8-1-2024

Missing Data Imputation With Longitudinal Data

Joseph Omar Alanis
The University of Texas Rio Grande Valley

MISSING DATA IMPUTATION WITH LONGITUDINAL DATA

A Thesis

by

JOSEPH O. ALANIS

Submitted in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Major Subject: Mathematics

The University of Texas Rio Grande Valley

August 2024

MISSING DATA IMPUTATION WITH LONGITUDINAL DATA

A Thesis
by
JOSEPH O. ALANIS

COMMITTEE MEMBERS

Dr. Tamer Oraby
Chair of Committee

Dr. Michael Machiorlatti
Committee Member

Dr. Hansapani Rodrigo
Committee Member

Dr. George Yanev
Committee Member

August 2024

ABSTRACT

Alanis, Joseph O., Missing Data Imputation With Longitudinal Data. Master of Science (MS), August 2024, 60 pp., 33 tables, 11 figures, 27 references.

In the repeated measures longitudinal datasets where missing data is a relatively common issue, we explored different imputation methods, including machine learning (ML) methods in order to examine potential efficiencies for using traditional versus newer computational methods. In order to accomplish said comparison, we used a Monte Carlo simulation experiment of a population of size N=70000 to mimic a clinical trial, with different scenarios of missing at random (MAR) data in the response variable. To compare the behavior of each method resulting from the difference between population and sample dataset, a real dataset was used from the Boston College on "National Longitudinal Survey" to simulate MAR. Moreover, we used both datasets to examine the effects of different sample sizes when using Bayesian neural networks and *k*-Nearest Neighbors (*k*-NN) for imputations and compared that to the more traditional methods of last observed carried forward, multiple imputation, and linear regression. Additionally, the cost of computing power is evaluated at different sample sizes for each scenario on both datasets.

DEDICATION

To Linda Lydia, Adrian Joseph and Mario Omar, Mandi, Marvel, Mjlnor, Stormi, and Britney,

I apologize for all the time I had to sacrifice from my family to complete my Thesis. I hope you can forgive me and just know that I love you 3000!

Sincerely,

Joseph Omar Alanis

ACKNOWLEDGMENTS

I would like to give a special thank you to Dr. Tamer Oraby, Dr. Hansapani Rodrigo, and Dr. Michael Machiorlatti! I have learned a great deal from all of you these past few years and I'm truly grateful. I'm very fortunate to be both mentored and lectured by all of you and look forward to publishing with you all in the near future. Although, I have never worked with you personally Dr. George Yanev I have heard amazing stories of generosity and genuineness. In addition, many stories about how you go above and beyond for faculty and students alike.

TABLE OF CONTENTS

Page

ABSTRACT iii

DEDICATION iv

ACKNOWLEDGMENTS v

LIST OF TABLES viii

LIST OF FIGURES x

CHAPTER I: INTRODUCTION 1

CHAPTER II: LONGITUDINAL DATA 6

2.1 Missing Data 11

CHAPTER III: METHODOLOGY 14

3.1 Last Observed Carried Forward 14

3.2 Regression 14

3.3 Multiple Imputation 16

3.3.1 Mice Imputed 2 Level Class Predictive Mean Matching 18

3.4 Bayesian Neural Network 19

3.4.1 Graph Theory 19

3.4.2 Graph Structure 19

3.4.3 Statistical Introduction 20

3.4.4 Probability Factorization 20

3.4.5 D-Separation 21

3.4.6 Fundamental Connections 22

3.4.7 Equivalent Structures 22

3.4.8 Markov Blanket 23

3.4.9 Inductive Causation Algorithm 23

3.4.10 Static Bayesian Network Modeling 24

3.4.11 Constraint-Based Structure Learning Algorithms 24

3.4.12 Score-Based Structure Learning Algorithms 26

3.4.13 Hill-Climbing Algorithm 26

3.4.14 Hybrid Structure Learning Algorithms 27

3.4.15 Sparse Candidate Algorithm 27

3.4.16 Bayesian Neural Network Models 27

3.5 *k*-Nearest Neighbor 28

3.5.1 Nearest Neighbor Classification 29

3.5.2 Nearest Neighbor Algorithms 29

3.5.3 Choice of Nearest Neighbor Algorithms 30

3.5.4 Fitting and Evaluating the *k*-NN Model 31

CHAPTER IV: SIMULATION STUDY 32

4.1 Methods and Motivation 32

4.1.1 Real Data Settings 33

4.1.2 Simulation Settings 34

4.2 Results 36

4.2.1 Result Tables 39

4.2.2 Average Summary Tables Comparison Graphs: 51

CHAPTER V: CONCLUSION AND DISCUSSION 55

REFERENCES 56

APPENDICES 58

BIOGRAPHICAL SKETCH 60

LIST OF TABLES

Page

Table 1.1: Literature Review 5

Table 4.1: Scenario 1 Results with Real Data at n=100 39

Table 4.2: Scenario 2 Results with Real Data at n=100 39

Table 4.3: Scenario 3 Results with Real Data at n=100 40

Table 4.4: Simulated Scenario 1 Results with n=100 40

Table 4.5: Simulated Scenario 2 Results with n=100 40

Table 4.6: Simulated Scenario 3 Results with n=100 41

Table 4.7: Scenario 1 Results with Real Data at n=200 41

Table 4.8: Scenario 2 Results with Real Data at n=200 41

Table 4.9: Scenario 3 Results with Real Data at n=200 42

Table 4.10: Simulated Scenario 1 Results with n=200 42

Table 4.11: Simulated Scenario 2 Results with n=200 42

Table 4.12: Simulated Scenario 3 Results with n=200 43

Table 4.13: Scenario 1 Results with Real Data at n=500 43

Table 4.14: Scenario 2 Results with Real Data at n=500 43

Table 4.15: Scenario 3 Results with Real Data at n=500 44

Table 4.16: Simulated Scenario 1 Results with n=500 44

Table 4.17: Simulated Scenario 2 Results with n=500 44

Table 4.18: Simulated Scenario 3 Results with n=500 45

Table 4.19: Scenario 1 Results with Real Data at n=1000 45

Table 4.20: Scenario 2 Results with Real Data at n=1000 45

Table 4.21: Scenario 3 Results with Real Data at n=1000 46

Table 4.22: Simulated Scenario 1 Results with n=1000 46

Table 4.23: Simulated Scenario 2 Results with n=1000 46

Table 4.24: Simulated Scenario 3 Results with n=1000 47

Table 4.25: Summary table for $n = 100$ with all scenarios for real data 47

Table 4.26: Summary table for $n = 100$ with all scenarios for simulated data 47

Table 4.27: Summary table for $n = 200$ with all scenarios for real data 48
Table 4.28: Summary table for $n = 200$ with all scenarios for simulated data 48
Table 4.29: Summary table for $n = 500$ with all scenarios for real data 49
Table 4.30: Summary table for $n = 500$ with all scenarios for simulated data 49
Table 4.31: Summary table for $n = 1000$ with all scenarios for real data 50
Table 4.32: Summary table for $n = 1000$ with all scenarios for simulated data 50

LIST OF FIGURES

Page

Figure 3.1: Directed Graphs 19

Figure 4.1: Diagram of Bayesian Neural Network of Real Data 34

Figure 4.2: Diagram of Bayesian Neural Network of Real Data 36

Figure 4.3: $\mu_{y_{2i}}$ Bias for Real Data 51

Figure 4.4: $\mu_{y_{2i}}$ Bias for Simulated Data 51

Figure 4.5: $\mu_{y_{3i}}$ Bias for Real Data 52

Figure 4.6: $\mu_{y_{3i}}$ Bias for Simulated Data 52

Figure 4.7: $\mu_{y_{2i}}$ RMSE for Real Data 53

Figure 4.8: $\mu_{y_{2i}}$ RMSE for Simulated Data 53

Figure 4.9: $\mu_{y_{3i}}$ RMSE for Real Data 54

Figure 4.10: $\mu_{y_{3i}}$ RMSE for Simulated Data 54

CHAPTER I

INTRODUCTION

Missing data imputation have been an interesting area of research since the 1970's by Donald B. Rubin and is a forever changing field as new methods are created [21]. Before the birth of such methods researchers would discard viable data and only use complete case datasets to insure quality of results. Unfortunately, this is still a primary method of choice to address missing data for many research in the health profession. But why the need to impute missing data? The issue of missingness greatly plagues longitudinal datasets due to it's time sensitive nature. If participates in an longitudinal experiment designed to measure the effect of blood pressure medication at different time intervals of the treatment plan are missing one or more key treatment measures is it ethical to remove said individuals from the dataset? This idea of imputing missing data is a foreign and or an unethical idea for many health professionals because replacing and individuals blood pressure with an estimated based on parameters of others measurements in the study seems absurd. Although the answer to this question is beyond the scope of this thesis, the question we can ask and answer is just how effective has imputing missing data evolved over the last 50 years? Additionally, how do simple methods like last observed carried forward and linear regression compare to more advanced methods like Bayesian Neural networks and k-NN machine learning ? Having difficulty in finding little to no literature that compares Bayesian Networks to a large range of methods; the comparative data analysis will consist of a Monte Carlo simulation experiment and real data analysis in a wide range of scenarios of missing at random (MAR) in longitudinal datasets response variables.

The comparative data analysis was inspired by Md. Hamidul Hugue research that computed 12 different MI methods on both empirical and simulated study data from the six waves of the longitudinal study of Australian Children [8]. The study used to create the simulated data examined the development and overall health of Australian children. The time interval is every two years (waves) with a sample size of $n = 4893$ kindergartens to measure the association of obesity as a predictor for quality of life (QoL). Specifically, they were interested in the connection between obesity in waves (1-5) and QoL z-scores at wave 6 and cross-sectional association between Qol z-scores and BMI z-scores across all 6 waves. At baseline, The QoL z-score at wave 6 and cumulative burden of overweight(OverWtCat) were adjusted based on sex, language,socio-economic standing (SEP),age, and family structure (FamStCat).

Both models were susceptible to missing data due to drop out when applied to LSAC. Data was missing for both BMI and *QoL* z-scores in all six waves, as well as all other features. If the *QoL* z-score was missing across all 6 waves the participate was removed, which resulted in a total of $n = 4661$ participants.The *QoLz* was generated using the linear mixed-effects model.

The parameters for the above model were based on the LSAC data to similar proportions of missing observations for each variable at each wave. A 1000 datasets were created with sample sizes of 5000 to compare behavior of the regression coefficient estimators using the 12 MI methods. Of the 12 methods used the most effective methods where the following:

(i) Fully conditional specification (FCS) imputes variables using conditional uni-variate regression models for each incomplete variable, conditional on the time-dependent variables at all waves. Note the repeated measurements of the time-dependent variables are imputed using hierarchical models.

(ii) Joint Modelling - Multivariate LMM (JM-MLMM) imputes missing data using a joint multivariate LMM and repeated measurements of time-dependent variables are imputed hierarchical models. Also, binary variables are imputed as continuous variables.

The sampling distribution of the estimated bias and the coverage of the regression coefficients for the analysis model was record for all models was compared for both simulated data and LSAC data. In addition, The conclusion was FCS-standard and JM-MVN provided reliable estimates for both models with better coverage probabilities then majority of the other methods. However, the methods need further study to determine if they are appropriate when data is collected at irregular time intervals, and if a generalized linear mixed model based approach should be used instead.

In a comparative study by Ahmed Mahmoud Gad [6], the simulation study evaluated the behavior of eight imputation methods listed below on a dataset for n subjects with five measurements. They chose $n = \{10, 50, 100\}$ to evaluate small to large sample sizes and assumed there were two covariates. The first being time $Time$ and the treatment group. Hence the simulated data followed the model below

$$y_{ij} = \beta_0 + \beta_1 \text{Time}_i + \beta_2 \text{Grp}_i + \varepsilon_{ij} \tag{1.1}$$

where $\text{Time}_i = \{0, 1, 2, 3, 4\}$ for the five time points and Grp_i is dichotomous variable for placebo and treatment group. A simple linear regression model for the mean profiles of $E(y_{ij})$, as well as, the variance-covariance structure was assumed first-order autoregressive. Additionally, the ε_i's were generated from a multivariate normal with zero and and sigma square of one. The data was simulated to satisfy the mulit-variable normal distribution where each dataset is based on the following assumptions:

(i) The first time point is fully observed

(ii) The missingness pattern is monotone for MAR, MCAR, and MNAR

(iii) The number of replications is fixed at 5000

The comparison between the model was recorded by the measure of both the Relative Bias and Mean Square Error. For the MCAR simulation, the missing rates consist of 0%, 25%, 50%, 75%, and 87.5% for all timepoints.

Methods Used: The complete case Analysis Method, The Mean Substitutions Method, The last Observation Carried forward Method, The k-NN method, the Hot Deck method, The regression imputation method, The Expectation Maximization Algorithm, the Multiple Imputation Method.

The results were that each method handled better under certain scenarios depended on type of missingness, missing rates (%), and sample size. For example, the k-NN is better equip to handle large dataset for MCAR and MAR. The CCA method the overall choice for the MCAR, but trembled in the MAR and MNAR settings with biased estimates but MSE values.

With limited literature on Bayesian neural networks compared to non-parametric, semi-parametric, parametric, and machine learning I used G. Frank Lui and James frost Monte-Carlo simulation-based statistical modeling research on missing data in simulated clinical trails longitudinal data. The goal of their research was to conduct sensitivity analyses under different assumptions to assess the robustness of the analysis results from a clinical trial [4]. By creating a simulation population of million from a monotone missing multivariate normal (MVN) data at different level of under different missing data models allowed users to specify the expected proportion of missing data at each longitudinal time point. Second, a "tipping-point" sensitivity analysis method to which a delta-adjustment is applied to measure the potential difference in the estimated treatment effects between models. Last, a Bayesian Markov chain Monte-Carlo (MCMC) method for control-based imputation was considered to provide a higher yielding variance estimate that conventional multiple imputation. Their conclusion was the simulation-based approach for missing data in longitudinal study was extremely useful in design stage to calculate needed sample size and power, as well as, the final analysis stage to conducted sensitivity analysis.

In addition, a Monte-Carlo approach for Control-Based Imputation (CBI) Analysis was conducted as another approach for sensitivity analysis. Where the missing in the control group are imputed under the assumption of MAR, while the treatment groups missing data was imputed with a imputation model built from the control group. The CBI methods used for their research were defined by specifying the mean profile after drop out in the treatment group using the profile in the control group as follows:

1. Copy Increments in Reference(CIR): The increment mean change from the time of drop out for a patient in the treatment group will be the same as the increment mean change for a patient in the control group.

2. Jump to Reference (J2R): The mean profile after drop out for the test drug group will equal the mean profile of the control group.

3. Copy References (CR): The mean profile for a drop-out patient in test drug group will equal the mean profile for the control group for all time points.

In comparing all their methods by p-value, confidence interval (CI), and Square Error, they noticed that the MCMC sampling had high auto-correlations and produced similar to mixed model analysis under MAR data. With CBI, the point estimates shrunk toward zeros but the standard errors where very similar to the SE's from the MAR analyses. Hence, the CBI analyses with regular MI have large p-values compared to the primary analysis under MAR. In fact, the result of J2R, analysis became insignificant.

