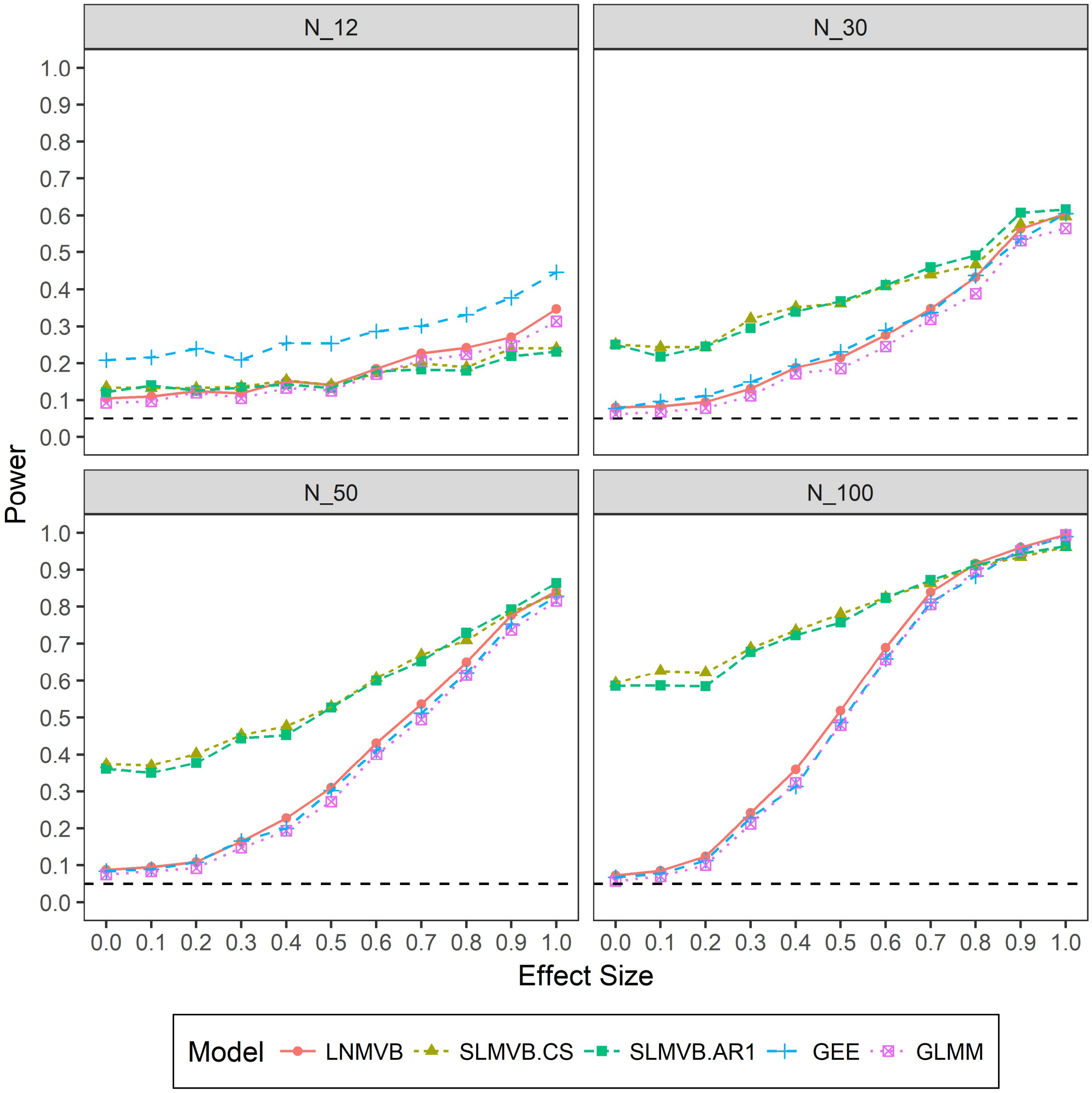


Figure B.17 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

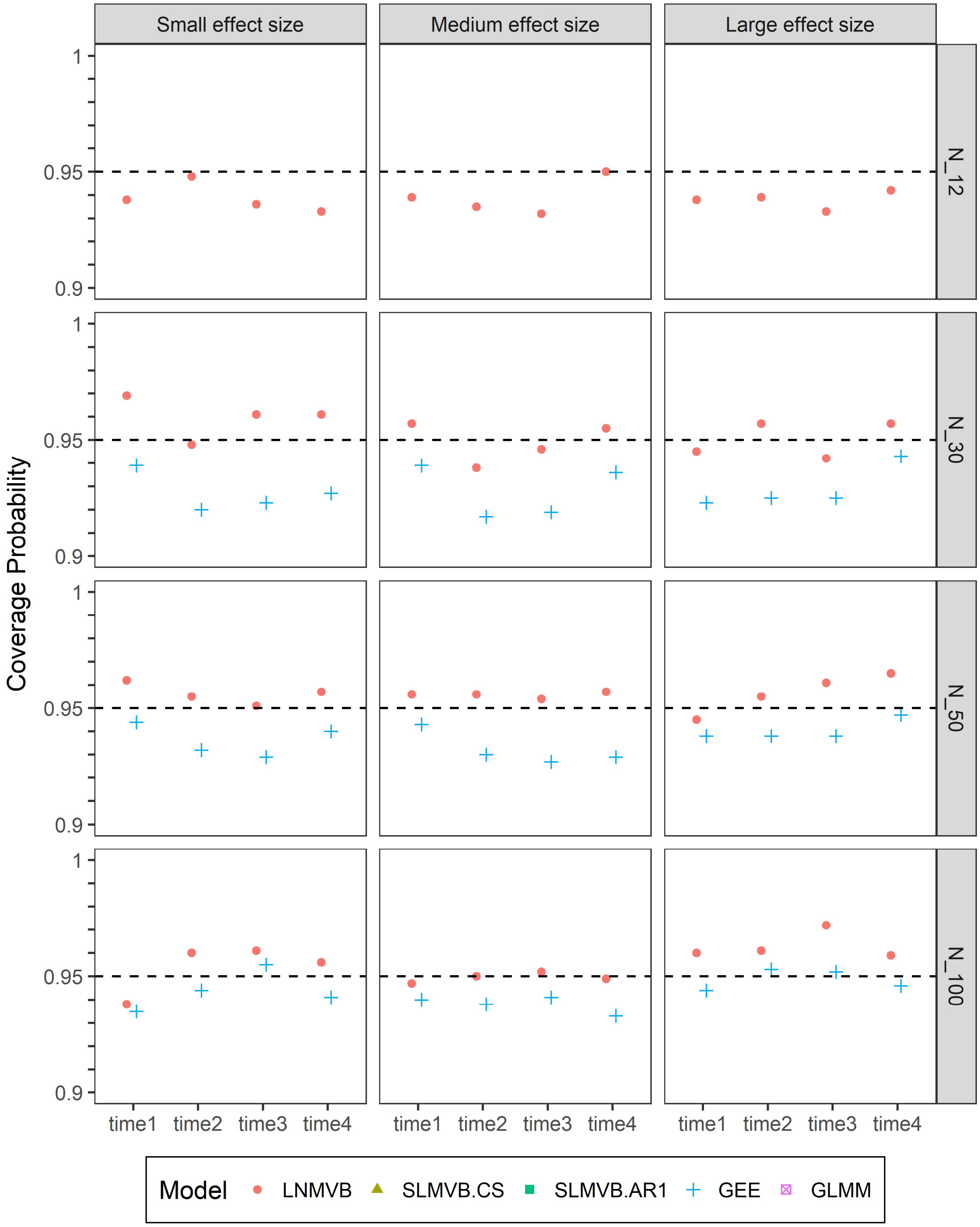


Figure B.18 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$.

Group with non-stationary location parameter displayed. If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

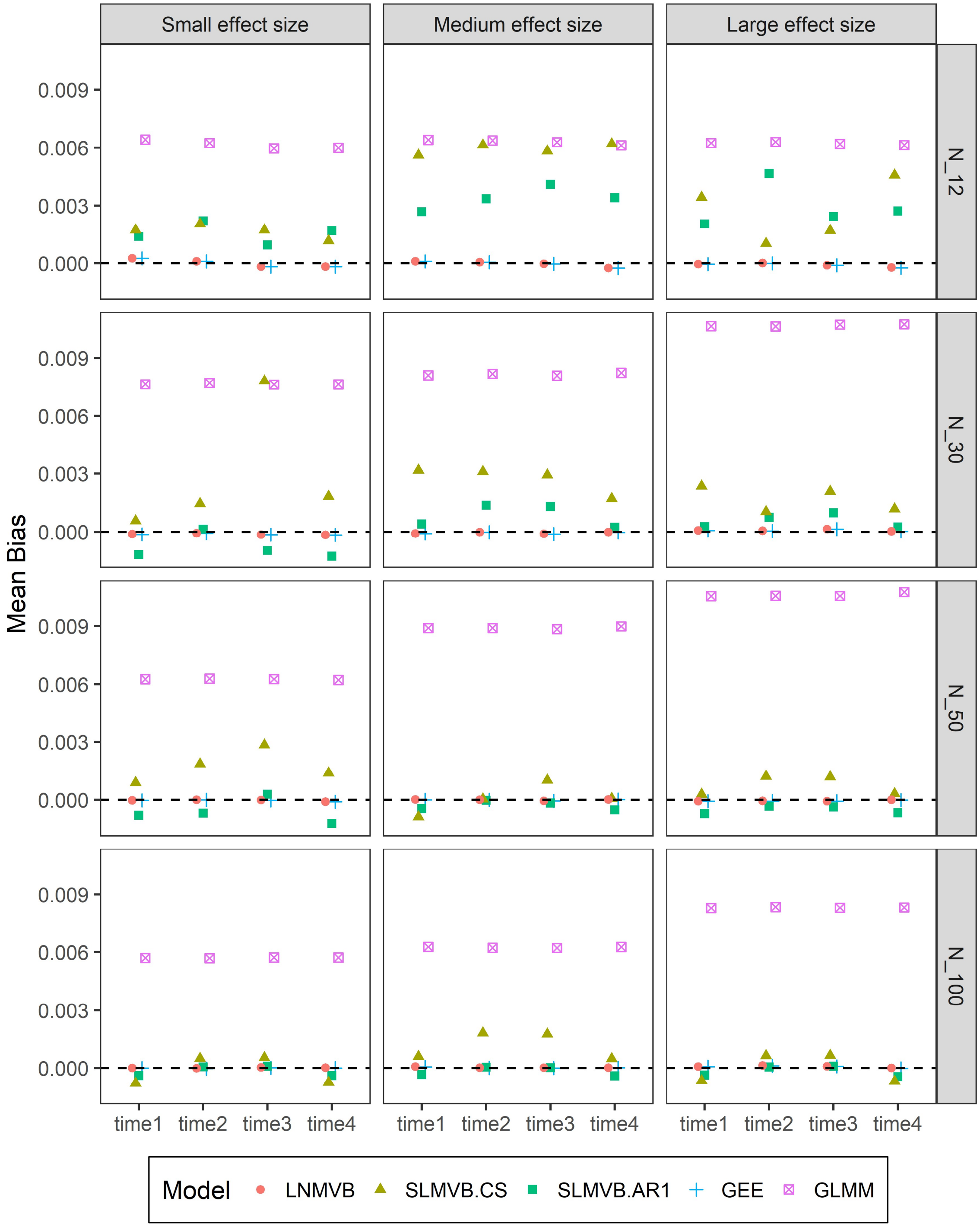


Figure B.19 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$. Group with non-stationary location parameter displayed.

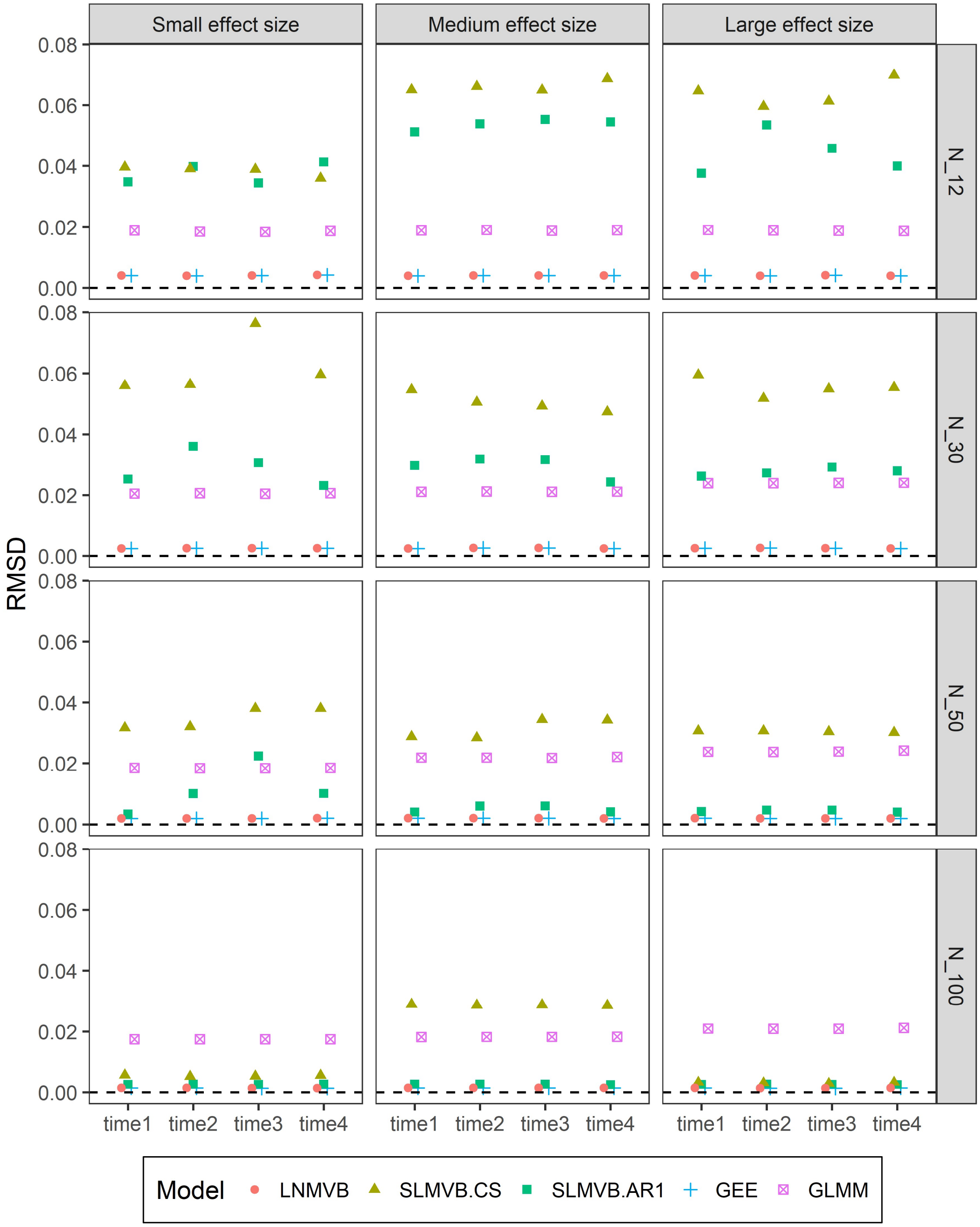


Figure B.20 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.3$.

Group with non-stationary location parameter displayed.

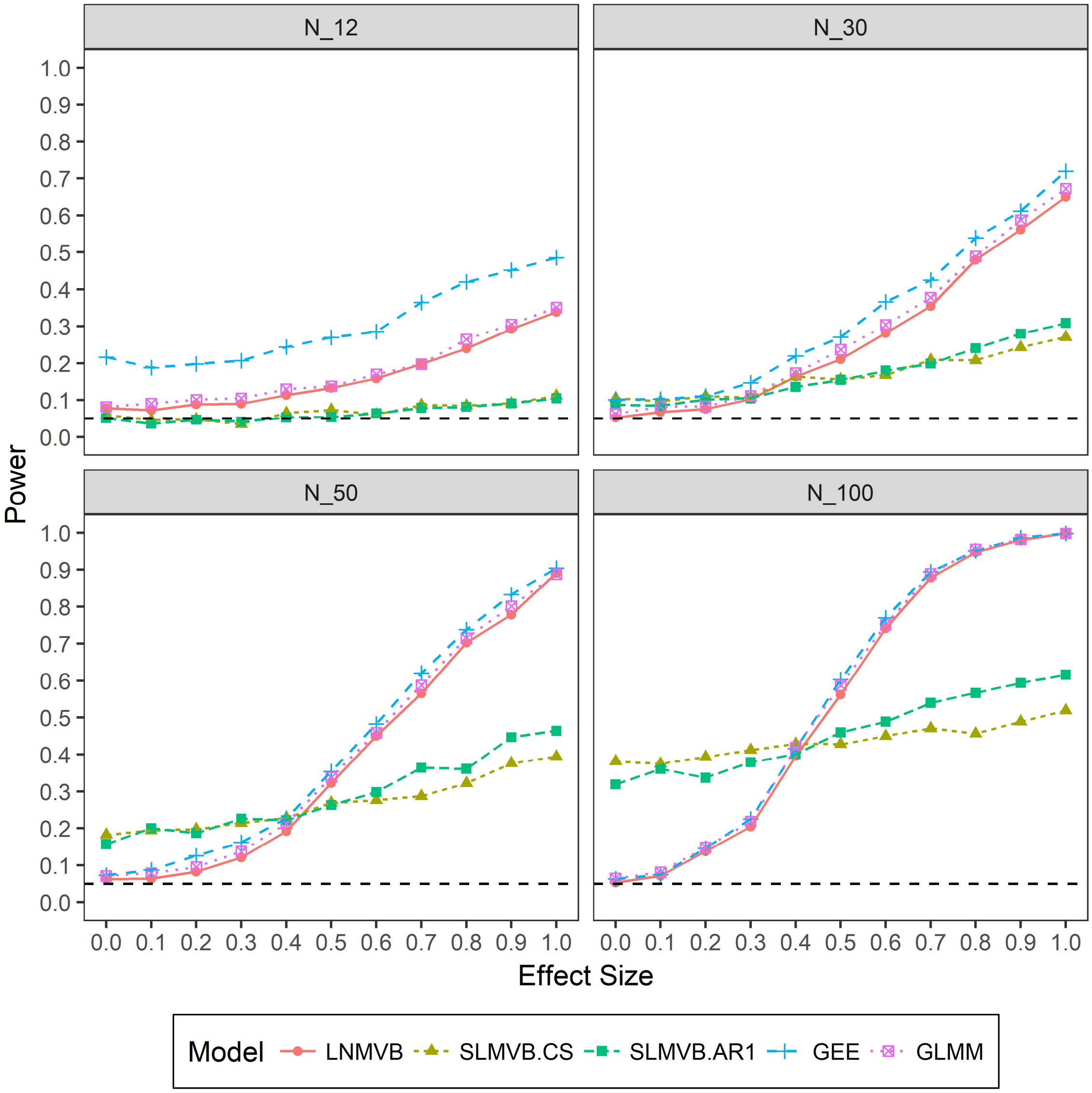


Figure B.21 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB models did not converge for all replicates; estimates of the power may be biased.

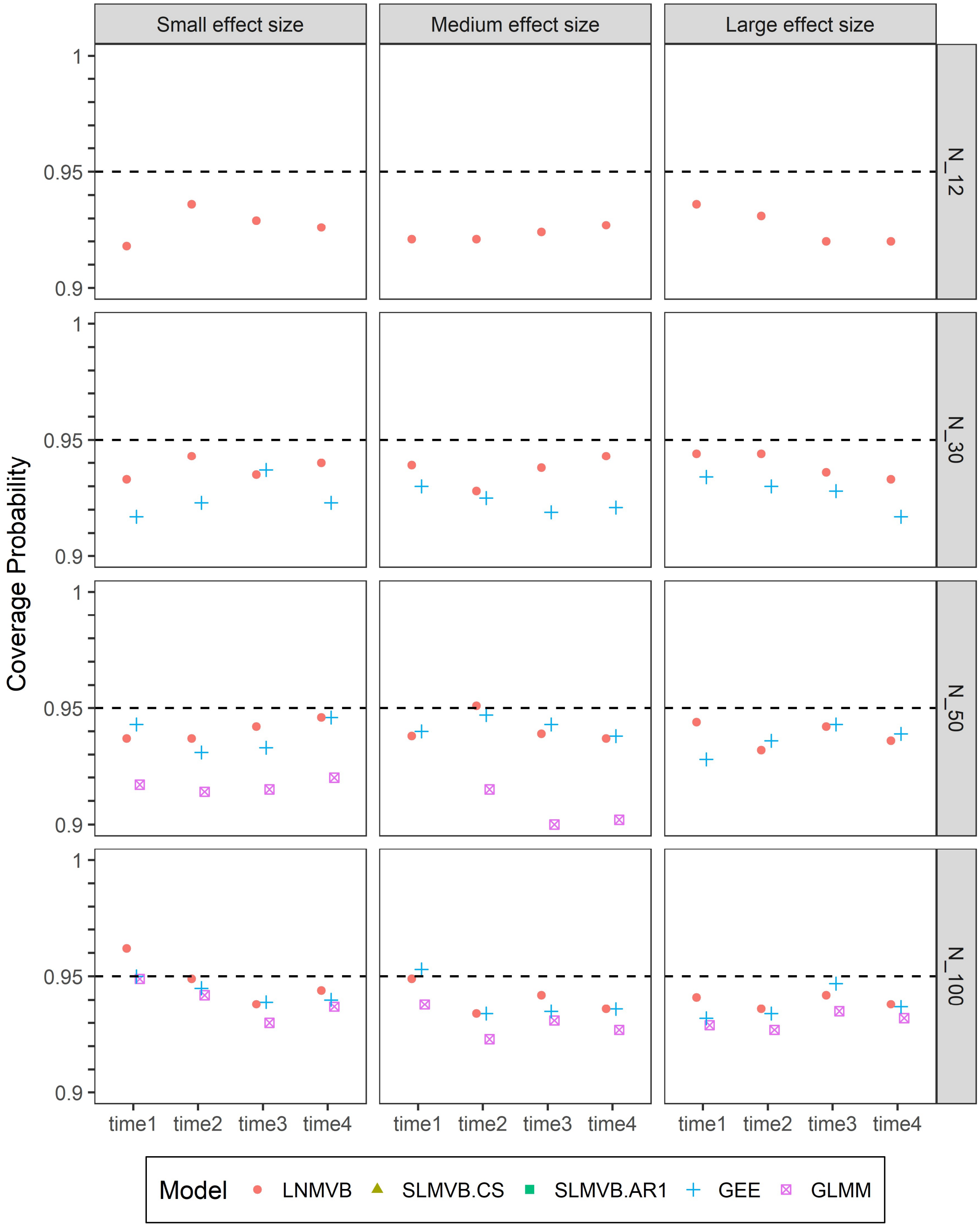


Figure B.22 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$.

Group with non-stationary location parameter displayed. If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

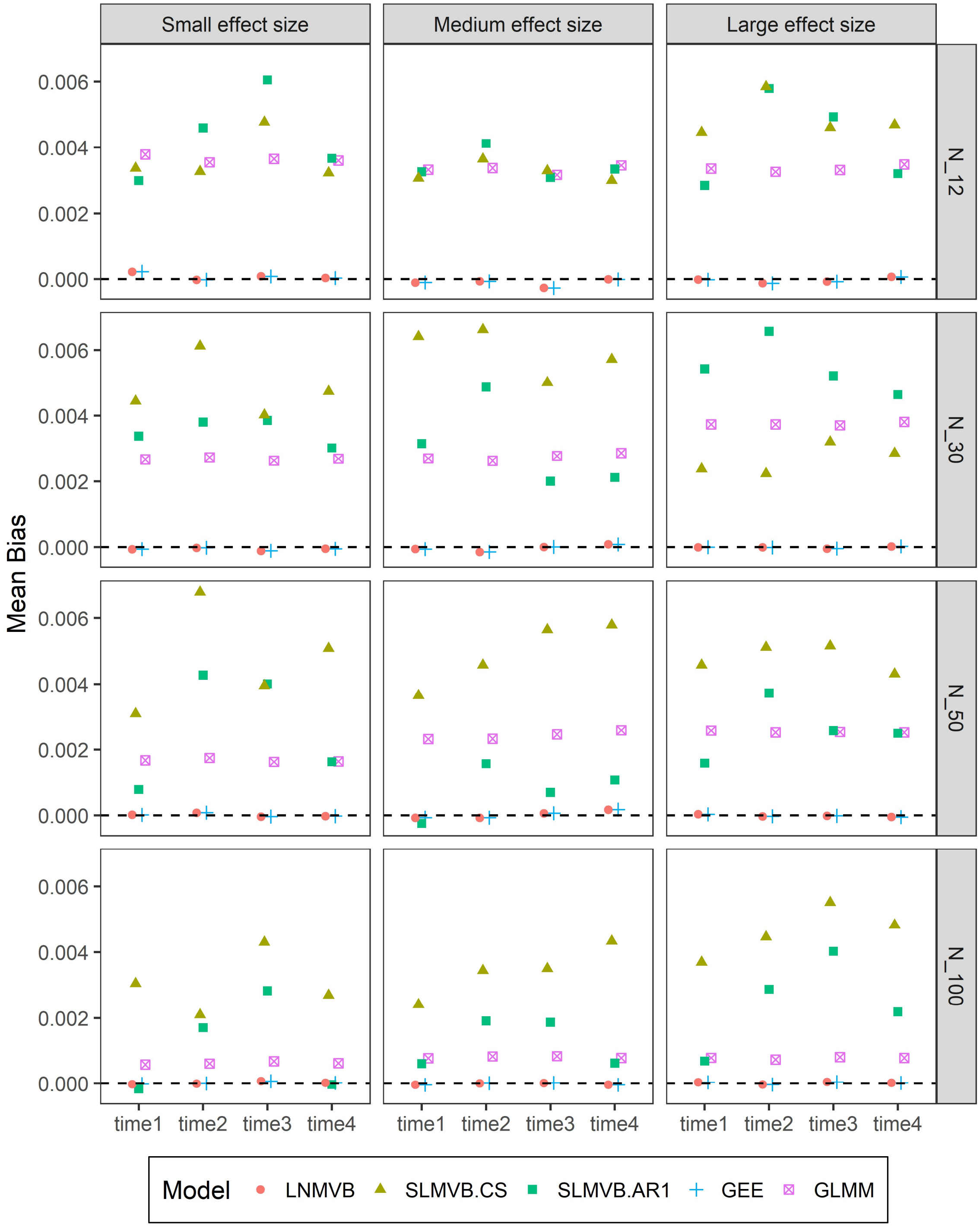


Figure B.23 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$.

Group with non-stationary location parameter displayed.

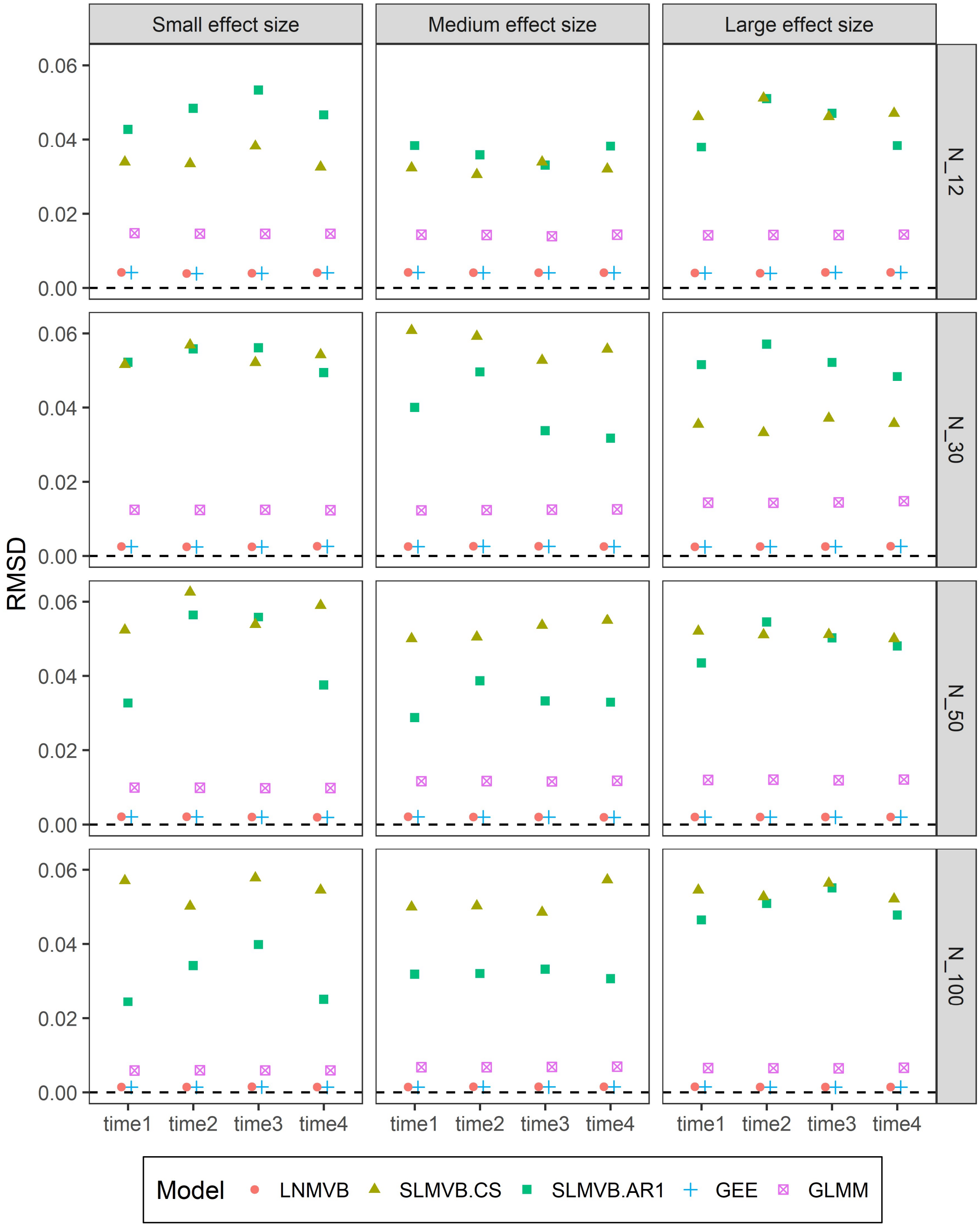


Figure B.24 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.3$ with AR(1) correlation structure, $\rho = 0.5$.

Group with non-stationary location parameter displayed.

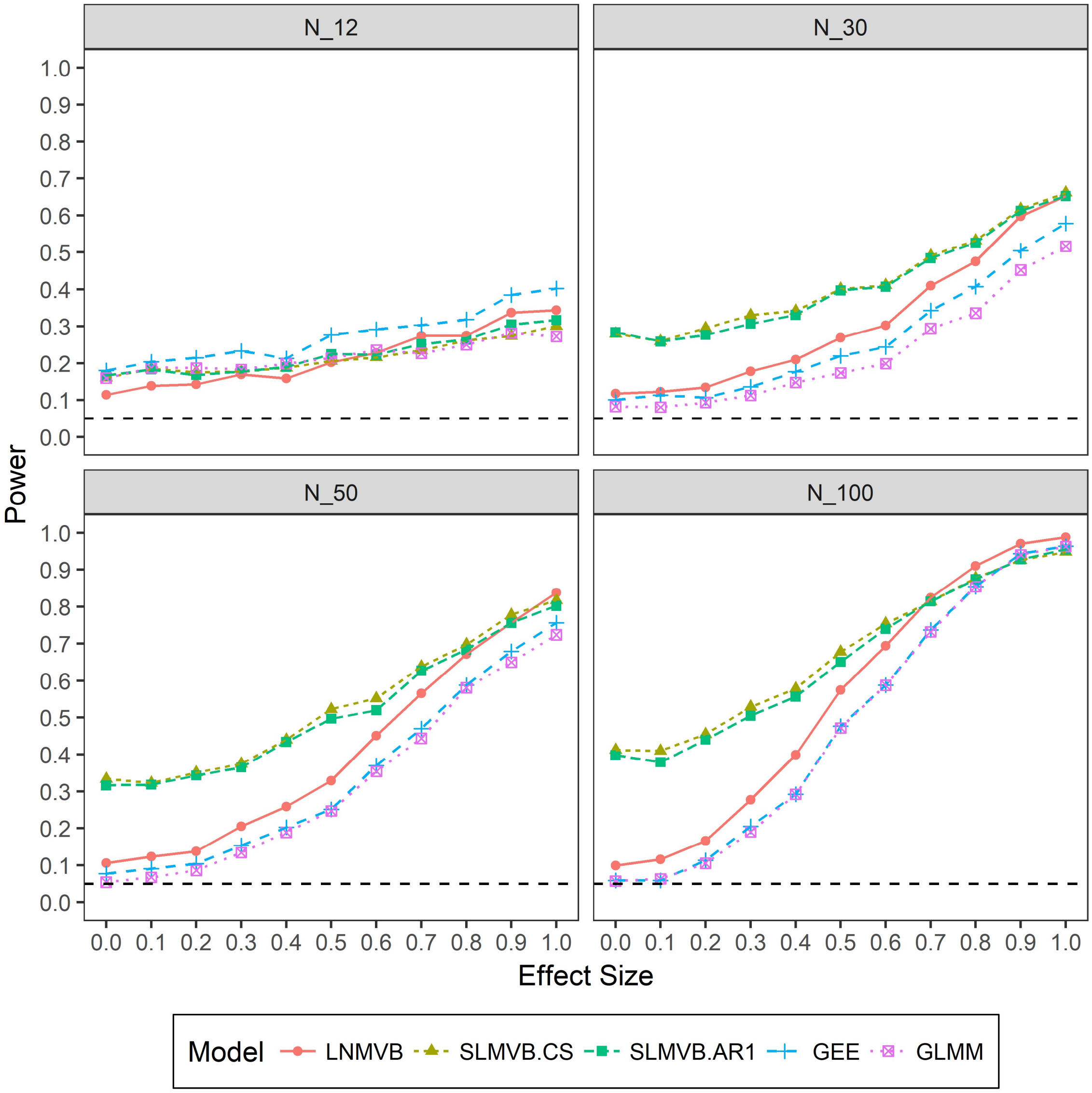


Figure B.25 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB and Beta GLMM models did not converge for all replicates; estimates of the power may be biased.

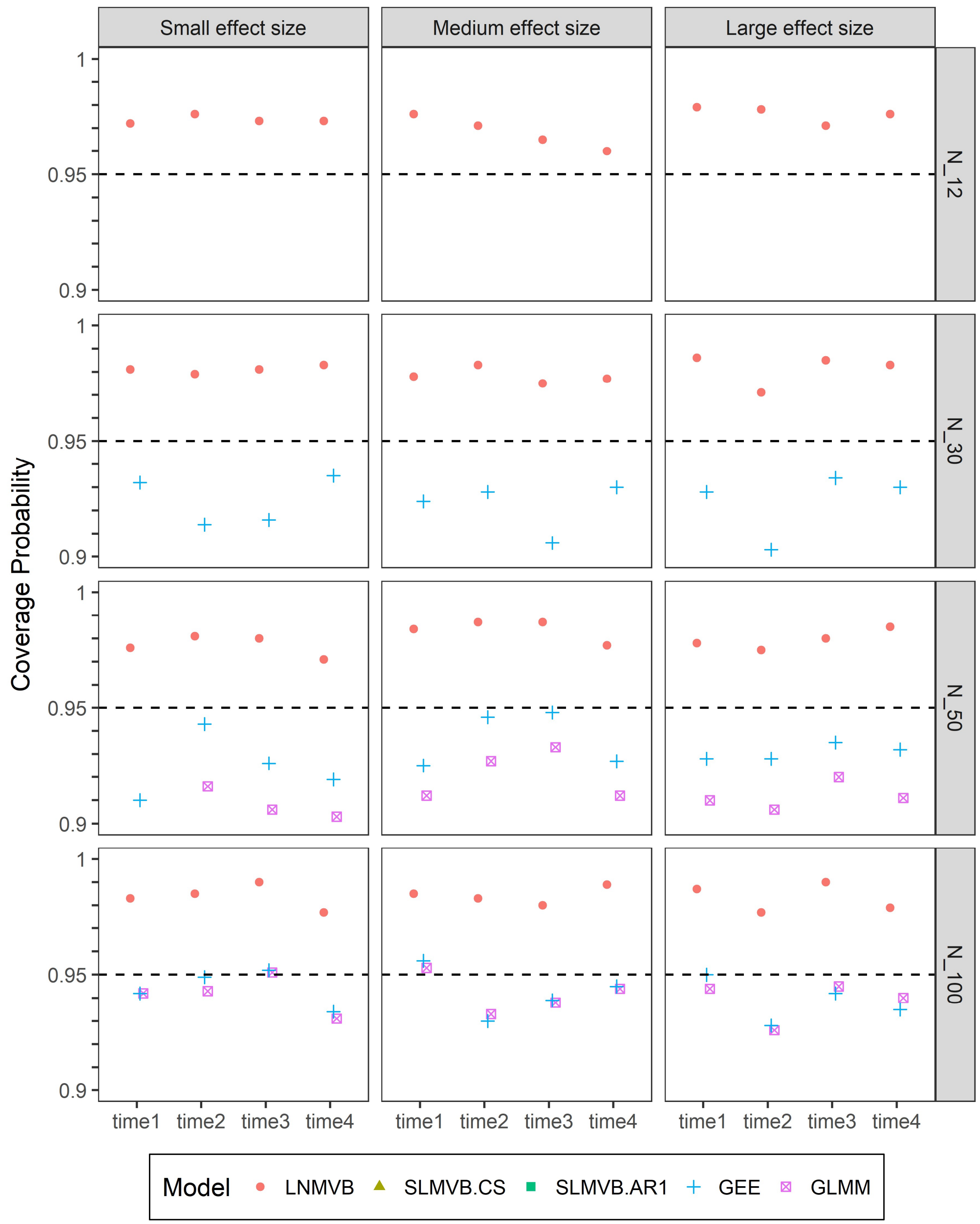


Figure B.26 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$.

Group with non-stationary location parameter displayed. If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

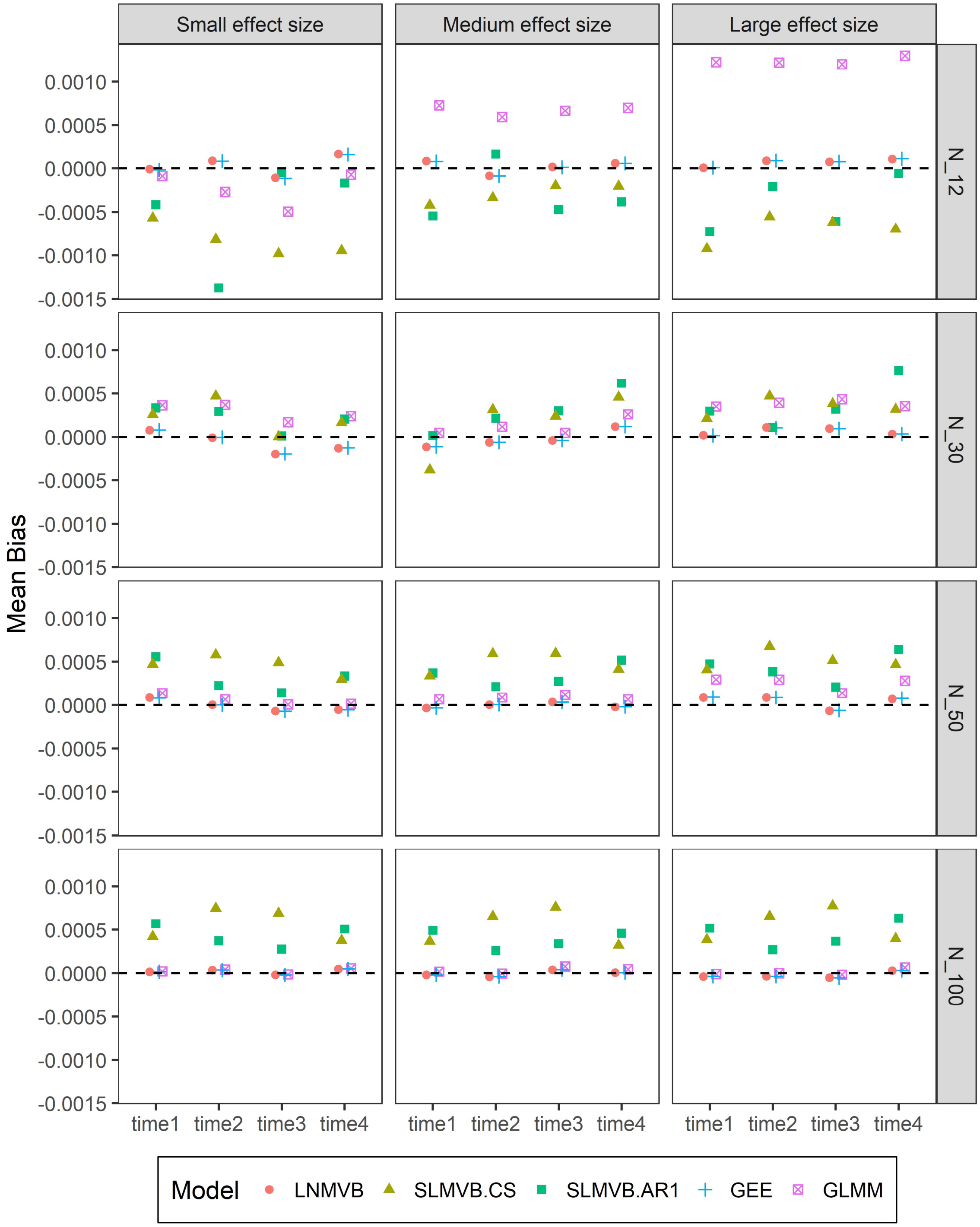


Figure B.27 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$. Group with non-stationary location parameter displayed.