Table 1.1: Literature Review

LITERATURE REVIEW					
Methods	Statistical Model	Type of variable	Missingness	Measurement	Ref.
FCS	LRM/LMM	Num.(Discrete)	MAR	Bias/MSE	[2]
JM-MLMM	LRM/LMM	Num.(Discrete)	MAR	Bias/MSE	[2]
JM-MVN	LRM/LMM	Num.(Discrete)	MAR	Bias/MSE	[2]
CAAM	LRM/LMM	Cont./Dich.	MCAR,MAR, MNAR	Bias/MSE	[3]
MSM	LRM/LMM	Cont./Dich.	MCAR,MAR, MNAR	Bias/MSE	[3]
LOCF	LRM/LMM	Cont./Dich.	MCAR,MAR, MNAR	Bias/MSE	[3]
k-NN	LRM/LMM	Cont./Dich.	MCAR,MAR, MNAR	Bias/MSE	[3]
Hot Deck	LRM/LMM	Cont./Dich.	MCAR,MAR, MNAR	Bias/MSE	[3]
Reg. Imp.	LRM/LMM	Cont./Dich.	MCAR,MAR, MNAR	Bias/MSE	[3]
EMA	LRM/LMM	Cont./Dich.	MCAR,MAR, MNAR	Bias/MSE	[2]
MCMC	Monte-Carlo Sim.	Cont./Dich.	MAR, MNAR	p-value, CI, SE	[4]
CBI	Monte-Carlo Sim.	Cont./Dich.	MAR, MNAR	p-value, CI, SE	[4]
Mixed Mod	Monte-Carlo Sim.	Cont./Dich.	MAR, MNAR	p-value, CI, SE	[4]

CHAPTER II

LONGITUDINAL DATA

Longitudinal study is a research design with repeated measurement of variables over extended periods of time defined as longitudinal data [26]. The primary goal of a longitudinal study is to characterize the change in response over time based on factors that affects the responses. Additionally, to determine if these within-individual changes to response are related to selected covariates. The main objective of a longitudinal analysis is to describe trends in these within-individual changes in the response and relate it to selected covariates. In some longitudinal studies, it may be of interest to make predictions about how specific individuals change over time.

Let Y_{ij} denote the response variable for the i^{th} individual at the j^{th} occasion where $i, j = 1, \cdots, N$. We represent the n observations on the N individuals as a two-dimension array i.e. $n \times N$. Thus for one individual, we have $n \times 1$ with n repeated measurements of response variables denoted by

$$Y_i = (Y_{i1,Y_{i2}}, \cdots, Y_{in})^T \, . \tag{2.1}$$

In the analysis of data from a longitudinal study, the main interest is the mean response time

$$\mu_{ij} = E(Y_{ij}). \tag{2.2}$$

Also, defined as the expectation of each response Y_{ij}.From the conditional expectation $E(Y_{ij})$, the conditional variance of Y_{ij} is defined as

$$\sigma_j^2 = E\left[Y_{ij} - E(Y_{ij})\right]^2 = E\left[Y_{ij} - \mu_{ij}\right]^2 \tag{2.3}$$

Thus, conditional covariance between responses at two different occasions Y_{ij} and Y_{ik} is denoted by

$$\sigma_{jk} = E\left[\left(Y_{ij} - \mu_{ij}\right)\left(Y_{ik} - \mu_{ik}\right)\right] \tag{2.4}$$

and determines the linear dependence between the two responses. If the covariance is zero, then there is no linear dependence between the two occasions. The magnitude of the covariance depends on both the degree of dependence and the units of measurement between Y_{ij} and Y_{ik}.

Next, we define the conditional correlation between Y_{ij} and Y_{ik} by

$$\rho_{jk} = \frac{E\left[\left(Y_{ij} - \mu_{ij}\right)\left(Y_{ik} - \mu_{ik}\right)\right]}{\sigma_j \sigma_k}, \tag{2.5}$$

where σ_j and σ_k are the conditional standard deviations of both response variables. The correlation ρ_{jk} is a measure of dependence that is unit-less.

In longitudinal data the repeated measures for an individual are predicted to be positively correlated. Thus, we define the variance-covariance matrix for $Y_i = (Y_{i1,Y_{i2}}, \cdots, Y_{in})^T$ to be a twodimensional array of conditional variances and covariances as defined below

$$\mathrm{Cov}\begin{pmatrix} Y_{i1} \\ Y_{i2} \\ \vdots \\ Y_{in} \end{pmatrix} = \begin{pmatrix} \mathrm{Var}(Y_{i1}) & \mathrm{Cov}(Y_{i1}, Y_{i2}) & \cdots & \mathrm{Cov}(Y_{i1}, Y_{in}) \\ \mathrm{Cov}(Y_{i2}, Y_{i1}) & \mathrm{Var}(Y_{i2}) & \cdots & \mathrm{Cov}(Y_{i2}, Y_{in}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}(Y_{in}, Y_{i1}) & \mathrm{Cov}(Y_{in}, Y_{i2}) & \cdots & \mathrm{Var}(Y_{in}) \end{pmatrix}$$

$$= \begin{pmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_{nn} \end{pmatrix} \tag{2.6}$$

where $\mathrm{Cov}(Y_{ik}, Y_{ik}) = \sigma_{jk}$. Recall that the covariance of a variable with itself is a variance. Thus,

$$\sigma_{kk} = \mathrm{Cov}(Y_{ik}, Y_{ik}) = \mathrm{Var}(Y_{ik}) = \sigma_k^2 \tag{2.7}$$

Therefor, the variance-covaraince matrix of Y_i is defined as

$$\mathrm{Cov}(Y_i) = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_n^2 \end{pmatrix} \tag{2.8}$$

Also, we can define the correlation matrix as

$$\mathrm{Corr}(Y_i) = \begin{pmatrix} 1 & \rho_{12} & \cdots & \rho_{1n} \\ \rho_{21} & 1 & \cdots & \rho_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1} & \rho_{n2} & \cdots & 1 \end{pmatrix} \tag{2.9}$$

For continuous data we define Y_i as a vector of dimension n_i for the N individuals. Associated with each response, Y_{ij}, there is a $p \times 1$ vector of covariates

$$X_{ij} = \begin{pmatrix} X_{ij1} \\ X_{ij2} \\ \vdots \\ X_{ijp} \end{pmatrix} \tag{2.10}$$

Note that X_{ij} is a vector of covariates associated with Y_{ij}, the response variable for the i^{th} individual at the j^{th} occasion. The p rows of X_{ij} correspond to different covariates. There is a corresponding vector of covraites associated with each of the n_i repeated measurements on the i^{th} subject.

Define the matrix X_i be an ordered collection of the values of the p covariates for the i^{th} individual at each n_i occasion. Thus,

$$X_i = \begin{pmatrix} X_{i11} & X_{i12} & \cdots & X_{i1p} \\ X_{i21} & X_{i22} & \cdots & X_{i2p} \\ \vdots & \vdots & \ddots & \vdots \\ X_{in_i1} & X_{in_i2} & \cdots & X_{in_ip} \end{pmatrix} \tag{2.11}$$

Next, we consider a linear regression model for changes in the mean response over time and for relating the changes to the covariates,

$$\begin{aligned} Y_{ij} &= \beta_1 X_{ij1} + \beta_2 X_{ij2} + \cdots + \beta_j X_{ijp} + \varepsilon_{ij} \qquad (2.12) \\ j &= 1, \cdots, n_i \end{aligned}$$

where $\beta_1, \cdots, \beta_p$ are unknown regression coefficients relating the mean of Y_{ij} to its corresponding covariates and ε_{ij} are random errors, with mean zero.

Hence, ε_{ij} represents the deviations of the responses from their corresponding predicted means

$$E(Y_i|X_{ij}) = \beta_1 X_{ij1} + \beta_1 X_{ij1} + \cdots + \beta_p X_{ijp}. \tag{2.13}$$

Generally, $X_{ij1} = 1$ for all i and j, and β_1 is the intercept of the model.Thus, we can rewrite are regression model as

$$Y_i = X_i\beta + \varepsilon_i \tag{2.14}$$

where β and ε_i are both vector transposes. As noted before the mean zero of ε_i, implies

$$E(Y_i|X_i) = \mu_i = X_i\beta \tag{2.15}$$

where μ_i is a transposed vector for the i^{th} individual. Hence,

$$\mu_{ij} = E(Y_{ij}|X_i) = E(Y_{ij}|X_{ij}). \tag{2.16}$$

This model explains how the mean response interact with the covariates. Next, Y_i is assumed to have a conditional distribution that is multivariate normal, with response vector (15) and a covariance matrix

$$\sum_i = \mathrm{Cov}(Y_i|X_i) \tag{2.17}$$

2.1 Missing Data

In longitudinal studies the design and collection of data for every individual results in missing data as the rule. The first major implications for missing data is data sets will become unbalanced over time since not all individuals have the same number of repeated measurement at a common set of occasion. The second is loss of precision is directly proportional to the amount of data missing, which determines how accurate we can estimate the change in mean response over time. Lastly, the missing data mechanism is required to tenable for validity of any method of analysis used. Understanding the reason for missingness i.e. MCAR and MAR lead way to alternative methods for handling missingness in longitudinal studies. The data mechanism can be understood as a model that describes the probability that a response is missing at any observation [13].

Missing data can be referred to as missing completely at random (MAR) or missing at random (MAR). Longitudinal data are defined as MCAR when missingness in Y_i does not depend on observed or unobserved components of Y_i. MNAR means that the probability of being missing varies for the reasons that are unknown to us. MCAR is often unrealistic for data at hand. For example a weighing scale mechanism may wear out over time, producing more missing data as time progresses, but we may fail to note this [12].

If the data is said to be MAR means the probability of being missing is the same for all cases. In other words, the probability that responses are missing depends on the set of observed response, but in unrelated to certain missing values. Implying that the cause of missing data is unrelated to the data. Example is a weight scale ran out of batteries. If the probability of being missing is the same only within groups defined by the observed data, then the data are missing at random (MAR). MAR is a much broader class. If neither MCAR nor MAR holds, then we speak of missing not at random (MNAR).

Let the response indicator variable be a value of 1 when the measurement is obtained and 0 otherwise, for n repeated measures of the response variable on the same individual. Thus, a complete set of responses has $n \times 1$ response vector denoted by

$$Y_i = (Y_{i1}, Y_{i2}, \cdots, Y_{im})' \tag{2.18}$$

Due to missingness, some components of Y_i are not observed for an individual, thus we define these instances $R_{ij} = 0$ and observed instances as $R_{ij} = 0$. Thus, R_i is an $n \times 1$ vector of response variables with missing and non-missing response indicators as defined below

$$R_i = (R_{i1}, R_{i2}, \cdots, R_{im})' \tag{2.19}$$

Missingness is not accounted for in the covariates; that is, we separate R_i into two components Y_i^O for observed and Y_i^M for missing response on the i^{th} subject. Additionally, R_i can be thought of as a stratification variable that divides the target population into sub-populations based on missing data patterns.

Longitudinal data are MCAR when R_i is independent of both Y_i^O and Y_i^M. Hence, consider the bivariate case $Y_i = (Y_{i1}, Y_{i2})'$, where Y_{i1} is fully observed and Y_{i2} is sometimes missing. If Y_{i2} is MCAR, then

$$\Pr\left(R_{i2} = 1 | Y_{i1}, Y_{i2}, X_i\right) = \Pr\left(R_{i2} = X_i\right). \tag{2.20}$$

Hence, the probability of Y_{i2} is missing does not depend on the observed value of Y_{i1} or Y_{i2} that should have been obtained. Missingness in Y_{i2} is a chance mechanism that does not depend on Y_i observations. Longitudinal data are MAR when R_i is conditionally independent of Y_i^M, give Y_i^O is

$$\Pr(R_i | Y_i^O, Y_i^M, X_i) = \Pr(R_i^O, X_i) \tag{2.21}$$

Missing data is said to not missing at random (NMAR) wen the probability that missing responses are related to certain values that should be obtained. Hence, the conditional distribution of R_i, given Y_i^0, is related to Y_i^M, and

$$\Pr(R_i|Y_i^O, Y_i^M, X_i) \tag{2.22}$$

depends on some elements of Y_i^M. NMAR is often refereed to as non-ignorable due to the fact that the missing data mechanism cannot be ignored in order to make inferences about the distribution of the complete longitudinal response.

Let dropout be defined as the special case where if Y_{ik} is missing, then $Y_{ik+1}, \cdots, Y_{in}$ is also missing. Hence, if, $R_{ik} = 0$, then $R_{ik+1} = R_{in} = 0$. When dropout occurs, the key issues is identifying as MAR, MCAR, or MNAR.

CHAPTER III

METHODOLOGY

3.1 Last Observed Carried Forward

A simple yet effective missing data imputation method in clinical trails is the Last Observed Carried Forward (LOCF) for longitudinal data. A previous FDA preferred method of analysis, LOCF is a conservative method that uses only observed data within in the variable to compute missingness. The LOCF simply takes the last observed non-missing value and fills in the missing value at a later point in the variable [15].

3.2 Regression

In regression methods for imputing longitudinal data, the monotone missing response values are imputed sequentially using all preceding responses as predictors [9]. Hence, a series of regression models Y_{ik}, given $Y_{i1}, \cdots, Y_{ik-1}$ and X_i are fitted to the observed d ata. For continuous response variables, standard linear regression models are used to generate imputations as defined below

$$E(Y_{ik}|Y_{i1}, \cdots, Y_{ik-1}, X_i) = \gamma_1 + \gamma_2 Y_{i1} + \cdots + \gamma_k Y_{ik-1}. \tag{3.1}$$

Where the linear regression model is fitted using non-dropped data on the k^{th} occasions. However, if there is no dependence on X_i, then it's defined as

$$E(Y_{ik}|Y_{i1}, \cdots, Y_{ik-1}, X_i) = Z'_{ik}\gamma. \tag{3.2}$$

Where the model is fitted to data on the N_k subjects who have not dropped out by the k^{th} occasion and Z_{ik} denotes a vector created from any Y_{ik-1} and subset of X_i. Also, γ denotes a $q \times 1$ vector of regression parameters relating Y_{ik} to the preceding responses and covariates. $E(Y_{ik}|Y_{i1})$ can be estimated via ordinary least squares, which produces estimates of the regression parameters $\hat{\gamma}$ and their associated covariance matrix,

$$\hat{\text{Cov}}(\hat{\gamma}) = \hat{\sigma}^2 \left(\sum_{i=1}^{N_k} Z_{ik} Z'_{ik} \right)^{-1}, \tag{3.3}$$

where $\hat{\sigma}^2$ is an estimate of the residual variance and Z_{ik} is the design vector for the regression of Y_{ik} on any Y_{ik-1} and subset of X_i. To account bias, a random variation is needed to account for the uncertainty of the imputations. Hence, adding the predicted value for a random draw Y_{ik} from the residual distributions Y_{ik} for any Y_{ik-1} and subset of X_i. This corresponds to adding a random error. To account for additional sources of variation, each imputation should be based on a set of estimated regression coefficients and an estimate of the residual variance σ^2 where the imputation process above treats them as fixed(known) instead of sample estimates. These values should be drawn randomly from what is known as the posterior distribution from Bayesian statistics.

To summarize, the regression methods let Y^*_{ik} be the produced imputed values from the missing Y_{ik}, γ^* the new regression parameters, and σ^{*2} be the residual variance. When all are drawn from their posterior distribution the account of uncertainty in estimating γ and σ^2, the residual variance is randomly drawn by

$$\sigma^{*2} = \frac{(N_k - q)\,\hat{\sigma}^2}{\chi^2}, \tag{3.4}$$

where $N_k - q$ denotes the degrees of freedom for the residual variance, and χ^2 is a random draw from a chi-square distribution with $N_k - q$ degrees of freedom. The regression parameters γ^* are randomly drawn from a multivariable normal distribution with mean equal to the estimated regression parameters $\hat{\gamma}$ and with covariance matrix,

$$\hat{\text{Cov}}(\hat{\gamma}) = \sigma^{*2} \left(\sum_{i=1}^{N_k} Z_{ik} Z'_{ik} \right)^{-1}, \tag{3.5}$$

where Z_{ik} is the design vector for the regression of Y_{ik} for any Y_{ik-1} and subset of X_i.
Let missing values for Y_{ik} be Y^*_{ik} be imputed by the following prediction:

$$Y^*_{ik} = Z'_{ik}\gamma^* + \varepsilon^*, \tag{3.6}$$

where, for each Y^*_{ik}, ε^* is randomly drawn from a normal distribution with mean zero and σ^*.

3.3 Multiple Imputation

Multiple imputation is the a simple technique of imputing plausible values for missing data. If the data is filled with only one plausible value, then subsequent analysis of the completed data is problematic[3]. Uncertainty around the surrounding imputed values are not accounted for with conventional methods for standard error estimation, which leads to anti-conservative where the nominal p-values and confidence intervals are s maller. Hence, Multiple imputation corrects this anti-conservative by filling the plausible values multiple times creating multiple completed data sets. The combined sets provide a single estimate of the parameter with standard errors that reflect the uncertainty inherent in the imputation data [23].

Assume $m > 1$ imputed data sets are created, then m different estimates of the regression parameters β, say $\hat{\beta}^k$ (for $k = 1 \cdots, m$) can be calculated from each individual m data sets [22].Additionally, the m sets produce m estimates of the covariance $\hat{\beta}^k$. The multiple imputation estimate of β is simple the unweighted average of the m estimates,

$$\hat{\beta} = \bar{\beta} = \frac{1}{m} \sum_{k=1}^{m} \hat{\beta}^k \tag{3.7}$$

The estimated covariance of $\hat{\beta}$ is given by

$$\hat{\text{Cov}}\left(\hat{\beta}\right) = W + (1 + m^{-}1)B, \tag{3.8}$$

where

$$W = \frac{1}{m}\sum_{k=1}^{m} \hat{\text{Cov}}\left(\hat{\beta}^{k}\right) \tag{3.9}$$

and

$$B = \frac{1}{m-1}\sum_{k=1}^{m} \left(\hat{\beta}^{k} - \bar{\beta}\right)\left(\hat{\beta}^{k} - \bar{\beta}\right)'. \tag{3.10}$$

$\hat{\text{Cov}}\left(\hat{\beta}\right)$ combines both the within-imputation (W) and between imputation (B) variability's.

The missing data is fill in m times to create m completed data sets, where they are analyzed using statistical methods. Then, the results from m analyses of the completed data sets are combined as mentioned above.

In the report we will cover five methods of multiple imputations for handle missing data. In general, proper imputation is should be drawn at random from the conditional distribution of the missing data given the observed data. Thus, Y_i^M is obtained randomly from $f(Y_i^M|Y_i^O, X_i)$ by assuming that the missingness is MAR and the predictive distribution of the missing data based on the observed data does not depend on R_i. Hence,

$$f(Y_i^M|Y_i^O, X_i) = f(y_i^m|Y_i^O, X_i, R_i) \tag{3.11}$$

When randomly sampling values from (27), they identify as either monotone missing data pattern or non-monotone. Before describing the specific methods it's worth noting that these methods assume the data set is structured in a "wide" rather than "long" format.

Longitudinal data can be coded into "long" and "wide" formats. A wide data set will have one record for each individual. The observations made at different time points are coded as different columns. In the long format there will be multiple records for each individual. Some variables that do not vary in time are identical in each record, whereas other variables vary across the records, and an "id" variable that groups the records from the same person. Note that the concepts of long and wide are general and apply to cross-sectional data. While wide has no redundancy or repetition, long format is better at handling irregular and missed visits and has an explicit time variable available that can be used for analysis. Long is used for ANOVA and MANOVA techniques for repeated measures and structural equation models for longitudinal data.

Monotone missing data patterns arise in longitudinal studies when missingness occurs through dropout. The first response Y_1 is observed but subsequent response are missing due to dropout. The missing values in the next response can be imputed with a regression model to predict Y_{i2} from Y_{i1} and X_i.

3.3.1 Mice Imputed 2 Level Class Predictive Mean Matching

The use of a single-level imputation methods ignore hierarchical group structure in longitudinal data. An example of a single-level imputation is LOCF. Also, a typical fix for missing values in a level-2 predictor was to delete all records in the cluster. Disregarding the potential impact on the analyses, the problem of incomplete level-2 predictors received less attention than missingness in level-1 predictors. To address thesis issues, a more recent attempt is to create two datasets, one with level-1 data, and one with level-2 data and run separate imputations within each dataset while using the results from one in the other [7].

The mice package used in this research was the 2lonly.pmm (2LPMM) method, which aggregates a level-1 predictors, and imputes the level-2 variables by the normal model and by predictive mean matching.

3.4 Bayesian Neural Network

Bayesian networks(BN) are statistical tools for encoding probabilistic relationships with direct acyclic graphs. Generally, applied to population health and social science equations [14].

3.4.1 Graph Theory

Let $H = (\mathbf{V}, \gamma)$ be an nonempty set of V of nodes and vertices and a finite set A of vertices where (u, v) define either an ordered pair or an unordered pair of nodes. If (u, v) are adjacent they are considered neighbors otherwise (u, v) are incident on an arc. Additionally, if (u, v) create an arc they can be represented as direct $u \rightarrow v$ where u is the head and v is the tail. Otherwise, they are undirected arcs or edges denoted by $u - v$.

Let $H = (\mathbf{V}, \gamma)$ exist where the node set $\mathbf{V} = \{A, B, C, D, E\}$ and γ is the diagrams depicted below in Figure 1, then H can be undirected, partially directed, or mixed graph when it contains undirected and/or directed arcs. In the partially directed graph from Figure 1, denoted by $H = (V, \gamma, \alpha)$ the set V is characterized by the edges set $\gamma = \{(A - C), (A - D), (D - C)\}$ and arc set $\alpha = \{(D \rightarrow E), (E \rightarrow B)\}$. In the directed graph from Figure 1, the arc set $\alpha = \{(C \rightarrow E), (C \rightarrow A), (C \rightarrow D), (D \rightarrow B), (A \rightarrow B)\}$ creates the parent nodes (A, D) for node B, where node C is the ancestor.

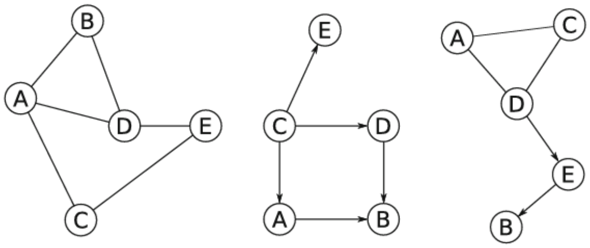

Figure 3.1: An undirected , a directed , and a partially directed graphs [14]

3.4.2 Graph Structure

As seen in Figure 1 graphs can be represent in many different structures with the simplest structure presented as an empty graph or by a saturated graph where every node is connected by and edge. For graphs that map real-world abstractions generally are consider either sparse or dense.

Paths are sequences of arcs or edges connecting two nodes denoted as the sequence of vertices $v_i = (v_1, v_2, \cdots, v_n)$ incident on those arcs.

If the path passes each arc only once then the arcs connecting v_i are assumed to be unique. Otherwise, if the path starts and ends at $v_1 = v_n$, then it's defined as a cycle. thus, a graph that contains no cycles or loops is defined as a cyclic. For acylic graphs each edge represents a direct conditional dependency and and pairs of nodes not connected are considered conditionally independent of each other. Each node is associated with a probability function that inputs a set of values from the parent nodes variables and outputs a probability or probability distribution of the variable represented by the node[18].

3.4.3 Statistical Introduction

Let A and B be events with probability of said event occurring defined as $P(A)$ and $P(B)$. Let $P(A|B)$ is a conditional probability defined as the probability of event A occurs given that event B has already occurred.

Thus, Bayes' Theorem is stated mathematically as the following equation

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{3.12}$$

where $P(B) \neq 0$.

3.4.4 Probability Factorization

Let $\perp\!\!\!\perp_H$ be defined as graphical separation created from the absences of a particular arc and $\perp\!\!\!\perp_P$ define the dependencies between variables [16]. A graph H is an independence map (I-map) of the probabilistic dependence structure, P of random variable X if there is a one-to-one mapping between X and the nodes V in H, such that for all disjoint subsets $A, B, C \in V$ of

$$A \perp\!\!\!\perp_P B|C \Leftarrow A \perp\!\!\!\perp_H B|C. \tag{3.13}$$

Similarly, H is a dependency map (D-map) of P of X defined by

$$A \perp\!\!\!\perp_P B|C \Rightarrow A \perp\!\!\!\perp_H B|C. \tag{3.14}$$

The combination of a dependence and independence maps creates a perfect map defined as

$$A \perp\!\!\!\perp_P B|C \Leftrightarrow A \perp\!\!\!\perp_H B|C, \tag{3.15}$$

where P is isomorphic to H.

3.4.5 D-Separation

Given a set of random variables $X = \{X_1, X_2, \cdots, X_p\}$ and a directed acyclic graph defined as $\text{DAG}_H = (V, \gamma)$, Bayesian networks are a class of graphical models that provide a concise representation of the probabilistic dependencies between the X and DAG_H where $v_i \in V$ corresponds to X. Let DAG_H be constructed by three disjoint subsets of nodes $A, B, C \in \text{DAG}_H$ where C is D-separate from the other nodes and $A \perp\!\!\!\perp_H B|C$ denotes a sequence of arcs between A, B and v that satisfies the following conditions:

(i) v has converging arcs i.e. parents and none of v descendants are in C.

(ii) v is in C and does not have converging arcs.

A direct result of D-separation is the Markov property of Bayesian networks, which enables the representation of the joint distribution in respect to X as a product of conditional probability distribution. Hence, it's a direct application of the chain rule [11], defined as the factorization of the joint probability distribution P_X for discrete X given by

$$P_X(X) = \Pi_{i=1}^{p} P_{x_i}(X_i|\Pi_{X_i}). \tag{3.16}$$

Note: Π_{X_i} is the set of parent nodes of X_i. In the case of a continuous X, the factorization of the joint density function f_X is given by

$$f_X(X) = \Pi_{i=1}^{p} f_{X_i}(X_i|\Pi_{X_i}). \tag{3.17}$$

Let $H = (V, \gamma_i)$ where $V = \{A, B, C\}$ and $\gamma_1 = \{(A \to C), (B \to C)\}$, then the probabilistic structure is $A \not\perp\!\!\!\perp_H B|C$ and the

$$P(A,B,C) = P(C|A,B)P(A)P(B). \tag{3.18}$$

If $\gamma_2 = \{(A \to C), (C \to B)\}$ or $\gamma_3 = \{(C \to A), (C \to B)\}$ with probabilistic structure $A \perp\!\!\!\perp_H B|C \Rightarrow A \perp\!\!\!\perp_P B|C$, then

$$P(A,B,C) = P(B|C)P(C|A)P(A) \tag{3.19}$$

$$= P(A|C)P(B|C)P(C) \tag{3.20}$$

3.4.6 Fundamental Connections

Considering the fundamental connections seen in γ_i, γ_1 C has parent nodes A,B meaning it's not D-separate resulting in (3.14) by the Markov property in (3.16) where $\Pi_A = \{\emptyset\}$, $\Pi_B = \{\emptyset\}$, and $\Pi_C = \{A, B\}$ [9]. Therefor, C depends on the joint distribution of A and B. Nonetheless, C is independent in both γ_2 and γ_3. Thus, for the serial connection in γ_2, we have (3.19) where $\Pi_A = \{\emptyset\}$, $\Pi_B = \{C\}$, and $\Pi_C = \{A\}$ by the Markov property in (3.16). For the diverging connection in γ_3, we have (3.20) where $\Pi_A = \{C\}$, $\Pi_B = \{C\}$, and $\Pi_C = \{\emptyset\}$ by the Markov property in (3.18).

3.4.7 Equivalent Structures

The serial and diverging connections result in equivalent factorization obtained by repeated application of Bayes' theorem from the other. Thus, such probabilistically equivalent structures are defined as Markov equivalent. By definition equivalence structures is symmetric, reflexive, and transitive each set of Markov equivalent forms an Markov class. Generally, only arcs whose direction is needed to identify an Markov class belong to at least one v-structure [1].Markov classes are generally represented by completed partially directed acyclic graphs (CPDAGs), whose v-structures and arcs that path into v-structures or cycles are directed. Such arcs are called compelled because their direction is determined by equivalence classes. Redirecting compelled arcs results in another network in the same equivalence class if a new v-structure or cycle is not added.

3.4.8 Markov Blanket

The Markov blanket [27] represents the set of nodes that completely D-separates a particular node from the graph. By definition the Markov blanket of a node $A \in V$ is the minimal subset S of V such that

$$A \perp\!\!\!\perp_P V - S - A | S, \tag{3.21}$$

which for any Bayesian network is the parents of A, the children of A, and all other nodes sharing a child with A.

Markov blankets facilitate the comparison of Bayesian networks with graphical models based on undirected graphs, communally referred to as Markov networks or random fields [10]. Hence, on a related note a DAG can be transformed in the undirected graph of the corresponding Markov networks by the following steps:

1. Connect the nonadjacent nodes in each v-structure by an undirected arc, which is equivalent to adding an undirected arc between a node centered upon the Markov blanket.

2. Ignore the direction of other arcs by replacing said arcs with edges.

3.4.9 Inductive Causation Algorithm

1. For each pair of variables A and B in V search for the set $S_{A,b} \subset V$ (including $S = \emptyset$) such that A and B are independent given $S_{A,b}$ and $A, B \notin S_{A,b}$. If there is no such set, place an undirected arc between A and B.

2. For each pair of non-adjacent variables A and B with a common neighbor C, check whether $C \in S_{A,b}$ If this is not true, set the direction of the arcs $A - C$ and C_B to $A \rightarrow C$ and $C \leftarrow B$.