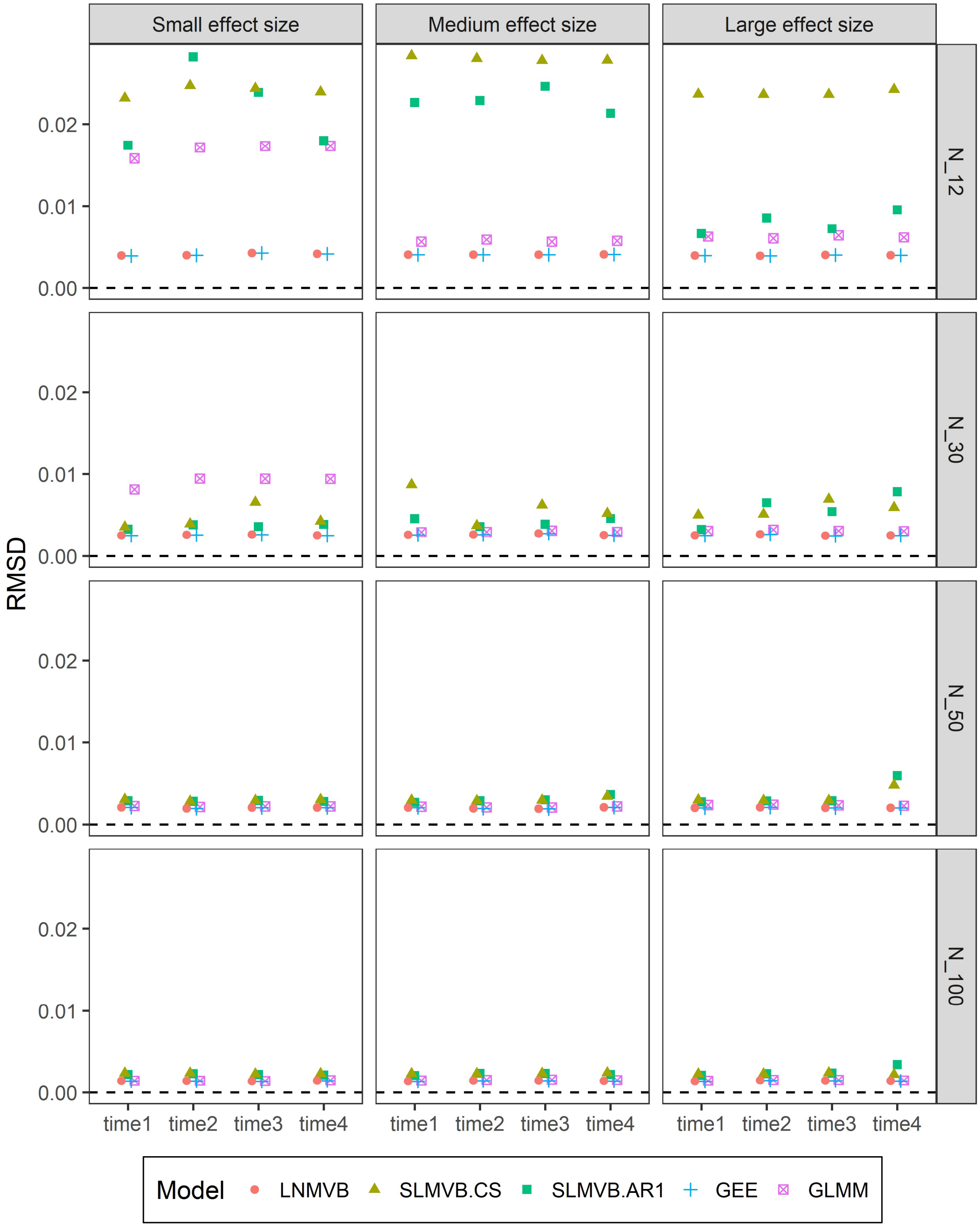


Figure B.28 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.1$.

Group with non-stationary location parameter displayed.

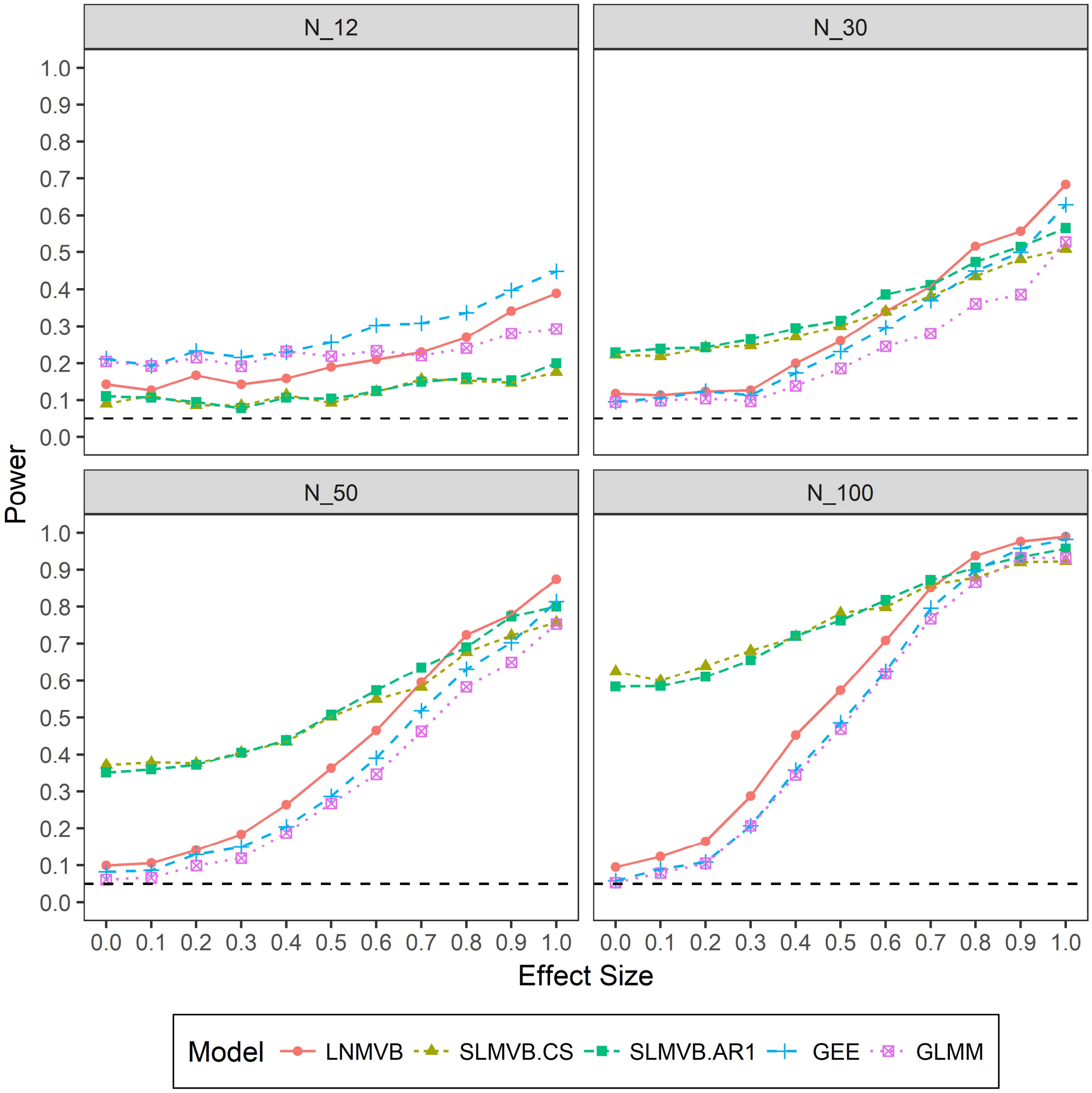


Figure B.29 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB and Beta GLMM models did not converge for all replicates; estimates of the power may be biased.

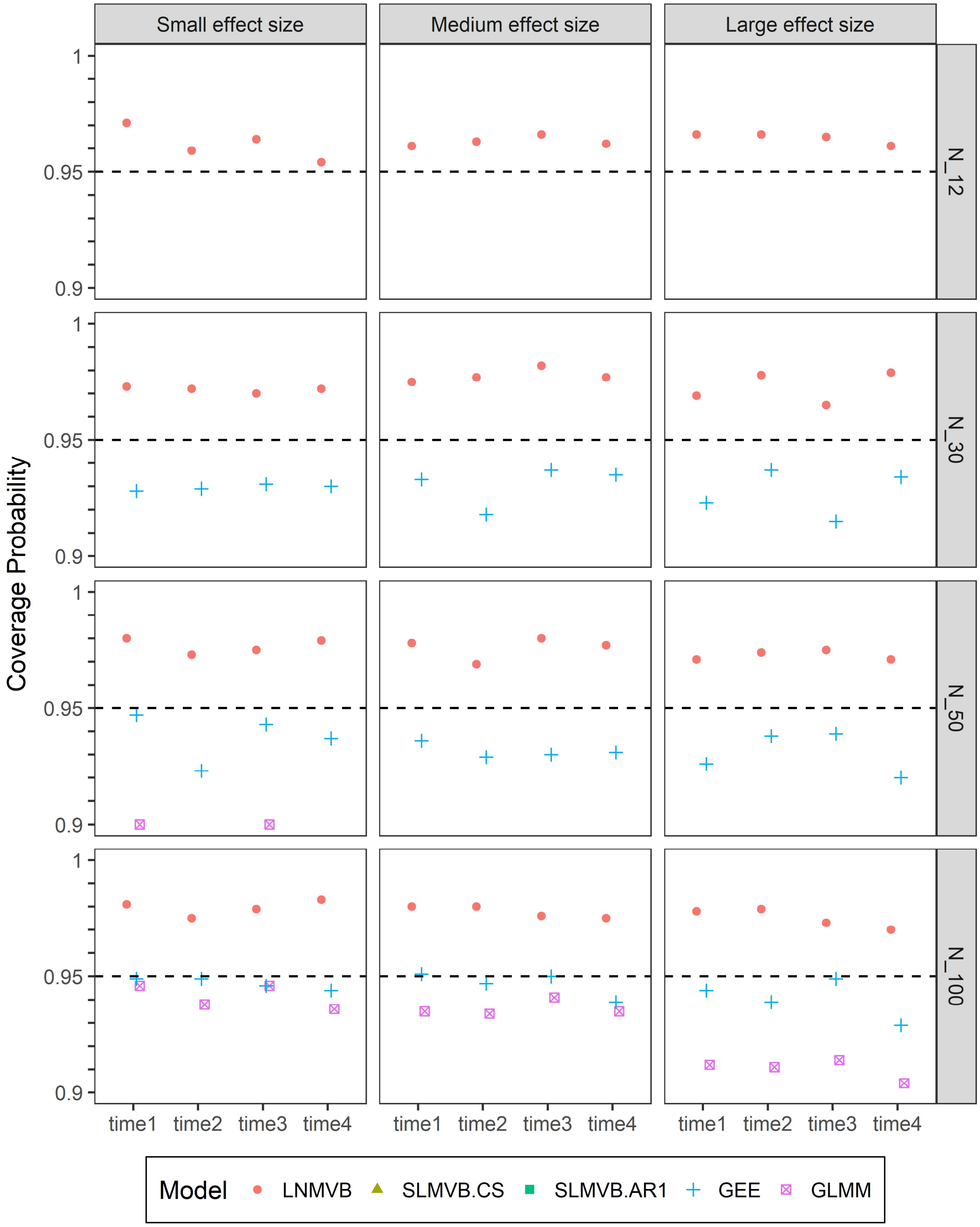


Figure B.30 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3$.

Group with non-stationary location parameter displayed. If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

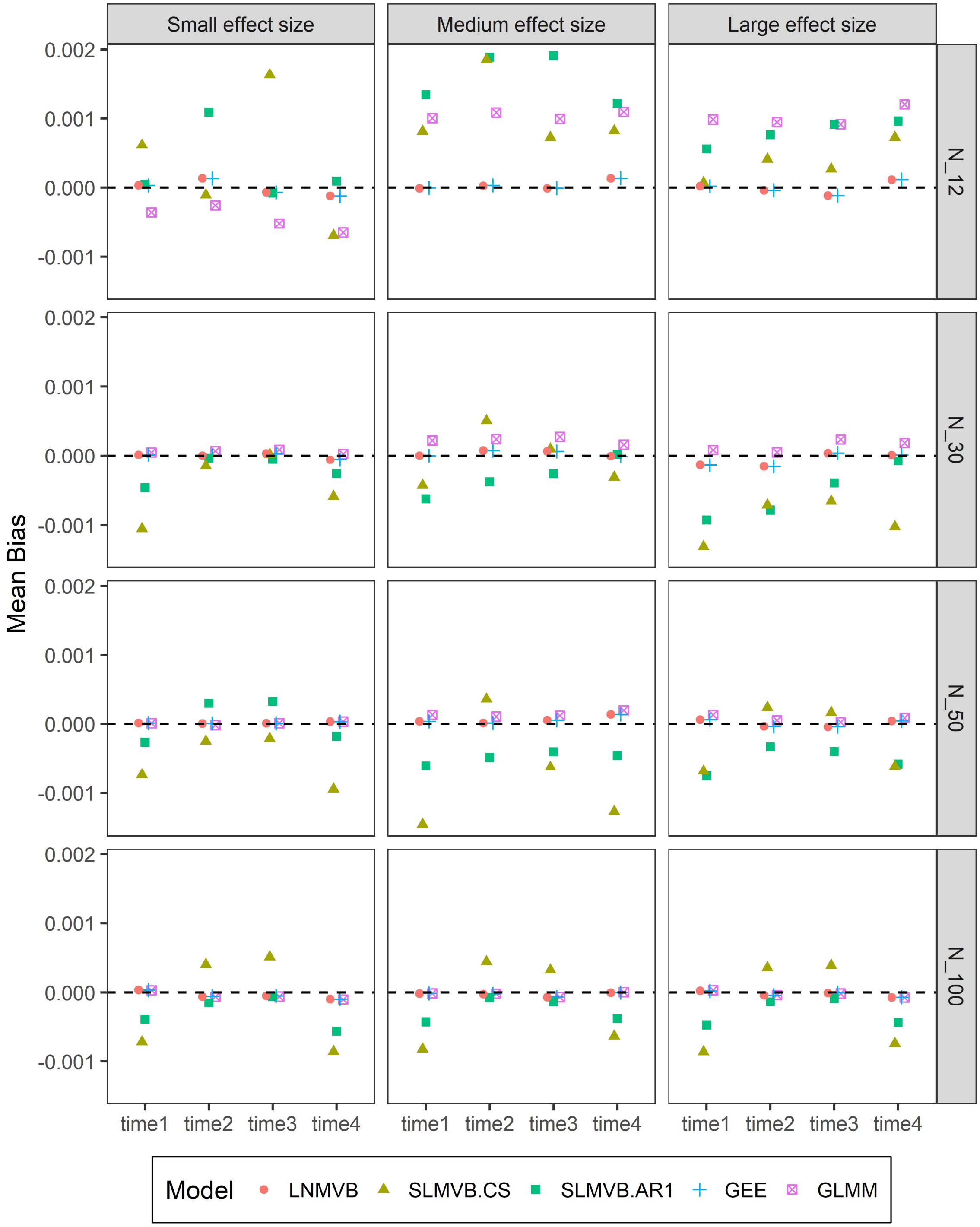


Figure B.31 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3$. Group with non-stationary location parameter displayed.

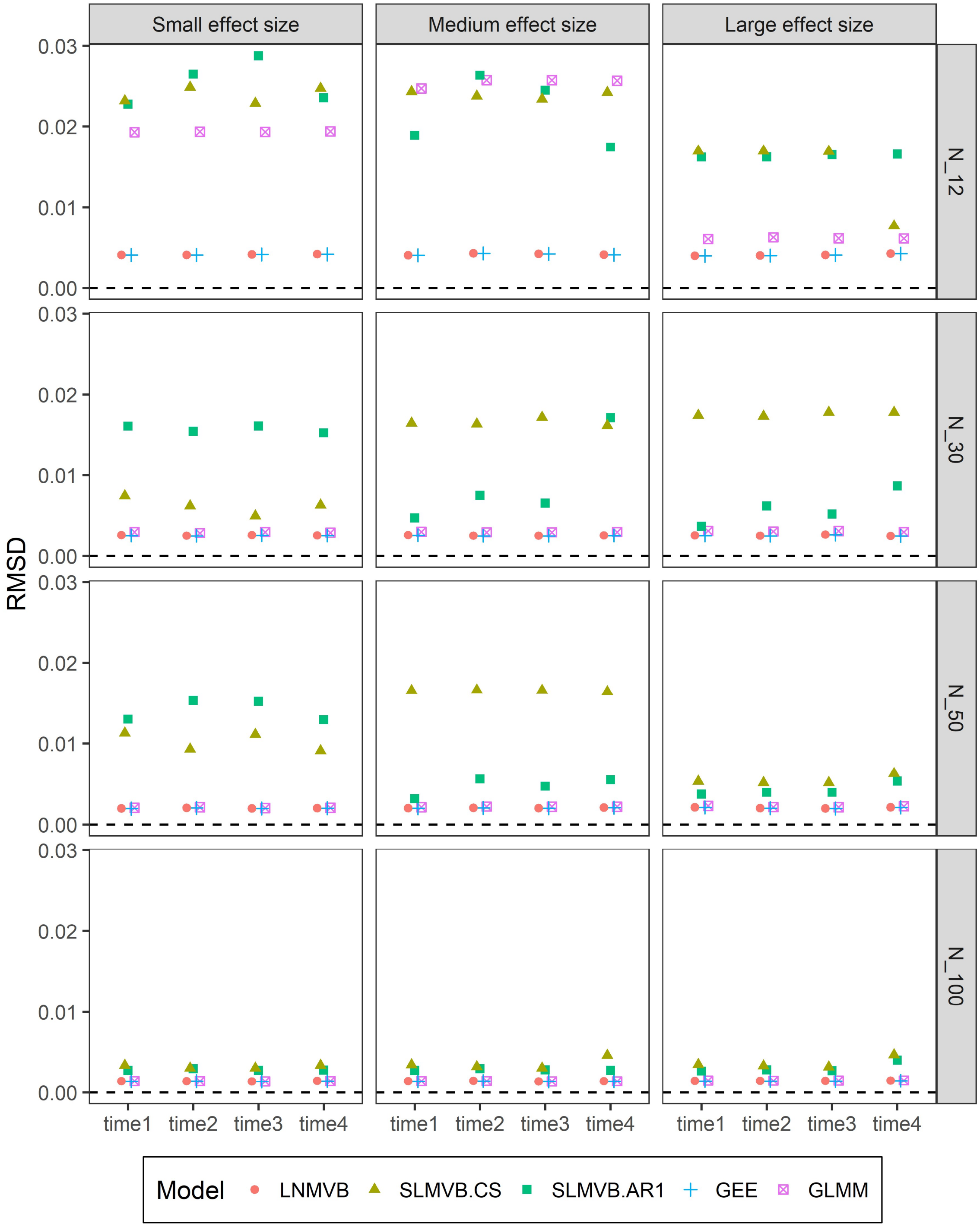


Figure B.32 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.3$.

Group with non-stationary location parameter displayed.

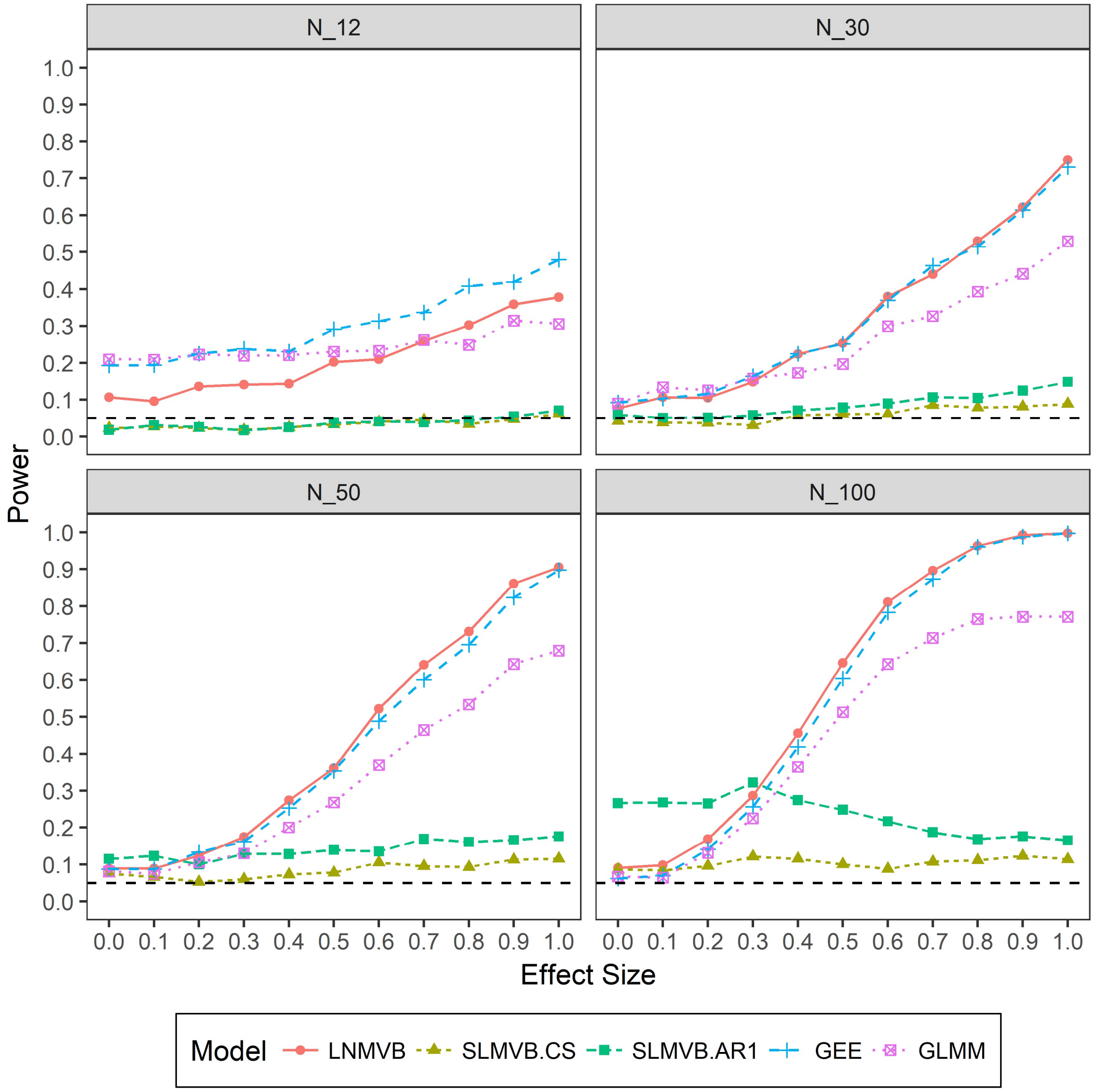


Figure B.33 Empirical power for the overall treatment x time effect for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5$.

Empirical power is calculated as the percentage of F-tests ≤ 0.05 out of 1000 replicates. Effect size 0 represents the Type I error. SLMVB and Beta GLMM models did not converge for all replicates; estimates of the power may be biased.

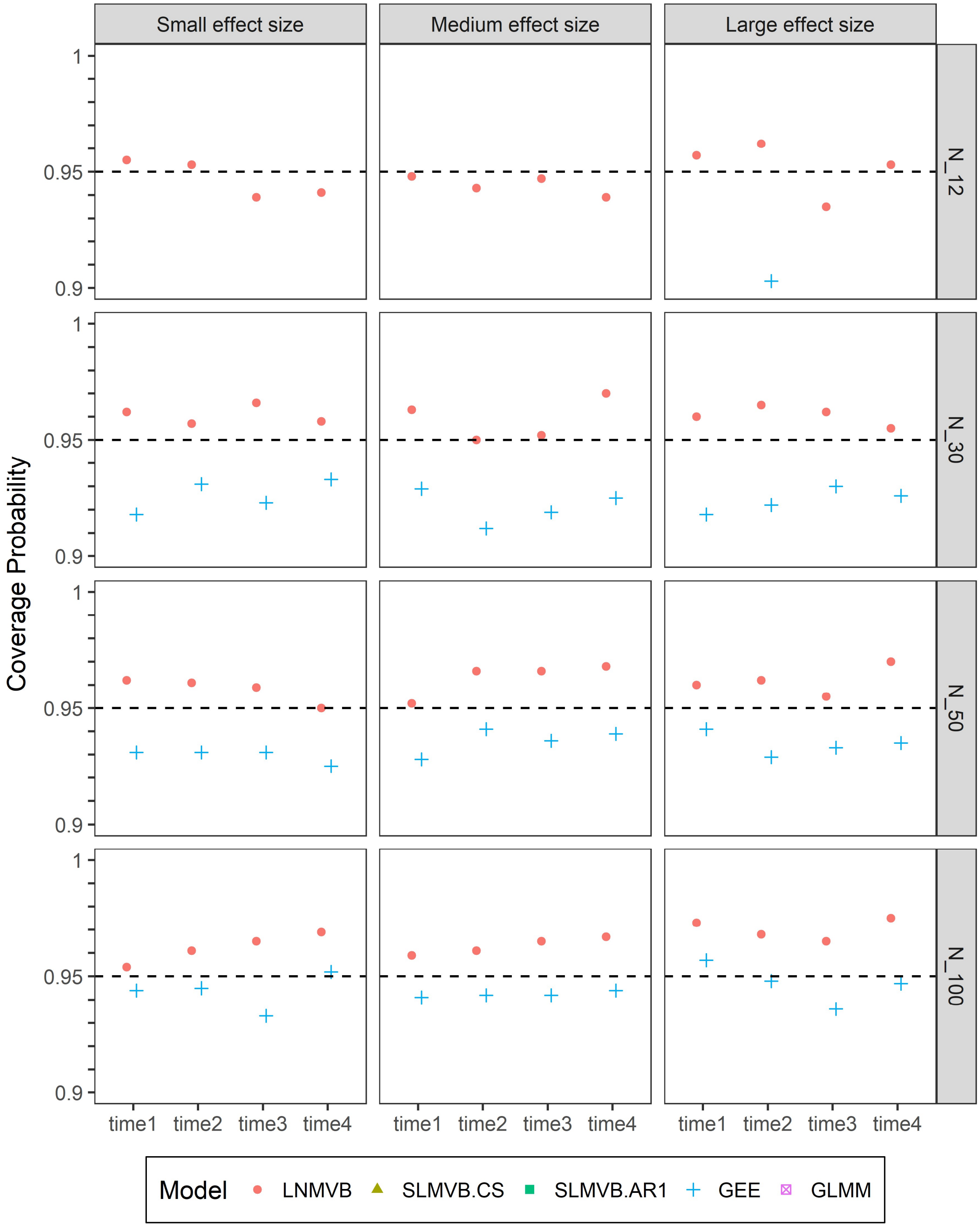


Figure B.34 Coverage probabilities of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5$.

Group with non-stationary location parameter displayed. If Hessian matrix was singular, we considered effected estimates to not contain the true parameter value.

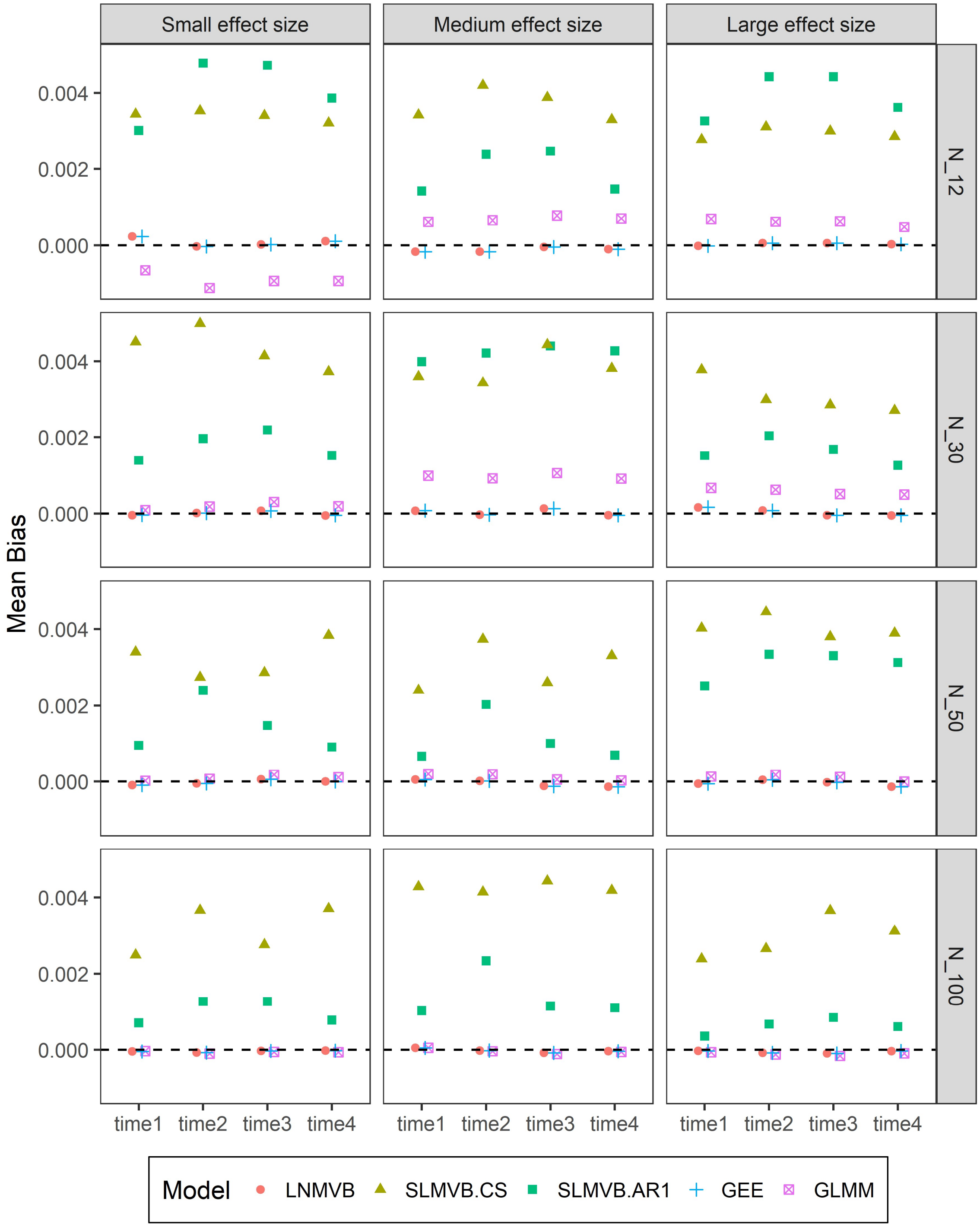


Figure B.35 Mean bias of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5$.

Group with non-stationary location parameter displayed.

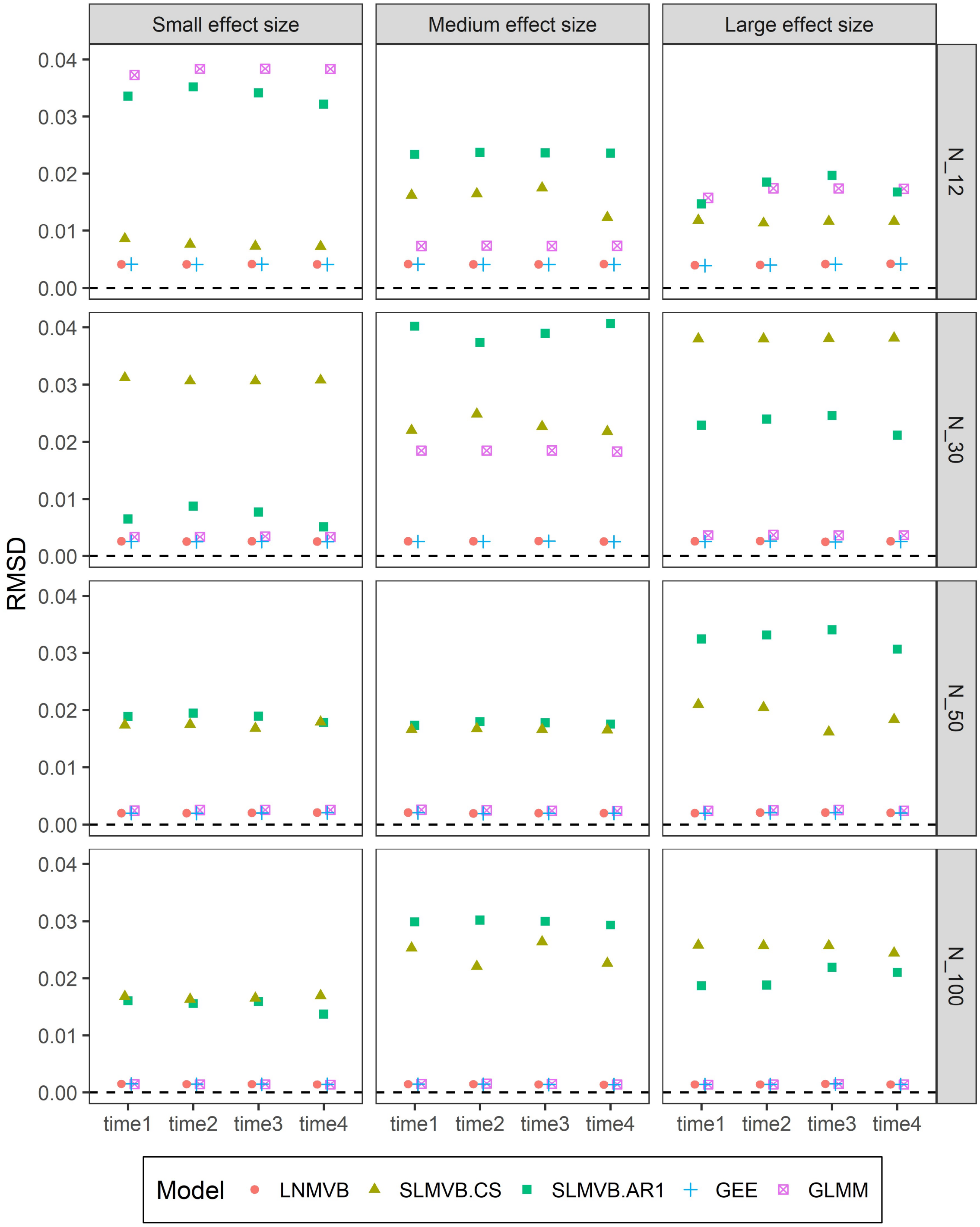


Figure B.36 Root mean squared deviation of the location estimate for two groups, 1000 replicates simulated around $\mu = 0.5$ with AR(1) correlation structure, $\rho = 0.5$.

Group with non-stationary location parameter displayed.

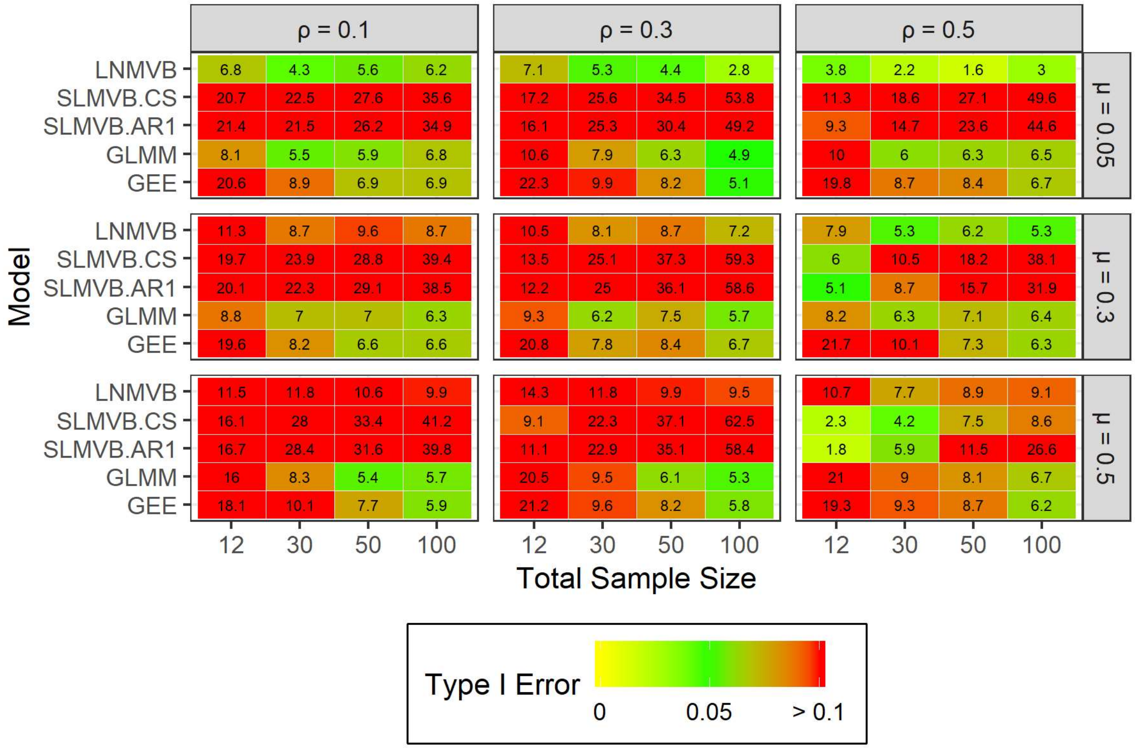


Figure B.37 Summary of Type I error for two group simulations using 1000 replicates with AR(1) correlation structure.