3. Set the direction of arcs which are still undirected by applying recursively the following two rules:

   (i) If A is adjacent to B and there is a strictly directed path from A to B (a path leading from A to B containing no undirected arcs) then set the direction of $A - B$ to $A \rightarrow B$

   (ii) If A and B are not adjacent but $A \rightarrow C$ and $C - B$, then change the latter to $C \rightarrow B$

4. Return the resulting completed partially directed acyclic graph.

3.4.10 Static Bayesian Network Modeling

Learning is defined as the task of fitting the Bayesian network [17] performed by two different steps corresponding to the model selection and parameter estimation techniques in the classic statistical models. The first step defined as the structure learning identifies the graph structure of the Bayesian network, which should be the minimal I-map of the dependence structure of the data. If not, then it should result in a distribution as close as possible to the I-maps probability space. Several algorithms with a variety of theoretical backgrounds and terminology have been proposed in literature for structure learning, but all of them fall under three broad categories as follows: constraint-based, score-based, and hybrid algorithms. Alternatively, the network structure can be built manually from the domain knowledge of an expert and prior information available data.

The second step is called parameter learning which implements the estimation of the parameters of the global distribution. Parameter learning can be performed efficiently by estimating the parameters of the local distributions implied by the structure from the first step.

3.4.11 Constraint-Based Structure Learning Algorithms

Constraint-based structure learning algorithms are based on seminal work of Pearl on maps and it's application to causal graphical models. The inductive causation (IC) algorithm [5] and conditional independence tests provides the framework for learning the structure of Bayesian networks. From the details in the IC algorithm the first step identifies the pairs of variables that will be connected by an arc by ignoring direction. The variables can not be D-separated, which can also be seen as a backwards selection procedure starting from the saturated model with a complete graph and pruning based on statistical test from conditional independence. The second step identifies the v-structures among all the pairs of nonadjacent nodes A and B with a common neighbor C. By definition, v-structures are the only fundamental connection where two adjacent nodes are dependent conditionally on the third node. Hence, if A,B, and C create a subset v-structure centered on C, then A and B are D-separate. Thus verified by performing a conditional independence test of A and B against every possible subset of their common neighbors that includes C. The last step of the IC algorithm identifies compelled arcs and orients them recursively to obtain the completed partially DAG (CPDAG) describing the equivalence class.

A major problem of the IC algorithm is the previous steps cannot be applied to any real-world problem due to the exponential number of possible conditional independence relationships, which led to the development of the following improved algorithms:

1. PC: a backward selection procedure from the saturated graph
2. Grow-Shrink (GS) a simple forward selection Markov blanket detection approach.
3. Incremental Association (IAMB): a two-phase selection scheme based on a forward selection followed by a backward one.
4. Fact Incremental Association (FAST-IAMB): A variant of IAMB which uses speculative stepwise forward selection to reduce the number of conditional independence test.
5. Interleaved Incremental Association (Inter-IAMB): Variant of IAMB which uses forward stepwise selection to avoid false positives in the Markov blanket detection phase.

Majority of these methods learn the Markov blanket of each node in the network, which greatly simplifies the identification of neighbors of each node. Hence, the number of conditional independence tests performed and it's complexity by the learning algorithm is reduced.

3.4.12 Score-Based Structure Learning Algorithms

Score-based structure learning algorithms also known as search-and-score, model the application of general heuristic optimization techniques to address the issue of learning the structure of a Bayesian network. Hence, a network score is given to each candidate network reflecting goodness of fit to which the algorithm attempts to maximize. Such examples of this class of algorithms are listed below:

1. Greedy search algorithms i.e. hill-climbing with random restarts or tabu search. Such algorithms explore the search space starting from a network structure and adding, deleting, or reversing arcs to maximize the algorithm.

2. Genetic algorithms mimic natural evolution through the iterative selection of the best fitting models and hybridization of their characteristics. Thus, the search space is explored via the crossover and mutation stochastic operators.

3. Simulated annealing performs a stochastic local search by accepting changes that increase the network score and simultaneously allows changes that decrease the score by a probability inversely proportional to said decrease score.

3.4.13 Hill-Climbing Algorithm

1. Choose a network structure H over V, generally empty.

2. Compute the score of H, denoted as S_H =Score(H).

3. Set Max $= S_H$

4. Repeat the following steps until max score is maximized:

   (a) For all arc addition, deletion, and or reversal no resulting in a cyclic network:

      (i) Compute the score of the modified network $H', S_{H'}$=Score(H'):

      (ii) If $S_{H'} > S_H$ set $H = H'$ and $S_H = S_{H'}$.

      (iii) Update max-score with the new value of S_H

5. Return the directed acyclic graph H.

3.4.14 Hybrid Structure Learning Algorithms

Hybrid structure learning algorithms combines score and constraint based algorithms to offset shortcomings and create a reliable network structures. The two well known Hybrid structures are Sparse Candidate algorithm (SC) [25] and Max-Min Hill-Climbing (MMHC)[19] , which both use a restrict and maximize step. The restrictive step, the candidate set for the parents oe each node X_i is reduced from the entire node set V to a smaller set $C_i \subset V$ of nodes whose behavior is related to that of X_i, which results in a smaller and regular search space. Next, the maximizing step finds the network that maximizes the score function subjected to constraints imposed by the C_i sets.

In the Sparse Candidate algorithms both steps are applied iterative until the network maximizes the network scores. Note, the heuristics used to perform both methods is left to the user. Also, while the MMHC algorithm's restrict and maximize process is only used once, the MMPC heuristic is used to learn the candidate sets C_i and a hill-climbing greedy search to find the optimal network.

3.4.15 Sparse Candidate Algorithm

1. Choose a empty or non-empty network structure H over V.

2. Repeat the following steps until convergence:

   (a) Restrict Step: Select a set C_i of candidate parents for each node $X_i \in V$ that includes the parents of X_i in H'.

   (b) Maximize Step: Find the network structure H' that maximizes $S_{H'}$ among the networks where parent of each node X_i are included in the correct corresponding set C_i.

   (c) Set $H = H'$.

3. Return the directed acyclic graph H.

3.4.16 Bayesian Neural Network Models

The Bayesian Network (BN) increase the computation time considerable. The structural expectation-maximization algorithm makes BN structures learning from incomplete data by structuring over the search space not depending on the parameters. One of the method used was a BN that calculated the predicted values by plugging in the new values for the parent of node in the local probability distribution of node extracted from the fitted.

The next method called BN Monte Carlo (BNMC) the predicted values are computed by averaging likelihood weighting simulations performed using all the available nodes as evidence. The number of random samples which are average for each new observation is controlled by the *n* optional argument. Since, the simulated data was discrete the predicted level is the highest conditional probability. The last method call BN exact the predicted values are computed using exact inference (BNEI), which are maximum posterior estimates calculated from a junction trees and belief propagation.

The Bayesian network created in both the real and simulated dataset was over the missing in each response variable by creating two dataset, where one contains a set of fully observed variables and the other with a set of completely unobserved variables. By defining a new DAG G^* over the union of both the latent (missing) and observed variables, the network models them by adding them as a column of NA to the data used for learning to prevent confounding. For MAR in the response variables, the causal network is augment again for precision. This is done by creation of the causal graph (missingness graph) where the vertices in are partitioned into disjoint subsets.

3.5 *k*-Nearest Neighbor

In statistics, the K-nearest neighbors algorithm (*k*-NN) is a non-parametric supervised learning method. In other words, it makes minimal assumptions regarding the underlying distribution of the data to build a function who's input and output values train a model. Hence, it maps new data on expected output values for unseen instances in a reasonable way measured by generalization error [20].

For the *k*-NN classification model, the output is a class membership from an object being classified by a plurality vote of its neighbors. The object being assigned to the class most common among it's k nearest neighbors.

3.5.1 Nearest Neighbor Classification

Neighbors-based classification,the most common used technique, is an instance based learning algorithm that does not construct a general internal model,but instead instances of the training data. The optimal choice of the value **k** is highly data-dependent and in general suppresses the effects of noise. The larger the **k** value the less distinct the classification boundaries[2].

The basic nearest neighbors uses uniform weights where the value assigned to the query point is computed from simple majority voting. Distances weights can be assigned where the weights are proportional to the inverse of the distance from the query points or a user-defined functions of the distance can be used to compute the distance weights [24].

3.5.2 Nearest Neighbor Algorithms

The nearest neighbor is calculated by three different methods as seen below:

Brute Force:

The most naive neighbor search implementation involves the brute-force computation of distances between all pairs of points in the dataset. For N samples in D dimensions, the brute force method is very competitive for small data samples which scales as $O[DN^2]$. However, as the number of samples N grows, quickly becomes infeasible.

K-D Tree:

To address the computational inefficiencies of the brute-force algorithm, a variety of tree-based data structures have been invented to reduce the required number of distances calculations by efficiently encoding aggregate distances information for the sample. Thus, the computational cost of a nearest neighbors can be reduced to $O[DN\log(N)]$, which is a significant improvement.

Ball Tree:

To address the inefficiencies of a KD Trees in higher dimension, the ball tree data structure was developed to partition data into series of nesting hyper-spheres. The nest result in a higher cost, but very efficient with higher dimension data structures.

The idea of the nest is to divide the data by a centroid C and radius r, such that each point in the node lies within the hyper-sphere defined r and C. Hence, by the triangle inequality a single distance between a test point and the centroid determines a lower and upper bound on the distance to all points within the node. Thus, by spherical geometry of the ball tree nodes it handles the higher dimension more efficiently than the KD tree.

3.5.3 Choice of Nearest Neighbor Algorithms

All calculations for k-NN where calculated using scikit-learn, which is a simple and efficient machine learning tool for predictive data analysis in Python. The optimal algorithm of a given dataset is automatically chosen by sklearn and is dependent on the following factors:

1. Number of N and D i.e. features.

   Intrinsic dimensionality of the data and/or sparsity of data. Intrinsic dimensionality refers to the dimension of $d \leq D$ manifold on which the data lies i.e. linearly or non-linearly embedded in the parameter space. Sparsity is the degree to which the data fill the parameter space.

   Hence, the larger the N and D the quicker the Query time grows. Hence, for smaller data size with less features, brute force is optimal. For large data size with less dimensions KD-Tree is optimal. Otherwise, Ball Tree algorithm is optimal.

2. The number of neighbors k requested for a query point.

   Brute force query time is largely unaffected by the value k. While, Ball and KD tree query time will become slower as k increases,which his caused by two reasons. The first because the larger k leads to necessity to search a larger portion of the parameter space. Second, using $k > 1$ requires internal queuing of results as the tree is traversed. Also, as k becomes larger compared to N, the ability to prune branches in a tree-based query is reduced.

3. The number of query points.

   Both the ball and KD tree requires a construction phase that is amortized over many queries. Hence, the smaller the queries the less cost and vice versa. If two few queries, then brute force is optimal.

3.5.4 Fitting and Evaluating the *k*-NN Model

Once, the *k*-NN model is created from our Nearest Neighbor Algorithm we use it to fit our training data. Hence, we pass both the x_{train} and y_{train} for the model to learn. The output is defined as *k*-NN fit model.Next, a prediction model defined as y_{pred} is created by passing x_{test} though the *k*-NN fit model. Thus, to evaluate the models accuracy we compare y_{pred} to the y_{test}. Accuracy is computed from how many times y_{pred} was able to correctly identify y_{test}.

Additionally, precision and F1-score are also calculated. Precision is a model evaluation and performance metric that corresponds to the fraction of values that actually belong to a positive class out of all of the values which are predicted to belong to that class. Precision is also known as the positive predictive value (PPV). F1-score computes the average of precision and recall, where the relative contribution of both of these metrics are equal to F1 score. The best value of F1 score is 1 and the worst is 0.

CHAPTER IV

SIMULATION STUDY

4.1 Methods and Motivation

The use of the Monte Carlo simulation is pertinent in predicting the probability of methods outcomes when the potential for missingness within random variables is present. The Monte Carlo simulations helps explain the impact of bias and uncertainty in imputing missing values at different sample sizes. Although the use of real data set is a crucial aspect of this imputation analysis, it unfortunately is only a sample. By simulating a population we can implement a range of 500-2000 simulations to analysis both local and asymptotic behavior at varying sample sizes of said population. Given the extreme flexibility that simulations afford, the use of simulations is especially common in trials with complicating factors such as

1. interim analyses for futility or for overwhelming efficacy,

2. multiplicity approaches covering multiple time points/endpoints,

3. or adaptations built into the designs adaptations built into the designs (e.g., dropping or adjusting the randomization ratio as a function of the accruing data).

Of course, power calculations can also be simulated for relatively straightforward clinical trial designs.

Additionally, with a simulated population of 70,000 we explore sensitivity analysis of each method in respect to each longitudinal time point to measure effectiveness. For example, LOCF method with small percentages of missingness and large enough sample size is arguable a better method than linear regression. In clinical trials it's still the preferred method for patient dropouts and uses only observed data within the response variable to impute missingness. From an ethical standpoints, I understand why it is a FDA favorite. Similarly, *k*-NN uses only observed data points across all response variables to create instance based learning algorithms to impute values. Bayesian neural networks allow for the incorporation of prior knowledge into the learning process, which improves model performance when data is limited or noisy.

4.1.1 Real Data Settings

The data set used was provided by Boston College on the "National Longitudinal Survey" over the years 1968-1988 (with gaps) of about 4,711 young working woman aging from 14–26 years. There are 28,534 observations in total and of those 13,452 observations contained no MAR/MNAR data. Within the complete case subset the sample has the following defined continuous variables:

1. Age: ranging from 14-26, normally distrusted.
2. Hours: usual hours worked within the week.
3. Tenure: job tenure in years.
4. wks-work: weeks worked in the past year.
5. ttl-exp: total work experience.
6. ln-wage: $ln\left(\frac{\text{Wage}}{\text{GNP deflator}}\right)$

To test the efficacy of the different methods inside of a real dataset we simulated similar scenarios of MAR in ttl-exp. We undertook this approach because there was no missing data in total-work experience and needy a way to compare results to that of our simulated data. Hence, to simulate missingness, samples sizes of $n = \{100, 200, 500, 1000\}$ where created at random with the following MAR percentages scenarios:

I Scenario 1: 0% y_{1i}, 10% y_{2i}, and 20% y_{3i}

II Scenario 2: 0% y_{1i}, 20% y_{2i}, and 30% y_{3i}

III Scenario 3: 0% y_{1i}, 30% y_{2i}, and 50% y_{3i}

Thus, we define y_{ij}^{S} to be total-work experience with MAR created from one of the scenarios where $i = \{1, \cdots, n\}$ for $n \in N$, $j = \{2, 3\}$ and $S = \{1, 2, 3\}$. Note: baseline total-work pressure,y_{i1} was not simulated with MAR.