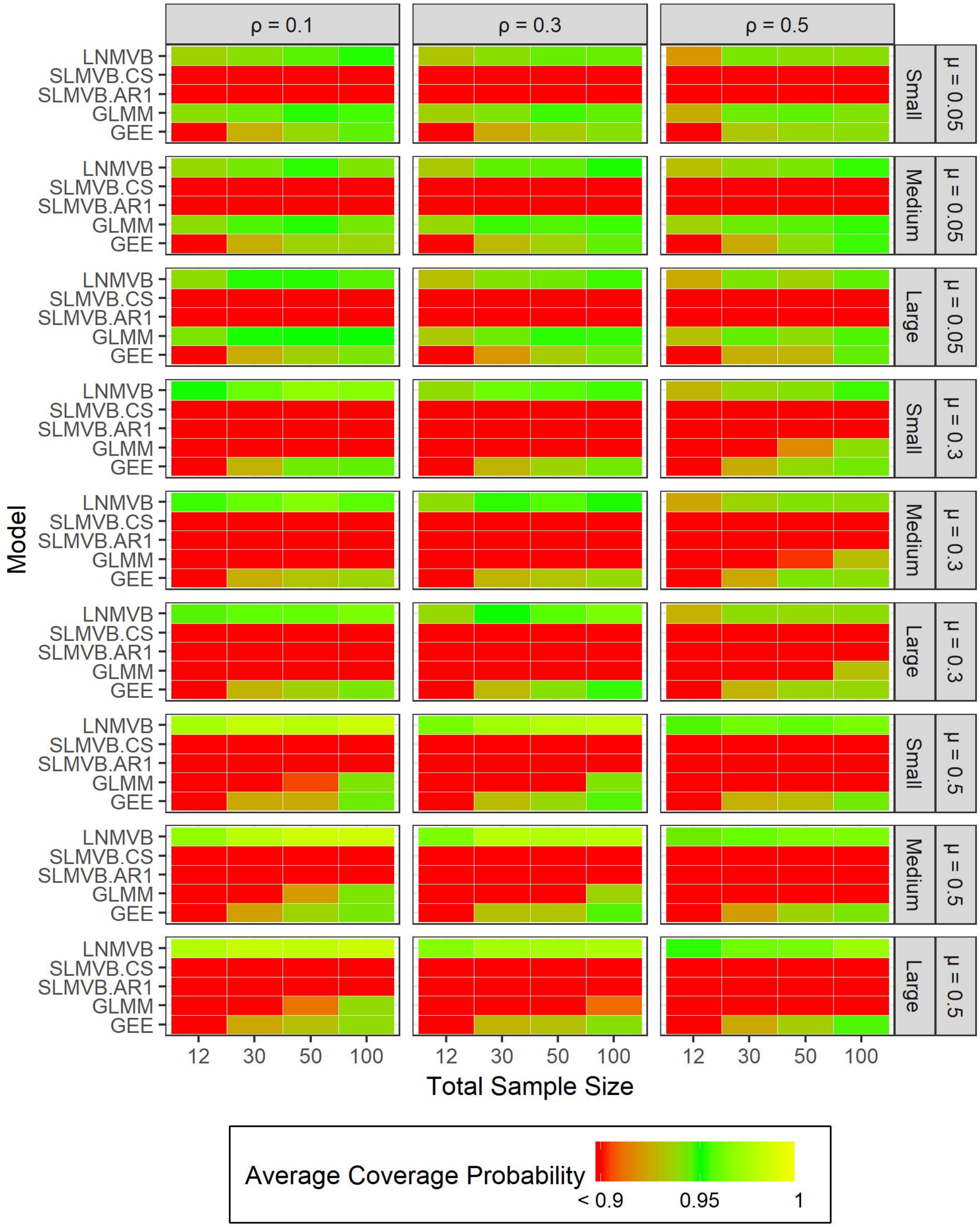


Figure B.38 Summary of coverage probabilities for two group simulations using 1000 replicates with AR(1) correlation structure.

Average coverage probabilities are the mean of the coverage probabilities of the 4 repeated-measures. Small, medium, and large represent the effect sizes.

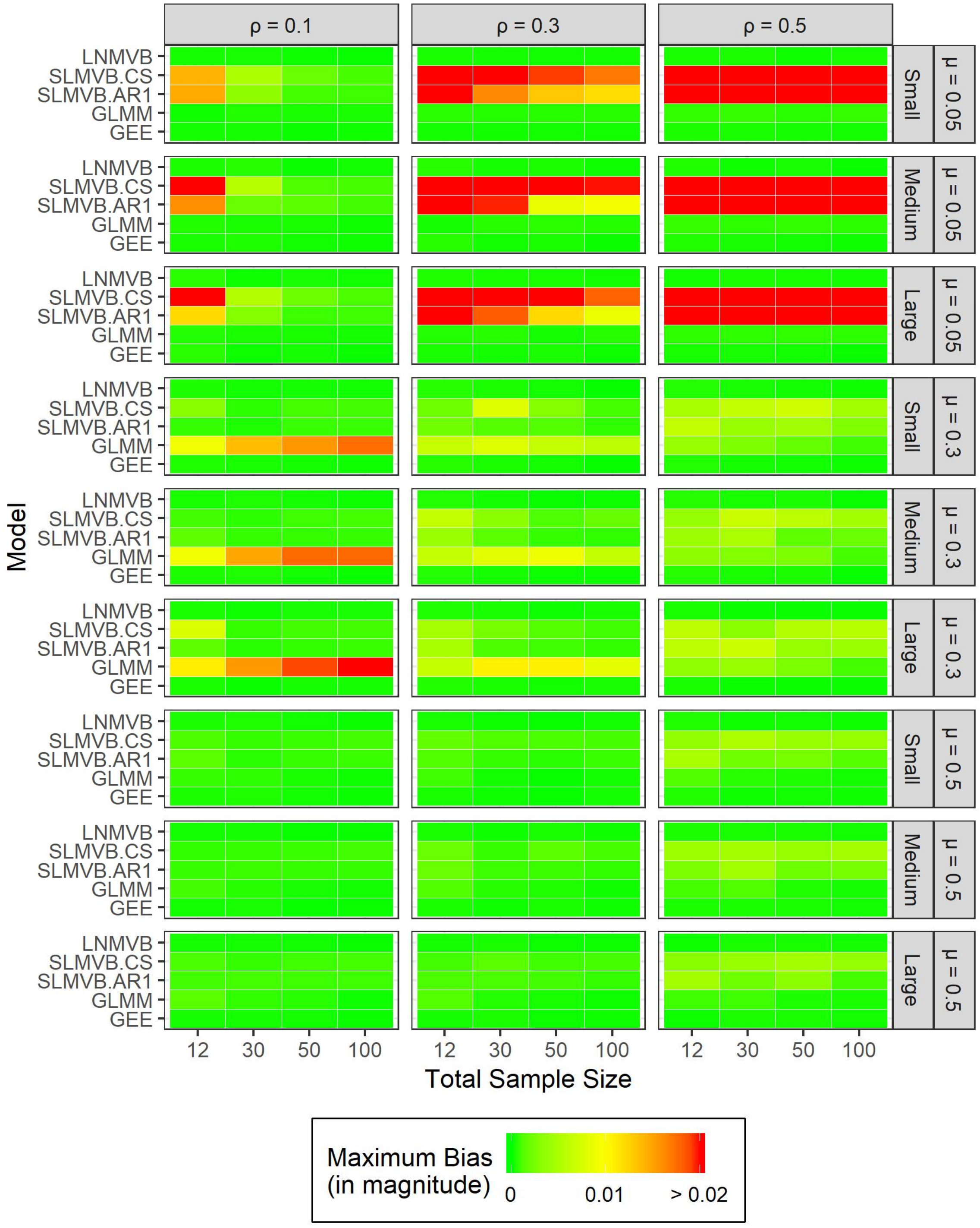


Figure B.39 Summary of bias of location parameter for two group simulations using 1000 replicates with AR(1) correlation structure.

Maximum bias is the bias that is furthest from zero of the 4 repeated-measures. Small, medium, and large represent the effect sizes.

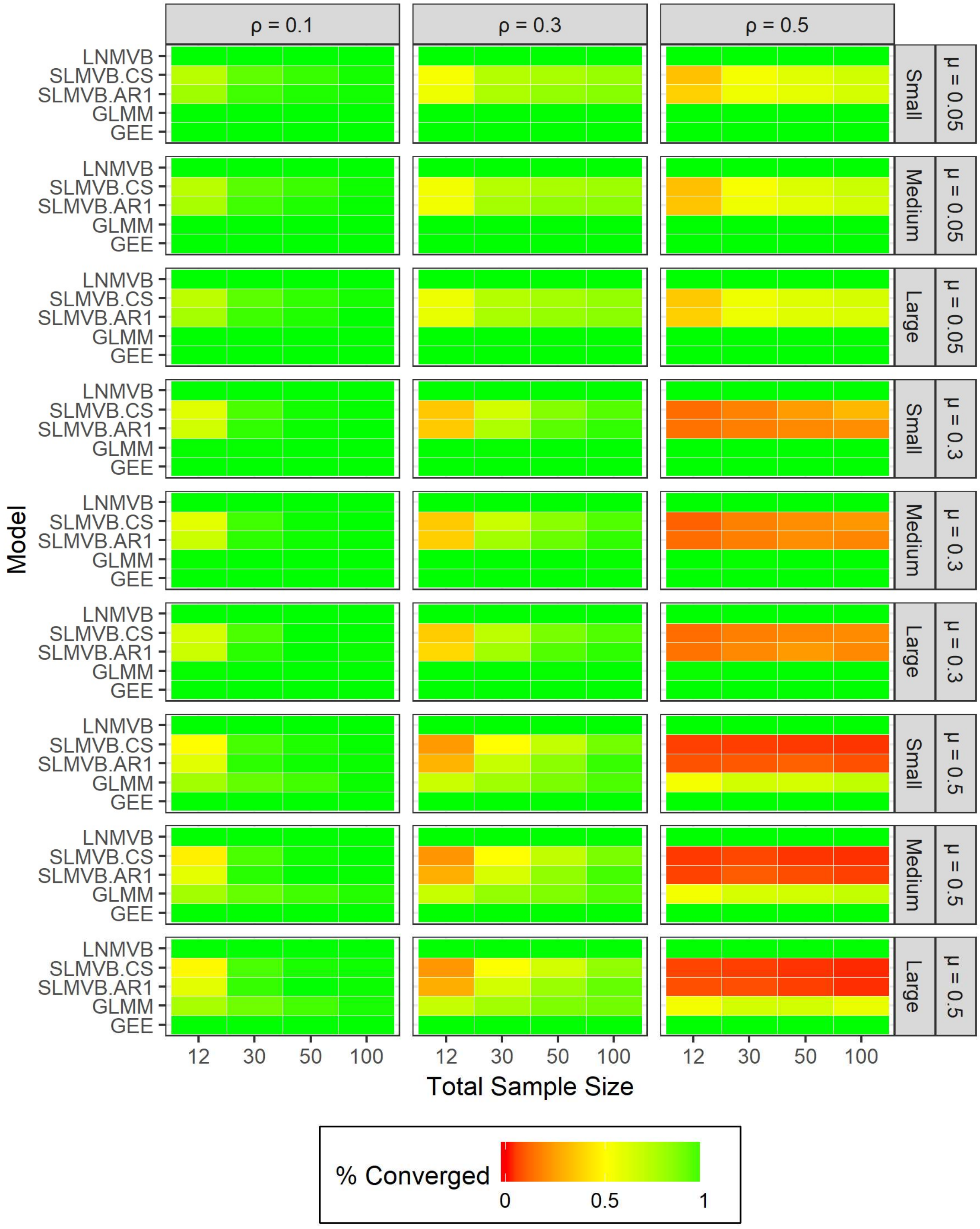


Figure B.40 Summary of model convergence for two group simulations using 1000 replicates with AR(1) correlation structure.

Appendix C: NNTC Results, SLMVB AR(1) Model

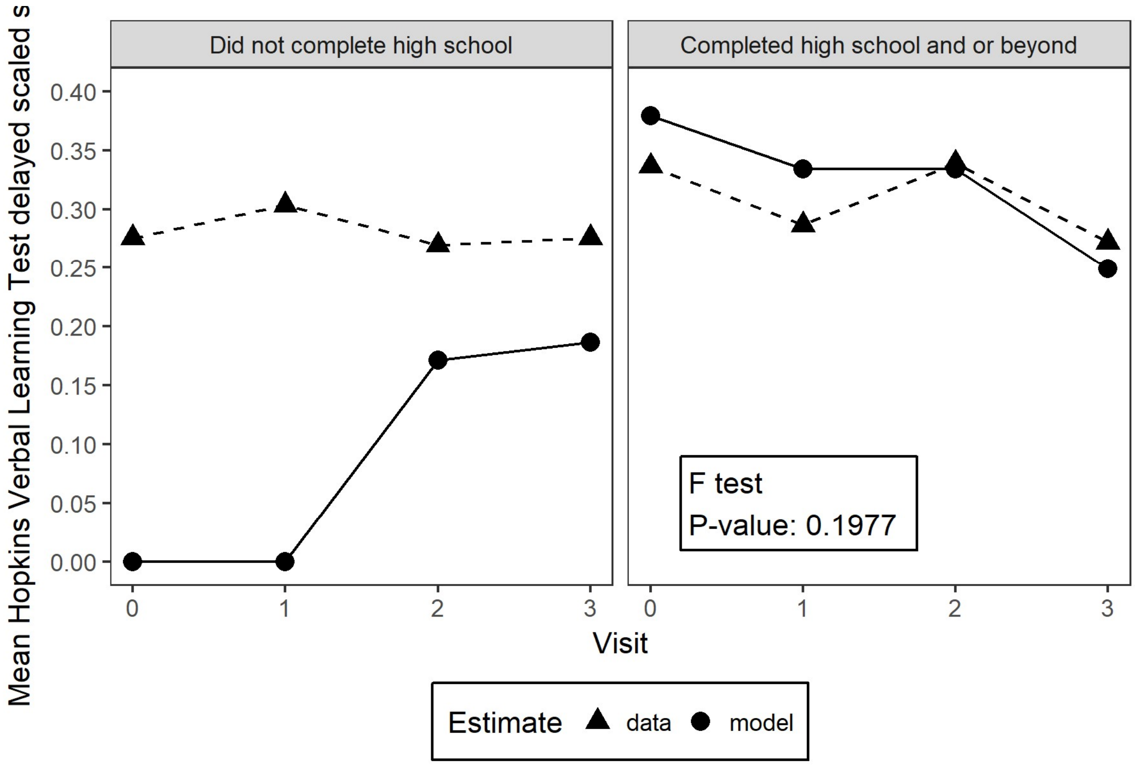


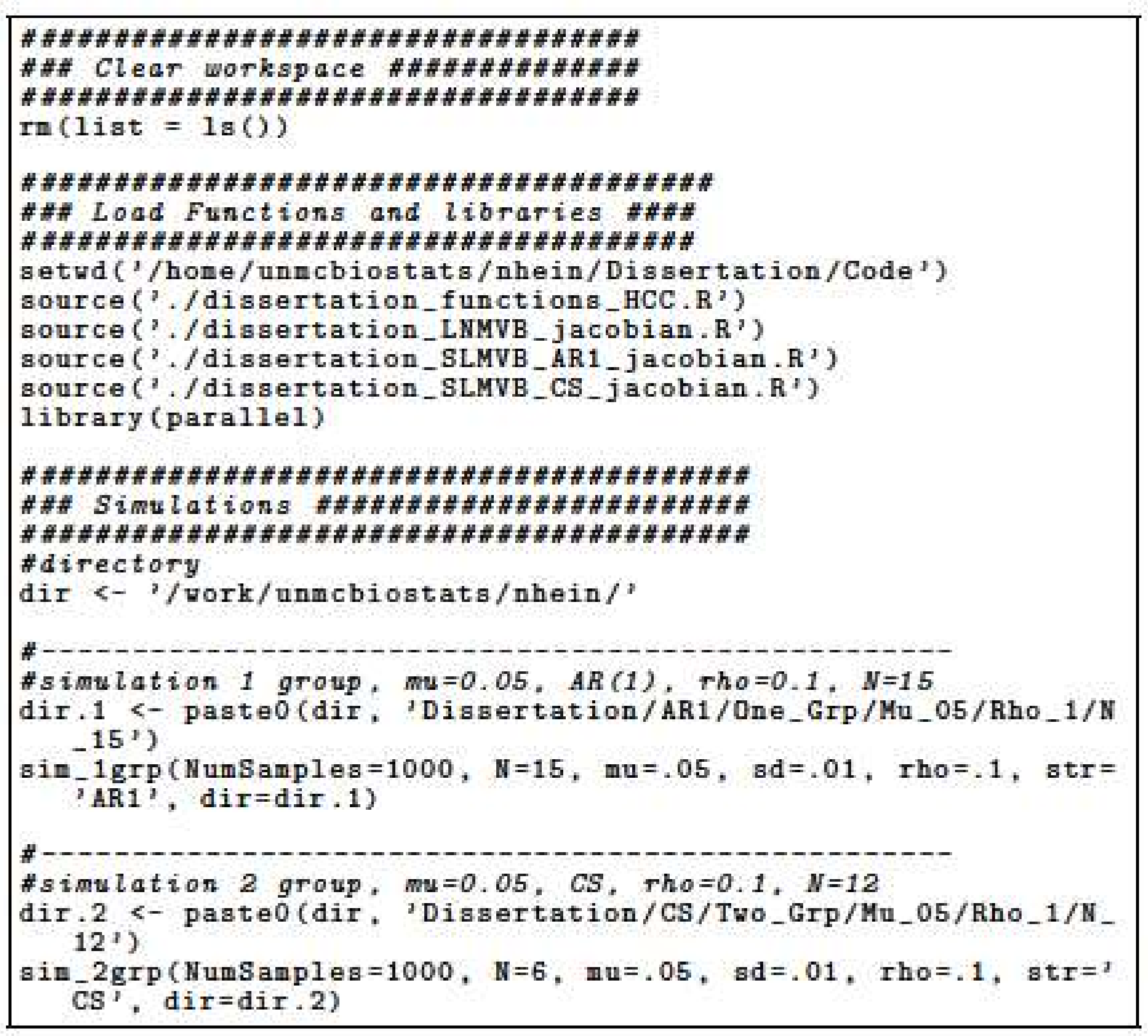
Figure C.1 Mean profiles plots of NNTC data using SLMVB AR(1) model.

Table C.1 SLMVB AR(1) model estimates of the NNTC data.

		Estimate	SE	95% C.I.	P-Value
SLMVB AR(1)					
	Intercept	-345.609	--	(--, --)	--
	Visit 1	164.953	--	(--, --)	--
	Visit 2	344.033	--	(--, --)	--
	Visit 3	344.138	--	(--, --)	--
	Educ	345.116	--	(--, --)	--
	Visit 1*Educ	-165.150	--	(--, --)	--
	Visit 2*Educ	-344.231	--	(--, --)	--
	Visit 3*Educ	-344.748	--	(--, --)	--
	Correlation	0.712			

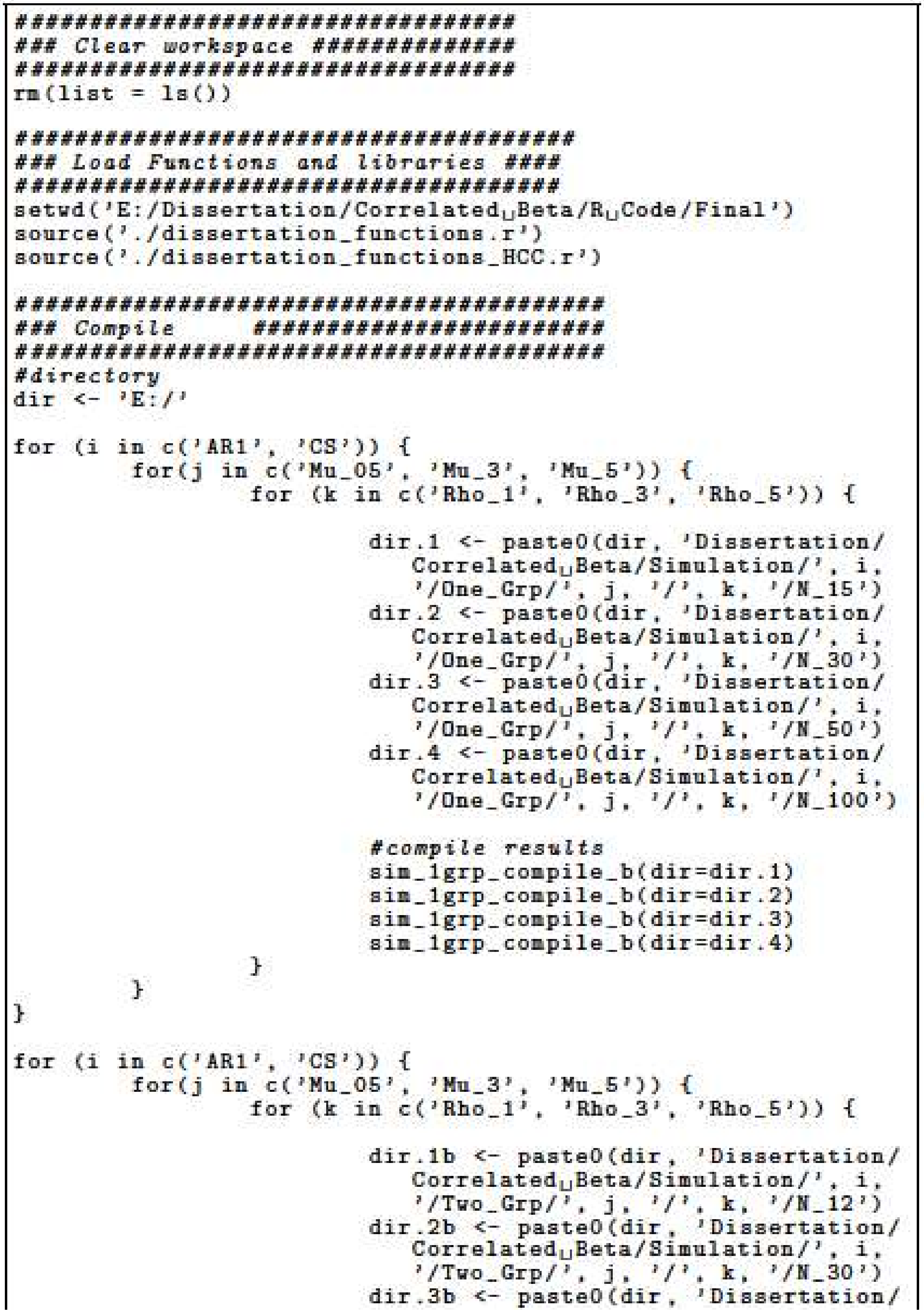
Appendix D: R Code

Below is a sample of the R code that implements the simulation. A one group (AR(1) correlation structure) and two group (CS correlation structure) simulation example was provided. The entire simulation can be performed by simulating all combinations of Table 3.1.



```
##################################
### Clear workspace ##############
##################################
rm(list = ls())

######################################
### Load Functions and libraries ####
#####################################
setwd('/home/unmcbiostats/nhein/Dissertation/Code')
source('./dissertation_functions_HCC.R')
source('./dissertation_LNMVB_jacobian.R')
source('./dissertation_SLMVB_AR1_jacobian.R')
source('./dissertation_SLMVB_CS_jacobian.R')
library(parallel)

########################################
### Simulations ########################
########################################
#directory
dir <- '/work/unmcbiostats/nhein/'

#--------------------------------------------------
#simulation 1 group, mu=0.05, AR(1), rho=0.1, N=15
dir.1 <- paste0(dir, 'Dissertation/AR1/One_Grp/Mu_05/Rho_1/N
   _15')
sim_1grp(NumSamples=1000, N=15, mu=.05, sd=.01, rho=.1, str=
   'AR1', dir=dir.1)

#--------------------------------------------------
#simulation 2 group, mu=0.05, CS, rho=0.1, N=12
dir.2 <- paste0(dir, 'Dissertation/CS/Two_Grp/Mu_05/Rho_1/N_
   12')
sim_2grp(NumSamples=1000, N=6, mu=.05, sd=.01, rho=.1, str='
   CS', dir=dir.2)
```

The following is the R code that implements the compiling of results.



```
####################################
### Clear workspace ##############
####################################
rm(list = ls())

########################################
### Load Functions and libraries ####
#######################################
setwd('E:/Dissertation/Correlated␣Beta/R␣Code/Final')
source('./dissertation_functions.r')
source('./dissertation_functions_HCC.r')

#########################################
### Compile     #######################
#########################################
#directory
dir <- 'E:/'

for (i in c('AR1', 'CS')) {
        for(j in c('Mu_05', 'Mu_3', 'Mu_5')) {
                for (k in c('Rho_1', 'Rho_3', 'Rho_5')) {

                        dir.1 <- paste0(dir, 'Dissertation/
                           Correlated␣Beta/Simulation/', i,
                           '/One_Grp/', j, '/', k, '/N_15')
                        dir.2 <- paste0(dir, 'Dissertation/
                           Correlated␣Beta/Simulation/', i,
                           '/One_Grp/', j, '/', k, '/N_30')
                        dir.3 <- paste0(dir, 'Dissertation/
                           Correlated␣Beta/Simulation/', i,
                           '/One_Grp/', j, '/', k, '/N_50')
                        dir.4 <- paste0(dir, 'Dissertation/
                           Correlated␣Beta/Simulation/', i,
                           '/One_Grp/', j, '/', k, '/N_100')

                        #compile results
                        sim_1grp_compile_b(dir=dir.1)
                        sim_1grp_compile_b(dir=dir.2)
                        sim_1grp_compile_b(dir=dir.3)
                        sim_1grp_compile_b(dir=dir.4)
                }
        }
}

for (i in c('AR1', 'CS')) {
        for(j in c('Mu_05', 'Mu_3', 'Mu_5')) {
                for (k in c('Rho_1', 'Rho_3', 'Rho_5')) {

                        dir.1b <- paste0(dir, 'Dissertation/
                           Correlated␣Beta/Simulation/', i,
                           '/Two_Grp/', j, '/', k, '/N_12')
                        dir.2b <- paste0(dir, 'Dissertation/
                           Correlated␣Beta/Simulation/', i,
                           '/Two_Grp/', j, '/', k, '/N_30')
                        dir.3b <- paste0(dir, 'Dissertation/
                                   Correlated␣Beta/Simulation/', i,
                                   '/Two_Grp/', j, '/', k, '/N_50')
                                dir.4b <- paste0(dir, 'Dissertation/
                                   Correlated␣Beta/Simulation/', i,
                                   '/Two_Grp/', j, '/', k, '/N_100')

                                #compile results
                                sim_2grp_compile_b(dir=dir.1b)
                                sim_2grp_compile_b(dir=dir.2b)
                                sim_2grp_compile_b(dir=dir.3b)
                                sim_2grp_compile_b(dir=dir.4b)
                      }
             }
}
```

```
##################################
### Clear workspace ##############
##################################
rm(list = ls())

######################################
### Load Functions and libraries ####
#####################################
setwd('E:/Dissertation/Correlated␣Beta/R␣Code/Dissertation')
source('./dissertation_functions.r')
source('./dissertation_functions_HCC.r')

########################################
### Compile     ########################
########################################
#directory
dir <- 'E:/'

for (i in c('AR1', 'CS')) {
        for(j in c('Mu_05', 'Mu_3', 'Mu_5')) {
                for (k in c('Rho_1', 'Rho_3', 'Rho_5')) {

                        dir.1 <- paste0(dir, 'Dissertation/
                           Correlated␣Beta/Simulation/', i,
                           '/One_Grp/', j, '/', k)

                        #compile results
                        sim_1grp_compile_aggregate(dir=dir
                           .1)

                }
        }
}

for (i in c('AR1', 'CS')) {
        for(j in c('Mu_05', 'Mu_3', 'Mu_5')) {
                for (k in c('Rho_1', 'Rho_3', 'Rho_5')) {

                        dir.1b <- paste0(dir, 'Dissertation/
                           Correlated␣Beta/Simulation/', i,
                           '/Two_Grp/', j, '/', k)

                        #compile results
                        sim_2grp_compile_aggregate(dir=dir.1
                           b)

                }
        }
}
```

The following produces the summary plots, i.e., heat maps

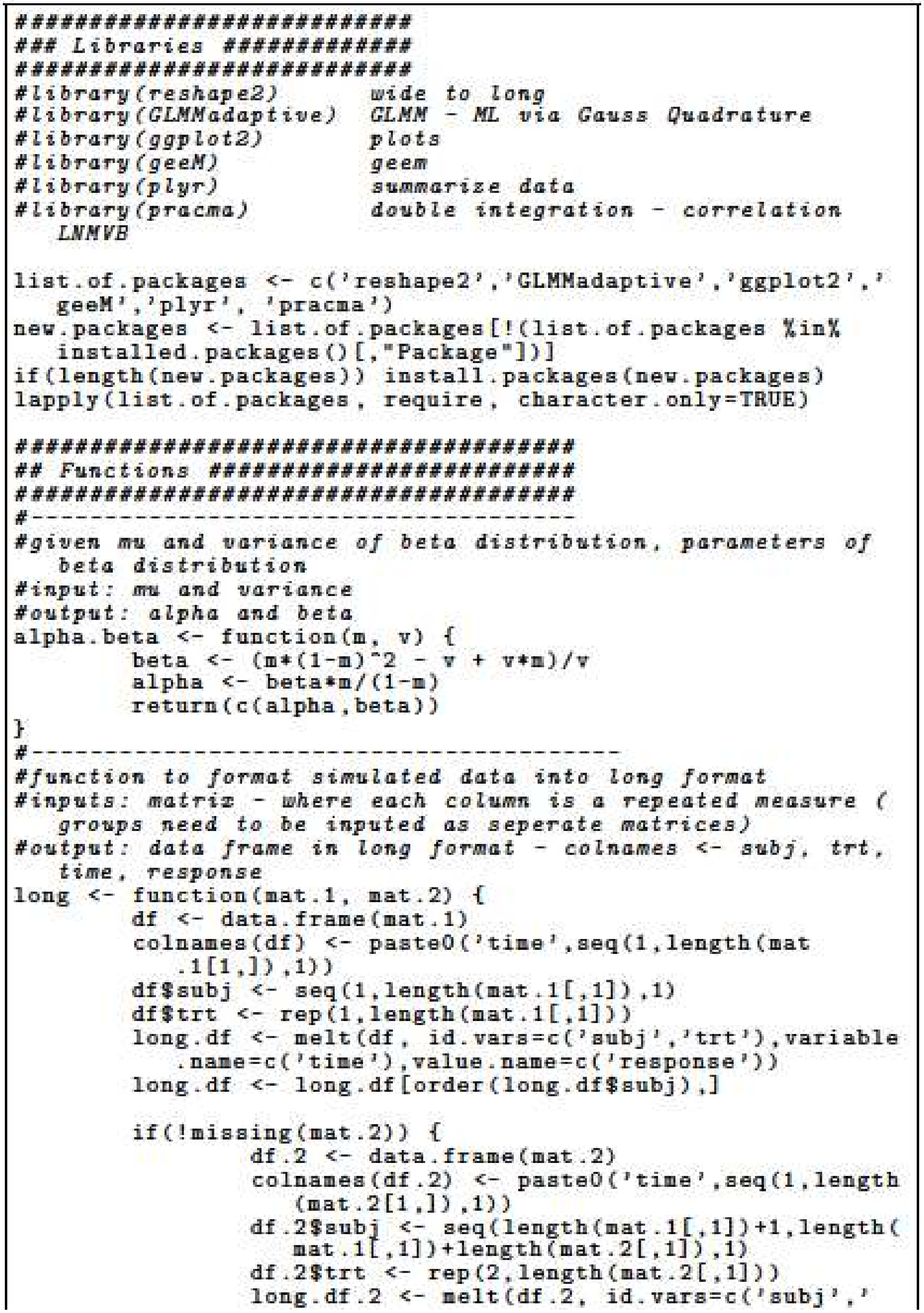
```
rm(list=ls())

library(xlsx)
library(ggplot2)
library(reshape2)

setwd('E:/Dissertation/Correlated␣Beta/Simulation')
save.loc <- 'c:/users/nicholas.hein/desktop'

mu <- c('Mu_05', 'Mu_3', 'Mu_5')
rho <- c('Rho_1', 'Rho_3', 'Rho_5')
n <- c('N_15', 'N_30', 'N_50', 'N_100')
n.2 <- c('N_12', 'N_30', 'N_50', 'N_100')

################################
### One Group #################
################################

#-----------------------------------
#Type I error
power <- data.frame()
for (w in c('AR1', 'CS')) {
        for (i in mu) {
                for (j in rho) {
                        for (k in n) {
                                tmp <- read.xlsx(paste0('./'
                                   , w, '/One_Grp/', i, '/',
                                    j, '/', k, '/results.
                                   xlsx'), 1)
                                dat <- data.frame(t(tmp
                                   [1,2:6]))
                                colnames(dat) <- 'power'
                                dat$model <- c('LNMVB', '
                                   SLMVB.CS', 'SLMVB.AR1', '
                                   GEE', 'GLMM')
                                dat$mu <- i
                                dat$rho <- j
                                dat$N <- k
                                dat$structure <- w
                                power <- rbind(power, dat)
                        }
                }
        }
}
power$model <- factor(power$model, levels=c('GEE', 'GLMM', '
   SLMVB.AR1', 'SLMVB.CS', 'LNMVB'))
power$N <- factor(power$N, levels=n, labels=c('15', '30', '
   50', '100'))
power$mu <- factor(power$mu, levels=mu, labels=c('\U03BC␣=␣
   0.05', '\U03BC␣=␣0.3', '\U03BC␣=␣0.5'))
power$rho <- factor(power$rho, levels=rho, labels=c('\U03C1␣
   =␣0.1', '\U03C1␣=␣0.3', '\U03C1␣=␣0.5'))
power.ar1 <- subset(power, structure=='AR1')
power.plot.ar1 <- ggplot(power.ar1, aes(N, model)) + geom_
   tile(aes(fill=power), color='white') + facet_grid(mu~rho)
```

```
    +
        scale_fill_gradient2(limits=c(0,.1), breaks=seq
           (0,.1, .05), labels=c('0', '0.05', '>␣0.1'),
                low='yellow', mid='green', high='red',
                   midpoint=0.05, na.value='red') +
                labs(x='Total␣Sample␣Size', y='Model', fill=
                   'Type␣I␣Error') +
        theme_bw() + theme(legend.position='bottom', legend.
           box.background=element_rect(color='black'),
                legend.box.margin=margin(1,10,1,1))
ggsave(filename=paste0(save.loc, '/power_ar1.png'), power.
   plot.ar1, width=6, height=4, units='in', dpi=300)

#-------------------------------------
#Coverage
coverage <- data.frame()
for (w in c('AR1', 'CS')) {
        for (i in mu) {
                for (j in rho) {
                        dat <- read.xlsx(paste0('./', w, '/
                           One_Grp/', i, '/', j,'/results.
                           xlsx'), 1)
                        dat$coverage <- apply(dat[,paste0('
                           time',1:4)], 1, mean)
                        dat$mu <- i
                        dat$rho <- j
                        dat$structure <- w
                        coverage <- rbind(coverage, dat)
                }
        }
}
coverage$method <- factor(coverage$method, levels=c('GEE', '
   GLMM', 'SLMVB.AR1', 'SLMVB.CS', 'LNMVB'))
coverage$effect.size <- factor(coverage$effect.size, levels=
   c(0.2, 0.5, 0.8), labels=c('Small', 'Medium', 'Large'))
coverage$mu <- factor(coverage$mu, levels=mu, labels=c('\
   U03BC␣=␣0.05', '\U03BC␣=␣0.3', '\U03BC␣=␣0.5'))
coverage$sample.size <- factor(coverage$sample.size, levels=
   n, labels=c('15', '30', '50', '100'))
coverage$rho <- factor(coverage$rho, levels=rho, labels=c('\
   U03C1␣=␣0.1', '\U03C1␣=␣0.3', '\U03C1␣=␣0.5'))
coverage.ar1 <- subset(coverage, structure=='AR1')
coverage.plot.ar1 <- ggplot(coverage.ar1, aes(sample.size,
   method)) + geom_tile(aes(fill=coverage), color='white') +
    facet_grid(mu+effect.size~rho) +
        scale_fill_gradient2(limits=c(.9,1), breaks=seq
           (.9,1,.05), labels=c('<␣0.9', '0.95', '1'),
                low='red', mid='green', high='yellow',
                   midpoint=0.95, na.value='red') +
        labs(x='Total␣Sample␣Size', y='Model', fill='Average
           ␣Coverage␣Probability') +
        theme_bw() + theme(legend.position='bottom', legend.
           box.background=element_rect(color='black'),
                legend.box.margin=margin(1,10,1,1))
ggsave(filename=paste0(save.loc, '/coverage_ar1.png'),
  coverage.plot.ar1, width=6, height=7.5, units='in', dpi
  =300)

#-----------------------------------
#bias
bias <- data.frame()
for (w in c('AR1', 'CS')) {
      for (i in mu) {
            for (j in rho) {
                  tmp <- read.xlsx(paste0('./', w, '/
                    One_Grp/', i, '/', j,'/results.
                    xlsx'), 2)
                  dat <- dcast(tmp, effect.size+method
                    +sample.size~time, value.var='
                    bias')
                  dat$bias <- apply(dat[,paste0('time'
                    ,1:4)], 1, function(x) x[which(
                    abs(x)==max(abs(x)))])
                  dat$mu <- i
                  dat$rho <- j
                  dat$structure <- w
                  bias <- rbind(bias, dat)
            }
      }
}
bias$method <- factor(bias$method, levels=c('GEE', 'GLMM', '
  SLMVB.AR1', 'SLMVB.CS', 'LNMVB'))
bias$effect.size <- factor(bias$effect.size, levels=c('Small
  ␣effect␣size', 'Medium␣effect␣size', 'Large␣effect␣size')
  , labels=c('Small', 'Medium', 'Large'))
bias$mu <- factor(bias$mu, levels=mu, labels=c('\U03BC␣=␣
  0.05', '\U03BC␣=␣0.3', '\U03BC␣=␣0.5'))
bias$sample.size <- factor(bias$sample.size, levels=n,
  labels=c('15', '30', '50', '100'))
bias$rho <- factor(bias$rho, levels=rho, labels=c('\U03C1␣=␣
  0.1', '\U03C1␣=␣0.3', '\U03C1␣=␣0.5'))
bias.ar1 <- subset(bias, structure=='AR1')
bias.plot.ar1 <- ggplot(bias.ar1, aes(sample.size, method))
  + geom_tile(aes(fill=bias), color='white') + facet_grid(
  mu+effect.size~rho) +
      scale_fill_gradient2(limits=c(-.02,.02), breaks=seq
        (-.02,.02,.02), labels=c('<␣-0.02', '0', '>␣0.02'
        ),
            low='yellow', mid='green', high='red',
              midpoint=0, na.value='red') +
      labs(x='Total␣Sample␣Size', y='Model', fill='Maximum
        ␣Bias') +
      theme_bw() + theme(legend.position='bottom', legend.
        box.background=element_rect(color='black'),
            legend.box.margin=margin(1,12,1,1))

ggsave(filename=paste0(save.loc, '/bias_ar1.png'), bias.plot
  .ar1, width=6, height=7.5, units='in', dpi=300)

#-----------------------------------
```

```
#Convergence
converge <- data.frame()
for (w in c('AR1', 'CS')) {
        for (i in mu) {
                for (j in rho) {
                        dat <- read.xlsx(paste0('./', w, '/
                           One_Grp/', i, '/', j,'/results.
                           xlsx'), 1)
                        #dat <- dcast(tmp, effect.size+
                           method+sample.size~time, value.
                           var='CONV')
                        #dat$converge <- apply(dat[,paste0('
                           time',1:4)], 1, function(x) x[
                           which(abs(x)==max(abs(x)))])
                        dat$mu <- i
                        dat$rho <- j
                        dat$structure <- w
                        converge <- rbind(converge, dat)
                }
        }
}
converge$method <- factor(converge$method, levels=c('GEE', '
   GLMM', 'SLMVB.AR1', 'SLMVB.CS', 'LNMVB'))
converge$effect.size <- factor(converge$effect.size, levels=
   c(0.2, 0.5, 0.8), labels=c('Small', 'Medium', 'Large'))
converge$mu <- factor(converge$mu, levels=mu, labels=c('\
   U03BC␣=␣0.05', '\U03BC␣=␣0.3', '\U03BC␣=␣0.5'))
converge$sample.size <- factor(converge$sample.size, levels=
   n, labels=c('15', '30', '50', '100'))
converge$rho <- factor(converge$rho, levels=rho, labels=c('\
   U03C1␣=␣0.1', '\U03C1␣=␣0.3', '\U03C1␣=␣0.5'))
converge.ar1 <- subset(converge, structure='AR1')
converge.plot <- ggplot(converge.ar1, aes(sample.size,
   method)) + geom_tile(aes(fill=CONV), color='white') +
   facet_grid(mu+effect.size~rho) +
        scale_fill_gradient2(limits=c(0,1), breaks=c(0,.5,1)
           , labels=c('0', '0.5', '1'),
                low='red', mid='yellow', high='green',
                   midpoint=0.5, na.value='red') +
        labs(x='Total␣Sample␣Size', y='Model', fill='%␣
           Converged') +
        theme_bw() + theme(legend.position='bottom', legend.
           box.background=element_rect(color='black'),
                legend.box.margin=margin(1,12,1,1))
ggsave(filename=paste0(save.loc, '/converge.png'), converge.
   plot, width=6, height=7.5, units='in', dpi=300)

#######################################
### Two Group #####################
######################################

#----------------------------------
#Type I error
power.2 <- data.frame()
for (w in c('AR1', 'CS')) {
        for (i in mu) {
                for (j in rho) {
                        for (k in n.2) {
                                tmp <- read.xlsx(paste0('./'
                                   , w, '/Two_Grp/', i, '/',
                                    j, '/', k, '/results.
                                   xlsx'), 1)
                                dat <- data.frame(t(tmp
                                   [1,2:6]))
                                colnames(dat) <- 'power'
                                dat$model <- c('LNMVB', '
                                   SLMVB.CS', 'SLMVB.AR1', '
                                   GEE', 'GLMM')
                                dat$mu <- i
                                dat$rho <- j
                                dat$N <- k
                                dat$structure <- w
                                power.2 <- rbind(power.2,
                                   dat)
                        }
                }
        }
}
power.2$model <- factor(power.2$model, levels=c('GEE', 'GLMM
   ', 'SLMVB.AR1', 'SLMVB.CS', 'LNMVB'))
power.2$N <- factor(power.2$N, levels=n.2, labels=c('12', '
   30', '50', '100'))
power.2$mu <- factor(power.2$mu, levels=mu, labels=c('\U03BC
   ␣=␣0.05', '\U03BC␣=␣0.3', '\U03BC␣=␣0.5'))
power.2$rho <- factor(power.2$rho, levels=rho, labels=c('\
   U03C1␣=␣0.1', '\U03C1␣=␣0.3', '\U03C1␣=␣0.5'))
power.2.ar1 <- subset(power.2, structure=='AR1')
power.2.plot.ar1 <- ggplot(power.2.ar1, aes(N, model)) +
   geom_tile(aes(fill=power), color='white') + facet_grid(mu
   ~rho) +
        scale_fill_gradient2(limits=c(0,.1), breaks=seq
           (0,.1, .05), labels=c('0', '0.05', '>␣0.1'),
                low='yellow', mid='green', high='red',
                   midpoint=0.05, na.value='red') +
                labs(x='Total␣Sample␣Size', y='Model', fill=
                   'Type␣I␣Error') +
        theme_bw() + theme(legend.position='bottom', legend.
           box.background=element_rect(color='black'),
                legend.box.margin=margin(1,10,1,1))
ggsave(filename=paste0(save.loc, '/power_2_ar1.png'), power
   .2.plot.ar1, width=6, height=4, units='in', dpi=300)

#-------------------------------------
#Coverage
coverage.2 <- data.frame()
for (w in c('AR1', 'CS')) {
        for (i in mu) {
                for (j in rho) {
                        dat <- read.xlsx(paste0('./', w, '/
                           Two_Grp/', i, '/', j,'/results.
                           xlsx'), 1)
                        dat$coverage <- apply(dat[,paste0('
                           time',1:4)], 1, mean)
                        dat$mu <- i
                        dat$rho <- j
                        dat$structure <- w
                        coverage.2 <- rbind(coverage.2, dat)
                }
        }
}
coverage.2$method <- factor(coverage.2$method, levels=c('GEE
   ', 'GLMM', 'SLMVB.AR1', 'SLMVB.CS', 'LNMVB'))
coverage.2$effect.size <- factor(coverage.2$effect.size,
   levels=c(0.2, 0.5, 0.8), labels=c('Small', 'Medium', '
   Large'))
coverage.2$mu <- factor(coverage.2$mu, levels=mu, labels=c('
   \U03BC␣=␣0.05', '\U03BC␣=␣0.3', '\U03BC␣=␣0.5'))
coverage.2$sample.size <- factor(coverage.2$sample.size,
   levels=n.2, labels=c('12', '30', '50', '100'))
coverage.2$rho <- factor(coverage.2$rho, levels=rho, labels=
   c('\U03C1␣=␣0.1', '\U03C1␣=␣0.3', '\U03C1␣=␣0.5'))
coverage.2.ar1 <- subset(coverage.2, structure=='AR1')
coverage.2.plot.ar1 <- ggplot(coverage.2.ar1, aes(sample.
   size, method)) + geom_tile(aes(fill=coverage), color='
   white') + facet_grid(mu+effect.size~rho) +
        scale_fill_gradient2(limits=c(.9,1), breaks=seq
           (.9,1,.05), labels=c('<␣0.9', '0.95', '1'),
                low='red', mid='green', high='yellow',
                   midpoint=0.95, na.value='red') +
        labs(x='Total␣Sample␣Size', y='Model', fill='Average
           ␣Coverage␣Probability') +
        theme_bw() + theme(legend.position='bottom', legend.
           box.background=element_rect(color='black'),
                legend.box.margin=margin(1,10,1,1))
ggsave(filename=paste0(save.loc, '/coverage_2_ar1.png'),
   coverage.2.plot.ar1, width=6, height=7.5, units='in', dpi
   =300)

#------------------------------------
#bias
bias.2 <- data.frame()
for (w in c('AR1', 'CS')) {
        for (i in mu) {
                for (j in rho) {
                        tmp <- read.xlsx(paste0('./', w, '/
                           Two_Grp/', i, '/', j,'/results.
                           xlsx'), 2)
                        dat <- dcast(tmp, effect.size+method
                           +sample.size~time, value.var='
                           bias')
                        dat$bias <- apply(dat[,paste0('time'
                           ,1:4)], 1, function(x) x[which(
                           abs(x)==max(abs(x)))])
                        dat$mu <- i
                        dat$rho <- j
                        dat$structure <- w
                        bias.2 <- rbind(bias.2, dat)
                }
        }
}
bias.2$method <- factor(bias.2$method, levels=c('GEE', 'GLMM
   ', 'SLMVB.AR1', 'SLMVB.CS', 'LNMVB'))
bias.2$effect.size <- factor(bias.2$effect.size, levels=c('
   Small␣effect␣size', 'Medium␣effect␣size', 'Large␣effect␣
   size'), labels=c('Small', 'Medium', 'Large'))
bias.2$mu <- factor(bias.2$mu, levels=mu, labels=c('\U03BC␣=
   ␣0.05', '\U03BC␣=␣0.3', '\U03BC␣=␣0.5'))
bias.2$sample.size <- factor(bias.2$sample.size, levels=n.2,
    labels=c('12', '30', '50', '100'))
bias.2$rho <- factor(bias.2$rho, levels=rho, labels=c('\
   U03C1␣=␣0.1', '\U03C1␣=␣0.3', '\U03C1␣=␣0.5'))
bias.2.ar1 <- subset(bias.2, structure=='AR1')
bias.2.cs <- subset(bias.2, structure=='CS')
bias.2.plot.ar1 <- ggplot(bias.2.ar1, aes(sample.size,
   method)) + geom_tile(aes(fill=bias), color='white') +
   facet_grid(mu+effect.size~rho) +
        scale_fill_gradient2(limits=c(-.02,.02), breaks=seq
           (-.02,.02,.02), labels=c('<␣-0.02', '0', '>␣0.02'
           ),
                low='yellow', mid='green', high='red',
                   midpoint=0, na.value='red') +
        labs(x='Total␣Sample␣Size', y='Model', fill='Maximum
           ␣Bias') +
        theme_bw() + theme(legend.position='bottom', legend.
           box.background=element_rect(color='black'),
                legend.box.margin=margin(1,12,1,1))
ggsave(filename=paste0(save.loc, '/bias_2_ar1.png'), bias.2.
   plot.ar1, width=6, height=7.5, units='in', dpi=300)

#-----------------------------------
#Convergence
converge.2 <- data.frame()
for (w in c('AR1', 'CS')) {
        for (i in mu) {
                for (j in rho) {
                        dat <- read.xlsx(paste0('./', w, '/
                           Two_Grp/', i, '/', j,'/results.
                           xlsx'), 1)
                        #dat <- dcast(tmp, effect.size+
                           method+sample.size~time, value.
                           var='CONV')
                        #dat$converge <- apply(dat[,paste0('
                           time',1:4)], 1, function(x) x[
                           which(abs(x)==max(abs(x)))])
                        dat$mu <- i
                        dat$rho <- j
                        dat$structure <- w
                        converge.2 <- rbind(converge.2, dat)
                }
        }
```

```
}
converge.2$method <- factor(converge.2$method, levels=c('GEE
   ', 'GLMM', 'SLMVB.AR1', 'SLMVB.CS', 'LNMVB'))
converge.2$effect.size <- factor(converge.2$effect.size,
   levels=c(0.2, 0.5, 0.8), labels=c('Small', 'Medium', '
   Large'))
converge.2$mu <- factor(converge.2$mu, levels=mu, labels=c('
   \U03BC␣=␣0.05', '\U03BC␣=␣0.3', '\U03BC␣=␣0.5'))
converge.2$sample.size <- factor(converge.2$sample.size,
   levels=n.2, labels=c('12', '30', '50', '100'))
converge.2$rho <- factor(converge.2$rho, levels=rho, labels=
   c('\U03C1␣=␣0.1', '\U03C1␣=␣0.3', '\U03C1␣=␣0.5'))
converge.2.ar1 <- subset(converge.2, structure='AR1')
converge.plot <- ggplot(converge.2.ar1, aes(sample.size,
   method)) + geom_tile(aes(fill=CONV), color='white') +
   facet_grid(mu+effect.size~rho) +
         scale_fill_gradient2(limits=c(0,1), breaks=c(0,.5,1)
            , labels=c('0', '0.5', '1'),
                  low='red', mid='yellow', high='green',
                     midpoint=0.5, na.value='red') +
         labs(x='Total␣Sample␣Size', y='Model', fill='%␣
            Converged') +
         theme_bw() + theme(legend.position='bottom', legend.
            box.background=element_rect(color='black'),
                  legend.box.margin=margin(1,12,1,1))
ggsave(filename=paste0(save.loc, '/converge2.png'), converge
   .plot, width=6, height=7.5, units='in', dpi=300)
```

disseration_functions_HCC.R



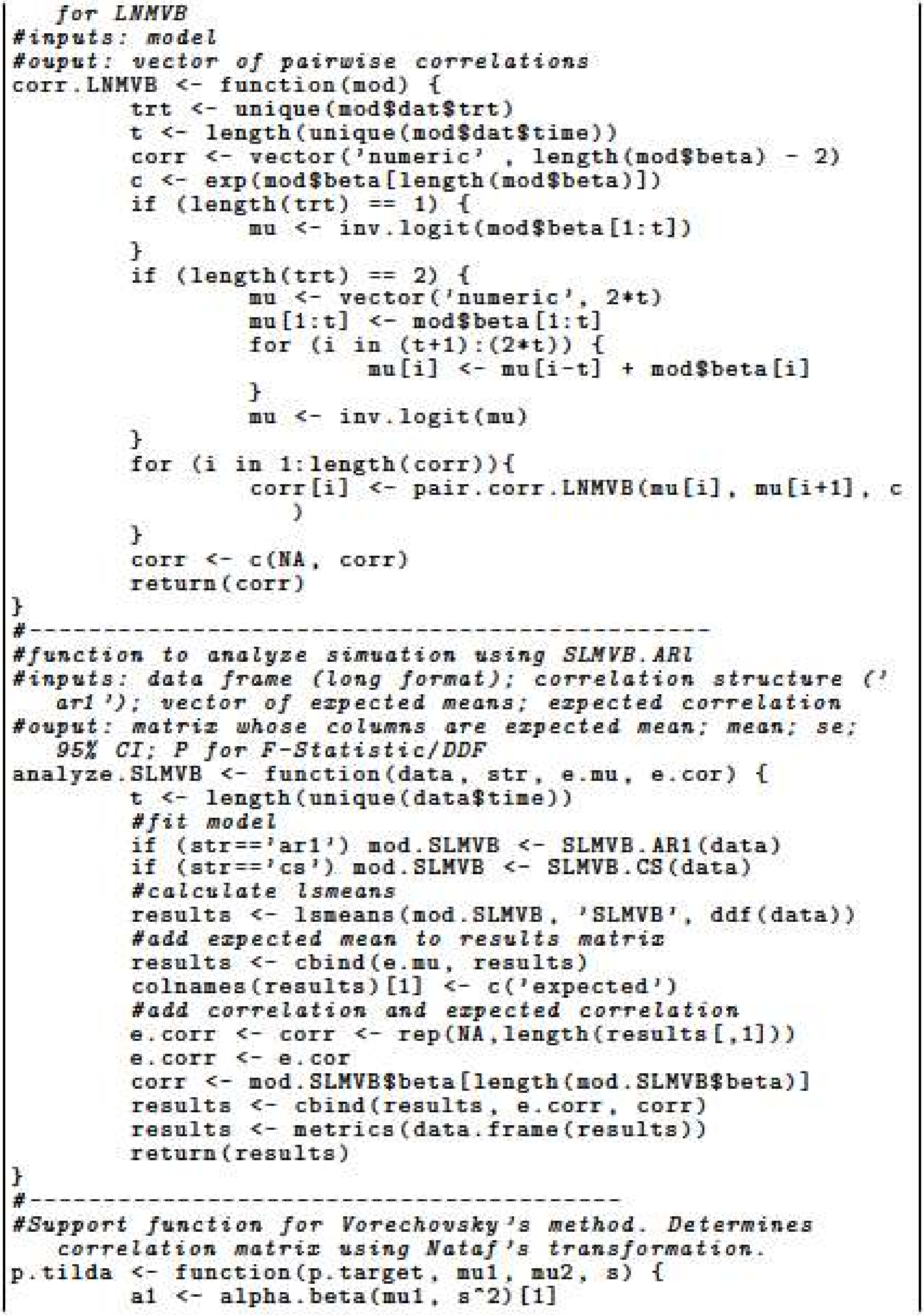
```
############################
### Libraries #############
############################
#library(reshape2)      wide to long
#library(GLMMadaptive)  GLMM - ML via Gauss Quadrature
#library(ggplot2)       plots
#library(geeM)          geem
#library(plyr)          summarize data
#library(pracma)        double integration - correlation
   LNMVB

list.of.packages <- c('reshape2','GLMMadaptive','ggplot2','
   geeM','plyr', 'pracma')
new.packages <- list.of.packages[!(list.of.packages %in%
   installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
lapply(list.of.packages, require, character.only=TRUE)

#######################################
## Functions #########################
#######################################
#-------------------------------------
#given mu and variance of beta distribution, parameters of
   beta distribution
#input: mu and variance
#output: alpha and beta
alpha.beta <- function(m, v) {
        beta <- (m*(1-m)^2 - v + v*m)/v
        alpha <- beta*m/(1-m)
        return(c(alpha,beta))
}
#----------------------------------------
#function to format simulated data into long format
#inputs: matrix - where each column is a repeated measure (
   groups need to be inputed as seperate matrices)
#output: data frame in long format - colnames <- subj, trt,
   time, response
long <- function(mat.1, mat.2) {
        df <- data.frame(mat.1)
        colnames(df) <- paste0('time',seq(1,length(mat
           .1[1,]),1))
        df$subj <- seq(1,length(mat.1[,1]),1)
        df$trt <- rep(1,length(mat.1[,1]))
        long.df <- melt(df, id.vars=c('subj','trt'),variable
           .name=c('time'),value.name=c('response'))
        long.df <- long.df[order(long.df$subj),]

        if(!missing(mat.2)) {
                df.2 <- data.frame(mat.2)
                colnames(df.2) <- paste0('time',seq(1,length
                   (mat.2[1,]),1))
                df.2$subj <- seq(length(mat.1[,1])+1,length(
                   mat.1[,1])+length(mat.2[,1]),1)
                df.2$trt <- rep(2,length(mat.2[,1]))
                long.df.2 <- melt(df.2, id.vars=c('subj','
                   trt'),variable.name=c('time'),value.name=
                   c('response'))
                long.df.2 <- long.df.2[order(long.df.2$subj)
                   ,]
                long.df <- rbind(long.df,long.df.2)
        }

        long.df$trt <- factor(long.df$trt)

        return(long.df)
}
#---------------------------------------------
#inverse logit function
inv.logit <- function(x) { exp(x)/(1+exp(x)) }

#---------------------------------------------
#delta method for inverse logit
#inputs:standard deviaton on logit scale
#output: standard deviation on the original scale
delta.logit <- function(theta, sd) {
        (exp(theta)/(1+exp(theta))^2)*sd
}
#-----------------------------------------------
#denominator degrees of freedom for treatment groups use
   between method - SAS documenation glimmix/Fitzmaurice
#inputs: data.frame - subj, response, time, trt
#output: denominator degrees of freedom for f-statistics
ddf <- function(data) {
        if(is.null(data$trt)) {
                N <- length(unique(data$subj))
                t <- length(unique(data$time))
                ddf <- (N-1) * (t-1)
                adj.df <- sapply(by(data$subj, data$subj,
                   function(w) length(w)),function(z) z)
                ddf <- ddf - sum(max(adj.df)-adj.df)
                return(ddf)
        } else {
                ddf.temp <- ddply(data, .(trt), summarize,
                        t=max(as.numeric((time))),
                        N=length(unique(subj))
                )
                ddf.final <- apply(ddf.temp, 1, function(x)
                    (as.numeric(x[3]) - 1)*(as.numeric(x[2])
                    - 1) )
                ddf.adj <- vector('numeric', length(ddf.temp
                   [,1]))
                for (i in 1:length(ddf.temp[,1])) {
                        temp.adj <- ddply(data[which(data$
                           trt==i),], .(trt, subj),
                           summarize,
                                N=length(unique(time))
                        )
                        ddf.adj[i] <- sum(ddf.temp$t[i] -
                           temp.adj$N)
                }
                return(sum(ddf.final - ddf.adj))
        }
}
#--------------------------------------------------
#function to calculate 95% confidence intervals based on t-
   distribution
#inputs: point estimate; standard error; degrees of freedom
#output: 2x1 vector with first entry lower.ci and 2nd entry
   upper.ci
t.confidence <- function(mean, se, df) {
        lower <- mean - se*qt(.975, df)
        upper <- mean + se*qt(.975,df)
        return(c(lower,upper))
}
#-------------------------------------------------
#family definition for Beta GLMM using package GLMMadaptive
beta.glmm <- function(link='logit') {
        stats <- make.link(link)
        log_dens <- function (y, eta, mu_fun, phis, eta_zi)
           {
                phi <- exp(phis)
                mu <- mu_fun(eta)
                comp.1 <- lgamma(phi) - lgamma(mu*phi) -
                   lgamma((1-mu)*phi)
                comp.2 <- (mu*phi-1)*log(y) + ((1-mu)*phi-1)
                   *log(1-y)
                out <- comp.1 + comp.2
                attr(out, "mu_y") <- mu
        out
        }
        structure(list(family = "Beta", link = stats$name,
           linkfun = stats$linkfun,
        linkinv = stats$linkinv, log_dens = log_dens),
                class = "family")
}
#-----------------------------------------------------
#function for lsmeans
#inputs:
#outputs:
lsmeans <- function(mod, type, den.df) {

        if (!is.list(mod)) {
                if (mod==1) {
                        results <- matrix(NA, nrow=4, ncol
                           =5)
                        colnames(results) <- c('mean','se','
                           lower.ci','upper.ci','P.DDF.CONV'
                           )
                        return(results)
                } else {
                        results <- matrix(NA, nrow=8, ncol
                           =5)
                        colnames(results) <- c('mean','se','
                           lower.ci','upper.ci','P.DDF.CONV'
                           )
                return(results)
            }
    }

    switch(type,
            glmm = {
                    #coefficients
                    b.hat <- mod$coefficients
                    #num repeated measures
                    t <- length(unique(mod$model_frames$
                        mfX$time))
                    #Hessian matrix
                    vbeta <- tryCatch({solve(mod$Hessian
                        )},
                            error=function(e){print('
                                GLMM␣singular␣hessian');
                                matrix(NA, nrow=2*t, ncol
                                =2*t)})

                    trt.len <- length(unique(mod$model_
                        frames$mfX$trt))
                    },
            gee =
                    {
                    b.hat <- mod$beta
                    vbeta <- as.matrix(mod$var)
                    t <- length(grep('timetime[[:digit
                        :]]$',mod$coefnames)) + 1
                    trt.len <- length(grep('^trt[[:digit
                        :]]$',mod$coefnames))
            },
            LNMVB = {
                    b.hat <- mod$beta[-length(mod$beta)]
                    t <- length(unique(mod$dat$time))
                    vbeta <- tryCatch({solve(-mod$
                        hessian)},
                            error=function(e){print('
                                LNMVB␣singular␣hessian');
                                 matrix(NA, nrow=2*t,
                                ncol=2*t)})
                    trt.len <- length(unique(mod$dat$trt
                        ))
            },
            SLMVB = {
                    b.hat <- mod$beta
                    #perturbation based on dennis 1996
                    t <- length(unique(mod$dat$time))
                    vbeta <- tryCatch({solve(-mod$
                        hessian)},
                            error=function(e){print('
                                SLMVB␣singular␣hessian');
                                 matrix(NA, nrow=2*t,
                                ncol=2*t)})
                    trt.len <- length(unique(mod$dat$trt
                        ))
```

```
        }
)

if (type == 'glmm' | type=='gee') {
if (trt.len==0) {
        #var-cov matrix
        vbeta <- vbeta[1:t,1:t]
        results <- matrix(NA, nrow=t, ncol=5)
        #Calculate lsmeans, se, 95% CI
        for (i in 1:t){
                L <- rep(0,t)
                L[1] <- L[i] <- 1
                results[i,1] <- L %*% b.hat
                results[i,2] <- sqrt(L %*% vbeta %*%
                    L)
                results[i,3:4] <- t.confidence(
                   results[i,1],results[i,2],den.df)
        }
        #Convert back to original scale
        results[,2] <- delta.logit(results[,1],
           results[,2])
        results[,c(1,3:4)] <- inv.logit(results[,c
           (1,3:4)])
        #f-statistic
        L <- diag(t)
        L <- L[-1,]
        f.stat <- t(L %*% b.hat) %*% solve(L %*%
           vbeta %*% t(L)) %*% L %*% b.hat/(t-1)
        results[1,5] <- pf(f.stat, t-1, den.df,
           lower.tail=FALSE)
        results[2,5] <- den.df
        results[3,5] <- mod$converged
        #column and row names
        colnames(results) <- c('mean','se','lower.ci
           ','upper.ci','P.DDF.CONV')
        rownames(results) <- paste0('time',seq(1,t
           ,1))
} else {
        results <- matrix(NA, nrow=(t*2), ncol=5)
        vbeta <- vbeta[1:(2*t),1:(2*t)]
        #Calculate lsmeans, se, 95% CI for 1st group
        for (i in 1:t){
                L <- rep(0,2*t)
                L[1] <- L[i] <- 1
                results[i,1] <- L %*% b.hat
                results[i,2] <- sqrt(L %*% vbeta %*%
                    L)
                results[i,3:4] <- t.confidence(
                   results[i,1],results[i,2],den.df)
        }
        #Convert back to original scale
        results[(1:t),2] <- delta.logit(results[(1:t
           ),1],results[(1:t),2])
        results[(1:t),c(1,3:4)] <- inv.logit(results
           [(1:t),c(1,3:4)])
        #Calculate lsmeans, se, 95% CI for 2nd group
        for (i in (t+1):(2*t)) {
                L <- rep(0,2*t)
                L[c(1,t+1)] <- 1
                L[c(i-t,i)] <- 1
                results[i,1] <- L %*% b.hat
                results[i,2] <- sqrt(L %*% vbeta %*%
                    L)
                results[i,3:4] <- t.confidence(
                   results[i,1],results[i,2],den.df)
        }
        #Convert back to original scale
        results[(t+1):(2*t),2] <- delta.logit(
           results[(t+1):(2*t),1],results[(t+1):(2*t
           ),2])
        results[(t+1):(2*t),c(1,3:4)] <- inv.logit(
           results[(t+1):(2*t),c(1,3:4)])
        #f-statistic
        L <- diag(2*t)
        L <- L[-c(1:(t+1)),]
        f.stat <- t(L %*% b.hat) %*% solve(L %*%
           vbeta %*% t(L)) %*% L %*% b.hat/(t-1)
        results[1,5] <- pf(f.stat, t-1, den.df,
           lower.tail=FALSE)
        results[2,5] <- den.df
        results[3,5] <- mod$converged
        #column and row names
        colnames(results) <- c('mean','se','lower.ci
           ','upper.ci','P.DDF.CONV')
        rownames(results) <- c(paste0('trt1_time',
           seq(1,t,1)),paste0('trt2_time',seq(1,t,1)
           ))
}
}

if (type=='LNMVB') {
if (trt.len==1) {
        #var-cov matrix
        vbeta <- vbeta[1:t,1:t]
        results <- matrix(NA, nrow=t, ncol=5)
        #Calculate lsmeans, se, 95% CI
        for (i in 1:t){
                L <- rep(0,t)
                L[i] <- 1
                results[i,1] <- L %*% b.hat
                results[i,2] <- sqrt(L %*% vbeta %*%
                    L)
                results[i,3:4] <- t.confidence(
                   results[i,1],results[i,2],den.df)
        }
        #Convert back to original scale
        results[,2] <- delta.logit(results[,1],
           results[,2])
        results[,c(1,3:4)] <- inv.logit(results[,c
           (1,3:4)])
        #f-statistic
        L <- matrix(c(rep(1,3),-1,0,0,0,-1,0,0,0,-1)
           ,nrow=3,byrow=FALSE)
        f.stat <- t(L %*% b.hat) %*% solve(L %*%
           vbeta %*% t(L)) %*% L %*% b.hat/(t-1)
        results[1,5] <- pf(f.stat, t-1, den.df,
           lower.tail=FALSE)
        results[2,5] <- den.df
        results[3,5] <- mod$converged
        #column and row names
        colnames(results) <- c('mean','se','lower.ci
           ','upper.ci','P.DDF.CONV')
        rownames(results) <- paste0('time',seq(1,t
           ,1))
} else {
        results <- matrix(NA, nrow=(t*2), ncol=5)
        vbeta <- vbeta[1:(2*t),1:(2*t)]
        #Calculate lsmeans, se, 95% CI for 1st group
        for (i in 1:t){
                L <- rep(0,2*t)
                L[i] <- 1
                results[i,1] <- L %*% b.hat
                results[i,2] <- sqrt(L %*% vbeta %*%
                    L)
                results[i,3:4] <- t.confidence(
                   results[i,1],results[i,2],den.df)
        }
        #Convert back to original scale
        results[(1:t),2] <- delta.logit(results[(1:t
           ),1],results[(1:t),2])
        results[(1:t),c(1,3:4)] <- inv.logit(results
           [(1:t),c(1,3:4)])
        #Calculate lsmeans, se, 95% CI for 2nd group
        for (i in 1:t) {
                L <- rep(0,2*t)
                L[c(i,i+t)] <- 1
                results[(i+t),1] <- L %*% b.hat
                results[(i+t),2] <- sqrt(L %*% vbeta
                    %*% L)
                results[(i+t),3:4] <- t.confidence(
                   results[(i+t),1],results[(i+t)
                   ,2],den.df)
        }
        #Convert back to original scale
        results[(t+1):(2*t),2] <- delta.logit(
           results[(t+1):(2*t),1],results[(t+1):(2*t
           ),2])
        results[(t+1):(2*t),c(1,3:4)] <- inv.logit(
           results[(t+1):(2*t),c(1,3:4)])
        #f-statistic
        L <- matrix(c(rep(0,12),rep(-1,3)
           ,1,0,0,0,1,0,0,0,1),nrow=3,byrow=FALSE)
        f.stat <- t(L %*% b.hat) %*% solve(L %*%
           vbeta %*% t(L)) %*% L %*% b.hat/(t-1)
        results[1,5] <- pf(f.stat, (t-1), den.df,
           lower.tail=FALSE)
        results[2,5] <- den.df
        results[3,5] <- mod$converged
        #column and row names
        colnames(results) <- c(’mean’,’se’,’lower.ci
           ’,’upper.ci’,’P.DDF.CONV’)
        rownames(results) <- c(paste0(’trt1_time’,
           seq(1,t,1)),paste0(’trt2_time’,seq(1,t,1)
           ))
}
}

if (type==’SLMVB’) {
if (trt.len==1) {
        #var-cov matrix
        b.hat <- b.hat[1:t]
        vbeta <- vbeta[1:t,1:t]
        results <- matrix(NA, nrow=t, ncol=5)
        #Calculate lsmeans, se, 95% CI
        for (i in 1:t){
                L <- rep(0,t)
                L[i] <- 1
                results[i,1] <- L %*% b.hat
                results[i,2] <- sqrt(L %*% vbeta %*%
                    L)
                results[i,3:4] <- t.confidence(
                   results[i,1],results[i,2],den.df)
        }
        #Convert back to original scale
        results[,2] <- delta.logit(results[,1],
           results[,2])
        results[,c(1,3:4)] <- inv.logit(results[,c
           (1,3:4)])
        #f-statistic
        L <- matrix(c(rep(1,3),-1,0,0,0,-1,0,0,0,-1)
           ,nrow=3,byrow=FALSE)
        f.stat <- t(L %*% b.hat) %*% solve(L %*%
           vbeta %*% t(L)) %*% L %*% b.hat/(t-1)
        results[1,5] <- pf(f.stat, t-1, den.df,
           lower.tail=FALSE)
        results[2,5] <- den.df
        results[3,5] <- results[3,5] <- mod$
           converged
        #column and row names
        colnames(results) <- c(’mean’,’se’,’lower.ci
           ’,’upper.ci’,’P.DDF.CONV’)
        rownames(results) <- paste0(’time’,seq(1,t
           ,1))
} else {
        results <- matrix(NA, nrow=(t*2), ncol=5)
        b.hat <- b.hat[1:(2*t)]
        vbeta <- vbeta[1:(2*t),1:(2*t)]
        #Calculate lsmeans, se, 95% CI for 1st group
        for (i in 1:t){
```

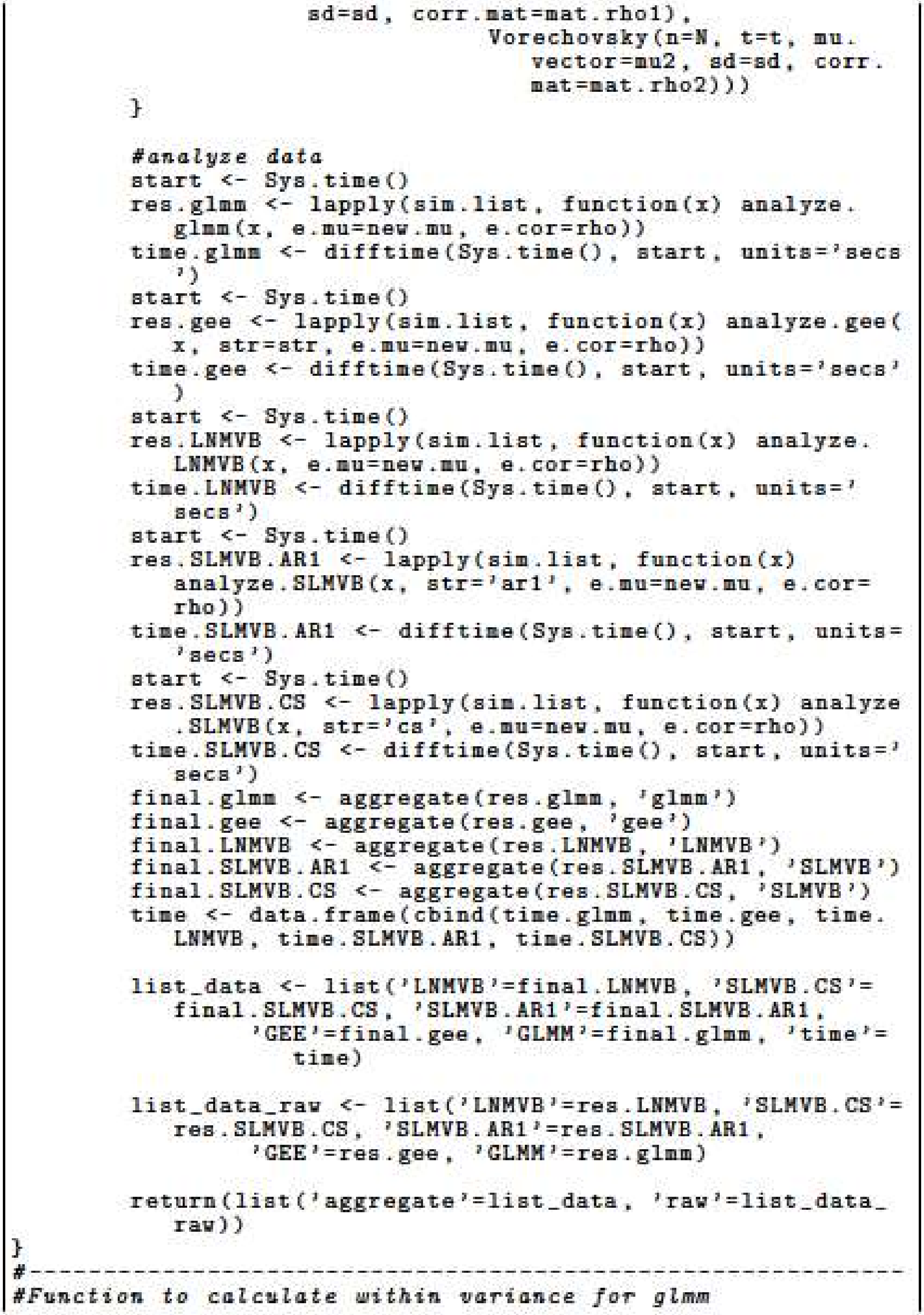
```
                        L <- rep(0,2*t)
                        L[i] <- 1
                        results[i,1] <- L %*% b.hat
                        results[i,2] <- sqrt(L %*% vbeta %*%
                            L)
                        results[i,3:4] <- t.confidence(
                           results[i,1],results[i,2],den.df)
                }
                #Convert back to original scale
                results[(1:t),2] <- delta.logit(results[(1:t
                   ),1],results[(1:t),2])
                results[(1:t),c(1,3:4)] <- inv.logit(results
                   [(1:t),c(1,3:4)])
                #Calculate lsmeans, se, 95% CI for 2nd group
                for (i in 1:t) {
                        L <- rep(0,2*t)
                        L[c(i,i+t)] <- 1
                        results[(i+t),1] <- L %*% b.hat
                        results[(i+t),2] <- sqrt(L %*% vbeta
                            %*% L)
                        results[(i+t),3:4] <- t.confidence(
                           results[(i+t),1],results[(i+t)
                           ,2],den.df)
                }
                #Convert back to original scale
                results[(t+1):(2*t),2] <- delta.logit(
                   results[(t+1):(2*t),1],results[(t+1):(2*t
                   ),2])
                results[(t+1):(2*t),c(1,3:4)] <- inv.logit(
                   results[(t+1):(2*t),c(1,3:4)])
                #f-statistic
                L <- matrix(c(rep(0,12),rep(-1,3)
                   ,1,0,0,0,1,0,0,0,1),nrow=3,byrow=FALSE)
                f.stat <- t(L %*% b.hat) %*% solve(L %*%
                   vbeta %*% t(L)) %*% L %*% b.hat/(t-1)
                results[1,5] <- pf(f.stat, (t-1), den.df,
                   lower.tail=FALSE)
                results[2,5] <- den.df
                results[3,5] <- results[3,5] <- mod$
                   converged
                #column and row names
                colnames(results) <- c('mean','se','lower.ci
                   ','upper.ci','P.DDF.CONV')
                rownames(results) <- c(paste0('trt1_time',
                   seq(1,t,1)),paste0('trt2_time',seq(1,t,1)
                   ))
        }
        }
return(results)
}
#----------------------------------------------
#function to analyse simuation using GLMM
#inputs: data frame (long format); vector of expected means;
    expected correlation
#ouput: matrix whose columns are expected mean; mean; se;
```

```
   95% CI; P for F-Statistic/DDF
analyze.glmm <- function(data, e.mu, e.cor) {
        t <- length(unique(data$time))
        #Fit GLMM to the simulated data using package
           glmmTMB
        #Type of model fit depends on number of treatment
           groups
        if (length(unique(data$trt)) == 1) {
                #fit model
                mod.glmm <- tryCatch({mixed_model(response~
                   time, random= ~ 1|subj, data=data, family
                   =beta.glmm, n_phis=1)},
                        error=function(e){print('error␣mod.
                           glmm'); 1})
                #calculate lsmeans
                results <- lsmeans(mod.glmm, 'glmm', ddf(
                   data))
                #add expected mean to results matrix
                results <- cbind(e.mu, results)
                colnames(results)[1] <- c('expected')
                #add correlation and expected correlation
                e.corr <- corr <- rep(NA,length(results[,1])
                   )
                e.corr <- e.cor
                #e.corr <- rho.beta(alpha, beta, p)
                corr <- tryCatch({as.numeric(mod.glmm$D)/(as
                   .numeric(mod.glmm$D) + sigma.e(inv.logit(
                   mod.glmm$coefficients[1]), exp(mod.glmm$
                   phis)))},
                        error=function(e){NA; NA})
                results <- cbind(results, e.corr, corr)
        } else {
                #fit model
                mod.glmm <- tryCatch({mixed_model(response~
                   time+trt+time:trt, random= ~ 1|subj, data
                   =data, family=beta.glmm, n_phis=1)},
                        error=function(e){print('error␣mod.
                           glmm'); 2})
                #calculate lsmeans
                results <- lsmeans(mod.glmm, 'glmm', ddf(
                   data))
                #add expected mean to results matrix
                results <- cbind(e.mu, results)
                colnames(results)[1] <- c('expected')
                #add correlation and expected correlation
                e.corr <- corr <- rep(NA,length(results[,1])
                   )
                e.corr <- e.cor
                corr <- tryCatch({as.numeric(mod.glmm$D)/(as
                   .numeric(mod.glmm$D) + sigma.e(inv.logit(
                   mod.glmm$coefficients[1]), exp(mod.glmm$
                   phis)))},
                        error=function(e){NA; NA})
                results <- cbind(results, e.corr, corr)
        }
        results <- metrics(data.frame(results))
        return(results)
}
#---------------------------------------
#function to calculate metrics
#input: data.frame with following columns - expected, mean,
   CI, P.Value for f statistic
#output: input data.frame with following columns added, bias
   , ...
metrics <- function(data){
        data$bias <- data$mean - data$expected
        data$coverage <- (data$expected > data$lower.ci &
           data$expected < data$upper.ci)
        data$bias.corr <- data$corr - data$e.corr
        return(data)
}
#-------------------------------------------
#function to aggregate results
#inputs: results list
#outputs: data.frame
aggregate <- function(list.res, type) {
        n.row <- length(list.res[[1]][,1])
        agg <- matrix(NA, nrow=n.row, ncol=13)
        agg[,1] <- list.res[[1]]$expected
        agg[,2] <- apply(sapply(list.res, function(x) x$mean
           ),1,mean, na.rm=TRUE)
        agg[,3:6] <- t(apply(sapply(list.res, function(x) x$
           bias[1:n.row]),1, function(y) {
                c(mean(y, na.rm=TRUE), min(y, na.rm=TRUE),
                   max(y, na.rm=TRUE), sqrt(mean(y^2, na.rm=
                   TRUE)))
        }))

        #determine power and convergence
        agg[1,7] <- sum(sapply(list.res, function(x) x$P.DDF
           .CONV[1]) <= 0.05, na.rm=TRUE)/length(list.res)
        agg[2,7] <- mean(sapply(list.res, function(x) x$P.
           DDF.CONV[2]), na.rm=TRUE)
        agg[3,7] <- sum(sapply(list.res, function(x) x$P.DDF
           .CONV[3]), na.rm=TRUE)/length(list.res)

        #non-estimated se's are imputed as FALSE coverage
        tmp <- sapply(list.res, function(x) x$coverage)
        tmp[is.na(tmp)] <- FALSE
        agg[,8] <- apply(tmp, 1, sum)/length(list.res)

        #missing length of missing ci's are imputed as
           random normal, using mean and sd of other obs of
           that time
        tmp2 <- sapply(list.res, function(x) x$upper.ci - x$
           lower.ci)
        for (i in 1:nrow(tmp2)) {
                if (sum(is.na(tmp2[i,])) > 0) {
                        tmp2[i,is.na(tmp2[i,])] <- rnorm(sum
                           (is.na(tmp2[i,])), mean(tmp2[i,],
                          na.rm=TRUE), sd(tmp2[i,], na.rm=
                         TRUE))
                }
        }

        agg[,9] <- apply(tmp2, 1, mean)
        agg[,10] <- list.res[[1]]$e.corr[1]
        agg[,11] <- apply(sapply(list.res, function(x) x$
           corr),1,mean, na.rm=TRUE)
        agg[,12] <- apply(sapply(list.res, function(x) x$
           bias.corr),1,mean, na.rm=TRUE)

        #determine percent se's not calaculated
        tmp3 <- sapply(list.res, function(x) x$se)
        agg[1,13] <- sum(is.na(tmp3))/(nrow(tmp3)*ncol(tmp3)
           )

        if (type=='LNMVB') {
                #EXY cannot always be calculated for the
                   correlation.
                #Impute missing values of correlation as a
                   random draw from normal using empirical
                   mean and sd
                tmp4 <- sapply(list.res, function(x) x$corr)
                #Percent of missing correlation
                if ( sum(is.na(tmp[-1,])) > 0 ) agg[2,13] <-
                    sum(is.na(tmp4[-1,]))/(nrow(tmp4[-1,])*
                   ncol(tmp4))
                #Imputation
                for (i in 2:length(list.res[[1]]$corr)) {
                        if (sum(is.na(tmp4[i,])) > 0) {
                                tmp4[i,is.na(tmp4[i,])] <-
                                   rnorm(sum(is.na(tmp4[i,])
                                   ), mean(tmp4[i,], na.rm=
                                   TRUE), sd(tmp4[i,], na.rm
                                   =TRUE))
                        }
                }
                agg[,11] <- apply(tmp4,1,mean)
                agg[,12] <- apply(tmp4-list.res[[1]]$e.corr,
                    1, mean)
        }

        colnames(agg) <- c('expected', 'mean', 'bias', 'min.
           bias', 'max.bias','RMSD','Power.DDF.CONV','
           Coverage','Mean.Len.CI','Exp.Corr','Corr','Bias.
           Corr','PctMiss.SE.COR')
        rownames(agg) <- rownames(list.res[[1]])
        return(data.frame(agg))
}
#-----------------------------------------
#GEE using package geeM - need to specify link function, etc
 .
LinkFun <- function(arg){log(arg/(1-arg))}
InvLink <- function(arg){exp(arg)/(1+exp(arg))}
```

```
InvLinkDeriv <- function(arg){exp(arg)/(1+exp(arg))^2}
VarFun <- function(arg){arg*(1-arg)}
FunList <- list(LinkFun, VarFun, InvLink, InvLinkDeriv)
#----------------------------------------------------
#function to analyze simuation using GEE
#inputs: data frame (long format); correlation structure ('
   ar1');  vector of expected means; expected correlation
#ouput: matrix whose columns are expected mean; mean; se;
   95% CI; P for F-Statistic/DDF
analyze.gee <- function(data, str, e.mu, e.cor) {
        t <- length(unique(data$time))

        gee.cor <- tolower(str)
        if (gee.cor == 'cs') gee.cor <- 'exchangeable'

        #Fit GEE to the simulated data using package geeM
        #Type of model fit depends on number of treatment
           groups
        if (length(unique(data$trt)) == 1) {
                #fit model
                mod.gee <- geem(response~time, id=subj,
                   family=FunList, corstr=gee.cor, data=data
                   )
                #calculate lsmeans
                results <- lsmeans(mod.gee, 'gee', ddf(data)
                   )
                #add expected mean to results matrix
                results <- cbind(e.mu,results)
                colnames(results)[1] <- c('expected')
                #add correlation and expected correlation
                e.corr <- corr <- rep(NA,length(results[,1])
                   )
                e.corr <- e.cor
                corr <- mod.gee$alpha
                results <- cbind(results, e.corr, corr)

        } else {
                #fit model
                mod.gee <- geem(response~time+trt+time:trt,
                   id=subj, family=FunList, corstr=gee.cor,
                   data=data)
                #calculate lsmeans
                results <- lsmeans(mod.gee, 'gee', ddf(data)
                   )
                #add expected mean to results matrix
                results <- cbind(e.mu,results)
                colnames(results)[1] <- c('expected')
                #add correlation and expected correlation
                e.corr <- corr <- rep(NA,length(results[,1])
                   )
                e.corr <- e.cor
                corr <- mod.gee$alpha
                results <- cbind(results, e.corr, corr)
        }
        results <- metrics(data.frame(results))
        return(results)
}
#-----------------------------------------------
#function to analyze simuation using LNMVB
#inputs: data frame (long format); vector of expected means;
    expected correlation
#ouput: matrix whose columns are expected mean; mean; se;
   95% CI; P for F-Statistic/DDF
analyze.LNMVB <- function(data, e.mu, e.cor) {
        t <- length(unique(data$time))
        #fit model
        mod.LNMVB <- LNMVB(data)
        #calculate lsmeans
        results <- lsmeans(mod.LNMVB, 'LNMVB', ddf(data))
        #add expected mean to results matrix
        results <- cbind(e.mu,results)
        colnames(results)[1] <- c('expected')
        #add correlation and expected correlation
        e.corr <- corr <- rep(NA,length(results[,1]))
        e.corr <- e.cor
        corr <- corr.LNMVB(mod.LNMVB)
        results <- cbind(results, e.corr, corr)
        results <- metrics(data.frame(results))
        return(results)
}
#-----------------------------------------------
#function to calculate pairwise correlation of LNMVB
#inputs: Means of pair and shared parameter
#ouput: correlation
pair.corr.LNMVB <- function(m1, m2, c){
        integ <- function(x,y){
                be.num <- gamma(c*(1 + m1/(1-m1) + m2/(1-m2)
                   ))
                be.den <- gamma(c)*gamma(c*m1/(1-m1))*gamma(
                   c*m2/(1-m2))
                num <- (x/(1-x))^(m1*c/(1-m1) - 1)*(1-x)
                   ^(-2)*(y/(1-y))^(m2*c/(1-m2) - 1)*(1-y)
                   ^(-2)*x*y
                den <- (1 + x/(1-x) + y/(1-y))^(c*(1 + m1/
                   (1-m1) + m2/(1-m2)))
                return(be.num*num/(be.den*den))
        }

        exy <- tryCatch({integral2(integ, 0, 1, 0, 1)$Q},
                error=function(e){print('error␣integral2');
                   NA})
        ex <- m1
        ey <- m2
        sx <- m1*(1-m1)^2/(1+c-m1)
        sy <- m2*(1-m2)^2/(1+c-m2)
        corr <- (exy-ex*ey)/sqrt(sx*sy)
        return(corr)
}
#----------------------------------------------
#function to calculate all adjoining pairwise correlations
```