Below is a diagram of the Bayesian Neural Network created from the real dataset

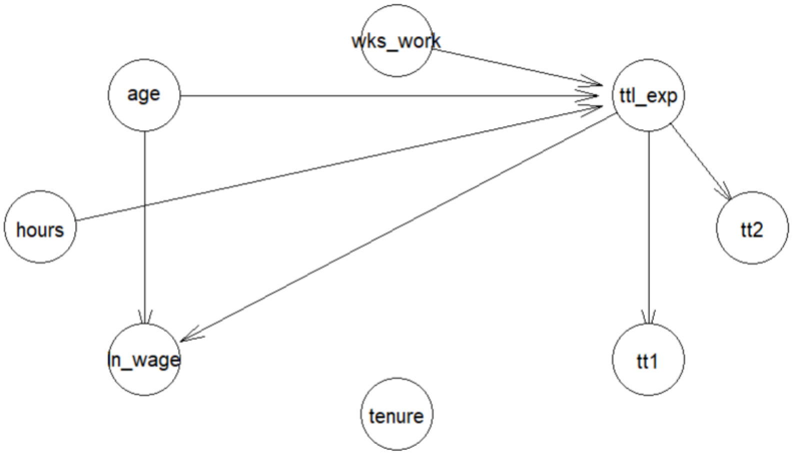

Figure 4.1: Diagram of Bayesian Neural Network of Real Data

4.1.2 Simulation Settings

In an effort to mimic a clinical trial, a simulated population of $N = 70,000$ was created to compare blood pressure at three different time intervals defined below (51). Define Sex and treatment (Trt) as dichotomous variables where treatment represents both the placebo and treatment patients. Also, let both age $\sim N(\mu = 50, \sigma = 5)$ and error, $\varepsilon \sim N(\mu = 0, \sigma = 2)$ be normally distributed.

Thus,

$$y_{ij} = \beta_0 + \beta_1 \text{Age}_i + \beta_2 \text{Sex}_i + \beta_3 \text{Trt}_i + \varepsilon_i, \tag{4.1}$$

where $i = \{1, \cdots, n\}$ for $n \in N$ and $j = \{1, 2, 3\}$ for each number of blood pressures. Hence, Baseline blood pressure was computed by

$$y_{i1} = \frac{59}{20}\text{Age}_i + \frac{5}{2}\text{Sex}_i + \varepsilon_i. \tag{4.2}$$

The second and third measurements for blood pressure were computed by

$$y_{i2} = -2 + y_{1i} + \frac{1}{2}\text{Trt}_i \tag{4.3}$$

and

$$y_{i3} = -4 + y_{1i} - \frac{3}{4}\text{Trt}_i. \tag{4.4}$$

To simulate missingness, samples sizes of $n = \{100, 200, 500, 1000\}$ where created at random with the following MAR percentages scenarios:

I Scenario 1: 0% y_{1i}, 10% y_{2i}, and 20% y_{3i}

II Scenario 2: 0% y_{1i}, 20% y_{2i}, and 30% y_{3i}

III Scenario 3: 0% y_{1i}, 30% y_{2i}, and 50% y_{3i}

Thus, we define y_{ij}^{S} to be blood pressure with MAR created from one of the scenarios where $i = \{1, \cdots, n\}$ for $n \in N$, $j = \{2, 3\}$ and $S = \{1, 2, 3\}$. Note: baseline blood pressure,y_{i1} was not simulated with MAR.

Below is a diagram of the Bayesian Neural Network created from the real dataset

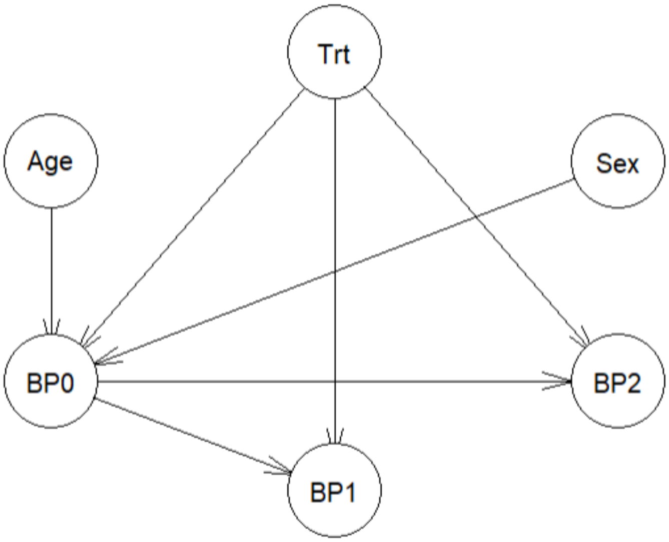

Figure 4.2: Diagram of Bayesian Neural Network of Real Data

4.2 Results

To compare results between all six methods,the average Bias and average Root Mean Square Error (RMSE) were computed. Let y_{ij}^{Sm} be the imputed baseline vector from any method where $m = \{1,2,3,4,5,6\}$. Thus, we define average Bias as the following:

$$\text{Ave Bias} = \frac{1}{k}\sum_{l=1}^{k}\sum_{i=1}^{n}\frac{y_{ij}^{Sm} - y_{ij}}{n}, \tag{4.5}$$

where $i = \{1,\cdots,n\}$, $l = \{1,\cdots,k\}$ and $k = 500$ is the number of iteration used for each method on n subsets with MAR. In the results table, a negative average bias is and underestimate, while a positive is an overestimate from the imputed data compared to the actual data.

Next, we define the average RMSE as

$$\text{Ave RMSE} = \sqrt{\frac{1}{k}\sum_{l=1}^{k}\sum_{i=1}^{n}\frac{\left(y_{ij} - y_{ij}^{Sm}\right)^2}{n}} \tag{4.6}$$

Let θ be a random variable (rv) and $\hat{\theta}$ an estimator, where the expected value of said rv and estimator is defined as $E(\theta)$ and $E\left(\hat{\theta}\right)$. Next, we define the mean square error of $\hat{\theta}$ as the following

$$\text{MSE}\left(\hat{\theta}\right) = E\left[\left(\hat{\theta}-\theta\right)^2\right]. \tag{4.7}$$

Hence, we define the variance of the estimator to be

$$\text{var}\left(\hat{\theta}\right) = E\left(\hat{\theta}\right)^2 - E^2\left(\hat{\theta}\right) \tag{4.8}$$

and the bias of the estimator as

$$\text{Bias}\left(\hat{\theta}\right) = E\left(\hat{\theta}\right) - \theta, \tag{4.9}$$

where for this particular case for bias the θ is a constant.Thus, if we expand

$$\begin{aligned}
\text{MSE}\left(\hat{\theta}\right) =& E\left[\left(\hat{\theta}-\theta\right)^2\right] \\
&= E\left[\left(\hat{\theta}-\theta\right)\left(\hat{\theta}-\theta\right)\right] \\
&= E[\hat{\theta}^2 - 2\hat{\theta}\theta + \theta^2] \\
&= E\left[\hat{\theta^2}\right] - 2\theta E\left[\hat{\theta}\right] + \theta.
\end{aligned} \tag{4.10}$$

Next, if we square the bias we can rewrite it as follows

$$\begin{aligned}
\text{Bias}^2\left(\hat{\theta}\right) &= \left(E\left(\hat{\theta}\right)-\theta\right)^2 \\
&= \left(E\left(\hat{\theta}\right)-\theta\right)\left(E\left(\hat{\theta}\right)-\theta\right) \\
&= E^2\left[\hat{\theta}\right] - 2\theta E\left[\hat{\theta}\right].
\end{aligned} \tag{4.11}$$

By combining (4.8) and (4.11), we derive

$$\begin{aligned}\operatorname{var}\left(\hat{\theta}\right)+\text{Bias}^2\left(\hat{\theta}\right) &= E\left(\hat{\theta}\right)^2 - E^2\left(\hat{\theta}\right) + E^2\left[\hat{\theta}\right] - 2\theta E\left[\hat{\theta}\right] \\ &= E\left[\hat{\theta^2}\right] - 2\theta E\left[\hat{\theta}\right] + \theta \\ &= \text{MSE}\left(\hat{\theta}\right).\end{aligned} \tag{4.12}$$

Therefore, by (4.12)

$$\text{MSE}\left(\hat{\theta}\right) = \operatorname{var}\left(\hat{\theta}\right) + \text{Bias}^2\left(\hat{\theta}\right). \tag{4.13}$$

4.2.1 Result Tables

Recall the following abbreviations for the methods used as:

1. 2lonely.PMM as 2LPMM
2. Bayesian Network as BN
3. Bayesian Network Imputing with Monte Carlo Posterior Inference as BNMC.
4. Bayesian Network Imputing with Exact Inference as BNEI
5. *k* Nearest Neighbor as *k*-NN

Note: For the *k*-NN model the optimal nearest neighbors for both the real and simulated data was a 3-NN.

Table 4.1: Scenario 1 Results with Real Data at *n*=100

Scenario 1 Results with Real Data at *n*=100				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-0.3960	-0.7935	1.7804	2.7430
LR	-0.4576	-0.7812	1.9480	2.6130
2LPMM	-0.3632	-0.8835	1.5414	2.7656
BN	0.0034	0.0070	0.0747	0.1222
BNMC	0.0066	0.0066	0.0725	0.0725
BNEI	0.0071	0.0071	0.0714	0.0714
3-NN	0.0042	0.0096	0.0784	0.1436

Table 4.2: Scenario 2 Results with Real Data at *n*=100

Scenario 2 Results with Real Data at *n*=100				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.0293	-1.5073	3.1740	3.6573
LR	-0.9714	-1.4431	2.9193	3.6244
2LPMM	-0.9540	-1.6423	3.1077	3.8101
BN	0.0021	0.0004	0.1581	0.2133
BNMC	-0.0017	-0.0017	0.1217	0.1217
BNEI	-0.0015	-0.0015	0.1259	0.1259
3-NN	0.0101	0.0155	0.1424	0.2057

Table 4.3: Scenario 3 Results with Real Data at n=100

Scenario 3 Results with Real Data at n=100				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.6699	-3.1369	4.2097	5.6705
LR	-1.3907	-2.7500	3.8422	5.3078
2LPMM	-1.5373	-2.9813	3.8787	4.8603
BN	0.0179	0.0245	0.3304	0.4041
BNMC	0.0236	0.0236	0.3138	0.3138
BNEI	0.0220	0.0220	0.3141	0.3141
3-NN	0.0140	0.0386	0.1928	0.3507

Table 4.4: Simulated Scenario 1 Results with n=100

Simulated Scenario 1 Results with n=100				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.247	-3.049	4.494	8.659
LR	-1.532	-3.151	5.204	8.665
2LPMM	-0.353	-0.422	1.428	1.804
BN	-0.033	-0.207	0.909	1.368
BNMC	-0.033	-0.147	0.913	1.331
BNEI	-0.032	-0.155	0.909	1.334
3-NN	-0.7333	-0.6877	0.8177	1.2272

Table 4.5: Simulated Scenario 2 Results with n=100

Simulated Scenario 2 Results with n=100				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-2.854	-4.448	7.962	9.666
LR	-3.114	-4.982	9.168	10.750
2LPMM	-0.249	-1.247	1.720	2.943
BN	-0.038	0.092	1.388	1.449
BNMC	-0.036	0.099	1.383	1.474
BNEI	-0.038	0.096	1.388	1.466
3-NN	-0.6177	-0.5651	1.1853	1.5067

Table 4.6: Simulated Scenario 3 Results with n=100

Simulated Scenario 3 Results with n=100				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-4.705	-7.89	10.354	12.781
LR	-4.673	-9.466	9.652	15.083
2LPMM	-1.772	-4.198	3.783	7.218
BN	-0.061	-0.088	1.491	1.827
BNMC	-0.061	-0.125	1.495	1.787
BNEI	-0.06	-0.119	1.491	1.779
3-NN	-0.9627	-0.9022	1.4714	2.1011

Sample Size of $n = 100$ for all scenarios and data:

Table 4.7: Scenario 1 Results with Real Data at n=200

Scenario 1 Results with Real Data at n=200				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-0.4636	-0.8273	1.9736	2.6089
LR	-0.4742	-0.8025	2.1629	2.6814
2LPMM	-0.5239	-1.0680	2.1678	3.0942
BN	0.0003	0.0013	0.0332	0.1044
BNMC	0.0028	0.0028	0.0641	0.0641
BNEI	0.0023	0.0023	0.0624	0.0624
3-NN	0.0014	0.0047	0.0445	0.0889

Table 4.8: Scenario 2 Results with Real Data at n=200

Scenario 2 Results with Real Data at n=200				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.0606	-1.5703	3.2220	3.8954
LR	-0.9347	-1.4951	3.0007	3.8743
2LPMM	-0.8658	-1.2628	2.6839	3.1675
BN	0.0043	0.0142	0.1763	0.2918
BNMC	0.0068	0.0068	0.1954	0.1954
BNEI	0.0069	0.0069	0.1916	0.1916
3-NN	0.0046	0.0067	0.0892	0.1188

Table 4.9: Scenario 3 Results with Real Data at n=200

Scenario 3 Results with Real Data at n=200				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.3047	-2.1139	3.2969	4.4581
LR	-1.5674	-2.3041	4.0256	4.7113
2LPMM	-1.7700	-2.9637	4.4482	5.4211
BN	-0.0013	0.0276	0.2998	0.5920
BNMC	-0.0002	-0.0002	0.3010	0.3010
BNEI	-0.0005	-0.0005	0.3033	0.3033
3-NN	0.0070	0.0172	0.1215	0.2185

Table 4.10: Simulated Scenario 1 Results with n=200

Simulated Scenario 1 Results with n=200				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.650	-3.568	6.262	9.369
LR	-1.680	-3.082	6.114	8.178
2LPMM	-0.120	-0.575	0.858	1.976
BN	0.036	-0.119	0.955	1.314
BNMC	0.035	-0.093	0.958	1.224
BNEI	0.036	-0.091	0.955	1.218
3-NN	-0.5971	-0.8089	0.781	1.1341

Table 4.11: Simulated Scenario 2 Results with n=200

Simulated Scenario 2 Results with n=200				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-3.275	-4.978	9.002	10.731
LR	-3.274	-3.971	9.335	8.596
2LPMM	-0.432	-1.043	2.022	2.559
BN	-0.006	-0.053	1.276	1.513
BNMC	-0.006	-0.067	1.275	1.494
BNEI	-0.006	-0.062	1.276	1.486
3-NN	-0.4402	-0.5364	1.1212	1.4099

Table 4.12: Simulated Scenario 3 Results with n=200

Simulated Scenario 3 Results with n=200				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-4.853	-7.809	10.529	12.972
LR	-4.351	-8.688	9.576	14.161
2LPMM	-0.860	-3.446	2.400	5.849
BN	0	-0.181	1.459	1.819
BNMC	0.003	-0.19	1.47	1.803
BNEI	0	-0.178	1.459	1.8
3-NN	-0.9623	-0.9101	1.3844	1.864

Sample Size of $n = 200$ for all scenarios and data:

Table 4.13: Scenario 1 Results with Real Data at n=500

Scenario 1 Results with Real Data at n=500				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-0.3618	-0.8716	1.6624	2.7246
LR	-0.4711	-0.8652	1.9906	2.8555
2LPMM	-0.3870	-1.2484	1.5587	3.6562
BN	0.0140	-0.0050	0.3448	0.4189
BNMC	0.0156	0.0156	0.3805	0.3805
BNEI	0.0155	0.0155	0.3726	0.3726
3-NN	0.0008	0.0014	0.0274	0.0414