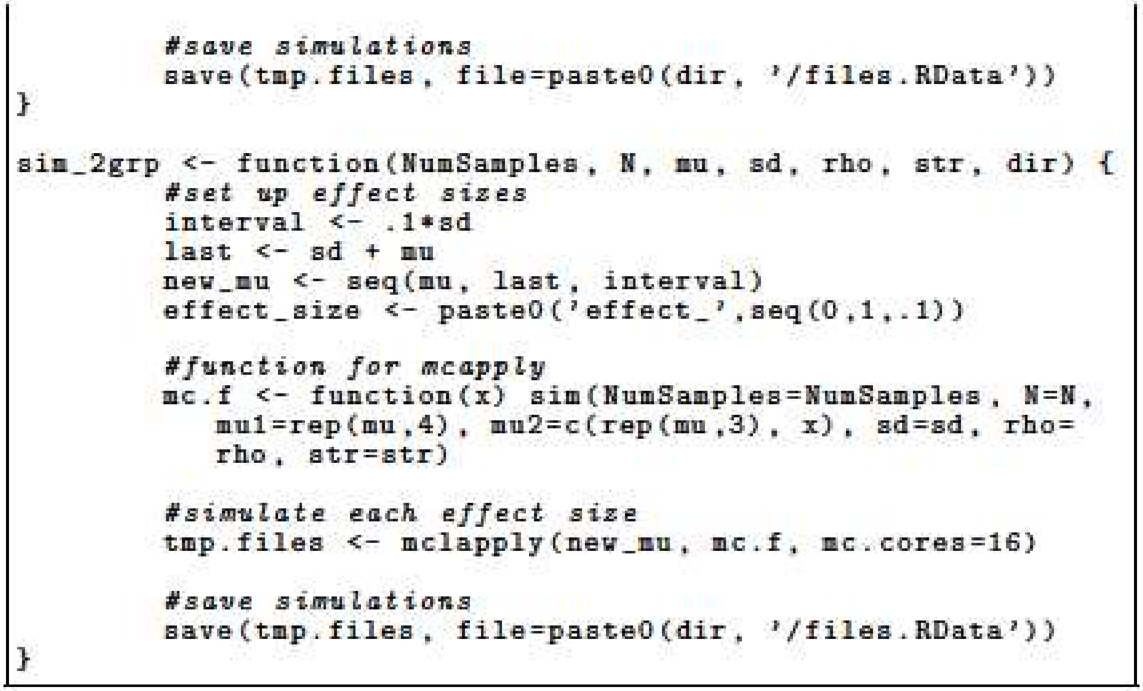


```
   for LNMVB
#inputs: model
#ouput: vector of pairwise correlations
corr.LNMVB <- function(mod) {
        trt <- unique(mod$dat$trt)
        t <- length(unique(mod$dat$time))
        corr <- vector('numeric' , length(mod$beta) - 2)
        c <- exp(mod$beta[length(mod$beta)])
        if (length(trt) == 1) {
                mu <- inv.logit(mod$beta[1:t])
        }
        if (length(trt) == 2) {
                mu <- vector('numeric', 2*t)
                mu[1:t] <- mod$beta[1:t]
                for (i in (t+1):(2*t)) {
                        mu[i] <- mu[i-t] + mod$beta[i]
                }
                mu <- inv.logit(mu)
        }
        for (i in 1:length(corr)){
                corr[i] <- pair.corr.LNMVB(mu[i], mu[i+1], c
                   )
        }
        corr <- c(NA, corr)
        return(corr)
}
#-----------------------------------------------
#function to analyze simuation using SLMVB.AR1
#inputs: data frame (long format); correlation structure ('
   ar1'); vector of expected means; expected correlation
#ouput: matrix whose columns are expected mean; mean; se;
   95% CI; P for F-Statistic/DDF
analyze.SLMVB <- function(data, str, e.mu, e.cor) {
        t <- length(unique(data$time))
        #fit model
        if (str=='ar1') mod.SLMVB <- SLMVB.AR1(data)
        if (str=='cs') mod.SLMVB <- SLMVB.CS(data)
        #calculate lsmeans
        results <- lsmeans(mod.SLMVB, 'SLMVB', ddf(data))
        #add expected mean to results matrix
        results <- cbind(e.mu, results)
        colnames(results)[1] <- c('expected')
        #add correlation and expected correlation
        e.corr <- corr <- rep(NA,length(results[,1]))
        e.corr <- e.cor
        corr <- mod.SLMVB$beta[length(mod.SLMVB$beta)]
        results <- cbind(results, e.corr, corr)
        results <- metrics(data.frame(results))
        return(results)
}
#-----------------------------------------
#Support function for Vorechovsky's method. Determines
   correlation matrix using Nataf's transformation.
p.tilda <- function(p.target, mu1, mu2, s) {
        a1 <- alpha.beta(mu1, s^2)[1]
        b1 <- alpha.beta(mu1, s^2)[2]
        a2 <- alpha.beta(mu2, s^2)[1]
        b2 <- alpha.beta(mu2, s^2)[2]
        p.tmp <- function(p) {
                f <- function(u1,u2) {
                        tmp1 <- (qbeta(pnorm(u1), shape1=a1,
                            shape2=b1) - mu1)/s
                        tmp2 <- (qbeta(pnorm(u2), shape1=a2,
                            shape2=b2) - mu2)/s
                        tmp3 <- 1/(2*pi*sqrt(1-p^2))*exp(-1/
                           (2*(1-p^2))*(u1^2 - 2*p*u1*u2 +
                           u2^2))
                        tmp1*tmp2*tmp3
                }
                integral2(f, -10, 10, -10, 10)$Q - p.target
        }
        nleqslv(0.5, p.tmp, method='Newton')$x
}
#---------------------------------------------------
#function to simulate beta responses for Vorechovsky's
   method
#inputs: n - num subjecs; t - num repeated measures; vector
   of means; common sd; target correlation matrix
#output: n x t matrix - rows are subjects and columns are
   repeated measures
Vorechovsky <- function(n, t, mu.vector, sd, corr.mat) {
        U <- chol(corr.mat)
        tmp <- matrix(rnorm(n*t), nrow=n, ncol=t)
        mat <- tmp %*% U

        for (i in 1:t) {
                a <- alpha.beta(mu.vector[i], sd^2)[1]
                b <- alpha.beta(mu.vector[i], sd^2)[2]
                mat[,i] <- qbeta(pnorm(mat[,i]), shape1=a,
                   shape2=b)
        }
        return(mat)
}
#--------------------------------------------------------
#Function to make correlation matrix for Vorechovsky's
   method
#input: rho, number of repeated measures, type of
   correlation - CS, AR1, UNSTR
#output: txt correlation matrix that has been Nataf
   transformed
make.corr <- function(rho, t, mu.vector, sd, type) {
        mat <- diag(t)

        if (toupper(type) == 'CS') {
                for (i in 1:t) {
                        for (j in i:t) {
                                if (i != j) mat[i,j] <- mat[
                                   j,i] <- rho
                        }
                }
        }

        if (toupper(type) == 'AR1') {
                for (i in 1:t) {
                        for (j in i:t) {
                                if (i != j) mat[i,j] <- mat[
                                   j,i] <- rho^(abs(i-j))
                        }
                }
        }

        if (toupper(type) == 'UNSTR') {
                for (i in 1:t) {
                        for (j in i:t) {
                                if (i != j) mat[i,j] <- mat[
                                   j,i] <- runif(1)
                        }
                }
        }

        for (i in 1:t) {
                for (j in i:t) {
                        if (i != j) {
                                mat[i,j] <- mat[j,i] <- p.
                                   tilda(mat[i,j], mu.vector
                                   [i], mu.vector[j], sd)
                        }
                }
        }
        return(mat)
}
#----------------------------------------------
#Function to run simulation
#input: Numsamples, N, t, mu1 (group 1), mu2 (group2), sd,
   rho, corr.structure
#output: list - results of LMMVB, SLMVB, GEE, GLMM, and time
sim <- function(NumSamples, N, t=4, mu1, mu2=NULL, sd, rho,
   str) {
        #Set up empty list
        sim.list <- vector('list', NumSamples)

        #Set up correlation matrix
        mat.rho1 <- make.corr(rho, t, mu1, sd, str)

        #Simulate data
        if (is.null(mu2)) {
                new.mu <- mu1
                sim.list <- lapply(sim.list, function(x)
                   long(Vorechovsky(n=N, t=t, mu.vector=mu1,
                    sd=sd, corr.mat=mat.rho1)))
        } else {
                mat.rho2 <- make.corr(rho, t, mu2, sd, str)
                new.mu <- c(mu1, mu2)
                sim.list <- lapply(sim.list, function(x)
                   long(Vorechovsky(n=N, t=t, mu.vector=mu1,
```



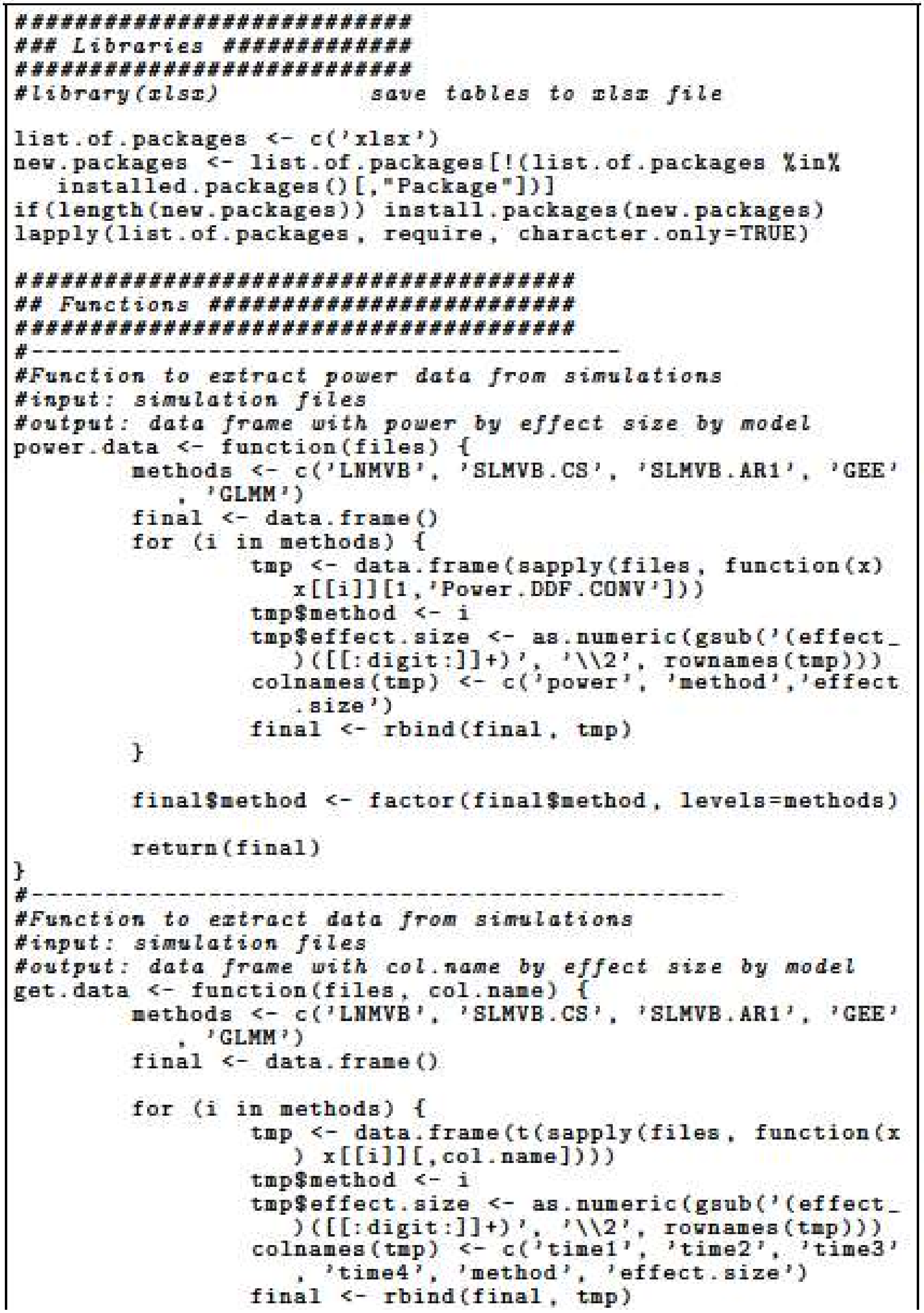
```
                    sd=sd, corr.mat=mat.rho1),
                                Vorechovsky(n=N, t=t, mu.
                                   vector=mu2, sd=sd, corr.
                                   mat=mat.rho2)))
        }

        #analyze data
        start <- Sys.time()
        res.glmm <- lapply(sim.list, function(x) analyze.
           glmm(x, e.mu=new.mu, e.cor=rho))
        time.glmm <- difftime(Sys.time(), start, units='secs
           ')
        start <- Sys.time()
        res.gee <- lapply(sim.list, function(x) analyze.gee(
           x, str=str, e.mu=new.mu, e.cor=rho))
        time.gee <- difftime(Sys.time(), start, units='secs'
           )
        start <- Sys.time()
        res.LNMVB <- lapply(sim.list, function(x) analyze.
           LNMVB(x, e.mu=new.mu, e.cor=rho))
        time.LNMVB <- difftime(Sys.time(), start, units='
           secs')
        start <- Sys.time()
        res.SLMVB.AR1 <- lapply(sim.list, function(x)
           analyze.SLMVB(x, str='ar1', e.mu=new.mu, e.cor=
           rho))
        time.SLMVB.AR1 <- difftime(Sys.time(), start, units=
           'secs')
        start <- Sys.time()
        res.SLMVB.CS <- lapply(sim.list, function(x) analyze
           .SLMVB(x, str='cs', e.mu=new.mu, e.cor=rho))
        time.SLMVB.CS <- difftime(Sys.time(), start, units='
           secs')
        final.glmm <- aggregate(res.glmm, 'glmm')
        final.gee <- aggregate(res.gee, 'gee')
        final.LNMVB <- aggregate(res.LNMVB, 'LNMVB')
        final.SLMVB.AR1 <- aggregate(res.SLMVB.AR1, 'SLMVB')
        final.SLMVB.CS <- aggregate(res.SLMVB.CS, 'SLMVB')
        time <- data.frame(cbind(time.glmm, time.gee, time.
           LNMVB, time.SLMVB.AR1, time.SLMVB.CS))

        list_data <- list('LNMVB'=final.LNMVB, 'SLMVB.CS'=
           final.SLMVB.CS, 'SLMVB.AR1'=final.SLMVB.AR1,
                'GEE'=final.gee, 'GLMM'=final.glmm, 'time'=
                   time)

        list_data_raw <- list('LNMVB'=res.LNMVB, 'SLMVB.CS'=
           res.SLMVB.CS, 'SLMVB.AR1'=res.SLMVB.AR1,
                'GEE'=res.gee, 'GLMM'=res.glmm)

        return(list('aggregate'=list_data, 'raw'=list_data_
           raw))
}
#-------------------------------------------------------------
#Function to calculate within variance for glmm
```

```
#See "The coefficient of determination r^2 and intra-class
   correlation from glmm revisted and expanded
#input: mu and phi
#output: sigma.e
sigma.e <- function(mu, phi) {
        tmp <- (1 - 2*mu)/(mu*(1-mu))
        tmp2 <-(mu*(1-mu))/(1+phi)
        (tmp*(1-tmp2)^(-2))^2*tmp2
}
#-------------------------------------------------------------
#Function to extract power data from simulations
#input: simulation files
#output: data frame with power by effect size by model
power.data <- function(files) {
        methods <- c('LNMVB', 'SLMVB.CS', 'SLMVB.AR1', 'GEE'
           , 'GLMM')
        final <- data.frame()
        for (i in methods) {
                tmp <- data.frame(sapply(files, function(x)
                   x[[i]][1,'Power.DDF.CONV']))
                tmp$method <- i
                tmp$effect.size <- as.numeric(gsub('(effect_
                   )([[:digit:]]+)', '\\2', rownames(tmp)))
                colnames(tmp) <- c('power', 'method','effect
                   .size')
                final <- rbind(final, tmp)
        }

        final$method <- factor(final$method, levels=methods)

        return(final)
}
#
   --------------------------------------------------------------

#Function to run 1 treatment group simulation and save
   output to specified directory
#input: NumSamples - Replicates, N - num of subjects, mu -
   mean of first 3 time points, sd - standard deviation
#       rho - correlation, str - 'AR1' 'CS' 'UNSTR', dir -
   where to save results
#output:
sim_1grp <- function(NumSamples, N, mu, sd, rho, str, dir) {
        #set up effect sizes
        interval <- .1*sd
        last <- sd + mu
        new_mu <- seq(mu, last, interval)
        effect_size <- paste0('effect_',seq(0,1,.1))

        #function for mcapply
        mc.f <- function(x) sim(NumSamples=NumSamples, N=N,
           mu1=c(rep(mu,3), x), sd=sd, rho=rho, str=str)

        #simulate each effect size
        tmp.files <- mclapply(new_mu, mc.f, mc.cores=16)
```



```
        #save simulations
        save(tmp.files, file=paste0(dir, '/files.RData'))
}

sim_2grp <- function(NumSamples, N, mu, sd, rho, str, dir) {
        #set up effect sizes
        interval <- .1*sd
        last <- sd + mu
        new_mu <- seq(mu, last, interval)
        effect_size <- paste0('effect_',seq(0,1,.1))

        #function for mcapply
        mc.f <- function(x) sim(NumSamples=NumSamples, N=N,
           mu1=rep(mu,4), mu2=c(rep(mu,3), x), sd=sd, rho=
           rho, str=str)

        #simulate each effect size
        tmp.files <- mclapply(new_mu, mc.f, mc.cores=16)

        #save simulations
        save(tmp.files, file=paste0(dir, '/files.RData'))
}
```

dissertation_functions.R



```
############################
### Libraries #############
############################
#library(xlsx)          save tables to xlsx file

list.of.packages <- c('xlsx')
new.packages <- list.of.packages[!(list.of.packages %in%
   installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
lapply(list.of.packages, require, character.only=TRUE)

########################################
## Functions ##########################
########################################
#-----------------------------------------
#Function to extract power data from simulations
#input: simulation files
#output: data frame with power by effect size by model
power.data <- function(files) {
        methods <- c('LNMVB', 'SLMVB.CS', 'SLMVB.AR1', 'GEE'
           , 'GLMM')
        final <- data.frame()
        for (i in methods) {
                tmp <- data.frame(sapply(files, function(x)
                   x[[i]][1,'Power.DDF.CONV']))
                tmp$method <- i
                tmp$effect.size <- as.numeric(gsub('(effect_
                   )([[:digit:]]+)', '\\2', rownames(tmp)))
                colnames(tmp) <- c('power', 'method','effect
                   .size')
                final <- rbind(final, tmp)
        }

        final$method <- factor(final$method, levels=methods)

        return(final)
}
#------------------------------------------------
#Function to extract data from simulations
#input: simulation files
#output: data frame with col.name by effect size by model
get.data <- function(files, col.name) {
        methods <- c('LNMVB', 'SLMVB.CS', 'SLMVB.AR1', 'GEE'
           , 'GLMM')
        final <- data.frame()

        for (i in methods) {
                tmp <- data.frame(t(sapply(files, function(x
                   ) x[[i]][,col.name])))
                tmp$method <- i
                tmp$effect.size <- as.numeric(gsub('(effect_
                   )([[:digit:]]+)', '\\2', rownames(tmp)))
                colnames(tmp) <- c('time1', 'time2', 'time3'
                   , 'time4', 'method', 'effect.size')
                final <- rbind(final, tmp)
        }

        long <- melt(final, id.vars=c('effect.size', 'method
           '), value.name=col.name, variable.name='time')
        long$method <- factor(long$method, levels=methods)

        return(long)
}

get.data.1 <- function(files, col.name) {
        methods <- c('LNMVB', 'SLMVB.CS', 'SLMVB.AR1', 'GEE'
           , 'GLMM')
        final <- data.frame()

        for (i in methods) {
                if (col.name == 'PctMiss') tmp <- data.frame
                   (sapply(files, function(x) x[[i]][1,'
                   PctMiss.SE.COR']))
                if (col.name == 'CONV') tmp <- data.frame(
                   sapply(files, function(x) x[[i]][3,'Power
                   .DDF.CONV']))
                if (col.name == 'Bias.Corr') tmp <- data.
                   frame(sapply(files, function(x) mean(x[[i
                   ]][,'Bias.Corr'], na.rm=TRUE)))
                tmp$method <- i
                tmp$effect.size <- as.numeric(gsub('(effect_
                   )([[:digit:]]+)', '\\2', rownames(tmp)))
                colnames(tmp) <- c(col.name, 'method', '
                   effect.size')
                final <- rbind(final, tmp)
        }

        final$method <- factor(final$method, levels=methods)

        return(final)
}

get.data_2grp <- function(files, col.name) {
        methods <- c('LNMVB', 'SLMVB.CS', 'SLMVB.AR1', 'GEE'
           , 'GLMM')
        final <- data.frame()

        for (i in methods) {
                tmp <- data.frame(t(sapply(files, function(x
                   ) x[[i]][,col.name])))
                tmp$method <- i
                tmp$effect.size <- as.numeric(gsub('(effect_
                   )([[:digit:]]+)', '\\2', rownames(tmp)))
                colnames(tmp) <- c('time1_1', 'time2_1', '
                   time3_1', 'time4_1', 'time1_2', 'time2_2'
                   , 'time3_2', 'time4_2','method', 'effect.
                   size')
                final <- rbind(final, tmp)
        }
        long <- melt(final, id.vars=c('effect.size', 'method
           '), value.name=col.name, variable.name='time')
        long$trt <- gsub('(time[[:digit:]]_)([[:digit:]])',
           '\\2', long$time)
        long$time <- gsub('(time[[:digit:]])(_[[:digit:]])',
            '\\1', long$time)
        long$method <- factor(long$method, levels=methods)

        return(long)
}
#-----------------------------------------------
#Function to extract bias from simulations
#input: simulation files
#output: data frame with bias by effect size by model
get.bias <- function(files) {
        bias.df <- get.data(files, 'bias')
        bias.min <- get.data(files, 'min.bias')
        bias.max <- get.data(files, 'max.bias')
        bias <- merge(x=bias.df, y=bias.min, by=c('effect.
           size', 'method', 'time'))
        bias <- merge(x=bias, y=bias.max, by=c('effect.size'
           , 'method', 'time'))
        bias$bias <- paste0(round(bias$bias,3), '␣(', round(
           bias$min.bias,3), ',␣', round(bias$max.bias,3), '
           )')
        bias <- bias[, c('effect.size', 'method', 'time', '
           bias')]
        bias <- dcast(bias, effect.size + method ˜ time,
           value.var='bias')
        return(bias)
}

get.bias_2grp <- function(files) {
        bias.df <- get.data_2grp(files, 'bias')
        bias.min <- get.data_2grp(files, 'min.bias')
        bias.max <- get.data_2grp(files, 'max.bias')
        bias <- merge(x=bias.df, y=bias.min, by=c('effect.
           size', 'method', 'time', 'trt'))
        bias <- merge(x=bias, y=bias.max, by=c('effect.size'
           , 'method', 'time', 'trt'))
        bias$bias <- paste0(round(bias$bias,3), '␣(', round(
           bias$min.bias,3), ',␣', round(bias$max.bias,3), '
           )')
        bias <- bias[, c('effect.size', 'method', 'time', '
           trt', 'bias')]
        bias <- dcast(bias, effect.size + method ˜ trt +
           time, value.var='bias')
        return(bias)
}
#-----------------------------------------------
#functions to compile results of simulation
sim_1grp_compile_b <- function(dir) {
        #used to label facets of plots
        facet.name <- c('0.2'='Small␣effect␣size', '0.5'='
           Medium␣effect␣size', '0.8'='Large␣effect␣size')
```

```
tmp.files <- get(load(paste0(dir, '/files.Rdata')))
files <- lapply(tmp.files, function(x) x$aggregate)

effect_size <- paste0('effect_',seq(0,1,.1))
names(files) <- effect_size

#plot power curve
power <- power.data(files)
power.plot <- ggplot(data=power, aes(x=effect.size,
   y=power, color=method, linetype=method, shape=
   method)) + geom_point() + geom_line() +
        theme_bw() + theme(panel.grid=element_blank
           (), legend.background=element_rect(color=
           'black', linetype=1, size=.3),
        legend.position=c(.12,.87)) + labs(x='Effect
           ␣Size', y='Power', color='Model',
           linetype='Model', shape='Model') +
        scale_x_continuous(lim=c(0,1), breaks=seq
           (0,1,.1)) + scale_y_continuous(lim=c(0,1)
           , breaks=seq(0,1,.1)) +
        geom_hline(yintercept=0.05, linetype=2)

#save power plot
ggsave(filename=paste0(dir, './power.png'), power.
   plot, width=6, height=7, units='in', dpi=600)

#save power data
write.xlsx(dcast(power, effect.size ~ method, value.
   var='power'), file=paste0(dir, './results.xlsx'),
    sheetName='Power', row.names=FALSE, append=FALSE
   )

#plot coverage probabilities
coverage <- get.data(files, 'Coverage')
coverage.df.plot <- subset(coverage, effect.size
   ==0.2 | effect.size==0.5 | effect.size==0.8)
coverage.plot <- ggplot(coverage.df.plot, aes(x=time
   , y=Coverage, color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(),legend.
           title.align=c(0.5), legend.position=c
           (.75,.25), legend.justification=c(.5,.5),
            legend.text=element_text(size=8)) +
        guides(color=guide_legend(ncol=3, nrow=2,
           byrow=TRUE, title.position='top'), shape=
           guide_legend(ncol=3, nrow=2, byrow=TRUE,
           title.position='top'))+
        scale_y_continuous(lim=c(0.9,1), breaks=seq
           (.9,1, .01), labels=c(.90,rep('',4),.95,
           rep('',4),1.00)) +
        geom_hline(yintercept=.95, linetype=2) +
           labs(x=NULL, y='Coverage␣Probability',
           color='Model', shape='Model') +
        facet_wrap(~effect.size, ncol=2, labeller=as
           _labeller(facet.name))

#save coverage plot
ggsave(filename=paste0(dir, './coverage.png'),
   coverage.plot, width=6, height=3.5, units='in',
   dpi=600)

#prepare coverage data
mean.length <- get.data(files, 'Mean.Len.CI')
pct.miss <- get.data.1(files, 'PctMiss')
conv <- get.data.1(files, 'CONV')
coverage.wide <- dcast(coverage, effect.size +
   method ~ time, value.var='Coverage')
mean.length.wide <- dcast(mean.length, effect.size +
    method ~ time, value.var='Mean.Len.CI')
colnames(mean.length.wide) <- c('effect.size', '
   method', 'Length1', 'Length2', 'Lengt3', 'Length4
   ')
coverage.final <- merge(x=coverage.wide, y=mean.
   length.wide, by=c('effect.size', 'method'))
coverage.final <- merge(x=coverage.final, y=pct.miss
   , by=c('effect.size', 'method'))
coverage.final <- merge(x=coverage.final, y=conv, by
   =c('effect.size', 'method'))
coverage.final <- coverage.final[order(coverage.
   final$effect.size, coverage.final$method),]

#save coverage data
write.xlsx(coverage.final, file=paste0(dir, './
   results.xlsx'), sheetName='Coverage', row.names=
   FALSE, append=TRUE)

#mean bias plot
bias.df <- subset(get.data(files, 'bias'), effect.
   size==0.2 | effect.size==0.5 | effect.size==0.8)
bias.plot <- ggplot(bias.df, aes(x=time, y=bias,
   color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(),legend.
           title.align=c(0.5), legend.position=c
           (.75,.25), legend.justification=c(.5,.5),
            legend.text=element_text(size=8)) +
        guides(color=guide_legend(ncol=3, nrow=2,
           byrow=TRUE, title.position='top'), shape=
           guide_legend(ncol=3, nrow=2, byrow=TRUE,
           title.position='top'))+
        labs(x=NULL, y='Mean␣Bias', color='Model',
           shape='Model') + geom_hline(yintercept=0,
            linetype=2) +
        facet_wrap(~effect.size, ncol=2, labeller=as
           _labeller(facet.name))

#save bias plot
```

```
ggsave(filename=paste0(dir, './bias.png'), bias.plot
   , width=6, height=3.5, units='in', dpi=600)

#save bias data
write.xlsx(get.bias(files), file=paste0(dir, './
   results.xlsx'), sheetName='Bias', row.names=FALSE
   , append=TRUE)

#rmsd plot
rmsd <- get.data(files, 'RMSD')
rmsd.plot.df <- subset(rmsd, effect.size==0.2 |
   effect.size==0.5 | effect.size==0.8)
rmsd.plot <- ggplot(rmsd.plot.df, aes(x=time, y=RMSD
   , color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(),legend.
           title.align=c(0.5), legend.position=c
           (.75,.25), legend.justification=c(.5,.5),
            legend.text=element_text(size=8)) +
        guides(color=guide_legend(ncol=3, nrow=2,
           byrow=TRUE, title.position='top'), shape=
           guide_legend(ncol=3, nrow=2, byrow=TRUE,
           title.position='top'))+
        labs(x=NULL, y='RMSD', color='Model', shape=
           'Model') + geom_hline(yintercept=0,
           linetype=2) +
        facet_wrap(~effect.size, ncol=2, labeller=as
           _labeller(facet.name))

#save bias plot
ggsave(filename=paste0(dir, './rmsd.png'), rmsd.plot
   , width=6, height=3.5, units='in', dpi=600)

#save rmsd data
write.xlsx(dcast(rmsd, effect.size + method ~ time,
   value.var='RMSD'), file=paste0(dir, './results.
   xlsx'), sheetName='RMSD', row.names=FALSE, append
   =TRUE)

#Bias correlation plot
bias.cor <- get.data.1(files, 'Bias.Corr')
bias.cor.plot.df <- subset(bias.cor, effect.size
   ==0.2 | effect.size==0.5 | effect.size==0.8)

bias.cor.plot <- ggplot(bias.cor.plot.df, aes(x=
   effect.size, y=Bias.Corr, color=method, shape=
   method)) +
        geom_point(position=position_dodge(width
           =0.05)) + theme_bw() +
        theme(panel.grid=element_blank(), legend.
           background=element_rect(color='black',
           linetype=1, size=0.3)) +
        labs(x='Effect␣Size', y='Mean␣Bias␣
           Correlation', color='Model', shape='Model
```

```
                    ') + geom_hline(yintercept=0, linetype=2)
                     +
                scale_x_continuous(lim=c(0.1,.9), breaks=c
                    (0.2,0.5,0.8), labels=c('Small', 'Medium'
                    , 'Large'))

        #save bias correlation plot
        ggsave(filename=paste0(dir, './bias_cor.png'), bias.
            cor.plot, width=6, height=2, units='in', dpi=600)

        #save bias correlation data
        cor.wide <- dcast(bias.cor, effect.size ~ method,
            value.var='Bias.Corr')
        cor.wide <- cor.wide[,c('effect.size', 'LNMVB', '
            SLMVB.CS', 'SLMVB.AR1', 'GEE', 'GLMM')]
        write.xlsx(cor.wide, file=paste0(dir, './results.
            xlsx'), sheetName='Bias_Cor', row.names=FALSE,
            append=TRUE)

        #time
        tmp <- data.frame(apply(sapply(files, function(x) as
            .numeric(x[['time']])),1,sum))
        tmp$method <- gsub('(time.)([[:alpha:]]+)', '\\2',
            colnames(files[[1]]$time))
        tmp$method <- factor(tmp$method, levels=c('LNMVB', '
            SLMVB.CS', 'SLMVB.AR1', 'gee', 'glmm'))
        colnames(tmp) <- c('time.sec', 'method')
        tmp <- tmp[order(tmp$method), c('method', 'time.sec'
            )]
        time.wide <- dcast(tmp, .~method, value.var='time.
            sec')
        time.wide <- time.wide[,colnames(time.wide) %in% c('
            LNMVB', 'SLMVB.CS', 'SLMVB.AR1', 'gee', 'glmm')]
        rownames(time.wide) <- c('time.sec')

        write.xlsx(time.wide, file=paste0(dir, './results.
            xlsx'), sheetName='Time', row.names=TRUE, append=
            TRUE)
}

sim_2grp_compile_b <- function(dir) {
        #used to label facets of plots
        facet.name.row <- c('0.2'='Small␣effect␣size', '0.5'
            ='Medium␣effect␣size', '0.8'='Large␣effect␣size')
        facet.name.col <- c('1'='Group␣1', '2'='Group␣2')

        tmp.files <- get(load(paste0(dir, '/files.Rdata')))
        files <- lapply(tmp.files, function(x) x$aggregate)

        effect_size <- paste0('effect_',seq(0,1,.1))
        names(files) <- effect_size

        #plot power curve
        power <- power.data(files)
        power.plot <- ggplot(data=power, aes(x=effect.size,
   y=power, color=method, linetype=method, shape=
   method)) + geom_point() + geom_line() +
        theme_bw() + theme(panel.grid=element_blank
           (), legend.background=element_rect(color=
           'black', linetype=1, size=.3),
        legend.position=c(.12,.87)) + labs(x='Effect
           ␣Size', y='Power', color='Model',
           linetype='Model', shape='Model') +
        scale_x_continuous(lim=c(0,1), breaks=seq
           (0,1,.1)) + scale_y_continuous(lim=c(0,1)
           , breaks=seq(0,1,.1)) +
        geom_hline(yintercept=0.05, linetype=2)

#save power plot
ggsave(filename=paste0(dir, './power.png'), power.
   plot, width=6, height=7, units='in', dpi=600)

#save power data
write.xlsx(dcast(power, effect.size ~ method, value.
   var='power'), file=paste0(dir, './results.xlsx'),
    sheetName='Power', row.names=FALSE, append=FALSE
   )

#plot coverage probabilities
coverage <- get.data_2grp(files, 'Coverage')
coverage.df.plot <- subset(coverage, effect.size
   ==0.2 | effect.size==0.5 | effect.size==0.8)
coverage.plot <- ggplot(coverage.df.plot, aes(x=time
   , y=Coverage, color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(),legend.
           title.align=c(0.5), legend.position='
           bottom', legend.text=element_text(size=8)
           ) +
        scale_y_continuous(lim=c(0.9,1), breaks=seq
           (.9,1, .01), labels=c(.90,rep('',4),.95,
           rep('',4),1.00)) +
        geom_hline(yintercept=.95, linetype=2) +
           labs(x=NULL, y='Coverage␣Probability',
           color='Model', shape='Model') +
        facet_grid(effect.size ~ trt, labeller=
           labeller(effect.size=facet.name.row, trt=
           facet.name.col))

#save coverage plot
ggsave(filename=paste0(dir, './coverage.png'),
   coverage.plot, width=6, height=5.25, units='in',
   dpi=600)

#prepare coverage data
pct.miss <- get.data.1(files, 'PctMiss')
conv <- get.data.1(files, 'CONV')
coverage.wide <- dcast(coverage, effect.size +
   method ~ trt + time, value.var='Coverage')
```

```
coverage.final <- merge(x=coverage.wide, y=pct.miss,
     by=c('effect.size', 'method'))
coverage.final <- merge(x=coverage.final, y=conv, by
   =c('effect.size', 'method'))
coverage.final <- coverage.final[order(coverage.
   final$effect.size, coverage.final$method),]
mean.length <- get.data_2grp(files, 'Mean.Len.CI')
mean.length.wide <- dcast(mean.length, effect.size +
     method ~ trt + time, value.var='Mean.Len.CI')

#save coverage data
write.xlsx(coverage.final, file=paste0(dir, './
   results.xlsx'), sheetName='Coverage', row.names=
   FALSE, append=TRUE)
write.xlsx(mean.length.wide, file=paste0(dir, './
   results.xlsx'), sheetName='Length␣CI', row.names=
   FALSE, append=TRUE)

#mean bias plot
bias.df <- subset(get.data_2grp(files, 'bias'),
   effect.size==0.2 | effect.size==0.5 | effect.size
   ==0.8)
bias.plot <- ggplot(bias.df, aes(x=time, y=bias,
   color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(),legend.
           title.align=c(0.5), legend.position='
           bottom', legend.text=element_text(size=8)
           ) +
        labs(x=NULL, y='Mean␣Bias', color='Model',
           shape='Model') + geom_hline(yintercept=0,
            linetype=2) +
        facet_grid(effect.size ~ trt, labeller=
           labeller(effect.size=facet.name.row, trt=
           facet.name.col))

#save bias plot
ggsave(filename=paste0(dir, './bias.png'), bias.plot
   , width=6, height=5.25, units='in', dpi=600)

#save bias data
write.xlsx(get.bias_2grp(files), file=paste0(dir, '.
   /results.xlsx'), sheetName='Bias', row.names=
   FALSE, append=TRUE)

#rmsd plot
rmsd <- get.data_2grp(files, 'RMSD')
rmsd.plot.df <- subset(rmsd, effect.size==0.2 |
   effect.size==0.5 | effect.size==0.8)
rmsd.plot <- ggplot(rmsd.plot.df, aes(x=time, y=RMSD
   , color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(),legend.
                    title.align=c(0.5), legend.position='
                    bottom', legend.text=element_text(size=8)
                    ) +
                 labs(x=NULL, y='RMSD', color='Model', shape=
                    'Model') + geom_hline(yintercept=0,
                    linetype=2) +
                 facet_grid(effect.size ~ trt, labeller=
                    labeller(effect.size=facet.name.row, trt=
                    facet.name.col))

#save bias plot
ggsave(filename=paste0(dir, './rmsd.png'), rmsd.plot
   , width=6, height=5.25, units='in', dpi=600)

#save rmsd data
write.xlsx(dcast(rmsd, effect.size + method ~ trt +
   time, value.var='RMSD'), file=paste0(dir, './
   results.xlsx'), sheetName='RMSD', row.names=FALSE
   , append=TRUE)

#Bias correlation plot
bias.cor <- get.data.1(files, 'Bias.Corr')
bias.cor.plot.df <- subset(bias.cor, effect.size
   ==0.2 | effect.size==0.5 | effect.size==0.8)
bias.cor.plot <- ggplot(bias.cor.plot.df, aes(x=
   effect.size, y=Bias.Corr, color=method, shape=
   method)) +
        geom_point(position=position_dodge(width
           =0.05)) + theme_bw() +
        theme(panel.grid=element_blank(), legend.
           background=element_rect(color='black',
           linetype=1, size=0.3)) +
        labs(x='Effect␣Size', y='Mean␣Bias␣
           Correlation', color='Model', shape='Model
           ') + geom_hline(yintercept=0, linetype=2)
            +
        scale_x_continuous(lim=c(0.1,.9), breaks=c
           (0.2,0.5,0.8), labels=c('Small', 'Medium'
           , 'Large'))

#save bias correlation plot
ggsave(filename=paste0(dir, './bias_cor.png'), bias.
   cor.plot, width=6, height=2, units='in', dpi=600)

#save bias correlation data
cor.wide <- dcast(bias.cor, effect.size ~ method,
   value.var='Bias.Corr')
cor.wide <- cor.wide[,c('effect.size', 'LNMVB', '
   SLMVB.CS', 'SLMVB.AR1', 'GEE', 'GLMM')]
write.xlsx(cor.wide, file=paste0(dir, './results.
   xlsx'), sheetName='Bias_Cor', row.names=FALSE,
   append=TRUE)

#time
tmp <- data.frame(apply(sapply(files, function(x) as
           .numeric(x[['time']])),1,sum))
        tmp$method <- gsub('(time.)([[:alpha:]]+)', '\\2',
           colnames(files[[1]]$time))
        tmp$method <- factor(tmp$method, levels=c('LNMVB', '
           SLMVB.CS', 'SLMVB.AR1', 'gee', 'glmm'))
        colnames(tmp) <- c('time.sec', 'method')
        tmp <- tmp[order(tmp$method), c('method', 'time.sec'
           )]
        time.wide <- dcast(tmp, .~method, value.var='time.
           sec')
        time.wide <- time.wide[,colnames(time.wide) %in% c('
           LNMVB', 'SLMVB.CS', 'SLMVB.AR1', 'gee', 'glmm')]
        rownames(time.wide) <- c('time.sec')

        write.xlsx(time.wide, file=paste0(dir, './results.
           xlsx'), sheetName='Time', row.names=TRUE, append=
           TRUE)
}
#------------------------------------------
#functions to compile results of simulation
#input: directory of sample sizes
sim_1grp_compile_aggregate <- function(dir) {
        #directories of results
        dir.name <- c('N_15', 'N_30', 'N_50', 'N_100')

        #read in files
        tmp.files <- lapply(dir.name, function(x) get(load(
           paste0(dir, '/', x, './files.Rdata'))))
        names(tmp.files) <- dir.name

        #aggregate the results
        files <- lapply(tmp.files, function(i) lapply(i,
           function(x) x$aggregate))
        effect_size <- paste0('effect_',seq(0,1,.1))
        for (i in names(files)) names(files[[i]]) <- effect_
           size

        #get type I error and power data
        power <- lapply(files, function(x) power.data(x))
        power.df <- data.frame()
        for (i in names(power)) {
                power[[i]]$sample.size <- i
                power.df <- rbind(power.df, power[[i]])
        }
        power.df$sample.size <- factor(power.df$sample.size,
            levels=c('N_15', 'N_30', 'N_50', 'N_100'))

        #create power plots
        power.plot <- ggplot(data=power.df, aes(x=effect.
           size, y=power, color=method, linetype=method,
           shape=method)) + geom_point() + geom_line() +
                theme_bw() + theme(panel.grid=element_blank
                   (), legend.background=element_rect(color=
                   'black', linetype=1, size=.3),
                legend.position='bottom', legend.direction='
```

```
              horizontal') + labs(x='Effect␣Size', y='
              Power', color='Model', linetype='Model',
              shape='Model') +
          scale_x_continuous(lim=c(0,1), breaks=seq
              (0,1,.1)) + scale_y_continuous(lim=c(0,1)
              , breaks=seq(0,1,.1)) +
          geom_hline(yintercept=0.05, linetype=2) +
              facet_wrap(~sample.size, ncol=2)

#save plot
ggsave(filename=paste0(dir, './power.png'), power.
    plot, width=6, height=6, units='in', dpi=600)

#get mean bias data
bias.df.tmp <- lapply(files, function(x) subset(get.
    data(x, 'bias'), effect.size==0.2 | effect.size
    ==0.5 | effect.size==0.8))
bias.df <- data.frame()
for (i in names(bias.df.tmp)) {
          bias.df.tmp[[i]]$sample.size <- i
          bias.df <- rbind(bias.df, bias.df.tmp[[i]])
}
bias.df$sample.size <- factor(bias.df$sample.size,
    levels=c('N_15', 'N_30', 'N_50', 'N_100'))
bias.df$effect.size <- factor(bias.df$effect.size,
    levels=c(0.2, 0.5, 0.8), labels=c('Small␣effect␣
    size', 'Medium␣effect␣size', 'Large␣effect␣size')
    )

#create mean bias plots
bias.plot <- ggplot(bias.df, aes(x=time, y=bias,
    color=method, shape=method)) +
          geom_point(position=position_dodge(width
              =0.25)) + theme_bw() +
          theme(panel.grid=element_blank(), legend.
              background=element_rect(color='black',
              linetype=1, size=.3),
          legend.position='bottom', legend.direction='
              horizontal') +
          labs(x=NULL, y='Mean␣Bias', color='Model',
              shape='Model') + geom_hline(yintercept=0,
               linetype=2) +
          facet_grid(sample.size~effect.size)

#save plot
ggsave(filename=paste0(dir, './bias.png'), bias.plot
    , width=6, height=7.5, units='in', dpi=600)

#get rmsd data
rmsd <- lapply(files, function(x) subset(get.data(x,
     'RMSD'), effect.size==0.2 | effect.size==0.5 |
    effect.size==0.8))
rmsd.df <- data.frame()
for (i in names(rmsd)) {
          rmsd[[i]]$sample.size <- i
```

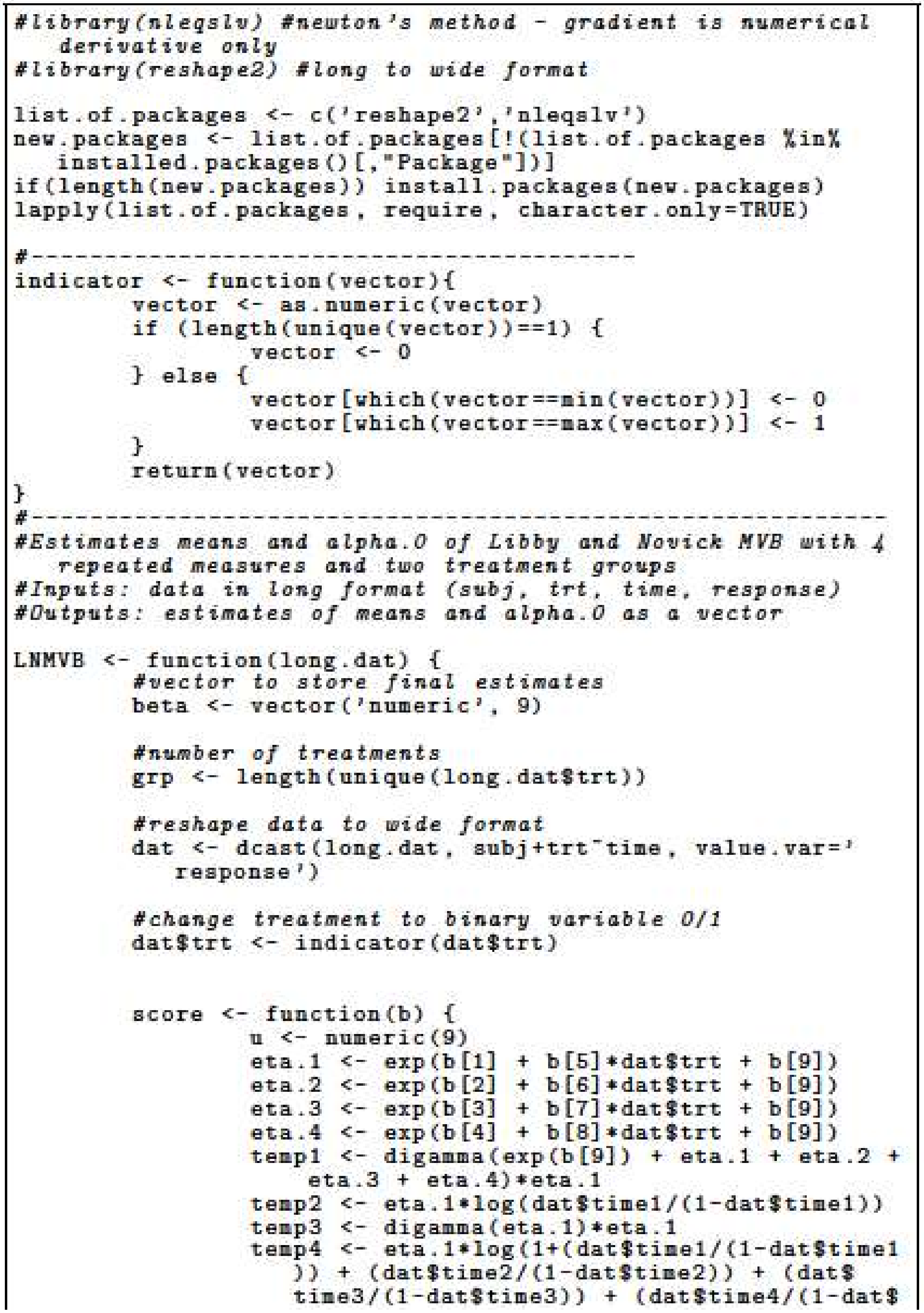
```
        rmsd.df <- rbind(rmsd.df, rmsd[[i]])
}
rmsd.df$sample.size <- factor(rmsd.df$sample.size,
   levels=c('N_15', 'N_30', 'N_50', 'N_100'))
rmsd.df$effect.size <- factor(rmsd.df$effect.size,
   levels=c(0.2, 0.5, 0.8), labels=c('Small␣effect␣
   size', 'Medium␣effect␣size', 'Large␣effect␣size')
   )

#creat rmsd plot
rmsd.plot <- ggplot(rmsd.df, aes(x=time, y=RMSD,
   color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(), legend.
           background=element_rect(color='black',
           linetype=1, size=.3),
        legend.position='bottom', legend.direction='
           horizontal')+
        labs(x=NULL, y='RMSD', color='Model', shape=
           'Model') + geom_hline(yintercept=0,
           linetype=2) +
        facet_grid(sample.size~effect.size)

#save plot
ggsave(filename=paste0(dir, './rmsd.png'), rmsd.plot
   , width=6, height=7.5, units='in', dpi=600)

coverage <- lapply(files, function(x) subset(get.
   data(x, 'Coverage'), effect.size==0.2 | effect.
   size==0.5 | effect.size==0.8))
coverage.df <- data.frame()
for (i in names(coverage)) {
        coverage[[i]]$sample.size <- i
        coverage.df <- rbind(coverage.df, coverage[[
           i]])
}

#get coverage data
coverage.df$sample.size <- factor(coverage.df$sample
   .size, levels=c('N_15', 'N_30', 'N_50', 'N_100'))
coverage.df$effect.size <- factor(coverage.df$effect
   .size, levels=c(0.2, 0.5, 0.8), labels=c('Small␣
   effect␣size', 'Medium␣effect␣size', 'Large␣effect
   ␣size'))

#create coverage plot
coverage.plot <- ggplot(coverage.df, aes(x=time, y=
   Coverage, color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(), legend.
           background=element_rect(color='black',
           linetype=1, size=.3),
        legend.position='bottom', legend.direction='
```

```
               horizontal')+
            scale_y_continuous(lim=c(0.9,1), breaks=seq
               (.9,1, .01), labels=c(.90,rep('',4),.95,
               rep('',4),1.00)) +
            geom_hline(yintercept=.95, linetype=2) +
               labs(x=NULL, y='Coverage␣Probability',
               color='Model', shape='Model') +
            facet_grid(sample.size~effect.size)

#save plot
ggsave(filename=paste0(dir, './coverage.png'),
   coverage.plot, width=6, height=7.5, units='in',
   dpi=600)

#prepare coverage, mean length CI, and convergence
   data
mean.length <- lapply(files, function(x) subset(get.
   data(x, 'Mean.Len.CI'),effect.size==0.2 | effect.
   size==0.5 | effect.size==0.8))
pct.miss <- lapply(files, function(x) subset(get.
   data.1(x, 'PctMiss'), effect.size==0.2 | effect.
   size==0.5 | effect.size==0.8))
conv <- lapply(files, function(x) subset(get.data.1(
   x, 'CONV'), effect.size==0.2 | effect.size==0.5 |
    effect.size==0.8))
coverage.wide <- lapply(coverage, function(x) dcast(
   x, effect.size + method ~ time, value.var='
   Coverage'))
mean.length.wide <- lapply(mean.length, function(x)
   dcast(x, effect.size + method ~ time, value.var='
   Mean.Len.CI'))
for (i in names(mean.length.wide)) colnames(mean.
   length.wide[[i]]) <- c('effect.size', 'method', '
   Length1', 'Length2', 'Lengt3', 'Length4')
coverage.final <- lapply(dir.name, function(i) {
                tmp <- merge(x=coverage.wide[[i]], y
                   =mean.length.wide[[i]], by=c('
                   effect.size', 'method'))
                tmp <- merge(x=tmp, y=pct.miss[[i]],
                    by=c('effect.size', 'method'))
                tmp <- merge(x=tmp, y=conv[[i]], by=
                   c('effect.size', 'method'))
                tmp <- tmp[order(tmp$effect.size,
                   tmp$method),]
                tmp
                })
names(coverage.final) <- dir.name
coverage.final.df <- data.frame()
for (i in names(coverage.final)) {
        coverage.final[[i]]$sample.size <- i
        coverage.final.df <- rbind(coverage.final.df
           , coverage.final[[i]])
}

#save to xlsx file
```

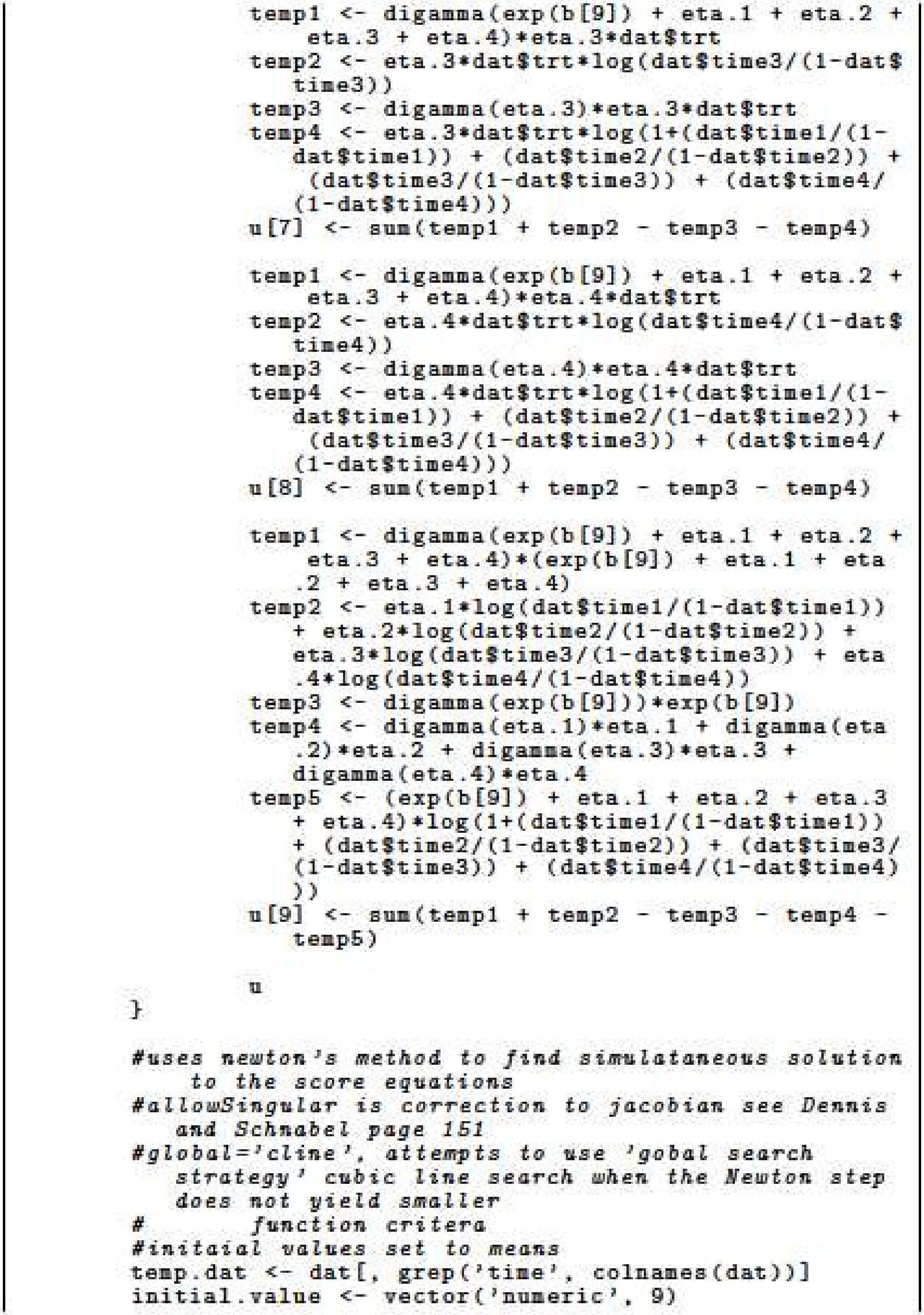
```
        write.xlsx(coverage.final.df, file=paste0(dir, './
           results.xlsx'), sheetName='Coverage', row.names=
           FALSE, append=FALSE)
        write.xlsx(bias.df, file=paste0(dir, './results.xlsx
           '), sheetName='Bias', row.names=FALSE, append=
           TRUE)
}

sim_2grp_compile_aggregate <- function(dir) {

        #directories of results
        dir.name <- c('N_12', 'N_30', 'N_50', 'N_100')

        #read in files
        tmp.files <- lapply(dir.name, function(x) get(load(
           paste0(dir, '/', x, './files.Rdata'))))
        names(tmp.files) <- dir.name

        #aggregate the results
        files <- lapply(tmp.files, function(i) lapply(i,
           function(x) x$aggregate))
        effect_size <- paste0('effect_',seq(0,1,.1))
        for (i in names(files)) names(files[[i]]) <- effect_
           size

        #get type I error and power data
        power <- lapply(files, function(x) power.data(x))
        power.df <- data.frame()
        for (i in names(power)) {
                power[[i]]$sample.size <- i
                power.df <- rbind(power.df, power[[i]])
        }
        power.df$sample.size <- factor(power.df$sample.size,
            levels=c('N_12', 'N_30', 'N_50', 'N_100'))

        #create power plots
        power.plot <- ggplot(data=power.df, aes(x=effect.
           size, y=power, color=method, linetype=method,
           shape=method)) + geom_point() + geom_line() +
                theme_bw() + theme(panel.grid=element_blank
                   (), legend.background=element_rect(color=
                   'black', linetype=1, size=.3),
                legend.position='bottom', legend.direction='
                   horizontal') + labs(x='Effect␣Size', y='
                   Power', color='Model', linetype='Model',
                   shape='Model') +
                scale_x_continuous(lim=c(0,1), breaks=seq
                   (0,1,.1)) + scale_y_continuous(lim=c(0,1)
                   , breaks=seq(0,1,.1)) +
                geom_hline(yintercept=0.05, linetype=2) +
                   facet_wrap(~sample.size, ncol=2)

        #save plot
        ggsave(filename=paste0(dir, './power.png'), power.
           plot, width=6, height=6, units='in', dpi=600)
```

```
#get mean bias data
bias.df.tmp <- lapply(files, function(x) subset(get.
   data_2grp(x, 'bias'), (effect.size==0.2 | effect.
   size==0.5 | effect.size==0.8) & trt==2))
bias.df <- data.frame()
for (i in names(bias.df.tmp)) {
        bias.df.tmp[[i]]$sample.size <- i
        bias.df <- rbind(bias.df, bias.df.tmp[[i]])
}
bias.df$sample.size <- factor(bias.df$sample.size,
   levels=c('N_12', 'N_30', 'N_50', 'N_100'))
bias.df$effect.size <- factor(bias.df$effect.size,
   levels=c(0.2, 0.5, 0.8), labels=c('Small␣effect␣
   size', 'Medium␣effect␣size', 'Large␣effect␣size')
   )

#create mean bias plots
bias.plot <- ggplot(bias.df, aes(x=time, y=bias,
   color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(), legend.
           background=element_rect(color='black',
           linetype=1, size=.3),
        legend.position='bottom', legend.direction='
           horizontal') +
        labs(x=NULL, y='Mean␣Bias', color='Model',
           shape='Model') + geom_hline(yintercept=0,
            linetype=2) +
        facet_grid(sample.size˜effect.size)

#save plot
ggsave(filename=paste0(dir, './bias.png'), bias.plot
   , width=6, height=7.5, units='in', dpi=600)

#get rmsd data
rmsd <- lapply(files, function(x) subset(get.data_2
   grp(x, 'RMSD'), (effect.size==0.2 | effect.size
   ==0.5 | effect.size==0.8) & trt==2))
rmsd.df <- data.frame()
for (i in names(rmsd)) {
        rmsd[[i]]$sample.size <- i
        rmsd.df <- rbind(rmsd.df, rmsd[[i]])
}
rmsd.df$sample.size <- factor(rmsd.df$sample.size,
   levels=c('N_12', 'N_30', 'N_50', 'N_100'))
rmsd.df$effect.size <- factor(rmsd.df$effect.size,
   levels=c(0.2, 0.5, 0.8), labels=c('Small␣effect␣
   size', 'Medium␣effect␣size', 'Large␣effect␣size')
   )

#creat rmsd plot
rmsd.plot <- ggplot(rmsd.df, aes(x=time, y=RMSD,
   color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(), legend.
           background=element_rect(color='black',
           linetype=1, size=.3),
        legend.position='bottom', legend.direction='
           horizontal')+
        labs(x=NULL, y='RMSD', color='Model', shape=
           'Model') + geom_hline(yintercept=0,
           linetype=2) +
        facet_grid(sample.size~effect.size)

#save plot
ggsave(filename=paste0(dir, './rmsd.png'), rmsd.plot
   , width=6, height=7.5, units='in', dpi=600)

coverage <- lapply(files, function(x) subset(get.
   data_2grp(x, 'Coverage'), (effect.size==0.2 |
   effect.size==0.5 | effect.size==0.8) & trt==2))
coverage.df <- data.frame()
for (i in names(coverage)) {
        coverage[[i]]$sample.size <- i
        coverage.df <- rbind(coverage.df, coverage[[
           i]])
}

#get coverage data
coverage.df$sample.size <- factor(coverage.df$sample
   .size, levels=c('N_12', 'N_30', 'N_50', 'N_100'))
coverage.df$effect.size <- factor(coverage.df$effect
   .size, levels=c(0.2, 0.5, 0.8), labels=c('Small␣
   effect␣size', 'Medium␣effect␣size', 'Large␣effect
   ␣size'))

#create coverage plot
coverage.plot <- ggplot(coverage.df, aes(x=time, y=
   Coverage, color=method, shape=method)) +
        geom_point(position=position_dodge(width
           =0.25)) + theme_bw() +
        theme(panel.grid=element_blank(), legend.
           background=element_rect(color='black',
           linetype=1, size=.3),
        legend.position='bottom', legend.direction='
           horizontal')+
        scale_y_continuous(lim=c(0.9,1), breaks=seq
           (.9,1, .01), labels=c(.90,rep('',4),.95,
           rep('',4),1.00)) +
        geom_hline(yintercept=.95, linetype=2) +
           labs(x=NULL, y='Coverage␣Probability',
           color='Model', shape='Model') +
        facet_grid(sample.size~effect.size)

#save plot
ggsave(filename=paste0(dir, './coverage.png'),
   coverage.plot, width=6, height=7.5, units='in',
           dpi=600)

        #prepare coverage, mean length CI, and convergence
           data
        mean.length <- lapply(files, function(x) subset(get.
           data_2grp(x, 'Mean.Len.CI'), (effect.size==0.2 |
           effect.size==0.5 | effect.size==0.8) & trt==2))
        pct.miss <- lapply(files, function(x) subset(get.
           data.1(x, 'PctMiss'), effect.size==0.2 | effect.
           size==0.5 | effect.size==0.8))
        conv <- lapply(files, function(x) subset(get.data.1(
           x, 'CONV'), effect.size==0.2 | effect.size==0.5 |
            effect.size==0.8))
        coverage.wide <- lapply(coverage, function(x) dcast(
           x, effect.size + method ~ time, value.var='
           Coverage'))
        mean.length.wide <- lapply(mean.length, function(x)
           dcast(x, effect.size + method ~ time, value.var='
           Mean.Len.CI'))
        for (i in names(mean.length.wide)) colnames(mean.
           length.wide[[i]]) <- c('effect.size', 'method', '
           Length1', 'Length2', 'Lengt3', 'Length4')
        coverage.final <- lapply(dir.name, function(i) {
                        tmp <- merge(x=coverage.wide[[i]], y
                           =mean.length.wide[[i]], by=c('
                           effect.size', 'method'))
                        tmp <- merge(x=tmp, y=pct.miss[[i]],
                            by=c('effect.size', 'method'))
                        tmp <- merge(x=tmp, y=conv[[i]], by=
                           c('effect.size', 'method'))
                        tmp <- tmp[order(tmp$effect.size,
                           tmp$method),]
                        tmp
                        })
        names(coverage.final) <- dir.name
        coverage.final.df <- data.frame()
        for (i in names(coverage.final)) {
                coverage.final[[i]]$sample.size <- i
                coverage.final.df <- rbind(coverage.final.df
                   , coverage.final[[i]])
        }

        #save to xlsx file
        write.xlsx(coverage.final.df, file=paste0(dir, './
           results.xlsx'), sheetName='Coverage', row.names=
           FALSE, append=FALSE)
        write.xlsx(bias.df, file=paste0(dir, './results.xlsx
           '), sheetName='Bias', row.names=FALSE, append=
           TRUE)
}
```

dissertation_LNMVB_jacobian.R



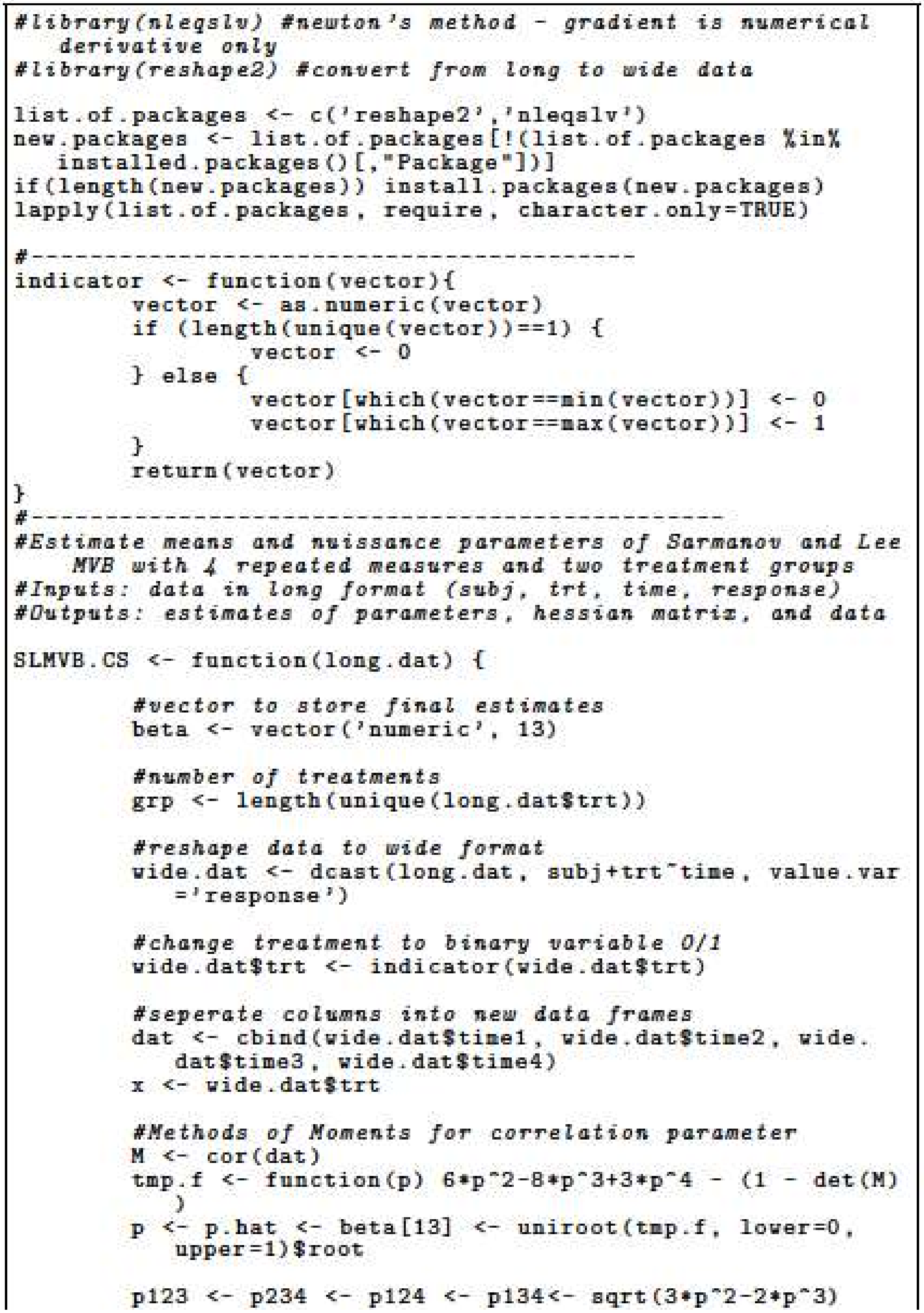
```
#library(nleqslv) #newton's method - gradient is numerical
   derivative only
#library(reshape2) #long to wide format

list.of.packages <- c('reshape2','nleqslv')
new.packages <- list.of.packages[!(list.of.packages %in%
   installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
lapply(list.of.packages, require, character.only=TRUE)

#-----------------------------------------
indicator <- function(vector){
        vector <- as.numeric(vector)
        if (length(unique(vector))==1) {
                vector <- 0
        } else {
                vector[which(vector==min(vector))] <- 0
                vector[which(vector==max(vector))] <- 1
        }
        return(vector)
}
#-----------------------------------------------------------
#Estimates means and alpha.0 of Libby and Novick MVB with 4
   repeated measures and two treatment groups
#Inputs: data in long format (subj, trt, time, response)
#Outputs: estimates of means and alpha.0 as a vector

LNMVB <- function(long.dat) {
        #vector to store final estimates
        beta <- vector('numeric', 9)

        #number of treatments
        grp <- length(unique(long.dat$trt))

        #reshape data to wide format
        dat <- dcast(long.dat, subj+trt~time, value.var='
           response')

        #change treatment to binary variable 0/1
        dat$trt <- indicator(dat$trt)


        score <- function(b) {
                u <- numeric(9)
                eta.1 <- exp(b[1] + b[5]*dat$trt + b[9])
                eta.2 <- exp(b[2] + b[6]*dat$trt + b[9])
                eta.3 <- exp(b[3] + b[7]*dat$trt + b[9])
                eta.4 <- exp(b[4] + b[8]*dat$trt + b[9])
                temp1 <- digamma(exp(b[9]) + eta.1 + eta.2 +
                    eta.3 + eta.4)*eta.1
                temp2 <- eta.1*log(dat$time1/(1-dat$time1))
                temp3 <- digamma(eta.1)*eta.1
                temp4 <- eta.1*log(1+(dat$time1/(1-dat$time1
                   )) + (dat$time2/(1-dat$time2)) + (dat$
                   time3/(1-dat$time3)) + (dat$time4/(1-dat$
   time4)))
u[1] <- sum(temp1 + temp2 - temp3 - temp4)

temp1 <- digamma(exp(b[9]) + eta.1 + eta.2 +
    eta.3 + eta.4)*eta.2
temp2 <- eta.2*log(dat$time2/(1-dat$time2))
temp3 <- digamma(eta.2)*eta.2
temp4 <- eta.2*log(1+(dat$time1/(1-dat$time1
   )) + (dat$time2/(1-dat$time2)) + (dat$
   time3/(1-dat$time3)) + (dat$time4/(1-dat$
   time4)))
u[2] <- sum(temp1 + temp2 - temp3 - temp4)

temp1 <- digamma(exp(b[9]) + eta.1 + eta.2 +
    eta.3 + eta.4)*eta.3
temp2 <- eta.3*log(dat$time3/(1-dat$time3))
temp3 <- digamma(eta.3)*eta.3
temp4 <- eta.3*log(1+(dat$time1/(1-dat$time1
   )) + (dat$time2/(1-dat$time2)) + (dat$
   time3/(1-dat$time3)) + (dat$time4/(1-dat$
   time4)))
u[3] <- sum(temp1 + temp2 - temp3 - temp4)

temp1 <- digamma(exp(b[9]) + eta.1 + eta.2 +
    eta.3 + eta.4)*eta.4
temp2 <- eta.4*log(dat$time4/(1-dat$time4))
temp3 <- digamma(eta.4)*eta.4
temp4 <- eta.4*log(1+(dat$time1/(1-dat$time1
   )) + (dat$time2/(1-dat$time2)) + (dat$
   time3/(1-dat$time3)) + (dat$time4/(1-dat$
   time4)))
u[4] <- sum(temp1 + temp2 - temp3 - temp4)

temp1 <- digamma(exp(b[9]) + eta.1 + eta.2 +
    eta.3 + eta.4)*eta.1*dat$trt
temp2 <- eta.1*dat$trt*log(dat$time1/(1-dat$
   time1))
temp3 <- digamma(eta.1)*eta.1*dat$trt
temp4 <- eta.1*dat$trt*log(1+(dat$time1/(1-
   dat$time1)) + (dat$time2/(1-dat$time2)) +
    (dat$time3/(1-dat$time3)) + (dat$time4/
   (1-dat$time4)))
u[5] <- sum(temp1 + temp2 - temp3 - temp4)

temp1 <- digamma(exp(b[9]) + eta.1 + eta.2 +
    eta.3 + eta.4)*eta.2*dat$trt
temp2 <- eta.2*dat$trt*log(dat$time2/(1-dat$
   time2))
temp3 <- digamma(eta.2)*eta.2*dat$trt
temp4 <- eta.2*dat$trt*log(1+(dat$time1/(1-
   dat$time1)) + (dat$time2/(1-dat$time2)) +
    (dat$time3/(1-dat$time3)) + (dat$time4/
   (1-dat$time4)))
u[6] <- sum(temp1 + temp2 - temp3 - temp4)
```



```
      temp1 <- digamma(exp(b[9]) + eta.1 + eta.2 +
          eta.3 + eta.4)*eta.3*dat$trt
      temp2 <- eta.3*dat$trt*log(dat$time3/(1-dat$
         time3))
      temp3 <- digamma(eta.3)*eta.3*dat$trt
      temp4 <- eta.3*dat$trt*log(1+(dat$time1/(1-
         dat$time1)) + (dat$time2/(1-dat$time2)) +
          (dat$time3/(1-dat$time3)) + (dat$time4/
         (1-dat$time4)))
      u[7] <- sum(temp1 + temp2 - temp3 - temp4)

      temp1 <- digamma(exp(b[9]) + eta.1 + eta.2 +
          eta.3 + eta.4)*eta.4*dat$trt
      temp2 <- eta.4*dat$trt*log(dat$time4/(1-dat$
         time4))
      temp3 <- digamma(eta.4)*eta.4*dat$trt
      temp4 <- eta.4*dat$trt*log(1+(dat$time1/(1-
         dat$time1)) + (dat$time2/(1-dat$time2)) +
          (dat$time3/(1-dat$time3)) + (dat$time4/
         (1-dat$time4)))
      u[8] <- sum(temp1 + temp2 - temp3 - temp4)

      temp1 <- digamma(exp(b[9]) + eta.1 + eta.2 +
          eta.3 + eta.4)*(exp(b[9]) + eta.1 + eta
         .2 + eta.3 + eta.4)
      temp2 <- eta.1*log(dat$time1/(1-dat$time1))
         + eta.2*log(dat$time2/(1-dat$time2)) +
         eta.3*log(dat$time3/(1-dat$time3)) + eta
         .4*log(dat$time4/(1-dat$time4))
      temp3 <- digamma(exp(b[9]))*exp(b[9])
      temp4 <- digamma(eta.1)*eta.1 + digamma(eta
         .2)*eta.2 + digamma(eta.3)*eta.3 +
         digamma(eta.4)*eta.4
      temp5 <- (exp(b[9]) + eta.1 + eta.2 + eta.3
         + eta.4)*log(1+(dat$time1/(1-dat$time1))
         + (dat$time2/(1-dat$time2)) + (dat$time3/
         (1-dat$time3)) + (dat$time4/(1-dat$time4)
         ))
      u[9] <- sum(temp1 + temp2 - temp3 - temp4 -
         temp5)

      u
}

#uses newton's method to find simulataneous solution
    to the score equations
#allowSingular is correction to jacobian see Dennis
   and Schnabel page 151
#global='cline', attempts to use 'gobal search
   strategy' cubic line search when the Newton step
   does not yield smaller
#       function critera
#initaial values set to means
temp.dat <- dat[, grep('time', colnames(dat))]
initial.value <- vector('numeric', 9)
```

```
        for (i in 1:ncol(temp.dat)) initial.value[i] <-
           logit(mean(temp.dat[,i]))

        #quasi-newton-raphson algorithm
        estimate <- nleqslv(initial.value, score, method='
           Newton', jacobian=TRUE, global='cline', control=
           list(allowSingular=TRUE, ftol=1e-8, maxit=20))

        #determine convergence
        converged <- 0
        if (estimate$termcd == 1) converged <- 1

        if (converged == 0) {
                initial.value <- estimate$x
                estimate <- nleqslv(initial.value, score,
                   method='Newton', jacobian=TRUE, global='
                   cline', xscalm='auto', control=list(
                   allowSingular=TRUE, ftol=1e-8))
                if (estimate$termcd == 1) converged <- 1
        }

        beta <- estimate$x
        H <- estimate$jac

        if (grp == 1) {
                beta <- beta[c(1:4,9)]
                names(beta) <- c('b1', 'b2', 'b3', 'b4', 'c'
                   )
                hessian <- H[c(1:4,9),c(1:4,9)]
                rownames(hessian) <- colnames(hessian)
                return(list(beta=beta, hessian=hessian, dat=
                   long.dat, converged=converged))
        } else {
                names(beta) <- c('b1', 'b2', 'b3', 'b4','b5'
                   , 'b6', 'b7', 'b8','c')
                hessian <- H
                rownames(hessian) <- colnames(hessian)
                return(list(beta=beta, hessian=hessian, dat=
                   long.dat, converged=converged))
        }
}
```