Table 4.14: Scenario 2 Results with Real Data at n=500

Scenario 2 Results with Real Data at n=500				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-0.9311	-1.4602	3.0329	3.8451
LR	-0.9383	-1.3587	2.9620	3.6188
2LPMM	-1.2975	-1.5524	3.9471	3.7114
BN	-0.0028	-0.0080	0.2073	0.2394
BNMC	-0.0028	-0.0028	0.2355	0.2355
BNEI	-0.0026	-0.0026	0.2298	0.2298
3-NN	0.0019	0.0032	0.0493	0.0710

Table 4.15: Scenario 3 Results with Real Data at n=500

Scenario 3 Results with Real Data at n=500				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.4767	-2.6720	3.7525	5.2803
LR	-1.5517	-2.6962	3.9744	5.4556
2LPMM	-1.0963	-3.2080	2.9361	6.2430
BN	-0.0036	-0.0118	0.2561	0.4489
BNMC	-0.0051	-0.0051	0.2761	0.2761
BNEI	-0.0047	-0.0047	0.2788	0.2788
3-NN	0.0031	0.0069	0.0724	0.1270

Table 4.16: Simulated Scenario 1 Results with n=500

Simulated Scenario 1 Results with n=500				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.633	-3.282	6.278	8.881
LR	-1.676	-3.068	6.411	8.160
2LPMM	-0.082	-0.582	1.008	1.875
BN	0.025	-0.055	0.992	1.111
BNMC	0.023	-0.04	0.991	1.083
BNEI	0.025	-0.039	0.992	1.079
3-NN	-0.7032	-0.9421	0.7684	1.102

Table 4.17: Simulated Scenario 2 Results with n=500

Simulated Scenario 2 Results with n=500				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-3.188	-4.928	8.245	10.200
LR	-3.407	-5.196	8.917	10.875
2LPMM	-0.557	-0.890	1.810	2.619
BN	0.01	-0.128	1.283	1.448
BNMC	0.009	-0.105	1.281	1.414
BNEI	0.01	-0.108	1.283	1.414
3-NN	-1.1326	-1.2793	1.0833	1.3494

Table 4.18: Simulated Scenario 3 Results with n=500

Scenario 3 Results with n=500				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-4.833	-8.262	10.280	13.292
LR	-4.646	-8.655	9.958	13.905
2LPMM	-1.325	-2.941	3.129	5.400
BN	0.09	-0.036	1.403	1.744
BNMC	0.086	-0.042	1.404	1.645
BNEI	0.09	-0.042	1.402	1.643
3-NN	-0.5112	-0.5889	1.3355	1.7548

Sample Size of $n = 500$ For All Scenarios and Data:

Table 4.19: Scenario 1 Results with Real Data at n=1000

Scenario 1 Results with Real Data at n=1000				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-0.4776	-0.9191	2.1420	2.9676
LR	-0.4440	-0.9296	2.0245	2.9880
2LPMM	-0.3915	-1.0693	1.7886	3.1476
BN	0.0034	0.0029	0.1322	0.2033
BNMC	0.0045	0.0045	0.1495	0.1495
BNEI	0.0043	0.0043	0.1510	0.1510
3-NN	0.0004	0.0008	0.0175	0.0304

Table 4.20: Scenario 2 Results with Real Data at n=1000

Scenario 2 Results with Real Data at n=1000				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-0.9481	-1.4406	2.9550	3.7013
LR	-0.8929	-1.4459	2.8910	3.7067
2LPMM	-0.9508	-1.3081	2.7704	3.1298
BN	0.0048	0.0035	0.2634	0.3408
BNMC	0.0041	0.0041	0.2809	0.2809
BNEI	0.0044	0.0044	0.2777	0.2777
3-NN	0.0008	0.0017	0.0324	0.0507

Table 4.21: Scenario 3 Results with Real Data at n=1000

Scenario 3 Results with Real Data at n=1000				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.4151	-2.4260	3.5591	4.8788
LR	-1.4631	-2.4335	3.6603	4.9197
2LPMM	-1.6174	-2.1180	3.3598	3.5118
BN	-0.0133	-0.0417	0.2602	0.4205
BNMC	-0.0131	-0.0131	0.3011	0.3011
BNEI	-0.0131	-0.0131	0.3016	0.3016
3-NN	0.0015	0.0034	0.0471	0.0828

Table 4.22: Simulated Scenario 1 Results with n=1000

Simulated Scenario 1 Results with n=1000				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-1.607	-3.216	5.995	8.512
LR	-1.736	-3.255	6.715	8.452
2LPMM	-0.124	-0.846	0.911	2.370
BN	-0.045	-0.07	1.097	1.204
BNMC	-0.045	-0.065	1.103	1.206
BNEI	-0.045	-0.062	1.097	1.203
3-NN	-0.098	-0.175	0.7584	1.0816

Table 4.23: Simulated Scenario 2 Results with n=1000

Simulated Scenario 2 Results with n=1000				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-3.277	-5.063	8.697	10.859
LR	-3.225	-5.136	8.532	10.966
2LPMM	-0.818	-1.296	2.343	3.012
BN	-0.005	-0.042	1.277	1.436
BNMC	-0.005	-0.038	1.281	1.43
BNEI	-0.005	-0.039	1.277	1.43
3-NN	-0.0382	-0.0422	1.0702	1.3248

Table 4.24: Simulated Scenario 3 Results with n=1000

Simulated Scenario 3 Results with n=1000				
Methods	y_{2i} Bias	y_{3i} Bias	y_{2i} RMSE	y_{3i} RMSE
LOCF	-4.853	-8.613	10.334	13.916
LR	-4.851	-8.730	10.271	14.065
2LPMM	-1.524	-3.740	3.415	6.307
BN	0.038	-0.031	1.47	1.743
BNMC	0.037	-0.011	1.477	1.7
BNEI	0.038	-0.01	1.47	1.694
3-NN	0.0636	-0.0677	1.3161	1.7215

Sample Size of $n = 1000$ For All Scenarios and Data:

Average Summary Tables Over All Scenarios For n: Define μ to be the average of the average calculated from all three scenarios for each individual method on both dataset as seen in the average summary tables below:

Table 4.25: Summary table for $n = 100$ with all scenarios for real data

Summary table for $n = 100$ with all scenarios for real data				
Methods	$\mu_{y_{2i}}$ Bias	$\mu_{y_{3i}}$ Bias	$\mu_{y_{2i}}$ RMSE	$\mu_{y_{3i}}$ RMSE
LOCF	-1.032	-1.813	3.055	4.024
LR	-0.940	-1.550	2.903	3.848
2LPMM	-0.316	0.152	2.843	3.812
BN	0.008	0.011	0.086	0.247
BNMC	0.010	0.011	0.169	0.169
BNEI	0.009	0.009	0.170	0.170
3-NN	0.009	0.021	0.138	0.233

Table 4.26: Summary table for $n = 100$ with all scenarios for simulated data

Summary table for $n = 100$ with all scenarios for simulated data				
Methods	$\mu_{y_{2i}}$ Bias	$\mu_{y_{3i}}$ Bias	$\mu_{y_{2i}}$ RMSE	$\mu_{y_{3i}}$ RMSE
LOCF	-2.935	-5.129	7.603	10.369
LR	-3.106	-5.866	8.008	11.499
2LPMM	-0.791	-1.956	2.310	3.988
BN	-0.044	-0.068	1.263	1.548
BNMC	-0.043	-0.058	1.264	1.531
BNEI	-0.043	-0.059	1.263	1.526
3-NN	-0.771	-0.718	1.158	1.612

As seen from individual tables for $n = 100$, the Bayesian Neural network methods and 3-NN did very well across all scenarios for average bias and average RMSE on both y_{2i} and y_{3i} through all scenarios with real data. As seen in the μ summary table for $n = 100$ of the real data for all scenarios, the BN method $\mu_{y_{2i}}\text{Bias} = 0.008$ and $\mu_{y_{3i}}\text{Bias} = 0.011$. The BNEI method had the lowest average for all three scenarios for simulated $\mu_{y_{3i}}\text{Bias} = -0.043$. For the simulated data the BNEI method had the lowest average for all three scenarios for simulated $\mu_{y_{2i}}\text{Bias} = -0.043$ and BNMC had the smallest $\mu_{y_{3i}}\text{Bias} = -0.058$,but 3-NN had the smallest $\mu_{y_{2i}}\text{RMSE} = 1.158$. The BN had the smallest $\mu_{y_{3i}}\text{RMSE} = 1.526$. Unfortunately, the linear regression had the largest values for all categories for both simulated and real data as seen in the summery tables for $n = 100$.

Table 4.27: Summary table for $n = 200$ with all scenarios for real data

Summary table for $n = 200$ with all scenarios for real data				
Methods	$\mu_{y_{2i}}$ Bias	$\mu_{y_{3i}}$ Bias	$\mu_{y_{2i}}$ RMSE	$\mu_{y_{3i}}$ RMSE
LOCF	-0.9430	-1.5038	2.8308	3.6541
LR	-0.9921	-1.5339	3.0631	3.7557
2LPMM	-1.0532	-1.7648	3.1000	3.8943
BN	0.0011	0.0144	0.1698	0.3294
BNMC	0.0031	0.0031	0.1868	0.1868
BNEI	0.0029	0.0029	0.1858	0.1858
3-NN	0.0043	0.0095	0.0851	0.1421

Table 4.28: Summary table for $n = 200$ with all scenarios for simulated data

Summary table for $n = 200$ with all scenarios for simulated data				
Methods	$\mu_{y_{2i}}$ Bias	$\mu_{y_{3i}}$ Bias	$\mu_{y_{2i}}$ RMSE	$\mu_{y_{3i}}$ RMSE
LOCF	-2.935	-5.129	7.603	10.369
LR	-3.106	-5.866	8.008	11.499
2LPMM	-0.791	-1.956	2.310	3.988
BN	0.010	-0.118	1.230	1.549
BNMC	0.011	-0.117	1.234	1.507
BNEI	0.010	-0.110	1.230	1.501
3-NN	-0.667	-0.752	1.096	1.469

As seen from individual tables for $n = 200$, the Bayesian Neural network methods and 3-NN yet again did very well across all scenarios for average bias and average RMSE on both y_{2i} and y_{3i} through all scenarios with real data. As seen in the μ summary table for $n = 200$ of the real data for all scenarios, the BN method had the smallest $\mu_{y_{2i}}$Bias $= 0.0011$ and $\mu_{y_{3i}}$Bias $= 0.00144$, while 3-NN had the smallest $\mu_{y_{2i}}$RMSE $= 0.0851$ and $\mu_{y_{3i}}$RMSE $= 0.1421$. For the simulated data the same holds true for the smallest $\mu_{y_{2i}}$Bias $= 0.01$, but the BNEI has the smallest $\mu_{y_{3i}}$Bias $= -0.110$. The 3-NN had the smallest $\mu_{y_{2i}}$RMSE $= 1.096$ and the smallest $\mu_{y_{3i}}$RMSE $= 1.469$. Unfortunately, the linear regression had the largest values for all categories for both simulated and real data as seen in the summery tables for $n = 200$.

Table 4.29: Summary table for $n = 500$ with all scenarios for real data

Summary table for $n = 500$ with all scenarios for real data				
Methods	$\mu_{y_{2i}}$ Bias	$\mu_{y_{3i}}$ Bias	$\mu_{y_{2i}}$ RMSE	$\mu_{y_{3i}}$ RMSE
LOCF	-0.923	-1.668	2.816	3.950
LR	-0.987	-1.640	2.976	3.977
2LPMM	-0.927	-2.003	2.814	4.537
BN	0.003	-0.008	0.269	0.369
BNMC	0.003	0.003	0.297	0.297
BNEI	0.003	0.003	0.294	0.294
3-NN	0.002	0.004	0.050	0.080

Table 4.30: Summary table for $n = 500$ with all scenarios for simulated data

Summary table for $n = 500$ with all scenarios for simulated data				
Methods	$\mu_{y_{2i}}$ Bias	$\mu_{y_{3i}}$ Bias	$\mu_{y_{2i}}$ RMSE	$\mu_{y_{3i}}$ RMSE
LOCF	-2.414	-4.118	6.201	8.093
LR	-2.432	-4.230	6.322	8.235
2LPMM	-0.491	-1.103	1.487	2.474
BN	0.0313	-0.0548	0.9195	1.0758
BNMC	0.0295	-0.0468	0.9190	1.0355
BNEI	0.0313	-0.0473	0.9193	1.0340
3-NN	-0.5868	-0.7026	0.7968	1.0516

As seen from individual tables for $n = 500$, the Bayesian Neural network methods and 3-NN improved performance across all scenarios for average bias and average RMSE on both y_{2i} and y_{3i} through all scenarios with real data. As seen in the μ summary table for $n = 500$ of the real data for all scenarios, the 3-NN method had the smallest $\mu_{y_{2i}}$Bias $= 0.002$ and BNMC had the smallest $\mu_{y_{3i}}$Bias $= 0.003$, while k-NN had the smallest $\mu_{y_{2i}}$RMSE $= 0.05$ and $\mu_{y_{3i}}$RMSE $= 0.08$. For the simulated data, the BNMC had the smallest $\mu_{y_{2i}}$Bias $= 0.0313$ and the smallest $\mu_{y_{3i}}$Bias $= -0.0468$, while 3-NN had the smallest $\mu_{y_{2i}}$RMSE $= 0.7968$ and BNEI had the smallest $\mu_{y_{3i}}$RMSE $= 1.0340$. Unfortunately, the linear regression had the largest values for all categories for both simulated and real data as seen in the summery tables for $n = 500$.

Table 4.31: Summary table for $n = 1000$ with all scenarios for real data

Summary table for $n = 1000$ with all scenarios for real data				
Methods	$\mu_{y_{2i}}$ Bias	$\mu_{y_{3i}}$ Bias	$\mu_{y_{2i}}$ RMSE	$\mu_{y_{3i}}$ RMSE
LOCF	-0.947	-1.595	2.885	3.849
LR	-0.933	-1.603	2.859	3.871
2LPMM	-0.987	-1.498	2.640	3.263
BN	-0.002	-0.012	0.219	0.322
BNMC	-0.002	-0.002	0.244	0.244
BNEI	-0.001	-0.001	0.243	0.243
3-NN	0.001	0.002	0.032	0.055

Table 4.32: Summary table for $n = 1000$ with all scenarios for simulated data

Summary table for $n = 1000$ with all scenarios for simulated data				
Methods	$\mu_{y_{2i}}$ Bias	$\mu_{y_{3i}}$ Bias	$\mu_{y_{2i}}$ RMSE	$\mu_{y_{3i}}$ RMSE
LOCF	-3.246	-5.631	8.342	11.096
LR	-3.271	-5.707	8.506	11.161
2LPMM	-0.822	-1.961	2.223	3.896
BN	-0.004	-0.048	1.281	1.461
BNMC	-0.004	-0.038	1.287	1.445
BNEI	-0.004	-0.037	1.281	1.442
3-NN	-0.024	-0.095	1.048	1.376

As seen from individual tables for $n = 1000$, the Bayesian Neural network methods and 3-NN improved performance yet again across all scenarios for average bias and average RMSE on both y_{2i} and y_{3i} through all scenarios with real data. As seen in the μ summary table for $n = 1000$ of the real data for all scenarios, the 3-NN method and BNEI had the smallest $\mu_{y_{2i}}$Bias $= \pm 0.001$ and BNEI had the smallest $\mu_{y_{3i}}$Bias $= -0.001$, while 3-NN had the smallest $\mu_{y_{2i}}$RMSE $= 0.032$ and $\mu_{y_{3i}}$RMSE $=$ 0.055. For the simulated data, all the Bayesian methods had the smallest $\mu_{y_{2i}}$Bias $= -0.004$ and BNEI had the smallest $\mu_{y_{2i}}$Bias $= -0.037$. The 3-NN had the smallest $\mu_{y_{2i}}$RMSE $= 1.048$, and smallest $\mu_{y_{3i}}$RMSE $= 1.376$. Unfortunately, the linear regression had the largest values for all categories for both simulated and real data as seen in the summery tables for $n = 1000$.