dissertation_SLMVB_CS_jacobian.R



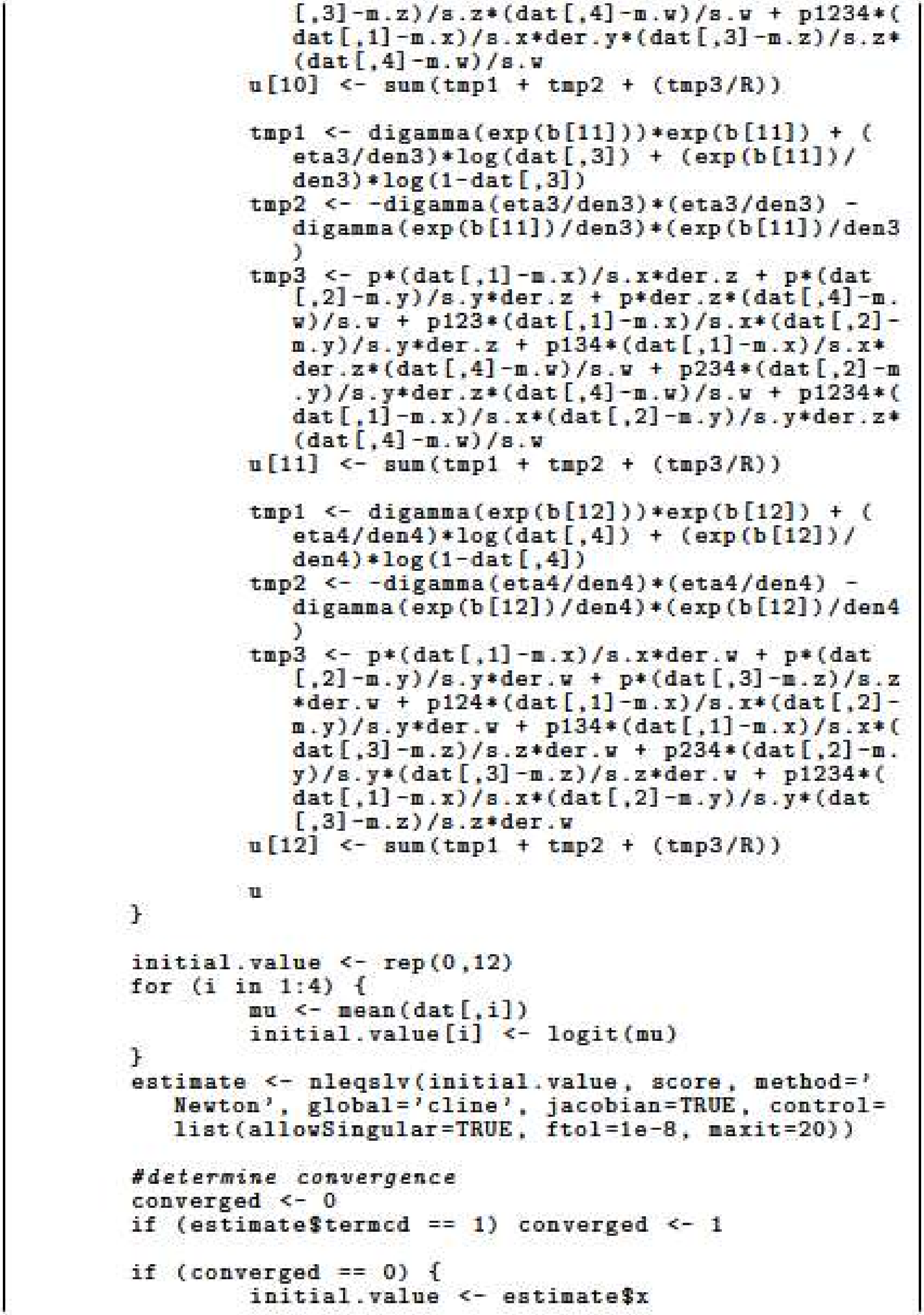
```
#library(nleqslv) #newton's method - gradient is numerical
   derivative only
#library(reshape2) #convert from long to wide data

list.of.packages <- c('reshape2','nleqslv')
new.packages <- list.of.packages[!(list.of.packages %in%
   installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
lapply(list.of.packages, require, character.only=TRUE)

#------------------------------------------
indicator <- function(vector){
        vector <- as.numeric(vector)
        if (length(unique(vector))==1) {
                vector <- 0
        } else {
                vector[which(vector==min(vector))] <- 0
                vector[which(vector==max(vector))] <- 1
        }
        return(vector)
}
#------------------------------------------------
#Estimate means and nuissance parameters of Sarmanov and Lee
    MVB with 4 repeated measures and two treatment groups
#Inputs: data in long format (subj, trt, time, response)
#Outputs: estimates of parameters, hessian matrix, and data

SLMVB.CS <- function(long.dat) {

        #vector to store final estimates
        beta <- vector('numeric', 13)

        #number of treatments
        grp <- length(unique(long.dat$trt))

        #reshape data to wide format
        wide.dat <- dcast(long.dat, subj+trt~time, value.var
           ='response')

        #change treatment to binary variable 0/1
        wide.dat$trt <- indicator(wide.dat$trt)

        #seperate columns into new data frames
        dat <- cbind(wide.dat$time1, wide.dat$time2, wide.
           dat$time3, wide.dat$time4)
        x <- wide.dat$trt

        #Methods of Moments for correlation parameter
        M <- cor(dat)
        tmp.f <- function(p) 6*p^2-8*p^3+3*p^4 - (1 - det(M)
           )
        p <- p.hat <- beta[13] <- uniroot(tmp.f, lower=0,
           upper=1)$root

        p123 <- p234 <- p124 <- p134<- sqrt(3*p^2-2*p^3)
```

```
p1234 <- sqrt(6*p^2-8*p^3+3*p^4)

score <- function(b) {
        u <- numeric(12)

        eta1 <- exp(b[1] + b[5]*x + b[9])
        den1 <- 1 + exp(b[1] + b[5]*x)
        eta2 <- exp(b[2] + b[6]*x + b[10])
        den2 <- 1 + exp(b[2] + b[6]*x)
        eta3 <- exp(b[3] + b[7]*x + b[11])
        den3 <- 1 + exp(b[3] + b[7]*x)
        eta4 <- exp(b[4] + b[8]*x + b[12])
        den4 <- 1 + exp(b[4] + b[8]*x)
        s.x <- (1+exp(b[1] + b[5]*x))^(-1)*sqrt(exp(
           b[1] + b[5]*x)/(1+exp(b[9])))
        m.x <- exp(b[1] + b[5]*x)/(1+exp(b[1] + b[5]
           *x))
        der.x <- 0.5*(dat[,1]*(exp(b[1] + b[5]*x)-1)
           -exp(b[1] + b[5]*x)) * (exp(b[1] + b[5]*x
           )/(1+exp(b[9])))^(-1/2)
        s.y <- (1+exp(b[2] + b[6]*x))^(-1)*sqrt(exp(
           b[2] + b[6]*x)/(1+exp(b[10])))
        m.y <- exp(b[2] + b[6]*x)/(1+exp(b[2] + b[6]
           *x))
        der.y <- 0.5*(dat[,2]*(exp(b[2] + b[6]*x)-1)
           -exp(b[2] + b[6]*x)) * (exp(b[2] + b[6]*x
           )/(1+exp(b[10])))^(-1/2)
        s.z <- (1+exp(b[3] + b[7]*x))^(-1)*sqrt(exp(
           b[3] + b[7]*x)/(1+exp(b[11])))
        m.z <- exp(b[3] + b[7]*x)/(1+exp(b[3] + b[7]
           *x))
        der.z <- 0.5*(dat[,3]*(exp(b[3] + b[7]*x)-1)
           -exp(b[3] + b[7]*x)) * (exp(b[3] + b[7]*x
           )/(1+exp(b[11])))^(-1/2)
        s.w <- (1+exp(b[4] + b[8]*x))^(-1)*sqrt(exp(
           b[4] + b[8]*x)/(1+exp(b[12])))
        m.w <- exp(b[4] + b[8]*x)/(1+exp(b[4] + b[8]
           *x))
        der.w <- 0.5*(dat[,4]*(exp(b[4] + b[8]*x)-1)
           -exp(b[4] + b[8]*x)) * (exp(b[4] + b[8]*x
           )/(1+exp(b[12])))^(-1/2)
        R <- 1 + p*(dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s
           .y + p*(dat[,1]-m.x)/s.x*(dat[,3]-m.z)/s.
           z + p*(dat[,1]-m.x)/s.x*(dat[,4]-m.w)/s.w
            + p*(dat[,2]-m.y)/s.y*(dat[,3]-m.z)/s.z
           + p*(dat[,2]-m.y)/s.y*(dat[,4]-m.w)/s.w +
            p*(dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w +
           p123*(dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*
           (dat[,3]-m.z)/s.z + p124*(dat[,1]-m.x)/s.
           x*(dat[,2]-m.y)/s.y*(dat[,4]-m.w)/s.w +
           p134*(dat[,1]-m.x)/s.x*(dat[,3]-m.z)/s.z*
           (dat[,4]-m.w)/s.w + p234*(dat[,2]-m.y)/s.
           y*(dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w +
           p1234*(dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y
           *(dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w
```

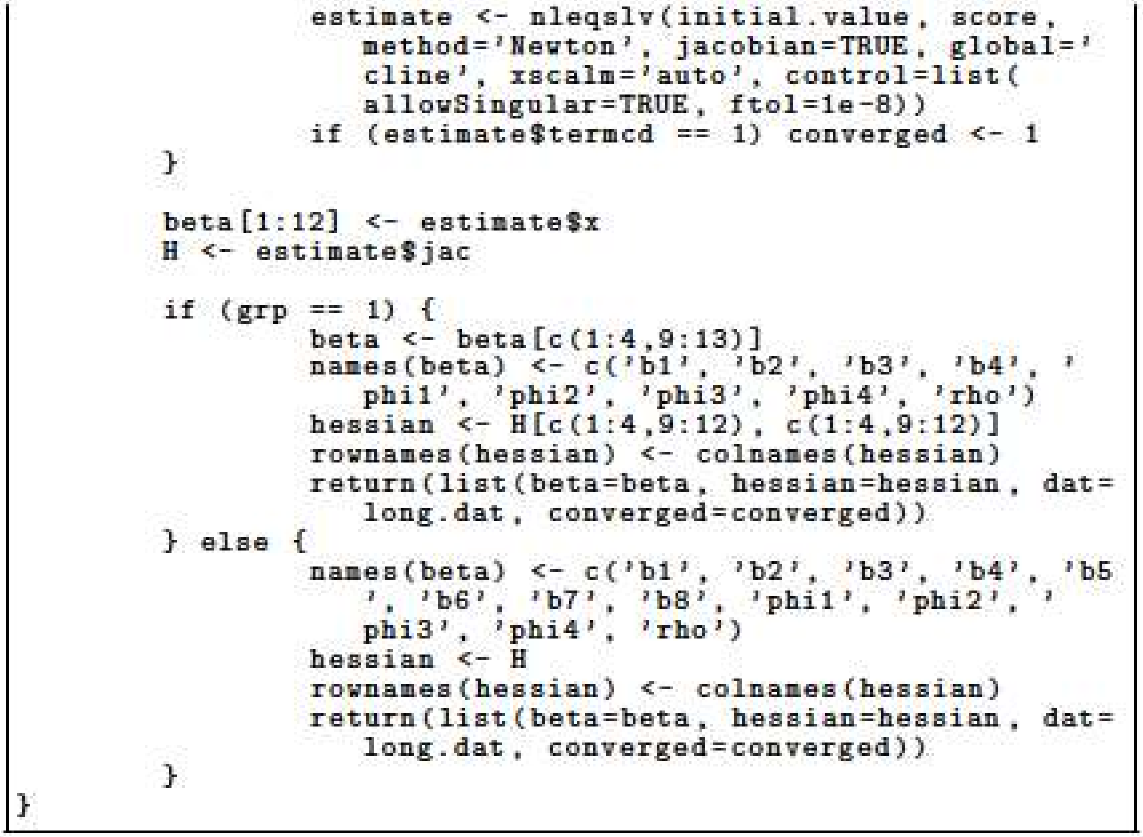
```
tmp1 <- digamma(exp(b[9])/den1)*(eta1/den1
   ^2) + (eta1/den1^2)*log(dat[,1]/(1-dat
   [,1]))
tmp2 <- -digamma(eta1/den1)*(eta1/den1^2)
tmp3 <- p*der.x*(dat[,2]-m.y)/s.y + p*der.x*
   (dat[,3]-m.z)/s.z + p*der.x*(dat[,4]-m.w)
   /s.w + p123*der.x*(dat[,2]-m.y)/s.y*(dat
   [,3]-m.z)/s.z + p124*der.x*(dat[,2]-m.y)/
   s.y*(dat[,4]-m.w)/s.w + p134*der.x*(dat
   [,3]-m.z)/s.z*(dat[,4]-m.w)/s.w + p1234*
   der.x*(dat[,2]-m.y)/s.y*(dat[,3]-m.z)/s.z
   *(dat[,4]-m.w)/s.w
u[1] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[10])/den2)*(eta2/den2
   ^2) + (eta2/den2^2)*log(dat[,2]/(1-dat
   [,2]))
tmp2 <- -digamma(eta2/den2)*(eta2/den2^2)
tmp3 <- p*(dat[,1]-m.x)/s.x*der.y + p*der.y*
   (dat[,3]-m.z)/s.z + p*der.y*(dat[,4]-m.w)
   /s.w + p123*(dat[,1]-m.x)/s.x*der.y*(dat
   [,3]-m.z)/s.z + p124*(dat[,1]-m.x)/s.x*
   der.y*(dat[,4]-m.w)/s.w + p234*der.y*(dat
   [,3]-m.z)/s.z*(dat[,4]-m.w)/s.w + p1234*(
   dat[,1]-m.x)/s.x*der.y*(dat[,3]-m.z)/s.z*
   (dat[,4]-m.w)/s.w
u[2] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[11])/den3)*(eta3/den3
   ^2) + (eta3/den3^2)*log(dat[,3]/(1-dat
   [,3]))
tmp2 <- -digamma(eta3/den3)*(eta3/den3^2)
tmp3 <- p*(dat[,1]-m.x)/s.x*der.z + p*(dat
   [,2]-m.y)/s.y*der.z + p*der.z*(dat[,4]-m.
   w)/s.w + p123*(dat[,1]-m.x)/s.x*(dat[,2]-
   m.y)/s.y*der.z + p134*(dat[,1]-m.x)/s.x*
   der.z*(dat[,4]-m.w)/s.w + p234*(dat[,2]-m
   .y)/s.y*der.z*(dat[,4]-m.w)/s.w + p1234*(
   dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*der.z*
   (dat[,4]-m.w)/s.w
u[3] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[12])/den4)*(eta4/den4
   ^2) + (eta4/den4^2)*log(dat[,4]/(1-dat
   [,4]))
tmp2 <- -digamma(eta4/den4)*(eta4/den4^2)
tmp3 <- p*(dat[,1]-m.x)/s.x*der.w + p*(dat
   [,2]-m.y)/s.y*der.w + p*(dat[,3]-m.z)/s.z
   *der.w + p124*(dat[,1]-m.x)/s.x*(dat[,2]-
   m.y)/s.y*der.w + p134*(dat[,1]-m.x)/s.x*(
   dat[,3]-m.z)/s.z*der.w + p234*(dat[,2]-m.
   y)/s.y*(dat[,3]-m.z)/s.z*der.w + p1234*(
   dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*(dat
   [,3]-m.z)/s.z*der.w
```

```
u[4] <- sum(tmp1 + tmp2 + (tmp3/R))

der.x <- 0.5*x*(dat[,1]*(exp(b[1] + b[5]*x)
   -1)-exp(b[1] + b[5]*x)) * (exp(b[1] + b
   [5]*x)/(1+exp(b[9])))^(-1/2)
der.y <- 0.5*x*(dat[,2]*(exp(b[2] + b[6]*x)
   -1)-exp(b[2] + b[6]*x)) * (exp(b[2] + b
   [6]*x)/(1+exp(b[10])))^(-1/2)
der.z <- 0.5*x*(dat[,3]*(exp(b[3] + b[7]*x)
   -1)-exp(b[3] + b[7]*x)) * (exp(b[3] + b
   [7]*x)/(1+exp(b[11])))^(-1/2)
der.w <- 0.5*x*(dat[,4]*(exp(b[4] + b[8]*x)
   -1)-exp(b[4] + b[8]*x)) * (exp(b[4] + b
   [8]*x)/(1+exp(b[12])))^(-1/2)

tmp1 <- digamma(exp(b[9])/den1)*(eta1/den1
   ^2)*x + (eta1/den1^2)*log(dat[,1]/(1-dat
   [,1]))*x
tmp2 <- -digamma(eta1/den1)*(eta1/den1^2)*x
tmp3 <- p*der.x*(dat[,2]-m.y)/s.y + p*der.x*
   (dat[,3]-m.z)/s.z + p*der.x*(dat[,4]-m.w)
   /s.w + p123*der.x*(dat[,2]-m.y)/s.y*(dat
   [,3]-m.z)/s.z + p124*der.x*(dat[,2]-m.y)/
   s.y*(dat[,4]-m.w)/s.w + p134*der.x*(dat
   [,3]-m.z)/s.z*(dat[,4]-m.w)/s.w + p1234*
   der.x*(dat[,2]-m.y)/s.y*(dat[,3]-m.z)/s.z
   *(dat[,4]-m.w)/s.w
u[5] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[10])/den2)*(eta2/den2
   ^2)*x + (eta2/den2^2)*log(dat[,2]/(1-dat
   [,2]))*x
tmp2 <- -digamma(eta2/den2)*(eta2/den2^2)*x
tmp3 <- p*(dat[,1]-m.x)/s.x*der.y + p*der.y*
   (dat[,3]-m.z)/s.z + p*der.y*(dat[,4]-m.w)
   /s.w + p123*(dat[,1]-m.x)/s.x*der.y*(dat
   [,3]-m.z)/s.z + p124*(dat[,1]-m.x)/s.x*
   der.y*(dat[,4]-m.w)/s.w + p234*der.y*(dat
   [,3]-m.z)/s.z*(dat[,4]-m.w)/s.w + p1234*(
   dat[,1]-m.x)/s.x*der.y*(dat[,3]-m.z)/s.z*
   (dat[,4]-m.w)/s.w
u[6] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[11])/den3)*(eta3/den3
   ^2)*x + (eta3/den3^2)*log(dat[,3]/(1-dat
   [,3]))*x
tmp2 <- -digamma(eta3/den3)*(eta3/den3^2)*x
tmp3 <- p*(dat[,1]-m.x)/s.x*der.z + p*(dat
   [,2]-m.y)/s.y*der.z + p*der.z*(dat[,4]-m.
   w)/s.w + p123*(dat[,1]-m.x)/s.x*(dat[,2]-
   m.y)/s.y*der.z + p134*(dat[,1]-m.x)/s.x*
   der.z*(dat[,4]-m.w)/s.w + p234*(dat[,2]-m
   .y)/s.y*der.z*(dat[,4]-m.w)/s.w + p1234*(
   dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*der.z*
   (dat[,4]-m.w)/s.w
```

```
u[7] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[12])/den4)*(eta4/den4
   ^2)*x + (eta4/den4^2)*log(dat[,4]/(1-dat
   [,4]))*x
tmp2 <- -digamma(eta4/den4)*(eta4/den4^2)*x
tmp3 <- p*(dat[,1]-m.x)/s.x*der.w + p*(dat
   [,2]-m.y)/s.y*der.w + p*(dat[,3]-m.z)/s.z
   *der.w + p124*(dat[,1]-m.x)/s.x*(dat[,2]-
   m.y)/s.y*der.w + p134*(dat[,1]-m.x)/s.x*(
   dat[,3]-m.z)/s.z*der.w + p234*(dat[,2]-m.
   y)/s.y*(dat[,3]-m.z)/s.z*der.w + p1234*(
   dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*(dat
   [,3]-m.z)/s.z*der.w
u[8] <- sum(tmp1 + tmp2 + (tmp3/R))

der.x <- 0.5*(dat[,1]-m.x)*den1*(exp(b[1] +
   b[5]*x)/(1+exp(b[9])))^(1/2)*exp(b[9])/
   exp(b[1] + b[5]*x)
der.y <- 0.5*(dat[,2]-m.y)*den2*(exp(b[2] +
   b[6]*x)/(1+exp(b[10])))^(1/2)*exp(b[10])/
   exp(b[2] + b[6]*x)
der.z <- 0.5*(dat[,3]-m.z)*den3*(exp(b[3] +
   b[7]*x)/(1+exp(b[11])))^(1/2)*exp(b[11])/
   exp(b[3] + b[7]*x)
der.w <- 0.5*(dat[,4]-m.w)*den4*(exp(b[4] +
   b[8]*x)/(1+exp(b[12])))^(1/2)*exp(b[12])/
   exp(b[4] + b[8]*x)

tmp1 <- digamma(exp(b[9]))*exp(b[9]) + (eta1
   /den1)*log(dat[,1]) + (exp(b[9])/den1)*
   log(1-dat[,1])
tmp2 <- -digamma(eta1/den1)*(eta1/den1) -
   digamma(exp(b[9])/den1)*(exp(b[9])/den1)
tmp3 <- p*der.x*(dat[,2]-m.y)/s.y + p*der.x*
   (dat[,3]-m.z)/s.z + p*der.x*(dat[,4]-m.w)
   /s.w + p123*der.x*(dat[,2]-m.y)/s.y*(dat
   [,3]-m.z)/s.z + p124*der.x*(dat[,2]-m.y)/
   s.y*(dat[,4]-m.w)/s.w + p134*der.x*(dat
   [,3]-m.z)/s.z*(dat[,4]-m.w)/s.w + p1234*
   der.x*(dat[,2]-m.y)/s.y*(dat[,3]-m.z)/s.z
   *(dat[,4]-m.w)/s.w
u[9] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[10]))*exp(b[10]) + (
   eta2/den2)*log(dat[,2]) + (exp(b[10])/
   den2)*log(1-dat[,2])
tmp2 <- -digamma(eta2/den2)*(eta2/den2) -
   digamma(exp(b[10])/den2)*(exp(b[10])/den2
   )
tmp3 <- p*(dat[,1]-m.x)/s.x*der.y + p*der.y*
   (dat[,3]-m.z)/s.z + p*der.y*(dat[,4]-m.w)
   /s.w + p123*(dat[,1]-m.x)/s.x*der.y*(dat
   [,3]-m.z)/s.z + p124*(dat[,1]-m.x)/s.x*
   der.y*(dat[,4]-m.w)/s.w + p234*der.y*(dat
```

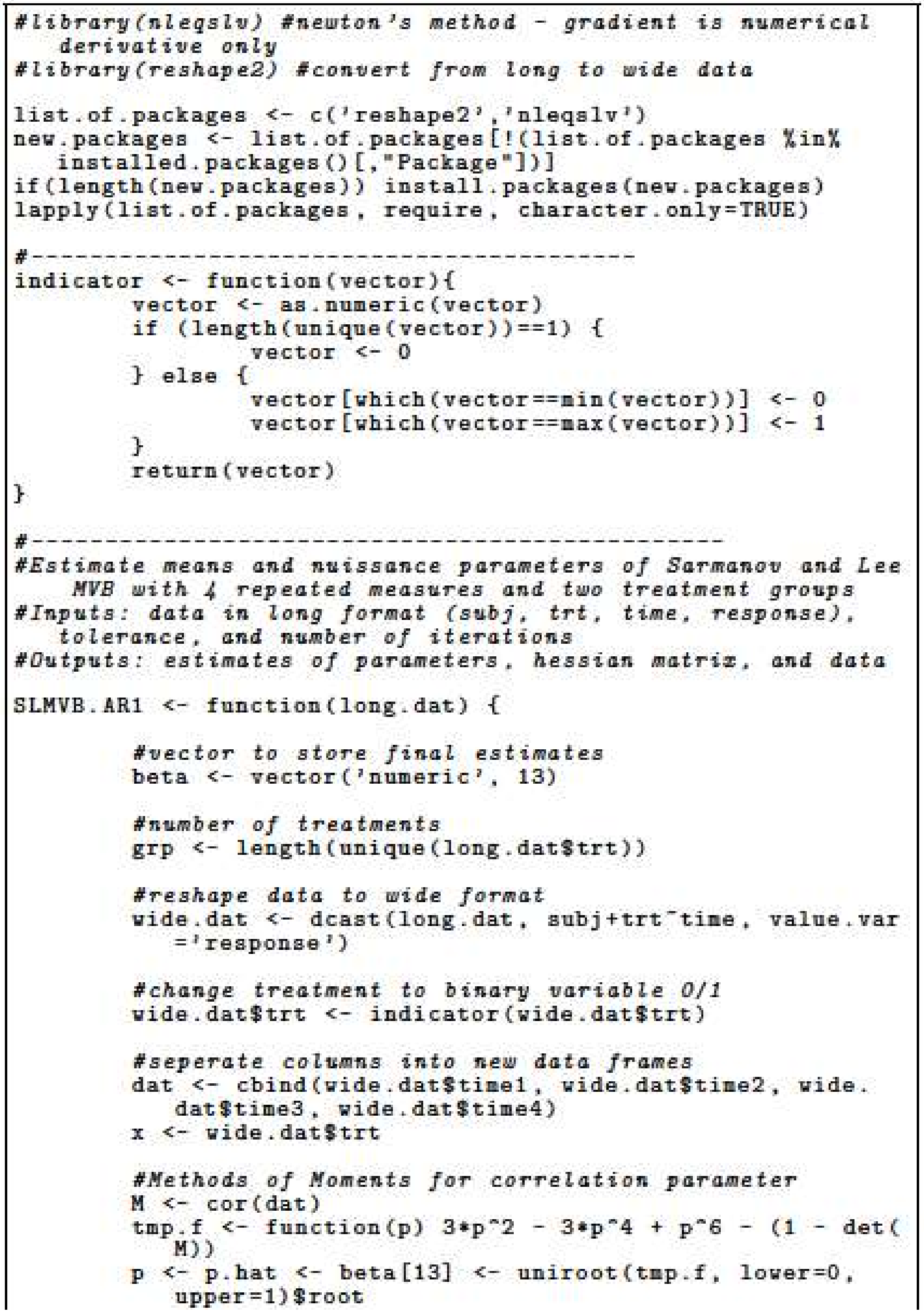


```
           [,3]-m.z)/s.z*(dat[,4]-m.w)/s.w + p1234*(
           dat[,1]-m.x)/s.x*der.y*(dat[,3]-m.z)/s.z*
           (dat[,4]-m.w)/s.w
        u[10] <- sum(tmp1 + tmp2 + (tmp3/R))

        tmp1 <- digamma(exp(b[11]))*exp(b[11]) + (
           eta3/den3)*log(dat[,3]) + (exp(b[11])/
           den3)*log(1-dat[,3])
        tmp2 <- -digamma(eta3/den3)*(eta3/den3) -
           digamma(exp(b[11])/den3)*(exp(b[11])/den3
           )
        tmp3 <- p*(dat[,1]-m.x)/s.x*der.z + p*(dat
           [,2]-m.y)/s.y*der.z + p*der.z*(dat[,4]-m.
           w)/s.w + p123*(dat[,1]-m.x)/s.x*(dat[,2]-
           m.y)/s.y*der.z + p134*(dat[,1]-m.x)/s.x*
           der.z*(dat[,4]-m.w)/s.w + p234*(dat[,2]-m
           .y)/s.y*der.z*(dat[,4]-m.w)/s.w + p1234*(
           dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*der.z*
           (dat[,4]-m.w)/s.w
        u[11] <- sum(tmp1 + tmp2 + (tmp3/R))

        tmp1 <- digamma(exp(b[12]))*exp(b[12]) + (
           eta4/den4)*log(dat[,4]) + (exp(b[12])/
           den4)*log(1-dat[,4])
        tmp2 <- -digamma(eta4/den4)*(eta4/den4) -
           digamma(exp(b[12])/den4)*(exp(b[12])/den4
           )
        tmp3 <- p*(dat[,1]-m.x)/s.x*der.w + p*(dat
           [,2]-m.y)/s.y*der.w + p*(dat[,3]-m.z)/s.z
           *der.w + p124*(dat[,1]-m.x)/s.x*(dat[,2]-
           m.y)/s.y*der.w + p134*(dat[,1]-m.x)/s.x*(
           dat[,3]-m.z)/s.z*der.w + p234*(dat[,2]-m.
           y)/s.y*(dat[,3]-m.z)/s.z*der.w + p1234*(
           dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*(dat
           [,3]-m.z)/s.z*der.w
        u[12] <- sum(tmp1 + tmp2 + (tmp3/R))

        u
}

initial.value <- rep(0,12)
for (i in 1:4) {
        mu <- mean(dat[,i])
        initial.value[i] <- logit(mu)
}
estimate <- nleqslv(initial.value, score, method='
   Newton', global='cline', jacobian=TRUE, control=
   list(allowSingular=TRUE, ftol=1e-8, maxit=20))

#determine convergence
converged <- 0
if (estimate$termcd == 1) converged <- 1

if (converged == 0) {
        initial.value <- estimate$x
```



```
                estimate <- nleqslv(initial.value, score,
                   method='Newton', jacobian=TRUE, global='
                   cline', xscalm='auto', control=list(
                   allowSingular=TRUE, ftol=1e-8))
                if (estimate$termcd == 1) converged <- 1
        }

        beta[1:12] <- estimate$x
        H <- estimate$jac

        if (grp == 1) {
                beta <- beta[c(1:4,9:13)]
                names(beta) <- c('b1', 'b2', 'b3', 'b4', '
                   phi1', 'phi2', 'phi3', 'phi4', 'rho')
                hessian <- H[c(1:4,9:12), c(1:4,9:12)]
                rownames(hessian) <- colnames(hessian)
                return(list(beta=beta, hessian=hessian, dat=
                   long.dat, converged=converged))
        } else {
                names(beta) <- c('b1', 'b2', 'b3', 'b4', 'b5
                   ', 'b6', 'b7', 'b8', 'phi1', 'phi2', '
                   phi3', 'phi4', 'rho')
                hessian <- H
                rownames(hessian) <- colnames(hessian)
                return(list(beta=beta, hessian=hessian, dat=
                   long.dat, converged=converged))
        }
}
```

dissertation_SLMVB_AR1_jacobian.R



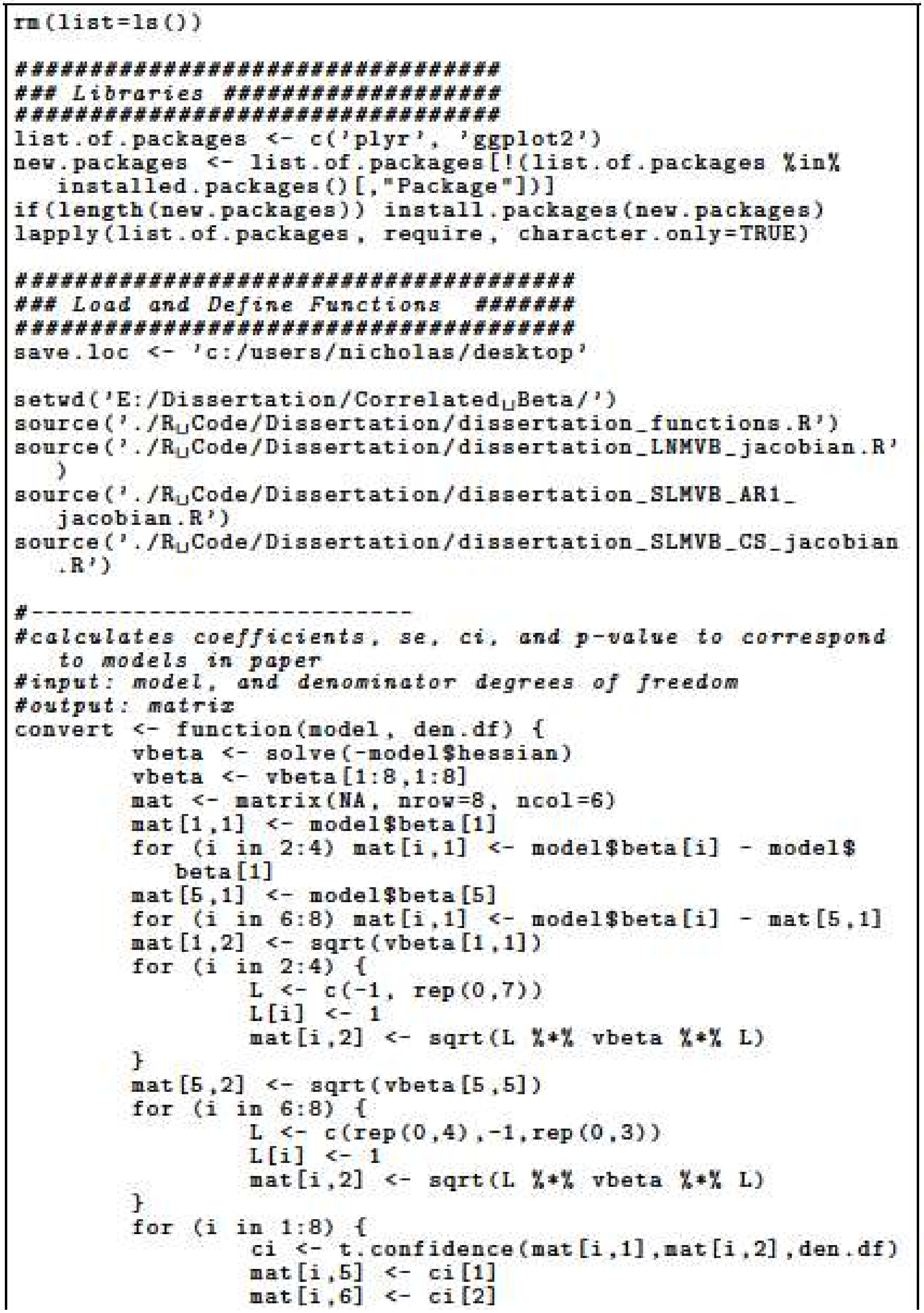
```
#library(nleqslv) #newton's method - gradient is numerical
   derivative only
#library(reshape2) #convert from long to wide data

list.of.packages <- c('reshape2','nleqslv')
new.packages <- list.of.packages[!(list.of.packages %in%
   installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
lapply(list.of.packages, require, character.only=TRUE)

#-----------------------------------------
indicator <- function(vector){
        vector <- as.numeric(vector)
        if (length(unique(vector))==1) {
                vector <- 0
        } else {
                vector[which(vector==min(vector))] <- 0
                vector[which(vector==max(vector))] <- 1
        }
        return(vector)
}

#-----------------------------------------------
#Estimate means and nuissance parameters of Sarmanov and Lee
    MVB with 4 repeated measures and two treatment groups
#Inputs: data in long format (subj, trt, time, response),
   tolerance, and number of iterations
#Outputs: estimates of parameters, hessian matrix, and data

SLMVB.AR1 <- function(long.dat) {

        #vector to store final estimates
        beta <- vector('numeric', 13)

        #number of treatments
        grp <- length(unique(long.dat$trt))

        #reshape data to wide format
        wide.dat <- dcast(long.dat, subj+trt~time, value.var
           ='response')

        #change treatment to binary variable 0/1
        wide.dat$trt <- indicator(wide.dat$trt)

        #seperate columns into new data frames
        dat <- cbind(wide.dat$time1, wide.dat$time2, wide.
           dat$time3, wide.dat$time4)
        x <- wide.dat$trt

        #Methods of Moments for correlation parameter
        M <- cor(dat)
        tmp.f <- function(p) 3*p^2 - 3*p^4 + p^6 - (1 - det(
           M))
        p <- p.hat <- beta[13] <- uniroot(tmp.f, lower=0,
           upper=1)$root
```

```
p123 <- p234 <- sqrt(2*p^2 - p^4)
p124 <- p134 <- sqrt(p^2 + p^4 - p^6)
p1234 <- sqrt(3*p^2 - 3*p^4 + p^6)

score <- function(b) {
        u <- numeric(12)
        eta1 <- exp(b[1] + b[5]*x + b[9])
        den1 <- 1 + exp(b[1] + b[5]*x)
        eta2 <- exp(b[2] + b[6]*x + b[10])
        den2 <- 1 + exp(b[2] + b[6]*x)
        eta3 <- exp(b[3] + b[7]*x + b[11])
        den3 <- 1 + exp(b[3] + b[7]*x)
        eta4 <- exp(b[4] + b[8]*x + b[12])
        den4 <- 1 + exp(b[4] + b[8]*x)
        s.x <- (1+exp(b[1] + b[5]*x))^(-1)*sqrt(exp(
           b[1] + b[5]*x)/(1+exp(b[9])))
        m.x <- exp(b[1] + b[5]*x)/(1+exp(b[1] + b[5]
           *x))
        der.x <- 0.5*(dat[,1]*(exp(b[1] + b[5]*x)-1)
           -exp(b[1] + b[5]*x)) * (exp(b[1] + b[5]*x
           )/(1+exp(b[9])))^(-1/2)
        s.y <- (1+exp(b[2] + b[6]*x))^(-1)*sqrt(exp(
           b[2] + b[6]*x)/(1+exp(b[10])))
        m.y <- exp(b[2] + b[6]*x)/(1+exp(b[2] + b[6]
           *x))
        der.y <- 0.5*(dat[,2]*(exp(b[2] + b[6]*x)-1)
           -exp(b[2] + b[6]*x)) * (exp(b[2] + b[6]*x
           )/(1+exp(b[10])))^(-1/2)
        s.z <- (1+exp(b[3] + b[7]*x))^(-1)*sqrt(exp(
           b[3] + b[7]*x)/(1+exp(b[11])))
        m.z <- exp(b[3] + b[7]*x)/(1+exp(b[3] + b[7]
           *x))
        der.z <- 0.5*(dat[,3]*(exp(b[3] + b[7]*x)-1)
           -exp(b[3] + b[7]*x)) * (exp(b[3] + b[7]*x
           )/(1+exp(b[11])))^(-1/2)
        s.w <- (1+exp(b[4] + b[8]*x))^(-1)*sqrt(exp(
           b[4] + b[8]*x)/(1+exp(b[12])))
        m.w <- exp(b[4] + b[8]*x)/(1+exp(b[4] + b[8]
           *x))
        der.w <- 0.5*(dat[,4]*(exp(b[4] + b[8]*x)-1)
           -exp(b[4] + b[8]*x)) * (exp(b[4] + b[8]*x
           )/(1+exp(b[12])))^(-1/2)
        R <- 1 + p*(dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s
           .y + p^2*(dat[,1]-m.x)/s.x*(dat[,3]-m.z)/
           s.z + p^3*(dat[,1]-m.x)/s.x*(dat[,4]-m.w)
           /s.w + p*(dat[,2]-m.y)/s.y*(dat[,3]-m.z)/
           s.z + p^2*(dat[,2]-m.y)/s.y*(dat[,4]-m.w)
           /s.w + p*(dat[,3]-m.z)/s.z*(dat[,4]-m.w)/
           s.w + p123*(dat[,1]-m.x)/s.x*(dat[,2]-m.y
           )/s.y*(dat[,3]-m.z)/s.z + p124*(dat[,1]-m
           .x)/s.x*(dat[,2]-m.y)/s.y*(dat[,4]-m.w)/s
           .w + p134*(dat[,1]-m.x)/s.x*(dat[,3]-m.z)
           /s.z*(dat[,4]-m.w)/s.w + p234*(dat[,2]-m.
           y)/s.y*(dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.
```

```
    w + p1234*(dat[,1]-m.x)/s.x*(dat[,2]-m.y)
    /s.y*(dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w

tmp1 <- digamma(exp(b[9])/den1)*(eta1/den1
    ^2) + (eta1/den1^2)*log(dat[,1]/(1-dat
    [,1]))
tmp2 <- -digamma(eta1/den1)*(eta1/den1^2)
tmp3 <- p*der.x*(dat[,2]-m.y)/s.y + p^2*der.
    x*(dat[,3]-m.z)/s.z + p^3*der.x*(dat[,4]-
    m.w)/s.w + p123*der.x*(dat[,2]-m.y)/s.y*(
    dat[,3]-m.z)/s.z + p124*der.x*(dat[,2]-m.
    y)/s.y*(dat[,4]-m.w)/s.w + p134*der.x*(
    dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w +
    p1234*der.x*(dat[,2]-m.y)/s.y*(dat[,3]-m.
    z)/s.z*(dat[,4]-m.w)/s.w
u[1] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[10])/den2)*(eta2/den2
    ^2) + (eta2/den2^2)*log(dat[,2]/(1-dat
    [,2]))
tmp2 <- -digamma(eta2/den2)*(eta2/den2^2)
tmp3 <- p*(dat[,1]-m.x)/s.x*der.y + p*der.y*
    (dat[,3]-m.z)/s.z + p^2*der.y*(dat[,4]-m.
    w)/s.w + p123*(dat[,1]-m.x)/s.x*der.y*(
    dat[,3]-m.z)/s.z + p124*(dat[,1]-m.x)/s.x
    *der.y*(dat[,4]-m.w)/s.w + p234*der.y*(
    dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w +
    p1234*(dat[,1]-m.x)/s.x*der.y*(dat[,3]-m.
    z)/s.z*(dat[,4]-m.w)/s.w
u[2] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[11])/den3)*(eta3/den3
    ^2) + (eta3/den3^2)*log(dat[,3]/(1-dat
    [,3]))
tmp2 <- -digamma(eta3/den3)*(eta3/den3^2)
tmp3 <- p^2*(dat[,1]-m.x)/s.x*der.z + p*(dat
    [,2]-m.y)/s.y*der.z + p*der.z*(dat[,4]-m.
    w)/s.w + p123*(dat[,1]-m.x)/s.x*(dat[,2]-
    m.y)/s.y*der.z + p134*(dat[,1]-m.x)/s.x*
    der.z*(dat[,4]-m.w)/s.w + p234*(dat[,2]-m
    .y)/s.y*der.z*(dat[,4]-m.w)/s.w + p1234*(
    dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*der.z*
    (dat[,4]-m.w)/s.w
u[3] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[12])/den4)*(eta4/den4
    ^2) + (eta4/den4^2)*log(dat[,4]/(1-dat
    [,4]))
tmp2 <- -digamma(eta4/den4)*(eta4/den4^2)
tmp3 <- p^3*(dat[,1]-m.x)/s.x*der.w + p^2*(
    dat[,2]-m.y)/s.y*der.w + p*(dat[,3]-m.z)/
    s.z*der.w + p124*(dat[,1]-m.x)/s.x*(dat
    [,2]-m.y)/s.y*der.w + p134*(dat[,1]-m.x)/
    s.x*(dat[,3]-m.z)/s.z*der.w + p234*(dat
    [,2]-m.y)/s.y*(dat[,3]-m.z)/s.z*der.w +
   p1234*(dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y
   *(dat[,3]-m.z)/s.z*der.w
u[4] <- sum(tmp1 + tmp2 + (tmp3/R))

der.x <- 0.5*x*(dat[,1]*(exp(b[1] + b[5]*x)
   -1)-exp(b[1] + b[5]*x)) * (exp(b[1] + b
   [5]*x)/(1+exp(b[9])))^(-1/2)
der.y <- 0.5*x*(dat[,2]*(exp(b[2] + b[6]*x)
   -1)-exp(b[2] + b[6]*x)) * (exp(b[2] + b
   [6]*x)/(1+exp(b[10])))^(-1/2)
der.z <- 0.5*x*(dat[,3]*(exp(b[3] + b[7]*x)
   -1)-exp(b[3] + b[7]*x)) * (exp(b[3] + b
   [7]*x)/(1+exp(b[11])))^(-1/2)
der.w <- 0.5*x*(dat[,4]*(exp(b[4] + b[8]*x)
   -1)-exp(b[4] + b[8]*x)) * (exp(b[4] + b
   [8]*x)/(1+exp(b[12])))^(-1/2)

tmp1 <- digamma(exp(b[9])/den1)*(eta1/den1
   ^2)*x + (eta1/den1^2)*log(dat[,1]/(1-dat
   [,1]))*x
tmp2 <- -digamma(eta1/den1)*(eta1/den1^2)*x
tmp3 <- p*der.x*(dat[,2]-m.y)/s.y + p^2*der.
   x*(dat[,3]-m.z)/s.z + p^3*der.x*(dat[,4]-
   m.w)/s.w + p123*der.x*(dat[,2]-m.y)/s.y*(
   dat[,3]-m.z)/s.z + p124*der.x*(dat[,2]-m.
   y)/s.y*(dat[,4]-m.w)/s.w + p134*der.x*(
   dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w +
   p1234*der.x*(dat[,2]-m.y)/s.y*(dat[,3]-m.
   z)/s.z*(dat[,4]-m.w)/s.w
u[5] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[10])/den2)*(eta2/den2
   ^2)*x + (eta2/den2^2)*log(dat[,2]/(1-dat
   [,2]))*x
tmp2 <- -digamma(eta2/den2)*(eta2/den2^2)*x
tmp3 <- p*(dat[,1]-m.x)/s.x*der.y + p*der.y*
   (dat[,3]-m.z)/s.z + p^2*der.y*(dat[,4]-m.
   w)/s.w + p123*(dat[,1]-m.x)/s.x*der.y*(
   dat[,3]-m.z)/s.z + p124*(dat[,1]-m.x)/s.x
   *der.y*(dat[,4]-m.w)/s.w + p234*der.y*(
   dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w +
   p1234*(dat[,1]-m.x)/s.x*der.y*(dat[,3]-m.
   z)/s.z*(dat[,4]-m.w)/s.w
u[6] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[11])/den3)*(eta3/den3
   ^2)*x + (eta3/den3^2)*log(dat[,3]/(1-dat
   [,3]))*x
tmp2 <- -digamma(eta3/den3)*(eta3/den3^2)*x
tmp3 <- p^2*(dat[,1]-m.x)/s.x*der.z + p*(dat
   [,2]-m.y)/s.y*der.z + p*der.z*(dat[,4]-m.
   w)/s.w + p123*(dat[,1]-m.x)/s.x*(dat[,2]-
   m.y)/s.y*der.z + p134*(dat[,1]-m.x)/s.x*
   der.z*(dat[,4]-m.w)/s.w + p234*(dat[,2]-m
   .y)/s.y*der.z*(dat[,4]-m.w)/s.w + p1234*(
   dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*der.z*
   (dat[,4]-m.w)/s.w
u[7] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[12])/den4)*(eta4/den4
   ^2)*x + (eta4/den4^2)*log(dat[,4]/(1-dat
   [,4]))*x
tmp2 <- -digamma(eta4/den4)*(eta4/den4^2)*x
tmp3 <- p^3*(dat[,1]-m.x)/s.x*der.w + p^2*(
   dat[,2]-m.y)/s.y*der.w + p*(dat[,3]-m.z)/
   s.z*der.w + p124*(dat[,1]-m.x)/s.x*(dat
   [,2]-m.y)/s.y*der.w + p134*(dat[,1]-m.x)/
   s.x*(dat[,3]-m.z)/s.z*der.w + p234*(dat
   [,2]-m.y)/s.y*(dat[,3]-m.z)/s.z*der.w +
   p1234*(dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y
   *(dat[,3]-m.z)/s.z*der.w
u[8] <- sum(tmp1 + tmp2 + (tmp3/R))

der.x <- 0.5*(dat[,1]-m.x)*den1*(exp(b[1] +
   b[5]*x)/(1+exp(b[9])))^(1/2)*exp(b[9])/
   exp(b[1] + b[5]*x)
der.y <- 0.5*(dat[,2]-m.y)*den2*(exp(b[2] +
   b[6]*x)/(1+exp(b[10])))^(1/2)*exp(b[10])/
   exp(b[2] + b[6]*x)
der.z <- 0.5*(dat[,3]-m.z)*den3*(exp(b[3] +
   b[7]*x)/(1+exp(b[11])))^(1/2)*exp(b[11])/
   exp(b[3] + b[7]*x)
der.w <- 0.5*(dat[,4]-m.w)*den4*(exp(b[4] +
   b[8]*x)/(1+exp(b[12])))^(1/2)*exp(b[12])/
   exp(b[4] + b[8]*x)

tmp1 <- digamma(exp(b[9]))*exp(b[9]) + (eta1
   /den1)*log(dat[,1]) + (exp(b[9])/den1)*
   log(1-dat[,1])
tmp2 <- -digamma(eta1/den1)*(eta1/den1) -
   digamma(exp(b[9])/den1)*(exp(b[9])/den1)
tmp3 <- p*der.x*(dat[,2]-m.y)/s.y + p^2*der.
   x*(dat[,3]-m.z)/s.z + p^3*der.x*(dat[,4]-
   m.w)/s.w + p123*der.x*(dat[,2]-m.y)/s.y*(
   dat[,3]-m.z)/s.z + p124*der.x*(dat[,2]-m.
   y)/s.y*(dat[,4]-m.w)/s.w + p134*der.x*(
   dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w +
   p1234*der.x*(dat[,2]-m.y)/s.y*(dat[,3]-m.
   z)/s.z*(dat[,4]-m.w)/s.w
u[9] <- sum(tmp1 + tmp2 + (tmp3/R))

tmp1 <- digamma(exp(b[10]))*exp(b[10]) + (
   eta2/den2)*log(dat[,2]) + (exp(b[10])/
   den2)*log(1-dat[,2])
tmp2 <- -digamma(eta2/den2)*(eta2/den2) -
   digamma(exp(b[10])/den2)*(exp(b[10])/den2
   )
tmp3 <- p*(dat[,1]-m.x)/s.x*der.y + p*der.y*
   (dat[,3]-m.z)/s.z + p^2*der.y*(dat[,4]-m.
   w)/s.w + p123*(dat[,1]-m.x)/s.x*der.y*(
           dat[,3]-m.z)/s.z + p124*(dat[,1]-m.x)/s.x
           *der.y*(dat[,4]-m.w)/s.w + p234*der.y*(
           dat[,3]-m.z)/s.z*(dat[,4]-m.w)/s.w +
           p1234*(dat[,1]-m.x)/s.x*der.y*(dat[,3]-m.
           z)/s.z*(dat[,4]-m.w)/s.w
        u[10] <- sum(tmp1 + tmp2 + (tmp3/R))

        tmp1 <- digamma(exp(b[11]))*exp(b[11]) + (
           eta3/den3)*log(dat[,3]) + (exp(b[11])/
           den3)*log(1-dat[,3])
        tmp2 <- -digamma(eta3/den3)*(eta3/den3) -
           digamma(exp(b[11])/den3)*(exp(b[11])/den3
           )
        tmp3 <- p^2*(dat[,1]-m.x)/s.x*der.z + p*(dat
           [,2]-m.y)/s.y*der.z + p*der.z*(dat[,4]-m.
           w)/s.w + p123*(dat[,1]-m.x)/s.x*(dat[,2]-
           m.y)/s.y*der.z + p134*(dat[,1]-m.x)/s.x*
           der.z*(dat[,4]-m.w)/s.w + p234*(dat[,2]-m
           .y)/s.y*der.z*(dat[,4]-m.w)/s.w + p1234*(
           dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y*der.z*
           (dat[,4]-m.w)/s.w
        u[11] <- sum(tmp1 + tmp2 + (tmp3/R))

        tmp1 <- digamma(exp(b[12]))*exp(b[12]) + (
           eta4/den4)*log(dat[,4]) + (exp(b[12])/
           den4)*log(1-dat[,4])
        tmp2 <- -digamma(eta4/den4)*(eta4/den4) -
           digamma(exp(b[12])/den4)*(exp(b[12])/den4
           )
        tmp3 <- p^3*(dat[,1]-m.x)/s.x*der.w + p^2*(
           dat[,2]-m.y)/s.y*der.w + p*(dat[,3]-m.z)/
           s.z*der.w + p124*(dat[,1]-m.x)/s.x*(dat
           [,2]-m.y)/s.y*der.w + p134*(dat[,1]-m.x)/
           s.x*(dat[,3]-m.z)/s.z*der.w + p234*(dat
           [,2]-m.y)/s.y*(dat[,3]-m.z)/s.z*der.w +
           p1234*(dat[,1]-m.x)/s.x*(dat[,2]-m.y)/s.y
           *(dat[,3]-m.z)/s.z*der.w
        u[12] <- sum(tmp1 + tmp2 + (tmp3/R))

        u
}

initial.value <- rep(0,12)
for (i in 1:4) {
        mu <- mean(dat[,i])
        initial.value[i] <- logit(mu)
}

estimate <- nleqslv(initial.value, score, method='
   Newton', global='cline', jacobian=TRUE, control=
   list(allowSingular=TRUE, ftol=1e-8, maxit=20))

#determine convergence
converged <- 0
if (estimate$termcd == 1) converged <- 1
```

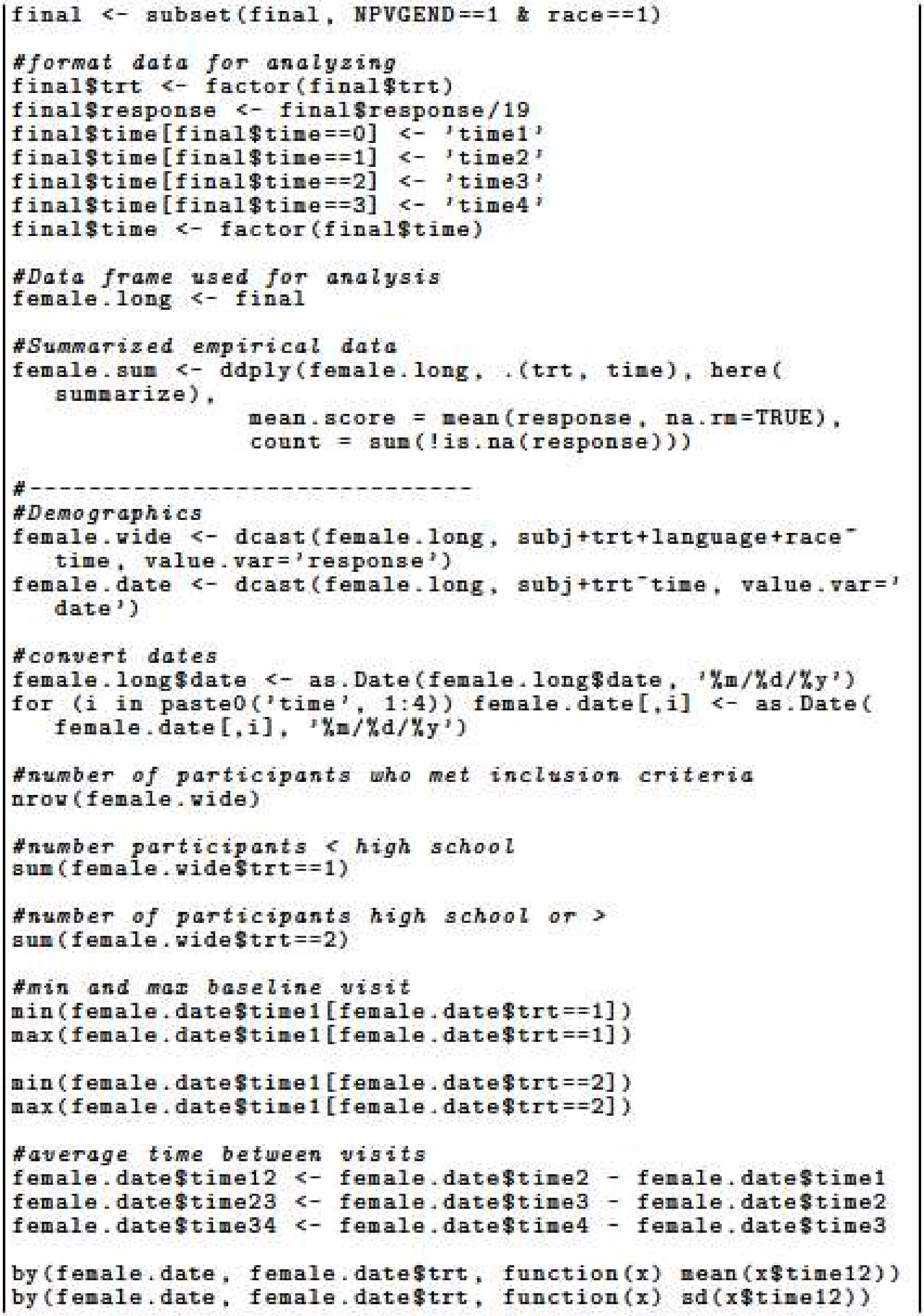
```
        if (converged == 0) {
                initial.value <- estimate$x
                estimate <- nleqslv(initial.value, score,
                   method='Newton', jacobian=TRUE, global='
                   cline', xscalm='auto', control=list(
                   allowSingular=TRUE, ftol=1e-8))
                if (estimate$termcd == 1) converged <- 1
        }

        beta[1:12] <- estimate$x
        H <- estimate$jac

        if (grp == 1) {
                beta <- beta[c(1:4,9:13)]
                names(beta) <- c('b1', 'b2', 'b3', 'b4', '
                   phi1', 'phi2', 'phi3', 'phi4', 'rho')
                hessian <- H[c(1:4,9:12), c(1:4,9:12)]
                rownames(hessian) <- colnames(hessian)
                return(list(beta=beta, hessian=hessian, dat=
                   long.dat, converged=converged))
        } else {
                names(beta) <- c('b1', 'b2', 'b3', 'b4', 'b5
                   ', 'b6', 'b7', 'b8', 'phi1', 'phi2', '
                   phi3', 'phi4', 'rho')
                hessian <- H
                rownames(hessian) <- colnames(hessian)
                return(list(beta=beta, hessian=hessian, dat=
                   long.dat, converged=converged))
        }
}
```

R Code used to analyze NNTC data



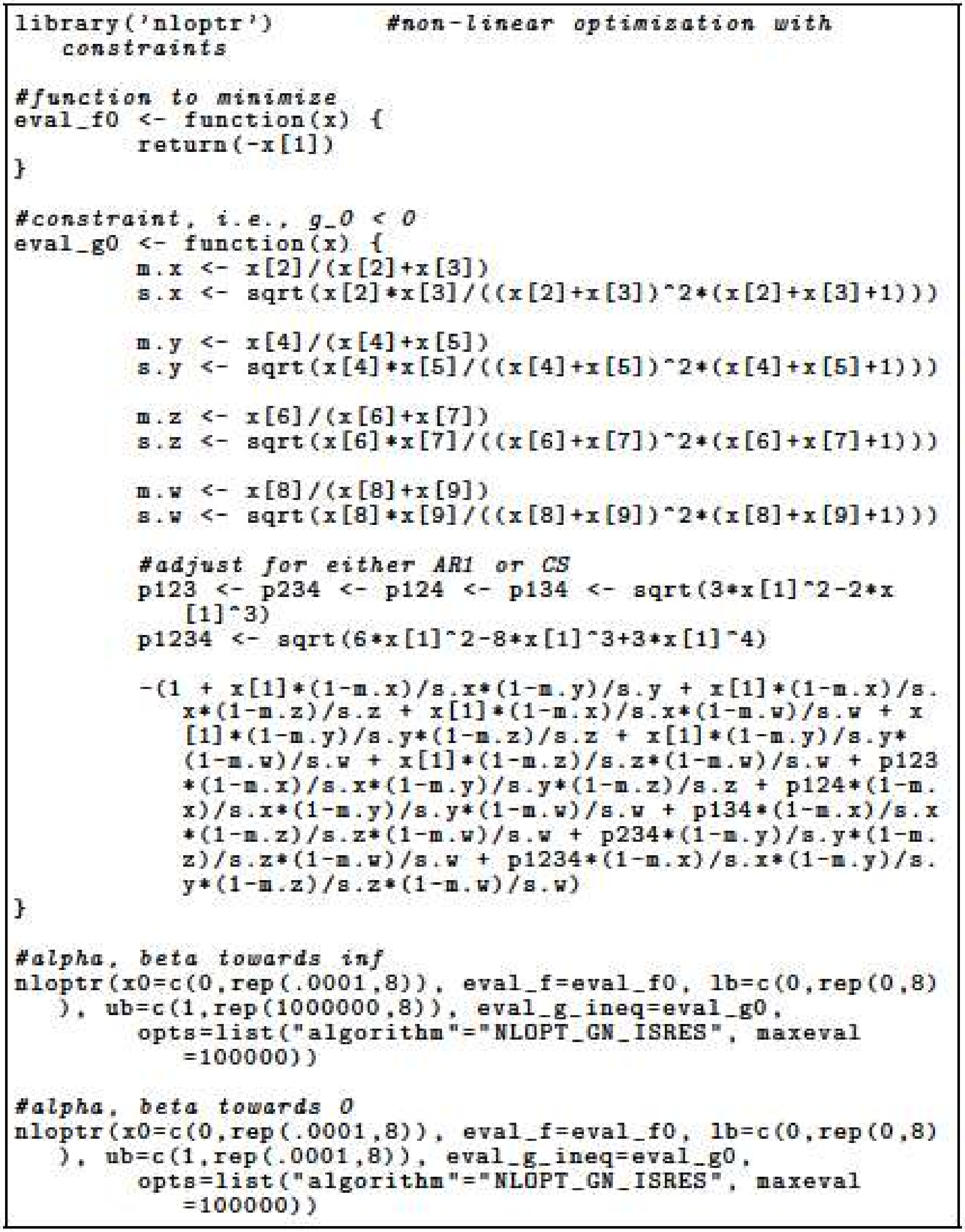
```
rm(list=ls())

#####################################
### Libraries ####################
#####################################
list.of.packages <- c('plyr', 'ggplot2')
new.packages <- list.of.packages[!(list.of.packages %in%
   installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)
lapply(list.of.packages, require, character.only=TRUE)

##########################################
### Load and Define Functions   #######
##########################################
save.loc <- 'c:/users/nicholas/desktop'

setwd('E:/Dissertation/Correlated␣Beta/')
source('./R␣Code/Dissertation/dissertation_functions.R')
source('./R␣Code/Dissertation/dissertation_LNMVB_jacobian.R'
   )
source('./R␣Code/Dissertation/dissertation_SLMVB_AR1_
   jacobian.R')
source('./R␣Code/Dissertation/dissertation_SLMVB_CS_jacobian
   .R')

#---------------------------
#calculates coefficients, se, ci, and p-value to correspond
   to models in paper
#input: model, and denominator degrees of freedom
#output: matrix
convert <- function(model, den.df) {
        vbeta <- solve(-model$hessian)
        vbeta <- vbeta[1:8,1:8]
        mat <- matrix(NA, nrow=8, ncol=6)
        mat[1,1] <- model$beta[1]
        for (i in 2:4) mat[i,1] <- model$beta[i] - model$
           beta[1]
        mat[5,1] <- model$beta[5]
        for (i in 6:8) mat[i,1] <- model$beta[i] - mat[5,1]
        mat[1,2] <- sqrt(vbeta[1,1])
        for (i in 2:4) {
                L <- c(-1, rep(0,7))
                L[i] <- 1
                mat[i,2] <- sqrt(L %*% vbeta %*% L)
        }
        mat[5,2] <- sqrt(vbeta[5,5])
        for (i in 6:8) {
                L <- c(rep(0,4),-1,rep(0,3))
                L[i] <- 1
                mat[i,2] <- sqrt(L %*% vbeta %*% L)
        }
        for (i in 1:8) {
                ci <- t.confidence(mat[i,1],mat[i,2],den.df)
                mat[i,5] <- ci[1]
                mat[i,6] <- ci[2]
                if (mat[i,1] < 0) {
                        mat[i,4] <- pt(mat[i,1]/mat[i,2],
                           den.df, lower.tail=T)*2
                } else {
                        mat[i,4] <- pt(mat[i,1]/mat[i,2],
                           den.df, lower.tail=F)*2
                }
        }
        mat <- format(round(mat, 3), nsmall=3)
        mat[,3] <- paste0('(', mat[,5], ',␣', mat[,6], ')')
        mat <- mat[,1:4]
        colnames(mat) <- c('Estimate', 'SE', 'CI', 'P-value'
           )
        return(mat)
}

convert.gee.glmm <- function(model, den.df) {
        if (class(model) == 'MixMod') {
                model$beta <- model$coefficients
                model$var <- solve(model$Hessian)
        }
        mat <- matrix(NA, nrow=8, ncol=6)
        for (i in 1:8) {
                mat[i,1] <- model$beta[i]
                mat[i,2] <- sqrt(model$var[i,i])
                ci <- t.confidence(mat[i,1],mat[i,2],den.df)
                mat[i,5] <- ci[1]
                mat[i,6] <- ci[2]

                if (mat[i,1] < 0) {
                        mat[i,4] <- pt(mat[i,1]/mat[i,2],
                           den.df, lower.tail=T)*2
                } else {
                        mat[i,4] <- pt(mat[i,1]/mat[i,2],
                           den.df, lower.tail=F)*2
                }
        }
        mat <- format(round(mat, 3), nsmall=3)
        mat[,3] <- paste0('(', mat[,5], ',␣', mat[,6], ')')
        mat <- mat[,1:4]
        colnames(mat) <- c('Estimate', 'SE', 'CI', 'P-value'
           )
        return(mat)
}

########################################
### Analyze Data #######################
########################################
#load data
tmp <- read.csv('./R626_Data/NNTC_Summary_scores.csv')

#Define treatment variable
tmp$educ <- 1
tmp$educ[tmp$NPVEDUC >= 12] <- 2
```

```
#keep required columns
tmp <- tmp[, c('PATID', 'VISNO', 'educ', 'NPVHVDSS', '
   NPVGEND', 'DTASSESS', 'NPVTLANG', 'NPVRACE', 'NPVAGE')]
colnames(tmp) <- c('subj', 'time', 'trt', 'response', '
   NPVGEND', 'date', 'language', 'race', 'age')

#number of individuals with 4 or more repeated measures
length(unique(tmp$subj))

#include only individuals who has complete data for first 4
   visits
tmp <- tmp[-grep('.S', tmp$time),]
tmp$time <- as.numeric(as.character(tmp$time))
tmp <- tmp[tmp$time < 4,]
tmp$subj <- as.character(tmp$subj)
#keep visits 0, 1, 2, 3
tmp.by <- by(tmp, tmp$subj, function(x) x)
final <- data.frame()
for(i in 1:length(tmp.by)) {
        if (nrow(tmp.by[[i]])==4) {
                if (sum(tmp.by[[i]]$time==c(0,1,2,3))==4) {
                        final <- rbind(final, tmp.by[[i]])
                }
        }
}
tmp <- final
#keep individuals whose education does not change
tmp.by <- by(tmp, tmp$subj, function(x) x)
final <- data.frame()
for(i in 1:length(tmp.by)) {
        if (nrow(tmp.by[[i]])==4) {
                if (sum(tmp.by[[i]]$trt)==4 | sum(tmp.by[[i
                   ]]$trt)==8) {
                        final <- rbind(final, tmp.by[[i]])
                }
        }
}
tmp <- final
#complete data only
tmp.by <- by(tmp, tmp$subj, function(x) x)
final <- data.frame()
for(i in 1:length(tmp.by)) {
        if (nrow(tmp.by[[i]])==4) {
                if (sum(is.na(tmp.by[[i]][,'response']))==0)
                    {
                        final <- rbind(final, tmp.by[[i]])
                }
        }
}
#counts
length(unique(final$subj))
length(unique(final$subj[final$NPVGEND==1]))

#keep african american females
```



```
final <- subset(final, NPVGEND==1 & race==1)

#format data for analyzing
final$trt <- factor(final$trt)
final$response <- final$response/19
final$time[final$time==0] <- 'time1'
final$time[final$time==1] <- 'time2'
final$time[final$time==2] <- 'time3'
final$time[final$time==3] <- 'time4'
final$time <- factor(final$time)

#Data frame used for analysis
female.long <- final

#Summarized empirical data
female.sum <- ddply(female.long, .(trt, time), here(
   summarize),
                 mean.score = mean(response, na.rm=TRUE),
                 count = sum(!is.na(response)))

#-------------------------------
#Demographics
female.wide <- dcast(female.long, subj+trt+language+race~
   time, value.var='response')
female.date <- dcast(female.long, subj+trt~time, value.var='
   date')

#convert dates
female.long$date <- as.Date(female.long$date, '%m/%d/%y')
for (i in paste0('time', 1:4)) female.date[,i] <- as.Date(
   female.date[,i], '%m/%d/%y')

#number of participants who met inclusion criteria
nrow(female.wide)

#number participants < high school
sum(female.wide$trt==1)

#number of participants high school or >
sum(female.wide$trt==2)

#min and max baseline visit
min(female.date$time1[female.date$trt==1])
max(female.date$time1[female.date$trt==1])

min(female.date$time1[female.date$trt==2])
max(female.date$time1[female.date$trt==2])

#average time between visits
female.date$time12 <- female.date$time2 - female.date$time1
female.date$time23 <- female.date$time3 - female.date$time2
female.date$time34 <- female.date$time4 - female.date$time3

by(female.date, female.date$trt, function(x) mean(x$time12))
by(female.date, female.date$trt, function(x) sd(x$time12))
by(female.date, female.date$trt, function(x) mean(x$time23))
by(female.date, female.date$trt, function(x) sd(x$time23))

by(female.date, female.date$trt, function(x) mean(x$time34))
by(female.date, female.date$trt, function(x) sd(x$time34))

#age
female.age <- female.long[female.long$time==’time1’,]
by(female.age, female.age$trt, function(x) mean(x$age))
by(female.age, female.age$trt, function(x) sd(x$age))

#correlation
female.cor <- female.wide[,paste0(’time’, 1:4)]
cor(female.cor)

#-----------------------------------------------
#LNMVB
mod.LNMVB <- LNMVB(female.long)
sum.LNMVB <- lsmeans(mod.LNMVB, type=’LNMVB’, ddf(female.
   long))

#-----------------------------------------------
#SLMVB.AR1
mod.SLMVB.AR1 <- SLMVB.AR1(female.long)
sum.SLMVB.AR1 <- lsmeans(mod.SLMVB.AR1, type=’SLMVB’, ddf(
   female.long))

#-----------------------------------------------
#SLMVB.CS
mod.SLMVB.CS <- SLMVB.CS(female.long)
sum.SLMVB.CS <- lsmeans(mod.SLMVB.CS, type=’SLMVB’, ddf(
   female.long))

#------------------------------------------------
#GLMM
mod.glmm <- mixed_model(response~time+trt+time:trt, random=
   ~ 1|subj, data=female.long, family=beta.glmm, n_phis=1)
sum.glmm <- lsmeans(mod.glmm, type=’glmm’, ddf(female.long))

#----------------------------------------------
#GEE
mod.gee <- geem(response~time+trt+time:trt, id=subj, family=
   FunList, corstr=’ar1’, data=female.long)
sum.gee <- lsmeans(mod.gee, type=’gee’, ddf(female.long))

#-----------------------------------------------
#plots

#set up results for plotting
mod.ls <- list(’LNMVB’=data.frame(sum.LNMVB), ’SLMVB.CS’=
   data.frame(sum.SLMVB.CS), ’SLMVB.AR1’=data.frame(sum.
   SLMVB.AR1), ’GLMM’=data.frame(sum.glmm), ’GEE’=data.frame
   (sum.gee))
mod.ls <- lapply(mod.ls, function(x) {
        x$trt <- 2
        x$trt[1:4] <- 1
        x$trt <- factor(x$trt, labels=c('Did␣not␣complete␣
           high␣school', 'Completed␣high␣school␣and␣or␣
           beyond'))
        x$time <- rep(1:4,2)
        x$model <- i
        x
})
df <- data.frame()
mod.ls.tmp <- mod.ls[!names(mod.ls) %in% 'SLMVB.AR1']
for (i in 1:4) df <- rbind(df, mod.ls.tmp[[i]])
df$model <- factor(df$model, levels=c('LNMVB', 'SLMVB.CS', '
   GLMM', 'GEE'))
female.sum$trt <- factor(female.sum$trt, labels=c('Did␣not␣
   complete␣high␣school', 'Completed␣high␣school␣and␣or␣
   beyond'))
a <- ggplot(df, aes(x=time, y=mean)) + geom_point(aes(shape=
   'model'), size=3) + geom_line(linetype=1) +
        geom_point(data=female.sum, aes(x=as.numeric(time),
           y=mean.score, shape='data'), size=3) +
        geom_line(data=female.sum, aes(x=as.numeric(time), y
           =mean.score), linetype=2) +
        facet_grid(model~trt) +
        scale_y_continuous(lim=c(.18,.34), breaks=seq
           (.18,.33,.03)) +
        scale_x_continuous(lim=c(1,4), breaks=1:4, labels
           =0:3) +
        annotate('text', x=1.25, y=.205,
                label=c(NA,paste0('F␣test\nP-value:␣', round
                   (df$P.DDF.CONV[1], 4)),
                        NA,paste0('F␣test\nP-value:␣', round
                           (df$P.DDF.CONV[9], 4)),
                        NA,paste0('F␣test\nP-value:␣', round
                           (df$P.DDF.CONV[17], 4)),
                        NA,paste0('F␣test\nP-value:␣', round
                           (df$P.DDF.CONV[25], 4))),
        hjust=0) +
        annotate('rect', xmin=1.2, xmax=2.75, ymin=.18, ymax
           =.23, color=rep(c(NA,'black'),4), alpha=0) +
        scale_shape_manual(name='Estimate', values=c('model'
           =16, 'data'=17)) + theme_bw() +
        labs(x='Visit', y='Mean␣Hopkins␣Verbal␣Learning␣Test
           ␣delayed␣scaled␣score') +
        theme(panel.grid=element_blank(), legend.position='
           bottom', legend.background=element_rect(size=.5,
           color='black'))
ggsave(filename=paste0(save.loc, '/Means_plot.png'), plot=a,
    height=7, width=6, units='in', dpi=300)

df <- data.frame()
df <- mod.ls[['SLMVB.AR1']]
b <- ggplot(df, aes(x=time, y=mean)) + geom_point(aes(shape=
   'model'), size=3) + geom_line(linetype=1) +
        geom_point(data=female.sum, aes(x=as.numeric(time),
           y=mean.score, shape='data'), size=3) +
        geom_line(data=female.sum, aes(x=as.numeric(time), y
           =mean.score), linetype=2) +
        facet_grid(.~trt) +
        scale_y_continuous(lim=c(0,.4), breaks=seq(0,.4,.05)
           ) +
        scale_x_continuous(lim=c(1,4), breaks=1:4, labels
           =0:3) +
        annotate('text', x=1.25, y=.05,
                label=c(NA,paste0('F␣test\nP-value:␣', round
                   (df$P.DDF.CONV[1], 4))),
        hjust=0) +
        annotate('rect', xmin=1.2, xmax=2.75, ymin=0.01,
           ymax=.09, color=rep(c(NA,'black'),1), alpha=0) +
        scale_shape_manual(name='Estimate', values=c('model'
           =16, 'data'=17)) + theme_bw() +
        labs(x='Visit', y='Mean␣Hopkins␣Verbal␣Learning␣Test
           ␣delayed␣scaled␣score') +
        theme(panel.grid=element_blank(), legend.position='
           bottom', legend.background=element_rect(size=.5,
           color='black'))
ggsave(filename=paste0(save.loc, '/Means_plot_SLMVB.png'),
   plot=b, height=4, width=6, units='in', dpi=300)

#-------------------------------------
#coefficients, se, ci, p-values

lnmvb <- convert(mod.LNMVB, ddf(female.long))
ar1 <- convert(mod.SLMVB.AR1, ddf(female.long))
cs <- convert(mod.SLMVB.CS, ddf(female.long))
glmm <- convert.gee.glmm(mod.glmm, ddf(female.long))
gee <- convert.gee.glmm(mod.gee, ddf(female.long))

write.csv(lnmvb, file=paste0(save.loc, '/lnmvb.csv)', row.
   names=FALSE)
write.csv(ar1, file=paste0(save.loc, '/ar1.csv'), row.names=
   FALSE)
write.csv(cs, file=paste0(save.loc, '/cs.csv'), row.names=
   FALSE)
write.csv(glmm, file=paste0(save.loc, '/glmm.csv'), row.
   names=FALSE)
write.csv(gee, file=paste0(save.loc, '/gee.csv'), row.names=
   FALSE)

#model correlations
mean(corr.LNMVB(mod.LNMVB), na.rm=TRUE)
mod.SLMVB.AR1$beta[length(mod.SLMVB.AR1$beta)]
mod.SLMVB.CS$beta[length(mod.SLMVB.CS$beta)]
as.numeric(mod.glmm$D)/(as.numeric(mod.glmm$D) + sigma.e(inv
   .logit(mod.glmm$coefficients[1]), exp(mod.glmm$phis)))
mod.gee$alpha
```

Maximum attainable correlation under SLMVB framework



```
library('nloptr')        #non-linear optimization with
   constraints

#function to minimize
eval_f0 <- function(x) {
        return(-x[1])
}

#constraint, i.e., g_0 < 0
eval_g0 <- function(x) {
        m.x <- x[2]/(x[2]+x[3])
        s.x <- sqrt(x[2]*x[3]/((x[2]+x[3])^2*(x[2]+x[3]+1)))

        m.y <- x[4]/(x[4]+x[5])
        s.y <- sqrt(x[4]*x[5]/((x[4]+x[5])^2*(x[4]+x[5]+1)))

        m.z <- x[6]/(x[6]+x[7])
        s.z <- sqrt(x[6]*x[7]/((x[6]+x[7])^2*(x[6]+x[7]+1)))

        m.w <- x[8]/(x[8]+x[9])
        s.w <- sqrt(x[8]*x[9]/((x[8]+x[9])^2*(x[8]+x[9]+1)))

        #adjust for either AR1 or CS
        p123 <- p234 <- p124 <- p134 <- sqrt(3*x[1]^2-2*x
           [1]^3)
        p1234 <- sqrt(6*x[1]^2-8*x[1]^3+3*x[1]^4)

        -(1 + x[1]*(1-m.x)/s.x*(1-m.y)/s.y + x[1]*(1-m.x)/s.
           x*(1-m.z)/s.z + x[1]*(1-m.x)/s.x*(1-m.w)/s.w + x
           [1]*(1-m.y)/s.y*(1-m.z)/s.z + x[1]*(1-m.y)/s.y*
           (1-m.w)/s.w + x[1]*(1-m.z)/s.z*(1-m.w)/s.w + p123
           *(1-m.x)/s.x*(1-m.y)/s.y*(1-m.z)/s.z + p124*(1-m.
           x)/s.x*(1-m.y)/s.y*(1-m.w)/s.w + p134*(1-m.x)/s.x
           *(1-m.z)/s.z*(1-m.w)/s.w + p234*(1-m.y)/s.y*(1-m.
           z)/s.z*(1-m.w)/s.w + p1234*(1-m.x)/s.x*(1-m.y)/s.
           y*(1-m.z)/s.z*(1-m.w)/s.w)
}

#alpha, beta towards inf
nloptr(x0=c(0,rep(.0001,8)), eval_f=eval_f0, lb=c(0,rep(0,8)
   ), ub=c(1,rep(1000000,8)), eval_g_ineq=eval_g0,
        opts=list("algorithm"="NLOPT_GN_ISRES", maxeval
           =100000))

#alpha, beta towards 0
nloptr(x0=c(0,rep(.0001,8)), eval_f=eval_f0, lb=c(0,rep(0,8)
   ), ub=c(1,rep(.0001,8)), eval_g_ineq=eval_g0,
        opts=list("algorithm"="NLOPT_GN_ISRES", maxeval
           =100000))
```