4.2.2 Average Summary Tables Comparison Graphs

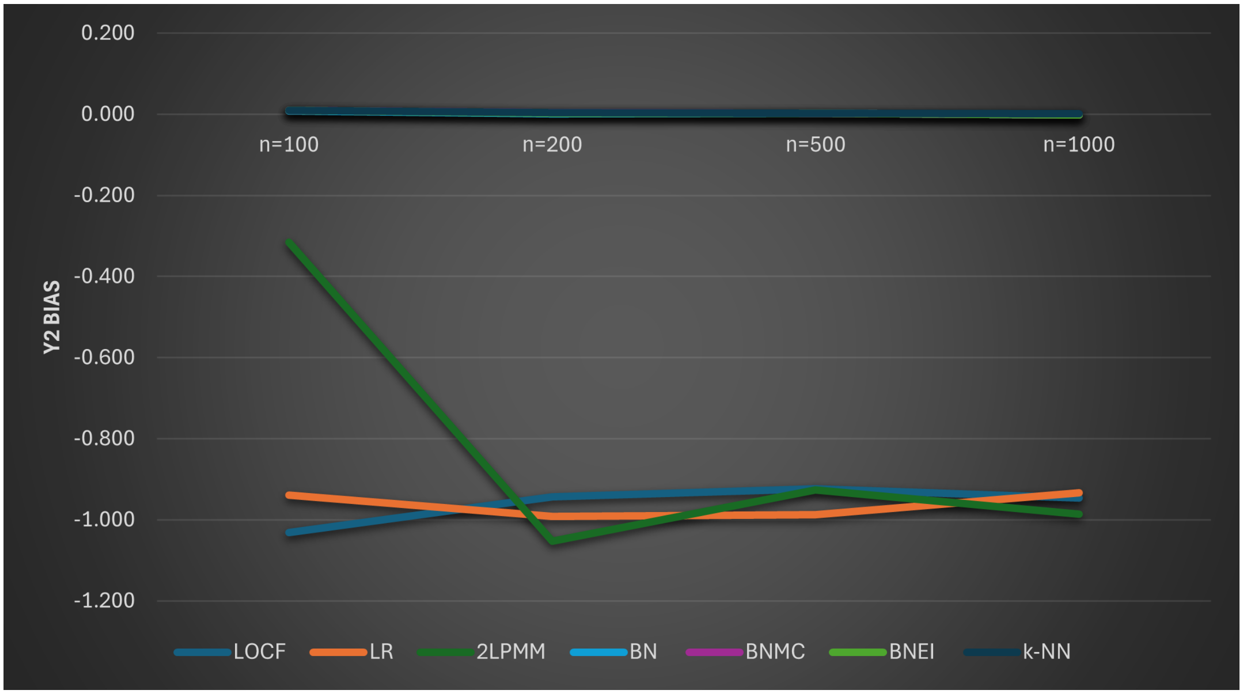

Figure 4.3: $\mu_{y_{2i}}$ Bias for Real Data

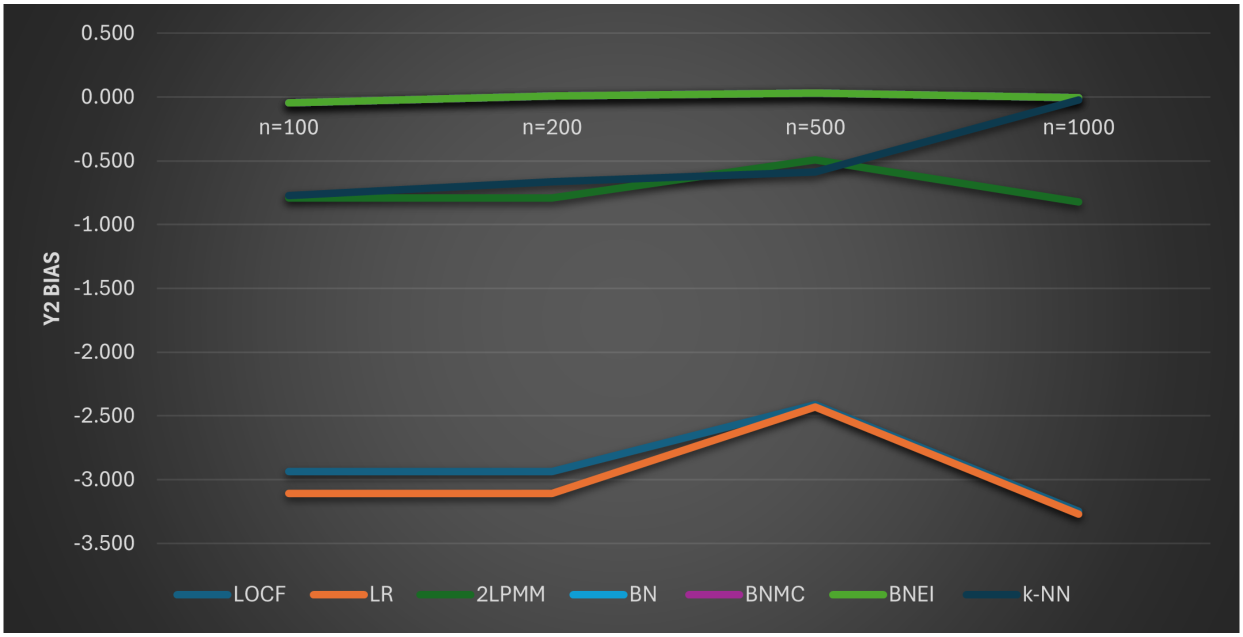

Figure 4.4: $\mu_{y_{2i}}$ Bias for Simulated Data

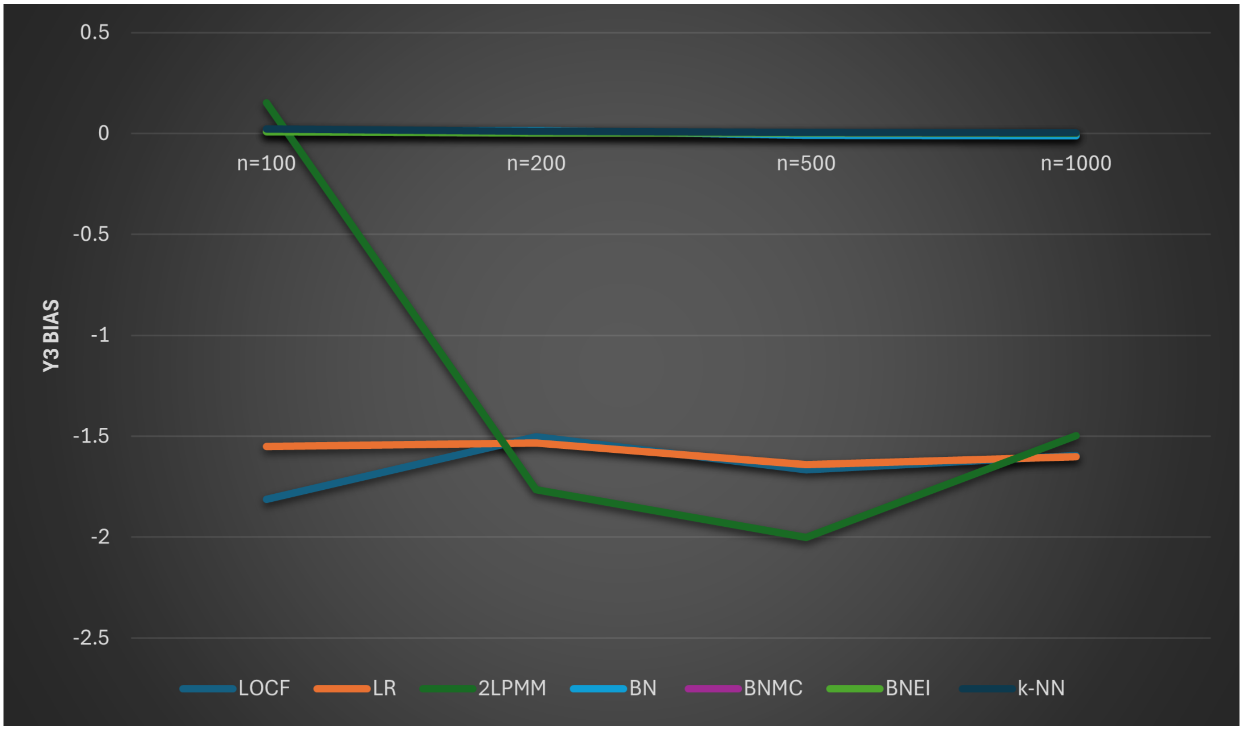

Figure 4.5: $\mu_{y_{3i}}$ Bias for Real Data

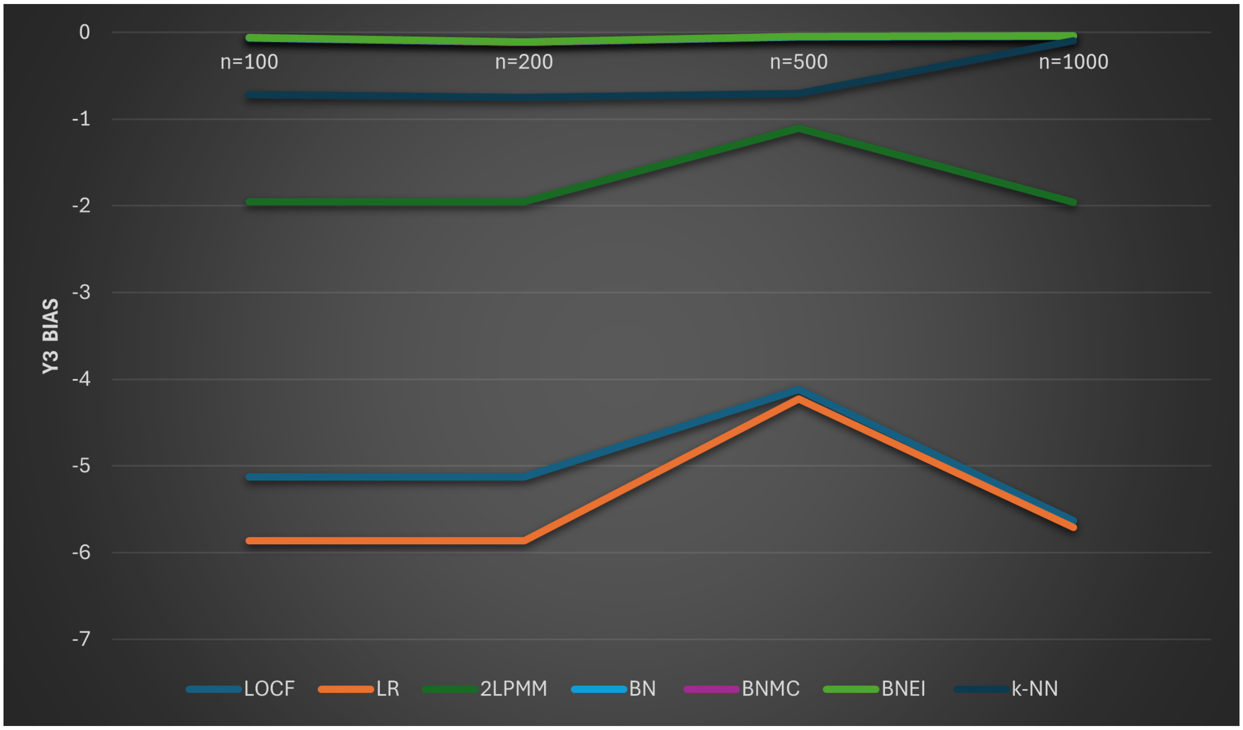

Figure 4.6: $\mu_{y_{3i}}$ Bias for Simulated Data

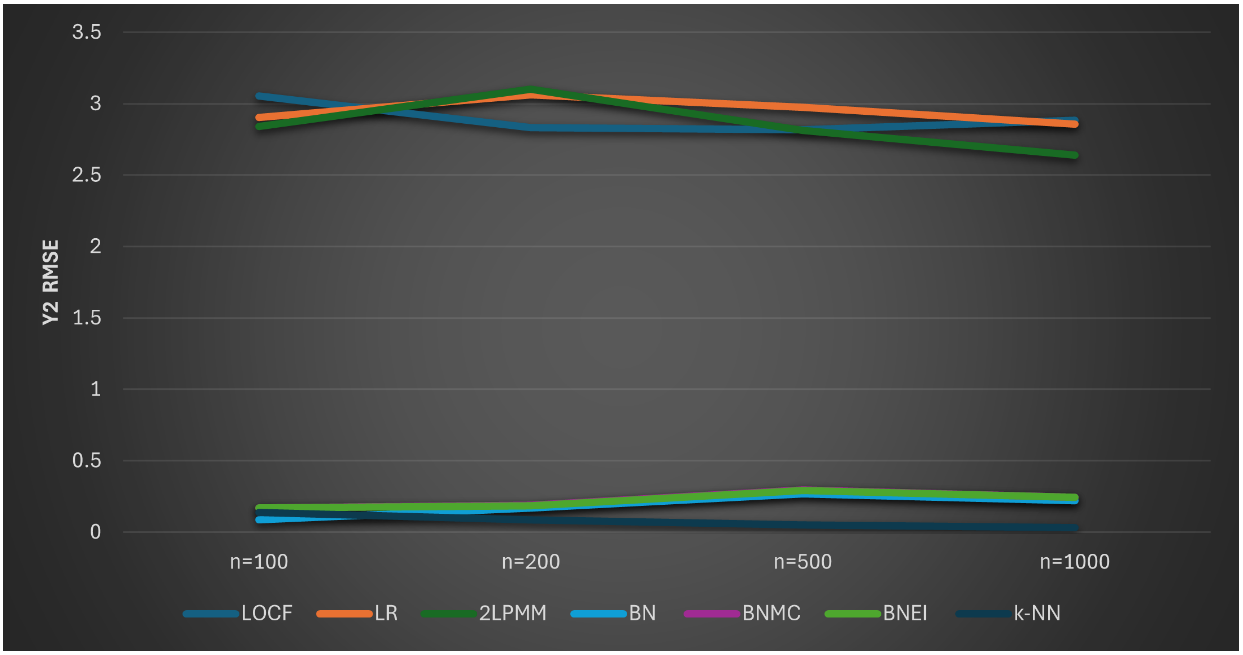

Figure 4.7: $\mu_{y_{2i}}$ RMSE for Real Data

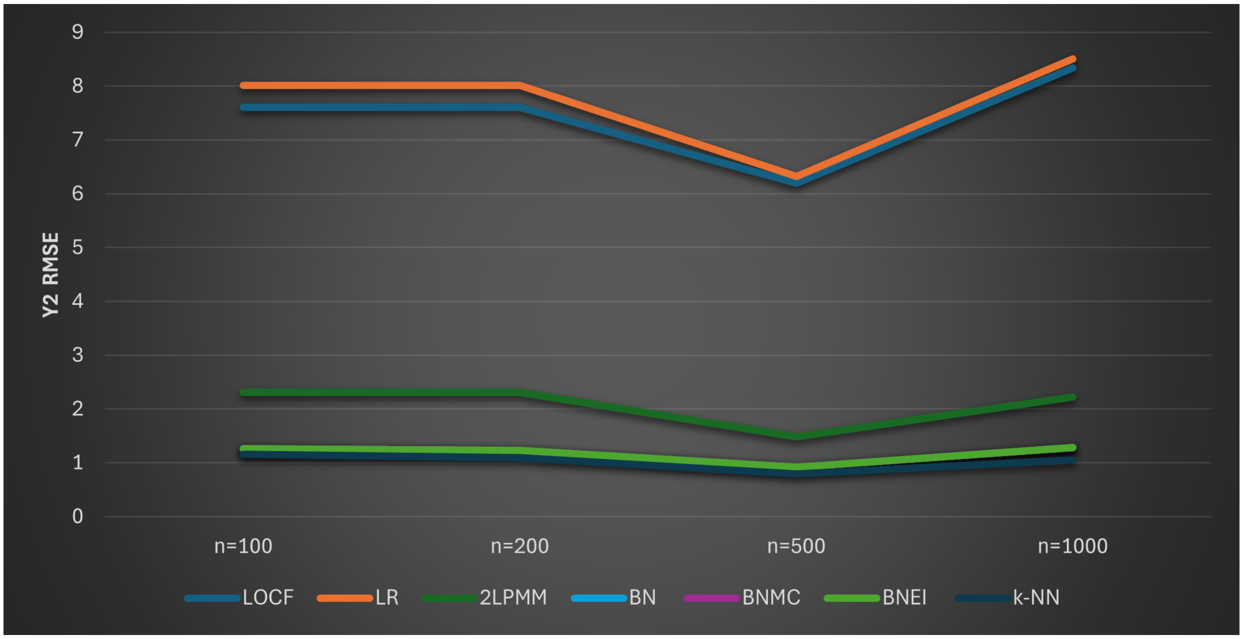

Figure 4.8: $\mu_{y_{2i}}$ RMSE for Simulated Data

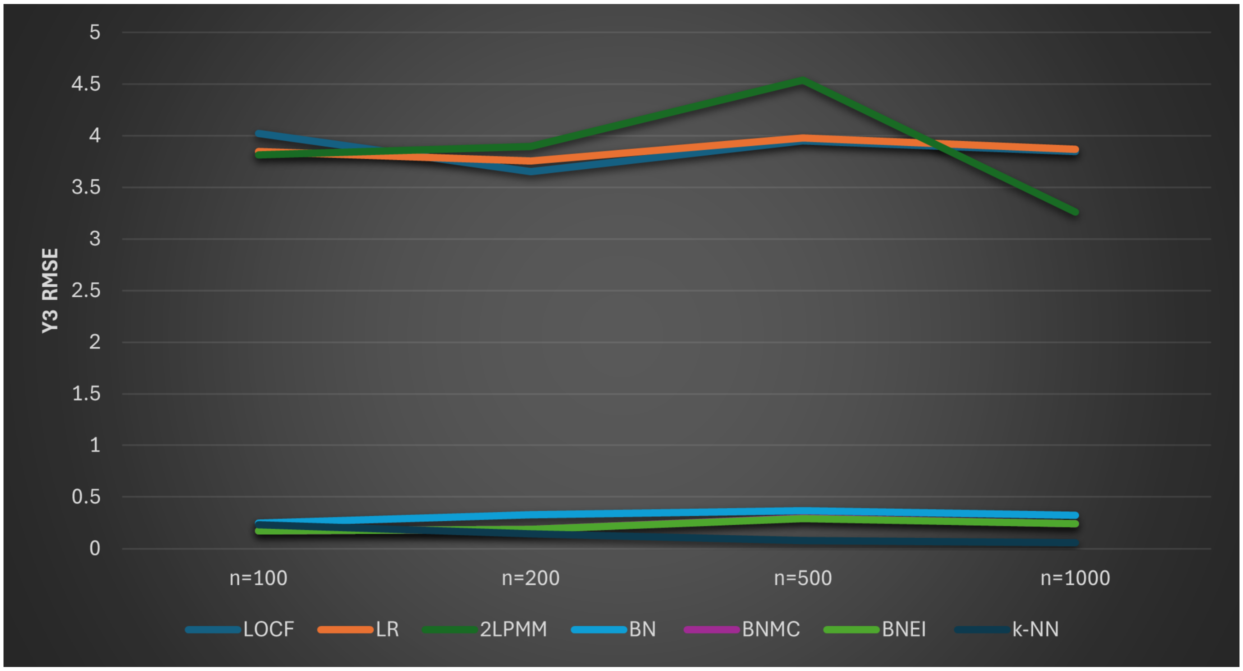

Figure 4.9: $\mu_{y_{3i}}$ RMSE for Real Data

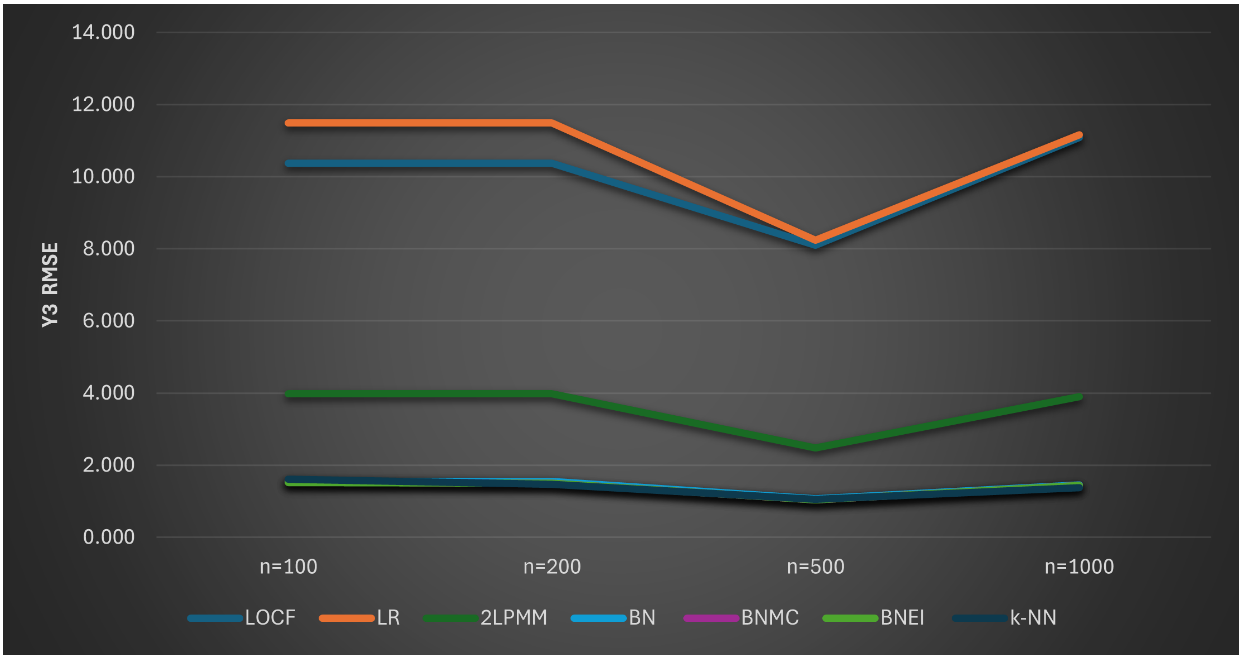

Figure 4.10: $\mu_{y_{3i}}$ RMSE for Simulated Data

CHAPTER V

CONCLUSION AND DISCUSSION

In this thesis, we addressed data missingness in the response variable of both simulated and real longitudinal datasets. Although the absence of data is usually considered a hindrance, I believe methods such as Bayesian Neural network and *k*-NN, as well as, LOCF at appropriate times are effective methods to imputed data. Thus, imputed data provides the researcher a complete case dataset to draw accurate conclusions upon. In addition, with little literature on comparative data analysis which includes Bayesian Neural network , I believe we can conclude how precise and accurate with little cost of computing power is need to effectively impute missing data. As seen in all average summary tables comparison graphs, all Bayesian Neural Networks outperformed all methods besides for *k*-NN at particular *n*-sized samples. As noted by Md. Hamidul Hugue the *k*-NN method is better equipped to handle large dataset for MAR.

One issue with simulation-based methods is the random variation from the simulations, which is critical to assess the potential variation and/or monitor the convergence. In simulation-based method for analysis of clinical trials, the analysis plan should predetermine all the algorithms, software packages, and random seeds for the computation. Generally, the analysis should use a sufficient number of imputations and/or simulations to reduce the random variation.The methods presented are only a few applications of simulation methods for missing data issues. Of course, many other simulation-based methods are available that can be used for missing data. For example, we considered only a linear mixed model for the real and imputed data to compare coefficients.

REFERENCES

[1] D. M. Chickering, *A transformational characterization of equivalent bayesian network structures*, arXiv preprint arXiv:1302.4938, (2013).

[2] B. V. Dasarathy, *Nearest neighbor (nn) norms: Nn pattern classification techniques*, IEEE Computer Society Tutorial, (1991).

[3] J. K. Dixon, *Pattern recognition with partly missing data*, IEEE Transactions on Systems, Man, and Cybernetics, 9 (1979), pp. 617–621.

[4] G. M. Fitzmaurice, N. M. Laird, and J. H. Ware, *Applied longitudinal analysis*, John Wiley & Sons, 2012.

[5] C. P. Friedman, A. S. Elstein, F. M. Wolf, G. C. Murphy, T. M. Franz, P. S. Heckerling, P. L. Fine, T. M. Miller, and V. Abraham, *Enhancement of clinicians' diagnostic reasoning by computer-based consultation: a multisite study of 2 systems*, Jama, 282 (1999), pp. 1851–1856.

[6] A. M. Gad and R. H. M. Abdelkhalek, *Imputation methods for longitudinal data: A comparative study*, International Journal of Statistical Distributions and Applications, 3 (2017), p. 72.

[7] S. Grund, O. Lüdtke, and A. Robitzsch, *Multiple imputation of missing data for multilevel models: Simulations and recommendations*, Organizational Research Methods, 21 (2018), pp. 111–149.

[8] M. H. Huque, J. B. Carlin, J. A. Simpson, and K. J. Lee, *A comparison of multiple imputation methods for missing data in longitudinal studies*, BMC medical research methodology, 18 (2018), pp. 1–16.

[9] M. Jensen, *Value maximisation, stakeholder theory, and the corporate objective function*, European financial management, 7 (2001), pp. 297–317.

[10] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*, MIT press, 2009.

[11] K. B. Korb and A. E. Nicholson, *Bayesian artificial intelligence*, CRC press, 2010.

[12] R. J. Little and D. B. Rubin, *The analysis of social science data with missing values*, Sociological methods & research, 18 (1989), pp. 292–326.

[13] G. F. Liu and J. Kost, *Applications of simulation for missing data issues in longitudinal clinical trials*, Monte-Carlo simulation-based statistical modeling, (2017), pp. 211–232.

[14] R. Nagarajan, M. Scutari, and S. Lèbre, *Bayesian networks in r*, Springer, 122 (2013), pp. 125–127.

[15] J. E. Overall, S. Tonidandel, and R. R. Starbuck, *Last-observation-carried-forward (locf) and tests for difference in mean rates of change in controlled repeated measurements designs with dropouts*, Social Science Research, 38 (2009), pp. 492–503.

[16] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, Elsevier, 2014.

[17] J. Pearl and T. S. Verma, *A theory of inferred causation*, in Studies in Logic and the Foundations of Mathematics, vol. 134, Elsevier, 1995, pp. 789–811.

[18] K. Polotskaya, C. S. Muñoz-Valencia, A. Rabasa, J. A. Quesada-Rico, D. Orozco-Beltrán, and X. Barber, *Bayesian networks for the diagnosis and prognosis of diseases: A scoping review*, Machine Learning and Knowledge Extraction, 6 (2024), pp. 1243–1262.

[19] C. E. Rasmussen and Z. Ghahramani, *Bayesian monte carlo*, Advances in neural information processing systems, (2003), pp. 505–512.

[20] S. Roweis, G. Hinton, and R. Salakhutdinov, *Neighbourhood component analysis*, Adv. Neural Inf. Process. Syst.(NIPS), 17 (2004), p. 4.

[21] D. B. Rubin, *An overview of multiple imputation*, Master's thesis, The University of Texas-Rio Grande Valley, August 2002.

[22] D. Sarkar, *Multivariate data visualization with r*, Use R, (2008).

[23] J. L. Schafer and M. K. Olsen, *Multiple imputation for multivariate missing-data problems: A data analyst's perspective*, Multivariate behavioral research, 33 (1998), pp. 545–571.

[24] G. Shakhnarovich, T. Darrell, and P. Indyk, *Nearest-neighbor methods in learning and vision*, IEEE Trans. Neural Networks, 19 (2008), p. 377.

[25] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, *The max-min hill-climbing bayesian network structure learning algorithm*, Machine learning, 65 (2006), pp. 31–78.

[26] S. Van Buuren, *Flexible imputation of missing data*, CRC press, 2018.

[27] J. Whittaker, *Graphical models in applied multivariate statistics*, Wiley Publishing, 2009.

APPENDIX A

APPENDIX A

Thesis code can be found in Github with the following URL:

https://github.com/josephalanis/thesis_longitudinal_data.git

BIOGRAPHICAL SKETCH

Joseph O. Alanis from McAllen TX has completed his third year of study in pursuit of his Masters in Mathematics emphasis in Statistics at the University of Texas Rio Grande Valley. In addition, Joseph O. Alanis is pursuing his Doctor of Philosophy in Mathematics and Statistics with Interdisciplinary Applications at the University of Texas Rio Grande Valley. In August 2024, he graduated with his second Masters in Mathematics degree emphasis in Statistics from the University of Texas Rio Grande Valley. Mr. Alanis obtained a Bachelor's of Science from Upper Iowa University majoring in Mathematics on December 2014 and a Master's of Liberal Arts emphasis in Mathematics for Teaching from Harvard Extension School at Harvard University on November 2018. Mr. Alanis is a Lecture II for the School of Mathematical and Statistical Science at the University of Texas Rio Grande Valley. Mr. Alanis's work email joseph.alanis@utrgv.edu.